Capítulo 1

Redes de Canais de Pagamento: Provendo Escalabilidade para Pagamentos em Criptomoedas

Gustavo F. Camilo, Gabriel Antonio F. Rebello, Lucas Airam C. de Souza, Guilherme A. Thomaz, Maria Potop-Butucaru, Marcelo Dias Amorim, Miguel Elias M. Campista, Luís Henrique M. K. Costa

Abstract

Blockchain revolutionized the transfer of assets in the 21st century and enabled the creation and wide adoption of cryptocurrencies. Despite the great success of cryptocurrencies, consensus protocols' low performance makes their adoption as a daily payment method infeasible. Other factors that inhibit the advancement of cryptocurrencies as an alternative payment method are the high confirmation latency and the high value of fees. Thus, the Payment Channel Network (PCN) technology presents a fast and secure solution to the blockchain scalability problem. Payment channel networks introduce a new way of transacting, displaying high transaction throughput by minimizing the number of transactions recorded at the blockchain. This chapter addresses the payment channel networks technology to provide an efficient and agile cryptocurrencies transfer. We present a hands-on activity that uses PCNsim, a modular simulator of payment channel networks developed by GTA (Grupo de Teleinformática e Automação). The goal of this chapter is to demonstrate the key concepts of payment channel networks and associate these concepts with research challenges in computer networks and information security. It is expected that, by the end of the chapter, the readers will master the fundamentals of payment channel networks and develop skills to identify their advantages and disadvantages critically. Moreover, readers are expected to understand the challenges related to privacy and routing and be on the cutting-edge of the technology to foster high-level research in the area.

Este trabalho foi realizado com recursos do CNPq, CAPES, FAPERJ e FAPESP (2015/24494-8, 2018/23292-0, 2015/24485-9, 2014/50937-1).

Resumo

A corrente de blocos revolucionou a transferência de ativos no século XXI e permitiu a criação e a ampla adoção de criptomoedas. Apesar do grande sucesso das criptomoedas, o baixo desempenho dos protocolos de consenso utilizados ainda dificulta sua adoção como método de pagamento no dia-a-dia. Outros fatores que impedem o avanço das criptomoedas como método de pagamento alternativo são a alta latência de confirmação e o alto valor das taxas. Assim, a tecnologia de rede de canais de pagamento (Payment Channel Network - PCN) apresenta uma solução rápida e segura para o problema da escalabilidade da corrente de blocos. As redes de canais de pagamento introduzem uma nova maneira de transacionar, exibindo alta vazão de transações ao minimizar o número de transações que vão para a corrente de blocos. Este capítulo aborda de forma prática a tecnologia de redes de canais de pagamentos para prover a troca eficiente e ágil de criptomoedas. A atividade prática utiliza o PCNsim, um simulador modular de redes de canais de pagamento desenvolvido pelo GTA (Grupo de Teleinformática e Automação). O objetivo deste capítulo é demonstrar os fundamentos-chave das redes de canais de pagamento e relacionar esses conceitos aos desafios de pesquisa em redes de computadores e segurança da informação. Espera-se que, ao final do capítulo, seus participantes dominem os fundamentos de redes de canais de pagamentos, desenvolvendo as habilidades de identificar de forma crítica as suas vantagens e desvantagens. Mais ainda, espera-se que os participantes entendam os desafios relacionados à privacidade e roteamento e estejam na vanguarda da tecnologia para promover pesquisas de alto nível na área.

1.1. Introdução

A tecnologia de corrente de blocos (*blockchain*) revolucionou a transferência de ativos digitais através da Internet [Nakamoto 2008]. A partir das criptomoedas, principal aplicação da corrente de blocos, usuários ao redor do mundo podem efetuar pagamentos e transferências monetárias de maneira segura e distribuída. Para isso, a corrrente de blocos apresenta uma estrutura de dados distribuída que armazena todo o histórico de transações do sistema. O processamento das transações, antes feito por bancos, agências e governos, passa a ser feito por usuários, que entram em acordo através de um algoritmo de consenso distribuído para manter a consistência do registro. As criptomoedas alavancaram este método de transacionar, que rapidamente atingiu sucesso, chegando a mais de 100 milhões de usuários [Blockchain.com 2022b]. No início de maio de 2022, as duas principais criptomoedas, Bitcoin [Nakamoto 2008] e Ethereum [Wood 2014], possuem juntas valor de mercado maior que 1 trilhão de dólares [CoinMarketCap 2022]. Esse valor seria o suficiente para posicionar as duas criptomoedas na 17^a posição da lista de países com maior PIB mundial, a frente de mais de 150 países, de acordo com o Banco Mundial [World Bank 2022].

Apesar do grande sucesso das criptomoedas, o baixo desempenho dos protocolos de consenso utilizados ainda dificulta sua adoção como método de pagamento no dia-a-dia. Enquanto as principais criptomoedas em valor de mercado[2], Bitcoin e Ethereum, apresentam vazão de somente 7 e 15 transações por segundo, respectivamente, métodos convencionais de pagamento, como cartão de crédito, atingem mais de 1.700 transações

[2]`https://coinmarketcap.com/`. Acessado em 06 de maio de 2022.

por segundo[3]. Outros fatores que impedem o avanço das criptomoedas como método de pagamento alternativo são a alta latência de confirmação e o alto valor de taxa. Uma transação na rede Bitcoin leva em torno de uma hora para ser confirmada e as taxas podem chegar a 320 reais[4]. Isto significa uma taxa de quase 500% em relação ao valor médio de uma transação de débito no Brasil, que é de 67 reais, contra os cerca de 3% cobrados por empresas de pagamentos [Ribeiro 2022, Damasceno 2022]. Esses fatores tornam a tecnologia pouco atrativa para o uso cotidiano, em que é necessário confirmar uma transação em poucos segundos, cobrando taxas bem menores do que o valor transferido. O desafio de desempenho para adoção em massa de criptomoedas é conhecidos na literatura como o problema de escalabilidade da corrente de blocos (*blockchain scalability problem*).

A tecnologia de rede de canais de pagamento (*Payment Channel Network* - PCN) apresenta uma solução rápida e segura para o problema da escalabilidade da corrente de blocos [Poon e Dryja 2016]. Essa solução atrai a atenção não somente da academia, mas também do público empresarial. Atualmente, a principal rede de canais de pagamento, a Rede Relâmpago (*Lightning Network* - LN), possui aproximadamente 764 milhões de reais alocados em mais 85.000 canais[5]. As redes de canais de pagamento introduzem uma nova maneira de transacionar, exibindo alta vazão de transações ao minimizar o número de transações que vão para a corrente de blocos. Para isso, os pagamentos são realizados fora-da-corrente (*off-chain*), eliminando a necessidade dos protocolos de consenso, sendo somente o resultado final publicado na corrente de blocos. Contratos bloqueados por tempo e por *hash* (*Hashed Timelock Contracts* - HTLC) e técnicas de criptografia garantem a segurança das transações. Apesar de fornecerem uma maneira rápida e segura de transacionar, as redes de canais de pagamento possuem desafios em aberto, relacionados principalmente ao roteamento de transações, privacidade e disponibilidade dos participantes [Rebello et al. 2021a]. As redes de canais de pagamento apresentam alto potencial de pesquisa nas áreas de segurança em redes de computadores e sistemas distribuídos.

Este capítulo aborda a tecnologia de redes de canais de pagamentos para prover a troca eficiente e ágil de criptomoedas. O capítulo é organizado em três partes principais, que abordam: i) a tecnologia de corrente de blocos e seu problema de escalabilidade; ii) as redes de canais de pagamento; e iii) uma atividade que demonstra o funcionamento das redes de canais de pagamento na prática.

A atividade prática se baseia no PCNsim [Rebello et al. 2022], um simulador modular de redes de canais de pagamento desenvolvido pelo Grupo de Teleinformática e Automação. O objetivo deste capítulo é mostrar de forma clara, direta e sucinta os fundamentos-chave das redes de canais de pagamento e relacionar esses conceitos aos desafios de pesquisa em redes de computadores e segurança da informação. O capítulo proposto possui uma abordagem teórico-prática, apresentando e informando sobre a tecnologia de rede de canais de pagamento e se aprofundando nos aspectos da tecnologia ligados à área de redes de computadores. Diferentemente de outros minicursos em simpósios brasileiros que abordaram correntes de blocos e algoritmos de consenso [Antonio et al. 2021, Rebello et al. 2019], este capítulo é o primeiro a abordar o

[3]`https://en.bitcoin.it/wiki/Scalability`. Acessado em 06 de maio de 2022.
[4]`https://www.blockchain.com/charts`. Acessado em 06 de maio de 2022.
[5]`https://1ml.com`. Acessado em 06 de maio de 2022.

tema de canais de pagamento e soluções fora-da-corrente para o problema de escalabilidade das correntes de blocos. Este capítulo procura mostrar objetivamente os principais conceitos, protocolos e características das redes de canais de pagamento. Tópicos importantes que nem sempre são assimilados e dominados por iniciantes são abordados, tais como o roteamento de pagamentos de maneira segura, os contratos bloqueados por tempo e por *hash* e a resolução de conflitos de pagamentos na corrente de blocos. O capítulo apresenta as principais direções de pesquisa, propostas, implementações e desafios na área de redes de canais de pagamento para capacitar o leitor em futuras pesquisas.

O restante deste capítulo está organizado da seguinte forma. A Seção 1.2 foca nos problemas de escalabilidade das correntes de blocos atuais, ressaltando as principais causas e comentando sobre propostas de solução. A Seção 1.3 apresenta a fundamentação teórica de canais de pagamentos, detalhando a implementação de um canal de pagamento, abertura e fechamento de canais, resolução de conflitos e o roteamento de pagamentos através de canais existentes. A atividade prática a ser realizada durante o capítulo é detalhada na Seção 1.4, que elabora exemplos de pagamentos e ataques à rede de canais de pagamento. Por fim, a Seção 1.5 conclui o capítulo discutindo as tendências e os desafios de pesquisa em redes de canais de pagamento, além de resultados de pesquisas.

1.2. O Problema de Escalabilidade das Correntes de Blocos

As correntes de blocos têm se transformado na tecnologia mais promissora dos últimos tempos devido à sua característica disruptiva e inovadora. Em um futuro próximo, espera-se que, assim como a Internet hoje permite a transferência de arquivos, a tecnologia de corrente de blocos permita a transferência de ativos, como dinheiro [Nakamoto 2008], dados médicos [de Oliveira et al. 2019], votos [Kshetri e Voas 2018], modelos de aprendizado de máquina [de Souza et al. 2020], entre outros, sem intermediários, proporcionando uma camada de confiança distribuída.

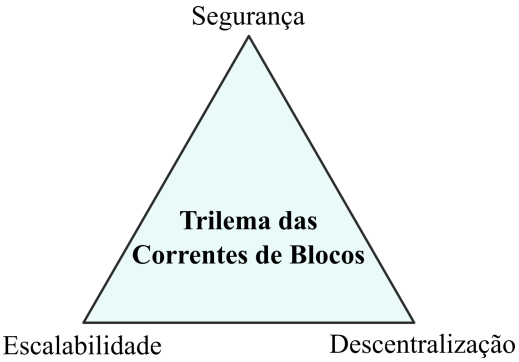

Figura 1.1. Trilema observável em sistemas baseados em corrente de blocos. Nenhuma corrente de blocos consegue prover, simultaneamente, um sistema altamente seguro, escalável e descentralizado.

No entanto, os principais sistemas atuais baseados em correntes de blocos, como as criptomoedas Bitcoin [Nakamoto 2008] e Ethereum [Wood 2014], ainda apresentam problemas significativos de latência, gasto de energia e desempenho. Publicar uma transação no Bitcoin leva aproximadamente uma hora, pode acarretar mais de 100 reais de taxa e gasta uma quantidade de energia suficiente para abastecer uma residência média brasileira

durante mais de dez meses [Digiconomist 2022, de Minas e Energia 2021]. O desempenho no Bitcoin é de aproximadamente 7 transações processadas por segundo, enquanto o Ethereum atinge 15 transações por segundo [Rebello et al. 2021b, Xiao et al. 2020]. Esses valores são incomparáveis com a média de mais de 4.500 transações por segundo registradas por grandes empresas de cartão de crédito [BitcoinWiki 2022, Visa Inc. 2022][6]. A dificuldade de tornar esses sistemas eficientes é conhecido popularmente como o "problema de escalabilidade em corrente de blocos" (*the blockchain scalability problem*) e é o principal desafio de pesquisa na área atualmente.

O problema da escalabilidade em corrente de blocos pode ser melhor compreendido através da enunciação de um trilema entre três propriedades: segurança, escalabilidade e descentralização. O trilema, proposto por Vitalik Buterin, criador do Ethereum, e ilustrado visualmente na Figura 1.1, enuncia que um sistema de corrente de blocos é capaz de prover no máximo duas das três propriedades simultaneamente. A ideia é que, por definição, a validação de transações no sistema ocorre através do acordo entre os participantes de um protocolo de consenso. Assim, quanto mais participantes o sistema possui, mais complexa e demorada é a tomada e divulgação de decisões na rede. Por outro lado, reduzir o número de participantes concentra o sistema em apenas alguns agentes, reduzindo o nível de descentralização e aumentando o monopólio financeiro da rede. Tentar prover uma alta vazão de transações com muitos participantes compromete a segurança, pois aumenta a probabilidade de blocos conflitantes serem propostos ao mesmo tempo, um problema conhecido como bifurcação na corrente de blocos. A bifurcação ocorre em protocolos de consenso que permitem a existência temporária de múltiplas soluções válidas para uma mesma rodada, devido à assincronia na comunicação entre os participantes. Quando os participantes eventualmente entram em acordo sobre a versão correta da corrente de blocos, o sistema descarta os blocos considerados inválidos. O sistema compromete sua segurança nesse caso, pois as transações presentes nos blocos inválidos são revertidas e tornam-se não-rastreáveis a partir da corrente de blocos. Assim, um ataque comum é se aproveitar das bifurcações para realizar pagamentos que são válidos temporariamente, mas que serão revertidos quando a rede invalidar o bloco. O trilema, apesar de partir de uma definição informal baseada em observação, ocorre em todos os principais sistemas de corrente de blocos conhecidos [Xiao et al. 2020, Rebello et al. 2021b, Gudgeon et al. 2020].

A escalabilidade das correntes de blocos depende principalmente de fatores relacionados aos processos de tomada e divulgação de decisões, como tempo de transmissão das mensagens, custo computacional de verificação das transações e tempo de formação de um novo bloco [Zhou et al. 2020, Kim et al. 2018, Xie et al. 2019]. Assim, o cerne do problema da escalabilidade está, na verdade, nos protocolos de consenso responsáveis pela adição de novos blocos na corrente. Consenso é o processo pelo qual um grupo de participantes independentes atinge a mesma decisão coletiva de aceitar ou recusar um novo bloco a ser incorporado na corrente de blocos. O protocolo de consenso em corrente de blocos é o algoritmo distribuído que garante que o sistema evolui corretamente, adicionando um novo bloco por vez. A Figura 1.2 ilustra o funcionamento de um protocolo de consenso genérico. Nele, um participante especial chamado líder do consenso reúne as

[6]Cálculo baseado no relatório de desempenho oficial da Visa que mostra aproximadamente 145 bilhões de transações processadas no ano de 2020 [Visa Inc. 2022].

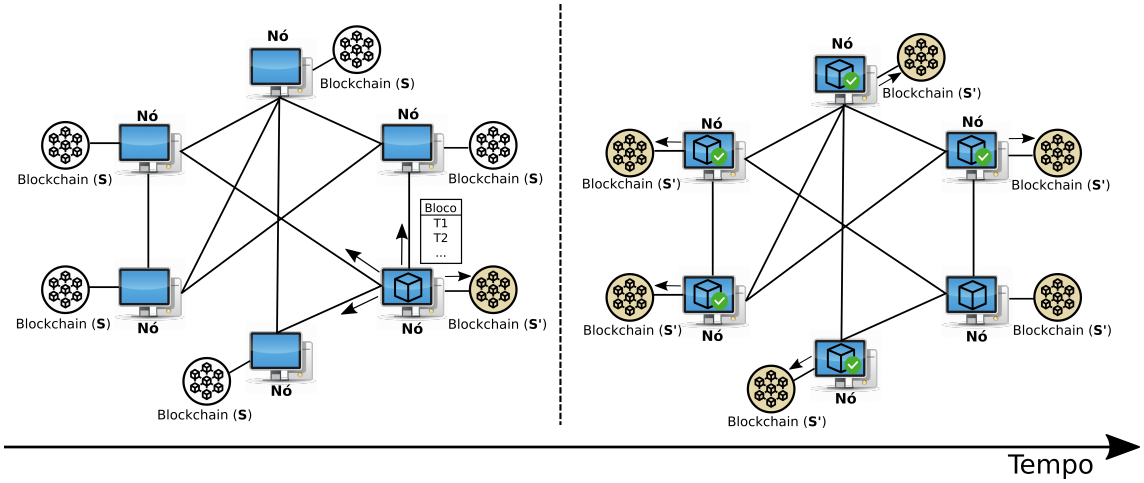

Figura 1.2. Atualização de uma corrente de blocos (*blockchain*) através de um protocolo de consenso genérico. Um dos participantes do consenso propõe, em difusão, um novo bloco que muda o estado global de S para S'. Os demais participantes verificam e adicionam independentemente o bloco proposto à corrente de blocos, replicando o novo estado global S' de maneira consistente.

transações recebidas em um novo bloco e o difunde na rede para ser validado localmente pelos demais participantes do consenso. Ao receber o bloco proposto, cada participante verifica o bloco independentemente e, caso aprove, o adiciona à corrente de blocos, atingindo localmente o estado S'. Quando a maioria dos participantes atinge o novo estado localmente, considera-se que houve consenso e que o sistema como um todo validou o novo bloco, incrementando o estado global para S'. Os procedimentos de proposição, difusão e verificação do bloco garantem a segurança das transações no sistema, mas implicam um gasto de tempo proporcional ao número de participantes. Definir quem será o participante responsável por publicar o próximo bloco e quais são as próximas transações que o constituirão também são tarefas árduas que consomem tempo e energia. Assim, o mecanismo de consenso torna-se o principal gargalo do sistema de corrente de blocos.

Como segurança é indispensável em correntes de blocos, na prática o trilema enunciado se transforma em um dilema para os protocolos de consenso: prover escalabilidade, medida em número de transações processadas por segundo, ou descentralização, medida em número de participantes. O compromisso entre as duas propriedades pode ser observado claramente na comparação entre os principais protocolos de correntes de blocos mostrada na Figura 1.3. Protocolos baseados em prova utilizam desafios computacionais como mecanismo de definição de líder, permitindo que qualquer usuário participe do processo. No entanto, esses protocolos apresentam baixa vazão de transações devido ao custo energético e ao tempo para resolver o desafio. Assim, os protocolos baseados em prova são pouco escaláveis, mas altamente descentralizados, o que os torna mais adaptados a sistemas públicos com muitos usuários. Os principais sistemas que utilizam esse tipo de consenso são as criptomoedas, como Bitcoin e Ethereum [Nakamoto 2008, Wood 2014], que utilizam a prova de trabalho (*Proof of Work* - PoW). Por outro lado, protocolos baseados em comitê elegem um grupo de participantes especiais para definir os blocos através de comunicação direta. A escolha de quem participa do comitê pode ser realizada de diversas formas, como, por exemplo, aleatoriamente, por voto direto dos usuários por quan-

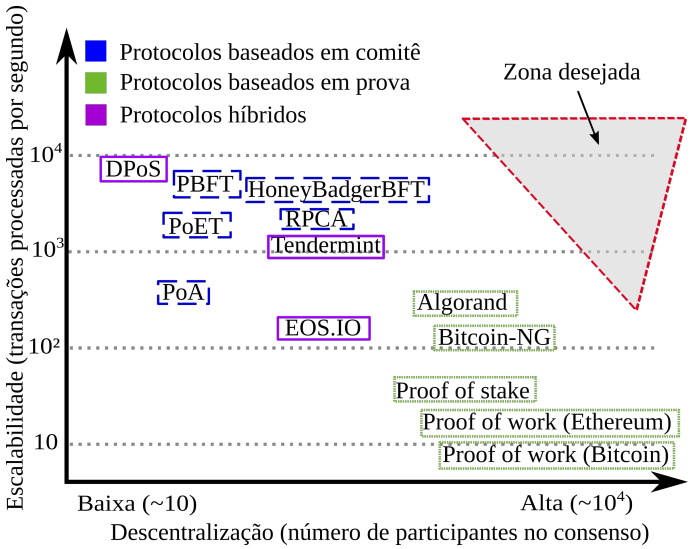

Figura 1.3. Comparação entre os principais protocolos de consenso de sistemas baseados em corrente de blocos. O compromisso observado entre desempenho e descentralização dificulta a proposta de um protocolo que seja simultaneamente escalável e tolerante a ataques de conluio.

tidade de moedas investidas. A organização em comitê sacrifica a descentralização, pois não é qualquer usuário que pode participar, mas aumenta a quantidade de transações processadas por segundo, uma vez que as decisões não dependem de custosos desafios computacionais. Os protocolos baseados em comitê são adaptados, portanto, a sistemas que já admitam algum tipo de centralização, como consórcios de empresas, bancos, e governos. Os principais exemplos deste tipo de sistema são o Hyperledger Fabric, Hyperledger Sawtooth e Ripple (RPCA) [Androulaki et al. 2018, Chen et al. 2017, Schwartz et al. 2014]. Assim, o projeto dos protocolos baseados em comitê deve incluir mecanismos robustos para garantir a segurança do sistema, pois a centralização facilita ataques de conluio e negação de serviço.

Há diversos protocolos de consenso que tentam resolver o problema da escalabilidade com descentralização através de soluções que visam combinar o melhor das abordagens baseadas em prova e das abordagens baseadas em comitê [Larimer 2017, Amoussou-Guenou et al. 2019, Yang et al. 2019]. O principal objetivo é prover cada propriedade em um momento específico enquanto mecanismos extra-consenso mitigam possíveis vulnerabilidades. No entanto, a abordagem híbrida também sofre com o compromisso entre escalabilidade e descentralização, atingindo valores intermediários nos dois quesitos para os principais protocolos híbridos, EOS.IO e Tendermint [Larimer 2017, Amoussou-Guenou et al. 2019]. Na prática, ainda não existe um protocolo de consenso que se destaque ao ponto de atingir milhares de transações por segundo com um baixo tempo de confirmação. Apesar dos esforços, até hoje existe uma "zona ideal" desejada mas não alcançada, vista na Figura 1.3, que permitiria escalar os sistemas de corrente de blocos sem comprometer sua descentralização.

As principais soluções de escalabilidade em correntes de blocos tendem a focar em aspectos extra-consenso que podem ser melhorados [Poon e Dryja 2016, Popov 2017, Rebello et al. 2021a, Zhou et al. 2020, Luu et al. 2016, Wang et al. 2019a]. Este capítulo

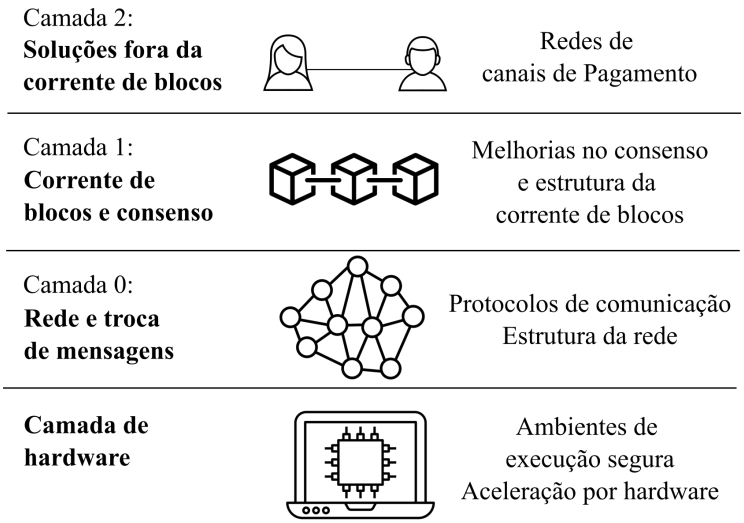

Figura 1.4. Camadas da arquitetura de sistemas baseados em corrente de blocos, adaptada de Gudgeon *et al.* [Gudgeon et al. 2020]. A maior parte das propostas para melhorar a escalabilidade desses sistemas encontra-se nas camadas 1 e 2, por serem as camadas mais fáceis de modificar em larga escala dentro de um ambiente descentralizado.

adota uma arquitetura em camadas como referência para melhor compreensão das propostas analisadas [Gudgeon et al. 2020]. A arquitetura, ilustrada na Figura 1.4, divide os sistemas baseados em corrente de blocos em quatro camadas em ordem crescente de complexidade: a camada de *hardware*, composta pelos dispositivos; a Camada 0 (zero), que abriga a infraestrutura de rede e troca de mensagens; a Camada 1, composta pelo protocolo de consenso e a corrente de blocos; e a Camada 2, que abrange tecnologias que realizam o máximo possível de processamento através de trocas de mensagens fora da corrente de blocos e sem o uso de protocolos de consenso, utilizando-a apenas quando necessário.

1.2.1. Camada de Hardware

Na camada mais baixa da arquitetura, a maior parte das propostas busca aumentar o desempenho das correntes de blocos através de soluções baseadas em hardware, como aceleração por hardware e ambientes de execução segura.

Dentre as propostas de aceleração, Sakakibara *et al.* propõem o uso da tecnologia de placas de rede programáveis baseadas em FPGA (*Field Programmable Gate Array*) para implementar uma memória cache que armazena os blocos mais procurados e os transmite rapidamente a usuários que os solicitem [Sakakibara et al. 2018]. A técnica acelera o acesso aos blocos permitindo que dispositivos obtenham informações diretamente da placa e, com isso, reduz em até 7x o tempo de acesso às transações pelos usuários. Outra proposta similar utiliza a mesma tecnologia para tornar mais rápida a verificação e divulgação de blocos dentro de uma rodada do protocolo de consenso [Javaid et al. 2021]. Os resultados indicam que o uso da FPGA para acelerar o consenso aumenta em até 12x a vazão total do sistema. Tecnologias de hardware ainda têm sido utilizadas para garantir a segurança de dispositivos e dados em sistemas de corrente de blocos para cenários específicos, como o de Internet das Coisas [Mohanty et al. 2020].

O uso da tecnologia de ambientes de execução segura (*Trusted Execution Environments* – TEE), como o Intel SGX (*Software Guard Extensions*), também previne diversas possíveis ações maliciosas dos participantes do consenso. Como consequência, o emprego das TEEs permite que os protocolos nas camadas superiores relaxem ou eliminem etapas de segurança que consomem tempo excessivo [Lind et al. 2019, Lind et al. 2017, Costan e Devadas 2016]. Essa tecnologia é a base de alguns protocolos de consenso altamente eficientes, como a prova de tempo decorrido (*Proof of Elapsed Time* - PoET) do Hyperledger Sawtooth [The Hyperledger Foundation 2022]. O protocolo se aproveita do Intel SGX para substituir o desafio computacional por temporizadores simples que não podem ser adulterados. Assim, a escolha do líder resume-se a sortear um valor aleatório para o temporizador de cada participante, cabendo aos participantes proporem blocos de forma ordenada conforme os temporizadores expiram. Esse processo reduz o tempo de uma rodada de consenso e aumenta a escalabilidade do protocolo para até 2.300 transações por segundo [Ampel et al. 2019].

A principal limitação de propostas que se beneficiam de hardwares mais eficazes é a heterogeneidade dos equipamentos dos usuários na rede. Uma vez que não há autoridade central em sistemas de corrente de blocos, dificilmente é possível garantir que todos os participantes possuam os mesmos equipamentos necessários para a implementação das melhorias. Assim, esse tipo de abordagem costuma ser mais eficaz em ambientes controlados e centralizados, que não representam as principais aplicações de corrente de blocos [Xiao et al. 2020, Ampel et al. 2019, Rebello et al. 2021b]. A maioria das propostas para ambientes altamente descentralizados com muitos usuários se concentra nas camadas superiores da arquitetura.

1.2.2. Camada 0: Rede e Troca de Mensagens

Na Camada 0, as soluções para a escalabilidade são relacionadas à comunicação eficiente. Assim, as principais alternativas para aumentar a vazão nesta camada são reduzir a quantidade de informações redundantes transmitidas para os nós da rede e otimizar o tamanho dos blocos transmitidos.

Uma solução ingênua para aumentar o número de transações por segundo inseridas na corrente de blocos é produzir no mesmo intervalo de tempo atual blocos maiores. O crescimento do tamanho máximo do bloco possibilita aumentar a quantidade de transações inseridas na corrente de blocos a cada rodada de consenso, uma vez que o bloco pode conter um número variável de transações. Por exemplo, a corrente de blocos Bitcoin, a pioneira na transferência de criptomoedas e atualmente a corrente com o maior número de participantes, totalizando aproximadamente 82 milhões, produz um bloco a cada 10 minutos [Nakamoto 2008] e um tamanho médio de 1,2 MB aproximadamente [Blockchain.com 2022c]. Como cada bloco possui em média 1.810 transações [Blockchain.com 2022a], atualmente a rede possui uma vazão média de 3 transações por segundo para transações com um tamanho típico de 663 B aproximadamente. Para atingir uma taxa de 24 mil transações por segundo, como a da administradora de cartões VISA [Visa 2022], fixando o tempo médio de produção de um bloco e o tamanho médio das transações, os blocos deveriam possuir 14,4 milhões de transações, e seu tamanho total seria de aproximadamente 9 GB. Entretanto, ao atingir um tamanho muito grande, a transmissão e validação das informações se tornariam um novo gargalo para o

sistema. Há o aumento do tempo de propagação na rede, pois além do tempo de envio da mensagem há o tempo de verificação das informações recebidas. Essa alternativa acaba se aproximando do tempo necessário para formar um novo bloco, e assim, há o favorecimento para a criação de bifurcações na corrente de blocos. Outro fator que desmotiva essa estratégia é o tempo de confirmação, pois a abordagem mantém o tempo entre blocos em 10 minutos, o que é irreal para aplicações cotidianas, como compras cotidianas em lojas físicas. Em geral, o tempo de confirmação é considerado 6 vezes maior do que o tempo para a confirmação de um bloco, para reduzir a probabilidade da transação fazer parte de um bloco órfão. Além disso, o custo de armazenamento das informações da corrente de blocos restringe a adoção de blocos muito grandes, pois nesse caso, apenas nós com alta capacidade de armazenamento conseguiriam manter a cópia completa da corrente de blocos. Esse fator enfraquece o propósito de descentralização da tecnologia de corrente de blocos, tornando a rede mais concentrada em alguns nós e mais próxima de soluções tradicionais de bancos de dados.

Uma segunda alternativa é reduzir o tamanho das transações a fim de aumentar o número de transações por bloco. Os dados que mais geram sobrecarga no tamanho das transações são as informações criptográficas utilizadas para verificar a autenticidade das transações. Nos sistemas de correntes de blocos, as assinaturas digitais representam cerca de 60 a 70 por cento do tamanho total de uma transação [Xie et al. 2019]. Uma abordagem para reduzir o espaço ocupado por esses dados nas transações é a Testemunha Segregada (*Segregated Witness* - SegWit) [Lombrozo et al. 2015].

O SegWit é uma proposta de mudança no protocolo Bitcoin, funcionando desde agosto de 2017, com o objetivo inicial de mitigar um problema de segurança denominado maleabilidade de transações [Decker e Wattenhofer 2014]. A maleabilidade é uma propriedade de algoritmos criptográficos. Algoritmos de encriptação maleáveis possibilitam a alteração de uma cifra mantendo o mesmo resultado de decriptação. A maleabilidade de transações ocorre quando uma transação válida em espera é duplicada com identificadores diferentes. A possibilidade do ataque ocorre devido ao armazenamento da assinatura da transação em um dos campos da própria transação. Como o identificador da transação é um resumo de seu conteúdo, qualquer modificação na transação gera um novo resumo complemente diferente. Porém, é possível replicar uma transação com uma assinatura válida e diferente, permitindo que a mesma transação possua uma cópia autêntica com um novo identificador. Assim, a solução propôs remover as assinaturas das transações para um campo externo, reduzindo notavelmente o tamanho da transação [Cheow 2020].

O campo externo, uma extensão de 3 MB nos blocos SegWit, previne que a assinatura faça parte do identificador da transação. Além disso, os nós legados recebem apenas a parte referentes às transações, 1 MB de dados, permitindo assim a visualização de mais transações por bloco. Assim, os clientes SegWit visualizam blocos maiores, enquanto clientes legados visualizam transações menores. A Figura 1.5 ilustra a diferença entre uma transação legada em relação a uma transação SegWit. As transações SegWit possuem duas partes: a primeira parte da transação contém os endereços da carteira do remetente e do destinatário e a segunda parte, denominada dados da testemunha (*witness data*), contém as assinaturas da transação. Portanto, a remoção das assinaturas da transação para o armazenamento externo no campo de testemunhas resolveu o problema de maleabilidade de transações e aumentou a vazão máxima do sistema. Entretanto, o au-

mento da vazão proporcionado pela solução ainda é distante dos valores desejados para atender clientes em uma escala global. Além disso, a solução mantém o alto tempo de confirmação inalterado.

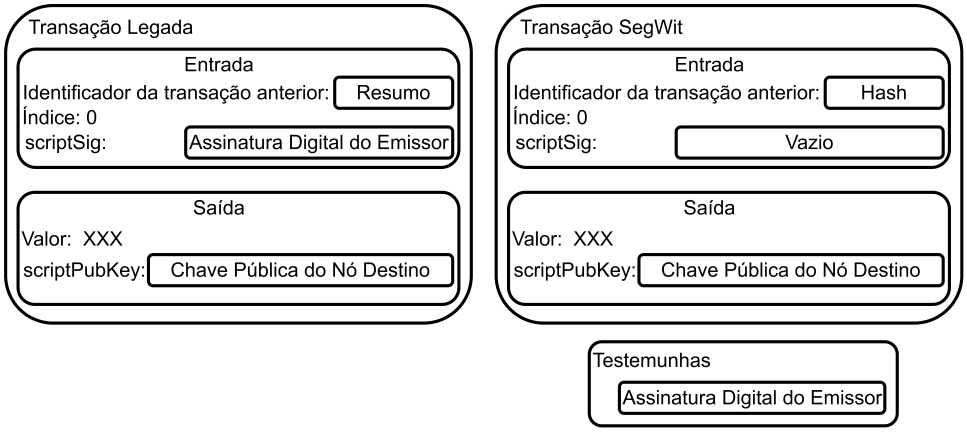

Figura 1.5. Comparação da estrutura de uma transação legada com uma transação SegWit. As transações legadas incluem a assinatura digital na própria transação, gerando uma sobrecarga no tamanho final da transação. Por outro lado, a transação SegWit dissocia a assinatura do emissor da transação através da criação de um campo externo denominado testemunhas.

Por fim, outra estratégia na Camada 0 é reduzir o tempo de criação entre os blocos. Entretanto, reduzir o tempo de criação dos blocos aumenta a probabilidade de bifurcações na corrente de blocos. Isso ocorre devido à probabilidade de existir uma bifurcação depender da razão do tempo de propagação de um bloco para a rede em relação ao tempo necessário para minerar um novo bloco. Quanto mais próximos os tempos de propagação e mineração forem, maior será a probabilidade de bifurcações. Por exemplo, a plataforma Ethereum [Wood 2014] reduz o tempo entre os blocos em relação ao Bitcoin [Nakamoto 2008] de 600 segundos para apenas 14 segundos. Porém, devido à maior probabilidade de existência de bifurcações, os nós da rede geralmente aguardam 250 blocos [LetsExchange 2021] para determinar que uma transação foi finalizada. Assim, o tempo de confirmação na corrente de blocos Ethereum é alto e da mesma ordem de grandeza que no Bitcoin, durando cerca de 58 minutos.

1.2.3. Camada 1: Corrente de Blocos e Consenso

As soluções de escalabilidade na camada 1 podem ser dividas em duas partes: soluções na corrente (*on-chain*) e fora-da-corrente (*off-chain*). Assim, esta seção discute primeiramente as soluções na corrente, e por fim discute soluções fora-da-corrente.

Na corrente. O livro-razão distribuído baseado em um grafo acíclico dirigido (*directed acyclic graph* - DAG) é outra tecnologia que possui o potencial de aumentar a vazão de transações por segundo enquanto garante propriedades similares à corrente de blocos [Gopalan et al. 2020]. Os DAGs são estruturas de dados de caminho imutável encadeadas por *hash*. Enquanto conjuntos de transações são encadeados em corrente de blocos, em um DAG, cada transação é encadeada de forma independente. A principal diferença entre uma corrente de blocos e um DAG é que uma nova transação pode referenciar qualquer transação predecessora, não apenas a última. Além disso, ao contrário

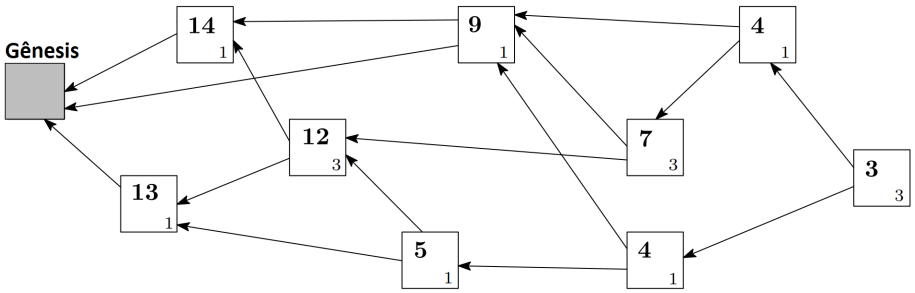

Figura 1.6. Exemplo da estrutura de dados do IOTA. As transações constituem um grafo acíclico dirigido. Cada transação é encadeada com duas transações predecessoras, com exceção das primeiras transações da rede. A estrutura garante propriedades similares às da corrente de blocos.

das correntes de blocos, os DAGs permitem a validação e o processamento de transações, sem depender de um protocolo de consenso. Essas características combinadas permitem uma vazão de transações muito maior, enquanto preservam os requisitos de segurança oferecidos pelas correntes de bloco [Alvarenga et al. 2021, Zhao e Yu 2019].

O IOTA [Popov 2017] apresenta uma criptomoeda construída para atender a micro-pagamentos máquina a máquina (*machine to machine* - M2M) característicos de um ambiente de Internet das Coisas. Seus mecanismos de pagamento e protocolo de consenso, formalizados por Popov em 2016 [Popov 2017], baseiam-se em uma estrutura de dados inovadora chamada de *Tangle*. A principal inovação do *Tangle* é a estrutura de livro-razão distribuído, que reúne as transações da rede em um DAG em vez de utilizar uma corrente de blocos. Além disso, o Tangle elimina a distinção entre clientes e mineradores: todos os usuários do sistema devem realizar trabalho para emitir uma nova transação. Uma característica notável do *Tangle* em comparação ao consenso em corrente de blocos é que diferentes participantes na rede podem ter diferentes visões das transações. Esta característica contrasta fortemente com a visão global da corrente de blocos, na qual todas as transações são idênticas em qualquer participante.

A Figura 1.6 mostra um exemplo da estrutura de dados do IOTA, o *Tangle*. Cada vértice do grafo representa uma transação e cada aresta representa o resultado da validação de uma transação. O usuário deve confirmar ao menos duas transações não-confirmadas para adicionar sua transação à *Tangle*. As transações não-confirmadas são chamadas de "pontas" (*tips*) do *Tangle*. Para adicionar uma transação à rede, o usuário adiciona os resumos das duas pontas escolhidas à sua transação, resolve um desafio baseado em prova de trabalho, e difunde o resultado na rede. A prova de trabalho, neste caso, tem dificuldade bem menor que a do Bitcoin e serve apenas como um mecanismo para prevenir *spam* de transações. O procedimento de adição de uma transação cria duas novas arestas direcionadas no grafo que confirmam as transações anteriores e funcionam como uma versão generalizada da sequência de funções resumo (*hashes*) da corrente de blocos. Não existe um mecanismo de consenso no IOTA Tangle que previna a adição de transações conflitantes, que realizam gasto duplo na rede. Atualmente, a confirmação da transação IOTA é centralizada por um coordenador, o que prejudica gravemente o tempo e a confiança da confirmação da transação [Wang et al. 2020].

Outros exemplos de sistemas que utilizam o DAG são o Byteball e o Hashgraph.

No Byteball, as transações anexadas convergem gradualmente para a cadeia principal usando nós especiais, denominados testemunhas (*witnesses*), com poder de votação baseado em reputação [Churyumov 2016]. Por outro lado, o Hashgraph é uma proposta inspirada em protocolos de consenso no estilo tolerante à falhas bizantinas (*Byzantine Fault Tolerance* - BFT) que são baseado em voto entre as testemunhas [Baird e Luykx 2020]. No entanto, no Byteball e no Hashgraph, os participantes exigem a mesma visualização de grafo para perceber a mesma ordem de transações. Assim, as propostas em DAG relaxam a propriedade de segurança, p. ex., permitindo a existência de visões discordantes da ordem das transações entre participantes do sistema, para garantir uma maior escalabilidade.

A fragmentação (*sharding*) [Zhou et al. 2020, Luu et al. 2016] é uma técnica originalmente proposta para o processamento ágil de bases de dados [Corbett et al. 2013], que divide os dados a serem processados em grupos menores, denominados fragmentos (*shards*). No contexto das correntes de blocos, os fragmentos são subgrupos constituídos de nós da rede. O objetivo da abordagem é paralelizar o processamento das transações em cada fragmento e diminuir a sobrecarga na corrente de blocos. Dessa forma, há a expectativa de que a quantidade de transações por segundo da rede cresça linearmente conforme o número de grupos existentes cresça. Enquanto algumas propostas de escalabilidade são limitadas a aplicações financeiras, p. ex., transferência de criptomoedas entre dois nós, a fragmentação possui a vantagem de lidar com diversas tarefas de computação, permitindo a escalabilidade de aplicações genéricas que utilizam contratos inteligentes.

O principal desafio da fragmentação é garantir a segurança da estrutura da corrente de blocos. Como o processamento de transações diferentes é realizado de forma paralela em cada fragmento, é necessário definir mecanismos que permitam transações entre fragmentos distintos e, ao mesmo tempo, evitem o gasto duplo [Huang et al. 2022]. A realização de transações entre fragmentos distintos gera sobrecarga na comunicação entre os nós devido à necessidade de sincronizar a informação de forma global. Um limitante para o aumento da vazão linear em relação ao número de fragmentos criados é o aumento das transações entre fragmentos. Estima-se que o número de transações entre fragmentos possa ultrapassar 90% do total de transações quando o número de fragmentos é superior a 64 e os nós são alocados nos fragmentos de forma aleatória [Wang et al. 2019a, Wang e Wang 2019]. Assim, alocar os nós que executam transações recorrentes nos mesmos fragmentos é uma alternativa para aumentar a vazão de transações e evitar gargalos nos sistemas baseados em fragmentos. Porém, a escolha aleatória dos nós participantes de cada fragmento garante uma maior segurança.

O Elastico é o primeiro sistema público de corrente de blocos baseado em fragmentação [Luu et al. 2016]. Cada fragmento da rede realiza a validação de um conjunto de transações de forma paralela executando o protocolo de consenso prático tolerante a falhas bizantinas (*Practical Byzantine Fault Tolerance* - PBFT) [Castro e Liskov 1999]. Os participantes de cada fragmento são escolhidos a cada época através de um desafio de prova de trabalho para evitar que um atacante crie diversas identidades e consiga corromper o resultado de um fragmento. Após essa etapa, um fragmento especial, denominado comitê de consenso, é responsável por formar o bloco global, um conjunto de todas as transações validadas de todos os fragmentos da rede. Entretanto, o sistema utiliza os benefícios da fragmentação apenas para o processamento de transações, enquanto as

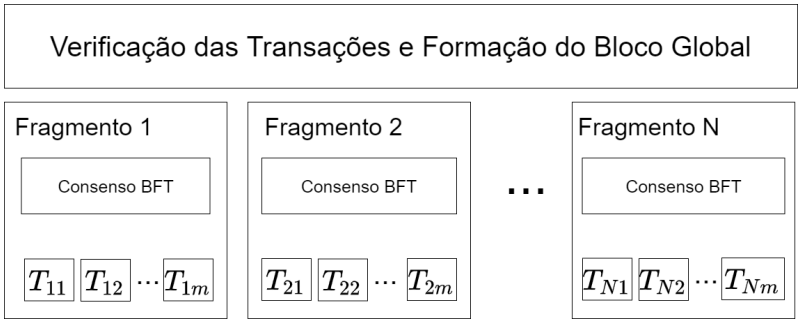

Figura 1.7. Execução de uma rodada de consenso em um sistema baseado em fragmentação. Cada fragmento gera um conjunto de transações válidas através de um acordo bizantino. Após essa etapa, os conjuntos de transações são combinados para formar o bloco global.

questões de armazenamento e comunicação permanecem desafios em aberto. Além disso, há sobrecarga computacional no processo de seleção dos membros dos fragmentos a cada época, prejudicando a escalabilidade da proposta.

O Omniledger surgiu como uma proposta alternativa ao Elastico, tendo como foco evitar as suas principais desvantagens [Kokoris-Kogias et al. 2018]. A proposta combina o RandHound [Syta et al. 2017] com o Algorand [Chen e Micali 2019] para prover um mecanismo público de escolha dos participantes dos fragmentos de forma aleatória e resistente a comportamentos maliciosos. O OmniLedger introduz um protocolo de duas fases, bloquear e desbloquear, denominado Atomix, para garantir atomicidade das transações entre fragmentos distintos. Outra vantagem da proposta é que os validadores salvam apenas parte do histórico de transações que resumem o estado do fragmento, a fim de diminuir a sobrecarga de armazenamento. Além disso, os autores propõem a substituição da corrente de blocos por um DAG para aumentar o paralelismo no processamento dos blocos. Entretanto, o uso de operações computacionalmente custosas e a necessidade de participação intensa dos nós da rede em transações entre fragmentos torna a proposta inconveniente para nós com baixo poder de processamento.

Outra solução para aumentar a vazão dos sistemas de correntes de blocos é o RapidChain [Zamani et al. 2018]. A proposta possui uma resiliência maior a falhas bizantinas, tolerando até 1/3 de participantes maliciosos como os acordos bizantinos determinísticos tradicionais [Lamport et al. 1982], superior ao 1/4 tolerado pelos sistemas Elastico [Luu et al. 2016] e OmniLedger [Kokoris-Kogias et al. 2018]. Os resultados mostram que a proposta atinge aproximadamente 4.000 transações por segundo para uma rede com 9 fragmentos, apresentando uma vazão 100 vezes maior do que a do Elastico e 8 vezes maior do que a do OmniLedger. Dang *et al.* propõem um sistema de corrente de blocos baseado em fragmentação com o uso de um ambiente de execução confiável (*Trusted Execution Environment* - TEE) para a seleção dos grupos [Dang et al. 2019]. Apesar de inovadora, a proposta dos autores obtém uma vazão de apenas 3.000 transações por segundo aproximadamente para uma rede com 36 fragmentos, cada um contendo 4 clientes. Entretanto, todas as propostas anteriores ainda apresentam uma vazão máxima muito inferior aos sistemas de pagamento tradicionais [Visa 2022].

O Pyramid é um sistema que prevê a interseção entre fragmentos distintos para

reduzir a sobrecarga gerada por transações entre fragmentos [Hong et al. 2021]. Dessa forma, os autores propõem a verificação das transações entre fragmentos de forma eficiente por nós que fazem parte dos fragmentos referenciados pela transação. Assim, a proposta reduz tanto a sobrecarga computacional de protocolos complexos para garantir a sincronia de informações entre os fragmentos, quanto o armazenamento de transações globais por todos os nós da rede, restringindo o armazenamento das transações entre os fragmentos aos nós de borda. Além dos sistemas destacados, outras propostas utilizam a fragmentação para o aumento da vazão de transações, como Zilliqa [Team e Barrett 2018], MultiVAC [Bugday et al. 2019], e Monoxide [Wang e Wang 2019]. Entretanto, prover escalabilidade, descentralização e segurança é um desafio atual nos tópicos de pesquisa que combinam a corrente de blocos e o uso de fragmentos.

O DAG e a fragmentação são propostas que alteram a estrutura da corrente de blocos ou o processo de validação de transações. Essas categorias de proposta são denominadas modificações na corrente (*on-chain*). Entretanto, há outras propostas que mantêm a estrutura da corrente de blocos principal inalterada, realizando modificações externas através de correntes de blocos secundárias e referenciando as alterações de estado entre os nós através da corrente de blocos principal. Essas propostas são denominadas modificações fora da corrente (*off-chain*).

Fora-da-corrente. A corrente cruzada (*cross-chain*) é uma tecnologia que surgiu com a finalidade de permitir a interoperabilidade entre correntes de blocos. A partir da difusão de diversas correntes de blocos no mercado digital, é atrativo garantir a comunicação entre correntes de blocos distintas. Dessa forma, os recursos existentes em um sistema de corrente de blocos são mapeados em um segundo sistema de forma coordenada para garantir a unicidade dos ativos e evitar ataques de gasto duplo. Alice que possui recursos na corrente de blocos X pode transferir ou receber ativos para Bob, usuário da corrente de blocos Y. Um protocolo de corrente cruzada deve garantir a transferência atômica de recursos entre os sistemas envolvidos. Para isso, geralmente há duas etapas para a transferência do recurso: bloqueio dos recursos na origem e repasse de recursos no destino. Apesar de sua origem ter como objetivo a interoperabilidade entre correntes de blocos, as correntes de blocos cruzadas têm sido usadas como alternativas para aumentar a escalabilidade dos sistemas. As correntes de blocos cruzadas constituem um pilar para a proposta da tecnologia de correntes laterais (*sidechain*), pois permitem mapear os recursos existentes em uma corrente de blocos principal, lenta e com diversos participantes, em correntes de blocos menores, rápida e com poucos participantes [Yang et al. 2020].

O conceito de corrente de blocos lateral foi apresentado inicialmente por Back *et al.* através da plataforma Pegged Sidechain [Back et al. 2014]. As arquiteturas propostas de correntes laterais para escalabilidade preveem o uso de uma corrente de blocos principal, mais lenta, que sincroniza de forma global todos os usuários do sistema. Paralela à corrente de blocos principal, existem correntes de blocos secundárias, mais rápidas, criadas sob demanda, que executam protocolos de consenso que podem inclusive diferir da corrente de blocos principal [Singh et al. 2020]. A velocidade das correntes de blocos secundárias é obtida pelo baixo número de participantes presentes. As correntes de blocos secundárias são criadas sob demanda dos usuários que necessitam executar tarefas genéricas de forma recorrente. Dessa forma, as correntes secundárias incorrem em taxas menores para realizar as transações ou executar contratos inteligentes [Poon e Buterin 2017].

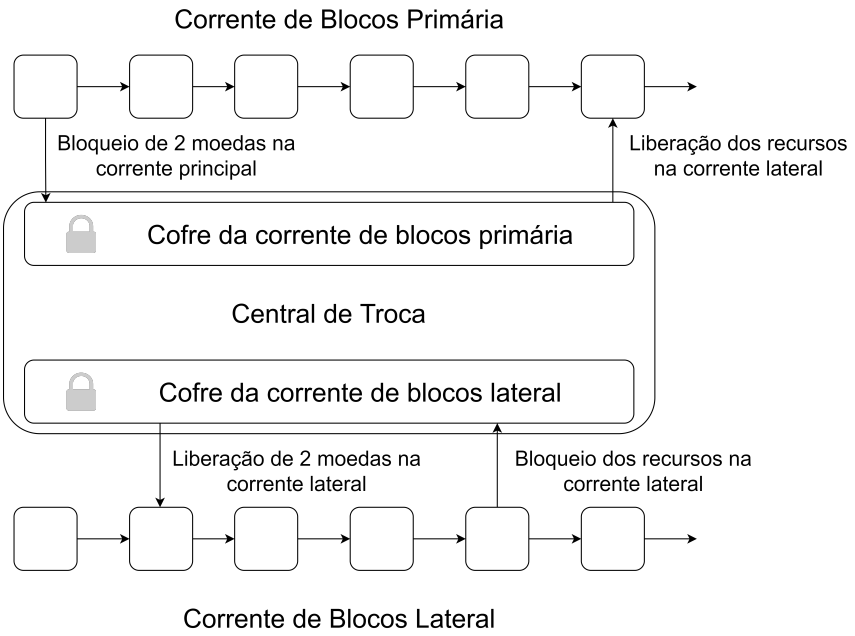

Figura 1.8. Exemplo de uma corrente lateral com o processo de troca entre correntes centralizado. Um usuário das duas correntes de blocos inicialmente bloqueia recursos na corrente de blocos principal e os obtêm na corrente de blocos lateral. A central de troca supervisiona as duas correntes de blocos e garantir a transferência correta dos ativos. Por fim, o usuário pode realizar o caminho reverso e obter os recursos novamente na corrente de blocos principal.

Por fim, quando os participantes da corrente de blocos secundária decidem, os estados alterados na corrente de blocos secundária são enviados à corrente de blocos principal. Outra vantagem para o sistema é que o dado publicado na corrente de blocos principal é apenas um resumo do estado atual da corrente de blocos secundária, economizando espaço em disco para os nós da corrente de blocos principal.

A central de troca é uma forma simples de realizar a troca entre as correntes de blocos. A central pode ser implementada por uma entidade confiável, como uma corretora de confiança das partes envolvidas, ou por um contrato inteligente associado às duas correntes de blocos. A Figura 1.8 exibe o exemplo de uma central de troca. Um usuário da corrente de blocos principal deseja realizar transações na corrente de blocos lateral. Assim, o usuário executa uma transação que bloqueia a quantidade de recursos desejada com o auxílio da central de troca. Uma vez que os recursos são bloqueados na corrente de blocos principal, a central de troca permite a liberação da quantia na corrente de blocos lateral. Após realizar as transações desejadas na corrente de blocos lateral, o usuário realiza o caminho inverso para obter seus ativos novamente na corrente de blocos principal. A escalabilidade do sistema tem o potencial de aumentar consideravelmente com a existência de correntes laterais, considerando que os resumos publicados na corrente de blocos principal são eventuais.

O principal desafio da abordagem é garantir a segurança nas correntes de blocos laterais. A criação de correntes de blocos menores favorece atacantes, pois a sua influência no resultado do consenso tende a aumentar com a diminuição do número de participantes. Assim, os participantes das correntes de blocos laterais de-

vem ser selecionados cuidadosamente, através da confiança e reputação, ou estabelecendo um número razoável de participantes de forma a minimizar o risco de ataques. Entre as principais implementações de correntes laterais existentes, pode-se destacar: Plasma [Poon e Buterin 2017], correntes satélites [Li et al. 2017], Chainlink [Ellis et al. 2017] e Cosmos [Kwon e Buchman 2019].

Plasma é uma alternativa para reduzir as taxas de execução de contratos inteligentes no Ethereum [Poon e Buterin 2017]. Os contrato inteligentes podem ser executados fora da corrente de blocos, e apenas um resumo de todas as execuções é publicado como um estado final na corrente de blocos principal. Li *et al.* apresentam o conceito de correntes satélites (*satellite chains*) [Li et al. 2017]. O objetivo dos autores é criar correntes de blocos que executem de forma independente, mas que possam trocar dados caso necessário. A aplicação prática visa um cenário industrial. O Chainlink é um sistema que permite a troca de informações entre correntes de blocos e informações externas através de contratos inteligentes. [Ellis et al. 2017, Breidenbach et al. 2021].

O Cosmos é um sistema de corrente de blocos que utiliza uma corrente central Cosmos (*Comos hub*) e correntes de blocos paralelas [Kwon e Buchman 2019]. Cada corrente de blocos executa um protocolo BFT baseado no Tendermint [Buchman 2016] para validar de forma ágil os blocos. Além disso, os autores propõem um novo protocolo de comunicação entre correntes de blocos (*Inter-Blockchain Communication* - IBC) para transferência de recursos entre as correntes de blocos secundárias.

A principal desvantagem de realizar modificações nas camadas inferiores é a complexidade apresentada para os usuários finais do sistema de corrente de blocos. As mudanças na camada zero alteram o formato das mensagens trocadas, gerando *soft forks* ou até mesmo *hard forks*. Para um usuário final isso pode significar a incompatibilidade com alguns nós da rede e impedir que este realize transações desejadas com clientes em versões diferentes. Por outro lado, a mudança na Camada 1 é mais crítica, pois altera a camada de consenso e dessa forma cria um sistema completamente novo. Portanto, a escalabilidade dos sistemas de correntes de blocos é preferencialmente obtida por mudanças na Camada 2, como as redes de canais de pagamento, pois estão dissociadas da corrente de blocos e geram um baixo impacto aos usuários finais.

1.3. Redes de Canais de Pagamento

A tecnologia de canais de pagamentos é uma solução para o problema de escalabilidade de pagamentos em correntes de blocos [Hearn 2013]. Ao contrário das soluções mencionadas anteriormente, essa tecnologia é de Camada 2, estabelecendo um canal de comunicação fora-da-corrente (*off-chain*) em que usuários podem transacionar livremente evitando os lentos protocolos de consenso. Outras soluções em Camada 2 existem, como os canais de estados (*state channels*) [Dziembowski et al. 2018], mas não são o foco deste capí. A remoção da necessidade de um mecanismo de consenso no processamento de transações aumenta consideravelmente a vazão transacional dos canais de pagamentos, que passa a depender somente da latência de comunicação entre os canais e não mais do estabelecimento de acordo entre múltiplos participantes.

Uma das principais vantagens dos canais de pagamentos é a facilidade de implementação. Enquanto parte das soluções de Camada 1, como *sharding* e protocolos de

consenso mais eficientes, requer mudanças significativas no núcleo da corrente de blocos, os canais de pagamento podem ser facilmente implementados com contratos inteligentes disponibilizados pela corrente de blocos. Além disso, outra grande vantagem dessa tecnologia em relação à corrente de blocos é a baixa tarifa de transação. Em correntes de blocos públicas, a baixa vazão transacional leva usuários a pagarem altas tarifas para mineradores, buscando uma priorização no processamento da própria transação. Canais de pagamentos dispensam tarifas, uma vez que as transações são enviadas diretamente ao destinatário, sendo cobradas somente nas redes de canais de pagamento, discutidas na Seção 1.3.1.

A principal ideia dos canais de pagamentos é minimizar as transações que passam pelo protocolo de consenso. Apesar da remoção da necessidade de um protocolo de consenso, os canais de pagamento ainda exigem uma corrente de blocos para garantia de segurança. Isso acontece porque a corrente de blocos é utilizada como ponto inicial dos canais de pagamento e eventuais disputas envolvendo usuários. A Figura 1.9 mostra o processo de abertura e fechamento de um canal de pagamento. Para estabelecer um canal, dois usuários da corrente de blocos emitem uma transação de financiamento (*funding transaction*), acordando em reservar parte de suas moedas em um endereço comum. Essas moedas ficam indisponíveis na corrente de blocos durante toda a duração do canal, mas podem ser utilizadas para emitir transações no canal. As transações no canal estabelecido são chamadas de transação de compromisso (*commitment transaction*) e atualizam o saldo de cada usuário no canal. Para fechar um canal de pagamento, os dois usuários entram em acordo no resultado final do canal, i.e., a quantidade de moedas que cada usuário possui após todas as transações feitas pelo canal. Para isso, a última transação de compromisso é emitida publicamente na corrente de blocos. Assim, somente a transação de abertura de canal e a de fechamento de canal são publicadas na corrente de blocos. Essas transações portanto são as únicas que passam pelo protocolo de consenso e estão sujeitas às altas tarifas cobradas por mineradores.

Os canais de pagamento, assim como as correntes de blocos, não requerem confiança mútua em seu modelo de segurança. Para estabelecer o canal de pagamento, um dos usuários emite uma transação com política de pagamento *2-of-2 multisig*, i.e., cada transação que utilize os fundos alocados neste endereço deve conter a assinatura dos dois usuários. Para emitir uma transação dentro do canal, um usuário cria uma transação que gaste moedas do fundo alocado, assina e transfere a transação assinada para a outra parte. Esta transação assinada funciona como uma garantia de pagamento à outra parte, que pode assinar e emitir a transação na corrente de blocos para resgatar seus fundos. Assim, torna-se inviável que uma das partes emita transações a si mesmo para se beneficiar e roubar moedas do canal, uma vez que é necessário o acordo entre os dois usuários.

Prender moedas em uma transação que requer a assinatura dos dois participantes, no entanto, pode apresentar vulnerabilidades de segurança. Supondo um canal formado pelos usuários A e B. O usuário B pode agir maliciosamente e se recusar a emitir transações e também se recusar a assinar todas as transações feitas por A. Dessa maneira, o usuário A fica com as moedas presas eternamente no canal de pagamento e indisponíveis para serem usadas na corrente de blocos. Para resolver essa possível vulnerabilidade, os usuários A e B devem gerar uma transação de reembolso (*refund*), que garante o reembolso dos usuários em caso de comportamento malicioso de uma das contrapartes. Essa

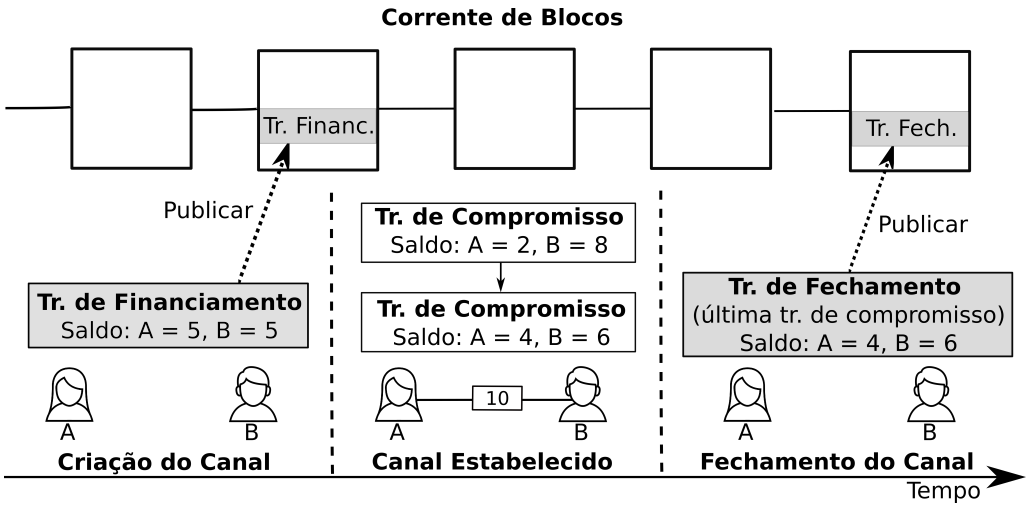

Figura 1.9. Abertura e fechamento de um canal de pagamento. Na figura, os usuários A e B contribuem com 5 moedas cada e emitem a transação de financiamento para criar um canal de pagamentos. O valor 10 no retângulo entre os dois usuários mostra a capacidade máxima do canal, que representa a quantidade máxima de moedas que pode ser transacionada no canal. Após a criação do canal, A e B podem efetuar pagamentos sem emitir transações na corrente de blocos. Assim, A emite um pagamento de 3 moedas e, logo depois, B realiza um pagamento de 2 moedas. Para fechar o canal, basta que um dos usuários emita a última transação na corrente de blocos com o saldo mais atualizado.

transação é criada antes do estabelecimento do canal por motivo de segurança e trocada pelos usuários. Dessa forma, o usuário B possui uma transação de reembolso assinada pelo usuário A e o usuário A possui uma transação de reembolso assinada pelo usuário B. Em caso de ação maliciosa por uma das partes, basta a contraparte emitir a transação de reembolso na corrente de blocos e tomar posse das moedas novamente. A transação de reembolso serve somente como uma garantia e deve deixar de ser válida após a primeira transação no canal de pagamento. Para esta invalidação, faz-se necessário uma maneira de atualizar o estado do canal, invalidando o anterior.

Uma parte fundamental da segurança em canais de pagamentos é a atualização do estado do canal [Gudgeon et al. 2020, Poon e Dryja 2016]. Um usuário malicioso pode se aproveitar e emitir uma transação anterior ao estado atual para se beneficiar. No exemplo da Figura 1.9, os usuários efetuam duas transações fora-da-corrente: a primeira transação TX_1 é feita pelo usuário A, enviando 3 moedas ao usuário B; a segunda transação TX_2 é feita pelo usuário B, enviando 2 moedas ao usuário A. Neste caso, o usuário B pode agir maliciosamente e fechar o canal emitindo a primeira transação feita no canal, uma vez que B teria um saldo maior, 8 moedas, do que o estado mais atual do canal, em que possui 6 moedas. Além disso, A poderia agir de maneira maliciosa e emitir a transação de reembolso na corrente de blocos, resgatando as 5 moedas que possuía inicialmente. Esta vulnerabilidade ocorre porque os canais de pagamento, ao remover a corrente de blocos, perdem o poder de ordenação das transações. Enquanto as criptomoedas utilizam a corrente de blocos para ordenação de transações, os canais de pagamento removem essa possibilidade ao levar os pagamentos para fora-da-corrente. Os canais de pagamento, no entanto, dispensam uma ordenação completa por estampas de tempo como as correntes

de blocos: basta saber qual estado é o mais atual. No caso do exemplo, basta descobrir que a transação TX_1 foi criada antes de TX_2, i.e., deve existir uma maneira de revogar a transação TX_1 e considerar somente o estado mais atual, representado pela TX_2.

Atualizações de estado. Diversas propostas foram criadas para resolver o problema de substituição de estados nos canais de pagamentos [Spilman 2013, Decker e Wattenhofer 2015, Poon e Dryja 2016]. Um dos primeiros mecanismos utilizados nos microcanais (*microchannels*) do Bitcoinj [Bitcoinj.org 2022] foi proposto por Spilman [Spilman 2013] em 2013 e funciona somente para pagamentos unidirecionais. Esse mecanismo impõe condições para o resgate do dinheiro utilizando o mecanismo de programação de transações disponibilizados pela rede do Bitcoin, o *Bitcoin script* [BitcoinWiki 2021]. No mecanismo proposto, antes de estabelecer um canal, os usuários A e B geram uma transação de reembolso que contém as seguintes condições para fechar o canal e resgatar as moedas presas: i) as moedas presas podem ser utilizadas após um tempo t do registro da transação de financiamento ou ii) as moedas podem ser utilizadas imediatamente se a contraparte concordar com o reembolso. A primeira condição garante que as moedas dos usuários não ficarão presas eternamente no canal, enquanto a segunda condição garante que, caso haja um acordo estabelecido por uma transação assinada pelos dois participantes, os dois usuários podem ser ressarcidos. Após estabelecer o canal, o usuário A passa a enviar moedas ao emitir transações de compromisso ao usuário B, que pode assinar as transações e publicá-las na corrente de blocos, fechando o canal, ou esperar uma nova mudança de estado. É fácil perceber que esse desenho de canal funciona somente como um canal unidirecional. Enquanto o usuário B possui transações com assinaturas de A, o mesmo não acontece na direção contrária. Além disso, mesmo que o usuário B devolva uma moeda ao usuário A, não há garantias de segurança que B não emitirá uma transação na corrente de blocos que o beneficie, ou seja, revele uma transação antiga em que possua um saldo maior. Dessa maneira, a substituição do estado é feita por incentivo, uma vez que, nesse modelo de canal unidirecional, o usuário que recebe o dinheiro não possui vantagens em postar uma transação antiga, em que possui menos moedas que a atual. Qualquer usuário racional que seja o destinatário dos pagamentos, sempre postará o estado mais atual, uma vez que este é o estado que o beneficia [Gudgeon et al. 2020].

A criação de um canal de pagamentos que funcione somente em uma direção, no entanto, restringe o potencial de aplicações dessa tecnologia. Grande parte das aplicações do dia-a-dia requerem pagamentos nas duas direções, p. ex., reembolso de compras e aplicações que fornecem *cashback*. Além disso, dois usuários que possuem um canal podem assumir papéis independentes, comprando e vendendo produtos entre si. Neste caso, canais unidirecionais eliminam a possibilidade de aproveitamento do canal já criado entre os dois usuários para pagamentos na direção contrária. Os usuários envolvidos interessados em inverter os papéis de remetente e destinatário de pagamentos teriam que criar mais um canal, o que implicaria em mais moedas indisponíveis na corrente de blocos. Vale ressaltar que Wang *et al.* mostram que 86% das transações feitas na Rede Ripple, uma rede de corrente de blocos não-permissionada, são recorrentes em um período de 24h, i.e., envolvem os mesmos pares de usuários [Wang et al. 2019b].

A criação de canais bidirecionais passa pela revogação do estado: para garantir que um usuário não aja de maneira maliciosa, é necessário revogar o estado ante-

rior. Entretanto, as correntes de blocos não permitem a criação de transações revogáveis [Poon e Dryja 2016]. Uma vez assinada e divulgada na rede, a transação não pode ser cancelada. É possível, no entanto, criar políticas que desincentivem usuários a emitirem transações com estados antigos e agirem maliciosamente [Decker e Wattenhofer 2015, Poon e Dryja 2016]. Uma das formas inicias de substituição de estados foi a substituição por bloqueio de tempo. Neste modelo, todas as transações no canal possuem um bloqueio de tempo, o que significa que as moedas envolvidas na transação só podem ser utilizadas após um intervalo de tempo t contado a partir da publicação da transação na corrente de blocos. Para garantia de sincronia do tempo t, este modelo utiliza a corrente de blocos como referência, sendo o tempo definido em número de blocos da corrente. O tempo t é decrementado toda vez que a direção do pagamento for invertida. Assim, em um canal entre dois usuários A e B, o usuário A pode emitir uma transação TX_1 a B com bloqueio de tempo de 30 minutos, aproximadamente 3 blocos na Rede Bitcoin. Caso B queira efetuar um pagamento a A, o usuário B emite TX_2 com bloqueio de tempo de 20 minutos, aproximadamente 2 blocos na Rede Bitcoin. Dessa maneira, se B agir maliciosamente e emitir a transação anterior, TX_1, na corrente de blocos, o usuário A pode emitir TX_2 e resgatar as moedas antes de B, uma vez que TX_2 possui menor bloqueio de tempo. Apesar de garantir a substituição de estados, esse modelo apresenta duas grandes desvantagens: (i) o usuário A deve estar disponível e constantemente verificando a corrente de blocos para monitorar B e (ii) o decremento de t possui o limite inferior de zero. O valor inicial de t, definido pelos dois usuários que compõem o canal, é uma escolha difícil. Um alto valor para t permite mais atualizações de estados, mas pode bloquear as moedas por um longo período de tempo no fechamento. Um baixo valor de t permite poucas atualizações, determinando um prazo de validade ao canal.

Poon e Dryja propuseram o modelo de substituição de estados utilizado pela Rede Relâmpago, a maior rede de canais de pagamentos atualmente [Poon e Dryja 2016]. O modelo gera canais bidirecionais desencorajando o comportamento malicioso de participantes com uma punição financeira: caso fique comprovado que um dos participante agiu maliciosamente e publicou um estado antigo na corrente de blocos, a contraparte pode contestar a transação e resgatar todo o dinheiro alocado no canal para si. Neste modelo, todo pagamento feito no canal gera um par de transações de compromisso assimétricas: o usuário A possui uma transação de compromisso com a assinatura do usuário B e o usuário B possui uma transação de compromisso com a assinatura do usuário A. Cada transação é associada a um segredo e apresenta a seguinte condição: se a contraparte souber o segredo associado a essa transação e o revelar dentro de um período t a partir da publicação da transação, ela pode pegar para si todo o dinheiro alocado no canal. Todo o processo é feito utilizando mecanismos da corrente de blocos como base, sendo público aos dois participantes. A verificação da chave revelada é feita de maneira automática através do *script*, mecanismo de programação de transações da Rede Bitcoin. Para efetuar um pagamento no canal, os usuários acordam em um novo estado trocando assinaturas e revelam o segredo da transação associada ao estado anterior. Assim, o usuário que publicar a transação de compromisso, fechando o canal, só pode resgatar o dinheiro após o tempo t, período em que a contraparte pode contestar o estado publicado na corrente de blocos.

O procedimento para substituição de estados adotado pela Rede Relâmpago funciona da seguinte maneira. Considerando um canal entre Alice e Bob em que, inicial-

mente, cada usuário possui 5 moedas que foram alocadas no canal. Alice deseja enviar 1 moeda para Bob através do canal. Para isso, Alice gera um par de chaves assimétricas (PK_A, SK_A), em que PK_A representa a chave pública de Alice e SK_A representa a chave secreta gerada por Alice. Bob executa o mesmo procedimento, gerando um par de chaves (PK_B, SK_B). Alice, então, cria uma transação de compromisso TX_{AB} contendo as seguintes informações: i) Alice possui 4 moedas que pode usar imediatamente se a transação for publicada; ii) Bob possui 6 moedas, que só podem ser usadas após um período t ou caso Alice autorize antes do período t acabar; iii) a chave pública dessa transação é PK_B. Se Alice revelar SK_B em um tempo $t_i, t_i < t$, Alice pode resgatar as outras 6 moedas. Alice assina a transação TX_{AB} e a envia para Bob. Bob executa o mesmo procedimento, gerando a transação TX_{BA}, que contém as seguintes informações: i) Bob possui 6 moedas que pode usar imediatamente se a transação for publicada; ii) Alice possui 4 moedas, que só podem ser usadas após um período t ou caso Bob autorize antes do período t acabar; iii) a chave pública dessa transação é PK_A. Se Bob revelar SK_A em um tempo $t_i, t_i < t$, Bob pode resgatar as outras 4 moedas. Bob assina a transação TX_{BA}, a envia para Alice e revela a chave secreta da transação referente ao estado anterior. Assim, nesse modelo, a parte que fecha o canal deve aguardar o período de disputa para utilizar as moedas.

1.3.1. Contratos de Bloqueio por Tempo e por Hash (*Hashed Timelock Contracts* - HTLC) e Redes de Canais de Pagamento

Apesar de permitir o pagamento rápido e com baixas taxas, a tecnologia de canais de pagamento atende somente a um par de usuários. Essa solução sozinha não resolve o problema de escalabilidade das correntes de blocos como um todo [Poon e Dryja 2016]. Um usuário que queira transacionar com múltiplos usuários deve: i) possuir uma alta quantia, que ficará indisponível na corrente de blocos, para alocar em múltiplos canais e ii) pagar altas taxas em cada transação de financiamento para abrir os múltiplos canais. Estes requisitos inviabilizam a utilização desta tecnologia no cotidiano, em que pagamentos a diversas entidades diferentes são comuns.

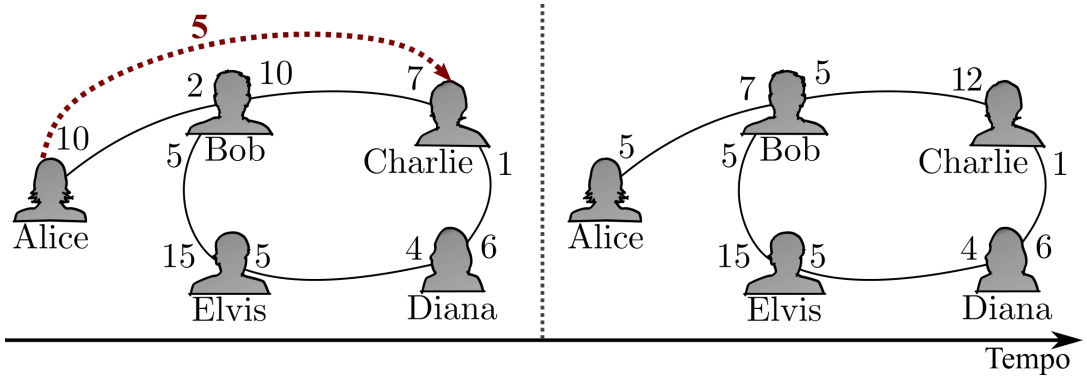

Figura 1.10. Exemplo de pagamento ocorrendo em uma rede de canais de pagamento. Para enviar um pagamento de 5 moedas a Charlie, Alice utiliza o canal pré-estabelecido com Bob, que as repassa ao destino. O pagamento modifica as capacidades dos canais envolvidos em seu caminho. A segurança desse procedimento é garantida por contratos de bloqueio por tempo e por *hash* (*Hashed Timelock Contracts* - HTLC).

O problema de escalabilidade pode ser resolvido com uma rede de canais de pa-

gamentos (*Payment Channel Network* - PCN), que interconecta os canais criados pelos usuários na corrente de blocos. As PCNs permitem que usuários possam transacionar entre si de maneira rápida e com baixas taxas, mesmo que não possuam um canal direto. A Figura 1.10 mostra um exemplo de uma PCN com 5 participantes. Na figura, Alice quer transferir 5 moedas para Charlie, com quem não compartilha um canal de pagamento. Entretanto, Alice possui um canal aberto com Bob, que possui um canal aberto com Charlie. As redes de canais de pagamento aproveitam os canais já existentes entre os usuários para encaminhar pagamentos por eles. Assim, Alice pode transferir 5 moedas em seu canal com Bob. Bob, então, transfere 5 moedas em seu canal com Charlie, completando o pagamento de Alice. Vale ressaltar que os canais de pagamento são independentes, o que significa que Bob não pode transferir as moedas de seu canal com Alice para o canal com Charlie. Bob recebe a transação de Alice em seu canal com Alice e gera uma transação de igual valor para Charlie.

As PCNs, assim como as correntes de blocos, devem dispensar confiança mútua entre os participantes. É possível perceber que, no exemplo da Figura 1.10, Bob pode agir maliciosamente recebendo as moedas de Alice sem repassar o pagamento em seu canal com Charlie. Para evitar tal ação, um tipo especial de contrato chamado contrato de bloqueio por tempo e por *hash* (*Hashed Timelock Contract* - HTLC) garante que as moedas em um canal serão transferidas apenas se duas condições específicas forem cumpridas [Poon e Dryja 2016, Bitcoin Wiki 2021]. A primeira condição é que Bob deve revelar o segredo de um *hash* para se apossar da moeda de Alice, e a segunda é que isso deve acontecer dentro de um limite de tempo. Como exemplo, Alice quer mandar dinheiro para Charlie, utilizando Bob como intermediário. Usando os contratos bloqueados por *hash* e tempo, a ideia é que Alice só entrega o dinheiro a Bob depois que ela souber que Bob repassou o dinheiro a Charlie: a prova de que Bob passou o dinheiro a Charlie é o "segredo" que Charlie gerou. Charlie só entrega o segredo a Bob depois que recebe o dinheiro (Bob cumpriu com o prometido). Em seguida, Bob pode repassar o "segredo" a Alice, provando que ele cumpriu com o prometido, e recebendo o dinheiro de Alice. O "segredo" gerado por Charlie funciona como se fosse um "recibo digital". Especificamente, cada HTLC possui um valor $y \mid H(x) = y$, em que $H(x)$ é o resultado da função resumo de um segredo x gerado pelo destinatário, e um tempo t. Caso o usuário não revele o valor x antes de t, o HTLC é invalidado e o valor retorna ao criador do HTLC. Cada usuário intermediário cria um HTLC com o próximo salto mantendo o mesmo valor y na condição e a consistência em toda a cadeia de pagamento. O destinatário possui conhecimento do valor x e o revela ao receber um HTLC, desbloqueando a cadeia de pagamentos.

No exemplo da Figura 1.11, Alice quer enviar uma moeda para Charlie. Charlie, destinatário do pagamento, gera um valor x aleatório, calcula o resultado da função resumo $H(x) = y$, e envia y para Alice. Alice gera um HTLC com Bob identificando Charlie como próximo salto e promete entregar uma moeda a Bob, sob condição da revelação do valor x por parte de Bob. O HTLC mostra Charlie como próximo salto de Bob. Bob, então, gera um HTLC com a mesma condição para Charlie. Charlie, que inicialmente gerou o valor de x, revela x para Bob para resgatar o seu pagamento. Bob, por sua vez, revela x a Alice para resgatar o pagamento no canal entre Alice e Bob, concluindo o pagamento. Devido às duas condições que devem ser cumpridas, os HTLCs são ditos bloqueados por

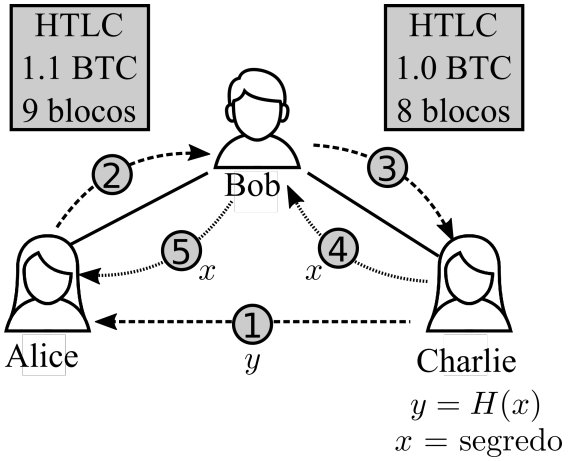

Figura 1.11. Etapas para efetuar um pagamento em uma rede de canais de pagamentos. 1) Charlie, o destinatário, gera um segredo e envia o *hash* deste segredo a Alice. 2) Alice não possui canal com Charlie e utiliza Bob como intermediário de pagamento, estabelecendo um contrato prometendo entregar dinheiro a Bob, caso Bob revele o segredo gerado por Charlie. 3) Bob executa o mesmo procedimento com Charlie. 4) Charlie revela o segredo para receber o pagamento, permitindo que Bob receba o pagamento em seu canal com Alice (5).

tempo e por *hash*. Apesar de serem formalmente chamados de contratos, o mecanismo de funcionamento dos HTLCs segue uma lógica simples que pode ser implementada mesmo em sistemas que não suportam contratos inteligentes, como o Bitcoin. Isto aumenta o alcance das redes de canais de pagamento, que podem ter sua funcionalidade estendida a praticamente qualquer corrente de blocos.

Cada intermediário no caminho de um pagamento cobra uma tarifa, que funciona como um incentivo pelo encaminhamento de pagamentos, como mostra a Figura 1.11. A quantidade máxima de moedas que um usuário pode encaminhar é limitada pela quantidade de moedas que o usuário possui no canal em que o HTLC vai ser estabelecido. No exemplo da Figura 1.10, Bob não consegue encaminhar pagamentos maiores que 10 moedas no canal que possui com Charlie. Além disso, cada intermediário na cadeia deve determinar um tempo de expiração menor que o definido no HTLC do salto anterior. Assim, enquanto Alice gera um HTLC com tempo de expiração t, Bob deve gerar um HTLC com Charlie com tempo de expiração $t_i \mid t_i < t$. Essa diferença no tempo de expiração garante que todos os intermediários da cadeia possuam tempo suficiente para resgatar o dinheiro.

1.3.2. A Rede Relâmpago (*Lightning Network*)

A Rede Relâmpago (*Lightning Network*), proposta por Poon e Dryja para a criptomoeda Bitcoin em 2016, é a primeira implementação da tecnologia de redes de canais de pagamento [Poon e Dryja 2016]. O sistema realizou sua primeira transação em 2017 e já atingiu a marca de mais de 14.000 nós e 86.000 canais de pagamento distribuídos pelo mundo em agosto de 2021, consolidando-se como a maior rede de canais de pagamento conhecida. A Figura 1.12 mostra a evolução do sistema. O resultado da figura mostra um conjunto de dados de trocas de mensagens de fofoca para sincronização dos

nós da topologia da Rede Lightning [Decker 2021]. O número de nós e canais triplicou entre janeiro de 2020 e agosto de 2021, demonstrando um crescimento exponencial da rede. Assim, a Rede Relâmpago é hoje a referência de implementação de canais de pagamento existente, provendo pagamentos digitais em tempo real em diversas aplicações [Lightning Network Developers 2022]. A Rede Relâmpago é inclusive mencionada como um dos motivos para a adoção do Bitcoin como forma de pagamento oficial em El Salvador [CloudTweaks 2021].

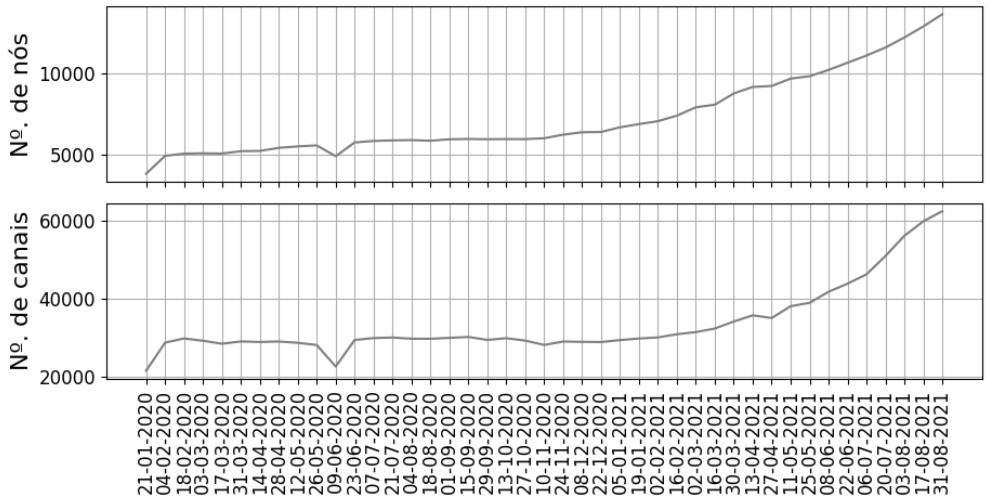

Figura 1.12. Crescimento do número de nós e de canais na Rede Relâmpago no período entre janeiro de 2020 e agosto de 2021.

O principal foco da Rede Relâmpago, assim como a criptomoeda na qual é baseada, é garantir o anonimato de seus usuários. Assim, a rede provê diversos mecanismos de privacidade, como identificação de usuários através apenas de chaves públicas, como no Bitcoin. O encaminhamento de pagamentos utiliza o roteamento em camadas (*onion routing*), conhecido por sua implementação na rede Tor [Dingledine et al. 2004], que criptografa o pagamento a cada salto. Isto garante que nenhum intermediário que encaminhe o pagamento conheça o caminho completo. Os intermediários cobram dois tipos de tarifas para encaminhar pagamentos: uma tarifa base e uma tarifa proporcional [Múltiplos Autores 2022c]. As tarifas base são um valor fixo cobrado a cada pagamento encaminhado pelo usuário, independentemente do valor do pagamento sendo roteado. As tarifas proporcionais são variáveis e cobradas em cima do valor a ser encaminhado. Apesar de apresentar dois tipos de tarifas, as tarifas da Rede Relâmpago são ordens de grandeza menores que as tarifas cobradas na corrente de blocos do Bitcoin, o que a torna especialmente atrativa para micro-pagamentos em tempo real [Zabka et al. 2021].

A rede implementa canais de pagamento exatamente da forma descrita anteriormente neste capí. No entanto, usuários podem optar por divulgar seus canais publicamente para que sejam utilizados como rotas de pagamentos, ou mantê-los privados para garantir maior privacidade. A divulgação de canais na rede ocorre através de mensagens de anúncio de canal (*channel announcement*) que a rede difunde no modelo de fofoca (*gossip*) [Múltiplos Autores 2022b]. Neste modelo, um participante divulga uma mensagem a um número específico de vizinhos, que repetem a mensagem aos próprios vizinhos

até que o canal seja conhecido por toda a rede. Os nós da rede constroem, a partir das mensagens recebidas, a topologia da rede contendo os canais ativos com seus respectivos participantes e capacidades. Canais privados não são divulgados e, portanto, não aparecem na topologia. Dessa forma, o número de canais conhecidos por um nó é um limite inferior para a real quantidade de canais existentes na rede, que é difícil de estimar. Os balanços dos canais, privados ou públicos, sempre são conhecidos apenas pelas duas partes diretamente envolvidas, enquanto a capacidade do canal pode ser divulgada ou não. Isto evita que observadores externos rastreiem pagamentos ao monitorar balanços de canais, o que comprometeria a privacidade dos usuários. Por outro lado, o fato de o balanço não ser conhecido dificulta a escolha de caminhos com capacidade suficiente para a realização de um pagamento específico

A Rede Relâmpago padroniza os formatos de mensagens e protocolos utilizados na rede através das "bases da tecnologia Relâmpago" (*Basis of Lightning Technology* - BOLT), documentos inspirados nas RFCs da Internet que descrevem formalmente como a rede deve ser implementada [Múltiplos Autores 2022a]. Atualmente, a Rede Relâmpago possui 11 BOLTs, que especificam formato e troca de mensagens para abertura e fechamento de canais, codificação de pagamentos, protocolo de roteamento e outras especificações. Este capítulo descreve as partes mais relevantes dos BOLTs na seção prática com objetivo de facilitar a compreensão do funcionamento do simulador utilizado.

Por ser a principal PCN, diversos trabalhos analisam a Rede Relâmpago [Seres et al. 2020, Lin et al. 2020, Rohrer et al. 2019]. O trabalho de Seres *et al.* analisa a topologia da Rede Relâmpago em janeiro de 2019. Os resultados dos autores mostram a forte centralização de conectividade da Rede Relâmpago, dado que poucos nós concentram grande parte dos canais da rede. Os autores verificam a robustez da rede, simulando ataques direcionados aos nós da rede que apresentam maior grau. Os resultados indicam que a remoção dos 37 nós mais de maior grau reduzem a capacidade da rede em 50%. De maneira similar, Lin *et al.* avaliam a concentração de renda na Rede Relâmpago no período entre janeiro de 2018 e julho de 2019 [Lin et al. 2020]. A concentração de renda de um nó da rede pode ser medida verificando os valores públicos de capacidade dos canais que este nó possui. Os autores verificam uma tendência de centralização na rede em torno dos nós de maior grau, formando estruturas do tipo núcleo-periferias em que o núcleo é formado por *hubs* e a periferia apresenta subestruturas do tipo estrela. Os resultados mostram que a remoção dos *hubs* particionam a rede em múltiplos componentes, tornando-a vulnerável a ataques [Rohrer et al. 2019].

1.3.3. Desafios em Rede de Canais de Pagamentos

Os principais desafios em redes de canais de pagamentos são relacionados ao roteamento de pagamentos [Sivaraman et al. 2018, Sivaraman et al. 2020, Wang et al. 2019b], rebalanceamento de canais e garantias de segurança e privacidade [Kappos et al. 2021, Malavolta et al. 2017].

Roteamento de pagamentos. Apesar de as redes de canais de pagamento serem consideradas redes par-a-par, o funcionamento dos canais de pagamento introduz características únicas que se refletem em desafios para o roteamento de pagamentos. A principal peculiaridade dos canais de pagamentos é que transferir moedas de uma parte para a outra de

um mesmo canal significa reduzir a capacidade do canal para transferências futuras na mesma direção, uma vez que cada encaminhamento está realocando o saldo da parte que envia para a parte que recebe. Portanto, a capacidade de encaminhamento de um canal depende diretamente de quantos pagamentos já foram enviados. Esta particularidade é a principal diferença do roteamento em redes de canais de pagamento em comparação a redes de computadores tradicionais, nas quais encaminhar pacotes reduz a capacidade de um enlace apenas durante o período em que os pacotes estão em trânsito. Por exemplo, em uma rede tradicional, ainda que um enlace de 1 Gb/s esteja encaminhando pacotes a 200 Mb/s e sua capacidade disponível seja temporariamente reduzida a aproximadamente 800 Mb/s, a capacidade retorna ao valor original assim que o encaminhamento terminar. Em uma rede de canais de pagamento, se um canal com capacidade de 1.000 moedas encaminhar 200 moedas numa direção específica, sua capacidade naquela direção se reduz permanentemente a 800 moedas. A única forma de retornar à capacidade original é receber pagamentos de 200 moedas na direção oposta. Essa característica evidencia a necessidade de equilíbrio entre os pagamentos em cada direção e dificulta a utilização de abordagens de fluxo máximo, muito presentes nas redes tradicionais.

Outra característica fundamental de redes de canais de pagamento é que os saldos de cada canal são privados, i.e., apenas as duas partes envolvidas diretamente no canal conhecem seu estado atual. A capacidade total do canal, por outro lado, é pública e está disponível na corrente de blocos. Isso gera uma desafio a ser considerado pelos algoritmos de roteamento: estimar o estado atual dos canais a partir de sua capacidade total. Se isso não for feito corretamente, o algoritmo de escolha de caminhos pode decidir por utilizar canais que possuíam saldo suficiente no momento de sua abertura, mas já foram exauridos por outros pagamentos. Nesse caso, o pagamento falha e deve ser refeito, o que aumenta a latência para o usuário e reduz a eficiência do sistema. Em protocolos que dividem o pagamento em múltiplos caminhos, a falha de parte de um pagamento também compromete a atomicidade do pagamento. Isto pode gerar situações nas quais o comprador paga corretamente n moedas por um produto, mas o vendedor recebe apenas $n - x$ moedas, onde x é a soma de valores dos pagamentos que falharam.

O procedimento padrão para descoberta de caminhos de pagamentos na Rede Relâmpago utiliza um protocolo de tentativa e erro baseado no algoritmo de Dijkstra para escolha do caminho atômico mais curto. O usuário executa o algoritmo de Dijkstra para encontrar o caminho mais curto até o destinatário do pagamento e, em seguida, tenta efetuar o pagamento neste caminho. Caso o pagamento falhe em algum canal porque um intermediário não possui saldo suficiente para encaminhar o pagamento, o protocolo remove o canal em que o pagamento falhou do grafo que o usuário possui e executa o algoritmo de Dijkstra novamente. A Rede Relâmpago utiliza como métrica padrão de caminho mais curto a soma das tarifas a serem pagas pelo usuário para cada canal, i.e., o algoritmo seleciona o caminho em que o usuário paga menos tarifas. Esse procedimento de escolha de caminhos, no entanto, é ineficiente, uma vez que desconsidera que pagamentos podem esgotar a capacidade dos canais em uma direção. O uso desse protocolo causa a necessidade constante de balanceamento dos canais com tarifas mais baixas, que são muito utilizados por possuírem menor custo. O balanceamento dos canais é geralmente realizado através do aumento da própria tarifa na direção do desbalanceamento ou da diminuição da tarifa na direção oposta para incentivar pagamentos no sentido inverso.

Ademais, o envio do pagamento por inteiro sobre um único caminho dificulta a transferência de pagamentos de alto valor. Essa dificuldade decorre do baixo número de canais que possuem alta quantidade de moedas alocadas [Seres et al. 2020].

O pagamento atômico multicaminhos (*Atomic Multipath Payments* - AMP) é um protocolo proposto por Osuntokun em 2018 para a Rede Relâmpago que tem sido adotado para aumentar a probabilidade de sucesso de uma transação [Osuntokun 2018]. Ao contrário do protocolo padrão, que utiliza apenas um caminho para transferência monetária, o AMP utiliza múltiplos caminhos para efetuar um pagamento. Dessa maneira, usuários conseguem enviar pequenas quantias de moedas por canais menores, que somam o valor completo do pagamento desejado. No exemplo da Figura 1.10, o valor máximo de pagamentos que Alice consegue transferir para Charlie utilizando o modelo padrão de roteamento da Rede Relâmpago é limitado pelo canal de menor capacidade, que possui 10 moedas. Portanto, supondo que Alice queira enviar 12 moedas para Charlie, não há caminho que suporte o pagamento. Entretanto, com o AMP, o pagamento pode ser dividido em duas partes com valores diferentes: Alice pode transferir 10 moedas pelo caminho Bob $\rightarrow$ Charlie e 2 moedas pelo caminho Bob $\rightarrow$ Elvis $\rightarrow$ Diana $\rightarrow$ Charlie. Assim, o AMP aumenta as chances de conseguir efetuar transferências de alto valor com sucesso ao dividir o pagamento em pequenas quantias que podem ser transmitidas por diversos canais. O AMP, no entanto, aumenta consideravelmente a quantidade de HTLCs na rede, uma vez que cada pequena quantia enviada deve gerar um novo HTLC. Este aumento no número de HTLCs também aumenta consideravelmente as tarifas pagas pelo usuário, já que cada HTLC independente gera uma tarifa base a ser paga. Além disso, a falha de um dos pagamentos ou caminhos implica em falha no pagamento completo, gastando recursos da rede desnecessariamente. Apesar dessas desvantagens, o AMP ainda é amplamente utilizado em propostas por apresentar uma melhora na taxa de sucesso de pagamentos em relação ao modelo padrão da Rede Relâmpago.

O Spider [Sivaraman et al. 2020, Sivaraman et al. 2018] é um protocolo de roteamento em PCNs proposto por Sivaraman *et al.* que, assim como o AMP, divide transações em pequenos pedaços para envio em múltiplos caminhos. No entanto, enquanto a quantidade de partes é programável no AMP, o Spider divide sempre a transação em pequenas quantias com o objetivo de maximizar a probabilidade de sucesso do pagamento. A principal novidade do Spider é a proposta de um controle de congestionamento de pagamentos nos múltiplos caminhos que visa manter os canais equilibrados. No Spider, os nós intermediários do caminho possuem filas que podem ser usadas para manter as transações em caso de falta de fundos para transferência e aguardar pagamentos na direção contrária. Os intermediários marcam transações que permanecem na fila por um determinado tempo e notificam os remetentes dos pagamentos, que passam a priorizar outros caminhos. Nesse sentido, o Spider busca mandar pagamentos em um caminho a uma taxa de envio igual à taxa na direção contrária. Essa característica busca minimizar os desbalanceamentos no caminho, uma vez que cada pagamento em uma direção é rapidamente reposto por outro na direção contrária, prevenindo o esgotamento de canais e diminuindo a necessidade de reposição de fundos no canal pela corrente de blocos. Entretanto, assim como no AMP, o Spider também apresenta um alto número de HTLCs se comparado com o padrão utilizado pela Rede Relâmpago devido à divisão do pagamento em múltiplos pedaços que são enviados de maneira independente. Além disso, o Spider desconsidera as tarifas de

encaminhamento no caminho, o que pode resultar em altas tarifas ao usuário remetente, além de não garantir atomicidade do pagamento.

Uma proposta adaptativa de roteamento é o protocolo Flash, que possui um algoritmo de roteamento que diferencia o roteamento de pagamentos de alto valor de pagamentos de baixo valor [Wang et al. 2019b]. Os autores argumentam que pagamentos de alto valor causam mais impacto na taxa de sucesso de transações da rede, uma vez que este tipo de transação está mais propenso a falhar por falta de canais com fundos disponíveis. Assim, o Flash desenvolve um algoritmo de roteamento complexo para pagamentos de alto valor baseado em uma versão modificada do algoritmo de maximização de fluxos, que considera o ambiente dinâmico das PCNs [Edmonds e Karp 1972]. Por outro lado, pagamentos de baixo valor, que são mais comuns [Wang et al. 2019b], utilizam roteamento simples baseado em tentativa e erro. Primeiro, o usuário tenta enviar o pagamento completo por um caminho aleatório. Caso o pagamento falhe por algum canal não possuir capacidade, o usuário envia parte do pagamento com valor igual à máxima capacidade encontrada no caminho e busca outro caminho para completar o pagamento. Os autores utilizam essa abordagem para minimizar a sobrecarga com a busca de caminhos e quebra de transações. Essa abordagem, no entanto, facilita o esgotamento de canais, uma vez que o máximo valor possível é enviado por um único caminho, o que pode prejudicar o encaminhamento de futuras transações no canal em uma direção.

Malavolta *et al.* propõem o SilentWhispers, um protocolo de roteamento em PCNs que contém garantias de privacidade e busca reduzir a sobrecarga gerada pelo armazenamento da topologia completa da rede em cada usuário [Malavolta et al. 2016]. O SilentWhispers implementa um roteamento baseado em marcos (*landmark routing*), em que nós dedicados, chamados de marcos, armazenam a topologia da rede e agem como intermediários para pagamentos. Assim, cada usuário precisa manter somente os caminhos até os nós dedicados. Caso queira efetuar uma transferência monetária, um usuário envia a quantia aos nós marcos, que os repassam até o destinatário. As principais críticas a esse tipo de roteamento são a centralização nos marcos, que podem agir maliciosamente, a utilização frequente dos mesmos canais, que pode levar ao colapso da rede e a concentração de renda nos marcos [Roos et al. 2017]. O uso de marcos também pode gerar caminhos mais longos do que o necessário. Por outro, a arquitetura hierárquica é mais adaptada a cenários com dispositivos limitados em recursos que não são capazes de calcular rotas e manter uma topologia completa da rede. Esse tipo de roteamento, apesar de apresentar uma solução com maior potencial de inclusão de dispositivos leves, ainda carece de observações práticas e testes em larga escala.

Pickhardt *et al.* propõem o único protocolo de roteamento conhecido que busca estimar os saldos dos canais de pagamento para minimizar as incertezas [Pickhardt e Richter 2021]. A estimativa inicial prevê que metade da capacidade total do canal está alocada para cada parte envolvida. Com essa estimativa, o protocolo seleciona caminhos cujos canais possuam a maior capacidade possível e tenta enviar os pagamentos por eles. Caso o pagamento falhe em algum canal devido à falta de saldo, o algoritmo sabe que o saldo daquele canal é menor que o valor do pagamento e atualiza sua topologia para refletir essa nova informação. O mesmo ocorre quando o pagamento é bem sucedido, pois existe a certeza de que o saldo do canal diminuiu do valor do pagamento. Através dessas atualizações, o protocolo consegue coletar informações sobre o estado da

rede enquanto realiza pagamentos, o que serve como controle de congestionamento para pagamentos futuros. O protocolo também introduz uma métrica mista que permite considerar as tarifas proporcionais cobradas no custo de utilização dos canais. O usuário pode decidir, através de um parâmetro regulador, dar maior peso às capacidades dos canais, maximizando a chance de sucesso do pagamento, ou às tarifas proporcionais, minimizando os custos financeiros. As principais críticas a esse protocolo são a alta latência devido à complexidade de atualização da topologia e seleção de caminhos, e a desconsideração das tarifas base no custo do canal. O protocolo também ainda não foi amplamente comparado com os demais protocolos implementados na Rede Relâmpago.

Tabela 1.1. Comparação multiquesito entre as principais propostas de algoritmos de roteamento em redes de canais de pagamento. As marcas de seleção indicam que a característica está presente na proposta, enquanto os traços indicam que a proposta não apresenta a característica correspondente.

Propostas	Rede Relâmpago (padrão)	Rede Relâmpago (AMP)	Spider	Flash	Silent Whispers	Pickhardt *et al.*
Roteamento pela Fonte	✓	✓	✓	✓	-	✓
Visão Global da Topologia	✓	✓	✓	✓	-	✓
Controle de Congestionamento	-	-	✓	-	-	✓
Múltiplos Caminhos	-	✓	✓	✓	✓	✓
Consideração de Tarifas	✓	✓	-	✓	-	✓
Garantias de Privacidade	-	-	-	-	✓	-
Filas de Transações	-	-	✓	-	-	-
Atomicidade de Pagamentos	✓	✓	-	-	-	-

A Tabela 1.1 resume os principais pontos das propostas apresentadas para os desafios de roteamento em PCN e as compara com a implementação da Rede Relâmpago. Na tabela, a linha visão global da topologia indica se o usuário deve conhecer a topologia completa da rede para efetuar um pagamento. As propostas baseadas em roteamento pela origem (*source routing*), como o modelo de roteamento da Rede Relâmpago, Spider, Flash e Pickhardt *et al.* delegam aos usuários remetentes de pagamentos a tarefa de encontrar caminhos até o destinatário. Assim, esse método de roteamento requer que os usuários conheçam toda a topologia da rede. O SilentWhispers, por outro lado, requer que o usuário saiba somente o caminho até um dos marcos para efetuar um pagamento, demandando menos recursos de armazenamento. Com exceção ao modelo adotado pela Rede Relâmpago, os protocolos apresentados utilizam múltiplos caminhos para envio de pagamentos, buscando aumentar a taxa de sucesso das transações ao permitir que pagamentos de qualquer valor sejam divididos e enviados por canais de baixa capacidade. Entretanto, apenas o modelo adotado pela Rede Relâmpago, o Flash e Pickhardt *et al.* buscam minimizar as tarifas pagas pelos usuários em seu modelo de roteamento. A atomicidade dos pagamentos ocorre exclusivamente com o uso do protocolo AMP, que atualmente só possui implementação para o protocolo da Rede Relâmpago. No entanto, o AMP pode teoricamente ser usado com qualquer protocolo de roteamento não-atômico [Pickhardt e Nowostawski 2020].

Rebalanceamento. O fato de o saldo de um canal influenciar diretamente na sua capacidade de encaminhar pagamentos causa uma preocupação constante com o equilíbrio dos canais na rede. E, visto que a maior parte das aplicações possui uma tendência bem

definida para o fluxo de pagamentos, p. ex. de compradores para vendedores, os canais de pagamento devem ser constantemente rebalanceados para se manterem ativos. Dessa forma, um desafio em redes de canais de pagamento é propor mecanismos de rebalanceamento eficientes, que preservem a vida útil dos canais.

A forma mais simples e mais utilizada de rebalanceamento é o incentivo de pagamentos através da variação das tarifas de encaminhamento cobradas. Nesse método, também chamado na literatura de rebalanceamento *passivo*, os intermediários aumentam as suas tarifas em um determinado canal toda vez que detectarem que este canal está com pouco saldo [Conoscenti et al. 2019]. A ideia é que, com o aumento da tarifa, menos usuários escolherão o canal em questão como caminho e a tendência é que o canal se equilibre conforme pagamentos chegam no sentido oposto. Em geral, os intermediários realizam o ajuste de tarifas manualmente de acordo com suas vontades e necessidades. Porém, hoje já existem ferramentas que automatizam o processo [Bosworth 2021, Otto 2022]. Essa abordagem funciona melhor na Rede Relâmpago, pois seu algoritmo padrão considera as tarifas como métrica principal para escolha dos caminhos. No entanto, é incerto que esta estratégia seja eficiente com algoritmos que consideram outros parâmetros.

Dentre os mecanismos *ativos* de rebalanceamento, Khalil e Gervais propõem o Revive, um algoritmo de rebalanceamento que aproveita ciclos na topologia da rede para rebalancear canais, diminuindo a necessidade de recorrer à corrente de blocos [Khalil e Gervais 2017]. No Revive, um líder eleito recebe requisições de rebalanceamento de múltiplos usuários e calcula um conjunto de transações que devem ser efetuadas. Este conjunto de transações busca atender aos requisitos dos usuários e deve garantir que usuários não percam dinheiro no processo. Dessa forma, o algoritmo proposto desloca moedas entre canais, respeitando as preferências de rebalanceamento fornecidas pelos usuários e conservando o crédito alocado por cada usuário na rede. O algoritmo, no entanto, fere a privacidade dos usuários, uma vez que, para calcular o conjunto de transações de rebalanceamento, o líder deve conhecer o saldo dos canais envolvidos.

Pickhardt e Nowostawski introduzem uma métrica de desbalanceamento global na rede e um método de rebalanceamento que minimiza o desbalanceamento local entre canais de um mesmo usuário [Pickhardt e Nowostawski 2020]. Nesta proposta, o usuário calcula constantemente a diferença de saldo entre todos os canais nos quais está envolvido, com o objetivo de mantê-los balanceados entre si. Se existe uma disparidade muito grande entre dois canais, o usuário realiza um pagamento cíclico para si mesmo partindo do canal de maior saldo para o canal de menor saldo, reequilibrando a distribuição de saldos. Assim, a ideia é que o usuário seja sempre capaz de rotear pagamentos igualmente em qualquer direção. O trabalho mostra que esta heurística local também contribui para a melhora do balanceamento global da rede caso todos os usuários a utilizem. No entanto, é difícil garantir isso em um sistema descentralizado no qual cada usuário age de forma independente. Além disso, os pagamentos de rebalanceamento custam tarifas ao usuário e podem comprometer o equilíbrio de canais que fazem parte do ciclo.

A Rede Relâmpago apresenta uma combinação entre os métodos ativos e passivos, bem como a recriação de canais através de transações na corrente de blocos. Os roteadores centrais da rede geralmente regulam seus saldos passivamente através das tarifas, pois recebem pagamentos constantemente de todos os lados. Os usuários menos centrais ado-

tam o método cíclico caso possuam canais com capacidade suficiente, ou simplesmente recriam o canal reservando mais moedas através de uma transação direto na corrente de blocos. Ainda não existe uma solução que garanta o equilíbrio da rede de forma sistemática.

Segurança e privacidade. As redes de canais de pagamento também apresentam desafios de segurança e privacidade. Malavolta *et al.* identificam três requisitos de segurança e privacidade em PCNs [Malavolta et al. 2017]:

- **Segurança de Saldo (*balance security*)**: Uma PCN deve garantir que um usuário intermediário, ao encaminhar um pagamento, não perca dinheiro mesmo que todos os outros nós do caminho sejam maliciosos ou corrompidos.
- **Privacidade de Valor Fora-do-caminho**: A PCN deve garantir que um usuário que esteja fora do caminho de um pagamento não possa descobrir o valor do pagamento sendo roteado em um caminho com usuários honestos.
- **Anonimidade de Relacionamento**: Um usuário intermediário no caminho de um pagamento desconhece outros nós do caminho, exceto o nó antecessor e sucessor na cadeia de pagamentos.

Além destes requisitos, Kappos *et al.* adicionam dois à lista, baseados na Rede Relâmpago [Kappos et al. 2021]:

- **Privacidade de canal**: Dois usuários, caso desejem, podem manter a existência do canal que compartilham sob segredo, sem revelar quaisquer informações sobre o canal, como capacidade e tarifas, a terceiros. A Rede Relâmpago permite que usuários mantenham o canal que compartilham sob segredo ao dispensar a obrigatoriedade de anunciar canais na rede. Neste caso, o canal dos usuários é desconsiderado para rotas de transações, uma vez que outros usuários desconhecem sua existência, e os participantes do canal não recebem tarifas de pagamento.
- **Segredo de saldo**: Apesar das capacidades dos canais serem públicas, o saldo que cada usuário possui dentro do canal, deve permanecer privado. A Rede Relâmpago permite que usuários anunciem seus canais, conforme definido pelo BOLT#2 [Múltiplos Autores 2022b]. Usuários podem revelar a capacidade do canal nos anúncios difundidos pela rede, mas o padrão de mensagens impede que os usuário revelem os saldos do canal. Além disso, questões de desempenho também protegem a propriedade de segredo de saldo. Devido à alta dinamicidade da rede provocada por múltiplos pagamentos concomitantes, a notificação a cada alteração de saldo inundaria a rede de mensagens de atualização.

Apesar de garantir os dois requisitos de privacidade e segurança listados acima, a Rede Relâmpago apresenta algumas vulnerabilidades de segurança. Malavolta *et al.* demonstram o ataque de buraco de minhoca [Malavolta et al. 2018]. Nesse tipo de ataque, um atacante no caminho de um pagamento entra em conluio ou controla um outro nó no caminho do pagamento. Apesar do caminho do pagamento ser confidencial, o atacante

pode facilmente perceber se outro nó, sob seu controle, está no caminho do pagamento, uma vez que os HTLCs na Rede Relâmpago possuem o mesmo resumo em todo o caminho. Assim, ao receber dois HTLCs em pontos diferentes com a mesma função resumo, o atacante infere que o pagamento está no mesmo caminho.

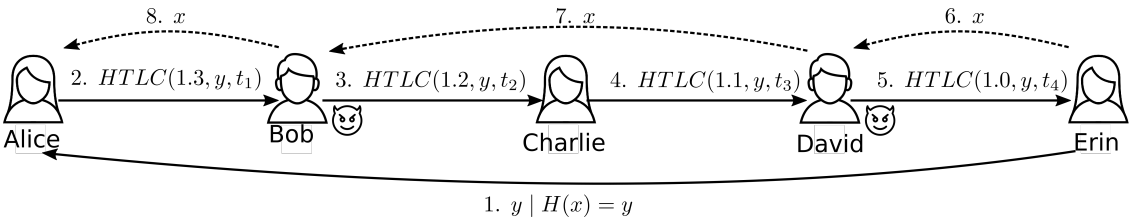

Figura 1.13. Exemplo de ataque de buraco de minhoca. Bob e David agem em conluio para roubar tarifas que seriam destinadas a Charlie. As linhas sólidas compreendem as etapas de estabelecimento de um pagamento, enquanto as linhas pontilhadas compreendem o resgate do pagamento.

A Figura 1.13 ilustra o ataque de buraco de minhoca. Na Figura, Alice deseja enviar uma moeda a Erin e cada intermediário cobra 0,1 moeda como tarifa de encaminhamento. Bob e David entram em conluio para efetuar o ataque. Na primeira parte, os HTLCs são estabelecidos no caminho do pagamento de maneira legítima com resumo bloqueado $y \mid H(x) = y$. Para desbloquear a cadeia de pagamentos, o destinatário revela o valor de x a David, que o repassa diretamente para Bob, sem resgatar o pagamento no canal que possui com Charlie. Bob, então, resgata o pagamento em seu canal com Alice. Do ponto de vista de Charlie, que não recebe o valor x, o HTLC falhou. O ataque de buraco de minhoca permite que o atacante receba tarifas que seriam destinadas a nós intermediários. No exemplo da figura, cada intermediário receberia 0,1 de tarifa se Bob e David agissem de maneira legítima. Ao efetuar o ataque de buraco de minhoca, no entanto, Bob recebe 0,4 de tarifa, que pode ser dividida com David. Apesar de não perder dinheiro, o ataque de buraco de minhoca prejudica Charlie, que não recebe as tarifas do pagamento e deve manter as moedas indisponíveis até o tempo de expiração do HTLC.

O ataque de buraco de minhoca é possível porque a anonimidade de relacionamento não é garantida. Todos os HTLCs do caminho apresentam o mesmo bloqueio por resumo. Dessa forma, um atacante que controla dois nós no caminho de um pagamento pode facilmente notar que o HTLC se trata do mesmo pagamento. Para resolver esse problema, Malavolta *et al.* propõem um novo tipo de contrato chamado HTLC multissalto [Malavolta et al. 2017]. Nesta construção, cada intermediário recebe dois valores de resumo y_i e y_{i+1}, sendo $y_i = H(x_i)$ e $y_{i+1} = H(x_i \oplus x_{i+1})$, em que $x_i \oplus x_{i+1}$ representa a operação lógica "ou-exclusivo" entre x_i e x_{i+1}. Além destes valores, os intermediários também recebem o valor x_{i+1} e uma prova que $\exists x_i \mid y_i = H(x_i)$ e $y_{i+1} = H(x_i \oplus x_{i+1})$. Essa prova utiliza técnicas de prova de conhecimento nulo (*Zero Knowledge Proof* - ZKP), que permite que o remetente prove uma afirmação sem revelar informações secretas pertinentes à afirmação [Goldwasser et al. 1985]. Dessa forma, assim como nos HTLCs, a revelação de x_i é suficiente para desbloquear toda a cadeia de pagamentos, uma vez que x_i é a única informação omitida aos intermediários. Essa construção garante a anonimidade de relacionamento, uma vez que cada intermediário possui um resumo diferente, dificultando a associação entre pagamentos na mesma cadeia.

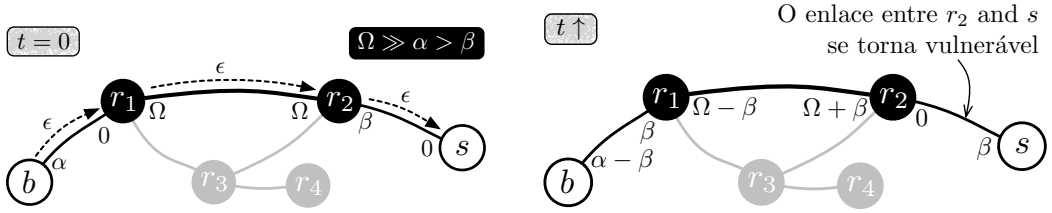

Figura 1.14. Um exemplo da vulnerabilidade de roubo de moedas em redes de canais de pagamento. À esquerda, uma quantidade contínua de ε moedas flui de uma origem b para um destino s até que a capacidade do canal entre o roteador r_2 e s se esgote. Então, à direta, s se torna altamente vulnerável se perder a conexão antes de fechar o canal porque r_2 não tem nada a perder se encerrar o canal com um balanço anterior.

Roubo de moedas. Em qualquer rede de canal de pagamentos, é possível roubar moedas caso uma das partes que constitui o canal se desconecte por um período excessivamente longo. Por isso, PCNs como a Rede Relâmpago [Poon e Dryja 2016] e a Rede Raiden [brainbot labs Est. 2020] assumem que qualquer nó que transaciona na rede permanece em linha (*online*) enquanto o canal está aberto. Caso contrário, uma das partes do canal pode encerrá-lo publicando uma transação antiga, que invalida moedas enviadas e efetivamente as recupera para a parte que enviou. O sistema pune este tipo de comportamento malicioso permitindo que a vítima gaste todas os moedas do canal, incluído as da parte maliciosa, caso se recupere durante uma janela de tempo de bloqueio predefinida. Portanto, só vale a pena tentar o ataque se o nó malicioso puder garantir que a outra parte não verificará a corrente de blocos durante o período de disputa, que permanece até a expiração da janela de tempo.

Em redes com conexões rápidas e confiáveis nas quais todos os usuários possuem uma cópia da corrente de blocos, um pequeno valor padronizado para janelas de tempo de bloqueio é suficiente para permitir que vítimas se recuperem e punam seus atacantes a tempo. Nesses casos, os usuários podem detectar o comportamento malicioso instantaneamente, sem confiar em terceiros, simplesmente sincronizando suas correntes de blocos e verificando os blocos mais recentes. No entanto, em cenários heterogêneos, com muitos usuários e principalmente com dispositivos móveis, alguns nós podem se desconectar por longos períodos ou mesmo indefinidamente. O tempo de inatividade do dispositivo é especialmente desafiador para casos de uso em que a direção dos pagamentos é tendenciosa para um dos lados, como quando um vendedor usa seu dispositivo para receber transações de vários compradores. Nesse caso, espera-se que os canais de pagamento sejam altamente desequilibrados em relação a uma das partes.

O desequilíbrio dos canais indica uma vulnerabilidade ainda maior ao problema de roubo de moedas, como demonstrado na formulação a seguir. Sejam dois dispositivos b e s, que representam dispositivos de um comprador e um vendedor, respectivamente, e que estão conectados aos roteadores r_1 e r_2 por meio de canais de pagamento conforme mostrado na Figura 1.14. Cada canal de pagamento $u \leftrightarrow v$ tem um balanço $bal_{u\leftrightarrow v}(t) = (bal_u(t), bal_v(t))$, onde $bal_u(t)$ e $bal_v(t)$ são os saldos dos nós u e v no tempo t, respectivamente. Observe que $bal_u(t) + bal_v(t)$ é constante. Para canais

de pagamento entre compradores e roteadores, por exemplo $b \leftrightarrow r_1$, o balanço inicial é $bal_{b \leftrightarrow r_1}(0) = (\alpha, 0)$, onde α é uma quantidade de moedas que o comprador b reserva para pagamentos no canal. Da mesma forma, o balanço inicial dos canais de pagamento entre vendedores e roteadores, por exemplo $r_2 \leftrightarrow s$ é $bal_{r_2 \leftrightarrow s}(0) = (\beta, 0)$ onde β é a quantidade de moedas que o roteador r_2 reserva para encaminhar pagamentos ao vendedor s. A formulação assume por simplicidade e sem perda de generalidade que s e b só participam de um canal de pagamento.

Considerando um cenário em que um pagamento de ε moedas ocorre de b para s, r_2 e s assinam uma transação de compromisso $Tx(1)$ contendo o novo balanço do canal $bal_{r_2 \leftrightarrow s}(1) = (\beta - \varepsilon, \varepsilon)$. Se s se desconectar indefinidamente após a assinatura, r_2 pode fechar o canal com a transação anterior $Tx(0)$ e recuperar ε moedas. Fazer isso é arriscado porque r_2 perderia todas as suas moedas se s se recuperar e detectar o comportamento malicioso antes do fim do período de disputa. No entanto, à medida que s recebe mais pagamentos, o balanço em $r_2 \leftrightarrow s$ converge para $bal_{r_2 \leftrightarrow s}(t) = (0, \beta)$. Se isso acontecer, r_2 tem pouco a perder fechando o canal com uma transação anterior, mesmo que s se recupere a tempo. Essa é a estratégia ótima para qualquer roteador racional r quando seu canal de pagamento para um vendedor se esgotar. Os nós maliciosos também podem decidir atacar em casos intermediários dependendo da relação risco-benefício. Portanto, os mecanismos tradicionais de segurança das redes de canais de pagamento não evitam que roteadores adotem essa estratégia. O vendedor s está sujeito a roubo de moedas mesmo na ausência de comportamento malicioso. Embora esteja formulado para um caso extremo de compradores e vendedores, o problema se aplica a qualquer situação em que um nó recebe pagamentos e se desconecta sem fechar o canal corretamente.

O problema de roubo de moedas torna-se ainda mais expressivo quando os nós da rede podem ser dispositivos móveis com conectividade intermitente e desconexões por longos períodos de tempo. Alguns trabalhos propõem melhorias como janelas de tempo adaptadas ao perfil de conectividade de dispositivos, contratação de nós "vigilantes" que constantemente verificam a corrente de blocos para detectar canais que foram fechados indevidamente, ou sistemas de reputação nos quais os nós puniriam o comportamento malicioso emitindo opiniões sobre roteadores [Rebello et al. 2021a, ION Lightning Network Wiki 2021]. No entanto, essas soluções ainda não são profundamente exploradas, possuem problemas de privacidade e levam à centralização no sistema [Camilo et al. 2020, Khojasteh e Tabatabaei 2021]. As principais implementações de redes de canais de pagamento não possuem uma solução eficaz para esses casos atualmente [Lin et al. 2020, brainbot labs Est. 2020].

Lind *et. al* propõem o Teechain, que permite o acesso assíncrono de usuários à corrente de blocos [Lind et al. 2019]. Para isso, os autores movem a raiz de confiança (*Root of Trust* - RoT) da corrente de blocos para ambientes de execução confiáveis (*Trusted Execution Environments* - TEE), garantindo que os nós ajam de forma honesta. Para implementar esses ambientes confiáveis, a arquitetura utiliza instruções especiais da CPU oferecidas pela tecnologia *Software Guard Extensions* (SGX) da Intel para criar regiões de memória isoladas até mesmo do sistema operacional, denominadas enclaves. No sistema proposto, a aplicação do cliente mantém depósitos dentro de enclaves e manipula os saldos desses depósitos ao receber e enviar transações pelos canais de pagamento. A aplicação com enclaves apenas interage com a corrente de blocos na criação e na finalização

de um depósito. Antes de criar um depósito, o usuário deve utilizar um mecanismo de atestação dos processadores da Intel para garantir que a aplicação é executada em um ambiente confiável genuíno, e que irá seguir o protocolo honestamente. Ademais, o sistema replica os dados entre os enclaves de dispositivos que participam de um comitê, evitando assim que uma vulnerabilidade no ambiente de execução confiável não acarrete em um ponto único de falha.

1.4. Atividade Prática (*Hands-On*): Efetuando Pagamentos em uma Rede de Canais de Pagamento

O objetivo da atividade prática deste capítulo é demonstrar o funcionamento padrão de uma rede de canais de pagamentos, mostrando as trocas de mensagens e comunicações entre os participantes envolvidos em um pagamento. A atividade consiste em explorar o comportamento de uma rede de canais de pagamento através de três cargas de trabalho de transações: i) um conjunto sintético de transações de baixo valor, que simulam pequenos pagamentos de até 20 reais na rede, ii) um conjunto sintético de transações de alto valor, que simulam grandes pagamentos de mais de 1.000 reais, e iii) um conjunto real de transações, que contém pagamentos de cartão de crédito realizados por consumidores ao longo de um mês. Para o desenvolvimento da atividade, este capítulo utiliza o PCNsim, simulador de rede de canais de pagamentos desenvolvido pelo Grupo de Teleinformática e Automação (GTA) em conjunto com o laboratório LIP6 [Rebello et al. 2022].

A atividade prática requer um simples computador com sistema operacional baseado em Linux e acesso à Internet. Apesar dos comandos da atividade seguirem o padrão da distribuição Ubuntu, a atividade pode ser efetuada em qualquer distribuição Linux, bastando substituir os comandos necessários por um comando equivalente na distribuição utilizada. Todos os programas necessários para a realização desta atividade prática são gratuitos. O PCNsim, principal componente da atividade, está disponível no sítio `https://github.com/gfrebello/pcnsim`. Os pré-requisitos para a execução da simulação podem ser facilmente encontrados na documentação do PCNsim em `https://pcnsim.readthedocs.io`.

1.4.1. PCNsim: Simulador Flexível e Modular de Redes de Canais de Pagamento

O PCNsim é um simulador de código-aberto, modular e flexível de redes de canais de pagamentos que reproduz o comportamento da Rede Relâmpago e permite a geração de diversos cenários em PCN para testes. A Figura 1.15 apresenta a arquitetura do PCNsim, que é composto pelo gerador de topologia, o gerador de carga, o núcleo do simulador, o visualizador de resultados e o armazenamento de resultados.

Os módulos de geração de topologia e carga de trabalho permitem que os usuários testem propostas e cenários específicos de topologia e carga de transações em uma PCN. O módulo de topologia do PCNsim permite a criação de topologias seguindo um modelo de rede-sem-escalas e de mundo pequeno, uma vez que trabalhos anteriores mostram que a Rede Relâmpago se comporta seguindo esses dois modelos [Seres et al. 2020, Rohrer et al. 2019]. Além disso, o gerador de topologia permite a modelagem de canais seguindo parâmetros reais da Rede Relâmpago, como capacidade de canais e taxas cobradas. O gerador da carga de trabalho define o conjunto de transações

que será usado durante a simulação. Ao contrário das transações na corrente de blocos, as transações na Rede Relâmpago não são publicamente disponibilizadas por questões de privacidade. Assim, para implementação de um cenário mais próximo ao real, o PCNsim permite que os usuários modelem valores de transações seguindo um conjunto de dados de pagamentos de cartão de crédito e de um *e-commerce*.

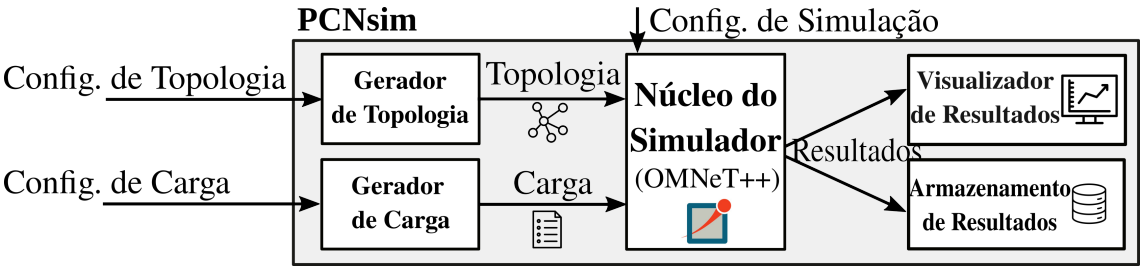

Figura 1.15. Arquitetura do PCNsim. Usuários podem facilmente configurar a topologia e carga de transações da simulação através dos geradores de topologia e de carga, além de visualizar os resultados da simulação.

O núcleo de simulação é composto pelo OMNET++ [OMNeT++ 2022] e simula o comportamento de uma rede de canais de pagamentos. O PCNsim segue os padrões definidos pelo BOLT#2 [Múltiplos Autores 2022b]. Dessa maneira, o PCNsim imita corretamente o comportamento padrão de uma implementação real de uma PCN, entregando uma simulação confiável aos usuários. Além disso, o núcleo de simulação também coleta estatísticas sobre canais, permitindo fácil visualização para os usuários. Por fim, ao contrário de implementações de PCN que exigem uma corrente de blocos para resolução de disputas e abertura de canais, o simulador foca somente na rede de canais de pagamento, retirando a necessidade de uma camada de corrente de blocos e tornando o armazenamento do simulador mais leve.

Por fim, os módulos visualizador e armazenamento de resultados disponibilizam as estatísticas coletadas durante a simulação para facilitar a análise de resultados ao usuário final. O PCNsim disponibiliza resultados como a taxa de sucesso de transações e evolução da capacidade dos canais por padrão, que podem ser visualizados através do módulo visualizador de resultados. Ademais, o módulo de armazenamento de resultados armazena de forma persistente as estatísticas em disco para análise futura dos usuários.

Para participar da atividade prática, é necessário instalar o PCNsim e preparar o ambiente de simulação. Assim, o usuário deve executar os seguintes passos:

1. Verificar se o computador possui os requisitos listados em `https://pcnsim.readthedocs.io/en/latest/prerequisites.html`.

2. Clonar o repositório do PCNsim utilizando a ferramenta Git: `git clone https://github.com/gfrebello/pcnsim`.

3. Dentro do repositório, executar `python3 -m pip install -r requirements.txt` para instalar as bibliotecas em Python necessárias para execução do simulador.

4. Executar o comando `export PCNSIM_DIR=$PWD` para criar uma variável de ambiente com o diretório do simulador.

5. Baixar o conjunto de dados de transações de cartão de crédito no Kaggle, disponível em `https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud` e mover o arquivo `csv` para o diretório adequado: `mv creditcard.csv $PCNSIM_DIR/scripts/datasets`.

Ao final desses passos, o ambiente de simulação está pronto para seguir a atividade prática descrita.

1.4.2. Gerando Topologias e Cargas no PCNsim

Antes de simular um cenário de uma rede de canais de pagamentos no PCNsim, é necessário gerar uma topologia de rede e o conjunto de transações utilizados pelo simulador. Iniciando pela criação da topologia da rede, o usuário deve executar, no diretório `scripts` o seguinte comando:

```
python3 generate_topology_workload.py genTopo
--lightning -n 10
```

O comando `genTopo` acima gera um arquivo no diretório `topology` que é utilizado pela simulação para gerar os nós da rede e as conexões entre eles. A opção `--lightning` gera os canais utilizando valores aleatórios de um conjunto de dados da Rede Relâmpago para modelar os parâmetros dos canais, como capacidade do canal e taxa cobrada. Dessa maneira, os usuários podem testar sua proposta em um cenário próximo ao de uma implementação real de PCN. A opção `-n 10` determina o número de nós da rede igual a 10 durante a simulação. Como o comando não especifica uma topologia, o PCNsim modela a rede gerada seguindo o modelo de uma rede sem escalas.

Após gerar a topologia utilizada durante a simulação, é necessário gerar a carga de trabalho. Vale ressaltar que a topologia deve ser gerada antes da carga de trabalho, uma vez que é necessário conhecer o conjunto de nós da rede para determinar os nós que efetuam pagamentos. Esta atividade prática compreende três cenários de carga de trabalho com: i) pagamentos de até 20 reais; ii) pagamentos de mais de 1.000 reais e iii) pagamentos com valores baseados em um conjunto de dados de cartão de crédito. Os comandos referentes aos cenários são, respectivamente:

1. `python3 generate_topology_workload.py genWork --n_payments 5 --max_payment 20`

2. `python3 generate_topology_workload.py genWork --n_payments 5 --max_payment 1000`

3. `python3 generate_topology_workload.py genWork --n_payments 5 --credit_card`

Nos comandos acima, a opção `--n_payments` define o número de pagamentos na simulação, `--max_payments` define o valor máximo do pagamento e

`--credit_card` modela o valor dos pagamentos de acordo com um conjunto de dados de transações de cartão de crédito. Vale ressaltar que os comandos são exclusivos, i.e., a carga de trabalho utilizada pelo simulador é baseada em somente um comando. O comando para o cenário escolhido gera um arquivo no diretório `workloads` que é utilizado pelo simulador para definir as transações entre os participantes. O PCNsim escolhe aleatoriamente os nós de origem e destino de pagamentos.

1.4.3. Efetuando e Visualizando Pagamentos

O PCNsim utiliza a ferramenta OMENT++ para simular o modelo de uma PCN. Assim, após gerar a topologia e o conjunto de transações utilizado pela simulação, é necessário inicializar a ferramenta utilizando o comando `omnetpp`. Para executar o PCNsim no OMNET++, o usuário deve importar o projeto do diretório raiz do PCNsim em `File > Import > General > Existing Projects in Workspace`. Ao selecionar o diretório `simulator` e clicar em "Executar", o usuário executa a simulação com o cenário gerado na etapa anterior.

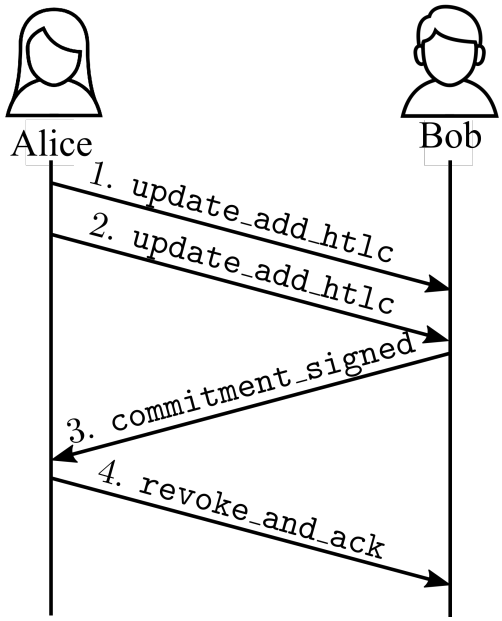

Figura 1.16. Troca de mensagens para o estabelecimento de um contrato bloqueado por tempo e por *hash*. Alice oferece um ou múltiplos contratos de pagamentos a Bob, que pode aceitá-los ou recusá-los. Bob atualiza os contratos aceitos localmente e envia uma mensagem assinada a Alice contendo os contratos aceitos. Alice recebe a mensagem, atualiza localmente os contratos efetivados por Bob e envia um reconhecimento à mensagem de Bob.

Através da interface do OMNET++, é possível perceber as trocas de mensagens envolvidas em um pagamento, além das atualizações nos balanços dos canais quando um pagamento é encaminhado. A primeira mensagem envolvida em um pagamento é a fatura (*invoice*). O destinatário do pagamento envia esta mensagem ao remetente, contendo o resumo que é usado no HTLC e o valor do pagamento. Ao receber a fatura, o remetente executa o algoritmo de Dijkstra para encontrar caminhos para o destinatário e oferece um HTLC com o primeiro salto do caminho utilizando a mensagem `update_add_htlc`,

definida pelo BOLT#2 [Múltiplos Autores 2022b]. Caso aceite o HTLC oferecido, o primeiro salto pode efetuar as atualizações localmente, reservando a quantia para o cumprimento do HTLC, e enviar uma mensagem assinada de `commitment_signed` ao remetente do pagamento atualizando o estado. Esta mensagem contém todos os HTLCs oferecidos no canal e aceitos pelo primeiro salto. Apesar de aceitar os HTLCs, o primeiro salto ainda deve aguardar a confirmação de revogação do estado anterior pelo remetente do pagamento para considerar o HTLC como efetivo. Assim, o usuário remetente verifica a lista de HTLCs aceitos pelo primeiro salto, atualiza localmente e envia uma mensagem de `revoke_and_ack`, revogando o estado que possuía anteriormente e enviando um reconhecimento (*acknowledgement*) à mensagem de `commitment_signed`. Esse processo é repetido com os outros intermediários que compartilham um canal até a formação da cadeia de pagamento fim-a-fim. Caso o pagamento falhe em algum canal, uma mensagem de `update_fail_htlc` é gerada e enviada por todo o caminho de volta até o remetente. A cada salto, os usuários devem atualizar o estado novamente, removendo o HTLC previamente acordado e recuperando as moedas alocadas. A Figura 1.16 ilustra a troca de mensagens entre os canais para estabelecimento e confirmação de um HTLC.

Para desbloquear a cadeia de pagamentos, o destinatário revela o valor gerado para o resumo utilizando a mensagem `update_fulfill_htlc`. O valor mantido em segredo e revelado pelo destinatário é chamado de pré-imagem. Ao receber a mensagem de cumprimento do HTLC, os participantes do canal iniciam outra etapa de atualização de estado, o que envolve as mensagens de `commitment_signed` e `revoke_and_ack`, como mostra a Figura 1.16. A mensagem de `update_fulfill_htlc` contendo a pré-imagem percorre todo o caminho do pagamento, do destinatário até o remetente do pagamento. A cada recebimento, o processo de atualização de estados é executado entre os pares que compartilham o canal.

1.5. Conclusão e Perspectivas Futuras

A tecnologia de corrente de blocos revolucionou a Internet ao permitir pagamentos em linha (*online*) sem necessidade de qualquer intermediário. Essa tecnologia, no entanto, apresenta diversos desafios de escalabilidade. Os lentos mecanismos de consenso e as altas tarifas das correntes de blocos impedem o uso cotidiano das criptomoedas como forma de pagamento. Enquanto métodos tradicionais confirmam pagamentos em segundos, as criptomoedas podem levar horas para confirmação de um pagamento.

O capítulo apresentou técnicas existentes para solucionar o problema de escalabilidade da corrente de blocos. A otimização da quantidade de informação transmitida pelos nós apresenta um baixo impacto na vazão final. A fragmentação agiliza o processamento das transações, mas cria desafios significativos em relação ao processamento de transações entre fragmentos distintos. Os grafos acíclicos dirigidos possuem uma estrutura diferente da corrente de blocos que apresenta riscos de segurança às aplicações de transferência de ativos, que são reduzidos através da adição de nós confiáveis. Por outro lado, as correntes laterais apresentam concepções similares aos canais de pagamento. Entretanto, a redução de participantes no consenso impacta significativamente a segurança das correntes de blocos secundárias, favorecendo o sucesso de ações maliciosas realizadas por atacantes. Além disso, os protocolos de consenso presentes nas propostas tendem a ser o gargalo da vazão de transações. Portanto, apesar de existirem outras alternativas para

o problema de escalabilidade das correntes de blocos, as redes de canais de pagamento são as que possuem menor latência e maior segurança associada.

Este capítulo apresentou a tecnologia de canais de pagamentos, que oferece uma solução ao problema de escalabilidade de correntes de blocos públicas. Esta tecnologia busca reduzir o número de transações que passam pelo mecanismo de consenso. Para isso, usuários criam canais de comunicação fora-da-corrente e utilizam a corrente de blocos somente como raiz de segurança, utilizada para iniciar e para eventuais disputas no fechamento dos canais. Dessa maneira, ao retirar a necessidade de um acordo global de transações, os canais de pagamento reduzem a sobrecarga de comunicação e atingem alta vazão de transações e rápida confirmação de pagamentos. Além disso, este capítulo introduziu os conceitos das redes de canais de pagamentos, utilizadas para escalar as soluções de canais de pagamentos a múltiplos usuários de maneira segura. O capítulo discutiu a implementação de contratos bloqueados por tempo e por *hash*, componentes importantes para garantia de segurança em PCNs. Por fim, o capítulo descreveu uma atividade prática, que pode ser seguida por leitores para visualizar na prática as operações de uma PCN, incluindo troca de mensagens e efeito da topologia de rede no sucesso de pagamentos.

Apesar de permitir transações rápidas, seguras e sem intermediários, as redes de canais de pagamentos ainda apresentam desafios em aberto. O modelo de roteamento de pagamentos utilizado pela principal implementação de PCN, a Rede Relâmpago, é ineficiente, provoca exaustão de canais e restringe a possibilidade de pagamentos de alto valor a poucos canais. Diversas propostas em redes de canais de pagamentos buscam resolver os problemas do modelo da Rede Relâmpago. Diferentemente do modelo adotado na Rede Relâmpago, as principais propostas atuais de roteamento utilizam múltiplos caminhos para o roteamento de pagamentos [Sivaraman et al. 2020, Wang et al. 2019b, Pickhardt e Richter 2021, Osuntokun 2018]. Esta abordagem apresenta a vantagem de permitir que pagamentos de alto valor sejam roteados por canais de baixa capacidade ao dividir o pagamento em múltiplos caminhos. Por outro lado, a divisão do pagamento em unidades independentes incorrem em maiores tarifas aos usuários e em um aumento na quantidade de contratos difundidos na rede. Além disso, grande parte das propostas não consideram as tarifas pagas pelos usuários no modelo de roteamento e não apresentam garantias de privacidade. Neste sentido, percebe-se que as propostas de roteamento apresentam um compromisso (*trade-off*): priorizar a eficiência do modelo de roteamento adotado implica em perda de privacidade e vice-versa [Tang et al. 2020]. Isso acontece porque, ao assumir que os balanços dos canais são públicos, modelos de roteamento podem tomar decisões mais conscientes de caminho, aumentando a probabilidade de sucesso das transações. Por outro lado, manter os balanços em segredo aumenta a privacidade, mas restringe a eficiência do modelo de roteamento adotado. Desta maneira, possíveis direções de roteamento devem explorar os limites desta escolha, buscando desenvolver um modelo que seja suficientemente eficiente sem expor informações privadas sobre os canais dos usuários. O mesmo compromisso entre privacidade e eficiência ocorre nas soluções relacionadas ao processo de rebalanceamento de canais. O conhecimento global dos balanços da rede permite a escolha mais eficiente de um conjunto de transações de rebalanceamento. Este conhecimento, no entanto, fere o princípio da segurança de balanço apresentado como requisito de privacidade na Seção 1.3. Por outro lado, o balanceamento de canais que considera apenas uma visão local da rede pode gerar um conjunto

de transações que desbalanceie outros canais no processo.

Na área de segurança e privacidade, as implementações de PCN atuais também apresentam vulnerabilidades. A utilização do mesmo resumo nos HTLCs em todo o caminho de pagamento quebra alguns requisitos de privacidade e permite que um agente malicioso efetue ataques de buraco de minhoca [Malavolta et al. 2017]. Além disso, o requisito de constante disponibilidade dos participantes e do armazenamento de toda a topologia da rede restringe o alcance desta tecnologia. Usuários que utilizam dispositivos móveis ou que possuam conexões intermitentes tornam-se vulneráveis a roubo de moedas [Rebello et al. 2021a]. Para resolver este problema, algumas soluções propõem serviços de vigilantes, que monitoram a corrente de blocos durante a indisponibilidade de um usuário [ION Lightning Network Wiki 2021]. Esses serviços, no entanto, centralizam a confiança no vigilante, que pode efetuar ataques de conluio e passa a ser um ponto único de falha. Outro desafio de segurança enfrentado pela Rede Relâmpago é a alta concentração de renda e conectividade, o que a torna vulnerável a ataques direcionados aos nós de maior grau [Rohrer et al. 2019, Seres et al. 2020]. Lange *et al.* verificam o impacto de políticas de preferência de conexão na Rede Relâmpago do ponto de vista individual e global [Lange et al. 2021]. Os autores verificam que a conexão aos nós mais centrais produz um benefício do ponto de vista individual, recebendo mais tarifas de pagamentos. Entretanto, do ponto de vista global da rede, os modelos de conexão descentralizados trazem mais benefícios de segurança a longo prazo. Os autores ressaltam que uma possível direção de pesquisa para este problema deve propor estratégias mescladas de conexões de novos nós para balancear a contradição dos dois tipos de políticas analisadas.

Referências

[Alvarenga et al. 2021] Alvarenga, I. D., Camilo, G. F., De Souza, L. A. e Duarte, O. C. M. (2021). Dagsec: A hybrid distributed ledger architecture for the secure management of the internet of things. Em *2021 IEEE International Conference on Blockchain (Blockchain)*, páginas 266–271. IEEE.

[Amoussou-Guenou et al. 2019] Amoussou-Guenou, Y., Del Pozzo, A., Potop-Butucaru, M. e Tucci-Piergiovanni, S. (2019). Dissecting tendermint. Em *International Conference on Networked Systems*, páginas 166–182. Springer.

[Ampel et al. 2019] Ampel, B., Patton, M. e Chen, H. (2019). Performance modeling of hyperledger sawtooth blockchain. Em *2019 IEEE International Conference on Intelligence and Security Informatics (ISI)*, páginas 59–61. IEEE.

[Androulaki et al. 2018] Androulaki, E. et al. (2018). Hyperledger Fabric: a distributed operating system for permissioned blockchains. Em *13th EuroSys Conference*, página 30.

[Antonio et al. 2021] Antonio, A. d. A., de Albuquerque, C. V., Loivos, E. B., Gondim, B. T., Vianna, A. A. e Ferreira, A. O. (2021). Segurança e escalabilidade em sharding blockchain. *Sociedade Brasileira de Computação*.

[Back et al. 2014] Back, A., Corallo, M., Dashjr, L., Friedenbach, M., Maxwell, G., Miller, A., Poelstra, A., Timón, J. e Wuille, P. (2014). Enabling Block-

chain Innovations with Pegged Sidechains. *URL: http://www. opensciencereview. com/papers/123/enablingblockchain-innovations-with-pegged-sidechains*, 72.

[Baird e Luykx 2020] Baird, L. e Luykx, A. (2020). The hashgraph protocol: Efficient asynchronous BFT for high-throughput distributed ledgers. Em *2020 International Conference on Omni-layer Intelligent Systems (COINS)*, páginas 1–7.

[Bitcoin Wiki 2021] Bitcoin Wiki (2021). Hash Time Locked Contracts. Disponível em `https://en.bitcoin.it/wiki/Hash_Time_Locked_Contracts`. Acessado em 9 de maio de 2022.

[Bitcoinj.org 2022] Bitcoinj.org (2022). bitcoinj. Disponível em `https://bitcoinj.org/`.

[BitcoinWiki 2021] BitcoinWiki (2021). Script - Bitcoin Wiki. Disponível em `https://en.bitcoin.it/wiki/Script`.

[BitcoinWiki 2022] BitcoinWiki (2022). Bitcoin Scalability. Acessado em 9 de maio de 2022.

[Blockchain.com 2022a] Blockchain.com (2022a). Average Transactions Per Block. Acessado em 9 de maio de 2022.

[Blockchain.com 2022b] Blockchain.com (2022b). Blockchain charts. Acessado em 9 de maio de 2022.

[Blockchain.com 2022c] Blockchain.com (2022c). Blockchain Size. Acessado em 9 de maio de 2022.

[Bosworth 2021] Bosworth, A. (2021). Balance of Satoshis. Disponível em: https://github.com/alexbosworth/balanceofsatoshis. Acessado em 9 de maio de 2022.

[brainbot labs Est. 2020] brainbot labs Est. (2020). The Raiden Network: Fast, cheap, scalable token transfers for Ethereum. Disponível em: https://raiden.network/. Acessado em 9 de maio de 2022.

[Breidenbach et al. 2021] Breidenbach, L. et al. (2021). Chainlink 2.0: Next Steps in the Evolution of Decentralized Oracle Networks. Acessado em 9 de maio de 2022.

[Buchman 2016] Buchman, E. (2016). *Tendermint: Byzantine Fault Tolerance in the Age of Blockchains*. PhD thesis, University of Guelph.

[Bugday et al. 2019] Bugday, A., Ozsoy, A. e Sever, H. (2019). Securing Blockchain Shards by Using Learning Based Reputation and Verifiable Random Functions. Em *2019 International Symposium on Networks, Computers and Communications (ISNCC)*, páginas 1–4. IEEE.

[Camilo et al. 2020] Camilo, G. F., Rebello, G. A. F., de Souza, L. A. C. e Duarte, O. C. M. (2020). A Secure Personal-Data Trading System Based on Blockchain, Trust, and Reputation. Em *2020 IEEE International Conference on Blockchain*, páginas 379–384.

[Castro e Liskov 1999] Castro, M. e Liskov, B. (1999). Practical byzantine fault tolerance. Em *Proceedings of the Third Symposium on Operating Systems Design and Implementation*, OSDI '99, páginas 173–186, Berkeley, CA, USA. USENIX Association.

[Chen e Micali 2019] Chen, J. e Micali, S. (2019). Algorand: A Secure and Efficient Distributed Ledger. *Theoretical Computer Science*, 777:155–183.

[Chen et al. 2017] Chen, L., Xu, L., Shah, N., Gao, Z., Lu, Y. e Shi, W. (2017). On security analysis of proof-of-elapsed-time (poet). Em *International Symposium on Stabilization, Safety, and Security of Distributed Systems*, páginas 282–297. Springer.

[Cheow 2020] Cheow, K. A. (2020). Something on Transaction Structure. Acessado em 9 de maio de 2022.

[Churyumov 2016] Churyumov, A. (2016). A decentralized system for storage and transfer of value. "https://obyte.org/Byteball.pdf".

[CloudTweaks 2021] CloudTweaks (2021). How Bitcoin Brought The Lightning Network To El Salvador. `https://cloudtweaks.com/2021/07/how-bitcoin-brought-lightning-network-el-salvador/`. Acessado em 3 de fevereiro de 2022.

[CoinMarketCap 2022] CoinMarketCap (2022). Cryptocurrency Prices, Charts and Market Capitalizations. Disponível em `https://coinmarketcap.com/`. Acessado em 9 de maio de 2022.

[Conoscenti et al. 2019] Conoscenti, M., Vetro, A. e De Martin, J. C. (2019). Hubs, rebalancing and service providers in the lightning network. *Ieee Access*, 7:132828–132840.

[Corbett et al. 2013] Corbett, J. C. et al. (2013). Spanner: Google's Globally Distributed Database. *ACM Transactions on Computer Systems*, 31(3).

[Costan e Devadas 2016] Costan, V. e Devadas, S. (2016). Intel sgx explained. Cryptology ePrint Archive, Report 2016/086. `https://ia.cr/2016/086`.

[Damasceno 2022] Damasceno, L. (2022). Qual máquina de cartão tem a melhor taxa de transação 2022? Disponível em `https://br.mobiletransaction.org/maquina-de-cartao-melhor-taxa/`. Acessado em 9 de maio de 2022.

[Dang et al. 2019] Dang, H., Dinh, T. T. A., Loghin, D., Chang, E.-C., Lin, Q. e Ooi, B. C. (2019). Towards Scaling Blockchain Systems via Sharding. Em *Proceedings of the 2019 international conference on management of data*, páginas 123–140.

[de Minas e Energia 2021] de Minas e Energia, M. (2021). Anuário estatístico de energia elétrica 2021. Relatório técnico, Ministério de Minas e Energia do Brasil. Acessado em 9 de maio de 2022.

[de Oliveira et al. 2019] de Oliveira, M. T. et al. (2019). Towards a Blockchain-based Secure Electronic Medical Record for Healthcare Applications. Em *IEEE International Conference on Communications (ICC)*, páginas 1–6.

[de Souza et al. 2020] de Souza, L. A. C. et al. (2020). DFedForest: Decentralized Federated Forest. Em *2020 IEEE International conference on blockchain (blockchain)*, páginas 90–97. IEEE.

[Decker 2021] Decker, C. (2021). Lightning network research; topology datasets. `https://github.com/lnresearch/topology`. Acessado em 29 de dezembro de 2021.

[Decker e Wattenhofer 2014] Decker, C. e Wattenhofer, R. (2014). Bitcoin Transaction Malleability and MtGox. Em *European Symposium on Research in Computer Security*, páginas 313–326. Springer.

[Decker e Wattenhofer 2015] Decker, C. e Wattenhofer, R. (2015). A fast and scalable payment network with bitcoin duplex micropayment channels. Em Pelc, A. e Schwarzmann, A. A., editors, *Stabilization, Safety, and Security of Distributed Systems*, páginas 3–18, Cham. Springer International Publishing.

[Digiconomist 2022] Digiconomist (2022). Bitcoin Energy Consumption Index. Acessado em 9 de maio de 2022.

[Dingledine et al. 2004] Dingledine, R., Mathewson, N. e Syverson, P. (2004). Tor: The Second-Generation onion router. Em *13th USENIX Security Symposium (USENIX Security 04)*, San Diego, CA. USENIX Association.

[Dziembowski et al. 2018] Dziembowski, S., Faust, S. e Hostáková, K. (2018). General state channel networks. Em *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, CCS '18, página 949–966, New York, NY, USA. Association for Computing Machinery.

[Edmonds e Karp 1972] Edmonds, J. e Karp, R. M. (1972). Theoretical improvements in algorithmic efficiency for network flow problems. *J. ACM*, 19(2):248–264.

[Ellis et al. 2017] Ellis, S., Juels, A. e Nazarov, S. (2017). Chainlink: A Decentralized Oracle Network. *Retrieved March*, 11:38. Acessado em 9 de maio de 2022.

[Goldwasser et al. 1985] Goldwasser, S., Micali, S. e Rackoff, C. (1985). The knowledge complexity of interactive proof-systems. Em *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing*, STOC '85, página 291–304, New York, NY, USA. Association for Computing Machinery.

[Gopalan et al. 2020] Gopalan, A., Sankararaman, A., Walid, A. e Vishwanath, S. (2020). Stability and Scalability of Blockchain Systems. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 4(2):1–35.

[Gudgeon et al. 2020] Gudgeon, L., Moreno-Sanchez, P., Roos, S., McCorry, P. e Gervais, A. (2020). SoK: Layer-two Blockchain Protocols. Em *International Conference on Financial Cryptography and Data Security*, páginas 201–226. Springer.

[Hearn 2013] Hearn, M. (2013). [ANNOUNCE] Micro-payment channels implementation now in bitcoinj. Bitcoin Forum. Disponível em `https://bitcointalk.org/index.php?topic=244656.0`.

[Hong et al. 2021] Hong, Z., Guo, S., Li, P. e Chen, W. (2021). Pyramid: A Layered Sharding Blockchain System. Em *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*, páginas 1–10. IEEE.

[Huang et al. 2022] Huang, H., Peng, X., Zhan, J., Zhang, S., Lin, Y., Zheng, Z. e Guo, S. (2022). BrokerChain: A Cross-Shard Blockchain Protocol for Account/Balance-based State Sharding. Em *IEEE INFOCOM*.

[ION Lightning Network Wiki 2021] ION Lightning Network Wiki (2021). Watchtowers. Disponível em: https://wiki.ion.radar.tech/tech/research/watchtowers. Acessado em 9 de maio de 2022.

[Javaid et al. 2021] Javaid, H., Yang, J., Santoso, N., Upadhyay, M., Mohan, S., Hu, C. e Brebner, G. (2021). Blockchain machine: A network-attached hardware accelerator for hyperledger fabric. *arXiv preprint arXiv:2104.06968*.

[Kappos et al. 2021] Kappos, G., Yousaf, H., Piotrowska, A., Kanjalkar, S., Delgado-Segura, S., Miller, A. e Meiklejohn, S. (2021). An empirical analysis of privacy in the lightning network. Em *International Conference on Financial Cryptography and Data Security*, páginas 167–186. Springer.

[Khalil e Gervais 2017] Khalil, R. e Gervais, A. (2017). Revive: Rebalancing Off-Blockchain Payment Networks. Em *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, página 439–453, New York, NY, USA. Association for Computing Machinery.

[Khojasteh e Tabatabaei 2021] Khojasteh, H. e Tabatabaei, H. (2021). A survey and taxonomy of blockchain-based payment channel networks. Em *2021 IEEE High Performance Extreme Computing Conference (HPEC)*, páginas 1–8. IEEE.

[Kim et al. 2018] Kim, S., Kwon, Y. e Cho, S. (2018). A Survey of Scalability Solutions on Blockchain. Em *2018 International Conference on Information and Communication Technology Convergence (ICTC)*, páginas 1204–1207.

[Kokoris-Kogias et al. 2018] Kokoris-Kogias, E., Jovanovic, P., Gasser, L., Gailly, N., Syta, E. e Ford, B. (2018). Omniledger: A Secure, Scale-out, Decentralized Ledger via Sharding. Em *2018 IEEE Symposium on Security and Privacy (SP)*, páginas 583–598. IEEE.

[Kshetri e Voas 2018] Kshetri, N. e Voas, J. (2018). Blockchain-Enabled E-Voting. *IEEE Software*, 35(4):95–99.

[Kwon e Buchman 2019] Kwon, J. e Buchman, E. (2019). Cosmos Whitepaper. *A Netw. Distrib. Ledgers*.

[Lamport et al. 1982] Lamport, L., Shostak, R. e Pease, M. (1982). The Byzantine Generals Problem. *ACM Transactions on Programming Languages and System*.

[Lange et al. 2021] Lange, K., Rohrer, E. e Tschorsch, F. (2021). On the impact of attachment strategies for payment channel networks. *arXiv preprint arXiv:2102.09256*.

[Larimer 2017] Larimer, D. (2017). EOS.IO White Paper. Disponível em https://developers.eos.io/-welcome/latest/protocol/consensus_protocol. Acessado em 9 de maio de 2022.

[LetsExchange 2021] LetsExchange (2021). What Is Block Confirmation on Ethereum and How Many Confirmations Are Required? Disponível em `https://letsexchange.io/blog/what-is-block-confirmation-on-ethereum-and-how-many-\confirmations-are-required/`. Acessado em 9 de maio de 2022.

[Li et al. 2017] Li, W., Sforzin, A., Fedorov, S. e Karame, G. O. (2017). Towards Scalable and Private Industrial Blockchains. Em *Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts*, páginas 9–14.

[Lightning Network Developers 2022] Lightning Network Developers (2022). Lightning App Directory. Disponível em `https://dev.lightning.community/lapps/`.

[Lin et al. 2020] Lin, J.-H., Primicerio, K., Squartini, T., Decker, C. e Tessone, C. J. (2020). Lightning network: a second path towards centralisation of the bitcoin economy. *New Journal of Physics*, 22(8):083022.

[Lind et al. 2017] Lind, J., Eyal, I., Kelbert, F., Naor, O., Pietzuch, P. e Sirer, E. G. (2017). Teechain: Scalable blockchain payments using trusted execution environments. *arXiv preprint arXiv:1707.05454*.

[Lind et al. 2019] Lind, J., Naor, O., Eyal, I., Kelbert, F., Sirer, E. G. e Pietzuch, P. (2019). Teechain: A secure payment network with asynchronous blockchain access. Em *Proceedings of the 27th ACM Symposium on Operating Systems Principles*, SOSP '19, página 63–79, New York, NY, USA. Association for Computing Machinery.

[Lombrozo et al. 2015] Lombrozo, E., Lau, J. e Wuille, P. (2015). BIP 141: Segregated Witness (Consensus Layer). Disponível em `https://www.omgwiki.org/dido/doku.php?id=dido:public:ra:xapend:xapend.b_stds:defact:bitcoin:bips:bip_0141`. Acessado em 9 de maio de 2022.

[Luu et al. 2016] Luu, L., Narayanan, V., Zheng, C., Baweja, K., Gilbert, S. e Saxena, P. (2016). A Secure Sharding Protocol for Open Blockchains. Em *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, páginas 17–30.

[Malavolta et al. 2016] Malavolta, G., Moreno-Sanchez, P., Kate, A. e Maffei, M. (2016). Silentwhispers: Enforcing security and privacy in decentralized credit networks. Cryptology ePrint Archive, Report 2016/1054. https://ia.cr/2016/1054.

[Malavolta et al. 2017] Malavolta, G., Moreno-Sanchez, P., Kate, A., Maffei, M. e Ravi, S. (2017). Concurrency and privacy with payment-channel networks. Em *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*.

[Malavolta et al. 2018] Malavolta, G., Moreno-Sanchez, P., Schneidewind, C., Kate, A. e Maffei, M. (2018). Anonymous Multi-Hop Locks for Blockchain Scalability and Interoperability. Cryptology ePrint Archive, Report 2018/472. https://ia.cr/2018/472.

[Mohanty et al. 2020] Mohanty, S. P., Yanambaka, V. P., Kougianos, E. e Puthal, D. (2020). Pufchain: A hardware-assisted blockchain for sustainable simultaneous device and data security in the internet of everything (ioe). *IEEE Consumer Electronics Magazine*, 9(2):8–16.

[Múltiplos Autores 2022a] Múltiplos Autores (2022a). BOLT #0: Introduction and index. https://github.com/lightning/bolts/blob/master/00-introduction.md. Acessado em 9 de maio de 2022.

[Múltiplos Autores 2022b] Múltiplos Autores (2022b). BOLT #2: Peer protocol for channel management. https://github.com/lightningnetwork/lightning-rfc/blob/master/02-peer-protocol.md. Acessado em 9 de maio de 2022.

[Múltiplos Autores 2022c] Múltiplos Autores (2022c). BOLT #3: Bitcoin transaction and script formats. https://github.com/lightning/bolts/blob/master/03-transactions.md#fees. Acessado em 9 de maio de 2022.

[Nakamoto 2008] Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system.

[OMNeT++ 2022] OMNeT++ (2022). Omnet++ discrete event simulator. Disponível em https://omnetpp.org/. Acessado em 9 de maio de 2022.

[Osuntokun 2018] Osuntokun, O. (2018). [Lightning-dev] AMP: Atomic Multi-Path Payments over Lightning. Disponível em https://lists.linuxfoundation.org/pipermail/lightning-dev/2018-February/000993.html. Acessado em 9 de maio de 2022.

[Otto 2022] Otto, C. (2022). Rebalance-LND. Disponível em: https://github.com/C-Otto/rebalance-lnd. Acessado em 9 de maio de 2022.

[Pickhardt e Nowostawski 2020] Pickhardt, R. e Nowostawski, M. (2020). Imbalance measure and proactive channel rebalancing algorithm for the lightning network. Em *2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, páginas 1–5. IEEE.

[Pickhardt e Richter 2021] Pickhardt, R. e Richter, S. (2021). Optimally reliable & cheap payment flows on the lightning network. *CoRR*, abs/2107.05322.

[Poon e Buterin 2017] Poon, J. e Buterin, V. (2017). Plasma: Scalable Autonomous Smart Contracts. *White paper*, páginas 1–47.

[Poon e Dryja 2016] Poon, J. e Dryja, T. (2016). The bitcoin lightning network: Scalable off-chain instant payments.

[Popov 2017] Popov, S. (2017). The tangle. *cit. on*, página 131. `http://www.descryptions.com/Iota.pdf`. Acessado em 31 de outubro de 2018.

[Rebello et al. 2021a] Rebello, G. A., Potop-Butucaru, M., de Amorim, M. e Duarte, O. C. (2021a). Protegendo redes de canais de pagamento sem fio com janelas de tempo de bloqueio mínimas. Em *Anais do XXI SBSeg*, páginas 295–308. SBC.

[Rebello et al. 2021b] Rebello, G. A. F., Camilo, G. F., Guimarães, L. C., de Souza, L. A. C., Thomaz, G. A. e Duarte, O. C. (2021b). A security and performance analysis of proof-based consensus protocols. *Annals of Telecommunications*, páginas 1–21.

[Rebello et al. 2022] Rebello, G. A. F., Camilo, G. F., Potop-Butucaru, M., Campista, M. E. M., de Amorim, M. D. e Costa, L. H. M. K. (2022). PCNsim: A Flexible and Modular Simulator for Payment Channel Networks. Demos do IEEE International Conference on Computer Communications. A ser publicado.

[Rebello et al. 2019] Rebello, G. A. F., Camilo, G. F., Silva, L. G., de Souza, L. A., Guimarães, L. C. e Duarte, O. C. M. (2019). Segurança na internet do futuro: Provendo confiança distribuída através de correntes de blocos na virtualização de funções de rede. *Sociedade Brasileira de Computação*.

[Ribeiro 2022] Ribeiro, M. (2022). Pela primeira vez, pix supera cartão em transações. Disponível em `https://valor.globo.com/financas/noticia/2022/03/30/pela-primeira-vez-pix-supera-cartao-em-transacoes.ghtml`. Acessado em 9 de maio de 2022.

[Rohrer et al. 2019] Rohrer, E., Malliaris, J. e Tschorsch, F. (2019). Discharged payment channels: Quantifying the lightning network's resilience to topology-based attacks. Em *2019 IEEE European Symposium on Security and Privacy Workshops (EuroS PW)*, páginas 347–356.

[Roos et al. 2017] Roos, S., Moreno-Sanchez, P., Kate, A. e Goldberg, I. (2017). Settling payments fast and private: Efficient decentralized routing for path-based transactions.

[Sakakibara et al. 2018] Sakakibara, Y., Morishima, S., Nakamura, K. e Matsutani, H. (2018). A hardware-based caching system on fpga nic for blockchain. *IEICE Transactions on Information and Systems*, 101(5):1350–1360.

[Schwartz et al. 2014] Schwartz, D., Youngs, N. e Britto, A. (2014). The ripple protocol consensus algorithm. *Ripple Labs Inc White Paper*. https://ripple.com/files/ripple_consensus_whitepaper.pdf.

[Seres et al. 2020] Seres, I. A., Gulyás, L., Nagy, D. A. e Burcsi, P. (2020). Topological analysis of bitcoin's lightning network. Em *Mathematical Research for Blockchain Economy*, páginas 1–12. Springer.

[Singh et al. 2020] Singh, A., Click, K., Parizi, R. M., Zhang, Q., Dehghantanha, A. e Choo, K.-K. R. (2020). Sidechain Technologies in Blockchain Networks: An Examination and State-of-the-Art Review. *Journal of Network and Computer Applications*, 149:102471.

[Sivaraman et al. 2018] Sivaraman, V., Venkatakrishnan, S. B., Alizadeh, M., Fanti, G. e Viswanath, P. (2018). Routing Cryptocurrency with the Spider Network. Em *Proceedings of the ACM Workshop on Hot Topics in Networks*, páginas 29–35.

[Sivaraman et al. 2020] Sivaraman, V., Venkatakrishnan, S. B., Ruan, K., Negi, P., Yang, L., Mittal, R., Fanti, G. e Alizadeh, M. (2020). High throughput cryptocurrency routing in payment channel networks. Em *17th USENIX NSDI 20*, páginas 777–796.

[Spilman 2013] Spilman, J. (2013). [Bitcoin-development] Anti DoS for tx Replacement. Disponível em `https://lists.linuxfoundation.org/pipermail/bitcoin-dev/2013-April/002433.html`.

[Syta et al. 2017] Syta, E., Jovanovic, P., Kogias, E. K., Gailly, N., Gasser, L., Khoffi, I., Fischer, M. J. e Ford, B. (2017). Scalable Bias-Resistant Distributed Randomness. Em *2017 IEEE Symposium on Security and Privacy (SP)*, páginas 444–460.

[Tang et al. 2020] Tang, W., Wang, W., Fanti, G. e Oh, S. (2020). Privacy-utility tradeoffs in routing cryptocurrency over payment channel networks. *Proc. ACM Meas. Anal. Comput. Syst.*, 4(2).

[Team e Barrett 2018] Team, Z. e Barrett, P. (2018). The Zilliqa Project: A Secure, Scalable Blockchain Platform. *Zilliqa*, páginas 1–18.

[The Hyperledger Foundation 2022] The Hyperledger Foundation (2022). Hyperledger sawtooth. Disponível em `https://sawtooth.hyperledger.org/`. Acessado em 9 de maio de 2022.

[Visa 2022] Visa (2022). Visa Acceptance for Retailers. Disponível em `https://usa.visa.com/run-your-business/small-business-tools/retail.html`. Acessado em 9 de maio de 2022.

[Visa Inc. 2022] Visa Inc. (2022). Q2 2021 operational performance data. Acessado em 9 de maio de 2022.

[Wang et al. 2019a] Wang, G., Shi, Z. J., Nixon, M. e Han, S. (2019a). SoK: Sharding on Blockchain. Em *Proceedings of the 1st ACM Conference on Advances in Financial Technologies*, páginas 41–61.

[Wang e Wang 2019] Wang, J. e Wang, H. (2019). Monoxide: Scale out Blockchains with Asynchronous Consensus Zones. Em *16th USENIX Symposium on Networked Systems Design and Implementation (NSDI 19)*, páginas 95–112.

[Wang et al. 2019b] Wang, P., Xu, H., Jin, X. e Wang, T. (2019b). Flash: efficient dynamic routing for offchain networks. Em *Proceedings of the 15th International Conference on Emerging Networking Experiments And Technologies*, páginas 370–381.

[Wang et al. 2020] Wang, Q., Yu, J., Chen, S. e Xiang, Y. (2020). SoK: Diving into DAG-based blockchain systems. https://arxiv.org/abs/2012.06128v2.

[Wood 2014] Wood, G. (2014). Ethereum: A Secure Decentralised Generalised Transaction Ledger.

[World Bank 2022] World Bank (2022). GDP (current US$). Acessado em 9 de maio de 2022.

[Xiao et al. 2020] Xiao, Y., Zhang, N., Lou, W. e Hou, Y. T. (2020). A survey of distributed consensus protocols for blockchain networks. *IEEE Communications Surveys & Tutorials*, 22(2):1432–1465.

[Xie et al. 2019] Xie, J., Yu, F. R., Huang, T., Xie, R., Liu, J. e Liu, Y. (2019). A Survey on the Scalability of Blockchain Systems. *IEEE Network*, 33(5):166–173.

[Yang et al. 2020] Yang, D., Long, C., Xu, H. e Peng, S. (2020). A Review on Scalability of Blockchain. Em *Proceedings of the 2020 The 2nd International Conference on Blockchain Technology*, páginas 1–6.

[Yang et al. 2019] Yang, F., Zhou, W., Wu, Q., Long, R., Xiong, N. N. e Zhou, M. (2019). Delegated Proof of Stake with Downgrade: A Secure and Efficient Blockchain Consensus Algorithm with Downgrade Mechanism. *IEEE Access*, 7:118541–118555.

[Zabka et al. 2021] Zabka, P., Förster, K.-T., Schmid, S. e Decker, C. (2021). Node classification and geographical analysis of the lightning cryptocurrency network. Em *International Conference on Distributed Computing and Networking 2021*, ICDCN '21, página 126–135, New York, NY, USA. Association for Computing Machinery.

[Zamani et al. 2018] Zamani, M., Movahedi, M. e Raykova, M. (2018). Rapidchain: Scaling blockchain via full sharding. Em *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, páginas 931–948.

[Zhao e Yu 2019] Zhao, L. e Yu, J. (2019). Evaluating DAG-based blockchains for IoT. Em *2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, páginas 507–513.

[Zhou et al. 2020] Zhou, Q., Huang, H., Zheng, Z. e Bian, J. (2020). Solutions to Scalability of Blockchain: A Survey. *IEEE Access*, 8:16440–16455.