FORTALEZA - CE

SBRC

XL SIMPÓSIO BRASILEIRO DE REDES DE COMPUTADORES E SISTEMAS DISTRIBUÍDOS

Minicursos da 40ª edição do Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos

Organizadores:
Rafael L. Gomes (UECE)
Rossana Maria C. Andrade (UFC)
Fátima Duarte-Figueiredo (PUC-MG)
Miguel Elias M. Campista (UFRJ)

SBRC

XL SIMPÓSIO BRASILEIRO DE REDES DE COMPUTADORES E SISTEMAS DISTRIBUÍDOS

Organizadores:
Rafael L. Gomes (UECE)
Rossana Maria C. Andrade (UFC)
Fátima Duarte-Figueiredo (PUC-MG)
Miguel Elias M. Campista (UFRJ)

Minicursos da 40ª edição do Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos

Porto Alegre
Sociedade Brasileira de Computação – SBC
2022

Dados Internacionais de Catalogação na Publicação (CIP)

S612 Simpósio Brasileiro de Redes de Computadores e de Sistemas Distribuídos (40. : 23 – 27 maio 2022 : Fortaleza)
Minicursos da 40ª Edição do Simpósio Brasileiro de Redes de Computadores e Sistema Distribuídos (SBRC 2022) [recurso eletrônico] / organização: Rafael L. Gomes ... [et al.]. Dados eletrônicos. – Porto Alegre: Sociedade Brasileira de Computação, 2022.
157 p. : il. : PDF ; 11.7MB

Modo de acesso: World Wide Web.
Inclui bibliografia
ISBN 978-85-7669-513-4 (e-book)

1. Computação – Brasil – Simpósio. 2. Redes de computadores. 3. Sistemas distribuídos. I. Gomes, Rafael L.. II. Andrade, Rossana Maria C.. III. Duarte-Figueiredo, Fátima. IV. Campista, Miguel Elias M.. V. Sociedade Brasileira de Computação. VI. Universidade Federal do Ceará. VII. Título.

CDU 004.7(063)

Ficha catalográfica elaborada por Jéssica Paola Macedo Müller – CRB-10/2662

Biblioteca Digital da SBC – SBC OpenLib

Índices para catálogo sistemático:

1. Ciência e tecnologia dos computadores : Informática : Comunicação de computadores :
Redes de computadores – Publicação de conferências, congressos e simpósios etc. ...
004.7(063)

Sumário

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.

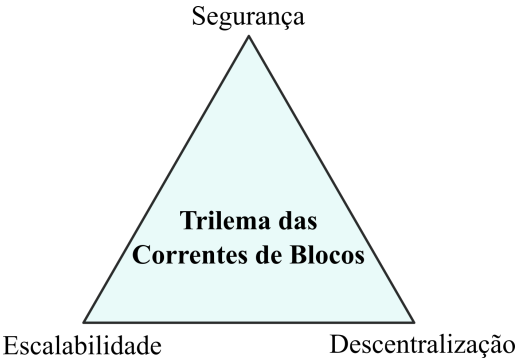


Figura 1.1. Trilema observável em sistemas baseados em corrente de blocos. Nenhuma corrente de blocos consegue prover, simultaneamente, um sistema altamente seguro, escalável e descentralizado.

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].

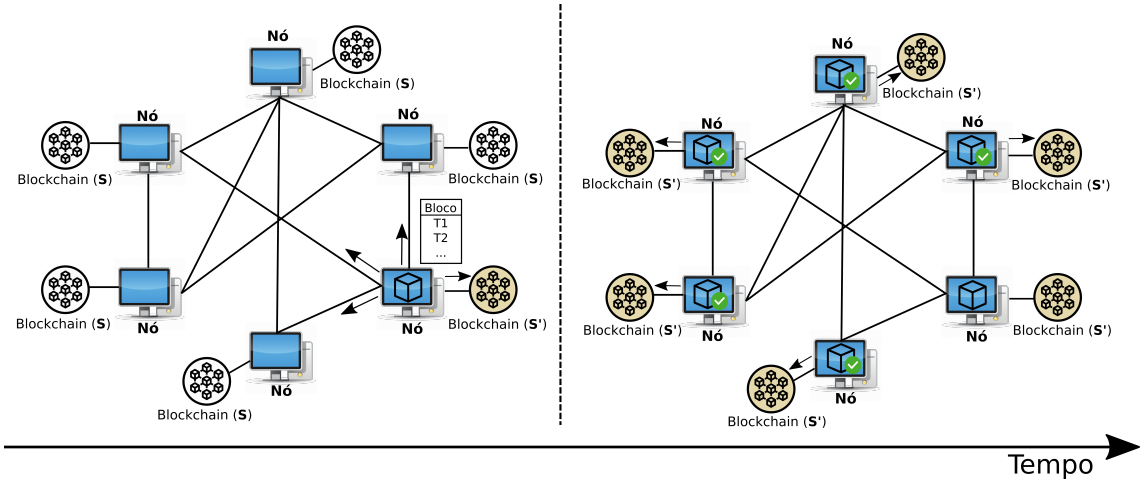


Figura 1.2. Atualização de uma corrente de blocos (*blockchain*) através de um protocolo de consenso genérico. Um dos participantes do consenso propõe, em difusão, um novo bloco que muda o estado global de S para S'. Os demais participantes verificam e adicionam independentemente o bloco proposto à corrente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

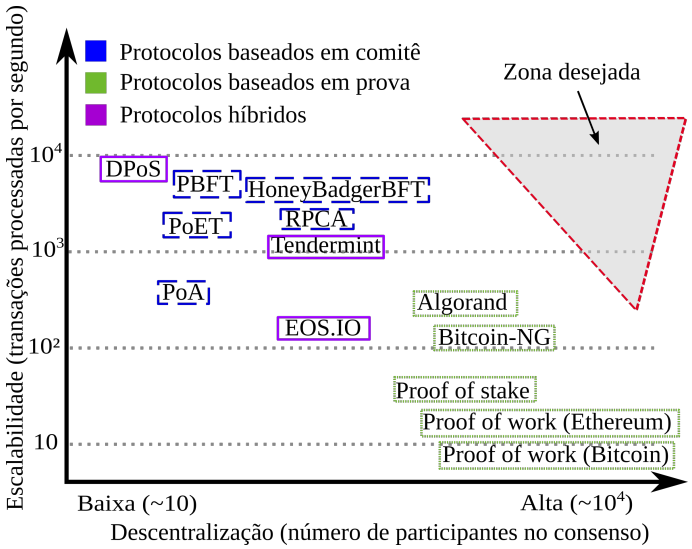


Figura 1.3. Comparação entre os principais protocolos de consenso de sistemas baseados em corrente de blocos. O compromisso observado entre desempenho e descentralização dificulta a proposta de um protocolo que seja simultaneamente escalável e tolerante a ataques de conluio.

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

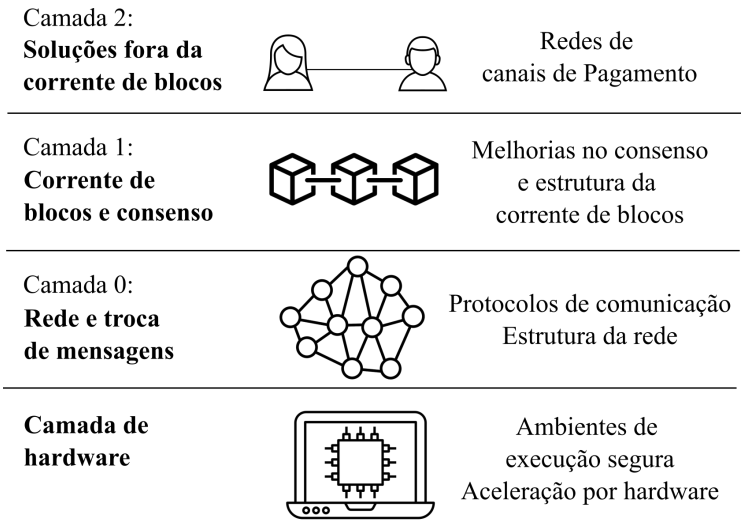


Figura 1.4. Camadas da arquitetura de sistemas baseados em corrente de blocos, adaptada de Gudgeon *et al.* [Gudgeon et al. 2020]. A maior parte das propostas para melhorar a escalabilidade desses sistemas encontra-se nas camadas 1 e 2, por serem as camadas mais fáceis de modificar em larga escala dentro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

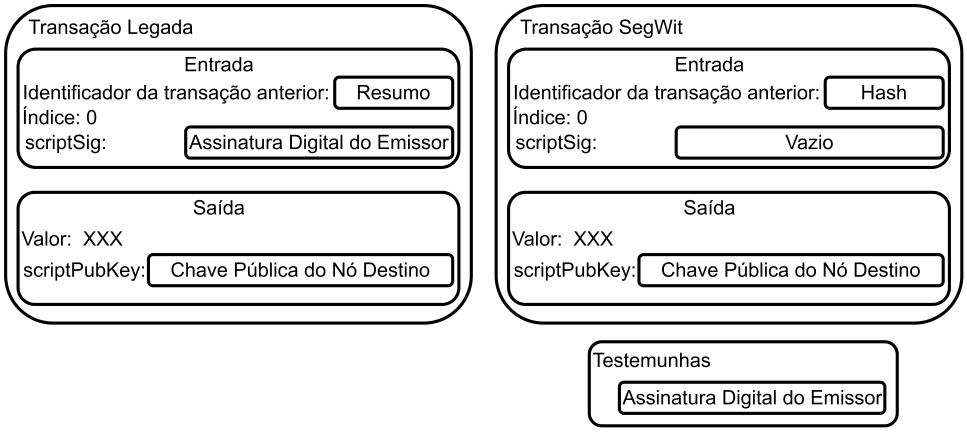


Figura 1.5. Comparação da estrutura de uma transação legada com uma transação SegWit. As transações legadas incluem a assinatura digital na própria transação, gerando uma sobrecarga no tamanho final da transação. Por outro lado, a transação SegWit dissocia a assinatura do emissor da transação através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

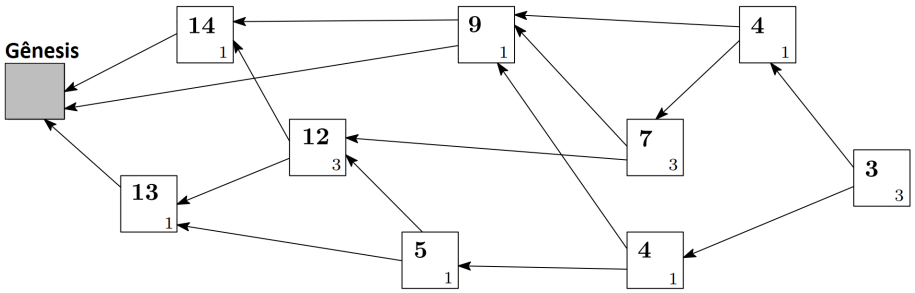


Figura 1.6. Exemplo da estrutura de dados do IOTA. As transações constituem um grafo acíclico dirigido. Cada transação é encadeada com duas transações predecessoras, com exceção das primeiras transações da rede. A estrutura garante propriedades similares às da corrente de blocos.

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

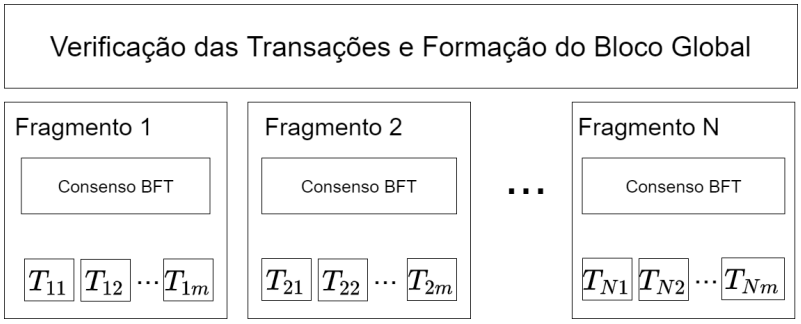


Figura 1.7. Execução de uma rodada de consenso em um sistema baseado em fragmentação. Cada fragmento gera um conjunto de transações válidas através de um acordo bizantino. Após essa etapa, os conjuntos de transações são combinados para formar o bloco global.

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].

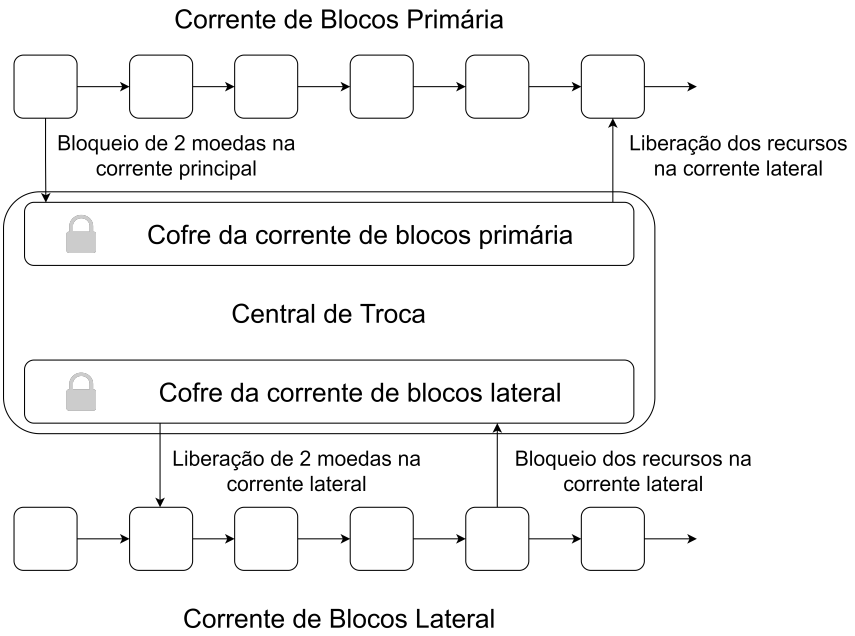


Figura 1.8. Exemplo de uma corrente lateral com o processo de troca entre correntes centralizado. Um usuário das duas correntes de blocos inicialmente bloqueia recursos na corrente de blocos principal e os obtêm na corrente de blocos lateral. A central de troca supervisiona as duas correntes de blocos 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

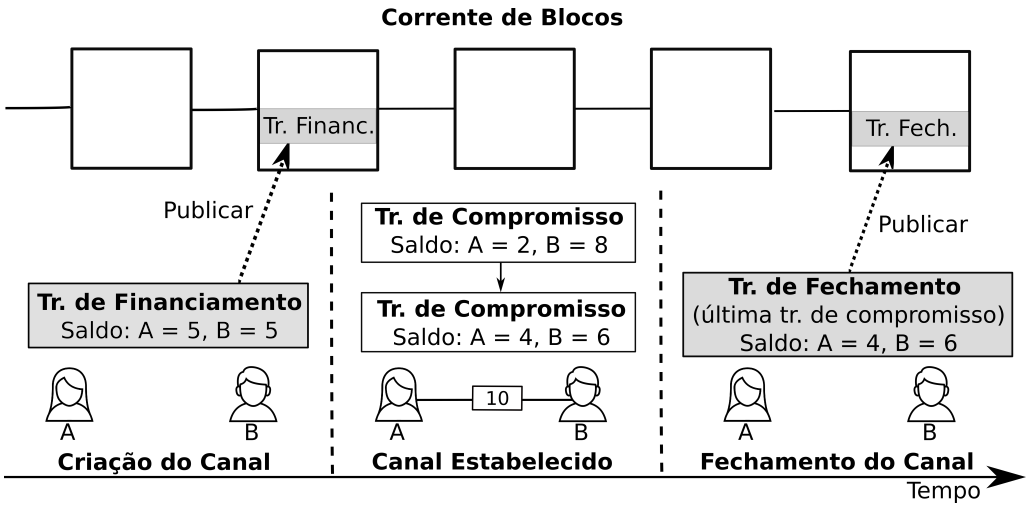


Figura 1.9. Abertura e fechamento de um canal de pagamento. Na figura, os usuários A e B contribuem com 5 moedas cada e emitem a transação de financiamento para criar um canal de pagamentos. O valor 10 no retângulo entre os dois usuários mostra a capacidade máxima do canal, que representa a quantidade 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.

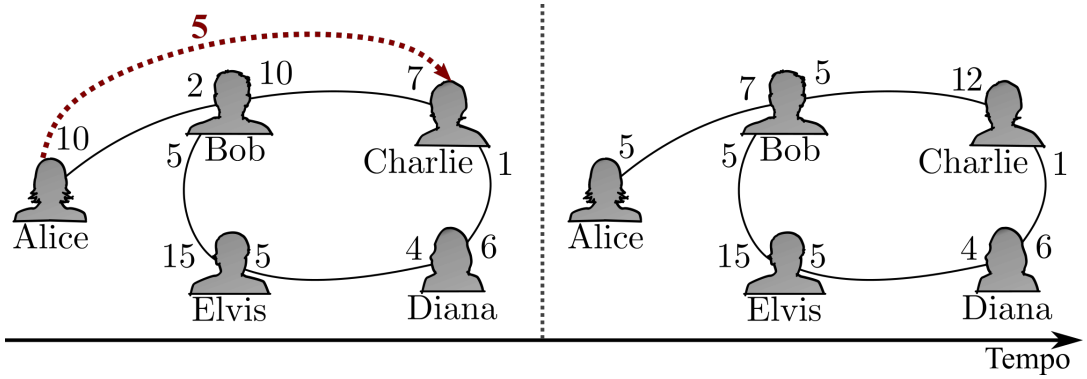


Figura 1.10. Exemplo de pagamento ocorrendo em uma rede de canais de pagamento. Para enviar um pagamento de 5 moedas a Charlie, Alice utiliza o canal pré-estabelecido com Bob, que as repassa ao destino. O pagamento modifica as capacidades dos canais envolvidos em seu caminho. A segurança desse procedimento 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

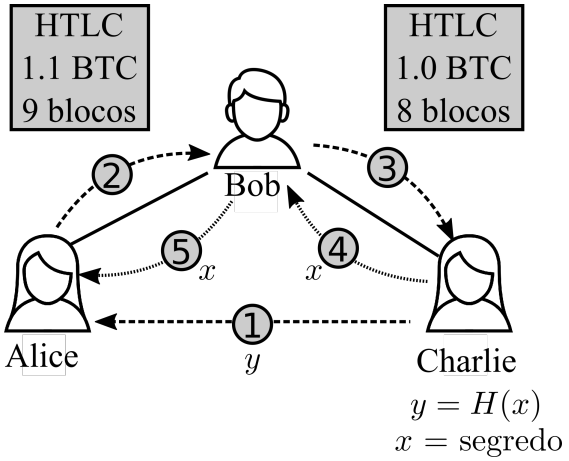


Figura 1.11. Etapas para efetuar um pagamento em uma rede de canais de pagamentos. 1) Charlie, o destinatário, gera um segredo e envia o *hash* deste segredo a Alice. 2) Alice não possui canal com Charlie e utiliza Bob como intermediário de pagamento, estabelecendo um contrato prometendo entregar dinheiro 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].

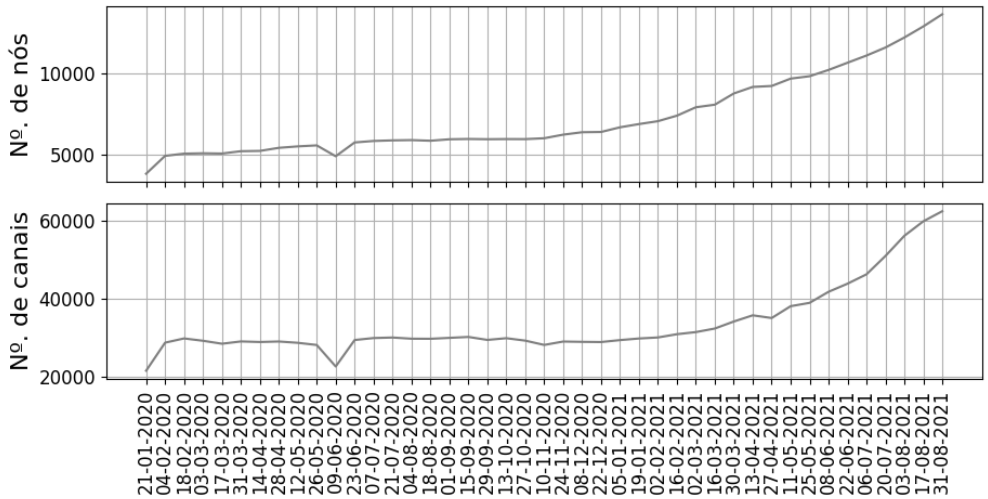


Figura 1.12. Crescimento do número de nós e de canais na Rede Relâmpago no período entre janeiro de 2020 e agosto de 2021.

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.

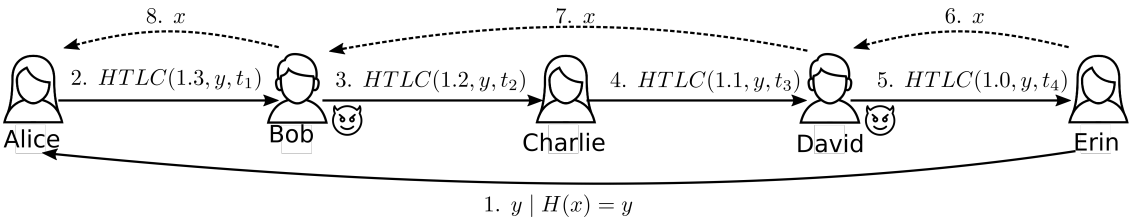


Figura 1.13. Exemplo de ataque de buraco de minhoca. Bob e David agem em conluio para roubar tarifas que seriam destinadas a Charlie. As linhas sólidas compreendem as etapas de estabelecimento de um pagamento, enquanto as linhas pontilhadas compreendem o resgate do pagamento.

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.

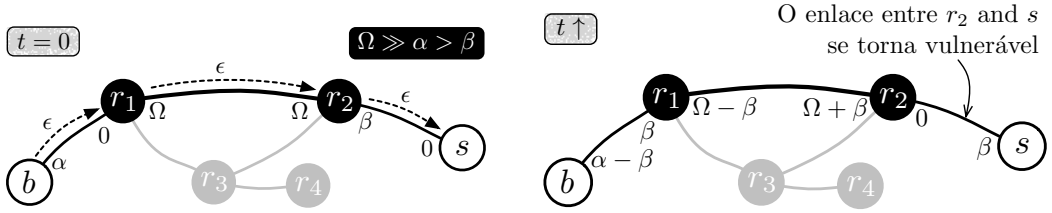


Figura 1.14. Um exemplo da vulnerabilidade de roubo de moedas em redes de canais de pagamento. À esquerda, uma quantidade contínua de ε moedas flui de uma origem b para um destino s até que a capacidade do canal entre o roteador r_2 e s se esgote. Então, à direta, s se torna altamente vulnerável se perder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

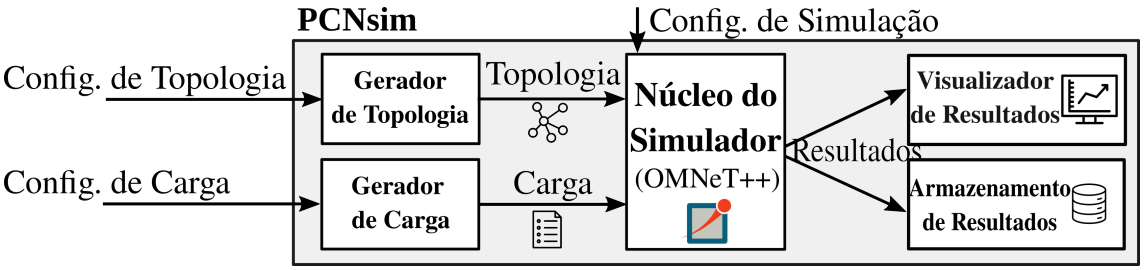


Figura 1.15. Arquitetura do PCNsim. Usuários podem facilmente configurar a topologia e carga de transações da simulação através dos geradores de topologia e de carga, além de visualizar os resultados da simulação.

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.

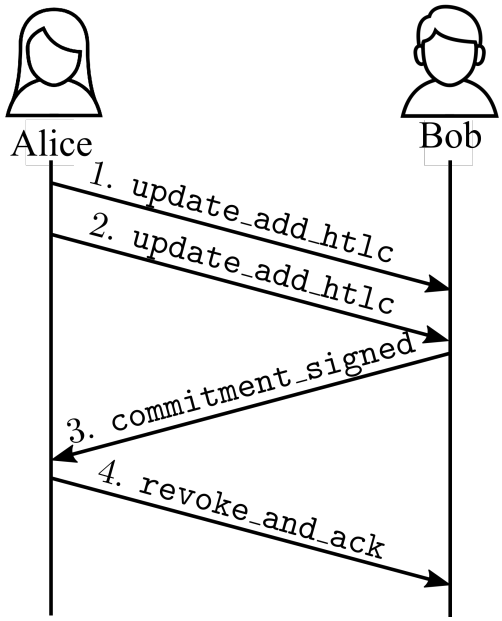


Figura 1.16. Troca de mensagens para o estabelecimento de um contrato bloqueado por tempo e por *hash*. Alice oferece um ou múltiplos contratos de pagamentos a Bob, que pode aceitá-los ou recusá-los. Bob atualiza os contratos aceitos localmente e envia uma mensagem assinada a Alice contendo os contratos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

Capítulo 2

Tokens Não Fungíveis (NFTs): Conceitos, Aplicações e Desafios

Ronan D. Mendonça (UFV), Josué N. Campos (UFV), Luiz F. M. Vieira (UFMG), Marcos A. M. Vieira (UFMG), Alex Borges Vieira (UFJF), José A. M. Nacif (UFV)

Abstract

Nowadays, we have been surprised by the popularization of Non-Fungible Token (NFT) markets due to their great success in the blockchain community. NFTs are records of asset ownership held by smart contracts on a blockchain. These records can be traded, transferred and rented through an exchange of ownership. The purpose of this work is to introduce the main concepts, applications and challenges related to NFTs. We present a theoretical view of the technologies involved in the creation and maintenance of NFTs, along with decentralized applications and marketplaces that trade the tokens. Moreover, we list the practical issues for using token standards and creating a market for trading and renting NFTs.

Resumo

Atualmente fomos surpreendidos com a popularização de mercados de Token Não-Fungível (NFT) devido ao seu grande sucesso obtido diante da comunidade blockchain. Os NFTs são registros da propriedade de ativos realizados por contratos inteligentes em uma blockchain. Estes registros podem ser negociados, transferidos e alugados por meio da troca de titularidade. O objetivo deste capítulo é apresentar os principais conceitos, aplicações e desafios relacionados aos NFTs. Apresentamos uma visão teórica das tecnologias envolvidas para a criação e manutenção dos NFTs, juntamente a aplicações descentralizadas e marketplaces que fazem negociações dos tokens. Além disso, apresentamos as questões práticas para a utilização dos padrões de tokens e a criação de mercado para comercialização e aluguel dos NFTs.

2.1. Introdução

Plataformas e aplicações baseadas em *blockchains*, em sua essência, são desenvolvidas para permitir transações financeiras entre usuários anônimos. Isso é possível com o uso de criptografia assimétrica, componente que faz parte da tecnologia *blockchain*, onde os usuários se identificam pelas suas chaves públicas e usam as chaves privadas secretas para autenticar as transações que eles emitem em favor de outros usuários [Wood 2014]. As *blockchains* armazenam as transações realizadas em forma de blocos interconectados por meio de *hashes* criptográficos e, um dos principais benefícios dessa tecnologia é que os dados são mantidos distribuídos pelos entes participantes da rede, sem a necessidade de uma autoridade central [Hewa et al. 2021a].

A relevância dada por pesquisadores e desenvolvedores à tecnologia *blockchain* aumentou significativamente a busca por novas formas de explorar suas características. De fato, as plataformas *blockchains* populares atuais, como Bitcoin e Ethereum, têm diversas funcionalidades com potencial para alavancar novos negócios. Por exemplo, aplicações financeiras descentralizadas (*DeFis*) [Schär 2020] oferecem serviços[1] para pagamentos, empréstimos, doações ou *royalties* por bens digitais (e.g., *tokens não fungíveis*). Há também serviços para análise de perfis dos usuários dessas plataformas, para analisar riscos de crédito e até mesmo para classificar usuários com envolvimento em esquemas fraudulentos [Bartoletti et al. 2020].

Dentre os diversos domínios possíveis de uso das *blockchains*, destaca-se atualmente o domínio dos *tokens*, no qual o uso de aplicações específicas deste domínio visa agregar forma de garantia de posse e representação de bens. Os bens são objetos físicos ou digitais de utilidade variada e que podem ser trocados ou vendidos. Podem ser classificados como um bem fungível e não fungível, sendo que um bem fungível significa que ele pode ser substituído por outro que representa o mesmo valor, e já um bem não fungível não admite substituição e são considerados com valor especial e individual [Fairfield 2021]. Um *token* não fungível (*Non-Fungible Token* (NFT)) é uma tecnologia que permite registrar de maneira distribuída a posse de um bem não fungível. Portanto, o NFT é um token ou certificado que comprova a propriedade de itens exclusivos. São utilizados para provar a propriedade de bens exclusivos como itens colecionáveis ou de investimento, pois se pode revender um NFT e obter lucros com base em seu valor atual[Valeonti et al. 2021]. Um exemplo de bens fungíveis é o próprio dinheiro onde cada nota comum tem o mesmo valor que todas as outras notas. Desta mesma forma, as criptomoedas também se aplicam a este mesmo conceito, onde um Ether e um Bitcoin têm o mesmo valor de outro um Ether e outro um Bitcoin respectivamente. Sendo assim, um item é substituível por qualquer outro item do mesmo tipo e considerados itens fungíveis. Mas para itens que não têm valores iguais a qualquer outro ou não podem ser igualmente trocados, recebem a denominação de itens não fungíveis.

As implementações de NFTs em *blockchains* é uma tecnologia relativamente nova e suas aplicações têm o potencial de mudar a forma de determinar a verificação do direito de posse. Os NFTs trouxeram para as plataformas *blockchains* novas possibilidades que ampliaram o oferecimento de aplicações inovadoras em vários setores como, por exemplo arte, música, moda e bebidas. Atualmente, os NFTs são amplamente conhecidos por

[1] Exemplos de DeFi Ecosystem - https://defiprime.com/ethereum

proporcionarem compras de obras de arte virtuais. Até mesmo um jogador de futebol brasileiro famoso virou notícia ao comprar uma obra de arte digital por meio de NFT em 2021 [2]. Além das obras de artes, NFTs também possibilitam o registro de posse de outros bens (ou a forma de determinar a verificação do direito de posse), como roupas, músicas, terrenos, personagens em jogos, ingressos para shows, entre outros.

Mais precisamente, os NFTs são descritos como registros individuais de posse e são registrados em plataformas *blockchains*, como por exemplo, o Ethereum. Por meio deles é possível definir a propriedade para um determinado item e, assim, atender diversas aplicações como, por exemplo, a certificação de direito autoral e de posse de objetos físicos e digitais [Valeonti et al. 2021]. Um NFT pode ser utilizado para identificar e descrever objetos do mundo real e ou do mundo digital. Tais objetos são representados de forma a possuir propriedades e características únicas e, desta maneira, não podem ser simplesmente trocados ou comparados por outro item deste mesmo objeto pois são únicos. No mundo real, podemos citar como exemplo qualquer objeto físico identificável. Já no mundo digital, toda criação é passível de se tornar um NFT [Nadini et al. 2021]. Em contrapartida, os itens ou *tokens* fungíveis podem ser trocados por outro ou outros itens de mesmo valor que não afetam assim a sua posse ou valorização. Por exemplo, as moedas em circulação nos países e que são permutáveis entre si.

A representação da propriedade de itens exclusivos por meio dos NFTs tornou-se cada vez mais comum ao longo do ano de 2021. Itens digitais ou do mundo real como criações artísticas, objetos colecionáveis, objetos proprietários e até mesmo direitos autorais podem ser identificados por um registro imutável de sua propriedade em *blockchains*. Atualmente, há inúmeros locais, chamados de mercados digitais ou marketplaces, especializados em manter NFTs, assim como plataformas de *blockchain*, oferecem recursos para sua criação e manutenção. Como por exemplo a plataforma Ethereum[3] e os mercados OpenSea[4] e Rarible[5]

Note que os NFTs surgiram juntamente com inúmeras aplicações, provenientes de diversos setores, em consequência da ampla atenção e recente exploração das *blockchains*. Nos últimos meses, as negociações sobre os NFTs alcançaram um crescimento gigantesco, registrando uma movimentação de $ 34.530.649,86 dólares em 2021[Wang et al. 2021]. O crescente volume de negociações de NFTs também impacta na forma como os indivíduos estão lidando com o dinheiro digital, já que os mercados de NFTs utilizam das criptomoedas para compra e venda de seus itens.

Alinhado a esse direcionamento, há um crescente interesse na criação de mercados digitais para a criação e negociação dos NFTs. Uma análise realizada por [Casale-Brunet et al. 2021], apresenta a movimentação de oito mercados, como exemplos de projetos NFTs, registrados na plataforma Ethereum e classificados em categorias de artes, fotos de perfil e metaverso. Estes mercados, endereçados conforme a Tabela 2.1, registraram um volume total de 315.491 ativos e movimentaram cerca de 425.245 transações entre compra, vendas e transferências.

[2]https://einvestidor.estadao.com.br/criptomoedas/investimento-nft-tendencia-famosos/
[3]`https://ethereum.org`
[4]`https://opensea.io`
[5]`https://rarible.com`

Tabela 2.1. Mercados NFTs, Categorias, Endereços dos contratos na plataforma Ethereum e suas movimentações. Adaptada de [Casale-Brunet et al. 2021].

Projeto NFT	Categoria	Endereço do contrato	Ativos	Transações
HashMasks	Arte Digital	0xc2c747e0f7004f9e8817db2ca4997657a7746928	16384	49404
Art Blocks Curated	Arte Digital	0xa7d8d9ef8d8ce8992df33d8b8cf4aebabd5bd270	57873	80660
CryptoPunks	Foto de perfil	0xb47e3cd837ddf8e4c57f05d70ab865de6e193bbb	10000	28576
Bored Ape Yacht Club	Foto de perfil	0xbc4ca0eda7647a8ab7c2061c2e118a18a936f13d	10000	32074
Acclimated Moon Cats	Foto de perfil	0xc3f733ca98e0dad0386979eb96fb1722a1a05e69	10765	14378
CryptoVoxels	Metaverso	0x79986af15539de2db9a5086382daeda917a9cf0c	14872	16607
Decentraland	Metaverso	0xf87e31492faf9a91b02ee0deaad50d51d56d5d4d	177094	174322
Meebits	Metaverso	0x7bd29408f11d2bfc23c34f18275bbf23bb716bc7	20000	292224

As primeiras menções aos conceitos ligados aos NFTs surgiram há alguns anos. Embora somente no ano de 2021 os NFTs ganharam maior atenção, na verdade, eles foram originados pelo trabalho "Overview of Colored Coins" de [Rosenfeld 2012] em 2012. Nesse trabalho surgiu a ideia de se utilizar a plataforma *blockchain* do Bitcoin para ativos colecionáveis digitais, cupons, certificação de propriedade, ações de empresas, etc. A Figura 2.1 mostra a linha do tempo com os principais fatos ocorridos em relação ao surgimento dos NFTs. Após o trabalho primário, de 2012, surgiu em 2014 a plataforma Counterparty utilizando também o Bitcoin e com o intuito de possibilitar aos usuários a criação de suas próprias moedas e ativos negociáveis. Em 2017 surgiram vários outros projetos NFTs importantes, tais como o Cryptopunks[6], CryptoKitties[7], Decentraland[8] e o mercado OpenSea.

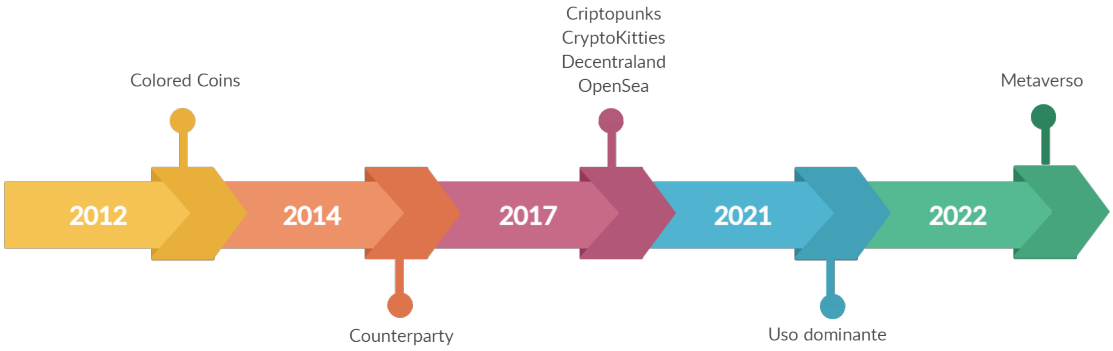


Figura 2.1. Linha do tempo histórica dos NFTs.

Os projetos CryptoPunks e CryptoKitties trabalham com o conceito de arte digital colecionável, que são ativos digitais únicos e nestes casos geradas por algoritmo e armazenados em servidores privados centralizados e com a prova de propriedade armazenada na plataforma *blockchain* Ethereum. CryptoPunks é uma coleção de 10.000 itens de imagens distintas e foi registrado na plataforma Ethereum pelo endereço de contrato

[6] https://www.larvalabs.com/cryptopunks/
[7] https://www.cryptokitties.co
[8] https://decentraland.org

0xb47e3cd837dDF8e4c57F05d70Ab865de6e193BBB. Este contrato serviu de inspiração para o padrão ERC-721 e foi precursor de outros projeto como o CryptoKitties que foi registrado no endereço 0x06012c8cf97BEaD5deAe237070F9587f8E7A266d e que pode ser facilmente verificados pela ferramenta Etherscan[9]. Neste projeto é permitido aos usuários negociar gatos virtuais exclusivos por meio do seu contrato inteligente.

O software Decentraland é uma aplicação que simula um mundo virtual compartilhado e suas transações são verificadas na plataforma Ethereum. Neste mundo virtual é possível criar conteúdos e aplicações que vão desde construções estáticas a jogos interativos. Os NFTs podem ser adicionados às construções ou utilizados nas interações criadas. A integração desta aplicação com os mercados externos de NFTs é possível devido os seus componentes serem construídos em camadas usando contratos inteligentes da plataforma Ethereum. OpenSea e Rarible são mercados para compra e venda de NFTs. Ambos possuem contratos descentralizados e armazenados na blockchain Ethereum. Um mercado NFT é uma camada de interface que torna mais fácil realizar as transações entre a blockchain e os consumidores. As ações mais comuns são criar, vender ou comprar NFTs de diferentes tipos. No entanto, por meio dos mercados também é possível descobrir novos projetos ou obter conhecimento histórico sobre a valorização e negociação de itens.

O cenário do ano de 2021 agilizou os comportamentos relacionados às interações online. Como consequência do trabalho remoto, aumento das compras online e a convivência digital, os NFTs se tornaram uma realidade com um sentido mais justificado já que muitas das interações ocorrem de maneira online. Sendo assim, o avanço desta tecnologia deve conectar esses diferentes mundos, replicando o mundo físico por meio de dispositivos digitais ao mundo virtual. Atualmente esta transição está sendo chamada de metaverso.

A importância dos NFTs para diversos segmentos é retratada através da utilização pelos mercados específicos de cada setor. Cartórios, museus, ateliês, jogos digitais e vários outros setores apresentaram interesses e projetos relacionados a certificação de propriedade utilizando NFTs. Dentre as diversas possibilidades de uso dos NFTs, destaca-se as chances de rentabilidade em suas negociações de compra e venda nos mercados específicos e genéricos. Entretanto, o inerente avanço da quantidade de mercados dos ambientes NFTs e os valores expressivos negociados, requerem soluções que permitam a sua utilização com orçamentos menores [Casale-Brunet et al. 2021]. Neste sentido, a funcionalidade de aluguel para os NFTs, isto é, o uso temporário de um NFT, demonstra ser uma funcionalidade promissora. De fato, é possível alugar com segurança um item e remunerar ao mercado e ao proprietário do *token*. O aluguel de NFTs pode trazer uma renda ao proprietário em um momento que a aquisição pode ser muito dispendiosa para um terceiro. Por parte do locatário, o aluguel de NFTs pode trazer benefícios como o de não ter que investir um grande volume de recursos financeiros na aquisição do item e sua utilização somente quando necessário. Por exemplo, um objeto de um personagem de um jogo online pode ser alugado para um jogador qualquer enquanto esse item NFT não está sendo utilizado por seu dono. Ao ser alugado, o proprietário se torna o locatário daquele bem, e recebe em troca uma rentabilidade paga pelo locador. O jogador que alugou o

[9] `https://etherscan.io`

objeto, ao deixar a plataforma do jogo, pode não ter mais interesse naquele item e assim, devolve ao proprietário original, pagando somente pelo momento que utilizou o objeto.

O ambiente que envolve a criação e negociação dos NFTs é bem mais complexo do que a simples implementação dos padrões estabelecidos para o registro e transferência de posse destes ativos. O arcabouço apresentado pela Figura 2.2 demonstra as partes envolvidas na complexa estrutura utilizada pelos NFTs. Todo o processo que vai desde a criação até a possibilidade de negociação dos *tokens*, utiliza camadas como os próprios *tokens*, os padrões estabelecidos, as bibliotecas implementadas destes padrões e as linguagens utilizadas para esta implementação. Na camada de **Linguagens**, podemos observar uma interdependência com plataforma *blockchain* a qual irá ser desenvolvida a aplicação. Por exemplo, na plataforma Ethereum a linguagem padrão para o desenvolvimento dos contratos é a Solidity. Partindo para a camada **Bibliotecas** encontramos as implementações criadas em uma determinada linguagem para os padrões já estabelecidos. O openZeppelin[10] é um exemplo de biblioteca que disponibiliza, como *Application Programming Interface* (API), interfaces de contratos exaustivamente testados. Os padrões dispostos pela camada **Padrões** descrevem um conjunto de regras para serem seguidos no desenvolvimento dos contratos de finalidades específicas. O ERC-20 e o ERC-721 são exemplos destes padrões e ditam como construir *tokens* fungíveis e não fungíveis na plataforma de *blockchain* Ethereum. Existem vários **Mercados**, camada que também é conhecida como Marketplaces, que surgiram em torno dos NFTs. Eles permitem que os usuários comprem e vendam seus *tokens* por meio de uma interface centralizada. Estes incluem OpenSea, Rarible, CrypoPunks e vários outros. A camada superior e que tem os **tokens** como finalidade, abrange a aplicação dos *tokens* fungíveis e não fungíveis e dentre estes os tangíveis e intangíveis. Para a criação e negociação dos NFTs, torna-se necessário o gasto com seu registro na plataforma e taxas pagas aos mercados. Esse pagamento utiliza normalmente um *token* fungível, como por exemplo o Ether, para efetivação das transações e gerenciados por uma aplicação de carteira. Consequentemente, para obter os valores em criptomoedas, o usuário necessita realizar um investimento com moedas ou qualquer outro ativo tangível por meio das Exchanges.

Os desafios enfrentados pelos NFTs vão além dos problemas já conhecidos para as *blockchains*. Estes desafios devem ser cuidadosamente enfrentados dada a grande aplicabilidade e movimentação financeira dos NFTs. O registro de NFTs pode ser realizado de maneira a atender qualquer tipo de dados, porém os padrões estabelecidos até aqui e as plataformas *blockchains* não fornecem suporte para armazenar conteúdos multimídia ou de grande volume. Uma vez que os conteúdos registrados como um NFT não são armazenados ou mantidos sob custódia da própria plataforma ou contrato, garantir a segurança de acesso apresenta como um desafio para os contratos de NFTs. Os ativos a qual os registros de NFTs representam, podem simplesmente serem alterados em seus locais de origem ou até mesmo não existirem mais, causando uma distorção do que foi negociado e do que se tem em posse. Entre os desafios dos NFTs, discutimos ainda a usabilidade, segurança e legislação na Seção 2.5.

Assim, o objetivo deste capítulo é apresentar e mais detalhes os principais conceitos, aplicações e desafios relacionados aos NFTs. Apresentamos uma visão teórica das

[10] `https://docs.openzeppelin.com/`

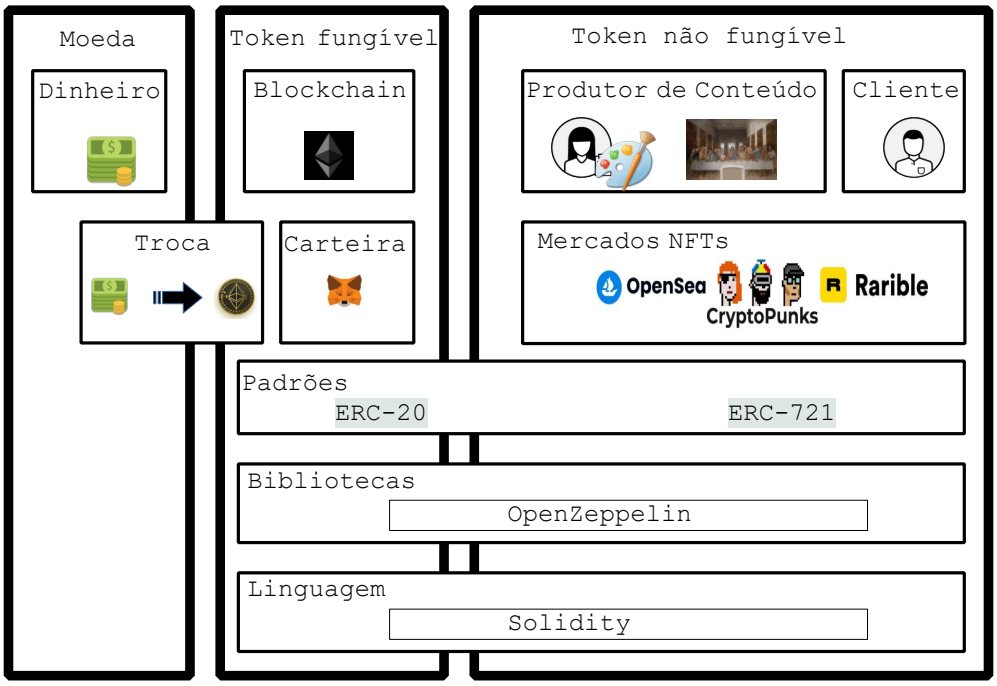


Figura 2.2. Arcabouço NFT.

tecnologias envolvidas para a criação e manutenção dos NFTs e aplicações de Marketplaces que fazem negociações dos *tokens*. Além disso, apresentamos as questões práticas para a utilização dos padrões de *tokens* e a criação de mercado para comercialização dos NFTs. Será abordada ainda a motivação para o estudo da área destacando o avanço tecnológico dos NFTs desde a sua concepção, popularização e utilização de forma autônoma em diversas aplicações. Além disso, será discutido onde e em quais situações utilizar os *tokens*, evidenciando o grande potencial dessa tecnologia em vários setores devido à sua inovação e aplicabilidade. Esse potencial não se restringe apenas a transações financeiras, sendo que há também a utilização em artes, games, certificado de propriedades e várias outras aplicações [Bao and Roubaud 2022]. Serão demonstradas as principais plataformas utilizadas para o desenvolvimento de um NFT e as aplicações descentralizadas (*decentralized applications* – DApps), que são invocadas de acordo com as condições e os códigos dos padrões de contratos NFTs [Wang et al. 2021]. O restante deste capítulo está organizado da seguinte forma. A Seção 2.2 apresenta a visão geral de todos os conceitos envolvidos para um bom entendimento dos NFTs. A Seção 2.3 define os *tokens* e os padrões criados para os mesmos na plataforma Ethereum. As questões de segurança são exploradas na Seção 2.4. A Seção 2.5 elenca os desafios de pesquisa e as perspectivas futuras em relação a usabilidade, privacidade e legislação. A prática envolvendo os padrões de *smart contract* para os *tokens*, bibliotecas e Dapps são desenvolvidas na Seção 2.6. Por fim, a Seção 2.7 conclui o capítulo.

2.2. Visão geral

Esta seção apresenta uma visão geral sobre *tokens* não fungíveis [Wang et al. 2021], cobrindo os temas dos conceitos de sistemas distribuídos, *blockchains*, consenso, padrões

de contratos e plataformas *blockchain* [Chohan 2021]. Primeiramente são apresentadas uma introdução e explicação sobre o funcionamento das *blockchains*. A seguir, são descritos os componentes que formam os contratos inteligentes, como eles trabalham, a definição de termos usados e os passos para o processo de criação e uma apresentação e discussão sobre as plataformas existentes, como Ethereum [Buterin et al. 2014] e Hyperledger [Androulaki et al. 2018]. Por fim, são definidos os conceitos de aplicações e finanças descentralizadas (Dapp, DeFi), oráculos e organizações autônomas descentralizadas (DAO). Finalmente, apresentamos exemplos de aplicações de uso dos NFTs e suas características em relação aos padrões.

2.2.1. Blockchain

Blockchain é uma tecnologia que contém funcionalidades de armazenar registros de transações de maneira imutável e distribuída. A topologia das redes *blockchains* pode se dar de diversas formas, porém na maioria das vezes são concebidas por inúmeros participantes independentes, e que não há necessidade de um controle centralizado. Os dados enviados à rede são organizados em blocos encadeados por meio de *hashes* criptográficos [Nakamoto and Bitcoin 2008]. Esses blocos são compostos por cabeçalho e uma lista de transações conforme ilustra a Figura 2.3 e que demonstra a arquitetura de um bloco e suas interligações apontadas pelas setas.

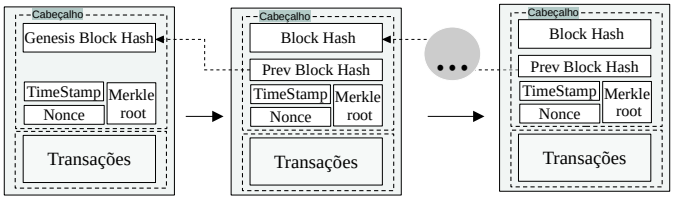


Figura 2.3. Cadeia de blocos.

Cada bloco é interligado ao outro por um endereçamento baseado em um *hash* criptográfico do bloco anterior e ilustrado pela seta pontilhada. À partir do bloco gênesis, isto é, o primeiro bloco da cadeia, o *hash* é calculado contendo o endereço do último bloco. Este processo garante que caso algum bloco da cadeia seja alterado de forma maliciosa, todo o restante será invalidado. As transações são mensagens submetidas ao consenso da *blockchain* e assinadas pelo endereço que as submeteram. Ao serem aceitas, elas alteraram o estado da rede e são armazenadas permanentemente e imutáveis. Além da lista de transações contidas dentro do bloco, o bloco contém os seguintes campos:

- ***Prev Block Hash:*** este campo faz uma referência aos pais, que é um link de um bloco ao seu anterior na cadeia. Todas as informações dentro do bloco anterior serão inseridas em uma função *hash* para obter um valor, então este valor será atribuído ao campo *Prev Block Hash* no novo bloco.
- ***Timestamp***: que contém a hora em que o bloco foi anexado.
- ***Merkle Root:*** contém o valor de *hash* de todas as transações validadas do bloco. Todas as transações são criptografadas em um valor de *hash* que representa a raiz de Merkle.

- ***Nonce:*** campo com valor que é usado pelo protocolo de consenso para comprovar o esforço despendido para anexar o bloco à cadeia.

Existem inúmeros e diversificados problemas que podem ser solucionados com a aplicação da tecnologia *blockchain*. Dentre estes problemas, podemos citar a validação de acesso, integridade e interoperabilidade de dados, rastreabilidade e até mesmo a contatação e verificação de propriedade [Gordon and Catalini 2018].

A utilização do conceito de contratos inteligentes pela tecnologia *blockchain* aumenta as possibilidades de uso e seu funcionamento. Os contratos são implementados em uma determinada linguagem de programação por meio de scripts e armazenados na rede. As regras dos contratos são executadas pela rede da forma como foram estabelecidas.

Os tipos de *blockchains* são classificados basicamente de acordo com a forma de acesso e proteção das transações armazenadas [Monrat et al. 2020]. Os tipos encontrados são as públicas, privadas e de consórcio. As especificações de protocolo de consenso, proteção de acesso às informações e controle da forma de distribuição diferenciam os tipos de *blockchain* e são apresentados na Tabela 2.2.

Tabela 2.2. Tipos de *blockchains*.

	Centralização	**Consenso**	**Acesso a Dados**
Pública	Nenhuma	Livre	Públicos
Privada	Sim	Restrito	Públicos ou restritos
Consórcio	Parcial	Restrito	Públicos ou restritos

Em uma *blockchain* privada as respostas são mais rápidas e seguras, porém o controle é exercido por um proprietário específico e os nós precisam de permissão para ingressarem na rede . A *blockchain* privada é mais rápida, eficiente e segura [Monrat et al. 2020]. Na *blockchain* pública, por sua vez, a rede é totalmente descentralizada e pode conter vários nós e qualquer nó pode se ingressar à rede [Xiao et al. 2020]. Porém, apenas nós sincronizados são utilizados para consenso. Uma *blockchain* de consórcio é composta por nós de organizações específicas que se organizam e controlam quem pode ter acesso à rede. A rede resultante do consórcio é parcialmente descentralizada [Rouhani and Deters 2017, Wang et al. 2018].

Os protocolos de consenso trabalham com algoritmos de forma distribuída, com o intuito de gerar um novo bloco e ser aceito entre os nós da rede. Os algoritmos consistem basicamente na solução de um dilema para obter um consenso entre os nós da rede sobre a validade de uma transação. O objetivo destes algoritmos é obter o consenso entre os nós validadores e ignorar qualquer tentativa de obstrução. A comparação entre os mecanismos de consenso seguem os critérios de eficácia energética, entrada de novos nós e tolerância a falhas. Um estudo mais aprofundado desses algoritmos pode apresentar mais detalhes sobre a capacidade de armazenamento e o tempo gasto para gerar os blocos em cada algoritmo. A escolha do mecanismo de consenso pode ter um impacto significativo no desempenho da solução pretendida. Portanto, deve-se considerar os requisitos da aplicação referente ao armazenamento de dados e tempo de resposta.

2.2.2. Contratos Inteligentes

Contratos Inteligentes (*Smart Contracts*) são programas auto executáveis e auto impositivos que funcionam de acordo com condições previamente acordadas [Hewa et al. 2021a]. Esses contratos são capazes de implementar determinadas operações dentro da *blockchain* e funcionam como parte das aplicações descentralizadas. Entre os benefícios da utilização de contratos inteligentes apontados por [Hewa et al. 2021a] estão a eliminação da necessidade de um terceiro confiável para efetuar transações, a integridade e a transparência das transações e a autonomia de execução e a precisão dos contratos, uma vez que são imutáveis e são executados quando as pré condições são atendidas.

Devido ao caráter complementar que os contratos possuem em relação à *blockchain*, eles se tornaram parte essencial das *blockchains* que surgiram após o Bitcoin, proposto por [Nakamoto and Bitcoin 2008]. Com a utilização desses contratos, as *blockchains* tiveram sua capacidade de armazenamento aprimorada. Um contrato permite definir um comportamento para um determinado estado e atender necessidades de aplicações diversas. Nesse sentido, os contratos inteligentes são capazes de executar transações muito mais complexas dentro da *blockchain* do que simplesmente a troca de moedas.

A partir dessas características, inúmeras aplicações podem ser desenvolvidas baseadas no uso de contratos inteligentes como, por exemplo, a certificação de propriedade feita pelos NFTs, aplicações financeiras (gerenciamento de moeda, serviço de garantia, procedimentos de auditoria, empréstimo), aplicações médicas (gestão de informações de saúde, proteção de dados de pesquisa clínica, monitoramento e tratamento automatizado de pacientes, gerenciamento de identidade e controle de acesso, proteção de dados de identidade), aplicações imobiliárias, aplicações de acordos contratuais, aplicações de internet das coisas, aplicações de serviços de telecomunicações, aplicações de gestão de logística, além de aplicações entre diferentes indústrias [Hewa et al. 2021a].

2.2.3. Plataformas *blockchains*

A evolução da tecnologia *blockchain* provocou a criação de novas plataformas e com diferentes características e parâmetros. As plataformas são basicamente a reunião de procedimentos que determinam o funcionamento da *blockchain* mediante os requisitos apresentados pelas aplicações definidas para seu uso. As características ou parâmetros específicos propostos pelas plataformas compreendem em qual domínio ela irá operar e qual tipo de *blockchain* ela implementa. As plataformas são do tipo permissionada ou não permissionada. O mecanismo de consenso adotado pela plataforma também a caracteriza juntamente com a capacidade e suporte de uso de *smart contracts* [Monrat et al. 2020]. A seguir, mencionamos duas destas plataformas e que estão diretamente envolvidas com o desenvolvimento dos NFTs.

Ethereum é uma plataforma popular do tipo pública, livre de permissão e possui o Ether como moeda principal [Buterin et al. 2014]. Ela possui tecnologia que possibilita o desenvolvimento e execução de contratos inteligentes. O uso deste recurso resulta na implementação das chamadas aplicações descentralizadas (DApps) que são executadas dentro da *blockchain*. Nesta plataforma é necessário o gasto de Ethers para realizar a inserção de um Contrato Inteligente que é utilizado como recompensa ao responsável por criar o bloco deste contrato. Por sua vez, uma rede *blockchain* permissionada

se mostra como uma opção à exposição pública da plataforma. Hyperledger Fabric é uma plataforma privada e permissionada que possibilita o desenvolvimento de aplicações que equilibram os gastos de custo e desempenho em relação às redes *blockchain* públicas [Cachin et al. 2016]. Por ser privada, a escrita de transações é controlada por um grupo de membros da rede, e o acesso à leitura de dados e transações são limitados aos membros e usuários autorizados. Neste caso, por ser uma rede permissionada, os usuários da rede são controlados por autenticação de identidade. Os contratos inteligentes são definidos como Chaincode e as chamadas a ele é que podem alterar o estado da rede. O consenso é realizado por uma implementação do algoritmo *Practical Byzantine Fault Tolerance*.

2.2.4. Carteiras

Uma carteira representa um par de chaves que compreende em uma chave pública e uma chave privada [Borkowski et al. 2019] . Normalmente, em *blockchain* é utilizada como um ambiente que permite aos usuários terem controle sobre as transações realizadas com suas criptomoedas e *tokens*. Por meio de uma aplicação de carteira é que o usuário de sistema *blockchain* consegue ingressar à rede e realizar transações [Hasanova et al. 2019]. A inserção e validação de transações normalmente é feita por uma aplicação de carteira que necessita da assinatura das transações por meio da chave privada do usuário. Este processo de assinatura é responsável por decifrar as transações utilizando a chave pública do usuário, procedimento denominado criptografia assimétrica, que proporciona segurança ao constatar a veracidade dos autores das transações [Pal et al. 2021].

A aplicação de carteira deve conter atributos que consideram principalmente a segurança e anonimato do usuário. Dentre estes atributos estão o controle, validação, transparência, vulnerabilidade e privacidade. Sendo que a contenção dos ativos devem ficar no controle do próprio usuário, não havendo possibilidade de congelamento ou perda dos mesmos [Borkowski et al. 2019]. A verificação e validação das transações deve ser realizada pela carteira sem processamento por terceiros, tornando-a independente. O código-fonte da carteira deve ser transparente ao ponto de possibilitar a auditoria e verificação das funções executadas. As vulnerabilidades devem ser contempladas no projeto ao ponto de indicar as fraquezas e solucioná-las. Quanto à privacidade, a carteira deve dispor de ferramentas que inibem o rastreamento do usuário e transações por meio, por exemplo, de anonimização e roteamento de endereços [Pillai et al. 2019].

2.2.5. Oráculos, Aplicações e Finanças descentralizadas

As aplicações descentralizadas ou *Decentralized Applications* (DApps) são um tipo especial de aplicação que são desenvolvidas baseadas em contratos inteligentes. Em outras palavras, os DApps são softwares que dependem de contratos inteligentes implantados em uma plataforma *blockchain*.

As finanças descentralizadas ou *Decentralized Finance* (DeFi) são uma opção ao sistema controlado por processos centralizados e mantidos por instituições financeiras. A DeFi oferece um controle transparente e descentralizado mediante um sistema aberto e criado especificamente para a era digital.

Por muitas vezes, as DApps necessitam recuperar dados externos a uma plata-

forma *blockchain*. Porém os contratos inteligentes são invocados por eventos ou funções externas por participantes do *blockchain* e não são capazes de realizar a recuperação de dados externos. Isto é, extrair dados externos ao da plataforma onde o contrato está implantado não é uma tarefa trivial. Portanto, os dados externos devem ser inseridos nos contratos. Os oráculos são basicamente uma ponte entre a parte interna e externa da plataforma *blockchain*. Estes oráculos precisam necessariamente advir de fontes confiáveis uma vez que inserem um nível de insegurança e complexidade nas DApps. Dado que este processo de consulta aos oráculos torna esta ação centralizada, é aplicado um consenso entre os oráculos para definir a confiança da informação advinda do processo [Stradling and Voorhees 2018].

2.2.6. Organizações autônomas e descentralizadas

Uma organização autônoma descentralizada, (*Decentralized Autonomous Organizations* – DAOs), é constituída de regras e princípios de governança automatizados por contratos inteligentes transparentes, são controlados pelos membros da organização, mas sem interferência centralizada [El Faqir et al. 2020]. O objetivo principal das DAOs é de criar uma forma de governança coletiva utilizando uma infraestrutura descentralizada. As regras e princípios são comumente escritas em linguagem de programação para contratos inteligentes e implantadas em uma *blockchain* pública. Os membros pertencentes a uma organização votam para o consenso de uma determinada proposta em um determinado tempo. O resultado da aprovação da proposta poderá ser consultado após processo da determinação do consenso. As DAOs podem resolver o problema de confiança no compartilhamento de NFTs com outros e reduzir o processo de colaboração remediando falhas da colaboração tradicional e permitindo a colaboração em escala [Jentzsch 2016].

2.2.7. Cenários e aplicações de NFTs

A inovação e criatividade no mundo dos NFTs tem realizado ainda mais avanços aos padrões e as plataformas que os implementam. À medida que o universo NFT evolui, a tecnologia é adotada por diversos cenários e são produzidas inúmeras aplicações que entregam soluções interessantes. Podemos citar vários exemplos de uso interessante que envolvem experiências de jogos *blockchain*, arte digital, registro seguro, propriedade de objetos/colecionáveis digitais e físicos, direito autoral e muito mais. Os casos de uso que envolvem em seus planejamentos o aluguel de NFTs trazem diversos benefícios em seus cenários e podem ser integrados aos mercados. Alguns exemplos reais de cenários que utilizam NFTs incluem:

- Arte digital

  A possibilidade de aluguel de itens para exposições de arte no mundo virtual. Sendo que este possa ter o tempo controlado e/ou expirado por contratos automatizados. A exibição nestes expositores pode ainda ser realizada de forma distinta e selecionada, gerando um rendimento para itens de arte menos conhecidos.

- Metaverso

  O aluguel de terrenos virtuais pode ser comparado ao modo realizado no mundo real. Possibilidades de aluguel de espaço em anúncio virtual são inseridas neste

cenário onde podem ser explorados espaços por tempo de exibição e até mesmo baseado em tráfego naquele local.

- Jogos

  No cenário de jogos já é muito comum a existência de itens escassos e raros que podem aumentar a capacidade e a jogabilidade. Assim estes itens alugáveis podem ajudar a avançar ou superar certos desafios dos jogos em um determinado momento ou fase.

- Domínio (URL)

  Um nome registrado em espaços virtuais pode ser explorado sob a demanda por eles sem a necessidade de venda ou troca de titularidade permanente. Resultando em uma renda passiva por meio do aluguel.

- Mercados

  Por se tratar de uma extensão do padrão ERC-721, a funcionalidades de aluguel dos NFTs podem ser diretamente integradas aos mercados preexistentes como por exemplo o OpenSea.

- Pessoa pública

  As pessoas públicas estão criando NFTs e disponibilizando às suas comunidades para permitirem benefícios como, por exemplo, acesso a fã clubes, exclusividades em visualização de conteúdos e descontos em produtos de marca própria.

2.3. *Tokens*

Nesta seção tratamos dos conceitos de *tokens* e os aspectos das estruturas de um *token*, tais como seus componentes e padrões. Descrevemos os *Ethereum Request for Comments* (ERC), tais como ERC-20, ERC-721 e ERC 1155, que são padrões de contratos criados no nível de aplicação no ecossistema Ethereum para *tokens* e propostos nos *Ethereum Improvement Proposals* [Wang et al. 2021]. Eles são um fator importante para o sucesso dos NFTs pois são padrões abertos que descrevem como construir *tokens* não Fungíveis em *blockchains* e possui um conjunto de regras que facilitam o trabalho do desenvolvedor. Também descrevemos as formas que os padrões podem ser utilizados para troca de propriedade e serem negociados por outros *tokens* ou criptomoedas.

Token pode ter o significado de uma ficha ou um código que representa algo. Na tecnologia, a palavra *token* é atribuída a dispositivos ou sistemas que geram codificações de acesso e autenticação. Este termo significa a representação digital de um ativo quando no universo de uma *blockchain*. Portanto é utilizado para descrever a criação de um registro digital de um ativo tangível, como por exemplo o ouro, dinheiro, obras de arte, imóveis, equipamentos, etc. Ou ainda um ativo intangível como por exemplo software, criptomoedas, artes digitais, etc.

Sendo assim, conforme representados na Figura 2.4, os ativos tangíveis ou intangíveis podem ser protegidos com a ajuda da tecnologia *blockchain* e representados como um ativo digital por um *token* fungível ou não. Podendo representar dinheiro, certificados

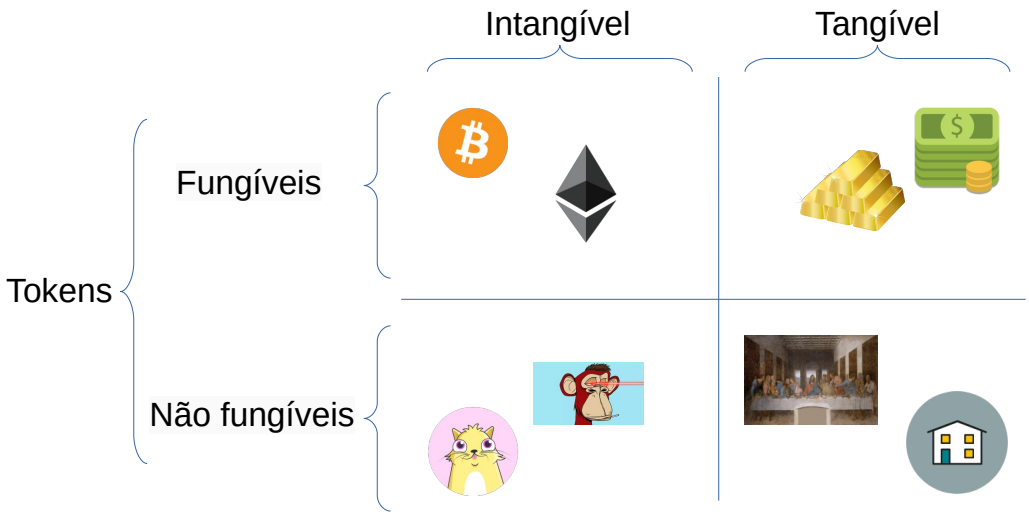


Figura 2.4. *Tokens* Fungíveis e Não fungíveis

de propriedades e outros ativos. As próprias criptomoedas podem ser consideradas como *tokens*. Existem, portanto, vários tipos de *tokens* e se distinguem de acordo com a proposta a qual forem designados. Dentre eles os principais tipos são: *tokens* fungíveis ou de pagamentos, utilidades, não fungíveis e de segurança.

- O *token* fungível (FT) ou de pagamento exerce o papel de dinheiro eletrônico e são operados em transferência e ou pagamentos. Os exemplos mais conhecidos deste tipo de *token* são o Bitcoin e o Ether.

  Definição 1. *Um FT é idêntico a todos os outros tokens em valor, para a mesma classe de tokens e é igualmente intercambiável com qualquer outro dentro de uma determinada classe.*

- Os *tokens* de utilidades são usados no oferecimento de algum benefício ao cliente ou usuário de um serviço ou produto. Eles fornecem utilidades como descontos, acessos a serviços especiais dentre outros.

- Os NFTs, são *tokens* que podem representar algo singular, i.e, não pode ser trocado por outro de mesmo valor como dinheiro porque é único. Eles correspondem à identificação de um item único e vinculado a um proprietário. Em suma, um NFT é um código que contém o registro do objeto e seu proprietário. A maioria dos projetos de NFTs são originados da plataforma Ethereum ou a utilizam para prover garantias. No entanto, existem uma gama de outros projetos baseados em plataformas diferentes e que demonstram a independência da utilização dos NFTs de qualquer plataforma[11].

  Definição 2. *Um NFT é um token exclusivo dentro de uma determinada classe de token e não é igual a nenhum outro NFT dentro de uma determinada classe.*

[11] https://chaindebrief.com/non-ethereum-nft-projects-zilliqa-solana/

- Os *tokens* de segurança são uma nova modalidade de propriedade de ações, títulos e produtos regulados por instituições financeiras. Representam uma participação externa em um ativo e podem ser emitidos em *blockchain* públicas ou permissionadas.

2.3.1. Estrutura e Padrão de *tokens* na plataforma Ethereum

A estrutura de um *token* na plataforma blockchain Ethereum é realizada fundamentalmente em um contrato inteligente. Esta estrutura contém basicamente um mapa de endereços de contas e seus saldos vinculados. O saldo, chamado de *token*, simboliza um valor, que também pode pode ser utilizado na representação de objetos, valores monetários e outras finalidades, conforme mencionadas nos itens da seção anterior. As operações sobre esta estrutura discorrem em transferências da propriedade destes *tokens*. Ao realizar esta transferência, os contratos atualizam o saldo das duas contas envolvidas, creditando-o em uma das contas e debitando na outra.

Os padrões de *tokens* foram sugeridos inicialmente na plataforma Ethereum. Na plataforma Ethereum, quando surge uma nova proposta e que altera um procedimento deve-se fazer por meios de uma Proposta de melhorias no ethereum ou *Ethereum Improvement Proposal* (EIP). Esta proposta consiste em um documento que detalha as alegações de mudanças e as questões técnicas envolvidas. Por meio dos EIPs é possível submeter proposições de comentários, mudanças em protocolos e outros detalhes da Ethereum. No caso dos *tokens*, os documentos são submetidos como solicitação de comentários Ethereum ou *Ethereum Request for Comments* (ERC) que é a forma de definir padrões de uso na Ethereum. As EIPs submetidas passam por um ciclo de vida e recebe um status em cada etapa, que vai desde o rascunho da ideia até a versão final. As EIPs que tratam de conteúdos técnicos e padrões referentes aos contratos inteligentes passarão a ser nomeadas com a sigla ERC após a aprovação na fase final. Existem vários EIPs/ERCs que foram submetidos e aprovados como padrões por meio deste ciclo e que são comumente utilizados para definir tipos de *tokens*. Dentre eles citamos os padrões ERC-20, ERC-721 e ERC-1155 que tratam dos *tokens* fungíveis e *tokens* não-fungíveis respectivamente.

Inicialmente o padrão ERC-20 foi introduzido como uma experiência de proporcionar recursos para padronização de criação de contratos de *tokens*. Este padrão trouxe muitos benefícios, permitindo uma integração de vários *tokens* com uma mesma carteira e listagem em plataformas digitais onde é possível comprar, vender, trocar e guardar os *tokens*. Estes benefícios avançaram para outros padrões de outros tipos de *tokens* conforme apresenta a Tabela 2.3.

2.3.2. Padrão ERC-20

O ERC-20 é um padrão de contratos inteligentes que permite a criação de *tokens* fungíveis no Ethereum, isto é, *tokens* que são iguais e possuem o mesmo valor independente do índice que o representa. O padrão foi proposto por [Vogelsteller and Buterin 2015]. Por meio da implementação da interface deste padrão é possível utilizar inúmeras das funcionalidades pré-estabelecidas. Tais funcionalidades como as de transferência de *tokens* entre contas, verificação o saldo da conta, verificação do total de *tokens* criados e aprovação de *tokens* para utilização de terceiros possibilitam a reutilização e compatibilidade por aplicações como carteiras e exchanges descentralizadas. Há vários *tokens* já implantados

Tabela 2.3. Padrões de *tokens* Ethereum com casos de uso

Padrão de *token*	Definições	Utilidade
ERC 20	Baseado em saldos Valor fungível	Oferta inicial de criptomoedas *token* de segurança *Tokens* de utilidade
ERC 721	Valor não fungível Cada elemento é único	Tokenização de ativos reais Registro de propriedade de ativos colecionáveis e virtuais
ERC 155	Combinação dos padrões ERC-20 e ERC-721 Permite que diferentes *tokens* sejam configurados de um único ponto	Conjunto de ativos heterogêneo

e seguindo o padrão ERC-20 na rede Ethereum. Podemos encontrar em [Charles 2013], implementações de exemplo e que são utilizadas pelas Dapps. A Listagem 2.1 apresenta alguns dos métodos deste padrão e que são relevantes para nossa discussão e prática.

```
1  // SPDX-License-Identifier: UNLICENSED
2  pragma solidity ^0.4.20;
3  interface ERC721{
4  event Transfer(address, address, uint256)
5  event Approval(address, address, uint256)
6  function name() public view returns (string)
7  function symbol() public view returns (string)
8  function decimals() public view returns (uint8)
9  function totalSupply() public view returns (uint256)
10 function balanceOf(address) public view returns (uint256)
11 function transfer(address, uint256) public returns (bool)
12 function transferFrom(address, address, uint256) public returns (bool)
13 function approve(address, uint256) public returns (bool)
14 function allowance(address, address) public view returns (uint256)
15 }
```

Listagem 2.1. Padrão para *token* fungível: EIP-20

2.3.3. Padrão ERC-721

O padrão ERC-721, também conhecido como o padrão de NFTs, é utilizado na identificação de itens únicos e exclusivos. Este tipo de padrão fornece a capacidade ao *token* de realizar funções como certificação de propriedade, identificação de credenciais, identificação de acesso, ingressos para eventos, itens colecionáveis, etc.

Os NFTs possuem as características básicas do *token* ERC-20, porém apresentam um identificador exclusivo para cada registro tornando-os únicos e exclusivos. Uma interface padrão para os NFTs foi apresentada por [Entriken et al. 2018]. Este padrão pode ser considerado como um certificado de posse que permite a implementação de uma API padrão para NFTs utilizando os contratos inteligentes. As funcionalidades básicas para rastrear e transferir NFTs são apresentadas na Lista de código 2.2.

```
1  // SPDX-License-Identifier: UNLICENSED
2  pragma solidity ^0.4.20;
3  interface ERC721{
4    event Transfer(address, address, uint256);
5    event Approval(address, address, uint256);
6    event ApprovalForAll(address, address, bool);
7    function balanceOf(address) external view returns (uint256);
8    function ownerOf(uint256) external view returns (address);
9    function safeTransferFrom(address, address, uint256, bytes) external payable;
```

```
10    function safeTransferFrom(address, address, uint256) external payable;
11    function transferFrom(address, address, uint256) external payable;
12    function approve(address, uint256) external payable;
13    function setApprovalForAll(address, bool) external;
14    function getApproved(uint256) external view returns (address);
15    function isApprovedForAll(address, address) external view returns (bool);
16 }
```

Listagem 2.2. Padrão para *token* não fungível: EIP-721

2.3.4. Padrão ERC-1155

O padrão ERC-1155 é chamado com padrão multi-token e foi idealizado para controlar um número maior de *tokens* simultaneamente. Para abranger *tokens* fungíveis e não fungíveis, o ERC-1155 contém as funções do *token* ERC-20 e do ERC-721 [Radomski et al. 2018]. Por meio de uma interface padrão utilizada por um único contrato inteligente é possível gerenciar vários tipos de *token*. As funcionalidade básica desta interface são apresentadas na Lista de código 2.3.

```
1  // SPDX-License-Identifier: UNLICENSED
2  pragma solidity ^0.5.9;
3  interface ERC1155
4      event TransferSingle(address, address, address, uint256, uint256);
5      event TransferBatch(address, address, address, uint256[], uint256[]);
6      event ApprovalForAll(address, address, bool);
7      event URI(string, uint256);
8      function safeTransferFrom(address, address, uint256, uint256, bytes) external;
9      function safeBatchTransferFrom(address, address, uint256[], uint256[], bytes) external;
10     function balanceOf(address, uint256) external view returns (uint256);
11     function balanceOfBatch(address[], uint256[]) external view returns (uint256[]);
12     function setApprovalForAll(address, bool) external;
13     function isApprovedForAll(address, address) external view returns (bool);
14 }
```

Listagem 2.3. Padrão para Multi *token*: EIP-1155

Os padrões ERC-20 e ERC-721 requerem a implantação de contratos distintos por cada tipo de *token*. O padrão ERC-721, por exemplo, contém um ID do *token* que é um índice que representa um item não fungível desse grupo e é implantado como um único contrato que abrange configurações para todo o grupo de *tokens*. Já o ERC-1155 permite que cada ID de *token* represente um novo tipo de *token* configurável, que pode ter seus próprios metadados, funções e atributos. Tarefas como a transferência de vários tipos de *token* simultaneamente são possíveis por meio das funcionalidades adicionadas a este padrão resultando em economia nos custos de transação.

2.3.5. Negociação de *tokens*

Os *tokens* ERC-721 são negociados de forma diferente das criptomoedas e outros tipos de *tokens*. O valor dos NFTs geralmente depende de sua relação fora da *blockchain* em que são negociados e também a raridade imposta a eles [Rogers et al. 2022]. Alguns NFTs só podem ser comprados com criptomoedas, portanto, é necessário possuir parte dessa criptomoeda e armazená-la em uma carteira digital. A partir daí é possível comprar NFTs por meio de qualquer um dos mercados NFT online, incluindo OpenSea, Rarible e Cryptopunks. Quando ocorre uma transferência de *tokens* fungíveis, por exemplo do tipo ERC-20, são submetidos ao consenso da rede o débito e o crédito nos saldos das contas envolvidas na transação. Já a transferência de NFTs, isto é, o *token* não fungível, ocorre

a mudança da propriedade daquele registro individual e que não será agregado a nenhum outro *token*. A Figura 2.5 apresenta duas funções executadas na transferência e obtenção de dados de um NFT em forma sequencial.

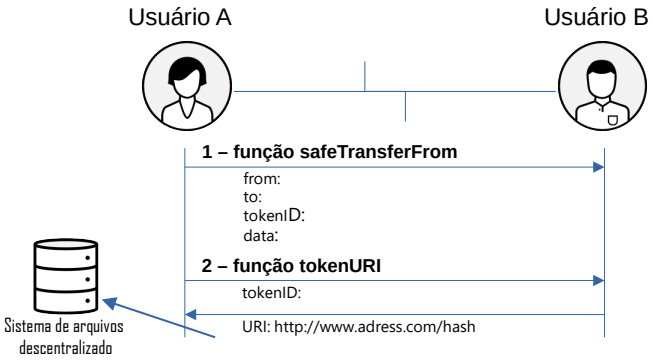


Figura 2.5. Sequência para transação de transferência de NFTs.

Neste caso, os usuários A e B realizam uma transferência da propriedade de um *token* com um ID específico. Onde o usuário A é o atual proprietário do *token* e o usuário B é quem receberá a propriedade do *token*. A transferência do *token* só se dá por meio da execução da função ***saveTransferFrom*** acionado pelo proprietário do *token*. Esta função recebe os dados de identificação do atual proprietário, identificação do usuário para quem vai ser transferido o *token*, juntamente com a identificação do *token* no contrato. A função ***tokenURI*** solicita o retorno do metadado URI armazenado ao criar o registro do *token*.

A compra e venda de NFTs deve ser de comum acordo entre as partes envolvidas, porém a efetivação da transferência só é realizada pelo vendedor. Apesar da iniciativa da compra ser comumente feita pelo comprador nos mercados de *tokens*, o comprador tem simplesmente a função de pagar o valor combinado pela transação. Normalmente os marketplaces realizam a intermediação de todo o procedimento da transferência e o pagamento ao proprietário do NFT.

O aluguel é uma outra modalidade e possibilidade de transação com os NFTs em que é permitido alugar e realizar o pagamento do aluguel de NFTs da mesma forma da venda. Porém, neste caso o NFT recebe o status de alugado, onde o usuário que o alugou pode usá-lo, mas não pode transferi-lo. Somente quando o aluguel terminar e o NFT voltar ao seu proprietário, o NFT poderá ser transferido novamente.

Como o pagamento nestes mercados é realizado normalmente por meio de criptomoedas, existem também mercados de criptomoedas, também chamados de exchanges, que oferecem aos usuários o recurso de poderem comprar e vender as criptomoedas. Este serviço é normalmente incrementado pelas exchanges para oferecerem trocas por moedas correntes tais como, por exemplo, o dólares ou euro. Há três tipos de exchanges de criptomoedas: exchanges centralizadas que são dirigidas por uma empresa ou organização, exchanges descentralizadas que fornecem processos automatizados para negociações entre o próprios usuários e exchanges híbridas que combinam as duas opções [Xia et al. 2020]. Os usuários se dispõem de saldos em sua aplicação de carteira, ao adquirirem as criptomoedas, que poderão ser utilizadas para negociação dos NFTs em seus marketplaces.

2.4. Segurança

Nesta seção serão apresentados os aspectos principais relacionados à segurança no contexto de *smart contract* e consequentemente os NFTs. De fato, nos últimos anos, diferentes estudos tratam das pesquisas relacionadas à segurança nos ambientes voltados aos contratos inteligentes [Rouhani and Deters 2019, Harz and Knottenbelt 2018], discutindo a importância e constante evolução deste aspecto. O conceito de segurança em uma rede está diretamente relacionado aos métodos de ataque, de modo a obter informações sensíveis dos elementos e do próprio estado da rede, e também aos mecanismos de defesa a esses ataques. Além disso, um aspecto fundamental para ambos é a privacidade dos usuários ativos na rede, garantindo que dados sensíveis de identificação e localização sejam preservados [Sayeed et al. 2020].

É importante salientar que os contratos inteligentes para NFTs sejam exaustivamente testados. Os prejuízos causados por falhas podem ser catastróficos se os contratos inteligentes contiverem erros. Portanto, devem ser testados para garantir que façam exatamente o que foi proposto e sem falhas. A qualidade e integridade dos testes devem passar por uma eficiente averiguação de eficácia e demonstrar as possíveis vulnerabilidades existentes [Hartel and Schumi 2020, Hewa et al. 2021b].

2.4.1. Avaliação de segurança

Os NFTs geralmente são seguros, uma vez que usam a tecnologia *blockchain* assim como as criptomoedas. A natureza distribuída das *blockchains* torna os NFTs difíceis de serem atacados. Porém existem vários problemas de segurança que permitem que mineradores ou usuários maliciosos os explorem e ganhem lucro de maneira injusta [Luu et al. 2016].

Boas práticas devem ser seguidas ao criar contratos inteligentes para NFTs para que sejam seguros. Com isso é possível identificar os ataques comumente utilizados e garantir que *tokens* não sejam perdidos devido a erros de programação. Alguns problemas de segurança encontrados em contratos inteligentes e que são comumente explorados por usuários mal intencionados estão descritos abaixo.

Dependência de ordem de transação - Um ataque de dependência da ordem de transação pode gerar um resultado inesperado para os usuários simplesmente se houver invocações simultâneas. Também pode ocorrer de um usuário mal-intencionado explorar as dependências de ordem dos contratos para obter vantagens. Nestes casos, a invocação do contrato para alterar o estado da rede pode não conhecer individualmente o estado, uma vez que outras transações também podem ter sido invocadas. Dado que o minerador do bloco é quem decide a ordem das transações, e consequentemente a ordem das atualizações, o estado final de um contrato pode depender da ordem em que foram invocadas as transações. Exemplificando, caso um item NFT seja anunciado por um preço, o usuário espera pagar esse preço pelo item. Uma tentativa desse ataque poderia alterar o preço do item antes do processamento da transação de compra. Isso demonstra que este contrato depende de um valor que pode ser alterado de acordo com a ordem das transações.

Dependência de *Timestamp* - Este problema de segurança afeta os contratos que utilizam o *timestamp* do bloco como condição para executar alguma tarefa. A definição de data e hora é realizada pelo nó minerador, sendo que este pode variar o valor de tempo

em aproximadamente 900 segundos, que mesmo assim o bloco pode ser validado na rede pelos mineradores. Um exemplo de contrato dependente de *timestamp* é o contrato de aluguel apresentado na Seção 2.6.5. Nele é possível observar que a função de finalizar o contrato pode se dar automaticamente dado um determinado período de expiração baseado no *timestamp* do bloco. Caso o valor do *timestamp* seja determinado de maneira a estender ou encurtar o tempo, dependendo da aplicação e cenário envolvido, os resultados podem ser diferentes do esperado para esta função.

Dependência de terceiros - Um outro grande risco de segurança para os NFTs está relacionado diretamente a confiança nos mercados que comercializam os *tokens*. Existe então a possibilidade de perda do acesso ao *token* se o mercado que hospeda o NFT deixar de existir ou não houver maneira de entrada na plataforma, seja por perda de dados das credenciais de acesso ou mesmo por acesso malicioso, o registro do *token* ficará inutilizado.

Sabemos que a *blockchain* é uma tecnologia dita como segura, mas o que se usa para comunicar com a *blockchain* pode não ser tão seguro assim. A camada de interface normalmente não é implementada de forma distribuída. Na maioria dos casos, estas interfaces são sites comuns ou com apenas algumas variações de tecnologia semelhante a isso. Sendo assim, o desenvolvedor da aplicação deverá implantar uma interface que será utilizada pelos usuários. Consequentemente, um nó será necessário para conectar essa interface à *blockchain*. Desta forma os desenvolvedores são e dependem de entidades centralizadas já que normalmente não executam seus próprios nós da *blockchain* e não criam suas interfaces específicas para cada aplicação. As questões relacionadas às interfaces para a *blockchain* não significa que haja problemas fundamentais com a própria *blockchain*. Isto apenas demonstra que há uma vulnerabilidade em camadas superiores e que devem ser levadas a sério como uma oportunidade disruptiva para que sejam construídas de maneiras mais robustas.

2.4.2. Mecanismos de defesa

A principal característica de segurança apresentada aos usuários pela *blockchain* é a transparência em que as transações são dispostas. Seja qual for o usuário, no caso de redes públicas, usuários sem necessidade de autorização, e em redes privadas, os usuários autorizados, eles conseguem auditar os dados históricos. Desta forma é extremamente difícil esconder algo na *blockchain*, necessitando da criptografia como requisito essencial de segurança dos dados para o funcionamento da tecnologia [Johnson et al. 2019].

A Tabela 2.4 apresenta alguns possíveis problemas de segurança e medidas de defesa que podem ser tomadas para sanar os problemas.

2.5. Desafios

Aqui discutimos os desafios de pesquisa e também os desafios da tecnologia além de algumas possíveis direções de pesquisa. Dada a grande aplicabilidade dos NFTs, a pesquisa apresenta várias possibilidades, desde desenvolvimento de técnicas para melhorar o desempenho até padronizações para prover verificação.

Como desafios de pesquisa encontramos ainda a necessidade de melhorias nos

Tabela 2.4. Potenciais problemas de segurança e soluções correspondentes de NFTs. Adaptada de [Wang et al. 2021]

Vulnerabilidade	Questão de segurança	Solução
Falsificação (Autenticidade)	• Um invasor pode explorar vulnerabilidades de autenticação • Um invasor pode roubar a chave privada de um usuário.	• Uma verificação formal do contrato inteligente. • Usar uma carteira para evitar o vazamento de chave privada.
Adulteração (Integridade)	• Os dados armazenados fora do *blockchain* podem ser manipulados.	• Envio de dados originais e dados de *hash* para o comprador de NFT ao negociar NFTs.
Repúdio (Não repudiabilidade)	• Os dados de *hash* podem ser vinculados ao endereço de um invasor.	• Uso parcial de um contrato com várias assinaturas.
Divulgação de informação (Confidencialidade)	• Um invasor pode facilmente explorar o *hash* e a transação para vincular um determinado comprador ou vendedor de NFT.	• Usar contratos inteligentes que preservam a privacidade em vez de contratos inteligentes para proteger a privacidade do usuário.
Negação de serviço (Disponibilidade)	• Os dados NFT podem ficar indisponíveis se o ativo for armazenado fora do *blockchain*.	• Usando a arquitetura *blockchain* híbrida com algoritmo de consenso fraco.
Elevação de privilégio (Autorização)	• Um contrato inteligente mal projetado pode fazer com que os NFTs percam tais propriedades.	• Uma verificação formal dos contratos inteligentes.

processos de desenvolvimento, como mais suporte para depuração, garantia de segurança, melhores ferramentas para facilitar o desenvolvimento e as dificuldades em realizar testes nas plataformas existentes [Zou et al. 2019]. O alto custo de falhas e a dificuldade de mudanças após implantados, torna os NFTs um desafio ainda maior [Dolgui et al. 2020]. Os direcionamentos e esforços de pesquisas para solucionar os desafios enfrentados pelos NFTs percorre dentre estes outros a revogação de certificados e *tokens*, geração de números aleatórios, paralelização de códigos e execução de aplicações em dispositivos com baixa capacidade de processamento e memória [Kemmoe et al. 2020]. Apesar de ser possível registrar um NFT para qualquer tipo de dados, os padrões estabelecidos até aqui e as plataformas *blockchains* não foram projetados para armazenar conteúdos multimídia e de grande volume. Isso apresenta como um desafio para como os NFTs mantêm protegidos algo que não está sob sua custódia. Entre os desafios da tecnologia, apresentamos a usabilidade, segurança, legislação e eficiência energética. Além destes desafios, os NFTs ainda são sujeitos aos desafios de escalabilidade e interoperabilidade que são inerentes à *blockchain*.

2.5.1. Usabilidade

O trabalho para aumentar a usabilidade e a eficácia das plataformas *blockchains* necessitam ainda de muitos esforços. Várias questões relacionadas a facilidade de uso da tecnologia precisam ser melhor estudadas para permitirem a viabilidade de uso eficaz das aplicações que demandam de muitos recursos para proverem uma interação facilitada [Pillai et al. 2017].

No contexto dos NFTs, a interoperabilidade entre plataformas contribui para a usabilidade ao implicar a capacidade de transferir um ativo entre mercados distintos, mantendo o estado consistente. A interoperabilidade de mercados deve atingir a eficiência de dois tipos, cada um dos quais trazendo considerações distintas porém contribuindo para a usabilidade. A troca de ativos digitais entre os mercados é um dos tipos de interoperabilidade. Ele deveria conter a capacidade de transferir e trocar ativos originários de diferentes mercados sem intermediários confiáveis, como trocas centralizadas. Um exemplo disso seria tornar um NFT originário do mercado Criptopunks negociável em qualquer outro mercado disponível. Outro tipo de interoperabilidade desejada se diz respeito a troca de

informações que mantém a capacidade de fazer algo em um mercado que reflete em outro mercado NFT. Esta troca deve permitir o rastreamento não só de ativos ou itens negociáveis, mas também as operações executadas. Como exemplo, o compartilhamento do histórico de transações de um determinado item contendo valores e negociação.

2.5.2. Privacidade

O risco de exposição de dados é iminente em tecnologias que utilizam a internet como meio de comunicação, uma vez que ficam suscetíveis a ameaças de segurança e privacidade. Sendo a segurança e a privacidade quesitos importantes para aplicações, como NFTs, que envolvem recursos financeiros, estas aplicações demandam mecanismos eficientes para o controle e validação de credenciais, integridade e interoperabilidade de dados. Nesse contexto, a utilização de *blockchain* torna-se atraente, pois essa tecnologia proposta por [Nakamoto and Bitcoin 2008], contempla a inibição de inúmeras vulnerabilidades e pode ser utilizada para garantir segurança e auditabilidade dos dados. As *blockchains* permitem o armazenamento seguro e imutável de dados, com aplicação direta na resolução de problemas de segurança e privacidade.

As pesquisas relacionadas à privacidade ainda não obtiveram aplicabilidades relacionadas ao cenário dos NFTs devido ao alto custo computacional das soluções existentes. No entanto, é necessário prover formas menos custosas para a disponibilização destes métodos, favorecendo assim a adesão pela utilização da tecnologia no intuito de melhorar a segurança e privacidade do usuário [Wang et al. 2021].

2.5.3. Legislação

Existem questões legais em relação aos direitos autorais e proteção de dados que precisam ser abordados como desafio desta aplicação de contratos inteligentes. Os NFTs dificilmente incluem informações relacionadas ao licenciamento ou transferência dos seus direitos [Çağlayan Aksoy and Özkan Üner 2021]. Isto se deve ao fato de o padrão ERC-721 fornecer a possibilidade de apenas criar um link entre o registro do *token* feito pelo criador do NFT e o conteúdo digital a ser representado quando implementado desta forma. Assim, o conteúdo digital é representado no contrato NFT apenas como um índice do *token* e um link para o conteúdo ou item que é representado pelo *token* e são armazenados fora da *blockchain*, normalmente em uma URL pública.

Uma questão que se pode ter é como a legislação pode proteger a criação e venda de um NFT em uma plataforma digital e se ela pode ser considerada como um trabalho artístico. De fato, no direito existe legislação que responde às criações do mundo artístico e somente com o passar do tempo é que será possível verificar se os NFTs serão considerados obras artísticas cabíveis de proteção ou não. Outra questão é que o ato de comprar uma NFT não transfere o direito autoral deste. Sendo assim, a compra transfere apenas o direito de posse, embora seja possível o detentor dos direitos autorais conceder ao comprador uma licença para determinados usos da obra. Muitos questionamentos são realizados acerca dos NFTs, porém é sabido que os instrumentos digitais só tem validade quando o mundo real valida o processo. E as leis que tangem os direitos autorais juntamente aos contratos de licença podem esclarecer muitos destes questionamentos [Çağlayan Aksoy and Özkan Üner 2021, Valeonti et al. 2021].

2.5.4. Eficiência energética

A blockchain, por ser uma rede descentralizada, necessita de um conjunto de nós interligados de forma ponto a ponto para manterem os dados replicados e atualizados. Nesta topologia de rede, o sistema torna-se altamente disponível mesmo que alguns nós saiam da rede ou fiquem inacessíveis. Ainda para manter a segurança, a blockchain utiliza uma estratégia tolerante a falhas, chamada de consenso, para obter o acordo necessário em um único estado de rede. O consenso é um método de tomada de decisão para um conjunto de nós que precisam chegar a um acordo para manter o funcionamento do sistema. Existem várias maneiras de implementar os mecanismos de consenso que geram grandes diferenças técnicas em relação a tolerância à falha e consumo de recursos como tempo e processamento [Xiao et al. 2020].

As principais implementações dos mecanismos de consenso são o *Proof of Work (PoW)* e *Proof of Stake (PoS)*. Não há diferença quanto à forma direta de interação com aplicações de mercados para os ativos digitais, porém há uma mudança em relação à taxas e tempo gastos em transações. O *PoW*, por exemplo, tem uma característica de ser extremamente seguro e descentralizado. Em contrapartida ele consome uma quantidade de tempo e energia excessiva para este trabalho. Isto se deve ao fato de que para a criação de um bloco, a rede depende de nós mineradores com hardwares de alta capacidade para verificar as transações e cunharem um novo bloco. As plataformas Bitcoin e Ethereum utilizam o mecanismo de consenso *PoW*, e no caso da Ethereum cobra altas taxas nas transações para compra e venda de NFTs ou qualquer outra transação. Plataformas *blockchains* que utilizam o mecanismo de consenso *PoS* tem a possibilidade de obter maior desempenho e taxas menores uma vez que a cunhagem dos blocos não dependem de um alto processamento e sim da escolha de mineradores que têm seus saldos tomados por contratos como garantia de boa índole. Consequentemente o consumo de energia em *blockchain* que utilizam o *PoS* é menor do que o *PoW* devido a economia de trabalho de processamento realizada pelos mineradores.

2.6. Prática

Nesta seção abordaremos como projetar, codificar, implantar e executar contratos inteligentes para NFTs e um *marketplace* que manipula NFTs alugáveis. Os contratos desenvolvidos serão construídos por meio da linguagem Solidity e, além da implementação e execução destes contratos, apresentaremos como testá-los e invocá-los a partir de uma aplicação descentralizada desenvolvida na linguagem Python, em um ambiente de desenvolvimento configurável. Por sua vez, neste ambiente de desenvolvimento serão utilizadas as plataformas:

- **Remix IDE:** Desenvolver, compilar e implantar os contratos inteligentes na rede *Blockchain*;
- **Ganache:** Simular um nó da rede *Blockchain Ethereum* localmente;
- **Sistema de Arquivos Interplanetário (IPFS):** Gerar os *hashes* ou CID (*Content Identifier*) dos NFTs criados.

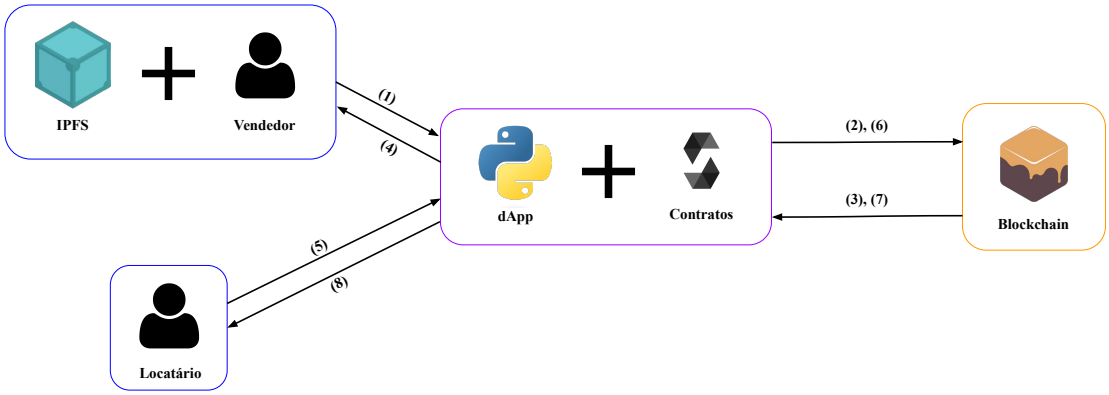


Figura 2.6. Fluxo de criação e aluguel de um NFT.

Conforme a Figura 2.6, ao final da prática teremos o seguinte fluxo: o vendedor enviará o CID gerado pelo IPFS para a *blockchain*, por meio da aplicação descentralizada (*dApp*) desenvolvida em Python (1). Por sua vez, a *dApp* utilizará as funcionalidades do contrato escrito em Solidity, com o objetivo de criar o NFT e disponibilizá-lo para aluguel na *blockchain* simulada pelo Ganache (2). Analogamente, a *dApp* receberá os dados das transações mineradas na *blockchain* (3), para transmiti-los de volta ao vendedor (4). Por outro lado, a pessoa que alugará o *token* utilizará o fluxo de maneira semelhante. O locatário solicitará tanto o aluguel quanto a devolução através da aplicação descentralizada (5), responsável por enviar as transações que serão mineradas (6). Posteriormente, os resultados deste procedimento serão enviados para a *dApp* (7) e, por fim, também poderão ser consultados (8).

Sendo assim, os objetivos desta prática são esclarecer características da programação de NFTs por contratos inteligentes que auxiliam a criação de *tokens* mais eficientes e seguros. Dentre as características mais relevantes a serem discutidas, destacamos a utilização dos padrões estruturais de *tokens* existentes, como as interfaces da OpenZeppelin, assim como os padrões das funções de empréstimo e devolução. Assim, objetivamos abordar os passos para a implementação de uma aplicação real para NFTs, bem como apresentar a utilidade dos *tokens* não fungíveis para controle de propriedade de objetos físicos e digitais.

2.6.1. Configuração do ambiente de desenvolvimento

Nesta subseção, iniciamos o processo de desenvolvimento realizando a configuração do ambiente de desenvolvimento, por meio da instalação e utilização das ferramentas Remix IDE, Ganache e IPFS. Estas ferramentas desempenham o papel de auxiliar na construção e implantação do contrato, bem como na simulação de uma *blockchain* real. Desta forma, a principal vantagem deste ambiente é o fato dele ser facilmente configurável e sem gastos reais, visto que toda execução é realizada localmente.

2.6.1.1. Remix IDE

A primeira ferramenta que abordaremos será o Remix[12], que nada mais é do que uma IDE desenvolvida em ambiente web, utilizada principalmente para desenvolver, compilar, implantar e utilizar contratos inteligentes, como é exibido na Figura 2.7.

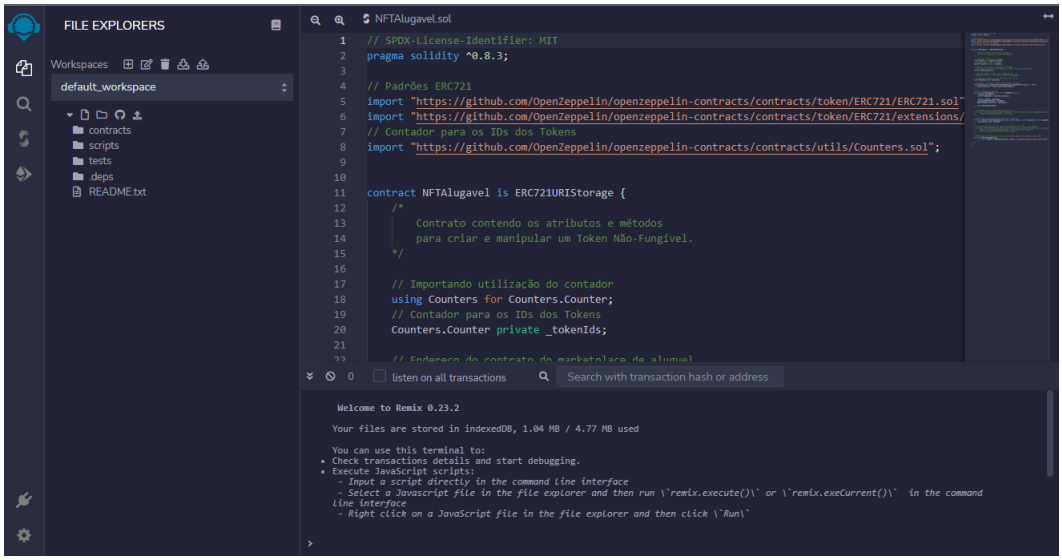


Figura 2.7. Tela principal da ferramenta Remix IDE.

Por meio do acesso a IDE[13], seremos capazes de realizar todas as etapas descritas anteriormente que são necessárias para o funcionamento da rede de NFTs alugáveis. Na primeira aba da esquerda da Figura 2.7 podemos criar novos arquivos da linguagem Solidity para desenvolvermos nossos contratos. Já na terceira aba seremos capazes de compilar os contratos, a fim de verificarmos possíveis erros de sintaxe e/ou de fluxo e, finalmente, na quarta aba poderemos implantá-los na rede *blockchain* de teste que configuraremos a seguir.

2.6.1.2. Ganache

A segunda ferramenta que necessitamos é o Ganache[14], utilizada para simular uma *blockchain Ethereum* localmente. Dessa maneira, é possível configurar a rede, realizar testes e monitorar transações da forma que o usuário desejar. Outra vantagem é a integração com o Remix IDE, além da possibilidade de criar um ambiente de trabalho de forma rápida que engloba a rede *blockchain* juntamente com as contas fictícias para teste, conforme na Figura 2.8.

[12] `https://remix-ide.readthedocs.io/en/latest/file_explorer.html`
[13] `http://remix.ethereum.org/`
[14] `https://trufflesuite.com/docs/ganache/`

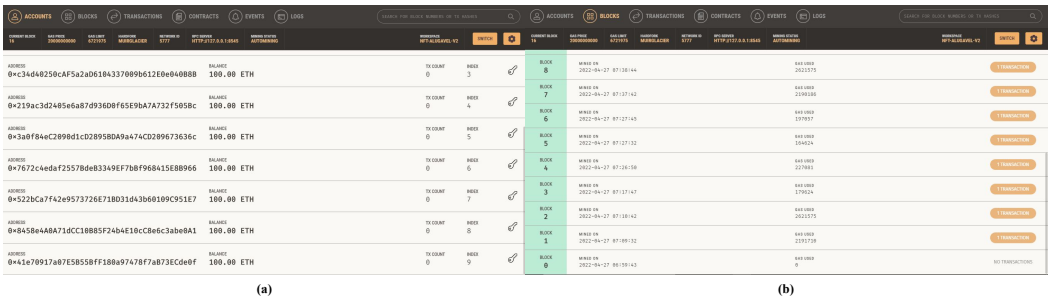


Figura 2.8. (a) Lista de contas criadas automaticamente. (b) Histórico de todas as transações da rede de teste.

2.6.1.3. IPFS - Interplanetary File System

O sistema de arquivos interplanetário, ou IPFS[15], é um sistema de arquivos descentralizado, que visa criar um armazenamento associado ponto-a-ponto e distribuído. Suas principais vantagens são garantir segurança e privacidade dos dados armazenados, e vem sendo utilizado juntamente com as tecnologias de *blockchain* e NFTs, pelo fato da descentralização da ferramenta possibilitar o acesso aos ativos de maneira simplificada.

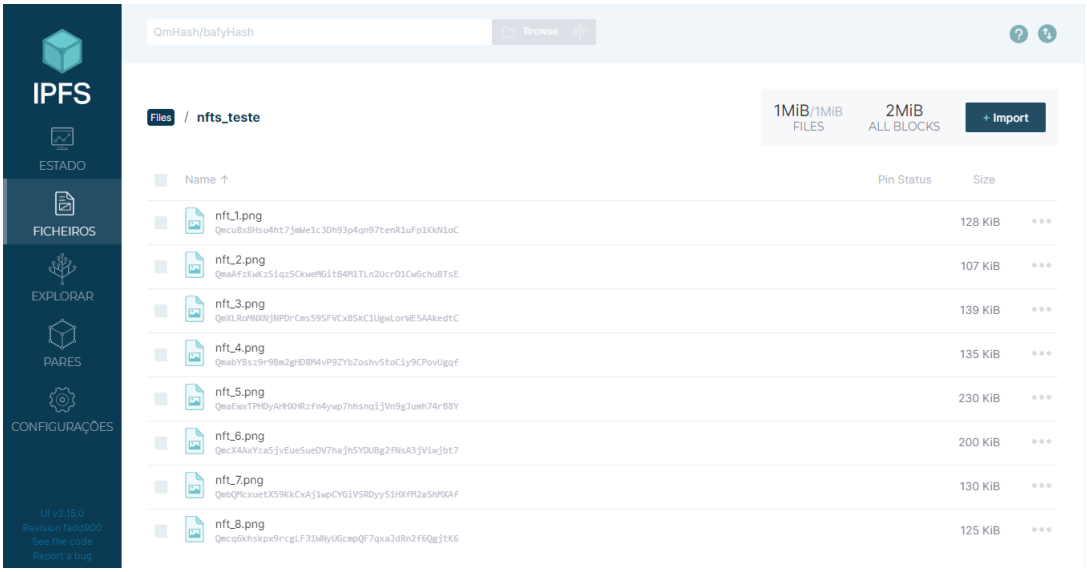


Figura 2.9. Tela que exibe os arquivos de um usuário alocados no IPFS.

Assim sendo, o IPFS torna-se uma solução efetiva para nossa prática, no que diz respeito a armazenar os NFTs e gerar os CIDs, que são o conteúdo das imagens codificadas por *hash*. Apesar desse sistema de arquivos ser utilizado por outras ferramentas, como o Piñata[16], que endereça arquivos na nuvem e podem ser acessados de qualquer lugar, optamos pelo IPFS Desktop[17] propriamente dito, para termos controle local dos

[15] https://docs.ipfs.io/
[16] https://www.pinata.cloud/
[17] https://docs.ipfs.io/install/ipfs-desktop/

NFTs, conforme a Figura 2.9.

2.6.2. Fundamentos da Linguagem Solidity

Nesta subseção apresentaremos os principais conceitos relativos à linguagem de descrição de contratos inteligentes para NFTs. O objetivo é ensinar os fundamentos da linguagem Solidity[18], que é de alto nível e criada por meio de uma combinação de JavaScript, Java e C++, para realizar a interação do conjunto de funções e eventos definidos nos padrões ERC dos *tokens* não fungíveis. A partir das características da linguagem descritas e apresentadas, trabalharemos com as estruturas dos ERC baseados nas funções de propriedades de metadados.

2.6.2.1. Tipos de dados

Os tipos de dados em Solidity apresentam similaridades com a linguagem C++ e JavaScript, no que diz respeito às operações entre eles. Destacamos os principais tipos de dados utilizados durante a prática. O primeiro tipo de dados e o tipo mais básico das linguagens de programação é o tipo booleano, o qual é representado pelos valores verdadeiro ou falso na linguagem Solidity.

```
1 bool verdadeiro = true;
2 bool falso = false;
```

Listagem 2.4. Exemplo de variáveis booleanas em Solidity.

Já o segundo tipo de dados muito utilizado nas linguagens é o tipo inteiro. Diferentemente de outras linguagens, em Solidity é possível especificar o tamanho do inteiro em bits e a sinalização, conforme mostra a Listagem 2.5.

```
1 int8 w = 3; // Inteiro com sinal de 8 bits
2 uint8 x = 1; // Inteiro sem sinal de 8 bits
3 int y = 10; // (int256) Inteiro sem sinal de 256 bits
4 uint256 z = 30; // (uint) Inteiro sem sinal de 256 bits
```

Listagem 2.5. Exemplo de variáveis inteiras em Solidity.

Outros dois tipos de dados existentes em Solidity são os Bytes e Strings, utilizados comumente para armazenar cadeias de caracteres. Assim como o tipo inteiro, os bytes podem ter tamanhos definidos. Por sua vez, variáveis do tipo String podem armazenar dados com a codificação UTF-8 de qualquer comprimento. A diferença dos dois tipos baseia-se na utilização, visto que o tipo Bytes é mais barato computacionalmente, logo, quando o texto pode possuir um tamanho definido, recomenda-se o uso do tipo Bytes em vista do tipo String.

```
1 bytes32 b = "Texto";
2 string s = "Ola, Mundo!";
```

Listagem 2.6. Exemplo de variáveis que armazenam caracteres em Solidity.

Além disso, a linguagem Solidity possui tipos de dados específicos: os mapeamentos e endereços. O tipo de dado mapeamento pode ser associado com a tabela *hash*.

[18] https://docs.soliditylang.org/en/v0.8.9/

Sua sintaxe é simples conforme a Listagem 2.7 e, como o próprio nome diz, mapeia chaves de um tipo para valores do mesmo ou outro tipo.

```
1 mapping(uint256 => bool) mapeamento; // Mapeia chaves do tipo inteiro para valores booleanos
```

Listagem 2.7. Exemplo de mapeamentos em Solidity.

No que diz respeito ao tipo de dado endereço, podemos dizer que este é um dos tipos mais importantes da linguagem, pois com ele é possível armazenar em variáveis os endereços *Ethereum* de contas, transações, contratos, além de realizar chamadas de funções específicas deste tipo. Variáveis do tipo endereço podem possuir o modificador *payable*, significando que o endereço desta variável pode receber transferências de outro endereço, bem como ter o seu balanço de *Ethers* consultado.

```
1 address payable add1 = address(0x123); // Indica que o endereco add1 e pagavel
2 address meuEndereco = address(this); // Armazena o endereco da entidade que chama uma funcionalidade do contrato
3 add1.transfer(10); // A entidade que chama o contrato realiza uma transferencia para o endereco add1
4 meuEndereco.call(...); // Realiza uma chamada de funcao de outro contrato, por exemplo
```

Listagem 2.8. Exemplo de endereços em Solidity.

Finalmente, o último tipo de dado que abordaremos durante a prática será o tipo composto de estrutura, semelhante ao tipo de estrutura existente na linguagem C. Com este tipo composto, podem ser definidas abstrações de uma entidade que possui atributos de tipos variados, desde tipos básicos como inteiros, booleanos, até mesmo tipos como endereços e mapeamentos.

```
1 struct Token{
2     uint id,
3     bool status,
4     bytes32 cid,
5     address payable dono
6 }
```

Listagem 2.9. Exemplo de estruturas em Solidity.

2.6.2.2. Variáveis especiais

Tendo em vista os tipos de dados mais utilizados em Solidity, apresentaremos também variáveis globais e funções reservadas que possuem papéis importantes ao tratar de manipulação de contratos e suas funcionalidades.

A primeira variável que abordaremos é a *block*, visto que como o próprio nome diz, possui valores referentes ao bloco atual da rede *blockchain*. Por exemplo, alguns valores possíveis de serem consultados são o número do bloco e a data/hora em segundos, desde uma época em Unix. Do mesmo modo, a variável *msg* possui valores referentes ao envio da transação, isto é, dados como o endereço de quem envia a transação, bem como o número de *Wei* transferido. Finalmente, a variável global *abi* possui as funcionalidades de decodificar ou codificar funções externas, tal como funções de outros contratos.

```
1 block.number; // Retorna o numero atual do bloco
2 block.timestamp; // Retorna a data e hora atual
3 msg.sender; // Endereco de quem envia a transacao
4 msg.value; // Valor em Wei transferido
5 abi.encodeWithSignature(...); // Codifica uma funcao enviando sua assinatura como primeiro parametro
```

Listagem 2.10. Exemplo de variáveis globais em Solidity.

2.6.2.3. Funções

As funções em Solidity possuem grande importância ao tratarmos de contratos inteligentes, pelo fato de definirem de forma exata as funcionalidades que podem ser realizadas e quem é capaz de acessá-las. Logo, cada função possui um tipo associado, indicando qual o escopo de sua utilização:

- ***public*:** Este tipo indica que a função pode ser acessada por todos;
- ***external*:** Este tipo informa que a função pode ser acessada apenas externamente ao contrato;
- ***internal*:** Este tipo assinala que uma função pode ser acessada apenas pelo próprio contrato, ou por contratos e bibliotecas derivados;
- ***private*:** Este tipo restringe o escopo da função para que apenas o próprio contrato possa acessá-la.

Adicionalmente, funções em Solidity possuem modificadores padrões que definem quais regras a função deve seguir em seu fluxo de execução:

- ***pure*:** Este modificador indica que a função não poderá ler dados de variáveis externas à função e tampouco alterá-las, isto é, funções com este modificador podem trabalhar apenas em seu escopo local e com os parâmetros recebidos;
- ***view*:** Já este modificador restringe uma função para que ela possa ler dados de variáveis externas à ela, porém ainda assim funções com este modificador não podem alterar dados destas variáveis;
- ***payable*:** Por fim, este modificador aponta que a função pode acessar variáveis externas, bem como alterá-las e, em adição, requisita que ao utilizar a função, um pagamento em *Ether* deve ser realizado.

Vale ressaltar que, para uma função conseguir ler e alterar os dados de variáveis externas, porém sem requisitar um pagamento, basta omitir estes modificadores. Porém, cada um deles adiciona uma camada de segurança e coerência importantes para a utilização de um contrato. Na Listagem 2.11 é possível observar o modelo de declaração de uma função.

```
1 function nomeFunc(<parametros>) {public|internal|external|private} [pure|view|payable] returns (<tipo retorno>)
```

Listagem 2.11. Exemplo de função em Solidity.

2.6.2.4. Alocação de dados

Em Solidity, as variáveis além de possuírem tipos associados, podem ser armazenadas de modos diferentes. Semelhantemente com a memória principal e secundária de um computador, existem os modificadores *memory* e *storage*, respectivamente.

Variáveis com o modificador *memory* possuem dados armazenados em tempo de execução do contrato, e normalmente são utilizadas para armazenar parâmetros ou retornos de funções. Por outro lado, o modificador *storage* é o modo de armazenamento padrão da linguagem, logo, variáveis com este modificador possuem dados armazenados de forma persistente e, consequentemente, consomem uma quantidade maior de gás em comparação com as variáveis que possuem o modificador anterior.

```
1 string storage texto = "Persistente"; // Armazenamento persistente
2 bool memory statusRetorno = "False"; // Armazenamento temporario
3 mapping(uint => bool) mapeamento; // Armazenamento persistente omitido
4 delete mapeamento[1]; // Metodo para desalocar um item
```

Listagem 2.12. Exemplo de alocação de dados em Solidity.

2.6.3. Contratos inteligentes

Em sequência, nesta subseção abordaremos os recursos necessários sobre contratos inteligentes em Solidity voltados para a nossa prática. Os contratos assemelham-se com as classes de linguagens de programação orientadas a objetos, isto é, cada contrato em Solidity pode possuir um construtor, variáveis com armazenamento persistente e métodos que manipulam estas variáveis, conforme a Listagem 2.13.

Embora haja similaridade com as classes de linguagens orientadas a objetos, os contratos possuem particularidades intrínsecas, como por exemplo os eventos, manipuladores de erros e modificadores, que proporcionam vantagens de segurança ao tratarmos sobre boas práticas de modelagem de contratos.

```
1  contract Marketplace{
2      address payable donoContrato;
3      uint256 taxaMarketplace = 0.02 ether;
4
5      constructor(){
6          donoContrato = payable(msg.sender);
7      }
8
9      function getTaxaMarketplace() public view returns (uint256) {
10         return taxaMarketplace;
11     }
12
13     function setTaxaMarketplace(uint256 _novaTaxa) external {
14         taxaMarketplace = _novaTaxa;
15     }
16 }
```

Listagem 2.13. Exemplo básico de um contrato em Solidity.

2.6.3.1. Eventos

Os eventos são recursos muito utilizados em contratos, com o objetivo de transmitirem informações para as aplicações descentralizadas. Além disso, os eventos são úteis para

retornarem informações em funções que geram transações na *blockchain*, pelo fato dos dados de um evento poderem ser acessados por meio dos *logs* da transação.

```
1 event RetornaToken(uint256 tokenId); // Declaracao de um evento
2 emit RetornaToken(1); // Modo de disparar um evento
```

Listagem 2.14. Exemplo de eventos em Solidity.

2.6.3.2. Manipuladores de erros

Outro recurso difundido na construção de um contrato são os manipuladores de erros, que são funcionalidades previamente definidas pela linguagem para tratamento de exceção ou para avaliar condições prévias que devem ser atendidas. Nesta prática utilizaremos o manipulador *require*, que avalia se uma condição é falsa, para então disparar uma exceção e finalizar o fluxo principal do escopo em que o manipulador foi chamado. Adicionalmente, uma mensagem customizada pode ser retornada, caso a mesma esteja definida.

```
1 require(msg.value == taxaMarketplace, "Pague exatamente a taxa que o Marketplace exige!");
```

Listagem 2.15. Exemplo de manipuladores de erros em Solidity.

2.6.3.3. Modificadores

No que tange aos modificadores criados internamente em contratos, os mesmos são semelhantes aos modificadores padrões da linguagem vistos em 2.6.2.3. Novos modificadores podem ser criados dentro dos contratos para serem utilizados em determinadas funções e delimitarem o uso das mesmas, como por exemplo, especificar quais funções apenas o dono do contrato poderá invocá-las.

```
1 modifier apenasMarketplace(){
2     require(msg.sender == marketplace, "Apenas o marketplace pode utilizar esse metodo!");
3     _;
4 }
```

Listagem 2.16. Exemplo de modificadores em Solidity.

2.6.4. Padrões ERC e OpenZeppelin

Até o momento, observamos como os contratos podem ser modelados, bem como os seus recursos internos existentes e como manipular estes recursos. No que diz respeito a segurança de um contrato, podemos destacar a utilização de modificadores e tratamentos de erros, porém existem outras técnicas de modelar um contrato inteligente de maneira eficiente e segura.

Com o advento dos NFTs, os padrões ERC tornaram-se referência para a modelagem de contratos de *tokens*, e juntamente com eles surgiram as demandas por segurança. Dessa forma, os padrões da OpenZeppelin[19] introduziram interfaces e contratos devidamente testados para que o processo de desenvolvimento de contratos seguros e eficientes torne-se mais simples.

[19] https://docs.openzeppelin.com/

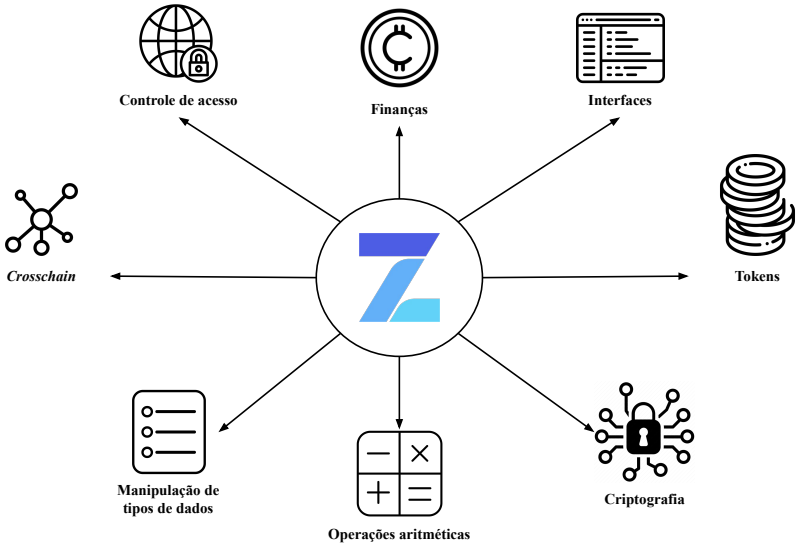


Figura 2.10. Subáreas de segurança abrangidas pela biblioteca OpenZeppelin.

A biblioteca OpenZeppelin conta com contratos e interfaces para diversos tipos de aplicações, além de possuir implementados os próprios padrões ERC. Na Figura 2.10 é possível observar como esta biblioteca perpassa várias aplicações importantes que podem ser desenvolvidas para *blockchain* e NFTs.

2.6.5. Desenvolvendo NFTs baseados nos padrões de *tokens* Ethereum

Nesta subseção, abordaremos o processo de desenvolvimento de *tokens* não fungíveis utilizando algumas das boas práticas de modelagem apresentadas nas seções anteriores. Ao final, planejamos obter um contrato capaz de criar NFTs e com funcionalidades disponíveis de serem utilizadas por meio de uma aplicação descentralizada e independente de plataforma.

O primeiro passo para modelar o contrato de NFTs de forma segura e eficiente, baseia-se em importar os contratos da biblioteca OpenZeppelin que já possuem funcionalidades bem testadas e implementadas. Nesta prática, optamos por utilizar o contrato do padrão ERC-721 para criar os *tokens*, bem como o contrato para armazenar o código da imagem que se tornará um NFT. Utilizamos ainda a biblioteca de contadores para termos controle dos índices dos *tokens* criados.

```
1 import "https://github.com/OpenZeppelin/openzeppelin-contracts/contracts/token/ERC721/ERC721.sol";
2 import "https://github.com/OpenZeppelin/openzeppelin-contracts/contracts/token/ERC721/extensions/ERC721URIStorage.
      sol";
3 import "https://github.com/OpenZeppelin/openzeppelin-contracts/contracts/utils/Counters.sol";
```

Listagem 2.17. Contratos e bibliotecas importadas.

Posteriormente, o contrato inteligente pode ser criado herdando as funcionalidades do contrato da Linha 2 na Listagem 2.17. Visto que esta importação herda o contrato do padrão ERC-721, não precisamos especificar novamente na definição do contrato de NFTs esta herança. Internos ao mesmo, teremos os atributos: o contador de identificador dos *tokens*, com o objetivo de armazenar o índice correto a cada criação de um novo NFT,

o endereço do contrato que representa o *Marketplace* de aluguéis, com a finalidade de demarcar qual *Marketplace* pode gerenciar os NFTs deste novo contrato. Além do mais, deve existir o evento de emitir o identificador de um novo *token* de volta para a aplicação descentralizada e um modificador para demarcar quais funções apenas o *Marketplace* estará elegível para utilizar.

```
1  contract NFTAlugavel is ERC721URIStorage {
2      using Counters for Counters.Counter;
3      Counters.Counter private _tokenIds;
4
5      address marketplace;
6
7      event TokenId(uint256 token_id);
8
9      constructor(address enderecoContratoMarketplace) ERC721("NFTAlugavel", "AFT") {
10         marketplace = enderecoContratoMarketplace;
11     }
12
13     modifier apenasMarketplace(){
14         require(msg.sender == marketplace, "Apenas o marketplace pode utilizar esse metodo!");
15         _;
16     }
17 }
```

Listagem 2.18. Atributos do contrato de NFTs.

Vale ressaltar que, conforme a Listagem 2.18, é possível observar a inicialização do construtor do padrão ERC-721. Este contrato demanda a inicialização do nome dos NFTs, bem como o símbolo destes *tokens*, ambos à cargo do programador. Em seguida, as funções do contrato de NFTs podem ser desenvolvidas, como na Listagem 2.19.

```
1  function criarNovoToken(string memory tokenURI) public {
2      _tokenIds.increment();
3      uint256 newItemId = _tokenIds.current();
4      _mint(msg.sender, newItemId);
5      _setTokenURI(newItemId, tokenURI);
6      approve(marketplace, newItemId);
7      emit TokenId(newItemId);
8  }
9
10 function transferirTokenExpirado(address de, address para, uint256 tokenId) external apenasMarketplace {
11     _transfer(de, para, tokenId);
12 }
```

Listagem 2.19. Funções de criar e retornar um Token Não-Fungível.

A primeira delas (e a mais importante) é a função de criação de um novo *token*, responsável por realizar as chamadas de funções do padrão ERC-721, como por exemplo, as funções de forjar um novo NFT (Linha 4), definir o código CID do novo *token* (Linha 5), bem como permitir o acesso do *Marketplace* ao NFT gerado (Linha 6). Por sua vez, a segunda função aborda o caso de um NFT em estado de alugado, porém com o prazo de aluguel expirado. Em um cenário como este, a função do contrato é chamada apenas pelo próprio *Marketplace* com a finalidade de retornar a titularidade do *token* para o dono de origem e, consequentemente, finalizar o aluguel deste NFT.

De forma complementar, a Listagem 2.20 apresenta o contrato referente ao *Marketplace*, responsável por gerenciar todo o processo de aluguel de um *token*, desde a disponibilização de um item para alugar por parte do vendedor, até o aluguel e finalização por parte do locatário. Os atributos deste contrato assemelham-se com a ideia do contrato

de NFTs, visto que existem os contadores de itens totais, alugados e devolvidos. Neste contrato, há também o atributo que armazena a conta do dono responsável pelo *Marketplace*, bem como a taxa que é cobrada para um vendedor colocar o seu item disponível e, portanto, sendo dessa maneira que o dono do *Marketplace* recebe sobre a utilização do seu sistema. Além disso, uma estrutura é definida para abstrair um item alugável, contendo dados como tempo de aluguel, identificador do *token*, etc. Vale ressaltar que, nesta estrutura há o campo que armazena o endereço do contrato do NFT existente, pelo fato do *Marketplace* ser desenvolvido para hospedar *tokens* de tipos variados, incrementando sua usabilidade. Finalmente, o mapeamento armazena estes itens criados de acordo com seus identificadores, assim como um evento de retorno é emitido após a criação de um item disponível para ser alugado, sendo que neste último existem campos com o modificador *indexed*, indicando que eles podem atuar como filtros em consultas pelo evento nos logs das transações.

```
1  contract MarketplaceAluguel{
2      using Counters for Counters.Counter;
3      Counters.Counter private _itemIds;
4      Counters.Counter private _countItensAlugados;
5      Counters.Counter private _countItensDevolvidos;
6
7      address payable donoContrato;
8      uint256 taxaMarketplace = 0.02 ether;
9
10     struct Item{uint256 itemId; bool statusAlugado; address contratoNFT;
11         uint256 tokenId; address payable vendedor; address locatario;
12         uint256 preco; uint256 expiraEm; bytes32 descricao;
13     }
14
15     mapping(uint256 => Item) private listaItens;
16
17     event ItemCriado(uint256 indexed itemId, bool statusAlugado,
18         address indexed contratoNFT, uint256 indexed tokenId,
19         address vendedor, address locatario,
20         uint256 preco, uint256 expiraEm, bytes32 descricao
21     );
22
23     constructor(){
24         donoContrato = payable(msg.sender);
25     }
26 }
```

Listagem 2.20. Contrato do Marketplace para NFTs alugáveis.

Após a definição dos atributos do contrato *Marketplace*, faz-se necessário definir os métodos que manipulam estes dados, isto é, as funções responsáveis por colocar um NFT disponível para aluguel, alugar um *token* e finalizar um aluguel. Cada uma destas funções possui suas particularidades, devido às verificações que devem ser feitas em conjunto com o fluxo principal de execução.

A função de disponibilizar um NFT para aluguel, conforme mostra a Listagem 2.21, baseia-se em verificar o preço do NFT, a fim de garantir que o *token* possua um preço válido, além de verificar se a taxa de alocação está sendo paga corretamente ao dono do *Marketplace*. Caso os dados da transação atendam aos requisitos, o fluxo principal da função é seguido. O item é criado e a titularidade do NFT é transferida para o *Marketplace*, com o intuito do mesmo ser capaz de permitir um locatário possuir o *token* durante o tempo de aluguel definido. Em adição, o dono real do NFT transfere a taxa ao

dono do *Marketplace* e o evento de retorno para a aplicação descentralizada é emitido.

```
function criaItemAlugavel(address contratoNFT, uint256 tokenId, uint256 preco, uint256 expiraEm, bytes32 descricao
    ) public payable{
    require(preco > 0, "Preco deve ser ao menos de 1 wei!");
    require(msg.value == taxaMarketplace, "Pague exatamente a taxa que o Marketplace exige!");
    _itemIds.increment();
    uint256 itemId = _itemIds.current();
    listaItens[itemId] = Item(itemId, false, address(contratoNFT), tokenId, payable(msg.sender), address(0),
    preco, expiraEm, descricao);

    IERC721(contratoNFT).transferFrom(msg.sender, address(this), tokenId);
    donoContrato.transfer(taxaMarketplace);
    emit ItemCriado(itemId, false, contratoNFT, tokenId, msg.sender, address(0), preco, expiraEm, descricao);
}
```

Listagem 2.21. Função de criar um NFT alugável.

Já a função de alugar um item disponível - Listagem 2.22 demanda que um NFT não esteja alugado, além de analisar se o *token* requerido pertence ao *Marketplace*, com o objetivo de garantir o sucesso na transferência para o locatário. A última verificação aborda o valor transferido, obrigando o locatário a realizar o pagamento exato pelo aluguel do NFT. O fluxo principal desta função segue a lógica de um processo de compra comum: o pagamento é feito ao vendedor, para logo em seguida os dados do item serem alterados, como o endereço do locatário, o status de *token* alugado e o prazo de expiração.

```
function alugarItem(address contratoNFT, uint256 itemId) public payable{
    uint256 preco = listaItens[itemId].preco;
    uint256 tokenId = listaItens[itemId].tokenId;
    require(!listaItens[itemId].statusAlugado, "Este token ja esta alugado!");
    require(IERC721(contratoNFT).ownerOf(tokenId) == address(this), "Token nao disponivel!");
    require(msg.value == preco, "Pague exatamente o valor do aluguel!");
    listaItens[itemId].vendedor.transfer(msg.value);
    IERC721(contratoNFT).transferFrom(address(this), msg.sender, tokenId);
    listaItens[itemId].expiraEm = listaItens[itemId].expiraEm + block.timestamp;
    listaItens[itemId].locatario = msg.sender;
    listaItens[itemId].statusAlugado = true;
    _countItensAlugados.increment();
}
```

Listagem 2.22. Função de alugar um NFT.

Concluindo as três funções principais do *Marketplace*, a função de finalizar um aluguel, como consta na Listagem 2.23, é responsável por verificar se um NFT está realmente alugado, bem como se o referido *token* ainda está em período de aluguel. É importante destacar que, o locatário de um NFT é capaz de finalizar o aluguel antes do prazo definido e, por outro lado, qualquer pessoa é capaz de finalizar o aluguel de um item após o prazo expirado. Sendo assim, com os requisitos atendidos, a função transfere o NFT de volta ao vendedor e exclui o item referente a este *token*, ou seja, caso o vendedor desejar colocar o mesmo item para aluguel novamente, uma nova taxa deve ser paga.

```
function finalizaAluguel(uint256 itemId) external {
    Item storage itemAlugado = listaItens[itemId];
    require(itemAlugado.statusAlugado, "Este token nao esta alugado!");
    require(msg.sender == itemAlugado.locatario || block.timestamp >= itemAlugado.expiraEm,
            "Este token ainda esta no periodo de aluguel!");
    itemAlugado.statusAlugado = false;
    (bool sucessoTransferencia, ) = (itemAlugado.contratoNFT).call(
        abi.encodeWithSignature(
            "transferirTokenExpirado(address,address,uint256)",
            itemAlugado.locatario,
```

```
            itemAlugado.vendedor,
            itemAlugado.tokenId
        )
    );
    require(sucessoTransferencia, "Nao foi possivel transferir o NFT de volta para o vendedor!");
    _countItensDevolvidos.increment();
    delete listaItens[itemId];
}
```

Listagem 2.23. Função de finalizar o aluguel de um NFT.

2.6.6. Aplicação descentralizada cliente

Nesta subseção, é implementado a *dApp* responsável por consumir as funcionalidades dos contratos criados anteriormente. Foi decidido utilizar a linguagem Python, juntamente com a biblioteca Web3 para mapear as funções dos contratos, pelo fato de proporcionarem aprendizado sobre o funcionamento do processo de realizar uma transação na rede *blockchain*.

Através destes recursos, podemos mapear as funções do contrato de NFTs, bem como do contrato *Marketplace*. Acerca do contrato de *tokens* não fungíveis, apenas a função de criar um novo *token* é viável ser mapeada, já que a outra funcionalidade deste contrato fica restringida para apenas o *Marketplace* gerenciador dos contratos de NFTs. Conforme a Listagem 2.24 sobre o mapeamento da função de criar um novo *token*, a biblioteca Web3 proporciona métodos que facilitam o processo de criar e enviar transações para a rede *blockchain*.

```
def criarNovoToken(tokenCID : str):
    nonce = web3.eth.get_transaction_count(public_key, 'latest')
    tx = contract.functions.criarNovoToken(tokenCID).buildTransaction({"nonce": nonce, "from": public_key})
    signed_tx = web3.eth.account.sign_transaction(tx, private_key)
    tx_hash = web3.eth.send_raw_transaction(signed_tx.rawTransaction)
    receipt = web3.eth.wait_for_transaction_receipt(tx_hash)
    token_id = Web3.toInt(receipt['logs'][0]['topics'][3])
    return token_id
```

Listagem 2.24. Função de criar um novo Token mapeada para a aplicação descentralizada.

Para uma transação ser validada na rede, é preciso primeiramente resgatar o *nonce* dado a chave pública de uma conta. Este valor nada mais é do que um número pseudo-aleatório, utilizado para meios de autenticação e criptografia durante o processo de mineração da transação. Em seguida, a transação pode ser modelada por meio deste valor gerado e a chave pública da conta que indica quem está realizando a transação. Além disso, a assinatura da função do contrato deve ser invocada, recebendo como parâmetro o CID do novo *token*. Posteriormente, a transação deve ser assinada com a chave privada da conta, sendo que em aplicações Web que utilizam as carteiras como o MetaMask[20], esta funcionalidade fica à cargo da carteira e confirmação do usuário. Ao final, a transação pode ser enviada para a rede e ter o seu *hash* retornado. Após o aguardo da conclusão deste processo, o evento emitido pela função no contrato pode ser consultado por meio dos logs da transação e, consequentemente, podemos armazenar o identificador do novo NFT criado.

[20]https://metamask.io

Analogamente, as funções do contrato *Marketplace* são mapeadas da mesma maneira, porém vale ressaltar o caso das transações que demandam o pagamento de *Ethers*. A Listagem 2.25 exemplifica este processo. No momento de modelar a transação, além dos parâmetros *nonce* e endereço público da conta de quem realiza a transação, são necessários outros valores, como o valor da transação convertidos em wei, isto é, a menor divisão de *ether* possível, o gás máximo que poderá ser consumido pela transação e o preço deste gás, que indica o quanto o usuário está disposto a pagar por cada unidade de gás, ou seja, para ter sua transação concluída de forma mais rápida é necessário um preço de gás mais alto. Novamente, este processo de gerar tais valores em uma aplicação Web fica à cargo da carteira utilizada pelo usuário na maioria dos casos.

```
1 def alugarItem(itemId: int, valor : float):
2     nonce = web3.eth.get_transaction_count(public_key, 'latest')
3     tx = contract.functions.alugarItem(contract_nft, itemId).buildTransaction({"nonce": nonce, "from": public_key,
          "value": web3.toWei(valor, 'ether'), "gas": 2000000, "gasPrice": web3.toWei('50', 'gwei')})
4     signed_tx = web3.eth.account.sign_transaction(tx, private_key)
5     tx_hash = web3.eth.send_raw_transaction(signed_tx.rawTransaction)
6     return tx_hash
```

Listagem 2.25. Função de alugar um *token* mapeada para a aplicação descentralizada.

2.6.7. Implantação e testes

Tendo em vista a construção de todo o fluxo de trabalho do sistema de aluguéis de NFTs, nesta subseção concluímos a prática apresentando o processo de implantar os contratos criados, assim como utilizá-los por meio da aplicação descentralizada desenvolvida. Com o auxílio do Remix IDE, há a possibilidade de compilar os contratos criados em diversas versões de compiladores, para logo em seguida obter dados como o ABI (*Application Binary Interface*) do contrato compilado e o *Bytecode*. O ABI nada mais é do que a interface utilizada para estabelecer a comunicação entre a aplicação descentralizada e o contrato implantado na rede *blockchain*. Já o *Bytecode* é em sua essência o contrato armazenado na rede.

Seguido da compilação de um contrato, o processo de implantação na rede *blockchain* pode ser realizado. Este processo inicia-se selecionando o ambiente em que o contrato será implantado. O Remix permite a utilização de máquinas virtuais para testes, assim como diversos provedores. No caso desta prática, podemos selecionar o Ganache diretamente pelas opções, ou utilizá-lo em conjunto com o MetaMask, sendo que desta última forma as contas fictícias geradas pelo Ganache são mapeadas na carteira MetaMask por meio das chaves privadas de cada conta.

Figura 2.11. Transação ao implantar um contrato na *blockchain*.

Através do contrato implantado, somos capazes de observar na interface do Gana-

che o registro e os dados da transação, conforme a Figura 2.11. Um dos dados importantes a serem analisados é o endereço em que o contrato foi criado na rede, pois por meio dele a *dApp* consegue buscar o contrato e fazer uso de suas funções, juntamente com o ABI. Em contraste, no Remix é possível utilizar as funções do contrato para teste por meio de uma interface previamente definida após a implantação, como na Figura 2.12.

Figura 2.12. Teste de implantação gerado pelo Remix.

De acordo com os processos anteriores, a aplicação descentralizada pode então ser inicializada por meio dos endereços dos contratos, em conjunto com os ABIs e endereços das contas que serão utilizadas. Adicionalmente, o provedor da rede deve ser especificado conforme a Listagem 2.26. Em suma, a execução da *dApp* é realizada via terminal, como mostra a Figura 2.13, local em que os dados como eventos emitidos e status de retorno podem ser conferidos.

```
1 public_key = "0x..."
2 private_key = "..."
3 web3 = Web3(Web3.HTTPProvider(ganache))
4 contract = self.web3.eth.contract(contract_address, contract_abi)
```

Listagem 2.26. Inicialização da aplicação descentralizada.

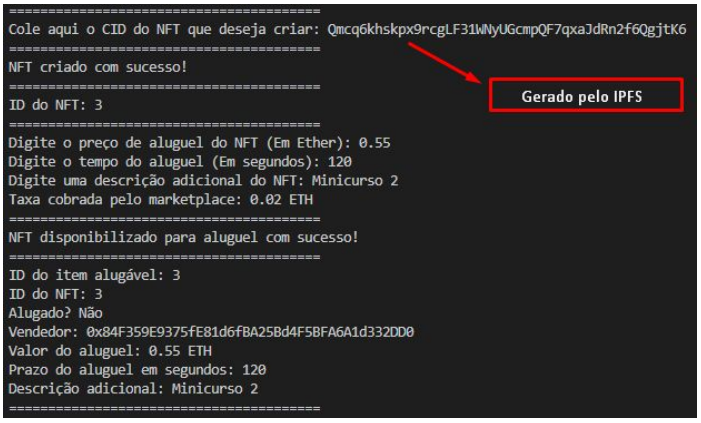


Figura 2.13. Exemplo de resultado obtido ao executar a dApp.

Em suma, a simulação prática objetivou associar a utilização dos contratos e a

comunicação entre *blockchain* e *dApp* com os *Marketplaces* existentes, como o OpenSea[21] e o Rarible[22]. Ademais, melhorias podem ser implementadas, bem como a expansão e teste da aplicação descentralizada em ambiente web, integrada com as carteiras de criptomoedas e *tokens*. Todo o código da seção prática é acessível pelo Github: `https://github.com/lesc-ufv/minicurso2-sbrc2022`.

2.7. Conclusão

Este capítulo discorreu sobre os conceitos, aplicações e desafios do padrão ERC-721 para *tokens* não fungíveis. Inicialmente foram definidos os fundamentos essenciais para um bom entendimento dos *tokens* passando pelas *blockchains*, contratos inteligentes, plataformas, carteiras, oráculos, aplicações e finanças descentralizadas, Daos e os cenários de aplicações. Em seguida, foram apresentados as estruturas e os padrões de *tokens* utilizados em rede *blockchain*. Padrões estes que deram origem aos NFTs e que tomou grande proporção no ano de 2021, com milhões de transações realizadas em *blockchains*. Este padrão, que representa objetos como artes, itens de jogos e colecionáveis, é negociado normalmente de forma online e com criptomoedas.

Foram elencados os principais aspectos de segurança relacionados ao desenvolvimento de contratos inteligentes para NFTs, bem como os desafios desta tecnologia emergente. Os padrões sugerem boas práticas que devem ser seguidas para manter a aplicação descentralizada segura e livre de vulnerabilidades que possam ser exploradas. Relacionamos a camada que provê a interface, que fica entre todas as tecnologias envolvidas na estrutura dos NFTs e os usuários, como a mais vulnerável. Portanto atenção a esta camada devem ser despendidas para que, sendo esta implementada normalmente de forma centralizada, não influencie de maneira negativa em uma estrutura descentralizada e segura.

Apresentamos como os NFTs podem fornecer a certificação de propriedade de arte digital, documentos, direitos de propriedade intelectual, licenças e marcas por meio de registros em *blockchain* permanentemente. Desta maneira vimos as funções de criação e comercialização de um NFT, podendo armazená-lo em uma carteira de ativos digitais e alugá-lo virtualmente. Como resultado destes registros podemos verificar a verdadeira propriedade e origem de ativos de forma confiável.

Abordamos na prática a criação de NFTs e como um *Marketplace* pode ser criado a partir dos recursos de contratos inteligentes para manipular os *tokens*. Observamos que é tecnicamente possível realizar a tokenização de um ativo físico e dividi-lo em partes como um NFT. Desta forma, pode-se permitir a concessão de direitos parciais a diferentes proprietários de NFTs representando parte de um mesmo ativo físico.

Por fim, este capítulo objetivou proporcionar uma base de conhecimento e visão do estado da arte para interessados na área de *blockchain* e NFTs, bem como possibilitar o aprendizado inicial destas tecnologias procedendo com implementações de práticas das soluções para NFTs.

[21] `https://opensea.io/`
[22] `https://rarible.com/`

Referências

[Androulaki et al. 2018] Androulaki, E., Barger, A., Bortnikov, V., Cachin, C., Christidis, K., De Caro, A., Enyeart, D., Ferris, C., Laventman, G., Manevich, Y., et al. (2018). Hyperledger fabric: a distributed operating system for permissioned blockchains. In *Proceedings of the thirteenth EuroSys conference*, pages 1–15.

[Bao and Roubaud 2022] Bao, H. and Roubaud, D. (2022). Recent development in fintech: Non-fungible token. *FinTech*, 1(1):44–46.

[Bartoletti et al. 2020] Bartoletti, M., Carta, S., Cimoli, T., and Saia, R. (2020). Dissecting ponzi schemes on ethereum: identification, analysis, and impact. *Future Generation Computer Systems*, 102:259–277.

[Borkowski et al. 2019] Borkowski, M., Sigwart, M., Frauenthaler, P., Hukkinen, T., and Schulte, S. (2019). Dextt: Deterministic Cross-Blockchain Token Transfers. *IEEE Access*, 7:111030–111042.

[Buterin et al. 2014] Buterin, V. et al. (2014). Ethereum: A next-generation smart contract and decentralized application platform. *URL: https://github.com/ethereum/wiki/wiki/White-Paper*, 7.

[Cachin et al. 2016] Cachin, C. et al. (2016). Architecture of the hyperledger blockchain fabric. In *Workshop on distributed cryptocurrencies and consensus ledgers*. Chicago.

[Çağlayan Aksoy and Özkan Üner 2021] Çağlayan Aksoy, P. and Özkan Üner, Z. (2021). Nfts and copyright: challenges and opportunities. *Journal Of Intellectual Property Law and Practice*, 16(10):1115–1126.

[Casale-Brunet et al. 2021] Casale-Brunet, S., Ribeca, P., Doyle, P., and Mattavelli, M. (2021). Networks of ethereum non-fungible tokens: A graph-based analysis of the erc-721 ecosystem. In *2021 IEEE International Conference on Blockchain (Blockchain)*, pages 188–195. IEEE.

[Charles 2013] Charles, P. (2013). Openzeppelin. `https://github.com/OpenZeppelin`.

[Chohan 2021] Chohan, U. W. (2021). Non-fungible tokens: Blockchains, scarcity, and value. *Critical Blockchain Research Initiative (CBRI) Working Papers*.

[Dolgui et al. 2020] Dolgui, A., Ivanov, D., Potryasaev, S., Sokolov, B., Ivanova, M., and Werner, F. (2020). Blockchain-oriented dynamic modelling of smart contract design and execution in the supply chain. *International Journal of Production Research*, 58(7):2184–2199.

[El Faqir et al. 2020] El Faqir, Y., Arroyo, J., and Hassan, S. (2020). An overview of decentralized autonomous organizations on the blockchain. In *Proceedings of the 16th international symposium on open collaboration*, pages 1–8.

[Entriken et al. 2018] Entriken, W., Shirley, D., Evans, J., and Sachs, N. (2018). Eip-721: Non-fungible token standard,"ethereum improvement proposals, no. 721, january 2018. https://eips.ethereum.org/EIPS/eip-721.

[Fairfield 2021] Fairfield, J. (2021). Tokenized: The law of non-fungible tokens and unique digital property. *Indiana Law Journal, Forthcoming*.

[Gordon and Catalini 2018] Gordon, W. J. and Catalini, C. (2018). Blockchain technology for healthcare: Facilitating the transition to patient-driven interoperability. *Computational and Structural Biotechnology Journal*, 16:224 – 230.

[Hartel and Schumi 2020] Hartel, P. and Schumi, R. (2020). Mutation testing of smart contracts at scale. In *International Conference on Tests and Proofs*, pages 23–42. Springer.

[Harz and Knottenbelt 2018] Harz, D. and Knottenbelt, W. (2018). Towards safer smart contracts: A survey of languages and verification methods. *arXiv preprint arXiv:1809.09805*.

[Hasanova et al. 2019] Hasanova, H., jun Baek, U., gon Shin, M., Cho, K., and Kim, M. S. (2019). A survey on blockchain cybersecurity vulnerabilities and possible countermeasures. *International Journal of Network Management*, 29(2):1–36.

[Hewa et al. 2021a] Hewa, T., Ylianttila, M., and Liyanage, M. (2021a). Survey on blockchain based smart contracts: Applications, opportunities and challenges. *Journal of Network and Computer Applications*, 177:102857.

[Hewa et al. 2021b] Hewa, T. M., Hu, Y., Liyanage, M., Kanhare, S. S., and Ylianttila, M. (2021b). Survey on blockchain-based smart contracts: Technical aspects and future research. *IEEE Access*, 9:87643–87662.

[Jentzsch 2016] Jentzsch, C. (2016). Decentralized autonomous organization to automate governance. *White paper, November*.

[Johnson et al. 2019] Johnson, S., Robinson, P., and Brainard, J. (2019). Sidechains and interoperability.

[Kemmoe et al. 2020] Kemmoe, V. Y., Stone, W., Kim, J., Kim, D., and Son, J. (2020). Recent advances in smart contracts: A technical overview and state of the art. *IEEE Access*, 8:117782–117801.

[Luu et al. 2016] Luu, L., Chu, D.-H., Olickel, H., Saxena, P., and Hobor, A. (2016). Making smart contracts smarter. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 254–269.

[Monrat et al. 2020] Monrat, A. A., Schelen, O., and Andersson, K. (2020). Performance Evaluation of Permissioned Blockchain Platforms. *2020 IEEE Asia-Pacific Conference on Computer Science and Data Engineering, CSDE 2020*.

[Nadini et al. 2021] Nadini, M., Alessandretti, L., Di Giacinto, F., Martino, M., Aiello, L. M., and Baronchelli, A. (2021). Mapping the nft revolution: market trends, trade networks, and visual features. *Scientific reports*, 11(1):1–11.

[Nakamoto and Bitcoin 2008] Nakamoto, S. and Bitcoin, A. (2008). A peer-to-peer electronic cash system. *Bitcoin.–URL: https://bitcoin. org/bitcoin. pdf*, 4.

[Pal et al. 2021] Pal, O., Alam, B., Thakur, V., and Singh, S. (2021). Key management for blockchain technology. *ICT Express*, 7(1):76–80.

[Pillai et al. 2019] Pillai, A., Saraswat, V., and V. R., A. (2019). Smart wallets on blockchain—attacks and their costs. In Wang, G., El Saddik, A., Lai, X., Martinez Perez, G., and Choo, K.-K. R., editors, *Smart City and Informatization*, pages 649–660, Singapore. Springer Singapore.

[Pillai et al. 2017] Pillai, B., Muthukkumarasamy, V., and Biswas, K. (2017). Challenges in designing a blockchain platform.

[Radomski et al. 2018] Radomski, W., Cooke, A., Castonguay, P., Therien, J., Binet, E., and Sandford, R. (2018). Eip-1155: Multi token standard,"ethereum improvement proposals, no. 1155, june 2018. `https://eips.ethereum.org/EIPS/eip-1155`.

[Rogers et al. 2022] Rogers, I., Carter, D., Morgan, B., and Edgington, A. (2022). Diminishing dreams: The scoping down of the music nft. *M/C Journal*, 25(2).

[Rosenfeld 2012] Rosenfeld, M. (2012). Overview of colored coins. *White paper, bitcoil. co. il*, 41:94.

[Rouhani and Deters 2017] Rouhani, S. and Deters, R. (2017). Performance analysis of ethereum transactions in private blockchain. In *2017 8th IEEE(ICSESS)*, pages 70–74. IEEE.

[Rouhani and Deters 2019] Rouhani, S. and Deters, R. (2019). Security, performance, and applications of smart contracts: A systematic survey. *IEEE Access*, 7:50759–50779.

[Sayeed et al. 2020] Sayeed, S., Marco-Gisbert, H., and Caira, T. (2020). Smart contract: Attacks and protections. *IEEE Access*, 8:24416–24427.

[Schär 2020] Schär, F. (2020). Decentralized finance: On blockchain-and smart contract-based financial markets. *Available at SSRN 3571335*.

[Stradling and Voorhees 2018] Stradling, A. and Voorhees, E. (2018). System and method of providing a multi-validator oracle. US Patent App. 15/715,770.

[Valeonti et al. 2021] Valeonti, F., Bikakis, A., Terras, M., Speed, C., Hudson-Smith, A., and Chalkias, K. (2021). Crypto collectibles, museum funding and openglam: Challenges, opportunities and the potential of non-fungible tokens (nfts). *Applied Sciences*, 11(21).

[Vogelsteller and Buterin 2015] Vogelsteller, F. and Buterin, V. (2015). Eip-20: Token standard, ethereum improvement proposals. `https://eips.ethereum.org/EIPS/eip-20`.

[Wang et al. 2021] Wang, Q., Li, R., Wang, Q., and Chen, S. (2021). Non-fungible token (nft): Overview, evaluation, opportunities and challenges. *arXiv preprint arXiv:2105.07447*.

[Wang et al. 2018] Wang, X., Feng, Q., and Chai, J. (2018). The research of consortium blockchain dynamic consensus based on data transaction evaluation. In *2018 11th International Symposium on Computational Intelligence and Design (ISCID)*, volume 2, pages 214–217. IEEE.

[Wood 2014] Wood, G. (2014). Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper*, 151:1–32.

[Xia et al. 2020] Xia, P., Wang, H., Zhang, B., Ji, R., Gao, B., Wu, L., Luo, X., and Xu, G. (2020). Characterizing cryptocurrency exchange scams. *Computers & Security*, 98:101993.

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

[Zou et al. 2019] Zou, W., Lo, D., Kochhar, P. S., Le, X.-B. D., Xia, X., Feng, Y., Chen, Z., and Xu, B. (2019). Smart contract development: Challenges and opportunities. *IEEE Transactions on Software Engineering*.

Capítulo 3

Metrologia na Era do Aprendizado de Máquina

Sidney Loyola, Antônio A. de A. Rocha, Aline Paes e José F. de Rezende

Abstract

The growth in the number of mobile and cellular networks arouses interest in end-to-end performance measurements of these networks and their impact on mobile applications [15]. Among these measurements, we find those based on the quality of the user experience, which is an excellent source of information about network management. In this context, the implementation of Artificial Intelligence and Machine Learning increases the efficiency of the monitoring process [21]. In this short course, we will discuss how we can use objective parameters of quality of services such as delay, throughput, packet loss, and jitter to estimate and predict the quality of user experience using Machine Learning [21]. Furthermore, it is possible to use Artificial Intelligence to estimate the multiple indicators of the quality of user experience, even in a network that adopts end-to-end encryption [49, 27]. In addition, Machine Learning methods help acquire knowledge about the functioning of networks, understand how the structures of existing networks are, and allow the establishment of new networks with less effort. Among the several Network Metrology problems that can benefit from Machine Learning, we can mention network attack detection, user experience quality prediction, application anomaly detection, and network failures.

Resumo

O crescimento da quantidade de redes móveis e celulares desperta o interesse em medições do desempenho fim-a fim dessas redes e o seu impacto em aplicações móveis [15]. Dentre essas medições encontramos aquelas baseadas em qualidade de experiência do usuário que constitui uma grande fonte de informações sobre o gerenciamento da rede. Nesse contexto, a implementação de Inteligência Artificial e Aprendizado de Máquina aumenta a eficiência do processo de monitoramento [21]. Abordaremos nesse minicurso de que forma podemos utilizar os parâmetros objetivos de qualidade do serviço tais como latência, vazão, perda de pacotes e jitter para estimar e predizer a qualidade de experiência do usuário utilizando Aprendizado de Máquina [21]. Além disso, é possível utilizar a

Inteligência Artificial para estimar os múltiplos indicadores de qualidade de experiência do usuário, mesmo em uma rede que adote criptografia fim-a-fim [49, 27]. Além disso, métodos de Aprendizado de Máquina auxiliam na aquisição do conhecimento sobre o funcionamento das redes, o entendimento de como são as estruturas de redes existentes, permitindo estabelecer novas redes com menor esforço. Entre os diversos problemas de Metrologia de Redes que podem se beneficiar da utilização do Aprendizado de Máquina podemos citar detecção de ataques de redes, predição de qualidade de experiência do usuário, detecção de anomalias em aplicações e falhas de redes.

3.1. Introdução

A expansão das funcionalidades da Internet fez com que o número de usuários crescesse, acarretando em mais de 4.66 bilhões de pessoas conectadas na web segundo relatórios do ano de 2021 [19]. Esse crescimento acelerado vem acompanhado da popularização de *smartphones*, *tablets*, *smart TVs* e *smart watchs*, criando um ambiente interconectado de dispositivos de diversos tipos conhecido como *Internet of Things* (IoT) [3]. Surgem, ainda, novos tipos de redes com características próprias, como as redes móveis, redes sem fio e as redes das *smart cities*. Além disso, a virtualização permite interconectar nós que são dissociados dos equipamentos físicos, criando, assim, diferentes topologias entre o mundo real e o mundo virtual, possibilitando a implantação de redes definidas e gerenciadas por software.

Redes móveis, redes sem fio e o ambiente IoT oferecem desafios singulares. O primeiro deles está relacionado à mobilidade e dinamicidade, uma vez que os dispositivos que fazem parte de tais redes não são fixos e nem estáticos; ao contrário, estão em constante deslocamento. Por exemplo, um celular conectado pode se mover de uma antena para outra vizinha da primeira, bem como pode ir, viajando de avião, por exemplo, de um continente para outro. Dessa forma, eles se conectam e se desconectam das redes com grande velocidade, dificultando a criação de modelos para entendimento e gerenciamento de tais conexões. Outra dificuldade está na heterogeneidade dos dados, sendo os mesmos dispositivos utilizados para assistir vídeos, jogar videogames, acessar contas bancárias e enviar mensagens de texto ou voz. Essa grande variedade de aplicações por si só adiciona complexidade às redes. Entretanto, com o avanço dos ambientes IoT não podemos nem determinar com precisão a origem do tráfego: por exemplo, é perfeitamente possível que uma cafeteira envie um e-mail, ou, até mesmo, envie uma publicação para uma rede social avisando aos colaboradores de uma empresa que o café está pronto.

Esse avanço tecnológico e utilização massiva de recursos computacionais impõem desafios que demandam a utilização de novos métodos e técnicas de gerenciamento, além do desenvolvimento de novas arquiteturas de redes. Por exemplo, a quantidade de informação transportada e a quantidade de dados disponíveis dos diferentes tipos de redes, torna os métodos analíticos para caracterização de tráfego ineficientes e custosos. Assim, algumas técnicas para entendimento e monitoramento, como as tarefas de classificação e predição de tráfego tornam-se um problema de *big data*. Outro desafio imposto pela evolução tecnológica refere-se à dificuldade de realizar medições diretas em redes, devido, em grande parte, à popularização da criptografia. É evidente que isso contribuiu para a privacidade e segurança das redes, porém dificulta o gerenciamento e monitoramento, forçando o desenvolvimento e aplicação de novas técnicas de medição [1].

A área de Metrologia de Redes aborda técnicas, ferramentas e métodos que permitem compreender o comportamento, dinâmica e propriedades das redes. Porém, atualmente, já não basta entendermos as redes, detectarmos falhas e anomalias; é necessário antever os acontecimentos para que ações preventivas possam ser realizadas. Para tanto, modelos de Aprendizado de Máquina (AM) podem ser utilizados para extrair informações úteis e padrões dos dados [6]. Com isso, o processo de tomada de decisão tende a ser mais eficiente e rápido, com o mínimo possível de interferência humana.

Entre as áreas em que modelos de AM têm contribuído para um melhor monitoramento e gerenciamento de redes encontram-se a classificação de tráfego, predição de tráfego, estimativa de qualidade de experiência do usuário, segurança de redes e gerenciamento de falhas. Modelos de AM de naturezas diversas têm sido aplicados com sucesso nos problemas apresentados, com destaque especial às redes neurais, que, além de lidarem com grande quantidade de dados, apresentam alto desempenho preditivo e a habilidade de aprender automaticamente representações, evitando a etapa de engenharia de atributos que seria realizada manualmente [1].

Outros dois cursos sobre metrologia já ocorreram em edições passadas do SBRC, um em 2005, Ziviani e Duarte [57], e outro em 2016, Rocha et al. [42]. Os conceitos apresentados neste minicurso revisitam os minicursos anteriores, apresentando as técnicas adotadas atualmente, diante das novas tecnologias e tendências de pesquisa. Dentre essas novas técnicas e tecnologias, a que mais se destaca é o aprendizado de máquina, em especial as redes neurais, que são aplicadas na resolução dos mais variados problemas.

Veremos no decorrer desse minicurso que os problemas de rede beneficiam-se de outras áreas da Inteligência Artificial, além do Aprendizado de Máquina. Como exemplos de outras áreas cujos métodos podem ser aplicados para modelar problemas de rede, temos a visão computacional e o processamento de linguagem natural. Em ambos os casos, os conceitos dessas áreas são utilizados para modelar o fluxo de dados, tanto por meio de representações de imagens, como por modelos de linguagem. Esse minicurso aborda os conceitos teóricos e fundamentos da Metrologia de Redes e o uso de Aprendizado de Máquina para resolver problemas envolvendo metrologia, de forma eficiente e rápida. Além da apresentação teórica contida nesse texto, forneceremos uma aplicação prática implementada e disponível em `https://github.com/loyoladesa/srbc2022`. A aplicação é autocontida e apresenta um exemplo de como aplicar os conceitos apresentados em um conjunto de dados extraídos de redes.

O principal objetivo deste presente minicurso, portanto, é apresentar as novas tecnologias de Metrologia de Redes. Isso inclui as recentes aplicações com Inteligência Artificial, a utilização em áreas emergentes e a discussão de problemas em aberto que podem fomentar novos trabalhos de pesquisa e desenvolvimento na área. Essas ferramentas e métodos têm influência direta em outras áreas de redes, como o planejamento de redes, engenharia de tráfego, garantia de qualidade de serviço e gerenciamento de redes [57].

O restante do minicurso está estruturado como segue. Primeiro, o minicurso oferece uma breve fundamentação teórica com conceitos de Aprendizado de Máquina, qualidade de serviços e qualidade de experiência do usuário na Seção 3.2. Em seguida, o minicurso discute as principais aplicações de AM para realizar medições e estimativas na Seção 3.3. Na Seção 3.4 é apresentado a forma como irá funcionar a apresentação do

estudo de caso. Finalmente, apresentamos nossas considerações finais na Seção 3.5.

3.2. Fundamentação Teórica

Esta seção tem como objetivo apresentar o estado da arte tanto de Metrologia de Redes quanto dos métodos de Aprendizado de Máquina. Além disso, serão discutidos conceitos de Qualidade de Serviço em redes (QoS) e Qualidade de Experiência do Usuário (QoE). A seção aponta as diversas definições e métodos existentes, de forma a conceitualizar de maneira hierárquica as tarefas, técnicas e problemas envolvidos.

Citaremos os métodos de Aprendizado de Máquina comumente utilizados, os principais experimentos e seus resultados e de que forma eles são utilizados para realizar medições em redes, tal que sejam identificados os modelos e variáveis de acordo com os problemas a serem investigados. Conforme apresentado em Casas [9], não há um método de Aprendizado de Máquina que atenda de forma genérica os problemas de medições de redes, sendo necessário encontrar um modelo específico para cada problema a ser investigado.

A metrologia de redes refere-se às técnicas utilizadas para monitorar, gerenciar e mensurar o tráfego em redes com um nível de detalhamento compatível com a tarefa desejada [1]. Esses conhecimentos são utilizados para entender como as redes trabalham, para monitorar a performance delas, descobrir como os recursos estão sendo consumidos pelo usuários e identificar como pode ser realizado um controle efetivo de modo que a rede forneça os requisitos dos acordos de níveis de serviço [1].

Para cumprir esses objetivos existem duas abordagens para realizar as mensurações. As medições passivas e ativas, como vistas na Figura 3.1:

- Medição passiva - basicamente realiza as medições observando o tráfego sem modificar o mesmo. Essas medições podem ser aplicadas em diversos pontos da rede e como exemplos cita-se o registro de perda de pacotes [42].
- Medição ativa - nesse caso é injetado tráfego padronizado na rede, também chamado de sonda, e monitorado o resultado da sua travessia pela rede, tornando possível extrair informações sobre os caminhos existentes [42].

3.2.1. Aprendizado de Máquina

Aprendizado de Máquina (AM) é um subcampo da Inteligência Artificial que visa equipar as máquinas com a habilidade de resolver problemas que requerem aprendizagem, obtida a partir de experiência. A principal motivação para o seu desenvolvimento é que nem todo problema pode ser modelado e resolvido utilizando um algoritmo pré-definido, ou seja, algoritmos que seguem um passo-a-passo para serem implementados.

Por exemplo, reconhecer uma pessoa a partir do seu rosto pode ser uma tarefa simples para os humanos, mas não é trivial especificar uma sequência de passos para a máquina resolver essa tarefa. Diante de cenários como esses, as técnicas de Aprendizado de Máquina constroem conhecimento sem serem previamente "programadas" para tal, ao invés disso elas vão "aprendendo" (melhorando o desempenho de alguma tarefa) a partir de exemplos [29].

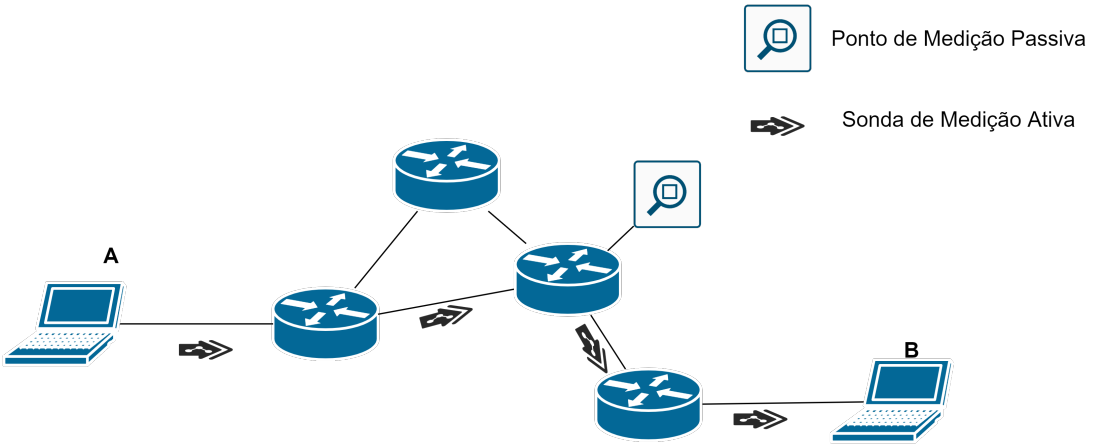


Figura 3.1. Exemplo de Medições. Figura adaptada de Ziviani e Duarte [57]

Este aprendizado é, na maioria das vezes, a busca por uma função alvo (desconhecida) capaz de resolver o problema proposto. Utilizando dados relacionados à tarefa (a experiência), os algoritmos induzem funções capazes de alcançar um determinado objetivo por si próprias. A experiência é comumente representada como o conjunto de dados compostos por exemplos – um exemplo é uma experiência individual e seus atributos – variáveis descrevendo a experiência [29].

Abaixo, apresentamos algumas definições de AM utilizadas neste minicurso:

- **Conjunto de Dados**. É a representação tabular dos atributos que representam os objetos estudados [29]. No caso desse minicurso seriam os dados medidos das redes como latência, vazão, perda de pacotes e jitter.
- **Atributo (*Feature*)**. Característica do tráfego da rede, obtida diretamente ou derivada (através de algum cálculo ou técnica). Cada atributo está associado a uma propriedade do objeto pesquisado, nesse caso o fluxo de rede [29].
- **Atributos preditivos**. São atributos utilizados como entradas para os modelos de AM. Normalmente a entrada é representada por um vetor de atributos [29].
- **Atributo alvo**. Também chamado de alvo ou saída, representa o fenômeno de interesse da previsão, em nosso caso pode ser a classe do tráfego, a estimativa de QoE ou apenas a identificação de um ataque malicioso [29].

Cada abordagem de AM pode escolher uma série de diferentes estratégias para aprender a função-alvo. Isso inclui a representação da experiência, incluindo matrizes de exemplos e atributos, pares de entrada e saída ou apenas entradas, interação com o meio ambiente; a representação da função aprendida, por exemplo, funções, regras, distribuições de probabilidade; e a forma como o método percorre o espaço de pesquisa para encontrar uma aproximação da função alvo [29].

Em relação ao tipo de experiência adquirida, os métodos de AM seguem três paradigmas principais, nas quais as tarefas de aprendizado são comumente classificadas:

aprendizado supervisionado, aprendizado não supervisionado e aprendizado por reforço[1].

Aprendizado Supervisionado

Nesse paradigma, as tarefas são preditivas e o conjunto de dados de treinamento deve ter atributos (características) de entrada e de saída (também chamada de alvo). As saídas devem ser rotuladas simulando a atividade de um supervisor, ou seja, alguém que sabe a "resposta". A tarefa de aprendizado supervisionado pode ser definida da seguinte forma [44]:

> Dado um conjunto de treinamento de N pares de exemplos de entrada e saída
>
> $$(x_1,y_1),(x_2,y_2),...(x_n,y_n),$$
>
> onde cada y_i foi gerado por uma função desconhecida $y = f(x)$, o algoritmo deve encontrar uma função h que se aproxime da função verdadeira f.

A função de hipótese h deve ser válida para outros objetos do mesmo domínio que não pertençam ao conjunto de treinamento. A essa propriedade dá-se o nome de generalização. A baixa capacidade de generalização significa que o modelo e os dados estão super ajustados (*overfitting*), ou subajustado aos dados (*underfitting*) [29].

Adota-se como boa prática, quando lidando com AM, utilizar três conjuntos de dados: treinamento, validação e teste. O conjunto de treinamento é utilizado para ajustar o modelo, ou seja, são fornecidos dados para que o algoritmo aprenda: encontre uma função que aproxime f, a partir de exemplos conhecidos. O conjunto de validação é importante para avaliar a capacidade de generalização do modelo, verificando se ele não está superajustado e nem subajustado, além de permitir a escolha do modelo por meio do ajuste dos hiperparâmetros. Por fim, com o conjunto de teste, avalia-se o desempenho preditivo do modelo, constatando se ele resolve ou não o problema proposto.

As tarefas preditivas podem ser divididas em problemas de **classificação**, quando a saída y for um conjunto de valores qualitativos, por exemplo, tipo de tráfego de rede (*email*, *streaming*, *gaming*). Já quando a saída é um valor numérico, ex: *estimativa de QoE*, a tarefa é chamada de **regressão**.

Russel e I.Norvig [44] apresentam as seguintes definições:

- Classificação: $y_i = f(x_i) \in \{c_1,...,c_m\}$, isto é, $f(x_i)$ assume valores em um conjunto discreto, não ordenado;
- Regressão: $y_i \in \mathbb{R}$, isto é, $f(x_i)$ assume valores em um conjunto infinito e ordenado de valores.

[1] Outros tipos de supervisão também existem, a saber, aprendizado semi-supervisionad, quando apenas um subconjunto dos exemplos tem uma saída; e auto-supervisionado, quando o rótulo é extraído da própria tarefa sem a supervisão humana

Aprendizado Não Supervisionado

O objetivo das tarefas neste paradigma é descrever o conjunto de dados, descobrindo alguma ordem ou padrão relevante que auxilie na representação dos dados. Assim, não é necessário que o conjunto de dados seja rotulado e os algoritmos utilizam agrupamentos ou regras de associação entre grupos de atributos. O agrupamento identifica a similaridade dos itens separando-os em grupos distintos e, além disso, busca-se maximizar a não similaridade entre os itens de grupos diferentes.

Já as regras de associação buscam padrões frequentes entre grupos de atributos. Esta técnica é bem explorada no contexto de cestas de produtos para descobrir os itens que são comprados em conjunto, e é bastante utilizada para guiar as ações de marketing de lojas online [29]. Essas regras de associação vêm sendo aplicadas em problemas de redes, auxiliando na classificação de tráfego e na identificação de dados que não obedecem a um padrão conhecido, permitindo identificar anomalias em redes. Outro problema em que esta técnica pode ser utilizada é a detecção de intrusos, tanto em redes cabeadas como em redes sem fio. As regras de associação aprendidas permitem observar dados anômalos realizando a detecção de agentes maliciosos [50].

Aprendizado por Reforço

Diferente dos outros dois paradigmas, não há a necessidade de um conjunto de dados para treinamento. As tarefas simulam aprendizado por tentativa e erro, premiando as ações positivas e punindo as ações negativas. Essa é a forma com que os robôs aprendem a andar e, até mesmo, a dançar. Muitas aplicações estão relacionadas com a robótica, com jogos de estratégia e atividades difíceis de programar, por exemplo, o controle de um helicóptero autônomo que realiza voos acrobáticos. Uma desvantagem significativa dessa abordagem é a necessidade de muitas repetições para a obtenção de bons resultados, o que gera um enorme custo computacional [44].

Alguns Algoritmos de Aprendizado de Máquinas

A seguir, apresentamos uma breve descrição dos principais algoritmos de Aprendizado de Máquina encontrados na literatura que também foram aplicados na metrologia de redes:

- **k-NN**. O algoritmo dos vizinhos mais próximos (k-NN) é um método baseado em distância. Assim, o exemplo de entrada será classificado de acordo com as instâncias que estão próximos a ele no conjunto de treinamento. O parâmetro k define a quantidade de instâncias a serem levados em conta para a previsão. É considerado um algoritmo preguiçoso (*lazy*) por apenas memorizar os dados do conjunto de treinamento, sem necessariamente aprender um modelo. Tem dificuldade em lidar com grande número de atributos (dimensionalidade), por causa do cálculo da similaridade em um espaço dimensional muito grande, e o processo de classificação é lento, já que no pior caso, tem de percorrer todo o conjunto de dados [29].

- ***Naive* Bayes**. O algoritmo *Naive* Bayes é um modelo probabilístico que utiliza o Teorema de Bayes, assumindo que os valores dos atributos são independentes entre si. Assim, a probabilidade de um item pertencer à determinada classe está representada na Equação 1 e o algoritmo associa o exemplo x à classe y_k para a qual

$P(y_k|x)$ é máxima [29].

$$P(y_i|x) \propto P(y_i) \prod_{j=1}^{d} P(x_j|y_i) \tag{1}$$

- **Árvore de Decisão**. Uma árvore de decisão apresentada é um grafo acíclico direcionado em que cada nó ou é um nó de divisão, com dois ou mais sucessores, ou um nó folha. Cada nó de divisão realiza um teste para decidir o caminho a ser percorrido. Os nós folhas são rotulados com valores, representando regras e conclusões [29]. Existem diversos algoritmos que utilizam a formalidade das árvores de decisão, sendo um dos mais conhecidos o C4.5 [39], que tem uma implementação em JAVA com a denominação "J48" [35].

- **Máquinas de Vetores de Suporte - SVM**. Foram desenvolvidas utilizando a teoria de aprendizado estatístico buscando-se hiperplanos que separem linearmente os conjuntos de dados [13]. A equação de um hiperplano é apresentada na Equação 2, em que $w \,.\, x$ é o produto escalar entre os vetores w *e* x, $w \in X$ é o vetor normal ao hiperplano descrito e $\frac{b}{||w||}$ corresponde à distância do hiperplano em relação à origem, com $b \in \Re$ [29].

$$h(x) = w \,.\, x + b \tag{2}$$

- **Redes Neurais Artificiais**. As redes neurais foram inspiradas no modelo de sinapses biológicas, tentando simular o cérebro humano. Elas são compostas por unidades de processamento simples, neurônios artificiais (foram apresentados pela primeira vez em McCulloch e Pitts [28]), dispostas em camadas e interligadas por um grande número de conexões. Quando todos os neurônios de uma camada estão conectados à camada seguinte, diz-se que é uma camada completamente conectada. Cada conexão tem um valor associado, chamado de peso.

  O neurônio artificial executa uma combinação linear de suas entradas com os pesos associados, seguido pelo cálculo de uma função de ativação não-linear sobre a combinação linear. Os pesos associados às conexões influenciam na importância de cada entrada. Dessa forma, a rede aprende por meio dos dados. Para aprender os pesos, a abordagem usual é o emprego do algoritmo de retropropagação juntamente com a técnica de otimização do gradiente descendente [29].

  As redes de uma única camada resolvem apenas problemas linearmente separáveis, mas adicionando-se camadas e utilizando funções de ativação não lineares nas camadas intermediárias, é possível resolver problemas não lineares, tornando-se um aproximador universal. As redes multicamadas também são chamadas de redes perceptron multicamadas (MLP, do inglês *multilayer perceptron*) [29]. Um exemplo de rede MLP pode ser observado na Figura 3.2.

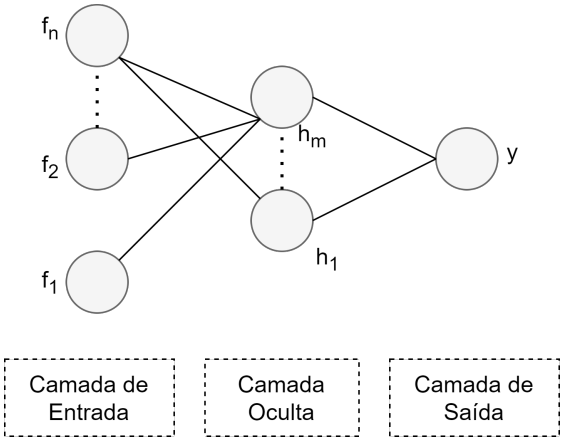


Figura 3.2. Rede MLP. Figura adaptada de Li et al. [23]

As redes neurais profundas são aquelas que possuem diversas camadas ocultas, podendo haver dezenas delas [2]. O aprendizado realizado por essas redes é chamado de *Deep Learning* possuindo algoritmos que apresentam excelentes resultados para uma variedade de problemas. Porém, tem como desvantagens o tempo de treinamento e a necessidade de uma grande quantidade de dados. Essas redes também são utilizadas para aprendizado de características especificas dos dados a serem analisados, eliminando a etapa de engenharia de atributos [1].

- **Redes Neurais Convolucionais (CNN)**. Similarmente às redes neurais, elas foram inspiradas na capacidade do cérebro de detectar e processar textos, imagens e vídeos. A etapa convolucional dessas redes realiza a detecção de atributos aplicando filtros que reduzem o tamanho da entrada, permitindo que o processamento seja mais rápido [6]. Elas utilizam pelo menos uma camada que realiza a operação de convolução [2]. Após isso, existe uma etapa conhecida como *pooling*, para reduzir a dimensionalidade e levar em consideração as características espaciais. As redes convolucionais podem ser utilizadas em uma gama diversificada de aplicações como detecção de objeto, classificação, processamento de texto e classificação de imagens. Dessa forma, as útlimas camadas da rede é que irão transformar a representação matricial em único vetor que determinará a saída da rede [34]. É comum essas redes serem utilizadas no domínio de duas dimensões, no caso de imagens, mas podem ser empregadas em problemas unidimensionais como os casos de séries temporais [6]. Um exemplo de rede convolucional pode ser visto na Figura 3.3.

- **Redes Neurais Recorrentes (RNN)**. Foram projetadas com o propósito de aprimorar as redes neurais fazendo com que fosse possível observar eventos anteriores. São compostas de unidades que conseguem observar a sequência dos elementos que passam pela rede [6]. São ideais para dados sequenciais como sentenças de textos, séries temporais e outras sequências como mapeamentos genéticos [2]. Elas possuem um estado oculto que guarda informações que irão interagir com o próximo elemento na sequência. A arquitetura básica pode ser vista na Figura 3.4.

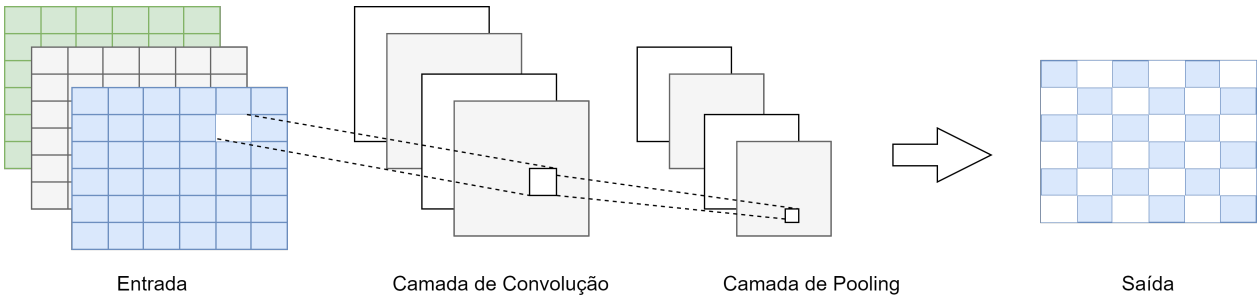


Figura 3.3. Rede Convolucional. Figura adaptada de Li et al. [23]

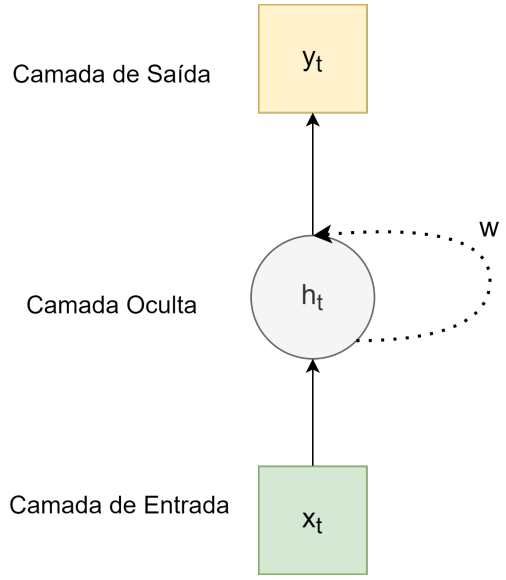


Figura 3.4. Rede RNN. Figura adaptada de Li et al. [23]

3.2.2. Qualidade de Serviços de Rede

Com a popularização de serviços oferecidos através da Internet e com a consequente convergência de infraestrutura de comunicação na web, além do desenvolvimento e utilização de diversos tipos de redes, por exemplo redes móveis, sem fio e dispositivos IoT, surgiu a preocupação com a qualidade de serviço da rede. Dessa forma, diversas métricas podem ser definidas para caracterizar o tráfego como *delay*, *jitter*, largura de banda e perda de pacotes [16]. Esses parâmetros são utilizados para medir a qualidade de serviço (QoS) em um fluxo de dados.

- **Jitter**. É a diferença entre o tempo estimado de entrada e o tempo real de entrada. Sinaliza uma variação no atraso da entrega dos dados, causando descontinuidade na sincronização do fluxo [32].
- ***Delay***. Tempo que o pacote leva para ir da origem até o destino. Essa métrica pode ser otimizada com a utilização de buffers [32].
- **Largura de banda**. Quantidade de dados que podem ser transmitidos em um período de tempo [8].

- **Taxa de perda de pacotes**. Quantidade de pacotes perdidos em relação à quantidade de pacotes enviados [8].

Cada aplicação demanda diferentes requisitos de qualidade de serviço, assim as métricas de QoS podem ser utilizadas para verificar a performance da rede. Portanto, para uma avaliação eficiente da rede é necessário que diferentes aplicações possuam distintos requisitos de QoS. Provedores de serviços elaboram acordos de níveis de serviço (SLA) comprometendo-se em garantir QoS para os usuários finais. Cada SLA acaba gerando especificações de níveis de serviço que irão guiar o monitoramento do QoS e SLA [43]. Uma das formas para se garantir o SLA é realizando classificação do tráfego de forma a prover os requisitos de QoS necessários em cada caso.

As métricas de qualidade foram projetadas para serem objetivas, focadas na avaliação do cumprimento dos requisitos de rede, não incluindo elementos qualitativos e nem subjetivos, por isso, em alguns casos, mesmo que os requisitos de QoS tenham sido cumpridos o usuário pode não ter ficado satisfeito [8, 16]. Por isso, foram desenvolvidas métricas de qualidade de experiência do usuário (QoE). Nessa busca pela satisfação do usuário, adota-se postura preventiva ao invés da reativa. Dessa forma, procura-se estimar os valores de QoS para que caso seja detectada uma degradação, medidas corretivas possam ser adotadas [41].

Ray [41] propôs uma rede formada por pacotes cognitivos. Esses pacotes teriam a capacidade de se rotearem e se comunicarem, buscando melhorar os valores de QoS, consumo de energia e até mesmo segurança. Esses pacotes cognitivos seriam ideais para utilizar em redes de IoT em que os dispositivos estão conectados entre si e o gerenciamento centralizado é dificultado. Esses pacotes utilizam algoritmos de aprendizado com redes neurais recorrentes para mapear atributos cognitivos e enriquecer a qualidade de serviço da rede.

Na busca pela performance das redes, além de medir as métricas de QoS pode-se medir a degradação da qualidade. Assim, em Nguyen e Thai [33] foi formulado o problema da degradação de QoS e os parâmetros pelos quais ele pode ser caracterizado. Além disso, foram propostos quatro algoritmos que tentam evitar esta degradação utilizando Aprendizado de Máquina. Os resultados foram otimistas encontrando limitações quanto ao tempo de execução, ou seja, nessa avaliação de QoS é interessante que as soluções possam ser aplicadas em tempo real [33].

3.2.3. Qualidade de Experiência do Usuário

A qualidade de experiência do usuário (QoE) é um indicador qualitativo da satisfação do usuário final com os serviços e aplicações utilizadas. Essa medida, por muitas vezes, é focada em avaliações subjetivas, interpretada em termos de percepção do usuário. Nessas avaliações, o usuário classifica o serviço ou aplicação com uma nota através de entrevistas ou questionários [21].

Esse conceito de QoE é muito importante, direcionando os investimentos e projetos de arquitetura na área de multimídias, principalmente nos serviços de vídeo para os usuários finais. Por exemplo, os grandes provedores de *streaming* armazenam os vídeos com diferentes qualidades de transmissão para que a experiência do usuário seja

otimizada, mesmo que os parâmetros de QoS da rede flutuem [8].

As métricas de QoE são afetadas por diversos fatores que podem ser divididos em três categorias principais: humanos, sistema e contexto. Dentre os fatores humanos que afetam a qualidade de experiência do usuário encontram-se o nível de escolaridade, idade, gênero, personalidade, humor, condições sócio-econômicas e até mesmo experiências de vida. Os fatores de sistema estão baseados em configurações técnicas e relacionadas ao conteúdo. Por exemplo, no caso de vídeos podemos citar a resolução, sincronização e as métricas de QoS. Os fatores de contexto relacionam-se ao ambiente em que o usuário recebe o conteúdo [8].

O conceito de QoE pode direcionar o entendimento e análise das operações de rede sobre a ótica do usuário final. Sendo, assim, um indicador importante para o controle da qualidade dos serviços, permitindo encontrar medidas que afetem a funcionalidade das redes complementando as métricas tradicionais de QoS [21]. Mesmo tendo naturezas distintas, QoS e QoE têm alto grau de correlação, por isso diversos estudos na literatura pesquisam a relação entre elas. Modelos de AM foram desenvolvidos para estimarem as métricas de QoE baseados nos valores de QoS como será visto em 3.3.4.

3.3. Uso de Aprendizado de Máquina em Metrologia de Redes

Nesta seção, trataremos os aspectos de Metrologia de Redes aplicados a áreas de redes sem fio e mobilidade, caracterização de topologia em diferentes níveis e redes definidas por software. Além disso, incluiremos discussão sobre medições voltadas aos usuários finais, principalmente as relacionadas com QoE.

As técnicas de metrologia buscam medir e monitorar o tráfego de redes nos seus diferentes níveis de granularidade, obtendo conhecimento da operação das redes, descobrindo como elas funcionam e analisando a performance delas em termos de QoS e QoE. Além disso, nessa utilização de técnicas para caracterização das redes faz sentido descobrir como os usuários consomem os serviços e como otimizar os recursos para melhor atendê-los. Unindo as informações a respeito da infraestrutura e do comportamento dos usuários é possível, propor, controlar e gerenciar acordos de níveis de serviço apropriados [1].

O crescimento da quantidade de redes móveis e celulares desperta o interesse em medições do desempenho fim-a fim dessas redes e o seu impacto em aplicações móveis [15]. Dentre essas medições encontramos aquelas baseadas em qualidade de experiência do usuário que constitui uma grande fonte de informações sobre o gerenciamento da rede. Nesse contexto, a implementação de Inteligência Artificial e Aprendizado de Máquina aumenta a eficiência do processo de monitoramento [21].

Abordaremos, ainda, de que forma podemos utilizar os parâmetros objetivos de qualidade do serviço tais como *delay*, *throughput*, perda de pacotes e *jitter* para estimar e predizer a qualidade de experiência do usuário utilizando Aprendizado de Máquina [21]. Além disso, é possível utilizar a Inteligência Artificial para estimar os múltiplos indicadores de qualidade de experiência do usuário, mesmo em uma rede que adote criptografia fim-a-fim [49, 27].

A caracterização de topologias sempre foi um dos objetivos das pesquisas de mo-

nitoração da Internet, que buscou ao longo dos anos a implementação e aperfeiçoamento de técnicas de descoberta de topologias nas camadas física, de rede e overlay. A descoberta de topologias em todos esses níveis é bastante útil no estudo de novos protocolos e serviços e análise das infraestruturas existentes. Então, métodos de Aprendizado de Máquina auxiliam na aquisição do conhecimento sobre o funcionamento das redes, o entendimento de como são as estruturas de redes existentes, permitindo estabelecer novas redes com menor esforço.

Dentre os diversos problemas de Metrologia de Redes que podem se beneficiar da utilização do Aprendizado de Máquina, podemos citar detecção de ataques de redes, predição de qualidade de experiência do usuário, detecção de anomalias em aplicações, estimativa de latência da rede [31] e falhas de redes. Nesse sentido, Zanotelli et al. [55] apresentam estudo para detecção de falhas na rede Ipê utilizando redes neurais, encontrando bons resultados quando regionalizaram os dados e as detecções. Sendo a globalização, ou seja, encontrar um modelo que conseguisse detectar falhas em qualquer ponto da rede nacional, um problema em aberto.

Em Streit et al. [48], os autores utilizam informações de tráfego residencial, fornecidas por um Provedor de Serviços de Internet, para analisar o tráfego de usuários domésticos. Neste trabalho, especificamente, eles comparam a QoE dos usuários de um período imediatamente antes com o período imediatamente depois do início da quarentena da COVID-19, em cidades do estado do Rio de Janeiro. Para isso, utilizam técnicas de decomposição tensorial, clusterização e classificação para identificar perfis de tráfego residencial dos clientes.

Dentre diversas aplicações, identificam-se trabalhos que utilizam medidas de telemetria de rede em plano de dados programável, em conjunto com modelos de Aprendizado de Máquina, para estimar métricas de qualidade de serviço [4]. Além dos desafios inerentes à Metrologia de Redes, discutiremos o estado da arte da medição em redes definidas por software (SDN), pontuando as principais características e dificuldades encontradas. Nota-se que a utilização de Aprendizado de Máquina para medições em rede é uma tendência atual. Diversos trabalhos têm lançado mão dessas técnicas para resolver problemas de medições de redes.

Entretanto, os dados capturados da rede para análise com os modelos de AM possuem características específicas que não são encontradas em aplicações mais comuns de Aprendizado de Máquina. Entre elas está a heterogeneidade, pois existe uma infinidade de dispositivos diferentes que podem estar conectados às redes estudadas, cada um deles consumindo e gerando tipos de tráfegos distintos [1]. Grande parte dos dados capturados estão criptografados, mantendo a privacidade dos usuários e a segurança das redes, porém dificultando a utilização desses dados para análises.

3.3.1. Definição de Fluxo de Tráfego

A maioria dos métodos de Aprendizado de Máquina aplicados em Metrologia de Redes utiliza o conceito de fluxo de tráfego. Esse conceito pode ser definido de diferentes formas. Neste minicurso utilizaremos a definição apresentada em Wei Wang et al. [54], por ser a mais utilizada na literatura.

Assim, um fluxo de tráfego é definido por meio dos pacotes que possuam os mesmos elementos, formando uma tupla : IP de origem, porta de origem, IP de destino, porta de destino e protocolo de transporte. Já uma sessão inclui fluxos em ambas direções, ou seja, temos o fluxo origem $\rightarrow$ destino e destino $\rightarrow$ origem [54]. Formalizando, temos:

- Tráfego bruto: o conjunto de pacotes é definido por $P = \{p^1,...,p^{|P|}\}$, onde cada pacote $p^i \in P$ é definido como $p^i = (x^i,b^i,t^i), i = 1,2....,|P|$. O primeiro elemento é a tupla x^i, o segundo elemento é o tamanho do pacote $b^i \in [0,\infty)$ em bytes e o último elemento $t^i \in [0,\infty)$ em segundos é o instante de início da transmissão.
- Fluxo de tráfego: é um subconjunto de um conjunto de tráfego bruto P, definido como $f = (x,b,d_t,t)$. O x representa a tupla, o b é a soma do tamanho de todos os pacotes em um fluxo, $d_t = t^n - t^1$ é a duração do fluxo e t é o início da transmissão do primeiro pacote.

3.3.2. Classificação de Tráfego

A tarefa de classificação de tráfego permite identificar os protocolos ou aplicações de determinado fluxo de tráfego, sendo uma importante ferramenta de gerenciamento que pode ser aplicada em tarefas tais como monitoramento de redes, provisionamento de qualidade de serviço, prioridade de tráfego, detecção de intrusão e aplicações de segurança [5, 3]. Existem três principais abordagens para essa tarefa, a saber, baseada em portas, baseada no conteúdo (payload) e baseada em informações estatísticas do fluxo [52].

- **Classificação baseada em portas.** A classificação baseada em portas identifica o tráfego de acordo com a padronização de portas de cada aplicação. Porém, este método é ineficiente ao lidar com a alocação de portas dinâmicas [5].
- **Classificação baseada em conteúdo.** A classificação baseada em conteúdo, também conhecida como *Deep Packet Inspection (DPI)*, identifica características do conteúdo dos pacotes (sequência de bytes) que diferem um protocolo de outro, sendo consideradas como a assinatura dos tipos de tráfego [42]. Essa abordagem enfrenta problemas de privacidade do conteúdo e não lida bem com fluxos criptografados, além de ser computacionalmente custosa [5].
- **Classificação baseada em fluxo.** A abordagem baseada em fluxo permite utilizar as características estatísticas para classificar o tráfego, permitindo a utilização de modelos de AM para identificar padrões existentes em cada fluxo. Essa abordagem utiliza parâmetros independentes, tais como comprimento dos pacotes, tempo de intervalo entre as chegadas e duração do fluxo. Dessa forma, ela consegue lidar com o problema da alocação dinâmica de portas, já que não depende das portas pelo qual o tráfego foi transmitido, também com o fluxo criptografado [52].

Um dos contextos em que estão sendo estudadas diversas aplicações de AM com classificação de tráfego refere-se às *smart cities*. As aplicações de Internet das Coisas (IoT do inglês *Internet of Things*) interconectam diversos objetos, aparelhos, sensores e dispositivos inteligentes criando uma rede complexa e que processa grande quantidade de

dados. Essas tecnologias são essenciais para a construção de *smart cities*, oferecendo serviços em diversas áreas de atuação, como educação, saúde, transporte e casas inteligentes [3]. Essas diferentes aplicações geram uma grande quantidade de dados e possuem diferentes requisitos de qualidade de serviço (QoS), como largura de banda, taxa de perda de pacotes, taxa de *delay* e *jitter* [5]. Assim, a classificação de tráfego permite implementar um mecanismo que diferencie o fluxo de tráfego de acordo com o tipo de aplicação (Ex: e-mail, jogos, *streaming*). Por fim, os recursos da rede podem ser alocados de forma a garantir os requisitos de QoS [5].

Nesse contexto, AlZoman e Alenazi [5] realizaram a comparação de quatro modelos de AM para classificação de tráfego, utilizando um conjunto de dados com características de tráfego semelhantes às *smart cities* como múltiplas fontes, vários tipos de dados e grande quantidade de tráfego. No conjunto de dados, existiam 11 tipos de tráfego e suas respectivas aplicações como navegação na web, acesso a banco de dados, jogos em rede, tráfego de ataques de segurança, multimídia entre outros. Os autores encontraram a melhor performance com algoritmos de árvore de decisão, com acurácia de 99.18%, superando os métodos tradicionais de classificação de tráfego. Ao avaliar a performance dos modelos, os autores compararam o tempo de execução dos diferentes algoritmos de AM, não utilizando apenas as métricas tradicionais, como acurácia, precisão e f1-Score, pois em aplicações de tempo real, o *delay* pode ser tão ou mais importante do que essas métricas [5].

A classificação de tráfego utilizando AM vêm se tornando um método de alta performance, embarcando inteligência nas funções de rede e melhorando o seu gerenciamento [5]. Apesar dessa evolução e da capacidade de lidar com uma grande quantidade de dados, ainda existem desafios na aplicação desses modelos e um deles é lidar com uma grande quantidade de atributos de entrada, ainda mais, quando eles são esparsos. Para solucionar este problema, uma das técnicas empregadas é a seleção de atributos que permite identificar os atributos essenciais para melhorar a performance dos modelos. Nesse processo de seleção, utiliza-se uma função de importância dos atributos que calcula o ganho de informação de todos os atributos e retorna os atributos com ganhos mais elevados, como visto em Alhumyani et al. [3].

Em Jonathan, Misra e Osamor [18], os autores compararam a aplicação de modelos de AM para classificação de tráfego sob duas perspectivas: executando diferentes métodos de seleção de atributos e sem realizar a seleção de atributos. Os resultados demonstram que a seleção de atributos possibilitou incrementar a performance dos modelos [18]. Dessa forma, a adoção de uma análise prévia do problema a ser tratado para a identificação da seleção de atributos ideal pode ser considerada como uma boa estratégia.

Além dos modelos tradicionais de AM, os pesquisadores têm proposto diversos modelos que se adaptem melhor aos tipos de dados utilizados. Pesquisas recentes utilizam redes neurais profundas (DNN) para classificação de tráfego, pois estas redes permitem a captura de características (que não são perceptíveis aos humanos) e conseguem lidar com grande quantidade de dados. Uma DNN possui uma camada de entrada, múltiplas camadas escondidas e uma camada de saída. Assim, em Alhumyani et al. [3] os autores desenvolveram uma DNN com sete camadas escondidas e um classificador que utiliza entropia máxima como função de custo na camada de saída para classificar o tráfego em

diferentes classes.

Resultados experimentais mostram que a rede proposta atinge acurácia máxima de 99.23%, superando alguns modelos tradicionais de AM, como SVM e KNN [3]. Para melhorar a performance, os autores realizaram uma seleção de atributos com o algoritmo *extra-trees* que cria diversas árvores de decisão para realizar a classificação. Assim, utilizando-se uma função de ganho de importância obtiveram-se os atributos mais importantes para melhorar o classificador, resultando em um total de 12 atributos, aplicando-os com a rede DNN proposta.

Outro modelo de rede neural utilizada em classificações de tráfego são as denominadas *autoencoders*. Essas redes são construídas com a mesma dimensionalidade nas camadas de entrada e de saída, mas as camadas intermediárias são implementadas com dimensionalidade reduzida. O objetivo desses modelos é reconstruir a informação da entrada na saída, a partir de transformações codificadoras e decodificadoras realizadas pelas camadas intemediárias. Espera-se que com esse processo, ocorra o aprendizado de outras características que permitam otimizar os classificadores [2].

Em Li et al. [24] é proposto um *stacked autoencoder* aprimorado para aprender as relações complexas sobre fontes múltiplas. Este modelo foi proposto para possibilitar que o modelo consiga lidar com a incerteza de grandes fluxos de redes. Especificamente, para modelar a incerteza, a rede é treinada utilizando uma estratégia não supervisionada que aplica a teoria Bayesiana de probabilidade para calcular as distribuições dos parâmetros do modelo. Os resultados demonstram que o modelo proposto supera os tradicionais em termos de acurácia.

O *stacked autoencoder* é treinado em duas fases. Na primeira, chamada de pré-treinamento, os parâmetros de cada camada oculta são aprendidos utilizando um aprendizado não supervisionado camada por camada. Na segunda fase, esses parâmetros são ajustados utilizando um método de aprendizado supervisionado para capturar as características finais. Uma importante propriedade do modelo proposto é a capacidade de lidar com a incerteza dos dados de uma maneira natural [24].

Redes Definidas por Software (em inglês, *Software Defined Networks* - SDN) é um paradigma de redes que permite a virtualização da infraestrutura de rede, desacoplando os planos de controle e dados e criando uma arquitetura dinâmica e flexível [52]. Além da camada de controle e da camada de dados, existe a camada de aplicação, completando a arquitetura SDN. O controlador central gerencia o fluxo de dados e conhece a topologia da rede, permitindo um controle de fluxo eficiente e facilitando as medições nessa camada [52]. Apesar disso, o gerenciamento de QoS fim-a-fim pelo controlador central pode ser dificultado devido aos diferentes requisitos de cada aplicação e usuários finais. Surge, então, a necessidade de se medir e monitorar as demais camadas de uma rede SDN [42].

Wang et al. [52] propuseram um arcabouço que potencializa os conceitos de SDN, permitindo monitoramento e gerenciamento em camadas separadas e garantindo controle e flexibilidade com otimização de recursos computacionais. Além disso, desenvolveram redes neurais denominadas Datanets baseadas em três arquiteturas de redes neurais: *multilayer perceptron*, *stacked autoencoder* e redes neurais convolucionais. As Datanets permitem classificar o tipo de tráfego, mesmo em redes criptografadas, sem ter acesso ao

conteúdo, mantendo a privacidade dos dados [52]. As Datanets são posicionadas no controlador SDN que realizam a classificação de tráfego. Este controlador central, ainda, cria e atualiza periodicamente essas redes conforme os dados são coletados dos diversos tipos de tráfego. Dessa forma, é possível realizar medições de QoS, monitoramento de redes, estimativa de QoE e detecção de malwares [52]. Assim, o controlador consegue entregar a QoS necessária para cada aplicação. Das redes criadas no estudo de Wang et al. [52], as redes convolucionais obtiveram os melhores resultados, com acurácia acima de 98%.

Para classificação de tráfego criptografado existem três abordagens principais. A primeira, diferencia tráfego criptografado do tráfego não criptografado. A segunda abordagem permite identificar o aplicativo que está gerando o fluxo de tráfego. A terceira abordagem visa identificar o tipo de tráfego, por exemplo, é possível determinar se o fluxo refere-se a um e-mail, a um serviço de streaming ou a uma rede social [53]. Quando redes neurais são utlilizadas para resolver problemas de classificação de tráfego, é possível aproveitar o conhecimento gerado em um campo distinto de aplicação para resolver problemas de rede. Nesse sentido, Wang et al. [53] propuseram um modelo fim-a-fim que aprende automaticamente atributos preditivos do fluxo de tráfego bruto aplicando uma modelagem feita em problemas de Processamento de Linguagem Natural (PLN), não necessitando da etapa de seleção de atributos. Na proposta dos autores, o tráfego de rede é sequencial e hierarquizado, estruturado em bytes, pacotes, sessões e fluxo de tráfego, que serve para comunicação de dois ou mais agentes, de maneira similar aos modelos de linguagem. Assim, há uma relação com a estrutura de caracteres, palavras, sentenças e artigos completos.

Então, utilizando-se de conhecimento para classificação de texto com redes convolucionais [20], Wang et al. [53] alimentam uma rede convolucional de uma dimensão com o tráfego bruto criptografado. A rede é alimentada com os primeiros 748 bytes de cada fluxo, extraindo os atributos preditivos e passando pela última camada que realiza a classificação de tipos de tráfego. Outra forma de modelo fim-a-fim em que não é necessário criar e selecionar atributos é proposta em Chen et al. [12]. Os autores propuseram uma rede convolucional de duas dimensões que é normalmente utilizada para classificar imagens. Dessa forma, o fluxo de tráfego é transformado em uma representação de imagens, adotando o *Reproducing Kernel Hilbert Space (RKHS)*. Após esse passo, a rede é alimentada com esses dados para realizar a classificação de tráfego. Esse modelo permite eliminar a etapa de engenharia de atributos não sendo necessário realizar atualizações periódicas da rede com novos exemplos de treinamento e lidar com fluxo de tráfego criptografado.

Outra abordagem para classificar tráfego está relacionada à aplicação de modelos de linguagens conhecidos como n-gramas. Zhao, Zhang e Sang [56] utilizam essa estratégia combinando *embeddings* de n-gramas, redes neurais e algoritmos de clusterização para construção de um esquema não supervisionado que resolve o problema de identificação de fluxo de tráfegos desconhecido. A etapa de aprendizado de atributos utiliza uma rede do tipo *autoencoder* que consegue aprender representações de dados não rotulados, como é o caso do fluxo de tráfego desconhecido nas redes. O tráfego dos pacotes é caracterizado via *embeddings* de n-gramas agrupando-os em cluster de tráfego desconhecido permitindo identificar dados não rotulados.

Mesmo com o avanço das aplicações de Aprendizado de Máquina na classificação de tráfego, em algumas situações a falta de uma base de dados rotuladas em tamanho suficiente impede que os modelos de AM alcancem resultados satisfatórios. Os resultados obtidos em Shahraki et al. [45] demonstram que o aprendizado ativo (AL, do inglês *Active Learning*) permite obter alto valor de acurácia, mesmo com uma quantidade pequena de dados. AL é um subcampo do Aprendizado de Máquina que tenta reduzir a quantidade de exemplos rotulados para o treinamento dos modelos. A ideia básica é consultar os exemplos rotulados de forma inteligente, buscando exemplos selecionados pelo algoritmo de AL para a construção do melhor modelo. Assim, um modelo aprendiz segue uma estratégia iterativa, selecionando exemplos de uma base não rotulada para que um oráculo, humano ou máquina, possa rotular.

Essa estratégia reduz o tempo e o custo de rotular todos os dados, pois utiliza uma quantidade significativa de menos exemplos [56]. Zhao, Zhang e Sang [56] demonstraram que os modelos implementados com AL superaram os modelos offline e alguns modelos que utilizavam redes neurais.

3.3.3. Predição de Tráfego

A predição de tráfego pode resolver diversos problemas de gerenciamento de redes, entre eles o provisionamento de recursos, a detecção de congestionamento e a tolerância a falhas. Apesar do sucesso obtido com a aplicação dos modelos de AM, ainda existem desafios a serem superados como alto custo computacional, dificuldade de treinamento e retreinamento, alta volatilidade dos dados e falta de dados rotulados [25]. As técnicas de predição de tráfego são aplicadas com diferentes escalas de tempo ou, até mesmo, em problemas independentes de tempo. Dessa forma, dividimos as aplicações de predição em curto prazo, quando se refere a escalas de tempo que vai de milissegundos a dias. Quando estamos falando de semanas, meses e anos chamamos de longo prazo [11].

O avanço da infraestrutura virtualizada possibilita a existência de inúmeras redes virtuais conectando diferentes dispositivos. Essas fatias de redes demandam serviços da infraestrutura de rede física subjacente, criando ambientes complexos e difíceis de controlar. Para resolver esse problemas foram propostas novas soluções que executam as tarefas de gerenciamento e alocação de recursos automaticamente, criando redes em que não há intervenção humana chamadas *zerotouch networks* [7]. Nessas redes autônomas é possível criar múltiplas instâncias lógicas da rede física, isolando o tráfego em cada uma delas. Assim, cada fatia é alocada para um tipo específico de tráfego de acordo com as aplicações. Dessa forma, é possível alocar diferentes serviços em uma mesma rede permitindo que eles coexistam [7].

A principal dificuldade imposta por essas redes autônomas é o gerenciamento de recursos. Essas fatias de redes exigem aumento dos requisitos de capacidade da rede, sendo necessário uma alocação de recursos eficiente, dinâmica e preemptiva para manter a infraestrutura operacional e controlar os custos da virtualização. Nesse contexto, em que as demandas de recursos mudam rapidamente, o tamanho e a complexidade das redes atingiram níveis em que não é possível ser gerenciada sob a ótica da percepção humana, uma orquestração e alocação de recursos orientada a dados vêm sendo implementada [7]. As redes virtualizadas são flexíveis e escaláveis, sendo muitas delas fornecidas como

software em serviços da Amazon AWS e Google Cloud. Apesar das vantagens e custos competitivos, nesses ambientes é comum ocorrer o erro de superdimensionamento, situação que aumenta a despesa sem necessidade pois os recursos alocados não são utilizados. Outro problema frequente é o subdimensionamento, em que poucos recursos são alocados, nesse caso o prejuízo financeiro é indireto, pois os serviços podem ficar fora de operação caso ocorram demandas que superem a capacidade da rede. Existe ainda o caso em que os recursos são alocados de forma correta, mas não se consegue mensurar o momento exato em que a infraestrutura deva ser reduzida ou aumentada, gastando mais do que o necessário [7].

O trabalho em Bega et al. [7] propõe um modelo que realiza a orquestração de redes virtualizadas e autônomas em duas escalas de tempo. Existe um orquestrador em escala de longo prazo que aloca uma capacidade dedicada para cada fatia e outro de longo prazo para capacidade compartilhada. Essas capacidades são constantes, porém cada fatia só tem acesso a sua capacidade dedicada e todas as fatias têm acesso às capacidades compartilhadas. Esse compartilhamento é gerenciado por um orquestrador de curto prazo, decidindo em que momento cada fatia terá esse recurso alocado. O framework apresentado chama-se AZTEC e é implementado em três blocos. O primeiro bloco é uma rede neural alimentada pelo fluxo de tráfego, transformado em imagens, que realiza a predição de longo prazo da capacidade dedicada. Esta rede consiste de três camadas 3D-CNN interligadas por camadas de *dropout*. O segundo bloco realiza a predição de longo prazo da capacidade compartilhada e possui uma rede neural com três camadas completamente conectadas com 128, 51 e 1 neurônio. O terceiro bloco possui uma rede semelhante à do primeiro bloco e realiza a predição da capacidade de curto prazo. Esse modelo proposto quando comparado com o estado da arte de orquestração possibilitou a redução de 47% do custo de alocação de recursos [7].

Outro problema preditivo que pode ser auxiliado com a utilização de AM é a predição do tamanho do fluxo em redes. Detectar fluxos muito grandes é importante para melhorar o roteamento, balanceamento e escalonamento nos diferentes tipos de redes. Essa detecção costuma acontecer após o congestionamento da rede devido ao fluxo. Aplicando modelos inteligentes é possível predizer o tamanho do fluxo de acordo com os dados do primeiro pacote, permitindo que o tráfego seja roteado e a rede não fique congestionada [37].

Esse problema foi atacado em Poupart et al. [37] que extrai os seguintes atributos dos primeiros pacotes: IP de origem, IP de destino, porta da origem, porta do destino, protocolo, identificação de servidor ou cliente e tamanho dos primeiros pacotes. Após essa extração, os autores analisaram regressores para determinar o tamanho do fluxo, encontrando os melhores resultados com um Processo Gaussiano de Regressão. Dessa forma, o modelo apresentado em Poupart et al. [37] permitiu estimar o tamanho do fluxo utilizando os primeiros pacotes e definir um limite para adoção de políticas de roteamento. Essa configuração permitiu que a rede funcionasse evitando congestionamentos devido ao tamanho dos fluxos. Outra vantagem desse modelo comparado às metodologias clássicas é que não existe a necessidade de alteração dos dispositivos de roteamento e conexão [37].

Outro modelo para calcular o tamanho do fluxo e tentar evitar congestionamentos é apresentado em Andreoletti et al. [6]. Eles empregam um modelo de AM baseado em

grafos chamado de Rede Neural Recorrente Convolucional de Difusão (DCRNN do inglês *Diffusion Convolutional Recurrent Neural Network*). A grande novidade desse modelo é capturar tanto as propriedades dos atributos, como as propriedades estruturais da rede. Esse modelo superou os demais modelos de redes neurais avaliados [6].

O objetivo é evitar a interferência humana no gerenciamento de redes, permitindo que sistemas inteligentes realizem tarefas de configuração, provisão, teste e detecção automaticamente. Dessa forma, Andreoletti et al. [6] tentam predizer o volume de tráfego em um enlace baseado na sequência histórica de tráfego. Então, a rede DCRNN executa uma tarefa de regressão minimizando o erro absoluto médio. O processo de difusão representa a relação entre dois nós do grafo, especificamente é a probabilidade que um caminho de K passos comece no primeiro nó e termine no segundo. A ideia das redes DCRNN é que a modelagem do processo de difusão permita descobrir a influência de cada nó para o resultado da predição. Assim, a representação do nó dentro do espaço de atributos é aprimorada com a aplicação dos filtros das camadas convolucionais [6].

Em Andreoletti et al. [6] foi utilizado o backbone da rede Abilene [47] que possui topologia pública, de 12 nós e 30 links, bem como dados estatísticos de tráfego real. Utilizando esses dados foi possível demonstrar que a rede DCRNN supera as demais na predição de volume de tráfego, sendo uma ferramenta eficiente para evitar congestionamentos. Os avanços na aplicação de Inteligência Artificial aos problemas de redes e o aumento da complexidade das mesmas criam a necessidade de criar gerenciadores adaptativos capazes de garantir boa qualidade de serviço (QoS) e oferecer qualidade de experiência do usuário (QoE) com segurança e baixo consumo de energia [14].

Dessa forma, Gelenbe et al. [14] propõem o desenvolvimento de redes auto-conscientes capazes de eliminar a complexidade de programação, gerenciar tarefas, de autoconfiguração, monitorar, gerenciar e corrigir falhas com mínima intervenção humana. Essas redes são possíveis desde que haja uma IA capaz de predizer o tráfego melhorando o roteamento, aumentando a confiabilidade, segurança e resiliência. A arquitetura proposta é uma rede SDN que utiliza pacotes cognitivos [41] capazes de realizar medições entre os nós de origem e destino. Esses pacotes compartilham informações com o controlador SDN que possui um mecanismo de decisão baseado em aprendizado por reforço com redes neurais recorrentes para modificar os caminhos do fluxo dinamicamente atingindo a melhor qualidade de serviço possível. Esse roteamento inteligente é executado predizendo o tráfego na rede SDN [14]. Além dos parâmetros de QoS essa rede auto-consciente busca melhorar a segurança e o consumo de energia. Eles implementaram um protótipo para análise e os resultados obtidos demonstraram performance superior às redes convencionais em termos de QoS, segurança e consumo de energia [14].

Foi visto que é possível utilizar Inteligência Artificial com Aprendizado de Máquina para criar redes *zerotouch* em que não há intervenção humana. Essas redes autônomas são mais eficientes, possuem melhor tempo de resposta, além de serem seguras e poderem oferecer até uma melhor controle do consumo de energia [14]. O gerenciador e tomador de decisão baseia-se na predição de fluxo de tráfego, evitando congestionamento de pacotes garantindo requisitos de QoS e buscando uma melhor experiência para o usuário. Essa análise em função do QoE foi realizada em Gelenbe et al. [14] e as redes autônomas superaram as redes tradicionais.

3.3.4. Estimativa de QoE

A estimativa de Qualidade de Experiência do Usuário (QoE) utilizando AM baseia-se no mapeamento de um conjunto de parâmetros de QoS objetivos e mensuráveis da rede, tais como latência, vazão, perda de pacotes e jitter, em uma medida da QoE do usuário. Dessa forma, é possível aplicar funções matemáticas nas medidas de QoS, encontrando valores para a QoE. O sucesso dos modelos para realizar essa estimativa deve-se ao grau de correlação entre essas medidas. Esse mapeamento ainda permite eliminar parte da subjetividade encontrada na estimativa de QoE [21]. O monitoramento de QoE é uma tarefa desafiadora em qualquer rede, nas redes móveis esse desafio é ainda maior. Devido à grande quantidade de usuários e dispositivos conectados, bem como à dificuldade de análise do tráfego de rede disponível. Além disso, a segurança e privacidade impõem outro desafio que é lidar com dados criptografados nas transmissões fim-a-fim.

O trabalho em Casas [9] investigou esse problema, realizando a inferência de QoE com a métrica *Speed Index* (SI) através de modelos de AM usando como entrada apenas dados de nível de pacote. Nele, os autores avaliam a QoE da navegação web em aplicativos de dispositivos móveis, como smartphones e tablets. O carregamento de uma simples página web pode envolver conteúdos que estejam localizados em diferentes servidores e diferentes fontes, por exemplo, é comum que uma página html exibida ao usuário possua dados provenientes de diferentes bancos de dados. Então, nessas situações a rede pode impactar a experiência do usuário. A abordagem para a solução do problema consiste na utilização de modelos supervisionados, por isso é necessário que o conjunto de dados contenha exemplos rotulados. Os atributos preditivos foram coletados e extraídos de tráfego criptografado no acesso das páginas web pelos dispositivos móveis [9].

A métrica SI indica o tempo necessário para que o usuário consiga visualizar a página web, mesmo que essa página não esteja completamente carregada. Por exemplo, se a imagem do rodapé demorar a carregar e o usuário for lendo e observando os outros elementos da página, ele pode não perceber essa demora. O que importa na experiência do usuário é que ele consiga visualizar os itens desejados, qualquer atraso no carregamento que não esteja visível será imperceptível. Os modelos foram treinados para inferir o SI como um problema de regressão, analisando os dados obtidos do fluxo de pacotes. Como existe diferença de QoE entre os diferentes tipos de dispositivos, os autores analisaram a inferência em três experimentos sendo que no primeiro eles utilizaram os smartphones, no segundo experimento foram utilizados apenas tablets. O terceiro experimento foi realizado com múltiplos dispositivos. Dessa forma, foi possível concluir que a abordagem baseada em AM para estimativa de QoE é confiável, produzindo uma baixa taxa de erro, além de conseguir utilizar atributos extraídos diretamente dos dados criptografados [9].

Essa estimativa de QoE a partir das métricas de QoS aplica-se a diversos casos, não estando restrita a um determinado tipo de conteúdo. Os autores em Ul Mustafa, Moura e Rothenberg [49] analisaram esse mapeamento utilizando AM no contexto de redes 5G para transmissão de vídeos. Nesse trabalho, eles verificaram o impacto na QoE dos usuários quando assistindo vídeos transmitidos com HTTP adaptive streaming (HAS), utilizando HTTPS para entregar criptografia fim-a-fim. Foi desenvolvido um algoritmo para extração de atributos de QoS que podem ser utilizados para mapear a QoE, com acurácia de 91%.

As transmissões de vídeos ao vivo (LIVE) cresceram enormemente nos últimos anos, principalmente durante a pandemia do COVID-19. Há uma variedade de tipos de vídeos como shows, transmissões de eventos esportivos, streaming de vídeos games, até aulas nas plataformas Twitch e YouTube. Essas transmissões ao vivo são mais suscetíveis a problemas de congestionamento de rede do que os vídeos sob demanda (VOD), necessitando de requisitos específicos para obter melhores medidas de QoE. Entretanto, existe um desafio para esse monitoramento devido a utilização da mesma infraestrutura para as transmissões ao vivo e VOD. Dessa forma, Madanapalli et al. [26] desenvolveram um método de aprendizado de máquina para detectar os vídeos que são transmitidos ao vivo e mensura a QoE baseado nas seguintes características de cada pedaço do fluxo: carimbo de tempo para o instante da requisição, início de transmissão do primeiro pacote, fim de transmissão do último pacote, número de pacotes, quantidade de bytes transmitidos.

O fato dos operadores de rede utilizarem a mesma infraestrutura para as transmissões de vídeos VOD e as transmissões ao vivo, dificulta a inspeção DPI que permitiria distingui-los. Porém, a extração de atributos do comportamento dos pacotes permite a construção de um modelo que realize essa separação. Madanapalli et al. [26] desenvolveram um método para estimar a resolução dos vídeos inferindo a medida de QoE, além disso eles apresentaram uma forma de detectar a presença de paradas de buffer, contribuindo para a estimativa de QoE. Tradicionalmente, a modelagem e medidas de QoE eram realizadas através da análise dos logs HTTP. Nessa abordagem, no entanto, os autores fornecem um preditor que atua em tempo real e com fluxo de tráfego criptografado. Assim, mesmo aqueles provedores que não têm acesso aos logs conseguem inferir a QoE das transmissões.

Poucos estudos são realizados para inferência de QoE em transmissões ao vivo, concentrando a maior parte dos trabalhos em vídeos VOD. Vídeos LIVE são gravados e transmitidos em tempo real, possuindo características distintas e requisitos específicos. A começar pelo buffer que é menor para os vídeos ao vivo, o usuário vai solicitando requisições HTTP e armazenando poucos segmentos do vídeo diminuindo a latência entre a transmissão e visualização pelo usuário final [26]. Já os vídeos VOD realizam requisições a um servidor que armazena os vídeos em diferentes resoluções pré-codificadas, permitindo sofisticados esquemas de compressão, bem como que o cliente mantenha um buffer maior evitando a deterioração das métricas de QoE. Outra diferença é que os clientes de transmissões ao vivo realizam downloads de segmentos do vídeo a cada dois segundos, enquanto que para vídeos VOD esse tempo sobe para dez segundos. Essa diferença de periodicidade no download de segmentos é uma das principais características que permite distinguir os dois tipos de vídeos [26].

Foi desenhada e implementada uma rede LSTM que aprende características comportamentais do fluxo de rede, evitando que seja necessário realizar a etapa de engenharia de atributos. A rede LSTM recebe um vetor de série temporal com a contagem das requisições de pacotes e passa por um classificador MLP que diz a probabilidade do fluxo ser uma transmissão ao vivo. Após isso, utiliza-se uma outra rede neural para predizer a parada de buffer, estimando o QoE entregue ao usuário [26]. Uma rede LSTM mantém um estado oculto e uma célula de estado. O estado da célula age como uma memória relembrando informações que serão utilizados na tarefa de classificação. O estado oculto é um canal de saída, que seleciona as informações da célula de estado que irão para o

classificador como visto na Figura 3.5.

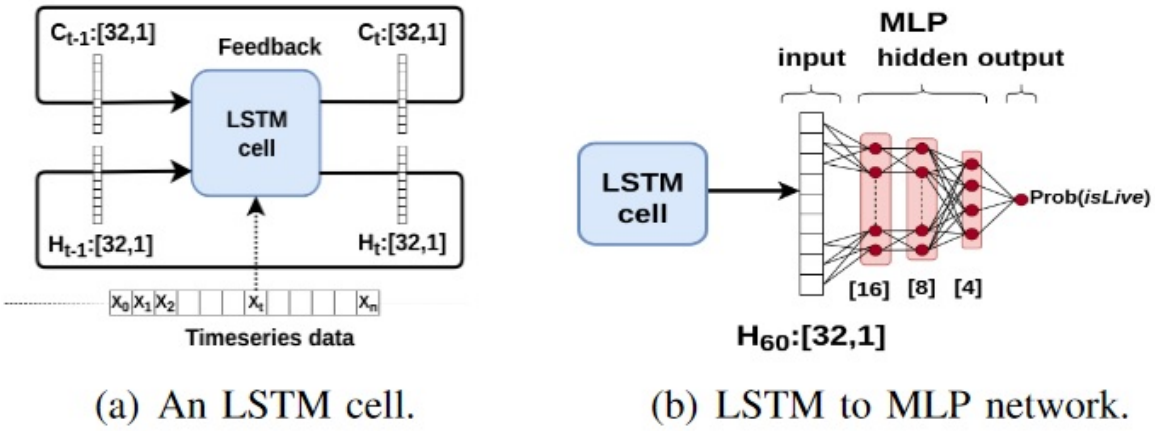


Figura 3.5. Rede LSTM Madanapalli et al. [26]

Com o crescimento e popularização da distribuição de vídeos na Internet, os usuários tornaram-se mais exigentes desejando assistir os vídeos online em alta resolução. Dessa forma, a QoE para os serviços de multimídia tornou-se uma métrica importante, já que os usuários esperam encontrar nos vídeos transmitidos a mesma qualidade encontrado no mundo offline. Sabe-se, no entanto, que diversos fatores influenciam a qualidade de uma transmissão através da rede [22].

Os provedores de serviços monitoram constantemente os parâmetros de rede para entregar a melhor experiencia para usuário. Entre as diversas técnicas encontra-se o mapeamento de qualidade de serviço para estimar a QoE, otimizando a qualidade dos vídeos. Porém, existem outros fatores que influenciam nessa experiência como o contexto social em que o usuário está inserido. Laiche et al. [22] estudam os fatores sociais e de engajamento de usuários que podem ser medidos e utilizados como atributos para estimar a QoE com modelos de AM, obtendo alta acurácia. A maior parte das métricas de QoE são subjetivas e refletem o grau de satisfação ou desapontamento do usuário. A mais utilizada é a MOS (do inglês mean opinion score), que é uma nota dada pelo usuário relacionada a experiência que ele experimentou. Esta medida é padronizada para ITU-T. O estimador proposto em Laiche et al. [22] tenta predizer o valor MOS de acordo com os atributos de contexto social.

A proposta dos autores em Laiche et al. [22] é que a contagem de visualizações, comentários e notas que ficam visíveis nas redes sociais refletem o grau de satisfação dos usuários com os vídeos relacionados. Pouco é conhecido sobre os efeitos da popularidade e do engajamento dos usuários sobre a medida final de QoE. Então as métricas de popularidade e engajamento nas redes sociais, foram utilizadas para predizer a qualidade de experiência do usuário. Assim, eles puderam identificar a influência que essas métricas possuem em relação à estimativa de QoE.

Laiche et al. [22] analisaram o modelo proposto com três algoritmos de AM: KNN, Random Forest e Árvore de Decisão. Foi encontrado o melhor resultado com a

árvore de decisão. Esse trabalho é relevante pois demonstrou que as informações do contexto social, popularidade e engajamento, contribuem para a estimativa de QoE. Os autores sugerem como trabalho futuro a utilização de redes neurais para incrementar a performance, além de demonstrar que o modelo proposto pode ser integrado em redes SDN.

3.3.5. Segurança de Redes

Um ataque de negação de serviço distribuído (DDoS do inglês *Distributed Denial of Service*) persiste como um problema de segurança das redes. Existem diversas abordagens para detectar tais ataques, incluindo a utilização de métodos baseados em AM. Os modelos propostos utilizam uma seleção manual de atributos baseada no entendimento de especialistas sobre o tema, o que gera um problema de generalização e os detectores acabam ficando muito especializados no contexto do ataque [51].

Ataques de DDoS enviam uma grande quantidade de tráfego para o sistema alvo, geralmente com a utilização de botnets. Esses bots frequentemente escravizam dispositivos que estão conectados à Internet para realizar esses ataques, inclusive beneficiando-se da enorme quantidade de dispositivos de IoT que acabam sendo fáceis de capturar. Essa grande quantidade de tráfego impede que os usuários reais tenham acesso ao serviço. Um dos grandes desafios para detectar esse tipo de ataque está associado ao fato dos agentes maliciosos utilizarem pacotes de dados normais, ou seja, não há indicação de tráfego malicioso, o que dificulta a utilização de classificadores de tráfego [51].

Wang, Lu e Qin [51] propuseram um classificador MLP para detectar ataques DDoS combinado com uma seleção automática de atributos com MLP que tenta encontrar um conjunto ótimo de atributos que incremente a performance do modelo detector do ataque. Além disso, no modelo proposto existe um mecanismo de feedback para detectar o momento em que o classificador deva ser atualizado. Os resultados encontrados demonstram que o modelo proposto é comparável aos demais encontrados na literatura com a vantagem de corrigir o detector quando a performance se deteriora. Esta é uma grande dificuldade dos modelos propostos com a utilização de AM, a acurácia obtida com os dados de treinamento não se reflete quando o modelo é posto em produção, em contato com dados do mundo real.

O modelo proposto utiliza uma classificação binária utilizando um classificador MLP. Na etapa de treinamento é um utilizado o algoritmo SBS (do inglês *Sequential Backward Selection*) para selecionar os atributos. Esse algoritmo funciona retirando cada atributo dos dados de treinamento e calculando o ganho ou a perda do modelo nessa situação. No final permanecem os atributos que incrementaram a performance do modelo. Dessa forma, o classificador é treinado com esses atributos finais. Na etapa de detecção existe um mecanismo de feedback que possui um limitador de erro, após esse limite ser atingido o classificador é atualizado com os novos exemplos que passaram pelo detector após a fase de treinamento. Essa estratégia tende a corrigir o detector quando a performance começa a deteriorar [51].

Os avanços tecnológicos recentes já permite o projeto de redes ad hoc veiculares (VANET do inglês *Vehicular Ad hoc Network*) que promete fornecer diversos serviços inteligentes de transporte em *smart cities*. Um modelo VANET pode ser visto na Figura

3.6, os veículos comunicam-se entre si, com os controladores SDN e com os RSU (Roadside Units) que facilita a transmissão de informações para outros veículos que passaraão no mesmo ponto. Porém, todo esse avanço vêm acompanhado de diversos perigos o que faz essas VANETs vulneráveis a vários tipos de ataques [46]. Os sistemas de detecção de intrusos (IDS do inglês *Intrusion Detection Systems*) tentam mitigar a possibilidade de ataques, mas eles ainda estão restritos a sub-redes, não abrangendo a VANET inteira. As SDN tentam mitigar esse problema oferecendo um gerenciamento centralizado de toda rede, permitindo que um IDS localizado no controlador possa verificar toda a VANET.

Para mitigar, ainda, o problema de ataque centralizado no controlador SDN é proposto um sistema colaborativo entre vários controladores SDN que mantém a comunicação e o funcionamento do IDS, mesmo que um controlador SDN fique inoperante. A arquitetura das VANETS interconecta a comunicação dos veículos aproveitando-se da utilização das RSUs, tudo sendo gerenciado por um controlador central SDN. Essas redes oferecem serviços como alertas de emergência, segurança das vias, tráfego eficiente, serviços para os motoristas como detecção de congestionamento, assistência para estacionamento e outros [46]. Shu et al. [46] utilizam redes neurais com *deep learning* e GAN (*Generative Adversarial Networks*) explorando SDN distribuídas para desenvolver um sistema de detecção de intrusão colaborativo (CIDS do inglês Collaborative Intrusion Detection System).

A mobilidade dos veículos nessas redes dificulta a localização de veículos maliciosos, porque no momento que um veículo com comportamento anormal é detectado ele pode se afastar do correspondente controlador SDN. Então, essa abordagem colaborativa entre múltiplos SDN permite que o aviso sobre o veículo malicioso seja transmitido para toda a rede VANET, identificando e gerenciando o elemento malicioso [46].

Uma botnet é uma rede composta de computadores que foram comprometidos e são controlados remotamente, executando tarefas comuns, espalhando vírus de computadores ou executando ataques de DDoS. Uma estratégia para se descobrir uma botnet é analisar o fluxo de comunicação entre dois pontos da rede para identificar a comunicação de um bot com o servidor de comando e controle (C&C), essa comunicação possui características estatísticas que podem ser detectadas utilizando AM. Entretanto, os modelos aplicados nesse problema precisam atuar em tempo real e serem confiáveis [36].

Pektaş e Acarman [36] apresentam uma rede neural para identificar botnets que combina camadas convolucionais, redes neurais recorrentes e extrai estatísticas baseadas em fluxo de redes entre dois hosts tais como duração, tamanho dos pacotes e tempo de chegada entre os pacotes. Essa abordagem pode ser aplicada com protocolos de comunicação que são criptografados, pois mesmo sem ter acesso ao conteúdo dos pacotes as informações estatísticas do fluxo de comunicação se mantém [36]. Na extração de atributos os autores utilizaram uma estrutura de grafos para representar o fluxo de comunicação entre dois hosts, assim cada fonte fonte e destino IP são representados por nós, e caso exista comunicação entre eles é acrescentada uma aresta no grafo. A rede neural é utilizada como um classificador binário e é composta de *embedding*, rede convolucional, LSTM e rede completamente conectada. O modelo proposto identifica botnets com acurácia de 99% com tempo de performance compatível com outros modelos, porém a fase de treinamento tem um custo de tempo elevado [36].

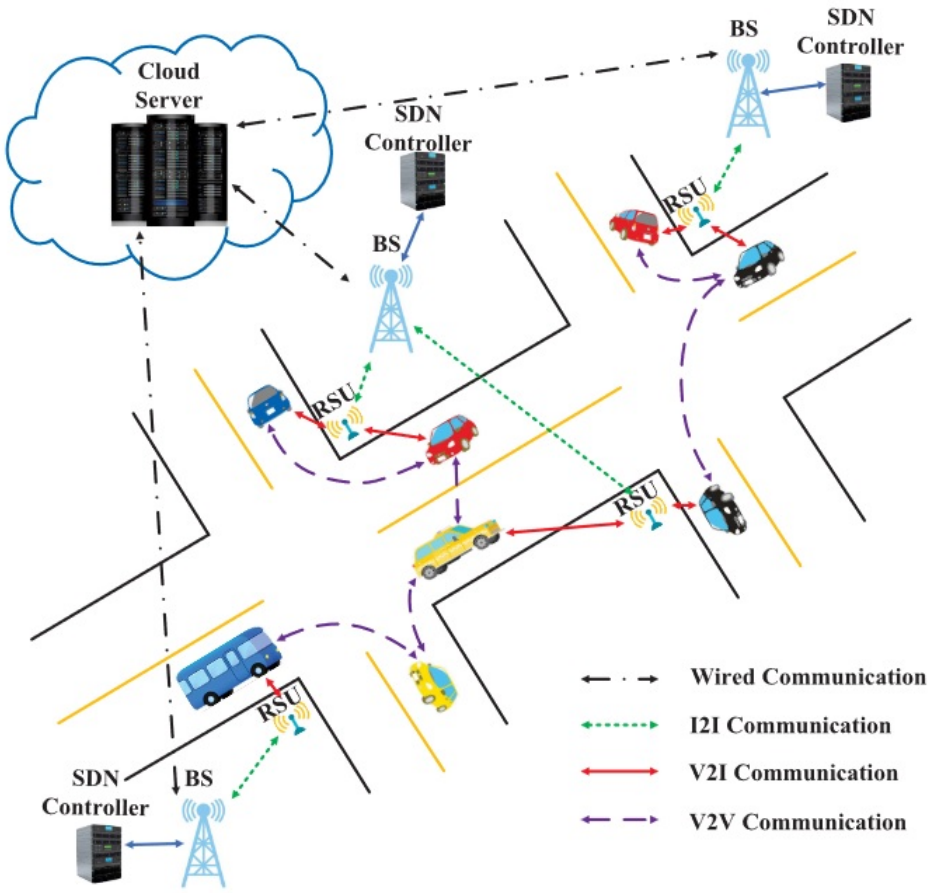


Figura 3.6. Modelo VANET Shu et al. [46]

A classificação de tráfego é utilizada para a detecção de anomalias na rede e exerce um papel importante no domínio de segurança de redes. Entretanto na detecção de malwares existe a dificuldade de se determinar os atributos relevantes para que o classificador seja bem sucedido. Tentando resolver este problema, Wei Wang et al. [54] propôs um classificador de malware que utiliza redes convolucionais para aprender uma representação dos dados do fluxo, que neste trabalho é representado como imagens. Os autores não extraem atributos dos dados, eles transformam o fluxo de tráfego bruto em imagens que alimentam uma rede CNN com diversas camadas. As camadas iniciais aprendem uma representação dos dados, enquanto que as camadas finais classificam as imagens em classes pré-determinadas como visto na Figura 3.7. Essa abordagem permite que o detector alcance acurácia de 99.41% mesmo com dados criptografados [54].

Devido a diferença de continuidade entre o fluxo de tráfego e sua representação em imagens, foram testadas diversas configurações sendo a que apresentou os melhores resultados as que utilizavam a sessão para alimentar as redes. [54]. Os tamanhos dos fluxos e sessões são variáveis de acordo com a transmissão realizada, mas a entrada da rede CNN deve ser uniforme. Então a solução encontrada foi a utilização dos primeiros 784 bytes da sessão. Caso a sessão excedesse esse número o restante da sessão era descartada, caso o tamanho fosse menor os bytes eram completados com 0 no final.

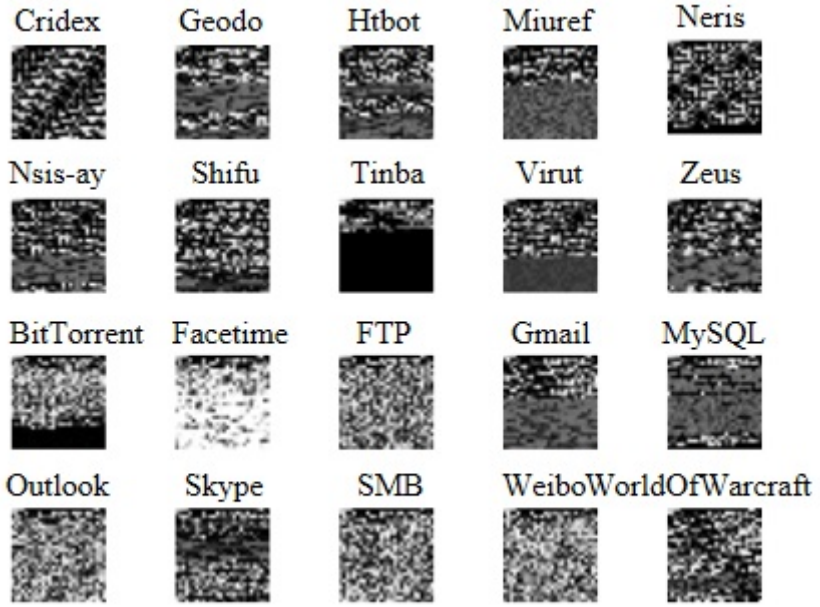


Figura 3.7. Visualização das Classes de Tráfego Wei Wang et al. [54]

As detecções de anomalias vêm se tornando cada vez mais importantes, conforme cresce o tráfego nas redes, principalmente devido à explosão de dispositivos de IoT. As aplicações de AM conseguem obter excelentes resultados nessa detecção, mas esse crescimento vertiginoso impõem dificuldades como velocidade e tempo de processamento devido à quantidade de dados excessiva. Dessa forma, além dos modelos de AM é necessário utilizar tecnologias Big Data desenvolvidas para oferecerem performance e capacidade adequada a tarefas com uma quantidade enorme de dados [38].

Pwint e Shwe [38] investigam a performance do Apache Spark aplicado aos problemas de detecção de anomalias valendo-se de diversos modelos de AM. Além disso, os autores testaram variedade de configuração de paralelismo de processamento e utilização de memória. A conclusão é de que o uso do Spark permite que o processamento dos modelos de AM ocorra em tempo hábil. Além disso, a distribuição e execução paralela das tarefas nos núcleos do Spark se adequam bem aos diversos modelos. Dessa forma, a utilização das tecnologias Big Data no domínio de segurança de redes permite superar as dificuldade de tempo de processamento.

Existem métodos não supervisionados que tentam detectar anomalias nas redes, prevenindo ataques cibernéticos. Entre essas propostas encontra-se o trabalho de Radford et al. [40], que se aproveita dos conceitos e modelagens de NLP para propor um detector de anomalias. Os autores demonstram que rede neural recorrente é capaz de aprender um modelo para representar sequências de comunicação entre computadores e detectar tráfegos anômalos. Espera-se que o fluxo de dados na rede compartilhe características com os modelos de linguagem, seguindo um conjunto de regras similares a uma gramática [40]. O fluxo de rede é tokenizado (separado em sequências menores) em sequência de bytes que foram considerados como "palavras"e que podem ser combinadas originando "sentenças"representativas da comunicação dos computadores. Essas "sentenças"foram

utilizadas para gerar um modelo que aprende semântica e gramática sintática dessa linguagem gerada. Foi utilizada uma rede recorrente com LSTM para capturar essas relações e nuances da linguagem [40]. Esse modelo é utilizado para predizer a comunicação entre dois IPs e o erro de predição é utilizado para identificar transmissões atípicas, caracterizando atividades maliciosas.

A rede implementada possuía duas camadas LSTM, uma camada densa de ativação, uma camada softmax completamente conectada como saída. Cada camada LSTM era composta de 50 células ocultas com ativação linear na primeira camada e ReLU na segunda camada. O treinamento realizado era não supervisionado, ou seja, não possuía exemplos rotulados. Os exemplos rotulados foram utilizados apenas para teste, verificando se as anomalias detectadas referiam-se a ataques de segurança. Para uma melhor avaliação do modelo proposto, realizou-se o treinamento da rede com duas bases distintas, uma com exemplos de ataques maliciosos e uma que não possuía tais ataques.

A ideia era que a base que não possuía ataques entre os exemplos permitisse que a rede detectasse melhor os tráfegos anômalos, já que esses tráfegos teriam características muito diferentes. Enquanto que o fato de possuir diversos exemplos de ataques poderia levar o modelo a "entender"que este tipo de tráfego é normal. Porém, isto não ocorreu. A rede treinada com exemplos maliciosos obteve performance superior ao detectar anomalias, os autores acreditam que isso possa ter acontecido devido ao desbalanceamento das bases de dados disponíveis [40]. Por fim, Radford et al. [40] demonstraram que o fluxo de tráfego pode ser associado a um modelo de linguagem que permite a detecção de tráfego anômalo, caraterizando assim fluxo malicioso. Entretanto, este modelo pode não detectar todos os tipos de ataques, sendo necessário combinar esse monitoramento com outros detectores.

3.3.6. Gerenciamento de Falhas

O crescimento das redes criou arquiteturas complexas e com uma grande quantidade de informações, tornando inviável que eles sejam monitoradas e gerenciadas manualmente. A evolução dos modelos de AM é possível criar ferramentas pra classificação de redes e detecção de falhas de forma autônoma. Os problemas de redes atuais, acabam sendo problemas de big data que são muito difíceis de serem resolvidos analiticamente [30]. A aplicação de algoritmos de AM permite não só resolver os problemas eficientemente, como mantêm o projeto de redes mais simples.

Em redes programáveis é possível criar regras para gerenciamento de falhas a ser entregue em forma de política para um orquestrador. Esse sistema centralizado então monitora a rede e quando detecta um serviço ou dispositivo com problema, ele desvia o tráfego ou aciona o dispositivo backup. Desabilita o aparelho, ou caminho, problemático e emite um aviso de problema informando a ação adotada [30]. Na operação clássica dos sistemas de rede, quando detectada uma falha ia ser emitido um alarme para uma equipe de prontidão que deveria localizar e corrigir a falha.

Zanotelli et al. [55] apresentam um método de predição de falhas com AM aplicados a Rede Ipê que interconecta universidade e centros de pesquisa de todo o país. A abordagem adotada utilizada uma rede LSTM para predizer se numa janela de tempo à frente a rede irá falhar ou não. Dessa forma, em redes programáveis caso a predição de

falha ocorra é possível adotar medidas para manter a disponibilidade da rede, sendo o preditor proposto uma excelente ferramenta de monitoramento.

Os modelos clássicos de interconectividade têm mudado de conexões de dispositivos físicos para ambiente altamente virtualizados em que planos programáveis permitem novas aboradagens para monitoramento e gerenciamento de redes. A aplicação de modelos de AM para predição de falhas é um deles, ao contrário dos modelos clássicos, eles podem ser aplicados em tempo real e tomarem decisões automáticas em tempos inimagináveis para a ação humana [55].

Para a realização dos testes e avaliação do modelo, foram conectados dados de diversos pontos de presença da Rede Ipê utilizando a ferramenta ViaIpê. Essa ferramenta reúne característica de qualidade da rede operada pela RNP, dando transparência para a qualidade dos enlaces da rede acadêmica brasileira. Os atributos retirados dos dados foram: perda de pacotes, RTT (milissegundos), Download e Upload (bits por segundo) [55]. A caracterização de falha utilizada em Zanotelli et al. [55]foi a taxa de perda de pacotes superior a 3%. O modelo tenta predizer se ocorrerá pelo menos uma falha no perído de 15 minutos a frente.

O modelo foi analisado em função das métricas de acurácia, precisão e revocação. Como a Rede Ipê apresenta poucas falhas, o conjunto de dados de treinamento era desbalanceado e o modelo obteve alta acurácia, mas com valores baixos de precisão e revocação. Ou seja, o modelo tinha problemas em detectar os casos positivos de falha. Os testes inciais foram realizados com um modelo global para todos os pontos de presença. Depois da constatação dessa baixa precisão, os autores regionalizaram o modelo, tendo um modelo treinado e utilizado para cada região do país. Com essa redução de heterogeneidade nos dados, os modelos obtiveram bons resultados em algumas regiões continuando insatisfatórios para outras regiões [55].

Os resultados são promissores para a utilização de modelos de AM para caraterização de falhas na Rede Ipê. Entretanto, ainda existe várias possibilidades de estudo para melhorar os resultados. Baseando-se em outros trabalhos já apresentados nesse minicurso, este problema poderia ser abordado com um modelo de aprendizado de representação que eliminasse a etapa de engenharia de atributos, bem como a transformação do fluxo em imagens para classificação entre falha de rede ou não.

Além da detecção de falhas, modelos de AM podem determinar classes para os estados da rede. Em Mohammed et al. [30], os autores determinam três estados que seria o normal, quando não há falhas, estado de falha, quando a comunicação é interrompida e congestionado, quando os parâmetros de QoS deterioraram comprometendo o QoE. Além disso, os autores pretendem identificar o enlace que ocorreu a falha para que ações corretivas possam ser tomadas.

Esse processo autônomo de identificação e localização de falhas pretende evitar custos operacionais e financeiros, automatizando uma importante tarefa para manter a disponibilidade das redes. Para a utilização do sistema foram utilizadas métricas de QoS e QoE como entrada do modelo. As métricas de QoS são. jitter, taxa de perda de pacotes, casos de pacotes fora de sequência, pacotes descartados. Para o QoE foi utilizada a velocidade de transferência de download [30].

Para realizar a classificação foram testados 4 modelos: árvore de decisão, os modelos de agrupamento GB (*Gradient Boosting*) e XGB (*eXtreme Gradiente Boosting*) e uma rede neural completamente conectada. O melhor resultado foi obtido com XGB obtendo acurácia superior a 99%. O conjunto de dados era desbalanceado com as classes de congestionamento e falha tendo menos exemplos do que o estado normal, porém essa é a realidade que o modelo irá se deparar no mundo real. Para tentar minimizar esse problema foi ajustado pesos diferentes para as classes na rede neural, mas os resultados não foram promissores. Ao analisar as métricas de precisão e revocação, constata-se que o modelo proposto com XGB é viável para a detecção e localização de falhas.

Redes de sensores sem fio (WSN - *Wireless Sensor Networks*) estão sujeitas a falhas, principalmente por causa dos ambientes em que são implantadas. Essa falahas, pode ocorrer em nível de software, hardware ou comunicação. Essa redes são compostas por sensores independentes conectados por canais sem fio. Sensores são equipamento projetados para realizar tarefas especificas, por muitas vezes eles são projetados para não serem intrusivos, por isso não dispõem de elevada capacidade de processamento. Exemplos de sensores são termômetros e medidores de pressão [34].

Redes WSN geralmente possuem restrições de energia e armazenamento, podendo ser utilizadas para monitorar saúde, em ambientes de vigilância, em aplicações militares e aplicações industriais. Muitas vezes os ambientes em que elas operam são de difícil acesso e não vão permitir intervenção humana em caso de falhas. Por essa razão, as WSNs necessitam de monitoramento de falhas eficiente e autônomo. Entretanto, o fato desses sensores não possuírem recursos computacionais faz com que o detector precise ser preciso e rápido e possa operam em ambientes com riscos elevados [34].

Então em Noshad et al. [34], os autores avaliam a utilização de 6 classificadores para detecção de falha em redes WSNs. Os experimentos foram realizados com medidores de umidade e termômetros, construindo uma rede artificial para que os dados pudessem ser coletados. As falhas foram introduzidas propositalmente simulando ações do mundo real. Avaliando diferentes classificadores como SVM, CNN, MLP, RF, redes neurais e gradiente descendente estocástico. Esses classificadores são posicionados nos nós centrais dos clusters de sensores, que são responsáveis pelo monitoramento da rede [34]. O melhor resultado encontrado foi com o Random Forest que superou os demais modelos em todas as métricas utilizadas.

Na busca por redes autônomas, que possam ser auto-gerenciadas, diversos modelos foram propostos. Em Huang et al. [17], os autores propõem uma arquitetura para gerenciamento de falhas que utilizam um algoritmo chamado GBRM baseado em redes neurais com auto-encoder. Esse modelo superou outros modelos populares de AM. Redes auto-gerenciadas pretendem funcionar sem intervenção humana, detectando falhas, analisando-as e corrigindo as mesmas. Esse processo é complexo e envolve diversas tomadas de decisão que utilizando AM podem ser realizadas em um intervalo de tempo ínfimo [17]. Esses modelos são orientados a dados combinado com análise estatísticas, demandando também a necessidade por equipamentos especificos que irão realizar medições e monitoramento.

As redes auto-encoders (DAE) possuem duas fases chamadas de encoder e decoder. Elas são similares e funcionam da seguinte forma: na fase de encoder os dados de

entrada passam por camadas que vão reduzindo a sua dimensionalidade, preservando as informações mais relevantes daquele exemplo, o decoder tenta reconstruir os dados originais. Como isso não é possível, o que a rede DAE realiza é minimizar o erro entre o exemplo original e a reconstrução [17]. Essas redes realizam treinamento não supervisionado por não necessitar de rótulos para realizar essa reconstrução. Um dos principais problemas dessas redes é a inicialização dos parâmetros, interferindo diretamente no desempenho da rede. Dessa forma, o algoritmo GBRBM proposto em Huang et al. [17] utiliza uma máquina restrita de Boltzmann(RBM) para pré-treinar a rede otimizando a performance do modelo. Os resultados contribuem para o desenvolvimento da arquitetura auto-gerenciado e alcança acurácia de 89.2%.

Nesta seção foram apresentados diversos modelos para serem aplicados em uma diversidade de problemas. Apesar do avanço realizado na abordagem orientada a dados com a utilização de Aprendizado de Máquina, ainda é difícil selecionar o melhor modelo para resolução de um problema específico. Realmente, não existe um modelo único que resolva todos os problemas [9]. Entretanto, existem iniciativas de agrupamento de modelos criando um super modelo de aprendizado que possa ser generalizado para mais de um tipo de problema [10].

Casas, Vanerio e Fukuda [10] introduzem o GML learning, um modelo genérico para analisar problemas de medições de redes, empregando técnicas de agrupamento seguindo os conceitos de modelo de super aprendizado. O GML learning é um agrupamento de modelos que busca encontrar a melhor combinação entre eles fornecendo uma predição mais precisa. Ele tem performance assintoticamente melhor que qualquer modelo individual do agrupamento [10]. Além disso, como a maioria dos métodos de agrupamento, ele exibe robustez quanto a incerteza apresentada nos dados [10].

Enquanto a aplicação de técnicas de aprendizado em problemas de medições de rede vêm sendo utilizadas intensamente, existem poucas abordagens a respeito dos métodos de agrupamento. Esse fenômeno acontece, mesmo sendo observado na prática que esses métodos atingem melhores resultados do que modelos simples. Os métodos de agrupamento, além de apresentarem melhor performance, combinam diferentes abordagens para resolver o mesmo problema, buscando complementaridade entre os diferentes modelos. Dessa forma, cada modelo potencializa o outro e compensa as limitações dos demais [10]. Entre os métodos tradicionais de agrupamento encontramos o *Bagging* e o *Boosting*.

- **Bagging**. Abreviação de *Bootstrap Aggregation* busca reduzir a variância dos modelos de previsão, gerando subconjuntos de dados de treinamento retirados do conjunto de dados original. Cada modelo individual é treinado com um subconjunto sorteado aleatoriamente, após isso ele é combinado com os demais utilizando um esquema de votação por maioria com mesmo peso para encontrar a predição final [10]. O exemplo mais conhecido dessa abordagem é o algorimto Random Forest.

- **Boosting**. Constrói incrementalmente um agrupamento, treinando cada nova instância de modelo com base no desempenho do modelo anterior. *Boosting* é uma abordagem de duas etapas, onde primeiro usa subconjuntos dos dados originais para produzir vários modelos e, em seguida, aumenta seu desempenho combinando-os,

também usando votação por maioria. A criação de novos conjuntos não é aleatória, depende do desempenho dos modelos anteriores, e cada novo subconjunto contém as instâncias mal classificadas pelos modelos anteriores. Ou seja, o modelo continua aumentando a quantidade de subconjunto enquanto estiver melhorando a sua performance [10]. Um exemplo conhecido é o algoritmo AdaBoost.

O GML learning foi testado com problemas diferentes incluindo detecção de ataques, detecção de anomalias e predição de QoE. Os resultados demonstraram que o GML learning superou os melhores modelos individuais aplicados em cada caso, bem como superou os métodos tradicionais de *bagging* , com a aplicação do Random Forest e *boosting*, com a utilização do AdaBoost [10]. Dessa forma, percebe-se que uma das principais vantagens do GML refere-se a capacidade de lidar com problemas diferentes utilizando o mesmo conjunto de dados e a mesma etapa de treinamento tornando-se uma iniciativa para uma generalização de melhores práticas em medições de redes.

Com base na análise de desempenho de diversas pesquisas e modelos, Casas, Vanerio e Fukuda [10] observaram que tanto as redes neurais quanto os modelos baseados em árvore de decisão fornecem, em geral, melhores resultados em termos de precisão e previsão do que outros modelos únicos. Além da vantagem de sobrecarga computacional muito menor para árvores de decisão em comparação com modelos baseados em redes neurais. Modelos baseados em árvore de decisão representam, portanto, um modelo de aprendizado de máquina muito atraente para análise de redes, não apenas por sua alta precisão e baixo custo computacional, mas também devido a uma série de propriedades incorporadas, como visibilidade do modelo, robustez ao ruído de entrada, entre outras [10].

3.4. Estudo de Caso

Para facilitar a compreensão do assunto, será proposto um estudo de caso com a utilização modelos de Aprendizado de Máquina para resolução de problemas de medições. Dentre os modelos apresentados, utilizaremos métodos interpretáveis, como Árvores de Decisão, e métodos com alto poder preditivo, como Random Forest e AdaBoost, Redes Neurais e Redes Neurais profundas. Além disso, serão utilizados alguns modelos clássicos e mais simples, como o método probabilístico Naive Bayes e o algoritmo baseado em distância KNN, para comparação. Incluiremos medições de tempo de execução dos modelos, para verificar quais poderiam ser utilizados em tempo real, o que sem dúvida é um desafio interessante.

Os resultados serão apresentados com uma discussão detalhada, incluindo suas vantagens e desvantagens, refinamentos adotados, possíveis melhorias e algumas explicações de resultados extremos. Explicaremos as métricas selecionadas para comparação, apresentando a motivação e justificativa para cada uma delas, enriquecendo o entendimento dos experimentos realizados. Durante a apresentação desse estudo de caso, pretende-se demonstrar o processo completo de construção de um modelo de Aprendizado de Máquina, desde a coleta de dados, perpassando as etapas de pré-processamento, construção do conjunto de dados, seleção de atributos, bem como as escolhas dos modelos a serem utilizados. O conjunto de dados a ser utilizado estará e será obtido de dados reais de monitoramento. Uma dessas fontes será a base de dados pública da RNP (Rede

Nacional de Ensino e Pesquisa). Os resultados, explicações, apresentação e implementações realizadas serão disponibilizadas em `https://github.com/loyoladesa/srbc2022`.

3.5. Conclusões

A quantidade de dispositivos móveis conectados às redes cresce exponencialmente impondo novos desafios para os administradores de redes. Esses profissionais buscam garantir os requisitos de qualidade de serviço (QoS) para proporcionar uma melhor qualidade de experiência (QoE) para o usuário final. Além dos dispositivos móveis, o crescimento de dispositivos IoT conectados contribuem para o aumento de tráfego, o crescimento da infraestrutura e da complexidade das redes. Dessa forma, novas maneiras de mensurar e gerenciar as redes foram propostas, aproveitando-se da evolução e capacidade dos modelos de Aprendizado de Máquina. Adotando uma metodologia orientada a dados, as ferramentas e técnicas de metrologia de redes contribuem para um melhor monitoramento e gerenciamento eficiente e preciso das redes.

Existe uma ampla gama de estudos demonstrando as oportunidades que se apresentam na utilização de AM no monitoramento de redes. Este trabalho focou em aplicações de metrologia apresentando modelos capazes de realizar a classificação e predição de tráfego, estimativa de QoE, gerenciamento de falhas e segurança de redes. Observa-se que uma das aplicações mais importantes é a classificação, tendo, consequentemente uma variedade de trabalhos a respeito. Ela é útil para que seja alocado os recursos necessários, de forma que se alcance os requisitos de qualidade de serviço para cada tipo de tráfego. Em alguns casos, pode bloquear serviços em ambientes corporativos (ex: bloquear o uso Netflix em empresas). A utilização de AM permite lidar e classificar tráfego criptografado, sem que seja preciso acessar o conteúdo dos dados, mantendo a privacidade dos usuários e segurança da rede.

A predição exerce um papel central no gerenciamento e alocação de recursos, principalmente em redes definidas por software. A identificação precoce de fluxos muito grandes permite que sejam adotadas medidas para evitar o congestionamento. Medidas essas, que com o auxílio de IA, são disparadas autonomamente sem intervenção humana em uma velocidade, antes inimaginável. Dessa forma, podemos utilizar as técnicas de predição para estimar QoE, aproveitando o poder dos modelos de AM. O mapeamento de QoE a partir das métricas de QoS é amplamente empregado para obter atributos preditivos do fluxo de tráfego, mesmo que ele esteja criptografado.

A segurança de redes é outro campo em que o Aprendizado de Máquina contribui significativamente para a performance das medições, pois grande parte da detecção de anomalias passa por algum sistema inteligente. Esses sistemas analisam os dados e descobrem padrões que permite classificar o tráfego em diferentes classes, detectando comportamentos maliciosos. A vantagem dessas aplicações fica visível no tratamento de ataques DDoS. A maneira tradicional de lidar com esse tipo de ataque é, uma vez que o ataque tenha sido detectado, disparar alarme e bloquear o tráfego ao servidor. Porém, nessa abordagem mesmo evitando um estrago maior o ataque já foi realizado. Com o auxilio dos modelos de AM é possível identificar o comportamento de DDoS, antes que o ataque se manifeste, adotando-se medidas preventivas.

As redes tornaram-se complexas e difíceis de gerenciar, mas utilizando-se ferramentas e técnicas de Metrologia é possível compreender o seu funcionamento, caracterizando-as e retornando mensurações para avaliação de performance. Os modelos de AM permitem atingir outro nível, melhorando o monitoramento e gerenciamento. Algumas propostas foram realizadas para que no futuro, as redes sejam auto-gerenciadas, diminuindo a intervenção humana, sendo capazes de detectar anomalias, analisar e disparar ações necessárias para a correção das falhas.

As técnicas, ferramentas e modelos apresentados abrangeram uma variedade de tipos de redes como redes móveis, redes sem fio, redes IoT, *smart cities*, backbone, redes domésticas e redes definidas por software. Essa ampla gama de contextos indica que a Inteligência Artificial pode ser utilizada para resolver problemas de redes, bem como caracteriza-las e ampliar o nosso entendimentos sobre as mesmas.

Os principais desafios para o futuro residem no gerenciamento de dados provenientes de fontes heterogêneas presentes em redes IoT e em *smart cities*. Além dessas, as redes de comunicação que irão auxiliar veículos autônomos tanto terrestres quanto aéreos precisa de um gerenciador de falhas preciso, autônomo e que funcione em tempo real em velocidade quase que instantânea. Esse é um problema desafiador para futuras pesquisas.

Agradecimentos

Este trabalho é parcialmente financiado pela RNP e CNPq (projeto 300123/2021-3). Agradecemos também à RNP pela sessão dos dados utilizados na parte prática deste minicurso.

Referências

[1] Mahmoud Abbasi, Amin Shahraki e Amir Taherkordi. "Deep Learning for Network Traffic Monitoring and Analysis (NTMA): A Survey". en. Em: *Computer Communications* 170 (mar. de 2021), pp. 19–41. ISSN: 01403664. DOI: 10.1016/j.comcom.2021.01.021. URL: https://linkinghub.elsevier.com/retrieve/pii/S0140366421000426 (acesso em 14/04/2022).

[2] Charu C. Aggarwal. *Neural Networks and Deep Learning: A Textbook*. en. Cham: Springer International Publishing, 2018. DOI: 10.1007/978-3-319-94463-0. URL: http://link.springer.com/10.1007/978-3-319-94463-0 (acesso em 29/04/2022).

[3] Hesham Alhumyani et al. "An Efficient Internet Traffic Classification System Using Deep Learning for IoT". Em: *Computers, Materials & Continua* 71.1 (2022), pp. 407–422. ISSN: 1546-2226. DOI: 10.32604/cmc.2022.020727. URL: https://www.techscience.com/cmc/v71n1/45387 (acesso em 22/04/2022).

[4] Leandro Almeida, Fábio Verdi e Rafael Pasquini. "Estimando métricas de serviço através de In-band Network Telemetry". Em: *Anais do XXXIX SBRC*. Disponível em https://sol.sbc.org.br/index.php/sbrc/article/view/16725 Acessado em 07 de maio de 2022. Uberlândia: SBC, 2021, pp. 252–265.

[5] Razan M. AlZoman e Mohammed J. F. Alenazi. "A Comparative Study of Traffic Classification Techniques for Smart City Networks". Em: *Sensors* 21.14 (8 de jul. de 2021), p. 4677. ISSN: 1424-8220. DOI: 10.3390/s21144677. URL:

https://www.mdpi.com/1424-8220/21/14/4677 (acesso em 22/04/2022).

[6] Davide Andreoletti et al. “Network Traffic Prediction based on Diffusion Convolutional Recurrent Neural Networks”. en. Em: *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. Paris, France: IEEE, abr. de 2019, pp. 246–251. ISBN: 978-1-72811-878-9. DOI: 10.1109/INFCOMW.2019.8845132. URL: https://ieeexplore.ieee.org/document/8845132/ (acesso em 03/05/2022).

[7] Dario Bega et al. “AZTEC: Anticipatory Capacity Allocation for Zero-Touch Network Slicing”. en. Em: *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*. Toronto, ON, Canada: IEEE, jul. de 2020, pp. 794–803. ISBN: 978-1-72816-412-0. DOI: 10.1109/INFOCOM41043.2020.9155299. URL: https://ieeexplore.ieee.org/document/9155299/ (acesso em 03/05/2022).

[8] Abdelkader Benmir et al. “Survey on QoE/QoS Correlation Models for Video Streaming over Vehicular Ad-hoc Networks”. en. Em: *Journal of Computing and Information Technology* 26.4 (mar. de 2019), pp. 267–287. ISSN: 13301136, 18463908. DOI: 10.20532/cit.2018.1004278. URL: http://cit.fer.hr/index.php/CIT/article/view/4278 (acesso em 05/05/2022).

[9] Pedro Casas. “On the Analysis of Network Measurements Through Machine Learning: The Power of the Crowd”. Em: (2018), pp. 1–8. DOI: 10.23919/TMA.2018.8506486.

[10] Pedro Casas, Juan Vanerio e Kensuke Fukuda. “GML learning, a generic machine learning model for network measurements analysis”. en. Em: *2017 13th International Conference on Network and Service Management (CNSM)*. Disponível em http://ieeexplore.ieee.org/document/8255998/ Acessado em 07 de maio de 2022. Tokyo: IEEE, nov. de 2017, pp. 1–9. ISBN: 978-3-901882-98-2. DOI: 10.23919/CNSM.2017.8255998. (Acesso em 22/04/2022).

[11] Aaron Chen, Jeffrey Law e Michal Aibin. “A Survey on Traffic Prediction Techniques Using Artificial Intelligence for Communication Networks”. en. Em: *Telecom* 2.4 (dez. de 2021), pp. 518–535. ISSN: 2673-4001. DOI: 10.3390/telecom2040029. URL: https://www.mdpi.com/2673-4001/2/4/29 (acesso em 03/05/2022).

[12] Zhitang Chen et al. “Seq2Img: A sequence-to-image based approach towards IP traffic classification using convolutional neural networks”. Em: *2017 IEEE International Conference on Big Data (Big Data)*. 2017 IEEE International Conference on Big Data (Big Data). Disponível em http://ieeexplore.ieee.org/document/8258054/ Acessado em 07 de maio de 2022. Boston, MA: IEEE, dez. de 2017, pp. 1271–1276. ISBN: 978-1-5386-2715-0. DOI: 10.1109/BigData.2017.8258054.

[13] Harris Drucker et al. “Support Vector Regression Machines”. Em: *Advances in neural information processing systems* 9 (2003), pp. 155–161.

[14] Erol Gelenbe et al. “Self-Aware Networks That Optimize Security, QoS, and Energy”. en. Em: *Proceedings of the IEEE* 108.7 (jul. de 2020), pp. 1150–1167. ISSN: 0018-9219, 1558-2256. DOI: 10.1109/JPROC.2020.2992559. URL: https://ieeexplore.ieee.org/document/9103525/ (acesso em 06/05/2022).

[15] Utkarsh Goel et al. “Survey of End-to-End Mobile Network Measurement Testbeds, Tools, and Services”. Em: *IEEE Communications Surveys Tutorials* 18.1 (2016), pp. 105–123. DOI: 10.1109/COMST.2015.2485979.

[16] A HajaAlaudeen, E Kirubakaran e D Jeya Mala. “Approaches for Utility-Based QOE/QOS Architecture for Streaming Server in a Heterogeneous Wireless Device Based on SVM”. en. Em: (2015), p. 6.

[17] Huakun Huang et al. “Machine Fault Detection for Intelligent Self-Driving Networks”. en. Em: *IEEE Communications Magazine* 58.1 (jan. de 2020), pp. 40–46. ISSN: 0163-6804, 1558-1896. DOI: 10.1109/MCOM.001.1900283. URL: https://ieeexplore.ieee.org/document/8970164/ (acesso em 24/04/2022).

[18] Oluranti Jonathan, Sanjay Misra e Victor Osamor. “Comparative Analysis of Machine Learning techniques for Network Traffic Classification”. Em: *IOP Conference Series: Earth and Environmental Science* 655.1 (1 de fev. de 2021), p. 012025. ISSN: 1755-1307, 1755-1315. DOI: 10.1088/1755-1315/655/1/012025. URL: https://iopscience.iop.org/article/10.1088/1755-1315/655/1/012025 (acesso em 24/04/2022).

[19] Kemp, Simon. *Digital 2021:Global Overview Report*. Disponível em https://datareportal.com/reports/digital-2021-global-overview-report Acessado em 07 de maio de 2022. 2021.

[20] Yoon Kim. “Convolutional Neural Networks for Sentence Classification”. en. Em: *arXiv:1408.5882 [cs]* (set. de 2014). arXiv: 1408.5882. URL: http://arxiv.org/abs/1408.5882 (acesso em 30/04/2022).

[21] Georgios Kougioumtzidis et al. “Machine Learning for QoE Management in Future Wireless Networks”. Em: *2021 XXXIVth General Assembly and Scientific Symposium of the International Union of Radio Science*. 2021, pp. 1–4. DOI: 10.23919/URSIGASS51995.2021.9560226.

[22] Fatima Laiche et al. “When Machine Learning Algorithms Meet User Engagement Parameters to Predict Video QoE”. en. Em: *Wireless Personal Communications* 116.3 (fev. de 2021). Disponível em https://link.springer.com/10.1007/s11277-020-07818-w Acessado em 07 de maio de 2022., pp. 2723–2741. ISSN: 0929-6212, 1572-834X. DOI: 10.1007/s11277-020-07818-w.

[23] Ming Li et al. “A deep learning method based on an attention mechanism for wireless network traffic prediction”. en. Em: *Ad Hoc Networks* 107 (out. de 2020), p. 102258. ISSN: 15708705. DOI: 10.1016/j.adhoc.2020.102258. URL: https://linkinghub.elsevier.com/retrieve/pii/S1570870519310923 (acesso em 03/05/2022).

[24] Peng Li et al. “An Improved Stacked Auto-Encoder for Network Traffic Flow Classification”. Em: *IEEE Network* 32.6 (nov. de 2018), pp. 22–27. ISSN: 0890-8044, 1558-156X. DOI: 10.1109/MNET.2018.1800078. URL: https://ieeexplore.ieee.org/document/8553650/ (acesso em 16/04/2022).

[25] Iraj Lohrasbinasab et al. “Fromstatistical-to machine learning-based network traffic prediction”. en. Em: *Transactions on Emerging Telecommunications Technologies* (2021), p. 102258. DOI: 10.1002/ett.4394. (Acesso em 03/05/2022).

[26] Sharat Chandra Madanapalli et al. “ReCLive: Real-Time Classification and QoE Inference of Live Video Streaming Services”. en. Em: *2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS)*. Tokyo, Japan: IEEE, jun. de 2021, pp. 1–7. ISBN: 978-1-66541-494-4. DOI: 10.1109/IWQOS52092.2021.9521288. URL: https://ieeexplore.ieee.org/document/9521288/ (acesso em 04/05/2022).

[27] Mateus Marim et al. “Caracterização e Classificação do Tráfego da Darknet com Modelos Baseados em Árvores de Decisão”. Em: *Anais do XXXIX SBRC*. Disponível em https://sol.sbc.org.br/index.php/sbrc/article/view/16716 Acessado em 07 de maio de 2022. Uberlândia: SBC, 2021, pp. 127–140.

[28] Warren S. McCulloch e Walter Pitts. “A logical calculus of the ideas immanent in nervous activity”. Em: *The bulletin of mathematical biophysics* 4 (1943), p. 19.

[29] Tom M Mitchell. *Machine Learning*. McGraw-Hill Science, 1997.

[30] Ayse Rumeysa Mohammed et al. “Machine Learning-Based Network Status Detection and Fault Localization”. en. Em: *IEEE Transactions on Instrumentation and Measurement* 70 (2021), pp. 1–10. ISSN: 0018-9456, 1557-9662. DOI: 10.1109/TIM.2021.3094223. URL: https://ieeexplore.ieee.org/document/9474482/ (acesso em 22/04/2022).

[31] Shady A. Mohammed, Shervin Shirmohammadi e Sa’di Altamimi. “A Multimodal Deep Learning-Based Distributed Network Latency Measurement System”. en. Em: *IEEE Transactions on Instrumentation and Measurement* 69.5 (mai. de 2020), pp. 2487–2494. ISSN: 0018-9456, 1557-9662. DOI: 10.1109/TIM.2020.2967877. URL: https://ieeexplore.ieee.org/document/8963623/ (acesso em 24/04/2022).

[32] K. Mor et al. “Evaluation of QoS Metrics in Ad-Hoc Wireless Sensor Networks using Zigbee”. en. Em: *International Journal of Computer Sciences and Engineering* 6.3 (mar. de 2018), pp. 90–94. ISSN: 23472693. DOI: 10.26438/ijcse/v6i3.9094. URL: http://www.ijcseonline.org/full_paper_view.php?paper_id=1766 (acesso em 05/05/2022).

[33] Lan N. Nguyen e My T. Thai. “Network Resilience Assessment via QoS Degradation Metrics: An Algorithmic Approach”. en. Em: *arXiv:1902.01701 [cs]* (fev. de 2019). arXiv: 1902.01701. URL: http://arxiv.org/abs/1902.01701 (acesso em 05/05/2022).

[34] Zainib Noshad et al. “Fault Detection in Wireless Sensor Networks through the Random Forest Classifier”. en. Em: *Sensors* 19.7 (abr. de 2019), p. 1568. ISSN: 1424-8220. DOI: 10.3390/s19071568. URL: https://www.mdpi.com/1424-8220/19/7/1568 (acesso em 24/04/2022).

[35] Poonam Pandey e Radhika Prabhakar. “An analysis of machine learning techniques (J48 AdaBoost)-for classification”. Em: *2016 1st India International Conference on Information Processing (IICIP)*. 2016, pp. 1–6.

[36] Abdurrahman Pektaş e Tankut Acarman. “Deep learning to detect botnet via network flow summaries”. en. Em: *Neural Computing and Applications* 31.11 (nov. de 2019), pp. 8021–8033. ISSN: 0941-0643, 1433-3058. DOI: 10.1007/s00521-018-3595-x. URL: http://link.springer.com/10.1007/s00521-018-3595-x (acesso em 01/05/2022).

[37] Pascal Poupart et al. “Online flow size prediction for improved network routing”. en. Em: *2016 IEEE 24th International Conference on Network Protocols (ICNP)*. Singapore: IEEE, nov. de 2016, pp. 1–6. ISBN: 978-1-5090-3281-5. DOI: 10.1109/ICNP.2016.7785324. URL: http://ieeexplore.ieee.org/document/7785324/ (acesso em 22/04/2022).

[38] Phyo Htet Pwint e Thanda Shwe. “Network Traffic Anomaly Detection based on Apache Spark”. Em: *2019 International Conference on Advanced Information Technologies (ICAIT)*. 2019 International Conference on Advanced Information Technologies (ICAIT). Yangon, Myanmar: IEEE, nov. de 2019, pp. 222–226. ISBN: 978-1-72815-173-1. DOI: 10.1109/AITC.2019.8920897. URL: https://ieeexplore.ieee.org/document/8920897/ (acesso em 20/04/2022).

[39] J. R. Quinlan. *C4.5: Programs for Machine Learning*. San Mateo, CA, USA: Morgan Kaufmann Publishers Inc., 1993.

[40] Benjamin J. Radford et al. “Network Traffic Anomaly Detection Using Recurrent Neural Networks”. Em: *arXiv:1803.10769 [cs]* (28 de mar. de 2018). Disponível em http://arxiv.org/abs/1803.10769 Acessado em 07 de maio de 2022. arXiv: 1803.10769.

[41] Partha Pratim Ray. “A survey on cognitive packet networks: Taxonomy, state-of-the-art, recurrent neural networks, and QoS metrics”. en. Em: *Journal of King Saud University - Computer and Information Sciences* (jun. de 2021), S1319157821001324. ISSN: 13191578. DOI: 10.1016/j.jksuci.2021.05.017. URL: https://linkinghub.elsevier.com/retrieve/pii/S1319157821001324 (acesso em 05/05/2022).

[42] Antonio Rocha et al. “Revisitando Metrologia de Redes: Do Passado às Novas Tendências”. Em: *Minicursos do SBRC*. 2016.

[43] Carlos Rodrigues et al. “An ontology for managing network services quality”. en. Em: *Expert Systems with Applications* 39.9 (jul. de 2012), pp. 7938–7946. ISSN: 09574174. DOI: 10.1016/j.eswa.2012.01.106. URL: https://linkinghub.elsevier.com/retrieve/pii/S0957417412001248 (acesso em 05/05/2022).

[44] Stuart J. Russel e Peter I.Norvig. *Artificial Intelligence: A Modern Approach*. Pearson, 2020.

[45] Amin Shahraki et al. "Active Learning for Network Traffic Classification: A Technical Study". Em: *IEEE Transactions on Cognitive Communications and Networking* 8.1 (mar. de 2022), pp. 422–439. ISSN: 2332-7731, 2372-2045. DOI: 10.1109/TCCN.2021.3119062. URL: https://ieeexplore.ieee.org/document/9566310/ (acesso em 22/04/2022).

[46] Jiangang Shu et al. "Collaborative Intrusion Detection for VANETs: A Deep Learning-Based Distributed SDN Approach". en. Em: *IEEE Transactions on Intelligent Transportation Systems* 22.7 (jul. de 2021), pp. 4519–4530. ISSN: 1524-9050, 1558-0016. DOI: 10.1109/TITS.2020.3027390. URL: https://ieeexplore.ieee.org/document/9216536/ (acesso em 01/05/2022).

[47] *SNDlib*. Acessado em 05/05/2022. URL: http://sndlib.zib.de/home.action.

[48] Ananda Streit et al. "Efeito do confinamento causado pela pandemia Covid-19 nos perfis de tráfego residencial". Em: *Anais do XXXIX SBRC*. Disponível em https://sol.sbc.org.br/index.php/sbrc/article/view/16724 Acessado em 07 de maio de 2022. Uberlândia: SBC, 2021, pp. 238–251. DOI: 10.5753/sbrc.2021.16724.

[49] Raza Ul Mustafa, David Moura e Christian Esteve Rothenberg. "Machine Learning Approach to Estimate Video QoE of Encrypted DASH Traffic in 5G Networks". Em: *2021 IEEE Statistical Signal Processing Workshop (SSP)*. 2021, pp. 586–589. DOI: 10.1109/SSP49050.2021.9513804.

[50] Hui Wang et al. "Mining Association Rules for Intrusion Detection". en. Em: *2009 Fourth International Conference on Frontier of Computer Science and Technology*. Disponível em http://ieeexplore.ieee.org/document/5392848/. Acessado em 12 de maio de 2022. Shanghai, TBD, China: IEEE, dez. de 2009, pp. 644–648. ISBN: 978-1-4244-5466-2 978-0-7695-3932-4. DOI: 10.1109/FCST.2009.22.

[51] Meng Wang, Yiqin Lu e Jiancheng Qin. "A dynamic MLP-based DDoS attack detection method using feature selection and feedback". en. Em: *Computers & Security* 88 (jan. de 2020). Disponível em https://linkinghub.elsevier.com/retrieve/pii/S0167404819301890 Acessado em 07 de maio de 2022., p. 101645. ISSN: 01674048. DOI: 10.1016/j.cose.2019.101645.

[52] Pan Wang et al. "Datanet: Deep Learning Based Encrypted Network Traffic Classification in SDN Home Gateway". Em: *IEEE Access* 6 (2018). Disponível em https://ieeexplore.ieee.org/document/8473682/ Acessado em 07 de maio de 2022., pp. 55380–55391. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2018.2872430.

[53] Wei Wang et al. “End-to-end encrypted traffic classification with one-dimensional convolution neural networks”. Em: *2017 IEEE International Conference on Intelligence and Security Informatics (ISI)*. 2017 IEEE International Conference on Intelligence and Security Informatics (ISI). Disponível em http://ieeexplore.ieee.org/document/8004872/ Acessado em 07 de maio de 2022. Beijing, China: IEEE, jul. de 2017, pp. 43–48. ISBN: 978-1-5090-6727-5. DOI: 10.1109/ISI.2017.8004872.

[54] Wei Wang et al. “Malware traffic classification using convolutional neural network for representation learning”. en. Em: *2017 International Conference on Information Networking (ICOIN)*. Da Nang, Vietnam: IEEE, 2017, pp. 712–717. ISBN: 978-1-5090-5124-3. DOI: 10.1109/ICOIN.2017.7899588. URL: http://ieeexplore.ieee.org/document/7899588/ (acesso em 30/04/2022).

[55] Vitor Zanotelli et al. “Caracterização e Previsão de Falhas em Serviços de Conectividade: uma Aplicação à Rede Ipê”. Em: *Anais do XXXIX SBRC*. Disponível em https://sol.sbc.org.br/index.php/sbrc/article/view/16717 Acessado em 07 de maio de 2022. Uberlândia: SBC, 2021, pp. 141–154.

[56] Shuyuan Zhao, Yongzheng Zhang e Yafei Sang. “Towards Unknown Traffic Identification via Embeddings and Deep Autoencoders”. Em: *2019 26th International Conference on Telecommunications (ICT)*. 2019 26th International Conference on Telecommunications (ICT). Disponível em https://ieeexplore.ieee.org/document/8798803/ Acessado em 07 de maio de 2022. Hanoi, Vietnam: IEEE, abr. de 2019, pp. 85–89. ISBN: 978-1-72810-273-3. DOI: 10.1109/ICT.2019.8798803.

[57] Artur Ziviani e Otto Carlos M. B. Duarte. “Metrologia na Internet”. Em: *Minicursos do SBRC*. Fortaleza, CE, 2005.

Capítulo

4

Redes Neurais de Grafos no Contexto das Cidades Inteligentes

Cláudio Gustavo Santos Capanema, Fabrício Aguiar Silva, Antonio Alfredo Ferreira Loureiro

Abstract

Graph neural networks, also known as GNNs, have been applied to solve problems in different domains, such as biology, chemistry, physics, natural language processing, computer vision, economics, among others. In particular, this class of neural network has been shown to be effective in modeling problems related to smart cities, such as traffic prediction; urban function classification of elements such as points of interest, roads and regions; forecasting the spread of diseases; autonomous agents; anomaly detection, among other activities. The objective of this chapter is to present the theoretical and practical foundations of graph neural networks in the context of smart cities. For this, the following topics are presented: types of tasks performed, taxonomy of GNNs layers, concepts, commonly used architectures, problems related to smart cities, modeling for the transformation of raw data to the graph structure and GNN structure, in addition to the implementation a model in practice.

Resumo

Redes neurais de grafos, também conhecidas como Graph Neural Networks (GNNs), têm sido aplicadas para resolver problemas em diferentes domínios, como biologia, química, física, processamento de linguagem natural, visão computacional, economia, dentre outros. Em particular, essa classe de rede neural tem-se mostrado eficaz na modelagem de problemas relacionados às cidades inteligentes, como predição de tráfego; classificação de função urbana de elementos como pontos de interesse, estradas e regiões; previsão de disseminação de doenças; agentes autônomos; detecção de anomalia, dentre outras atividades. O objetivo deste capítulo é apresentar os fundamentos teóricos e práticos das redes neurais de grafos no contexto das cidades inteligentes. Para isto, são apresentados os seguintes tópicos: tipos de tarefas realizadas, taxonomia das camadas GNNs, conceitos, arquiteturas comumente utilizadas, problemas relacionados

às cidades inteligentes, modelagens para a transformação de dados brutos para a estrutura de grafo e rede GNN, além da implementação de um modelo na prática.

4.1. Introdução

Grafo é um objeto matemático cuja representação computacional pode modelar elementos individuais, seus relacionamentos e estrutura. O seu potencial de utilização em diferentes contextos tem atraído a atenção de muitos pesquisadores que, por sua vez, têm explorado grafos no contexto de redes neurais para a solução de diversos problemas que envolvem grandes volumes de dados. As redes neurais de grafos (do inglês *Graph Neural Networks*, ou simplesmente GNNs) têm demonstrado maior capacidade do que métodos tradicionais (e.g., algoritmos de aprendizado de máquina) em capturar relacionamentos diretos (e.g., usuário-item) e colaborativos (e.g., estrutura topológica entre elementos que se relacionam) (S. Wu, Sun, et al., 2020). Os principais problemas nos quais as *GNNs* são empregadas podem ser organizados nos seguintes tópicos:

- **Cidades inteligentes**: O estudo sobre cidades inteligentes é crescente e evolui constantemente uma vez que novos elementos e soluções são criados. Apesar da grande variedade de definições, o conceito de cidades inteligentes está comumente relacionado ao uso de tecnologia da informação para a resolução de problemas e desafios enfrentados por governos, empresas, ou indivíduos (Yin et al., 2015). Como resultado, as soluções desenvolvidas têm como objetivo trazer benefícios como eficiência e sustentabilidade. No presente capítulo, os problemas que envolvem cidades inteligentes são categorizados da seguinte forma: sistemas de recomendação, previsão de tráfego, classificação de função urbana, previsão de disseminação de doenças, veículos autônomos, detecção de anomalia, reidentificação de veículos e reconhecimento de atividades.

- **Biologia e química**: No contexto de biologia e química, os principais tópicos são: predição de reação química, predição de interface de proteína e engenharia biomédica (Zhou et al., 2020). Em (Do et al., 2019), moléculas interagem com outras a partir da presença ou não de moléculas reagentes. Assim, são geradas "moléculas produto" a partir da adição ou quebra de conexões. Cada vértice representa um átomo e cada aresta representa o tipo de ligação entre átomos. Por fim, são gerados grafos intermediários para a predição de pares de vértices. Rhee et al. (2017) utilizam uma rede *GNN* com aprendizagem por reforço para classificar subtipos de câncer a partir da interação de proteínas.

- **Física**: Neste contexto, a rede neural deve apreender as leis que governam o comportamento e a interação de partículas ou objetos para simular o seu próximo estado (Zhou et al., 2020). Problemas de rastreamento e reconstrução de partículas podem ser divididos em: rastreamento de partículas carregadas, reconstrução de vértice secundário, mitigação de acumulação, reconstrução de calorímetro e reconstrução de fluxo de partícula (Duarte & Vlimant, 2022). Em (T. Kipf et al., 2018), cada grafo representa uma trajetória de objetos (vértices) que são caracterizados pela sua posição no espaço e velocidade, além de interagirem entre si

(arestas). O modelo prevê, simultaneamente, a trajetória futura dos objetos e os tipos das arestas que os conectam.

- **Processamento de linguagem natural**: Redes neurais de grafos são utilizadas para os seguintes tópicos de processamento de linguagem natural: classificação de texto, rotulagem de sequência, tradução de máquina, extração de relação, verificação de fatos, dentre outros (Zhou et al., 2020). Neste contexto, os vértices frequentemente representam palavras e documentos de texto e as arestas associam palavras dispostas em uma mesma frase ou documentos relacionados (Malekzadeh et al., 2021). Yao et al. (2019) apresentaram um modelo que classifica os tipos de documentos em termos dos seus assuntos, onde cada grafo representa um documento e as suas palavras e arestas estabelecem relações entre palavra-palavra e documento-documento.

- **Visão computacional**: Em visão computacional, os principais tópicos são: classificação de imagem, detecção de objeto, detecção de interação, classificação de região e segmentação semântica (Zhou et al., 2020). Um problema particular de grande relevância é o *MAIC* (*Multi Label Aerial Image Classification*). Por exemplo, dada uma imagem de satélite, é possível identificar diferentes elementos como árvore, água, areia, grama, carro, barco, dentre outros. Lin et al. (2021) combinam redes *CNN* (*Convolutional Neural Network*) e *GNN* para gerar representações de imagens aéreas ao mesmo tempo que gera novas representações semânticas de cada rótulo (objeto do mundo real) com base nas suas inter-relações presentes em um grafo de conhecimento.

- **Outras áreas**: Redes neurais de grafos também podem ser empregadas em outras áreas, como: (1) redes de citação de artigos (T. N. Kipf & Welling, 2016a); (2) mineração de grafo, o que inclui correspondência (Li et al., 2019) e agrupamento (Tsitsulin et al., 2020) de grafo; (3) economia, onde preços de ações são previstos (Cheng et al., 2022).

Em particular, as Redes Neurais de Grafos (*GNNs*) têm demonstrado grande potencial na modelagem de problemas relacionados ao contexto de cidades inteligentes, um domínio específico de sistemas distribuídos e sistemas de computadores. A computação de borda (*edge computing*) tem se desenvolvido com o avanço das redes neurais de grafos, onde dispositivos de borda se tornaram capazes, por exemplo, de lidar com ruídos inerentes aos dados relacionados ao contexto de reconhecimento de atividade humana (Sanchez et al., 2021). No entanto, para o desenvolvimento de soluções que envolvam *GNNs* é necessário compreender quais são as características específicas desse tipo de rede neural. Dessa forma, este capítulo tem como objetivo apresentar os seguintes aspectos:

- Fundamentos teóricos: tipos de entradas, tarefas, treinamentos e camadas *GNN* (e.g. *message passing, pooling, skip connections* e módulos de amostragem).

- Aplicações de *GNN* no contexto das cidades inteligentes (e.g., sistemas de recomendação, previsão de tráfego, detecção de anomalia, dentre outras).

- Abordagens utilizadas para modelar dados brutos de diferentes tipos para a estrutura de grafo e posterior construção de rede neural de grafo, dentro do contexto de cidades inteligentes.
- Implementação prática de um modelo de rede neural de grafo.

Este capítulo segue a seguinte organização: na Seção 4.2 são apresentados os tipos possíveis de entradas para redes neurais de grafos; na Seção 4.3 são apresentados os tipos de tarefas e treinamentos existentes; a Seção 4.4 contém os fundamentos teóricos das principais camadas *GNN*; a Seção 4.5 descreve os principais tipos de problemas em que as redes neurais de grafos são empregadas no contexto das cidades inteligentes; na Seção 4.6 são sumarizadas as principais abordagens de conversão de dados brutos para a estrutura de grafo e posterior modelagem de rede neural; na Seção 4.7 são descritos os desafios e questões abertas de maior relevância; na Seção 4.8, um problema e a sua respectiva implementação são apresentados; Por último, as considerações finais são apresentadas na Seção 4.9.

4.2. Grafo como entrada para *GNN*

Nesta seção, são apresentadas as notações e os conceitos utilizados neste capítulo. A Tabela 4.1 sumariza as principais notações utilizadas ao longo deste capítulo.

Seja um grafo direcionado $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ composto por um conjunto $\mathcal{V}$ de vértices e $\mathcal{E}$ de arestas, onde $v_i \in \mathcal{V}$ denota cada vértice e $(v_i, v_j) \in \mathcal{E}$ denota uma aresta de v_i para v_j. Além disso, assuma que $N = |\mathcal{V}|$ representa o número de vértices e $M = |\mathcal{E}|$ o número de arestas de $\mathcal{G}$, e que $\mathcal{N}(v_i) = \{v_j \in \mathcal{V} | (v_i, v_j) \in \mathcal{E}\}$ representa a vizinhança de v_i. O grafo não direcionado é um caso particular onde para cada aresta (v_i, v_j) existe uma aresta (v_j, v_i) no sentido contrário (Z. Wu et al., 2020). Ambos os tipos de grafos têm suporte nas *GNNs* dependendo das camadas utilizadas. A partir de $\mathcal{G}$ são geradas três matrizes de entrada para modelos *GNN*:

- **Matriz de adjacência** $A \in \mathbb{R}^{N \times N}$, com $A_{ij} \neq 0$ se $(v_i, v_j) \in \mathcal{E}$ e $A_{ij} = 0$, caso contrário.
- **Matriz de atributos de vértices** $X \in \mathbb{R}^{N \times D}$ onde cada vetor $x_i \in \mathbb{R}^D$ contém os valores dos D atributos do vértice v_i.
- **Matriz de atributos de arestas** $E \in \mathbb{R}^{M \times C}$ onde cada vetor $e_{ij} \in \mathbb{R}^C$ contém os valores dos C atributos da aresta $(v_i, v_j) \in \mathcal{E}$. Essa matriz é utilizada em contextos onde os vértices possuem diferentes tipos de relacionamentos (e.g., conhecidos, amigos ou parceiros) (Bianchi et al., 2020).

É importante notar que a maioria das soluções utilizam a própria matriz de adjacência de um grafo ponderado para caracterizar as arestas. Isto é, cada aresta contém um valor que representa o seu peso. Dessa forma, ainda são poucas as abordagens que tiram proveito da matriz de atributos das arestas E.

A utilização das matrizes de adjacência, de atributos de vértices e de atributos de arestas, varia de acordo com cada tipo de camada de *GNN*. Por exemplo, as camadas

de *message passing* necessariamente utilizam como entrada a matriz de adjacência. Além da matriz de adjacência, a maioria dessas camadas utilizam em conjunto apenas a matriz de atributos de vértices, sendo menor o número de métodos que aproveitam da matriz de atributos de arestas. Já as camadas de *pooling*, geralmente têm como entrada as matrizes de atributos, podendo ou não utilizar a matriz de adjacência. Ambos os tipos de camadas serão apresentados e discutidos na Seção 4.4.

Tabela 4.1. Notações utilizadas

Notação	Descrição
$\mathcal{G} = (\mathcal{V}, \mathcal{E})$	Um grafo um conjunto $\mathcal{V}$ de vértices e $\mathcal{E}$ de arestas.
N	O número de vértices, $N = \lvert\mathcal{V}\rvert$
M	O número de arestas, $M = \lvert\mathcal{E}\rvert$.
D	A dimensão do vetor de atributos dos vértices.
C	A dimensão do vetor de atributos das arestas.
$A \in \mathbb{R}^{N \times N}$	matriz de adjacência.
$X \in \mathbb{R}^{N \times D}$	Matriz de atributos dos vértices.
$E \in \mathbb{R}^{M \times C}$	Matriz de atributos das arestas.
$x \in \mathbb{R}^{N}$	Sinal de grafo. Corresponde a uma coluna na matriz de atributos.
$x_i \in \mathbb{R}^{D}$	Vetor de atributos do vértice v_i.
$e_{ij} \in \mathbb{R}^{C}$	Vetor de atributos da aresta (v_i, v_j).

Além disso, os grafos podem ser categorizados de acordo com diferentes aspectos, dentre eles: (1) **direcionado/não direcionado** onde a existência de uma aresta (v_i, v_j) implica em uma aresta (v_i, v_j) no caso do grafo não direcionado e onde a existência de uma aresta (v_i, v_j) não implica em uma aresta (v_i, v_j) no caso do grafo direcionado; (2) **ponderado/não ponderado** onde a existência de uma aresta (v_i, v_j) implica em $A_{ij} = 1$ no caso do grafo não ponderado e a existência da mesma implica em $A_{ij} \neq 0$ no caso do grafo ponderado. (3) **homogêneo/heterogêneo** onde vértices e arestas são de tipos iguais (homogêneo) ou tipos diferentes (heterogêneo); (4) **estático/dinâmico** onde o grafo dinâmico tem os seus atributos e a sua topologia variando ao longo do tempo.

Para cada um dos aspectos acima, existe um conjunto mais adequado de camadas e abordagens no contexto das redes neurais para grafos. Por exemplo, a camada *GCN* (*Graph Convolutional Network*) é frequentemente utilizada em grafos não direcionados, ponderados, homogêneos e estáticos. No entanto, de acordo com a implementação, alterações nesses aspectos podem ser realizadas. Este capítulo, no entanto, tem maior foco nos tipos de grafos mais comumente encontrados na literatura como homogêneos e estáticos.

4.3. Tarefas e tipos de treinamentos em uma *GNN*

Redes neurais para grafos realizam tarefas específicas sobre os elementos que compõem um grafo. Essas tarefas são divididas nas seguintes classes: nível de vértice, nível de aresta e nível de grafo.

- **Nível de vértice**: os vértices podem ser classificados, agrupados e associados a

valores contínuos (ou seja, regressão). No processo de agrupamento, os vértices são divididos em conjuntos disjuntos onde vértices similares pertencem a um mesmo grupo. Em particular, o agrupamento de vértices está frequentemente associado às camadas de *pooling*, como em (Bianchi et al., 2020).

- **Nível de aresta**: nesta categoria, as tarefas possíveis são classificação e predição de arestas (*link prediction*), onde o objetivo da última é prever a existência ou não de arestas entre os vértices. Considerando as representações ocultas de dois vértices, é possível prever a classe/conexão entre eles considerando uma função de similaridade ou uma rede neural (S. Zhang et al., 2019).
- **Nível de grafo**: inclui classificação de grafo, regressão e correspondência. Normalmente, as tarefas de nível de grafo incluem camadas de *pooling* responsáveis por gerar uma representação de alto nível, já que a estrutura original do grafo não precisa ser preservada.

A escolha do tipo de tarefa a ser realizada é importante para a definição da função de perda a ser utilizada. Por exemplo, em uma regressão de grafo a função de perda *mean squared error* pode ser a mais adequada, enquanto que para a classificação de grafo a *categorical cross entropy* pode ser a melhor opção. Em determinadas soluções, é comum que as predições não sejam diretamente calculadas sobre as representações latentes encontradas pela *GNN*. Ao contrário, a predição é calculada a partir de uma combinação (e.g., produto, concatenação, dentre outras operações) das representações latentes da *GNN* com representações encontradas por outros componentes da rede neural, como ocorre em (Capanema et al., 2021b).

Com relação ao aspecto da configuração do treinamento, as redes neurais podem ser treinadas nas seguintes configurações:

- **Supervisionado**, onde rótulos estão disponíveis. As classificações de vértice e grafo são exemplos de tarefas comumente relacionadas.
- **Semi-supervisionado**, onde se treina o modelo considerando uma pequena quantidade de amostras rotuladas e uma grande quantidade de amostras não rotuladas. A solução de (T. N. Kipf & Welling, 2016a) emprega a camada *GCN* no contexto semi-supervisionado.
- **Não supervisionado**, onde os dados utilizados não são rotulados. Esse tipo de treinamento está frequentemente associado com a tarefa de agrupamento de vértices. Além disso, treinamentos não supervisionados têm como variantes os métodos de *auto-encoder* e aprendizagem contrastiva. Em redes *auto-encoder*, tanto as matrizes de adjacência quanto as de atributos podem ser reconstruídas e comparadas com as originais para se computar o erro. Os autores de *VGAE* (*Variational Auto-Encoder*) (T. N. Kipf & Welling, 2016b) treinaram o modelo com a adição de ruído, e aplicam a seguinte equação:

$$\begin{aligned} Z &= \mathrm{GCN}(X, A), \\ \hat{A} &= \phi\left(ZZ^T\right), \end{aligned} \tag{1}$$

onde Z são os *embeddings* atualizados de vértices e $\hat{A}$ é a matriz de adjacência reconstruída. No contexto de aprendizagem contrastiva não supervisionada, You et al. (2020) apresentaram quatro soluções de *data augmentation* para grafos onde são aplicadas as seguintes variações: vértices e arestas são adicionados ou retirados; alguns atributos de vértices são removidos; amostragem de um subgrafo. Após o processo de *encoder*, as representações latentes do grafo criado e do grafo original tendem a ser semelhantes.

4.4. Camadas *GNN*

Nesta seção, as camadas *GNN* são categorizadas de acordo com as tarefas que elas podem desempenhar bem como de acordo com as motivações teóricas por trás de cada classe de camadas.

As camadas *GNN* são divididas em *message passing* e *pooling*, e a utilização delas está intimamente relacionada com o tipo de tarefa desempenhada pela rede neural. Por exemplo, as camadas de *pooling* alteram a topologia original do grafo, fornecendo uma representação de alto nível. Assim, esse tipo de camada está mais associada às tarefas a nível de grafo. Por outro lado, as camadas de *message passing* não alteram a estrutura do grafo em termos dos vértices e das arestas, o que permite que esse tipo de camada seja utilizada, durante alguma etapa, em todos os tipos de tarefas.

Por último, esta seção apresenta uma discussão sobre a complexidade das camadas de *GNN*.

4.4.1. Camadas de *message passing*

As camadas de *message passing* ou de convolução, são responsáveis por computar novas representações de cada vértice considerando a informação local (mensagem) dos seus vizinhos (Grattarola & Alippi, 2021). Assim, esse tipo de camada generaliza o conceito de convolução presente nas *CNNs* (*Convolutional Neural Networks*) para o contexto de grafos.

Em arquiteturas *GNN*, é comum que existam camadas convolucionais consecutivas/empilhadas. Neste sentido, uma dada camada de nível k representa uma operação de convolução para cada vértice central sobre os seus vizinhos de ordem k. Por exemplo, a primeira camada *GCN* (*Graph Convolutional Network*) (T. N. Kipf & Welling, 2016a) aplica a convolução sobre os vizinhos de primeira ordem de cada nó, a segunda camada aplica a convolução sobre os vizinhos de segunda ordem, e assim por diante. Isto significa que, as matrizes de atributos dos vértices são atualizadas a cada passo de convolução. No entanto, algumas abordagens não se constituem de um conjunto de camadas empilhadas, adotando-se mecanismos específicos para agregar as características dos vértices vizinhos em diferentes níveis (e.g., (Defferrard et al., 2016) e (Atwood & Towsley, 2016)).

Considerando os fundamentos teóricos empregados, uma camada de *message passing* pode ser categorizada em abordagem espectral ou espacial. Na primeira, as representações de cada vértice são mapeadas para o domínio espectral através da transformação de *Fourier*. Na segunda, a convolução é realizada considerando a vizinhança

de cada vértice, sem a necessidade de conversão para o domínio espectral. Essas duas abordagens são detalhadas a seguir.

4.4.1.1. Abordagem espectral

Esse tipo de camada utiliza a teoria espectral de grafos. Os filtros de grafos são introduzidos para reduzir ruídos dos sinais de grafos. Em operações espectrais, o sinal de grafo denotado por $x \in \mathbb{R}^N$ é um vetor de atributo de todos os vértices de um grafo (Z. Wu et al., 2020), ou seja, ele representa uma determinada coluna da matriz $X \in \mathbb{R}^{N \times D}$. Esse vetor é convertido para o domínio espectral pela transformação de *Fourier*, seguida da operação de *message passing* e convertendo o resultado de volta pela transformada inversa de *Fourier* $\mathscr{F}^{-1}$ (Zhou et al., 2020). Esse processo, é apresentado pela Equação 2:

$$\begin{aligned} \mathscr{F} &= U^T x, \\ \mathscr{F}^{-1} &= Ux, \end{aligned} \tag{2}$$

onde U é a matriz de auto-vetores do grafo Laplaciano normalizado $L = I_N - D^{\left(-\frac{1}{2}\right)} A D^{\left(-\frac{1}{2}\right)}$, sendo $I \in \mathbb{R}^{N \times N}$ a matriz identidade, D a matriz de graus e A a matriz de adjacência. Uma vez que o grafo Laplaciano normalizado é real, simétrico, positivo semi-definido, ele pode ser fatorado como $L = U \Lambda U^T$, onde Λ é a matriz diagonal dos auto-valores (i.e., $\Lambda_{ii} = \lambda_i$). Dessa forma, a operação de convolução é definida como:

$$g \cdot x = \mathscr{F}^{-1}\left(\mathscr{F}\left(g\right) \odot \mathscr{F}\left(x\right)\right) = U\left(U^T g \odot U^T x\right), \tag{3}$$

onde $\odot$ denota o produto elementar e $U^T g$ representa o filtro no domínio espectral. Ao simplificar o filtro como uma matriz diagonal treinável $g_\theta = diag(U^T g)$, temos

$$g_\theta \cdot x = U g_\theta U^T x, \tag{4}$$

onde o filtro de grafo g_θ remove ruídos do sinal de grafo x. Em geral, as soluções espectrais se diferenciam uma das outras na forma como o filtro de grafo g_θ é definido.

A camada *Spectral CNN* (Bruna et al., 2013) considera que $g_\theta = \text{diag}(w)$, é uma matriz diagonal com parâmetros treináveis onde $w \in \mathbb{R}^N$ (Zhou et al., 2020). Esta abordagem requer a autodecomposição da matriz Laplaciana, o que resulta em três problemas (Z. Wu et al., 2020): (1) qualquer variação em um grafo resulta na mudança de sua base de auto-valores e auto-vetores; (2) os filtros de grafo computados não podem ser aplicados a grafos com estruturas diferentes porque são baseados no domínio onde foram treinados; (3) a autodecomposição tem complexidade computacional de $O(N^3)$. As abordagens *ChebNet* (Defferrard et al., 2016) e *GCN* (T. N. Kipf & Welling, 2016a) reduzem a complexidade para $O(M)$.

A solução *ChebNet* (Defferrard et al., 2016) utiliza os polinômios de Chebyshev da matriz ortogonal de auto-valores para aproximar o filtro do grafo. Esses polinômios são recursivamente definidos por $T_i(x) = 2xT_{i-1}(x) - T_{i-2}(x)$ com $T_0(x) = 1$ e $T_1(x) = x$, onde i indica a ordem do polinômio. Ao invés de calcular a autodecomposição da matriz Laplaciana, o que tem alto custo, o filtro do grafo é definido em função do polinômio de *Chebyshev* até a ordem K, como $g_\theta(\Lambda) = \sum_{i=0}^{K} \theta_i T_i(\tilde{\Lambda})$, onde θ_i é um parâmetro

treinável, $\tilde{\Lambda} = \frac{2}{\lambda_{max}}\Lambda - I_N$ representa os auto-valores reescalados, λ_{max} é o maior auto-valor e cada auto-valor de Λ está compreendido no intervalo $[-1, 1]$. A normalização de Λ é necessária por causa da base ortonormal dos polinômios de *Chebyshev* (Z. Zhang et al., 2020). Substituindo a nova aproximação de g_θ na Equação 4, temos

$$g_\theta \cdot x = U\left(\sum_{i=0}^{K} \theta_i T_i\left(\tilde{\Lambda}\right)\right) U^T x. \tag{5}$$

Uma vez que $T_i(\tilde{L}) = UT_i(\tilde{\Lambda})U^T$, onde $\tilde{L} = \frac{2}{\lambda_{max}}L - I_N$, a Equação 5 pode ser reescrita como:

$$g_\theta \cdot x = \sum_{i=0}^{K} \theta_i T_i\left(\tilde{L}\right) x. \tag{6}$$

Assim, a solução *ChebNet* atualiza os atributos de cada vértice com base nos seus vizinhos de ordem K.

A camada *GCN* (*Graph Convolutional Network*) (T. N. Kipf & Welling, 2016a) simplifica a Equação 6 assumindo que $K = 1$ para evitar sobreajuste, e considera que o maior auto-valor é $\lambda_{max} = 2$, fazendo

$$g_\theta \cdot x = \theta_0 x + \theta_1 x\left(L - I_N\right) x = \theta_0 x - \theta_1 D^{\left(-\frac{1}{2}\right)} A D^{\left(-\frac{1}{2}\right)} x, \tag{7}$$

com dois parâmetros livres θ_0 e θ_1. Ao se considerar $\theta_0 = -\theta_1$, temos

$$g_\theta \cdot x = \theta\left(I_N + D^{\left(-\frac{1}{2}\right)} A D^{\left(-\frac{1}{2}\right)}\right) x. \tag{8}$$

Em seguida, é aplicado um *truque de renormalização* para evitar a perda de gradiente, com $I_N + D^{\left(-\frac{1}{2}\right)} A D^{\left(-\frac{1}{2}\right)} \rightarrow \tilde{D}^{\left(-\frac{1}{2}\right)} \tilde{A}\, \tilde{D}^{\left(-\frac{1}{2}\right)}$. Ao final do processo de simplificação, cada camada *GCN* é definida como:

$$X^{(k+1)} = \sigma\left(\tilde{D}^{\left(-\frac{1}{2}\right)} \tilde{A}\, \tilde{D}^{\left(-\frac{1}{2}\right)} X^{(k)} W^{(k)}\right), \tag{9}$$

onde σ é uma função de ativação, auto-conexões são adicionadas via $\tilde{A} = A + I_N$ e os valores dos atributos são normalizados a nível de vértice através da matriz de graus $\tilde{D}$, onde $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$.

4.4.1.2. Abordagem espacial

As camadas espaciais de *message passing* tendem a ser mais flexíveis e eficientes se comparadas com as espectrais, uma vez que não necessitam de computar auto-vetores ou processar todo o grafo em um mesmo passo. Além disso, dependendo da camada espacial, é possível processar grafos direcionados/não direcionados, diferentes entradas como as matrizes de atributos de arestas, grafos heterogêneos, dentre outros aspectos.

As abordagens espaciais aplicam a operação de convolução diretamente considerando a topologia do grafo. Neste sentido, as operações baseadas no espaço se

assemelham às camadas de convolução das redes CNN. Por exemplo, uma imagem pode ser considerada um caso especial de um grafo onde cada pixel corresponde a um nó, e os seus vértices vizinhos são os *pixels* adjacentes. Na operação de convolução, um filtro 3×3 é aplicado sobre o *pixel*/vértice e os seus vizinhos obtendo a média dos valores calculados. Da mesma forma, as camadas baseadas no espaço realizam a convolução, isto é, atualizam a representação de cada vértice, ao agregar as representações dos vértices vizinhos de cada nó central (Z. Wu et al., 2020). Como maior desafio, as abordagens espaciais buscam realizar a operação de convolução considerando diferentes tamanhos de vizinhança e mantendo a invariância local (Zhou et al., 2020).

A *DCNN* (*Diffusion Convolutional Neural Network*) (Atwood & Towsley, 2016; Z. Wu et al., 2020) assume que as características de um vértice são agregadas com base em uma probabilidade de transição que equilibra a distribuição de informação a cada passo de convolução. O método *DCNN* é definido como:

$$X^{(k+1)} = \sigma\left(W^{(k+1)} \odot P^{(k+1)} X^{(0)}\right), \tag{10}$$

onde σ é uma função de ativação, a matriz $P \in \mathbb{R}^{N \times N}$ é calculada como $P = D^{-1}A$. O resultado final é uma concatenação de $X^{(1)}, X^{(2)}, \ldots, X^{(K+1)}$. Essa camada tem suporte para grafos ponderados e direcionados.

Camadas que utilizam mecanismos de atenção são frequentemente classificadas como um caso particular de métodos de convolução espacial (Zhou et al., 2020). A camada *GAT* (*Graph Attention Network*) (Velickovic et al., 2017; Z. Wu et al., 2020) possui as vantagens de ser paralelizável na computação dos *attention heads*, e pode ser aplicada em contextos onde os vértices que têm diferentes graus ao especificar pesos para os seus vizinhos. A camada *GAT* utiliza o mecanismo de atenção para aprender o peso entre vértices conectados e a saída da k-ésima camada é definida como:

$$x_i^{(k+1)} = \sigma\left(\sum_{j \in \mathcal{N}(i)} \alpha_{ij}^{(k+1)} W x_j^{(k)}\right), \tag{11}$$

onde $x_i^{(0)} = x_i$. O peso do coeficiente de atenção normalizado $\alpha_{ij}^{(s+1)}$ representa o quão conectados estão os vértices v_i e v_j no mecanismo de nível $s+1$, e é definido como:

$$\alpha_{ij}^{(s+1)} = \text{softmax}\left(g\left(a^T\left[Wx_i^{(k)} || Wx_j^{(k)}\right]\right)\right), \tag{12}$$

onde $g(.)$ é a função de ativação *LeakyRelu*, e a é um vetor treinável. Em particular, a Equação 11 tem duas variações ao se utilizar o *multi-head attention* (veja as Equações 13 e 14). Na Equação 13, cada uma das S *heads* são calculadas e concatenadas (Zhou et al., 2020):

$$x_i^{k+1} = \|_{s=0}^{S} \sigma\left(\sum_{j \in \mathcal{N}(i)} \alpha_{ij}^{(s+1)} W^{(s+1)} x_j^{(k)}\right), \tag{13}$$

Por outro lado, a Equação 14 é aplicada quando o mecanismo de atenção é executado na última camada da rede.

$$x_i^{k+1} = \sigma\left(\frac{1}{S}\sum_{s=0}^{S}\sum_{j\in\mathcal{N}(i)}\alpha_{ij}^{(s+1)}W^{(s+1)}x_j^{(k)}\right), \tag{14}$$

onde $\|$ é o operador de concatenação e $\alpha_{ij}^{(s+1)}$ é o coeficiente de atenção normalizado do *attention head* de nível $s+1$.

A maioria das camadas de *message passing* atualizam a matriz de atributos dos vértices X. No entanto, a camada *XENet* (Maguire et al., 2021) atualiza tanto as matrizes de atributos dos vértices X quanto as matrizes de atributos das arestas E.

$$S_{ij} = \phi^{(s)}\left((x_i^k \| x_j^k \| e_{ij}^k \| e_{ji}^k)\right) \tag{15}$$

$$S_i^{(\text{out})} = \sum_{j\in\mathcal{N}(i)} a^{(out)}\left(S_{ij}\right) S_{ij} \tag{16}$$

$$S_i^{(\text{in})} = \sum_{j\in\mathcal{N}(i)} a^{(in)}\left(S_{ij}\right) S_{ij} \tag{17}$$

$$x_i^{k+1} = \phi^{(n)}\left((X_i^k \| S_i^{(\text{out})} \| S_i^{(\text{in})})\right) \tag{18}$$

$$e_{ij}^{k+1} = \phi^{(e)}\left(S_{ij}\right), \tag{19}$$

onde $\phi^{(s)}$, $\phi^{(n)}$ e $\phi^{(e)}$ são *multi-layer perceptrons* com a função de ativação *PRELU*, $a^{(out)}$ e $a^{(in)}$ são camadas *Dense* que utilizam a função de ativação *softmax* e têm apenas um escalar como saída. Na Equação 15, para cada par de vértices são concatenados os respectivos atributos de vértices e de arestas. Nas Equações 16 e 17 as mensagens em direção de saída $S_i^{(out)}$ e de entrada $S_i^{(in)}$ são calculadas para cada vértice v_i. O vetor de atributos de cada vértice x_i^{k+1} do nível $k+1$ é atualizado com base na sua representação de nível anterior x_i^k concatenado com as mensagens de direção de entrada e saída, que carregam dados de vértices e arestas (veja a Equação 18). Os atributos de cada aresta são atualizados de acordo com a Equação 19, onde a nova representação e_{ij}^{k+1} de cada aresta depende das mensagens trocadas entre seus respectivos vértices. De acordo com a implementação desta camada na biblioteca *Spektral* (Bianchi et al., 2020), as mensagens de saída e entrada podem ser calculadas utilizando o mecanismo de *self-attention*. Além disso, na versão para processamento em *batch*, a matriz de adjacência A é utilizada, através de multiplicação, como mascaramento para cada S_{ij}.

4.4.1.3. *Skip connections*

As *skip connections* são operações comumente adicionadas às camadas de *message passing* com o objetivo de permitir que o modelo tenha maior profundidade em termos de camadas *GNN*. Para isso, as *skip connections* tentam preservar a representação histórica do grafo e, assim, reduzir as chances de sobre-suavização (Zhou et al., 2020).

A camada *ARMA* (Bianchi et al., 2021, 2020) (veja a Figura 4.1) é composta de atualizações recursivas da camada *Graph Convolutional Skip (GCS)*, que é definida como:

$$X^{(t+1)} = \sigma\left(\tilde{L}X^{(t)}W + X^{(0)}V\right), \tag{20}$$

onde $X^{(t+1)}$ são atualizações dos atributos dos vértices na iteração $t+1$, σ é uma função de ativação, $X^{(t)}$ é a saída da última camada *GCS*, $X^{(0)}$ é matriz inicial de atributos dos vértices e representa a *skip connection* desta camada, e $W \in \mathbb{R}^{out \times out}$ e $V \in \mathbb{R}^{out \times out}$ são parâmetros treináveis. *out* é o hiperparâmetro do tamanho da saída. $\tilde{L}$ é a matriz Laplaciana modificada $\tilde{L} = D^{(-\frac{1}{2})}AD^{(-\frac{1}{2})}$, onde D é a matriz de graus. Por fim, a saída da camada *ARMA*, $\tilde{X}$, é a média das saídas empilhadas das camadas *GCS*, o que é computado como:

$$\tilde{X} = \sigma\left(\frac{1}{K}\sum_{k=1}^{K} X_k^{(T)}\right), \tag{21}$$

onde K é o número de camadas *GCSs*, $X_k^{(T)}$ é a última saída da k-ésima camada *GCS* e σ é uma função de ativação que pode ser *softmax* ou *sigmoid*, por exemplo, caso essa seja a camada que gera a predição final.

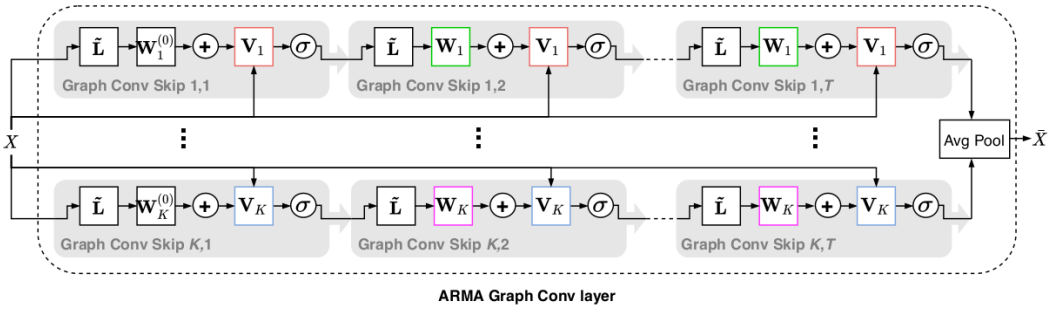


Figura 4.1. Diagrama da camada *ARMA* de (Bianchi et al., 2021)

4.4.1.4. Módulos de amostragem

Os módulos de amostragem (do inglês *sampling modules*) são principalmente empregados em camadas de *message passing* e são úteis para grafos grandes e para reduzir o problema de explosão de vizinhança. Existem três tipos de módulos de amostragem: por vértice, camada e por subgrafo (veja a Figura 4.2).

Na amostragem por vértice, um subconjunto de vértices vizinhos é selecionado em cada passo de propagação de mensagem. Hamilton et al. (2017) selecionam um subconjunto de tamanho fixo de vizinhos de cada vértice. R. Ying et al. (2018) aplicam caminhos aleatórios partindo de cada vértice e selecionam aqueles com o maior número de visitas normalizadas.

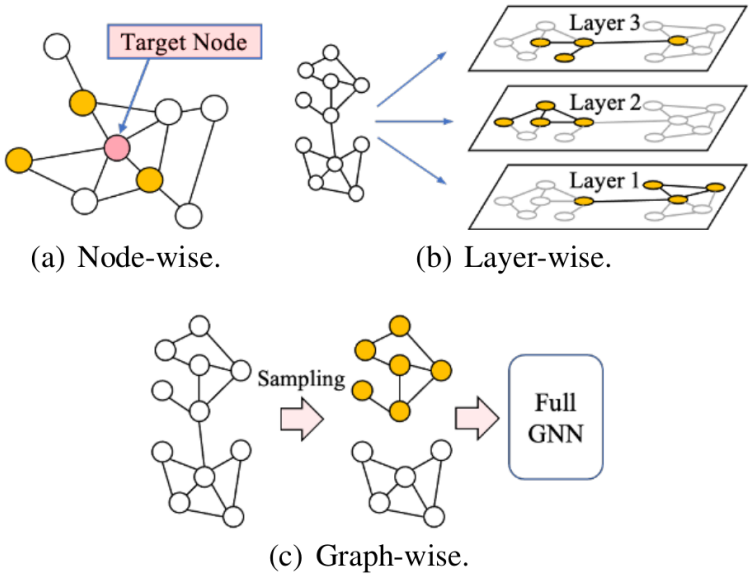


Figura 4.2. Módulos de amostragem de (L. Wu et al., 2022). a) indica a amostragem por vértice b) indica a amostragem por camada c) indica a amostragem por subgrafo

Em métodos de amostragem por camada, ao invés de se selecionar um subconjunto de vizinhos para cada vértice, apenas um subconjunto de vértices de todo o grafo será utilizado para a agregação de informação no processo de propagação de mensagem (Zhou et al., 2020). Esse tipo de método sofre com conexões esparsas entre vértices (Zou et al., 2019). J. Chen et al. (2018) selecionam os vértices com maior grau para construir uma matriz de adjacência menor que será utilizada na propagação de mensagem e representará menor custo computacional. No entanto, de uma camada para outra, o conjunto de vértices selecionados pode não estar suficientemente conectado (e.g., um dos vértices selecionados pode não estar conectado com os outros vértices de maior grau), podendo gerar uma matriz esparsa e até mesmo com valores zero em todas as posições (Zou et al., 2019). Zou et al. (2019) aliviam o problema de conexões esparsas ao selecionar amostras a partir dos vizinhos dos vértices selecionados na camada anterior.

Os métodos de amostragem por subgrafo aplicam a propagação de mensagem apenas dentro dos múltiplos subgrafos selecionados. Chiang et al. (2019) obtêm esses subgrafos são obtidos a partir de algoritmos de agrupamento, onde cada subgrafo representa um *cluster*.

4.4.2. Camadas de *pooling*

Analogamente ao contexto de visão computacional, as camadas de *pooling* são operações de redução de amostragem utilizadas para gerar uma representação simplificada de um dado elemento. Na literatura, elas também são comumente referidas como operações *readout*.

Nas arquiteturas de *GNNs*, as camadas de *pooling* estão, em geral, associadas às tarefas a nível de grafo, de modo que vértices são agregados para gerar uma repre-

sentação simplificada do grafo. Como caso particular, esse tipo de camada também é utilizado em operações de agrupamento de vértices. Essas camadas estão posicionadas após as operações de convolução. Em termos de implementação, esta classe de operações gera como saídas novas matrizes de adjacência e de atributos, onde ambas representam uma quantidade reduzida de vértices. Esta operação de redução de amostragem/vértices tem os seguintes benefícios (Z. Wu et al., 2020):

- Redução do custo computacional das operações, das chances de sobreajuste e de invariância de permutação.
- Melhoria no desempenho de tarefas a nível de grafo.

Como desafio, as camadas *pooling* devem manter a *invariância de ordem* dos vértices (Z. Zhang et al., 2020), i.e., ao se trocar os índices dos vértices e arestas usando uma função bijetora entre um par de vértices, a representação simplificada do grafo após a operação de *pooling* deve ser a mesma.

Existem dois tipos de camadas *pooling* para grafos: *pooling* direto e *pooling* hierárquico (Zhou et al., 2020).

4.4.2.1. *Pooling* direto

Esta categoria de camadas gera representações simplificadas de grafos ao diretamente aplicar estratégias de seleção de vértices (Zhou et al., 2020).

No *pooling* global, todo o grafo é reduzido a um vértice representado por um vetor de atributos. Neste caso, operações de soma, média, máximo e *gated attention* (*GAP*), dentre outras, podem ser utilizadas (Grattarola & Alippi, 2021). As representações calculadas para cada vértice são resumidas/simplificadas pelos operadores de *pooling*.

4.4.2.2. *Pooling* hierárquico

Esta classe de operações explora a hierarquia ou topologia de um grafo para gerar uma nova representação. A camada *DiffPool* (Z. Ying et al., 2018) realiza um agrupamento hierárquico, e é definida por:

$$\begin{aligned} S^k &= \text{softmax}\left(GNN_{k,pool}\left(A^k, X^k\right)\right), \\ A^{k+1} &= \left(S^k\right)^T A^k S^k, \\ X^{k+1} &= \left(S^k\right)^T \text{ConvGNN}_{k,embed}\left(A^{(k)} X^k\right), \end{aligned} \tag{22}$$

onde X^k é a matriz de atributos dos vértices, A^k é a matriz de adjacência do nível k, $S^k \in \mathbb{R}^n$ contém as probabilidades para que cada vértice no nível k, e A^{k+1} e X^{k+1} são as matrizes de adjacência e de atributos de vértices simplificadas, ou seja, com tamanho reduzido no nível $k+1$.

4.4.3. Complexidade

A complexidade de tempo de cada camada pode ser dominada pelo custo do pré-processamento, se existir, ou da operação de *message passing* em si. Camadas espectrais que calculam a decomposição de auto-valores sem que haja nenhuma simplificação no pré-processamento, como *Spectral CNN*, requerem $O(N^3)$. O modelo de (Tran et al., 2018) tem custo de $O(N^3)$ devido ao cálculo de menor caminho entre pares de diversos vértices. Quanto a operação em si de *message passing*, a sua complexidade na maioria das camadas é de $O(N^2)$ quando a matriz de adjacência é densa e $O(M)$ quando a matriz é esparsa (Z. Wu et al., 2020).

4.5. *GNN* para cidades inteligentes

O termo cidades inteligentes é definido pelo uso inteligente da tecnologia para melhorar a qualidade de vida das pessoas, o que é frequentemente associado com a melhoria da saúde de indivíduos, da eficiência de produtos e serviços, da sustentabilidade, do planejamento urbano, da mobilidade urbana, dentre outros aspectos.

As redes neurais para grafos se inserem nesse contexto a partir da modelagem em grafo de elementos presentes no dia-a-dia de ambientes urbanos como ruas/estradas, estabelecimentos, fluxo de veículos em rodovias, movimentação de massas no ambiente urbano, dentre outros. Nesta seção, serão apresentadas as principais áreas dentro de cidades inteligentes onde pesquisadores têm obtido relevantes avanços a partir do uso das *GNNs*. Dentre estes tópicos estão: sistemas de recomendação, previsão de tráfego, classificação de função urbana, previsão de disseminação de doenças, agentes autônomos, reidentificação de indivíduos e reconhecimento de atividades. Para cada um dos tópicos, são apresentadas aplicações reais e como os dados originais são mapeados para o contexto de grafos.

4.5.1. Sistemas de recomendação

Sistemas de recomendação usualmente inferem a preferência de indivíduos por itens. Neste sentido, as interações Usuário-Item podem ser modeladas a partir de um grafo bipartido heterogêneo (i.e., Usuário-item) ou a partir de grafos homogêneos (i.e., Usuário-Usuário e Item-Item) para se encontrar representações latentes que sejam úteis para estimar a preferência de um usuário por um item (S. Wu, Sun, et al., 2020), o que frequentemente se traduz em tarefas a nível de aresta de predição de *link*.

Com relação ao processo de recomendação, existem dois tipos principais: recomendação geral e recomendação sequencial. Na recomendação geral, assume-se que os usuários têm preferências estáticas, isto é, que não mudam ao longo do tempo. A maior parte dos modelos constroem as representações de usuários e itens a partir dos dados históricos das interações entre ambos. Essa relação pode ser descrita por:

$$y_{u,i} = f\left(h_u^*, h_i^*\right), \tag{23}$$

onde $y_{u,i}$ é a preferência do usuário u pelo item i, $f(.)$ é uma função que utiliza as representações h_u^* de usuário e h_i^* de item para gerar a saída $y_{u,i}$.

Por outro lado, as recomendações sequenciais assumem que as preferências de cada usuário são dinâmicas e evoluem ao longo do tempo. Dada uma sequência de n

interações Usuário-Item, o objetivo é prever qual é o item mais provável na interação $n+1$, como descrito por:

$$i^*_{s,n+1} = \text{argmax}_{i \in I} \text{P}\left(i_{s,n+1} = i | S^u\right), \tag{24}$$

onde $S^u = [i_{s,1}, i_{s,2}, \ldots, i_{s,n}]$ é a sequência de itens $i_{s,t}$ que o usuário u interagiu em cada tempo $t \leq n$.

Com relação às recomendações sequenciais, S. Wu, Zhang, et al. (2020) e Capanema et al. (2021b) combinam redes *RNN* e *GNN* para estimar preferências de usuários por *PoIs* (*Points of Interest*) e categorias de *PoIs*, respectivamente. S. Wu, Zhang, et al. (2020) utilizam um bloco com duas camadas consecutivas *GCN* para modelar as características geográficas de cada *PoI*, onde cada vértice indica a influência geográfica entre cada par de *PoIs* (modelagem Item-Item), de modo que quanto maior a proximidade maior a influência. Na matriz de atributos dos vértices, são utilizadas as representações latentes encontradas de cada *PoI* através da camada de *Embedding* da componente recorrente. Na camada recorrente, são processadas sequências de locais visitados por um usuário. As componentes de *RNN* e *GNN* são combinadas a partir do produto entre as representações latentes de cada componente, o que se traduz na preferência de um usuário por cada *PoI*.

O modelo apresentado em (Capanema et al., 2021b) prevê a categoria do próximo local a ser visitado, o que é importante para que provedores de serviços móveis realizem ações de *marketing* mais assertivas. Neste trabalho, cada vértice do grafo corresponde a uma categoria de *PoI* (modelagem Item-Item) e diferentes matrizes de atributos de vértices (e.g., matrizes de distância e duração de tempo entre visitas aos estabelecimentos de cada categoria) são utilizadas em blocos separados de convolução. É importante ressaltar que, um dos *datasets* utilizados contém *check-ins* de usuários de uma rede social, de modo que cada estabelecimento visitado é considerado como um *PoI*. Por outro lado, a segunda base de dados utilizada contém trajetórias de *GPS* de usuários móveis e, dessa forma, foi necessário utilizar o algoritmo de identificação de *PoIs* descrito em (Capanema et al., 2021a, 2019; Capanema & Silva, 2021). A componente de camada *RNN* indica o comportamento recente do usuário, enquanto a componente *GNN* indica o comportamento global/geral do usuário em termos da sua mobilidade entre *PoIs* e as suas respectivas categorias. A predição da categoria do próximo local se dá pela combinação dos resultados de cada componente, o que caracteriza a solução como uma abordagem híbrida.

No contexto de abordagem não sequencial, Xiao et al. (2020) utilizam os grafos Usuário-Usuário e Usuário-Item para prever relacionamentos de amizade entre indivíduos (*link prediction*) e a probabilidade de aquisição de produtos/itens no contexto de redes sociais. Para isso, os autores consideram que usuários que são amigos em uma rede social têm propensão a consumir os mesmos produtos, ao mesmo tempo que indivíduos que têm hábitos de consumo similares tendem a se tornarem amigos. Neste sentido, vértices são associados aos usuários e produtos, e representações latentes são calculadas para cada um, tendo um papel similar à matriz de atributo de vértices.

Liang et al. (2021) apresentam uma rede neural para a recomendação de *apps* do *Google Play* considerando o contexto de grafos heterogêneos onde cada vértice

pode representar um usuário, um aplicativo, ou uma categoria de aplicativo. Além disso, as arestas são divididas em dois tipos: avaliação de usuários a aplicativos e associação entre aplicativo e a sua categoria. Um exemplo de meta caminho é: considerando a preferência de um usuário por um aplicativo, percorre-se o grafo considerando apenas aplicativos da mesma categoria. Neste meta caminho, será alcançado um conjunto específico de usuários que têm preferência pela mesma categoria de aplicativo. O processo de *message passing* é realizado através de meta caminhos onde se agrega os atributos de vértices de arestas de diferentes tipos.

4.5.2. Previsão de tráfego

A previsão de tráfego é um problema importante para o desenvolvimento de sistemas de transportes inteligentes. Neste sentido, as redes neurais de grafos têm contribuído para o desenvolvimento de soluções no estado da arte. Essas soluções partem do princípio de que o estado de tráfego (e.g., fluxo de tráfego e velocidade de tráfego em um dado local) influenciará, em certo grau, o estado de tráfego futuro de outros locais. Para isso, as soluções processam, em geral, sequências históricas de grafos que representam a variação do estado de tráfego ao longo do tempo.

Os sistemas de previsão de tráfego são inicialmente divididos de acordo com o tema (Jiang & Luo, 2021): fluxo de tráfego, velocidade, demanda por recursos, dentre outros. Cada um desses problemas podem ser analisados em diferentes níveis: estrada, estação e região. Além disso, cada nível exige uma abstração diferente dos dados. A nível de estrada, cada vértice pode representar uma interseção de ruas e cada aresta pode representar uma rua. Em outro cenário, cada vértice pode representar um sensor em um segmento de estrada ao mesmo tempo em que o peso de cada aresta indica a distância entre cada par de sensores. A nível de estação, cada estação de ônibus ou metrô é representada por um vértice e as linhas/trajetórias realizadas são correlacionadas com as arestas. A nível de região, cada vértice é associado a uma região e cada aresta indica o fluxo de elementos entre duas regiões.

Com relação à geração do grafo a partir de dados de fluxo de tráfego, sejam eles obtidos através de sensores ou dados de GPS de usuários móveis, existem diversos tipos de matrizes de adjacência. Os principais tipos são: (1) matrizes baseadas em conectividade de transporte, onde dois locais v_i e v_j podem estar associados, isto é $A_{ij} = 1$, se uma viagem pode ser realizada de modo conveniente ou dentro de um tempo máximo, utilizando os meios de transporte disponíveis; (2) matrizes baseadas em distância, onde a aresta entre dois vértices v_i e v_j é ponderada pela distância (e.g., distância geográfica) entre os elementos representados pelos vértices; (3) matrizes de similaridade, onde a posição A_{ij} indica o nível de similaridade entre dois locais considerando o histórico de seus estados de tráfego.

4.5.2.1. Fluxo de tráfego

O fluxo de tráfego é definido como o número de veículos que passam por um local em um dado intervalo de tempo. A predição eficaz do fluxo de tráfego é importante para o controle de tráfego ou o controle de semáforos, por exemplo. Nesse último, o tempo

do semáforo pode ser otimizado de modo que os veículos permaneçam parados por um período menor. Em problemas a nível de estrada, são previstos os seguintes tipos de fluxos: estrada, origem-destino (OD) e vazão de tráfego em uma interseção. Em problemas a nível de região, cada área pode ser dividida regularmente (i.e., grids de tamanhos iguais) ou irregularmente (e.g., com base no CEP ou limites de um bairro). Por outro lado, nos problemas a nível de estação se prevê o fluxo em uma estação de metrô ou de ônibus.

Um aspecto desafiador é a coleta de dados de tráfego. A utilização de sensores em rodovias, por exemplo, tem um alto custo de implantação e manutenção. Como alternativa, sensores de GPS em dispositivos móveis podem fornecer dados de movimentação de indivíduos a um baixo custo. Ao mesmo tempo, essa abordagem apresenta desafios com relação à qualidade dos dados, por exemplo, dados faltantes.

Dai et al. (2020) utilizam dados de navegação de veículos para caracterizar o fluxo de tráfego em segmentos de estradas (vértices) e, a partir disso, estimar o tempo de viagem em uma região. Cada grafo representa o tráfego de uma região, e é construído para cada intervalo de tempo de 30 minutos. As arestas da matriz de adjacência conectam estradas do grafo não direcionado, e o peso é obtido por uma operação composta que considera a proximidade entre as estradas (i.e., menor caminho) e a correlação do tempo de viagem entre elas. Esse trabalho utiliza a camada *STGCN* (*Spatio-Temporal Graph Convolutional Network*) (Yu et al., 2017) para realizar o treinamento sobre sequências de grafos que representam os estados de tráfego anteriores. Essa solução se enquadra na tarefa de regressão de grafo.

No contexto de transporte ferroviário, vértices podem estar associados a estações de metrô, como em (Ye et al., 2020). Nesse trabalho, as arestas do grafo direcionado são ponderadas pelo menor tempo de viagem entre cada par de estações (Origem-Destino). Em particular, esses valores de tempo são obtidos a partir do algoritmo de *Dijkstra*. A partir disto, são gerados diferentes tipos de matrizes de adjacência, onde duas estações estão conectadas apenas se a aresta tem um peso dentro de um intervalo específico de valores. A matriz de atributos dos vértices contém os fluxos de saída e chegada de passageiros em cada estação. No modelo proposto, existem duas componentes de redes neurais. Na primeira, camadas *RNN* processam sequências de matrizes de atributos para diferentes intervalos de tempos. Na segunda, camadas *GNN* processam sequências de grafos. A solução se enquadra na tarefa de regressão de vértice, onde se pretende prever o fluxo de passageiros em cada estação.

4.5.2.2. Velocidade de tráfego

A velocidade de tráfego é definida como a velocidade média de veículos que passam por um local. A predição da velocidade de tráfego é útil para estimativas de melhores rotas, tempo de chegada no destino e duração de viagem. Yu et al. (2017) apresentam um modelo para prever a velocidade do tráfego em cada estação de sensor (vértice) ao longo da estrada, configurando-se como uma tarefa de regressão de vértice. Cada aresta, indica a conectividade entre esses sensores e é ponderada pela proximidade geográfica entre eles.

4.5.2.3. Demanda de tráfego

Esta classe de problemas se refere à previsão da demanda de sistemas de transporte considerando o potencial de viagens a serem realizadas. Isto pode se traduzir em demandas por táxis, veículos compartilhados, bicicletas, dentre outros. Em (Ke et al., 2021), cada vértice representa um par de regiões Origem-Destino (OD) e cada aresta indica a conexão entre ODs. A partir do conceito de grafo OD, são construídos quatro grafos onde as matrizes de adjacência têm os pesos das arestas ponderados pelos seguintes aspectos: vizinhança de regiões, similaridade de função urbana de regiões, distância entre regiões, correlação de padrões de mobilidade entre regiões. Os vértices são caracterizados pelas demandas de viagem para cada par de regiões OD considerando diferentes períodos. Assim, é possível estimar a demanda futura por viagens de táxi em cada par de regiões OD.

4.5.2.4. Outros problemas

Outros tópicos estão relacionados com as seguintes predições (Jiang & Luo, 2021): disponibilidade de vagas de estacionamento, atraso de linhas de trem, emissão de poluentes por veículos, acidentes, dentre outras.

4.5.3. Classificação de função urbana

Nesta categoria de problemas, as redes neurais de grafos são utilizadas para prever a função ou o tipo de cada elemento urbano. Essa tarefa é importante para o planejamento urbano e governança, uma vez que com o passar do tempo, os elementos urbanos podem ter a sua semântica alterada. A classificação de função urbana pode ser realizada a nível de local específico, rua/estrada, área e região. Como esse tópico não está relacionado com a previsão de fluxos de tráfego, a utilização de grafos direcionados não é frequentemente necessária.

(S. Hu et al., 2021) classificam segmentos de estradas em: (1) tráfego, que são caracterizadas pela alta velocidade, faixas mais largas e pelo baixo volume de pedestres; (2) comercial, onde há predominância de shoppings e comércio em geral; (3) pública, onde a velocidade de tráfego tende a ser menor, as faixas são levemente mais estreitas e as ruas são cercadas por estabelecimentos que oferecem serviços gerais, como estacionamentos e edifícios públicos. Além disso, cada vértice é associado a um segmento de estrada, e as arestas indicam segmentos de estradas adjacentes. Considerando essa modelagem, o trabalho realiza a tarefa de classificação de vértices. São utilizadas três fontes de dados: rede de estradas urbanas, dados de GPS de trajetórias de táxis e dados de *PoIs* que contêm as categorias de estabelecimentos. A partir desses dados, são criadas matrizes de atributos que caracterizam cada rua com base nos *PoIs* ao redor e nos padrões das viagens de táxis que passam por elas.

Yang et al. (2022) classificam regiões em industrial, comercial, residencial e outro. Cada vértice representa um edifício de uma dada região e cada aresta conecta vértices de edifícios vizinhos sendo, portanto, um grafo não direcionado. Como a conexão entre os edifícios é construída utilizando a Triangulação *Delaunay*, as arestas

são ponderadas de acordo com o tamanho da aresta que liga um centroide a outro na triangulação. Cada vértice é caracterizado pelos seguintes atributos geométricos referentes ao diagrama de *Delaunay* obtido: raio, área, perímetro, média de orientação da aresta, compacidade, alongamento e concavidade. Este trabalho se enquadra na categoria de tarefas de classificação de vértice.

4.5.4. Previsão de disseminação de doenças

Como uma tendência crescente, grafos têm sido utilizados para representar a movimentação de massas entre diferentes regiões geográficas para modelar a disseminação de doenças. A previsão de disseminação de doenças pode ser a nível de indivíduo (i.e., de pessoa para pessoa) até a nível de região (i.e., de região para região). Essa última, é um caso particular da predição de fluxos de massas. Além disso, nesse tipo de modelagem, a matriz de atributos dos vértices normalmente carrega o estado de saúde de cada indivíduo ou de grupos de pessoas (e.g., pessoas que vivem em uma dada região).

Tomy et al. (2022) utilizam dados de redes sociais para estimar a disseminação de Covid-19. Cada grafo não direcionado representa uma rede de usuários, onde cada indivíduo é associado a um vértice e o contato entre duas pessoas é representado por uma aresta. A matriz de atributos dos vértices indica estado de saúde de cada indivíduo. Ao final, a solução proposta prevê a condição de saúde de cada pessoa, podendo ser: recuperado, saudável ou contaminado. Dessa forma, essa solução se enquadra na tarefa de classificação de vértice.

La Gatta et al. (2020) apresentam um modelo que processa sequências de grafos direcionados a partir da combinação de redes *GNN* e *RNN*, onde cada $G^{(t)}$ com $t \in [1, T]$ representa um *snapshot* de grafo no instante t, onde T é o tamanho da sequência. Cada vértice representa uma região, e é caracterizado por atributos estáticos (e.g., população e densidade demográfica) e dinâmicos (e.g., fração de indivíduos não viajantes, fluxo de entrada de indivíduos e raio de giro), onde os valores variam ao longo do tempo. Além disso, as arestas são ponderadas com base nos fluxos de usuários móveis entre as regiões analisadas. Essas informações são obtidas através de dados governamentais e de operadores de serviços móveis. Por fim, para cada região/vértice são previstos os seus estados futuros, como números de novos casos e de indivíduos recuperados, o que se caracteriza como uma tarefa de regressão de vértice.

4.5.5. Agentes autônomos

A modelagem de redes *CAV* (*Connected Autonomous Vehicle*) em grafos e a subsequente utilização de *GNNs* é importante para o auxílio de tomadas de decisões de veículos autônomos.

S. Chen et al. (2021) modelam *CAVs* para assegurar a comunicação e cooperação entre veículos na tarefa de mudança de faixa, isto é, mover-se para a esquerda, permanecer na faixa ou mover-se para a direita (veja a Figura 4.3). No processo de modelagem da matriz de adjacência do grafo não direcionado, cada veículo é associado a um vértice e seus vizinhos estão conectados por arestas. A matriz de atributos dos vértices contém quatro características: velocidade, posição, localização e intenção de movimento. Posição e intenção são variáveis categóricas, e portanto são representa-

das por *one hot encoding*. Por fim, cada veículo/vértice é classificado de acordo com as ações de mudar para a faixa da esquerda, direita, ou manter a posição.

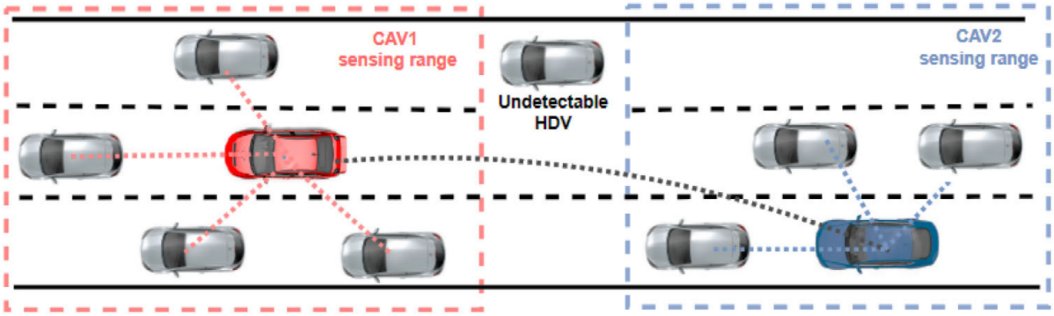


Figura 4.3. Exemplo de rede *CAV* de (S. Chen et al., 2021)

4.5.6. Detecção de anomalia

A modelagem em grafos de sistemas inteligentes de transporte, energia e de empresas (e.g., indústrias e prestadoras de serviços) tem se mostrado promissora para o desenvolvimento de soluções que empregam *GNNs* (Y. Wu et al., 2021). Neste sentido, a Internet das Coisas (*IoT*) e o advento das redes 5G tem um papel fundamental na coleta dos dados utilizados na modelagem de grafos para a detecção de anomalias.

No contexto de sistemas inteligentes de transporte, anomalias no trânsito estão frequentemente associadas a acidentes, eventos públicos, crimes, dentre outros. (Y. Hu et al., 2020) apresentam uma arquitetura *auto-encoder* de grafo, onde cada vértice corresponde a uma região e as arestas correspondem às viagens realizadas entre cada região. A matriz de adjacência é ponderada pelo quão rápidas são as viagens entre cada par de regiões. Cada vértice é caracterizado pelos valores das latitudes e longitudes mínimas e máximas da região correspondente. O erro de reconstrução do *auto-encoder* é utilizado para indicar o grau de anomalia de cada grafo. Nesse processo é realizada a predição de peso de arestas, o que enquadra a solução no contexto da tarefa de predição de *link*.

Deng et al. (2022) utilizam uma arquitetura *GAN* (*Generative Adversarial Network*) para sequências de grafos direcionados, onde o gerador aprende a criar grafos falsos e os adiciona na última posição de cada sequência, e o discriminador aprende a distingui-las. Neste trabalho, cada vértice representa uma região ou faixa de trânsito, de acordo com o contexto avaliado. Cada aresta conecta vértices de localidades adjacentes. O peso de cada aresta é determinado pela distância entre as regiões, de modo que regiões mais próximas têm maior valor associado. Com relação à matriz de atributos de vértices, cada atributo é caracterizado de acordo com os fluxos de entrada e saída de viagem em cada região ou o volume de tráfico e a velocidade média em cada faixa, dependendo do *dataset* avaliado. Esse estudo se encaixa na tarefa de regressão de vértice, de modo que o cálculo da função de erro envolve a matriz X e a gerada X'.

Os sistemas de energia inteligentes estão relacionados pelo advento de fontes renováveis (e.g., eólica e fotovoltaica), bem como com as suas redes de geração distribuída, transmissão e consumo. Anomalias que causem a interrupção ou a sobrecarga

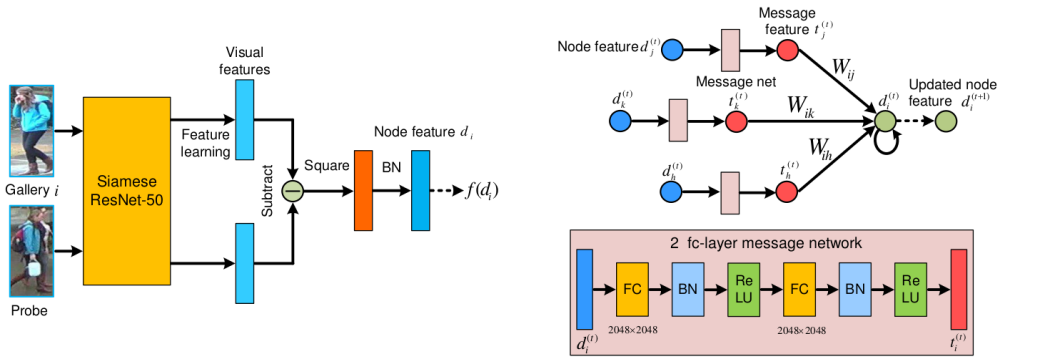
desses sistemas podem afetar sistemas de produção, em especial de empresas altamente tecnológicas como na Indústria 4.0. Neste sentido, a detecção e predição de anomalias é importante para que organizações mantenham a segurança e a previsibilidade. (Owerko et al., 2018) desenvolveram um modelo baseado em redes neurais de grafos para prever a interrupção de transmissão elétrica tendo como base as condições meteorológicas. Em particular, cada vértice corresponde a uma estação meteorológica e cada aresta indica a correlação entre elas, onde estações próximas têm um alto valor associado na matriz de adjacência. Com relação à matriz de atributos dos vértices, cada coluna representa uma medição meteorológica, como pressão, temperatura, velocidade do vento, umidade, nível de precipitação dentre outras.

No contexto de empresas inteligentes, anomalias podem ser detectadas ou previstas na linha de produção, nos produtos em si, ou nos serviços prestados (Y. Wu et al., 2021). Com isso, atividades de reparos podem ser antecipadas ou mais bem planejadas e prejuízos são eventualmente reduzidos. (D. Chen et al., 2021) detectam falhas em processos industriais a partir da seguinte modelagem: os vértices representam sensores e as arestas representam a relação entres os sensores. Cada posição da matriz de adjacência tem um peso treinável. A matriz de atributos dos vértices contém as características coletadas por cada sensor. A matriz de atributos das arestas indica o tipo de cada aresta. Na matriz de adjacência, o peso de cada aresta (i.e., relação entre cada sensor) não é obtido diretamente, como ocorre na maioria dos métodos. Ao contrário, esses pesos são calculados utilizando o mecanismo de atenção sobre os vetores de atributos de cada vértice, indicando a importância que cada sensor tem para o outro em ocorrências de falhas no sistema.

4.5.7. Reidentificação de indivíduos

Em sistemas de vigilância por vídeo, a identificação de um mesmo indivíduo em imagens de diferentes câmeras tem sido tratada utilizando-se *GNNs* juntamente com redes *CNNs* para extrair as similaridades entre imagens (Shen et al., 2018). A partir da identificação automática de indivíduos que representem perigo potencial a uma população, eleva-se a sensação de segurança e de bem-estar de uma sociedade. A utilização de recursos tecnológicos para este fim é um importante aspecto para as cidades inteligentes.

Em (Shen et al., 2018), cada vértice representa um par de uma imagem de referência (e.g., imagem do indivíduo alvo) e uma imagem de galeria (i.e., amostra de imagem de câmera), onde cada aresta é ponderada pela similaridade entre cada par de amostras de galeria. Cada vértice possui uma representação latente gerada por uma componente baseada em camadas *CNN* que codificam a similaridade entre a imagem de referência e a imagem da galeria (veja a Figura 4.4). O processo de *message passing* é guiado pela similaridade (i.e., peso das arestas) de imagens de galeria, de modo que imagens similares têm as suas representações agregadas com maior peso, resultando em um melhor desempenho na tarefa de reidentificação de indivíduo, onde cada vértice é classificado como pertencente ou não à imagem de um indivíduo de referência.



(a) Node input feature generating. (b) Deep message passing of SGGNN.

Figura 4.4. Modelo de reidentificação de indivíduos de (Shen et al., 2018)

4.5.8. Reconhecimento de atividade humana

O reconhecimento de atividade humana, também conhecido como *HAR* (*Human Activity Recognition*), tem aplicações na saúde humana, esportes, investigação criminal dentre outros (Mondal et al., 2020). Caminhar, correr e descansar são exemplos de atividades de pessoas em uma cidade que podem ser detectadas a partir de uma modelagem em grafo capaz de agregar informações de diversos sensores. O tratamento desses dados de forma isolada e/ou combinada (fusão) aparece frequentemente em sistemas distribuídos.

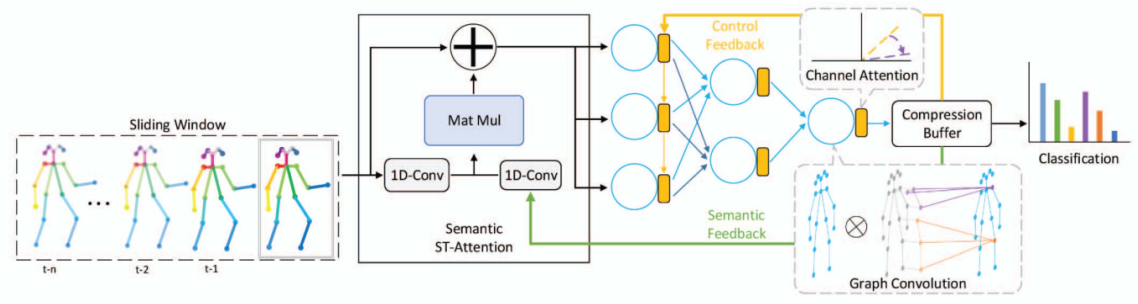


Figura 4.5. Modelo de reconhecimento de atividade humana de (Mondal et al., 2020)

Mondal et al. (2020) utilizam dados coletados de sensores de *smartphones* (e.g., acelerômetro, giroscópio, dentre outros) para prever o tipo de atividade desempenhada por um usuário em um dado momento (veja a Figura 4.5). Como esses dados são coletados continuamente, eles representam uma série temporal. Na modelagem do grafo, cada vértice representa uma atividade realizada em uma janela de tempo na série temporal. Essa atividade é caracterizada por dados agregados dos sensores embutidos (atributos dos vértices). Cada vértice é conectado com o vértice da janela de tempo anterior e posterior, considerando a série temporal. Em outras palavras, essa modelagem assume que cada atividade realizada está associada com os comportamentos anteriores e posteriores do indivíduo. A classificação de atividades (e.g., correr, caminhar, permanecer parado, andar de bicicleta, dentre outras) é realizada tanto a nível de vértice quanto a nível de grafo.

O reconhecimento de atividade humana também pode ser utilizado para sistemas de vigilância inteligente por vídeo (*smart video surveillance*). (Sanchez et al., 2021) consideram que uma atividade pode ser identificada com base em uma sequência de poses/ações contidos em um conjunto de *frames*. Nesse trabalho, é proposta uma rede espaço-temporal para o reconhecimento de atividade humana onde cada *frame* representa um corpo humano, que por sua vez é mapeado para um grafo, onde cada vértice corresponde a uma junção (*keypoint*) do corpo e as arestas conectam partes adjacentes. A matriz de atributos de vértices contém atributos que caracterizam as possíveis poses ou movimentos de cada indivíduo em um *frame*, e o processo de convolução agrega os movimentos realizados para se classificar o tipo de atividade realizada. Essa solução processa sequências de grafos utilizando uma arquitetura *ST-GCN* e se enquadra no contexto de tarefas de classificação de grafo.

4.5.9. Discussão

As redes neurais para grafos têm se mostrado flexíveis para a inserção en novos contextos. Porém, um ponto a se notar é a pouca utilização da matriz de atributos das arestas E. Isso se deve a três motivos:

1. **Camadas**: a quantidade de camadas capazes de processar a matriz E juntamente com as matrizes A e X é pequena.
2. **Modelagem**: na maior parte dos trabalhos, os autores utilizam a matriz de adjacência ponderada, de modo que os pesos são utilizados para caracterizar as arestas, em detrimento da utilização da matriz E.
3. **Contexto**: nem todos os contextos são abrangentes o suficiente para exigir a utilização da matriz E, onde cada aresta indica um tipo de relacionamento entre os vértices.

4.6. Dos dados brutos para o modelo *GNN*

Uma das principais razões para o advento das redes neurais de grafo é a sua capacidade de modelar problemas de diferentes contextos. Neste sentido, um dos aspectos relevantes que surge é o processo de transformação dos dados originais para a estrutura de grafo. Dependendo da modelagem utilizada, dados brutos de diferentes formas podem ser transformados em grafo, como: imagens, dados de *check-ins* de localização, trajetórias coletadas por sensores de *GPS*, sensores espalhados por estradas, dentre outros. Para isso, é necessária uma sequência de passos, desde o pré-processamento dos dados brutos até a implementação da rede neural. Além disso, séries temporais correspondem a um tipo de dado muito importante no contexto de *GNNs*, uma vez que elas podem ser utilizadas tanto para gerar grafos estáticos quanto para grafos dinâmicos, dependendo do contexto.

4.6.1. Representatividade dos vértices

Vértices podem representar usuários, pontos de interesse, estações de metrô, segmento de estrada, regiões, veículos, estações meteorológicas, sensores, imagens, junções do corpo humano, dentre outros elementos. A Tabela 4.2 sumariza os principais

elementos representados pelos vértices em cada tópico onde as redes neurais de grafos são empregadas no contexto das cidades inteligentes.

4.6.2. Tipos de matrizes de adjacência

Além de modelar a representatividade dos vértices, é importante definir como eles interagem entre si. As matrizes de adjacência podem indicar diferentes tipos de interações entre vértices, como: distância, duração, conectividade de transporte, relacionamento composto, relacionamento/contato, frequência, similaridade e vizinhança. A Tabela 4.3 apresenta os tipos de matrizes de adjacência e os estudos relacionados no contexto das cidades inteligentes.

4.6.3. Série temporal

Uma série temporal é uma sequência ordenada de observações comumente realizadas entre períodos iguais ou dentro de um intervalo máximo de tempo. As séries temporais podem ser utilizadas para se gerar grafos estáticos ou dinâmicos, dependendo do contexto e da abordagem utilizada. Elas são essenciais para as redes neurais de grafos e, em especial, para problemas relacionados às cidades inteligentes. Por exemplo, a partir de séries temporais de dados de navegação, são construídas sequências de grafos dinâmicos que caracterizam a movimentação de veículos em um conjunto de ruas no contexto de previsão de tráfego (Dai et al., 2020).

No contexto das cidades inteligentes, grafos podem ser gerados a partir de séries temporais de diferentes tipos: trajetórias de *GPS* de usuários móveis (S. Wu, Zhang, et al., 2020; Capanema et al., 2021b); trajetórias de *GPS* de veículos (Dai et al., 2020), sequências de dados coletados por sensores de tráfego (Yu et al., 2017; Deng et al., 2022), embutidos em dispositivos móveis (e.g., acelerômetro e giroscópio (Mondal et al., 2020)); dados coletados periodicamente sobre a movimentação de massas (La Gatta et al., 2020); imagens pré-processadas (e.g., algoritmos que detectam junções do corpo humano para posterior construção de grafo no problema de reconhecimento de atividade humana) (Sanchez et al., 2021).

A partir de séries temporais pode ser gerado um grafo ou uma sequência de grafos. Capanema and Silva (2021) utilizam a trajetória histórica de um usuário para gerar um grafo, que representa a sua mobilidade em todo o período. Por outro lado, sequências de grafos também podem ser criadas, como em (Ye et al., 2020), o que é comumente associado aos grafos dinâmicos, onde as características do grafo variam ao longo do tempo. Considerando que os grafos dinâmicos adicionam a dimensão temporal em sua definição, eles são divididos em dois tipos (L. Wu et al., 2022): grafos dinâmicos contínuos temporalmente e grafos dinâmicos discretos temporalmente. O primeiro não é criado a partir da agregação temporal dos dados, sendo mais rico em informação mas, ao mesmo tempo, mais complexo. O segundo, agrega dados temporalmente em sua criação, sendo mais simples e amplamente utilizado no contexto das cidades inteligentes. Dessa forma, os grafos dinâmicos discretos são descritos nos parágrafos seguintes.

Um grafo dinâmico discreto temporalmente (do inglês *discrete-time dynamic graph* ou simplesmente *DTDG*) é composto por uma sequência de *snapshots* de grafos

Tabela 4.2. Representatividade dos vértices para cada tópico.

Tópico	Vértice	Atributos
Sistemas de recomendação	Usuário	Representação latente de cada usuário (Xiao et al., 2020)
	Usuário-PoI (Usuário-Item)	Representação latente de usuário e *PoI* (Xiao et al., 2020)
	PoI-PoI (Item-Item)	Representação latente de cada *PoI* (S. Wu, Zhang, et al., 2020) e Distância e duração entre visitas (Capanema et al., 2021b)
Previsão de tráfego	Estação de sensor	Velocidade (Yu et al., 2017)
	Estação de metrô	Fluxos de passageiros (Ye et al., 2020)
	Segmento de estrada	Volume de tráfego e tempo de viagem (Dai et al., 2020)
	Região	Fluxo de viagens (Ke et al., 2021)
Classificação de função urbana	Segmento de estrada	(S. Hu et al., 2021)
	Região	(Yang et al., 2022)
Disseminação de doenças	Usuário	Estado de saúde (Tomy et al., 2022)
	Região	População, densidade demográfica e fluxo de entrada (La Gatta et al., 2020)
Agentes autônomos	Veículo	Velocidade, posição, localização e intenção de movimento (S. Chen et al., 2021)
Detecção de anomalia	Região	Fluxo de entrada e saída (Deng et al., 2022) e dados da fronteira geográfica da região (Y. Hu et al., 2020)
	Estação meteorológica	Dados de pressão atmosférica, temperatura, velocidade do vento, humidade, nível de precipitação, dentre outros (Owerko et al., 2018)
	Sensor	Dados coletados pelo sensor (D. Chen et al., 2021), (Deng et al., 2022)
Reidentificação de indivíduo	Imagem de referência - Imagem de galeria	Representação codificada da similaridade entre imagens (Shen et al., 2018)
Reconhecimento de atividade humana	Amostra de atividade	Dados agregados de acelerômetro, giroscópio, dentre outros (Mondal et al., 2020). esses dados são coletados em uma janela de tempo da atividade
	Junção do corpo humano (*keypoint*)	Atributos categóricos que indicam o tipo da pose/movimento em cada *frame* (Sanchez et al., 2021)

Tabela 4.3. Principais tipos de matrizes de adjacência em termos dos significados das arestas.

Tipo de matriz de adjacência	Definição	Estudo(s)
Distância	$A_{ij} = dis_{ij}$ onde dis_{ij} é a distância entre v_i e v_j. Um caso particular é a matriz de **proximidade de distância**, $f(dis_{ij})$ onde $f(.)$ é uma função aplicada na distância que atribui maior peso às distâncias menores	(S. Wu, Zhang, et al., 2020), (Yu et al., 2017), (Ke et al., 2021), (Yang et al., 2022), (Deng et al., 2022) e (Owerko et al., 2018)
Duração	$A_{ij} = dur_{ij}$ onde dur_{ij} é a duração da viagem entre v_i e v_j. Um caso particular é a matriz de **proximidade temporal**, $f(dur_{ij})$ onde $f(.)$ é uma função aplicada na distância que atribui maior peso às distâncias menores	(Y. Hu et al., 2020)
Conectividade de transporte	$A_{ij} \neq 0$ (ponderado por alguma métrica) se é possível viajar de v_i para v_j e $A_{ij} = 0$ caso contrário	(Ye et al., 2020)
Composta	$A_{ij} = f^c(ij)$ onde $f^c(.)$ é uma função que agrega dados de proximidade, tempo de viagem, dentre outros	(Dai et al., 2020) e (D. Chen et al., 2021)
Relacionamento/ contato	$A_{ij} = 1$ se v_i e v_j estabelecem algum relacionamento ou contato e $A_{ij} = 0$ caso contrário	(Xiao et al., 2020), (Tomy et al., 2022), (Mondal et al., 2020), (Sanchez et al., 2021) e (Sanchez et al., 2021)
Frequência	A_{ij} indica a frequência de transições/viagens entre v_i e v_j	(Capanema et al., 2021b) e (La Gatta et al., 2020)
Similaridade	A_{ij} representa o grau de similaridade entre as representações de v_i e v_j	(Shen et al., 2018)
Vizinhança	$A_{ij} = 1$ se v_i e v_j forem vizinhos e $A_{ij} = 0$ caso contrário. Esse é um caso particular de relacionamento/contato para entidades geográficas	(S. Hu et al., 2021) e (S. Chen et al., 2021)

$[G^{(1)}, G^{(2)}, \ldots, G^{(t)}]$ coletados em intervalos de tempo regulares. Cada grafo é representado como $G^{(t)} = (V^{(t)}, A^{(t)}, X^{(t)})$, onde o conjunto de vértices $V^{(t)}$, a matriz de adjacência $A^{(t)}$ e a matriz de atributos dos vértices $X^{(t)}$ podem variar ao longo de intervalos discretos de tempo. Um tipo específico de grafo dinâmico discreto temporalmente é o grafo espaço-temporal. Nesta categoria de grafo, a topologia se mantém a mesma enquanto que os valores dos atributos podem mudar ao longo do tempo (Skardinga et al., 2021), como em (Yu et al., 2017; Deng et al., 2022; Mondal et al., 2020; La Gatta et al., 2020; Sanchez et al., 2021). No contexto das cidades inteligentes, é frequente a utilização de redes neurais para grafos dinâmicos espaço-temporais na área de previsão de tráfego onde os vértices representam sensores em estradas, o que significa que não ocorrerá nenhuma alteração na estrutura do grafo (Dai et al., 2020).

Um aspecto importante é o intervalo de tempo utilizado para coletar os dados de cada grafo na sequência (i.e., *snapshot*). Caso esse período de tempo seja muito curto, o grafo eventualmente possuirá poucas arestas e as matrizes de atributos terão, eventualmente, pouca informação. Por outro lado, períodos longos para a coleta de *snapshots* podem resultar na perda de informação precisa entre as mudanças de um grafo para o outro.

4.6.4. Principais tarefas para cada tipo de problema

Compreender as tarefas mais comuns realizadas em cada área é importante para que pesquisadores possam direcionar melhor os seus esforços. A Tabela 4.4 apresenta as principais tarefas realizadas em cada tipo de tópico de *GNN* em cidades inteligentes, como descrito a seguir:

- Em **sistemas de recomendação**, as principais tarefas são: (1) predição de *link*, onde relacionamento entre usuários ou usuário-item são previstos; (2) classificação de vértice, onde se recomenda o item (e.g., *PoI*) de maior associação.

- Os problemas de **previsão de tráfego** se concentram nas seguintes tarefas: (1) regressão de grafo, onde o tempo de viagem em uma região ou o fluxo de tráfego de uma rede são previstos, por exemplo; (2) regressão de vértice, onde a velocidade, fluxos de passageiros, dentre outros aspectos, são previstos.

- No contexto de **classificação de função urbana**, a principal tarefa é a classificação de vértice, onde segmentos de estradas e regiões, por exemplo, têm os seus tipos classificados.

- Os problemas de **disseminação de doenças** estão frequentemente associados às tarefas de: (1) classificação de vértice, onde o estado de um indivíduo (vértice) é previsto; (2) em regressão de vértice, onde valores epidemiológicos (e.g., quantidade de indivíduos contaminados) podem ser previstos para cada localidade/região.

- Os problemas de **agentes autônomos** se concentram em tarefas de classificação de vértice onde, por exemplo, as possíveis ações (e.g., mudança de faixa de trânsito) de veículos são previstas.

- Em problemas de **detecção de anomalia**, as principais tarefas são: (1) predição de *link* em uma arquitetura *auto-encoder* ao se reconstruir a matriz de adjacência; (2) regressão de vértice em uma arquitetura *GAN*, onde os valores previstos dos atributos da matriz X são utilizados na detecção de anomalia.
- Os problemas de **reidentificação de indivíduos** se concentram em tarefas de classificação de vértice. Nesse caso, cada vértice pode conter um conjunto de atributos que representam a similaridade entre duas imagens. Por fim, cada vértice é classificado como pertencente a um dado indivíduo ou não.
- Em sistemas de **reconhecimento de atividade humana**, as principais tarefas são: (1) classificação de vértice para a predição do tipo de atividade realizada; (2) classificação de grafo para a predição da atividade global composta por um conjunto de ações (vértices).

Tabela 4.4. Principais tópicos e tarefas de *GNN* associadas

Tópico	Tarefa	Trabalho(s)
Sistemas de recomendação	Predição de *link*, classificação de vértice	(Xiao et al., 2020)
Previsão de tráfego	Regressão de grafo Regressão de vértice	(Dai et al., 2020) (Yu et al., 2017), (Ye et al., 2020)
Classificação de função urbana	Classificação de vértice	(S. Hu et al., 2021), (Yang et al., 2022), (Capanema et al., 2021b)
Disseminação de doenças	Classificação de vértice Regressão de vértice	(Tomy et al., 2022) (La Gatta et al., 2020)
Agentes autônomos	Classificação de vértice	(S. Chen et al., 2021)
Detecção de anomalia	Predição de *link* Regressão de vértice	(Y. Hu et al., 2020) (Deng et al., 2022)
Reidentificação de indivíduo	Classificação de vértice	(Shen et al., 2018)
Reconhecimento de atividade humana	Classificação de vértice Classificação de grafo	(Mondal et al., 2020) (Mondal et al., 2020), (Sanchez et al., 2021)

4.6.5. Passo a passo

Considerando a capacidade dos grafos em modelar diversos problemas, é necessário estruturar o processo de conversão de dados brutos para a estrutura de grafo. Essa modelagem é composta por uma sequência de passos, onde são definidos os seguintes aspectos:

1. **Estrutura de grafo**: o grafo por ser direcionado/não direcionado, homogêneo/heterogêneo e estático/dinâmico.

2. **Tarefa**: a tarefa pode ser a nível de vértice, aresta ou de grafo.

3. **Função de perda**: com base na tarefa selecionada, se define a função de perda. Por exemplo, se a tarefa é uma classificação de vértice, então uma função de perda apropriada pode ser a *categorical cross entropy*.

4. **Tipo de supervisão**: com base nos dados utilizados e a disponibilidade de rótulos, deve-se definir o tipo de supervisão no treinamento: supervisionado, semi-supervisionado e não supervisionado.

5. **Camadas**: a escolha das camadas utilizadas depende do tipo do grafo, das entradas disponíveis (i.e., as matrizes de entrada A, X e E) e da tarefa escolhida.

4.7. Desafios e questões abertas

As redes neurais de grafos apresentam tanto desafios teóricos quanto práticos. Além disso, alguns tópicos e técnicas têm se sobressaído e demonstrado serem promissoras para futuros trabalhos.

4.7.1. Desafios

O processo de aprendizagem em redes neurais de grafos é caracterizado pela agregação recursiva das informações de vértices vizinhos, e à medida que o modelo se torna cada vez mais profundo, dois problemas emergem (Zhou et al., 2020):

- **Explosão de vizinhança**: ao se utilizar consecutivamente múltiplas camadas de *message passing*, o número de vértices vizinhos cresce exponencialmente. Como alternativa, ao invés de agregar as informações de todos os vértices vizinhos a cada iteração, as camadas de amostragem agregam apenas subconjuntos de vértices a cada iteração, o que reduz a quantidade de tempo e memória necessários para a operação (D. Chen et al., 2020).

- **Sobre-suavização**: a intuição por trás do processo de *message passing* é que os vértices mais relacionados entre si assumem representações similares, o que é comumente chamado de suavização, um processo natural das redes *GNN*. No entanto, à medida que mais camadas de *message passing* são adicionadas, o número de vértices vizinhos cresce exponencialmente (explosão de vizinhança) e tanto informações úteis quanto ruídos são adicionados nas novas representações. Por exemplo, interações entre vértices de uma mesma classe são benéficos, enquanto que interações entre vértices de classes diferentes trazem ruídos. Neste sentido, as representações dos vértices tendem a se tornarem sobre-suavizadas (i.e., todos os vértices tendem a assumir representações semelhantes), o que torna difícil para que os modelos possam distinguir corretamente vértices de diferentes classes. A sobre-suavização é especialmente negativa para tarefas a nível de vértice (L. Wu et al., 2022). Para mitigar esse problema, operações de *skip connection* no contexto de grafos têm sido utilizadas para propagar representações históricas de vértices (D. Chen et al., 2020).

Além desses problemas, outro aspecto importante é a **invariância**: as camadas de redes neurais de grafos devem ser equivariantes em tarefas a nível de vértice e invariantes em tarefas a nível de grafo (Z. Wu et al., 2020). Isto significa que, quando a ordem dos vértices é alterada, apenas a ordem (equivariância) em que eles estão dispostos é alterada na nova representação gerada por uma camada de *message passing*. Por outro lado, a variação na ordem dos vértices do grafo não deve alterar a representação final gerada em uma tarefa a nível de grafo, em especial quando se utiliza camadas de *pooling* para a redução no tamanho do grafo.

4.7.2. Questões abertas

As principais questões abertas e os temas mais promissores de possíveis trabalhos futuros estão apresentados a seguir:

- **Enriquecimento de Dados**: diferentemente do domínio da imagem, onde rotações, adição de ruído e outras técnicas são suficientes para gerar novos dados, os grafos devem ser enriquecidos de acordo com a estrutura de dados complexa (e.g., correlação entre vértices e arestas). Quanto ao contexto das cidades inteligentes, a necessidade de enriquecimento é alta uma vez que, é comum que os dados utilizados sejam esparsos (e.g., dados de *check-in*), levando à construção de poucas instâncias de grafos ou grafos com poucas arestas. Zhao et al. (2021) melhoraram a classificação semi-supervisionada de vértices através de uma nova abordagem para *data augmentation* para grafos, onde arestas são adicionadas entre vértices que supostamente pertencem a uma mesma classe, ao mesmo tempo em que arestas de vértices de classes distintas são removidas.

- **Uso de múltiplas técnicas**: A utilização das *GNNs* juntamente com as *RNNs* (Capanema et al., 2021b) e *CNNs* (Shen et al., 2018; Lin et al., 2021) tem se mostrado promissora em diversas áreas. Com exceção de (Capanema et al., 2021b), a combinação de *GNNs* com *RNNs* ou *CNNs* está frequentemente associada ao processamento de sequências de grafos dinâmicos, onde é necessário capturar a evolução dos vértices e das arestas ao longo do tempo. Grafos espaço-temporais têm sido frequentemente utilizados para modelar fluxos de tráfego, uma vez que a topologia do grafo permanece a mesma. No entanto, grafos dinâmicos a nível estrutural, com adição e remoção de vértices, ainda foram pouco explorados nos problemas de cidades inteligentes apresentados neste capítulo.

- **Aprendizado multi-tarefa**: a estrutura de grafo pode ser utilizada para integrar o conhecimento sobre diferentes fontes de dados no aprendizado multi-tarefa, como: imagens, texto e bases de conhecimento (L. Wu et al., 2022).

- **Escalabilidade**: os módulos de amostragem têm sido extensivamente estudados como alternativa para se processar grafos grandes. No entanto, é importante que sejam desenvolvidos métodos adequados para grafos heterogêneos e dinâmicos (L. Wu et al., 2022).

4.8. Prática

Nesta seção, é apresentado um experimento prático no contexto das cidades inteligentes. Para isto, são descritos os principais conceitos da biblioteca *Spektral* que pertencente à linguagem *Python*. Além disso, é apresentada a modelagem da solução considerando a geração do grafo e o processo de desenvolvimento da rede neral, além da apresentação de um código de exemplo seguido de sugestões de melhorias.

4.8.1. Biblioteca *Spektral*

A biblioteca *Spektral* (Bianchi et al., 2020) na sua versão 1.1.0 é utilizada neste capítulo para o treinamento de redes neurais de grafos. O pacote é baseado no *Tensorflow*, se beneficiando dos seus recursos para ambiente de produção.

O treinamento de redes de grafos não é trivial como em outros contextos. Por exemplo, com relação às redes *CNN*, imagens podem ser cortadas ou preenchidas, através do *padding* para assumirem um mesmo tamanho. Da mesma forma, em redes *RNN* as sequências podem ser preenchidas. O domínio de grafo não permite que operações “diretas"como essas sejam realizadas, pois elas assumem, por exemplo, a proximidade espacial dos *pixels*, o que não existe em grafos. A retirada de um vértice pode, por exemplo, desconectar partes de um grafo.

Dessa forma, a biblioteca possui o conceito de *modo de dados*, que organiza os modos como as entradas são processadas pela rede neural. Os quatro modos de dados são apresentados a seguir:

1. *Single*: apenas um grafo é utilizado.

2. *Disjoint*: um *batch* de grafos é representado pelas suas uniões disjuntas. Caso não se deseje utilizar matrizes com representação densa, nem realizar o preenchimento com zero, esse é o modo mais adequado.

3. *Batch*: os grafos devem ter o mesmo número de vértices, o que é comumente alcançado com o preenchimento de zeros (i.e., *zero-padding*). esse modo tem maior comunalidade com os recursos do *Tensorflow*.

4. *Mixed*: todos os grafos compartilham de uma mesma matriz de adjacência, variando apenas as matrizes de atributos. Para evitar replicações e, portanto, melhorar o desempenho, a matriz de adjacência é processada no modo *single*, enquanto que a matriz de atributos é processada no modo *batch*.

4.8.2. Experimento

O experimento realizado neste capítulo consiste da modelagem e teste de uma rede neural de grafos para o problema de predição de categoria de pontos de interesse (*PoIs*). Um ponto de interesse é um local/estabelecimento frequentemente visitado por um indivíduo. Esse problema se insere no contexto de classificação de função urbana, onde cada *PoI* corresponde a um vértice e as arestas representam as viagens realizadas por um usuário móvel entre pares de *PoIs*.

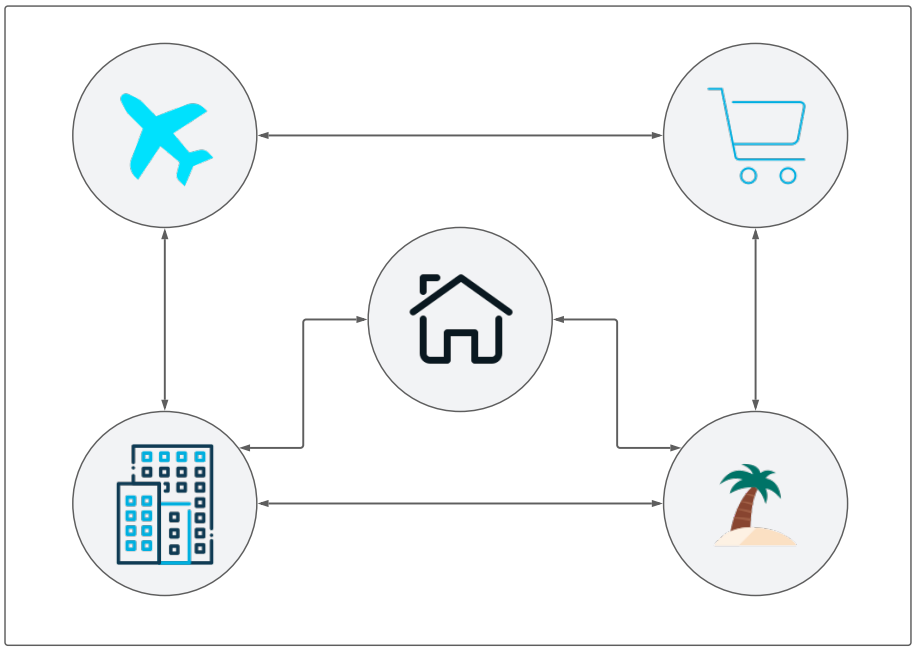


Figura 4.6. Exemplo de grafo não direcionado *PoIxPoI*, onde cada *PoI* é representado por um vértice e as viagens realizas por um indivíduo entre cada par de *PoIs* são representadas por arestas

4.8.2.1. Motivação

Em alguns serviços baseados em geolocalização em que a mobilidade dos usuários é frequentemente registrada, é comum que nenhuma informação adicional seja fornecida além dos dados brutos do *GPS*. Serviços de mapas como *OSM* (*Open Street Map*) e *Google Maps* podem ser usados para anotar a semântica dos *PoIs*; no entanto, essa abordagem direta nem sempre é suficiente, pois é comum que muitos lugares ainda não estejam anotados no sistema de mapas, ou haja um custo para usar esse tipo de serviço. Desse forma, é necessário fornecer um modelo *offline* e sem custo, capaz de extrair os padrões de visitas a cada tipo de categoria de *PoI* para realizar, dessa forma, o enriquecimento semântico de outras bases de dados.

4.8.2.2. Modelagem

A base de dados utilizada contém *check-ins* de localização de usuários da rede social *Gowalla* (Liu et al., 2014). Cada registro contém as seguintes informações: identificador do usuário e local visitado (assumimos que esse local é um *PoI*), data e horário, latitude e longitude, além da categoria do local (i.e., rótulo do problema de classificação).

Com base nos *check-ins* de localização de um usuário, é construído um grafo, onde cada vértice corresponde a um *PoI* e cada aresta conecta *PoIs* visitados consecutivamente, como é mostrado na Figura 4.6.

Com relação às entradas para o modelo, a matriz de adjacência A é ponderada

pela quantidade de viagens consecutivas entre cada par de *PoIs*. A matriz de atributos dos vértices $X \in \mathbb{R}^{N \times 48}$ representa a quantidade de visitas feitas à cada *PoI* para cada hora do dia (i.e., 24 horas para dias de semana e 24 horas para finais de semana). No processo de *message passing* cada *PoI*, representado por um vértice, agrega os padrões de visita (i.e., horários de visitação) dos estabelecimentos visitados antes e depois (i.e., vértices vizinhos). Como o grafo é ponderado, os pares de *PoIs* que um dado indivíduo fez mais visitas consecutivas possui um maior peso na etapa de agregação dos valores de seus atributos. Ao final desse processo, a representação de cada *PoI* contêm as características de visitação dos estabelecimentos vizinhos considerando o histórico de mobilidade de um usuário móvel.

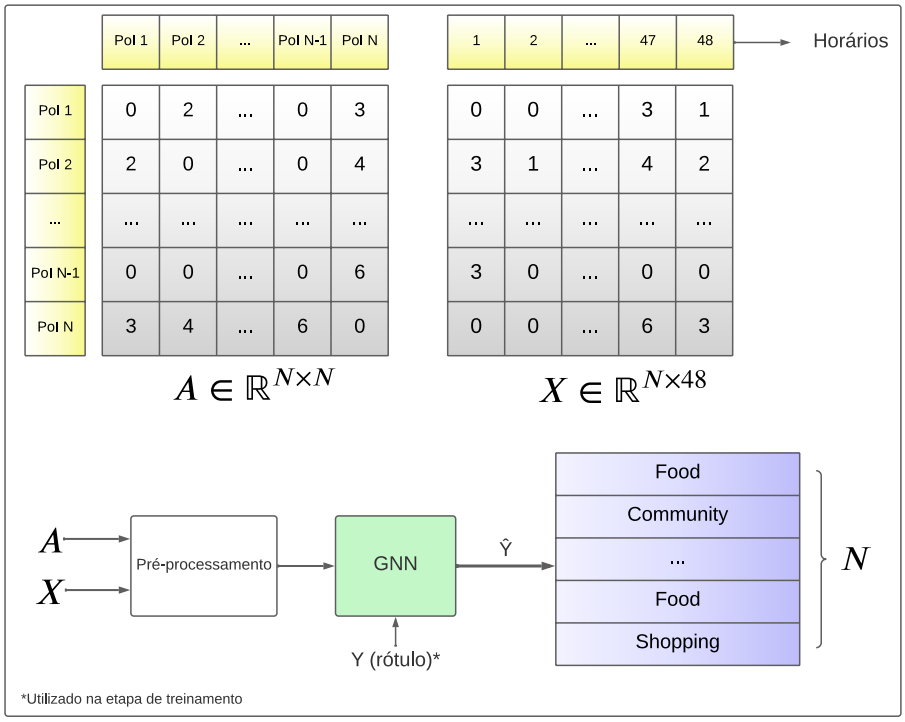


Figura 4.7. Diagrama de modelagem da solução

A Figura 4.7 exibe o diagrama de modelagem da solução. As matrizes de entrada A e X estão, inicialmente, não normalizadas. Na etapa de pré-processamento, a matriz de adjacência é modificada de acordo com a camada de *message passing* utilizada que, no presente exemplo, corresponde à camada *ARMA*. A biblioteca *Spektral* fornece essas operações de pré-processamento. Adicionalmente, a matriz de atributos dos vértices X pode ser normalizada. Por fim, as entradas são enviadas para o modelo *GNN*, onde são previstas as categorias dos *PoIs* representados por cada vértice do grafo. A base de dados *Gowalla* contém sete tipos de categorias de estabelecimentos: *Community*, *Entertainment*, *Food*, *Nightlife*, *Outdoors*, *Shopping* e *Travel*.

4.8.2.3. Exemplo de implementação

A Figura 4.8 apresenta um trecho de código de rede neural de grafo para a classificação de vértice e que pode ser utilizado no contexto de classificação de *PoIs*. São utilizadas duas camadas *ARMA* para o processo de *message passing*. Como entrada, são utilizadas as matrizes de adjacência *A_input* e de atributos de vértices *X_input*. Além dos hiperparâmetros comuns de uma camada de rede neural, é importante observar que essa camada requer valores para *order* e *iterations* que indicam a quantidade de blocos de camadas e a quantidade de camadas *GCS* em cada bloco. Além disso, apenas a matriz de atributos de vértices *X* é atualizada a cada iteração, e a matriz de adjacência é mantida com os mesmos valores iniciais. Outro aspecto importante, é que essa camada possui dois parâmetros para funções de ativação, sendo o primeiro utilizado pelas camadas *GCS* internas e o segundo aplicado sobre a saída da camada *ARMA*. Como esse é um problema de classificação com diferentes rótulos, a função de ativação da última camada *ARMA* é a *softmax*.

```
class GNN(Model):
    def __init__(self):
        super().__init__()
        self.mask = GraphMasking()
        self.conv1 = ARMAConv(
        16,
        iterations=1,
        order=2,
        share_weights=True,
        dropout_rate=0.75,
        activation="elu",
        gcn_activation="elu",
        kernel_regularizer=l2(5e-5)
    )
        self.dropout = Dropout(0.6)
        self.conv2 = ARMAConv(
        7,
        iterations=1,
        order=1,
        share_weights=True,
        dropout_rate=0.75,
        activation="softmax",
        gcn_activation=None,
        kernel_regularizer=l2(5e-5),
    )
    def call(self, inputs):
        X_input, A_input = inputs
        X = self.mask(X_input)
        X = self.conv1([X, A_input])
        X = self.dropout(X)
        output = self.conv2([X, A_input])
        return output
```

Figura 4.8. Definição do modelo *GNN*

A Figura 4.9 apresenta o trecho de código onde o modelo final é criado considerando as matrizes de entrada e a saída. A rede neural é compilada utilizando o otimizador *Adam*, a função de perda *categorical crossentropy* e a métrica acurácia. Após isso, é realizado o treinamento e a validação com os dados de teste utilizando o método *fit*. O código completo juntamente com os resultados estão disponíveis no repositório deste capítulo[1]. O algoritmo do exemplo apresentado alcança cerca de 40% de acurácia.

```
model = GNN()
opt = Adam(lr=0.0001)
model.compile(optimizer=opt,
              loss="categorical_crossentropy",
              metrics=["acc"])

model.fit(
    loader_tr.load(),
    steps_per_epoch=loader_tr.steps_per_epoch,
    epochs=epochs,
    validation_data=loader_va.load(),
    validation_steps=loader_va.steps_per_epoch,
    callbacks=[EarlyStopping(patience=3,
                             restore_best_weights=True)],
)
```

Figura 4.9. Configuração do treinamento do modelo *GNN*

Considerando o contexto desse experimento e as entradas disponíveis para o modelo, são propostas direções para melhorias no desempenho da rede neural:

- Alteração nos valores dos hiperparâmetros.
- Modificar a quantidade de camadas *GNN* utilizadas.
- Avaliar o emprego de outras camadas de *message passing*. A biblioteca *Spektral* contém diversas camadas que podem ser utilizadas no presente problema, como: *GCN*, *GCS*, *APPNP* (*Approximate Personalized Propagation of Neural Predictions*) (Klicpera et al., 2018), dentre outras. É importante destacar que ao se mudar a camada utilizada, o método de pré-processamento aplicado sobre a matriz de adjacência também deve ser modificado utilizando-se a função apropriada. Outros exemplos de como isto é realizado estão presentes na documentação[2] e no repositório[3] da biblioteca.
- Avaliar o uso de outros tipos de camadas.

[1] https://github.com/claudiocapanema/minicurso_gnn_sbrc2022. Acessado em 13/5/2022.
[2] https://graphneural.network/. Acessado em 3/5/2022.
[3] https://github.com/danielegrattarola/spektral/. Acessado em 3/5/2022.

4.9. Considerações finais

Este capítulo discorreu sobre a redes neurais de grafos no contexto das cidades inteligentes, incluindo os fundamentos teóricos e aplicações práticas das *GNNs*. Inicialmente, foram apresentados os principais conceitos, os tipos de entradas possíveis, as tarefas existentes, os tipos de treinamentos e as principais camadas *GNN* juntamente com o fundamento teórico e as motivações para cada classe de camadas.

Em seguida, foram apresentadas as principais aplicações das redes neurais de grafos no contexto das cidades inteligentes, o que inclui os tópicos de sistemas de recomendação, previsão de tráfego, classificação de função urbana, disseminação de doenças, agentes autônomos, detecção de anomalia, reidentificação de indivíduo e reconhecimento de atividade humana. Além disso, foi discutido o processo de conversão de dados brutos para a estrutura de grafo com foco especial em séries temporais. Dois aspectos que foram apresentados e que são importantes para o desenvolvimento de novas soluções são: (1) principais tarefas utilizadas de *GNN* em cada tópico de problema; (2) desafios e questões abertas que são importantes para indicar a direção de novos métodos.

Por último, foi apresentada a modelagem de um experimento prático para o problema de classificação de pontos de interesse envolvendo redes neurais de grafos através da biblioteca *Spektral*.

4.10. Agradecimento

Este trabalho contou com o apoio da CAPES.

Referências

Atwood, J., & Towsley, D. (2016). Diffusion-convolutional neural networks. *Advances in neural information processing systems*, *29*.

Bianchi, F. M., Grattarola, D., & Alippi, C. (2020). Spectral clustering with graph neural networks for graph pooling. In *International conference on machine learning* (pp. 874–883).

Bianchi, F. M., Grattarola, D., Livi, L., & Alippi, C. (2021). Graph neural networks with convolutional arma filters. *IEEE transactions on pattern analysis and machine intelligence*.

Bruna, J., Zaremba, W., Szlam, A., & LeCun, Y. (2013). Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*.

Capanema, C. G. S., & Silva, F. A. (2021). Detecção de pontos de interesse e predição de próximo local de visita de usuários móveis com base em dados esparsos. In *Anais estendidos do xxxix simpósio brasileiro de redes de computadores e sistemas distribuídos* (pp. 129–136).

Capanema, C. G. S., Silva, F. A., & Silva, T. R. M. B. (2019). Identificacão e classificacão de pontos de interesse individuais com base em dados esparsos. In *Anais do xxxvii simpósio brasileiro de redes de computadores e sistemas distribuídos* (pp. 15–28).

Capanema, C. G. S., Silva, F. A., Silva, T. R. M. B., & Loureiro, A. A. F. (2021a). Dcluster: Geospatial analytics with poi identification. *Journal of Information and Data Management*, *12*(2).

Capanema, C. G. S., Silva, F. A., Silva, T. R. M. B., & Loureiro, A. A. F. (2021b). Poirgnn: Using recurrent and graph neural networks to predict the category of the next point of interest. In *Proceedings of the 18th acm symposium on performance evaluation of wireless ad hoc, sensor, & ubiquitous networks* (pp. 49–56).

Chen, D., Lin, Y., Li, W., Li, P., Zhou, J., & Sun, X. (2020). Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. In *Proceedings of the aaai conference on artificial intelligence* (Vol. 34, pp. 3438–3445).

Chen, D., Liu, R., Hu, Q., & Ding, S. X. (2021). Interaction-aware graph neural networks for fault diagnosis of complex industrial processes. *IEEE Transactions on Neural Networks and Learning Systems*.

Chen, J., Ma, T., & Xiao, C. (2018). Fastgcn: fast learning with graph convolutional networks via importance sampling. *arXiv preprint arXiv:1801.10247*.

Chen, S., Dong, J., Ha, P., Li, Y., & Labi, S. (2021). Graph neural network and reinforcement learning for multi-agent cooperative control of connected autonomous vehicles. *Computer-Aided Civil and Infrastructure Engineering*, *36*(7), 838–857.

Cheng, D., Yang, F., Xiang, S., & Liu, J. (2022). Financial time series forecasting with multi-modality graph neural network. *Pattern Recognition*, *121*, 108218.

Chiang, W.-L., Liu, X., Si, S., Li, Y., Bengio, S., & Hsieh, C.-J. (2019). Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks. In *Proceedings of the 25th acm sigkdd international conference on knowledge discovery & data mining* (pp. 257–266).

Dai, R., Xu, S., Gu, Q., Ji, C., & Liu, K. (2020). Hybrid spatio-temporal graph convolu-

tional network: Improving traffic prediction with navigation data. In *Proceedings of the 26th acm sigkdd international conference on knowledge discovery & data mining* (pp. 3074–3082).

Defferrard, M., Bresson, X., & Vandergheynst, P. (2016). Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems, 29*.

Deng, L., Lian, D., Huang, Z., & Chen, E. (2022). Graph convolutional adversarial networks for spatiotemporal anomaly detection. *IEEE Transactions on Neural Networks and Learning Systems*.

Do, K., Tran, T., & Venkatesh, S. (2019). Graph transformation policy network for chemical reaction prediction. In *Proceedings of the 25th acm sigkdd international conference on knowledge discovery & data mining* (pp. 750–760).

Duarte, J., & Vlimant, J.-R. (2022). Graph neural networks for particle tracking and reconstruction. In *Artificial intelligence for high energy physics* (pp. 387–436). World Scientific.

Grattarola, D., & Alippi, C. (2021). Graph neural networks in tensorflow and keras with spektral [application notes]. *IEEE Computational Intelligence Magazine, 16*(1), 99–106.

Hamilton, W., Ying, Z., & Leskovec, J. (2017). Inductive representation learning on large graphs. *Advances in neural information processing systems, 30*.

Hu, S., Gao, S., Wu, L., Xu, Y., Zhang, Z., Cui, H., & Gong, X. (2021). Urban function classification at road segment level using taxi trajectory data: A graph convolutional neural network approach. *Computers, Environment and Urban Systems, 87*, 101619.

Hu, Y., Qu, A., & Work, D. (2020). Graph convolutional networks for traffic anomaly. *arXiv preprint arXiv:2012.13637*.

Jiang, W., & Luo, J. (2021). Graph neural network for traffic forecasting: A survey. *arXiv preprint arXiv:2101.11174*.

Ke, J., Qin, X., Yang, H., Zheng, Z., Zhu, Z., & Ye, J. (2021). Predicting origin-destination ride-sourcing demand with a spatio-temporal encoder-decoder residual multi-graph convolutional network. *Transportation Research Part C: Emerging Technologies, 122*, 102858.

Kipf, T., Fetaya, E., Wang, K.-C., Welling, M., & Zemel, R. (2018). Neural relational inference for interacting systems. In *International conference on machine learning* (pp. 2688–2697).

Kipf, T. N., & Welling, M. (2016a). Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Kipf, T. N., & Welling, M. (2016b). Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*.

Klicpera, J., Bojchevski, A., & Günnemann, S. (2018). Predict then propagate: Graph neural networks meet personalized pagerank. *arXiv preprint arXiv:1810.05997*.

La Gatta, V., Moscato, V., Postiglione, M., & Sperli, G. (2020). An epidemiological neural network exploiting dynamic graph structured data applied to the covid-19 outbreak. *IEEE Transactions on Big Data, 7*(1), 45–55.

Li, Y., Gu, C., Dullien, T., Vinyals, O., & Kohli, P. (2019). Graph matching networks for learning the similarity of graph structured objects. In *International conference on*

machine learning (pp. 3835–3845).

Liang, T., Sheng, X., Zhou, L., Li, Y., Gao, H., Yin, Y., & Chen, L. (2021). Mobile app recommendation via heterogeneous graph neural network in edge computing. *Applied Soft Computing*, *103*, 107162.

Lin, D., Lin, J., Zhao, L., Wang, Z. J., & Chen, Z. (2021). Multilabel aerial image classification with a concept attention graph neural network. *IEEE Transactions on Geoscience and Remote Sensing*, *60*, 1–12.

Liu, Y., Wei, W., Sun, A., & Miao, C. (2014). Exploiting geographical neighborhood characteristics for location recommendation. In *Proceedings of the 23rd acm international conference on conference on information and knowledge management* (pp. 739–748).

Maguire, J. B., Grattarola, D., Mulligan, V. K., Klyshko, E., & Melo, H. (2021). Xenet: Using a new graph convolution to accelerate the timeline for protein design on quantum computers. *PLoS computational biology*, *17*(9), e1009037.

Malekzadeh, M., Hajibabaee, P., Heidari, M., Zad, S., Uzuner, O., & Jones, J. H. (2021). Review of graph neural network in text classification. In *2021 ieee 12th annual ubiquitous computing, electronics & mobile communication conference (uemcon)* (pp. 0084–0091).

Mondal, R., Mukherjee, D., Singh, P. K., Bhateja, V., & Sarkar, R. (2020). A new framework for smartphone sensor-based human activity recognition using graph neural network. *IEEE Sensors Journal*, *21*(10), 11461–11468.

Owerko, D., Gama, F., & Ribeiro, A. (2018). Predicting power outages using graph neural networks. In *2018 ieee global conference on signal and information processing (globalsip)* (pp. 743–747).

Rhee, S., Seo, S., & Kim, S. (2017). Hybrid approach of relation network and localized graph convolutional filtering for breast cancer subtype classification. *arXiv preprint arXiv:1711.05859*.

Sanchez, J., Neff, C., & Tabkhi, H. (2021). Real-world graph convolution networks (rw-gcns) for action recognition in smart video surveillance. In *2021 ieee/acm symposium on edge computing (sec)* (pp. 121–134).

Shen, Y., Li, H., Yi, S., Chen, D., & Wang, X. (2018). Person re-identification with deep similarity-guided graph neural network. In *Proceedings of the european conference on computer vision (eccv)* (pp. 486–504).

Skardinga, J., Gabrys, B., & Musial, K. (2021). Foundations and modelling of dynamic networks using dynamic graph neural networks: A survey. *IEEE Access*.

Tomy, A., Razzanelli, M., Di Lauro, F., Rus, D., & Della Santina, C. (2022). Estimating the state of epidemics spreading with graph neural networks. *Nonlinear Dynamics*, 1–15.

Tran, D. V., Navarin, N., & Sperduti, A. (2018). On filter size in graph convolutional networks. In *2018 ieee symposium series on computational intelligence (ssci)* (pp. 1534–1541).

Tsitsulin, A., Palowitch, J., Perozzi, B., & Müller, E. (2020). Graph clustering with graph neural networks. *arXiv preprint arXiv:2006.16904*.

Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., & Bengio, Y. (2017). Graph attention networks. *stat*, *1050*, 20.

Wu, L., Cui, P., Pei, J., Zhao, L., & Song, L. (2022). Graph neural networks. In *Graph neural networks: Foundations, frontiers, and applications* (pp. 27–37). Springer.

Wu, S., Sun, F., Zhang, W., & Cui, B. (2020). Graph neural networks in recommender systems: a survey. *arXiv preprint arXiv:2011.02260*.

Wu, S., Zhang, Y., Gao, C., Bian, K., & Cui, B. (2020). Garg: anonymous recommendation of point-of-interest in mobile networks by graph convolution network. *Data Science and Engineering*, *5*(4), 433–447.

Wu, Y., Dai, H.-N., & Tang, H. (2021). Graph neural networks for anomaly detection in industrial internet of things. *IEEE Internet of Things Journal*.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Philip, S. Y. (2020). A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, *32*(1), 4–24.

Xiao, Y., Yao, L., Pei, Q., Wang, X., Yang, J., & Sheng, Q. Z. (2020). Mgnn: Mutualistic graph neural network for joint friend and item recommendation. *IEEE Intelligent Systems*, *35*(5), 7–17.

Yang, M., Kong, B., Dang, R., & Yan, X. (2022). Classifying urban functional regions by integrating buildings and points-of-interest using a stacking ensemble method. *International Journal of Applied Earth Observation and Geoinformation*, *108*, 102753.

Yao, L., Mao, C., & Luo, Y. (2019). Graph convolutional networks for text classification. In *Proceedings of the aaai conference on artificial intelligence* (Vol. 33, pp. 7370–7377).

Ye, J., Zhao, J., Ye, K., & Xu, C. (2020). Multi-stgcnet: A graph convolution based spatial-temporal framework for subway passenger flow forecasting. In *2020 international joint conference on neural networks (ijcnn)* (pp. 1–8).

Yin, C., Xiong, Z., Chen, H., Wang, J., Cooper, D., & David, B. (2015). A literature survey on smart cities. *Science China Information Sciences*, *58*(10), 1–18.

Ying, R., He, R., Chen, K., Eksombatchai, P., Hamilton, W. L., & Leskovec, J. (2018). Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th acm sigkdd international conference on knowledge discovery & data mining* (pp. 974–983).

Ying, Z., You, J., Morris, C., Ren, X., Hamilton, W., & Leskovec, J. (2018). Hierarchical graph representation learning with differentiable pooling. *Advances in neural information processing systems*, *31*.

You, Y., Chen, T., Sui, Y., Chen, T., Wang, Z., & Shen, Y. (2020). Graph contrastive learning with augmentations. *Advances in Neural Information Processing Systems*, *33*, 5812–5823.

Yu, B., Yin, H., & Zhu, Z. (2017). Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. *arXiv preprint arXiv:1709.04875*.

Zhang, S., Tong, H., Xu, J., & Maciejewski, R. (2019). Graph convolutional networks: a comprehensive review. *Computational Social Networks*, *6*(1), 1–23.

Zhang, Z., Cui, P., & Zhu, W. (2020). Deep learning on graphs: A survey. *IEEE Transactions on Knowledge and Data Engineering*.

Zhao, T., Liu, Y., Neves, L., Woodford, O., Jiang, M., & Shah, N. (2021). Data augmentation for graph neural networks. In *Proceedings of the aaai conference on artificial intelligence* (Vol. 35, pp. 11015–11023).

Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., … Sun, M. (2020). Graph neural networks: A review of methods and applications. *AI Open, 1*, 57–81.

Zou, D., Hu, Z., Wang, Y., Jiang, S., Sun, Y., & Gu, Q. (2019). Layer-dependent importance sampling for training deep and large graph convolutional networks. *Advances in neural information processing systems, 32*.

Capítulo 5

Monitoramento de Sinais Vitais Utilizando Redes Wi-Fi

Julio C. H. Soto, Iandra Galdino, Egberto Caballero, Vinicius Ferreira, Débora Muchaluat-Saade, Célio Albuquerque

Resumo

A pandemia de COVID-19 destacou a necessidade de usar procedimentos de monitoramento remoto de baixo custo para pacientes médicos. O uso das informações do canal de transmissão (Channel State Information – CSI) das redes Wi-Fi para monitorar pacientes remotamente é uma ferramenta capaz de suprir informações médicas de forma não invasiva e com baixo custo. Este capítulo apresenta as técnicas de captura de dados Wi-Fi CSI, as técnicas de processamento de sinais utilizadas e as aplicações para o monitoramento de sinais vitais. É comentada uma demonstração prática da captura dos sinais de respiração de um indivíduo utilizando essa tecnologia. Por fim, apresentam-se os desafios e perspectivas para pesquisas na área.

Abstract

The COVID-19 pandemic has highlighted the need to use low-cost remote monitoring procedures for medical patients. The use of Channel State Information (CSI) of Wi-Fi networks is a tool capable of providing non-invasive medical information in a low-cost manner. This chapter presents CSI Wi-Fi data capture techniques, signal processing techniques, and applications for vital signs monitoring. We also comment on a practical demonstration of capturing an individual's breathing signal using this technology. Finally, we present the research challenges and perspectives of this area.

5.1. Introdução

Diversos dispositivos médicos têm sido usados para ajudar a monitorar, diagnosticar e tratar doenças. Usualmente, esses dispositivos permitem a comunicação com uma rede interna ou até mesmo externa, para monitoramento, configuração e controle ou mesmo troca de informação em tempo real. O monitoramento contínuo da saúde do paciente

oferece um melhor conhecimento de sua condição e permite um melhor fluxo de informações para supervisão, tratamento e recuperação [Tan et al. 2018]. Devido à pandemia da COVID-19, observou-se um número crescente de pacientes que demandam cuidados de saúde. Por se tratar de uma doença altamente contagiosa com potencial de agravamento da condição de saúde, o monitoramento dos pacientes infectados deve ser cuidadoso, reduzindo ao máximo o contato entre os indivíduos. Os profissionais de saúde que tratam de pacientes contaminados pela COVID-19 e demais doenças infectocontagiosas devem utilizar equipamentos de proteção individual (EPI) de forma a minimizar o risco de contágio [Sharma et al. 2021, Li et al. 2019].

Diversas propostas têm sido apresentadas na literatura, relacionadas ao monitoramento sem contato de pacientes contaminados com doenças contagiosas, com o objetivo de lidar com essa demanda. Em [Zhao et al. 2018], por exemplo, os autores propuseram usar a tecnologia *Frequency Modulated Carrier Wave* (FMCW) para detectar atividades humanas através de sinais de radiofrequência. No entanto, o FMCW tem um custo elevado, o que torna essa tecnologia não acessível a todos. Outra solução possível para o monitoramento sem contato de pacientes é o uso da tecnologia de identificação por radiofrequência (*Radio Frequency IDentification* – RFID) [Khan 2017]. A utilização de RFID é uma abordagem interessante, entretanto, depende da aquisição de etiquetas RFID a serem conectadas aos pacientes. Portanto, a busca por uma abordagem nova, menos dispendiosa e sem o uso de dispositivos invasivos mostrou que os sinais de rádio Wi-Fi podem ser usados para detectar a presença de pessoas, rastrear atividades, movimentos e ainda captar sinais vitais humanos [Ma et al. 2019a, Yousefi et al. 2017].

Atualmente, os dispositivos Wi-Fi estão disponíveis em quase todos os ambientes médicos e residenciais. A tecnologia Wi-Fi está amplamente difundida no mundo e sua aplicação ao monitoramento da saúde é considerada de baixo custo, uma vez que se aproveita de dispositivos já utilizados para estabelecer uma rede Wi-Fi. Ainda, sua utilização é considerada não invasiva, uma vez que não há a necessidade de intervenção física no paciente. As ondas eletromagnéticas dos sinais Wi-Fi têm a particularidade de atravessar paredes, ou seja, em algumas aplicações não se necessita sequer de transmissão com linha de visada para o paciente, o que facilita sua utilização.

Sinais de rádio (ondas eletromagnéticas) podem ser usados para rastrear atividades humanas. As ondas de rádio são afetadas pelos movimentos humanos alterando as características das ondas que chegam até o receptor [Gu et al. 2017, Gu et al. 2018]. Essas alterações podem ser reconhecidas em um conjunto de dados chamado informação de estado do canal (*Channel State Information* - CSI). O CSI fornece informações de estado do canal na camada física (*Physical layer* - PHY), através de informações como amplitude, fase, e/ou indicador de intensidade do sinal recebido (*Received Signal Strength Indicator* - RSSI). Essa informação pode ser obtida em cada uma das subportadoras envolvidas em uma transmissão multiportadora [Wang et al. 2019, Liu et al. 2019]. Os padrões atuais de Wi-Fi, como o 802.11n/ac, utilizam a modulação por divisão ortogonal de frequências (*Orthogonal Frequency Division Multiplexing* - OFDM) na camada física. OFDM é uma técnica de modulação na qual a banda de frequência de transmissão é dividida em várias sub-bandas, cada uma com sua portadora própria, denominadas subportadoras. Cada uma das subportadoras pode fornecer informações detalhadas sobre o estado do canal [Lee et al. 2018]. O sinal CSI representa a resposta em frequência do canal (*Channel Fre-*

quency Response - CFR) para cada subportadora entre os pares de antenas de transmissão e recepção.

O CSI pode capturar as interferências que o corpo humano causa no sinal eletromagnético nos domínios do tempo, da frequência e domínios espaciais. Essas informações podem ser usadas para diferentes aplicações, como a detecção da presença humana, detecção de movimentos, identificação humana, detecção de queda, reconhecimento de gestos, localização humana e monitoramento das condições de saúde. Para monitoramento das condições de saúde, as subportadoras OFDM são usadas como vários sensores para detectar a mudança física de uma pessoa. Uma análise de forma de onda CSI é realizada para detectar atividades mínimas do corpo humano, como a respiração, os batimentos cardíacos, dentre outros [Gu et al. 2018, Damodaran et al. 2020, Lee et al. 2018].

Na literatura, encontra-se vários estudos que enfatizam o uso do sinal CSI como uma tecnologia acessível a todos para o monitoramento das atividades humanas [Yousefi et al. 2017, Ma et al. 2019a, Wang et al. 2019, Liu et al. 2019]. Além disso, o CSI é considerado uma ferramenta não invasiva para o paciente, o que gera maior aceitação do seu uso.

Muitos trabalhos de pesquisa foram desenvolvidos com o foco na comparação de diferentes tecnologias de detecção de características do ambiente ou de atividades humanas sem fio [Uchiyama et al. 2021, Wang et al. 2019, Ma et al. 2019a], reconhecimento de comportamento [Yousefi et al. 2017, Liu et al. 2019] e localização [Xiao et al. 2016, Yang et al. 2013]. Em [Uchiyama et al. 2021], os autores se concentraram em revisar as diferenças entre o sinal CSI de dispositivos Wi-Fi (Wi-Fi CSI), RFID e retrodifusão. Os autores de [Wang et al. 2019] analisaram os principais componentes e características centrais da arquitetura do sistema de reconhecimento do comportamento humano. Em [Ma et al. 2019a], os autores apresentaram uma revisão de técnicas de processamento de sinais, algoritmos, aplicações e resultados de desempenho. Youssefi et al. [Yousefi et al. 2017] apresentaram os avanços no reconhecimento passivo do comportamento humano. Em [Liu et al. 2019], os autores pesquisaram os sistemas de detecção sem fio existentes em termos de seus princípios básicos, técnicas e estruturas de sistema. Xiao et al. in [Xiao et al. 2016] também fornecem uma pesquisa sobre a localização interna sem dispositivos e com dispositivos, e [Yang et al. 2013] apresentou uma pesquisa sobre localização com ênfase nos princípios básicos e tendências futuras. Este último também destacou as diferenças entre CSI e RSSI em termos de camadas de rede, resolução de tempo, resolução de frequência, estabilidade e acessibilidade.

Diferentemente dos trabalhos existentes, este capítulo tem como foco a análise de dados de sinais CSI para monitoramento de sinais vitais humanos. Discutem-se técnicas de detecção, reconhecimento e estimativa de sinais vitais através de CSI. Dessa maneira há uma diretriz abrangente para adotar o CSI para fins de saúde sob uma perspectiva segura, escalável e de baixo custo. Por fim, apresentam-se as tendências e desafios futuros para aprimorar os recursos de detecção de sinais vitais através de CSI existentes e habilitar novas aplicações para o monitoramento da saúde.

Na Seção 5.2, são apresentados os principais conceitos relacionados a Wi-Fi CSI, com os conceitos básicos de modulação OFDM, a modelagem do canal sem fio Wi-Fi

e apontam-se as ferramentas utilizadas para a extração de características do canal, utilizando o *Channel State Information* (CSI). Na Seção 5.3, são discutidas as principais ferramentas utilizadas para processar os dados de CSI e algoritmos de detecção dos sinais vitais de indivíduos em ambientes monitorados por CSI. Na Seção 5.4, apresentam-se algumas aplicações de monitoramento de sinais vitais através de Wi-Fi CSI encontradas na literatura. Na Seção 5.5, uma atividade prática é discutida, mostrando todos os passos para se realizar o monitoramento da respiração de um indivíduo através do processamento de dados CSI. Já na Seção 5.6, destacam-se os desafios da obtenção e processamento dos dados de CSI e uma prospecção da diversidade de cenários e possíveis aplicações futuras dessa tecnologia. Por fim, a Seção 5.7 apresenta as considerações finais do capítulo.

5.2. Wi-Fi CSI

Esta seção apresenta uma visão geral de como os dados CSI são obtidos utilizando dispositivos Wi-Fi e como eles podem ser processados e utilizados em algumas aplicações. A Figura 5.1, adaptada de [Ma et al. 2019a], mostra a arquitetura geral do sistema usado para coleta, tratamento e estimativa de atividades humanas usando dados Wi-Fi CSI.

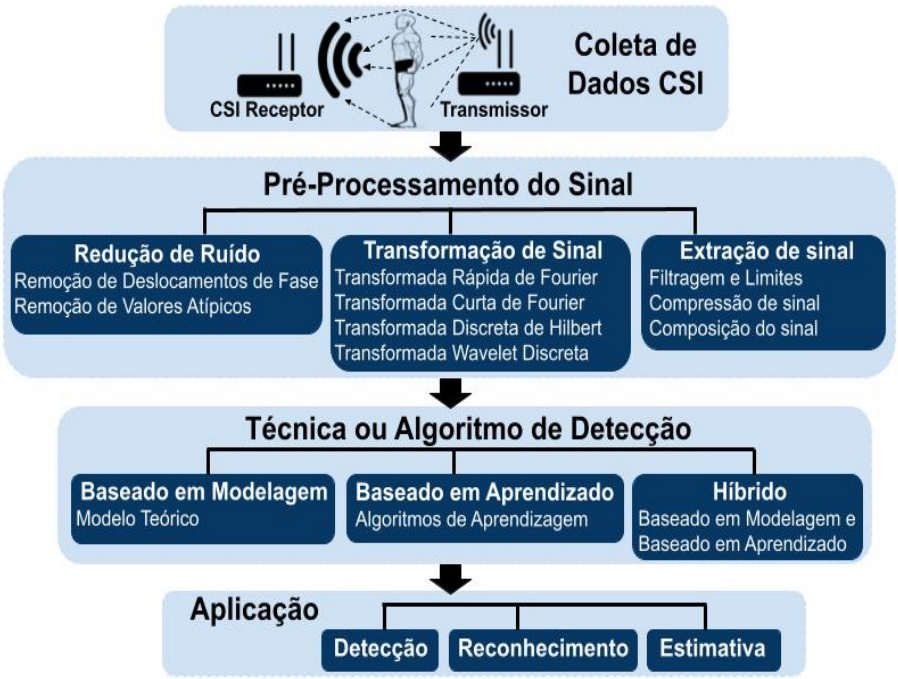


Figura 5.1. Estrutura do sistema Wi-Fi CSI

Em geral, o processo de coleta do CSI é realizado por um dispositivo equipado com placa de interface de rede (NIC - *Network Interface Card*). Então, o sinal de base adequado escolhido, como amplitude e/ou fase, deve ser extraído das informações coletadas. O sinal extraído alimenta o módulo de pré-processamento de sinal em uma segunda fase. Nesta fase, a fim de remover o ruído do sinal e obter dados CSI mais precisos, abordagens de pré-processamento são essenciais. O pré-processamento é realizado através de técnicas de redução de ruído, transformação de sinal e filtragem [Ma et al. 2019a, Gu et al. 2017, Gu et al. 2018]. Após o pré-processamento do sinal, segue-se a análise das ativi-

dades humanas por meio de algoritmos baseados em modelagem teórica e/ou algoritmos baseados em aprendizado. O algoritmo baseado em modelagem geralmente utiliza modelos típicos, como modelo de zona de Fresnel e ângulo de chegada (do inglês, *Angle-of-Arrival* - AoA). A abordagem baseada em modelagem enfrenta desafios na construção de um modelo. A abordagem baseada em aprendizado é usada principalmente em aplicações de identificação de movimento. Apesar de exigir uma etapa de treinamento, essa abordagem pode alcançar um bom desempenho. Finalmente, a aplicação pode detectar, estimar ou reconhecer algumas atividades humanas e sinais vitais [Ma et al. 2019a, Bowen et al. 2019a, Duan et al. 2018a]. Em suma, a arquitetura apresentada fornece uma visão geral da estrutura baseada em CSI para monitoramento de sinais vitais. Uma análise mais detalhada de cada uma das etapas será apresentada nas seções a seguir.

5.2.1. Modelagem Matemática do Canal

Esta seção apresenta um modelo matemático do sistema utilizado para coletar os dados CSI. Na especificação IEEE 802.11g/n/ac [IEEE 802.11 Working Group 2003, IEEE 802.11 Working Group 2009, IEEE 802.11 Working Group 2013], a camada física dos sistemas de comunicação Wi-Fi utiliza a técnica de modulação OFDM para bandas de frequência de 2.4GHz e 5GHz. O OFDM é uma técnica de modulação que utiliza um número pré-definido de subportadoras ortogonais entre si [Weinstein and Ebert 1971]. Consequentemente, as informações podem ser transmitidas independentemente entre si e também entre diferentes símbolos OFDM. As características intrínsecas ao OFDM o tornam uma boa alternativa para canais com múltiplos percursos e também para sistemas com múltiplas entradas e múltiplas saídas (do inglês, *Multiple-Input Multiple-Output* - MIMO).

Para coletar o sinal CSI, o transmissor Wi-Fi envia um sinal pré-definido (do inglês, *Long Training Fields* - LTFs), que contém informações em cada uma das subportadoras do preâmbulo do bloco. Na recepção, o receptor Wi-Fi estima as informações CSI representadas pela matriz ($\mathbf{H} \in \mathbb{C}^{M \times M}$), utilizando para isso o sinal recebido e os LTFs transmitidos. Desta forma, o sinal recebido y pode ser modelado no domínio da frequência como $y = \mathbf{H}x + n$, onde x e $y \in \mathbb{C}^{M \times 1}$ representam os símbolos OFDM transmitidos e recebidos respectivamente, $\mathbf{H}$ é uma matriz complexa que contém o sinal CSI, e $n \in \mathbb{C}^{M \times 1}$ representa o ruído [Lee et al. 2018].

Considerando um sistema Wi-Fi MIMO com m antenas transmissoras e n antenas receptoras, operando sob a especificação IEEE 802.11n, o sinal que contém a informação CSI estimada de cada fluxo de dados entre as antenas transmissoras e receptoras pode ser expresso como

$$\mathbf{H} = \begin{pmatrix} \mathbf{h}_{1,1} & \mathbf{h}_{1,2} & \dots & \mathbf{h}_{1,n} \\ \mathbf{h}_{2,1} & \mathbf{h}_{2,2} & \dots & \mathbf{h}_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{h}_{m,1} & \mathbf{h}_{m,2} & \dots & \mathbf{h}_{m,n} \end{pmatrix}, \tag{1}$$

onde $\mathbf{h}_{i,j} \in \mathbb{C}^{M \times M}$ representa os dados CSI entre a i-ésima antena de transmissão e a j-ésima antena de recepção.

Seja M o número de subportadoras do bloco OFDM utilizadas para estimar os dados CSI, a informação de estado do canal estabelecida entre um par de antenas (i, j),

definida por $\mathbf{h}_{i,j} \in \mathbb{C}^{(M\times 1)}$, pode ser representada matematicamente por um vetor contendo M elementos. Usa-se $\mathbf{h}$ para representar um $h_{i,j}$ genérico como

$$\mathbf{h} = [h_1, h_2, \ldots, h_M]^{\mathrm{T}}. \tag{2}$$

Esta análise pode ser desenvolvida utilizando tanto a técnica de RSSI quanto a de Wi-Fi CSI. Entretanto, a análise utilizando Wi-Fi CSI pode fornecer mais informações do que utilizando RSSI, pois a matriz de dados coletados é semelhante a uma imagem digital com alta resolução espacial.

5.2.2. Extração de dados CSI

Após apresentar o modelo matemático do sistema utilizado para obter os dados CSI, esta seção descreve brevemente algumas ferramentas utilizadas para capturar e coletar os dados CSI. Várias ferramentas têm sido propostas e estudadas na literatura para acessar os dados CSI com placas de rede de dispositivos Wi-Fi. A Tabela 5.1 resume algumas das ferramentas mais conhecidas [Gringoli et al. 2019].

Tabela 5.1. Ferramentas para extração de dados CSI

Ferramenta	Chipsets	Max. BW	Tecnologia
Linux 802.11n CSI Tool [Halperin et al. 2011]	IWL5300	40MHz	802.11n
Atheros CSI Tool [Xie et al. 2018]	AR9580, AR9590 AR9344, QCA9558	40MHz	802.11n
OpenFWWF CSI Tool [Gringoli and Nava 2009]	BCM4318	20MHz	802.11g
Nexmon CSI Extractor [Schulz et al. 2018]	BCM4365, 66 BCM4339, 58, 455	80MHz	802.11ac
GNU Radio [Khan et al. 2017]	USRP B200	80MHz	802.11ac

Linux 802.11n CSITool [Halperin et al. 2011] é uma ferramenta de extração que utiliza o chipset Intel Wi-Fi *Wireless Link* 5300 802.11n rádio MIMO. Esse é um *firmware* modificado que inclui todos os procedimentos adequados para capturar, coletar e analisar os dados CSI de um canal sem fio usando um *driver* de rede sem fio Linux de código aberto. Essa ferramenta inclui todos os *softwares* e *scripts* necessários para executar experimentos, ler e analisar medições de estado do canal, o que a torna uma boa alternativa para monitoramento de sinais vitais por meio de dados Wi-Fi CSI. O *chipset* IWL5300 802.11n fornece informações de estado do canal em um formato matricial. Cada entrada da matriz de canal é um número complexo, com resolução de 8 bits com sinal para as partes real e imaginária. Ele especifica o ganho e a rotação de fase de cada um dos caminhos entre um único par de antenas de transmissão e recepção.

Atheros-CSI-Tool [Xie et al. 2018] é outra ferramenta utilizada para capturar, coletar e analisar dados CSI. Ela permite a extração de informações detalhadas de comunicação sem fio das placas de rede Wi-Fi Atheros. Atheros-CSI-Tool é um *firmware* modificado em um *kernel* Linux de código aberto que utiliza interface de rede 802.11n e opera em várias distribuições Linux, por exemplo, Ubuntu, OpenWRT e Linino. Portanto, está disponível para computadores pessoais, dispositivos embarcados, como roteadores Wi-Fi e dispositivos de Internet das Coisas (do inglês, *Internet of Things* - IoT).

OpenFWWF CSI Tool (do inglês, *Open FirmWare for Wi-Fi networks*) [Gringoli and Nava 2009] é um projeto focado em fornecer uma plataforma completa e econômica para implementar novos protocolos de controle de acesso ao meio (do inglês, *Medium Access Control* - MAC), nos quais a realização de testes é simplificada. Ele também oferece *firmware* de código aberto para chips Broadcom e adota um método de engenharia reversa para obter dados CSI.

O Nexmon CSI Extractor [Schulz et al. 2018] foi proposto para permitir aos pesquisadores acesso ao processamento de quadros de camada inferior e funcionalidades avançadas na camada física. Nexmon é um *firmware* baseado na linguagem de programação C, desenvolvido para chips Wi-Fi *Broadcom/Cypress*, que se concentra principalmente em habilitar o modo de monitoramento para extração de dados CSI. Ele permite a extração de informações de canais de comunicação sem fio entre dois dispositivos Wi-Fi. O Nexmon CSI opera em dispositivos Wi-Fi MIMO com largura de banda de até 80 MHz, além de ser compatível com os padrões 802.11a/n/ac. Ele pode ainda ser usado em dispositivos móveis, como *smartphones*, e em dispositivos de baixo custo, como *Raspberries*.

GNU Radio é um *software* [Khan et al. 2017] de código aberto que permite o desenvolvimento gráfico de módulos de processamento de sinal. É utilizado sem *hardware* específico de forma simulada, ou com *hardware* de RF (Radiofrequência) externo como o USRP. O USRP é projetado para aplicações de RF a 6 GHz, incluindo sistemas MIMO. Exemplos de áreas de aplicação incluem telefonia celular, segurança pública, monitoramento de espectro, redes de rádio, rádio cognitivo, navegação por satélite e rádio amador. O monitoramento de espectro é como o rádio GNU e o USRP são usados para coletar, extrair e analisar dados CSI.

5.3. Processamento do Wi-Fi CSI para Monitoramento de Sinais Vitais

Após a coleta dos dados CSI, são necessários procedimentos para limpar do sinal os ruídos inerentes ao canal de comunicação sem fio e aplicar algoritmos que permitam a detecção de sinais vitais. Nesta seção são abordadas essas técnicas.

5.3.1. Processamento dos sinais de CSI

Seguindo a arquitetura apresentada na Figura 5.1, na etapa 2 é realizado um procedimento de pré-processamento do sinal capturado que contém os dados CSI. Com base em vários estudos encontrados na literatura [Liu et al. 2019, Wang et al. 2019, Ma et al. 2019b, Uchiyama et al. 2021], esta seção apresenta uma classificação e subclassificação das técnicas mais conhecidas utilizadas na etapa de pré-processamento do sinal recebido.

O sinal obtido através de uma ferramenta de extração, e.g. aquelas apresentadas

na Seção 5.2.2, usualmente está contaminado por ruídos e/ou *outliers*, que podem reduzir e prejudicar o desempenho da detecção. Com o objetivo de mitigar ou diminuir os efeitos causados pelo ruído e *outliers*, são aplicadas técnicas de **redução de ruído**. As técnicas de redução de ruído podem ser classificadas em dois grupos: compensação de fase e remoção de *outliers*. A compensação de fase é importante pois em sistemas de comunicação sem fio sempre há deslocamentos de sinal devido a falhas ocasionadas pelo *hardware* ou *software* e também devido a características dos canais multipercurso. Algumas técnicas de remoção de deslocamentos de fase comumente usadas são baseadas em compensação de tempo e/ou frequência de amostragem [Wang et al. 2017c, Wang et al. 2017b], compensação de frequência da portadora [Wang et al. 2017c, Wang et al. 2017b], compensação de erros de sincronização entre dispositivos [Wang et al. 2017c], atraso de detecção de pacotes, diferença de fase e ainda, regressão linear múltipla.

Enquanto isso, as técnicas de remoção de *outliers* servem para eliminar ruídos de alta frequência presentes no sinal recebido. Algumas técnicas neste grupo são baseadas em média móvel [Wang et al. 2016b, Wu et al. 2017, Li et al. 2021, Liu et al. 2018], filtro mediano [Ma et al. 2016, Liu et al. 2015a, Shang and Wu 2016, Yang et al. 2018], filtro passa-baixa, filtro *wavelet* [Liu et al. 2015b, Liu et al. 2014], filtro de Hampel [Ma et al. 2016, Wang et al. 2016b, Zhang et al. 2018, Zhang et al. 2019, Dou and Huan 2021, Wang et al. 2017c, Li et al. 2021, Liu et al. 2015a, Liu et al. 2018, Wang et al. 2017b, Liu et al. 2015b, Liu et al. 2014, Gu et al. 2021], filtro Savitzky-Golay [Dou and Huan 2021, Zeng et al. 2019], fator *Outlier* Local, anulação de sinal, entre outros.

Outro conjunto de técnicas que também podem ser aplicadas nesta etapa é a **transformação do sinal**, que é utilizada para analisar o sinal recebido no domínio da frequência. Algumas transformações que auxiliam na análise da frequência são: a transformada rápida de Fourrier (do Inglês, *Fast Fourier Transform* - FFT) [Wang et al. 2017a, Liu et al. 2015a, Lee et al. 2018, Wang et al. 2020, Wang et al. 2017b, Khan et al. 2017] amplamente utilizada para encontrar frequências dominantes, a transformada curta de Fourrier (do Inglês, *Short Time Fourier Transform* - STFT) [Dou and Huan 2021, Li et al. 2021, Liu et al. 2015b], que divide o sinal em segmentos iguais e calcula a FFT em cada segmento independente, a transformada discreta de Hilbert (do Inglês, *Discrete Hilbert Transform* - DHT) que incorpora o deslocamento de fase e é útil para encontrar mudanças instantâneas em um determinado instante de tempo dentro do sinal, e também a transformada discreta de *wavelet* (do Inglês, *Discrete Wavelet Transform* - DWT) [Wang et al. 2017a, Wang et al. 2020, Wang et al. 2017b, Liu et al. 2014], que fornece uma análise do sinal recebido em alta resolução, porque descompõe o sinal em conjuntos de tempos que fornecem a evolução temporal do sinal nas frequências correspondentes.

A **extração de sinal** é a última etapa do pré-processamento. Ele pode ser realizado filtrando e limitando as componentes de frequência do sinal, onde os filtros passa-alta [Zhang et al. 2019], passa-baixa [Shang and Wu 2016] e passa-banda [Zhang et al. 2018] são amplamente utilizados para extrair sinais com determinadas frequências dominantes. A compressão do sinal é importante para reduzir os sinais a poucas ou a uma única dimensão mas que ainda assim represente a enorme quantidade de sinais capturados. Para isso, algumas das técnicas utilizadas incluem Análise de Componentes Principais (do Inglês, *Principal Component Analysis* - PCA) [Li et al. 2021], Análise de Componentes Independentes (do Inglês, *Independent Component Analysis* - ICA) [Zeng et al. 2020, Zhang

et al. 2017], Decomposição de Valor Singular (do Inglês, *Singular Value Decomposition* - SVD) [Liu et al. 2014] , autocorrelação, correlação cruzada, distância euclidiana, função distribuição, entre outros. Finalmente, a Composição do Sinal é uma técnica usada para estimar ou detectar um fenômeno usando vários dispositivos ou características de banda de frequência.

5.3.2. Algoritmos de Detecção de Sinais Vitais

A terceira etapa da arquitetura apresentada na Figura 5.1 é o uso de algoritmos de detecção. Três grupos de algoritmos podem ser usados nesta etapa: aqueles baseados em modelagem, aqueles baseados em aprendizado e também os algoritmos híbridos. Os algoritmos **baseados em modelagem** são fundamentados pela teoria física, como modelos de análise de sinal. Para esta análise, usa-se o sinal obtido após a etapa de pré-processamento e examinam-se os efeitos causados nas informações CSI através de diversos fenômenos. Por exemplo, a atenuação de amplitude e a mudança de fase do sinal recebido podem ser afetadas pela distância entre o transmissor e o receptor, assim como os efeitos dos múltiplos percursos, incluindo reflexão de rádio, refração, difração, absorção, polarização e espalhamento [Ma et al. 2016, Wang et al. 2016b, Zhang et al. 2018, Zhang et al. 2019, Dou and Huan 2021, Wang et al. 2017c, Wu et al. 2017, Wang et al. 2017a, Zeng et al. 2018, Liu et al. 2015a, Shang and Wu 2016, Wang et al. 2017b, Khan et al. 2017, Yang et al. 2018, Gu et al. 2021]. Dentre os algoritmos baseados em modelagem, existem ainda os modelos estatísticos, baseados em medições empíricas, e os modelos probabilísticos, como o modelo usado para determinar o estado do canal sem fio, e.g. densidade espectral de potência, coerência tempo/frequência, auto correlação e correlação cruzada, etc. [Wang et al. 2016b, Chen et al. 2017a, Liu et al. 2015a, Liu et al. 2018, Wang et al. 2020, Wang et al. 2017b, Zhang et al. 2017]. Esses tipos de algoritmos são amplamente utilizados para estimar os sinais vitais humanos.

O segundo grupo de algoritmos utilizados é **baseado em aprendizado** (*learning-based*). Algoritmos baseados em aprendizado são usados principalmente para reconhecimento de gestos humanos, posição e detecção da presença de pessoas. O aprendizado é realizado usando o conjunto de dados de treinamento, processado previamente aos dados que se deseja analisar, onde o efeito do fenômeno a ser detectado é refletido no CSI. Alguns dos algoritmos mais usados são: *Naive Bayes, k Nearest Neighbor, Support Vector Machine, Convolutional/Recurrent Neural Network* e *Long Short-Term Memory* [Wang et al. 2017c, Lee et al. 2018, Zhang et al. 2017].

Além dos já mencionados, a fusão de algoritmos levou ao uso de algoritmos ditos **híbridos** (*hybrid algorithms*). Os algoritmos híbridos combinam os benefícios de algoritmos baseados em modelos com aqueles dos algoritmos baseados em aprendizado. Essa combinação pode ser benéfica para o desenvolvimento de uma detecção mais robusta e completa, dependendo da aplicação.

5.4. Aplicações de Wi-Fi CSI para Monitoramento da Saúde

A maioria das pesquisas baseadas em análise CSI mostra três direções claras: detecção, reconhecimento e estimativa. A detecção está relacionada à classificação binária como detecção de presença humana, movimento de queda, mudança de postura, fadiga ao di-

rigir, intrusão, tabagismo, entre outros. Já as pesquisas sobre reconhecimento são basicamente uma classificação multiclasse como reconhecimento de atividades que incluem atividades diárias, exercícios, fala, reconhecimento de gestos com o corpo, cabeça, braço, mão, perna, dedo e gestos, reconhecimento de linguagem de sinais, identificação, autenticação de usuário e reconhecimento de objetos. A terceira direção de pesquisa baseada em análise CSI é a estimativa. Essa categoria envolve quantidade, valores de tamanho, comprimento, ângulo, distância, duração, frequência, contagens, movimento, desenho à mão, velocidade, localização/rastreamento humano baseado em dispositivo, estimativa de umidade, respiração/frequência respiratória e estimativa de frequência cardíaca. Essas estimativas podem envolver uma única pessoa, ou várias pessoas, para aplicações de contagem humana, indicando se estão estáticos ou em movimento, entre outras.

Vários estudos relacionados a diferentes atividades humanas aplicam a análise de dados CSI. [Gu et al. 2018, Gu et al. 2017, Bowen et al. 2019a, Duan et al. 2018a, Damodaran et al. 2020]. Esses trabalhos podem servir como ponto de partida para desenvolver aplicativos de monitoramento de saúde, levando a um sistema robusto de detecção e monitoramento de sinais vitais. Em [Gu et al. 2018] por exemplo, os autores propuseram um sistema chamado EmoSense para detectar emoções humanas. O EmoSense analisa as impressões digitais de tempo e frequência nos dados do canal sem fio casadas pela expressão física das emoções. Em [Gu et al. 2017], os autores propuseram um sistema denominado MoSense para detectar os movimentos que são indicadores críticos da presença humana e das atividades humanas. Além disso, em [Bowen et al. 2019a], foi proposta uma tomografia radioelétrica para detectar pessoas e a passagem de água. Outra abordagem foi apresentada em [Duan et al. 2018a], onde os autores propuseram um sistema denominado WiDriver para monitorar as atividades de um motorista de veículo pesado. Foram detectadas atividades como movimentos do volante, utilização de celulares e escrita de mensagens de texto no celular. Mais recentemente, em [Damodaran et al. 2020], os autores abordaram várias atividades humanas como andar, sentar, ficar em pé e correr.

A análise dos dados CSI apresenta alto potencial para se tornar uma tecnologia poderosa para monitorar os aspectos físicos do ambiente em geral. Este capítulo concentra-se na utilização de dados CSI para monitorar os sinais vitais humanos. Esses sinais são classificados principalmente em (i) frequência respiratória, que é o número de ciclos de respiração que uma pessoa faz por minuto; (ii) frequência cardíaca, que é o número de vezes que o coração se contrai durante um determinado período de tempo, geralmente um minuto (bpm). Esses dois sinais vitais, contados em várias respirações ou batimentos, oferecem informações importantes para determinar o estado de saúde atual do indivíduo. Vale ressaltar que podemos encontrar na literatura diversos estudos para detectar, reconhecer e estimar esses sinais vitais. As técnicas baseadas em Wi-Fi CSI para monitoramento de sinais vitais apresentam-se bastantes atraentes em comparação com as demais devido às suas características como o baixo custo, facilidade de implantação e ausência de dispositivos em contato direto com o corpo do indivíduo.

Nesta seção serão descritos estudos que utilizam dados Wi-Fi CSI para monitorar sinais vitais. Para tanto, destacam-se os sinais vitais monitorados e os recursos relevantes do aplicativo, monitoramento em tempo real ou não e consideração de múltiplas pessoas ou cada indivíduo, conforme apresentado na Figura 5.2. A seguir, são descritas as aplicações de monitoramento de sinais vitais, e as técnicas utilizadas no processo são

apresentadas de acordo com a classificação proposta na Seção 5.2.

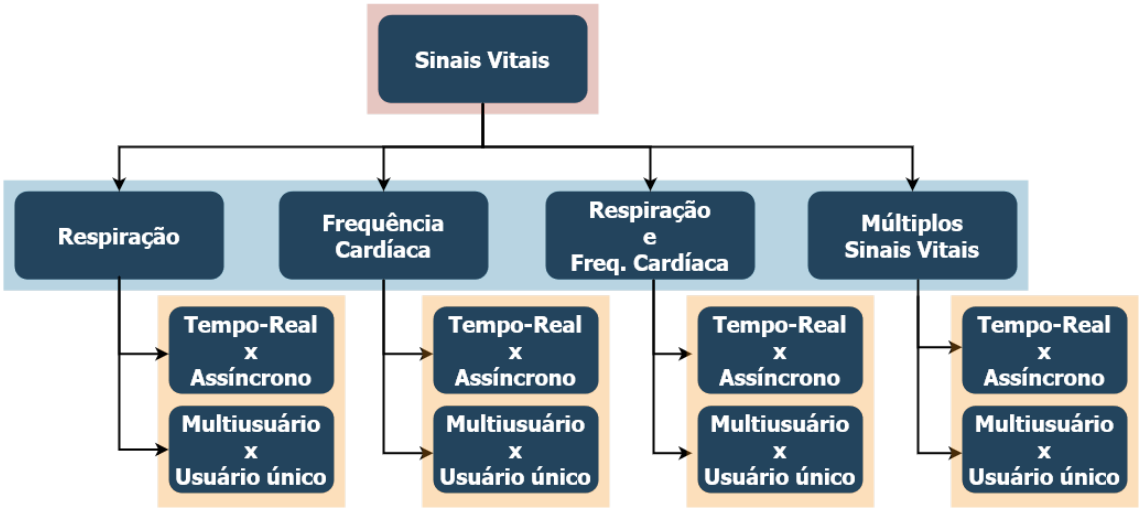


Figura 5.2. Diagrama de aplicações de sinais vitais

5.4.1. Monitoramento da Respiração

Muitos estudos recorrem aos dados CSI para monitorar os sinais vitais de um indivíduo. O WiSleep [Liu et al. 2014] foi o primeiro trabalho a detectar a taxa de respiração humana para monitoramento do sono com base em dados CSI utilizando dispositivos Wi-Fi comuns. A abordagem proposta, sem dispositivos físicos fixados ao indivíduo, tem potencial para ser amplamente implantada não somente em ambientes clínicos, como hospitais, mas também em ambientes não clínicos.

Desde então, diversos estudos foram desenvolvidos na tentativa de melhorar o monitoramento da frequência respiratória utilizando dados Wi-Fi CSI. Em [Wang et al. 2016b] por exemplo, os autores inicialmente introduziram a utilização da Zona de Fresnel (do Inglês, *Fresnel Zone* - FZ) no espaço livre, em seguida verificaram sua aplicabilidade em propagação de rádio Wi-Fi considerando ambiente interno. Eles desenvolveram uma teoria para relacionar a profundidade, localização e orientação da respiração de uma pessoa com a detectabilidade da respiração. Com a teoria desenvolvida, não só fica claro quando e por que a respiração humana é detectável utilizando dispositivos Wi-Fi, mas também esclareceu-se a compreensão do limite físico e a fundamentação dos sistemas de detecção baseados em Wi-Fi.

Seguindo a mesma linha de raciocínio, os autores de [Wu et al. 2017] compararam as abordagens baseadas em padrões e as baseadas em modelos para monitorar a taxa de respiração. Eles propuseram expandir o alcance de detecção do modelo Fresnel usado para as vastas regiões fora da primeira Zona de Fresnel. Eles mostraram a superioridade da detecção humana baseada no modelo da Zona de Fresnel em relação às abordagens baseadas em padrões e argumentaram que as abordagens baseadas no modelo de Zona de Fresnel têm grande potencial para alcançar escala centimétrica e até milimétrica na detecção de atividade humana, permitindo um amplo espectro de aplicações. Além disso, os autores [Ma et al. 2016] também usaram o modelo baseado em Zona de Fresnel e

mostraram como uma mudança de posição na escala de centímetros afeta o desempenho da detecção da respiração.

Também com foco no modelo de Zona de Fresnel, os autores de [Zhang et al. 2018] utilizaram o modelo de difração de Fresnel para quantificar com precisão a relação entre o ganho de difração e o sutil deslocamento do tórax do alvo humano, e assim transformar com sucesso a difração de obstrução destrutiva anteriormente considerada na Primeira Zona Fresnel (do Inglês, *First Fresnel Zone* - FFZ), em capacidade de detecção benéfica. Eles foram capazes de apresentar o mapa de calor detalhado da capacidade de detecção em cada local dentro da FFZ, para orientar o sensor de respiração para que os usuários saibam claramente onde estão as posições adequadas para o monitoramento da respiração, assim como identificar se estão em uma posição desfavorável.

Outro sistema, denominado *BreathTrack*, foi proposto por [Zhang et al. 2019] para rastrear o estado da respiração humana utilizando sinais CSI Wi-Fi. Nele os autores propuseram métodos de correção via *hardware* e *software* para remover as distorções de fase invariantes e variantes no tempo e.g. deslocamento de frequência da portadora (do Inglês, *Carrier Frequency Offset* - CFO), deslocamento de frequência de amostragem (do Inglês, *Sampling Frequency Offset* - SFO), atraso de detecção de pacotes (do Inglês, *Packet Detection Delay* - PDD) e *PLL Phase Offset* (PPO), e assim obter dados CSI precisos. Eles também propuseram um método de recuperação esparsa conjunta: ângulo de chegada e tempo de voo (AoA-ToF, do inglês Angle of Arrival - Time of Flight) para obter o coeficiente de atenuação complexo correspondente, eliminar o efeito de multipercurso no ambiente interno e também extrair a informação do caminho dominante para rastrear o status da respiração. Além disso, em relação à fase dos sinais Wi-Fi, os autores [Zeng et al. 2018] descobriram que sua amplitude e fase são perfeitamente complementares entre si. Eles detalharam o modelo matemático e exploraram a natureza complementar para projetar e implementar um sistema de detecção de respiração em tempo real com dispositivos Wi-Fi comuns. Eles também utilizaram o modelo de Zona de Fresnel. Além disso, os autores de [Wang et al. 2017c] utilizaram a diferença de fase dos dados CSI para estimar de forma inteligente as taxas de respiração para várias pessoas com dispositivos Wi-Fi comuns. Inicialmente, os dados de diferença de fase dos dados CSI entre pares de antenas no receptor Wi-Fi foram usados para criar tensores CSI. Em seguida, foi aplicada a decomposição poliádica canônica [Sorber et al. 2013] (do Inglês, *Canonical Polyadic Decomposition* - CPD) para obter os sinais respiratórios desejados.

Em outra abordagem, os autores de [Wang et al. 2017a] exploraram a detecção da respiração de várias pessoas simultaneamente. Para mitigar o efeito causado por outras pessoas, foi colocado um receptor ao lado de cada usuário, em seguida foram selecionados os dados cujo tempo de chegada (do inglês, *Time of Arrival* - ToA) era maior que um limite de truncamento pré-definido. Também com foco no contexto de várias pessoas em um mesmo ambiente, em [Chen et al. 2017a], os autores introduziram o TR-BREATH, um sistema de monitoramento de respiração baseado em reversão de tempo (do inglês, *Time-Reversal* - TR). Esse sistema é capaz de detectar a respiração e estimar a taxa de respiração de várias pessoas em um curto período de tempo. O TR-BREATH projeta o sinal no espaço de recursos de força ressonante TR (do inglês, *TR Resonating Strength* - TRRS) e analisa o TRRS usando o *Root-MUSIC* e algoritmos de propagação de afinidade para ampliar as variações de CSI. Se a respiração for detectada, o TR-BREATH estima

as taxas de respiração de várias pessoas por meio de propagação de afinidade, atribuição de probabilidade e mesclagem de *cluster*. Além disso, é possível estimar o número de pessoas presentes no ambiente com um erro de aproximadamente 1 ao comparar com o conhecimento prévio do número real de pessoas.

O sistema MultiSense [Zeng et al. 2020] foi desenvolvido para detectar de forma robusta e contínua os padrões detalhados de respiração de várias pessoas, mesmo que tenham taxas de respiração muito semelhantes e estejam fisicamente próximas. Atualmente, os *hardwares* de Wi-Fi mais utilizados geralmente são equipados com várias antenas. Assim, cada antena individual pode receber uma cópia de misturas de diferente sinais refletidos de várias pessoas. Os autores provaram com sucesso que os sinais refletidos são misturados linearmente em cada antena e propuseram modelar o sensor de respiração de várias pessoas como um problema de separação de fontes cegas (do inglês, *Blind Source Separation* - BSS). Eles resolveram o problema utilizando o ICA para separar o sinal misto e obter as informações de reparação de cada pessoa.

Em [Dou and Huan 2021], os autores usaram ainda a variação da energia espectral Doppler extraída do CSI coletado por dispositivos Wi-Fi para rastrear o deslocamento do tórax induzido pela respiração. O sinal de respiração é extraído da mudança de energia espectral acumulada do deslocamento Doppler na frequência zero de acordo com a periodicidade da ação da respiração.

A proposta FarSense, apresentada em [Zeng et al. 2019], propôs o primeiro sistema que trabalha em tempo real e que pode monitorar a respiração humana de forma confiável quando o alvo está longe (dentro do limite de 8 metros) do par de transceptores Wi-Fi, preenchendo a lacuna entre protótipo de laboratório e implantação na vida real. Os autores propuseram um método chamado CSI-ratio que combina a amplitude e a fase de duas antenas adjacentes para uma melhor estimativa da respiração.

Mais recentemente, o sistema chamado Wi-COVID foi introduzido em [Li et al. 2021]. Essa é uma tecnologia não invasiva e livre de dispositivos fixados ao indivíduo para monitorar indivíduos e rastrear a taxa de respiração para o profissional de saúde. Os autores exploraram a possibilidade de usar a tecnologia baseada em Wi-Fi para monitorar em tempo real pacientes diagnosticados com COVID-19 que estão realizando autoisolamento. Eles propuseram o uso de um *Raspberry* Pi que funciona como uma espécie de ponto de acesso. Na parte do *software*, eles usaram códigos abertos para implementar o processamento de dados CSI em um *Raspberry* Pi.

Em [Li et al. 2021], os autores usaram o Nexmon para extrair dados CSI do quadro Wi-Fi modulado com OFDM 802.11n por quadro com largura de banda de até 80 MHz no chip Broadcom Wi-Fi de um *Raspberry* Pi. Assim, sua implementação foi mais simples do que outras propostas, pois eles precisavam apenas de um roteador Wi-Fi pronto para uso e um *Raspberry* Pi.

A Tabela 5.2 apresenta um resumo e comparação dos estudos citados sobre monitoramento da respiração usando análise de dados Wi-Fi CSI juntamente com suas próprias características. Apresentamos as ferramentas de processamento de sinal utilizadas, como filtragem de ruído (do inglês, *noise filtering* - NF), transformada de sinal (do inglês, *Transformação do sinal* - ST) e ferramentas de extração de sinal (do inglês, *Signal Extraction* -

SE). Detalhes de quais algoritmos usam quais técnicas de processamento de sinal e para quais aplicativos de detecção de Wi-Fi eles são usados também são discutidos. Também é indicado se a operação é realizada em tempo real ou não.

Na Tabela 5.2, pode-se observar que a maioria dos estudos utiliza a ferramenta de extração Linux 802.11n CSI Tool. Esse fato se deve à natureza da maior parte dos dispositivos Wi-Fi que utilizam Linux. Vale ressaltar que uma nova proposta foi aplicada recentemente aos sistemas de monitoramento de saúde Wi-Fi CSI usando Nexmon. Essa é uma tecnologia promissora, pois oferece uso simples em dispositivos como *smartphones* e *Raspberries*.

Outro ponto importante a ser observado é a implementação em tempo real. Alguns estudos têm sido desenvolvidos com o objetivo de desenvolver o monitoramento em tempo real, o que os torna mais adequados para trabalhar em ambientes reais. Por outro lado, várias das propostas encontradas têm alta precisão, especialmente quando apenas um indivíduo é considerada no ambiente de teste. A precisão diminui com o aumento do número de indivíduos. Além disso, as configurações do ambiente e o posicionamento do indivíduo influenciam na precisão da proposta, oferecendo os melhores resultados quando os pacientes estão em boa posição, ou seja, na Primeira Zona de Fresnel.

5.4.2. Monitoramento da Frequência Cardíaca

O monitoramento da frequência cardíaca é outra tarefa relevante no acompanhamento dos sinais vitais para monitoramento da condição de saúde do indivíduo. Vários estudos monitoram a frequência cardíaca e a respiração simultaneamente, no entanto, foi encontrada apenas uma proposta na literatura para monitorar apenas a frequência cardíaca.

O sistema CardioFi, proposto em [Khamis et al. 2018], monitora a frequência cardíaca via *hardware* Wi-Fi com antenas omnidirecionais. O principal desafio observado foi o considerável nível ruído de radiofrequência que afeta as transmissões Wi-Fi em ambientes do mundo real. O CardioFi utiliza um esquema chamado *Dynamic-Window* para identificar um comportamento anômalo no sinal e descartar os sinais que não representam maior sensibilidade. Assim, eles obtiveram frequências altamente sensíveis. O CardioFi foi testado em cenários fora da linha de visão (do inglês, *Non-Line-Of-Sight* - NLOS) com uma baixa relação sinal-ruído (do inglês, *Signal-to-Noise Ratio* - SNR) e melhorou seu erro percentual. Essa solução considera o monitoramento de uma pessoa, que pode ser realizado em tempo real. No entanto, é apresentada apenas a arquitetura utilizada para a estimativa da frequência cardíaca, necessitando ser estendida para possibilitar a utilização em aplicações médicas de tempo real.

5.4.3. Monitoramento da Respiração e da Frequência Cardíaca

Também podemos encontrar na literatura trabalhos mais ambiciosos, que se concentraram não apenas no monitoramento da respiração, ou no monitoramento da frequência cardíaca, mas em ambos ao mesmo tempo. Por exemplo, em [Liu et al. 2015a], Liu et al. propuseram rastrear a respiração e a frequência cardíaca durante o sono utilizando dispositivos Wi-Fi prontos para uso. O algoritmo desenvolvido faz uso das informações do canal no domínio do tempo e da frequência para estimar a respiração e os batimentos cardíacos simultaneamente, e funciona bem quando um indivíduo ou duas pessoas estão

Tabela 5.2. Monitoramento da respiração utilizando sinais Wi-Fi CSI

Ref.	Ferramenta de extração	Pre-processamento	Algoritmos de detecção	Múltiplas pessoas	Tempo real	Desempenho
[Ma et al. 2016]	Linux 802.11n CSI Tool	Redução de ruído	Baseado em modelos	Não	Não	N/D
[Wang et al. 2016b]	Linux 802.11n CSI Tool	Redução de ruído	Baseado em modelos	Não	Não	N/D
[Zhang et al. 2018]	Linux 802.11n CSI Tool	Redução de ruído Extração do sinal	Baseado em modelos	Não	Não	Indivíduos bem posicionados 98.8%, mal posicionados 61.5%
[Zhang et al. 2019]	Linux 802.11n CSI Tool	Redução de ruído Extração do sinal	Baseado em modelos	Não	Não	Mais de 99%
[Dou and Huan 2021]	Linux 802.11n CSI Tool	Redução de ruído Transformação do sinal	Baseado em modelos	Não	Não	Erro máximo < 0.7 bpm, média de erro $\approx$ 0.15 bpm
[Zeng et al. 2020]	Linux 802.11n CSI Tool	Extração do sinal	Baseado em modelos	Sim	Não	Taxa de erros de 0.73 bpm (respirações por minuto)
[Wang et al. 2017c]	Linux 802.11n CSI Tool	Redução de ruído	Hibrido	Sim	Não	Erro de estimação: 1 pessoa 96% menos que 0.5bpm, 2 e 3 pessoas 93% menos que 0.5 bpm, e 5 pessoas 62% menos que 0.5bpm
[Wu et al. 2017]	Linux 802.11n CSI Tool	Redução de ruído	Baseado em modelos	Não	N/D	Erro médio: 0.09 bpm, 0.15 bpm, 0.06 bpm para três regiões detectáveis diferentes
[Wang et al. 2017a]	Linux 802.11n CSI Tool	Transformação do sinal	Baseado em modelos	Sim	Sim	$>$ 95% (1 pessoa) $>$ 88% (2 pessoas)
[Chen et al. 2017a]	Linux 802.11n CSI Tool	Transformação do sinal	Baseado em modelos	Não	Sim	Desempenho médio: uma pessoa NLOS 99%, 12 pessoas LOS 98.65%, 9 pessoas NLOS 98.07%
[Zeng et al. 2018]	Linux 802.11n CSI Tool	Redução de ruído	Baseado em modelos	Não	Sim	100% in LOS
[Zeng et al. 2019]	Linux 802.11n CSI Tool	Redução de ruído	Baseado em modelos	Não	Sim	A taxa geral de detecção 100%; Média absoluta de erros menor que 0.3bpm para taxa de respiração
[Li et al. 2021]	Nexmon CSI Extractor	Redução de ruído Extração do sinal	Baseado em modelos	Não	Sim	N/D

na cama.

Também em [Liu et al. 2018], Liu et al. reutilizaram a rede Wi-Fi existente para rastrear a respiração e os batimentos cardíacos simultaneamente durante o sono. Os resultados mostraram que o sistema proposto fornece uma estimativa precisa da frequência respiratória e da frequência cardíaca não apenas em configurações típicas, mas também abrangendo cenários desafiadores, incluindo a longa distância entre o dispositivo Wi-Fi e o ponto de acesso (do inglês, *Access Point* - AP), situações sem linha de visão direta (NLOS) e diferentes posturas de sono.

Também seguindo a linha de pesquisa qu considera o monitoramento de vários sinais vitais, o PhaseBeat [Wang et al. 2017b] aproveita as diferenças de fase do sinal coletado entre duas antenas de recepção em dispositivos Wi-Fi para detectar e monitorar a frequência respiratória e cardíaca em tempo real. Os autores descobriram que os dados de diferença de fase são bastante estáveis após calibração adequada. Eles também provaram que, para ambientes com múltiplos percursos internos sob desvanecimento de pequena escala, os dados de diferença de fase representam um sinal periódico com a mesma frequência que o sinal de respiração quando o sinal sem fio é refletido no peito de uma pessoa. Eles também mostraram que o PhaseBeat é altamente robusto para estimativa da taxa de respiração em vários ambientes, como diferentes distâncias entre o transmissor e o receptor.

Em [Shang and Wu 2016], os autores projetaram um sistema de reconhecimento de respiração e batimentos cardíacos baseado em sinal Wi-Fi denominado Wi-Health. O sistema proposto tem a capacidade de determinar se um ser humano está vivo ou não, e o número de batimentos cardíacos e respirações por unidade de tempo. Eles também propuseram eliminar atividades humanas semelhantes, que têm frequência semelhante à respiração (0Hz - 1Hz), como acenar as mãos. A eliminação da frequência dessas atividades de forma eficaz se faz necessária, caso contrário, elas podem introduzir picos extras no espectro e ocasionar em estimativas imprecisas de respiração e batimentos cardíacos.

Além disso, em [Lee et al. 2018], os autores propuseram um método para reconhecer e distinguir as mudanças no padrão de respiração e frequência cardíaca de uma pessoa utilizando o sinal Wi-Fi CSI. A amplitude das ondas do sinal pode representar os movimentos periódicos do tórax para cima e para baixo causados pela respiração e batimentos cardíacos, e mudanças proeminentes no padrão do sinal podem ser detectadas usando o algoritmo DTW (do inglês, *Dynamic Time Warping*). Os autores mostraram que esse método pode identificar o estado físico da pessoa. Além disso, avaliaram a eficácia do método proposto através de vários experimentos com 10 participantes.

Mais recentemente, outro sistema que detecta respiração e frequência cardíaca, denominado PhaseBeat, foi apresentado em [Wang et al. 2020]. Foi realizada uma análise rigorosa da diferença de fase da informação do estado do canal em relação à sua estabilidade e periodicidade. Os autores mostraram que, para ambientes internos com múltiplos percursos e sob desvanecimento em pequena escala, os dados de diferença de fase do sinal coletado são periódicos e têm a mesma frequência que o sinal de respiração quando o sinal Wi-Fi é refletido no peito de uma pessoa. Além disso, a diferença de fase do CSI também é mais robusta do que o RSSI em vários cenários de implantação, como diferentes distâncias, obstáculos/paredes e orientações. Nessa proposta, a subportadora

mais sensível foi selecionada e processada por DTW para obtenção da informação sobre a respiração e dos batimentos cardíacos reconstruídos. Finalmente, eles aplicaram FFT para medir as frequências respiratória e cardíaca.

O resumo das aplicações desenvolvidas para monitoramento da respiração e dos batimentos cardíacos baseados em sinais Wi-Fi é apresentado na Tabela 5.3. Como pode-se notar, quando o foco está na aplicação dos sinais vitais monitorados, a maior parte das pesquisas desconsidera a operação em tempo real. Pode-se observar ainda que em geral a acurácia das propostas é maior e o erro médio é menor para a detecção da frequência respiratória em relação à detecção da frequência cardíaca. Além disso, a maioria desses estudos considera apenas o monitoramento de apenas um indivíduo de cada vez.

5.4.4. Monitoramento de Múltiplos Sinais Vitais

Além da frequência respiratória e dos batimentos cardíacos, alguns estudos propuseram novas modalidades de sensoriamento das atividades humanas, como mudança de posição, micromovimentos, tremores, detecção de quedas, entre outros. A proposta do WiSleep [Liu et al. 2014], por exemplo, teve como foco extrair do CSI padrões rítmicos associados à respiração e mudanças bruscas devido ao movimento do corpo. A proposta do WiSleep foi estendida em [Liu et al. 2015b] para identificar de forma confiável a taxa de respiração na presença de ruído. A respiração é considerada periódica, no entanto, uma consequência indesejável é que informações como respiração anormal (por exemplo, apneia do sono) que violam a suposição periódica não são facilmente identificadas. Comparado com os trabalhos existentes, o sistema proposto em [Liu et al. 2015b] pode rastrear a respiração anormal (por exemplo, apneia do sono) e também pode fornecer informações sobre a respiração quando a pessoa está em diferentes posições de sono.

O sistema WiCare [Zhang et al. 2017] é outro exemplo de trabalho que utiliza sinais Wi-Fi CSI para monitorar diferentes sinais vitais em paralelo: frequência respiratória com a coexistência de alguns micromovimentos (por exemplo, ler, escrever, usar o telefone). Mais especificamente, o WiCare é capaz de distinguir os micromovimentos de um indivíduo específico de sua respiração. Esta abordagem baseia-se no fato de que a respiração resulta em flutuações da informação CSI com banda de frequência mais estreita em comparação com os micromovimentos.

Em [Khan et al. 2017], os autores apresentaram um sistema de extração de fase bidimensional usando sensor Wi-Fi passivo para monitorar três atividades básicas de cuidados com idosos, incluindo frequência respiratória, tremor essencial e quedas. Toda a implementação foi realizada usando rádios definidos por *software*. Os autores também usaram técnicas de processamento de sinais para analisar a função de ambiguidade cruzada e identificar variações de fase em dois planos separados.

Os autores de [Gu et al. 2021] propuseram um método para aprimorar a capacidade de detecção de movimento baseado na teoria Rice-K e na teoria de Fresnel. Movimentos como o giro afetam a precisão do monitoramento dos sinais vitais, assim, eles também propuseram um algoritmo de detecção da alteração da posição baseado na detecção de regularidade para distinguir rapidamente esses movimentos.

A dificuldade de monitorar a respiração do sono de várias pessoas geralmente

Tabela 5.3. Monitoramento da frequência respiratória e da frequência cardíaca usando sinais Wi-Fi CSI

Ref.	Ferramenta de extração	Pre-processamento	Algoritmos de detecção	Múltiplas pessoas	Tempo real	Desempenho
[Liu et al. 2015a]	Linux 802.11n CSI Tool	Redução de ruído Transformação do sinal	Hibrido	Não	Não	Erro da taxa de respiração: $<$ 1.1bpm (1 pessoa), $<$ 1.2bpm (2 pessoas); Erro da taxa cardíaca: $<$ 5bpm (1 pessoa)
[Lee et al. 2018]	Linux 802.11n CSI Tool	Transformação do sinal Extração do sinal	Baseado em modelos	Não	Não	Taxa de respiração 94% frequência cardíaca 82%
[Liu et al. 2018]	Linux 802.11n CSI Tool	Redução de ruído	Baseado em modelos	Não	Não	Erros de estimação 80% $<$ 0.5rpm para a taxa de respiração, 90% $<$ 4bpm para a taxa cardíaca
[Wang et al. 2020]	Linux 802.11n CSI Tool	Transformação do sinal	Baseado em modelos	Não	Não	Erro médio: 0.25 respirações por minuto (rpm)para taxa de respiração; 1.19 bpm para taxa cardíaca
[Shang and Wu 2016]	Linux 802.11n CSI Tool	Redução de ruído	Baseado em modelos	Não	N/D	Erro médio de estimativa abaixo de: 0.6rpm para taxa de respiração; 6bpm para taxa cardíaca
[Wang et al. 2017b]	Linux 802.11n CSI Tool	Redução de ruído Transformação do sinal	Baseado em modelos	Não	Sim	Erro de estimação: $<$ 0.85rpm para taxa de respiração, $<$ 10bpm para taxa cardíaca

vem da necessidade de separar os efeitos da respiração de cada uma das pessoas. Outro problema é que, embora a separação possa ser viável com alguns algoritmos, ainda é complexo mapear os múltiplos estados respiratórios identificados para as pessoas correspondentes. Para resolver este problema, os autores de [Yang et al. 2018] propuseram uma abordagem através da implantação de transceptores Wi-Fi. Um transceptor Wi-Fi cuidadosamente colocado pode ser afetado apenas pela pessoa em um determinado local. Além disso, eles consideraram a movimentação do indivíduo durante o sono, bem como a mudança de postura, isso para melhorar a robustez do sistema. Assim, empregaram a diferença entre o valor máximo e mínimo e a variação da amplitude do pico extraída do domínio da frequência entre os fluxos CSI de todas as subportadoras para detectar apneia.

A Tabela 5.4 resume e compara os estudos citados sobre o monitoramento de diferentes sinais vitais usando análise Wi-Fi CSI. São apresentadas também algumas de suas características como ferramentas de processamento e extração de sinais, modelo utilizado e desempenho. Também é indicada se a operação é realizada em tempo real ou não. Conforme mostrado na Tabela 5.4, a maioria dos estudos focados no monitoramento de diversos sinais vitais considera também uma implementação em tempo real e algoritmos de detecção baseados em modelos. Nesses, a precisão da detecção da frequência respiratória é ligeiramente inferior quando comparada às propostas de detecção de apenas uma pessoa mencionadas nas seções anteriores.

Tabela 5.4. Monitoramento de múltiplos sinais vitais usando Wi-Fi CSI

Ref.	Ferramenta de extração	Pre-processamento	Algoritmos de detecção	Múltiplas pessoas	Tempo real	Sinal Extra	Desempenho
[Liu et al. 2015b]	Linux 802.11n CSI Tool	Redução do ruído Transformação do sinal	Baseado em modelos	Não	Não	Postura	Estimação da taxa de respiração: maior que 85%; Estimação da apneia: 82.1%, Mudança de posição: maior que 80%
[Zhang et al. 2017]	Linux 802.11n CSI Tool	Redução de ruído Extração do sinal	Baseado em modelos	Não	Não	Micromovimentos	Erros de estimação $<$ 2bpm para 80% dos esperimentos
[Liu et al. 2014]	Linux 802.11n CSI Tool	Redução de ruído Extração do sinal	Baseado em modelos	Não	Sim	Postura	Estimação da taxa de respiração: 85%; Mudança de postura $\approx$ 80%
[Khan et al. 2017]	USRP B200	Transformação do sinal	Baseado em modelos	Não	Sim	Movimentos	Precisão: respiração 87%, detecção de quedas 98%, classificação de tremor 93%
[Yang et al. 2018]	Linux 802.11n CSI Tool	Redução de ruído	Baseado em modelos	Sim	Sim	Postura	Média absoluta de erro: Respiração 0.614bpm no meio da ZF, 3.130bpm nas bordas; Falso alarme de apneia 6.8%, apneia perdida 7.09%
[Gu et al. 2021]	Linux 802.11n CSI Tool	Redução do ruído	Baseado em modelos	Não	Sim	Postura	96.618% para taxa de respiração e 94.708% para batimentos cardíacos

5.5. Atividade Prática

Esta seção apresenta uma atividade prática. São mostrados os passos necessários para o processo de captura e processamento de dados CSI. Inicialmente, é apresentada uma configuração do ambiente de captura, destacando os requisitos básicos de *software* e *hardware*, além do cenário de coleta de dados. Posteriormente, é mostrada a extração da informação CSI a partir dos sinais coletados e o seu processamento, utilizando técnicas de filtragem e algoritmos conhecidos, como o PCA (do inglês, *Principal Component Analysis*) e a Transformada Rápida de Fourier – FFT (do inglês, *Fast Fourier Transform*). Finalmente, são apresentados dois casos que consideram a estimativa da taxa de respiração e do batimento cardíaco de um indivíduo a partir dos dados CSI capturados utilizando dispositivos Wi-Fi.

5.5.1. Configuração do Ambiente de Captura

Para a captura e coleta do sinal utilizou-se o *firmware* NEXMON. O *firmware* é instalado em um dispositivo Raspberry Pi modelo 4B. Para a instalação utilizou-se o Raspbian, um Sistema Operacional (SO) compatível com o *firmaware* NEXMON. Para a instalação do SO no Raspberry Pi, são necessários os equipamentos listados na Tabela 5.5.

Tabela 5.5. Dispositivos utilizados pelo NEXMON

Dispositivo	Descrição
Raspberry Pi 4 Modelo B	Hardware compatível com o Raspbian e NEXMON.
Cartão Micro SD	Memória para o SO
Leitor de cartão Micro SD	Para a instalação do SO
Cabo HDMI	Para conectar o monitor ao Raspberry.
Monitor	Tela para visualizar o Raspberry.
Teclado USB	Para trabalhar no Raspberry.
Cabo Ethernet	Para configurar o roteador sem fio.
Roteador TP-Link Archer C6	Para criar e configurar a rede Wi-Fi.
PC	Para instalar e transferir arquivos ao Raspbian.
Mouse	Para trabalhar com o Raspberry.

Primeiramente, deve-se introduzir o cartão SD no leitor de cartão Micro SD para conectá-lo ao slot USB do computador. Em seguida, deve-se instalar a imagem do Raspbian `2021-01-11-raspios-buster-armhf-full` no cartão SD. Essa imagem contém o Kernel 5.4, compatível com o *firmware* NEXMON, que será instalado. Para instalar a imagem do Raspbian no cartão SD utilizamos uma ferramenta chamada Etcher. Assim que a imagem do Raspbian estiver instalada no cartão SD, deve-se retirar o cartão do PC do leitor de cartão Micro SD, e inseri-lo no Raspberry Pi 4B.

O Raspberry Pi 4B deve estar conectado ao monitor através de um cabo HDMI. Além disso, conectamos um teclado e mouse às interfaces USB do Raspberry para controlá-lo. Finalizadas as conexões e inserção do cartão SD, o Raspberry Pi 4B pode ser ligado e a inicialização do Raspbian será realizada. Ao inicializar o SO, utiliza-se o usuário e senha padrão do Raspian para efetuar o login (user:pi, pass:raspberry). Posteriormente,

deve-se proceder com as configurações básicas do dispositivo, como as interfaces e habilitar os serviços que serão utilizados, como o idioma, a interface sem fio WLAN e o acesso remoto via SSH (*Secure SHell*). Para acessar essas configurações, executa-se o seguinte comando no terminal:

```
~$ sudo raspi-config
```

O comando abre uma janela com as opções de configuração. Na opção *Interfacing Options*, habilite a conexão SSH para se conectar futuramente ao Raspberry a partir do PC. Caso queira utilizar uma interface gráfica na conexão remota, habilite o servidor VNC. Configure o *Time Zone*, WLAN *Country* e teclado. Recomenda-se utilizar a opção Estados Unidos da América (USA) como país na configuração, para que habilite o uso de todos os canais Wi-Fi. Após as configurações anteriores, é necessário verificar a versão do kernel instalado, através do comando:

```
~$ uname -a
```

O resultado dessa consulta deve mostrar a versão do kernel conforme o exemplo a seguir:

```
Linux raspberrypi 5.4.83-v7l+ #1379 SMP Mon Dec 14
13:11:54 GMT 2020 armv7l GNU/Linux
```

Observando-se o resultado da consulta `~$ uname -a`, o kernel instalado é o `5.4.83-v7l+`, o qual é compatível com a versão do *firmware* NEXMON. Deve-se considerar que, durante todo o trabalho com o Raspbian e NEXMON, não se pode fazer um *upgrade* do SO, porque esse procedimento alteraria a versão do kernel e geraria uma incompatibilidade com a versão do *firmware* NEXMON.

Além das configurações já realizadas, é necessário ter acesso à conexão Internet para a instalação do NEXMON. Essa conexão pode ser feita utilizando a interface WLAN ou Ethernet. A configuração de rede também pode ser realizada no menu do comando `|sudo` raspi-config|. Deve-se alertar que após a instalação do NEXMON, o acesso à Internet só poderá ser feito pela interface Ethernet, pois a interface WLAN é modificada e fica exclusivamente para o uso do *firmware* NEXMON. Portanto, recomenda-se utilizar a interface Ethernet para a conexão à Internet desde o início.

Instalação do firmware NEXMON

O NEXMON é um firmware que modifica o kernel e a interface sem fio do Raspberry para poder capturar e coletar dados CSI. A instalação é realizada executando os comandos diretamente no Raspberry Pi 4B mediante o teclado, ou a partir do PC com conexão remota, utilizando o SSH ou VNC. A instalação segue diversos passos que são descritos a seguir:

1. **Instalar dependências**
   Para instalar as dependências, são executados os seguintes comandos:

   ```
   ~$ sudo apt update
   ```

   Logo que executar esta linha de comando, aparecerá na tela a pergunta: *Do you want to accept these changes and continue updating from this repository? [y/n]*, Respondemos "*y*"para que a atualização ocorra de acordo à versão do kernel instalada e as dependências possam ser instaladas corretamente. Logo instalam-se as seguintes dependências:

   ```
   ~$ sudo apt install libgmp3-dev gawk
   qpdf bc bison flex libssl-dev make
   automake texinfo libtool-bin tmux
   libncurses5-dev git tcpdump

   ~$ sudo reboot
   ```

   Deve-se verificar a versão do kernel após reiniciar o SO, utilizando o comando "**uname -a**". A versão deverá ser a mesma verificada inicialmente.

   ```
   Linux raspberrypi 5.4.83-v7l+  $#1379 SMP
   Mon Dec 14 13:11:54 GMT 2020 armv7l GNU/Linux
   ```

   Alerta-se que **NÃO DEVE-SE** executar o comando *"sudo apt upgrade"*, para não alterar a versão do kernel. Apenas kernels até a versão 5.4 são compatíveis com o NEXMON.

2. **Obter Kernel Headers**
   Ao utilizar o kernel *Linux raspberrypi 5.4.83-v7l+*, há ainda um problema de compatibilidade entre os headers disponíveis no repositório *apt*, que estão disponíveis apenas para versões mais atualizadas. Assim, deve-se sincronizar os headers para a versão correta do kernel. Para isso, utiliza-se o *rpi-source project*, executando os seguintes comandos:

   ```
   ~$ sudo wget https://raw.githubusercontent.
   com/RPi-Distro/rpi-source/master/rpi-source
   -O /usr/local/bin/rpi-source && sudo chmod
   +x /usr/local/bin/rpi-source && /usr/local/
   bin/rpi-source -q --tag-update

   ~$ rpi-source

   ~$ sudo reboot
   ```

3. **Instalar NEXMON e NEXMON-CSI**

   Finalmente, o firmware NEXMON e seu complemento NEXMON-CSI podem ser instalados. O firmware NEXMON modifica o kernel com múltiplos propósitos, um deles é a captura e coleta de dados CSI, através do complemento NEXMON-CSI. Para a instalação, deve-se executar os seguintes comandos:

```
~$ sudo su

~$ wget https://raw.githubusercontent.com/
zeroby0/nexmon_csi/pi-5.4.51/install.sh
-O install.sh

~$ tmux new -c /home/pi -s nexmon 'bash
install.sh | tee output.log'
```

Configuração do Cenário de Teste

Para realizar a coleta de dados CSI, é necessário configurar uma rede Wi-Fi, formada por um roteador e um dispositivo Wi-Fi cliente, conforme apresentado na Figura 5.3. Nessa rede, é gerado um tráfego entre o cliente e o roteador utilizando o utilitário "ping". O Raspberry, com o NEXMON instalado, mantém a interface sem fio em modo de escuta, capturando os quadros do meio de transmissão sem fio. Capturam-se os quadros da comunicação entre o roteador e o cliente Wi-Fi. O Raspberry coleta os dados de requisição (echo-request) e de resposta (echo-reply), ou apenas um deles. No exemplo a seguir, a captura é feita apenas nos quadros que têm como origem o cliente Wi-Fi, que envia o ping para o roteador.

A rede Wi-Fi formada para a captura e coleta de dados CSI deve trabalhar nas faixas frequência ISM de 2.4 GHz ou 5 GHz. Recomendamos utilizar a faixa de frequências de 5 GHz, pois possui menos interferências quando comparada a versão de 2.4 GHz e permite a utilização de canais de maior largura de banda. Ao utilizar a faixa de 5GHz, configuramos uma rede com largura de banda que vai de 20 MHz a 160 MHz. Já na faixa de 2.4 GHz, apenas canais com largura de banda de 20 MHz e 40 MHz são permitidos. No exemplo de extração de dados da Seção 5.5.2, utilizou-se uma rede com o roteador configurado para trabalhar na faixa de frequência de 5GHz, utilizando o canal 36 e largura de banda de 80 MHz. Por outro lado, a configuração do dispositivo cliente Wi-Fi basicamente consiste em se conectar à rede do roteador, indicando o SSID e senha da rede Wi-Fi configurada no roteador.

Para gerar o tráfego, executa-se o comando *ping* no cliente com destino ao roteador. Assim, gera-se um fluxo pela interface sem fio do laptop para o roteador. Um detalhe importante é a configuração do intervalo entre pacotes do *ping*, que deve ser configurado de acordo com a taxa de amostragem necessária para a estimação, detecção ou reconhecimento de atividade humana que é desejado. Neste exemplo, deseja-se obter os sinais de batimento cardíaco, que possuem uma frequência má-

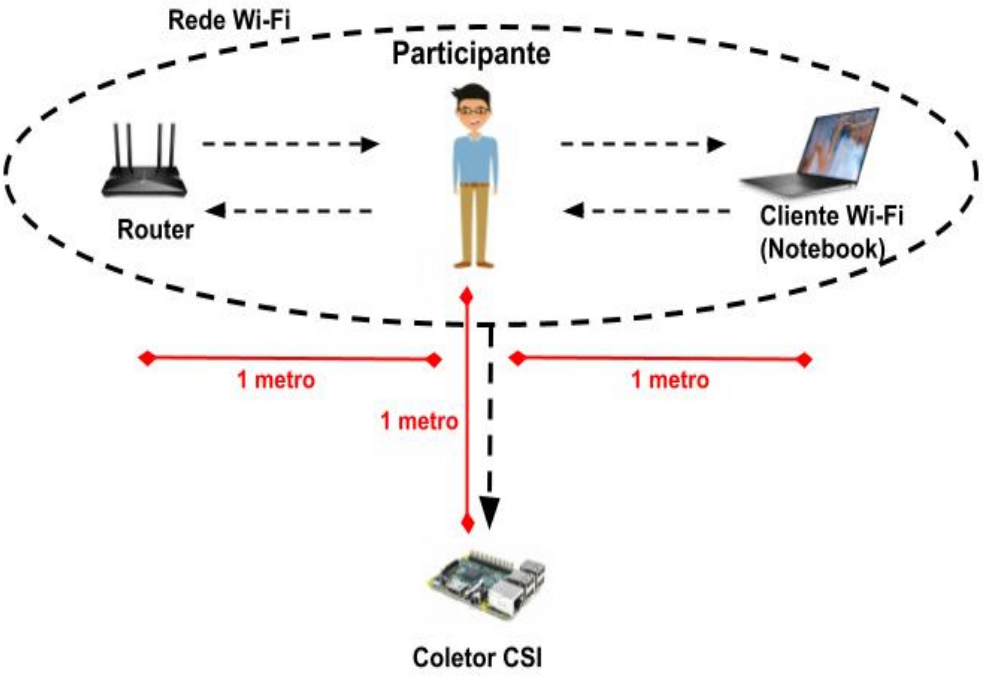


Figura 5.3. Cenário de teste

xima de 220 batimentos por minuto[1], ou seja, 3,67 Hz. De acordo com o teorema de Nyquist, a taxa de amostragem mínima deve ser o dobro da frequência máxima do sinal amostrado, ou seja, 7,34 Hz. Portanto, o intervalo máximo entre pacotes configurado para o ping deve ser de 0,13 seg.

O comando `|ping|` a seguir foi parametrizado para ser executado pela interface sem fio wlan0 (-I wlan0), no modo *flooding* (-f) para que seja exibido cada envio e resposta, com intervalo entre pacotes de 0.13 s (-i 0.13), enviados para um roteador configurado com o IP 192.168.1.1:

```
~$ sudo ping -I wlan0 -f -i 0.13 192.168.1.1
```

Finalmente, como mostrado na Figura 5.3, a distância entre o participante e os equipamentos é de 1 metro, formando assim o cenário de captura. O participante permanece no meio dos equipamentos a fim de que os sinais eletromagnéticos possam se propagar através ou ao redor dele e essas interferências no sinal sejam capturadas pelo Raspberry através do NEXMON.

5.5.2. Extração de Dados de Wi-Fi CSI Utilizando o NEXMON

Após instalar o NEXMON-CSI e configurar o cenário de teste, inicia-se a captura e coleta de dados CSI para posterior extração dos sinais vitais. Esta seção é subdividia, portanto, no passo-a-passo da captura e coleta e em seguida a extração de dados CSI a partir dos arquivos coletados.

[1] https://www.cdc.gov/physicalactivity/basics/measuring/heartrate.htm

Captura e Coleta de Dados CSI com NEXMON

Para capturar dados, pode-se executar o NEXMON no Raspberry de duas formas: executando comando a comando no terminal, ou através de um script. Neste exemplo,o utilizou-se a forma iterativa, para fins de demonstração. Portanto, os seguintes comandos devem ser executados no terminal:

```
~$ sudo su
~$ mcp -c 36/80 -C 1 -N 1
```

O comando mcp (*makecsiparams*) configura o canal 36 e largura de banda 80 MHz com um *core* e uma antena. Esta linha de comando retorna uma informação própria do NEXMON, similar à mostrada abaixo, que deve ser copiada pois será utilizada nos comandos seguintes:

```
m+IBEQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA==
```

Apos da executção dos comando anteriores, iniciamos a configuração da interface utilizada:

```
~$ sudo ifconfig wlan0 up

~$ nexutil -Iwlan0 -s500 -b -l34
 -vm+IBEQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA==
```

O comando *ifconfig* habilita a interface *wlan*. O comando *nexutil* configura a interface. Perceba que a informação copiada anteriormente deve ser colocada ao final da linha. Logo, a interface *wlan*0 é colocada em modo *monitor*. Para obter os dados coletados em um arquivo *.pcap* é preciso utilizar o comando *tcpdump* que escuta os pacotes pela porta 5500 (porta padrão de saída do NEXMON). Esses passos são mostrados a seguir:

```
~$ sudo iw dev wlan0 interface add mon0 type monitor

~$ sudo ip link set mon0 up

~$ sudo tcpdump -i wlan0 dst port 5500 -vv
   -w output.pcap -c 500
```

Extração de Dados CSI

Para extração de dados CSI a partir dos arquivos coletados *.pcap*, utilizam-se scripts desenvolvidos em python por um grupo de pesquisadores. Para maior detalhe pode ser encontrado no repositório do GitHub[2]. Esses scripts configuram a extração a partir do dispositivo indicado, extraem os dados dos arquivos *.pcap* utilizando uma contagem de bit no payload capturado. Para começar a extração, deve-se configurar o dispositivo de captura e coleta de dados CSI. Neste exemplo, utilizaou-se o chipset relativo ao Rapberry Pi 4B, o bcm43455c0. Seleciona-se essa opção de chipset no arquivo de configuração *config.py*:

```
# chip = 'bcm4339'    # Nexus 5
chip = 'bcm43455c0' # Raspberry Pi 3B+ and 4B
# chip = 'bcm4358'    # Nexus 6P
# chip = 'bcm4366c0'  # Asus RT-AC86U
```

Em seguida, remove-se do conjunto de dados as informações relativas às subportadores nulas e piloto do OFDM[3], denominadas "NULL" e "PILOT". Assim recuperam-se apenas informações relativas às subportadores utilizadas na transmissão de dados. Logo, obtêm-se os dados CSI capturados, além disso, pode-se obter outros dados que são de controle. Segundo a Tabela 5.6, os dados coletados têm diferente tamanho em bytes e representam diversas informações.

Tabela 5.6. Payload dos pacotes CSI coletados

Bytes	Tipo	Nome	Descrição
4	uint32	Magic Bytes	0x11111111
6	uint8[6]	Source Mac	Endereço MAC de origem do quadro Wi-Fi
2	uint16	Sequence Number	Número de sequência do quadro Wi-Fi
2	uint16	Core and Spatial Stream	3 bits indicam o Core e os próximos 3 bits o Spatial Stream number
2	uint16	Chanspec	Especificação do Canal usado na extração. Veja nexutil -k
2	uint16	Chip Version	Versão do Chipset
Variável	int16[]	CSI Data	Cada amostra CSI tem 4 bytes com intervalos de uma parte real e outra imaginária

Na extração de dados é importante esclarecer que os dados CSI são números complexos. Esses números complexos contêm uma parte real e outra parte imaginária. A manipulação desses números complexos é importante porque a partir desse processo podemos obter a amplitude, fase ou resultados mais complexos como detecção, reconhecimento ou detecção de alguma atividade humana. A seguir, é mostrado um exemplo de obtenção de amplitude e fase. Esses dados são obtidos a partir

[2]https://github.com/nexmonster/nexmon_csi/tree/feature/python/utils/python
[3]https://www.oreilly.com/library/view/80211ac-a-survival/9781449357702/ch02.html

dos dados CSI extraídos em forma de números complexos. Foi aplicada a função *np.abd*() para obter a amplitude e *np.angle*() para obter fase do sinal coletado. Nas funções aplicadas, foi passada como referência a matriz *csi* que contém os dados coletados em forma de uma matriz bidimensional.

```
self.ax_amp.plot(self.x_amp, np.abs(csi))
self.ax_pha.plot(self.x_pha,
        np.angle(csi, deg=True))
```

5.5.3. Pré-processamento de Sinais CSI e Obtenção de Sinais Vitais com Python

Após a extração dos dados CSI dos arquivos coletados, inicia-se a fase de processamento desses dados.

Para a extração dos sinais vitais, realiza-se uma sequência de procedimentos que serão categorizados como processamento dos dados CSI e algoritmos de detecção dos sinais vitais.

A etapa de processamentos dos dados CSI é realizada a partir da extração dos dados, como mostrado na Seção 5.5.2. Esses dados passam por diversos tipos de filtros, para remoção de ruído ou *outliers* (valores atípicos) que possam ter sido capturados.

Os dados CSI são representados por uma matriz, e suas dimensões são referentes à quantidade de subportadoras, que dependem da largura de banda utilizada e da quantidade de amostras. A matriz contém todos os dados CSI coletados válidos, sem considerar as subportadoras PILOTs e NULLs, mas contendo uma grande quantidade de ruído próprio do meio do transmissão.

A Figura 5.4(a) apresenta um exemplo de dados coletados, ainda com o ruido e outliers. Para remover esses valores indesejados, podemos utilizar uma série de filtros. Na aplicação desta atividade, o filtro Hampel é utilizado para este fim. O objetivo do filtro Hampel é identificar e substituir valores atípicos em uma série temporal. Esse filtro analisa uma janela deslizante para percorrer os dados e calcula a mediana e o desvio padrão nessa janela. Há duas variáveis que podem ser configuradas, o tamanho da janela deslizante (*window_size*) e a quantidade de desvios padrão (*n*) a partir da qual se classifica um valor como atípico. Os valores atípicos são substituídos pela mediana da janela. O trecho abaixo mostra como é aplicado o filtro Hampel no código de exemplo:

```
hampel(series[key], window_size=31,
    n=3, imputation=True)
```

Após a aplicação do filtro Hampel, ainda podem restar ruídos e *outliers*, que podem interferir na estimação dos sinais vitais. Para garantir que os dados sejam mais confiáveis, aplica-se outro filtro. O filtro aplicado é o filtro de média, ou *Moving Average*. O *Moving Average* é utilizado para suavizar variáveis aleatórias (outliers), em termos de reduzir as amplitudes fora do intervalo normal do sinal coletado. Este filtro se calcula obtendo a média dos dados do sinal em um período de tempo ou janela (*windows*) com

uma quantidade miníma de observações ($min_periods$). Neste exemplo, é aplicado como segue:

```
series[key].rolling(window=10, min_periods=1,
    center=True).mean()
```

Após o processamento dos dados CSI inicia-se a estimação de sinais vitais. Para os sinais vitais de interesse, frequência cardíaca e frequência respiratória, aplica-se um filtro passa-banda, para limitar o espectro de frequências em que estão localizados esses sinais vitais. Nesse estudo de caso, o filtro passa-banda deve ser configurado com uma frequência de corte entre 0.2 Hz e 0.4 Hz, para estimação da respiração. O movimento de inalar e exalar faz uma movimentação da caixa peitoral, que gera uma perturbação nos sinais electromagnéticos capturadas pelos dados CSI. Essas pertubações são observadas nas frequências acima mencionadas [Khamis et al. 2018, Zhang et al. 2019]. No caso da frequência cardíaca, as perturbações ocasionadas pelo batimento do coração podem ser sobrepostas pelo movimento do peito ao respirar. Para observar esses batimentos do coração, configura-se o filtro passa-banda com as frequências de corte 0.6 Hz e 3.5 Hz [Khamis et al. 2018, Wang et al. 2020]. A seguir é mostrada essa parte da configuração no código:

```
fs = 7.64 %taxa de amostragem
lowcut = 0.2 %0.6
highcut = 0.4 %3.5
butter(5, [lowcut / (fs / 2), highcut / (fs / 2)],
    'band', analog=False, output='ba')
```

Com o filtro passa-banda, recupera-se um conjunto de sinais que estão entre as frequências recomendadas para estimar os sinais vitais requeridos. Entretanto, é necessário estabelecer um único sinal que represente esse grande conjunto de sinais CSI. Na Figura 5.4(b) estão os resultados após o uso dos filtros Hampel e Moving Average. Além disso, a Figura 5.4(c) mostra os dados CSI processados pelo filtro passa-banda, para obter só as frequências de interesse. Existe diversas formas de obter sinais representativos, uma delas segundo a literatura é aplicar o PCA (do inglês, *Principal Component Analysis*) [Li et al. 2021]. O PCA faz com que esse conjunto de sinais seja representado por um conjunto mínimo de componentes. No exemplo, limita-se a representação dos sinais de todas as subportadoras a um único componente, que represente o total de sinais CSI. A seguir, apresenta-se o trecho de código da aplicação do PCA, enquanto a Figura 5.4(c) mostra o resultado da aplicação do PCA.

```
PCA(n_components=1)
pca.fit_transform(series)
```

A partir do resultado do PCA como uma só componente, utilizamos a FFT (do inglês, *Fast Fourier Transform*). A FFT é utilizada para a conversão do sinal que está no domínio do tempo para uma representação no domínio de frequência. A FFT resulta em um

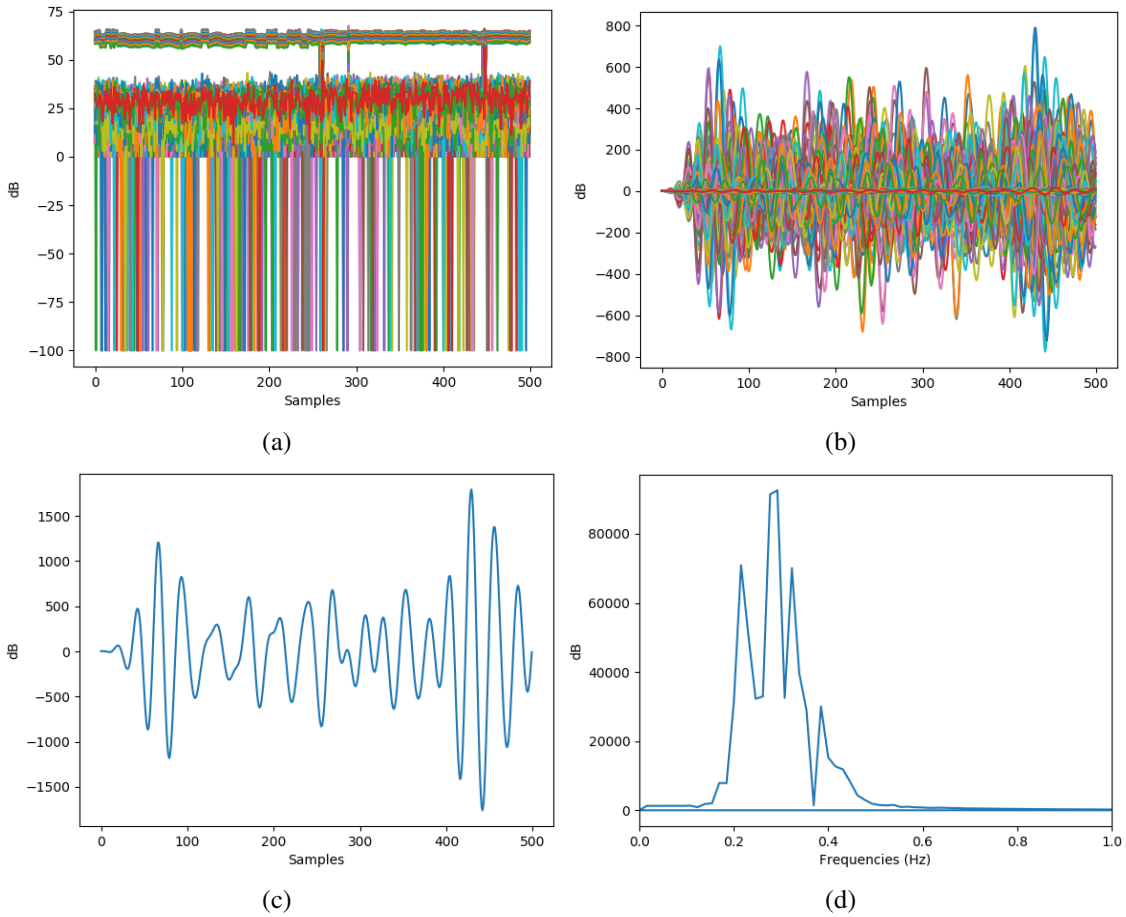


Figura 5.4. (a) Dados CSI coletados sem processamento. (b) Dados CSI pre-processados pelos filtros Hampel e Moving Average. (c) Sinal CSI obtido pelo PCA, que representa o conjunto de sinais coletados. (d) Exemplo de frequências obtidas para a estimação da frequência respiratória com o uso de FFT.

conjunto discreto de valores de frequências, que estão dentro das faixas de interesse selecionadas no filtro passa-banda. Para a frequência respiratória, a faixa de interesse é entre 0.2 Hz e 0.4 Hz, e estima-se a frequência respiratória como sendo a frequência de maior amplitude dentro dessa faixa do espectro e ela representa a taxa de rpm (respirações por minuto). Por exemplo, os resultados estimados para um adulto normal são mostrados em uma faixa entre 15 e 20 rpm. No caso da estimação do batimento cardíaco, as frequências são limitadas entre 0.6 Hz e 3.5 Hz com o filtro passa-banda. Nesse intervalo, a frequência cardíaca é estimada a partir de uma média entre as quatro frequências com maior amplitude. Os resultados da frequência cardíaca para uma pessoa normal em repouso são exibidos em uma faixa de 60 a 100 bpm (batimentos por minuto). A Figura 5.4(d) mostra um exemplo das frequências consideradas para o cômputo da frequência respiratória (sinal vital), que estão de 0.2 Hz e 0.4 Hz.

5.6. Desafios e Tendências Futuras

O monitoramento de sinais vitais e atividade humana usando Wi-Fi CSI pode oferecer suporte a aplicações de saúde. No entanto, os desafios devem ser superados para que se

possa obter a implementação prática. No que se referir à detecção de sinais vitais, a definição de cenários para captura de dados CSI não é trivial. Os movimentos e as atividades diárias têm maior impacto no CSI quando comparados à respiração e à frequência cardíaca. Portanto, a detecção de sinais vitais impõe restrições na recepção e processamento do sinal. Além disso, desafios como interferência eletromagnética, reconhecimento de movimentos simultâneos de diferentes partes do corpo, ambientes com muitas pessoas e cenários sem linha de visão direta (NLOS) devem ser enfrentados. Esta seção aborda esses desafios e também as perspectivas de aplicações futuras e melhorias do uso de Wi-Fi CSI para monitoramento de sinais vitais.

5.6.1. Desafios

O primeiro estágio da arquitetura CSI é a captura de dados. Nessa etapa, as antenas colocadas no ambiente para transmissão e recepção de sinais Wi-Fi podem variar em quantidade e formato. Existem cenários definidos para usar uma única antena transmissora e uma única antena receptora (do inglês, *Single-Input Single-Output* - SISO) em uma configuração ponto a ponto. Outra configuração corresponde à tecnologia MIMO, com várias antenas utilizadas para transmitir e receber os dados CSI. A configuração MIMO é necessariamente regida pelo padrão 802.11n/ac. Em ambas as configurações, o ponto de convergência é a quantidade de informações CSI capturadas. Quanto maior o número de antenas ou dispositivos envolvidos, maior a quantidade de dados.

Especificamente, o número de dispositivos necessários para obter as informações necessárias e aumentar a precisão do reconhecimento do comportamento é uma questão que precisa ser estudada. Estudos como [Zhu et al. 2017] e [Li et al. 2018] confirmam que aumentar o número de dispositivos Wi-Fi pode melhorar o desempenho do sistema. No entanto, aumentar o número de dispositivos também aumenta a interferência eletromagnética, e esse é um dos principais desafios na identificação de sinais vitais.

Aumentar o número de dispositivos Wi-Fi permite ampliar a área de cobertura e melhorar o desempenho de modelos baseados em zonas de Fresnel, por exemplo. No entanto, as zonas de Fresnel de vários links são complexas [Chen et al. 2017b]. Então, detectar vários limites de zona de Fresnel é um grande desafio. É difícil identificar as melhores localizações do transceptor e orientações corretas [Wang et al. 2017a]. A configuração do cenário é uma etapa crítica para a coleta eficiente de dados CSI que permite uma detecção mais precisa dos sinais vitais. Além disso, o número de dispositivos depende exclusivamente da quantidade de informação capaz de se analisar e se a ferramenta utilizada suporta esta captura. As ferramentas de captura de dados CSI mencionadas na Seção 5.2.2 diferem de acordo com as características do cenário escolhido.

Uma discussão fundamental para o desempenho de captura CSI é a linha de visão (*Line-Of-Sight* - LOS) de antenas e dispositivos. Em um cenário em que as antenas e dispositivos possuem uma linha de visão entre eles e o paciente, os dados capturados são mais robustos e descrevem de forma completa os sinais vitais a serem detectados. No entanto, em um cenário onde os dispositivos e antenas não possuem linha de visão (NLOS) entre eles e nem com o paciente, os resultados da captura de dados podem ser insuficientes. Isso torna a detecção de sinais vitais em cenários NLOS um desafio importante. Essas consequências ocorrem devido à interferência em um cenário LOS ser menor do

que em um cenário NLOS.

Em um cenário com linha de visão, a interferência detectada deve-se principalmente ao paciente, no qual os sinais interferem (refletem, espalham, atenuam, etc.), podendo-se obter o CSI para posteriormente inferir os sinais vitais. Por outro lado, em um cenário sem linha de visão, não apenas o paciente gera interferência, mas também outros objetos, por exemplo, paredes de concreto, que podem distorcer o sinal e as medições CSI. Consequentemente, as informações capturadas não geram a confiabilidade adequada para seu tratamento posterior e inferência de sinais vitais.

Uma vez que os dados CSI foram capturados, a próxima etapa é o pré-processamento do sinal. Este estágio se concentra em limpar o sinal o máximo possível do ruído externo, Gaussiano, branco, térmico ou qualquer outro ruído inerente à comunicação sem fio, que são acoplados ao sinal com a informação transmitida. Vários tipos de filtros são aplicados na tentativa de limpar o sinal, conforme descrito na Seção 5.3.1. Para uma análise mais aprofundada do sinal, filtros devem ser aplicados para obter um sinal resultante o mais confiável possível, assim, a limpeza do sinal torna-se um desafio. É necessário que o sinal limpo defina as formas de onda mais confiáveis, ou seja, quando a frequência respiratória é detectada, o sinal é analisado inicialmente em um intervalo de $0,2$Hz a $0,4$Hz, e a variação da onda, causada pelo movimento do tórax do paciente inspirando ou expirando, pode ser observado.

Após a detecção da frequência respiratória, é realizada a detecção da frequência cardíaca, que devido ao seu processo inerente de batimentos cardíacos é representada em menor intensidade em relação à respiração. Essa detecção cardíaca pode ser afetada pela frequência respiratória, pois o efeito da respiração no CSI é consideravelmente mais forte quando comparado ao efeito do batimento cardíaco. No monitoramento da respiração, os requisitos para filtragem de sinal podem ser menos complexos e exigentes em comparação com o monitoramento cardíaco. A detecção cardíaca requer que o sinal resultante exiba os picos de onda em detalhes enquanto conserva as distorções de ondas pequenas (picos) causadas pelo batimento cardíaco. Por isso é importante usar a técnica de filtragem mais adequada para limpar o sinal sem perder o efeito de pulsação no CSI capturado.

A próxima etapa é a detecção de sinais vitais usando vários algoritmos mostrados na Seção 5.3.2. O que encontramos nos estudos apresentados na literatura é que existem dois aspectos na detecção de sinais vitais ou atividades humanas em geral. Quando falamos em detectar uma pessoa, reconhecer gestos ou estimar movimentos, os estudos propostos induzem ao uso de algoritmos de inteligência artificial, especificamente aprendizado de máquina. No entanto, quando nos referimos à detecção de sinais vitais, as técnicas envolvidas são baseadas em modelos teóricos. Isso porque, conforme descrito anteriormente, quando detectamos sinais vitais como a frequência cardíaca, é necessário que as formas de onda (picos) sejam detalhadas da forma mais clara possível e, para isso, as técnicas teóricas de modelagem de sinais apresentam bom desempenho. No entanto, a falta de estudos que usem algoritmos de aprendizado de máquina para detectar sinais vitais nos leva a pensar em quão bem eles podem atuar nesse tipo de detecção. Partindo do ponto de que, por exemplo, para detecção de frequência cardíaca, os picos produzidos pelos batimentos que afetam o sinal capturado representam padrões de distorção, então esses padrões poderiam ser identificados por um algoritmo de aprendizado de máquina

como uma recorrência ao longo do monitoramento, em que o real valor da frequência cardíaca pode ser determinado. Em resumo, nesta fase o uso de técnicas teóricas de modelagem de sinais e algoritmos de aprendizado de máquina para a detecção de sinais vitais varia de acordo com o objetivo final da detecção. Da mesma forma, os algoritmos e/ou técnicas utilizadas têm um comportamento diferente dependendo do sinal vital detectado, o que desencadeia a necessidade de padronização das técnicas ou algoritmos de detecção.

A detecção de sinais vitais resulta em uma tarefa desafiadora, principalmente devido aos movimentos simultâneos de diferentes partes do corpo e/ou a presença de várias pessoas no mesmo ambiente. Comumente, as pessoas realizam vários movimentos simultaneamente e o CSI coletado contém o efeito misto resultante de todos esses movimentos. Distinguir a mudança de sinal causada por cada movimento do corpo é a base para garantir o melhor desempenho na detecção de sinais vitais. No entanto, a detecção simultânea de movimentos corporais é um desafio a ser enfrentado. Por outro lado, alguns estudos como TR-BREATH [Chen et al. 2016, Wang et al. 2017a] mostraram que o aumento no número de usuários geralmente diminui a precisão do reconhecimento. Também é importante identificar a presença de várias pessoas no mesmo ambiente e, eventualmente, poder diferenciar o efeito de cada pessoa no CSI. Dessa forma, seria possível monitorar mais de uma pessoa ao mesmo tempo. Além disso, evitando a interferência de outras pessoas nos sinais vitais do paciente que está sendo monitorado. No entanto, reconhecimento de comportamento multiusuário torna-se um desafio à medida que o número de pessoas aumenta.

Finalmente, o último estágio de aplicação leva à detecção, estimativa ou reconhecimento de alguma atividade humana. No campo da saúde, é necessário que a detecção seja tomada como principal aplicação. Esta aplicação leva à detecção de sinais vitais de pessoas/pacientes. Uma das perspectivas é que a detecção dos sinais vitais deve ser realizada de forma contínua, em tempo real. Essa avaliação contínua é feita com o objetivo de fornecer informações médicas atualizadas sobre o paciente à equipe médica e tomar melhores decisões em benefício do paciente. Os estudos na literatura carecem dessa adaptação da detecção contínua em tempo real e não fornecem uma abordagem aplicada à detecção de sinais vitais. Além disso, novos desafios como segurança, confiabilidade e portabilidade das informações de detecção surgem e podem gerar complicações na detecção contínua de sinais vitais. Esses desafios surgem em todo o processo de detecção e no cenário sem fio inerente, onde a detecção ocorre. Assim, a segurança pode ser afetada na captura de dados, uma vez que é realizada em um ambiente sem fio. Ele pode estar sujeito a ataques maliciosos que podem distorcer os dados CSI capturados. Na área de confiabilidade, as informações obtidas pelo médico assistente podem não ser confiáveis ou mesmo não contíguas. Considerando a portabilidade, há a necessidade de equipamentos de monitoramento (antenas, dispositivos) serem tão portáteis quanto possível para acoplamento a diferentes tipos de aplicações. Da mesma forma, as aplicações finais devem ser acopladas às informações fornecidas por esses dispositivos de monitoramento. Portanto, os problemas e desafios aumentam à medida que novos elementos são adicionados à detecção de sinais vitais. É importante superar esses desafios e propor tecnologias que trabalhem em conjunto com a detecção de sinais vitais e retornem um sistema médico robusto que utilize a tecnologia CSI, mas sem afetar sua finalidade inicial, que é a detecção de sinais vitais para aplicações médicas.

5.6.2. Tendências Futuras

O monitoramento em tempo real permite reduzir o risco de vida. Quando um paciente enfrenta uma situação de risco de vida, ele pode ser identificado imediatamente e um sinal de alerta pode ser acionado instantaneamente. Assim, o paciente pode receber a assistência médica necessária. Muitos eventos como: Reconhecimento de Atividade Humana (do inglês, *Human Activity Recognition* - HAR), detecção de queda, detecção de respiração e batimentos cardíacos e até diagnóstico de doenças, são de interesse para aplicações de saúde digital. Um desses eventos ou uma combinação de alguns deles pode indicar a necessidade de ajuda médica que uma pessoa pode ter. O reconhecimento de doenças usando CSI é o menos abordado na literatura e é um campo de pesquisa aberto interessante [Liu et al. 2015b, Yang et al. 2018].

Para extração de recursos e classificação de atividades, as técnicas de aprendizado de máquina (do inglês, *Machine Learning* - ML) se tornaram uma alternativa promissora. Na literatura, entre os principais algoritmos de ML utilizados para a detecção e reconhecimento da atividade humana e sinais vitais estão: LSTM [Damodaran et al. 2020, Bowen et al. 2019b, Shi et al. 2018], SVM [Damodaran et al. 2020, Li et al. 2019, Zhu et al. 2017], k-NN [Gu et al. 2018] e *Backpropagation Neural Network* (BPNN) [Duan et al. 2018b, Wu et al. 2018, Xiao et al. 2017]. Esses estudos demonstram a capacidade dos algoritmos de ML para detecção e reconhecimento da atividade humana e sinais vitais e abrem caminho para novas pesquisas focadas na detecção de doenças e na melhoria das aplicações de saúde digital.

Na literatura, estudos mostram que o algoritmo LSTM é capaz de extrair características dos dados de entrada automaticamente sem a necessidade de utilizar técnicas complexas para o tratamento de sinais Wi-Fi CSI. Portanto, o LSTM é considerado um bom candidato para a tarefa de classificação em aplicações de saúde digital em tempo real. Por isso, é uma opção interessante usar o LSTM para realizar uma análise profunda da respiração e do sinal de batimento cardíaco extraídos dos sinais CSI. Como o componente de fase do sinal é sensível ao movimento do tórax causado pela respiração [Tan et al. 2018], considera-se que algoritmos de classificação podem ser usados para estimar a atividade respiratória humana e eventualmente inferir ou detectar problemas respiratórios, como apneia do sono.

Uma combinação de detecção de atividade humana e taxa de respiração, semelhante ao BodyScan [Fang et al. 2016], é uma abordagem interessante. No BodyScan, como o CSI é capturado por dois dispositivos vestíveis projetados, ele não se encaixa na classificação de detecção de atividade sem dispositivo. No entanto, essa proposta apresenta bons resultados na determinação da taxa de respiração do usuário quando o movimento não é detectado e o corpo está em estado estacionário.

Identificar o efeito de atividades humanas comuns no CSI pode melhorar o desempenho da extração de sinais vitais. Além disso, conhecer a atividade imediatamente antes de uma anomalia no sinal respiratório e no batimento cardíaco pode ajudar na classificação e identificação de problemas de saúde ou mesmo de uma emergência médica. Por exemplo, quando há uma anormalidade no sinal respiratório ou nos batimentos cardíacos após uma queda ou após uma longa caminhada. Essas anomalias de sinal podem estar associadas a deficiência pulmonar, taquicardia, bradicardia ou arritmia. Desta forma, as

técnicas híbridas são uma opção interessante para melhorar o desempenho na detecção de sinais vitais.

Na literatura, alguns estudos [Wang et al. 2016a, Gu et al. 2018] mostram como o CSI é sensível a pequenos movimentos do corpo e que é possível detectar e analisar reflexões de rádio de granularidade fina a partir de movimentos faciais. O fato de emoções e expressões comuns de pessoas poderem ser reconhecidas a partir de CSI [Gu et al. 2018] abre a possibilidade de também identificar expressões associadas à dor e assim identificar quando uma pessoa está sofrendo alguma dor. Outra possibilidade interessante resulta em inferir o risco de uma pessoa sofrer depressão quando um padrão de tristeza se torna repetitivo. Além disso, como é possível ler os movimentos da boca e saber do que uma pessoa está falando a partir do CSI, como mostrado em [Wang et al. 2016a], palavras ou sequências de palavras usadas pelas pessoas para solicitar ajuda podem ser identificadas. Esse aplicativo pode ser usado para identificar o pedido de socorro de uma pessoa com parada cardiorrespiratória, em que a pessoa não consegue emitir o som da fala.

5.7. Conclusão

Este capítulo forneceu um levantamento sobre a detecção de sinais vitais usando informações de estado do canal sem fio através de dispositivos Wi-Fi comuns. Foi apresentada uma visão geral das informações de estado do canal, discutida brevemente a arquitetura geral de um sistema baseado em CSI, e seu modelo matemático. Estudos encontrados na literatura que tratam da detecção de sinais vitais humanos como respiração, frequência cardíaca, ou ambos, utilizando os dados CSI foram apresentados de acordo com as ferramentas utilizadas para extração de dados, técnicas empregadas no processamento dos sinais e algoritmos de detecção de sinais vitais. Assim, uma classificação e descrição foi apresentada, levando em conta também recursos como monitoramento em tempo real ou multiusuário e desempenho nos cenários propostos. Esses estudos de Wi-Fi CSI para monitoramento de sinais vitais mostram um desempenho promissor em vários cenários de aplicação.

Além disso, também foram apontadas as limitações das atuais abordagens de monitoramento de sinais vitais baseadas em Wi-Fi CSI e discutidos alguns desafios ainda a serem superados para que a análise de CSI possa ser totalmente usada na prática para monitorar sinais vitais e apoiar uma variedade de aplicações de saúde. Apresentando algumas tendências futuras, diversos temas de pesquisa foram elencados motivando mais trabalhos na área.

Agradecimentos

Este trabalho teve o apoio de CNPq, CAPES, CAPES Print, FAPERJ, FAPESP, INCT-MACC.

Referências

[Bowen et al. 2019a] Bowen, L., Hulbert, R., Fong, J., Rentz, Z., and Debruhl, B. (2019a). Democratized radio tomography: Using consumer equipment to see through walls. *IEEE Vehicular Technology Conference*, pages 1–6.

[Bowen et al. 2019b] Bowen, L., Hulbert, R., Fong, J., Rentz, Z., and DeBruhl, B. (2019b). Democratized radio tomography: Using consumer equipment to see through walls. In *IEEE Vehicular Technology Conference*, pages 1–6. IEEE.

[Chen et al. 2017a] Chen, C., Han, Y., Chen, Y., Lai, H.-Q., Zhang, F., Wang, B., and Liu, K. R. (2017a). TR-BREATH: Time-reversal breathing rate estimation and detection. *IEEE Transactions on Biomedical Engineering*, 65(3):489–501.

[Chen et al. 2016] Chen, C., Han, Y., Chen, Y., and Liu, K. R. (2016). Multi-person breathing rate estimation using time-reversal on wifi platforms. In *IEEE Global Conference on Signal and Information Processing*, pages 1059–1063. IEEE.

[Chen et al. 2017b] Chen, L., Chen, X., Ni, L., Peng, Y., and Fang, D. (2017b). Human behavior recognition using wi-fi csi: Challenges and opportunities. *IEEE Communications Magazine*, 55(10):112–117.

[Damodaran et al. 2020] Damodaran, N., Haruni, E., Kokhkharova, M., and Schäfer, J. (2020). Device free human activity and fall recognition using WiFi channel state information (CSI). *Transactions on Pervasive Computing and Interaction*, 2(1):1–17.

[Dou and Huan 2021] Dou, C. and Huan, H. (2021). Full respiration rate monitoring exploiting doppler information with commodity wi-fi devices. *Sensors*, 21(10):3505.

[Duan et al. 2018a] Duan, S., Yu, T., and He, J. (2018a). WiDriver: Driver Activity Recognition System Based on WiFi CSI. *International Journal of Wireless Information Networks*, 25(2):146–156.

[Duan et al. 2018b] Duan, S., Yu, T., and He, J. (2018b). Widriver: Driver activity recognition system based on wifi csi. *International Journal of Wireless Information Networks*, 25(2):146–156.

[Fang et al. 2016] Fang, B., Lane, N. D., Zhang, M., Boran, A., and Kawsar, F. (2016). Bodyscan: Enabling radio-based sensing on wearable devices for contactless activity and vital sign monitoring. In *Proceedings of the international conference on mobile systems, applications, and services*, pages 97–110.

[Gringoli and Nava 2009] Gringoli, F. and Nava, L. (2009). Open firmware for wifi networks.

[Gringoli et al. 2019] Gringoli, F., Schulz, M., Link, J., and Hollick, M. (2019). Free your CSI: A channel state information extraction platform for modern Wi-Fi chipsets. *Mobile Computing and Networking (MOBICOM)*, pages 21–28.

[Gu et al. 2018] Gu, Y., Liu, T., Li, J., Ren, F., Liu, Z., Wang, X., and Li, P. (2018). Emosense: Data-driven emotion sensing via off-the-shelf wifi devices. In *IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE.

[Gu et al. 2017] Gu, Y., Zhan, J., Ji, Y., Li, J., Ren, F., and Gao, S. (2017). MoSense: An RF-Based Motion Detection System via Off-the-Shelf WiFi Devices. *IEEE Internet of Things Journal*, 4(6):2326–2341.

[Gu et al. 2021] Gu, Y., Zhang, X., Yan, H., Liu, Z., and Ren, F. (2021). Wital: WiFi-based Real-time Vital Signs Monitoring During Sleep.

[Halperin et al. 2011] Halperin, D., Hu, W., Sheth, A., and Wetherall, D. (2011). Tool release: Gathering 802.11 n traces with channel state information. *ACM SIGCOMM Computer Communication Review*, 41(1):53–53.

[IEEE 802.11 Working Group 2003] IEEE 802.11 Working Group (2003). Ieee 802.11g-2003 - ieee standard for information technology. Technical report, IEEE.

[IEEE 802.11 Working Group 2009] IEEE 802.11 Working Group (2009). Ieee 802.11n-2009 - ieee standard for information technology. Technical report, IEEE.

[IEEE 802.11 Working Group 2013] IEEE 802.11 Working Group (2013). Ieee 802.11ac-2013 - ieee standard for information technology. Technical report, IEEE.

[Khamis et al. 2018] Khamis, A., Chou, C. T., Kusy, B., and Hu, W. (2018). Cardiofi: Enabling heart rate monitoring on unmodified COTS WiFi devices. *ACM International Conference Proceeding Series*, pages 97–106.

[Khan 2017] Khan, S. F. (2017). Health care monitoring system in internet of things (iot) by using rfid. In *2017 6th International Conference on Industrial Technology and Management (ICITM)*, pages 198–204. IEEE.

[Khan et al. 2017] Khan, U. M., Kabir, Z., and Hassan, S. A. (2017). Wireless health monitoring using passive wifi sensing. In *International Wireless Communications and Mobile Computing Conference*, pages 1771–1776. IEEE.

[Lee et al. 2018] Lee, S., Park, Y. D., Suh, Y. J., and Jeon, S. (2018). Design and implementation of monitoring system for breathing and heart rate pattern using WiFi signals. *IEEE Annual Consumer Communications and Networking Conference*, pages 1–7.

[Li et al. 2021] Li, F., Valero, M., Shahriar, H., Khan, R. A., and Ahamed, S. I. (2021). Wi-covid: A covid-19 symptom detection and patient monitoring framework using wifi. *Smart Health*, 19:100147.

[Li et al. 2019] Li, H., He, X., Chen, X., Fang, Y., and Fang, Q. (2019). Wi-motion: A robust human activity recognition using WiFi signals. *IEEE Access*, 7:153287–153299.

[Li et al. 2018] Li, H., Ota, K., Dong, M., and Guo, M. (2018). Learning human activities through wi-fi channel state information with multiple access points. *IEEE Communications Magazine*, 56(5):124–129.

[Liu et al. 2018] Liu, J., Chen, Y., Wang, Y., Chen, X., Cheng, J., and Yang, J. (2018). Monitoring vital signs and postures during sleep using WiFi signals. *IEEE Internet of Things Journal*, 5(3):2071–2084.

[Liu et al. 2019] Liu, J., Liu, H., Chen, Y., Wang, Y., and Wang, C. (2019). Wireless sensing for human activity: A survey. *IEEE Communications Surveys & Tutorials*, 22(3):1629–1645.

[Liu et al. 2015a] Liu, J., Wang, Y., Chen, Y., Yang, J., Chen, X., and Cheng, J. (2015a). Tracking vital signs during sleep leveraging off-the-shelf wifi. In *Proceedings of the ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pages 267–276.

[Liu et al. 2014] Liu, X., Cao, J., Tang, S., and Wen, J. (2014). Wi-sleep: Contactless sleep monitoring via wifi signals. In *IEEE Real-Time Systems Symposium*, pages 346–355. IEEE.

[Liu et al. 2015b] Liu, X., Cao, J., Tang, S., Wen, J., and Guo, P. (2015b). Contactless respiration monitoring via off-the-shelf wifi devices. *IEEE Transactions on Mobile Computing*, 15(10):2466–2479.

[Ma et al. 2016] Ma, J., Wang, Y., Wang, H., Wang, Y., and Zhang, D. (2016). When can we detect human respiration with commodity wifi devices? In *ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*, pages 325–328.

[Ma et al. 2019a] Ma, Y., Zhou, G., and Wang, S. (2019a). WiFi sensing with channel state information: A survey. *ACM Computing Surveys*, 52(3).

[Ma et al. 2019b] Ma, Y., Zhou, G., and Wang, S. (2019b). Wifi sensing with channel state information: A survey. *ACM Computing Surveys*, 52(3):1–36.

[Schulz et al. 2018] Schulz, M., Gringoli, F., Link, J., and Hollick, M. (2018). Shadow Wi-Fi: Teaching smartphones to transmit raw signals and to extract channel state information to implement practical covert channels over Wi-Fi. *ACM International Conference on Mobile Systems, Applications, and Services*, pages 256–268.

[Shang and Wu 2016] Shang, J. and Wu, J. (2016). Fine-grained vital signs estimation using commercial wi-fi devices. In *Proceedings of the Eighth Wireless of the Students, by the Students, and for the Students Workshop*, pages 30–32.

[Sharma et al. 2021] Sharma, A., Li, J., Mishra, D., Batista, G., and Seneviratne, A. (2021). Passive WiFi CSI sensing based machine learning framework for COVID-Safe occupancy monitoring. In *IEEE International Conference on Communications Workshops*, pages 1–6. IEEE.

[Shi et al. 2018] Shi, Z., Zhang, J. A., Xu, R., and Fang, G. (2018). Human activity recognition using deep learning networks with enhanced channel state information. In *IEEE Globecom Workshops*, pages 1–6. IEEE.

[Sorber et al. 2013] Sorber, L., Van Barel, M., and De Lathauwer, L. (2013). Optimization-based algorithms for tensor decompositions: Canonical polyadic decomposition, decomposition in rank-(l_r,l_r,1) terms, and a new generalization. *SIAM Journal on Optimization*, 23(2):695–720.

[Tan et al. 2018] Tan, B., Chen, Q., Chetty, K., Woodbridge, K., Li, W., and Piechocki, R. (2018). Exploiting WiFi Channel State Information for Residential Healthcare Informatics. *IEEE Communications Magazine*, 56(5):130–137.

[Tan et al. 2018] Tan, B., Chen, Q., Chetty, K., Woodbridge, K., Li, W., and Piechocki, R. (2018). Exploiting wifi channel state information for residential healthcare informatics. *IEEE Communications Magazine*, 56(5):130–137.

[Uchiyama et al. 2021] Uchiyama, A., Saruwatari, S., Maekawa, T., Ohara, K., and Higashino, T. (2021). Context recognition by wireless sensing: A comprehensive survey. *Journal of Information Processing*, 29:46–57.

[Wang et al. 2016a] Wang, G., Zou, Y., Zhou, Z., Wu, K., and Ni, L. M. (2016a). We can hear you with wi-fi! *IEEE Transactions on Mobile Computing*, 15(11):2907–2920.

[Wang et al. 2016b] Wang, H., Zhang, D., Ma, J., Wang, Y., Wang, Y., Wu, D., Gu, T., and Xie, B. (2016b). Human respiration detection with commodity wifi devices: do user location and body orientation matter? In *Proceedings of the ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 25–36.

[Wang et al. 2017a] Wang, P., Guo, B., Xin, T., Wang, Z., and Yu, Z. (2017a). Tinysense: Multi-user respiration detection using wi-fi csi signals. In *IEEE International Conference on e-Health Networking, Applications and Services*, pages 1–6. IEEE.

[Wang et al. 2017b] Wang, X., Yang, C., and Mao, S. (2017b). PhaseBeat: Exploiting CSI phase data for vital sign monitoring with commodity WiFi devices. In *IEEE International Conference on Distributed Computing Systems*, pages 1230–1239. IEEE.

[Wang et al. 2017c] Wang, X., Yang, C., and Mao, S. (2017c). Tensorbeat: Tensor decomposition for monitoring multiperson breathing beats with commodity wifi. *ACM Trans. Intell. Syst. Technol.*, 9(1).

[Wang et al. 2020] Wang, X., Yang, C., and Mao, S. (2020). On CSI-Based Vital Sign Monitoring Using Commodity WiFi. *ACM Transactions on Computing for Healthcare*, 1(3):1–27.

[Wang et al. 2019] Wang, Z., Jiang, K., Hou, Y., Dou, W., Zhang, C., Huang, Z., and Guo, Y. (2019). A survey on human behavior recognition using channel state information. *IEEE Access*, 7:155986–156024.

[Weinstein and Ebert 1971] Weinstein, S. and Ebert, P. (1971). Data transmission by Frequency-Division Multiplexing using the Discrete Fourier Transform. *IEEE Transactions on Communication Technology*, 19(5):628–634.

[Wu et al. 2017] Wu, D., Zhang, D., Xu, C., Wang, H., and Li, X. (2017). Device-free wifi human sensing: From pattern-based to model-based approaches. *IEEE Communications Magazine*, 55(10):91–97.

[Wu et al. 2018] Wu, X., Chu, Z., Yang, P., Xiang, C., Zheng, X., and Huang, W. (2018). Tw-see: Human activity recognition through the wall with commodity wi-fi devices. *IEEE Transactions on Vehicular Technology*, 68(1):306–319.

[Xiao et al. 2017] Xiao, F., Guo, Z., Zhu, H., Xie, X., and Wang, R. (2017). Ampn: Real-time los/nlos identification with wifi. In *IEEE International Conference on Communications (ICC)*, pages 1–7. IEEE.

[Xiao et al. 2016] Xiao, J., Zhou, Z., Yi, Y., and Ni, L. M. (2016). A survey on wireless indoor localization from the device perspective. *ACM Computing Surveys*, 49(2):1–31.

[Xie et al. 2018] Xie, Y., Li, Z., and Li, M. (2018). Precise power delay profiling with commodity wi-fi. *IEEE Transactions on Mobile Computing*, 18(6):1342–1355.

[Yang et al. 2018] Yang, Y., Cao, J., Liu, X., and Xing, K. (2018). Multi-person sleeping respiration monitoring with cots wifi devices. In *IEEE International Conference on Mobile Ad Hoc and Sensor Systems*, pages 37–45. IEEE.

[Yang et al. 2013] Yang, Z., Zhou, Z., and Liu, Y. (2013). From rssi to csi: Indoor localization via channel response. *ACM Computing Surveys*, 46(2):1–32.

[Yousefi et al. 2017] Yousefi, S., Narui, H., Dayal, S., Ermon, S., and Valaee, S. (2017). A survey on behavior recognition using wifi channel state information. *IEEE Communications Magazine*, 55(10):98–104.

[Zeng et al. 2018] Zeng, Y., Wu, D., Gao, R., Gu, T., and Zhang, D. (2018). Fullbreathe: Full human respiration detection exploiting complementarity of csi phase and amplitude of wifi signals. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2(3):1–19.

[Zeng et al. 2020] Zeng, Y., Wu, D., Xiong, J., Liu, J., Liu, Z., and Zhang, D. (2020). Multisense: Enabling multi-person respiration sensing with commodity wifi. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 4(3):1–29.

[Zeng et al. 2019] Zeng, Y., Wu, D., Xiong, J., Yi, E., Gao, R., and Zhang, D. (2019). FarSense: Pushing the range limit of WiFi-based respiration sensing with CSI ratio of two antennas. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 3(3):1–26.

[Zhang et al. 2019] Zhang, D., Hu, Y., Chen, Y., and Zeng, B. (2019). BreathTrack: Tracking indoor human breath status via commodity WiFi. *IEEE Internet of Things Journal*, 6(2):3899–3911.

[Zhang et al. 2018] Zhang, F., Zhang, D., Xiong, J., Wang, H., Niu, K., Jin, B., and Wang, Y. (2018). From Fresnel Diffraction Model to Fine-grained Human Respiration Sensing with Commodity Wi-Fi Devices. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, pages 1–23.

[Zhang et al. 2017] Zhang, J., Xu, W., Hu, W., and Kanhere, S. S. (2017). Wicare: Towards in-situ breath monitoring. In *EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, pages 126–135.

[Zhao et al. 2018] Zhao, M., Adib, F., and Katabi, D. (2018). Emotion recognition using wireless signals. *Communications of the ACM*, 61(9):91–100.

[Zhu et al. 2017] Zhu, D., Pang, N., Li, G., and Liu, S. (2017). Notifi: A ubiquitous wifi-based abnormal activity detection system. In *International Joint Conference on Neural Networks*, pages 1766–1773. IEEE.