ANALYZING AND MITIGATING THE CENTRALIZATION TENDENCY OF THE LIGHTNING NETWORK

Gustavo Franco Camilo

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia Elétrica, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia Elétrica.

Orientadores: Luís Henrique Maciel Kosmalski Costa
Miguel Elias Mitre Campista

Rio de Janeiro
Dezembro de 2023

ANALYZING AND MITIGATING THE CENTRALIZATION TENDENCY OF THE LIGHTNING NETWORK

Gustavo Franco Camilo

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Orientadores: Luís Henrique Maciel Kosmalski Costa
Miguel Elias Mitre Campista

Aprovada por: Prof. Avelino Francisco Zorzo
Prof. Célio Vinícius Neves de Alquerque
Prof. Luís Henrique Maciel Kosmalski Costa
Prof. Miguel Elias Mitre Campista

RIO DE JANEIRO, RJ – BRASIL
DEZEMBRO DE 2023

Franco Camilo, Gustavo

Analyzing and Mitigating the Centralization Tendency of the Lightning Network/Gustavo Franco Camilo. – Rio de Janeiro: UFRJ/COPPE, 2023.

XIV, 63 p.: il.; 29, 7cm.

Orientadores: Luís Henrique Maciel Kosmalski Costa
Miguel Elias Mitre Campista

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia Elétrica, 2023.

Referências Bibliográficas: p. 47 – 55.

1. blockchain. 2. payment channel networks. 3. off-chain payments. I. Maciel Kosmalski Costa, Luís Henrique *et al.* II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Elétrica. III. Título.

To my family.

Agradecimentos

Agradeço primeiramente aos meus pais, Moisés e Sandra, às minhas irmãs, Rafaela, Juliana e Isabela, e toda a minha família por estarem sempre comigo e me incentivarem ao longo de toda minha formação escolar e acadêmica. Agradeço também aos membros da minha família que me incentivaram durante toda a minha vida e hoje não estão mais aqui para acompanhar o meu crescimento. Obrigado por tudo. Sem vocês, nada disso seria possível. Espero retribuir todo esforço, amor, carinho e confiança depositados em mim durante todos esses anos.

Agradeço também a todos os amigos que fiz durante a faculdade pela motivação e por me acompanharem durante o curso. Em especial, agradeço ao meu grupo mais próximo durante a graduação: Bernardo, Diogo, João Felipe, João Pedro, Pedrosa, Rafael e Walter. A companhia de vocês nos altos e baixos da graduação e da pós-graduação durante esses anos tornaram minha vida no curso melhor, mais fácil e mais alegre.

Agradeço aos meus orientadores, os professores Luís e Miguel, por todos os conselhos, dedicação, paciência e confiança fornecidos a mim. Agradeço a eles também por aceitarem me orientar após a graduação. Estes são os responsáveis pela minha formação e pela qualidade desta dissertação. Obrigado pela motivação e por reforçar o meu desejo de seguir o caminho acadêmico. Agradeço também ao professor Otto Duarte por me orientar por três anos e por despertar meu desejo de efetuar pesquisa científica.

Aos professores e a todos os meus companheiros do Grupo de Teleinformática e Automação (GTA), por proporcionarem um ambiente acolhedor em que pude trabalhar nos últimos 5 anos. Em especial, deixo um agradecimento ao Lucas Airam Castro de Souza e Gabriel Antonio Fontes Rebello, os quais participaram comigo em diversas publicações e projetos de pesquisa.

Agradeço aos professores Avelino Zorzo e Célio Albuquerque por aceitarem o convite para participar da banca examinadora deste trabalho.

Agradeço à UFRJ e a todos os meus professores, por sempre darem seu máximo e contribuírem para minha formação. Às agências de fomento à pesquisa brasileira CNPq, CAPES, FAPERJ, FAPESP e RNP por viabilizarem este trabalho e incentivarem continuamente a pesquisa de alto nível no Brasil.

Por fim, agradeço ao povo brasileiro por manter e garantir o meu acesso e de milhares de pessoas à educação pública de qualidade desde o primeiro ano do Ensino Fundamental até a Pós-Graduação. Este trabalho é uma maneira de retribuir todo o investimento e confiança em mim depositados.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

ANÁLISE E MITIGAÇÃO DA TENDÊNCIA DE CENTRALIZAÇÃO DA REDE RELÂMPAGO

Gustavo Franco Camilo

Dezembro/2023

Orientadores: Luís Henrique Maciel Kosmalski Costa
Miguel Elias Mitre Campista

Programa: Engenharia Elétrica

As redes de canais de pagamento (*Payment Channel Networks* - PCN) são a principal solução de escalabilidade para criptomoedas públicas, oferecendo um método alternativo de pagamento que evita lentos mecanismos de consenso. Em PCNs, a topologia da rede influencia diretamente o desempenho, o custo e a taxa de sucesso de pagamentos. Esta dissertação analisa a evolução da topologia da Rede Relâmpago (*Lightning Network* - LN), a principal rede de canais de pagamento, e propõe uma solução para mitigar a sua tendência de centralização. Primeiro, o grafo da rede é reconstruído usando dados reais de um conjunto de mensagens coletado entre janeiro de 2020 e agosto de 2021. A análise usa métricas tradicionais de teoria dos grafos para avaliar o estado e a evolução da rede. Os resultados mostram uma forte tendência na Rede Relâmpago para centralização de recursos e conectividade, tendo apenas $0,38\%$ dos nós concentrando 50% da capacidade da rede.

Em seguida, propõe-se o ProfitPilot, uma estratégia de posicionamento de nós que incentiva a criação de ciclos na LN para permitir o rebalanceamento de baixo custo e reverter a tendência de centralização. Do ponto de vista do usuário, o ProfitPilot oferece canais lucrativos e rotas de baixo custo, incentivando a adesão dos usuários. Do ponto de vista da rede, o ProfitPilot cria redundâncias ao forçar a formação de ciclos, aumentando a robustez da rede. O modelo proposto é comparado com heurísticas disponíveis na Rede Relâmpago. De todas as heurísticas avaliadas, o ProfitPilot apresenta o aumento mais rápido na transitividade da rede, diminuindo a tendência de centralização e reduzindo o impacto de ataques topológicos direcionados em mais de 17%.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

ANALYZING AND MITIGATING THE CENTRALIZATION TENDENCY OF THE LIGHTNING NETWORK

Gustavo Franco Camilo

December/2023

Advisors: Luís Henrique Maciel Kosmalski Costa
Miguel Elias Mitre Campista

Department: Electrical Engineering

Payment channel networks (PCNs) are the leading scalability solution for public cryptocurrencies, offering an alternative payment method that avoids slow consensus mechanisms. In PCNs, the network topology directly influences the performance, cost, and payment success rate. This thesis analyzes the evolution of the Lightning Network (LN) topology, the leading payment channel network, and proposes a solution to mitigate its centralization trend. First, we reconstruct the network graph using real data from a set of channel announcement messages collected between January 2020 and August 2021. Our analysis uses traditional graph theory metrics to evaluate the state and evolution of the network. The results show a strong trend in the Lightning Network for resource and connectivity centralization with only 0.38% of nodes concentrating 50% of the network capacity.

Then, we design ProfitPilot, a node positioning strategy that encourages cycle creation in PCNs to reverse the current trend in centralization and enable cheap off-chain rebalancing. From the user's point of view, ProfitPilot offers profitable channels and low-cost routes, encouraging user adoption. From the network's point of view, ProfitPilot creates redundancies by forcing the formation of cycles, enhancing network robustness. We compare our proposed model with some heuristics available in the Lightning Network. Out of all the heuristics evaluated, ProfitPilot presents the fastest increase in network transitivity, decreasing the trend in centralization, and mitigates the impact of targeted topological attacks by over 17%.

Contents

List of Figures

ACRONYMS

BOLT - Basis of the Lightning Technology

CAPES - Coordenação de Aperfeiçoamento de Pessoal de Nível Superior

CNPq - Conselho Nacional de Desenvolvimento Científico e Tecnológico

COPPE - Instituto Alberto Luiz Coimbra de Pós-Graduação e Pesquisa em Engenharia

DAG - Directed Acyclic Graph

DoS - Denial of Service

FAPERJ - Fundação Carlos Chagas Filho de Amparo à Pesquisa do Estado do Rio de Janeiro

FAPESP - Fundação de Amparo à Pesquisa do Estado de São Paulo

GTA - Grupo de Teleinformática e Automação

LN - Lightning Network

LND - Lightning Network Daemon

PCN - Payment Channel Networks

PEE - Programa de Engenharia Elétrica

RFC - Request for Comments

RNP - Rede Nacional de Ensino e Pesquisa

SPF - Shortest Path First

UFRJ - Universidade Federal do Rio de Janeiro

Chapter 1

Introduction

Public cryptocurrencies still suffer from low scalability despite their recent success. While traditional payment methods, such as credit cards, process over 24,000 transactions per second [1], the two largest cryptocurrencies, Bitcoin and Ethereum, achieve a throughput of only 7 and 15 transactions per second, respectively [2]. This issue, commonly referred to as the *blockchain scalability problem* [3], stems primarily from the requirement imposed by public blockchains of processing every transaction through a time-consuming global consensus mechanism [2, 4–6]. The high transaction confirmation time also leads to block proposers prioritizing transactions with high fees, which makes micropayments infeasible. Overall, the low scalability of public blockchains hinders the adoption of cryptocurrencies for daily purchases and financial operations, which require fast transaction confirmation time and high throughput.

Payment channels are one of the main solutions to the scalability problem of public cryptocurrencies [6–8]. In payment channels, two users can establish an *off-chain communication channel* to issue payments directly to each other, bypassing the blockchain. These channels can be connected to form a payment channel network (PCN), which allows users to route payments through intermediaries that forward them in exchange for fees. This network of channels enables payments between users who do not share a direct channel with each other. PCNs have been implemented in the two most popular cryptocurrencies, namely Bitcoin's Lightning Network (LN) [6] and Ethereum's Raiden [9], and have effectively improved transaction throughput and latency in public blockchain systems. In fact, PCNs have been cited as one of the reasons for the adoption of Bitcoin as an official payment method in El Salvador [10].

The great success of the Lightning Network and its consolidation as a leading PCN implementation motivates the analysis of its growth once the performance of routing and channel balancing proposals depends on the topological characteristics of the network [11, 12]. Even though the topology of the Lightning Network has been explored in other works [13–15], few of them analyze the evolution of network

metrics. Furthermore, due to the rapid growth and high dynamism of the network, which allows easy opening and closing of channels, some analyses are already lagging behind the most recent state of the network [13, 14]. Thus, the analysis of network evolution and trends is essential to assess the feasibility of current and future routing proposals in PCN.

1.1 Contributions

This thesis starts by evaluating the evolution of the LN's topological metrics, analyzing recent trends and implications for the future of the network. Our goal is to answer the following question: *How has LN's topology changed with its growth and where is it heading?*. Therefore, we make the following contribution:

Contribution 1: Analysis of the LN's topology evolution. We use real information collected from gossip messages sent in the network to reconstruct the complete graph from January 2020 to August 2021. First, we analyze the state of the network in August 2021, checking the distribution of resources and connectivity. We also analyze the time series of classical graph theory and complex network metrics to evaluate the evolution of the network structure.

The analysis shows that the Lightning Network is becoming even more centralized and that some proposed channel rebalancing techniques are expensive or infeasible for most nodes in the network. This work yielded two publications:

- Camilo, G. F., Rebello, G. A. F., de Souza, L. A. C., Potop-Butucaru, M., Amorim, M. D., Campista, M. E. M., Costa, L. H. M. K. - "Análise da Evolução Topológica da Rede Lightning de Canais de Pagamento", in XXII Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais (SBSeg 2022), Santa Maria, Brazil, September 2022.

- Camilo, G. F., Rebello, G. A. F., de Souza, L. A. C., Potop-Butucaru, M., Amorim, M. D., Campista, M. E. M., Costa, L. H. M. K. - "Topological Evolution Analysis of Payment Channels in the Lightning Network", in 14th IEEE Latin-American Conference on Communications (LATINCOM 2022) - Rio de Janeiro, Brazil, November 2022.

The analysis of the Lightning Network topology also showed that, despite off-chain rebalancing being the most efficient approach for restoring channels [16–18], Lightning's current topology restricts such operations to a subset of nodes. Moreover, we demonstrate the impact of cycle scarcity in network centralization and how it weakens the network against targeted attacks. We simulate the attacks on a snapshot of the Lightning Network and show their alarming long-term consequences.

Thus, in the second part of the thesis we aim to answer the following question: *Is it possible to mitigate the centralization tendency of LN while enabling off-chain rebalancing to nodes?* To mitigate this centralization effect, we propose the following contribution:

Contribution 2: ProfitPilot. ProfitPilot is a node attachment strategy for PCNs that prioritizes cycle creation, enabling cheap rebalancing and improving network robustness. ProfitPilot focuses on the creation of cycles in the network. In particular, the core of our strategy is to encourage the arrangement of 3-node cycles to create low-cost routes and enhance network robustness. As creating cycles requires users to open more than one channel, which is costly, ProfitPilot strategically creates profitable channels to encourage user adoption.

We implement ProfitPilot and compare our proposal to state-of-the-art attachment heuristics available in the Lightning Network. We evaluate our algorithm on the Lightning Network and two synthetic topologies. ProfitPilot achieves 2× higher fee collection in all evaluated topologies when compared to other heuristics and efficiently generates cheaper routes. Moreover, ProfitPilot rapidly increases network transitivity and mitigates the effect of topological attacks. This work has resulted in the following publication, which was awarded by the Brazilian Computer Society:

- Camilo, G. F., Rebello, G. A. F., de Souza, L. A. C., Campista, M. E. M., Costa, L. H. M. K. - "Posicionamento Lucrativo de Nós e Criação de Rotas de Baixo Custo na Rede Relâmpago", in XLI Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC 2023), Brasília, DF, Brazil, May 2023. **Best paper runner-up.**

The above-cited work is also under submission to an international journal.

Besides these two contributions and the above-cited papers, three other significant works on payment channel networks were produced during the period of this thesis, including a mini-course, a survey, and a PCN simulator. Nevertheless, the below-cited works do not compose this thesis:

- Camilo, G.F., Rebello, G. A. F., de Souza, L. A. C., Thomaz, G. A., Potop-Butucaru, M., Amorim, M. D., Campista, M. E. M., Costa, L. M. K. - "Redes de Canais de Pagamento: Provendo Escalabilidade para Pagamentos em Criptomoedas", in Minicursos do XL Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC 2022), Fortaleza, Brazil, May 2022.

- Rebello, G. A. F., Camilo, G.F., de Souza, L. A. C., Thomaz, G. A., Potop-Butucaru, M., Amorim, M. D., Campista, M. E. M., Costa, L. M. K. - "A Survey on Layer-Two Protocols for Blockchains", submitted to IEEE Communications Surveys and Tutorials.

- Rebello, G. A. F., Camilo, G. F., Potop-Butucaru, M., Campista, M. E. M., Amorim, M. D., Costa, L. H. M. K. - "PCNsim: A Flexible and Modular Simulator for Payment Channel Networks", IEEE International Conference on Computer Communications Workshops (INFOCOM WKSHPS), Virtual Conference, May 2022.

Several other papers have been published during the writing of this master thesis. We refer to Append C for the complete list of publications.

1.2 Thesis Outline

The thesis contains two main parts. In the first part, we present some background in blockchain technology and its scalability problem in Chapter 2. We also introduce Contribution 1 in Chapter 3 by analyzing the Lightning Network topology and discussing its trends.

After analyzing the Lightning Network, we move to the second part of the thesis, which aims to solve one of the problems found during the first analysis of the network topology. Thus, we propose our node attachment strategy as introduced in Contribution 2 in Chapter 4. We evaluate the proposed strategy and display the results in Chapter 5. We discuss the state-of-the-art research in PCNs in Section 6. Finally, we conclude our thesis in Section 7, demonstrating possible directions for future work.

Chapter 2

Blockchain and Payment Channel Networks

This chapter introduces the background and concepts used throughout this thesis. We first introduce the blockchain technology data structure and the scalability problem of public blockchains. Then, we present the payment channel networks, one of the most popular scalability solutions for public blockchains.

2.1 The Blockchain Technology

Blockchain is a disruptive technology first proposed by Satoshi Nakamoto in 2008 to solve the *double-spending problem* and allow money transfers through the Internet without the need for intermediaries [19, 20]. The double spending problem occurs when the same asset is used more than once. Double spending is more challenging to prevent in digital settings, given that bits can be easily replicated by a computer. Blockchain technology solves the double-spending problem by offering a data structure based on an immutable and distributed ledger to log the transaction history and keep track of all asset operations in a financial environment.

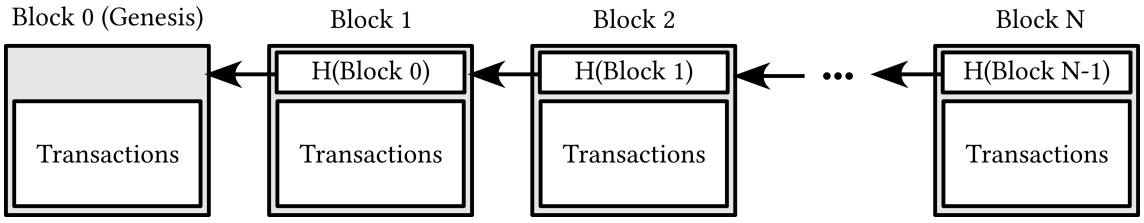


Figure 2.1: Blockchain data structure. The H(•) operator represents the hash function. The hash chain guarantees the integrity of the whole blockchain.

Figure 2.1 depicts the blockchain data structure, which is replicated in every participant of a blockchain network. The transactions are organized in blocks that are chained using cryptographic hash functions. Each block, except the first one

(genesis block), contains its hash and a field that "points" to the hash of the previous block. Modifying a single bit in any block of the blockchain alters the block hash and breaks the hash chain and the chain integrity. Because the blockchain is replicated in each consensus participants, malicious acts that might compromise its integrity can be easily detected by any participant. Each block also contains digitally signed transactions of asset operations in the network, such as payments and purchases, guaranteeing the non-repudiation of transactions.

To assert chain consistency, network participants must agree on the block sequence through a distributed consensus mechanism [19]. In distributed systems, consensus is defined as the process by which a set of independent participants reach a common decision [21, 22]. These participants exchange messages through the network or use shared memories to reach this common decision [23]. In blockchain-based systems, network participants interested in receiving fees propose blocks to be appended to the blockchain. The proposed blocks are broadcast through the network, which decides to either accept or reject it based on the correctness of the block [20].

2.1.1 The Blockchain Scalability Problem

Although blockchain technology has revolutionized distributed digital transfers, public cryptocurrencies, the main blockchain application, still suffer from scalability problems [2]. It is worth noting that, unlike the usual notion of scalability in distributed systems that indicates a system's ability to maintain its performance with an increasing number of participants, scalability here refers to the blockchain (in)ability to process a high amount of transactions per second. Public cryptocurrencies pale in comparison to traditional payment methods, which discourages the everyday use of this disruptive technology. While credit cards achieve over $24,000$ transactions per second and take seconds to confirm a payment, Bitcoin, the biggest cryptocurrency in market capitalization, achieves only 7 transactions per second and takes on average one hour to confirm a transaction [2]. The low throughput of cryptocurrencies also results in high fees, as some participants are willing to pay more to get their transactions prioritized. Thus, while credit cards process transactions with little to no fees, Bitcoin's average fee reached over U$ 60 in April of 2021 [24], making micropayments infeasible. This scalability problem happens because every transaction and block on the blockchain must go through a slow and global consensus.

Vitalik Buterin, Ethereum 's [25] co-founder, formalized the blockchain scalability problem with the concept of the *blockchain scalability trilemma* [26]. This concept states that a blockchain-based system can only achieve two out of the three

following properties:

1. **Decentralization:** The capacity of the blockchain to run without relying on a small group of large centralized actors.

2. **Security:** The capacity of the blockchain to resist collusion attacks. Ideally, the blockchain should resist up to 50% of nodes attacking it.

3. **Scalability:** The capacity of the blockchain to process transactions faster than a single node can verify.

Traditional blockchain systems, such as Bitcoin, rely on a large number of participants verifying every transaction and block through the consensus mechanism. Hence, these systems achieve decentralization and security properties of the blockchain trilemma but fail to achieve scalability.

Several solutions were proposed to improve consensus throughput in public blockchains [5, 27–30]. These proposals usually delegate consensus to a small group of participants to reduce the number of transmitted messages and accelerate consensus. Thus, they usually achieve security and scalability but neglect decentralization. Other proposals focus on altering the network or blockchain data structure, such as using directed acyclic graphs (DAG) [31–33], appendable blockchains [34–36], and sharding [37–40]. Despite the success of these proposals, they are hard to implement in existing public blockchains. This happens because most proposals require significant changes to the blockchain structure, which may lead to hard-forks[1].

2.2 Payment Channel Networks

This thesis is based on another popular blockchain scalability solution called payment channels [6, 7]. Unlike previously cited solutions, payment channels operate *off-chain*, meaning that payment channels do not usually require significant changes to the blockchain structures. Payment channels were first implemented in 2013 as *micropayment channels*, a mechanism in Bitcoinj, Bitcoin's Java implementation, to allow sending and receiving small payment amounts in a trustless setting without using the blockchain [41–43].

Payment channels are a blockchain-based technology that allows two users to pay each other privately and in real-time. The main premise of this technology is that the consensus protocol, despite being the main mechanism that guarantees blockchain security, is also the main cause of delays and excessive fees. Figure 2.2 shows the regular operation of opening and closing a payment channel. Users who are

[1]Hard forks are backward-incompatible modifications to the blockchain that force all nodes to update their software.

interested in transacting create a payment channel by issuing a funding transaction on the blockchain, allocating part of their funds to an "off-chain" channel. The allocated funds are locked on a 2-of-2 multi-sig address, meaning they can only be used in the channel and with the signatures of both parties. In this channel, users can transact freely and securely through signed cryptographic messages that can be published in the blockchain for resolution in case of a dispute. Thus, payment channels allow the maximum possible number of transactions to be carried out directly between the parties involved, without being published in a block, and to use the consensus protocol only when necessary. Off-chain transactions only require a signature from each party involved [6].

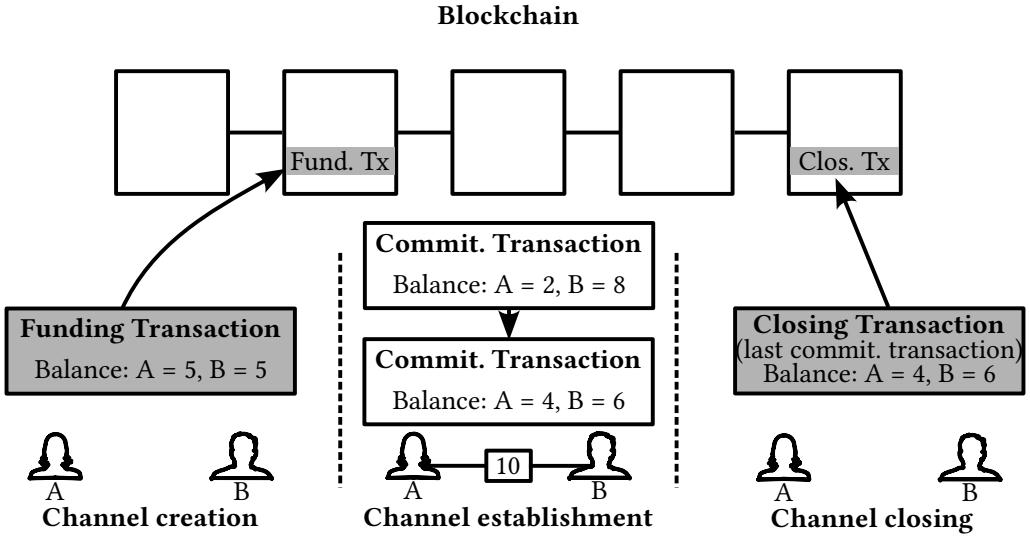


Figure 2.2: Opening and closing a payment channel. In the figure, users A and B contribute 5 coins each and issue the funding transaction to create a payment channel. The value 10 in the rectangle between the two users shows the maximum capacity of the channel, which represents the maximum amount of coins that can be transacted on the channel. After the channel is created, A and B can make payments without issuing blockchain transactions. Thus, A issues a payment of 3 coins and, shortly afterward, B makes a payment of 2 coins. To close the channel, all it takes is for one of the users to issue the last transaction in the blockchain with the most up-to-date balance.

In payment channel networks, users use established channels to route payments through intermediaries in exchange for a small fee. The set of payment channels established between system users forms a network of payment channels, as shown in Figure 2.3. Each link has a capacity, shown inside the rectangle in the figure, which indicates the total *tokens* reserved by the parties on that channel. This information is accessible and stored in the funding transaction, which is issued in the blockchain to create the payment channel. The balances of each party of the channel, represented by the numbers at each end of the edge, indicate the current state of the link, i.e., how much each participant has to transact. The balances of a

two-party channel are private to the rest of the network for security, since payments could be tracked if the balances were public, and for scalability reasons, given that it would be necessary to flood the network with update messages with each payment.

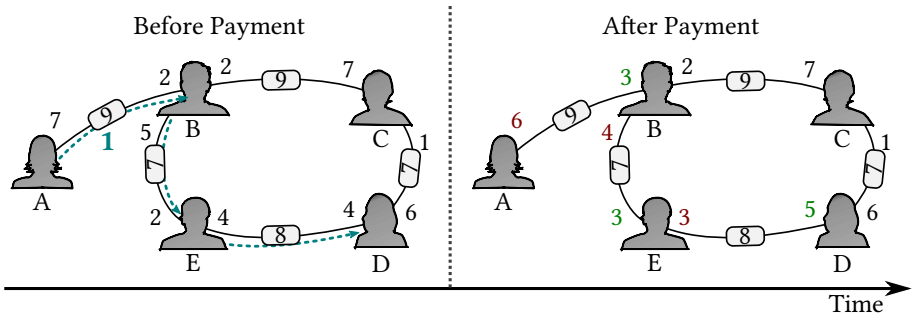


Figure 2.3: An example of 1 *token* payment from user A to user D in a payment channel network. The payment goes through the indicated path, modifying the channel balances along the payment path. Channel capacity, indicated inside the rectangles, represents the total amount allocated by both parties and is constant during a channel's lifespan.

We provide a formal payment channel network definition based on the one proposed by Malavolta et al. [44]:

Definition 2.2.1: Payment Channel Networks (PCN)

A PCN is a bidirectional graph $\mathbb{G} \coloneqq (\mathbb{V}, \mathbb{E})$*, where* $\mathbb{V}$ *is the set of nodes and* $\mathbb{E}$ *is the set of opened payment channels. A payment channel network depends on an underlying blockchain* B *and has the following operations:* `openChannel`*,* `closeChannel`*,* `pay`*. We describe these operations below.*

- `openChannel`$(i, j, \beta, t, f) \rightarrow \{1, 0\}$*. To open a channel, two users use their blockchain address* i *and* j *to issue a funding transaction containing information on the channel capacity* β*, a timeout* t*, and a fee value* f*. If the operation is authorized by* i *and* i *owns* β *coins, a new channel is created by registering its opening on the blockchain* B *with attributes* $(c_{ij}, \beta, t, f) \in \mathbb{E}$ *and the operation returns 1. Otherwise, it returns 0.*

- `closeChannel`$(c_{ij}, b) \rightarrow \{1, 0\}$*. To close a channel, one of the participants only needs to inform the channel identifier* $c_{ij} \in \mathbb{E}$ *and the channel balance* b*. The operation removes the corresponding channel from* $\mathbb{G}$ *and registers the closing transaction with balance* v *in* B*.*

- `pay`$((c_{i1}, c_{12}, ..., c_{nr}), v) \rightarrow \{1, 0\}$*. To issue a payment to user* r*, user* i *identifies the payment path through the channel identifiers* $p = c_{i1}, c_{12}, ..., c_{nr}$ *and the payment value* v*. The payment only succeeds if each of the channels on the way, e.g.,* c_{34}*, have balance* $b_{34} \geq v_k^{'}$*, where* $v_k^{'} = \sum_{j=1}^{k-1} f(p)$ *and* $f(p)$ *is the fee charged at each channel of the path. In that case, the operation returns* 1*. Otherwise, it returns* 0*.*

2.2.1 Hashed Timelock Contracts (HTLCs)

PCNs assume that intermediary nodes cannot be trusted. As payments are routed through multiple independent channels, there must be a scheme to secure this payment from possible malicious actors. Hashed Timelock Contracts (HTLCs) establish a mechanism that guarantees the security of payment delivery. These contracts are set at each hop on a payment path among users sharing a payment channel and ensure that the intermediary is only paid when the payment is complete.

When a user wants to route a payment to a destination going through intermediaries, the payment is not delivered right away. Instead, the user locks the payment using HTLCs, which are contracts that are locked with an associated hash and a time. HTLCs establish two conditions to unlock the payment:

1. The destination must reveal a secret that generates the associated hash.

2. If the time associated with the HTLC passes, the hash condition is no longer

valid and the user who established the HTLC can retrieve her payment. Thus, the payment timeouts.

Condition 1 guarantees that intermediaries in the payment path are not able to claim the payment before the payment is complete. The secret associated with the HTLC is usually generated by the final payment destination before the payment begins. The secret is hashed and the hashed version is sent to the payment source to create the first HTLC. The hash condition is the same along the complete payment path. The destination awaits the establishment of the payment and the HTLCs along the way to reveal this secret. Condition 2 guarantees that the payment lock is finite.

In the payment example depicted by Figure 2.3, user D first generates a secret x and sends the hash of this secret $H(x)$ to user A. User A finds a payment path through users B and E and establishes the first HTLC with user B, locking 1 *token* in the channel A-B. User B generates an HTLC with user E using the same hash condition initialized by user A. User E also generates an HTLC in channel E-D. When user D receives an HTLC with hash $H(x)$, user D reveals x, which unlocks the whole payment chain and the payment is complete. Using this mechanism, a malicious intermediary, e.g. B, cannot steal coins from A before the payment is complete, given that x is unknown to B before the payment reaches its destination.

2.2.2 Routing and Rebalancing in PCNs

Routing payments in PCNs is challenging and radically differs from routing datagrams in computer networks. As described in the `pay` operation, when a payment is routed on a channel in PCNs, the channel loses the capacity to forward future payments in that direction. Thus, when payments concentrate on one direction, the channel can become *depleted* and unable to route payments in the direction of these payments. Figure 2.4 shows an example where the channel from n_1 to n_2 is depleted (the capacity is zero). The user, then, has two options to reactivate (rebalance) the depleted payment channel. The first approach is closing and reopening the channel or acquiring liquidity through a liquidity provider [45]. As these approaches rely on blockchain transactions, it is slow and costly. Furthermore, by closing the channel for reopening an active one, the user interrupts the channel's lifespan, halting payments and hindering fee collection. Besides, while the reopening transactions or liquidity purchases are unconfirmed in the blockchain, the node cannot issue payments or collect fees in the depleted channel. In the example of the figure, n_1 is unable to route payments in the direction of n_2 and, therefore, cannot collect fees or issue payments.

The second and most efficient approach to rebalancing depleted channels is to issue a self-payment through a circular route in the network [16–18, 46, 47]. Fig-

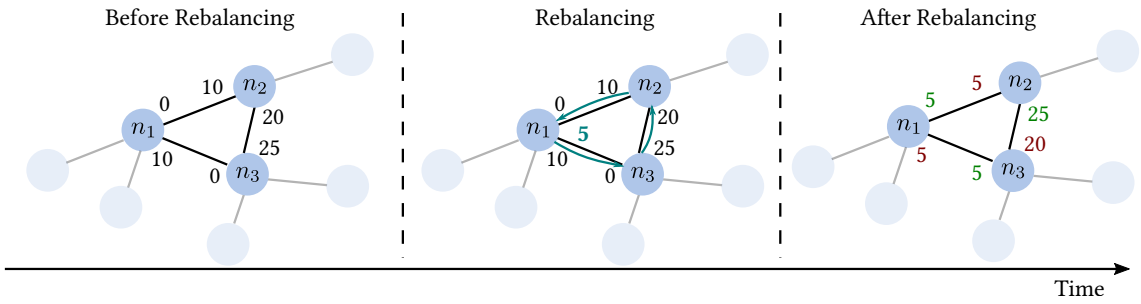


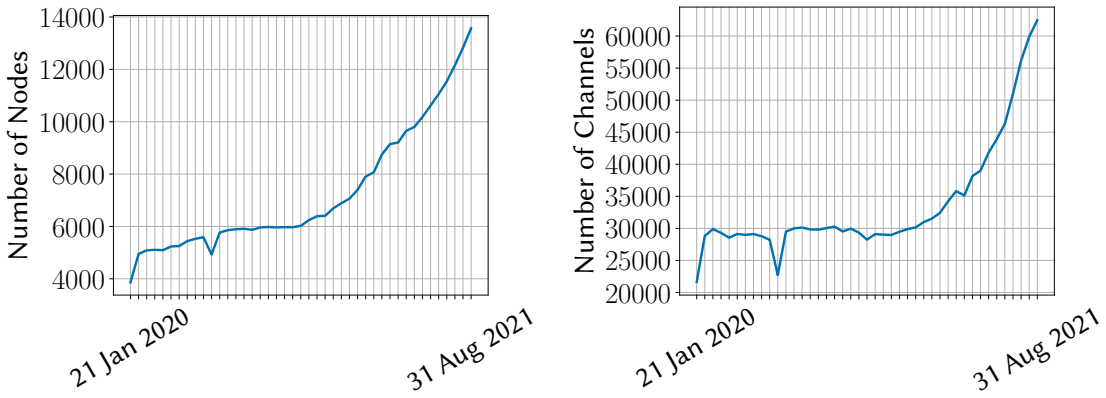
Figure 2.4: Channel rebalancing through self-payment in a circular route. Node n_1 finds a cycle and issues a self-payment of value 5, rebalancing its channel with n_2.

ure 2.4 illustrates the process of off-chain rebalancing. The key idea is to move funds from low-demand channels to high-demand ones, keeping the high-demand channels alive for a longer period of time. Although this approach offers a low-cost alternative to closing and reopening the payment channel, it depends on the existence and availability of cycles in the network topology. If cycles are unavailable, users have no other option than to resort to the blockchain. In this thesis, we verify that 36% of nodes in the Lightning Network, the most popular PCN currently, do not participate in cycles. Furthermore, even nodes that do participate in cycles usually have to pay high rebalancing fees, which restricts user adoption to off-chain rebalancing (see Chapter 3).

2.3 The Lightning Network

PCNs became more popular after the implementation of the Lightning Network in 2017, Bitcoin's and first PCN ever launched. The Lightning Network was proposed by Poon and Dryja [6] in 2016. The Lightning Network has experienced significant growth from 2020 to 2021. Figure 2.5 shows the growth of LN in terms of nodes and channels. In 18 months, the number of nodes in LN tripled, going from under $4,000$ to over $13,000$ nodes. Similarly, the number of channels went from a little over $20,000$ to over $60,000$ in the same period. As of November 2023, LN is the biggest PCN, having over $\$190,000,000.00$ distributed among $14,000$ nodes and $61,644$ public channels [48]. Due to its size and rapid growth, LN is the reference PCN currently and is the object of several studies [13, 49–51].

In LN, router nodes receive a small fee in exchange for routing services. Particularly, LN defines two types of fee: a fixed base fee, which is charged for every forwarded transaction, and a proportional fee, which varies with the payment value. A user A who is interested in paying a user B, calculates a route to B using Dijkstra's Shortest Path First (SPF) algorithm and computes the fee value charged by every node on the path. Usually, the selected path is the one that presents the



(a) Number of nodes in the Lightning Network.

(b) Number of channels in the Lightning Network.

Figure 2.5: Lightning Network's growth from January 2020 to August 2021.

smallest fee sum, meaning the cheapest path to user A. Then, A adds the sum of the fee values to the payment value and sends it through the computed route.

The Lightning Network standardizes the message formats and protocols through the Basis of the Lightning Technology (BOLTs) [52], documents based on the Internet Request for Comments (RFC) that formally describe how the network should be implemented. Currently, LN has three major implementations: `c-lightning` [53], Lightning Network Daemon (LND) [54], and `eclair` [55].

This thesis uses messages received at one of the network nodes to reconstruct the topology at different moments in history. In particular, the BOLT 2 document defines the announcements of public channels through channel announcement messages that are broadcast in the network through a gossip protocol [56]. In the gossip protocol, a participant disseminates a message to a specific number of neighbors, who repeat the message to their neighbors. The procedure continues until the information reaches the entire network. The network nodes can build, from the received messages, a local topology containing the active channels with their respective participants and capabilities.

In this thesis, we also use different graphs to represent the LN topology. Figure 2.6 depicts multiple representations used in this work. First, the channel graph is used in the first part of the topological analysis of the LN. In this representation, the LN topology is modeled as an undirected graph and the only considered edge attribute is channel capacity. This representation suits well the objective of analyzing some network topology features that do not require computing distances, given that it ignores channel fees. Thus, we use the first representation in the following chapter of the thesis in metrics such as transitivity, number of components, and average degree, unless the analysis requires distance measurements.

In the second part of the thesis, we move from analyzing the network topology

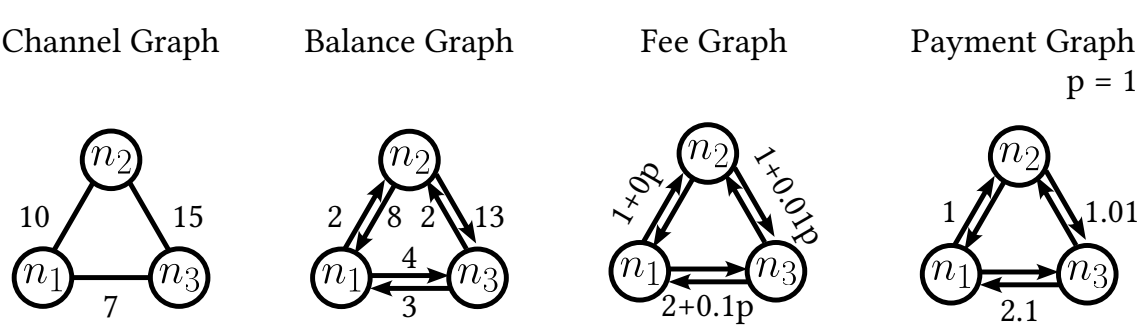


Figure 2.6: Types of graph representations of the Lightning Network considered in this thesis. Some fee attributes in the fee and payment graphs are removed from the image for clarity.

to positioning nodes to increase its probability of routing payments and simulating payments in the network. The balance graph is particularly important for simulating payments as it correctly captures the state of the network and how the capacity is divided in both directions of a channel. Nevertheless, as mentioned before, LN keeps channel balances private due to scalability and security concerns. This feature makes it difficult to determine how the capacity is exactly divided among participants. Therefore, we build the balance graph to simulate some payments in Chapters 3 and 4 by assuming that the top 10% of central nodes have channels with equally split capacity while periphery nodes have 99% of the capacity allocated towards one side of the channel.

Finally, we also use a payment graph for most of the second part of this thesis. The payment graph is directed and built from the fee graph, which is a graph with the charged fees as attributes, for a fixed payment value. Figure 2.6 shows an example of a payment graph built from the fee graph for a payment value of 1. In the figure, the fee values are presented in only one of the channel directions for clarity. Nevertheless, the fees could be different in both directions of a channel. Unless stated otherwise, the payment value used as a reference in this thesis is derived from a transaction dataset from the Ripple credit network [57, 58].

Chapter 3

Topological Analysis of the Lightning Network

The rapid growth of LN motivates the study of its topology evolution to identify possible features and trends. This chapter presents an analysis of the LN topology, discusses the centralization tendency that it presents, and how it affects the network in terms of security.

3.1 Network State, Evolution and Trends

The default routing of the Lightning Network is source-based. Thus, Lightning requires that nodes know the full topology of the network to make payments, information that can be gathered from announcement messages. In the present thesis, we use a dataset collected by Lightning Network developers of channel announcement messages transmitted using a gossip protocol for node synchronization in the Lightning Network [59]. The dataset contains three types of messages collected by a node using the Lightning Network implementation in C language, called `c-lightning`: (i) channel announcement messages, which make a channel public in the network; (ii) channel update messages, used to update the parameters of a given channel; and (iii) node announcement messages, used to inform the metadata of a node. To create the dataset, raw messages were stored in a file and a filter was used to remove duplicate messages. The file contains messages from the Lightning Network from August 2018 through August 2021. Given that the period 2018 and 2019 has already been covered by other papers [13, 14], we consider the period between January 2020 and August 2021 unless stated otherwise. All messages have a timestamp, which tells the time at which they were created using the UNIX system time base. Thus, using a time window, it is possible to reconstruct the topology of the Lightning Network at any instant in time. This work uses two-week time windows for topology analysis,

i.e., it ignores messages prior to a two-week period when reconstructing the network at a given instant.

Network state. The first set of experiments checks the state of the message dataset from August 17 to August 31, 2021, to estimate the resource concentration and connectivity in the Lightning Network. Figure 3.1 shows the distribution of channel routing capacity in the network. The channel routing capacity is the total number of tokens allocated by the two parties during its creation. We note an intense concentration of funds in the network, where few channels have high capacity and multiple channels have low resources. The capacity of the channels is a decisive factor for transaction routing. The low amount of high-value channels, e.g., only 13 channels (0.02% of the network) with capacity above US$ 85,678.60 ($\approx$ 2 BTC)[1], restricts the amount of paths that high-value payments can use. Although the value seems high, payments of 2 BTC or more are not uncommon on the Bitcoin Network [60]. By concentrating high-capacity channels on a few paths, most of the network must resort to blockchain to make high-value payments, experiencing high confirmation time and high fees.

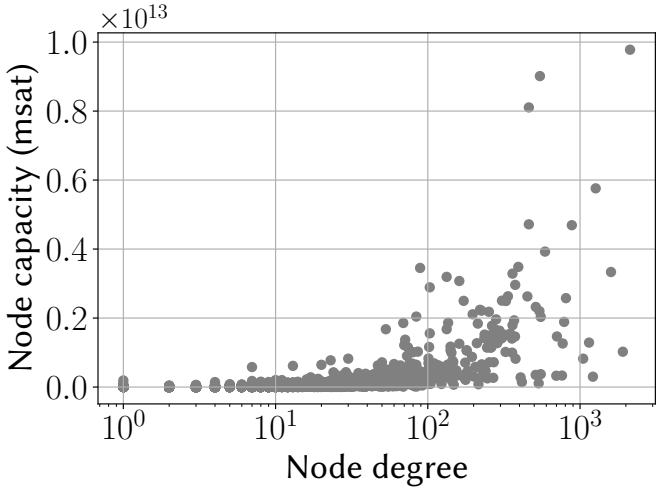
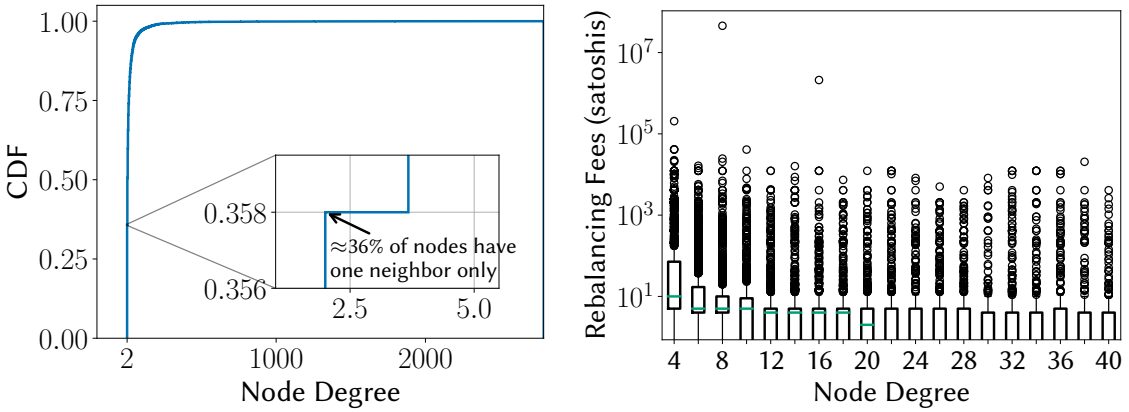


Figure 3.1: Relationship between node degree and node capacity in the Lightning Network.

Figure 3.2a shows the degree distribution in the Lightning Network in July 2022. While the majority of nodes in the Lightning Network have a low degree, a small number of nodes concentrate the network connectivity. In fact, we see that almost 36% of nodes in the Lightning Network have only one neighbor. Although these nodes may create cycles, they still have to endure high fees due to the blockchain, and current automatic channel creation heuristics present no financial benefits to the user. Thus, single-neighbor nodes are unable to rebalance their channels through off-chain methods and have to rely on slow and expensive blockchain payments to

[1]Exchange rate on January 13th, 2024.

keep their channels alive once they get depleted. To make matters worse, these nodes usually present low-capacity, in contrast with high-capacity nodes that can easily perform off-chain rebalancing. This contrast deepens the network inequality, which recent works show to already be extremely high [51, 61].



(a) Node degree cumulative distribution in the Lightning Network in July 2022.

(b) Rebalancing fees as a function of the node degree in the Lightning Network.

Figure 3.2: Network metrics of the Lightning Network in July 2022. We verify the scarcity of cycles in the network for a high number of nodes, hindering several rebalancing operations. Furthermore, we verify how rebalancing fees are highly unequal to nodes that do participate in cycles.

We also verify the availability of cheap rebalancing routes among nodes that have more than one neighbor. We run Dijkstra's Shortest Path First (SPF) algorithm that uses fees as a metric to select the shortest path. As fees on PCNs depend on the payment value, we set the payment value to the average value of a transaction dataset collected in the Ripple credit network[2] [62]. We find that even using off-chain rebalancing, the distribution of rebalancing fees is highly unequal in the network. Figure 3.2b shows that, while high-degree nodes usually find cheap routes to rebalance their channels, low-degree nodes have fewer options and end up paying more for each rebalancing operation.

Network Evolution and Trends. The second set of experiments checks the evolution of network metrics to assess whether the degree of network concentration has been increasing. The experiments evaluate the network in the period between January 21, 2020 and August 31, 2021. Figure 3.3 shows the evolution of the Lightning Network metrics. All metrics consider only the largest component of the graph, except the number of components result. Although the number of components in the network grows from 6 to 53 during the evaluated period, as shown in Figure 3.3a, the number of nodes of the smallest components is negligible compared to the number

[2]The Lightning Network keeps payment values private by design and thus no payment dataset is available. We hold that Ripple's credit network works similarly to PCNs and, therefore, has similar payment value ranges.

of nodes of the largest component. As an example, the network snapshot with the largest number of components has 13,462 nodes in the largest component, while the smallest components have at most 3 nodes. Thus, using only the largest component does not affect the reliability of the results, given that the smallest components do not make payments with more than 1 hop and cannot meet the demands of most of the network. The increase in the number of components is natural due to the growth in the number of nodes in the network. Some nodes establish connections with each other to make recurring payments and do not seek to establish connections with other participants.

The assortativity r of a graph indicates the connection preference of nodes in the network. A dissortative graph has $r < 0$, indicating that low-degree nodes in the network prefer to connect to high-degree nodes, while an assortative graph has $r > 0$, indicating that high-degree nodes prefer to connect to high-degree nodes. The network's assortativity coefficient, in Figure 3.3b, shows that the Lightning Network is disassortative, indicating that low-degree nodes tend to connect to high-degree nodes. This is because some Lightning Network implementations, such as Lightning Daemon (LND), adopt policies that connect new nodes to more central nodes in the network [13]. This strategy is adopted because nodes that intend to make payments to more than one entity may prioritize connecting to more central nodes, seeking lower paths and fees. Despite the disassortativity of the network, we can see a growth tendency in this metric, pointing to an assortative network in the future. This can be explained by the establishment of a connection between the central hubs after they consolidate, increasing the assortativity. The upward trend in assortativity can be explained by the emergence of new hubs as a result of the rapid growth of the network.

The graph transitivity measures the fraction of cycles existing in the network relative to triads, which are subgraphs formed by 3 nodes. Thus, the transitivity measure only considers 3-node cycles and varies between zero and one. The transitivity T of a graph is given by

$$T = 3\frac{l_3}{t}, \tag{3.1}$$

where l_3 is the number of cycles between 3 nodes in the network and t is the number of triads. In PCNs, establishing cycles of a few hops is crucial to enable low-cost rebalancing proposals [11, 12]. These proposals take advantage of cycles so that a node can make payments to itself, rebalancing paths by moving money between channels. Keeping channels balanced is of utmost importance to increase the probability of a successful transaction [63]. Nevertheless, the low transitivity, presented in Figure 3.3c and its downward trend in the network point to the existence of few

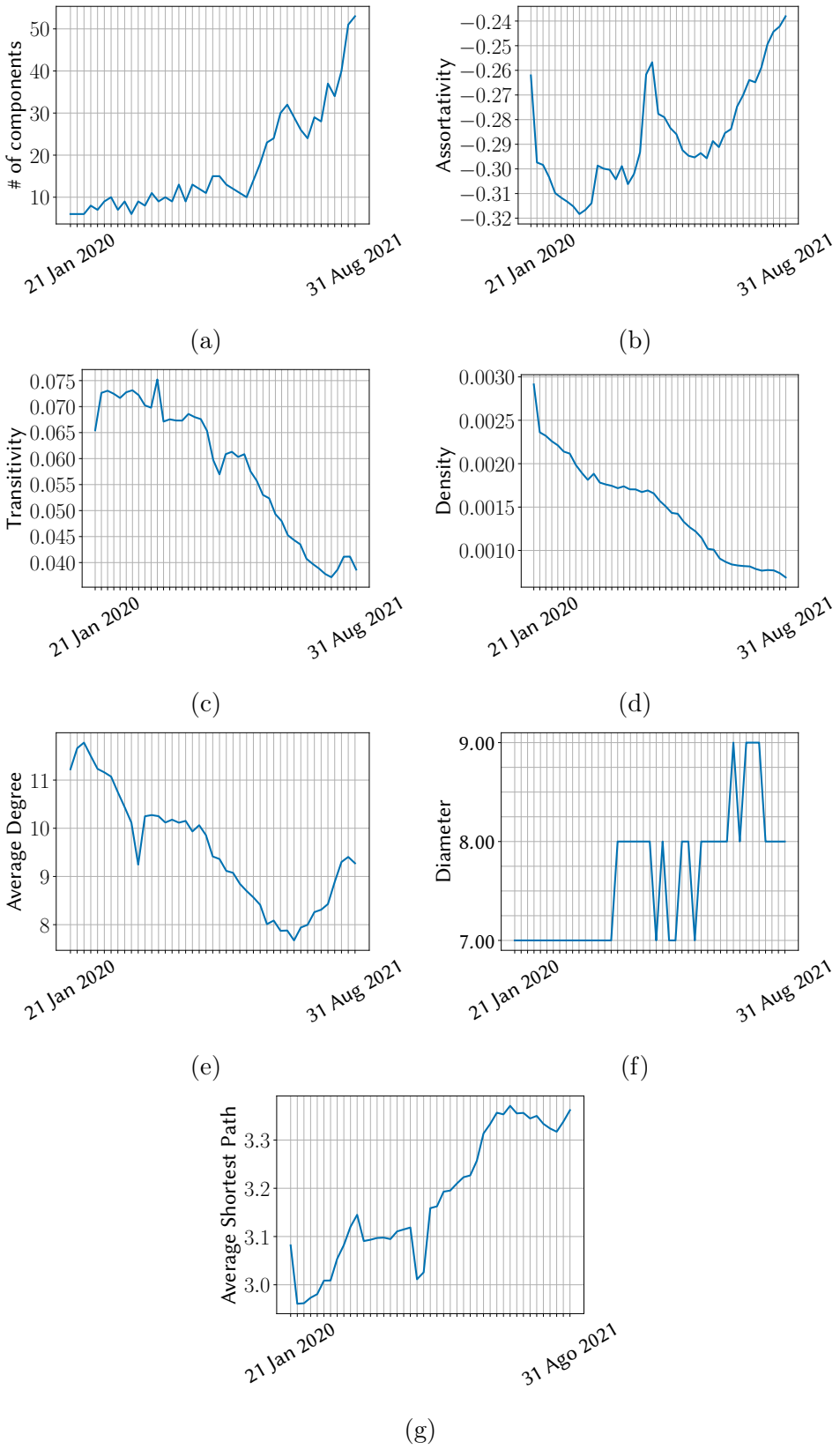


Figure 3.3: Evolution of the graph metrics of the Lightning Network over the period from January 21, 2020 to August 31, 2021.

3-node cycles, which makes proposals expensive and possibly unfeasible for most participants who have low financial power. This low transitivity is a natural consequence of the establishment of connections between low-degree nodes to high-degree nodes, generating several possibilities of cycles that are not formed. The centralized topology tends towards a core-periphery structure, in which most of the nodes are at the extremities, i.e., at the periphery. Thus, these nodes have only one neighbor, which does not contribute to the formation of triangles in the network, lowering transitivity.

The density of a graph is given by the fraction of the edges existing in the graph and the edges of a complete graph with the same number of nodes. Thus, the density of an undirected graph can be calculated by

$$d = \frac{m}{n(n-1)}, \tag{3.2}$$

where m is the number of edges in the network and n is the number of nodes. The low density observed in Figure 3.3d shows that the Lightning Network is extremely sparse. The downward trend in density can also be explained by the connection of new nodes to more central nodes. This type of behavior is common in scale-free networks, where a small portion of nodes concentrate a large part of the connections. The behavior of the Lightning Network as a scale-free network can also be observed in the drop in the average network degree, as shown in Figure 3.3e. The entry of one-connection only nodes contributes to the drop in the average network degree even as the degree of the most central nodes increases.

Figures 3.3f and 3.3g expose the low diameter and average shortest-path in the Lightning Network. The diameter of a graph is defined by the longest shortest path between any pair of nodes. The average shortest path is obtained through the sum of the minimum distance for each pair of vertices of the graph divided by the total number of vertices. Despite the number of nodes in the network tripling in the period, the two metrics present low variation if compared to the other metrics, varying by 28% in the case of the diameter and approximately 15% in the case of the shortest path.

The centralization trend becomes even clearer when analyzing Figure 3.4, which shows the resource concentration of the nodes over time. It can be seen that while in January 2020 approximately 2.0% and 0.9% of nodes concentrated 80% and 50% of the total network capacity, respectively, the numbers dropped to 1.1% and 0.3% in August 2021. The downward trend, albeit smooth, shows that a small set of nodes concentrates much of the allocated revenue and exposes future vulnerabilities and security challenges in the network.

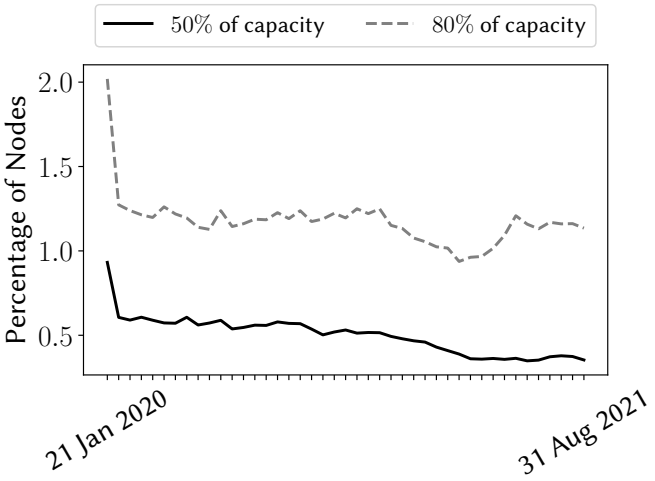


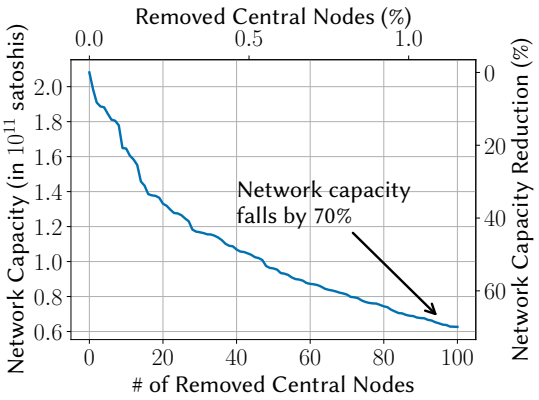
Figure 3.4: Resource concentration in the Lightning Network from January 2020 to August 2021. The dashes indicate the percentage of nodes that concentrate 50% and 80% of the total network capacity in each two-week period.

3.2 Impact of Topological Attacks

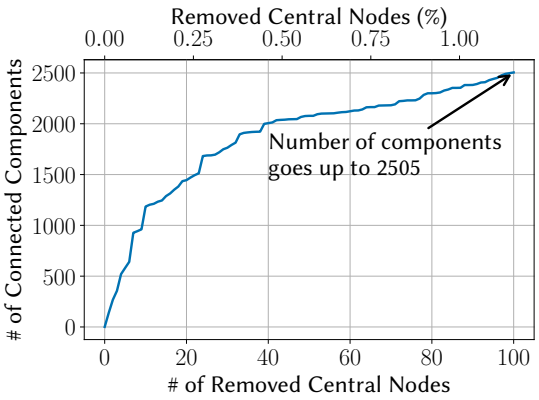
Creating cycles in the network is important not only to enable rebalancing operations but also to improve network robustness against targeted topology attacks [64, 65]. Several works point out that, despite being proposed as a completely decentralized network, the Lightning Network is becoming more centralized by concentrating connectivity and capacity in a small number of nodes [51, 61]. This centralization trend leads to vulnerabilities against attacks targeted at the highest-degree nodes. We simulate some of these attacks by considering an attacker that has access to the network topology and targets nodes with the highest degree in the network. This attacker can perform a DDoS attack on each of the target nodes and impair their operation temporarily in the network.

Figure 3.5a shows that impairing only 44 nodes ($\approx 0.5\%$ of nodes in the network) reduces the network capacity by half. Furthermore, extending the attack to 100 highest-degree nodes ($\approx 1.2\%$ of the network) leads to a decrease in capacity by 70%. Reducing the network capacity directly affects the payment success rate, leading to multiple failures due to a lack of available channels [8, 66]. Furthermore, Figure 3.5b also demonstrates the impact of targeted attacks on the number of network components. We verify that attacking 100 nodes increases the number of components by over $2,500\times$. In fact, removing the highest-degree node from the network leads to network partitioning in 140 components. This network partitioning pattern has drastic effects on PCNs, as it makes several payments infeasible.

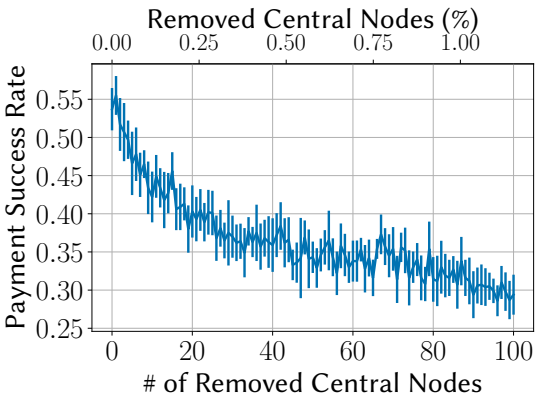
Figure 3.5c shows how the network partition affects the payment success rate in



(a) Network capacity in satoshis as high-degree nodes are removed.



(b) Network components as high-degree nodes are removed.



(c) Payment success rate as high-degree nodes are removed.

Figure 3.5: Impact of topological attacks in the Lightning Network as of July 2022. We simulate central nodes being impaired in a targeted attack and verify the effect on network capacity and partitioning.

the network. We implement a Lightning Network simulator[3] and simulate a PCN using a Lightning Network snapshot from July 2022 as network topology and a Ripple transaction dataset as workload [57, 62]. With the removal of 100 highest-degree nodes, the payment success rate falls from over 55% to less than 30%. The failed payments must be completed on the blockchain, incurring high fees and slow confirmation time. It is worth noting that these attacks are not restricted to theory. The Lightning Network was the target of a DDoS attack that affected 20% of the network nodes [51, 67].

[3]Available at `https://github.com/gFrancoCamilo/ln-looprebalance`

3.3 Discussion

Despite presenting itself as a fully decentralized alternative for fast payments in blockchains, the results show a clear centralization of resource and network connectivity. The resource concentration is evident when analyzing the distribution of capacity and channel degrees in Figure 3.1. Higher capacity nodes hold a monopoly on routing high-value payments in the network, getting richer by collecting more fees. These higher capacity nodes attract connections from new participants, creating hubs, as new participants seek to open channels in a location that can reach multiple destinations in a smaller number of hops. Thus, participants choose to connect to the more central nodes seeking to lower the routing fee paid when making a payment.

Low connectivity of nodes in the periphery also makes proposals that take advantage of cycles to rebalance channels unfeasible or expensive in practice. These proposals are restricted to central nodes, that have greater connectivity and are more likely to find paths with cycles for rebalancing. While nodes with higher connectivity can easily rebalance their own channels using the network topology, most network participants must resort to the blockchain. This phenomenon further accentuates network inequality, given that central nodes are less likely to experience periods of routing unavailability when waiting for transaction confirmation in the blockchain, nor do they pay high fees to rebalance their own channels.

Possible solutions to the centralization problem include network-attachment proposals that prioritize distributing connectivity when opening channels. In this case, the creation of alternative paths in the network ensures less dependence of peripheral nodes on centralized nodes and reduces the effects of split attacks on the network. Furthermore, connecting new nodes by creating cycles in the topology mitigates the impact of problems related to low transitivity, such as rebalancing channels through the network topology. Thus, generating cycles when opening channels ensures that nodes can rebalance their channels without having to resort to blockchain.

Chapter 4

ProfitPilot: Profitable Cycles for Low-Cost Rebalancing in LN

We present ProfitPilot, an algorithm for strategically adding new nodes to payment channel networks that focuses on the creation of 3-node cycles, enabling efficient rebalancing operations to mitigate the scarcity of cycles and the centralization tendency cited in Chapter 2. ProfitPilot enables channel rebalancing to promote a high channel lifespan, eliminating the need for frequent channel closure and reopening, which is costly for users. We choose to focus on 3-node cycles for two main reasons. First, as fees vary over time, smaller cycles provide higher tolerance against this fluctuation. Instead of relying on $n-1$ nodes not changing their fees in a n-length cycle, the user only relies on 2 nodes, which are his/her neighbors. Furthermore, by focusing on creating 3-node cycles, we improve the overall network robustness that would otherwise be lost by only considering connections to central nodes as in previous work [68]. ProfitPilot builds complete subgraphs on top of the original network graph, providing alternative paths and mitigating the effect of targeted attacks. To make the cycle creation economically viable, our strategy offers financial incentives to encourage user adoption. Finally, 3-node cycles merge into forming cycles with higher lengths, increasing possible rebalancing paths [46]. Our focus is on establishing channels that maximize the likelihood of routing payments through them and minimize the average number of hops to other network participants. This optimization reduces transaction fees and enhances the overall efficiency of the network.

4.1 Assumptions and Network Model

Publicly-available topology. ProfitPilot relies on the network topology, including node and channel attributes, being publicly available to nodes before creating

channels. The two most popular PCNs currently, Lightning and Raiden, offer implementations that allow users to download the network topology before establishing a channel in the network. Even before adhering to the Lightning or Raiden network, users may verify several network explorers that store and display information about nodes and channels on public websites [48]. We also assume that substantial changes in the network topology are time-consuming, given that these updates are done through the blockchain [69]. Thus, only minor changes may occur during the execution of the algorithm.

Freedom to create channels. We assume that a new node coming into the network can create nodes with any node publicly announced in the network. To do so, the new node uses the `openChannel` operation described in Chapter 2. The new node, however, funds all costs of creating the channel. Although the other party can theoretically reject connections, we assume that the other party acts rationally and accepts channel creation as they do not have to fund the channel and may gain fees from payment routing. This assumption is also present in previous works in the literature [69, 70].

Source-routed payments. We strategically position a node to receive routing fees and pay, on average, low fees when compared to other nodes. Thus, we need to clearly define how payments are routed in the network. ProfitPilot assumes that nodes adopt the standard routing strategy in the Lightning Network, which uses source routing with Dijkstra's SPF algorithm to select the payment route. Although several proposals to modify routing algorithms in PCNs have been proposed [8, 62, 66, 71, 72], the current implementation of the Lightning Network still uses Dijkstra's SPF algorithm with fees as the distance metric. Furthermore, like previous work [49, 69], we assume a uniform distribution of possible sender-receiver pairs when placing the node on the network.

Network model. We model a PCN as a directed graph $G = (V, E)$, where V is the set of nodes and E is the set of channels in the network (balance graph in Figure 2.6). Each bidirectional channel, represented by two directed edges, holds a set of attributes, such as information about the routing fees. In particular, an edge $e_{ij} \in E$ stores two attributes regarding fees: a base fee f^B and a proportional fee $f^P(p_v)$. While the base fee is fixed for every transaction that is forwarded in the channel, the proportional fee depends on the payment value p_v being routed. Thus the total amount of fees accounted for in a channel e_{ij} that transports a payment of value p_v is given by $f^T_{ij}(p_v) = f^B_{ij} + f^P_{ij} \cdot p_v$.

Like a similar work [64], we account for the weight of the fees by generating a payment graph $G_P = (V, E, W)$ from the original network G, where W is the set of edge weights, as described in Chapter 2.3. Basically, the payment graph has the same nodes and edges as the original network graph but the edges are weighted by

the total fees. Thus, given a channel e_{ij}, the weight w_{ij} is defined as $f_{ij}^{T}(p_v)$ for a fixed value of p_v. Users may define different fees in both directions of the payment channel, i.e., given an edge $e_{ij} \in E$, it is possible that $w_{ij} \neq w_{ji}$. As total fees depend on payment values, which are kept private on the Lightning Network [6], we use the average payment value of a transaction dataset collected from the real-world Ripple credit network [62].

4.2 Requirements

We identify the following requirements for our attachment strategy design:

1. **Profitability:** As channel creation incurs expenses and the creation of cycles requires more than one channel to be established, we focus on the creation of profitable channels that compensate for the opening costs during its lifespan.

2. **Economy:** As the user does not know beforehand who is a potential payment destination, we must create channels as close as possible to all other nodes in the network so the new user pays fewer fees overall.

3. **Cycle-enabled:** As we focus on enabling rebalancing operations, newly-created channels must have at least one cycle to rebalance. Particularly, we focus on short cycles to add redundancies and increase network robustness.

The above requirements aim to encourage user adoption. The list presents significant incentives from a user's point of view, but nothing is said of the network. From the network perspective, we hold that the requirement of being cycle-enabled creates redundancies that attenuate the current trend of network centralization in the long-term [51, 65].

4.3 Problem Formulation

We formulate the following problem. Consider a payment channel network described by a directed graph $G = (V, E)$ and a user u interested in creating new channels in a PCN. The user u can create channel a e_{uv} with any $v \in V$. The fee is composed of a fixed base fee f_{uv}^{B} and a proportional fee $f_{uv}^{P}(p_v)$ proportional to the value of payment p_v. These fees combine into a total fee $f_{uv}^{T}(p_v) = f_{uv}^{B} + f_{uv}^{P}(p_v)$ charged to route p_v on the channel e_{uv}. Like Ersoy *et al.* [69], we define an event of a transaction with source s and destination t and value p_v passing through the established channel as $X(p_v, s, t)$. We model the expected gain of the user u in the newly created channel e_{uv} as a product of probabilities, given by [69]:

$$\mathbb{E}[G_u] = \sum_{\substack{\forall s,t \in V \\ s \neq t \neq u}} \sum_{p_v=1}^{T_{max}} \sum_{v \in C} Pr[P = (s,t)] \times Pr[T = p_v] \times f_{ui}(p_v) Pr[X(p_v, s, t)], \quad (4.1)$$

where $Pr[P = (s,t)]$ is the probability of a payment occurring between nodes s and t, modeled by a distribution P; $Pr[T = p_v]$ is the probability of a payment, modeled by the statistical variable T, which can assume a maximum value of T_{max}, having a value p_v. Finally, $Pr[X(p_v, s, t)]$ is the probability of the payment of value p_v going from s to t passing through channel e_{uv}, given $C = V - \{u\}$. The gain is given by the product of all three probabilities.

Besides maximizing the financial gain, described by Equation 4.1, the user u also expects to pay fewer fees when issuing payments. Unlike previous work [69, 70] that solely focuses on collecting fees, we consider that the new user might want to issue payments in the network. As we assume that u does not know who he/she is going to transact with before joining the network, we aim to reduce u's distance to every other node in the network.

$$\mathbb{E}[C_u] = \sum_{\substack{\forall t \in V \\ t \neq u}} \sum_{p_v=1}^{T_{max}} \sum_{\forall t \in V t \neq u} Pr[P = (u,t)] \times Pr[T = p_v] \times f_{ut}^{T}, \quad (4.2)$$

where $f_{ut}^{T}(p_v)$ is the fee charged by the channel to forward a payment of value p_v from u to t.

To simplify the problem, we assume the probability of payment from s to t in Equation 4.1 to be modeled as a uniform distribution. Thus, $Pr[P] = \frac{1}{(|V|-1)(|V|-2)}$ is constant, and $|V|$ is the number of vertex in the network. We also consider a fixed p_v value, eliminating the second sum and the fee charged $f_{uv}^{T}(p_v)$. Equation 4.1 can, then, be rewritten as $\mathbb{E}[G_u] = \sum_{v \in C} Pr[X(p_v, s, t)]$. Finally, we adopt the betweenness centrality to model the probability $Pr[X(p_v, s, t)]$ of the event X. As PCNs use Dijkstra's SPF, we consider that the betweenness centrality of a node provides an accurate prediction on the probability of that node forwarding payments.

Definition 4.3.1. ***Node betweenness centrality.*** *A node's u betweenness centrality is proportional to the number of shortest paths that traverses u and is defined as*

$$bc_u = \sum_{\substack{s \neq t, \\ \sigma_{st} \neq 0}} \frac{\sigma_{st[u]}}{\sigma_{st}},$$

where σ_{st} is the number of shortest paths between s and t, and $\sigma_{st[e_{uv}]}$ is the number of shortest paths between s and t that goes through u.

Thus, we rewrite Equation 4.1 as $\mathbb{E}[G_u] = bc_u$. The same logic is applied to Equation 4.2, associating it with closeness centrality.

Definition 4.3.2. ***Closeness centrality.*** *The closeness centrality of a node u is inversely proportional to its distance to other nodes and is defined as*

$$cc_u = \sum_{v \in V} \frac{1}{d_{uv}},$$

where d_{uv} is the distance between u and v in total fees.

As we set the distance of a node to another as the total fee charged by the channel, we can use closeness centrality to infer the average amount of fees paid by our node. Thus, the sum of distances acts as a natural average for the fees encountered by u assuming a uniform distribution of transaction values [49]. We rewrite Equation 4.2 as $1/\mathbb{E}[C_u] = cc_u$, where cc_u is the closeness centrality of u. As closeness centrality is inversely proportional to the sum of distances, the higher a node's closeness centrality the smaller the distance between the node and the rest of the network. Therefore, instead of minimizing costs, we combine both closeness and betweenness centrality into a metric that we aim to maximize. We introduce the expected incentive $\mathbb{E}[I(u)]$ of node u as

$$\mathbb{E}[I(u)] = bc_u + cc_u. \tag{4.3}$$

4.4 Greedy Algorithm for Node Attachment

We propose a greedy algorithm that recommends channel connections to ingress nodes in the network. The goal of the algorithm is to create 3-node cycles while maximizing a user's probability of forwarding a payment and reducing the average amount paid in fees when issuing a transaction. We achieve both of these features through centrality metrics. First, as current PCNs use a simple Dijkstra's SPF algorithm to select the cheapest payment route, we can fulfill our goal of increasing the node's probability of forwarding a payment by creating channels that belong to as many shortest paths as possible in the PCN. In other words, for each channel created the user is interested in having more payments routed through her node. To that end, we employ the *node betweenness centrality*, which measures the fraction of shortest paths crossing the node, as one of the optimization objectives (Appendix 4.3 gives the mathematical formulation). Increasing a node's probability of routing is equivalent to increasing its betweenness centrality.

Second, we employ the *closeness centrality* to measure how far from every other node the new user is on the network. ProfitPilot aims to minimize the overall

distance to everyone else in the network, as it assumes that the new user does not know beforehand to whom she is going to send payments. Closeness centrality suits well in our proposal as it measures the overall distance of a node to the rest of the network. We set the weight of the closeness centrality as total fees in our payment graph. Thus, the higher the closeness centrality, the closer the node is to all other nodes, meaning fewer routing fees for the new node.

Both betweenness and closeness centralities depend on the fees that will be charged on the created channel. This happens because the default Dijkstra's SPF algorithm used by some PCNs, including the Lightning Network, selects the cheapest path, i.e., the one that charges fewer fees. Thus, setting low base and proportional fees results in higher betweenness and closeness centrality, whereas setting high fees yields the opposite result. As choosing channel fees is beyond the scope of our work [1], ProfitPilot considers a user that sets the channel fees as the median of the entire network. Thus, we set the base fee to 100 milisatoshis (msat) and the proportional fee to 50 satoshis (sat)[2].

We combine the node's betweenness and closeness centrality into one incentive metric (see Chapter 4.3), R_n, described as

$$R_n = \alpha \cdot bc_n + (1 - \alpha) \cdot cc_n, \tag{4.4}$$

where bc_n represents the betweenness centrality and cc_n is the closeness centrality of node n. ProfitPilot offers a tuning parameter $\alpha, 0 \leq \alpha \leq 1$, that can be set by the user to control the weight given to the betweenness centrality and the closeness centrality in the incentive computation. If the new user on issuing a small number of transactions, she can set $\alpha > 0,5$, prioritizing gains from fee collection. Conversely, if the user plans on issuing several transactions and wants to pay fewer fees on average, the user may set $\alpha < 0,5$.

At each step, ProfitPilot greedily looks for the neighbor with the highest incentive metric increase. ProfitPilot's algorithm, detailed in Algorithm 1 receives the PCN topology G and the maximum number of channels k to create as input and returns the recommended neighbors for a new node. ProfitPilot simulates the creation of multiple channels offline and computes the incentive return. The algorithm iterates through the nodes of the network creating channels with each one and stores the node N_+ that offers the highest incentive R_M on each round in a list N. This procedure continues until the incentive no longer increases, meaning it has reached a local maximum, or until the list N reaches k channels. If the incentive starts to

[1] Readers interested in fee selection may refer to Ersoy et al. [69].

[2] A satoshi is the atomic unit of Bitcoin and is equivalent to 10^{-8} BTC. Nevertheless, operations in milisatoashis (1 satoshi = 1,000 milisatoshis) are common in the Lightning Network. The milisatoshis are usually rounded down when closing the channel.

Algorithm 1: ProfitPilot's main algorithm.

Input : $G = (V, E) \rightarrow$ payment channel network as a directed graph;
$n \rightarrow$ node entering the network;
$\alpha \rightarrow$ parameter that indicates whether the user prioritizes collecting fees or paying low fees;
$k \rightarrow$ maximum number of channels the user wants to create.

Output : $\mathcal{N} \rightarrow$ set of neighbors recommended by the algorithm.

```
Initialize empty set of neighbors N ← ∅;
while |N| < k do
    Initialize maximum reward R_M ← 0;
    Initialize selected node N_+ ← None;
    foreach n_i ∈ V do
        Simulate channel opening (n, n_i);
        Compute incentive R_n ← ComputeIncentive(G, n, α);
        Simulate channel closure (n, n_i);
        if R_n ≥ R_M then
            if |N| > 0 then
                if n_i is neighbor of n_s ∈ N then
                    R_M ← R_n;
                    N_+ ← n_i;
                end
            end
            else
                R_M ← R_n;
                N_+ ← n_i;
            end
        end
    end
    Compute current incentive C_R ← ComputeIncentive(G, n, α);
    if R_M ≤ C_R then
        Break out of the loop;
    end
    N ← N ∪ N_+;
    Add channel (n, N_+) to network G;
end
return N
```

decrease, the algorithm stops, given that any selected channel will provide a smaller incentive than the current list.

ProfitPilot aims to create triangles in the network. After selecting the first node that presents the highest incentive, it is possible to create a 3-node cycle by verifying the selected node's neighbor instead of iterating through every node in the network. Thus, our algorithm executes the same procedure of looking for the highest incentive but in a reduced search space, saving execution time. Through this procedure, ProfitPilot guarantees the creation of $|\mathcal{N}| - 1$ triangles (see proof in Appendix A) that the node participates in, where $\mathcal{N}$ is the set of the node's

suggested neighbors, including already established ones, and $|\mathcal{N}|$ is the size of the set.

Although ProfitPilot only considers new nodes joining the network, it can easily be extended to nodes that already participate in the PCN. In that case, instead of iterating through every node in the network, ProfitPilot attempts to form a cycle in a procedure similar to closing a triangle or a triadic closure [73]. ProfitPilot connects to the node that provides the highest increase in incentive among the neighbors of the user's already established neighbors.

Chapter 5

Development of the Prototype and Experimental Results

We implement ProfitPilot and evaluate it using real data collected from the LN topology. We use Python v3.8 and the NetworkX library for graph analysis[1]. We build the network graph from node and channel announcement messages, used by nodes to announce their channel on the network publicly. The messages were collected using the `c-lightning` implementation and comprise the period from January 2020 to July 2022 [59]. Similar to the literature [8, 46], we snowball sample [74] the complete Lightning Network topology starting from the highest degree node, going from a topology with $8,676$ nodes and $80,220$ edges to a network with 512 nodes and $3,212$ edges. Our analysis focuses on evaluating ProfitPilot in the Lightning Network. Nonetheless, ProfitPilot is PCN-agnostic and can be easily adapted to other PCNs.

Besides LN, we evaluate ProfitPilot using two synthetic topologies: a scale-free Barabasi-Albert graph [75] and a small-world Watts-Stogratz [76] graph, both with 512 nodes. We chose these two topologies because the Lightning Network behaves both as a scale-free and as a small-world topology [51, 61, 65]. We sample node and channel attributes, such as channel capacity, base fee, and proportional fee, from the real Lightning Network and assign them to nodes and edges in the synthetic topology. During this assignment, node correspondence was kept, meaning that the central nodes in the synthetic topology receive their attribute from the central nodes in the Lightning Network snapshot. In particular, the 10% highest degree nodes in the synthetic topology receive attributes from the 10% highest degree nodes in the original LN topology. Unless stated otherwise, we repeat our experiments 10 times to account for statistical variations and error bars represent 95% confidence intervals. The results of ProfitPilot on LN, however, do not present error bars given that the

[1] Available at: `https://github.com/gFrancoCamilo/ln-looprebalance`.

LN topology is fixed and the greedy algorithm is deterministic.

Attachment algorithms. We compare our solution to the following attachment heuristics used by a publicly available LN autopilot [68]:

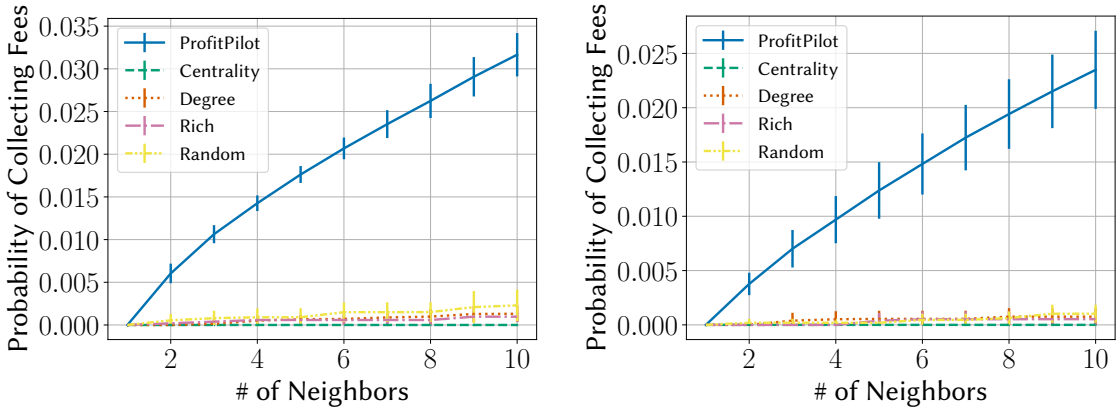
1. *Centrality:* This heuristic selects nodes as potential neighbors from a distribution proportional to the betweenness centrality of nodes.
2. *Rich:* This heuristic selects nodes from a distribution proportional to the capacity of nodes in the network.
3. *Random:* This heuristic follows the Erdos-Renyi [77] model by drawing nodes from a uniform distribution.
4. *Degree:* This heuristic draws nodes from a distribution proportional to the degree of nodes.

The above-listed heuristics were chosen due to their presence in the most popular Lightning Network implementations. The first three heuristics are available in `lib_autopilot`, the default autopilot for the `c-lightning` implementation of the Lightning Network [68].

5.1 Results

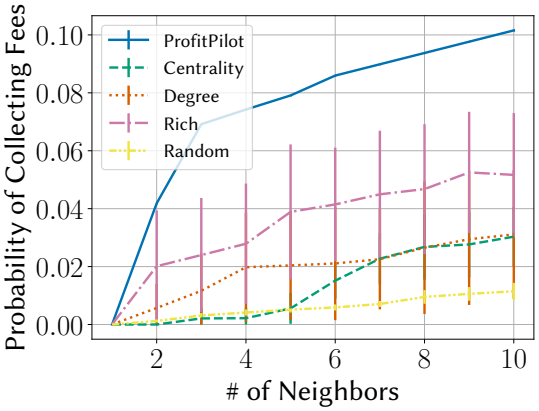
We divide our performance evaluation into two parts. First, we evaluate our proposal from the new node's point of view. We verify if our proposed solution achieves higher rewards than current solutions. Our evaluation compares the approach that creates cycles and the approach that doesn't create cycles to verify if there is a significant loss in incentives. Finally, we evaluate the difference in collected fees from our algorithm to the c-lightning ones. Then, we verify the impact of mass adoption from the network point of view and how it affects the network centralization.

ProfitPilot without cycles. One of ProfitPilot's main benefits is allowing nodes to compensate for channel creation costs by collecting routing fees. We run an experiment that compares ProfitPilot with the currently available autopilot heuristics to quantify this profit boost. The evaluation scenario consists of a node that creates ten channels, each set to charge the network median base and proportional fees, and $\alpha = 0.5$. Figure 5.1 shows the probability of forwarding a payment for each heuristic in the evaluated topology. ProfitPilot increases the likelihood of forwarding a payment and thereby collecting a fee by over 5× in the Barabasi-Albert and Watts-Strogatz topologies, as shown in Figures 5.1a and 5.1b respectively, and by over 2× in the Lightning (Figure 5.1c). Thus, users are compensated for the cost of opening multiple channels using ProfitPilot.



(a) Barabasi-Albert topology.

(b) Watts-Strogatz topology.

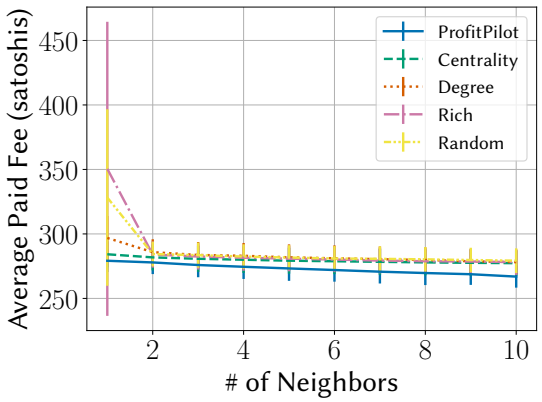


(c) LN topology.

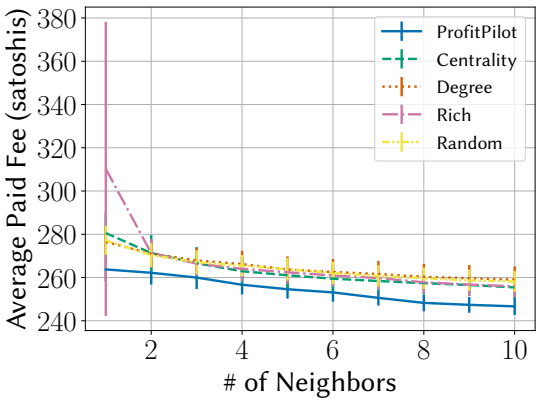
Figure 5.1: Probability of forwarding a payment and, therefore, collecting fees in the evaluated topologies. ProfitPilot more than doubles the probability of collecting fees.

ProfitPilot also creates cost-effective payment routes (Figure 5.2). The average paid fee is computed based on the shortest paths from the ingress to all other nodes in the network and averaging the path fees. We consider a payment value equal to the average transaction of a Ripple dataset [62] to calculate the fees. ProfitPilot presents equal or lower-cost paths compared with the other heuristics for the three topologies evaluated. Moreover, creating more channels decreases the average paying fees, notably for the Watts-Strogatz topology (Figure 5.2b).

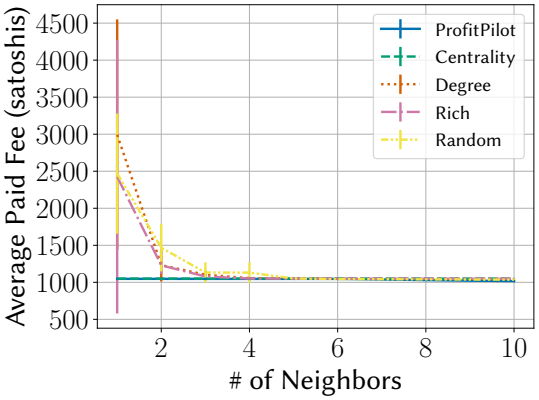
Impact of cycles. We repeat the previous experiments for the cycle creation approach. Figures 5.3 and 5.4 show the results for the average paid fees and the probability of collecting fees, respectively. Even by forcing the creation of triangles and reducing search space, ProfitPilot achieves a higher probability of collecting fees in all three evaluated topologies. In particular, ProfitPilot stands out in the Lightning Network topology, significantly improving fee collection probability. Furthermore, similarly to the approach that ignores cycle creation, ProfitPilot also offers a smaller average paid fee compared to other available heuristics from the first channel cre-



(a) Barabasi-Albert topology.



(b) Watts-Strogatz topology.



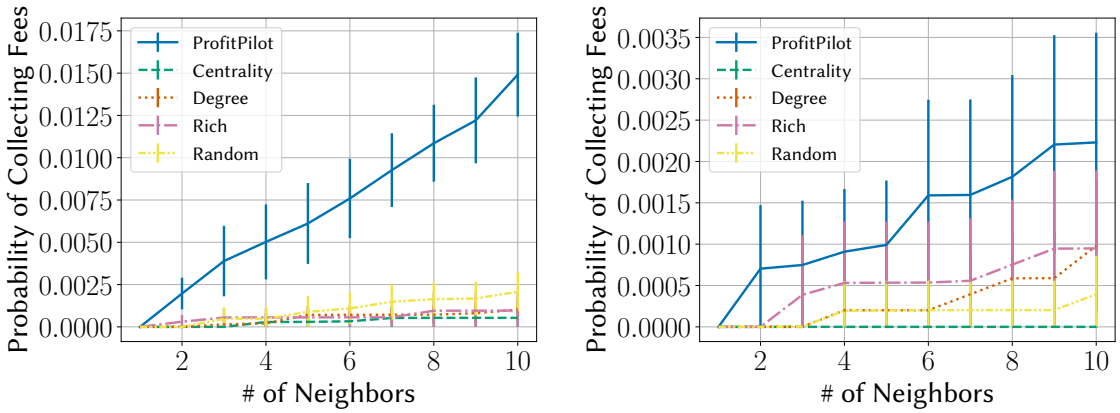
(c) LN topology.

Figure 5.2: Average paid fee for a transaction in the evaluated topologies. ProfitPilot creates cheaper routes when compared with state-of-the-art heuristics in all evaluated topologies.

ated. Thus, users are not required to create a high number of channels to receive the benefits of ProfitPilot.

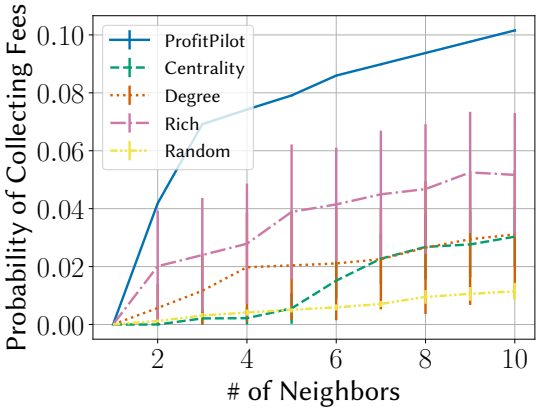
We also verify the number of triangles the node participates in using ProfitPilot or the other evaluated heuristics. Triangles are valuable in PCN topologies to allow off-chain rebalancing and create redundancies. As stated in Chapter 4.4 and demonstrated in Appendix A, ProfitPilot guarantees the creation of at least $|\mathcal{N}|-1$ triangles in all topologies, where $|\mathcal{N}|$ is the number of neighbors. Figure 5.5c shows that the centrality and rich heuristics generate a higher number of triangles in the LN topology. One possible reason for this difference is that central nodes, which are usually high-capacity nodes [51], are highly connected to each other, increasing the probability of creating triangles in the Centrality and Rich heuristics. Nevertheless, these heuristics are unable to ensure the creation of triangles under different topologies, as shown in the results of the Barabasi-Albert and Watts-Strogatz topologies in Figures 5.5a and 5.5b, respectively, unlike ProfitPilot.

Figure 5.6 shows the rebalancing fees paid by the user in each topology after



(a) Barabasi-Albert topology.

(b) Watts-Strogatz topology.

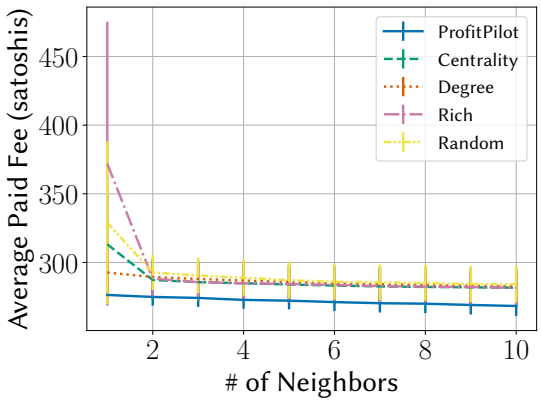


(c) LN topology.

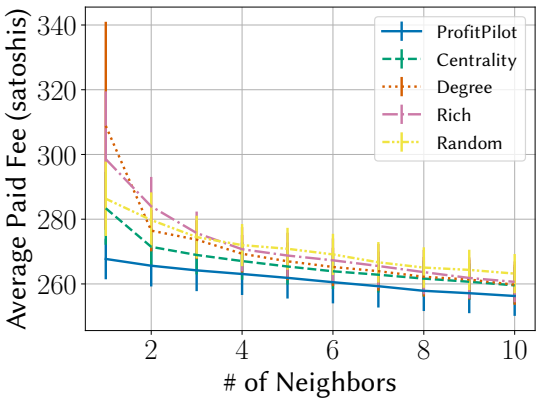
Figure 5.3: Probability of forwarding a payment and, therefore, collecting fees in the evaluated topologies using the strategy that prioritizes the formation of 3-node cycles. Even by forcing the creation of triangles, ProfitPilot presents a higher probability of collecting fees than currently available heuristics.

creating 10 channels. As ProfitPilot focuses on creating 3-node cycles, it may not produce least-cost cycles. Figure 5.6a shows that Centrality produces cheaper rebalancing cycles than ProfitPilot for the LN topology. Still, ProfitPilot presents equally low fees in both Barabasi-Albert and Watts-Strogatz topologies. While the centrality heuristic outperforms ProfitPilot in rebalancing fees in the Lightning Network, ProfitPilot's benefits, such as channel profitability, compensate the user for this difference. Figure 5.6 shows that ProfitPilot compares to the other heuristics regarding rebalancing fees, except for the Lightning Network.

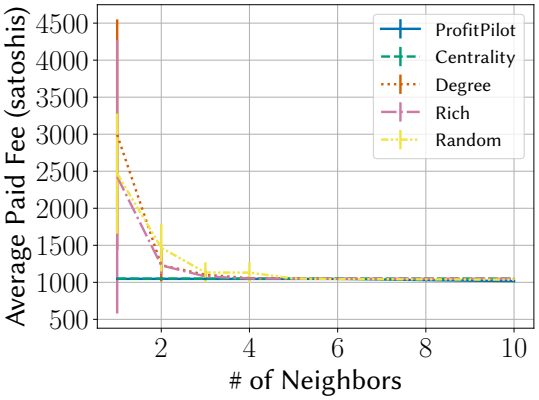
Impact on the network. Finally, we apply ProfitPilot to multiple existing nodes in the network using the procedure described in Chapter 4.4. Basically, ProfitPilot connects to the node that provides the highest incentive among the neighbors of the existing node's neighbor. Our goal is to verify the impact of ProfitPilot on network transitivity and robustness. In particular, we randomly select 15 nodes with a single neighbor and use ProfitPilot to create 3 bidirectional channels for each



(a) Barabasi-Albert topology.



(b) Watts-Strogatz topology.

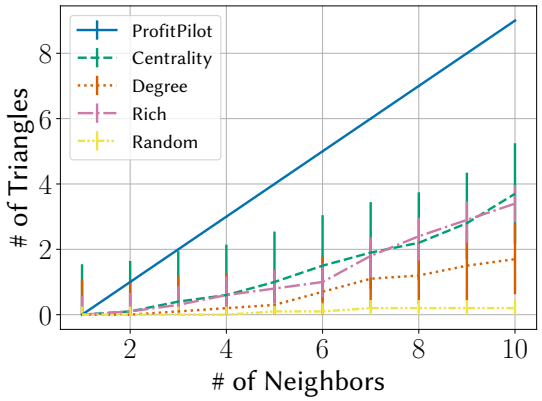


(c) LN topology.

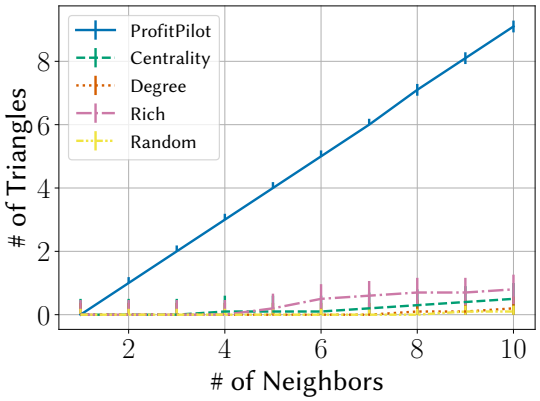
Figure 5.4: Average paid fee for a transaction in the evaluated topologies using ProfitPilot to prioritize the formation of 3-node cycles.

node. Figure 5.7a shows the impact of ProfitPilot and the `lib_autopilot` heuristics on the network transitivity as more nodes adopt these designs. ProfitPilot offers the fastest transitivity increase among all of the strategies. This increase is mostly due to ProfitPilot creating connections in previously intransitive triads in a process similar to closing triangles. While other heuristics also create triangles, they also create more intransitive triads, slowing the increase in network transitivity. We also note that as more nodes adopt ProfitPilot, the quicker the network transitivity reverses its downtrend [51, 61].

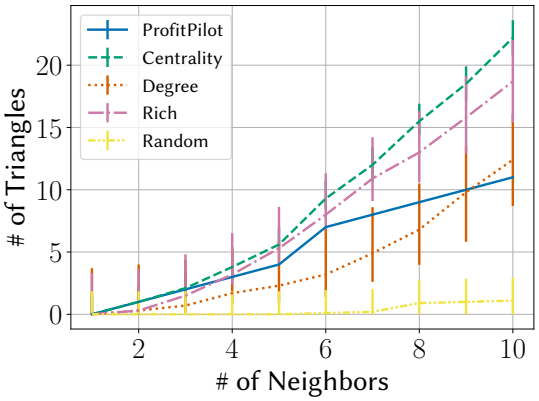
We also examine the robustness of the network against attacks on central nodes. This type of attack is critical in the current Lightning topology as discussed in Chapter 3. Figure 5.7b demonstrates that ProfitPilot efficiently mitigates the effect of targeted attacks by 18% with only 45 nodes adopting it with 3 channels each. We also verify that the higher the number of nodes adopting ProfitPilot, the more robust the network becomes. Although our test is restricted to nodes creating 3 channels, we also expect that an increase in the number of channels results in a more robust network. ProfitPilot's features, i.e. profitability and low-cost routing,



(a) Barabasi-Albert topology.

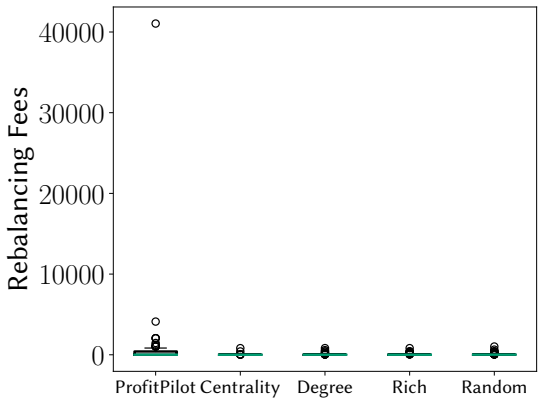


(b) Watts-Strogatz topology.

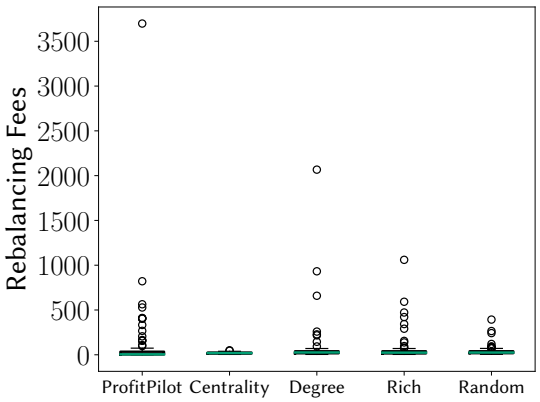


(c) LN topology.

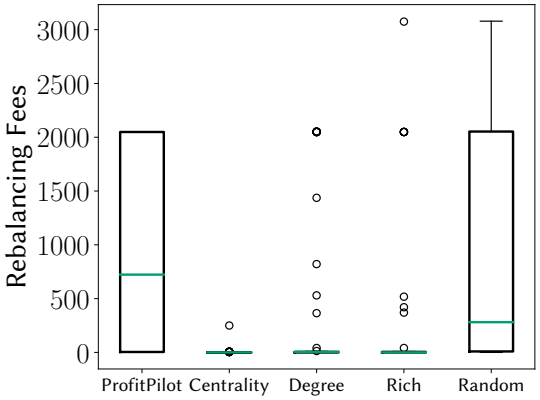
Figure 5.5: Number of triangles created by ProfitPilot and state-of-the-art heuristics in the evaluated topologies.



(a) Barabasi-Albert topology.

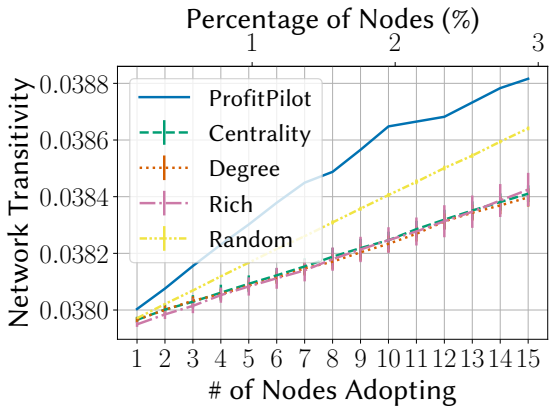


(b) Watts-Strogatz topology.

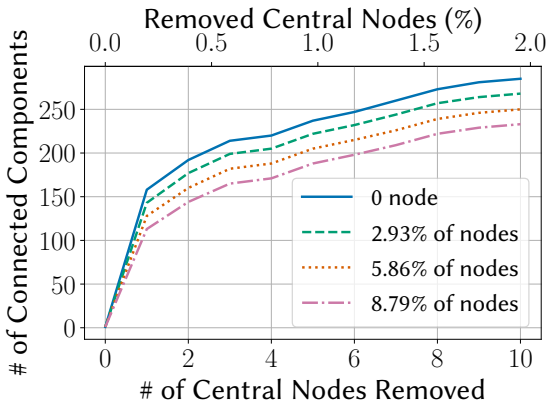


(c) LN topology.

Figure 5.6: Rebalancing fees in satoshis using ProfitPilot and state-of-the-art heuristics in the evaluated topologies.



(a) Improvement in network transitivity as a function of the number of nodes adopting each heuristic in the LN topology.



(b) Robustness against targeted attacks in the LN topology. The curves illustrate the number of nodes adopting ProfitPilot.

Figure 5.7: Impact of our proposal on the Lightning Network topology as more nodes adopt ProfitPilot.

encourage mass adoption, enhancing the network's robustness even further.

Chapter 6

Related Work

6.1 Payment Channel Networks

Most of PCN works in the literature focus on improving the standard routing algorithm [8, 62, 66, 72] or proposing rebalancing strategies [8, 17, 18, 46]. The problem of making rebalancing cheaper and faster is addressed by multiple proposals in the literature [8, 16, 46, 78]. Sivaraman *et al.* propose Spider, a PCN routing algorithm that uses a congestion control protocol to keep channels balanced [8]. Spider defines that payment sources must maintain an adjusted flow window to issue transactions at the same rate they receive, keeping the channel balanced according to demand. In Spider, if the flow of payments in one direction is greater than the flow in the opposite direction, the transaction waits in the queue for another transaction of the same amount in the opposite direction to guarantee equal flows in both directions. This solution, despite improving channel balancing, has some major disadvantages. First, Spider relies on the existence of unpredictable payment demands in the opposite direction to equalize the flow in both directions. If the demands do not exist, the user may have his payment halted indefinitely. Second, Spider does not consider payment fees in its routing algorithm, which can result in high costs for the user. Finally, the implementation of queues and flow control requires structural changes in Lightning Network, which currently does not have these features.

One of the most popular forms of rebalancing in payment channel networks involves issuing self-payments through circular routes. This method allows users to move funds from a low-demand channel to a high-demand channel, reactivating the channel without resorting to blockchains and slow consensus mechanisms. Khalil and Gervais propose REVIVE, a secure payment method on circular routes [16]. In REVIVE, an elected leader receives rebalance requests from multiple users and calculates a set of transactions that must be performed. This set of transactions seeks to meet user requirements and must ensure that users do not lose money in the pro-

cess. The algorithm, however, violates users' privacy, given that to calculate the set of rebalancing transactions, the leader must know users' channel balance. Awathare *et al.* propose REBAL, a circular rebalancing method that uses the transaction flow history of the channel to define the amount of funds that should be moved between channels [46]. In REBAL, participants run the rebalancing protocol locally, which removes the need to share private information with third parties. Otto presents a tool to automate the rebalancing of channels through self-payments in the implementation of the Lightning Network in Go [78] language. The tool allows users to configure the amount of funds he wants in each channel and calculates, based on an optimization problem, the best set of rebalancing transactions locally.

All of these proposals, however, depend on the availability of cycles in the network. As we show in this thesis, these operations are unavailable for a significant number of nodes in the network. Instead, these nodes have to resort to the blockchain, paying high fees and waiting for long periods of time for transaction confirmation. Our approach guarantees the creation of cycles, enabling off-chain rebalancing for nodes in the network.

6.2 Topology Evaluation

Some papers investigate the Lightning Network topology [13–15, 50]. Seres *et al.* analyze the Lightning Network topology using graph metrics and a network snapshot from January 2019 [13]. From the analysis, the authors conclude that the Lightning Network exhibits behavior similar to scale-free networks and small-world networks, and exhibits robustness against random failure attack and vulnerabilities to rational attackers. The authors evaluate the state of the network in January 2019, without considering historical growth. Due to the rapid growth and high dynamicity of the network, the results are limited to the evaluated date and do not consider the changing growth prospects of the network, as demonstrated in the present work. Moreover, Seres *et al.* analysis does not consider some financial parameters, such as the fees paid by users and the direct effect of attacks directed at the end user. Lin *et al.* evaluate the growth of the Lightning Network in relation to the resource concentration of the network [14]. The authors evaluate indices such as Gini, proximity centrality, and betweenness centrality of the network over the period between January 2018 and July 2019. The results show that the network exhibits core-periphery type structures where the core is formed by hubs and star-like substructures, verifying a strong centralization in the network around the highest capacity nodes. The authors expose that removing hubs would lead to partitioning of the network into multiple components, leaving the network more vulnerable to attacks. Similarly, Beres *et al.* analyze Lightning Network metrics, such as number

of edges, average degree, and transitivity, over the period between January 2018 and May 2019 [50] to build a traffic simulator. The papers, however, are limited to the analysis of few centralization metrics, disregarding other metrics, such as evolution of density, components, s-metric and assortativity, and the evolution and distribution of channel characteristics. These metrics allow for a greater understanding of the state of the network and its prospects. Rohrer *et al.* focus on security analysis, quantifying the resilience of the Lightning Network to topology-based attacks [15]. The authors show that the network is susceptible to channel exhaustion and node isolation attacks, which affects the payment success rate and the average payment flow in the network. The result shows that attackers with available resources should follow the strategy of removing higher centrality nodes to perform effective attacks, while nodes with low resource should focus on low capacity channels whose removal partitions the graph.

Unlike the above cited articles, this work evaluates the evolution of centralization of the Lightning Network based on graph-theory metrics and complex networks, as well as evaluating the result of targeted attacks and their effects on end users. The analysis uses real data collected from Lightning Network messages from January 2020 to August 2021 and evaluates the centralization trend of network connectivity and resources, and discusses the consequences and trends of the current growth model of the network for future proposals.

6.3 Attachment Strategies

Several works study different strategies for adding new nodes in PCNs [64, 68, 69]. Pickhardt presents lib_autopilot, a software that automates the creation of channels in the network that allows the user to select heuristics for creating channels, such as random, central, and reduced diameter [68]. Lib_autopilot has been adopted as a plug-in in the c-lightning implementation of the Lightning Network. Lange *et al.* study models of attachment strategies applied to the context of payment channel networks [64]. The authors verify a trade-off between security and efficiency in the evaluated models. While connecting to higher-degree nodes usually results in more efficient payment routing, they also expose the network to several security threats. Ersoy *et al.* focus on making payment channels profitable for users [69]. The authors formalize the maximum reward problem, show that the problem is NP-hard, and propose a channel creation algorithm that returns the maximum reward for a channel. The authors, however, consider a scenario in which the nodes only seek to act as routers, without issuing payments. This consideration prevents the development of a solution close to the real scenario of payment channel networks, in which participants assume roles of both sellers and buyers [6]. In addition, all the

above-cited solutions ignore the issue of rebalancing the channels, which can result in a low lifespan for the channel and high costs in channel operation.

Unlike previous works, this work presents a channel-opening strategy that prioritizes the increase of financial gains and decreases possible costs to users by creating channels. Furthermore, this thesis develops an algorithm that allows users to establish cycles in the network topology, enabling rebalancing operations and extending the channel lifespan. The proposed model, which prioritizes the user's financial gains, compensates for the cost of opening multiple channels to establish cycles. As far as we know, our approach is the first to propose creating cycles to enable rebalancing and improve network robustness.

Chapter 7

Conclusion and Future Work

Payment channel networks present a fundamental mechanism for enabling fast, secure, and decentralized cryptocurrency payments in everyday life. Because of the way the routing of payments through payment channels is implemented, PCNs are heavily influenced by their topology. This thesis analyzes the evolution of the topology of the Lightning Network, the main payment channel network, using graph theory metrics and real messages collected from the network. The results show that the connections and capacity allocated to the network are extremely concentrated in a few nodes, which are mostly controlled by companies linked to blockchain development. This centralization is a natural consequence of how the entry of new participants takes place, seeking smaller paths to perform transactions and, consequently, pay lower fees. In addition, the thesis finds that the results indicate an evolution towards a weakly transitive network, which makes channel rebalancing techniques unfeasible for most nodes.

This thesis presented ProfitPilot, a strategy for connecting new nodes in payment channel networks. Our proposed model presents financial advantages to users, such as compensating the costs of channel creation by establishing profitable and low-cost routes. Furthermore, the proposed scheme is also beneficial to the network as it promotes redundancies and makes it more robust to some security threats. We expect that these features promote user adoption. We implement and test ProfitPilot on real-world Lightning Network topology and the results show that the presented solution rewards the user up to 3× more than the traditional methods of adding nodes in the network. Although we focus on applying ProfitPilot to our PCN scenario, ProfitPilot can also be used by any other network application with similar routing characteristics as PCNs. ProfitPilot also presents cost-efficient routes with as low as one channel being created by the user when compared to state-of-the-art heuristics. ProfitPilot also benefits network security, as it quickly increases network transitivity and reduces the effects of targeted topological attacks.

Future work should offer fee-setting strategies, which allow the entry node to

maintain its profitable position after joining the network.

References

[1] "Visa acceptance for retailers". Available at `https://usa.visa.com/run-your-business/small-business-tools/retail.html`. Access: April 23rd, 2023.

[2] "Bitcoin Scalability". Available at `https://en.bitcoin.it/wiki/Scalability`. Access: January 13th, 2024.

[3] CROMAN, K., DECKER, C., EYAL, I., et al. "On Scaling Decentralized Blockchains". In: *Financial Cryptography and Data Security*, pp. 106–125, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg.

[4] WANG, J., WANG, H. "Monoxide: Scale out Blockchains with Asynchronous Consensus Zones". In: *16th USENIX Symposium on Networked Systems Design and Implementation (NSDI 19)*, pp. 95–112, Boston, MA, fev. 2019. USENIX Association.

[5] GILAD, Y., HEMO, R., MICALI, S., et al. "Algorand: Scaling Byzantine Agreements for Cryptocurrencies". In: *Proceedings of the 26th Symposium on Operating Systems Principles*, SOSP '17, p. 51–68, New York, NY, USA, 2017. Association for Computing Machinery.

[6] POON, J., DRYJA, T. "The bitcoin lightning network: Scalable off-chain instant payments". 2016. Available at `https://lightning.network/lightning-network-paper.pdf`. Access: January 13th, 2024.

[7] DZIEMBOWSKI, S., FAUST, S., HOSTÁKOVÁ, K. "General State Channel Networks". In: *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, CCS '18, p. 949–966, New York, NY, USA, 2018. Association for Computing Machinery.

[8] SIVARAMAN, V., VENKATAKRISHNAN, S. B., RUAN, K., et al. "High throughput cryptocurrency routing in payment channel networks". In: *Proceedings of the 17th Usenix Conference on Networked Systems Design and Implementation*, NSDI'20, pp. 777–796, USA, fev. 2020. USENIX Association.

[9] "Raiden Network". Available at `https://raiden.network/`. Access: January 13th, 2024.

[10] CLOUDTWEAKS. "How Bitcoin Brought The Lightning Network To El Salvador". 2021. Available at `https://cloudtweaks.com/2021/07/how-bitcoin-brought-lightning-network-el-salvador/`. Access: January 13th, 2024.

[11] PICKHARDT, R., NOWOSTAWSKI, M. "Imbalance measure and proactive channel rebalancing algorithm for the Lightning Network". In: *2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, pp. 1–5. IEEE, 2020.

[12] KHALIL, R., GERVAIS, A. "Revive: Rebalancing Off-Blockchain Payment Networks". In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, p. 439–453, New York, NY, USA, 2017. Association for Computing Machinery.

[13] SERES, I. A., GULYÁS, L., NAGY, D. A., et al. "Topological Analysis of Bitcoin's Lightning Network", *arXiv preprint arXiv:1901.04972*, 2019.

[14] LIN, J.-H., PRIMICERIO, K., SQUARTINI, T., et al. "Lightning Network: a second path towards centralisation of the Bitcoin economy", *New Journal of Physics*, v. 22, n. 8, pp. 083022, 2020.

[15] ROHRER, E., MALLIARIS, J., TSCHORSCH, F. "Discharged payment channels: Quantifying the lightning network's resilience to topology-based attacks". In: *IEEE EEuroS&PW*, 2019.

[16] KHALIL, R., GERVAIS, A. "Revive: Rebalancing Off-Blockchain Payment Networks". In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, pp. 439–453, New York, NY, USA, out. 2017. ACM.

[17] AVARIKIOTI, Z., PIETRZAK, K., SALEM, I., et al. "Hide&Seek: Privacy-Preserving Rebalancing On Payment Channel Networks". In: *Financial Cryptography and Data Security: 26th International Conference, FC 2022, Grenada, May 2–6, 2022, Revised Selected Papers*, p. 358–373, Berlin, Heidelberg, 2022. Springer-Verlag. ISBN: 978-3-031-18282-2.

[18] HONG, Z., GUO, S., ZHANG, R., et al. "Cycle: Sustainable Off-Chain Payment Channel Network with Asynchronous Rebalancing". In: *2022 52nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pp. 41–53, 2022. doi: 10.1109/DSN53405.2022.00017.

[19] NAKAMOTO, S. "Bitcoin: A peer-to-peer electronic cash system". 2008. Available at `https://bitcoin.org/bitcoin.pdf`. Access: January 13th, 2024.

[20] REBELLO, G. A. F., CAMILO, G. F., GUIMARAES, L. C., et al. "A security and performance analysis of proof-based consensus protocols", *Annals of Telecommunications*, pp. 1–21, 2021.

[21] LAMPORT, L. "Paxos made simple", *ACM SIGACT News (Distributed Computing Column) 32, 4 (Whole Number 121, December 2001)*, pp. 51–58, 2001.

[22] CASTRO, M., LISKOV, B. "Practical Byzantine Fault Tolerance". In: *3rd Symposium on Operating Systems Design and Implementation (OSDI 99)*, New Orleans, LA, fev. 1999. USENIX Association.

[23] ATTIYA, H., BAR-NOY, A., DOLEV, D. "Sharing memory robustly in message-passing systems", *Journal of the ACM (JACM)*, v. 42, n. 1, pp. 124–142, 1995.

[24] BITINFOCHARTS. "Bitcoin Average Transaction Fee Chart". 2022. Available at `https://bitinfocharts.com/comparison/bitcoin-transactionfees.html`. Access: January 13th, 2024.

[25] BUTERIN, V., OTHERS. "A next-generation smart contract and decentralized application platform", 2014. Available at `https://blockchainlab.com/pdf/Ethereum_white_paper-a_next_generation_smart_contract_and_decentralized_application_platform-vitalik-buterin.pdf`. Access: January 13th, 2024.

[26] BUTERIN, V. "Why sharding is great: demystifying the technical properties". 2021. Available at `https://vitalik.ca/general/2021/04/07/sharding.html`. Access: November 2nd, 2023.

[27] YIN, M., MALKHI, D., REITER, M. K., et al. "HotStuff: BFT Consensus with Linearity and Responsiveness". In: *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing*, PODC '19, p. 347–356, New York, NY, USA, 2019. Association for Computing Machinery.

[28] MILLER, A., XIA, Y., CROMAN, K., et al. "The Honey Badger of BFT Protocols". In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, p. 31–42, New York, NY, USA, 2016. Association for Computing Machinery.

[29] GUO, B., LU, Z., TANG, Q., et al. "Dumbo: Faster Asynchronous BFT Protocols". In: *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, CCS '20, p. 803–818, New York, NY, USA, 2020. Association for Computing Machinery.

[30] CARRARA, G. R., MATTOS, D. M. F., DE ALBUQUERQUE, C. V. N. "Vicinity-based Consensus: A Fast in-Neighborhood Convergence Consensus Mechanism for Blockchain". In: *2021 IEEE Global Communications Conference (GLOBECOM)*, pp. 01–06, 2021. doi: 10.1109/GLOBECOM46510.2021.9685110.

[31] POPOV, S. "The tangle", 2018. Available at `https://assets.ctfassets.net/r1dr6vzfxhev/2t4uxvsIqk0EUau6g2sw0g/45eae33637ca92f85dd9f4a3a218e1ec/iota1_4_3.pdf`. Access: January 13th, 2024.

[32] BAIRD, L. "The swirlds hashgraph consensus algorithm: Fair, fast, byzantine fault tolerance", *Swirlds Tech Reports SWIRLDS-TR-2016-01, Tech. Rep*, v. 34, pp. 9–11, 2016.

[33] AMORES-SESAR, I., CACHIN, C., TEDESCHI, E. "When Is Spring Coming? A Security Analysis of Avalanche Consensus". In: *26th International Conference on Principles of Distributed Systems (OPODIS 2022)*, v. 253, *Leibniz International Proceedings in Informatics (LIPIcs)*, pp. 10:1–10:22, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. ISBN: 978-3-95977-265-5. doi: 10.4230/LIPIcs.OPODIS.2022.10.

[34] MICHELIN, R. A., DORRI, A., STEGER, M., et al. "SpeedyChain: A Framework for Decoupling Data from Blockchain for Smart Cities". In: *Proceedings of the 15th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, MobiQuitous '18, p. 145–154, New York, NY, USA, 2018. Association for Computing Machinery.

[35] LUNARDI, R. C., MICHELIN, R. A., NEU, C. V., et al. "Impact of Consensus on Appendable-Block Blockchain for IoT". In: *Proceedings of the 16th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, MobiQuitous '19, p. 228–237, New York, NY, USA, 2020. Association for Computing Machinery.

[36] NUNES, H. C., LUNARDI, R. C., ZORZO, A. F., et al. "Context-based Smart Contracts For Appendable-block Blockchains". In: *2020 IEEE Interna-*

tional Conference on Blockchain and Cryptocurrency (ICBC), pp. 1–9, 2020. doi: 10.1109/ICBC48266.2020.9169466.

[37] DANG, H., DINH, T. T. A., LOGHIN, D., et al. "Towards Scaling Blockchain Systems via Sharding". In: *Proceedings of the 2019 International Conference on Management of Data*, SIGMOD '19, p. 123–140, New York, NY, USA, 2019. Association for Computing Machinery.

[38] ZAMANI, M., MOVAHEDI, M., RAYKOVA, M. "RapidChain: Scaling Blockchain via Full Sharding". In: *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, CCS '18, p. 931–948, New York, NY, USA, 2018. Association for Computing Machinery.

[39] HONG, Z., GUO, S., LI, P., et al. "Pyramid: A Layered Sharding Blockchain System". In: *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications*, pp. 1–10, 2021. doi: 10.1109/INFOCOM42981.2021.9488747.

[40] HUANG, C., WANG, Z., CHEN, H., et al. "RepChain: A Reputation-Based Secure, Fast, and High Incentive Blockchain System via Sharding", *IEEE Internet of Things Journal*, v. 8, n. 6, pp. 4291–4304, 2021. doi: 10.1109/JIOT.2020.3028449.

[41] HEARN, M. "Micro-payment channels implementation now in bitcoinj". 2013. Available at `https://bitcointalk.org/index.php?topic=244656.0`. Access: January 13th, 2024.

[42] SPILMAN, J. "[Bitcoin-development] Anti DoS for tx replacement". 2013. Available at `https://lists.linuxfoundation.org/pipermail/bitcoin-dev/2013-April/002433.html`. Access: January 13th, 2024.

[43] BITCOINJ.ORG. "bitcoinj". Available at `https://bitcoinj.org`. Access: January 13th, 2024.

[44] MALAVOLTA, G., MORENO-SANCHEZ, P., KATE, A., et al. "Concurrency and Privacy with Payment-Channel Networks". In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, p. 455–471, New York, NY, USA, 2017. Association for Computing Machinery.

[45] DECKER, C. "Splicing. [Lightning-dev] Channel top-up". Available at `https://lists.linuxfoundation.org/pipermail/lightning-dev/2017-May/000696.html`, 2017. Access: January 13th, 2024.

[46] AWATHARE, N., SURAJ, AKASH, et al. "REBAL: Channel Balancing for Payment Channel Networks". In: *2021 29th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*, pp. 1–8, nov. 2021.

[47] PICKHARDT, R., NOWOSTAWSKI, M. *Imbalance measure and proactive channel rebalancing algorithm for the Lightning Network*. Relatório Técnico arXiv:1912.09555, arXiv, dez. 2019. Disponível em: <`http://arxiv.org/abs/1912.09555`>. arXiv:1912.09555 [cs] type: article.

[48] "1ML - Lightning Network Search and Analysis Engine". Available at `https://1ml.com/`. Access: January 13th, 2024.

[49] AVARIKIOTI, Z., HEIMBACH, L., WANG, Y., et al. "Ride the Lightning: The Game Theory of Payment Channels". In: Bonneau, J., Heninger, N. (Eds.), *Financial Cryptography and Data Security*, pp. 264–283, Cham, 2020. Springer International Publishing.

[50] BERES, F., SERES, I. A., BENCZUR, A. A. "A Cryptoeconomic Traffic Analysis of Bitcoin's Lightning Network". 2019. Disponível em: <`https://arxiv.org/abs/1911.09432`>.

[51] SERES, I. A., GULYÁS, L., NAGY, D. A., et al. "Topological Analysis of Bitcoin's Lightning Network". In: Pardalos, P., Kotsireas, I., Guo, Y., et al. (Eds.), *Mathematical Research for Blockchain Economy*, pp. 1–12, Cham, 2020. ISBN: 978-3-030-37110-4.

[52] "Basis of Lightning Technology (BOLTs)". Available at `https://github.com/lightning/bolts`. Access: January 13th, 2024.

[53] LIGHTNING, C. "Core Lightning (CLN): A specification compliant Lightning Network implementation in C". Available at `https://github.com/ElementsProject/lightning`. Access: January 13th, 2024.

[54] LABS, L. "Lightning Network Daemon". Available at `https://github.com/lightningnetwork/lnd`. Access: January 13th, 2024.

[55] ACINQ. "eclair". Available at `https://github.com/ACINQ/eclair`. Access: January 13th, 2024.

[56] POON, J., OSUNTOKUN, O. "BOLT #2: Peer Protocol for Channel Management". `https://github.com/lightningnetwork/lightning-rfc/blob/master/02-peer-protocol.md`, 2021. Access: January 13th, 2024.

[57] ARMKNECHT, F., KARAME, G. O., MANDAL, A., et al. "Ripple: Overview and Outlook". In: *Trust and Trustworthy Computing*, pp. 163–180. Springer International Publishing, 2015. ISBN: 978-3-319-22846-4.

[58] ROOS, S., MORENO-SANCHEZ, P., KATE, A., et al. "Settling payments fast and private: Efficient decentralized routing for path-based transactions", *arXiv preprint arXiv:1709.05748*, 2017.

[59] DECKER, C. "Lightning Network Research; Topology Datasets". `https://github.com/lnresearch/topology`, 2021. Access: January 13th, 2024.

[60] BLOCKCHAIN.COM. "Unconfirmed transactions". `https://www.blockchain.com/btc/unconfirmed-transactions`, 2022. Access: August 5th, 2022.

[61] CAMILO, G. F., REBELLO, G. A. F., DE SOUZA, L. A. C., et al. "Topological Evolution Analysis of Payment Channels in the Lightning Network". In: *2022 IEEE Latin-American Conference on Communications (LATINCOM)*, pp. 1–6, 2022. doi: 10.1109/LATINCOM56090.2022.10000445.

[62] ROOS, S., MORENO-SANCHEZ, P., KATE, A., et al. "Settling Payments Fast and Private: Efficient Decentralized Routing for Path-Based Transactions". In: *Proceedings 2018 Network and Distributed System Security Symposium*, San Diego, CA, 2018. Internet Society.

[63] SIVARAMAN, V., OTHERS. "High throughput cryptocurrency routing in payment channel networks". In: *17th USENIX Symposium on Networked Systems Design and Implementation (NSDI 20)*, pp. 777–796, 2020.

[64] LANGE, K., ROHRER, E., TSCHORSCH, F. "On the Impact of Attachment Strategies for Payment Channel Networks". In: *2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, pp. 1–9, 2021. doi: 10.1109/ICBC51069.2021.9461104.

[65] ROHRER, E., MALLIARIS, J., TSCHORSCH, F. "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)*, pp. 347–356, 2019. doi: 10.1109/EuroSPW.2019.00045.

[66] WANG, P., XU, H., JIN, X., et al. "Flash: Efficient Dynamic Routing for Offchain Networks". In: *Proceedings of the 15th International Conference on Emerging Networking Experiments And Technologies*, CoNEXT '19, p.

370–381, New York, NY, USA, 2019. Association for Computing Machinery.

[67] "Lightning Network DDoS Sends 20% of Nodes Down". Available at `https://www.trustnodes.com/2018/03/21/lightning-network-ddos-sends-20-nodes`. Access: January 13th, 2024.

[68] PICKHARDT, R. "lightning-network-autopilot". Available at `https://github.com/renepickhardt/lightning-network-autopilot`, 2019. Access: January 13th, 2024.

[69] ERSOY, O., ROOS, S., ERKIN, Z. "How to Profit from Payments Channels". In: Bonneau, J., Heninger, N. (Eds.), *Financial Cryptography and Data Security*, pp. 284–303, Cham, 2020. Springer International Publishing. ISBN: 978-3-030-51280-4.

[70] AVARIKIOTI, G., WANG, Y., WATTENHOFER, R. "Algorithmic Channel Design", p. 12 pages, 2018. Artwork Size: 12 pages Medium: application/pdf Publisher: Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik GmbH, Wadern/Saarbruecken, Germany.

[71] ZHANG, Y., YANG, D. "RobustPay+: Robust Payment Routing With Approximation Guarantee in Blockchain-Based Payment Channel Networks", *IEEE/ACM Transactions on Networking*, v. 29, n. 4, pp. 1676–1686, 2021. doi: 10.1109/TNET.2021.3069725.

[72] CHEN, W., QIU, X., HONG, Z., et al. "Proactive Look-Ahead Control of Transaction Flows for High-Throughput Payment Channel Network". In: *Proceedings of the 13th Symposium on Cloud Computing*, SoCC '22, p. 429–444, New York, NY, USA, 2022. Association for Computing Machinery.

[73] HUANG, H., TANG, J., LIU, L., et al. "Triadic closure pattern analysis and prediction in social networks", *IEEE Transactions on Knowledge and Data Engineering*, v. 27, n. 12, pp. 3374–3389, 2015.

[74] HU, P., LAU, W. C. "A survey and taxonomy of graph sampling", *arXiv preprint arXiv:1308.5865*, 2013.

[75] BARABÁSI, A.-L., ALBERT, R. "Emergence of Scaling in Random Networks", *Science*, v. 286, n. 5439, pp. 509–512, 1999.

[76] WATTS, D. J., STROGATZ, S. H. "Collective dynamics of 'small-world' networks", *Nature*, v. 393, n. 6684, pp. 440–442, jun. 1998. ISSN: 1476-4687. Number: 6684 Publisher: Nature Publishing Group.

[77] ERDŐS, P., RÉNYI, A., OTHERS. "On the evolution of random graphs", *Publ. Math. Inst. Hung. Acad. Sci*, v. 5, n. 1, pp. 17–60, 1960.

[78] OTTO, C. "Rebalance-LND". 2022. Available at `https://github.com/C-Otto/rebalance-lnd`. Access: January 13th, 2024.

Appendix A

Creating Triangles with ProfitPilot

As mentioned in Section 4.4, ProfitPilot guarantees the creation of $|\mathcal{N}|-1$ triangles. We refrain from a more formal proof that would require showing a loop invariant proof of the inner loop in Algorithm 1 to demonstrate that it only creates channels with already selected nodes' neighbors. Instead, we intuitively state that the inner loop traverses the nodes of the network and only selects nodes that are neighbors of already selected ones. This can be logically observed by the '*if*' condition inside the inner loop that checks if the analyzed node is a neighbor of a node before storing it as a potential node as a suggestion.

Theorem A.0.1 (ProfitPilot's triangles.)**.** *At each interaction of Algorithm 1, the ingress node n participates in at least $|\mathcal{N}|-1$ triangles, for $|\mathcal{N}| \geq 1$.*

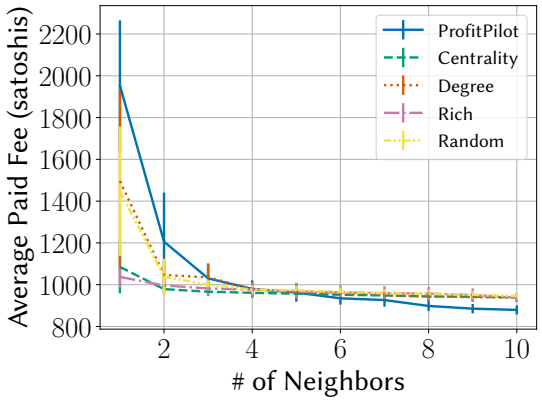
Proof. We prove Theorem A.0.1 using the loop invariant method.
Initialization: We first show that the loop invariant holds for $|\mathcal{N}| = 1$. In this scenario, the list of suggested node connections $\mathcal{N}$ is composed of a single element, meaning that n has only one channel. Thus, the loop invariant holds, given that one channel is not enough to form a triangle, i.e., n participates in $|\mathcal{N}|-1 = 0$ triangles.
Maintenance: Next, we show that each iteration maintains the invariant. First, assume that the loop invariant holds for an iteration j, that is $|\mathcal{N}| = j$ and node n participates in $j-1$ triangles. The inner loop only selects nodes that share a channel with a node in $\mathcal{N}$. Thus, if a node is selected by the algorithm, it will be added to $\mathcal{N}$ and $|\mathcal{N}| = j+1$. The algorithm will, then, establish a channel with this selected node and, as it shares a channel with a node already in $\mathcal{N}$, the node will participate in at least one more triangle, i.e., at least j triangles. If the algorithm traverses the network without selecting a node, $|\mathcal{N}|$ will remain unchanged, maintaining the number of triangles at the beginning of the interaction. Hence, the loop invariant holds at the end of the iteration.
Termination: Finally, we analyze the loop termination. There are two possible ways that the algorithm terminates. First, if the maximum number of channels

that the user wants to create k has been achieved, meaning $|\mathcal{N}| = k$. In that case, node n participates in at least $k-1$ triangles. The second possibility of terminating the algorithm occurs if ProfitPilot fails to find a higher incentive than the current setting. At that point, the algorithm breaks out of the loop in the last **if** condition presented at Algorithm 1 and the node participates in at least $|\mathcal{N}|-1$ triangles. □

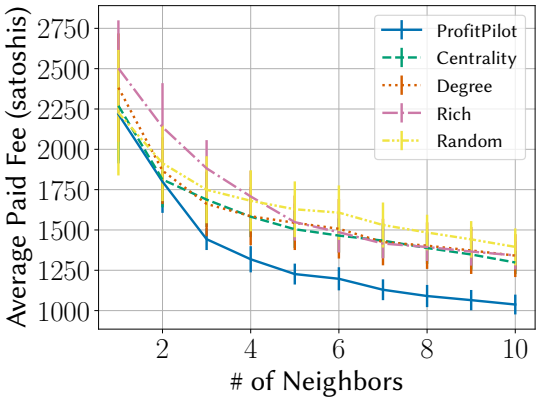
Appendix B

Additional Results

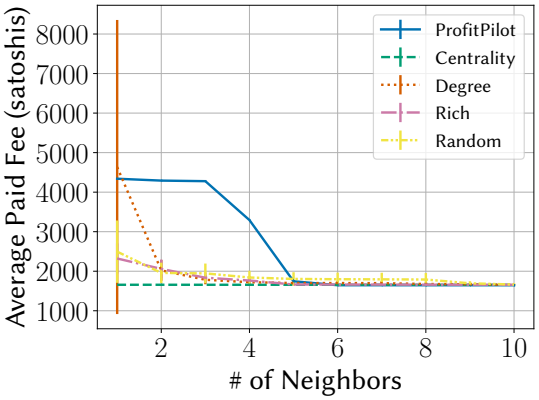
We verify the node's financial incentive with a varying α parameter. Figures B.1 and B.2 show the incentives nodes receive with parameter $\alpha = 1.0$. In this scenario, the node is not interested in paying low fees or issuing payments. Instead, the new node is only interested in creating channels that will result in financial gain. This strategy may be used by service providers or router nodes.



(a) Barabasi-Albert topology.

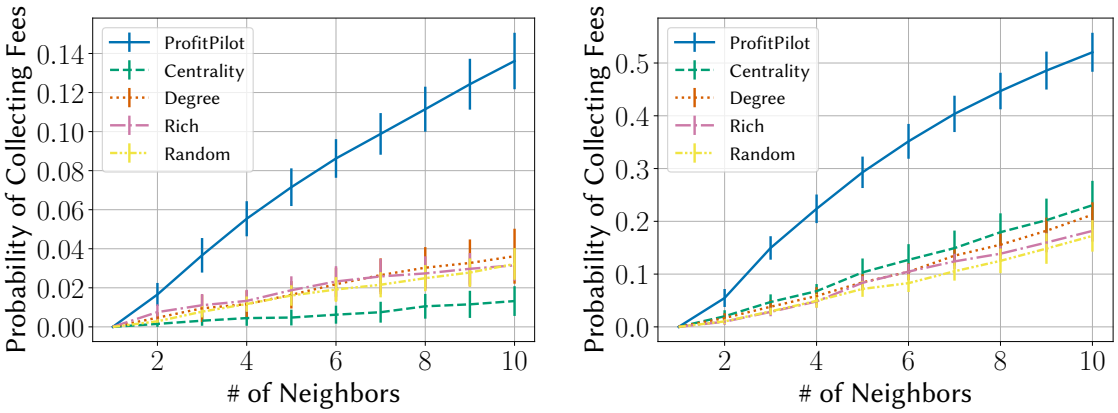


(b) Watts-Strogatz topology.



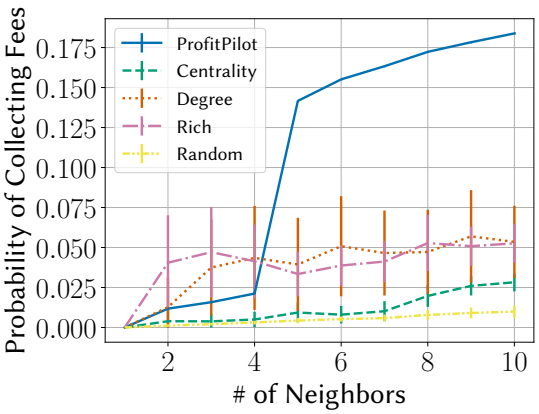
(c) LN topology.

Figure B.1: Average paid fee for a transaction in the Barabasi-Albert, Watts-Strogatz, and the Lightning Network topology considering $\alpha = 1.0$.



(a) Barabasi-Albert topology.

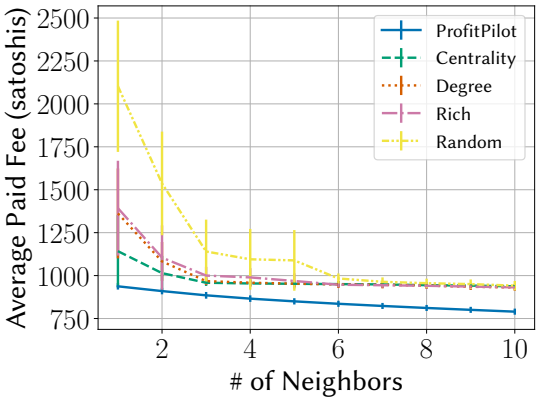
(b) Watts-Strogatz topology.



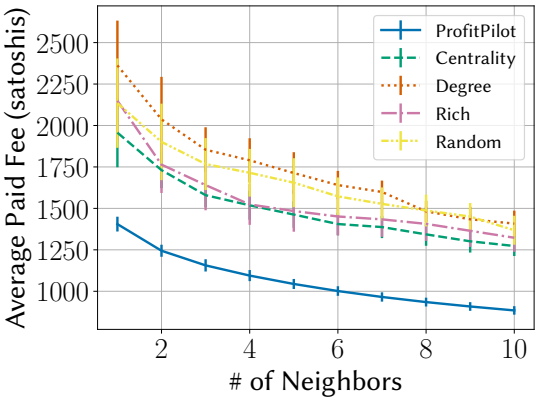
(c) LN topology.

Figure B.2: Probability of forwarding a payment and, therefore, collecting fees in the Barabasi-Albert, Watts-Strogatz, and the Lightning Network topology considering $\alpha = 1.0$.

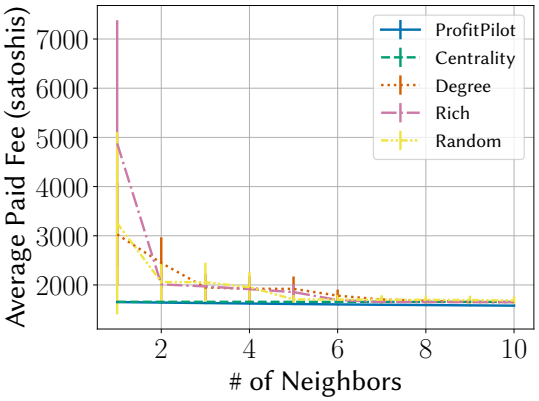
Figures B.3 and B.4 show the incentives nodes receive with parameter $\alpha = 0.0$. In this scenario, the node is not interested in creating channels that will result in financial gain. Instead, the new node is only interested in paying low fees or in issuing payments. This strategy may be used by buyers.



(a) Barabasi-Albert topology.

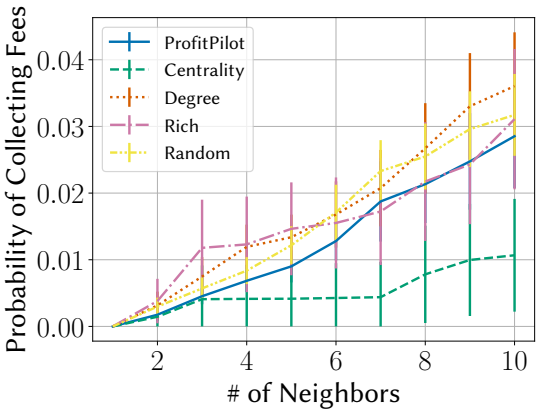


(b) Watts-Strogatz topology.

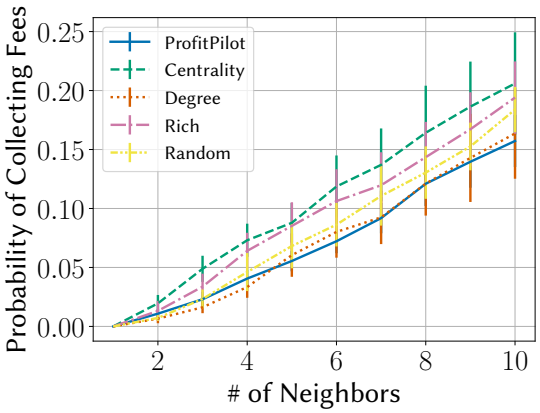


(c) LN topology.

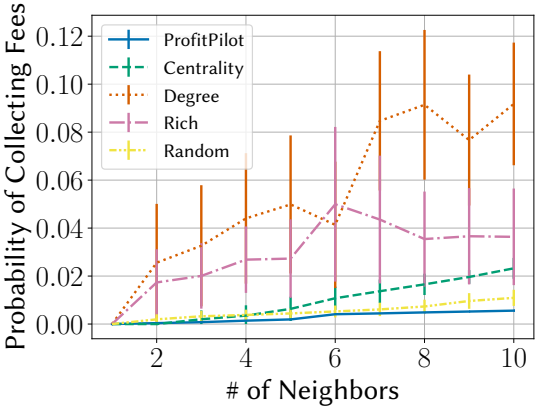
Figure B.3: Average paid fee for a transaction in the Barabasi-Albert, Watts-Strogatz, and the Lightning Network topology considering $\alpha = 0.0$.



(a) Barabasi-Albert topology.



(b) Watts-Strogatz topology.



(c) LN topology.

Figure B.4: Probability of forwarding a payment and, therefore, collecting fees in the Barabasi-Albert, Watts-Strogatz, and the Lightning Network topology considering $\alpha = 0.0$.

Appendix C

List of Publications

The following works were published during the elaboration of this master thesis:

- de Souza, L. A. C., **Camilo, G. F.**, Rebello, G. A. F., Sammarco, M., Campista, M. E. M., Costa, L. H. M. K. - "ATHENA-FL: Evitando a Heterogeneidade Estatística através do Um-contra-Todos no Aprendizado Federado". In Anais do VII Workshop de Computação Urbana (pp. 40-53). (2023, May). SBC. **Honorable Mention**.

- de Souza, L. A. C., Rebello, G. A. F., **Camilo, G. F.**, Campista, M. E. M., Costa, L. H. M. K. - "GITI-CB: Gestao de Identidade com Troca de Informaçoes entre Correntes de Blocos". In Anais do VI Workshop em Blockchain: Teoria, Tecnologias e Aplicações (pp. 43-56). (2023, May). SBC.

- **Camilo, G. F.**, Rebello, G. A. F., de Souza, L. A. C., Campista, M. E. M., Costa, L. H. M. K. - "Posicionamento Lucrativo de Nós e Criaçao de Rotas de Baixo Custo na Rede Relâmpago". In Anais do XLI Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (pp. 57-70). (2023, May). SBC. **Honorable Mention**.

- de Souza, L. A. C., **Camilo, G. F.**, Campista, M. E. M., Costa, L. H. M., Duarte, O. C. M. - "Enhancing Automatic Attack Detection through Spectral Decomposition of Network Flows". In Global Communications Conference (GLOBECOM) (pp. 2074-2079). (2022, December). IEEE.

- **Camilo, G. F.**, Rebello, G. A. F., de Souza, L. A. C., Potop-Butucaru, M., Amorim, M. D., Campista, M. E. M., Costa, L. H. M. K. - "Topological Evolution Analysis of Payment Channels in the Lightning Network". In Latin-American Conference on Communications (LATINCOM) (pp. 1-6). (2022, November). IEEE.

- Rebello, G. A. F., **Camilo, G. F.**, Guimarães, L. C., de Souza, L. A. C., Duarte, O. C. M. - "Security and Performance Analysis of Quorum-based Blockchain Consensus Protocols". In 6th Cyber Security in Networking Conference (CSNet) (pp. 1-7). (2022, October). IEEE.

- de Souza, L. A. C., **Camilo, G. F.**, Rebello, G. A. F., Campista, M. E. M., Costa, L. H. M. K. - "Gestão Segura e Escalável de Identidades através de Múltiplas Corrente de Blocos". In Anais Estendidos do XXII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (pp. 167-170). (2022, September). SBC.

- **Camilo, G. F.**, Rebello, G. A. F., de Souza, L. A. C., Potop-Butucaru, M., Amorim, M. D., Campista, M. E. M., Costa, L. H. M. K. - "Análise da Evolução Topológica da Rede Lightning de Canais de Pagamento". In Anais do XXII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (pp. 71-84). (2022, September). SBC.

- de Souza, L. A. C., **Camilo, G. F.**, Sammarco, M., Campista, M. E. M., Costa, L. H. M. - "Aprendizado Federado com Agrupamento Hierárquico de Clientes para Aumento da Acurácia". In Anais do XL Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (pp. 545-558). (May 2022). SBC.

- **Camilo, G. F.**, de Souza, L. A. C., Campista, M. E. M., Costa, L. H. M., Duarte, O. C. M. - "A Blockchain-based System for Secure and Distributed Virtual Network Functions Orchestration". In International Conference on Communications (ICC) (pp. 347-352). (2022, May). IEEE.

- Thomaz, G. A., **Camilo, G. F.**, de Souza, L. A. C., Duarte, O. C. M. - "Architecture and Performance Comparison of Permissioned Blockchains Platforms for Smart Contracts". In Global Communications Conference (GLOBECOM) (pp. 1-6). (2021, December). IEEE.

- Alvarenga, I. D., **Camilo, G. F.**, de Souza, L. A. C., Duarte, O. C. M. - "DAGSec: A Hybrid Distributed Ledger Architecture for the Secure Management of the Internet of Things". In International Conference on Blockchain (Blockchain) (pp. 266-271). (2021, December). IEEE.

- Rebello, G. A. F., **Camilo, G. F.**, Guimaraes, L. C., de Souza, L. A. C., Thomaz, G. A., Duarte, O. C. M. - "A Security and Performance Analysis of Proof-based Consensus Protocols". Annals of Telecommunications. (2021). 1-21.