



XXXVI Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos

Minicursos do SBRC 2018

6 a 10 de maio
Campos do Jordão | SP

www.sbrc2018.ufscar.br

36º SIMPÓSIO BRASILEIRO DE REDES DE COMPUTADORES E SISTEMAS DISTRIBUÍDOS

Campos do Jordão, SP, 6 a 10 de maio de 2018

MINICURSOS

Editora

Sociedade Brasileira de Computação – SBC

Organizadores

Fábio Luciano Verdi (UFSCar)

Jó Ueyama (USP)

Daniel L. Guidoni (UFSJ)

Realização

Universidade Federal de São Carlos (UFSCar)

Universidade de São Paulo (USP)

Sociedade Brasileira de Computação (SBC)

Laboratório Nacional de Redes de Computadores (LARC)



Esta obra está sob a licença Creative Commons Atribuição 4.0 (CC-BY). Você pode redistribuir este livro em qualquer suporte ou formato e copiar, remixar, transformar e criar a partir do conteúdo deste livro para qualquer fim, desde que cite a fonte.

Dados Internacionais de Catalogação na Publicação (CIP)

S612 Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (36. : 06 – 10 maio 2018 : Campos do Jordão, SP)

Minicursos do SBRC 2018 [recurso eletrônico] / organização: Fábio Luciano Verdi ... [et al.]. – Dados eletrônicos. – Porto Alegre : Sociedade Brasileira de Computação, 2018.

256 p. : il. : PDF.

Modo de acesso: World Wide Web.

Inclui bibliografia

ISBN 978-85-7669-684-1 (e-book)

1. Computação – Brasil – Evento. 2. Redes de computadores. 3. Sistemas computacionais. 4. Sistemas distribuídos. I. Verdi, Fábio Luciano. II. Ueyama, Jó. III. Guidoni, Daniel L.. IV. Sociedade Brasileira de Computação. V. Título.

CDU 004(063)

Ficha catalográfica elaborada por Annie Casali – CRB-10/2339

Biblioteca Digital da SBC – SBC OpenLib



Sociedade Brasileira de Computação

Av. Bento Gonçalves, 9500

Setor 4 | Prédio 43.412 | Sala 219 | Bairro

Agronomia Caixa Postal 15012 | CEP 91501-970

Porto Alegre - RS

Fone: (51) 99252-

6018

sbc@sbc.org.br

Sociedade Brasileira de Computação – SBC

Presidência

Thais Vasconcelos Batista (UFRN), Presidente

Cristiano Maciel (UFMT), Vice-Presidente

Diretorias

Renata de Matos Galante (UFRGS), Diretora Administrativa

Alirio Santos de Sá (UFBA), Diretor de Comunicação

Leila Ribeiro (UFRGS), Diretora de Computação na Educação Básica

Carlos Eduardo Ferreira (USP), Diretor de Competições Científicas

Ronaldo Alves Ferreira (UFMS), Diretor de Cooperação com Sociedades Científicas

Rodrigo Silva Duran (IFB), Diretor de Educação

Denis Lima do Rosário (UFPA), Diretor de Eventos e Comissões Especiais

Francisco Dantas Medeiros Neto (UERN), Diretor de Finanças

Flávia Maria Santoro (INTELI), Diretora de Inovação

André Luís de Medeiros Santos (UFPE), Diretor de Planejamento e Programas Especiais

José Viterbo Filho (UFF), Diretor de Publicações

Michelle Silva Wangham (UNIVALI), Diretora de Relações Profissionais

Eunice Pereira dos Santos Nunes (UFMT), Diretora de Secretarias Regionais

Marcelo Antonio Marotta (UNB), Diretoria Extraordinária de Tecnologia da Informação

Contato

Av. Bento Gonçalves, 9500

Setor 4 - Prédio 43.412 - Sala 219

Bairro Agronomia

91.509-900 – Porto Alegre RS

CNPJ: 29.532.264/0001-78

<http://www.sbrc.org.br>

Mensagem do coordenador de Minicursos

O Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC) é o mais importante fórum da comunidade de pesquisa e desenvolvimento em Redes de Computadores e Sistemas Distribuídos no Brasil. Entre as principais atividades técnicas do SBRC encontram-se os minicursos, que têm como objetivo a atualização em temas normalmente não cobertos nas grades curriculares ou que despertem grande interesse entre acadêmicos e profissionais. Este livro contém os minicursos selecionados para apresentação no XXXVI SBRC, realizado em Campos do Jordão - SP, entre os dias 6 e 10 de Maio de 2018. Cada minicurso compõe um capítulo deste livro e foi projetado para uma apresentação de 3 a 4 horas no evento.

Nessa edição, montamos um Comitê de Avaliação com 22 pesquisadores de diferentes regiões do Brasil, de diferentes áreas dentro da ciência da computação e com atuações expressivas em suas respectivas áreas. Para esta edição, foram submetidas 23 propostas para avaliação, das quais cinco foram selecionadas para apresentação no evento, correspondendo a uma taxa de aceitação de 21,7%. Infelizmente, várias propostas também de excelente qualidade não foram aceitas para apresentação. Cada proposta foi avaliada por, pelo menos, três avaliadores, que contribuíram com comentários relevantes para a criação da versão final dos capítulos. A qualidade dos capítulos se deve, fundamentalmente, ao esforço de cada um dos autores. A organização do evento é grata pelo esforço e horas trabalhadas. Encorajamos a comunidade de Redes de Computadores e Sistemas Distribuídos a continuar prestigiando o SBRC e enviando novas propostas de minicursos para futuras edições do SBRC.

Os minicursos selecionados para esta edição abordam temas de grande relevância e que vem despertando interesse da comunidade acadêmica e profissional no Brasil. Os capítulos deste livro são: Capítulo 1: Humanos no ciclo de comunicação: facilitadores das redes de próxima geração; Capítulo 2: Internet-das-Bionano-coisas: Conectando-se às nano-máquinas; Capítulo 3: Aprendizado de Máquina em Plataformas de Processamento Distribuído de Fluxo: Análise e Detecção de Ameaças em Tempo Real; Capítulo 4: Introdução à Linguagem P4 - Teoria e Prática; Capítulo 5: Blockchain e a Revolução do Consenso sob Demanda.

Gostaria de expressar meus sinceros agradecimentos aos colegas Jó Ueyama (USP) e Fábio Luciano Verdi (UFSCar), coordenadores do SBRC 2018, pela confiança depositada e pelo apoio recebido durante a condução de todo o processo. Um especial agradecimento aos Membros do Comitê de Avaliação pelas revisões e boas discussões que contribuíram para o processo de seleção. Meu agradecimento aos autores de propostas que prestigiaram a trilha de minicursos do SBRC.

Daniel Ludovico Guidoni
Coordenador de Minicursos do SBRC 2018

Comitê de Avaliação de Minicursos

Alberto Schaeffer-Filho (UFRGS)
Alex Vieira (UFJF)
Artur Ziviani (LNCC)
Bruno Kimura (UNIFESP)
Carlos Kamienski (UFABC)
Christian Esteve (UNICAMP)
Cristiano Silva (UFSJ)
Denis Rosário (UFPA)
Eduardo Cerqueira (UFPA)
Fabíola Greve (UFBA)
Felipe Domingos da Cunha (PUC Minas)
Fernanda Sumika Souza (UFSJ)
Flavia Delicato (UFRJ)
Guilherme Maia (UFMG)
Heitor Ramos (UFAL)
Horácio Oliveira (UFAM)
José Augusto Suruagy Monteiro (UFPE)
Leandro Villas (UNICAMP)
Luciano Paschoal Gaspary (UFRGS)
Michele Nogueira (UFPR)
Pedro Olmo Vaz de Melo (UFMG)
Thiago Silva (UTFPR)

Sumário

1.	Humanos no ciclo de comunicação: facilitadores das redes de próxima geração	1
2.	Internet-das-Bionano-Coisas: Conectando-se às Nanomáquinas	52
3.	Aprendizado de Máquina em Plataformas de Processamento Distribuído de Fluxo: Análise e Detecção de Ameaças em Tempo Real	102
4.	Introdução à Linguagem P4 - Teoria e Prática	158
5.	Blockchain e a Revolução do Consenso sob Demanda	201

Capítulo

1

Humanos no ciclo de comunicação: facilitadores das redes de próxima geração

Rafael Lima Costa (INRIA/UFBA/Université Paris-Saclay), Aline Carneiro Viana (INRIA), Artur Ziviani (LNCC), Leobino Nascimento Sampaio (UFBA)

Abstract

Next-generation mobile networks (including its models, architectures, protocols, types of communication and related technologies) shall take human-behavior into account to predict situations and shape dynamically requirements to be provided. This short-course aims to unravel and explore the “human-aware” paradigm including human characteristics such as mobility, personality traits, traffic profile, routines and social interactions. In addition to discussing the subject, relations with future network-concepts, informations for experiments and evaluation, research challenges and additional lessons from other areas in the same context are presented.

Resumo

As redes móveis do futuro (incluindo seus modelos, arquiteturas, protocolos, tipos de comunicação e tecnologias relacionadas) deverão levar em conta o comportamento humano para prever situações e moldar dinamicamente os requisitos a serem fornecidos. Esse minicurso objetiva desvendar e explorar o paradigma “ciente do humano”, que engloba características humanas como mobilidade, traços de personalidade, perfil de tráfego, rotinas e interações sociais. Além de discutir o tema, serão evidenciadas as relações com conceitos de redes do futuro, apresentadas informações para avaliação e experimentação, desafios de pesquisa e lições adicionais de outras áreas no mesmo contexto.

1.1. Introdução

Este minicurso explora e desvenda o paradigma “ciente do humano” no contexto de redes móveis, de forma abrangente e atualizada. Nesta seção, primeiramente expomos a contextualização, motivações e relevância do tema do minicurso. Em seguida, a evolução

dos paradigmas de comunicação em direção ao “ciente do humano” é analisada bem como alguns cenários de aplicação são discutidos. Por fim, são apresentados o objetivo e a organização do restante do texto.

1.1.1. Contextualização, motivações e relevância

A visão de computação ubíqua (ou computação pervasiva), em que a informática se tornaria onipresente no cotidiano das pessoas já é uma realidade. Isso tem acontecido, dentre outros fatores, através da disseminação de aparelhos móveis e das redes sem-fio. Ademais, o surgimento, implementação e popularização de tecnologias como Internet das Coisas (*Internet of Things – IoT*), equipamentos vestíveis, redes veiculares, aplicações de *big data analytics* e plataformas multimídia “famintas por banda” trazem desafios na perspectiva de redes de computadores.

Relatório da Ericsson [eri, 2017] de junho/2017 prevê mais de 1 milhão de novas adesões às redes móveis por dia nos próximos cinco anos, alcançando 29 bilhões de nós conectados (62% relativos à *IoT*) até 2022. Já a Cisco [cis, 2017a] prevê tráfego IP global de 3.3 *zettabytes/ano* até 2021 (3 vezes maior em relação a 2016), sendo cerca de 63% do total originado por *smartphones*. Investimentos de bilhões de dólares em infraestrutura não estão sendo suficientes para suprir a demanda e outras alternativas passaram a ser buscadas, dentre elas o *offloading* de dados através de equipamentos dos usuários [Rebecchi et al., 2015]. De acordo com [cis, 2017b], em 2016, 60% do tráfego móvel foi descarregado via *WiFi* ou *femtocells*. Ainda assim, as redes móveis estão sobrecarregadas, possuem limitações de espectro e alto custo de atualização infraestrutural. A partir desses estudos, podemos inferir que se tais técnicas de *offloading* não fossem buscadas, as redes móveis entrariam em colapso. Além do problema de sobrecarga, em padrões de redes móveis anteriores como o LTE, a preocupação maior sempre foi com parâmetros da rede e não com os objetivos de cada usuário. Isso fez com que requisitos-chave de desempenho tivessem implementações sub-ótimas [Lauridsen et al., 2017].

A comunicação centrada no usuário (do inglês *user-centric communication*) [Afra-siabi e Guérin, 2016] apareceu como uma das primeiras soluções para *offloading de dados* e aumento de capilaridade com foco nas características de uso da rede do utilizador. Em paralelo, a abstração “centrada no usuário” (onde soluções dependem da participação do mesmo) ganhou força em diversos tópicos de pesquisa na áreas de redes, incluindo Redes Definidas por Software e Redes Sociais Móveis. Ultimamente, o paradigma *user-centric* tem avançado em direção ao que é chamado de *human-aware* (ciente do humano) ou *human-behavior leveraging* e a novos padrões como 5G, Redes Veiculares, Redes Ultra-Densas, *IoT* e novas aplicações baseadas em interações diretas entre dispositivos [Panwar et al., 2016, Chen et al., 2016, Iosifidis et al., 2017, Li et al., 2017]. Isso tem acontecido devido à preocupação recente da comunidade científica de observar características do usuário como um ser humano e não apenas como um nó da rede.

A arquitetura das redes móveis do futuro será cada vez mais centrada nos requisitos comportamentais dos usuários. Ou seja, nas características ou hábitos rotineiros do ser humano por trás de um dispositivo de comunicação, de forma a melhor servi-lo. Sendo assim, o estudo do comportamento humano [Thilakarathna et al., 2017] tem sido essen-

cial ao avanço de pesquisas na área. Um indivíduo possui características como mobilidade, personalidade, interações sociais, caráter, humor e rotinas que podem ser estudadas para oferecer um serviço de rede mais adequado [Oliveira et al., 2016, Oliveira et al., 2017, Chen et al., 2017b]. Historicamente, diversos trabalhos examinaram características do usuário como a mobilidade [Lau et al., 2017, Xia et al., 2018]. No entanto, pouco foi feito no sentido de tentar utilizar essas informações para identificar determinadas rotinas e prever o comportamento futuro do humano como usuário de um dispositivo. É importante ressaltar que o perfil de cada pessoa no contexto de redes móveis é ditado por diversos fatores do seu comportamento humano. Dessa forma, para aproximar mais as redes e os seus usuários, cada vez mais se faz necessário estudar suas características.

O papel do usuário como um elemento ativo na rede rompe com o atual modelo fim-a-fim da Internet e será essencial para garantir requisitos de novos padrões de comunicação e aplicações, além de questões antes preteridas como a eficiência energética [I et al., 2014]. O objetivo da comunicação centrada no usuário é melhorar a experiência do mesmo independente do dispositivo, rede, recursos utilizados, localização, condições de mobilidade e sobretudo basear-se no seu contexto. Assim, os paradigmas *user-centric* e *human-aware* são vistos como essenciais nas redes do futuro e vão além em relação a outras soluções baseadas na participação dos usuários como *Mesh*, *Ad-hoc* e *MANETs* [Conti et al., 2015]. Pela definição e arquitetura dessas propostas anteriores, observamos que os usuários eram tratados como elementos de rede. No caso, eles serviam a objetivos específicos como o aumento da capilaridade ou encaminhamento de mensagens. Grande parte dos desafios dessas soluções, tais como roteamento, eficiência energética, mecanismos de incentivo e gerenciamento de recursos esbarraram justamente na ausência de um entendimento dos aspectos individuais de cada indivíduo como nó da rede. Já as soluções no contexto *user-centric* e *human-aware* caminham em direção a cenários de aplicação e mecanismos de incentivo bem definidos, dão suporte a novos modelos de negócio e sobretudo são do interesse das operadoras e de prestadoras de serviços. A implementação do “ciente do humano” evidencia uma situação vantajosa dos dois lados, possibilitando melhor *QoE* do usuário ao utilizar serviços da operadora, que por sua vez pode maximizar o lucro com menor custo, oferecer novos serviços e atrair novos clientes. Além disso, o entendimento dos aspectos humanos podem ser úteis para o dimensionamento das redes e para oferecer um serviço mais próximo do que o usuário espera naturalmente.

Levar em conta o comportamento dos usuários de redes baseado nas suas características humanas é um dos entendimentos necessários para as arquiteturas do futuro. O termo “centrado no usuário” é mais comumente encontrado na área de redes, no entanto, diversas pesquisas de outras áreas como física, sociologia e psicologia estudam o comportamento humano no contexto de telecomunicações e não utilizam o termo. Além de avançar em direção a novos padrões de comunicação, o “centrado no usuário” do futuro será orientado ao contexto e comportamento humano, dando início a esse novo paradigma “ciente do humano”. Um dos focos do minicurso é discutir como e porque o humano estará no centro das atenções das arquiteturas de redes de próxima geração. Ao longo do texto os termos *user-centric* (centrado no usuário) e *human-aware* (ciente do humano) serão usados para evidenciar a diferença e foco de cada um. No entanto, o elemento central é o “ciente do humano”, ou seja que engloba o estudo do comportamento humano como facilitador das redes de próxima geração, justificando o título do minicurso.

As operadoras passaram a ter interesse em padrões de comunicação centrados no usuário, tais como, o *D2D (Device-to-Device Communication)* [Gandotra et al., 2017] ao perceber que poderiam controlar e definir as regras da comunicação. Além do aumento da capilaridade a custo reduzido, novas aplicações e modelos de negócio, com modelos de comunicação cientes do humano é possível garantir requisitos como baixa latência e maior vazão, comuns a arquiteturas como 5G e V2V (Vehicle-to-Vehicle) [Xia et al., 2018]. Conforme citado em [Akyildiz et al., 2016], o D2D combinado com padrões, tecnologias e arquiteturas, como SDN, NFV, *Big Data* e Computação em Nuvem Móvel, pode ser usado para: (i) Garantir latência reduzida de 1 ms fim-a-fim; (ii) Reduzir em até 1000 vezes o consumo energético por bit; (iii) Obter maiores taxa de transmissão; (iv) Permitir o reuso eficiente do espectro; (v) Prover melhor conectividade para usuários na borda da rede. Sendo assim, a concepção de modelos de comunicação cientes do humano vem ganhando importância também em tecnologias como *IoT* [Kozioł et al., 2017] e *Machine-to-Machine Communication* [Cheng et al., 2015].

Tópicos de pesquisa atuais na área de redes como *SDN*, *NFV* e *IoT* apresentam trabalhos com soluções onde o usuário é um elemento central. O tema aparece em chamadas e publicações de conferências e periódicos relevantes, tais como *IEEE Communications Magazine*, *ACM Computer Communication Review*, *The International Journal for the Computer and Telecommunications Industry (Elsevier)*, *Elsevier Computer Networks*, *IEEE Transactions on Mobile Computing*, *IEEE GLOBECOM* e *ACM SIGCOMM*. Já o *human-aware* é recente e tem sido encontrado em periódicos internacionais como *Nature*, *Pervasive and Mobile Computing (Elsevier)* e em conferências de *Mobile Networking*. Este minicurso apresenta o paradigma de comunicação orientada ao contexto e comportamento humano do usuário de forma abrangente e mostra como ele está relacionado com diversos temas atuais em redes de computadores e sistemas distribuídos.

1.1.2. Evolução de paradigmas de rede até o “ciente do humano”

Ao longo do tempo, as redes de computadores foram sendo construídas através da perspectiva dos provedores, oferecendo um serviço chamado de “melhor esforço” (do inglês *best-effort*). O resultado disso foi a engenharia de protocolos e serviços voltados às aplicações e à conveniência dos provedores, porém distantes das expectativas reais dos usuários. Essa lacuna pode ser observada em técnicas de otimização de desempenho que buscam adaptar-se às condições da rede (ex: condições físicas dos links, mudanças de topologia) e são específicas em termos de protocolo ou serviço (ex: entrega de mensagens com sucesso, cobertura geográfica de rede). Portanto, é possível afirmar que as redes geralmente não se preocupam com o comportamento e necessidade dos usuários.

Soluções “centradas no usuário”, apesar de representarem uma mudança relativa ao tradicional “orientado ao provedor”, ainda enxergam o usuário mais como um elemento da rede do que como uma pessoa interagindo através de um dispositivo de comunicação. Dito isso, essas soluções consideram aspectos do usuário do ponto de vista de rede (tais como perfil de tráfego e possibilidade de compartilhar sua conexão), ignorando características do comportamento individual humano. A mudança de paradigma que esse minicurso chama atenção é a de uma rede construída para controlar, localizar e monitorar usuários para uma rede sensível às suas necessidades e capaz de se adaptar ao comportamento humano deles.

Atualmente, os usuários esperam muito mais das redes do que apenas um melhor desempenho. Revisitando a história recente das últimas gerações de redes móveis, observamos que a maximização ou minimização de alguns parâmetros de rede (ex: vazão, atraso, etc) foram as maiores preocupações. De acordo com [Akyildiz et al., 2016], ao contrário das gerações passadas, o 5G deverá suportar uma grande variedade de dispositivos, serviços e aplicações como veículos autônomos, *IoT*, comunicação dispositivo-a-dispositivo e equipamentos vestíveis. Já que as pessoas estão cada vez mais propensas a interagir através de dispositivos inteligentes, suas rotinas e comportamento humano estarão diretamente ligados a como elas irão requisitar recursos. A ‘inteligência’ dos dispositivos e das redes deverá ser justamente no sentido de levar em conta essas informações para entregar soluções e serviços de comunicação mais eficientes e sensíveis ao comportamento humano, assim como já acontece de certa forma com sistemas de recomendação.

As redes ciente do humano podem ser definidas precisamente como *redes que levam em conta o comportamento humano para prever necessidades e ações conscientes e inconscientes das pessoas, se adaptando à incerteza e heterogeneidade do que pode ser predito e sendo capaz de oferecer uma melhor qualidade de experiência*. Dados a serem utilizados na arquitetura de redes do futuro podem ser obtidos através dos próprios dispositivos dos usuários. Atualmente, os mesmos já são equipados com uma série de sensores capazes de traduzir com acurácia vários fenômenos físicos na forma de dados. Isso vai muito além das tradicionais medições de padrões de uso e métricas de rede. Além de microfone, *GPS*, câmera e giroscópios, espera-se no futuro a incorporação de sensores de pressão, monitores de ar e de saúde (ex: batimentos cardíacos), etc. Esses sensores se tornam ferramentas capazes de identificar a atividade humana no momento. Consequentemente, uma grande quantidade de dados sensoriais do contexto humano (*big data*) pode ser coletada e processada. Isso dá a oportunidade de inferir interações e comportamento numa escala global, já que a população de usuários de dispositivos inteligentes cresce a cada dia. Se usados corretamente, esses dados sensoriais podem alavancar os serviços de redes a um patamar jamais visto antes.

Os primeiros avanços no estudo do comportamento humano em redes de computadores foram relativos à análise de mobilidade [CC et al., 2016]. Grande parte dos trabalhos nesse contexto exploram particularidades dos contatos entre usuários como frequência e tempo entre eles, porém não levam em conta outras características humanas da mobilidade como interação com o ambiente e a forma como os caminhos são escolhidos. Outras peculiaridades como rotinas, o comportamento relativo à traços de personalidade individuais, necessidades, tipo de ambiente e interesses particulares podem ser utilizadas para personalizar serviços e aplicações de rede. O entendimento do comportamento subjetivo de cada ser humano irá guiar novas soluções e arquiteturas de rede futuras. Do ponto de vista evolutivo, o “centrado no usuário” se move em direção ao “ciente do humano”, conforme linha do tempo apresentada na Figura 1.1. Na Figura estão destacados fatos relacionados a pesquisas relevantes realizadas de 2008 até o presente momento, além do que pode ser esperado num futuro próximo. Nos limitaremos a trazer referências dos trabalhos mais recentes, por questões óbvias de espaço e também pela maior relevância para futuras pesquisas.

Em meados dos anos 2000, a mobilidade humana já era estudada pela comunidade de Redes Tolerantes ao Atraso (do inglês *Delay-Tolerant Networks*) [Mtibaa et al., 2006,

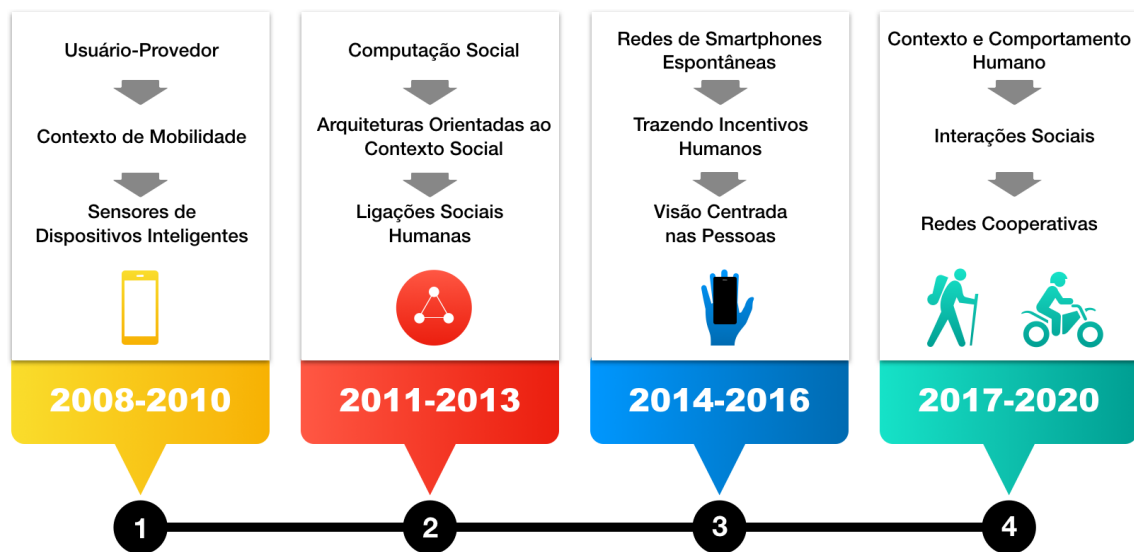


Figura 1.1. Linha do Tempo mostrando a evolução da visão do paradigma “centrado no usuário” em direção ao “ciente do humano”.

Conan et al., 2007]. Em 2008, a comunicação centrada no usuário apareceu como solução para extensão de capilaridade e *offloading* de dados. A ideia era o compartilhamento de conectividade e serviços do usuário aplicado em cenários específicos como desastres, alta densidade de pessoas e borda da rede. Desafios foram identificados e parcialmente solucionados, porém a maior parte das discussões continuaram a ser de uma perspectiva de rede, onde o usuário era visto como um nó compartilhador.

Durante 2009, a mobilidade foi estudada no contexto de modelos centrados no usuário e *frameworks* de gerenciamento baseados em redes sociais móveis foram propostos. Ademais, a visão futura de uma Internet voltada ao usuário foi introduzida. Já em 2010, foi observada a importância da coleta de dados dos sensores presentes em dispositivos inteligentes para auxiliar modelos e arquiteturas de rede do futuro. Surgiram também as primeiras propostas centradas no usuário para o 4G e redes veiculares.

No ano de 2011, aspectos sociais humanos passaram a ser considerados em gerenciamento de confiabilidade e modelagem de mobilidade. Aplicações baseadas na comunicação direta entre dispositivos e modelos centrados no usuário para 5G surgiram. Ao longo de 2012, observamos trabalhos baseados em arquiteturas orientadas a aspectos sociais, incorporando princípios e comportamento, sobretudo de usuários em redes sociais. Ideias de roteamento contemplando as rotinas dos usuários também foram propostas. Em 2013, surgiram iniciativas baseadas em interações sociais humanas, incluindo gerenciamento com SDN, modelos de confiabilidade e sistemas de reputação.

Já no ano de 2014, foi observado o grande potencial de redes oportunísticas criadas a partir de *smartphones* para tecnologias como 5G, redes veiculares e *IoT*, além de requisitos como eficiência energética. Alguns trabalhos chamaram a atenção sobre a grande quantidade de informação do contexto humano que pode ser coletada e extraída para modelos e arquiteturas de rede do futuro.

De 2014 a 2017, a visão do humano por trás de um dispositivo do usuário começou

a ser mais frequente [Thilakarathna et al., 2014, Barbera et al., 2014, Katsarakis et al., 2015, Afrasiabi e Guérin, 2016, Iosifidis et al., 2017, Zou et al., 2017]. Em 2014, houveram trabalhos de disseminação de conteúdo e *data offloading* baseados na noção de comportamento em redes sociais móveis. No segundo, houveram discussões da perspectiva do humano relacionada a mecanismos de incentivo e tarifação, segurança e privacidade. Já em 2016, termos como *people-centric* ou *people-to-people* passaram a ser mais comuns. Dito isso, o usuário passou a ser visto como um humano com características próprias, contrariando a visão antiquada do usuário apenas como portador de um dispositivo consumidor. Trabalhos relacionados à dinâmica de encontros, comunicação *multi-hop* baseada em aspectos sociais ou de mobilidade e outros foram propostos nesse período até 2017. Espera-se que num futuro próximo, surjam modelos e arquiteturas de rede cientes do humano. As gerações de rede do futuro deverão levar em conta o contexto, comportamento e outras características do humano, considerando aspectos como mobilidade, interações, perfil de tráfego, personalidade e conexões sociais. Os mesmos serão extensivamente discutidos nas seções a seguir, bem como exemplos de possíveis aplicações.

1.1.3. Cenários de aplicação

Nesta sub-seção, apresentamos cenários de aplicação do paradigma “ciente do humano”. Conforme o título do minicurso sugere, acreditamos que o papel dos humanos será o de facilitadores de redes de próxima geração. Sendo assim, o cenário de aplicação do paradigma “ciente do humano” é bastante amplo. Aspectos humano-comportamentais (apresentados a seguir, na Seção 1.2) como o contexto, interações sociais, perfil de tráfego, traços de personalidade, mobilidade, encontros e rotinas, podem ser aplicados na modelagem de soluções de redes. Arquiteturas, protocolos, tipos de comunicação e tecnologias relacionadas deverão ser “ciente do humano”. Dito isso, além de cenários tradicionais de redes e outros frequentes como desastres, falha da infraestrutura principal, grande concentração de pessoas, borda da rede, etc, o “ciente do humano” fomentará soluções em tecnologias promissoras. São exemplos delas *IoE/IoT*, Veículos Autônomos, *5G* e *6G*, Redes Ultra-densas e Dispositivos Vestíveis. Ao longo do minicurso apresentaremos exemplos e compartilharemos visões de como o paradigma poderá ser útil em cada caso.

Segundo [Aldini e Bogliolo, 2016], o sucesso de novos modelos para a Internet não seria o mesmo sem o envolvimento dos usuários no provisionamento de conteúdo, aplicações e serviços. A ideia central é de que eles possam ir muito além de consumir o que é disponibilizado pelos provedores. Essa visão participativa do usuário em novos modelos segue o caminho arquitetural evolucionário das redes móveis do futuro, presente também no conceito de Redes do tipo “Faça você mesmo” (do inglês *Do-it-Yourself - DiY Networks*). As soluções em redes *DiY* dependem da participação e engajamento dos indivíduos e antecederam a visão do humano por trás do dispositivo móvel. A importância do “ciente do humano” nos diferentes cenários citados acima é reforçada pela existência de uma grande infraestrutura disponível acessada por um número crescente de indivíduos portando dispositivos móveis e ao surgimento de diversas tecnologias para auxiliar no dia-a-dia das pessoas. Já que serviços de rede, aplicações e tecnologias relacionadas são destinadas aos humanos, podemos afirmar que suas expectativas podem ser melhor atendidas a partir do momento em que aspectos individuais do seu comportamento sejam levados em conta na arquitetura.

1.1.4. Objetivo e organização do minicurso

De acordo com o cenário exposto, este minicurso explora e desvenda o paradigma “ciente do humano” de forma abrangente e atualizada. Um dos objetivos é que os participantes se familiarizem com o tema, suas aplicações e desafios em diferentes áreas emergentes no domínio de redes. O minicurso também será importante para atualizar participantes sobre requisitos de tecnologias e modelos de comunicação do futuro, os quais oferecem diversas oportunidades de pesquisa.

O restante do texto está organizado conforme a seguir. Na Seção 1.2 serão discutidas particularidades essenciais do comportamento humano e social do usuário no contexto desse trabalho. Detalharemos cada aspecto do comportamento humano que pode influir no sucesso dos modelos de comunicação do futuro. Completando, o comportamento humano é relacionado com métodos de aquisição e processamento das informações de forma a auxiliar futuras arquiteturas de rede para melhor servir os usuários. A Seção 1.3 faz um paralelo do conceito com tipos de comunicação e discute como o paradigma aqui apresentado poderá satisfazer requisitos de tecnologias como *5G*, *IoT*, Redes Veiculares, Redes Ultra-Densas e Redes Centradas na Informação. Na Seção 1.4, modelos e soluções relacionados para *offloading de dados*, redes sociais *online*, segurança e privacidade. A Seção 1.5 traz detalhes de traces de usuários que podem ser utilizados para validação de pesquisas no contexto desse minicurso e *testbeds* de avaliação e experimentação. Já na Seção 1.6 serão apresentados diversos desafios identificados na literatura sobre o tema, fomentando novas oportunidades de pesquisa para estudantes. A Seção 1.7 apresentará lições adicionais com sugestões comentadas da literatura pertinente ao tema identificada em outras áreas como física, psicologia e sociologia. Por fim, na Seção 1.8 estão as considerações finais, e em seguida as referências bibliográficas.

1.2. Influência do contexto e comportamento humano do usuário

Conforme estatísticas apresentadas na Seção 1.1, num futuro próximo, espera-se cada vez mais portadores de dispositivos “inteligentes” como *smartphones* e aparelhos vestíveis. O comportamento de cada indivíduo irá impactar diretamente não só na forma como cada um requisitará recursos, mas também que tipos deles serão necessários em cada situação (contexto). Sendo assim, aspectos individuais do comportamento humano irão ditar exatamente como um indivíduo irá interagir como entidade da rede e também com outros usuários. Por isso, arquiteturas, modelos e soluções de redes do futuro deverão levar em conta aspectos do comportamento individual humano. Isso será motivado não apenas pelo oferecimento de melhor qualidade de experiência aos usuários, mas também pela otimização de parâmetros e aspectos de rede de uma forma mais natural.

A mobilidade foi um dos primeiros aspectos do comportamento dos usuários a ser identificado e explorado como potencial em soluções no âmbito de redes. Diversos trabalhos foram propostos, porém grande parte deles se restringiram a estudar particularidades específicas da mobilidade. Dentre elas, a repetitividade e frequência de contatos entre usuários e o tempo de ocorrência entre os mesmos. Pouco foi feito em relação à forma em que as pessoas instintivamente se movem, como os caminhos são escolhidos, que aspectos contextuais do ambiente ao redor podem influenciar, dentre outros fatores que podem impactar na mobilidade. Na perspectiva de rede, graças à ubiquidade dos dis-

positivos, os usuários se tornaram muito mais que uma entidade móvel. Individualmente, hoje uma pessoa gera e demanda uma grande quantidade de conteúdo, participa em redes sociais *online* e segue buscando constantemente por informações e serviços diversos que a mantenham cada vez mais “conectada”. Tudo isso é ligado a aspectos individuais que são relativamente estáveis ao longo da vida de um humano, influenciam diretamente no seu comportamento, e que variam em cada indivíduo. Interesses, necessidades, rotinas, ambiente, hábitos, traços de personalidade (de acordo com [McCrae e John, 1992], um elemento central para descrever uma pessoa), caráter, humor, dentre outros são determinantes não só para especificar o perfil de uma pessoa, mas também irão ditar que tipo de usuário de rede ela será. Sendo assim, o entendimento de padrões mais subjetivos do comportamento dos usuários, ou seja, o paradigma “ciente do humano” será essencial às soluções de rede do futuro.

O paradigma “ciente do humano” se refere ao objetivo de adicionar sentidos de percepção à rede, ligando-os com características humanas como interpretação, observação e reação relativas ao seu dia-a-dia e entidades que fazem parte dele. Sendo assim, uma rede “ciente do humano” do futuro será uma rede que leva em conta características comportamentais do ser humano para: (i) Prever necessidades conscientes ou subconscientes dos usuários; (ii) Se auto-adaptar à heterogeneidade e incerteza trazida pelo que é previsto; (iii) Ao mesmo tempo, oferecendo melhor qualidade de experiência.

A evolução dos dispositivos móveis permite a coleta de padrões de utilização de rede e outras informações dos usuários em qualquer momento ou local [Matarazzo et al., 2018]. Essas informações precisam ser correlacionadas com o comportamento humano e processadas. Com isso, será possível inferir e conhecer informações do comportamento e das interações em grande escala. A Figura 1.2 ilustra aspectos comportamentais diários dos humanos, relacionando com tecnologias e modelos de redes atuais e do futuro. A coleta, processamento e análise desses dados do comportamento humano irão alavancar tecnologias de redes como 5G, 6G, D2D, Redes Ultra-Densas, Carros Autônomos, dentre outras e também novas oportunidades de negócio.

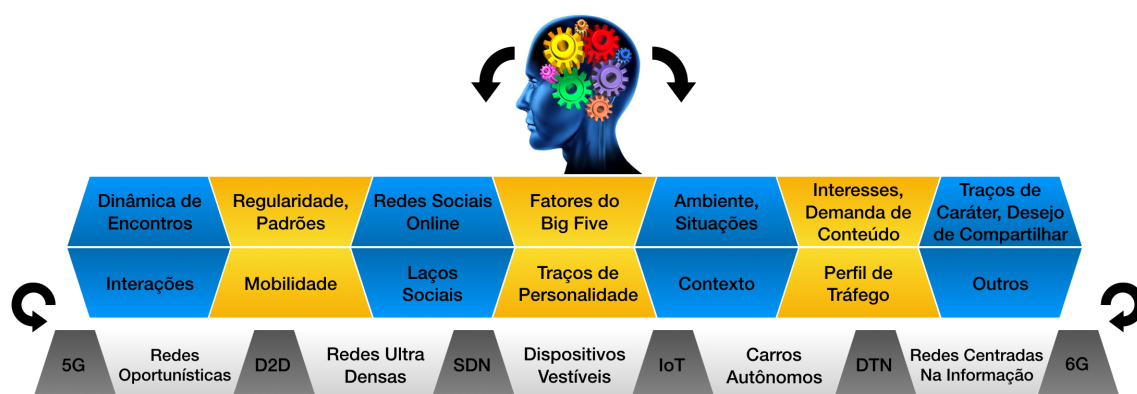


Figura 1.2. Representação do comportamento individual humano extraído a partir de aspectos multi-dimensionais como facilitadores de tecnologias de redes móveis do futuro.

A partir do paradigma “ciente do humano”, sistemas e serviços de rede podem se adequar às expectativas dos seus usuários finais. Isso se dá pela oportunidade de extrair e

prever o comportamento inerente de humanos por trás de dispositivos de rede num espaço multi-dimensional de tempo, localização, conteúdo, desejos e interesses. A relação do comportamento humano com aspectos como ‘quando’, ‘onde’, ‘o que’, ou ‘por que’ dirá muito sobre cada usuário individualmente. Sendo assim, o primeiro objetivo é identificar e extrair conhecimento do contexto e comportamento humano. Na sequência, trazer esse conhecimento para uma perspectiva de rede, analisando como o mesmo pode ser útil em novas tecnologias, paradigmas, protocolos, modelos, dentre outros.

Os aspectos que serão levantados aqui são comportamentos ditados pelo fato de sermos humanos. Dentre alguns exemplos, podemos citar: (i) As pessoas agem habitualmente como entidades semi-rationais, interagindo e se movendo rotineiramente num espaço geográfico previsível [Oliveira et al., 2016]; (ii) Ao escolher um itinerário, um indivíduo geralmente segue o caminho mais curto para seu destino, ou seja, o caminho disponível mais próximo do que seria uma linha traçada entre os dois pontos, exceto em ocasiões específicas como por exemplo visitação turística; (iii) O conjunto de itinerários e pontos de interesse frequentes são caracterizados pelo confinamento. Ou seja, as pessoas tendem a construir sua rotina diária em áreas particulares que também podem estar relacionadas com sua personalidade ou ligações sociais [Oliveira et al., 2016]; (iv) Um indivíduo tende a encontrar as mesmas pessoas por conta de interesses, personalidade, ou ambientes em comum [de Melo et al., 2015]. Todos esses aspectos podem ser entendidos através do estudo da mobilidade e ligados com outras características e expressões do comportamento humano. A Tabela 1.1 resume aspectos multi-dimensionais do comportamento humano que poderão alavancar serviços e tecnologias de rede do futuro. Nas sub-seções a seguir, esses aspectos são discutidos detalhadamente.

1.2.1. Aspectos do comportamento humano

Nessa seção discutiremos detalhadamente mobilidade, interações, traços de personalidade, perfil de tráfego, laços sociais, contexto e outros aspectos do comportamento humano capazes de alavancar arquiteturas de redes móveis. Na sequência apresentamos etapas pelas quais os dados do paradigma “ciente do humano” precisam passar para se tornarem úteis em soluções de redes do futuro e também em outras áreas.

1.2.1.1. Mobilidade

A popularização dos dispositivos móveis equipados com *GPS* abriu espaço para estudos mais profundos e granulares da mobilidade humana. Isso favoreceu a disponibilidade de *traces* de usuários e também as redes sociais baseadas em localização, das quais podem ser extraídas informações de mobilidade [Silva et al., 2016]. Os primeiros trabalhos a estudar trajetórias individuais mostraram que a mobilidade humana não é randômica, mas sim apresenta regularidade espacial e temporal [Gonzalez et al., 2008]. Dito isso, pessoas tendem a frequentar os mesmos lugares com duração temporal similar e com certa regularidade. Tudo isso acontece devido às rotinas diárias do comportamento humano, tornando a mobilidade algo previsível. Por exemplo, um funcionário que trabalha no escritório de uma empresa seguirá uma rotina onde pela manhã num determinado horário se deslocará de casa para o trabalho (utilizando sempre o menor caminho possível ou o de menor duração por conta de elementos externos como tráfego). Durante a permanência no

Tabela 1.1. Aspectos Multi-Dimensionais do Paradigma “Ciente do Humano”.

Aspecto	Descrição	Exemplos
Contexto	Contexto lógico ou físico no qual um indivíduo está inserido. Ambiente, situações vividas, padrões de visita, preferências (aplicações, áreas geográficas, ferramentas, interfaces) e outros.	Padrões utilizados como entrada para recomendação e disseminação de propaganda ou serviços direcionados; Uso do contexto para melhorar algoritmos de predição [Jeong et al. 2016], [Cuttone et al. 2018].
Interações	Estudo dos encontros físicos com outros indivíduos, incluindo regularidade, duração e similaridade.	Estimular a utilização de aplicações específicas, data offloading, encaminhamento de dados e implantação de novos tipos de redes sociais.
Laços Sociais	Extração de laços sociais e encontros em redes sociais online, incluindo regularidade, duração, similaridade e encontros físicos.	Modelos de confiabilidade baseados em relações sociais humanas.
Mobilidade	Informação contextual da mobilidade humana. Localização, similaridade, pontos de interesse, perfil de deslocamento, detecção de rotinas, regularidade, confinamento, e outros.	Estimular comunicação D2D; Alocação de recursos na borda da rede; Alocação de recursos em redes celulares para lidar com heterogeneidade e incertezas; Melhorar implantação de infraestrutura de rede.
Perfil de Tráfego	Tipo de conteúdo de interesse e quantidade de tráfego associados ao perfil. Inclui aplicações, sites e outros para oferecer recomendações e modelar o serviço de rede.	Sistemas de recomendação e disseminação de conteúdo gastando menos recursos de rede; Alocação de recursos em redes celulares para lidar com heterogeneidade e incertezas; Melhorar implantação de infraestrutura de rede.
Traços de Personalidade	Influência dos traços de personalidade no uso da rede, dos dispositivos e no contexto social. Identificação de ações e comportamento baseados na personalidade para predições.	Uma pessoa aberta a experiências tende a colaborar com compartilhamentos e possui mobilidade dinâmica, facilitando predições.
Outros	Outros aspectos ainda carecem de identificação. Informações e dados importantes podem estar escondidos em diferentes hábitos ou rotinas inerentes à vida humana.	A frequente atitude de compartilhar (opiniões, amigos, fotos, etc) já vem sendo explorada por redes sociais online e empresas como Airbnb, Uber, etc; Traços de caráter podem ser analisados por provedores de serviços para estimular compartilhamento.

trabalho, a mobilidade é restrita. Ao fim do expediente o indivíduo retorna para casa, geralmente seguindo o mesmo caminho (ou um conjunto reduzido deles). Eventuais paradas em estabelecimentos próximos de casa (ex: supermercado) são características do confinamento, onde a pessoa dá preferência a locais próximos. Esses deslocamentos podem ser esperados, assim como é possível prever com grande taxa de acerto que diariamente, num determinado período não haverá mobilidade já que o usuário estará em casa dormindo.

Um dos pilares do paradigma “ciente do humano” em redes móveis é justamente utilizar o conhecimento do perfil de deslocamento individual dos usuários para prever e modelar dinamicamente os serviços de rede a serem oferecidos a eles. Complementarmente, o perfil de mobilidade pode ser associado com outros aspectos que serão discutidos a seguir e que também influenciam na forma como as pessoas se deslocam. A associação dessas informações dará origem a conjuntos de dados valiosos e mais ricos dos que os já disponíveis nos últimos anos.

1.2.1.2. Interações

No contexto humano, o entendimento das interações físicas entre as pessoas pode alavancar uma série de serviços de rede e também novas aplicações e oportunidades de negócio. Por conta de aspectos diversos como o confinamento, movimentos repetidos, rotinas, personalidade, laços sociais, dentre outros, os humanos tendem a encontrar fisicamente (com uma certa repetição) determinados grupos de pessoas [Wang et al., 2009]. Isso se dá

por situações diversas, por exemplo ambiente de trabalho ou universidade em comum, ou por preferência de ambientes de lazer que são escolhidos de acordo com a personalidade individual.

No caso das interações, o estudo de aspectos como a regularidade, duração e similaridade são essenciais para arquiteturas de rede do futuro. Essas informações são úteis em diversas perspectivas do ponto de vista de rede, como para estimular comunicação *D2D* [Nunes et al., 2016], gerenciar recursos de rede, realizar *offloading* de dados [Chen e Yang, 2016], disseminar conteúdo direcionado a determinados grupos em comum, construir mecanismos de incentivo e reputação, dentre outros. Ao prever interações entre os usuários (e consequentemente entre os dispositivos que os mesmos carregam), recursos de redes móveis podem ser orquestrados mais eficientemente.

1.2.1.3. Traços de personalidade

Nos últimos anos, esforços de pesquisa estão sendo direcionados à um campo da psicologia capaz de trazer informações úteis às arquiteturas de redes móveis do futuro - o estudo dos traços de personalidade. O entendimento desses aspectos já vem sendo usado em outras áreas como redes sociais e aplicações móveis, sistemas de recomendação, interação humano-computador e processamento de vídeo. O modelo *Big Five* [McCrae e John, 1992], dividiu traços de personalidade humana em cinco dimensões: (i) Neuroticismo (*Neuroticism* ou Instabilidade Emocional): tendência a ser preocupado, vulnerável e temperamental; (ii) Extroversão (*Extraversion*): ser comunicativo, enérgico, afirmativo; (iii) Agradabilidade (*Agreeableness*): ser bondoso, afetivo, simpático, disposto a ajudar; (iv) Nível de Consciência (*Conscientiousness*): um indivíduo que gosta de fazer planos ao invés de ser espontâneo, organizado; (v) Abertura à Experiência (*Openness to Experience*): pessoa com interesses amplos, imaginativa.

Os traços de personalidade das pessoas estão ligados, dentre outros aspectos, aos seus gostos, hábitos e rotinas [Kopacz, 2005], tipos de ambientes favoritos, relações sociais, desempenho intelectual e profissional [Judge et al., 2006], expressões faciais comuns [Teijeiro-Mosquera et al., 2015], capacidade de lidar com emoções (ansiedade, stress, etc) [Orzeck e Lung, 2005], liderança ou passividade [De Hoogh et al., 2005], aspectos culturais [Favaretto et al., 2017] e até ao perfil de mobilidade e lugares que elas visitam [Courneya e Hellsten, 1998]. Por exemplo, uma pessoa extrovertida e aberta a experiências irá gostar de lugares mais movimentados e barulhentos como bares e restaurantes, terá a tendência de buscar novos ambientes e sair após o trabalho ou durante os finais de semana. Por outro lado, pessoas mais reservadas e introvertidas irão a locais mais acolhedores e calmos ou irão preferir permanecer em casa assistindo um filme.

O que podemos tirar dos traços de personalidade é a possibilidade de melhorar significativamente os serviços e tecnologias para cada tipo individual de personalidade, aumentando o grau de satisfação. Por exemplo, sugestões de novos amigos em redes sociais móveis baseadas na afinidade de personalidade e interesses, aplicativos de recomendação que sugerem lugares e experiências mais adequadas, dentre outros. Ao ligar traços de personalidade com perfis de mobilidade, uma grande variedade de informações sobre tendências ou rotinas de uma pessoa no contexto de rede pode ser alavancada. Pesqui-

sas na área de psicologia relacionam a personalidade a uma série de comportamentos que podem ser previstos, tanto individualmente como na interação social do indivíduo. Apresentaremos alguns exemplos e referências na Seção 1.7

1.2.1.4. Perfil de tráfego

Na visão das arquiteturas de redes, o perfil de tráfego dos usuários sempre foi relacionado com parâmetros e métricas de desempenho, tais como vazão (mínima/média/máxima), requisitos de atraso, taxas de *download* e *upload*, dentre outras. No entanto, o perfil de tráfego de um usuário de rede está ligado diretamente com características humanas [Oliveira et al., 2016, Chen et al., 2017a]. Dentre elas, os traços de personalidade (que pode determinar por exemplo aplicações, sites ou tipo de conteúdo favoritos), idade, gênero, contexto em que se encontram ou outros possíveis como o humor ou o quanto ele gosta de “compartilhar”. O correlacionamento entre aspectos humanos e o perfil de tráfego pode alavancar e auxiliar em desafios de rede, além de fomentar novas aplicações. Por exemplo, estímulo à disseminação de conteúdo baseada no conhecimento do perfil de tráfego em determinado ambiente.

1.2.1.5. Laços sociais

A disseminação dos dispositivos móveis e das redes sociais online favorecem o entendimento dos laços sociais humanos. Diversos estudos de outras áreas como sistemas de recomendação já tratam informações de redes sociais móveis (ex: informação de *check-in*) para prever deslocamento de usuários e recomendar pontos de interesse [Wagih et al., 2017]. No contexto de redes, a extração de informações dos encontros e laços sociais, incluindo regularidade, duração, similaridade e encontros físicos pode auxiliar em diversos desafios. Dentre eles, dimensionamento das redes e gerenciamento de recursos, estabelecimento de conexões baseadas na confiabilidade dos laços sociais, estimulação de comunicação *D2D* [Nunes et al., 2016], *data offloading* [Thilakarathna et al., 2017] protocolos de roteamento orientados aos encontros, dentre outros.

1.2.1.6. Contexto

Informações do contexto em que uma pessoa se encontra estão diretamente ligadas à forma como ela irá demandar recursos e que tipos de requisitos de redes irá precisar [Feki et al., 2016]. Dentre as informações que podem ser consideradas, estão o tipo de ambiente físico, o padrão de mobilidade ou tipo de deslocamento no momento e as preferências individuais (determinados pontos de interesse, ferramentas, aplicações, etc). O correlacionamento de informações de contexto com outras características do comportamento humano irá servir para prever como a rede deverá se comportar para oferecer melhor qualidade de experiência. Em áreas geográficas remotas por exemplo (na borda da rede), interações e compartilhamento podem ser encorajados. Um determinado tipo de atividade (ex: deslocamento em automóvel) pode ser previsto para moldar os requisitos da rede ou estimular comunicação *D2D* [Ren et al., 2015].

1.2.1.7. Outros

Acreditamos que outros aspectos do comportamento humano ainda poderão ser identificados e relacionados com a forma na qual os usuários interagem através de seus dispositivos móveis. Aspectos como humor, traços de caráter, desejo de compartilhar informações, dados sócio-demográficos, dentre outros podem representar fontes de dados a serem levadas em conta em diversas camadas de rede.

1.2.2. Aquisição e processamento de informações do contexto humano

Na seção anterior foram discutidos quais aspectos do comportamento humano podem ser úteis às arquiteturas e modelos de rede do futuro. A seguir, serão apresentadas as etapas (coleta, armazenamento/processamento, modelagem, extração de conhecimento, análise e validação) que esses dados precisam passar até se tornarem úteis nesse contexto. A Figura 1.3 resume os passos necessários que são discutidos detalhadamente a seguir.

Primeiramente, é necessária a aquisição dos dados do contexto e comportamento do usuário. As fontes (discutidas detalhadamente na Seção 1.5) são variadas e incluem redes sociais online, formulários de pesquisa e sensores de dispositivos como *smartphones*. Aplicações, *APIs* específicas ou *crawlers* são os métodos mais comuns para coletar dados nesse cenário. No caso das *APIs*, as informações podem ser coletadas quase em tempo real ou através de requisições (sob-demanda). Redes sociais como *Facebook*, *Twitter* e *Foursquare* disponibilizam *APIs* [fac, 2018] para o desenvolvimento de aplicações que podem acessar dados compartilhados pelos seus usuários. Já quando não há um meio pré-definido (como no caso das *APIs*) para requisitar os dados, são utilizados *crawlers* capazes de minerar informações de interesse de páginas web (ex: analisando tags HTML ou XML [Almendros-Jiménez e Becerra-Terón, 2015]). Outros métodos são possíveis, como o do *OpenStreetMap*, onde os dados são selecionados através de uma interface *web* e baixados num arquivo *zip* contendo as coordenadas. Já no caso de dados relativos aos traços de personalidade, o formulário do *Big Five* [Goldberg, 1999] desenvolvido por psicólogos é bastante utilizado.



Figura 1.3. Etapas pelas quais os dados do contexto e comportamento humano precisam passar para serem utilizados posteriormente em soluções de redes do futuro e também em outras áreas.

Sequencial à aquisição, os dados precisam ser processados (em alguns casos, enriquecidos) e armazenados. Dada a potencial grande quantidade de informação gerada, plataformas de armazenamento seguras, escaláveis e tolerante a falhas devem ser utilizadas. Esse é um dos pontos críticos, já que envolve a privacidade dos usuários e também é preciso ser capaz de lidar com múltiplas requisições de sistemas baseados nesse produto.

A depender dos tipos de dados (publicações em redes sociais, coordenadas geográficas obtidas por *GPS*, tomadas de decisão, etc), associações e agrupamento entre eles poderão ser necessários para que informações mais ricas sejam disponibilizadas. De forma complementar, em alguns casos, os dados apresentam lacunas, erros semânticos e inconsistência. Portanto, talvez seja necessário tratar as falhas, normalizar e preencher informações ausentes. Por fim, como os dados representam um agregado de diversas dimensões e tipos heterogêneos, antes da análise, as características desejadas precisam ser selecionadas.

Finalizando a etapa de gerenciamento dos dados (representada pelas 3 primeiras sub-etapas da Figura 1.3), é necessária a modelagem num formato onde possam ser extraídas informações espaço-temporais ou da relação entre componentes heterogêneos desses dados. No caso da modelagem de comportamento espaço-temporal humano relacionados com contexto ou interações (ex: laços sociais), os grafos são o método mais utilizado [Das et al., 2017]. Nesse caso, os vértices representam usuários numa rede ou lugares que eles visitaram, enquanto as arestas conectam os vértices na ocorrência de encontros ou quando lugares são visitados sequencialmente. Séries temporais de pontos ordenados cronologicamente ou trajetórias espaço-temporais também são exemplos de modelagem. Por exemplo, um perfil de mobilidade com coordenadas geográficas modelado como um conjunto de trajetórias espaço-temporais representando o deslocamento de um indivíduo.

Após a modelagem e anterior à validação (produto final), é necessário extrair conhecimento e analisar os dados. Essas etapas são essenciais para obter uma melhor visão das informações disponíveis. Na extração de conhecimento são avaliadas as propriedades dos dados e que tipos de problemas podem ser tratados. Isso é essencial para que as técnicas de análise e identificação de padrões sejam escolhidas propriamente. Dentre elas, podemos citar mineração de dados e análise estatística ou visual. Quando se trata de informações do contexto e comportamento humano (ou seja, informações espaço-temporais diversas), há diversos desafios na associação dos dados [Quintero et al., 2016]. Descobrir relação e associação entre conjuntos heterogêneos de dados, detectar objetos anormais e classificá-los não são tarefas triviais. Na extração de conhecimento, técnicas de aprendizado de máquina podem ser utilizadas. Citando um exemplo de redes sociais online, textos de publicações de usuários podem ser relacionados com seus traços de personalidade ou emoções para prover um conjunto de dados valioso para arquiteturas de redes móveis [Alsadhan e Skillicorn, 2017]. Para isso, são necessárias técnicas de mineração de dados textuais e processamento de linguagem. Conforme mostrado na Figura 1.3, a extração de conhecimento e análise podem ser repetidas até a validação. Com isso, surgem adaptações, novas ideias sobre os dados e também identificação de erros.

Por fim, a validação dos dados é realizada cruzando as informações obtidas com o que é chamado de *ground truth*, que podem ser dados oficiais obtidos tradicionalmente ou *traces* detalhados com informações mais precisas que permitam validar uma estratégia. Estatísticas oficiais ou pesquisas com baixa margem de erro [Pentland, 2012] são um exemplo de *ground truth*. Em alguns casos, a depender do tipo de informação presente no conjunto final gerado, esses dados oficiais não estarão disponíveis. Com isso, são necessários experimentos, simulação ou emulação em diversos cenários de condições variáveis para que a validação seja feita. Com o produto final da validação, espera-se o desenvolvimento “ciente do humano” de serviços, arquiteturas, aplicações, tecnologias

relacionadas, dentre outros.

1.3. Redes móveis e o papel da comunicação “ciente do humano”

Nessa seção, o paradigma “ciente do humano” é relacionado com tipos de comunicação fundamentais na evolução das redes móveis como *Mesh*, *MANETs*, *DTNs* e Redes Centradas no Usuário. Ademais, desafios tradicionais nesse contexto são abordados sob o ponto de vista “ciente do humano”. Completando a seção, o paradigma foco do minicurso é discutido diante de conceitos, tecnologias e modelos de comunicação de redes móveis, dentre eles *5G*, *IoT* e Redes Veiculares.

1.3.1. O paradigma “ciente do humano” e tipos de comunicação

Em redes móveis, diversos tipos de comunicação como Redes *Mesh*, *MANETs*, Redes Tolerantes ao Atraso (*DTNs*), Redes Oportunísticas, Dispositivo-a-Dispositivo (*D2D*) e Redes Centradas no Usuário foram apresentados ao longo dos anos [Conti et al., 2015, Aldini e Bogliolo, 2016, Akyildiz et al., 2016]. As principais motivações, foram a redução de sobrecarga nas redes, aumento de capilaridade, melhor reuso do espectro, fomento de novos tipos de aplicações, e atuação em cenários específicos (ex: desastres, na borda da rede ou alta densidade de nós). Cada proposta contribuiu com ideias relevantes e evolucionárias, no entanto, houveram dificuldades na adoção, disseminação e implementação, sobretudo por parte da indústria [Haus et al., 2017]. Em alguns casos, isso aconteceu por complicações com segurança e privacidade, falta de alternativas de código aberto, ausência de incentivos específicos e de modelos de negócio bem definidos, ou por estarem distantes da expectativa dos usuários.

Por exemplo, as redes *ad hoc* que surgiram há cerca de 30 anos não foram utilizadas massivamente. Isso se deu por falta de incentivos claros tanto aos usuários como organizações para que elas fossem criadas. Com isso, modelos mais simplificados, como as redes *MESH*, surgiram [Aldini e Bogliolo, 2016]. Porém, da mesma forma, a falta de uma percepção clara sobre os incentivos sociais, econômicos e técnicos não sugeriram como a adoção das redes *MESH* poderia prosseguir. No caso da comunicação centrada nos usuários, requisitos dos mesmos passaram a ser uma preocupação, mas as discussões seguiram focadas em atributos de rede [Aldini e Bogliolo, 2016]. Aspectos de mobilidade ou das interações sociais (característicos do comportamento humano) foram utilizados, porém mais no contexto de distribuição de credenciais de acesso ou de mecanismos de *handover*. Esse cenário promete mudar com a disseminação dos dispositivos móveis e com a percepção das soluções passando a considerar outros fatores além da perspectiva de rede, geralmente focada no desempenho.

Nos dias de hoje, os usuários finais se conectam à internet por uma variedade de métodos de acesso. Geralmente, o segmento final (última milha) é provido por tecnologias de curto alcance (ex: *WiFi*), cada vez mais utilizadas nos dispositivos móveis. De acordo com a Cisco [cis, 2017b], até 2021 haverá cerca de 5.5 bilhões (71% da população mundial) de usuários de dispositivos móveis, sendo *smartphones* mais de 50% do número global. A Figura 1.4 prova a representatividade individual dos usuários, dada previsão que num futuro próximo os mesmos irão gerar e consumir cerca de 4 vezes mais tráfego que corporações e governo. Com o surgimento de novos paradigmas como as

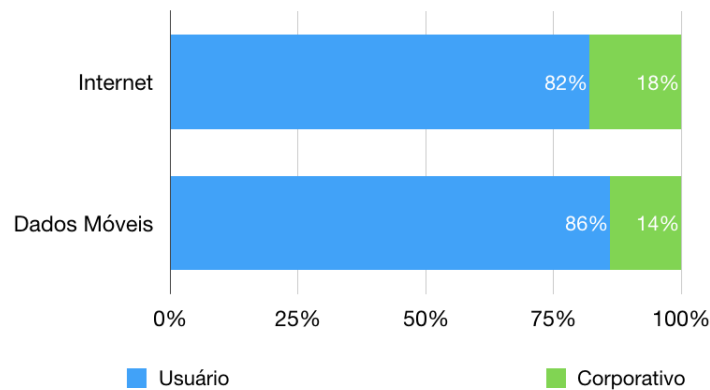


Figura 1.4. Panorama geral de tráfego previsto até o fim de 2021 evidenciando a representatividade do usuário na Internet. Dados provenientes de [cis, 2017b]

Redes Definidas por Software, e o *WiFi Direct*, essa crescente população de usuários de dispositivos móveis poderá alavancar a adoção massiva de tipos de comunicação como o *D2D* e favorecer aplicações como veículos autônomos. Simultaneamente, será possível atacar problemas antigos como a sobrecarga nas redes móveis, alcançar objetivos de tipos de comunicação anteriores e fomentar o surgimento de aplicações e redes sociais baseadas na interação direta entre dispositivos [Pambudi et al., 2017]. Ressaltamos que um dos pontos-chave para que isso ocorra é o estudo do comportamento humano e a incorporação de aspectos do mesmo nas soluções para redes móveis do futuro.

Não é novidade que as redes móveis disponíveis hoje foram criadas para atender as pessoas. No entanto, nem sempre percebemos que os anseios desses indivíduos foram levados em conta na arquitetura da rede. Quando se trata de tipos de comunicação, o sucesso de modelos atuais ou futuros dependerá diretamente da personalização dinâmica do serviço, ou seja, ele deve ser capaz de se adaptar ao perfil do usuário (humano) para melhor atendê-lo. Dito isso, a noção que o paradigma “ciente do humano” traz é que *as redes (através de suas soluções) deverão levar em conta aspectos dos usuários (humanos) em diversas camadas do modelo OSI*. Os tipos de comunicação do futuro deverão usar técnicas de predição do comportamento e rotinas do usuário para garantir uma melhor qualidade de experiência. Avanços nessa questão do humano por trás do dispositivo não apenas beneficiam os usuários (através de uma melhor experiência) mas também os provedores, já que a satisfação com o serviço atrai mais consumidores.

Nos trabalhos de pesquisa com propostas relativas aos tipos de comunicação (mais antigos ou recentes) em redes móveis, os desafios identificados se repetem em muitos casos [Aldini e Bogliolo, 2016]. A seguir, relacionamos desafios identificados nesse contexto com aspectos do paradigma “ciente do humano” e mostramos como os mesmos podem auxiliar no sucesso desses tipos de comunicação.

- **Eficiência Energética** - Apesar da coleta de informação, do processamento e da capacidade de comunicação e armazenamento dos dispositivos móveis vir melhorando ao longo dos anos, ainda faltam estudos para que tais atividades sejam realizadas levando em conta o baixo consumo [Masucci et al., 2017]. A eficiência energética em tipos de comunicação baseados na participação do usuário está di-

retamente ligada a tarefas que consomem muito recursos. Dentre elas, a coleta de informação de mobilidade, o encaminhamento e armazenamento de mensagens, o *lookup* de outros dispositivos e redes, e o roteamento (muitas vezes adaptado de algoritmos de redes convencionais). Em todos os casos, informações do contexto humano podem ser úteis para prever situações e otimizar o consumo de energia. Por exemplo, utilizando o conhecimento da dinâmica de encontros de um usuário para encaminhar as mensagens a outros dispositivos que tenham mais chance de levá-las ao destino final (reduzindo o *overhead* na rede). Já o *lookup* pode ser realizado em momentos específicos, ao identificar mobilidade do usuário, prevenindo que o mesmo irá em breve necessitar de *handover*. A coleta de informações de mobilidade de usuários pode também ser reduzida com a identificação de hábitos rotineiros dos mesmos, reduzindo a frequência de amostragem do *GPS*, que tende a drenar rapidamente a bateria de um dispositivo móvel [Katsikouli et al., 2017].

- **Mecanismos de Incentivo e Reputação** - Um dos grandes desafios em tipos de comunicação baseados na participação do usuário são os mecanismos de incentivo e reputação [Aldini e Bogliolo, 2016]. Nesse cenário, cada indivíduo possui um dispositivo com recursos limitados, dentre eles banda, bateria e memória. Por conta dessas limitações, a colaboração precisa ser estimulada de alguma forma, já que o sucesso desses tipos de comunicação é diretamente proporcional à adesão de usuários. Trabalhos anteriores como [Mantas et al., 2017] resumem benefícios propostos como os financeiros e de reciprocidade. No entanto, os incentivos podem funcionar de forma mais natural “imitando” o comportamento social e humano. Efetivamente sabemos que humanos tendem a cooperar uns com os outros quando há algo em comum. Por exemplo, as interações sociais e a dinâmica de encontros podem ser utilizadas para criar mecanismos de incentivo entre usuários que naturalmente compartilhem similaridades no seu dia-a-dia. O mesmo vale para sistemas de reputação, já que usuários que possuem características em comum tendem a não agir de forma gananciosa com os seus semelhantes (no mundo *P2P*, os conhecidos *leechers*).
- **Mobilidade** - Como o alcance das redes sem fio depende de fatores como obstáculos, potência dos transmissores e também de interferências externas, em redes móveis a mobilidade é um ponto chave a ser considerado. Em trabalhos anteriores, a mobilidade foi estudada e associada com tarefas orientadas à rede, como *handover* [Zang et al., 2017] e eleição de nós provedores. No entanto, a mobilidade no contexto humano é uma característica que tende a seguir rotinas e também ser associada a uma dinâmica de encontros [Gonzalez et al., 2008, Oliveira et al., 2016]. Essas informações podem ser utilizadas para prever requisitos de rede e oferecer melhor qualidade de experiência. Por exemplo, ao prever o caminho no qual um determinado usuário irá seguir num horário, a rede pode alocar dinamicamente os recursos necessários [Oliveira e Viana, 2014].
- **Roteamento** - Em tipos de comunicação de redes móveis, o roteamento é um dos tópicos que mais possui contribuições. Mais recentemente, estratégias de roteamento estão se aproximando do comportamento humano, sobretudo no que diz respeito ao estudo de mobilidade [Altahrawi et al., 2017]. Características e informações

da rotina dos usuários podem ser exploradas para otimizar a taxa de entrega de mensagens e diminuir o *overhead* da rede [Nunes et al., 2016]. Por exemplo, roteamento baseado nos interesses do usuário e realizado de forma oportunística para alguns tipos de aplicações.

- **Segurança, Privacidade e Confiabilidade** - A rotatividade de usuários conectados à uma rede móvel e a grande quantidade de informação que trafega por ela também traz desafios relativos à segurança e privacidade dos dados dos usuários. Mecanismos de autenticação e autorização para garantir que indivíduos mal intencionados não interfiram também precisam ser desenvolvidos nesse sentido [Aldini e Bogliolo, 2016]. No caso do gerenciamento de confiabilidade, a noção de interações em redes sociais móveis já vem sendo utilizada [Wang et al., 2018]. Redes e círculos de confiança podem ser formados oportunisticamente a partir de encontros de grupos de pessoas com interesse comum, como estudantes numa sala de aula por exemplo. A confiabilidade pode ser propagada de forma direta ou indireta entre os usuários. Tudo isso deve ser feito se preocupando com a privacidade dos usuários, evitando que suas informações pessoais sejam utilizadas maliciosamente.
- **Gerenciamento de Recursos** - O gerenciamento e alocação de recursos em redes móveis também pode ser favorecido pelo conhecimento de padrões e rotinas do comportamento humano aliado a tecnologias como SDN e NFV. Por exemplo, um estudo de frequência de usuários num determinado ponto de interesse em espaços temporais distintos fornece dados úteis para gerência dinâmica de recursos [Oliveira e Viana, 2014]. Dados de status do dispositivo do usuário como tempo de bateria, largura de banda disponível ou qualidade do sinal também podem ser obtidas ou previstas para auxiliar em decisões nesse contexto. Em [Ning et al., 2016], os autores consideram laços sociais dos usuários para melhorar a conectividade, serviço e qualidade de experiência, gerenciando recursos com SDN.

1.3.2. Conceitos e redes móveis

Essa seção discute a comunicação “ciente do humano” diante de conceitos, tecnologias e modelos de comunicação de redes móveis. São eles: *5G*, *IoT*, Redes Veiculares, Redes Ultra-Densas e Redes Centradas na Informação.

1.3.2.1. 5G

O *5G*, a próxima geração de redes móveis surgiu com a premissa de prover acesso a um número crescente de usuários num cenário atual de sobrecarga e limitações. Os desafios vão desde alocação de espectro (reutilização mais eficiente e alocação de novo espectro) à oferecer suporte a novas aplicações e tipos de dispositivos.

Em [Akyildiz et al., 2016], os autores relacionam dez tecnologias facilitadoras para que objetivos do *5G* sejam alcançados, mantendo ainda compatibilidade com gerações anteriores. São elas: (i) Redes Móveis Definidas por Software (*WSDN*); (ii) Virtualização de Funções de Rede (*NFV*); (iii) Comunicação em Espectro de Altas Frequência (*Millimeter Wave Communications*); (iv) *MIMOs* (*Multiple Input Multiple Output*) Simultâneos

(utilização de *streamings* de dados simultâneos para alcançar maiores taxas de transmissão); (v) Ultra-densificação; (vi) Computação Móvel em Nuvem; (vii) Internet das Coisas; (viii) Comunicação Dispositivo-a-Dispositivo (*D2D*); (ix) Comunicação Verde (sustentável) e (x) Técnicas de *Radio-Access*. A intuição é que no futuro (tanto no 5G como no 6G), informações do contexto e comportamento humano sejam combinadas com tecnologias como as citadas para chegarmos mais próximos da especificação, evitando soluções sub-ótimas como no LTE [Lauridsen et al., 2017]. No caso do 5G, na Figura 1.5 são resumidos alguns dos requisitos potencialmente possíveis de atingir com soluções “ciente do humano”. A seguir são discutidos quais aspectos do comportamento humano podem ser aproveitados nesse contexto e como.

No 5G, espera-se uma redução de consumo energético na ordem de 1000 vezes por bit transmitido [Akyildiz et al., 2016]. Essa meta está relacionada não apenas com as limitações de bateria de dispositivos tradicionais, mas outros ainda mais restritos como sensores de *IoT*. Nesse sentido, por exemplo, o “ciente do humano” pode ser estudado para favorecer comunicação dispositivo-a-dispositivo com gerência de recursos mais eficiente. Além de consumir menos energia (já que as conexões são de curto alcance), com o *D2D* também é possível atingir maiores taxas de transmissão (requisito do 5G). De fato o *D2D* já vem sendo utilizado em *IoT*, *V2V* e dispositivos vestíveis [Kozioł et al., 2017]. O *D2D* ciente do humano ainda é capaz de influenciar na diminuição da latência (requisito do 5G), dos atuais *15ms* para cerca de *1ms* e melhorar a conectividade de usuários na borda da rede (requisito do 5G) [Akyildiz et al., 2016]. Nesse caso, ideias de compartilhamento de conexão propostas em redes centradas no usuário são úteis, favorecendo o aumento da capilaridade na borda. Além das intuições citadas, o *D2D* “ciente do humano” é capaz de estimular melhor reuso do espectro. Já que as conexões nesse contexto possuem alcance reduzido, há um ganho (diminuição) também no número de saltos. A disseminação da comunicação dispositivo-a-dispositivo é útil em cenários diferentes como *offloading* de dados, mas também fomenta novas aplicações em arquiteturas de redes do futuro. O

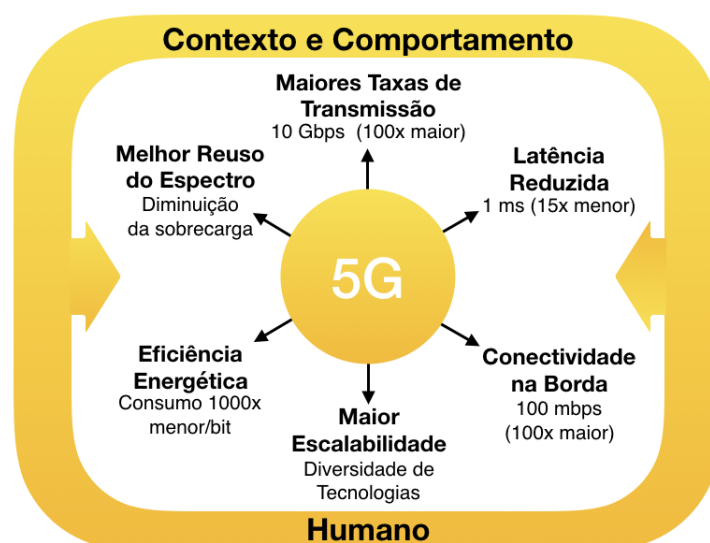


Figura 1.5. Requisitos do 5G que podem ter soluções favorecidas pelo conhecimento de informações do Contexto Humano combinadas com tecnologias de redes móveis do futuro.

cenário do 5G é de uma crescente quantidade de dispositivos (com requisitos diferentes) sendo conectados à rede, da qual espera-se também uma maior escalabilidade. *IoT*, Redes Veiculares, Veículos Autônomos, dentre outras tecnologias, trazem desafios que irão demandar modificações nas camadas de rede [Akyildiz et al., 2016]. Por exemplo, na camada física, utilização de informações do perfil de tráfego do usuário e do contexto (ex: tipo de ambiente em que ele se encontra) para melhor aproveitamento do espectro. Nesse caso surgem mecanismos de transmissão de dados e sinalização “cientes do humano” mais eficientes, já que os recursos são melhores orquestrados à medida em que se conhece o usuário (e as suas intenções) do serviço. Nas camadas de rede e de transporte, algoritmos de roteamento levando em conta aspectos como mobilidade, traços de personalidade e outros podem oferecer melhor taxa de entrega e menor *overhead*. Para usuários em deslocamento, o histórico de mobilidade fornece dados para prever situações como de *handover* ao longo de rotas prováveis, onde recursos devem ser reservados.

Em termos de gerenciamento, apesar do SDN já ser uma tecnologia um pouco mais consolidada, o WSDN ainda requer muitos avanços [Akyildiz et al., 2016]. No mesmo trabalho citado anteriormente, os autores reforçam a necessidade de um *framework* de gerenciamento capaz de gerenciar a mobilidade, interação entre protocolos e tecnologias e garantir *QoS* e serviços móveis. Nesse caso, uma intuição é que aspectos humanos relacionados às interações sociais ou relações de confiança no contexto social servem como parâmetro para garantir qualidade de serviço e experiência.

1.3.2.2. Internet das coisas (IoT)

Do ponto de vista de redes de computadores, há uma série de desafios a serem solucionados relativos à implementação de serviços de dados para *IoT* (do inglês *Internet of Things*) [Tayeb et al., 2017, Hejazi et al., 2018]. Uma das premissas desses serviços é que eles estejam disponíveis sob-demanda aos seus usuários, em qualquer momento e independente da localização. A *IoT* tem uma forte ligação com aspectos do comportamento humano, já que em muitos casos, sensores de *IoT* são utilizados para obter informações de contexto. Por esse motivo, é natural que o paradigma “ciente do humano” em redes seja útil para modelar os serviços de comunicações que atendem aos objetivos de uma infraestrutura de *IoT*. Nesse caso, o “ciente do humano” além de ser útil na perspectiva de serviços de rede, é capaz de auxiliar a modelagem de soluções de *IoT* que englobem dados sensoriais do contexto e/ou comportamento de um indivíduo.

De acordo com [Barnaghi e Sheth, 2016], num futuro próximo, a *IoT* funcionará como uma extensão das redes atuais. Cada objeto irá interagir através de uma infraestrutura de rede e interoperar com serviços de dados e comunicações, recursos disponíveis e outros objetos. Dessa forma, a grande quantidade de dados gerados e de novos dispositivos conectados causará impactos nas redes. A análise dos dados e a resposta aos eventos em muitos casos precisará ser em tempo real. Com isso, redes eficientes, sensíveis ao contexto e de alta velocidade são essenciais para o futuro da *IoT*.

As arquiteturas de rede que servem à *IoT* são heterogêneas e incluem modelos como o *WiFi*, Redes *Mesh*, Comunicação Dispositivo-a-Dispositivo, Redes Móveis (3G/4G/5G) e Redes de Sensores. Na literatura, alguns trabalhos estudam propostas de

protocolo de roteamento específicos para *IoT*. Em [Qiu et al., 2018], os autores apresentam uma arquitetura de *IoT* baseada em camadas. A camada de rede serve para a comunicação entre os sensores (ex: som, temperatura, movimento) e também para transferir os dados coletados à uma camada superior de computação em nuvem. Ainda na camada de rede, modelos de topologia são criados para encaminhar os dados, como por exemplo através de um nó *relay* central. Nesse contexto, surgem os desafios de redes, muitos presentes em outros cenários. Dentre eles, a eficiência energética, maximização da vazão dos dados e segurança contra ataques maliciosos. Espera-se que esses tipos de comunicação citados sirvam não apenas para ligar dispositivos individuais dos usuários, mas também favoreça a comunicação M2M (machine-to-machine) [Qiu et al., 2018]. O paradigma “ciente do humano” é capaz de alavancar a *IoT* utilizando dados em alguns casos produzidos pelos próprios sensores para favorecer tipos de comunicação (*D2D*, *V2X*, etc). O estudo do perfil de mobilidade e rotinas de usuários presentes nas proximidades serve para eleição de quais nós tem maior potencial de encaminhar dados dos sensores (ou de um nó *relay* que reúne as informações dos sensores) de *IoT* até a nuvem. Já existem trabalhos relacionados a tipos de comunicação ou agendamento de recursos que levam em conta o contexto social, mobilidade e outras informações [Wang et al., 2017, Name et al., 2017].

Aspectos humanos como os laços sociais (extraídos por exemplo das redes sociais *online*) servem para o estabelecimento de relações de confiança e como métrica de qualidade de serviço na comunicação e composição de dados oriundos de sensores de usuários diferentes. O estudo da rotina individual pode ser utilizado para prever os melhores momentos para capturar e/ou compartilhar dados de determinados sensores, mantendo-os em modo de espera em certas janelas de tempo, favorecendo a eficiência energética. Outro exemplo inclui situações de perigo, identificadas a partir de sensores de *IoT* coletando dados do contexto humano que servem como base para estimular comunicação oportunística na falha da infraestrutura principal de rede.

De acordo com os exemplos citados, concluímos que a aproximação do paradigma “ciente do humano” da *IoT* favorece não apenas uma nova perspectiva de redes, mas alavancará soluções que auxiliarão no dia-a-dia das pessoas num futuro próximo.

1.3.2.3. Redes veiculares (VANETs)

Com as Redes Veiculares (*VANETs*), uma série de aplicações para maior eficiência no tráfego, segurança dos passageiros, assistência ao motorista, dentre outras, estão sendo incorporadas em veículos modernos. Informações em tempo-real sobre congestionamentos, perigo e acidentes na rota, condições climáticas e sugestão de pontos de interesse prometem revolucionar os transportes urbanos.

As *VANETs* são consideradas como um caso especial de *MANETs*, onde veículos com dispositivos de comunicação e processamento de dados podem criar redes oportunísticas e espontâneas enquanto se movem. O padrão de comunicação *C-V2X* (*Cellular Vehicle to Everything*) vem emergindo com grande potencial e permite que usuários em deslocamento por estradas se comuniquem através de padrões como o *LTE* e futuramente *5G* [MacHardy et al., 2018]. Através de informações sensoriais (ex: condições climáticas

e da estrada, estado do veículo), os veículos serão capazes de ajudar seus motoristas a tomar decisões. Na Figura 1.6, são ilustradas visões de como dados do paradigma “ciente do humano” podem favorecer a tecnologia V2X.

Apesar da dinâmica dessas redes, alguns aspectos são recorrentes e ligados ao comportamento ou contexto individual dos seus usuários. A mobilidade por exemplo é constante, porém restrita, pois os veículos se restringem a trafegar por estradas ou ruas específicas e são submetidos a certas condições como tráfego, sinais, locais de travessia de pedestres. etc. Sendo assim, com o estudo do padrão de mobilidade de um usuário, é possível prever posições futuras em que o veículo irá se encontrar e até prever situações de *handover*. A dinâmica de encontros, apesar de possuir grande rotatividade de nós, pode ser útil para identificar grupos que estão seguindo uma determinada rota e que são capazes de interagir entre eles para formar redes oportunísticas e compartilhar dados. Na



Figura 1.6. Dados do comportamento humano que podem ser extraídos no cenário de Redes Veiculares V2X e que servirão para prever situações, favorecer interações, gerenciar recursos, criar mecanismos de incentivo, estabelecer confiabilidade, dentre outros.

camada de rede, os protocolos de roteamento devem implementar estratégias para prover comunicação constante. Esses protocolos dependem diretamente de fatores como mobilidade, tráfego de dados e também os formatos das estradas. A análise da mobilidade de usuários pode ajudar a identificar pontos comuns de troca de tráfego e dados, na intersecção de estradas por exemplo. No futuro, espera-se que outros sensores sejam incorporados a dispositivos do usuário, incluindo monitores de saúde. Alarmes sobre o estado de saúde do indivíduo podem sugerir mudanças de rotas (direcionamento a um hospital no caso de um veículo autônomo) ou a interrupção do veículo em caso de risco temporário.

1.3.2.4. Redes ultra-densas

De acordo com [López-Pérez et al., 2015], de 1950 a 2000, o desempenho das redes móveis aumentaram 25 vezes por conta da implementação de frequências de espectro mais largas, 10 vezes devido a avanços em técnicas de modulação e codificação e 2700 vezes por causa da diminuição dos tamanhos das células e consequentemente da distância de comunicação entre os pares. Dessa forma, a implantação das redes móveis tem migrado das tradicionais macro-células, para redes híbridas heterogêneas, onde além das macro-

células, células menores são densamente distribuídas na área de cobertura - as chamadas Redes Ultra-Densas (do inglês *Ultra-Dense Networks* - *UDNs*). Uma Rede ultra-densa é formada pelos seguintes componentes: (i) Pequenas células implantadas em grande densidade; (ii) Macro Estações-base; (iii) Nós-móveis (que podem ser de redes veiculares por exemplo); (iv) Controle da rede (potencialmente através de *SDN*) e (v) Dispositivos dos usuários finais.

As Redes Ultra-Densas são essenciais às próximas gerações de redes móveis como o 5G e 6G. Associadas com o paradigma “ciente do humano”, diversas soluções e requisitos podem ser alcançados. De acordo com [Yu et al., 2016], os principais desafios das Redes Ultra-Densas são: (i) Controle de Interferência; (ii) Mobilidade e (iii) Gerenciamento de Recursos (incluindo questões energéticas). A mobilidade em *UDNs* é estudada do ponto de vista de rede, no caso da seleção/re-seleção de uma estação-base (*handover/handoff*). A análise de mobilidade de indivíduos pode servir para prever situações de *handover/handoff* e também auxiliar na alocação de recursos de rede. Já na borda da rede, a comunicação *D2D multi-hop* “ciente do humano” pode aumentar a capilaridade, garantir o encaminhamento das mensagens e auxiliar no *offloading* de dados. Outros aspectos como a concentração de pessoas em janelas de tempo e espaço físico similares, a dinâmica de encontros e o contexto, podem ser usados para otimização no posicionamento das micro-células e para prever necessidade de alocação de recursos em grandes concentrações de pessoas. A redução do congestionamento das macro-células também pode ser feita através de técnicas de *offloading* de dados baseada na comunicação *D2D* “ciente do humano”. Ainda de acordo com [Yu et al., 2016], a maior parte do consumo nas redes móveis de celulares é relativo às estações-base (cerca de 60% a 80%). A regularidade e os padrões de uso dos usuários podem ser estudados para o dimensionamento dinâmico de capacidade das estações-base, assim como já se faz em data-centers, aumentando a eficiência energética [Araújo et al., 2017].

1.3.2.5. Redes centradas na informação

Arquiteturas de Redes Centradas na Informação (do inglês, *Information-Centric Networking* – *ICN*) são baseadas num modelo de consumo/requisição centralizado no detentor do conteúdo e encaminhamento baseado em nomes. As *ICNs* oferecem suporte nativo à mobilidade, segurança ao nível do conteúdo, comunicação não orientada à conexão e realização de *caching* na camada de rede [Jacobson et al., 2009, Xylomenos et al., 2014, Zhang et al., 2014]. Por conta dessas características, informações do comportamento humano oferecem novas possibilidades para a oferta de um melhor serviço de rede em *ICNs* a partir: (i) da identificação de padrões de comportamento de produtores de conteúdo, buscando minimizar efeitos do *handoff*; (ii) do uso de políticas de cache de acordo com contexto dos usuários; (iii) da escolha de localização de cache a partir das suas rotinas diárias. Cada uma dessas possibilidades são discutidas a seguir.

Alguns dos desafios de mobilidade em *ICN* estão ligados ao deslocamento dos nós consumidores e também ao *handoff* de produtores, que podem acarretar dentre outros problemas, na indisponibilidade de conteúdo [Lehmann et al., 2016, Araújo e Sampaio, 2017]. Trabalhos como [Lehmann et al., 2016, Farahat e Hassanein, 2016, Ge et al., 2016]

buscam resolver o problema da mobilidade de produtores à partir da alteração da característica reativa da rede, enquanto outras iniciativas buscam a cooperação dos nós na borda da rede [Panwar et al., 2017, Rehman e Kim, 2017, Hahm et al., 2017]. Por outro lado, esse problema pode ser mitigado a partir de um modelo de ICN em que dados do comportamento humano sejam utilizados para prever o deslocamento do produtor. Com isso, seria possível implementar técnicas de armazenamento oportunista na rede e priorizar o armazenamento e manutenção no domínio daqueles conteúdos do produtor móvel, a partir de uma política de *cache* específica.

Em estratégias como [Chen e Kountouris, 2015, Hajri e Assaad, 2016] é sugerida a adoção de *ICN* em redes celulares, buscando manter os conteúdos mais populares em estações-base na borda da rede e agrupando usuários com interesses similares. Apesar dos ganhos nas taxas de acerto com a adoção de grupos de usuários afins, as mesmas políticas de *cache* são adotadas, independentemente do padrão de requisição dos usuários. O desempenho nesse cenário pode variar a depender do perfil dos usuários de demanda por conteúdo [Ioannou e Weber, 2016]. Ou seja, padrões de comportamento do humano podem auxiliar as propostas na escolha das políticas de *cache* mais apropriadas para atender agrupamentos e, assim, aumentar a taxa de acerto de conteúdo em geral.

A localização dos *caches* numa topologia de rede também pode influenciar consideravelmente a taxa de acerto e, consequentemente, a qualidade do serviço de rede oferecida. Diversos trabalhos na literatura têm explorado informações de rotinas diárias dos usuários para prever por onde os mesmos devem passar e, assim, disponibilizar conteúdos em locais estratégicos visando aumentar a taxa de acerto. Nesta mesma linha, propostas são apresentadas em cenários de redes veiculares, implementando *caches* proativos em locais estratégicos da rede tendo, visando distribuição de conteúdos [da Silva et al., 2016]. Os maiores desafios para tais propostas é que dados de monitoramento e análise de tráfego não são suficientes para tomada de decisão sobre localização ótima de *caches* e de réplicas de conteúdos em cenários nos quais *caches* são armazenados na rede (i.e., *on-path caching*). Em cenários de mobilidade fortemente caracterizados por ambientes altamente dinâmicos, esse problema torna-se ainda mais evidente. Por tais motivos, é necessário o uso de informações contextuais de curto prazo, envolvendo não somente disponibilidade de nós e popularidade do conteúdo, mas também outras relacionadas ao hábitos e rotinas dos usuários.

1.4. Modelos e soluções

A seguir, são apresentados modelos e soluções em redes móveis relacionados a *offloading* de dados, redes sociais *online*, segurança e privacidade. Nesse contexto, aspectos do comportamento humano são relacionados com trabalhos propostos nos últimos anos nessas áreas, reforçando a progressiva presença do “ciente do humano” nas soluções.

1.4.1. Descarregamento (*offloading*) de dados

Conforme discutido na Seção 1.1, o aumento de tráfego nas redes móveis é um grande desafio para as operadoras que nem sempre estão preparadas para enfrentar a demanda. O descarregamento de dados (do inglês *data offloading*) surgiu como uma das formas de enfrentar esse panorama. Em [Zhou et al., 2018], os autores classificam o descarregamento

de dados em quatro categorias: (a) *Small cells* - micro, pico ou femto-células são distribuídas pela infraestrutura para descarregar o tráfego; (b) *WiFi* - pontos de acesso *WiFi* são “espalhados” em áreas congestionadas como alternativa às redes celulares; (c) Oportunística - estimula comunicação oportunística entre os dispositivos dos usuários, e; (d) Heterogêneas - combina técnicas anteriores tentando minimizar os problemas individuais de cada uma.

Ainda segundo [Zhou et al., 2018], apesar da popularidade de estratégias de *offloading* baseadas em *small cells* e *WiFi*, ambas dependem de instalação e manutenção de infraestrutura e se limitam a uma determinada área de cobertura. Sendo assim, o descarregamento de dados através de comunicação oportunística é uma alternativa promissora, dada a grande presença de dispositivos de usuários em áreas urbanas congestionadas. Por exemplo, o tipo de deslocamento que uma pessoa utiliza, seja ele a pé, de bicicleta ou de ônibus implica em trajetórias e também em interações (contatos) diferentes com outras pessoas em ambientes urbanos. Aspectos como esses são capazes de revelar ideias fundamentais em termos de utilização da rede e por conseguinte, servem a novas técnicas de *offloading* de dados. Através de tipos de comunicação como o *D2D*, essas técnicas de *offloading* podem se basear em informações dos usuários. Mobilidade, dinâmica de contatos, laços sociais, dentre outros aspectos são informações úteis para solucionar desafios como a eleição de nós para distribuição do conteúdo ou escolha do tipo de conteúdo direcionado a um perfil de usuário. Além disso, é sabido que os dispositivos dos usuários possuem capacidade limitada de recursos incluindo bateria, memória, processamento e alcance de transmissão. Esse tipo de informação individual também pode ser levado em consideração nas técnicas de *offloading* oportunísticas. Complementarmente, essas informações do contexto e comportamento humano são úteis também em soluções das outras três classes de *offloading*. Por exemplo, o histórico de mobilidade e utilização dos usuários serve para as operadoras identificarem os melhores pontos de instalação de *small cells* ou *APs WiFi* [Oliveira e Viana, 2014]. A seguir, citamos alguns trabalhos da literatura relacionados com o paradigma “ciente do humano” ordenados cronologicamente, estabelecendo uma relação com a linha do tempo apresentada na Seção 1.1.

Em [Whitbeck et al., 2011], os autores buscaram maximizar o *offloading* de dados estudando *quantas* cópias de um conteúdo deveriam ser injetadas na rede, *quando* e para *quais* nós. O terceiro desafio foi solucionado baseado em aspectos do usuário como a mobilidade obtida através de coordenadas GPS e obtenção de lista de contatos vizinhos. Já em [Chuang e Lin, 2012], a estratégia de *offloading* é baseada não apenas na frequência de contatos, mas também no relacionamento social entre os nós. Essa ideia tem o intuito de distribuir o conteúdo a usuários que participam de comunidades diferentes, garantindo assim que uma maioria dos nós tenham acesso ao objeto. Simulações com *traces* reais de usuário mostram que o esquema proposto tem desempenho muito superior comparado a outras alternativas apenas baseadas nos contatos.

No trabalho [Ding et al., 2013], os autores propuseram uma técnica de *offloading* heterogênea agregando operadoras, provedores de *WiFi* e usuários finais. A solução é baseada em informações como localização e mobilidade, utilizando as mesmas para prever aspectos do descarregamento de dados. Em [Al-Kanj et al., 2014], os autores também utilizam estudos de mobilidade dos nós para promover descarregamento de dados oportunístico em três cenários: (a) disseminação de dados *multi-hop*; (b) transmissão

multi-cast entre nós móveis, e (c) cooperação levando em conta restrições de consumo de energia entre os nós móveis.

O comportamento dos usuários em redes sociais também pode fornecer informações importantes à modelagem de técnicas de *offloading*. Justamente, em [Li et al., 2014], os autores analisam características de redes sociais para melhorar o desempenho de descarregamento de sua proposta. Informações como laços sociais, comunidades sociais, centralidade social e pontes sociais foram utilizadas nessa solução. Outro exemplo de utilização de redes sociais pode ser encontrado em [Wang et al., 2014]. Os autores propuseram um *framework* para descarregamento de dados da rede de celular baseado no impacto de disseminação de conteúdo dos nós em uma rede social e também nos seus padrões de mobilidade. Essas informações serviram para escolha dos nós disseminadores do conteúdo. Outros trabalhos como [Thilakarathna et al., 2014, Thilakarathna et al., 2017] levam em conta características sociais para escolher os disseminadores dos dados.

Em [Zhang et al., 2017], os autores utilizam noções de participação móvel através da mobilidade e de características sociais entre grupos de usuários, propondo um esquema chamado de “ciente do humano”. A solução se baseia na noção de interesse social similar advinda do fenômeno da homofilia (estudado em sociologia), que descreve que pessoas com atributos similares mantêm contato mais frequentemente que com estranhos.

1.4.2. Redes sociais online

As Redes Sociais Online (do inglês *Online Social Networks - OSNs*) favorecem a interação entre indivíduos com interesses ou objetivos similares através dos seus dispositivos móveis [Hu et al., 2015]. Aplicações, modelos, arquiteturas de redes, tecnologias relacionadas, etc podem se basear no “conhecimento” das *OSNs* para estimular colaboração, oferecer serviços, promover interação social oportunística, compartilhar informação, dentre outros. Ao contrário de redes sociais tradicionais, as *OSNs* podem tirar proveito de informações do contexto e comportamento humano, tais como as que podem ser capturadas através dos sensores dos dispositivos móveis (*GPS*, acelerômetro, câmera, etc). Complementarmente, as *OSNs* servem como fonte de dados. Por exemplo, interações, dinâmica de encontros, mobilidade e laços sociais podem ser inferidos através de dados provenientes das *OSNs*. Sendo assim, há uma forte ligação entre *OSNs* e o escopo desse minicurso. A seguir, apresentaremos alguns trabalhos relacionados.

O surgimento de serviços baseados na localização tem sido favorecido pela disseminação dos dispositivos móveis com *GPS* e estudos de mobilidade. Dentre esses serviços, estão as Redes Sociais Baseadas em Localização (*LBSNs*), que permitem estudo do comportamento individual *online* e *offline*, englobando modelos de mobilidade, análise comportamental e de anonimização, recomendações de relacionamentos sociais, dentre outros. Em abril de 2014 o *Facebook* lançou um serviço chamado “Amigos Próximos”, que permite aos usuários encontrar amigos através das suas coordenadas geográficas. O potencial desse tipo de serviço é grande, segundo [Thilakarathna et al., 2016], já que novos relacionamentos podem surgir baseados na localização em tempo real. No trabalho citado, é analisada a estrutura social de uma rede nesse contexto com foco na relação entre comunidades, usuários e outras redes. Os autores identificaram que os usuários mesmo tendo um alto padrão de atividade nessas redes, regularmente estão confinados

nos mesmos espaços geográficos (ressaltando a noção de rotina do comportamento humano). Adicionalmente, resultados mostram que é possível contactar mais de 50% dos usuários em até 3 dias através de aproximadamente 15% das localizações, abrindo oportunidades para redes de cache, comunicação *D2D* e *DTNs*. Por fim, foi descoberta uma grande similaridade de trajetórias em pessoas de mesmas comunidades, o que pode ser explicado pelo contexto social e humano.

A seleção e associação de pares em comunicação cooperativa dispositivo-a-dispositivo (*D2D*) é um dos desafios nessa tecnologia, já que desconexões ocorrem frequentemente (devido à mobilidade e curto alcance das conexões). Em [Meng et al., 2017], os autores avaliam métodos de seleção ótimos de Estações-Base ou pares colaborativos baseados na noção das interações sociais. A ideia é extrair vantagens das redes móveis para enriquecer as redes sociais e também utilizar o conhecimento dos relacionamentos sociais entre usuários para melhorar a eficiência da comunicação. Dentre as métricas avaliadas nesse contexto estão o histórico de contatos, similaridade social e histórico de contribuição (quando um usuário age como *relay* para dados de outro). Em trabalhos como [Thilakarathna et al., 2014, Wang et al., 2015, Thilakarathna et al., 2017], os autores apresentam soluções para *offloading* de dados oportunístico, baseado em noções obtidas de redes sociais móveis, incluindo aspectos sociais e mobilidade.

Usuários de redes sociais frequentemente enfrentam problemas como excesso de informação e falta de interação entre as suas redes, que ainda agravam o primeiro com publicações e publicidade repetidas. Em [Vu et al., 2015], os autores propuseram um esquema de filtragem de publicações, primeiro agregando dados de diferentes redes sociais, e depois apresentando resultados baseados nos interesses dos usuários, definidos pelos mesmos através da análise dos seus perfis. Essa solução também leva em conta o uso de informações do contexto humano para oferecer serviços direcionados aos usuários finais.

1.4.3. Segurança e privacidade

A disseminação de serviços baseados em localização e outros dados multidimensionais do comportamento humano traz preocupações à segurança e privacidade dos dados dos usuários [Hubaux e Juels, 2016]. Isso acontece pois esses dados podem revelar a identidade dos indivíduos, então a aplicação de mecanismos de privacidade é necessária antes, durante e após o envio às plataformas para garantir a privacidade dos usuários. Violações são observadas constantemente, como o recente vazamento de dados de cerca de 50 milhões de usuários do *Facebook* para utilização na campanha presidencial do norte-americano *Donald Trump* [tru, 2018].

Assim como a mobilidade foi o primeiro aspecto humano que passou a ser considerado em aplicações e arquiteturas de rede, a segurança e privacidade nesse contexto possui uma série de propostas. Uma pesquisa realizada por [Fawaz e Shin, 2014] com 180 usuários de *smartphones* revela que 78% acreditam que aplicativos que acessam sua localização representam ameaças à privacidade e 85% se importam com quem acessa esse tipo de dado. O estudo de regularidade e rotinas em dados de localização pode levar ainda a outras informações confidenciais como a inferência do local de trabalho, casa, preferência sexual, religiosidade, preferência política, identificação dos familiares, etc. Conforme chamamos atenção anteriormente, os dados do contexto e comportamento

humano são úteis em várias áreas, no entanto eles contém características privadas dos usuários que podem causar consequências desastrosas se utilizados maliciosamente. À seguir, apresentamos trabalhos relacionados à segurança e privacidade de dados do contexto e comportamento humano, tanto com propostas de soluções como identificação de falhas.

Em [Chatzikokolakis et al., 2015], os autores propuseram uma definição formal de privacidade, chamada geo-indistinguibilidade. Ela permite ao usuário divulgar informações suficientes para obter o serviço desejado, ao mesmo tempo mantendo a privacidade. Por exemplo, alguém no centro de uma cidade gostaria de obter informações sobre restaurantes próximos mantendo sua privacidade. Para garantir isso, é empregada a chamada “ofuscação”, que consiste em fornecer uma localização aproximada ao invés da exata. De acordo com uma pesquisa em [Fawaz e Shin, 2014], apenas 18% rejeitariam prover uma localização imprecisa. O estudo mostra que usuários se preocupam em manter privada sua localização, mas não sua identidade, aceitando autenticar-se em serviços confiáveis para obter recomendações personalizadas por exemplo.

Em segurança e privacidade, a anonimização dos dados, historicamente é um dos assuntos mais discutidos. No entanto, no caso de dados espaço-temporais, o trabalho [de Montjoye et al., 2013a] prova que é possível identificar indivíduos correlacionando conjuntos de dados de mobilidade com outras informações externas obtidas facilmente (ex: dados de um perfil numa rede social). Um *dataset* anonimizado não contém nome, endereço, telefone ou outros identificadores pessoais óbvios, mas se o perfil do usuário for distinto o suficiente, é possível fazer inferências sobre os dados e identificá-lo. Assim como uma impressão digital, a mobilidade humana possui unicidade, e esse trabalho estima o número de pontos necessários para identificar um indivíduo dentro de um *trace* de mobilidade enriquecido com informações externas. A pesquisa foi feita com um conjunto de dados anonimizado de uma operadora móvel com cerca de 1,5 milhões de usuários.

Para conectar a um ponto de acesso *WiFi*, nossos dispositivos enviam periodicamente requisições buscando as redes disponíveis. Essas requisições contém informações como o endereço *MAC* (identificador do dispositivo) e do lado do provedor, os *SSIDs* (identificadores das redes) estão repletos de informação semântica. Em [Luzio et al., 2016], os autores mostram como essas requisições aparentemente banais podem ser utilizadas para descobrir aspectos da vida humana que nada têm a ver com tecnologia. Dentre eles, relacionamentos entre pessoas, previsão de encontros, preferência política e até a origem de pessoas presentes em grandes aglomerações.

Atualmente, nossos cartões de crédito, *smartphones*, navegadores e até carros geram informações que dizem muito a respeito de quem nós somos, como nos movemos, quem contactamos ou quanto gastamos. Os cientistas têm comparado essa disponibilidade de dados comportamentais com a invenção do microscópio e chamado à atenção sobre a sensibilidade dessas informações. Em [de Montjoye et al., 2015], os autores estudam transações de cartões de crédito de 1,1 milhão de pessoas anônimas e mostram como informações espaço-temporais são suficientes para reidentificar 90% dos indivíduos. O conhecimento do preço da transação é o ponto que mais aumenta o risco de reidentificação (na ordem de 22%). O trabalho ainda mostra que mesmo em conjunto de dados com informações bem esparsas, a anonimização pode ser quebrada através da correlação de

valor, local e tempo. Curiosamente, o trabalho mostra que mulheres são cerca de 1.214 vezes mais reidentificáveis que homens em transações de cartão de crédito. Isso reforça o fato que características humanas são aspectos-chave para entender como as pessoas se comportam e agem.

A privacidade de dados sensíveis dos usuários é geralmente invadida de duas formas: obtendo dados de servidores considerados confiáveis, mas que possuem vulnerabilidades de segurança; ou, correlacionando dados espaço-temporais anonimizados, conforme trabalhos mencionados anteriormente. Em [Ren et al., 2018], os autores aplicam um novo método chamado de privacidade diferencial local, onde algoritmos são utilizados para variar a forma como os dados são anonimizados em cada serviço, dificultando a correlação. Com isso, serviços baseados nesses dados podem ser desenvolvidos enquanto a privacidade dos usuários não é colocada em risco.

1.5. Avaliação e experimentação

Nessa seção são relacionadas e detalhadas as fontes de dados identificadas para informações do contexto e comportamento humano. Exemplos de *traces* e *datasets* também são trazidos, buscando instrumentar futuras pesquisas na área. Completando, comentaremos sobre alguns ambientes de avaliação e experimentação em redes móveis.

1.5.1. Fontes de dados

Para alavancar futuras arquiteturas de rede “cientes do humano”, dados do contexto e comportamento dos indivíduos devem ser adquiridos ou gerados. A Figura 1.7 resume fontes de dados e a seguir, apresentamos detalhadamente cada uma delas.



Figura 1.7. Fontes e Tipos de Dados do Contexto e Comportamento Humano que podem ser utilizados em pesquisas facilitadoras de arquiteturas de redes do futuro.

- Sensores físicos - instalação de sensores dedicados a objetivos específicos, como por exemplo monitorar a qualidade do ar, obter níveis de ruído ou rastrear carros (ex: *GPS* em frota de táxis). Os desafios nesse cenário são o alto custo de instalação, e em alguns casos necessidade de autorizações para o uso, além da adaptação de veículos ou usuários.
- Dados de Infraestrutura: obtenção de dados de serviços públicos. Dentre eles, sistemas de transporte como ônibus e metrô e infraestrutura de telefonia. A dificuldade

nesse caso é o acesso, já que geralmente esses conjuntos de dados são fornecidos apenas para certas empresas ou universidades. Em [par, 2018] há um exemplo de fonte de dados de infraestrutura da cidade de Paris/França.

- **Estatísticas** - engloba estudos estatísticos populacionais (ex: índices de saúde, índices demográficos e aspectos sociais), dados de dinâmica urbana (preços de moradia e localização), segurança (taxas de criminalidade), energia (consumo de eletricidade ou gás), dentre outros. Esse tipo de fonte de informações está se popularizando com iniciativas governamentais de disponibilização como *data.gov*. Ainda assim, os dados podem não estar disponíveis para o local onde o estudo deseja ser conduzido. Outro desafio é agregar esses dados, devida à variedade de formatos nos quais eles são disponibilizados incluindo tabelas, gráficos, mapas, formulários, etc.
- **Redes Sociais Baseadas em Localização** - aplicações ou serviços que combinam redes sociais com informações espaço-temporais. Rotinas e preferências de indivíduos podem ser extraídas. Geralmente envolve participação voluntária dos usuários que determinam como, quando, o que e onde compartilhar dados. Esses serviços se popularizaram com a disseminação dos *smartphones*. Essa fonte de dados é uma das mais poderosas, já que técnicas tradicionais como censo, recrutamento voluntário e dados de GPS não estão disponíveis com o mesmo alcance global. Dentre os exemplos de Redes Sociais Baseadas em Localização, estão: *Foursquare*, que permite o compartilhamento de informações sobre lugares visitados; *Waze*, onde usuários compartilham informações de trânsito em tempo real e *Twitter*, onde indivíduos compartilham atualizações pessoais por texto até 140 caracteres.

Além dos métodos citados acima, pesquisas podem ser baseadas em *traces* (conjuntos de dados com informações dos usuários) já disponíveis. Idealmente, esses dados devem cobrir populações (ex: pessoas de uma cidade, país ou de uma universidade) e por um tempo (semanas a meses) que permita observar hábitos e rotinas dos indivíduos. Populações apresentam diversidade de comportamento e dão suporte a soluções mais abrangentes - que devem ser validadas através de fontes de dados diferentes para reduzir o risco de conclusões restritas. Os *traces* são obtidos geralmente através de duas formas, conforme a seguir:

- **Coleta de Tráfego Móvel** (do inglês *Mobile Traffic Gathering* - baseada nos *logs* de tráfego de redes móveis de operadoras. A grande quantidade de dispositivos móveis interagindo continuamente com a rede da operadora permite que informações temporais e geo-espaciais sejam armazenadas, servindo não apenas para o gerenciamento da rede e tarifação dos usuários.
- **Crowdsensing Móvel** (do inglês *Mobile Crowdsensing*) - consiste em extrair dados sensoriais multi-dimensionais diretamente dos dispositivos móveis por aplicações específicas. Os usuários dessas aplicações agem como voluntários colaborando geralmente com alguma pesquisa. Esses aplicativos são capazes de obter diversos dados dos sensores dos dispositivos (ex: aceleração, coordenadas geográficas do

GPS, conexões a redes *WiFi* e *Bluetooth*, força do sinal, nível de bateria e rotação). Nessa modalidade de *traces*, apesar da diversidade de informações disponíveis, geralmente a população de usuários é bem mais reduzida, pela dificuldade de abranger comunidades maiores.

No caso dos *datasets* de operadoras, os mais conhecidos são os chamados *CDR* (*Charging Data Records* ou *Call Detail Record*) [Blondel et al., 2015, Silveira et al., 2016, Marques-Neto et al., 2018]. Os registros *CDR* são gerados quando transações (ex: ligação telefônicas ou *SMS*) ocorrem na rede de uma operadora móvel. Dentre as informações armazenadas, estão os identificadores dos envolvidos na transação, tempo de início, duração, status (sucesso ou falhas detectadas), localização da antena de onde a transação foi iniciada, rota que a transação seguiu, etc. A vantagem dos *CDRs* oferecidos pelas operadoras são a quantidade de registros e o tempo de cobertura. Alguns chegam a ter milhões de usuários que produziram bilhões de registros e se deslocaram por milhares de antenas. No entanto, esses dados podem ser esparsos e nesse caso carecem de métodos para completar as lacunas antes de serem utilizados.

A seguir citamos exemplos de conjuntos de dados obtidos através de *Crowdsensing Móvel*. O *Geolife* [Zheng et al., 2009] foi um dos primeiros *datasets* públicos a disponibilizar informações de mobilidade. Ele fornece dados espaço-temporais de 182 pessoas (maior parte delas de *Beijing*, capital chinesa), obtidos por *GPS* entre abril de 2007 e agosto de 2012. Uma das deficiências desse *trace* são as lacunas temporais, já que nem todos usuários apresentam dados suficientes para serem analisados. Sendo assim, critérios precisam ser estabelecidos para selecionar os usuários em qualquer tipo de análise que vier a ser feita.

A aplicação *MACACO* foi desenvolvida no projeto Europeu CHIST-ERA [mac, 2015], focado em comunicação adaptativa ao conteúdo e baseada no contexto. Dentre os dados coletados, estão tráfego (*download/upload*), coordenadas geográficas, redes disponíveis (*WiFi*, *Bluetooth*), status de bateria e memória do dispositivo, aceleração, histórico do navegador e aplicativos instalados. As informações são requisitadas a cada 5 minutos e contemplam uma população de cerca de 180 usuários de diferentes países, dentre eles pesquisadores, professores e estudantes de universidades. Os usuários também são convidados a preencher um questionário não obrigatório que coleta informações sobre os traços de personalidade e as preferências de aplicação das pessoas. O *trace* do *MACACO* não está disponível publicamente, mas é um exemplo de como aplicações podem ser desenvolvidas para auxiliar na obtenção de dados.

Outro *trace* bastante utilizado é o *Friends and Family* do projeto Reality Mining do MIT [Eagle e (Sandy) Pentland, 2006], que contempla uma população de 100 usuários observados por 9 meses. Dentre as informações coletadas estão *logs* de chamadas, outros dispositivos *bluetooth* próximos, identificadores das torres de celular, localização, status do telefone (carregando ou ocioso), consumo de mídia e padrões de difusão de comportamento. Os participantes também responderam pesquisas em intervalos regulares: as pesquisas mensais incluem perguntas sobre a autopercepção de relacionamentos, afiliação de grupo, interações e teste de personalidade, o *Big-Five*; Pesquisas diárias incluem perguntas como humor, sono e outros registros de atividades.

O *CRAWDAD* [Kotz et al., 2009] é uma comunidade de disponibilização de *traces* para a comunidade de pesquisa. Diversos tipos de dados são oferecidos para propósitos de pesquisa diferentes, incluindo modelagem de comportamento humano, serviços de localização, conectividade oportunística e estudo de mobilidade. Por exemplo, recentemente (fevereiro/2018) foi disponibilizado um *trace* de mobilidade de ônibus na cidade do Rio de Janeiro. Dados de localização em tempo real de mais de 12.000 ônibus são reportados a cada minuto, incluindo horário, identificação do veículo, linha, coordenadas geográficas e velocidade.

O *OpenCellID* [ope, 2018] é um projeto comunitário colaborativo que busca documentar torres de celular e *APs WiFi* ao redor do mundo, juntamente com sua localização. A comunidade tem mais de 75.000 colaboradores, contribuindo com novas medições (na ordem de milhões) diariamente. O projeto disponibiliza também uma lista de aplicações que podem ser usadas para permitir a coleta.

Como o *OpenCellID*, o *OpenStreetMap* [Almendros-Jiménez e Becerra-Terón, 2015] é um projeto colaborativo com mais de 1,9 milhões de usuários registrados. Um dos recursos é a possibilidade dos usuários enviarem suas trajetórias georreferenciadas para melhorar o mapeamento de localidades no mundo inteiro. Cada trajetória detalhada do *GPS* do usuário é um conjunto de pontos georreferenciados classificados de maneira ascendente por instante de tempo (*timestamp*).

Já o *Device Analyzer* [Wagner et al., 2014] é uma aplicação projetada pela Universidade de Cambridge que coleta informações de *smartphones* de voluntários para fins científicos. Dentre os dados coletados, estão aplicações utilizadas, quando o usuário faz uma ligação ou envia *SMS*, localização (baseada na rede), e redes *WiFi* ou dispositivos *bluetooth* disponíveis.

Além dos exemplos citados, o *Google Maps* (versão *Android*) permite que o usuário registrado baixe o seu próprio conjunto de dados de deslocamento, incluindo coordenadas geográficas associadas à informações temporais. Com os avanços do paradigma “ciente do humano”, a intuição é que a variedade e disponibilidade de *traces* de usuários aumente ao longo dos próximos anos.

1.5.2. Ambientes de avaliação e experimentação

Historicamente, sempre houve uma escassez de plataformas de avaliação e experimentação flexíveis e de código aberto para validar soluções em diferentes camadas das redes móveis. De acordo com [Banerjee et al., 2015], a mudança começou imposta pela disponibilidade de novas tecnologias para gerenciar recursos nas redes como *SDN*, *NFV* e Computação em Nuvem e também pela diversidade de novos dispositivos a serem conectados. Com isso, alguns *testbeds* e ambientes experimentais de redes sem fio têm surgido, tais como o *PhantomNet* [Cho et al., 2016].

O *PhantomNet* é um *testbed* de redes móveis com um conjunto de recursos que desenvolvedores podem utilizar para desenvolver, avaliar e experimentar propostas de mobilidade. Dentre os recursos disponíveis e que podem ser instanciados através de uma interface *web* e terminais, estão interfaces *OpenEPC*, *OpenLTE* e *Open Air*, pontos de acesso de *hardware*, dispositivos móveis de usuários, nós virtuais e muitos outros ele-

mentos da infraestrutura do *Emulab* (ambiente de emulação de redes desenvolvido pelo mesmo grupo do *PhantomNet*). Além dos recursos, o *PhantomNet* disponibiliza diretivas de configuração e *scripts* para auxiliar pesquisadores com seus experimentos de mobilidade. Há uma série de exemplos disponíveis que podem ser modificados, além da possibilidade de introduzir propostas totalmente novas (*clean-slate*) no núcleo de funcionamento. Para utilizar o *Phantomnet*, o pesquisador deve efetuar um registro gratuito no portal do projeto.

Quando se trata de avaliação de desempenho, experimentação e validação de protocolos, mecanismos de *handover*, ou outras estratégias de redes, os ambientes de simulação e emulação são bastante utilizados. De acordo com [Fontes et al., 2015], no caso de experimentos em redes móveis com gerenciamento *SDN*, apenas alguns simuladores estão disponíveis (tais como o *NS-3* e *OMNet++*). Por conta da carência de um ambiente de emulação nesse contexto, os autores desenvolveram o *Mininet-WiFi*, uma extensão do emulador *Mininet* para prototipar, testar e avaliar novos protocolos, arquiteturas e aplicações de redes móveis. Com o *Mininet-WiFi* é possível emular uma rede móvel com *APs* e estações ou dispositivos, introduzindo parâmetros de mobilidade.

Conforme evidenciamos durante o texto, há uma série de informações do contexto e comportamento humano que podem ser correlacionadas com dados espaço-temporais. Sendo assim, plataformas de avaliação mais completas nesse sentido ainda são escassas.

1.6. Desafios de pesquisa

À medida em que mais dispositivos inteligentes ganham mercado, impactos têm sido observados na vida das pessoas. As arquiteturas de Internet do futuro irão conectar uma série de entidades distintas que causarão mudanças na forma como interagimos com as redes de computadores. Para que elas estejam preparadas para essa transição de paradigma, será necessário ir além de lidar com aspectos tecnológicos, mas entender o comportamento das pessoas e utilizá-lo nas soluções. Juntamente com essa noção do “ciente do humano”, surgem diversos desafios e direções de pesquisa.

Primeiramente, é preciso mudar as práticas de modelagem de soluções de rede. Apesar de muitas aplicações ou serviços serem direcionados ao usuário final, a preocupação com sua qualidade de experiência geralmente está associada com métricas e focada em condições da rede (ex: adaptação de taxa de transmissão) ou ligada à percepção do usuário do desempenho da rede. Para colocar o humano em destaque é preciso: (i) envidar esforços de análise de *big data* relacionado a informações do contexto e comportamento humano, e; (ii) trazer conhecimento sobre o comportamento humano às soluções de rede combinando ideias e métodos de diferentes áreas como aprendizado de máquina, psicologia, sociologia, redes de computadores, *data science* e estatística. Isso envolve ainda questões de disponibilidade de grandes quantidades de informação e métodos de processamento que permitam coletar, extrair e inferir comportamento humano. Ao longo do tempo, o correlacionamento desses dados será utilizado para analisar rotinas e prever situações e comportamentos [Gonzalez et al., 2008, de Montjoye et al., 2013b, Jeong et al., 2016, Silveira et al., 2016, Alsadhan e Skillicorn, 2017, Nedunchezian e Jacob, 2017, Cuttone et al., 2018]. No contexto de redes, esse tipo de informação será utilizado para melhor orquestrar como a rede deverá se comportar e para que os recursos sejam gerenciados de

forma mais eficiente. A falta de dados adequados é um dos obstáculos para um entendimento mais completo dos aspectos humanos [Naboulsi et al., 2016]. A análise deve ser estendida a um conjunto maior de características de locais, de recursos tecnológicos, dentre outros que podem ser obtidos através de dispositivos como os *smartphones*. Esses dados também precisam abranger populações maiores, preferencialmente de regiões metropolitanas completas.

Segundo, serão necessárias mudanças substanciais no modelo de comunicação tradicional. A emergência de novas aplicações, massificação de serviços de *cloud*, implantação da *IoT*, comunicação *Machine-to-Machine*, dentre outras tecnologias, contribuem com o congestionamento na borda da Internet [Akyildiz et al., 2016]. Além disso, à medida em que os dispositivos inteligentes vêm se tornando mais poderosos e ubíquos, mais pessoas estarão envolvidas e conectadas. Esse aumento de tráfego (conforme discutido na Seção 1.1) traz uma série de desafios ao núcleo e borda da Internet. Os modelos centralizados de comunicação de redes móveis em produção (ex: *4G* ou *LTE*) serão confrontados com essa crescente demanda, portanto uma mudança de paradigma é esperada na direção de soluções centradas nas rotinas e contexto das pessoas para uma gerência de recursos mais eficiente. Neste contexto, a novidade estará nos algoritmos projetados que utilizarão o conhecimento resultante do comportamento dos usuários, da heterogeneidade da rede e da incerteza existente no ambiente (seja esta trazida pelo comportamento humano ou condições físicas da rede).

Apesar das grandes taxas de crescimento do tráfego nos últimos anos, os protocolos de comunicação seguem limitados e em muitos casos baseados em estratégias desenvolvidas no passado, onde não se esperava um cenário como o atual, muito menos o futuro. Ainda há uma compreensão limitada das características que os protocolos devem levar em conta, dentre elas, aspectos do tráfego transportado e o contexto no qual o mesmo é gerado. Sendo assim, serão necessários protocolos inteligentes capazes de transportar as informações solicitadas com o menor custo possível para a rede, simultaneamente oferecendo qualidade de serviço e de experiência para os usuários [Yu et al., 2016]. Uma das características das arquiteturas de rede do futuro é que elas serão utilizadas não apenas para acessar informações, mas também para processá-las de forma distribuída. Por exemplo, bilhões de dispositivos *IoT* que serão conectados através de conexões sem fio na borda da rede necessitarão lidar com a incerteza e inconfiabilidade do meio sem fio. Boa parte desses dispositivos terão também limitações de processamento, bateria e memória, sendo assim a pilha de protocolos de rede executada nos mesmos deverá ser repensada, bem como as plataformas de *software* que serão utilizadas.

Mesmo com uma vasta literatura disponível, ainda há lacunas na predição do comportamento humano sob influência de fatores psicológicos, sociais, demográficos, dentre outros que devem impactar em modelos de predição [Subrahmanian e Kumar, 2017]. São necessários estudos quantitativos para desvendar um grau ou precisão esperada dessas predições, quais técnicas são mais adequadas para prever o comportamento de um indivíduo, e como fatores como os mencionados acima interferem na acurácia. Métodos de predição propostos na literatura requerem um grande histórico de dados e alta regularidade de eventos, o que reforça a necessidade de disponibilização de *datasets*.

As camadas de funções e protocolos da Internet possuem soluções que em cenários atuais com diversidade de dispositivos, comportamentos e requisitos não são ótimas [Nu-

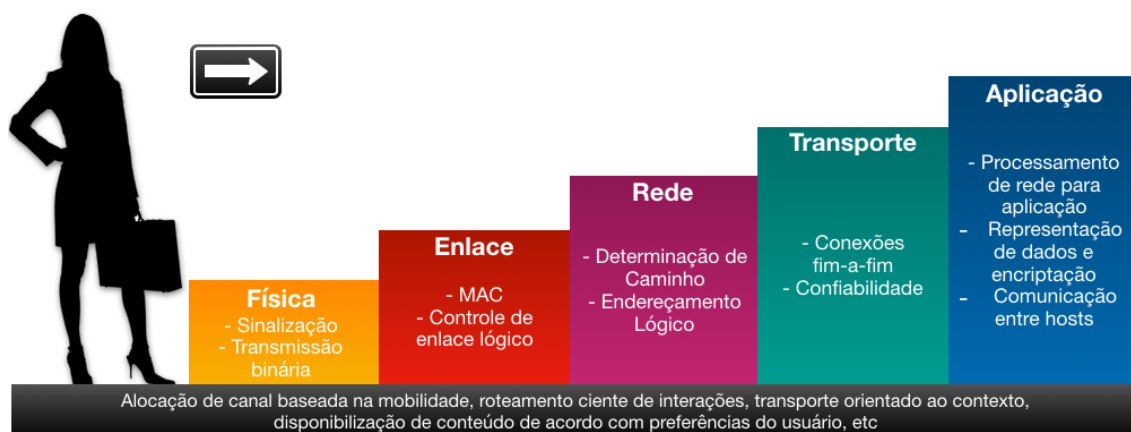


Figura 1.8. Informações do Contexto e Comportamento Humano deverão ser extraídas e incorporadas em soluções de diversas camadas da Internet.

nes et al., 2016]. Um dos desafios será identificar que tipo de informação do contexto e comportamento humano pode ser extraída para ser aplicada em novas soluções em cada uma das camadas para favorecer o gerenciamento de recursos e a qualidade de serviço e experiência dos usuários. Esse cenário é ilustrado na Figura 1.8.

Com o surgimento de tecnologias de gerenciamento e virtualização de rede como *SDN* e *NFV*, há uma progressiva mudança no paradigma de gerência de redes [Medhat et al., 2017]. Sendo assim, tornou-se possível a criação de redes virtuais fim-a-fim, possibilitando aos provedores de serviço acesso e controle total da rede que eles utilizam. Com isso, recursos de rede podem ser melhor gerenciados de acordo com o que os usuários requisitam. A utilização de dados do contexto e comportamento humano podem tornar essa gerência de recursos ainda mais eficiente, fornecendo a esses sistemas informações para prever a demanda dos usuários.

1.7. Lições adicionais

Ao longo desse minicurso foi mostrado como arquiteturas de rede, conceitos e tecnologias relacionadas estão se aproximando de aspectos do comportamento humano nas suas soluções. Isso ocorre também em outras áreas e essa seção traz alguns exemplos como lições adicionais. Esses estudos mostram que além das telecomunicações, setores como lazer, *marketing*, finanças, governo, saúde, comércio e propaganda podem se beneficiar de informações multi-dimensionais capazes de serem obtidas através de dispositivos dos usuários ou outras fontes (conforme mostradas na Seção 1.5).

No relatório [Lic, 2015], os autores chamam a atenção das vantagens que as operadoras têm em relação a outros modelos de negócio, podendo alavancar oportunidades bilionárias (sobretudo relativas ao *marketing*) baseadas em conjuntos de dados do comportamento humano. O argumento é que essas empresas deveriam primeiro procurar oferecer serviços que compartilhem “visões” sobre a localização de usuários para outros clientes corporativos, ao invés de focar em serviços de localização orientados ao consumidor e que exigem recursos adicionais. Os chamados serviços baseados em localização permitem que uma empresa forneça um determinado serviço ou apresente algo mais relevante e direcionado ao seu cliente. A identificação de padrões e repetitividade na mobilidade

de usuários ao longo do tempo (informação espaço-temporal) permite uma compreensão maior do consumidor em termos de comportamento e motivação, e também evidencia uma imagem mais clara do perfil do visitante. As operadoras *Orange* [flu, 2016] e *Telefonica* [tel, 2017] já fornecem esse tipo de serviço. Corporações contratantes podem visualizar informações estatísticas de tráfego, origem e rotas que os seus consumidores estão seguindo no seu ponto de interesse. Os dados dos clientes são anonimizados.

Em sistemas de transporte inteligentes (*ITSs*), informações sobre condições do tráfego podem ser obtidas de diversas fontes. Esses dados geralmente apresentam lacunas, e o preenchimento das mesmas é um dos grandes desafios nesse contexto. Em [Asif et al., 2016], os autores apresentam métodos de preenchimento dessas lacunas baseados na extração de padrões globais de tráfego encontrados nos dados. Esses métodos são analisados em diferentes cenários (categorias de estradas) e divididos por dia da semana (ou seja, observando a rotina e comportamentos diferentes em cada dia).

No campo da Interação Humano-Computador (*IHC*), auxiliar os usuários a realizarem suas tarefas é um dos desafios. Em [Czibula et al., 2009], os autores propuseram um auxiliar pessoal inteligente capaz de descobrir hábitos, habilidades, preferências e metas de um indivíduo, antecipando suas intenções. Esse assistente é capaz de aprender continuamente sobre o usuário baseando-se nas suas experiências e ações anteriores que cumpriram uma determinada tarefa.

O uso de *Big Data* também pode ser aplicado na melhoria de qualidade de serviço e experiência de redes. Em [Zheng et al., 2016], os autores exploram diversos meios de *Big Data Analytics* para otimização de redes, baseados em dados coletados tanto de equipamentos dos usuários como das redes móveis. Os dados dos usuários são relacionados ao seu perfil e comportamento, localização, mobilidade e padrões de comunicação pessoais. Dentre os dados do usuário obtidos através do núcleo da rede, estão: desempenho, chamadas completadas e índices de utilização por aplicação.

O comportamento humano é formado por uma série de características individuais e que interagem entre si. Anteriormente citamos os traços de personalidade, aspectos identificados em trabalhos na área de psicologia [Digman, 1990], [Goldberg, 1993] que são geralmente estabelecidos no início da vida adulta e permanecem relativamente estáveis ao longo do tempo. Por exemplo, usuários com personalidade neurótica tendem a ser menos móveis (confinados), enquanto que indivíduos extrovertidos e com traços de agradabilidade apresentam e mantêm mais contatos [de Montjoye et al., 2013a], [Chittaranjan et al., 2011]. Além dos traços de personalidade, pesquisas da área de economia e sociologia apontam relação do comportamento humano com características sócio-demográficas. Dentre elas, idade, sexo, escolaridade e status sócio-econômico [DonaldJ.Bogue, 2009]. De acordo com [Mcpherson et al., 2006], mulheres tendem a manter um menor e menos diverso número de contatos; indivíduos nascidos mais recentemente e com status educacional e sócio-econômico superiores apresentam um perfil mais móvel, tendem a possuir um número maior e diverso de contatos e são mais confortáveis com o uso de tecnologia. Defendemos que aspectos como os citados deverão ser avaliados para melhor entender o comportamento humano do usuário e levados em consideração nas redes do futuro. Alguns trabalhos como [Aharony et al., 2011], [de Montjoye et al., 2013a] já mostram como certas características interferem no perfil de mobilidade e comunicação

dos usuários.

Em [Sabharwal e Veeraraghavan, 2017], os autores citam evidências que ligam o comportamento humano à saúde e bem estar. Há um crescimento de evidências que determinados comportamentos trazem benefícios à saúde (ex: alimentação saudável e prática de exercícios). Desde 2000, um grande número de trabalhos se dedicaram a identificar em como a biologia molecular influencia no comportamento de um indivíduo e vice-versa. A partir disso, surgiu o que é chamado de *Bio-Behavioral Sensing*, ou seja a captura de informações de saúde e comportamento. Isso se tornou uma realidade com a disseminação de dispositivos vestíveis (ex: *Fitbit* e *Apple Watch*), antes artefatos apenas de pesquisas mas que hoje já estão disponíveis ao alcance de usuários finais. Por exemplo, dados obtidos por esses dispositivos podem servir para correlacionar o tipo e quantidade de uma atividade física com métricas diferentes de saúde (ex: peso, pressão arterial). Já em [Torous, 2017], os autores citam como o *Big Data* irá revolucionar o tratamento de doenças como depressão e esquizofrenia. Isso se tornará possível com a já esperada incorporação de outros tipos de sensores em *smartphones* e dispositivos vestíveis. Por exemplo, informações como diferenças na linguagem, diminuição do nível de certas atividades e problemas com o sono de uma pessoa poderão ser observadas. Essas informações processadas por *software* especializado auxiliarão no diagnóstico de doenças como depressão e a sugerir tratamento médico ao paciente, além de manter monitoração pós-ocorrido.

Muitas corporações passaram a enxergar as redes sociais como fonte de dados que podem ser úteis em cenários diferentes. De acordo com [Nedunchezian e Jacob, 2017], na perspectiva do usuário, perfis cooperativos ou não, são influenciados pela personalidade. Sendo assim, o comportamento humano tem sido colocado como aspecto diretamente ligado à influência social de um indivíduo [Gillen et al., 2017]. Para modelar características humanas complexas, pesquisadores têm utilizado técnicas de mineração de dados, aprendizado de máquina e mecanismos de extração de informação. Ainda em [Nedunchezian e Jacob, 2017], são citados alguns aspectos do comportamento humano: (a) Contexto - comportamento baseado em características espaço-temporais; (b) Conteúdo - referente ao ato de gostar ou não de um conteúdo; (c) Intenção - classificação de comportamento legítimo ou malicioso; e (d) Social - impacto causado pela atividade de terceiros.

1.8. Considerações finais

À medida em que os humanos ficam cada vez mais íntimos e conectados com seus dispositivos móveis, as futuras arquiteturas de rede terão que entender suas necessidades e expectativas. Sendo assim, conforme discutido neste minicurso, a compreensão do comportamento humano é essencial não apenas para projetar futuras redes móveis e tecnologias relacionadas, mas também para novos modelos de negócios.

Este minicurso abrangeu o paradigma “ciente do humano”, explicando por que e como os seres humanos estarão nos holofotes dos futuros padrões de comunicação. Inicialmente contextualizamos o tema diante do cenário de sobrecarga das redes móveis, disseminação de dispositivos móveis e tecnologias relacionadas. Revisamos a pesquisa relacionada por meio de uma linha do tempo, mostrando como há cerca de uma década a visão centrada no usuário tem caminhado em direção ao “ciente do humano”. Apresenta-

mos aspectos do contexto e comportamento já identificados e que poderão ser utilizados em futuros modelos e arquiteturas de redes móveis. Propusemos uma estrutura para o gerenciamento e a análise de aspectos humanos, contemplando coleta, armazenamento, modelagem, extração de conhecimento, análise e validação. Sequencialmente, contextualizamos e exemplificamos como o “ciente do humano” pode alavancar arquiteturas como *5G*, *IoT*, Redes Veiculares, Redes Ultra-Densas e Redes Centradas na Informação e também modelos/soluções como *Offloading* de dados, Redes Sociais *Online*, Segurança e Privacidade. Finalmente, apresentamos uma seção de avaliação e experimentação com fontes de dados do contexto e comportamento humano, ambientes de validação, desafios de pesquisa e lições adicionais de outras áreas.

Para que o paradigma “ciente do humano” progrida e, assim, a rede se torne mais próxima do humano, há a necessidade de mais análises e extração de conhecimento do “*big data*” dos indivíduos. Estes são essenciais para capturar aspectos comportamentais humanos do usuário da rede e possibilitar propostas com consciência humana. Além disso, há uma demanda por mais pesquisas inter-domínio, agrupando estudos e colaborações de psicologia, estatística, sociologia, redes, aprendizado de máquina, e assim por diante.

Agradecimentos

Agradecemos ao apoio da CAPES, STIC-AmSud, INRIA Saclay, Equipe Associada do INRIA EMBRACE, UFBA, CNPq, FAPERJ e INCT em Ciência de Dados (INCT-CiD).

Referências

- [lic, 2015] (2015). Fast-tracking operator plans to win in the \$5bn location insights market. Disponível em: <https://stlpartners.com/research/fast-tracking-operator-plans-to-win-in-the-5bn-location-insights-market/>.
- [mac, 2015] (2015). Mobile context-adaptive caching for content-centric networking. Disponível em: <http://macaco.inria.fr/>.
- [flu, 2016] (2016). Get to know your customers better with flux vision data. Disponível em: <https://www.orange-business.com/fr/produits/flux-vision>.
- [cis, 2017a] (2017a). Cisco visual networking index: Forecast and methodology, 2016-2021, white paper. Disponível em: <https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.html>.
- [cis, 2017b] (2017b). Cisco visual networking index: Global mobile data traffic forecast update, 2016-2021 white paper. Disponível em: <https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html>.
- [eri, 2017] (2017). Ericsson mobility report. Disponível em: <https://www.ericsson.com/assets/local/mobility-report/documents/2017/ericsson-mobility-report-june-2017.pdf>.

- [tel, 2017] (2017). Transform your value proposition and business. Disponível em: <https://www.business-solutions.telefonica.com/en/products/big-data/business-insights/smart-steps/>.
- [fac, 2018] (2018). Facebook android sdk. Disponível em: <https://developers.facebook.com/docs/android/getting-started>.
- [par, 2018] (2018). Paris data. Disponível em: <https://opendata.paris.fr/page/home/>.
- [tru, 2018] (2018). Revealed: 50 million facebook profiles harvested for cambridge analytica in major data breach - whistleblower describes how firm linked to former trump adviser steve bannon compiled user data to target american voters. <https://www.theguardian.com/news/2018/mar/17/cambridge-analytica-facebook-influence-us-election>.
- [ope, 2018] (2018). The world's largest open database of cell towers. Disponível em: <https://opencellid.org/>.
- [Afrasiabi e Guérin, 2016] Afrasiabi, M. H. e Guérin, R. (2016). Exploring user-provided connectivity. *IEEE/ACM Transactions on Networking*, 24(1):542–554.
- [Aharony et al., 2011] Aharony, N., Pan, W., Ip, C., Khayal, I., e Pentland, A. (2011). Social fmri: Investigating and shaping social mechanisms in the real world. *Pervasive and Mobile Computing*, 7(6):643 – 659. The Ninth Annual IEEE International Conference on Pervasive Computing and Communications (PerCom 2011).
- [Akyildiz et al., 2016] Akyildiz, I. F., Nie, S., Lin, S.-C., e Chandrasekaran, M. (2016). 5g roadmap: 10 key enabling technologies. *Computer Networks*, 106(Supplement C):17 – 48.
- [Al-Kanj et al., 2014] Al-Kanj, L., Poor, H. V., e Dawy, Z. (2014). Optimal cellular offloading via device-to-device communication networks with fairness constraints. *IEEE Transactions on Wireless Communications*, 13:4628–4643.
- [Aldini e Bogliolo, 2016] Aldini, A. e Bogliolo, A. (2016). *User-Centric Networking: Future Perspectives*. Springer Publishing Company, Incorporated, 1st edition.
- [Almendros-Jiménez e Becerra-Terón, 2015] Almendros-Jiménez, J. M. e Becerra-Terón, A. (2015). Querying open street map with xquery. In *2015 1st International Conference on Geographical Information Systems Theory, Applications and Management (GISTAM)*, pages 1–11.
- [Alsadhan e Skillicorn, 2017] Alsadhan, N. e Skillicorn, D. (2017). Estimating personality from social media posts. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 350–356.
- [Altahrawi et al., 2017] Altahrawi, M. A., Ismail, M., Mahdi, H., e Ramli, N. (2017). Routing protocol in a hybrid sensor and vehicular network for different mobility scenario. In *2017 IEEE 13th Malaysia International Conference on Communications (MICC)*, pages 113–118.

- [Araújo et al., 2017] Araújo, A. C. S., Sampaio, L. N., e Ziviani, A. (2017). BEEP: Balancing energy, redundancy, and performance in fat-tree data center networks. *IEEE Internet Computing*, 21(4):44–53.
- [Araújo e Sampaio, 2017] Araújo, F. R. C. e Sampaio, L. N. (2017). Mobilidade em NDN: Consumidores versus Produtores. In *XIII Workshop de Redes P2P, Dinâmicas, Sociais e Orientadas a Conteúdo (WP2P+)*, pages 8–13, Belém, PA. SBC.
- [Asif et al., 2016] Asif, M. T., Mitrovic, N., Dauwels, J., e Jaillet, P. (2016). Matrix and tensor based methods for missing data estimation in large traffic networks. *IEEE Transactions on Intelligent Transportation Systems*, 17(7):1816–1825.
- [Banerjee et al., 2015] Banerjee, A., Cho, J., Eide, E., Duerig, J., Nguyen, B., Ricci, R., Van der Merwe, J., Webb, K., e Wong, G. (2015). Phantomnet: Research infrastructure for mobile networking, cloud computing and software-defined networking. *GetMobile: Mobile Computing and Communications*, 19(2):28–33.
- [Barbera et al., 2014] Barbera, M. V., Viana, A. C., de Amorim, M. D., e Stefa, J. (2014). Data offloading in social mobile networks through vip delegation. *Ad Hoc Networks*, 19:92 – 110.
- [Barnaghi e Sheth, 2016] Barnaghi, P. e Sheth, A. (2016). On searching the internet of things: Requirements and challenges. *IEEE Intelligent Systems*, 31(6):71–75.
- [Blondel et al., 2015] Blondel, V. D., Decuyper, A., e Krings, G. (2015). A survey of results on mobile phone datasets analysis. *EPJ Data Science*, 4(1):10.
- [CC et al., 2016] CC, S., Raychoudhury, V., Marfia, G., e Singla, A. (2016). A survey of routing and data dissemination in delay tolerant networks. *J. Netw. Comput. Appl.*, 67(C):128–146.
- [Chatzikokolakis et al., 2015] Chatzikokolakis, K., Palamidessi, C., e Stronati, M. (2015). Geo-indistinguishability: A principled approach to location privacy. In Natarajan, R., Barua, G., e Patra, M. R., editors, *Distributed Computing and Internet Technology*, pages 49–72, Cham. Springer International Publishing.
- [Chen e Yang, 2016] Chen, B. e Yang, C. (2016). Energy costs for traffic offloading by cache-enabled d2d communications. In *2016 IEEE Wireless Communications and Networking Conference*, pages 1–6.
- [Chen et al., 2017a] Chen, G., Hoteit, S., Viana, A. C., Fiore, M., e Sarraute, C. (2017a). Spatio-temporal predictability of cellular data traffic. *Research Report - INRIA Saclay - Ile-de-France*, (0483).
- [Chen et al., 2017b] Chen, G., Hoteit, S., Viana, A. C., Fiore, M., Sarraute, C., undefined, undefined, undefined, e undefined (2017b). The spatiotemporal interplay of regularity and randomness in cellular data traffic. *2017 IEEE 42nd Conference on Local Computer Networks (LCN)*, 00:187–190.

- [Chen et al., 2016] Chen, S., Qin, F., Hu, B., Li, X., e Chen, Z. (2016). User-centric ultra-dense networks for 5g: challenges, methodologies, and directions. *IEEE Wireless Communications*, 23(2):78–85.
- [Chen e Kountouris, 2015] Chen, Z. e Kountouris, M. (2015). Cache-enabled Small Cell Networks with Local User Interest Correlation. In *IEEE 16th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pages 680–684.
- [Cheng et al., 2015] Cheng, R.-S., Huang, C.-M., e Cheng, G.-S. (2015). A d2d co-operative relay scheme for machine-to-machine communication in the lte-a cellular network. In *2015 International Conference on Information Networking (ICOIN)*, pages 153–158.
- [Chittaranjan et al., 2011] Chittaranjan, G., Blom, J., e Gatica-Perez, D. (2011). Who’s who with big-five: Analyzing and classifying personality traits with smartphones. In *2011 15th Annual International Symposium on Wearable Computers*, pages 29–36.
- [Cho et al., 2016] Cho, J., Sundaresan, K., Mahindra, R., der Merwe, J. K. V., e Rangarajan, S. (2016). Demo: ACACIA – context-aware edge computing for continuous interactive applications over mobile networks. In *ACM MobiCom*.
- [Chuang e Lin, 2012] Chuang, Y. J. e Lin, K. C. J. (2012). Cellular traffic offloading through community-based opportunistic dissemination. In *2012 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 3188–3193.
- [Conan et al., 2007] Conan, V., Leguay, J., e Friedman, T. (2007). Characterizing pairwise inter-contact patterns in delay tolerant networks. In *Proceedings of the 1st International Conference on Autonomic Computing and Communication Systems, Autonomics ’07*, pages 19:1–19:9, ICST, Brussels, Belgium, Belgium. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).
- [Conti et al., 2015] Conti, M., Boldrini, C., Kanhere, S. S., Mingozi, E., Pagani, E., Ruiz, P. M., e Younis, M. (2015). From manet to people-centric networking: Milestones and open research challenges. *Computer Communications*, 71(Supplement C):1 – 21.
- [Courneya e Hellsten, 1998] Courneya, K. S. e Hellsten, L. (1998). Personality correlates of exercise behavior, motives, barriers and preferences: An application of the five-factor model. *Personality and Individual Differences*, 24:625–633.
- [Cuttone et al., 2018] Cuttone, A., Lehmann, S., e González, M. C. (2018). Understanding predictability and exploration in human mobility. *EPJ Data Science*, 7(1):2.
- [Czibula et al., 2009] Czibula, G., Guran, A. M., Czibula, I. G., e Cojocar, G. S. (2009). Ipa - an intelligent personal assistant agent for task performance support. In *2009 IEEE 5th International Conference on Intelligent Computer Communication and Processing*, pages 31–34.

- [da Silva et al., 2016] da Silva, V. B. C., Campista, M. E. M., e Costa, L. H. M. K. (2016). Trac: A trajectory-aware content distribution strategy for vehicular networks. *Vehicular Communications*, 5:18 – 34.
- [Das et al., 2017] Das, S., Leopold, J., Ghosh, S., e Das, S. K. (2017). Graph compaction in analyzing large scale online social networks. In *2017 IEEE International Conference on Communications (ICC)*, pages 1–6.
- [De Hoogh et al., 2005] De Hoogh, A., Den Hartog, D., e L. Koopman, P. (2005). Linking the big five-factors of personality to charismatic and transactional leadership; perceived dynamic work environment as a moderator. *Journal of Organizational Behavior*, 26:839 – 865.
- [de Melo et al., 2015] de Melo, P. O. V., Viana, A. C., Fiore, M., Jaffrès-Runser, K., Mouël, F. L., Loureiro, A. A., Addepalli, L., e Guangshuo, C. (2015). Recast: Telling apart social and random relationships in dynamic networks. *Performance Evaluation*, 87:19 – 36. Special Issue: Recent Advances in Modeling and Performance Evaluation in Wireless and Mobile Systems.
- [de Montjoye et al., 2013a] de Montjoye, Y.-A., Hidalgo, C. A., Verleysen, M., e Blondel, V. D. (2013a). Unique in the crowd: The privacy bounds of human mobility. *Nature Scientific Reports*, 3(1376).
- [de Montjoye et al., 2013b] de Montjoye, Y.-A., Quoidbach, J., Robic, F., e Pentland, A. (2013b). Predicting personality using novel mobile phone-based metrics. In Greenberg, A. M., Kennedy, W. G., e Bos, N. D., editors, *Social Computing, Behavioral-Cultural Modeling and Prediction*, pages 48–55, Berlin, Heidelberg. Springer Berlin Heidelberg.
- [de Montjoye et al., 2015] de Montjoye, Y.-A., Radaelli, L., Singh, V. K., e Pentland, A. “. (2015). Unique in the shopping mall: On the reidentifiability of credit card metadata. *Science*, 347(6221):536–539.
- [Digman, 1990] Digman, J. M. (1990). Personality structure: Emergence of the five-factor model. *Annual Review of Psychology*, 41(1):417–440.
- [Ding et al., 2013] Ding, A. Y., Han, B., Xiao, Y., Hui, P., Srinivasan, A., Kojo, M., e Tarkoma, S. (2013). Enabling energy-aware collaborative mobile data offloading for smartphones. In *2013 IEEE International Conference on Sensing, Communications and Networking (SECON)*, pages 487–495.
- [DonaldJ.Bogue, 2009] DonaldJ.Bogue (2009). Century of difference: How america changed in the last one hundred years by claude s. fischer and michael hout. *American Journal of Sociology*, 114(5):1534–1536.
- [Eagle e (Sandy) Pentland, 2006] Eagle, N. e (Sandy) Pentland, A. (2006). Reality mining: Sensing complex social systems. *Personal Ubiquitous Comput.*, 10(4):255–268.

- [Farahat e Hassanein, 2016] Farahat, H. e Hassanein, H. (2016). Optimal caching for producer mobility support in Named Data Networks. In *2016 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE.
- [Favaretto et al., 2017] Favaretto, R. M., Dihl, L., Musse, S. R., Vilanova, F., e Costa, A. B. (2017). Using big five personality model to detect cultural aspects in crowds. In *2017 30th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*, pages 223–229.
- [Fawaz e Shin, 2014] Fawaz, K. e Shin, K. G. (2014). Location privacy protection for smartphone users. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, CCS '14*, pages 239–250, New York, NY, USA. ACM.
- [Feki et al., 2016] Feki, A., Duarte, M., Valentin, S., e Rose, L. (2016). Exploiting geographical context in d2d communications. In *2016 IEEE 84th Vehicular Technology Conference (VTC-Fall)*, pages 1–6.
- [Fontes et al., 2015] Fontes, R. R., Afzal, S., Brito, S. H. B., Santos, M. A. S., e Rothenberg, C. E. (2015). Mininet-wifi: Emulating software-defined wireless networks. In *2015 11th International Conference on Network and Service Management (CNSM)*, pages 384–389.
- [Gandotra et al., 2017] Gandotra, P., Jha, R. K., e Jain, S. (2017). A survey on device-to-device (d2d) communication: Architecture and security issues. *Journal of Network and Computer Applications*, 78(Supplement C):9 – 29.
- [Ge et al., 2016] Ge, J., Wang, S., Wu, Y., Tang, H., e E, Y. (2016). Performance improvement for source mobility in named data networking based on global–local FIB updates. *Peer-to-Peer Networking and Applications*, 9(4):670–680.
- [Gillen et al., 2017] Gillen, J. J., Freeman, M., e Tootell, H. (2017). Human behavior in online social networks. In *2017 IEEE International Symposium on Technology and Society (ISTAS)*, pages 1–6.
- [Goldberg, 1993] Goldberg, L. R. (1993). The structure of phenotypic personality traits. *American Psychologist*, 48:26–34.
- [Goldberg, 1999] Goldberg, L. R. (1999). A broad-bandwidth, public domain, personality inventory measuring the lower-level facets of several five-factor models. *I. Mervielde, I. Deary, F. De Fruyt and F. Ostendorf, Eds., Personality Psychology in Europe, Tilburg University Press, Tilburg*, pages 7–28.
- [Gonzalez et al., 2008] Gonzalez, M. C., Hidalgo, C. A., e Barabasi, A.-L. (2008). Understanding individual human mobility patterns. *Nature*, 453(7196):779–782.
- [Hahm et al., 2017] Hahm, O., Baccelli, E., Schmidt, T. C., Wählisch, M., Adjih, C., e Massoulié, L. (2017). Low-power Internet of Things with NDN & Cooperative Caching. In *Proceedings of the 4th ACM Conference on Information-Centric Networking, ICN '17*, pages 98–108, New York, NY, USA. ACM.

- [Hajri e Assaad, 2016] Hajri, S. E. e Assaad, M. (2016). Caching Improvement Using Adaptive User Clustering. *arXiv preprint arXiv:1605.09602*.
- [Haus et al., 2017] Haus, M., Waqas, M., Ding, A. Y., Li, Y., Tarkoma, S., e Ott, J. (2017). Security and privacy in device-to-device (D2D) communication: A review. *IEEE Communications Surveys & Tutorials*, 19(2):1054–1079.
- [Hejazi et al., 2018] Hejazi, H., Rajab, H., Cinkler, T., e Lengyel, L. (2018). Survey of platforms for massive iot. In *2018 IEEE International Conference on Future IoT Technologies (Future IoT)*, pages 1–8.
- [Hu et al., 2015] Hu, X., Chu, T. H. S., Leung, V. C. M., Ngai, E. C. H., Kruchten, P., e Chan, H. C. B. (2015). A survey on mobile social networks: Applications, platforms, system architectures, and future research directions. *IEEE Communications Surveys & Tutorials*, 17(3):1557–1581.
- [Hubaux e Juels, 2016] Hubaux, J.-P. e Juels, A. (2016). Privacy is dead, long live privacy. *Commun. ACM*, 59(6):39–41.
- [I et al., 2014] I, C. L., Rowell, C., Han, S., Xu, Z., Li, G., e Pan, Z. (2014). Toward green and soft: a 5g perspective. *IEEE Communications Magazine*, 52(2):66–73.
- [Ioannou e Weber, 2016] Ioannou, A. e Weber, S. (2016). A survey of caching policies and forwarding mechanisms in information-centric networking. *IEEE Communications Surveys & Tutorials*, 18(4):2847–2886.
- [Iosifidis et al., 2017] Iosifidis, G., Gao, L., Huang, J., e Tassiulas, L. (2017). Efficient and fair collaborative mobile internet access. *IEEE/ACM Transactions on Networking*, 25(3):1386–1400.
- [Jacobson et al., 2009] Jacobson, V., Smetters, D. K., Thornton, J. D., Plass, M. F., Briggs, N. H., e Braynard, R. L. (2009). Networking named content. In *Proceedings of the 5th international conference on Emerging networking experiments and technologies*, pages 1–12. ACM.
- [Jeong et al., 2016] Jeong, J., Leconte, M., e Proutiere, A. (2016). Cluster-aided mobility predictions. In *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, pages 1–9.
- [Judge et al., 2006] Judge, T. A., Higgins, C. A., Thoresen, C. J., e Barrick, M. R. (2006). The big five personality traits, general mental ability, and career success across the life span. *Personnel Psychology*, 52:621 – 652.
- [Katsarakis et al., 2015] Katsarakis, M., Theodosiadis, V., e Papadopouli, M. (2015). Evaluation of a user-centric qoe-based recommendation tool for wireless access. In *Proceedings of the 2015 ACM SIGCOMM Workshop on Crowdsourcing and Crowdsharing of Big (Internet) Data, C2B(1)D '15*, pages 39–44, New York, NY, USA. ACM.

- [Katsikouli et al., 2017] Katsikouli, P., Viana, A., Fiore, M., e Tarable, A. (2017). On the sampling frequency of human mobility. In *Proceedings of the GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, pages 1–6. IEEE.
- [Kopacz, 2005] Kopacz, M. (2005). Personality and music preferences: The influence of personality traits on preferences regarding musical elements. *Journal of music therapy*, 42:216–39.
- [Kotz et al., 2009] Kotz, D., Henderson, T., Abyzov, I., e Yeo, J. (2009). CRAWDAD dataset dartmouth/campus (v. 2009-09-09). Downloaded from <https://crawdad.org/dartmouth/campus/20090909>.
- [Koziol et al., 2017] Koziol, D., Moya, F. S., Yu, L., Phan, V. V., e Xu, S. (2017). Qos and service continuity in 3gpp d2d for iot and wearables. In *2017 IEEE Conference on Standards for Communications and Networking (CSCN)*, pages 233–239.
- [Lau et al., 2017] Lau, C. P., Alabbasi, A., e Shihada, B. (2017). On the analysis of human mobility model for content broadcasting in 5g networks. In *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pages 1–7.
- [Lauridsen et al., 2017] Lauridsen, M., Gimenez, L. C., Rodriguez, I., Sorensen, T. B., e Mogensen, P. (2017). From lte to 5g for connected mobility. *IEEE Communications Magazine*, 55(3):156–162.
- [Lehmann et al., 2016] Lehmann, M. B., Barcellos, M. P., e Mauthe, A. (2016). Providing producer mobility support in NDN through proactive data replication. In *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*, pages 383–391. IEEE.
- [Li et al., 2017] Li, Y., Gao, Z., Huang, L., Du, X., e Guizani, M. (2017). Resource management for future mobile networks: Architecture and technologies. *Computer Networks*, 129:392–398.
- [Li et al., 2014] Li, Y., Wu, T., Hui, P., Jin, D., e Chen, S. (2014). Social-aware d2d communications: qualitative insights and quantitative analysis. *IEEE Communications Magazine*, 52(6):150–158.
- [Luzio et al., 2016] Luzio, A. D., Mei, A., e Stefa, J. (2016). Mind your probes: De-anonymization of large crowds through smartphone wifi probe requests. In *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, pages 1–9.
- [López-Pérez et al., 2015] López-Pérez, D., Ding, M., Claussen, H., e Jafari, A. H. (2015). Towards 1 gbps/ue in cellular systems: Understanding ultra-dense small cell deployments. *IEEE Communications Surveys & Tutorials*, 17(4):2078–2101.
- [MacHardy et al., 2018] MacHardy, Z., Khan, A., Obana, K., e Iwashina, S. (2018). V2x access technologies: Regulation, research, and remaining challenges. *IEEE Communications Surveys & Tutorials*. Aceito para publicação.

- [Mantas et al., 2017] Mantas, N., Louta, M., Karapistoli, E., Karetsos, G. T., Kraounakis, S., e Obaidat, M. S. (2017). Towards an incentive-compatible, reputation-based framework for stimulating cooperation in opportunistic networks: a survey. *IET Networks*, 6(6):169–178.
- [Marques-Neto et al., 2018] Marques-Neto, H. T., Xavier, F. H. Z., Xavier, W. Z., Malab, C. H. S., Ziviani, A., Silveira, L. M., e Almeida, J. M. (2018). Understanding human mobility and workload dynamics due to different large-scale events using mobile phone data. *Journal of Network and System Management*. Aceito para publicação.
- [Masucci et al., 2017] Masucci, A. M., Elayoubi, S. E., e Gati, A. (2017). D2d-assisted sleep mode strategies for green mobile networks. In *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pages 1–6.
- [Matarazzo et al., 2018] Matarazzo, T. J., Santi, P., Pakzad, S. N., Carter, K., Ratti, C., Moaveni, B., Osgood, C., e Jacob, N. (2018). Crowdsensing framework for monitoring bridge vibrations using moving smartphones. *Proceedings of the IEEE*, 106(4):577–593.
- [McCrae e John, 1992] McCrae, R. R. e John, O. P. (1992). An introduction to the five-factor model and its applications. *Journal of personality*, 60:175–215.
- [Mcpherson et al., 2006] Mcpherson, M., Smith-Lovin, L., e Brashears, M. (2006). Social isolation in america: Changes in core discussion networks over two decades. 71:353–375.
- [Medhat et al., 2017] Medhat, A. M., Carella, G. A., Pauls, M., e Magedanz, T. (2017). Extensible framework for elastic orchestration of service function chains in 5g networks. In *2017 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, pages 327–333.
- [Meng et al., 2017] Meng, Y., Jiang, C., Chen, H. H., e Ren, Y. (2017). Cooperative device-to-device communications: Social networking perspectives. *IEEE Network*, 31(3):38–44.
- [Mtibaa et al., 2006] Mtibaa, A., Chaintreau, A., e Diot, C. (2006). Detail characterization of paths in pocket switched networks. In *Proceedings of the 2006 ACM CoNEXT Conference*, CoNEXT '06, pages 59:1–59:2, New York, NY, USA. ACM.
- [Naboulsi et al., 2016] Naboulsi, D., Fiore, M., Ribot, S., e Stanica, R. (2016). Large-scale mobile traffic analysis: A survey. *IEEE Communications Surveys & Tutorials*, 18(1):124–161.
- [Name et al., 2017] Name, H. A. M., Oladipo, F. O., e Ariwa, E. (2017). User mobility and resource scheduling and management in fog computing to support iot devices. In *2017 Seventh International Conference on Innovative Computing Technology (IN-TECH)*, pages 191–196.

- [Nedunchezian e Jacob, 2017] Nedunchezian, P. e Jacob, S. G. (2017). Social influence algorithms and emotion classification for prediction of human behavior: A survey. In *2017 Second International Conference on Recent Trends and Challenges in Computational Models (ICRTCCM)*, pages 55–60.
- [Ning et al., 2016] Ning, Z., Xia, F., Kong, X., e Chen, Z. (2016). Social-oriented resource management in cloud-based mobile networks. *IEEE Cloud Computing*, 3(4):24–31.
- [Nunes et al., 2016] Nunes, I. O., de Melo, P. O. S. V., e Loureiro, A. A. F. (2016). Leveraging d2d multihop communication through social group meeting awareness. *IEEE Wireless Communications*, 23(4):12–19.
- [Oliveira et al., 2017] Oliveira, E. M. R., Viana, A., Naveen, K. P., e Sarraute, C. (2017). Mobile data traffic modeling: Revealing temporal facets. *Computer Networks*, 112:176–193.
- [Oliveira e Viana, 2014] Oliveira, E. M. R. e Viana, A. C. (2014). From routine to network deployment for data offloading in metropolitan areas. In *2014 Eleventh Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, pages 126–134.
- [Oliveira et al., 2016] Oliveira, E. M. R., Viana, A. C., Sarraute, C., Brea, J., e Alvarez-Hamelin, I. (2016). On the regularity of human mobility. *Pervasive and Mobile Computing*, 33(Supplement C):73 – 90.
- [Orzeck e Lung, 2005] Orzeck, T. e Lung, E. (2005). Big-five personality differences of cheaters and non-cheaters. *Current Psychology*, 24:274–287.
- [Pambudi et al., 2017] Pambudi, S. A., Wang, W., e Wang, C. (2017). From isolation time to node resilience: Impact of cascades in d2d-based social networks. In *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, pages 1–6.
- [Panwar et al., 2017] Panwar, G., Tourani, R., Mick, T., Mtibaa, A., e Misra, S. (2017). DICE: Dynamic Multi-RAT Selection in the ICN-enabled Wireless Edge. In *Proceedings of the Workshop on Mobility in the Evolving Internet Architecture, MobiArch '17*, pages 31–36, New York, NY, USA. ACM.
- [Panwar et al., 2016] Panwar, N., Sharma, S., e Singh, A. K. (2016). A survey on 5g: The next generation of mobile communication. *Physical Communication*, 18, Part 2:64 – 84. Special Issue on Radio Access Network Architectures and Resource Management for 5G.
- [Pentland, 2012] Pentland, A. . (2012). Society’s nervous system: Building effective government, energy, and public health systems. *Computer*, 45(1):31–38.
- [Qiu et al., 2018] Qiu, T., Chen, N., Li, K., Atiquzzaman, M., e Zhao, W. (2018). How can heterogeneous internet of things build our future: A survey. *IEEE Communications Surveys Tutorials*, PP(99):1–1.

- [Quintero et al., 2016] Quintero, D. M. M., Mejia, J. C. G., e Builes, J. A. J. (2016). Model of human knowledge extraction for a kbs using reasoning cases. *IEEE Latin America Transactions*, 14(3):1466–1472.
- [Rebecchi et al., 2015] Rebecchi, F., de Amorim, M. D., Conan, V., Passarella, A., Bruno, R., e Conti, M. (2015). Data offloading techniques in cellular networks: A survey. *IEEE Communications Surveys & Tutorials*, 17(2):580–603.
- [Rehman e Kim, 2017] Rehman, R. A. e Kim, B. (2017). LOMCF: Forwarding and Caching in Named Data Networking Based MANETs. *IEEE Transactions on Vehicular Technology*, 66(10):9350–9364.
- [Ren et al., 2018] Ren, X., Yu, C. M., Yu, W., Yang, S., Yang, X., McCann, J. A., e Yu, P. S. (2018). Lopub: High-dimensional crowdsourced data publication with local differential privacy. *IEEE Transactions on Information Forensics and Security*, PP(99):1–1.
- [Ren et al., 2015] Ren, Y., Wang, C., Liu, D., Liu, F., e Liu, E. (2015). Applying lte-d2d to support v2v communication using local geographic knowledge. In *2015 IEEE 82nd Vehicular Technology Conference (VTC2015-Fall)*, pages 1–5.
- [Sabharwal e Veeraraghavan, 2017] Sabharwal, A. e Veeraraghavan, A. (2017). Bio-behavioral sensing: An emerging engineering area. *GetMobile: Mobile Comp. and Comm.*, 21(3):11–18.
- [Silva et al., 2016] Silva, T., Celes, C., Borges Neto, J., Mota, V., Felipe, C., Ferreira, A., Ribeiro, A., Vaz de Melo, P., Almeida, J., e Loureiro, A. (2016). Users in the urban sensing process: Challenges and research opportunities. In *Pervasive Computing: Next Generation Platforms for Intelligent Data Collection*, pages 45–95.
- [Silveira et al., 2016] Silveira, L. M., Almeida, J. M., Marques-Neto, H. T., Sarraute, C., e Ziviani, A. (2016). Mobhet: Predicting human mobility using heterogeneous data sources. *Computer Communications*, 95:54–68.
- [Subrahmanian e Kumar, 2017] Subrahmanian, V. S. e Kumar, S. (2017). Predicting human behavior: The next frontiers. *Science*, 355(6324):489–489.
- [Tayeb et al., 2017] Tayeb, S., Latifi, S., e Kim, Y. (2017). A survey on iot communication and computation frameworks: An industrial perspective. In *2017 IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 1–6.
- [Teijeiro-Mosquera et al., 2015] Teijeiro-Mosquera, L., Biel, J. I., Alba-Castro, J. L., e Gatica-Perez, D. (2015). What your face vlogs about: Expressions of emotion and big-five traits impressions in youtube. *IEEE Transactions on Affective Computing*, 6(2):193–205.
- [Thilakarathna et al., 2014] Thilakarathna, K., Seneviratne, A., Viana, A. C., e Petander, H. (2014). User generated content dissemination in mobile social networks through infrastructure supported content replication. *Pervasive and Mobile Computing*, 11:132 – 147.

- [Thilakarathna et al., 2016] Thilakarathna, K., Seneviratne, S., Gupta, K., Kaafar, M. A., e Seneviratne, A. (2016). A deep dive into location-based communities in social discovery networks. *Computer Communications*, 100:78–90.
- [Thilakarathna et al., 2017] Thilakarathna, K., Viana, A. C., Seneviratne, A., e Petander, H. (2017). Design and analysis of an efficient friend-to-friend content dissemination system. *IEEE Transactions on Mobile Computing*, 16(3):702–715.
- [Torous, 2017] Torous, J. (2017). Digital psychiatry. *IEEE Spectrum*, 54(7):45–50.
- [Vu et al., 2015] Vu, X.-T., Abel, M.-H., e Morizet-Mahoudeaux, P. (2015). A user-centered and group-based approach for social data filtering and sharing. *Computers in Human Behavior*, 51:1012–1023.
- [Wagih et al., 2017] Wagih, H., Mokhtar, H., e Ghoniemy, S. (2017). Location recommendation based on social trust. In *2017 13th International Conference on Semantics, Knowledge and Grids (SKG)*, pages 50–55.
- [Wagner et al., 2014] Wagner, D. T., Rice, A., e Beresford, A. R. (2014). Device analyzer: Large-scale mobile data collection. *SIGMETRICS Perform. Eval. Rev.*, 41(4):53–56.
- [Wang et al., 2018] Wang, E. K., Li, Y., Ye, Y., Yiu, S. M., e Hui, L. C. K. (2018). A dynamic trust framework for opportunistic mobile social networks. *IEEE Transactions on Network and Service Management*, 15(1):319–329.
- [Wang et al., 2014] Wang, X., Chen, M., Han, Z., Wu, D. O., e Kwon, T. T. (2014). Toss: Traffic offloading by social network service-based opportunistic sharing in mobile social networks. In *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*, pages 2346–2354.
- [Wang et al., 2015] Wang, X., Li, X., e Leung, V. C. M. (2015). Tasa: traffic offloading by tag-assisted social-aware opportunistic sharing in mobile social networks. In *The 21st IEEE International Workshop on Local and Metropolitan Area Networks*, pages 1–6.
- [Wang et al., 2009] Wang, Y., Krishnamachari, B., e Valente, T. W. (2009). Findings from an empirical study of fine-grained human social contacts. In *2009 Sixth International Conference on Wireless On-Demand Network Systems and Services*, pages 153–160.
- [Wang et al., 2017] Wang, Z., Sun, L., Zhang, M., Pang, H., Tian, E., e Zhu, W. (2017). Propagation- and mobility-aware d2d social content replication. *IEEE Transactions on Mobile Computing*, 16(4):1107–1120.
- [Whitbeck et al., 2011] Whitbeck, J., Amorim, M., Lopez, Y., Leguay, J., e Conan, V. (2011). Relieving the wireless infrastructure: When opportunistic networks meet guaranteed delays. In *2011 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks*, pages 1–10.

- [Xia et al., 2018] Xia, F., Wang, J., Kong, X., Wang, Z., Li, J., e Liu, C. (2018). Exploring human mobility patterns in urban scenarios: A trajectory data perspective. *IEEE Communications Magazine*, 56(3):142–149.
- [Xylomenos et al., 2014] Xylomenos, G., Ververidis, C. N., Siris, V. A., Fotiou, N., Tsiolopoulos, C., Vasilakos, X., Katsaros, K. V., e Polyzos, G. C. (2014). A survey of information-centric networking research. *IEEE Communications Surveys & Tutorials*, 16(2):1024–1049.
- [Yu et al., 2016] Yu, W., Xu, H., Zhang, H., Griffith, D., e Golmie, N. (2016). Ultra-dense networks: Survey of state of the art and future directions. In *2016 25th International Conference on Computer Communication and Networks (ICCCN)*, pages 1–10.
- [Zang et al., 2017] Zang, S., Bao, W., Yeoh, P. L., Chen, H., Lin, Z., Vucetic, B., e Li, Y. (2017). Mobility handover optimization in millimeter wave heterogeneous networks. In *2017 17th International Symposium on Communications and Information Technologies (ISCIT)*, pages 1–6.
- [Zhang et al., 2014] Zhang, L., Afanasyev, A., Burke, J., Jacobson, V., Crowley, P., Papadopoulos, C., Wang, L., Zhang, B., et al. (2014). Named data networking. *ACM SIGCOMM Computer Communication Review*, 44(3):66–73.
- [Zhang et al., 2017] Zhang, X., Guo, L., Li, M., e Fang, Y. (2017). Motivating human-enabled mobile participation for data offloading. *IEEE Transactions on Mobile Computing*, PP(99):1–1.
- [Zheng et al., 2016] Zheng, K., Yang, Z., Zhang, K., Chatzimisios, P., Yang, K., e Xiang, W. (2016). Big data-driven optimization for mobile networks toward 5g. *IEEE Network*, 30(1):44–51.
- [Zheng et al., 2009] Zheng, Y., Zhang, L., Xie, X., e Ma, W.-Y. (2009). Mining interesting locations and travel sequences from gps trajectories. In *Proc. of the WWW, WWW '09*, pages 791–800. ACM.
- [Zhou et al., 2018] Zhou, Z., Yu, H., Xu, C., Zhang, Y., Mumtaz, S., e Rodriguez, J. (2018). Dependable content distribution in d2d-based cooperative vehicular networks: A big data-integrated coalition game approach. *IEEE Transactions on Intelligent Transportation Systems*, 19(3):953–964.
- [Zou et al., 2017] Zou, Y., Liu, W., Wu, K., e Ni, L. M. (2017). Wi-fi radar: Recognizing human behavior with commodity wi-fi. *IEEE Communications Magazine*, 55(10):105–111.

Capítulo

2

Internet-das-Bionano-Coisas: Conectando-se às Nanomáquinas

Michael Taynnan Barros (TSSG - Waterford Institute of Technology, Irlanda), Ligia Francielle Borges (Universidade Federal do Paraná), Carlos Danilo Miranda Regis (Instituto Federal de Educação, Ciência e Tecnologia da Paraíba), Michele Nogueira (Universidade Federal do Paraná), Antonio Loureiro (Universidade Federal de Minas Gerais)

Abstract

The Internet-of-things attracts the attention of many researchers in computer networks with the challenge of providing connectivity to a huge quantity of devices. This reality can be further complicated once again with the recent proposed Internet-of-bionano-things. Nanomachines, natural or synthetic, will be able to communicate to each other and to the Internet through the means of communication systems that are being developed at the nano-scale with the goal of cooperatively executing complex tasks. This technology requires a complete revision of the TCP/IP architecture to accommodate the requirements and demands of the nanonetworks. This chapter aims at introducing this research field to the computer network community, presenting the different types of communicating networks, an initial reformulation of the TCP/IP architecture, research challenges and the applications for the nanonetworks. This technology enables a revolution in the society and affects directly areas, such as medicine, agriculture, pollution and even industry.

Resumo

A Internet-das-coisas atrai a atenção de muitos pesquisadores da área de redes de computadores pela possibilidade de prover conectividade a uma vasta quantidade de dispositivos. Esta realidade está mais uma vez mudando com os avanços na concepção da Internet-das-bionano-coisas. As nanomáquinas, naturais ou sintéticas, podem comunicar entre si e com a Internet por intermédio de sistemas de comunicações que vêm sendo desenvolvidos na escala nano e com o objetivo de cooperativamente executar tarefas complexas. Essa tecnologia requer uma revisão da atual arquitetura TCP/IP a fim de acomodar os requisitos e as demandas das nanorredes. Este capítulo tem o objetivo de introduzir essa área de pesquisa à comunidade de redes de computadores, apresentando os diferentes tipos de redes de comunicações, uma reformulação inicial da arquitetura

TCP/IP, desafios de pesquisa e as aplicações das nanorredes. O desenvolvimento dessa tecnologia permite uma revolução da sociedade que conhecemos, e afetará diretamente áreas como medicina, agricultura, poluição e até indústria.

2.1. Introdução

Desde a introdução das nanomáquinas pelo vencedor do prêmio Nobel de 2016, Sir Fraser Stoddart [1], a comunidade científica de muitas áreas está passando por mudanças que nos permitem explorar o sensoriamento, o processamento e a manipulação de informação na escala nano. As nanomáquinas são dispositivos limitados que possuem a característica de executar funções bem simples [2]. Entretanto, existem propostas para permitir a comunicação entre essas nanomáquinas em uma rede, favorecendo o trabalho colaborativo para executar tarefas complexas, e deste modo, aumentar a capacidade funcional desses pequenos dispositivos.

A nanorrobótica é uma tecnologia que cria máquinas à escala de um nanômetro (10^{-9} metros). Mais especificamente, a nanorrobótica recorre em grande parte à disciplina teórica da engenharia da nanotecnologia, da disciplina de projeto e construção de nanorrobôs. As nanomáquinas são dispositivos que variam no tamanho de 0.1-10 micrômetros e são construídas à escala nanométrica ou de componentes moleculares [3]. Uma outra definição para nanomáquinas usada por vezes, refere-se a um robô que permite a interação da precisão com objetos à escala nanométrica, ou pode manipular com definição a essa escala. Esta última definição não é utilizada neste capítulo, haja visto que poderia englobar um instrumento grande, como um microscópio atômico quando configurado para executar manipulações à escala nanométrica. O que não segue a ideia principal das nanomáquinas enfocada neste trabalho.

Existem dois tipos de nanomáquinas: as naturais e as sintéticas. As nanomáquinas naturais são encontradas em todos os lugares na natureza e são responsáveis pela contração muscular, pela locomoção das bactérias e dos espermatozoides, pela divisão celular, pela replicação de DNA e outros. As nanomáquinas sintéticas são aquelas produzidas pelo homem. Na Figura 2.1, ilustramos esses dois tipos de nanomáquinas. Na Figura 2.1(a), percebemos uma nanomáquina natural, a enzima F0F1-ATP *sintase*. Esta enzima está localizada na membrana interna da mitocôndria e a função está intimamente relacionada com a cadeia respiratória mitocondrial. Na Figura 2.1(b), ilustramos uma nanomáquina sintética que armazena energia da luz [4]. Tibor Kudernac desenvolveu blocos químicos básicos que se agrupam naturalmente por automontagem para formar túbulos, pequenos canos com até um micrômetro de comprimento e poucos nanômetros de diâmetro. Quando os túbulos são iluminados, eles absorvem os fótons, fazendo com que uma tensão mecânica vá se acumulando em sua estrutura até atingir um valor limite. Quando a estrutura não suporta mais energia, ela desmorona abruptamente, liberando a energia. Os microtúbulos sintéticos convertem a energia da luz em uma energia de deformação, como se fosse uma minúscula explosão. Esta explosão pode então ser usada para alimentar a resposta mecânica específica que se desejar. Embora as nanomáquinas sintéticas sejam mais fáceis de serem manipuladas, as nanomáquinas naturais, quando manipuladas, são as que permitem um impacto que jamais foi visto na comunidade científica.

A biotecnologia, a nanotecnologia e, mais recentemente, a teoria e tecnologia da

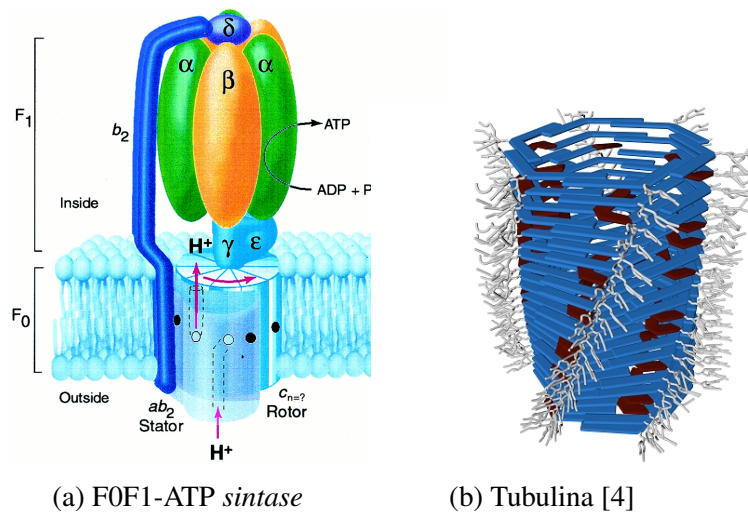


Figura 2.1: Nanomáquinas.

informação vêm se unindo cada vez mais para desenvolver sistemas de comunicações biológicos conectados entre si e à Internet. Esse movimento denomina-se de Internet-das-bionano-coisas. Ele tem a promessa de fazer também o sensoriamento, processamento e manipulação de informação em sistemas de comunicações biológicos, incluindo as nanomáquinas sintéticas. Com isso podemos imaginar que as tecnologias futuras compostas por nanomáquinas sejam organizadas ao ponto de executarem tarefas complexas, mas que sejam controladas remotamente, bem como também, autonomicamente. O sucesso dessa tecnologia permitirá uma mudança total da sociedade em que vivemos, por impactar em diversas áreas como: medicina, agricultura, poluição, e até indústria.

A comunidade de redes de computadores tem uma função fundamental no desenvolvimento da Internet-das-bionano-coisas, que é desenvolver as redes de comunicações em nanoescala. Mas também tem a missão de criar protocolos que viabilizem o acesso eficiente a informações dessa escala a partir da Internet. Como essa tecnologia não está necessariamente ligada a transmissão de informação por ondas eletromagnéticas, mas também por ondas acústicas e moleculares, todo o conceito de redes de comunicações deve ser avaliado e possivelmente reformulado para acomodar a nova tecnologia. Isso deve levar em conta o aumento exponencial da quantidade de dispositivos conectados (nós) e demandas da rede. Para introduzir essa área para a comunidade científica de redes de computadores no Brasil, este capítulo de livro apresenta de uma maneira geral a Internet-das-bionano-coisas.

2.1.1. Como as nanorredes podem transformar a sociedade de hoje

As nanorredes estão transformando a sociedade atual por possibilitar que informações mais precisas possam ser extraídas do ambiente em que vivemos. Isso também vem afetando diretamente o paradigma atual da Internet por expandir seu horizonte de acesso ao meio, coletar informações e desafiar a construção de novas formas de codificação, protocolos e transmissões. A Internet é, sem dúvida, a tecnologia decisiva da Era da Informação e, com a explosão da comunicação sem fio no início do século XXI, podemos dizer que a humanidade está quase totalmente conectada em um nível macro, embora com grandes desigualdades em largura de banda, eficiência e preço. As pessoas, empresas e

instituições vêm sentindo nas últimas décadas o impacto gerado por essa mudança tecnológica, desencadeando diferentes percepções, algumas utópicas e outras distópicas. Por exemplo, a mídia frequentemente relata que o intenso uso da Internet aumenta o risco de isolamento, alienação e afastamento da sociedade, mas as evidências disponíveis mostram que a Internet não isola as pessoas nem reduz sua sociabilidade; na verdade, aumenta a sociabilidade, o engajamento cívico e a intensidade das relações familiares e de amizade, em todas as culturas.

No momento, percebemos uma nova fase de transformações acontecerem na nossa atual “sociedade em rede”. Nossa sociedade já é o produto da revolução digital no nível macro e de importantes mudanças socioculturais. Uma delas é a ascensão da “sociedade centrada no eu”, marcada por um foco maior no crescimento individual e um declínio na comunidade, entendido em termos de espaço, trabalho, família e atribuição em geral. Mas a individualização não significa isolamento ou o fim da comunidade. Em vez disso, as relações sociais estão sendo reconstruídas com base em interesses, valores e projetos individuais. Com os avanços e desenvolvimentos nas redes em escala nano e suas interconexões com a Internet, um espaço para novas percepções, talvez utópicas e distópicas novamente, se abre. Diferentes aplicações são previstas para as nanorredes em áreas como medicina, neurociência, agricultura de precisão, meio ambiente e outras, conforme detalhado na Seção 2.5. Exemplos concretos são encontrados na defesa contra o câncer, na agricultura de precisão e outros.

Com as nanorredes, as fronteiras da Internet se expandem. Teremos acesso a dados online com granularidade mais fina, extraídos de nossos corpos, dos vegetais e do nosso meio ambiente. A sociedade se enriquecerá com dados que permitirão tomadas de decisões mais acertadas favorecendo uma maior eficiência nos nossos sistemas econômicos, biológicos, tecnológicos, ambientais, de transporte e de saúde. Tais redes permitirão uma interação através da Internet com sistemas em escala nano contribuindo para sanar problemas ou doenças o quanto antes. Um melhor planejamento da sociedade será possível graças aos dados coletados e analisados. Todos esses aspectos têm o potencial de trazer impactos positivos para a nossa sociedade acelerando o seu desenvolvimento. Porém, em paralelo à coleta e transmissões desses dados através da Internet, as nanorredes podem trazer desvantagens para a sociedade relacionadas à privacidade dos dados, vulnerabilidades de segurança, injeção de dados, e ameaças relacionadas ao controle de nanomáquinas à distância pela Internet. De qualquer forma, neste minicursos a lista de possíveis desvantagens é vista como oportunidades e desafios de pesquisa que precisam ser tratados ao logo do desenvolvimento do campo das nanorredes.

2.1.2. Estágio atual de desenvolvimento da tecnologia das nanorredes

As comunidade dos pesquisadores das nanorredes nos últimos 10 anos vêm se concentrando mais no desenvolvimento teórico do que no desenvolvimento prático. Como essa área de pesquisa se iniciou dentro da engenharia de telecomunicações, e logo depois, das redes de computadores, a integração com experimentos que envolvem tanto nanomáquinas sintéticas e biológicas ainda é um desafio. Existe um extensivo diálogo sendo feito para que financiamento seja disponibilizado para o puro desenvolvimento dos modelos de comunicação que iremos detalhar ao longo deste capítulo. Dos pesquisadores mais experientes aos mais jovens, é esperada uma constante integração das suas pesquisas com

biólogos, engenheiros de materiais e especialistas em nanotecnologia. Com uma equipe interdisciplinar, o avanço dessa tecnologia para experimentos em direção da produção de novos produtos e a comercialização desses será bem mais eficiente.

Baseado-se na arquitetura de rede TCP/IP convencional, podemos inferir o cumulativo desenvolvimento de trabalhos desde o nascimento da área das nanorredes até o período de desenvolvimento deste capítulo. A maioria dos esforços dos pesquisadores se encontram na camada física. O interesse pelos métodos de propagação de moléculas ainda é fonte de produção de novos sistemas, métodos e ferramentas. Esses métodos, propostos teoricamente, são os componentes necessários para a construção dessa camada. Pouco se vê trabalhos que foquem em camadas de enlace e de rede. Como o conhecimento sobre os processos que regulam as moléculas em ambientes biológicos ainda é pouco, e diante do baixo desempenho desses sistemas de comunicações, a expansão dessas redes de comunicações é um dos maiores desafios da Internet das Bionano-Coisas.

2.1.3. Motivações e objetivos do capítulo

Este capítulo de livro tem o objetivo de esclarecer porque a comunidade de Redes de Computadores deve investigar mais agressivamente esta área, especificando, por meio deste documento, em que os diversos pesquisadores da área podem se concentrar. Os objetivos específicos do capítulo são:

- Introduzir as diferentes opções de redes de comunicações na escala nano, e especificar para cada uma delas os materiais, dispositivos, sistemas de comunicação e protocolos de transmissão necessários para a execução da tecnologia.
- Apresentar uma tentativa inicial de reformulação da arquitetura TCP/IP para acomodar as nanorredes. Isso terá como base os esforços que a comunidade científica têm apresentado até o momento de escrita deste capítulo.
- Identificar e analisar os diversos desafios de pesquisa e aplicações que vem sendo o foco de pesquisa nos últimos anos. Essa parte servirá como um guia de inserção para a comunidade de redes de computadores na área de nanorredes.
- Apresentar as principais ferramentas, tais como simuladores de rede, existentes e em uso para avaliar propostas de protocolos concebidas para as nanorredes.

2.2. As redes de comunicação em nanoescala

As redes de comunicação desempenham um papel essencial em uma ampla gama de aplicações em nanoescala, desde a computação em nanoescala até aplicações em nanomedicina. A conectividade e os enlaces entre os nanodispositivos (ou nanonós) suportam as nanorredes, assim como a nanocomunicação. A comunicação em nanoescala permite que um conjunto de nanomáquinas realizem tarefas coordenadas. Por exemplo, as nanomáquinas podem funcionar como portas lógicas básicas [5] e, conseqüentemente, podem realizar computação distribuída. As nanomáquinas médicas [6] com capacidades de comunicação podem realizar um monitoramento coordenado da saúde humana [7]. Esta seção apresenta as principais definições associadas às nanorredes e à comunicação em

nanoescala. Da mesma forma, são elencadas as características gerais que podem auxiliar na compreensão das demais seções deste capítulo.

2.2.1. Definição e características gerais

O termo nanotecnologia foi definido pela primeira vez da seguinte forma: “A nanotecnologia consiste principalmente do processamento, separação, consolidação e deformação de materiais por um átomo ou por uma molécula” [8]. Na década de 80 esse conceito foi expandido, acrescentando a ideia de nanodispositivos (ou nanomáquinas) e a ideia de que eles poderiam se replicar por meio do controle do computador. As nanomáquinas são a unidade funcional mais básica de uma nanorrede. Elas são pequenos componentes consistindo em um conjunto de moléculas organizadas capazes de realizar tarefas de computação, detecção e/ou atuação muito simples. As nanomáquinas podem ser utilizadas como blocos de construção para o desenvolvimento de sistemas mais complexos, como nanorrobôs e dispositivos computacionais, como nanoprocessadores, nanomemória ou nanorrelógios [9]. As nanomáquinas podem ser interligadas, dando origem às nanorredes, com o intuito de executar tarefas colaborativas de forma distribuída e superando as limitações das nanomáquinas.

A estrutura das nanorredes é apresentada de forma geral pelos elementos: nanonós, nanorroteadores, interface micro-nano e *gateway*, como apresentado na Figura 2.2. Esses elementos são importantes para o funcionamento das nanorredes e seguem as características [10]:

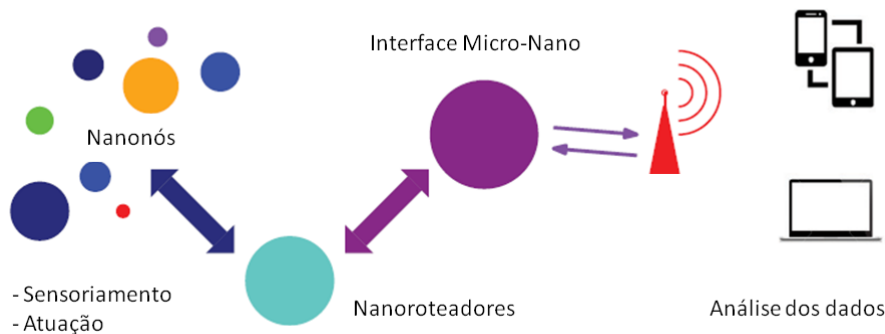


Figura 2.2: Visão geral da estrutura de uma nanorrede.

- **Nanonós:** os nanodispositivos pequenos e simples. Devido à capacidade energética, de memória e de comunicação reduzidas, eles só podem realizar tarefas de computação simples, como por exemplo o controle de execução de atividades, comunicação, sensoriamento e atuação ao meio. Os nanonós também têm a capacidade de transmissão de curtas distâncias.
- **Nanorroteadores:** são nanodispositivos com recursos computacionais um pouco maiores do que os nanonós e podem agregar informações das nanomáquinas e também podem controlar o comportamento dos nanonós enviando uma ordem extremamente simples (como ligar/desligar, dormir, ler, etc.).
- **Interface micro-nano:** Eles são usados para coletar as informações encaminhadas pelos nanorroteadores e enviar a informação aos microdispositivos. Ao mesmo

tempo, eles podem enviar a informação em microescala para nanoescala. As interfaces micro-nano são dispositivos híbridos que não só podem se comunicar em nanoescala, usando as técnicas de nanocomunicação, mas também podem usar paradigmas clássicos de comunicação em as redes de comunicação micro/macro.

- **Gateway:** permite aos usuários um controle ou monitoramento de todo o sistema remotamente pela Internet.

A comunicação entre nanomáquinas consiste na troca de dados (mensagens) em nanoescala e é a base para as interconexões com fio/sem fio em uma nanorrede. As formas populares de mensagens são texto, imagens, números, vídeos e som. A maneira pela qual as nanomáquinas se comunicam depende fortemente da maneira como elas são implementadas. Além disso, a aplicação para a qual as nanorredes são implantadas restringe a escolha do tipo de nanocomunicação. A comunicação em nanoescala tem algumas dificuldades, dentre elas podemos citar: a dimensão extremamente pequena entre as nanomáquinas, a relação entre o tamanho da antena e o comprimento de onda do sinal, a rápida degradação que limita a cobertura do sinal na faixa de Terahertz, baixo poder de processamento das nanomáquinas e de alimentação, e a degradação do sinal molecular devido aos efeitos da concentração de moléculas de água no ambiente.

Na literatura, existem três principais paradigmas de comunicação em nanoescala, podendo fornecer a interconexão entre as nanomáquinas em uma nanorrede. A Figura 2.3 ilustra a classificação desses paradigmas em comunicação eletromagnética, acústica e molecular [9, 11]. Na energia acústica, a variação de pressão é usada para transmitir informações em comunicação acústica. A comunicação eletromagnética é habilitada pela modulação de ondas eletromagnéticas para transmitir informações. Além disso, a comunicação molecular permite que as nanomáquinas se comuniquem umas com as outras usando moléculas como portadoras da comunicação.

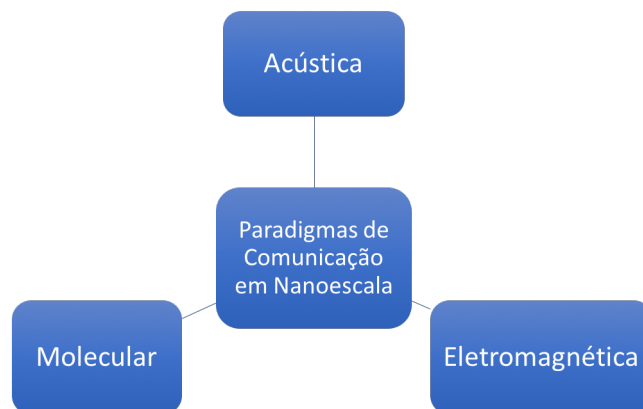


Figura 2.3: Paradigmas de comunicação em nanoescala.

De forma geral, o sistema de comunicação (em nano, micro e macro escalas) segue os passos: *i*) codificação dos dados (mensagem) no sinal, *ii*) transmissão, *iii*) propagação do sinal no meio de comunicação, *iv*) recepção do sinal, *v*) decodificação dos dados (mensagem). A Figura 2.4 ilustra esses passos sendo que os passos (*i*) e (*ii*) são realizados na

nanomáquina transmissora e os passos (iv) e (v) realizados na nanomáquina receptora. No passo (iii), a propagação do sinal ocorre no canal de comunicação. A nanomáquina transmissora é aquela que envia a mensagem de dados, enquanto a receptora é aquela que recebe a mensagem de dados. O canal de comunicação é o caminho físico em que o sinal se propaga entre a nanomáquina transmissora e a receptora. No caso das nanomáquinas, esse canal de comunicação pode ser o ar, o sangue, ou cabos, por exemplo. Para controlar essa comunicação é necessário um protocolo, ou seja, um conjunto de regras que irá auxiliar as duas nanomáquinas a trocarem mensagens. Todo dado a ser transmitido precisa ser codificado em um sinal e este pode ser analógico ou digital. Exemplos de sinais analógicos são as ondas eletromagnéticas, acústicas e moleculares, os quais são essencialmente tratados neste capítulo por serem os mais utilizados na comunicação em nanoescala até o momento. Na sequência focaremos em apresentar as definições referentes aos três tipos de comunicação e depois apresentaremos como esses tipos de comunicação são utilizados para a construção de nanorredes.

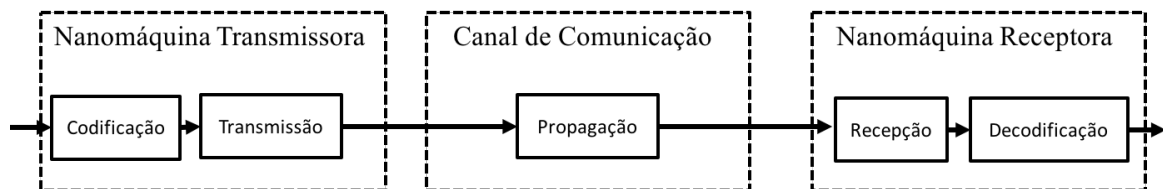


Figura 2.4: Sistema de comunicação em nanoescala baseado em uma nanomáquina transmissora, um canal de comunicação e uma nanomáquina receptora. A transmissora é responsável pela codificação e envio da mensagem para o canal. Esse, por sua vez, é responsável em propagar o sinal. Finalmente, a mensagem chega na nanomáquina receptora que a decodifica.

- **Comunicação por ondas eletromagnéticas:** é baseada na modulação de ondas eletromagnéticas para transmitir informações, entre muitos nanodispositivos de comunicação, sendo necessário o desenvolvimento de nanoantenas e o correspondente transceptor eletromagnético.
- **Comunicação acústica:** é realizada a partir de variações de pressão em um meio, na qual a mensagem é codificada usando a energia acústica dessas variações.
- **Comunicação molecular:** é baseada nos caminhos de sinalização, que são cadeias de reações químicas que processam sinais de informação modulada por características químicas, como concentração de moléculas, tipo e estado de energia, para propagá-los de uma fonte (transmissor) para um destino (receptor).

2.2.2. Nanorredes baseadas em ondas eletromagnéticas

As nanorredes baseadas em ondas eletromagnéticas estão fundadas na transmissão e recepção de radiação eletromagnética a partir de componentes compostos por novos nanomateriais [12]. A comunicação baseada em ondas eletromagnéticas na banda Terahertz é uma das técnicas promissoras para a troca de dados nessas nanorredes. Esta banda é subutilizada e pode contribuir significativamente para potencializar tecnologias médicas no

futuro por causa de sua menor susceptibilidade a efeitos de propagação, como a dispersão e sua vantagem de segurança para tecidos biológicos, isto é, não-ionizados [10].

Para a construção das nanoantenas é proposto o uso de nanotubos de carbono e nanofitas de grafeno. Uma nanoantena baseada em grafeno não é apenas uma mera redução de uma antena clássica, pois existem vários fenômenos quânticos que afetam a propagação de ondas eletromagnéticas no grafeno. Como resultado, a frequência de ressonância dessas nanoestruturas pode ter magnitude duas vezes menor que as antenas não carbonadas. No entanto, sua eficiência de radiação também pode ser prejudicada por causa desse fenômeno. Em segundo lugar, os nanotubos de carbono também são propostos como base de um nanotransceptor eletromecânico ou nanorrádio, capaz de modular e demodular uma onda eletromagnética por meio de ressonância mecânica. Esta técnica foi utilizada na recepção, mas exige fontes de energia em nanoescala muito altas para transmissão ativa.

Existem duas possíveis bandas de frequência de operação para redes de nanomáquinas: a banda de Terahertz e a parte superior da banda de megahertz. A transmissão sendo realizada em frequências mais baixas, as nanomáquinas podem se comunicar por longas distâncias. No entanto, a eficiência energética do processo para gerar as ondas EM em um nanodispositivo é baixa. Por isso, acredita-se que os nanodispositivos não se comunicarão entre si utilizando a faixa de frequência megahertz. A banda Terahertz fornece transmissão em uma grande largura de banda. Por um lado, isso pode ser usado para suportar comunicação de alta velocidade entre nanodispositivos. Por outro lado, uma largura de banda muito grande permite novas técnicas de acesso ao canal, o que pode facilitar as tarefas do protocolo MAC (*Medium Access Control*). Por exemplo, ao usar pulsos de *femtosegundos* para comunicação entre nanodispositivos, as chances de haver uma colisão entre transmissões de nanomáquinas diferentes são quase inexistentes [13]. Como resultado, protocolos MAC simples podem ser usados. Por exemplo, as nanomáquinas podem transmitir apenas quando tiverem alguma informação pronta e, em seguida, aguardar por uma confirmação. Novas formas de verificação da integridade da mensagem e informar adequadamente o transmissor são necessárias.

Uma possibilidade é usar a banda Terahertz para controlar remotamente os nanodispositivos. Essa ideia decorre do fato de que as ondas EM na faixa megahertz podem ser facilmente geradas usando transmissores clássicos no macro-domínio. Ondas de energia mais altas podem ser usadas para controlar milhares ou mesmo milhões de dispositivos de nanosensores implantados em áreas muito amplas. Nesse caso, será necessário desenvolver modelos de canais precisos, bem como protocolos de comunicação para interfaces macro-nano sem fio. Assim, acredita-se que os nanodispositivos se comunicarão potencialmente entre eles na banda Terahertz (0,1-10 THz). Entretanto, o primeiro desafio de pesquisa para a comunicação do nanodispositivo é desenvolver novos modelos de canais para a banda Terahertz. O ruído do canal de Terahertz é formado principalmente pelo ruído molecular. A absorção de moléculas presentes no meio atenua o sinal transmitido e introduz ruído [12]. Além disso, o ruído molecular não é gaussiano, nem branco. Devido a cada tipo de molécula ter diferente frequência de ressonância, a densidade espectral de potência do ruído não é plana. Além disso, este tipo de ruído só aparecerá na transmissão, ou seja, não haverá ruído a menos que o canal esteja sendo usado.

2.2.3. Nanorredes baseadas em ondas acústicas

Na comunicação acústica, a mensagem transmitida é codificada usando energia acústica, isto é, variações de pressão em um meio fluido ou sólido. A propagação acústica introduz pequenas variações de pressão no meio fluido ou sólido, que satisfazem a equação de onda. O comportamento dos nanorrobôs é relevante por suas propriedades físicas, meio ao seu redor e a frequência de trabalho [10, 14]. Tomando como referência o sistema genérico de comunicação em nanoescala ilustrado na Figura 2.4, definem-se as partes das nanorredes baseadas em ondas acústicas como:

1. *Nanomáquina transmissora*: A codificação e modulação de ondas acústicas dependem do meio que compõe o canal de comunicações. Entretanto não existe, ainda, um interesse ativo no desenvolvimento de novas técnicas porque elas podem ser diretamente utilizadas das redes baseadas em ondas eletromagnéticas. Entretanto, devido à diferença de comportamento e de desempenho, novas técnicas precisam ser desenvolvidas. Os grandes problemas, então, se encontram na miniaturização desses transmissores.
2. *Canal de comunicação acústico*: As ondas acústicas são caracterizadas pela propagação em diferentes tipos de ambientes com um melhor desempenho, principalmente quando comparado com ondas eletromagnéticas. O atrativo do uso desse meio de comunicação, principalmente dentro da Internet das Bionano-Coisas é pela alta eficiência dessas ondas ao penetrarem tecidos e organismos biológicos. Mas isso não é só limitado a esses meios. Além disso, as ondas acústicas também servem de transmissão de informação em ambientes aéreos, aquáticos e até dentro do solo.
3. *Nanomáquina receptora*: A recepção de sinais de ultrassom são baseados na conversão de ondas acústicas para ondas elétricas. Na escala nano o sistema mais comumente usado são baseados em nanocabos. Um conjunto desses nanocabos entram em movimento mediante a uma determinada potência da onda acústica. Esses movimentos podem ser convertidos em energia elétrica mediante a uma conversão mecânica-eletr por circuitos retificadores. Outra técnica que pode ser usada é o sensor de pressão, que é um transdutor que transforma a pressão medida em energia elétrica detectável ou sinais ópticos. Estes sensores são fabricados através das mais diversas tecnologias como o piezoelétrico, cristais de quartzo, fotocondutores e diafragma de metal. Atualmente os nanodispositivos de pressão são fabricados em massa utilizando a tecnologia de *wafes* de silício.

A comunicação acústica tem um outro benefício que é a simplicidade dos componentes necessários para a construção dos dispositivos e o fato de que muitos instrumentos de diagnóstico já estão baseados em microondas, que são consideradas não perigosas para os seres humanos [14]. Os tecidos do corpo variam em propriedades acústicas, incluindo velocidade de som, densidade e atenuação. Os limites entre os diferentes tipos de tecidos podem espalhar ondas acústicas. Para as distâncias relativamente curtas, considera-se ($100\mu\text{m}$, cerca de 5 a 10 diâmetros celulares) um único tipo de tecido com características homogêneas. Assim, é razoável ignorar as reflexões dos limites dos tecidos nessa escala

porque a amplitude da reflexão depende da diferença na impedância acústica em ambos os lados do limite, que é muito pequena [15].

O aquecimento é um dos efeitos acústicos nos tecidos biológicos. Durante a propagação de ultrassom, uma parte da energia é absorvida e convertida em calor, o que pode levar a um aumento de temperatura. O tempo de exposição desempenha um papel importante no efeito do aquecimento, uma vez que a geração de calor devido à absorção de ultrassom é maior do que a dissipação de calor pela circulação sanguínea no tecido. A medida que a intensidade da onda aumenta, a temperatura aumenta e, se ficar acima de 38,5°C, pode ocorrer efeitos biológicos adversos. Como o aquecimento é causado pela intensidade da onda, as transmissões impulsivas com um ciclo de trabalho baixo podem reduzir esse efeito por um fator proporcional à duração do ciclo de trabalho [15]. As variações de pressão da onda de ultrassom causam um outro efeito, que são bolhas no meio de propagação para contrair e expandir. Durante este processo, a pressão interna e a temperatura na bolha podem atingir valores elevados, causando sérios efeitos biológicos e danificando os tecidos próximos. Esse efeito é um fenômeno dependente de frequência. Para frequências altas há oscilações de pressão mais curta e o tempo de expansão da bolha é reduzido, tendendo a desaparecer [15].

Sistemas baseados em ondas ultrassônicas também são atrativos para as nanorredes quando a questão é recepção e detecção de sinais. Mesmos os transmissores sendo maior e menos confiáveis, esses sistemas ainda possuem um grande potencial principalmente para redes de *broadcast*, onde um transmissor envia uma mesma mensagem para vários outros receptores. Isso possibilita uma implementação e interação de redes micro para nano, como foi proposto em aplicações em neurociências que são abordados na Seção 2.5 neste capítulo.

2.2.4. Nanorredes baseadas em comunicação molecular

Na natureza, a troca de informações entre células baseia-se na síntese, transformação, emissão, propagação e recepção de moléculas por meio de processos bioquímicos e físicos. Esta troca de informações foi recentemente classificada como comunicação molecular [16]. Tomando como referência o modelo genérico de um sistema de comunicação em nanoescala ilustrado na Figura 2.4, o sistema de comunicação molecular é organizado em três blocos principais: nanomáquina transmissora, canal molecular e nanomáquina receptora. Esses blocos são definidos como segue:

1. *Nanomáquina transmissora*: a depender das propriedades do canal molecular, as moléculas devem ser codificadas de maneira diferente. A abordagem comumente utilizada é a técnica de modulação por concentração, na qual a informação é codificada como a concentração de uma determinada molécula [17, 18].
2. *Canal de comunicação molecular*: o processo de propagação pode ser passivo ou ativo. No processo passivo as moléculas se propagam sem que haja gasto energético envolvido no transporte entre as nanomáquinas transmissora e receptora (por exemplo, difusão facilitada, microfluidos, canais iônicos, canais mecânico-sensitivos). Quando um terceiro agente, chamado propagador, é o responsável pelo transporte das moléculas de informação entre a nanomáquina transmissora e a receptora, o

processo é chamado de ativo (por exemplo, bactéria e nanomotores) [17, 18].

3. *Nanomáquina receptora*: a entidade responsável pela captura das moléculas de informação e sua posterior decodificação é chamada receptora ou destinatária. A abordagem mais comum de recepção se baseia nas reações químicas produzidas pela adesão das moléculas à superfície da nanomáquina receptora [17, 18].

As redes nanomoleculares são diretamente inspiradas por redes de comunicação entre entidades presentes na natureza. Exemplos podem ser apresentados, diferindo na forma como as moléculas se propagam através da difusão de um nó de rede (células nos exemplos) para outro [19].

2.2.4.1. Comunicações moleculares baseadas em difusão

O modelo de canal de comunicação mais estudado em comunicações moleculares é aquele baseado em difusão. Nesse modelo, as moléculas se propagam livremente pelo espaço entre as nanomáquinas transmissora e receptora executando movimento Browniano (Figura 2.5). Desta maneira, os pesquisadores se restringem a estudar outros tópicos, incluindo o ruído, a modulação e a detecção dos sinais difundidos no espaço [17, 18].

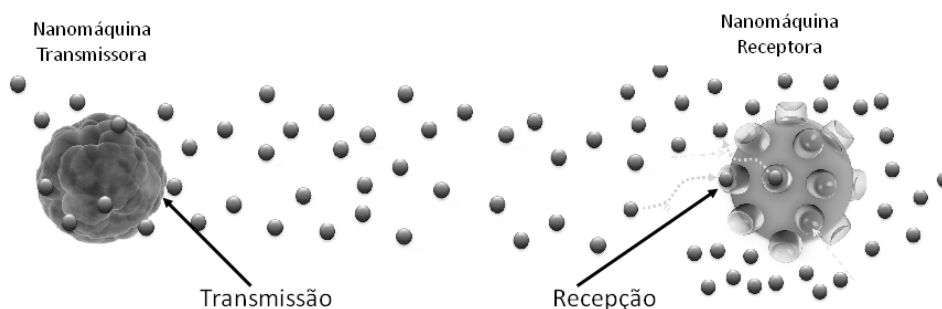


Figura 2.5: Comunicações moleculares baseadas em difusão. A informação é codificada na concentração de determinada molécula.

2.2.4.2. Comunicações moleculares baseadas em sinalização por Ca^{2+}

Uma célula sensorial libera ou absorve moléculas de Ca^{2+} em resposta a vários estímulos que abrem ou fecham canais particulares na membrana celular. A informação molecular na variação da concentração de íons de Ca^{2+} é propagada dentro e fora da célula, causando uma variação na carga elétrica da membrana celular e a transdução da informação para um sinal elétrico. A propagação de Ca^{2+} tem dois estágios de sinalização: intracelular e intercelular. A informação para esse tipo de comunicação é codificada com o uso de picos ou oscilações da concentração de Ca^{2+} na célula. Devido ao estímulo externo, sua propagação permite a comutação e a amplificação do sinal. A propagação de Ca^{2+} é baseada em canais iônicos intercelulares. Esse processo depende de abertura de junções comunicantes, que são portas que interconectam o citosol de uma célula com o da outra. Quando são abertas as junções, os íons se propagam entre as células.

2.2.4.3. Comunicações moleculares baseadas em neurônios

A interação natural entre neurônios também é estudada como forma de habilitar a comunicação entre as nanomáquinas [20]. A Figura 2.6 ilustra um sistema ponto-a-ponto de comunicação molecular baseada em neurônios. A informação é codificada em ação potencial, que é uma relação eletroquímica entre a concentração de Ca^{2+} e a tensão da membrana celular. As comunicações de longa distância dentro do corpo humano são possíveis devido ao uso do sistema nervoso como rede centralizadora (*backbone*). Pesquisas estão sendo realizadas para construir modelos para esse sistema de comunicação, mas também para tornar mais eficiente a transmissão neuronal. Em geral, os autores estão observando a comunicação não apenas dos neurônios, mas também dos astrócitos e entre neurônios e astrócitos [21]. Os astrócitos fornecem nutrientes que mantêm o bom funcionamento dos neurônios. Acredita-se que os astrócitos tenham uma importante função na comunicação entre os neurônios.

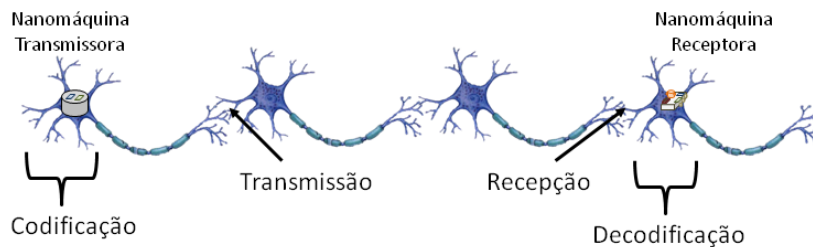


Figura 2.6: Comunicações moleculares baseadas em difusão. A informação é codificada na concentração de determinada molécula.

2.2.4.4. Comunicações moleculares baseadas em bactérias

Nesse sistema de comunicação, a informação é codificada no plasmídeo da bactéria e então se propaga entre as nanomáquinas por meio do seu movimento em um fluido, como pode ser observado na Figura 2.7. Esse sistema ativo é considerado o mais fácil de ser realizado. Atualmente, os pesquisadores estão concentrados em encontrar as melhores formas de estabelecer comunicações desse tipo e em determinar quais aplicações podem ser mais importantes [22, 23].

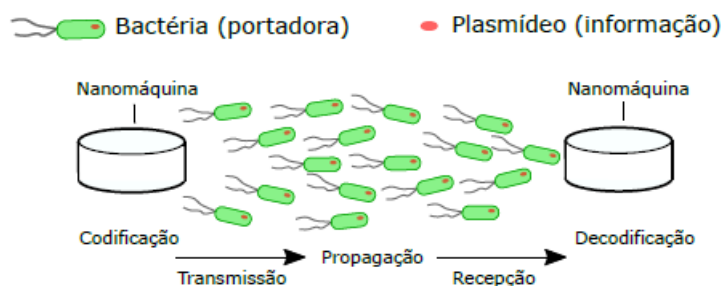


Figura 2.7: Representação de um sistema de comunicação baseado em bactérias. A informação é armazenada no plasmídeo para que seja transportada entre as nanomáquinas [3].

2.2.4.5. Comunicações moleculares baseadas em vírus

As comunicações moleculares baseadas em vírus são uma proposta para transmitir e realizar a transdução de informação genética entre nanomáquinas. Essa técnica é muito importante para o estudo de doenças, bem como tratamentos utilizando engenharia genética, como por exemplo, a transferência genética e a terapia genética. Esse sistema é apresentado na Figura 2.8 [24].

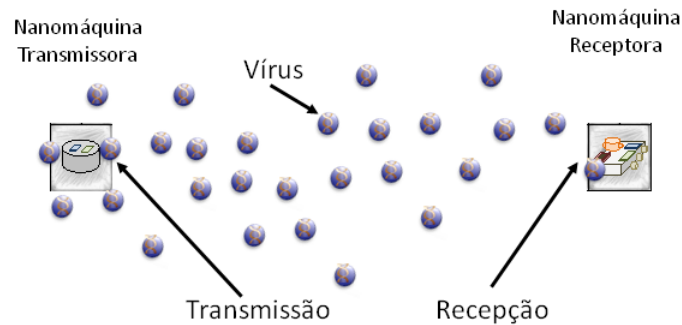


Figura 2.8: Representação de um sistema de comunicação baseado em vírus.

2.3. Arquiteturas para nanorredes

Esta seção apresenta uma visão geral sobre como os trabalhos da literatura vêm tratando a definição de uma arquitetura de redes de computadores a fim de acomodar as nanorredes. Descrevemos os recentes esforços da comunidade científica para solucionar este problema seguindo três perspectivas identificadas a partir de nossas investigações e organizadas como segue: *i)* modelo de camada única, *ii)* modelo em camadas *clean-slate* e *iii)* modelo em camadas com base na arquitetura TCP/IP (Figura 2.9). Antes de detalharmos os esforços em direção à proposição de uma arquitetura para as nanorredes, fazemos um breve histórico da evolução das redes de computadores de forma geral e a consolidação da arquitetura TCP/IP.

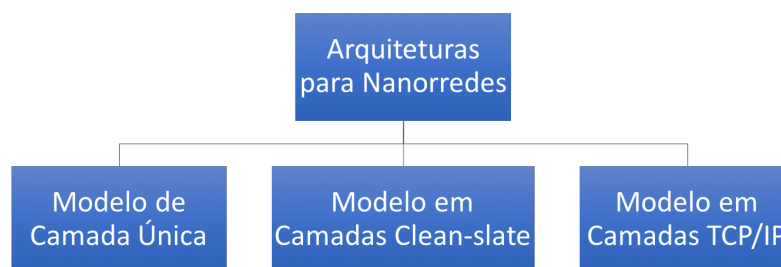


Figura 2.9: Perspectivas seguidas na proposição de uma arquitetura para nanorredes.

Os primeiros computadores foram construídos durante a segunda guerra mundial e eles eram caros e isolados. Contudo, com a redução gradativa dos preços e a popularização dos computadores pessoais, os primeiros experimentos começaram a conectar computadores. No início da década de 1960, diversos pesquisadores incluindo Paul Baran, Donald Davies ou Joseph Licklider independentemente publicaram os primeiros ar-

tigos descrevendo a idéia de construir redes de computadores [25, 26]. Dado o custo dos computadores, compartilhar seus recursos como processamento, armazenamento e outros dispositivos de entrada e saída a uma longa distância foi uma idéia atrativa às empresas e usuários em geral. Nos EUA, o ARPANET começou em 1969 e continuou até meados da década de 1980 [27]. Muitas outras redes de pesquisa foram construídas durante a década de 1970 [28]. Ao mesmo tempo, as indústrias de telecomunicações e informática se interessaram pelas redes de computadores. A indústria de telecomunicação apostou no padrão X25. A indústria de computadores tomou uma abordagem completamente diferente ao projetar as redes locais (*Local Area Networks* – LANs). Muitas tecnologias LAN, como Ethernet ou Token Ring, foram projetadas. Durante a década de 1980, a necessidade de interconectar mais e mais computadores levou a maioria dos vendedores de computadores a desenvolver seus próprios conjuntos de protocolos de rede. A Xerox desenvolveu a Arquitetura de Sistema de Rede (*Xerox Network System Architecture*) [29], a DEC escolheu DECNet [30], a IBM desenvolveu SNA [31], a Microsoft introduziu NetBIOS, a Apple apostou no Appletalk.

Dentro da comunidade de pesquisa, a ARPANET foi desativada e substituída pela arquitetura TCP/IP [27] e a implementação de referência foi desenvolvida dentro do BSD Unix [32]. As Universidades que já estavam executando o Unix poderiam assim adotar a arquitetura TCP/IP facilmente e fornecedores de estações de trabalho Unix, como Sun ou Silicon Graphics incluíam o TCP/IP na sua variante do Unix. Paralelamente, o ISO, com o apoio dos governos, trabalhou no desenvolvimento de um modelo de referência. No final, o modelo TCP/IP tornou-se o padrão de fato que não é apenas usado dentro da comunidade de pesquisa. Durante a década de 1990 e no início dos anos 2000, o uso do TCP/IP continuou. Hoje, os protocolos proprietários raramente são usados.

Com a evolução da microeletrônica, a miniaturização dos dispositivos e os avanços na comunicação sem fio, percebemos nas últimas décadas uma enorme busca pela ubiquidade das redes. A Internet se expandiu exponencialmente suportada pela arquitetura TCP/IP. Em paralelo, o advento de redes sensores e da Internet das Coisas vêm revolucionando as formas como desempenhamos atividades corriqueiras. Os serviços de acesso à Internet estão disponíveis em quase qualquer lugar e a qualquer hora. A mobilidade e a portabilidade dos dispositivos são uma realidade. Para se chegar a esta evolução, um grande esforço científico foi realizado para definir a melhor forma de conectar esses dispositivos portáteis à Internet. A própria construção e desenvolvimento das redes de sensores sem fio e da Internet das Coisas ofereceram aprendizados, os quais passaram por questionamentos sobre a arquitetura de rede a seguir. Mais uma vez se observou a predominância e a continuidade do modelo TCP/IP. No momento, os pesquisadores discutem novamente qual arquitetura de rede a seguir na nova era que se abre com as nanorredes e a Internet das bio-nano-coisas.

Mais uma vez, este capítulo de livro está sendo escrito em um momento de possível reconstrução ou reformulação da arquitetura de redes a fim de integrar os dispositivos em nanoescala com as redes clássicas e, em última instância, com a Internet. A interconexão de dispositivos em escala nano impulsiona a proposição de um novo paradigma de redes de computadores. Devido às inúmeras propriedades provenientes da nanoescala e as limitações esperadas para as nanomáquinas, a maior parte do conhecimento de redes de comunicação existente não é adequado para as nanorredes. A comunicação entre nanomá-

quinas exige soluções inovadoras que estejam de acordo tanto com as características dos componentes da rede quanto a natureza dos processos de comunicação molecular [33].

Em redes de computadores para que seja possível a comunicação entre dois dispositivos computacionais (nós), diferentes funções devem ser realizadas em cada componente desse nó. Essas funcionalidades são normalmente definidas pela arquitetura de redes e posteriormente implementadas. O conjunto de protocolos possibilita a interconexão entre os diferentes tipos de computadores de diversos fabricantes, equipados com arquiteturas de hardware distintas fazendo uso de diferentes tecnologias de acesso [34]. Contudo, o modelo tradicional, originalmente projetado considerando o alto processamento de rede e de seus nós, não é considerado viável para a implementação em nanorredes. Diferentes desafios para o projeto de protocolos para nanorredes ainda estão sendo investigados, porém sem soluções totalmente definidas ou desenvolvidas até o momento [35]. Apesar da implementação completa da arquitetura TCP/IP na comunicação molecular ser difícil ou inviável, algumas de suas funções podem ser reproduzidas para este ambiente tais como: o controle de fluxo e a conexão orientada a capacidade, confiabilidade e congestionamento [36].

Na literatura atual, várias pesquisas propõem protocolos para nanorredes. Algumas propostas sugerem um modelo de camada única, outras propõem um novo modelo em camadas projetadas especificamente para as nanorredes, e ainda outras adaptam o modelo TCP/IP para as nanorredes na intenção de já conectar as nanorredes à Internet. Estas três perspectivas já ilustradas na Figura 2.9 são exploradas nas próximas subseções. Nelas, descrevemos as abordagens seguidas nas definições das novas arquiteturas de redes e apresentamos exemplos de protocolos.

2.3.1. Modelo de camada única

Seguir uma pilha de protocolos baseada em camada assume que as nanorredes mantêm uma arquitetura hierárquica de várias camadas dinâmicas podendo compreender nanomáquinas, nanorroteador e gateway [35]. Entretanto, as limitações esperadas para as nanorredes motivam o desenvolvimento do paradigma de camada única, sendo necessário combinar as funcionalidades indispensáveis em uma única camada. Contudo, deve-se avaliar o verdadeiro impacto de cada nó em uma nanorede de acordo com o ambiente e a sua aplicação incluindo apenas o essencial.

Para Akyildiz e Jornet [37] não se deve considerar que cada nanonó será capaz de transmitir diretamente ao seu nanorroteador mais próximo. Na abordagem apresentada pelos autores, nanomáquinas idênticas que se comportam da mesma maneira em termos de comunicação, propósito e área de atuação (dentro do corpo humano) são agrupadas e gerenciadas por um cabeçalho que classifica os atributos para cada agrupamento. Se o cenário é um sistema de transmissão baseado em pulsos, as nanomáquinas podem ter uma noção da distância entre elas, o que permitiria um nanorroteador atribuir o mesmo endereço (identificador) para as diferentes nanomáquinas que estão à mesma distância. Com as características propostas por esse modelo, as funcionalidades de roteamento podem ser acopladas com as funcionalidades de endereçamento e aos serviços de descoberta e associação de rede, o que consequentemente poderia atribuir para o projeto de um modelo de camada única.

Na pilha de protocolos TCP/IP, cada camada é responsável por um conjunto de tarefas que fornecem serviços para o protocolo da camada superior. Em um modelo de camada única, as funções das camadas de enlace, rede, transporte e aplicação podem ser agrupadas na camada física ao combinar técnicas de inundação de sinal (*flooding*) com métodos de endereçamento. A inundação de sinal é um algoritmo estático que visa à divulgação de dados em toda a rede por retransmissão cega, ou seja, cada pacote de entrada é enviado para todas as linhas de saída, exceto para aquela que chegou [38]. A utilização deste recurso em nanorredes poderia suprimir a necessidade dos requisitos utilizados para o endereçamento, identificação, encaminhamento e roteamento [33].

A técnica de disseminação de dados por inundação foi utilizada por Liaskos e Tsioliaridou [39] para propor um modelo de rede de camada única. A estrutura do modelo combina as funcionalidades das camadas de enlace, rede e aplicação em uma única camada de baixa complexidade. O esquema descarta os requisitos de endereçamento, entrega e roteamento ao utilizar álgebra de rede e inundação de pacotes. O modelo é baseado na arquitetura de rede em grade quadrada na qual, os nanodispositivos ficam densamente distribuídos nos vértices da grade. Os nós que fazem parte dessa rede são classificados em duas categorias conforme sua capacidade de recepção: nó usuário (leitura) ou nó de infraestrutura (retransmissor). O processo de inundação é desencadeado pela fonte e adota as seguintes etapas:

1. A propagação de pacotes – tem como base o protocolo de enlace CSMA/CA (Acesso Múltiplo com Verificação de Portadora e Prevenção de Colisão, do inglês, *Carrier Sense Multiple Access with Collision Avoidance*). Este é um protocolo MAC simples, amplamente utilizado no padrão IEEE 802.11. Devido às limitações do ambiente o protocolo RTS/CTS, uma extensão do CSMA/CA que permite a uma estação anunciar a sua intenção de enviar dados, não foi utilizado. Em vez disso, cada nó é equipado com uma fila capaz de acomodar 10 pacotes diferentes, enquanto o intervalo de retrocesso é escolhido uniformemente em (0,10) nanosegundos.
2. A emissão de pacotes – uma vez que um nanonó obtém uma medição útil de seu módulo sensorial, ele se torna uma fonte e emite pulsos (“pacotes”) periodicamente. Os nós de recepção adotam um esquema de inundação, inicialmente retransmitindo o pacote incondicionalmente. Durante o pulso cada nó processa estatisticamente os níveis de sinal para interferência (SINR) e posteriormente amadurece para se tornar um nó de “infraestrutura” ou de “usuário”. A simetria pode ser usada para dirigir as transmissões de pacotes para direções específicas, alcançando os mesmos efeitos que a forma de feixe ou roteamento direcional. Para escalabilidade e ganhos em eficiência energética, apenas os nós de infraestrutura podem encaminhar os pacotes recebidos. Entretanto, a classificação é dinâmica e adaptável. Segundo os autores dessa técnica, a arquitetura proposta é facilmente implementável com tecnologias existentes como as empregadas na fabricação de CPU e opera de forma eficiente independentemente do número total de nós.

2.3.2. Nova arquitetura em camadas

Piro *et al.* [40] consideram que a arquitetura de rede TCP/IP não pode ser diretamente adotada em comunicações nano. Nesta perspectiva, os autores pensaram em um protótipo

de pilha de protocolos para ser utilizado como um ponto de partida para soluções futuras. A pilha de protocolo proposta consiste em quatro camadas: *unidade de processamento de mensagem*, *camada de rede*, *controle de acesso ao meio* e *camada física*. A camada de unidade de processamento de mensagem é responsável por fornecer e processar mensagens. A camada de rede é encarregada de manipular as operações de roteamento. O controle de acesso ao meio trata-se de um protocolo MAC transparente e simplificado que possui a única função de gerenciar o fluxo de pacotes. Este protocolo transfere os dados da camada de rede para a camada física. Como não há cabeçalhos adicionais nesta camada, este protocolo não lida com o reconhecimento e a retransmissão de pacotes. A camada física segue a técnica de transmissão do sinal no meio físico utilizando como base pulsos curtos. A técnica de modulação por tempo de transmissão *TS-OOK* é utilizada, sendo o pulso curto codificado como 1 e ausência de pulso como 0 [41].

Para permitir a comunicação entre nanodispositivos biológicos, Walsh *et al.* [42] propuseram uma comunicação baseada em plataforma celular para criar protocolos necessários para redes de comunicação molecular. A solução proposta é híbrida, inclui DNA e computação baseada em enzimas, onde cada uma contribui para funções de protocolo específicas. A abordagem utiliza alguns protocolos convencionais de redes de comunicação adaptados para comunicação molecular. A interconexão de componentes é baseada em protocolos soltos, onde cada componente realiza uma técnica específica de computação. Embora, o componente de cada camada é mapeado a partir de protocolos convencionais utilizados nas redes de comunicação, as camadas da pilha de protocolos proposta foram reorganizadas de acordo com uma série de características encontradas na comunicação molecular. A Figura 2.10 ilustra os componentes da pilha de protocolos e os protocolos para diferentes operações no nanodispositivo (por exemplo, nó de transmissão, nó de recepção, nó de roteamento intermediário).

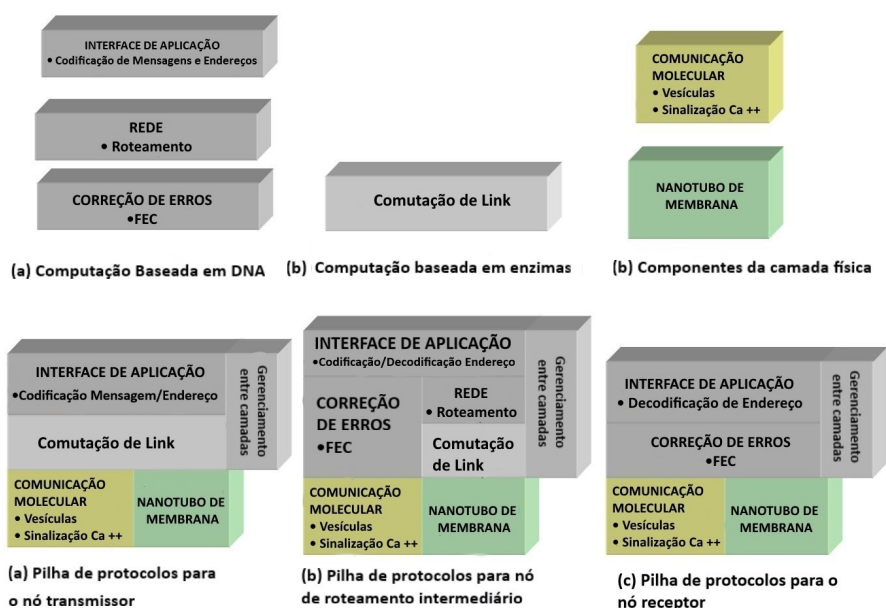


Figura 2.10: Pilha de protocolos da comunicação molecular.

A abordagem é baseada na interconexão de componentes de protocolo soltos, onde em cada componente realiza uma técnica específica da computação molecular. Para evitar o aumento desnecessário da complexidade computacional, os autores decidiram não incorporar todos os componentes em uma única pilha de protocolos genérica. Embora os componentes de cada camada sejam mapeados a partir dos protocolos convencionais, cada camada do modelo é reorganizada para se adequar às características encontradas na comunicação molecular.

Na solução proposta, a computação baseada em DNA – *DNA Based Computation* é utilizada para as camadas de interface de aplicação (*Application Interface*), rede (*Network*) e correção de erro (*Error Correction*), enquanto a computação baseada em enzimas é usada para a camada de troca de link (*Link Switching*). O modelo apresenta o seguinte fluxo de operação entre as camadas:

1. A interface da camada de aplicação executa a codificação da mensagem para as biomoléculas de informação (Figura 2.10(a)).
2. A biomolécula codificada recebe o endereço específico do destino pretendido a partir de uma tabela de endereços.
3. Na camada de aplicação, as células podem interagir com um dispositivo físico ou células artificiais com funcionalidades embutidas. Uma vez que o processo de codificação é executado, a informação da biomolécula está pronta para transmissão e se submete à camada *Link Switching*, que seleciona a junção de espaço correta para transmissão.
4. No roteamento intermediário, ilustrado na Figura 2.10(b), quando a biomolécula é recebida pela célula, a correção de erro é realizada pela primeira vez na biomolécula. Em seguida, ocorre a decodificação do endereço e a codificação baseado na tabela de roteamento para o próximo nó. Uma vez que isso é executado, a operação de comutação de ligação segue e transmite a biomolécula para o enlace subjacente.
5. A biomolécula de informação é recebida no dispositivo receptor (*receiving node*) ilustrado na Figura 2.10(c). Ela passa novamente através da camada de correção de erro, logo após ocorre a decodificação da mensagem na camada de aplicação.

2.3.3. Modelo em camadas com base na arquitetura TCP/IP

Publicações recentes exploram os problemas e as vantagens em seguir o modelo de camadas TCP/IP para projetar as nanorredes. Alguns desses estudos são apresentados a seguir. Os conceitos teóricos para lidar com a comunicação entre nanosensores sobre camadas com características distintas, como a biocompatibilidade e escalabilidade, foram apresentados por [43]. Na documentação da pesquisa são abordados os processos básicos de comunicação tais como a codificação, envio de mensagens, propagação, recepção e decodificação para desenvolver um modelo de arquitetura inicial para nanorredes intracorporais baseadas em um sistema de comunicação molecular que utiliza nanomotores.

A comunicação molecular permite as nanomáquinas biológicas se comunicarem através da troca moléculas em um ambiente aquoso e realizar trabalhos colaborativos

possibilitando um grupo de nanomáquinas executar tarefas que não podem ser realizadas por bionanomáquinas individuais. Dada a importância da comunicação molecular entre bionanomáquinas, o tema foi discutido em [44]. Os autores apresentaram uma visão completa de uma arquitetura de rede em camadas para comunicação molecular. O trabalho apresenta um modelo formal baseado na arquitetura de redes tradicionais como o modelo OSI (*Open System Interconnection*) e o modelo de TCP/IP. O estudo contribui para questões abertas que precisam ser abordadas em cada camada. Os autores utilizam como exemplo uma aplicação de entrega de medicamentos para explicar como cada camada se comporta nesta arquitetura a fim de acelerar a pesquisa em cada camada e facilitar o projeto e o desenvolvimento de aplicações para comunicação molecular.

Durante a investigação para entender exatamente como ocorre a transmissão em uma instância específica das nanopartículas foram constatadas semelhanças entre o sistema de comunicação de nanopartículas com o sistema elétrico. A partir desta percepção, Sharp *et al.* [45] propuseram a utilização de uma pilha de protocolos semelhante ao TCP/IP para uma nanorede de entrega de medicamentos. O modelo apresenta um conjunto de protocolos para o transmissor e o receptor que possuem as mesmas camadas (Aplicação, Transporte, Rede e Física) e as seguintes funções:

- A **camada de aplicação** possui a função de iniciar uma transmissão, executar uma mensagem recebida e fazer o papel de interface entre o remetente e o receptor;
- A **camada de transporte** é responsável pelo controle de erro e fluxo;
- A **camada de rede** é utilizada para o endereçamento e a entrega de pacotes;
- A **camada física** realiza a transmissão de um pacote para o seu destino através da corrente sanguínea.

Segundo os autores, a arquitetura de redes desenvolvida é robusta o suficiente para que possa ser usada para compreender e modelar várias outras ferramentas baseadas em nanopartículas. No entanto, devido ao baixo armazenamento e ao processamento dos nanodispositivos, o trabalho proposto pode não ser viável, pois tenta implementar toda a pilha de protocolos do modelo TCP/IP [35] e não segue os mecanismos básicos dos processos de comunicação na nanotecnologia, como DNAs, enzimas e implantações dos vários mecanismos de comunicação para o curto, médio e longo alcances [46].

2.3.3.1. Camada física

No modelo TCP/IP tradicional, a camada física é responsável por garantir a integração do hardware com as demais camadas fornecendo rotinas de acesso ao meio físico. Nesta camada, ocorrem as especificações das interfaces para o meio físico, no qual o sinal será transmitido (par trançado, fibra óptica ou sem fio), a definição do tipo de sinalização (voltagem do sinal, duração/frequência de cada bit) e a codificação para garantir a confiabilidade na comunicação (detecção e correção de erros e controle da frequência de envio). A maioria dos esforços na área da comunicação molecular são focados nos vários meios de comunicação da camada física devido ao fato de que este campo de pesquisa ser

extremamente novo e as formas de comunicação ainda não estarem consolidadas. Os pesquisadores têm explorado a capacidade de informação e as características físicas como, por exemplo, atraso, atenuação do sinal, amplificação e requisitos energéticos das comunicações moleculares [36].

Conforme apresentado na Seção 2.2, a comunicação entre nanomáquinas pode ser realizada principalmente por meio de ondas eletromagnéticas, acústicas e comunicação química molecular. A propagação de sinal em vários meios e ambientes é um problema tratado na camada física. Durante o processo de transmissão de informação de um ponto a outro ocorre a codificação dos símbolos para a propagação em um meio físico. Em redes de comunicação tradicionais, a informação é codificada em sinais eletromagnéticos, acústicos ou ópticos, enquanto que em nanorredes, a mensagem é codificada utilizando moléculas [47]. O processo de codificação ocorre quando uma nanomáquina emissora detecta alguma informação (por exemplo, uma reação biológica, dentro ou fora da nanomáquina) e traduz este dado em moléculas de informação de forma que receptor seja capaz de detectar ou capturar [48].

Mensagem

Nas redes de computadores tradicionais, a mensagem é representada utilizando o sistema binário e os sinais, onde a mensagem é transportada, agregam conjuntos de bits representando texto, voz e vídeo. Na comunicação molecular, a mensagem é uma molécula que deve possuir características a serem reconhecidas pelo receptor, uma vez que a mensagem é uma molécula, a informação transmitida está mais relacionada a fenômenos, processos e estados químicos, tais como uma estrutura externa predefinida e seu status (ativo/inativo) [49, 6].

Codificação

As seguintes técnicas de codificação podem ser consideradas para representar a informação em nanorredes: Uma técnica similar à utilizada pelo sistema nervoso central para propagar os impulsos neurais, isto é, codificar a informação através de sequências temporais que alteram o nível de concentração de moléculas específicas no meio (número de moléculas por volume) [47]. Utilizar parâmetros relativos de posicionamento dos elementos moleculares ou polarização que é semelhante ao uso de pacotes criptografados nas redes de comunicação, em que apenas o receptor pretendido é capaz de ler as informações. A codificação da informação também pode ocorrer em um subcomponente da molécula como por exemplo em uma subsequência de uma sequência de DNA ou até mesmo codificadas no ambiente pelo emissor. Neste caso, são emitidas moléculas que modificam o ambiente e o receptor detecta essas alterações [48].

Receptor/transmissor

As nanomáquinas poderão desempenhar o papel de transmissor ou receptor. Nanomáquinas transmissoras devem ser capazes de obter, a partir de uma fonte externa, moléculas

cruas e armazená-las para um uso posterior, como mensagens. O receptor deve ser capaz de extrair a mensagem do meio e lidar com a informação. Em nível molecular, isto pode significar reencaminhar a mensagem, armazená-la ou para reagir a ela [47].

Em [50], os autores definiram uma interface molecular de comunicação que usa vesículas, com proteínas de junção, para transportar moléculas codificadas por mensagens. As vesículas que incorporam as moléculas de informação (como metabólitos ou nucleotídeos pequenos) serão então utilizadas como portadoras de sinal entre os nanodispositivos remetente e receptor. [51] propuseram que nanodispositivos podem se comunicar codificando informações através da frequência e amplitude das ondas de cálcio intercelulares (sinalização de cálcio que ocorre entre as células). Para garantir a detecção adequada de símbolos a sintonização entre o transmissor e o receptor é necessária. Entretanto em nanorredes não é possível simplesmente reutilizar as soluções existentes para os esquemas convencionais de comunicação de alta frequência, como os sistemas *Impulse Radio Ultra-wideband* (IRUWB), *Milimeter Wave (mm-wave)*, *Free Space Optical* (FSO) ou sistemas de alta velocidade analógica para digital (ADCs). Para superar as limitações, em [33], os autores afirmam que novos algoritmos de sincronização de tempos e frequências são necessários. Para uma sincronização de baixa taxa de amostragem, [52] desenvolveram uma nova arquitetura para a recepção baseada em média móvel contínua para a comunicação por pulso (*Continuous-time Moving Average - CTMA*). O método pode ser utilizado para futuras operações de comunicação sem fio em nanodispositivos de Grafeno utilizando a faixa de frequência em Terahertz (0,1-10 THz). Segundo os autores, o novo esquema de detecção de símbolos supera os detectores existentes com base em pulso de rádio quando usados na faixa de banda Terahertz reduzindo a taxa de erro do símbolo e aumentando a distância de transmissão alcançável.

Controle de erro

Os esquemas de comunicação molecular estão sujeitos a processos aleatórios. Como resultado, a recepção de uma mensagem transmitida pelo receptor certo não pode ser garantida. Algumas aplicações exigem mecanismos confiáveis para monitorar ou interromper transmissões seletivas ou processos em andamento [37]. Em nanorredes os nós que só recebem as mensagens de comando podem dispensar componentes associados à codificação e à transmissão de mensagens, enquanto os dispositivos que recebem dados de comando requerem recursos de correção de erros [42]. A codificação eficiente é afetada pela capacidade de caracterizar com precisão e modelar as fontes de erro tanto na comunicação molecular quanto na comunicação eletromagnética. Contudo é importante frisar que a codificação em nanoescala precisa ter uma complexidade significativamente baixa devido à capacidade de processamento limitada das nanomáquinas [53] e a probabilidade de retransmissão requer otimização, o que é realizado por troca de dados entre nós. Para exemplo, em [54], os nós compartilham seus padrões de movimento, aproximando sua distribuição local ao longo do tempo e adota um protocolo MAC adicional para limitar com as interferências mais distantes. Para mitigar a taxa elevada de pacotes redundantes na divulgação de dados em toda a rede por retransmissão cega um nó pode retransmitir uma mensagem probabilisticamente para evitar erros [39].

Conforme descrito em [33], a combinação da baixa transmissão, com o ruído de absorção molecular e a interferência multiusuário em nanorredes gera redes sem fio propensas a erros. Técnicas de correção de erro direta são muito complexas para os nanodispositivos. O tempo necessário para codificar e decodificar um pacote pode ser muito mais do que o tempo de transmissão de pacotes [33]. Esquemas de codificação muito mais simples e adaptados às capacidades dos nanodispositivos e as peculiaridades do canal para a banda Terahertz devem ser desenvolvidas. Nessa direção, o uso de controle de erros de baixo peso e códigos de prevenção (EPCs) foram propostos por [55] que defendem redução do número médio de lógica transmitida por o pacote resultando em uma diminuição da absorção molecular geral do ruído e interferência.

Para as redes corporais é necessário considerar a comunicação de rádio sem fio e as necessidades que a topologia demanda. A alta taxa de perda de caminho e a baixa energia dos nanodispositivos, juntamente com a transmissão de mídia ruidosa, aumentam a probabilidade de erros de comunicação, independentemente da grande largura de banda do link. Os esquemas de codificação e prevenção de colisões podem reduzir as retransmissões e a interferência de múltiplos usuário. Neste sentido, é necessário um protocolo de controle de acesso ao meio (MAC) para definir e aplicar mecanismos para assegurar um uso justo do canal de transmissão. Os protocolos clássicos MAC podem ser adaptados para trabalhar em nanorredes baseadas em comunicação via rádio ou acústica. Diferentes pesquisadores defendem o design de protocolos MAC transparentes para nanorredes. Neste protocolo simplificado, o pacote recebido da camada de rede é transmitido para a interface física sem manipular nenhum controle de fluxo, controle de erros ou adicionar cabeçalhos ao pacote [56]. Espera-se ainda, uma grande variedade de protocolos MAC cobrindo recursos de comunicação para as redes intracorporais de tempo real. Muitos desses protocolos já estão disponíveis em simuladores de rede padrão. Contudo a utilização deste recurso exigirá pacotes de dados maiores, existindo a possibilidade de sobrecarga, o que pode gerar um maior consumo de energia no nanotransmissor e no nanorreceptor quando comparado com a camada MAC transparente.

Os nanodispositivos carregados são capazes de transmitir apenas alguns pacotes antes de esgotar completamente a sua energia. Assim, os nanonós devem aguardar para recarregar usando coleta de energia antes de serem capazes de transmitir novamente. Esta característica pode apresentar um atraso inevitável que torna a retransmissão de pacotes um desperdício de recurso [35]. Esquemas de economia de energia para a codificação, correção e detecção de erros na camada MAC foram propostas em [53, 57, 58].

Em pesquisa apresentada por Mohrehkesh e autores [58] foi proposto um protocolo MAC com coleta racional de energia, inspirado pelo CSMA /CA-CD chamado RIH-MAC. O esquema é orientado para o receptor e baseado no protocolo *Receiver-Oriented* MAC (RI-MAC), onde um receptor envia um sinal para todas as fontes emissoras interessadas, uma vez que possuir a energia coletada suficiente para receber um pacote. O protocolo utiliza ainda o conceito de distribuição agendada em nanorredes. D'oro e autores [57] propuseram um protocolo MAC que visava a utilização de um canal de comunicação lógica no qual a informação é codificada através da temporização de transmissões, um canal de temporização para nanorredes.

2.3.3.2. Camada de rede

A formatação e o endereçamento dos pacotes para o meio de transmissão com base em um endereço, assim como a verificação de erros nos pacotes entregues no destino, são funcionalidades da camada de rede no modelo TCP/IP [34]. Em redes tradicionais um esquema de endereçamento é indispensável para viabilizar as comunicações bidirecionais e multi-salto (arquitetura baseada em múltiplos saltos em que os nós são capazes de retransmitir as informações recebidas de outro nó, para um dos nós vizinhos). O pacote deve incluir a informação sobre sua origem e destino. Entretanto, o endereçamento na comunicação molecular é uma questão em aberto, não se sabe ainda como estas informações serão adicionadas em uma molécula de informação [47]. Além disso, a comunicação e o encaminhamento multi-salto é outro aspecto crítico para as nanorredes. Espera-se que a faixa de comunicação disponível para os nanonós seja entre 1 cm e 1 m para comunicação baseada em Terahertz e 1 nm a 1 cm para comunicação molecular o que torna o alcance da transmissão extremamente limitado [43].

Nos modelos naturais de nanorredes, a informação geralmente é codificada e transmitida sem incluir um endereço específico da nanomáquina transmissora. Neste cenário, presume-se que apenas os transmissores autorizados podem ter acesso ao canal e enviar a mensagem codificada, quando esta mensagem chega ao receptor, ele reage de acordo com a informação molecular ignorando sua origem. Este modelo de comunicação é definido como rede centrada em dados. Os padrões de comunicação como este são normalmente encontrados na comunicação por feromônio ou redes usando sinalização de cálcio. Nestes exemplos, as reações no receptor são desencadeadas pelo próprio sinal molecular, que não contém nenhum dado específico sobre a origem. Em contrapartida, o endereço de destino é crucial [47].

Inspirados em soluções baseadas em nanorredes naturais conforme a descrita acima, Dressler e Fischerb [59] apresentam a solução para o endereçamento de redes ópticas e outros *gateways* externos com nanodispositivos corporativos. Para os autores nenhum endereçamento é necessário na direção de saída, se todas as mensagens do interior do corpo forem direcionadas exclusivamente para um dispositivo de análise externa que possui mais recursos. Nesta arquitetura, quando um *gateway* recebe uma mensagem de dentro do corpo, ele simplesmente encaminha para esse dispositivo. Com relação a direção da entrada dos dados, o dispositivo de análise deve ser responsável por enviar comandos para uma ou mais das nanorredes no corpo para pedir que executem uma determinada tarefa de detecção ou atuação. Contudo o endereçamento não estará no nível do endereço de um dispositivo, mas sim em funções, ou seja, na camada de aplicação. Esta solução é semelhante ao trabalho apresentado anteriormente em que Liaskos e Tioliaridou [39] propuseram um esquema de disseminação de dados de inundação para suprir os requisitos para os esquemas de endereçamento nós.

Em nanorredes, atribuir um endereço diferente a cada nanodispositivo não é uma tarefa simples, principalmente devido ao fato de que isso exigiria sincronização complexa e coordenação entre nanomáquinas. Para amenizar este problema, diferentes tipos de componentes podem ter endereços diferentes, mas as nanomáquinas idênticas que se comportar da mesma maneira em termos de comunicação podem assumir o mesmo endereço. Nesse caso, diferentes nós irão reagir da mesma maneira, dependendo da infor-

mação que está sendo detectada [37]. O endereçamento pode, portanto, ser baseado por agrupamento em vez de ser por nó. Isso torna possível abordar um grupo de nós com base na funcionalidade que eles executam ou fenômenos biológicos que monitoram [60].

As nanomáquinas que participam da via de propagação devem ser capazes de encaminhar um pacote para o receptor correto com base no endereço de destino [47]. As nanorredes podem representar a informação transmitida como dados armazenados dentro de um componente de DNA (semelhante a um pacote IP) ou em forma binária, em que o valor geralmente representa uma concentração de moléculas que são transmitidas entre os nós, por exemplo, 1 representa uma concentração específica, enquanto 0 não representa nenhuma transmissão molecular [61]. Pensando em uma rede única que contém milhares de nanomáquinas e muitos nanorroteadores, a comunicação entre todas elas exigiria o uso de endereços muito longos. No entanto, algumas alternativas mais simples e mais viáveis são possíveis. Por exemplo, levando em consideração a arquitetura de rede hierárquica, os nanonós que coordenados por um mesmo nanorroteador podem ter diferentes endereços. Por exemplo, um endereço como G8.I3.R1.N4, pode ser usado para se referir ao nanonó 4, dentro do domínio do nanorroteador 1, conectado à interface nanomicro 3, vinculado ao *gateway* 8 [37].

O roteamento é um requisito essencial para a transmissão de informações através de redes de comunicação. Os protocolos de roteamento tradicionais exigem uma grande capacidade de memória e processadores poderosos para executar funções como atualizações de rota regulares, pesquisa de tabela de roteamento, cálculos de verificação e armazenagem do programa e tabela de roteamento [56]. As nanomáquinas podem ter que responder a uma consulta específica de um centro de comando ou podem precisar relatar novos eventos impulsionados pela base (*pushbased*). Esse fluxo de informações requer o estabelecimento de rotas [37]. Entretanto, as nanomáquinas não têm capacidade de memória para armazenar a tabela de roteamento, nem possuem capacidade de processamento para realizar a pesquisa de rota [56].

Os recursos limitados das nanomáquinas determinam que o projeto de roteamento não deve assumir que os nós tenham conhecimento da topologia da rede [35]. Não é possível sequer considerar que cada nanonó seja capaz de transmitir diretamente ao seu nanorroteador e que a rota será armazenada ou lembrada entre as transmissões [37]. Nesse sentido, é necessário um algoritmo de encaminhamento simples que não deve incluir procedimentos de tomada de decisão, descoberta de rota e outros cálculos complexos. A inundação (*flood*), por exemplo, pode ser previsto como o protocolo de roteamento que funcionará em nanomáquinas [56].

A mobilidade das nanomáquinas dentro do corpo pode ser utilizada para o roteamento e redução do atraso decorrente das limitações de propagação biológica. A direção de uma rota de comunicação não é determinista e depende da velocidade do fluxo das nanomáquinas dentro do corpo [35]. Devido à limitada gama de moléculas ou outros componentes que transportam a mensagem, o roteamento dentro de nanorredes moleculares sem fio deve ocorrer através de múltiplos saltos (*multi-hop*) [61]. Isto é, a informação parte do nó fonte sendo transmitida através de outros nós da rede até chegar ao nó de destino. Por exemplo, em [17], o esquema de roteamento proposto assume uma arquitetura hierárquica baseada em agrupamento. A transmissão de pacotes da fonte para o nanor-

roteador pode ser direta ou *multihop* com base na probabilidade de economizar energia através da transmissão.

Alguns autores defendem a utilização de um esquema de roteamento geográfico para as nanorredes. O método utiliza as informações de localização para decidir se a mensagem deve ser encaminhada ou não. A principal desvantagem deste roteamento é que cada nó deve possuir a informação da localização dos nós vizinhos. Uma solução para este problema seria a troca dos dados de localização entre os nós, contudo isto pode resultar em muita sobrecarga nas trocas de mensagens [62].

Algumas funcionalidades de roteamento podem ser acopladas aos serviços de descoberta e associação da rede. Em uma comunicação baseada em pulso, os nanonós podem ter uma noção da distância entre eles quando ocorre a troca coordenada de pulsos entre dois nós. Deste modo, um nanorroteador pode atribuir endereços lógicos mais baixos aos nós que estão mais próximo e o mesmo endereço para os que estão à mesma distância do nanorroteador. Os vizinhos desses nós que podem estar longe e não receber a transmissão do nanorroteador, podem assumir um endereço mais alto e transmiti-lo [37].

No protocolo de roteamento geográfico apresentado por Tsioliaridou e autores [63], os nós das nanorredes assumem que há pelo menos dois nós de apoio que possuem maior capacidade de comunicação e processamento do que os demais (usuários). Os autores assumem que a topologia da rede é quadrada com quatro nós de apoio distribuídos nos quatro cantos. Para o esquema de roteamento, os nós de usuário medem suas distâncias dos apoios. Em seguida, um nó de origem seleciona os nós de apoio e incorpora essa informação no cabeçalho de um pacote. Um nó receptor verifica sua localização, a localização do destino e a localização da fonte para decidir sobre o encaminhamento ou o descarte do pacote.

2.3.3.3. Camada de transporte

No modelo TCP/IP, a camada de transporte é responsável por gerenciar o fluxo de transmissão, garantir a integridade dos dados até o destino e reenvio de mensagens caso ocorra a perda de pacotes ou algum erro na transferência. Estas funções são normalmente realizadas pelos protocolos TCP e UDP [64]. Em nanorredes as soluções da camada de transporte são necessárias para lidar com possíveis falhas nos dispositivos e constantes desconexões devido a oscilação de energia ou efeitos temporários causados pela composição dos canais moleculares [33]. Nanomáquinas biológicas podem apresentar transmissão não confiável devido ao alto nível de ruído biológico. Por exemplo, o vapor de água pode interferir nos sinais eletromagnéticos, o movimento dos fluidos podem ser contrários e prejudicar a difusão molecular [61] enquanto que uma explosão repentina de moléculas pode criar desconexões temporais da rede em diferentes pontos. Se estes problemas forem apenas um efeito local, um protocolo de roteamento pode determinar um caminho alternativo, caso contrário, afetará toda a rede [37].

Para algumas aplicações específicas, uma opção pode ser simplesmente aumentar o número de nanomáquinas cobrindo a mesma área [37]. Uma aplicação densa de nanomáquinas poderia permitir a comunicação mais confiável de forma abrangente compensando a perda potencial de pacotes, uma vez que mais nanomáquinas com a mesma

funcionalidade podem relatar os mesmos dados. Em contrapartida, o controle do congestionamento, especialmente em um cenário com muitas nanomáquinas é um desafio não abordado na literatura. Embora algumas técnicas de comunicação eletromagnética possam ser adaptadas para a banda Terahertz, ainda não está claro como o congestionamento será tratado na comunicação molecular [35].

Quando se trata de interferência molecular transitória, são necessárias soluções mais complexas. Por exemplo, moléculas absorventes para criar picos de atenuação. Protocolos conscientes de percepção, em que as nanomáquinas podem detectar quais janelas estão disponíveis por meio de nanosensores químicos, podem ser uma solução [37]. A conexão orientada as capacidades permitem que um canal de controle seja configurado entre o remetente e o receptor, que será capaz de controlar a taxa de liberação da molécula no remetente com base no número de moléculas absorvidas ou detectadas em nós alvos, no ambiente circundante [44].

Em redes de computadores tradicionais, a maioria dos protocolos de aplicação faz uso do protocolo de controle de transmissão TCP (*Transmission Control Protocol*) [65]. Um protocolo bidirecional de rede que opera na camada de transporte para fornecer confiabilidade na transmissão dos dados e garantir o controle de congestionamento e de fluxo para os aplicativos que funcionam na ponta da rede [64]. Considerando que maioria do tráfego sobre a Internet é transportada pelo TCP. Parece razoável a ideia de modificá-lo mantendo a compatibilidade, em vez de propor protocolos radicalmente novos para as nanoredes. No entanto, para que isto seja possível, novas extensões para o protocolo precisam ser desenvolvidas [33].

Enquanto a implementação completa do TCP na comunicação molecular é difícil ou inviável, algumas de suas funções, como a conexão orientada capacidade, confiabilidade, congestionamento e controle de fluxo podem ser traduzidos para as redes moleculares. Em [44] um protocolo orientado a conexão foi proposto. O protocolo faz uso de sondagem semelhante ao TCP para encontrar uma taxa de transmissão adequada entre um transmissor e um receptor a fim de evitar o congestionamento. Uma vez que os atrasos típicos nas comunicações moleculares são muito maiores, e uma solução baseada em *feedback* não é adequada pois aumentaria o atraso na comunicação. No protocolo proposto os sinais de *feedback* são enviados pelo receptor para conter a taxa de envio no transmissor. De forma mais detalhada, uma vez que o remetente recebeu o sinal inicial do receptor, o remetente começa liberando moléculas em rajadas, e aumenta linearmente o número de moléculas transmitidas durante cada rajada, da mesma maneira que o TCP faz com o número de bytes para evitar o congestionamento. Quando chega um *feedback* negativo, o destinatário interpreta como a indicação para reduzir a taxa de transmissão.

2.3.3.4. Camada de aplicação

A camada de aplicação possui a funcionalidade de realizar a comunicação entre os aplicativos e o protocolo de transporte. O processo começa com um programa enviando uma requisição para a camada de aplicação. Cada programa interage com um protocolo diferente conforme o seu propósito. O protocolo da camada de aplicação processa a requisição e transmite para outro protocolo da camada de transporte (geralmente o TCP),

que irá particionar o dado em pacotes para ser encaminhado e processado nas camadas inferiores. Existem vários protocolos que operam na camada de aplicação em uma rede convencional, entre eles estão: FTP (*File Transfer Protocol*) utilizado para a transferência de arquivos, o HTTP (*Hypertext Transfer Protocol*) utilizado como base da comunicação na Internet, SMTP (*Simple Mail Transfer Protocol*) aplicado em requisições de e-mail, o Telnet utilizado para a comunicação baseada em texto interativo e o RTP (*Real-time Transport Protocol*) utilizado em aplicações de tempo real, tal como, a entrega de dados de áudio (voz sobre IP) [34].

A natureza heterogênea das nanomáquinas e seu uso para vários fins dentro do corpo resultará em diferentes formatos e representações de dados. A fusão de dados, portanto, precisa ser otimizada, dinâmica e tolerante ao atraso principalmente para os aplicativos que dependem da integração com diversas outras fontes de dados [61]. As nanorredes implementadas dentro do corpo humano precisam de um mecanismo não invasivo para reconhecer o cenário, lembrando que algumas condições podem depender da mobilidade do nó. Além disso, as aplicações podem precisar coordenar múltiplos serviços. Por exemplo, uma rede no corpo pode detectar a mudança no quadro do paciente e ser alertado por uma rede ambiental sobre a presença de um componente alérgeno. A rede intracórpore é então notificada e libera a medicação no organismo para controlar a reação alérgica [66]. O aspecto crítico das aplicações médicas faz com que a operação em tempo real seja um requisito fundamental. No entanto, devido ao meio de transmissão imprevisível e ao intervalo significativamente curto da comunicação intracórpore, é esperado um atraso aleatório [35].

Em [67], o meio de comunicação é sobre neurônios naturais. O protocolo chamado Neuronal TDMA, foi proposto para otimizar a programação de sinalização para nós individuais na rede. O objetivo do protocolo é prevenir o atraso e evitar possíveis interferências no sinal. Os autores definem que a camada de aplicação deve ser utilizada para a transdução de sinal neural (processo através do qual uma célula converte um tipo de sinal ou estímulo em outro), para o controle de neuroprótese (comunicação do neurônio motor que permite controle de equipamentos pelo pensamento) e para o sensoramento fisiológico. Entretanto, a definição de um protocolo para a camada não é explorado.

O estudo de [68] aborda a transmissão de vídeo em tempo real através de nanorredes utilizando o protocolo da camada de aplicação RTP. O modelo é desenvolvido e simulado pensando em nanoantenas de grafeno como fonte de transmissão de dados e segue o esquema: No lado do transmissor, a fonte fragmenta o vídeo original em pacotes. As informações de cabeçalho como identificação do pacote, tamanho de carga útil e vistos de data/hora são adicionados. O receptor extrai o cabeçalho de cada pacote e cria o vídeo reconstruído. Segundo os autores, para evitar o congelamento do vídeo não deve ocorrer a retransmissão de pacotes perdidos, nesse sentido, as futuras camadas nano sem fio provavelmente terão que ser ajustadas para o *streaming* de vídeo. Os resultados de simulação indicaram que as ferramentas e os modelos existentes não são adequados para esse estudo. A simulação do *jitter* (a diferença entre atrasos de pacotes) do NanoSim e as perdas no lado do receptor não podem ser consideradas realistas, pois não levam em conta a contenção do canal e os atrasos na transmissão que ocorrem nas nanorredes.

A maioria das pesquisas publicadas exploram a camada física para o projeto de

comunicação das nanorredes. Ainda não está claro como serão as aplicações e consequentemente poucos autores abordam a camada de aplicação no ambiente nano. Em contrapartida, as soluções que são apresentadas são adaptadas para cada caso específico. Por exemplo, em [69], os autores assumem que a camada de aplicação estará contida na nanomáquina enquanto uma célula assume o papel de interface de aplicação em uma implementação que utiliza a capacidade dos vírus para transportar e transduzir a informação genética entre células biológicas. Na pilha de protocolo para nanorredes sem fio de [70], a camada de aplicação fornece funções para desenhar circuitos lógicos para a comunicação. Já em [71], a camada de aplicação é responsável pela perda de pacotes, pois no modelo o protocolo já entrega os dados em ordem e sem perdas através do handshaking entre o solicitante e o responsável. As pesquisas atuais sobre os desafios para implementar os protocolos tradicionais da camada de aplicação em nanorredes devem ser exploradas. Para tanto existe a necessidade de fornecer modelos matemáticos e físicos confiáveis, bem como simulações computacionais que podem ser aplicadas no meio biológico [35].

2.4. Desafios de pesquisa

Com a intenção de expor os principais tópicos de pesquisa para os próximos anos em nanorredes, comunicações moleculares e Internet-das-bionano-coisas, nesta seção descrevemos os desafios de pesquisa para: nanorredes baseadas em comunicações eletromagnéticas, acústicas, e moleculares, reformulação da camada TCP/IP e integração com os redes avanços de redes de computadores e o 5G.

Grande parte da comunidade das nanorredes consideram as comunicações eletromagnéticas uma área interessante e de grande contribuição para redes sem fio, como um todo. A frequência alta da portadora permitem que envie-se Terabits de dados por segundo, mas os desafios ficam em torno das qualidades dos links entre os dispositivos e a maneira que uma rede pode ser composta. Mesmo para dispositivos em escalas maiores que a nano, a propagação de ondas Terahertz possuem um coeficiente de absorção alto tanto para o meio como o ar, como também para supercies e obstáculos. Isso significa que as conexões por Terahertz se limita tanto na distância entre os dispositivos quando na manutenção de linha de visada. Mesmo proposta já tem aparecido para combater esses problemas usando refletores ou espelhos [72], esse problema encontra-se longe de ser resolvido também pelo baixo nível controle dos feixes do sinal. Esse problema é bastante correlacionado com a distância entre os dispositivos, que a partir de um metro de distância já decresce a taxa de transmissão em quase 100x. A solução mais direta que pode-se encontrar é o aumento da potência no transmissor e a sensibilidade no receptor. Entretanto, a eficiência energética desses dispositivos irá gradativamente ser afetado, tornando a aplicação dessa tecnologia em nanorredes bem mais desafiador. Acredita-se que se a escala dos dispositivos, e da rede, diminuem, a complexidade de todas as partes do sistema deve diminuir equivalentemente [73]. Essa lição já fora obtido em redes de sensores sem fio e agora a estória também se repete para as nanorredes. Para solucionar esses problema, acreditamos que pesquisadores devem se concentrar em protocolos de comunicações robustos mais não flexíveis. Mesmo sendo protocolos limitados, podemos imaginar que no futuro dessas redes existirá muitos tipos de dispositivos que fazem funções diferentes e possuem protocolos diferentes. Dessa maneira eles também podem ser aplicados em ambientes diversos.

As nanorredes baseadas em comunicações acústicas possuem desafios parecidos com as eletromagnéticas, entretanto, a miniaturização dos transmissores acústicos é bem mais difícil. Isso se deve pelo fato de o transmissor fazer a transdução do sinal eletromagnético para o acústico, e possuir uma estrutura composta com diversos cristais que vibram os nanocabos. Pouco foi o interesse de se fazer essas estruturas na escala nano, até anos atrás, em que a comunidade se deparou com a possibilidade de fazer o sensoriamento dentro do corpo humano em tempo real e remotamente, a partir da disponibilidade de uma tecnologia que faça essas conexões serem ligadas à Internet. A partir desse evento, outros desafios podem surgir subsequentemente. As diferentes características de sinais eletromagnéticos e acústicos, mediante a uma certa frequência de ressonância, podem também implicar em dispositivos de recepção complexos que consigam manter a sincronia entre dispositivos que se comunicam por diferentes meios, o que também implica na miniaturização dos dispositivos. Apesar de a comunidade das nanorredes ainda ter pouco interesse nesse tipo de comunicações por esses desafios, a construção de uma nanorede inteiramente baseada em ondas acústicas ainda é considerada a forma mais atrativa de comunicação dentro do corpo humano.

Por causa da grande concentração de trabalhos na camada física nas nanorredes baseadas em comunicações moleculares, pouco é investigado sobre a camada de enlace e roteamento. Outro desafio é o questionamento sobre o que caracteriza a informação, pois a pluralidade de moléculas e seus arranjos podem atingir níveis complexos de codificação de informação em amino-ácidos, proteínas e até DNA [74, 75]. Dessa forma, os protocolos de acesso ao meio na camada de enlace, que ainda não são desenvolvidos, provavelmente deveriam primeiramente levar em conta como essa codificação das moléculas podem ser usada para diferentes nanomáquinas possam se comunicar ao mesmo tempo. O uso simplesmente de tipos diferentes de moléculas não é suficiente para acomodar o número de nanomáquinas que é previsto para serem implantadas tanto no corpo humano quanto fora. Essa pluralidade de mecanismos celulares também podem ser a chave para obtenção de mecanismos de endereçamento, bem como também roteamento. Recentes descobertas em biologia sugerem que as configurações interna das células e interações entre as proteínas podem criar estruturas únicas chamadas de BioID [76]. Elas podem também serem usadas para criar reações que podem ser observadas nos microscópicos dependendo dessas interações entre proteínas. Esses mecanismos ainda pouco foram investigados na comunidade de comunicações moleculares, em que comunicações baseados em interações de proteínas ainda é inexistente.

Grande parte dos desafios para a reformulação da arquitetura TCP/IP já foram abordados na seção anterior, mas aqui traremos uma visão mais abrangente dos desafios que se relacionam com essa iniciativa. Independentemente do tipo de organização entre camadas, ou até camadas únicas, as nanorredes devem se estabelecer como uma parte fundamental da infraestrutura da Internet mas sim como uma extensão que permite uma maior flexibilidade. Isso vai do tipo de informação que podemos obter, e que será transmitido pela rede. Ainda pouco se sabe sobre quanto de dados biológicos serão ultimamente transmitido na rede quando toda essa tecnologia esteja disponível. Entretanto, esses tipos de dados possam não fazer sentido para os diversos métodos usados dentro do todo sistema que envolve o TCP/IP. Isso significa que novos tipos de tráfego de dados surgirão, que poderão não ser caracterizados como o tráfego atual da Internet atualmente se comporta.

Isso deverá gerar uma movimentação em torno de todos os pesquisadores para o problema de interoperabilidade desses dados e como isso vai afetar a rede. Com toda essa lógica que permite visualizar o impacto futuro dessas redes de comunicações, a comunidade de redes de computadores deve-se concentra na construção de *middleware* tanto no *hardware* e no *software* que fara esse tipo de integração das nanorredes com o TCP/IP. Mesmo com conceitos e desenvolvimento inicial dessa tecnologia, já poderíamos avaliar qual realmente será o impacto de se conectar milhões de nanomáquinas à Internet e como isso pode virar a mudar a arquitetura TCP/IP.

Com os avanços recentes de redes de computadores em expandir o conceito de comunicações, como um todo, para um ambiente virtual, dinâmico e flexível também deve ser considerado nas nanorredes bem como elas devem ser integradas. As nanorredes podem ser caracterizadas pela baixa complexidade e funcionamento limitado, mas a reprogramação desses dispositivos pode trazer muitos avanços para essa tecnologia. Uma nanorede definida por software ainda não é proposta pela comunidade, mas acredita-se que tal tecnologia pode trazer uma modularidade de funcionamento das nanomáquinas em prol de funções mais complexas. Isso seria uma alternativa para o problema de se gerenciar um quantidade enorme de nanomáquinas, em então, quando reprogramadas, possam fazer diferentes funções e do mesmo modo manter as dimensões da nanorede. Podemos imaginar, para as nanorredes baseadas em comunicações moleculares, que a biologia sintética é uma ferramenta fundamental no desenvolvimento de redes de comunicações biológicas reprogramáveis. Essas e mais outras iniciativas podem ser fundamentais para a integração dessas redes com a infraestrutura visada como o futuro para as redes de comunicações e sistemas móveis, como o 5G. Como essa tecnologia é baseada em redes e rádios definidos por software, as nanorredes deverão possuir também níveis de programação e abstração de suas funções que permita uma integração mais eficiente. Claramente esse tipo de pesquisa também deve ser pensada e iniciada nos próximos ano. Como o 5G também é visada para se inserir exponencialmente nos verticais da sociedade, como nas nanorredes (ver próxima Seção), a integração dessas duas tecnologias permitirá uma revolução mais rápida nos serviços que compõem todos os setores da sociedade.

2.5. Aplicações

Com o progresso do desenvolvimento deste novo paradigma de redes de comunicações, percebemos um gradativo surgimento de aplicações em todos os setores da sociedade. As nanorredes apresentam-se com o objetivo principal de acessar informações na escala nano, mas também como uma forma colaborativa de executar tarefas através da rede a qual pertencem. Com essa mentalidade, essas redes podem desde compor materiais e produtos, bem como acessar informações em lugares de difícil acesso para humanos executar tarefas e serviços. Nesta seção, apresentaremos as recentes propostas de aplicações para os seguintes setores da sociedade que acreditamos ser de imediata aceitação dessas redes: tecnologia da informação, medicina, neurociência, agricultura de precisão, poluição e indústria.

2.5.1. Tecnologia da Informação

A área de tecnologia da informação poderá sofrer uma drástica mudança nos serviços que ela promove pela quantidade e heterogeneidade das informações acessadas com o

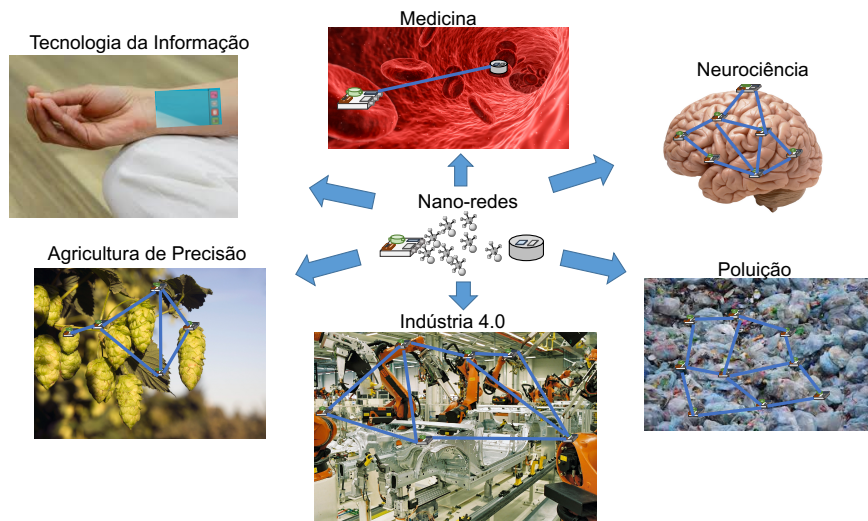


Figura 2.11: Aplicações das nanorredes.

desenvolvimento da Internet das Bionano-Coisas. Os novos serviços podem variar de acordo com sua aplicabilidade. Isso pode mudar a caracterização do tráfego atual, o qual é composto em sua maioria pela transmissão multimídia. No futuro, os milhares de nanonós espalhados pelo nosso ambiente estarão constantemente enviando novas informações para os servidores desses novos serviços. Esse novo processo pode fazer com que outros tipos de dados sejam os mais trafegados na rede. Também podemos prever que o número de dispositivos por pessoa crescerá exponencialmente, e todas as pesquisas atualmente sendo desenvolvidas para dimensionamento da rede global de redes de comunicações deverão incluir essa potencial adição na competição por ainda mais conexões na rede.

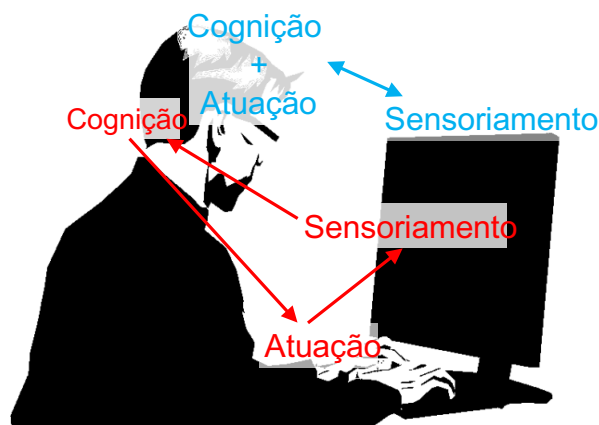


Figura 2.12: Aplicações das nanorredes em interações homem máquina.

A interação humano-máquina poderá mudar, com serviços que habilitem mais eficiência tanto nas experiências dos usuários de sistemas de informação, como na aquisição de informações desses usuários. Por exemplo, com sistemas de nanorredes baseados em interface entre cérebro-máquina podem mudar o sistema de entrada de comandos em sistemas digitais por fazer uma ponte direta entre a consciência humana e os dispositivos elétricos. Isso poderia eventualmente eliminar o uso de outros dispositivos que fazem

esse trabalho, como teclados ou mouse. Essa tecnologia pode ser imaginada para outros setores que usam de sistemas motores do corpo humano para execução de tarefas, como por exemplo, os sistemas industriais ou automobilísticos. Essa maior integração humano-máquina já começou a ser desenvolvida de uma maneira mais próxima da realidade. A próxima geração de telefones móveis possuirão bio-chips ou bio-máquinas que são capazes de analisar sinais biomoleculares do corpo humano (e.g., células sanguíneas ou substâncias no suor) que são capturadas por intermédio do contato das mãos ou ouvidos pelo telefone celular [77]. As sugestões mais ambiciosas já propõem a idéia de que esses telefones celulares do futuro farão parte do nosso próprio corpo, com a ideia de que células reprogramáveis possam formar um *display dermal* sobre nossa pele com um sistema de comandos baseados no processamento do tato [6, 78].

2.5.2. Medicina

As comunicações moleculares dentro do corpo humano são diretamente ligadas à saúde do organismo e, também, às práticas medicinais que permitem o diagnóstico, o tratamento e a cura de doenças. As comunicações entre células, órgãos e sistemas no corpo humano envolvem a transmissão, propagação e recepção de inúmeras moléculas e proteínas que transmitem informações necessárias para as reações energéticas, ajudando no metabolismo das partes envolvidas. Por exemplo, podemos observar o sistema de aceleração cardíaca mediante a estímulo visual. As imagens processadas pelo cérebro ativam uma transmissão de pulsos elétricos dos neurônios do hipocampo até a medula adrenal através da medula espinhal. Esses pulsos elétricos, após recebidos pela medula, irão iniciar a produção e a transmissão de adrenalina na corrente sanguínea. Essa molécula servirá de informação para vários órgãos e células, que globalmente fará o corpo humano responder adequadamente a esse estímulo visual.

Quando células, órgãos ou até sistemas falham tanto na transmissão, quanto na recepção de informações moleculares no corpo humano, doenças naturalmente surgem. Essas doenças podem ser internas, ou seja, uma falha na comunicação celular que causa um grave dano no funcionamento organizado das células de um órgão e depois sistema, até que o organismo sistematicamente venha a responder a essa doença. Essas doenças podem também ser causadas por organismos externos, como bactérias, vírus, ou moléculas que danifiquem o funcionamento correto e a comunicação de moléculas e proteínas.

A medicina, de certo modo, já vem aplicando essas práticas de correção de comunicações moleculares a vários anos. Entretanto, só recentemente, práticas médicas vêm sendo redesenhadas com os avanços de outras áreas do conhecimento, como a engenharia eletromecânica, a nanotecnologia e biotecnologia. A engenharia eletromecânica vem continuamente produzindo sistemas de diagnose e tratamento de doenças. Esses sistemas variam entre sensores móveis e máquinas robustas que conseguem acessar camadas posteriores à pele, como por exemplo, scanners ultrassom, MRI, entre outros. De um outro lado, robôs cirúrgicos vêm sendo desenvolvidos continuamente desde os anos 80, onde eles vêm contribuindo para assistir os médicos cirurgiões em prol de uma maior precisão e bons resultados nas salas cirúrgicas. E ainda mais recentemente, a tecnologia da informação veio para permitir que a Internet possa ajudar ambientes médicos, tanto no gerenciamento de ambientes médicos com sistemas robustos autônomos, quanto no processamento de dados em escalas moleculares. A nanotecnologia vem também apoi-

ando o desenvolvimento de técnicas menos invasivas para diagnose, tratamento e cura de doenças. A nanomedicina é oficialmente uma área que promete materiais e sistemas minimamente invasivos ao corpo humano para acessar níveis de detalhes sobre o seu funcionamento e também repensar novas práticas medicinais com resultados inimagináveis. Por fim, a biotecnologia propõe o uso de biologia sintética para tratamento e cura de doenças. Esse tipo de tecnologia atrai mais atenção pelo fato de regenerar tecidos e órgãos e promete trazer maior eficiência em todas as práticas médicas. Entretanto o custo e a complexidade desse sistema dificulta a popularização dessas técnicas agora e por muitos anos. As técnicas baseadas em regeneração possui longo processo ético e de segurança, onde a burocracia e os sistemas reguladores estatais interferem diretamente no desenvolvimento industrial dessas técnicas e inviabilizam a rápida disponibilidade no mercado.

As comunicações moleculares prometem ser um área que reúne tanto a engenharia eletromecânica, como também a nanotecnologia e biotecnologia para uma reformulação das práticas médicas. Com o objetivo de trazer a personalização e a integração de diversas formas de diagnose, tratamento e cura de doenças, essa tecnologia de comunicação pode fazer com que falhas de comunicações de sistemas celulares dentro do corpo humano sejam detectadas com a ajuda de técnicas de nanotecnologia, remotamente observadas por médicos por meio da Internet-das-bionano-coisas. Estas podem acionar, também remotamente, tratamento e curas com medicina regenerativa por meio de avanços na biotecnologia. Isso também não impede que essas comunicações moleculares contribuam para essas áreas individualmente, como é abordado em outras subseções. Mas em termos de medicina, as comunicações moleculares já vêm ativamente propondo sistemas que apresentam grande potencial revolucionário.

A primeira aplicação que chamou atenção de comunicações moleculares em medicina foi proposta por Chahibi *et al.* [79]. Os autores propuseram a idéia de que a Entrega Inteligente de Medicamentos (do inglês, *Intelligent Drug Delivery*), que consiste em controlar adaptativamente a taxa e frequência de medicamento a entrar no corpo humano por uma seringa, dependendo de que local no corpo humano deseja-se obter uma maior concentração desses medicamentos. Esse trabalho abre o espaço para modelar o sistema sanguíneo do corpo humano como um sistema de comunicação de dados para fins de otimização. Os mesmos autores também usaram dessa metodologia para adaptar a modelagem previamente usada para fins de modelar a propagação de anticorpos e antígenos. Esses trabalhos foram estendidos por Chude *et al.* [80], no qual investigou uma maneira de maximizar a entrega de medicamentos em múltiplos pontos no corpo humano. Esse mesmo autor também investigou como as nanomáquinas podem ser removidas do corpo humano, trabalho que tem impacto na aplicações clínicas desta tecnologia.

A biologia sintética já está sendo utilizada como um objetivo dual de solucionar problemas na engenharia biomédica como também uma ferramenta básica para sistemas de comunicações moleculares. Em [81], um sistema microfluídico foi criado para fazer um processamento sanguíneo por meio de dispositivos de microfluidos, que efetivamente filtra o vírus Ebola. Baseado num processo de diálise, esse método permite que todo o sangue passe pelo dispositivo de microfluidos, sendo completamente processado. Dentro do dispositivo, existe uma colônia de bactérias engenhas para permitir o encaixe do vírus em sua superfície, e por meio do mecanismo de nado natural das bactérias, vírus podem ser coletados por toda essa área. Mas nem sempre comunicação das bactérias

são benéficas. Como por exemplo, biofilmes são colônias de bactérias que podem causar infecções e inflamações em organismos hospedeiros. Elas apresentam um sistema de comunicação robusto que serve para manter uma colônia com diferentes tipos de bactérias. Elas se organizam dessa maneira para otimizar o consumo de nutrientes encontrados no ambientes nas quais se instalam. Com base nessas informações, em [82], os autores apresentaram um sistema de redes de comunicação de bactérias que se organizam para fazer uma parede que impede que nutrientes cheguem ao alcance do biofilme. Com isso, esse sistema de redes de comunicação permite que o biofilme passe fome e acabe quebrando com sucesso.

As bactérias são atrativas pelo fato de que são nanomáquinas naturais e podem ser literalmente reprogramadas a desejo de humanos para servi-los com simples tarefas até processamento lógico de informação. Mas não são a única opção. Em [24], os autores exploram como os vírus podem ser utilizados para transmitir informação de modo ativo. Esse trabalho abre as portas para outros métodos de transmissão de informação em vírus, que podem servir também de base para estudo de como os vários vírus que assustam a humanidade podem ser controlados para fins de tratamento e cura de doenças. Sabe-se atualmente que vacinas e terapias genéticas já utilizam vírus como um propagador de informação genética, mas com a ajuda das nanorredes, esses sistemas podem ganhar mais flexibilidade com o maior nível de segurança e eficiência.

2.5.3. Neurociência

As nanorredes podem se comunicar com o cérebro e ajudar a contribuir para os esforços da neurociência em entender o funcionamento do mesmo. As nanorredes juntamente com as comunicações moleculares podem ajudar a neurociência do mesmo modo que a neurociência vem contribuindo para inteligência artificial e aprendizagem de máquina. Entretanto, não se exclui a possibilidade de que os mesmos princípios já mencionados anteriormente, em medicina, podem ser reaplicados para tratar de doenças que afetam o cérebro. Com os avanços na expectativa de vida, nota-se que mais e mais idosos são afetados com doenças neurodegenerativas. Essas doenças afetam os mais básicos sistemas de comunicações entre neurônios, astrócitos, micróglia e oligodendrócitos, e apresentam diversos sintomas como perda de memória, demência, perda de função motora até que a perda de funções parassintéticas levem colapso do corpo humano.

Com o interesse de entender como as redes neurais se comunicam, a comunidade de nanorredes e comunicações moleculares se concentrou inicialmente na modelagem desses sistemas usando teoria da informação e codificação. Os sistemas de comunicação molecular neuronal foram previamente estudados usando os princípios de informação e comunicação nos últimos oito anos com uma intensa concentração no projeto de sistemas de comunicação e cálculo de capacidade de população única e de neurônios. Uma rede de backbone neuronal foi proposta como um novo paradigma de comunicação por Walsh *et al.* [20]. Este trabalho foi ampliado por Balasubramaniam *et al.* [83], no qual um protocolo de acesso múltiplo foi apresentado usando uma técnica de otimização de algoritmo genético. Trabalhos mais recentes se concentraram na modelagem e também no desempenho de comunicação da rede, tais como a modelagem sobre a transmissão sináptica baseada em terminais pré-sinápticos e vetores de transmissores [84]. Caccipuoti *et al.* mostraram que a informação neuronal é codificada nos padrões de liberação dos

terminais pré-sinápticos e analisou como esse sistema processa informações. Ramezani *et al.*, por outro lado, estavam mais interessados em como a forma do potencial de ação durante a propagação axonal pode afetar um sistema de comunicação molecular baseado em neurônios. Em [85], os autores analisaram a comunicação das populações de neurônios e astrócitos. Eles investigaram o estímulo-resposta do sistema astrócitos-neurônio, analisando as sinapses tripartidas sob a densidade espectral de potência, a resposta ao impulso e o canal de comunicação.

Mais recentemente, a comunidade está focando também em trabalhos que apliquem conceitos e sistemas de comunicações moleculares para novas tecnologias que ajudem nas doenças cerebrais como neurodegeneração. Essas doenças vêm crescendo a partir do aumento da expectativa de vida humana. Por exemplo, na Europa essas doenças já totalizam 35% de pacientes tornando a segunda doença que mais mata. Nos EUA, quase 500 bilhões de dólares americanos já são acumulados em gastos para tratar pacientes em diversas doenças neurodegenerativas. A maioria dos tratamentos dessas doenças ainda se baseiam em medicamentos que só tratam os sintomas, mas não curam a doença. Esses medicamentos também são acompanhados por grandes níveis de intoxicação como também efeitos colaterais que prejudicam a vida dos pacientes. Por isso novos métodos de prevenção e tratamento de doenças neurodegenerativas ainda é uma área que atrai bastante esforços de pesquisas científicas. Existem dois métodos de tratamento que a comunidade de redes moleculares e nanorredes propuseram contribuições, que são os métodos de estimulação cerebral profunda e biologia sintética.

Em [86], um método de estimulação cerebral profunda foi proposto usando uma arquitetura multinível de nanorredes. Os nós fundamentais dessa rede são chamados de *neural dust*, um dispositivo em nanoescala que possui um LED, um circuito retificador e um conjunto de nanofios. Os nanofios são balançados com ondas de ultra-som e por meio do circuito retificador são capazes de ativar o LED pela conversão eletromecânica do balanço dos nanofios. O LED precisa ser ativado para iluminar as superfícies dos neurônios e ativá-los, fazendo assim uma estimulação dos neurônios a nível celular, o que é chamado de optogenética. Um conjunto desses dispositivos podem se organizar efetuar uma estimulação em um conjunto de neurônio de maneira controlada. O controle vem a partir das camadas superiores desta rede, que compõem um dispositivo subdural e um dispositivo externo. O dispositivo externo se comunica por meio de ondas eletromagnéticas com o dispositivo subdural que se comunica com os *neural dusts* por meio de ultrassom. O motivo por esse sistema heterogêneo se baseia na propagação dessas ondas dentro do corpo humano, sendo que o ultrassom possui mais eficácia em atingir níveis profundos em tecidos biológicos. Essa hierarquia também permite um controle externo das estimulações cerebrais, o que torna essa técnica mais atrativa pelo fato de ser menos invasiva e não requer cirurgia quando é implantada. Entretanto, os autores não fornecem detalhes em quais protocolos podem ser utilizados nessas redes e também quais limitações aparecem na utilização de longo prazo e de como essa rede se comporta em larga escala.

Visando tratamentos em nível celular para doenças neurodegenerativas, Barros e Dey [87] propuseram um método de controle indireto de sinapses quando manipulam-se células chamadas astrócitos. Esse método de controle baseia-se em fornecer uma estimulação em uma determinada taxa que regula a sinalização de cálcio dentro dessas células. O cálcio é responsável pela transmissão de gliotransmissores no canal sináptico o que

modulariza o sinal entre um neurônio pre-sináptico e um neurônio pós-sináptico. Essa relação faz com que a qualidade na transmissão desses sinais sinápticos também sejam controlados e a plasticidade dessas conexões sejam manipuladas. Eles demonstram como o cálcio pode ser controlado a ponto de estabilizar a concentração de cálcio em um ponto preciso por um tempo pré-determinado, consequentemente, a transmissão de gliotransmissores pode também ser estabilizada. Mesmo com a criação dessa técnica inovadora, os autores não exploraram ainda como de fato essa técnica pode ser implementada como um tratamento de doença degenerativa, e muitas questões ainda estão sem resposta como o controle distribuído dessas células e que técnica de estimulação dos astrócitos devem ser apropriadamente escolhida [88].

2.5.4. Agricultura de precisão

A agricultura de precisão é uma área que visa otimizar todos os processos que envolvem o agronegócio. Esse método visa reduzir os custos, doenças e efeitos no ambiente e aumentar a segurança dos alimentos, bem como a produção de derivados do leite, carne e afins. Todo esse processo funcionará em torno da internet-das-coisas, em que uma contínua coleta e processamento de dados faz com que o gerenciamento dos processos na agricultura sejam precisamente sintonizados. O impacto no agronegócio é considerável, visto que a sustentabilidade do agronegócio atingirá níveis superiores o que aumentará a segurança de alimentos, que por meio da internet-das-coisas, transforma as fazendas com um processo mais inteligente.

Entretanto, pesquisadores já percebem que o modelo atual de internet-das-coisas para o agronegócio possui falhas no sensoriamento do sistema. Os sensores de baixa tecnologia não são eficientes em adquirir dados precisos do ambiente. Se esses sensores obterem dados em nível microscópico para inferências de mais processos biológicos que afetam o solo, gramado e animais, a precisão em gerenciar os processos no agronegócio podem obter acurácia em níveis altos. Como por exemplo, as bactérias têm um processo fundamental tanto para o solo como para a saúde de animais como as vacas. As diversas plantas que existem em uma fazenda interagem com o micro-ecossistema de bactérias no solo onde o processo de simbiose existe na qual nutrientes são trocados entre essas partes. Bactérias também são encontradas dentro dos estômagos de vacas e ajudam no processamento de comida para nutrição desses animais tendo um impacto tremendo na saúde deles. Informações sobre esses processos biológicos são capaz de produzir novos métodos de monitoramento em micro-escala, o que transformaria os processos de estimação do ambiente para níveis bem mais altos em termos de performance do que os encontrados atualmente. Por esse motivo, a Internet-das-bionano-coisas pode ser essa nova plataforma, onde processos biológicos podem também interagir com sistemas digitais para fins do gerenciamento das fazendas.

Bhargava *et al.* investigaram quais os desafios que se encontram em implementar uma tecnologia em nanoescala para incorporação em ambientes de agricultura [89]. Eles destacam dois aspectos, o desenvolvimento de dispositivos em nanoescala e a comunicação desses dispositivos entre eles e com a Internet. Dispositivos em nanoescala possuem a vantagem de ser extremamente simples de ser projetados. Entretanto, a miniaturização envolve processos complexos de produção de materiais e até metamateriais. Esses dispositivos são dedicados ao ambiente, ou seja, dispositivos desenvolvidos para interagir

com as bactérias não são os mesmos para interagir com as plantas. Com isso, faz com que diversos tipos diferentes de dispositivos sejam previstos para serem desenvolvidos nos próximos anos. Com isso a comunicação de dados se torna bem mais complexa em termos de interoperabilidade do que se é transmitido, o que requer uma constante investigação de como sistemas atuais de redes, como *software-defined networks*, podem ajudar em resolver problemas como esse [90].

Mais recentemente, os pesquisadores investigaram a propagação de informação em ambientes de agricultura usando ondas Terahertz. Embora a propagação dessa faixa de frequência em ambientes abertos é ainda bem questionada, existe um atrativo de que as antenas são extremamente pequenas, o que ajuda no desenvolvimento de dispositivos em nanoescala. Afsharinejad *et al.* desenvolveram um sistema de comunicação e protocolos para formar um framework cross-layer para monitoramento de plantas [91]. Como a faixa de frequência Terahertz é seletiva, esses pesquisadores indicam que pode-se otimizar a transmissão de dados das plantas ao utilizar frequências específicas em determinados períodos.

2.5.5. Poluição

O ar e a água são bastante prejudicados por poluição produzida por combustíveis e sistemas de dutos, no caso das águas, e por indústrias e transporte no caso do ar. Mesmo com a utilização de vários métodos de prevenção, que basicamente são estipulados por órgãos reguladores, esses métodos ainda são difíceis de serem aplicados em diversos países, mas suas atividades ainda possuem efeitos degradáveis no ecossistema terrestre. A idéia de produzir sistemas de comunicações moleculares para monitorar esse tipo de ambiente já fora explorada por alguns autores, ainda num âmbito bem superficial [9]. Entretanto, é imaginado que esses sistemas podem ajudar na coleta de dados em alta resolução, e também acesso em lugares de inóspitos.



Figura 2.13: Aplicações das nanorredes em poluição.

Em termos de alta resolução de coleta de dados, um sistema de redes de comunicações moleculares desenvolvido para monitorar a poluição de águas poderia identificar partes do ambiente que estão contaminados ou centros de iniciação de poluição, fazendo com que esses problemas sejam mais facilmente resolvidos. Do mesmo modo que esses nanodispositivos poderiam teoricamente acessar sistemas de dutos e tratamento de água, o que habilitaria a possibilidade de coleta de informação sobre a possível contaminação de água para consumo de uma sociedade. É imaginado que esses sistemas também podem aumentar a resolução na coleta de dados sobre o ar, onde uma distribuição em larga-escala de nanodispositivos capazes de sensoriar os componentes do ar em um ambiente, poderia

que quase em tempo real detectar possíveis contaminações por acidentes ou por ataques bioquímicos, ou seja, informações que poderia teoricamente salvar vidas.

Um dos grandes centros de uma sociedade bio-sustentável é o tratamento de lixo e esgoto. Um sistema de redes de comunicações moleculares poderia acessar níveis profundos em grandes quantidades de lixo, conseguindo obter informações em detalhes sobre níveis de toxicidade. Isso tem um impacto significativo no processamento de lixo, em que poderia-se identificar em quase tempo real se o lixo é reusável, ou até, se partes desse lixo poderia ser reusado sem qualquer risco de contaminação. O lixo contaminado poderia então ser rapidamente removido para que o restante não seja contaminado.

Embora esses sistemas ainda estejam sendo conceitualizados podem trazer um impacto tremendo no monitoramento e controle do ambiente, o que permite uma maior segurança do ambiente que vivemos para aumentar a longevidade de uma sociedade sustentável pela utilização de sistemas de redes de comunicações.

2.5.6. Indústria

A indústria 4.0 se inicia com o propósito de prover uma revolução industrial pela inclusão de sistemas computacionais de diversas dimensões na indústria, o que também depende de um robusto sistema de comunicação e informação. Uma grande parte dessa revolução claramente se baseia na Internet-das-coisas, pela relação da mesma com os grandes pilares da indústria 4.0: interoperabilidade, transparência e decisões descentralizadas. Essa nova indústria contará com um grande número de sensores e dispositivos que permitem com que a experiência da indústria possa se renovar com a possibilidade de conexão de humanos com ela. Essa nova experiência permitiria não só uma construção adaptativa de produtos mas bem como maior facilidade de produção de novos produtos e disponibilidade no mercado. Isso deve contar com um processo de transparência de alto nível entre o processo de produção de produtos bem como o interesse popular por processos industriais. Por último, os sistemas descentralizados de decisões podem compor uma larga rede de dispositivos que possam otimizar o sistema industrial por uma crescente demanda de mais organização automatizada.

Pouco ainda fora explorado como sistemas de redes de comunicações moleculares podem contribuir consideravelmente na indústria 4.0. Os autores em [74] comentaram a possibilidade de como essas nanorredes podem compor dispositivos robustos e como esse poderia ser utilizado para criar mais flexibilidade no processo de produção por formação de padrão e estruturas moleculares. É imaginado que moléculas poderiam ser controladas de uma maneira precisa para compor sistemas de produção de específicos tipos de conjuntos de moléculas. Isso poderia ajudar a indústria de materiais e química em seus produtos, ou qualquer indústria que dependa de composição de moléculas como meio de produção.

2.6. Ferramentas

Devido ao número considerável de propostas para canais de comunicação para nanorredes baseados em sistemas biológicos ou não, inúmeras ferramentas de simulações vêm sendo propostas nos últimos anos. Nesta seção, nós iremos apresentar uma discussão sobre essas ferramentas, comentando também sobre as vantagens e as desvantagens delas.

2.6.1. BiNS - Biological Nano-Scale communication simulator

BiNS é um simulador que apresenta uma abordagem “*plug-in-play*” para diferentes tipos de transmissores, canais e receptores para simulações baseadas em difusão [92]. Essa ferramenta segue uma programação orientada à objetos, com uma arquitetura centralizada baseada no conceito de nano-objetos para mais flexibilidade do simulador. Todos os elementos da rede possuem características em comum, mas podem ser estendidas por funções e características mais específicas. Como parte da validação da metodologia apresentada, os autores implementaram no BiNS o sistema imunológico do corpo humano. As moléculas portadoras de informação são as IL-4 citocinas, vindo de um portão de entrada fluidico de um transmissor no sangue que é o canal de difusão. A informação é amplificada por células linfócitos T, e recebidas por um linfócito B. Os resultados obtidos por esse ambiente de simulação foram validados por experimentos reais.

2.6.2. CalComSim - Calcium Signalling-Based Molecular Communication System Simulator

CalComSim é um simulador de comunicações moleculares baseado em sinalização de cálcio projetado para ambas as comunicações sintéticas e naturais encontradas dentro de um tecido humano 3D [7]. Uma integração dos modelos bio-físicos do processo de sinalização governado com o algoritmo estocástico de Gillespie é usado para simular reações múltiplas e paralelas em cada célula do tecido. Além disso, implementa-se três tipos diferentes de tecidos, incluindo: não excitáveis (tecidos que não podem propagar corrente elétrica), excitável (tecidos que pode propagar corrente elétrica) e híbrido (tecidos capazes de se comunicar com ambos tecidos excitáveis e não excitáveis). Cada tipo de células simuladas são as seguintes: epitelial (não-excitável), células musculares lisas (excitáveis) e astrócitos (híbrido). Os modelos biológicos que são incorporados no simulador são baseados em dados experimentais reais. O simulador também inclui o fechamento e abertura estocásticos das junções comunicantes entre as células. O simulador pode ser usado não apenas por engenheiros de telecomunicações, por biólogos cientistas, mas também por pesquisadores farmacêuticos que pode projetar novos medicamentos e tratamentos para doenças que surgem do prejudicado sinalização do cálcio. O CalComSim pode ser acessado em [93]

2.6.3. Neuron Simulator

O simulador NEURON é uma ferramenta desenvolvida pela Universidade de Yale que implementa sistemas neuronais como um todo e as comunicações entre neurônios. Esse simulador possui bastante popularidade na área de neurociência pela sua precisão em implementar modelos bio-físicos do processo de propagação de impulsos nas membranas celulares dos neurônios como também a propagação das sinapses [94]. Esses modelos bio-físicos podem ser adaptados facilmente para simular experimentos *in-vitro*, onde a partir de uma imagem da célula, a configuração morfológica dela pode ser obtida para compor um novo modelo dos compartimentos das células. Mais recentemente, esse simulador foi utilizado pelo *Blue Brain Project*, uma iniciativa européia de digitalização de microcolunas do cortex, que também pode ser acessada virtualmente e seus modelos podem ser baixados para uso de pesquisadores e afins. Como o NEURON possui uma extensão para a linguagem de programação Python, esses modelos podem ser reprogra-

mados com tudo o que essa linguagem pode oferecer, e abrir um leque de possibilidades de interação desse simulador com vários outros modelos e sistemas. O simulador pode ser acessado em [95].

2.6.4. NanoNS

O NanoNS é um simulador para nanorredes baseado no Network Simulator 2 (NS2) [96]. Os autores apresentam uma biblioteca baseada na linguagem utilizada pelo NS2 para interpretação dos seus scripts, o TCL. As nanorredes baseadas em difusão podem ser reconfiguradas a partir do canal de comunicação que é utilizado pela simulação, seja em espaço livre ou aquoso. Com a integração das nanorredes com o NS2 é possível simular a mobilidade dos nanonós da rede, e estender o conceito de nanorredes para incorporação de mobilidade e, conseqüentemente, as vantagens e desvantagens desses sistemas.

2.6.5. NCSim

O NCSim simula nanorredes biológicas com a transmissão de informação por bactéria flagelada [97]. As informações são codificadas em diferentes formatos pelo DNA dessas bactérias, que pelo processo de conjugação, transmitem essas informações para nós *relays* ou receptores que são outras bactérias do mesmo tipo. Os autores utilizam o tipo de bactéria E. Coli que possui uma característica estocástica de mobilidade, modelada por [98]. NCSim consiste em três módulos: (1) camada física de nanorredes bacterianas, incluindo mobilidade de bactérias e conjugação, técnicas de codificação/decodificação; (2) gerador de cenários e monitor de simulação; e (3) ferramenta de plotagem, destinada ao pós-processamento de dados e parcelas de simulação bruta geração. O módulo PHY, como o mais intensivo computacional, é implementado em C++. Os dois últimos módulos são escritos em Python para simplicidade de manutenção e extensão. O usuário interage com NCSim escrevendo pequenos scripts em Python para definição de cenários. Apesar do atrativo, esse simulador não se encontra disponível na Internet até o momento.

2.6.6. MolComML

Mais recentemente, e visando convergência desses simuladores propostos pela comunidade, os autores em [99] propuseram um modelo de Markup-Language para nano-redes e comunicações moleculares (MolComML) inspirados pelos projetos *Synthetic Biology Markup Language* ou *CellML*. Uma arquitetura de interpretação de definição de cenários por arquivos XML permite que tantos usuários com outros sistemas possam usar do MolComML como ferramenta de simulação. Essa metodologia permite também o uso de vários outros simuladores, como ferramentas de simulação, que apresentem heterogeneidade em termos de linguagem de programação e desempenho, para criação de complexos cenários de simulação. O MolComML também possui a possibilidade de fácil validação de resultados de simulação com resultados experimentais que são disponíveis pela ferramenta. Bem como o NCSim, essa ferramenta ainda não está disponível para download até o momento.

2.7. Considerações finais

O surgimento das nanorredes, embora recente, já permite que se imagine um grande impacto na sociedade. Os vários setores que variam da medicina até indústria permitem um

rápido desenvolvimento de ideias que se posicionam em grandes níveis de inovação. Mas ainda existem muitos desafios tanto no desenvolvimento teórico, quando prático dessas nanorredes. Por meio deste documento, é apresentado de maneira introdutória o que são essas nanorredes, que tipos de comunicações elas usam, e como as redes se comportam. Pelo que foi observado, o número de dispositivos que eventualmente fazem parte da Internet crescerá mais do que se tem previsto até hoje. Por isso não é somente necessário que protocolos de endereçamento sejam expandidos em termos de capacidade, como foi observado em mudanças recentes na Internet. Agora, com a incorporação dessas nanorredes e sua interface com sistemas biológicos, podemos imaginar que no futuro esses sistemas irão também estar conectados à grande rede. Abordamos então os esforços que a comunidade está tendo em direção a uma possível reformulação do modelo TCP/IP seguido na Internet. Tanto os modelos de camada única e de várias camadas, independente de complexidade, já sugerem que o TCP/IP terá mudanças de caráter vertical. Com isso, os desafios de pesquisa foram definidos nessa linha. Bem como também abordamos diversos tipos de aplicações que incluem: tecnologia da informação, medicina, neurociência, agricultura de precisão, poluição e indústria. Por último, discutimos que ferramentas de simulação estão disponíveis para os pesquisadores e alunos que desejam investigar melhor sobre as recentes descobertas realizadas pela comunidade das nanorredes.

A recém formada área de nanorredes encontra-se em uma crescente expansão da comunidade que permitirá o desenvolvimento completo dessa tecnologia e também as aplicações que são imaginadas para ela. Um pesquisador ou aluno interessado em ingressar nessa área pode fazer uma transição mais rápida do que pode parecer. Mesmo pelo fato dessa área exigir uma expansão do conhecimento do pesquisador para outras áreas. Esse processo pode ser feito a partir do refinado escopo que uma determinada pesquisa se encontra. Deste modo, pode-se em alguma outra área do conhecimento contribuir com grande impacto para o desenvolvimento das nanorredes.

Da mesma forma que a interdisciplinaridade é um desafio para pesquisador que deseja ingressar na área, ela também contribui bastante para o desenvolvimento técnico do pesquisador. O impacto que as nanorredes podem causar na sociedade ainda é pouco estimado pelo simples fato de que as nanomáquinas podem ser reprojatadas ou até reorganizadas para se adaptar em um ambiente. A comunidade já vislumbra mais aplicações além das que foram abordadas, como: militar, transporte e entretenimento. Tudo só poderá atingir um nível grande de transformação da sociedade se for alavancado por uma tecnologia revolucionária, como a Internet, e deste modo, o que irá expor ainda mais o poder das redes de comunicações como agente transformador social.

Referências

- [1] D. C. Richard Van Noorden, “World’s tiniest machines win chemistry nobel,” *Nature*, vol. 538, no. 7624, pp. 152–153, 2016.
- [2] M. Peplow, “The tiniest lego: a tale of nanoscale motors, rotors, switches and pumps,” *Nature*, vol. 525, no. 7567, pp. 18–21, 2015.
- [3] M. T. Barros, D. P. Martins, and M. S. de Alencar, “Comunicações moleculares: Um novo paradigma de comunicações para aplicações em nanomedicina,” *Revista*

de Tecnologia da Informação e Comunicação, vol. 7, no. 1, pp. 31–38, 2017.

- [4] J. W. Fredy, A. Méndez-Ardoy, S. Kwangmettatam, D. Boichichio, B. Matt, M. C. A. Stuart, J. Huskens, N. Katsonis, G. M. Pavan, and T. Kudernac, “Molecular photoswitches mediating the strain-driven disassembly of supramolecular tubules,” *Proceedings of the National Academy of Sciences*, vol. 114, no. 45, pp. 11 850–11 855, 2017.
- [5] R. Weiss, S. Basu, S. Hooshangi, A. Kalmbach, D. Karig, R. Mehreja, and I. Netravali, “Genetic circuit building blocks for cellular computation, communications, and signal processing,” *Natural Computing*, vol. 2, no. 1, pp. 47–84, 2003.
- [6] R. A. Freitas, *Nanomedicine, volume I: basic capabilities*. Landes Bioscience Georgetown, TX, 1999, vol. 1.
- [7] M. T. Barros, S. Balasubramaniam, and B. Jennings, “Comparative end-to-end analysis of ca 2+-signaling-based molecular communication in biological tissues,” *IEEE Transactions on Communications*, vol. 63, no. 12, pp. 5128–5142, 2015.
- [8] M. Pierobon and I. F. Akyildiz, “Noise analysis in ligand-binding reception for molecular communication in nanonetworks,” *IEEE Transactions on Signal Processing*, vol. 59, pp. 4168–4182, 2011.
- [9] I. F. Akyildiz, F. Brunetti, and C. Blazquez, “Nanonetworks: A new communication paradigm,” *Computer Networks*, vol. 52, pp. 2260–2279, 2008.
- [10] Q. H. Abbasi, K. Yang, N. Chopra, J. M. Jornet, N. A. Abuali, K. A. Qaraqe, and A. Alomainy, “Nano-communication for biomedical applications: A review on the state-of-the-art from physical layers to novel networking concepts,” *IEEE Access*, vol. 4, pp. 3920–3935, 2016.
- [11] B. Atakan, S. Balasubramaniam, and O. B. Akan, “Body area nanonetworks with molecular communications in nanomedicine,” *IEEE Communications Magazine*, vol. 50, pp. 28–34, 2012.
- [12] I. F. Akyildiz and J. M. Jornet, “Electromagnetic wireless nanosensor networks,” *Nano Communication Networks*, vol. 1, no. 1, pp. 3–19, 2010.
- [13] M. Pierobon, J. M. Jornet, N. Akkari, S. Almasri, and I. F. Akyildiz, “A routing framework for energy harvesting wireless nanosensor networks in the terahertz band,” *Wireless networks*, vol. 20, no. 5, pp. 1169–1183, 2014.
- [14] V. Loscri and A. M. Vegni, “An acoustic communication technique of nanorobot swarms for nanomedicine applications,” *IEEE transactions on nanobioscience*, vol. 14, no. 6, pp. 598–607, 2015.
- [15] G. E. Santagati and T. Melodia, “Opto-ultrasonic communications for wireless intra-body nanonetworks,” *Nano Communication Networks*, vol. 5, no. 1, pp. 3 – 14, 2014.

- [16] I. Akyildiz, M. Pierobon, S. Balasubramaniam, and Y. Koucheryavy, “The internet of bio-nano things,” *IEEE Communications Magazine*, vol. 53, no. 3, pp. 32–40, March 2015.
- [17] M. Pierobon and I. F. Akyildiz, “Capacity of a diffusion-based molecular communication system with channel memory and molecular noise,” *IEEE Transactions on Information Theory*, vol. 59, no. 2, pp. 942–954, 2013.
- [18] —, “A physical end-to-end model for molecular communication in nano-networks,” *IEEE Journal on Selected Areas in Communications*, vol. 28, pp. 602–611, 2010.
- [19] I. A. F. Fekri, R. Sivakumar, C. R. Forest, and B. K. Hammer, “Monaco: fundamentals of molecular nano-communication networks,” *IEEE Wireless Communications*, vol. 19, no. 5, pp. 12–18, 2012.
- [20] F. W. et al., “Artificial backbone neuronal network for nano scale sensors,” in *IEEE Conference on Computer Communications Workshops*, 2011, pp. 449–454.
- [21] F. Mesiti, M. Veletć, P. A. Floor, and I. Balasingham, “Astrocyte-neuron communication as cascade of equivalent circuits,” *Nano Communication Networks*, 2015.
- [22] S. Balasubramaniam and P. Lio’, “Multi-hop conjugation based bacteria nano-networks,” *IEEE Transactions on NanoBioscience*, vol. 12, no. 1, pp. 47–59, 2013.
- [23] A. S. Cacciapuoti, M. Caleffi, and A. Piras, “Neuronal communication: Presynaptic terminals as transmitter array,” in *Proceedings of the Second Annual International Conference on Nanoscale Computing and Communication*, ser. (NANOCOM), 2015, pp. 12:1–12:5.
- [24] F. Walsh and S. Balasubramaniam, “Reliability and delay analysis of multihop virus-based nanonetworks,” *IEEE Transactions on Nanotechnology*, vol. 12, no. 5, pp. 674–684, 2013.
- [25] I. F. Akyildiz and J. M. Jornet, “The internet of nano-things,” *IEEE Wireless Communications*, vol. 17, no. 6, 2010.
- [26] P. Baran, “On distributed communications series,” 1964. [Online]. Available: <http://www.rand.org/about/history/baran.list.html>
- [27] B. M. Leiner, V. G. Cerf, D. D. Clark, R. E. Kahn, L. Kleinrock, D. C. Lynch, J. Postel, L. G. Roberts, and S. Wolff, “A brief history of the internet,” *SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 5, pp. 22–31, Oct. 2009.
- [28] R. Moore, “Packet switching history.” [Online]. Available: <http://rogerdmoore.ca/PS/>
- [29] A. K. Bhushan and D. G. Frahmman, “Xerox network systems architecture,” in *Computer Network Architectures and Protocols*. Springer, 1989, pp. 417–447.

- [30] C. Malamud, *Analyzing DECnet/OSI Phase V*, ser. VNR computer library. Van Nostrand Reinhold, 1991. [Online]. Available: <https://books.google.com.br/books?id=fPJSAAAAMAAJ>
- [31] J. H. McFadyen, “Systems network architecture: An overview,” *IBM Systems Journal*, vol. 15, no. 1, pp. 4–23, 1976.
- [32] C. DiBona, S. Ockman, and M. Stone, Eds., *Open Sources: Voices from the Open Source Revolution*. Sebastopol, CA, USA: O’Reilly & Associates, Inc., 1999.
- [33] Q. H. Abbasi, K. Yang, N. Chopra, J. M. Jornet, N. A. Abuali, K. A. Qaraqe, and A. Alomainy, “Nano-communication for biomedical applications: A review on the state-of-the-art from physical layers to novel networking concepts,” *IEEE Access*, vol. 4, pp. 3920–3935, 2016.
- [34] D. Comer, *Interligação de Redes com TCP/IP-Vol. 1: Princípios, Protocolos e Arquitetura*. Elsevier Brasil, 2017, vol. 6.
- [35] N. A. Ali, W. Aleyadeh, and M. AbuElkhair, “Internet of nano-things network models and medical applications,” in *2016 International Wireless Communications and Mobile Computing Conference (IWCMC)*, 2016, pp. 211–215.
- [36] L. Felicetti, M. Femminella, G. Reali, T. Nakano, and A. V. Vasilakos, “Tcp-like molecular communications,” *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 12, pp. 2354–2367, 2014.
- [37] I. F. Akyildiz and J. M. Jornet, “The internet of nano-things,” *IEEE Wireless Communications*, vol. 17, no. 6, pp. 2260–2279, 2010.
- [38] B. Williams and T. Camp, “Comparison of broadcasting techniques for mobile ad hoc networks,” in *ACM international symposium on Mobile ad hoc networking & computing*. ACM, 2002, pp. 194–205.
- [39] Liaskos and Tsioliaridou, “A promise of realizable, ultra-scalable communications at nano-scale: a multi-modal nano-machine architecture,” *IEEE Transactions on Computers*, vol. 64, no. 5, pp. 1282–1295, May 2015.
- [40] G. Piro, L. A. Grieco, G. Boggia, and P. Camarda, “Nano-sim: simulating electromagnetic-based nanonetworks in the network simulator 3,” in *Proceedings of the 6th International ICST Conference on Simulation Tools and Techniques*. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2013, pp. 203–210.
- [41] B. A. Forouzan and S. C. Fegan, *Protocolo TCP/IP-3*. AMGH Editora, 2009.
- [42] F. Walsh, S. Balasubramaniam, D. Botvich, T. Suda, T. Nakano, S. F. Bush, and M. Ó. Foghlú, “Hybrid dna and enzyme based computing for address encoding, link switching and error correction in molecular communication,” in *International Conference on Nano-Networks*. Springer, 2008, pp. 28–38.

- [43] T. Nakano, K. Hosoda, Y. Nakamura, and K. Ishii, “A biologically-inspired intra-body nanonetwork: design considerations,” in *International Conference on Body Area Networks*. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2013, pp. 484–487.
- [44] T. Nakano, T. Suda, Y. Okaie, M. J. Moore, and A. V. Vasilakos, “Molecular communication among biological nanomachines: A layered architecture and research issues,” *IEEE transactions on nanobioscience*, vol. 13, no. 3, pp. 169–197, 2014.
- [45] A. T. Sharp, S. M. Raja, B. J. Wysocki, and T. A. Wysocki, “Layered communication protocol for macro to nano-scale communication systems,” in *IEEE International Conference on Communications (ICC)*. IEEE, 2010, pp. 1–6.
- [46] F. Ullah and I. Khan, “Bnmpps: Biomolecular nanomachine protocol stack for human disease diagnoses: A new paradigm,” *VAWKUM Transactions on Computer Sciences*, vol. 5, no. 2, pp. 8–18, 2015.
- [47] I. F. Akyildiz, F. Brunetti, and C. Blázquez, “Nanonetworks: A new communication paradigm,” *Computer Networks*, vol. 52, no. 12, pp. 2260–2279, 2008.
- [48] M. Moore, A. Enomoto, T. Nakano, R. Egashira, T. Suda, A. Kayasuga, H. Kojima, H. Sakakibara, and K. Oiwa, “A design of a molecular communication system for nanomachines using molecular motors,” in *Pervasive Computing and Communications Workshops, 2006. PerCom Workshops 2006. Fourth Annual IEEE International Conference on*. IEEE, 2006, pp. 6–20.
- [49] M. J. Berridge, “The am and fm of calcium signalling,” *Nature*, vol. 386, pp. 759–760, 1997.
- [50] Y. Moritani, S. Hiyama, S. Nomura, K.-n. Akiyoshi, and T. Suda, “A communication interface using vesicles embedded with channel forming proteins in molecular communication,” in *Bio-Inspired Models of Network, Information and Computing Systems*. IEEE, 2007, pp. 147–149.
- [51] T. Nakano, T. Suda, T. Koujin, T. Haraguchi, and Y. Hiraoka, “Molecular communication through gap junction channels: System design, experiments and modeling,” in *Bio-Inspired Models of Network, Information and Computing Systems(Bionetics)*. IEEE, 2007, pp. 139–146.
- [52] R. G. Cid-Fuentes, J. M. Jornet, I. F. Akyildiz, and E. Alarcón, “A receiver architecture for pulse-based electromagnetic nanonetworks in the terahertz band,” in *IEEE International Conference on Communications (ICC)*. IEEE, 2012, pp. 4937–4942.
- [53] I. Akyildiz, J. Jornet, and C. Han, “Teranets: Ultra-broadband communication networks in the terahertz band,” *IEEE Wireless Communications*, vol. 21, no. 4, pp. 130–135, 2014.
- [54] H. Shah-Mansouri, M. R. Pakravan, and B. H. Khalaj, “Analytical modeling and performance analysis of flooding in csma-based wireless networks,” *IEEE Transactions on Vehicular Technology*, vol. 60, no. 2, pp. 664–679, 2011.

- [55] J. M. Jornet and I. F. Akyildiz, “Low-weight channel coding for interference mitigation in electromagnetic nanonetworks in the terahertz band,” in *IEEE International Conference on Communications (ICC)*. IEEE, 2011, pp. 1–6.
- [56] S. R. Neupane, “Routing in resource constrained sensor nanonetworks,” Ph.D. dissertation, TAMPERE UNIVERSITY OF TECHNOLOGY.
- [57] S. D’Oro, L. Galluccio, G. Morabito, and S. Palazzo, “A timing channel-based mac protocol for energy-efficient nanonetworks,” *Nano Communication Networks*, vol. 6, no. 2, pp. 39–50, 2015.
- [58] S. Mohrehkesh and M. C. Weigle, “Rih-mac: Receiver-initiated harvesting-aware mac for nanonetworks,” in *ACM International Conference on Nanoscale Computing and Communication*. ACM, 2014, pp. 18:1–18:6.
- [59] F. Dressler and S. Fischer, “Connecting in-body nano communication with body area networks: Challenges and opportunities of the internet of nano things,” *Nano Communication Networks*, vol. 6, no. 2, pp. 29–38, 2015.
- [60] N. Agoulmine, K. Kim, S. Kim, T. Rim, J.-S. Lee, and M. Meyyappan, “Enabling communication and cooperation in bio-nanosensor networks: toward innovative healthcare solutions,” *IEEE Wireless Communications*, vol. 19, no. 5, pp. 42–51, 2012.
- [61] S. Balasubramaniam and J. Kangasharju, “Realizing the internet of nano things: challenges, solutions, and applications,” *Computer*, vol. 46, no. 2, pp. 62–68, 2013.
- [62] S. Oh, J. Kang, and M. Gruteser, “Location-based flooding techniques for vehicular emergency messaging,” in *Annual International Conference on Mobile and Ubiquitous Systems-Workshops, 2006. 3rd*. IEEE, 2006, pp. 1–9.
- [63] A. Tsioliariidou, C. Liaskos, S. Ioannidis, and A. Pitsillides, “Corona: A coordinate and routing system for nanonetworks,” in *Proceedings of the Second Annual International Conference on Nanoscale Computing and Communication*. ACM, 2015, pp. 1–18.
- [64] L. B. SOUSA, “Redes de computadores-guia total,” *São Paulo: Editora Érica Ltda*, 2009.
- [65] A. S. Tanenbaum and W. D. J., “Computer networks, /andrew s. tanenbaum, david j. wetherall,” *Cloth: Prentice Hall*, 2011.
- [66] N. A. Ali and M. Abu-Elkheir, “Internet of nano-things healthcare applications: Requirements, opportunities, and challenges,” in *IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. IEEE, 2015, pp. 9–14.
- [67] J. Suzuki, P. Boonma, and D. H. Phan, “Neuronal signaling optimization for intra-body nanonetworks,” in *International Conference on Digital Information and Communication Technology and it’s Applications (DICTAP)*. IEEE, 2014, pp. 69–74.

- [68] E. Dedu, J. Bourgeois, and M. A. Zainuddin, “A first study on video transmission over a nanowireless network,” in *International Conference on Nanoscale Computing and Communication*. ACM, 2014, pp. 18:1–18:6.
- [69] F. Walsh and S. Balasubramaniam, “Reliability and delay analysis of multihop virus-based nanonetworks,” *IEEE Transactions on Nanotechnology*, vol. 12, no. 5, pp. 674–684, 2013.
- [70] F. Walsh, S. Balasubramaniam, D. Botvich, W. Donnelly, and S. Sergeyev, “Development of molecular based communication protocols for nanomachines,” in *international conference on Nano-Networks*. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2007, pp. 19:1–19:5.
- [71] D. Zhao, “Ultraperformance wireless interconnect nanonetworks for heterogeneous gigascale multi-processor socs,” in *Workshop on Chip Multiprocessor*, pp. 1–3.
- [72] M. T. Barros, R. Mullins, and S. Balasubramaniam, “Integrated terahertz communication with reflectors for 5g small-cell networks,” *IEEE Transactions on Vehicular Technology*, vol. 66, no. 7, pp. 5647–5657, 2017.
- [73] I. F. Akyildiz and J. M. Jornet, “The internet of nano-things,” *IEEE Wireless Communications*, vol. 17, pp. 58–63, 2010.
- [74] T. Nakano, M. J. Moore, F. Wei, A. V. Vasilakos, and J. Shuai, “Molecular communication and networking: Opportunities and challenges,” *IEEE Transactions on NanoBioscience*, vol. 11, pp. 135–148, 2012.
- [75] M. T. Barros, “Ca 2+-signaling-based molecular communication systems: Design and future research directions,” *Nano Communication Networks*, 2017.
- [76] K. J. Roux, D. I. Kim, and B. Burke, “Bioid: a screen for protein-protein interactions,” *Current protocols in protein science*, pp. 19–23, 2013.
- [77] T. Nakano, T. Suda, Y. Okaie, M. J. Moore, and A. V. Vasilakos, “Molecular communication among biological nanomachines: A layered architecture and research issues,” *IEEE Transactions on NanoBioscience*, vol. 13, no. 3, pp. 169–197, Sept 2014.
- [78] T. Nakano, S. Kobayashi, T. Suda, Y. Okaie, Y. Hiraoka, and T. Haraguchi, “Externally controllable molecular communication,” *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 12, pp. 2417–2431, 2014.
- [79] Y. Chahibi, M. Pierobon, O. S. Sang, and I. F. Akyildiz, “A molecular communication system model for particulate drug delivery systems,” *IEEE Transactions on Biomedical Engineering*, vol. 60, no. 12, pp. 3468–3483, 2013.
- [80] U. A. Chude-Okonkwo, R. Malekian, B. T. Maharaj, and A. V. Vasilakos, “Molecular communication and nanonetwork for targeted drug delivery: A survey,” *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 3046–3096, 2017.

- [81] D. Martins, M. T. Barros, S. Balasubramaniam, M. Pierobon, M. Kandhavelu, and P. Lio, “Computational models for trapping ebola virus using engineered bacteria,” *To appear in IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2018.
- [82] D. P. Martins, M. T. Barros, and S. Balasubramaniam, “Using competing bacterial communication to disassemble biofilms,” in *ACM International Conference on Nanoscale Computing and Communication*. ACM, 2016, pp. 18:1–18:6.
- [83] S. Balasubramaniam, N. T. Boyle, A. Della-Chiesa, F. Walsh, A. Mardinoglu, D. Botvich, and A. Prina-Mello, “Development of artificial neuronal networks for molecular communication,” *Nano Communication Networks*, vol. 2, pp. 150–160, 2011.
- [84] A. S. Cacciapuoti, A. Piras, and M. Caleffi, “Modeling the dynamic processing of the presynaptic terminals for intrabody nanonetworks,” *IEEE Transactions on Communications*, vol. 64, no. 4, pp. 1636–1645, 2016.
- [85] M. Veletic, P. A. Floor, Y. Chahibi, and I. Balasingham, “On the Upper Bound of the Information Capacity in Neuronal Synapses,” *IEEE Transactions on Communications*, vol. PP, no. 99, pp. 1–12, 2016.
- [86] S. A. Wirdatmadja, M. T. Barros, Y. Koucheryavy, J. M. Jornet, and S. Balasubramaniam, “Wireless optogenetic nanonetworks for brain stimulation: Device model and charging protocols,” *IEEE Transactions on NanoBioscience*, 2017.
- [87] M. T. Barros and S. Dey, “Feed-forward and feedback control in astrocytes for ca^{2+} -based molecular communications nanonetworks,” *bioRxiv*, pp. 1–13, 2017.
- [88] —, “Set point regulation of astrocyte intracellular ca^{2+} signalling,” in *IEEE International Conference on Nanotechnology (IEEE-NANO)*, July 2017, pp. 315–320.
- [89] K. Bhargava, S. Ivanov, and W. Donnelly, “Internet of nano things for dairy farming,” in *Annual International Conference on Nanoscale Computing and Communication*. ACM, 2015, pp. 24:1–24:2.
- [90] A. Tzounis, N. Katsoulas, T. Bartzanas, and C. Kittas, “Internet of things in agriculture, recent advances and future challenges,” *Biosystems Engineering*, vol. 164, pp. 31–48, 2017.
- [91] A. Afsharinejad, A. Davy, B. Jennings, and C. Brennan, “Performance analysis of plant monitoring nanosensor networks at thz frequencies,” *IEEE Internet of Things Journal*, vol. 3, no. 1, pp. 59–69, 2016.
- [92] L. Felicetti, M. Femminella, and G. Reali, “A simulation tool for nanoscale biological networks,” *Nano Communication Networks*, vol. 3, no. 1, pp. 2–18, 2012.
- [93] (2018) Calcomsim: Calcium signaling molecular communications simulator. [Online]. Available: <http://calcomsim.droppages.com/>

- [94] M. L. Hines and N. T. Carnevale, “The neuron simulation environment,” *Neural computation*, vol. 9, no. 6, pp. 1179–1209, 1997.
- [95] (2018) The neuron simulator. [Online]. Available: <https://www.neuron.yale.edu/neuron/>
- [96] E. Gul, B. Atakan, and O. B. Akan, “Nanons: A nanoscale network simulator framework for molecular communications,” *Nano Communication Networks*, vol. 1, no. 2, pp. 138–156, 2010.
- [97] S. Balasubramaniam and P. Lio, “Multi-hop conjugation based bacteria nano-networks,” *IEEE Transactions on NanoBioscience*, vol. 12, no. 1, pp. 47–59, 2013.
- [98] Z. Wang, M. Kim, and G. Rosen, “Validating models of bacterial chemotaxis by simulating the random motility coefficient,” in *IEEE International Conference on BioInformatics and BioEngineering (BIBE)*. IEEE, 2008, pp. 1–5.
- [99] L. Felicetti, S. S. Assaf, M. Femminella, G. Reali, E. Alarcon, and J. Sole-Pareta, “The molecular communications markup language (molcomml),” *Nano Communication Networks*.

Capítulo

3

Aprendizado de Máquina em Plataformas de Processamento Distribuído de Fluxo: Análise e Detecção de Ameaças em Tempo Real

Martin Andreoni Lopez (UFRJ), Igor Jochem Sanz (UFRJ), Antonio G. P. Lobato (UFRJ), Diogo M. F. Mattos (UFF) e Otto Carlos M. B. Duarte (UFRJ)

Abstract

In this chapter, we focus on stream processing architectures for real-time threat detection in networks. We present and compare Apache Storm, Apache Spark Streaming, and Apache Flink. The most used feature selection techniques and machine learning algorithms for threat detection are described. We present the common datasets targeted for evaluating the performance of threat detection architectures. Finally, we evaluate proposals based on the lambda architecture, with previous or adaptive training, and zero-day threat detection. The CATRACA real-time threat detection tool, which uses the Apache Spark Streaming platform, is presented.

Resumo

Este capítulo foca em arquiteturas para processamento distribuído de fluxo para detecção de ameaças em redes em tempo real. O capítulo apresenta e compara as plataformas de código aberto para processamento distribuído de fluxo Apache Storm, Apache Spark Streaming e Apache Flink. As principais técnicas de seleção de características e algoritmos de aprendizado de máquina usados para classificar os tráfegos em benigno ou malicioso são descritos. Os conjuntos de dados de tráfego usados para a avaliação de desempenho das arquiteturas de detecção de ameaças são mostrados. Por fim, o capítulo avalia propostas baseadas na arquitetura lambda, com treinamento prévio e adaptativo e com detecção de ameaças inéditas (zero-day threats). A ferramenta CATRACA de detecção em tempo real de ameaças usando a plataforma Apache Spark Streaming é apresentada.

Este capítulo foi realizado com recursos do CNPq, CAPES, FAPERJ e FAPESP (2015/24514-9, 2015/24485-9 e 2014/50937-1).

3.1. Introdução

Os ataques a redes de computadores estão cada vez mais comuns e com volumes cada vez maiores. A exploração de novas vulnerabilidades (*zero-day attack*), ataques distribuídos de negação de serviço e uso de *softwares* maliciosos sofisticados têm aumentado. O impacto dos ataques de negação de serviço distribuídos (*Distributed Denial of Service - DDoS*) é cada vez maior, chegando a taxas de ataques da ordem de 1 Tb/s [Chandrasekar et al. 2017]. A popularização dos ataques é tão grande que é possível comprar uma hora de ataque de negação de serviço distribuído por U\$S 10 a hora na *deep web* [ARMOR 2018]. Os ataques deixaram de ser apenas motivados por fins econômicos e passaram a ter motivações políticas [Chandrasekar et al. 2017], a exemplo de ataques que visam manipular resultados de eleições, como a suspeita de manipulação russa sobre as eleições dos Estados Unidos em 2016 [ICA 2017].

Detectar ameaças em tempo real e reagir prontamente a ataques são fundamentais para diminuir os impactos de segurança [Andreoni Lopez et al. 2016a]. Atualmente, a detecção de ameaças de segurança pode levar semanas ou meses e espera-se reduzir esse tempo para minutos ou segundos. O cenário para o futuro é ainda mais adverso devido à introdução dos mais de 80 bilhões de dispositivos conectados, até 2025, na Internet das Coisas (IoT - *Internet of Things*) [Gluhak et al. 2011, Mattos et al. 2018]. Esses dispositivos produzem uma grande massa de dados (*Big Data*), que precisa ser gerenciada, processada, transferida e armazenada de forma segura e em tempo real. No entanto, as tecnologias atuais não foram projetadas para essa demanda [Cárdenas et al. 2013].

Os atuais sistemas de segurança, como os sistemas de Gerenciamento e Correlação de Eventos de Segurança (*Security Information and Event Management - SIEM*), não possuem um desempenho satisfatório, pois cerca de 85% das intrusões de redes são detectadas semanas depois de terem ocorrido [Clay 2015]. É essencial que o tempo de detecção seja o mínimo possível para que a prevenção da intrusão possa ser efetiva. A simples análise e a filtragem de pacotes pelos cabeçalhos IP e TCP não são eficientes, já que os atacantes procuram ocultar-se das ferramentas de segurança falsificando o IP de origem e alterando dinamicamente a porta TCP. Além disso, a tecnologia de virtualização, que é empregada em sistemas de computação em nuvem, introduz novas ameaças, gerando novos desafios para sistemas de segurança voltados para a proteção das nuvens [Sanz et al. 2017]. Nesse complexo cenário, uma alternativa promissora para classificar tráfego e detectar ameaças de forma automática baseia-se em técnicas de aprendizado de máquina.

O aprendizado de máquina fornece aos sistemas a capacidade de aprender e melhorar a partir da experiência, sem que sejam explicitamente programados. Baseados em técnicas de aprendizado de máquina, os sistemas são capazes de detectar e reagir automaticamente a ameaças [Lobato et al. 2017, Henke et al. 2011]. As técnicas de aprendizado de máquina se beneficiam da quantidade elevada de amostras para treinar, pois os métodos tendem a ter maior acurácia [Mayhew et al. 2015]. No entanto, para processar grandes quantidades de dados, os métodos tradicionais de aprendizado de máquina apresentam alta latência devido ao alto consumo de recursos computacionais, o que impede respostas em tempo real. A detecção de ameaças é um problema de processamento de uma quantidade massiva de dados (*Big Data*) e, portanto, sistemas que se baseiam no processamento centralizado para o treinamento dos modelos são inviáveis de serem aplicados.

A velocidade e a variabilidade dos dados da Internet são outros elementos complicadores para a detecção de ameaças. A identificação de ameaças em redes deve ser realizada em tempo real e, se possível, enquanto o fluxo da ameaça esteja ativo, para que contramedidas eficazes possam ser aplicadas, diminuindo assim os impactos econômicos e de privacidade. Além disso, devido aos diferentes perfis de uso da rede, as características do tráfego podem variar significativamente. Logo, é importante que as técnicas de aprendizado de máquina usadas sejam capazes de reagir às mudanças no tráfego, através da adaptação dos modelos de aprendizado ou através do retreinamento dos modelos, quando identificada uma mudança no padrão do tráfego [Lobato et al. 2018, Polikar et al. 2001]. Portanto, há a necessidade de se adotar novas ferramentas e plataformas para realizar o aprendizado de máquina sobre as grandes massas de dados coletados na rede, a uma rápida velocidade e levando em conta a grande variabilidade do tráfego da Internet.

A técnica de processamento distribuído de dados em fluxos realizada em aglomerados computacionais (*clusters*) é uma alternativa para processar com rapidez o grande volume de tráfego de redes que é potencialmente ilimitado [Gaber et al. 2005, Stonebraker et al. 2005]. Para enfrentar esse desafio, plataformas de processamento distribuído de fluxo de dados, de código aberto, como a Apache Storm [Zhao et al. 2015], a Apache Flink [Carbone et al. 2015a] e a Apache Spark Streaming [Franklin 2013] vêm sendo propostas. Essas plataformas diferem entre si no modelo de distribuição dos dados, na resiliência a falhas e no desempenho que alcançam ao serem submetidas a uma dada carga de trabalho [Andreoni Lopez et al. 2016c].

Este capítulo apresenta uma avaliação comparativa das três principais plataformas de processamento de fluxo de código aberto: a Apache Storm, a Apache Flink e a Apache Spark Streaming. O foco do capítulo são as arquiteturas de processamento distribuído de fluxo nas plataformas de código aberto mencionadas para a detecção de ameaças em tempo real. São apresentados os conjuntos de dados usados para avaliar o desempenho de algumas propostas apresentadas. Conceitos e técnicas usadas para redução de dimensionalidade e seleção de características são descritas. As principais métricas usadas na avaliação de desempenho de sistemas de aprendizado de máquina em detecção de intrusão são apresentadas. Arquiteturas de processamento distribuído de fluxos que seguem a arquitetura lambda, adaptativas e de processamento incremental são descritas e resultados são mostrados. Direções de pesquisas futuras são apresentadas. Por fim, a ferramenta CATRACA [Andreoni Lopez et al. 2017b] que realiza a detecção de anomalias em fluxos de rede é apresentada como uma parte prática de detecção de anomalias em fluxos de rede usando a plataforma Spark Streaming.

3.2. Plataformas de Processamento Distribuído de Fluxo

O processamento de fluxo (*stream processing*) possibilita extrair valores de dados em movimento, de forma similar à qual o processamento em lotes (*batch processing*) faz para dados estáticos. O objetivo do processamento de fluxo é possibilitar tomadas de decisões em tempo real, ou tempo quase real, ao prover capacidade de inspecionar, correlacionar e analisar os fluxos de dados à medida que os dados atravessam o sistema de processamento. Exemplos de cenários que requerem processamento de fluxo são: aplicações de monitoramento de tráfego para segurança de redes de computadores; aplicações de redes sociais, como o Twitter ou o Facebook; aplicações de análise financeira que mo-

nitoram fluxos de dados de ações reportadas em bolsas de valores; detecção de fraudes de cartão de crédito; controle de estoque; aplicações militares que monitoram leituras de sensores usados por soldados, como pressão arterial, frequência cardíaca e posição; processos de manufatura; gerenciamento energético; entre outras. Muitos cenários requerem capacidades de processamento de milhões ou centenas de milhões de eventos por segundo.

Os sistemas de gerenciamento de base de dados (*Database Management System - DBMS*) convencionais armazenam e indexam registros de dados antes de disponibilizá-los para a atividade de consulta, o que os torna inadequado para aplicações em tempo real ou respostas na ordem dos sub-segundos [Abadi et al. 2005]. As bases de dados estáticas não foram projetadas para o carregamento rápido e contínuo de dados. Logo, não suportam diretamente o processamento contínuo que é típico dos aplicativos de fluxo de dados. Além disso, se o processo não for estritamente estacionário como a maioria das aplicações do mundo real não o são, a saída poderia gradualmente mudar ao longo do tempo. Ameaças à segurança em redes TCP/IP, foco deste capítulo, são um típico exemplo de dados em movimento, no qual a saída muda ao longo do tempo.

Os fluxos de dados diferem do modelo convencional de dados armazenados [Gama and Rodrigues 2007] em: i) os elementos de dados no fluxo chegam em linha (*online*); ii) o sistema não tem controle sobre a ordem em que os elementos de dados chegam para serem processados; iii) os fluxos de dados são potencialmente ilimitados em tamanho; iv) uma vez que um elemento de um fluxo de dados foi processado, ele é descartado ou arquivado e não pode ser recuperado facilmente, a menos que seja armazenado explicitamente na memória, que normalmente é pequena em relação ao tamanho dos fluxos de dados. No processamento de fluxo, deve ser feito a computação antes de armazenar o dado, esquema conhecido como *compute-first, store-second*. A Tabela 3.1 resume as principais diferenças entre o processamento em lotes de dados estáticos e o processamento de fluxo de dados em movimento.

Tabela 3.1. Resumo de comparação de características entre o processamento em lotes e o processamento de fluxo.

	Em Lotes	De Fluxo
Núm. vezes que pode processar dado	Múltiplas vezes	Uma única vez
Tempo de processamento	Ilimitado	Restrito
Uso de memória	Ilimitado	Restrito
Tipo de resultado	Acurado	Aproximado
Topologia de processamento	Centraliz. / Distrib.	Distribuída
Tolerância a falhas	Alta	Moderada

O processamento de fluxo de dados é modelado através de um grafo acíclico dirigido (GAD) com nós fontes de dados, que continuamente emitem amostras, e nós de processamento interconectados. Um fluxo de dados ϕ é um conjunto infinito de dados, em que $\phi = \{D_t \mid t \geq 0\}$ e o ponto D_t é um dado representado por um conjunto de atributos e uma estampa de tempo. Formalmente, um ponto é descrito por $D_t = (\mathbf{V}, \tau_t)$, em que $\mathbf{V}$ é uma n -tupla na qual cada valor corresponde a um atributo e τ_t é a estampa de tempo para o t -ésimo dado. Os nós fontes emitem tuplas ou mensagens que são recebidas por

nós que realizam processamento, chamados de elementos de processamento (EP). Cada EP recebe dados em suas filas de entrada, executa algum processamento sobre os dados e produz uma saída para suas filas de saída.

Uma série de requisitos devem ser atendidos em plataformas de processamento distribuído de fluxo, Stonebraker *et al.* realçam os mais importantes [Stonebraker et al. 2005]. A capacidade de processar o dado em linha, sem a necessidade de armazená-lo para efetuar as operações, é fundamental para manter a baixa latência, uma vez que operações de armazenamento, como escrita e leitura em disco, acrescentam atrasos inaceitáveis no processamento. Além disso, é ideal que o sistema seja ativo, ou seja, que possua políticas próprias para operar sobre os dados sem depender de instruções externas. Devido ao grande volume, os dados devem ser separados em partições para tratá-los em paralelo. A alta disponibilidade e a recuperação de falhas também são críticas em sistemas de processamento de fluxo. Em aplicações de baixa latência, a recuperação deve ser rápida e eficiente, proporcionando garantias de processamento. Assim, as plataformas de processamento de fluxo devem fornecer mecanismos de resiliência contra imperfeições ou falhas, como atrasos, perda de dados ou amostras fora de ordem, que são comuns no processamento distribuído de fluxo em aglomerados computacionais (*clusters*). Além disso, os sistemas de processamento devem ter um mecanismo de execução altamente otimizado para fornecer resposta em tempo real para aplicações com altas taxas de dados. Portanto, a capacidade de processar milhões de mensagens por segundo com baixa latência, dentro de microssegundos, é essencial. Para alcançar esse desempenho, as plataformas devem minimizar a sobrecarga de comunicação entre os processos distribuídos.

As plataformas de processamento de fluxo vêm sendo pesquisadas desde os anos 90, apresentando uma evolução em três gerações. As plataformas de primeira geração foram baseadas nos sistemas de banco de dados que avaliam as regras expressas como pares condição-ação quando novos eventos chegam. Esses sistemas eram limitados em funcionalidades e também não eram dimensionados para fluxos com grandes volumes de dados. Exemplos de sistemas dessa geração incluem Starburst [Widom 1992], Postgres [Stonebraker and Kemnitz 1991] e NiagaraCQ [Chen et al. 2000]. A empresa Apama², fundada em 1999, foi a primeira empresa de aplicações de análise em tempo real orientada a eventos com foco em tomada de decisões de negócios. A tecnologia provida pela plataforma Apama permitia monitorar eventos, analisá-los e realizar ações em milissegundos.

Os sistemas de segunda geração focam em estender a linguagem de consulta estruturada (*Structured Query Language* - SQL) para processar fluxos, explorando as semelhanças entre um fluxo e uma consulta (*query*). Em maio de 2003, na Universidade de Stanford, foi criado o projeto STanford stREam datA Manager (STREAM) [Arasu et al. 2004]. O projeto STREAM é considerado um dos primeiros sistemas de gerenciamento de fluxo de dados de uso geral (*Data Stream Management System* - DSMS). O projeto STREAM impulsionou a fundação, ainda em 2003, da empresa Coral8³. Em 2007, Coral8 lança uma plataforma comercial, baseada em tecnologias da Microsoft, capaz de processar e analisar milhares de requisições SQL por segundo. O projeto Aurora [Carney

²A primeira empresa de análise de eventos em tempo real, Apama Stream Analytics, foi vendida em 2005 para a Progress Software Cooperation por 25 milhões de dólares. Acessado em abril de 2018. https://www.softwareag.com/corporate/products/apama_webmethods/analytics/default.html

³Vendida para a empresa Aleri em 2009.

et al. 2002] foi lançado em 2002 em uma colaboração com a Universidade Brandeis, a Universidade de Brown e o MIT. A principal desvantagem do Aurora é que o projeto foi concebido como um mecanismo único e centralizado de processamento de fluxo. Uma nova versão distribuída foi lançada em 2003, chamada Aurora*. Uma última versão foi lançada oficialmente com o nome de Borealis [Abadi et al. 2005], com algumas outras melhorias, como a tolerância a falhas. O projeto Medusa [Balazinska et al. 2004] usou a distribuição do Borealis para criar um sistema de processamento de fluxo federado. Borealis e Medusa ficaram obsoletos em 2008. Os projetos Aurora/Borealis impulsionaram em 2003 a fundação da empresa StreamBase System⁴, que lançou comercialmente a plataforma StreamBase para processamento de eventos complexos (*Complex Event Processing* - CEP) com propósito de tomadas de decisões. A universidade de Berkeley, em 2003, cria uma linguagem para executar continuamente consultas SQL com base no sistema de banco de dados Postgres chamada TelegraphCQ [Chandrasekaran et al. 2003]. Com base no TelegraphCQ, a empresa da Truviso⁵ foi criada em 2009, e em 2012 a Truviso foi adquirida pela Cisco. Em 2006, a Universidade de Cornell criou Cayuga [Demers et al. 2007], um sistema de publicação-assinatura (*publish/subscribe*) com estados, que desenvolveu uma linguagem simples de consulta para realizar um processamento que escala tanto com a taxa de chegada de eventos quanto com o número de consultas. Cayuga foi substituído por Cougar⁶ e é ainda uma pesquisa ativa.

Os sistemas de terceira geração surgiram para atender a necessidade das empresas associadas à Internet de processar grandes volumes de dados produzidos à alta velocidade. O foco principal passa a ser o processamento distribuído escalável de fluxos de dados em aglomerados computacionais. A Google revoluciona o processamento distribuído propondo o modelo de programação MapReduce [Jiang et al. 2010] para o processamento paralelo escalável de grandes volumes de dados em aglomerados. A ideia chave de espalhar-processar-combinar é usada para realizar de forma escalável diferentes tarefas em paralelo em servidores de um aglomerado computacionais. A plataforma Hadoop [Kala Karun and Chitharanjan 2013] é a implementação de código aberto de MapReduce para realizar análíticas em grandes massas de dados. No entanto, devido à alta latência que o MapReduce produz, alguns projetos foram propostos para realizar análíticas de fluxo de dados em tempo real. O projeto Spark substitui o MapReduce do Hadoop para executar em memória operações que o Hadoop executa sobre dados recuperados do disco. As plataformas de código aberto Storm e Flink são propostas para processamento de fluxo. A plataforma Spark propõe uma extensão para processamento de fluxo em microlotes, o Spark Streaming. Em seguida, são descritas as plataformas escaláveis de terceira geração de código aberto Apache Hadoop, Apache Storm, Apache Flink e Apache Spark Streaming. Hadoop é apresentado por fins didáticos.

3.2.1. A Plataforma Apache Hadoop⁷

O Hadoop permite o processamento distribuído de grandes conjuntos de dados através de um aglomerado de computadores usando um modelo simples de programação.

⁴Vendida para a empresa TIBCO Software em 2013.

⁵Truviso Analytic <http://jtonedm.com/2009/03/03/first-look-truviso/> acessado em abril de 2018.

⁶Cougar processing <http://www.cs.cornell.edu/database/cougar/> acessado em abril de 2018.

⁷Esta subseção foi baseada em [Costa et al. 2012].

O Hadoop é projetado com o objetivo de escalar de um único até milhares de servidores, cada um com processamento e memória local [Costa et al. 2012]. Além da escalabilidade, o Hadoop provê uma robusta tolerância a falhas que é obtida por *software* projetado para detectar e tratar falhas, o que o torna apropriado para aglomerados. O Hadoop é composto por duas partes principais:

- o sistema Hadoop de arquivos distribuídos (*Hadoop Distributed File System* - HDFS), que é um sistema de arquivos para dividir, espalhar, replicar e gerenciar dados ao longo dos nós em um aglomerado;
- o MapReduce, que é um mecanismo computacional para executar aplicações em paralelo. As aplicações são executadas através da divisão em tarefas que manipulam apenas uma parcela dos dados, coletando e redistribuindo resultados intermediários e gerenciando falhas através de todos os nós do aglomerado.

No HDFS um arquivo consiste em blocos com tamanhos iguais e múltiplos dos tamanhos dos blocos de armazenamento. Normalmente os blocos de arquivo são de 64 MB, enquanto os blocos de armazenamento são de 512 kB. O HDFS usa o bloco de arquivos como a unidade a ser empregada para distribuir partes de arquivo entre os discos rígidos dos nós. Como os núcleos de processadores e os discos em um nó e também os nós em um bastidor (*rack*) podem falhar, um bloco de arquivo é armazenado em múltiplos nós do o aglomerado, conforme mostrado na Figura 3.1. O número de cópias pode ser configurado, mas por padrão, é tipicamente igual a três. O sistema de arquivos do Hadoop é classificado como “distribuído” porque ele gerencia o armazenamento para todas as máquinas da rede e os arquivos são distribuídos entre diversos nós. O Hadoop trata todos os nós como nós de dados, o que significa que eles podem armazenar dados. Entretanto, ele elege ao menos um nó para ser o “Name Node”. O Name Node decide em qual disco rígido cada uma das cópias de cada um dos blocos de arquivo é armazenada. Além disso, o Name Node mantém todas as informações em tabelas armazenadas localmente em seus discos. Quando um nó falha, o Name Node identifica todos os blocos de arquivo que foram afetados; recupera as cópias desses blocos de arquivo de nós operacionais (sem falha); encontra novos nós para armazenar cópias dos dados afetados; armazena essas cópias no nó escolhido e atualiza a informação em sua tabela. Quando uma aplicação precisa ler um arquivo, ele primeiro se conecta ao Name Node para obter o endereço dos blocos do disco onde os blocos do arquivo estão armazenados. Assim, em seguida, a aplicação pode ler esses blocos diretamente sem outra intervenção do Name Node. Um dos maiores problemas apontados do HDFS é o fato do Name Node poder se tornar um ponto único de falha. Logo, se o nó com o Name Node falhar, todas as informações de mapeamento entre nomes de arquivos e endereços de seus respectivos blocos de arquivo podem ser perdidos. Então, um novo nó precisa ser designado como o Name Node com o mesmo endereço IP do anterior que falhou. Para abordar tal questão, o Hadoop salva cópias das tabelas criadas pelo Name Node em outros nós do aglomerado computacional.

O segundo mecanismo fundamental do Hadoop é o MapReduce. Como o próprio nome sugere, o MapReduce enxerga uma tarefa computacional como consistindo de duas fases, a fase de mapeamento (Map) e a fase de redução (Reduce), que são executadas nessa mesma sequência. Durante a fase de mapeamento, todos os nós desempenham a

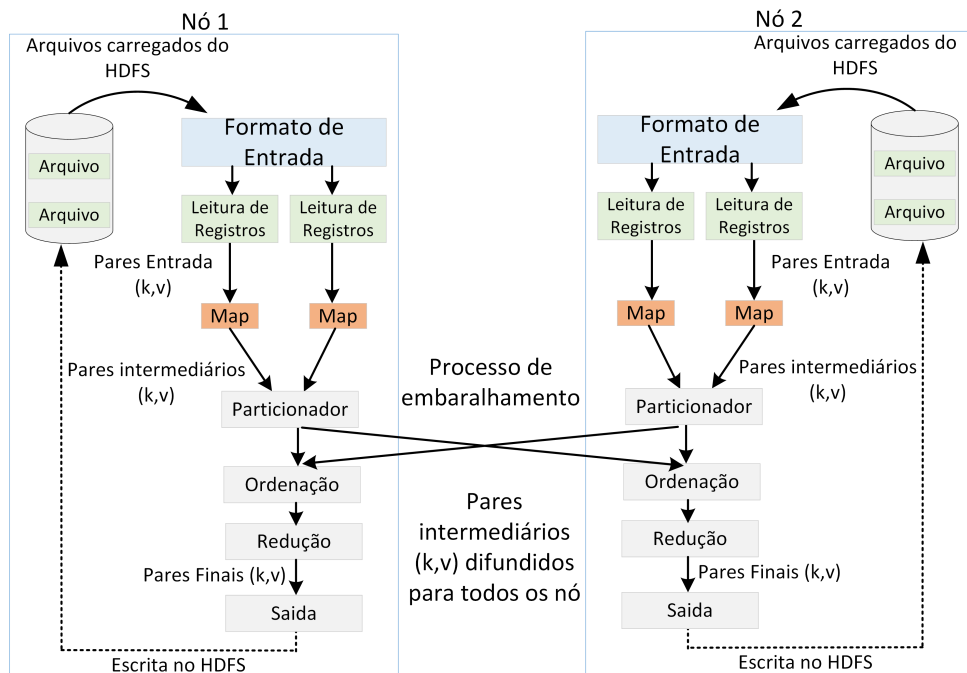


Figura 3.1. Funcionamento do MapReduce. Primeiramente os arquivos são carregados do HDFS. Logo o formato de entrada define como esses arquivos são divididos e lidos. Os arquivos são divididos em blocos e são criados os pares $\langle \text{chave}, \text{valor} \rangle$ (k,v). Os pares $\langle \text{chave}, \text{valor} \rangle$ são obtidos localmente, para logo ser intercambiados entre os nós do sistema pela função de embaralhamento. Uma vez que os pares são recebidos, eles são ordenados e reduzidos para obter o par $\langle \text{chave}, \text{valor} \rangle$ final que será armazenado no HDFS.

mesma tarefa computacional a partir de um subconjunto dos dados que está localizado no próprio nó ou próximo dele. Em outras palavras, o MapReduce usa o princípio da localidade dos dados para aumentar o seu desempenho e para minimizar a movimentação dos dados pelo aglomerado. É importante notar que devido a todos os blocos de arquivos no sistema de arquivos distribuídos do Hadoop terem o mesmo tamanho, a computação na fase de mapeamento pode ser igualmente dividida entre os nós. Se os blocos de arquivo não tivessem o mesmo tamanho, o tempo de processamento seria predominantemente ditado pelo tempo necessário para processar o bloco de arquivo mais longo, enquanto os outros nós permaneceriam ociosos. A saída da fase de redução consiste em uma lista de pares $\langle \text{chave}, \text{valor} \rangle$.

Após a execução da função de mapeamento, cada nó produz uma lista de pares chave-valor, na qual cada chave é o identificador da parcela dado a ser processado e o valor é o resultado da computação sobre a parcela de dados identificada pelo valor. Em um exemplo como a execução de uma consulta sobre uma base dados, mostrado na Figura 3.2, a chave é a identificação da consulta e o valor é o resultado da consulta. Pode-se, então, utilizar a fase de redução para consolidação dos resultados obtidos por cada nó, através, por exemplo, da soma de valores identificados pelas mesmas chaves, para “reduzir” a saída das funções de mapeamento a uma única lista de pares chave-valor. No MapReduce, um nó é selecionado para executar a função de redução. Todos os outros nós precisam enviar a lista $\langle \text{chave}, \text{valor} \rangle$ criada pela própria função de mapeamento ao nó designado. O primeiro procedimento executado agrupa os pares $\langle \text{chave},$

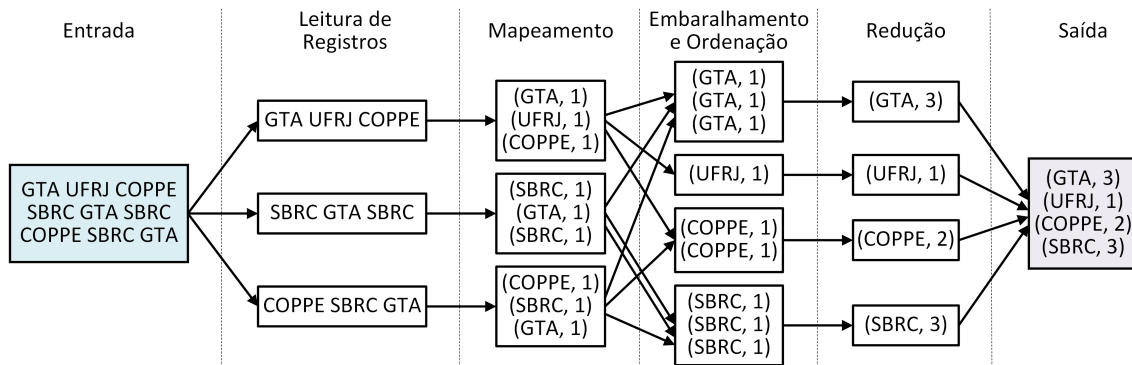


Figura 3.2. Exemplo do MapReduce. As funções principais são a entrada de dados, a leitura de registros e divisão em blocos, o mapeamento em <chave, valor>, o embaralhamento e a ordenação dos pares, a redução e finalmente a saída.

`valor`> com a mesma chave de forma ordenada. Em seguida, o procedimento executado é o procedimento de combinação que agrupa os pares <chave, valor> com a mesma chave em uma única entrada. O agrupamento é realizado de forma que cada entrada seja composta de uma chave e uma lista com todos os valores relacionados a mesma chave no procedimento anterior. Finalmente, o procedimento de redução soma os valores associados a cada chave existente do par <chave, valor>. No arcabouço MapReduce, ambas operações de mapeamento e redução são consideradas rotinas, que combinadas formam uma tarefa.

3.2.2. A plataforma Apache Storm

O Apache Storm⁸ [Toshniwal et al. 2014] é uma plataforma de código aberto para processamento distribuído de fluxo escrito nas linguagens Java e Clojure. Nesta plataforma, um fluxo de dados consiste em um identificador e um conjunto ilimitado de tuplas n-dimensionais. As aplicações são abstraídas em topologias, análogas a um trabalho (*job*) de MapReduce no Hadoop, que consistem em grafos acíclicos dirigidos nos quais as arestas representam fluxos e os vértices representam os nós de entrada (*spouts*) ou nós de processamento (*bolts*) de dados. Os *spouts* são responsáveis pela abstração dos dados de entrada em tuplas que fluem pelo grafo. Os *bolts* são elementos processadores que executam uma operação atômica definida pelo usuário à medida que o fluxo de dados avança no grafo. Tanto *bolts* como *spouts* são paralelizáveis e podem ser definidos com um grau de paralelismo que indica a quantidade de tarefas concorrentes presentes em cada nó. Um exemplo de topologia com dois *spouts* e três *bolts* é mostrado na Figura 3.3(a).

Outro conceito importante no Storm é o agrupamento de fluxos (*stream grouping*), que define como o fluxo é distribuído entre as tarefas de um *bolt*. Assim, a decisão sobre o tipo de agrupamento permite que o usuário defina como os dados devem fluir na topologia. O Storm possui oito tipos de agrupamento de fluxos que representam diferentes maneiras de enviar tuplas ao *bolt* seguinte, dos quais se destacam: i) por embaralhamento (*shuffle*),

⁸Nathan Marz, doutor pela Universidade de Stanford, trabalhando na empresa BackType, desenvolve o Storm em 2011, um arcabouço para processamento distribuído de fluxo, para tratar em tempo real a grande quantidade de mensagens (*tweets*) recebidas pela empresa Twitter. A empresa BackType é adquirida pelo Twitter e o Storm torna-se de código aberto, migrando para a Fundação Apache em 2013.

no qual cada tarefa em um *bolt* recebe o mesmo número de tuplas, distribuídas de maneira aleatória e uniforme; ii) por campos (*fields*), no qual as tuplas que possuem valores iguais para um conjunto de campos são enviadas à mesma tarefa de um *bolt*; iii) por difusão (*all*), na qual todas as tuplas são enviadas a todas as tarefas do *bolt* seguinte; e iv) global, na qual todo o fluxo é enviado à tarefa de menor identificador. Cada envio de fluxo, isto é, cada aresta na topologia, deve possuir um tipo de agrupamento decidido pelo usuário, que deve levar em conta as características da sua aplicação.

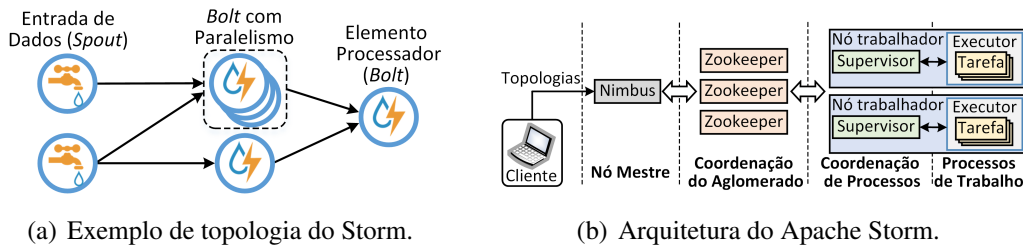


Figura 3.3. Apache Storm. a) uma topologia do Storm com elementos *Spouts*, nós de entrada, e *Bolts*, nós que realizam um processamento nas tuplas; b) a arquitetura do Apache Storm. O Nimbus controla as topologias, se comunicando com os supervisores que coordenam os processos nos trabalhadores. O controle do estado da topologia é feito pelo *Zookeeper*.

A Figura 3.3(b) mostra a coordenação dos processos do Storm em um aglomerado computacional. Um usuário envia suas topologias ao Nimbus, o nó mestre da arquitetura, que coordena a implementação das tarefas de *bolts* e *spouts* em nós trabalhadores através do Apache Zookeeper. O Zookeeper é responsável pelo gerenciamento dos nós trabalhadores e pelo armazenamento estado de todos os elementos do sistema. Em cada nó trabalhador, um supervisor monitora os executores, que são processos responsáveis por executar uma ou mais tarefas. Os supervisores informam o estado e a disponibilidade dos executores através de um mecanismo de sinais periódicos *heartbeat*, permitindo que o Nimbus identifique falhas no sistema. Falhas de executores são tratadas pelos próprios supervisores, que reiniciam os processos correspondentes no nó trabalhador. Uma falha de supervisor é tratada pelo Nimbus, que pode realocar todas as tarefas do supervisor em falha para outro nó trabalhador. Se o Nimbus falha, o sistema ainda é capaz de executar todas as topologias pendentes, porém o usuário não é capaz de submeter novas topologias. Após recuperação, o Nimbus e os supervisores podem retomar o último estado armazenado no Zookeeper.

O Apache Storm usa mecanismos de armazenamento e reconhecimento (ACK) para garantir o processamento das tuplas mesmo após uma falha. Para isto, todas as tuplas são identificadas pelos spouts e seus identificadores são enviadas a um *bolt* especial, denominado *bolt* de reconhecimento (*acker bolt*), que armazena o estado de cada tupla. Um exemplo de topologia com *acker bolt* é mostrado na Figura 3.4. A cada tupla processada, um *bolt* deve enviar um reconhecimento positivo (ACK) para o *acker bolt*. Caso todas as tuplas recebam um ACK para cada *bolt*, o *bolt* de reconhecimento descarta os IDs e informa ao *spout* que o processamento foi realizado com sucesso. Caso contrário, o *bolt* de reconhecimento solicita ao *spout* o reenvio de todas as tuplas e o sistema retrocede até o ponto de falha. O não-recebimento de um ACK é reconhecido pela expiração de um

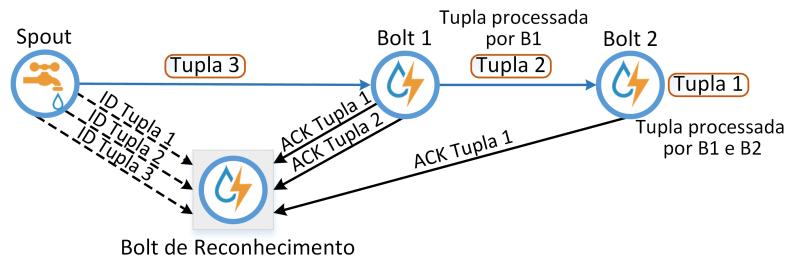


Figura 3.4. Semântica de entrega de mensagens “pelo menos uma vez” (*at least once*) utilizada no Apache Storm. Cada vez que uma tupla é emitida pelo *spout*, um registro é emitido ao *bolt* de reconhecimento. Logo quando a tupla é processada por um *bolt* um ACK é emitido reconhecendo o processamento.

temporizador (*timeout*) definido no *bolt* de reconhecimento. Isto garante a semântica de entrega de mensagens “pelo menos uma vez” (*at least once*), em que cada tupla seja processada uma ou várias vezes, no caso de reenvio. Ainda é possível desativar o *acker bolt* para aplicações que não necessitem de garantias de processamento.

3.2.3. A Plataforma Apache Flink

O Apache Flink⁹ [Carbone et al. 2015a] é uma plataforma de processamento híbrido, que suporta processamento de fluxo e em lotes. O núcleo do Flink é o processamento de fluxo, tornando o processamento em lotes um caso especial. As tarefas de análise do Flink são abstraídas em grafos acíclico dirigido (GAD) formado por quatro componentes: fontes; operadores; torneiras de saída; e registros que percorrem o grafo. A abstração da topologia é realizada através da programação em Java ou Scala. Assim como no Storm, a divisão de trabalho é baseada em um modelo mestre-trabalhador. A Figura 3.5(a) mostra a arquitetura do Apache Flink. O nó mestre do Flink é denominado gerente de trabalho e interage com aplicações de clientes com responsabilidades semelhantes ao nó mestre do Storm (Nimbus). O gerente de trabalho recebe aplicações de clientes, organiza as tarefas e as envia para os nós trabalhadores, que recebem o nome de gerente de tarefas. Além disso, o gerente de trabalho mantém o estado de todas as execuções e de cada trabalhador. Os estados dos trabalhadores são informados através de um mecanismo de sinais periódicos (*heartbeat*). O gerente de tarefas tem uma função semelhante ao nó trabalhador no Storm. Os gerentes de tarefas executam tarefas atribuídas pelo gerente de trabalho e trocam informações com outros gerentes de tarefas quando necessário. Cada gerente de tarefas disponibiliza *slots* de processamento para o aglomerado computacional, que são utilizados para executar tarefas em paralelo.

A abstração do fluxo de dados no Flink é chamada *DataStream* e é definida como uma sequência de registros parcialmente ordenados. Parcialmente, pois não há garantia de ordem caso um elemento operador receba mais de um fluxo de dados como entrada. *DataStreams* são semelhantes às tuplas do Storm e recebem dados de fluxo de fontes externas, como filas de mensagens, *sockets* e outros. A programação de *DataStream* oferece suporte a várias funções nativas para operar fluxos de dados, como *map*, *filtering*, *reduc-*

⁹Flink nasceu em 2010 de um projeto de pesquisa Europeu denominado “Stratosphere: Gestão da Informação na Nuvem” desenvolvido em colaboração da Universidade Técnica de Berlim, Humboldt-Universität zu Berlin e Hasso-Plattner-Institut Potsdam. Em 2014, o Stratosphere muda o nome do projeto para Flink e abre o seu código na Fundação Apache.

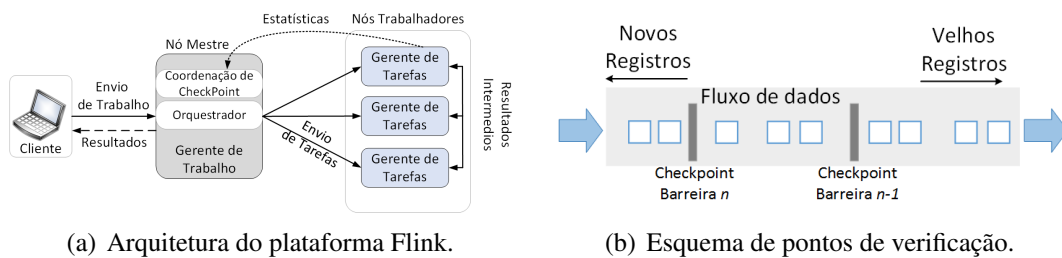


Figura 3.5. A arquitetura da plataforma Apache Flink: a) o gerente de trabalho recebe os trabalhos desde o cliente, divide o trabalho em tarefas e envia para os trabalhadores que comunicam as estatísticas e os resultados; b) as barreiras são injetadas nos elementos de origem e passam pelo grafo junto com as amostras. Quando o fluxo passa pelos operadores capturas instantâneas do estado são disparadas. Quando um operador recebe uma barreira de todos os fluxos de entrada, ele verifica seu estado para o armazenamento.

tion, join etc., que são aplicadas de forma incremental a cada entrada, gerando um novo *DataStream*. Cada uma destas operações pode ser paralelizada configurando-se um parâmetro de paralelismo. Assim, as instâncias paralelas das operações são atribuídas aos *slots* de processamento disponíveis dos gerentes de tarefas para tratar partições do *DataStream* simultaneamente. Este método permite a execução distribuída de operações nos fluxos de dados.

O Flink possui uma semântica de entrega confiável de mensagens do tipo “exatamente uma”. Esta semântica se baseia no esquema de tolerância a falha com pontos de verificação, ou barreiras (*checkpoints barriers*), para que o sistema possa retornar em caso de falha. As barreiras funcionam como registros de controle e são injetadas regularmente no fluxo de dados pelos elementos fontes para fluírem pelo grafo juntamente com os registros de dados. A Figura 3.5(b) mostra a implementação do esquema de pontos de verificação por barreiras. Quando uma barreira atravessa um elemento operador, este realiza uma captura momentânea do estado do sistema (*snapshot*). A captura consiste em armazenar o estado do operador, por exemplo, o conteúdo de uma janela deslizante ou uma estrutura customizada de dados, e a sua posição no fluxo de dados. Após uma fase de alinhamento entre operadores para certificar de que a barreira atravessou todos os operadores que recebem aquele fluxo, os operadores escrevem a captura gerada em um sistema de arquivos duráveis, como o HDFS. Em caso de falha de *software*, nó ou rede, o Flink interrompe o *DataStream*. Em seguida, o sistema reinicia imediatamente os operadores e recomeça a partir do última captura bem-sucedida armazenada. Assim como no Storm, tolerância a falhas do Flink é garantida partindo da premissa que o sistema é precedido por um sistema de mensagens com reenvio persistente, como o Apache Kafka. No caso especial de processamento em lotes, não há esquema de tolerância a falhas, e caso ocorra uma falha, toda a operação deve ser recomeçada do zero.

3.2.4. A Plataforma Apache Spark

Spark¹⁰ [Franklin 2013] é uma plataforma para processamento distribuído de dados, escrita em Java e Scala. O Spark é uma implementação do MapReduce do Hadoop,

¹⁰A plataforma Spark foi desenvolvida, em 2012, pelo AmpLab da Universidade de Berkeley. Em 2011, a Universidade de Berkeley recebeu 30 milhões de dólares para criar o AmpLab e financiar pesquisas análise de dados em tempo real. O projeto Spark foi doado em 2013 para a Fundação Apache.

que processa as tarefas em memória, evitando a escrita e a leitura em disco de resultados intermediários. A principal diferença entre o Spark e o Hadoop é que todo o envio de dados entre os nós do Hadoop é realizado através de escritas e leituras no sistema de arquivo distribuído (HDFS), enquanto no Spark, os envios de dados entre processos são realizados por comunicação entre processos que mantêm os dados em memória. Assim, o Spark não introduz a necessidade de realização de repetidas escritas e leituras em disco, operações reconhecidamente lentas. O processamento em memória torna o Spark mais rápido que o MapReduce, obtendo muito sucesso nas aplicações que realizam diversas rodadas de transformações sobre grandes conjuntos de dados, como as de aprendizado de máquina. Vale ressaltar que o Spark, assim como o Hadoop, realiza processamento em lotes e, portanto, não é apropriado para análise em fluxos em tempo real. Por sua vez, a extensão Spark Streaming [Zaharia et al. 2013], baseada em microlotes, foi desenvolvida em 2014 como uma biblioteca executando no topo do Spark Engine para realizar análise em tempo real.

A biblioteca Spark Streaming discretiza um fluxo de dados em lotes pequenos de dados, chamados microlotes. Cada microlote é um conjunto de tuplas de dados armazenado em memória. O lote é a unidade de dados tratada pelo Spark Engine. A abstração do fluxo é chamada de fluxo discreto (*Discrete Stream* - D-Stream) e é definida como um conjunto de tarefas curtas, sem estado e discretizadas. No Spark, como mostra a Figura 3.6(b), o processamento de fluxos é tratado como uma série de processamento em lotes discretos em intervalos de tempo menores, os microlotes. Semelhante ao MapReduce, um trabalho no Spark é definido como uma computação paralela que consiste em várias tarefas. A tarefa é uma unidade de trabalho que é enviada para o Gerenciador de Tarefas. Quando um fluxo chega ao Spark, os dados são discretizados em microlotes que são distribuídos de forma a garantir a resiliência e, portanto, os microlotes são implementados por uma estrutura de dados chamada de Conjunto de Dados Distribuídos Resilientes (*Resilient Distributed Dataset* - RDD). O RDD é a principal estrutura de dados do Spark Engine que armazena os dados em memória e os distribui garantindo a resiliência. É importante ressaltar que há a necessidade de se garantir a resiliência na distribuição dos dados em fluxo, pois os dados são sempre armazenados em memória volátil, ao contrário do que ocorre com o Hadoop em que os dados são sempre armazenados em disco, memória não volátil. Um D-Stream é uma sequência potencialmente infinita de conjuntos de dados distribuídos resilientes (RDD). Em seguida, o Spark Engine é executado gerando trabalhos para processar os microlotes representados pela estrutura de dados RDD.

As operações realizadas com Spark Streaming são realizadas sobre D-Streams. As operações do Spark Engine ocorrem sobre os RDDs que estão contidos em um D-Stream. Assim, ao operar com Spark Streaming são realizados dois níveis de distribuição de dados. O primeiro nível itera e distribui os RDDs contidos em uma fração do D-Stream entre diversos nós, semelhante a uma operação de mapeamento (*map*), em que cada nó recebe uma partição do conjunto de RDDs. No segundo nível, opera-se sobre os RDDs, em que os nós realizam o mapeamento e a redução das tuplas de dados contidas em cada RDD. As operações de ações e transformações sobre os dados ocorrem através do mapeamento e redução sobre os RDDs.

A Figura 3.6(a) mostra a configuração de um aglomerado computacional Spark. Os trabalhos no Spark são executados como processos independentes no aglomerado,

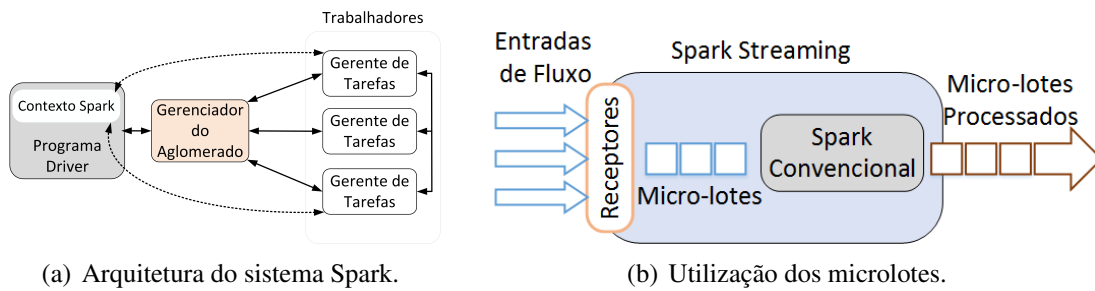


Figura 3.6. Distribuição de dados e tarefas no Spark. a) arquitetura do aglomerado do sistema Spark Streaming: o programa de driver, o gerenciador do aglomerado e os nós de trabalho, que executam os gerenciadores de tarefas; b) processamento de microlote usado no Spark Streaming, no qual os fluxos de entrada são recebidos pelos receptores e são transformados em microlotes que são executados em um mecanismo tradicional Map-Reduce.

que é coordenado pelo mestre ou o *Driver Program*, responsável pelo agendamento de tarefas e pela criação do *Spark Context*. O *Spark Context* se conecta a vários tipos de gerenciadores de aglomerado, como o Spark StandAlone, Mesos ou Hadoop YARN (*Yet Another Resource Negotiator*). Esses gerenciadores de aglomerado são responsáveis pela alocação de recursos entre aplicativos. Uma vez conectado, o Spark executa a tarefa dentro dos gerenciadores de tarefas, que executam o processamento e o armazenamento de dados, equivalentes aos trabalhadores Storm, e os resultados são então comunicados ao *Spark Context*. O mecanismo descrito para o Storm, no qual cada processo de trabalho é executado dentro de uma topologia, pode ser aplicado ao Spark, no qual aplicativos ou trabalhos são equivalentes as topologias. A desvantagem deste conceito no Spark é a troca de mensagens entre diferentes programas, que é feito indiretamente, como escrever dados em um arquivo, piora a latência que pode ser da ordem de segundos em aplicações de várias operações.

Como o Spark opera sobre dados armazenados na memória volátil, há a necessidade de se prover a tolerância a falhas para os dados enquanto estão sendo processados, e não somente após a gravação em disco como feito no Hadoop. O Spark tem semântica de entrega de mensagens “exatamente uma vez”. A ideia é processar uma tarefa em vários nós de trabalho distintos e, no evento de uma falha, o processamento do microlote pode ser redistribuído e recalculado. O estado dos RDDs é periodicamente replicado para outros nós de trabalho. As tarefas são então discretizadas em tarefas menores executadas em qualquer nó, sem afetar a execução. Assim, as tarefas com falhas podem ser lançadas em paralelo, distribuindo uniformemente a tarefa, sem afetar o desempenho. Este procedimento é chamado de recuperação paralela. A semântica de “exatamente uma vez” reduz a sobrecarga em comparação ao *upstream backup*, em que todas as tuplas devem ser positivamente reconhecidas, como no Storm. No entanto, o processamento de microlote tem desvantagens. O processamento de microlotes leva mais tempo nas operações para baixo (*downstream*). A configuração e distribuição de cada microlote pode demorar mais do que a taxa de chegada do fluxo nativo. Consequentemente, microlotes podem ficar enfileirados na fila de processamento.

Os criadores da Spark fundaram a DataBricks¹¹ em 2013, uma empresa para dar suporte na ferramenta Spark. O Twitter ainda gerencia o projeto Storm. Da mesma forma que a DataBricks está para o Spark, os proprietários do Apache Flink criam a Data Artisan¹². Em 2011, o Yahoo desenvolveu o S4 [Neumeyer et al. 2010] e doou para a fundação Apache, mas o projeto nunca evoluiu para incubação e foi interrompido em 2015. As grandes empresas desenvolvem seus próprios produtos. Google Millwheel [Akidau et al. 2013], IBM InfoSphere Streams [Ballard et al. 2014], Microsoft Azure Analytics¹³ entre outros são exemplos de soluções proprietárias.

Esquemas de Tolerância a Falhas nas Plataformas de Processamento Distribuído

Um esquema robusto de tolerância a falha é essencial para as plataformas de processamento distribuído que rodam em aglomerados, que são sensíveis a falhas de nó, de rede e de *software*. Deve ser observado que um centro de dados possui uma estrutura em aglomerados computacionais, nos quais os nós são servidores de baixo custo (*commercial off-the-shelf* - COTS). Nos sistemas de processamento em lote, a latência é aceitável e, consequentemente, o sistema não precisa se recuperar rapidamente de uma falha. No entanto, em sistemas em tempo real, as falhas podem significar perda de dados, uma vez que os dados não estão armazenados. Portanto, a recuperação rápida e eficiente é importante para evitar a perda de informações [Kamburugamuve et al. 2013].

A forma mais comum de recuperação de falhas é o armazenamento para reenvio (*upstream backup*). Ao considerar a topologia de processamento, o algoritmo usa os nós pais para atuar como *backups*, armazenando e preservando temporariamente as tuplas em suas filas de saída até que seus vizinhos, descendentes, as processem e enviem um reconhecimento positivo (*acknowledgement* - ACK). Toda tupla deve ser individualmente reconhecida positivamente com um ACK. Se algum desses vizinhos descendentes falhar, um ACK não será enviado e, por estouro de temporizador, o nó pai reproduz as tuplas em um outro nó. Uma outra forma de reconhecimento positivo é por grupo de tuplas. Basta identificar uma tupla que falta que o grupo inteiro de tuplas é reproduzido.

Uma desvantagem desta abordagem é o tempo de recuperação longo, já que o sistema deve aguardar até o nó protegido assumir o controle. Para abordar este problema, em [Zaharia et al. 2013] é proposto o algoritmo de recuperação paralela. Neste algoritmo, o sistema verifica periodicamente os estados replicando de maneira assíncrona para outros nós. Quando um nó falha, o sistema detecta as partições em falta e lança tarefas para recuperá-las do último ponto de verificação. Muitas tarefas podem ser lançadas ao mesmo tempo para calcular diferentes partições em diferentes nós. Assim, a recuperação paralela é mais rápida do que o *backup* a montante.

Outra solução é proposta em [Carbone et al. 2015b] com base no algoritmo de instantâneos de barreira assíncrona (*Asynchronous Barrier Snapshotting*- ABS). A ideia central é fazer uma marcação do estado global de um sistema distribuído. Na proposta, um *snapshot*, ou um instantâneo, é o estado global do grafo de processamento, capturando todas as informações necessárias para reiniciar o cálculo desse estado de execução específico. Uma barreira separa o conjunto de registros que acompanham o instantâneo

¹¹DataBricks analytics <https://databricks.com/>

¹²Data Artisan Analytics <http://data-artisans.com/>

¹³Análise Microsoft Azure <https://azure.microsoft.com/en-us/services/stream-analytics/>

atual dos registros que se inserem no próximo instantâneo. As barreiras não interrompem o fluxo de dados. Várias barreiras de instantâneos diferentes podem estar no fluxo ao mesmo tempo, o que significa que vários instantâneos podem ocorrer simultaneamente. Quando uma fonte recebe uma barreira, a fonte tira um *snapshot* do seu estado atual e, então, transmite a barreira para todas as saídas. Quando uma tarefa não-fonte recebe uma barreira de uma de suas entradas, ela bloqueia essa entrada até receber uma barreira de todas as entradas. Quando as barreiras foram recebidas de todas as entradas, a tarefa faz um *snapshot* de seu estado atual e transmite a barreira para suas saídas. Então, a tarefa desbloqueia seus canais de entrada para continuar sua computação. Assim, a recuperação de falhas reverte todos os estados do operador para seus respectivos estados retirados do último instantâneo bem-sucedido e reinicia os fluxos de entrada a partir da última barreira para a qual há um instantâneo.

A semântica de garantia de entrega que um sistema oferece para processar ou não uma amostra pode ser dividida em três tipos: “exatamente uma vez” (*exactly once*), “pelo menos uma vez” (*at least once*), e “no máximo uma vez” (*at most once*). A semântica mais simples a “no máximo uma vez”, em que não há recuperação de erro, ou seja, as amostras são processadas ou perdidas. Quando ocorre uma falha, os dados podem ser encaminhados para outro elemento de processamento sem perder informações. Na semântica “exatamente uma vez” os reconhecimentos positivos são individuais por tupla. Na semântica “pelo menos uma vez”, a correção de erro é feita em conjunto para um grupo de amostras, dessa forma, se ocorrer um erro com qualquer uma das amostras, o grupo inteiro é reprocessado e, portanto, é possível que algumas amostras sejam processadas mais de uma vez. A semântica “pelo menos uma vez” é menos custosa do que “exatamente uma vez”, que requer a confirmação individual para cada tupla processada.

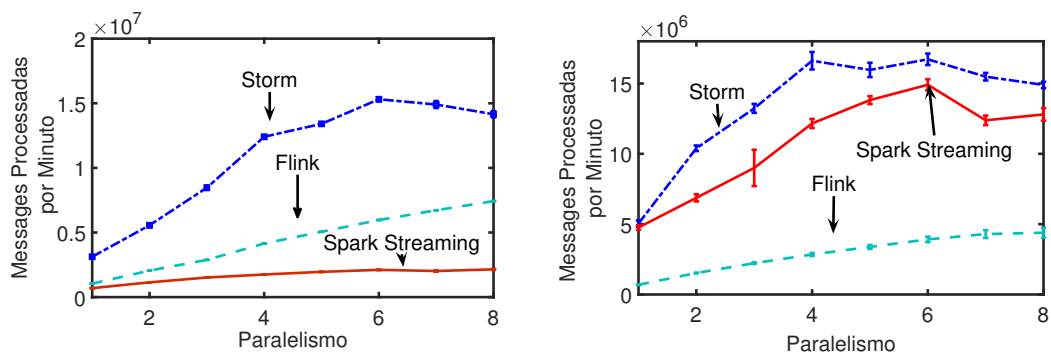
A Tabela 3.2 apresenta um resumo das características de comparação dos sistemas de processamento de fluxo. O modelo de programação pode ser classificado como composicional e declarativo. A abordagem composicional fornece blocos de construção básicos, como *Spouts* e *Bolts* em Storm e devem ser conectados entre si para criar uma topologia. Por outro lado, os operadores do modelo declarativo são definidos como funções de ordem superior, que permitem escrever código funcional com tipos abstratos e o sistema cria automaticamente a topologia.

Tabela 3.2. Comparação entre plataformas de processamento distribuído de fluxo.

	Storm	Flink	Spark Streaming
Abstração do Fluxo	Tuplas	DataStream	D-Stream
Ling. Programação	Java/Clojure	Java/Scala	Java/Scala
Tolerância Falhas	Mínimo 1 vez	Exatamente 1 vez	Exatamente 1 vez
API	Composicional	Declarativa	Declarativa
Mecanismo Falha	<i>Backup</i>	Barreiras	Recuperação
Subsist. de Falhas	Nimbus, Zookeeper	Não há	Não há

3.3. Avaliação de Desempenho das Plataformas para a Classificação com Aprendizado de Máquina

Esta seção avalia a taxa de processamento e o comportamento durante a falha do nó das três plataformas de processamento de fluxo apresentadas: Apache Storm versão 0.9.4, Apache Flink versão 0.10.2 e Apache Spark Streaming versão 1.6.1, com tamanho de microlotes estabelecido em 0,5 segundos. A aplicação avaliada é um sistema de detecção de ameaças com um classificador de rede neural programado em Java. As experiências foram realizadas em um ambiente com oito máquinas virtuais executadas em um servidor com o processador Intel Xeon E5-2650 a 2,00 GHz e 64 GB de RAM. A configuração de topologia de experiência é um mestre e sete nós de trabalho para os três sistemas avaliados. Os resultados são apresentados com um intervalo de confiança de 95%. O Apache Kafka na versão 0.8.2.1, que opera como um serviço de publicação/inscrição, foi usado para inserir dados a taxas elevadas nos sistemas de processamento de fluxo. No Kafka, as amostras, ou eventos, são chamadas de mensagens, nome usado a partir de agora. O Kafka abstrai o fluxo de mensagens em tópicos que atuam como *buffers* ou filas, ajustando diferentes taxas de produção e consumo. Portanto, os produtores registram os dados em tópicos e os consumidores leem os dados desses tópicos. O conjunto de dados usado foi criado pelos autores [Andreoni Lopez et al. 2017a], que foi replicado para obter dados suficientes para avaliar o máximo de processamento no qual o sistema pode operar.



(a) Mensagens processadas por minuto aplicação de detecção de intrusão.

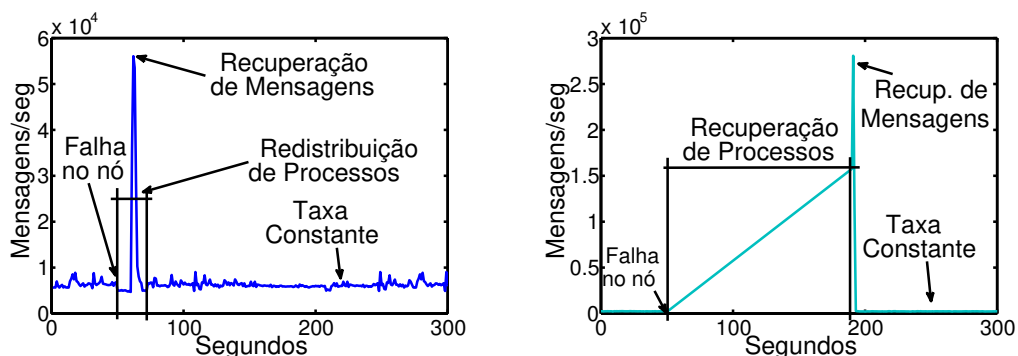
(b) Mensagens processadas por minuto aplicação de contagem de palavras

Figura 3.7. Comparação do paralelismo em plataformas de processamento de fluxo. a) Número de mensagens processadas por minuto em função do paralelismo de tarefas para a aplicação de detecção de intrusão no conjunto de dados de segurança. b) Número de mensagens processadas por minuto em função do paralelismo de tarefas para a aplicação de contagem de palavras no conjunto de dados do Twitter.

O primeiro experimento avalia o desempenho das plataformas em termos de processamento [Andreoni Lopez et al. 2018]. O conjunto de dados é injetado no sistema em sua totalidade e replicado quantas vezes for necessário para criar um grande volume de dados. O experimento calcula a taxa de consumo e processamento de cada plataforma. Também foi variado o parâmetro de paralelismo, que representa o número total de núcleos disponíveis para o *cluster* processar amostras em paralelo. A Figura 3.7(a) mostra os resultados da experiência. O Apache Storm apresenta a maior taxa de transferência.

Para um único núcleo, sem paralelismo, o Storm já mostra um melhor desempenho, com uma vazão pelo menos 50% maior quando comparado a Flink e Spark Streaming. Flink tem um crescimento linear, mas com valores sempre inferiores aos de Apache Storm. A taxa de processamento de Apache Spark Streaming, quando comparada a Storm e a Flink, é muito inferior e isso se deve ao uso de microlote. Cada microlote é agrupado antes do processamento, gerando um atraso no processamento por amostras. O comportamento de Apache Storm é linear até o paralelismo de quatro núcleos. Então, a taxa de processamento cresce até o paralelismo de seis, em que o sistema satura. Esse comportamento também foi observado em Apache Spark Streaming com o mesmo paralelismo de seis núcleos.

Para avaliar o desempenho outro experimento foi criado. Esse experimento conta o número de vezes que cada palavra aparece em um texto, usando um conjunto de dados que contém mais de 5.000.000 tweets [Cheng et al. 2010]. As três plataformas oferecem o aplicativo de contagem de palavras como exemplos de tutoriais, portanto, mostramos esse resultado para obter uma comparação imparcial que não é afetada pela implementação do código em cada plataforma. Figura 3.7(b) mostra o desempenho dos três sistemas no programa `wordcount`. Esse experimento mostra um resultado semelhante ao mostrado anteriormente.

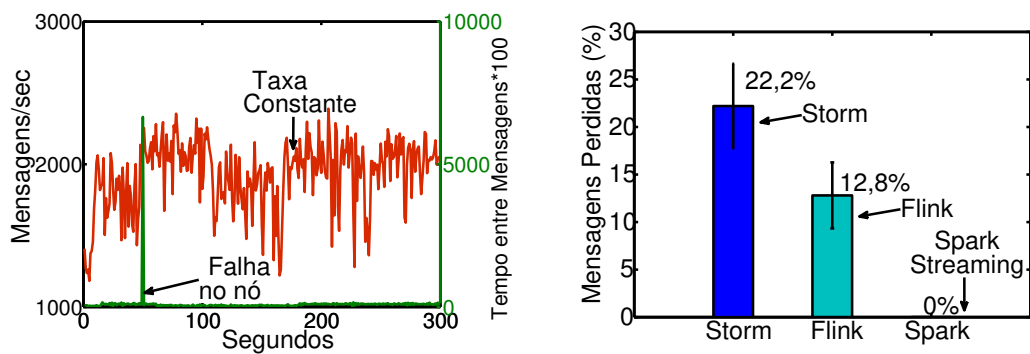


(a) Comportamento da ferramenta Storm ante uma falha de nó.

(b) Comportamento da ferramenta Flink ante uma falha de um nó.

Figura 3.8. Comportamento de Storm e Flink durante uma falha de nó. Uma falha é produzida em 50 segundos. a) Storm e b) Flink. Comportamento dos sistemas após detectar a falha consiste em procedimentos de redistribuição de processos e recuperação de mensagens.

O próximo experimento mostra o comportamento do sistema quando um nó falha. As mensagens são enviadas a uma taxa constante para analisar o comportamento do sistema durante a falha. A falha do nó é simulada desligando uma máquina virtual. As figuras 3.8(a), 3.8(b) e 3.9(a) mostram o comportamento dos três sistemas antes e depois de uma falha do nó do trabalhador em 50 segundos. O Apache Storm leva algum tempo nos processos de redistribuição após a detecção da falha. Esse tempo é devido à comunicação com o *Zookeeper*. O *Zookeeper* tem uma visão geral do *cluster* e reporta o estado ao Nimbus no Storm, que redistribui os processos em outros nós. Logo após esta redistribuição, o sistema recupera mensagens do Kafka em aproximadamente 75 segundos. Embora o sistema possa recuperar rapidamente da falha do nó, durante o processo



(a) Comportamento da ferramenta Spark ante uma falha de um nó.

(b) Comparação da perda de mensagens nos três sistemas durante a falha do nó.

Figura 3.9. Comportamento do Spark Streaming ante uma falha de nó e comparação de perdas de mensagens. a) O comportamento do sistema Spark sob falha, indicando que ele se mantém estável e não perde mensagens. b) Porcentagem de perdas de mensagens.

há uma perda de mensagem significativa. Um comportamento semelhante é observado no Apache Flink. Depois de detectar a falha em aproximadamente 50 segundos, o sistema redistribui os processos para nós ativos. O Flink faz esse processo internamente sem a ajuda de nenhum subsistema, ao contrário do Apache Storm que usa o *Zookeeper*.

A Figura 3.8(b) mostra o período de tempo em que o Flink redistribui processos é muito maior do que o tempo gasto no Apache Storm. No entanto, a recuperação de mensagens também é maior, perdendo algumas mensagens durante a redistribuição do processo. A Figura 3.9(a) mostra o comportamento de Spark Streaming durante uma falha. Quando ocorre uma falha em aproximadamente 50 segundos, o comportamento do sistema é basicamente o mesmo que antes. Isto é devido ao uso de tarefas em micro-lote que são distribuídas rapidamente, sem afetar o desempenho. O Spark Streaming não mostra perda de mensagem durante a falha. Assim, apesar do baixo desempenho do Spark Streaming, esse pode ser uma boa escolha em aplicações em que a resiliência e o processamento de todas as mensagens são necessários. Figura 3.9(b) mostra a comparação de mensagens perdidas entre o Storm, o Flink e o Spark. O Spark não teve perda durante a falha. O resultado mostra o percentual de mensagens perdidas por sistemas, calculada pela diferença de mensagens enviadas pelo Apache Kafka e por mensagens analisadas pelos sistemas. Assim, Apache Flink tem uma menor perda de mensagens durante uma falha com cerca de 12,8% em comparação com 22,2% em Storm. Os resultados foram obtidos com intervalo de confiança de 95%. Pode-se concluir que o Apache Storm apresentou os melhores resultados em relação a vazão de dados. Por outro lado, se a tolerância a falhas e o processamento de mensagens “exatamente uma vez” for necessário, o Apache Spark e o Apache Flink são boas opções.

3.4. O Aprendizado de Máquina e o Problema de Detecção de Ameaças em Tempo Real

Em Segurança da Informação, define-se como **ameaça** um evento externo que explora uma **vulnerabilidade**, interna a um sistema, causando um **incidente** de segurança que tem como efeito algum **impacto**. Nesse contexto, a vulnerabilidade é intrínseca do

sistema e pode nunca vir a ser descoberta. Ao ser descoberta uma nova vulnerabilidade, essa passa a ser possível de ser explorada por um agente externo, configurando uma ameaça. Ao longo deste capítulo, são definidas como ameaças a exploração, a tentativa de exploração ou até mesmo a busca por vulnerabilidades em redes de computadores. Tais ações apresentam comportamentos característicos e identificá-los entre o tráfego legítimo é um desafio em segurança. Outro desafio é detectar as ameaças de forma rápida, e se possível, em tempo real. A detecção tardia de ameaças, pela análise de conjuntos de dados armazenados previamente, impede o acionamento de contramedidas eficazes. Além disso, na era das Grandes Massas de Dados (*big data*), a alta capacidade de sistemas de defesa de processar grandes volumes de dados para detectar ameaças é fundamental. Quando a ameaça explora uma vulnerabilidade recém descoberta, chama-se de ameaça inédita (*zero-day threat*). Ameaças inéditas configuram outro grave problema para a segurança, pois passam despercebidas por sistemas de defesas, como sistemas de detecção de intrusão e antivírus, podendo permanecer indetectáveis por semanas e causar impactos catastróficos. Nesse sentido, as técnicas de aprendizado de máquina permitem aprender com o comportamento da rede, se adaptar a mudanças no tráfego de forma automática e detectar ameaças, até então, desconhecidas. Aplicar estas técnicas em linha (*online*) é fundamental para detectar rapidamente e até impedir a ocorrência de incidentes de segurança, limitando os impactos causados por uma ameaça [Chandrasekar et al. 2017]. Portanto, o aprendizado de máquina integrado à tecnologia de processamento distribuído de fluxo é uma poderosa estratégia para solucionar o problema de detecção de ameaças em tempo real.

O problema da detecção de ameaças em tempo real refere-se à classificação dos dados, representados como amostras, assim que chegam ao sistema de classificação. Em geral, as amostras são classificadas em normal ou ameaça. Normal refere-se ao tráfego benigno, enquanto ameaça refere-se ao tráfego suspeito de ser malicioso. O problema de detecção de ameaças em tempo real é um problema de classificação de dados de fluxo potencialmente ilimitados, em que a tomada de decisão deve ser limitada a tempos próximo ao tempo real [Gaber et al. 2005]. Portanto, a classificação de dados em fluxo em tempo real apresenta os desafios da escolha do conjunto de dados adequado, do algoritmo de classificação, das características de representação dos dados, da arquitetura de detecção em fluxo, e, por fim, da avaliação e comparação de classificadores.

Algoritmos de aprendizado de máquina têm a capacidade de aprender de forma automática com os dados, ou seja, sem serem explicitamente programados para este fim. O algoritmo procura aprender e descobrir como realizar tarefas importantes a partir de generalizações dos exemplos. Como cada vez mais dados estão sendo disponibilizados, a tomada de decisão com a intervenção humana torna-se cada vez mais difícil e, então, a proposta de usar aprendizado de máquina para automatizar tarefas fica cada vez mais atrativa. No entanto, o sucesso no desenvolvimento de aplicações de aprendizado de máquina requer diferentes aspectos que não são fáceis de serem seguidos. Um primeiro desafio é escolher qual algoritmo de aprendizado de máquina que deve ser usado, pois existem milhares de algoritmos e todo ano centenas de novos algoritmos são propostos [Domingos 2012]. Esta seção foca em algoritmos de aprendizado de máquina aplicados à arquitetura de detecção de ameaças em tempo real em redes de computadores. Para isso, são introduzidos conceitos e exemplos de conjunto de dados, as técnicas de seleção e de redução

de características de tráfego e as métricas de avaliação de desempenho de classificadores mais utilizadas na detecção de ameaças. As arquiteturas de detecção em tempo real, tanto com treinamento prévio quanto com treinamento adaptativo para a detecção de ataques inéditos (*zero-day attacks*), são apresentadas e a implementação dos algoritmos nas arquiteturas de detecção é avaliada.

Para avaliar os mecanismos de defesa contra ataques de rede, o primeiro desafio é obter um conjunto de dados adequado para as avaliações. A disponibilidade de conjuntos de dados na literatura é limitada, pois há a preocupação com a privacidade e o receio de vazamento de informações confidenciais contidas na carga útil de pacotes [Heidemann and Papadopoulos 2009]. Um dos principais conjuntos de dados disponíveis é o **DARPA** [Lippmann et al. 2000a], composto por tráfego TCP/IP em formato cru (*raw data*) e dados do sistema operacional UNIX de uma rede simulada, obtidos ao longo de sete semanas de coleta totalizando 9,87 GB de dados. Como o DARPA 98 consiste em arquivos brutos, é preciso extrair as características desses arquivos para usá-los em algoritmos de aprendizado de máquina. Uma maior quantidade de tráfego de fundo e diferentes tipos de ataques foram adicionados para construir o DARPA 99. As primeiras duas semanas foram livres de ataque, portanto, é adequado para treinar algoritmos de detecção de anomalia. Nas cinco semanas seguintes vários ataques simulados foram usados contra a base. Novos ataques foram inseridos no DARPA 99 [Haines et al. 2001], principalmente ataques em diferentes sistemas operacionais como SunOS, Solaris, Linux, e Windows NT. A maioria das pesquisas utiliza uma mistura dos dois conjuntos de dados referindo-se ao conjunto de dados do DARPA. O conjunto de dados **KDD99**, por sua vez, foi criado a partir dos arquivos do conjunto DARPA 98 para uma competição de detecção de intrusão no ano de 1999 [Lee et al. 1999], e é composto de amostras definidas por 41 características e uma classe de saída. O conjunto de dados possui duas semanas de ataques. As classes são divididas em cinco categorias que contêm 24 tipos de ataques para o treinamento e mais 14 tipos de ataques no conjunto de treino, totalizando 38 ataques. O conjunto de treinamento possui 494.021 fluxos e o de teste 311.029 fluxos. As classes incluem Negação de Serviço (*Denial of Service* - DoS), sondagem (*Probe*), *Root2Local* (R2L), *User2Root* (U2R) e normal. Um dos problemas do KDD99 é o desbalanceamento. Aproximadamente 80% das amostras são consideradas ataques, o que difere largamente da realidade. O conjunto contém poucos tipos de ataques U2R e R2L e muitos destes ataques são duplicados. Ao duplicar amostras, os classificadores se tornam tendenciosos a ataques de negação de serviço (DoS) e ao tráfego normal, os quais são os mais abundantes no KDD99. O **NSL-KDD** é uma modificação do conjunto original KDD-99 e possui as mesmas 41 características e as mesmas cinco categorias que o KDD 99 [Tavallae et al. 2009b]. As melhoras do NSL-KDD em relação ao KDD 99 são a eliminação de amostras redundantes e duplicadas, para evitar uma classificação tendenciosa e o sobreajuste (*overfitting*), e um melhor balanceamento entre classes para evitar a escolha aleatória. Apesar da redução em tamanho, o NSL-KDD mantém as proporções entre as classes como no KDD 99, sendo 125.973 amostras de treino e 22.544 de teste. No entanto, o DARPA, o KDD, assim como o NLS-KDD são criticados, pois os seus tráfegos são sintéticos e, portanto, não representam fielmente cenários reais de rede de computadores [Tavallae et al. 2009a]. Existem dados redundantes, que afetam o desempenho dos algoritmos de classificação de dados. Outra crítica importante é que esses conjuntos de dados estão

desatualizados, uma vez que foram simulados há mais de 15 anos [Sommer and Paxson 2010] e muitas aplicações, bem como ataques, surgiram desde então. Desde a criação do KDD99, outros conjuntos de dados foram publicados e disponibilizados introduzindo vantagens e desvantagens. Dessa forma, não há um conjunto de dados que atenda a todos os casos e a escolha de qual usar depende do cenário e da aplicação. Alguns exemplos de conjunto de dados frequentemente encontrados na literatura são o **UNB ISCX IDS 2012** [Shiravi et al. 2012], tráfego da simulação de um cenário real, o **CTU-13** [Garcia et al. 2014], tráfego simulado de uma *botnet*, o **CAIDA DDoS**¹⁴, tráfego de um ataque DDoS real, o **MAWI** [Fontugne et al. 2010], tráfego real de um *backbone* entre EUA e Japão, o **Kyoto**, tráfego real de *honeypots* e o **LBNL**¹⁵ tráfego real de roteadores de borda da LBNL.

O Grupo de Teleinformática e Automação da Universidade Federal do Rio de Janeiro (**GTA/UFRJ**) elaborou um conjunto de dados para testes e avaliações de desempenho [Lobato et al. 2016]. O conjunto de dados corresponde à captura de pacotes de tráfego reais do laboratório e contém comportamentos legítimos de usuários e ameaças de rede reais. Os ataques reais foram realizados de forma controlada pela ferramenta de teste de penetração *Kali Linux*, com a base de ataques mais recente até sua data de criação, contendo um total de sete tipos de ataques de negação de serviço e nove tipos de varreduras de porta. A principal vantagem desse conjunto de dados é que os ataques são mais atuais em relação a outros conjuntos de dados disponíveis na literatura. Após a captura, os pacotes são agrupados no formato de fluxos, em que o fluxo é definido como uma sequência de pacotes com os mesmos endereços IP de origem e destino dentro de uma janela de tempo. Então, foram extraídos dados de cabeçalhos TCP/IP de cada pacote e abstraídos em 26 características de fluxo, como as taxas de pacotes TCP, UDP e ICMP, a quantidade de pacotes com portas de origem e de destino, a quantidade de pacotes com cada sinal (*flag*) TCP, entre outras. Uma análise detalhada das características e ataques pode ser encontrada em [Andreoni Lopez et al. 2017a]. O conjunto de dados tem mais de 95 GB em pacotes capturados, resultando em 214.200 fluxos de tráfego malicioso e normal. A classe normal representa aproximadamente 70% do conjunto de dados, enquanto a classe de negação de serviço 10% do conjunto de dados e a classe de varredura da porta 20% das amostras.

O **NetOp** é um conjunto de dados reais de uma operadora do Brasil [Andreoni Lopez et al. 2017c, Andreoni Lopez et al. 2017d], usado na avaliação de desempenho de arquiteturas propostas por Andreoni *et al.*. O conjunto de dados possui tráfego anonimizado de acesso de 373 usuários de banda larga da Zona Sul da cidade do Rio de Janeiro. Os dados foram obtidos durante 1 semana de coleta de dados ininterrupta, de 24 de fevereiro a 4 de março de 2017. O conjunto de dados contém 5 TB de dados reais em formato bruto (*raw data*) que representam um total de 5,3 milhões de fluxos, definidos pela quintupla IPs e portas de origem e de destino e protocolo de transporte. Os dados foram previamente rotulados pelo sistema de detecção de intrusão (*Intrusion Detection System* - IDS) Suricata para validar os fluxos entre tráfego legítimo ou malicioso. Os pacotes são resumidos em um conjunto de dados de 46 características de fluxos associadas

¹⁴The Cooperative Analysis for Internet Data Analysis, <http://www.caida.org>

¹⁵Lawrence Berkeley National Laboratory - LBNL/ICSI Enterprise Tracing Project, <http://www.icir.org/enterprise-tracing/download.html>

à uma classe de alarme do IDS ou à classe de tráfego legítimo [Andreoni Lopez et al. 2017d].

A Tabela 3.3 resume as principais características dos conjuntos de dados utilizar na literatura de detecção de intrusão.

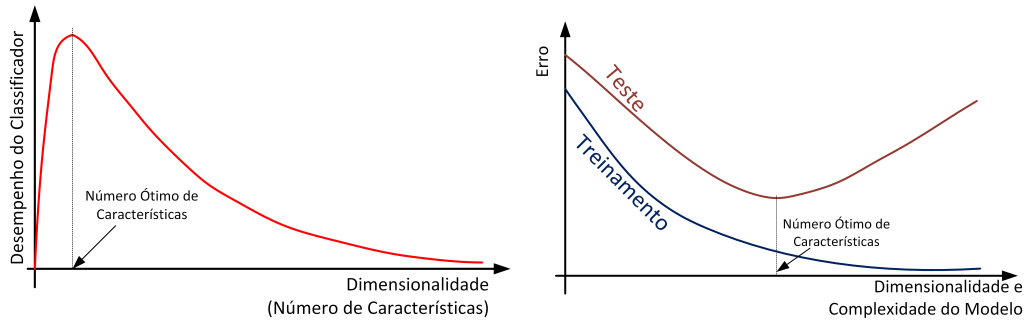
Tabela 3.3. Resumo dos principais conjuntos de dados disponíveis na literatura (Ca: características; FI: fluxos).

Conjunto	Formato	Tamanho	Ataques	Tipo	Ano
DARPA 98/99	pcap	9.87 GB	80.1 %	Sintético	1998/99
KDD99	csv (41 Ca)	805.050 FI	80.1 %	Sintético	1998
NLS-KDD	csv (41 Ca)	148.517 FI	80.5 %	Sintético	1998
LBNL	pcap	11 GB	-	Real	2005
Kyoto	txt (24 Ca)	14 GB	100%	Real	2006
CAIDA DDoS	pcap	21 GB	100%	Real	2007
ISCX IDS	pcap	84.42 GB	2.8 %	Sintético	2012
CTU-13	pcap	697 GB	11.69 %	Sintético	2014
MAWI	pcap	~ 2 GB/ano	-	Real	2001–18
GTA/UFRJ	csv (26 Ca)	95 GB	30 %	Real/Controlado	2016
NetOp	csv (46 Ca)	5.320.955 FI	-	Real	2017

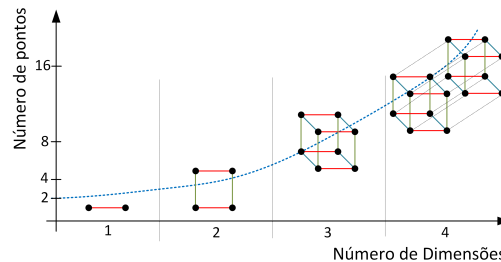
3.4.1. Seleção de Características e Redução da Dimensionalidade

Uma informação pode ser representada ou classificada por suas características ou atributos. A quantidade de características ou atributos usados para representar uma informação varia bastante. Um problema relevante é que o aumento do número de atributos nem sempre melhora a acurácia da classificação da informação. Este problema é conhecido como “a maldição da dimensionalidade” [Zhai et al. 2014] que afirma existir um número ótimo de características que podem ser selecionados em relação ao tamanho da amostra para maximizar o desempenho do classificador. A Figura 3.10(a) mostra que a medida que a dimensionalidade aumenta, o desempenho do classificador aumenta até que o número ótimo de características seja atingido e, a partir desse valor ótimo, o desempenho cai. Logo, a partir desse valor ótimo do número de características, aumentar a dimensionalidade sem aumentar o número de amostras de treinamento resulta em uma diminuição no desempenho do classificador. A Figura 3.10(b) mostra a degradação do desempenho dos algoritmos. O incremento na dimensionalidade é diretamente proporcional à complexidade do modelo, obtendo uma taxa de erro baixa durante a fase de treinamento. No entanto, esses modelos apresentam o problema de sobreajuste (*overfitting*) ao treinamento, fazendo com que o modelo apresente um desempenho insatisfatório na etapa de teste. A Figura 3.10(c) mostra que o número de pontos aumenta exponencialmente com o aumento da dimensionalidade. Nos espaços com muitas dimensões, os pontos tornam-se esparsos e pouco similares, com pontos muito distantes uns dos outros e aproximadamente equidistantes, o que leva um classificador a cometer mais erros. Existem também outros problemas ao utilizar algoritmos de aprendizado de máquina com um número alto de dimensões, pois algumas métricas tradicionais de distância, como a distância Euclidiana, deixam de ter sentido em dimensões altas, sendo necessária a utilização de outros tipos de métricas, como a distância cosseno, que possuem maior custo computacional.

Ainda, a maioria das técnicas de aprendizado de máquina são ineficazes no tratamento de dados dimensionais elevados, incorrendo em problemas de sobreajuste (*overfitting*) durante a fase de classificação. Portanto, é comum reduzir o número de características ou a dimensionalidade antes da aplicação de um algoritmo de aprendizado de máquina.



(a) Desempenho do classificador com o aumento da dimensionalidade. (b) Efeito de sobreajuste aos dados do treinamento, o que reduz o desempenho no teste.



(c) Crescimento exponencial do número de pontos com o aumento da dimensionalidade.

Figura 3.10. O problema da “maldição da dimensionalidade” afirma existir um número ótimo de características que podem ser selecionadas em relação ao tamanho da amostra para maximizar o desempenho do classificador.

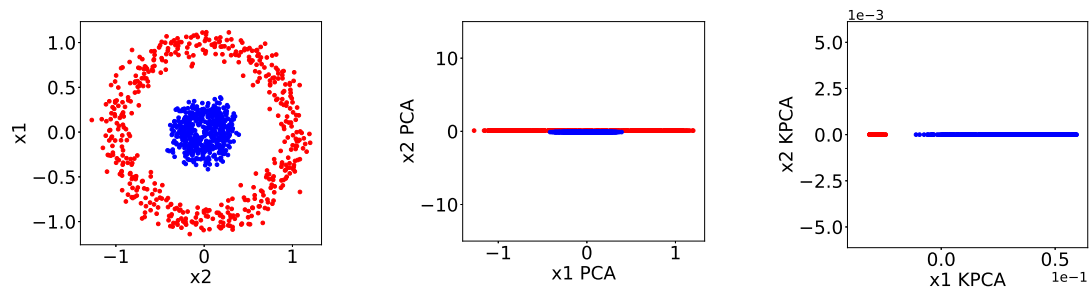
A seleção de características ou a redução de dimensionalidade são duas técnicas bastante usadas para se obter mais desempenho no aprendizado. A seleção de características mantém os atributos mais relevantes do conjunto de dados original, criando um subconjunto das características. Por outro lado, a redução da dimensionalidade aproveita a redundância dos dados de entrada, calculando um conjunto menor de novas dimensões, ou atributos sintéticos. Os novos atributos sintéticos são uma combinação linear ou não linear dos atributos de entrada. A ideia principal dos métodos é remover toda informação redundante, mantendo unicamente a informação necessária e suficiente para representar a informação original. O objetivo tanto da seleção de características como da redução da dimensionalidade é produzir um conjunto mínimo de características para que o novo subconjunto mantenha um desempenho o mais semelhante possível ou superior ao do conjunto gerador. Portanto, a seleção de características e a redução da dimensionalidade melhoram o desempenho da classificação, provendo classificadores mais rápidos e economicamente viáveis. A seleção de características possui uma propriedade adicional, pois permite um melhor entendimento do processo que gera o dado. Na redução de dimensionalidade as características selecionadas são sintéticas e compostas pela combinação das características originais, o que dificulta o entendimento do processo.

A **redução da dimensionalidade** também pode ser vista como o processo de derivar um conjunto de graus de liberdade, que pode ser usado para reproduzir a maior parte da variabilidade de um conjunto de dados [Van Der Maaten et al. 2009]. Idealmente, a representação reduzida deve ter uma dimensionalidade que corresponde à dimensionalidade intrínseca dos dados. A dimensionalidade intrínseca dos dados é o número mínimo de parâmetros necessários para atender as propriedades observadas dos dados. Geralmente, na redução de dimensionalidade é criado um novo espaço de características através de algum tipo de transformação do espaço de características original. Na redução da dimensionalidade, dada a variável aleatória de p -dimensões $\mathbf{x} = (x_1, x_2, \dots, x_p)$, é possível encontrar uma representação dimensional inferior, $\mathbf{s} = (s_1, s_2, \dots, s_k)$ com $k \leq p$.

Muitos algoritmos com abordagens diferentes foram desenvolvidos para reduzir a dimensionalidade que podem ser classificadas em dois grupos: os lineares e os não-lineares. A redução linear de dimensionalidade é uma projeção linear, na qual os dados p -dimensionais são reduzidos em dados k -dimensionais usando k combinações lineares das p dimensões. Dois importantes exemplos de algoritmos de redução de dimensão lineares são a análise de componentes principais (*Principal Component Analysis* - PCA) e a análise de componentes independentes (*Independent Component Analysis* - ICA). O objetivo do PCA é encontrar uma transformação linear ortogonal que maximize a variância das características. O primeiro vetor da base PCA, a componente principal, é o que melhor descreve a variabilidade de seus dados. O segundo vetor é a segunda melhor descrição e deve ser ortogonal ao primeiro e assim por diante em ordem de importância. Por outro lado, o objetivo da ICA é encontrar uma transformação linear na qual os vetores base são estatisticamente independentes e não gaussianos, ou seja, a informação mútua entre duas variáveis do novo espaço vetorial é igual a zero. Ao contrário do PCA, os vetores de base na ICA não são nem ortogonais nem classificados em ordem, todos os vetores são igualmente importantes. O PCA é utilizado quando se quer encontrar uma representação reduzida dos dados. Por outro lado, o ICA é utilizado para se obter a extração de características, selecionando as características que melhor se adequam à aplicação.

Nos casos em que os dados de alta dimensionalidade apresentam uma natureza não-linear, os algoritmos lineares não apresentam bom desempenho. Isto significa que a relação entre as classes não é descrita em um subespaço linear, como mostrado na Figura 3.11(a). Para estes casos, é possível utilizar a técnica do PCA com funções de *kernels*. A função *kernel* transforma os vetores de entrada do espaço original a um espaço dimensional maior, no qual o problema se torna linearmente separável [Schölkopf et al. 1999]. A Figura 3.11 mostra uma comparação dos métodos de redução linear e não linear. O conjunto de dados original, mostrado na Figura 3.11(a), é um caso de dois círculos concêntricos, cada círculo é uma classe. O objetivo é reduzir a 2 dimensões ($\mathbb{R}^2$) para espaço de 1 dimensão ($\mathbb{R}^1$). Depois de aplicar uma redução linear, Figura 3.11(b), as componentes principais não conseguem um subespaço em que as classes estejam separadas linearmente no espaço ($\mathbb{R}^1$). Isso ocorre porque os dois círculos concêntricos são duas classes não-linearmente separáveis. Depois de aplicar um método não-linear, como um *Gaussian Kernel PCA*, mostrado na Figura 3.11(c), o método consegue criar um subespaço onde as classes se separaram corretamente.

Existem duas abordagens para a separação de classes em dados que não podem ser separados de forma linear, como mostra a Figura 3.12(a). A primeira consiste em



(a) Conjunto de dados original de duas classes. (b) Resultado da análise da PCA linear. (c) Resultado da PCA com função de *kernel*, não-linear.

Figura 3.11. Exemplo do “artifício do *kernel*” para o PCA: a) duas classes com separação não-linear, b) dificuldade da separação de classes aplicando PCA linear e c) resultado da separação de classes ao aplicar PCA com função de *kernel* não-linear.

mapear os dados em um espaço de maiores dimensões, no qual as classes possam ser separadas linearmente, ou seja, por retas, como mostra a Figura 3.12(b). No exemplo da Figura 3.12 é um exemplo binário, no qual no plano de $\mathbb{R}^1$ não existe nenhum hiperplano linear que seja capaz de separar as duas classes. Se o problema é levado a um plano com dimensionalidade superior, $\mathbb{R}^1 \rightarrow \mathbb{R}^2$, é possível encontrar um hiperplano, traços na Figura 3.12(b), que separe as classes. O conjunto de dados pode ser mapeado em um espaço de dimensão superior ao original, $\mathbb{R}^1 \rightarrow \mathbb{R}^2$, e com isso possibilitar um hiperplano que separe as classes, como mostra a Figura 3.12(b). Esse hiperplano, quando é trazido para uma dimensão menor, $\mathbb{R}^2 \rightarrow \mathbb{R}^1$, corresponde a uma superfície não-linear como mostrado na Figura 3.12(c). No entanto, ao realizar o mapeamento para dimensões maiores se incorre na “maldição da dimensionalidade” explicada anteriormente, o que gera um alto custo computacional. Para resolver esse problema é aplicado o “artifício do *kernel*”. Uma função *Kernel* é uma função de similaridade que corresponde ao produto escalar em um espaço vetorial expandido. A ideia é encontrar uma função não-linear na qual não seja necessário fazer o mapeamento de dimensões e que a computação seja independentemente do número de características. Se existe uma transformação não-linear $\Phi(x)$ do espaço de características D-dimensional original para um espaço de características M-dimensional, onde geralmente $M \gg D$. Então, cada ponto de dados x_i é projetado para um ponto $\Phi(x_i)$. Se a matriz $K()$ que contém os produtos escalares entre todos os pares de pontos de dados, agora é calculada como $K(x_i, x_j) = \Phi(x_i)^T \Phi(x_j)$. Os *kernels* mais utilizados são o núcleo polinômico, gaussiano e tangente (tangente hiperbólica). Se um *kernel* usado é linear, podemos ver o PCA padrão como um caso do *Kernel PCA*.

A **seleção de características** produz um subconjunto das características originais, que são as melhores representantes dos dados, portanto não há perda de significado, ao contrário da redução de dimensionalidade. As técnicas de seleção de características podem ser divididas em três tipos de algoritmos [Mladenić 2006]: empacotamento (*wrappers*), filtragem e embarcados.

Os métodos de empacotamento, como mostra a Figura 3.13(a), usam algoritmos de aprendizado de máquina, como máquina de vetores de suporte (*Support Vector Machine* - SVM), árvore de decisão, entre outros, para medir a qualidade dos subconjuntos

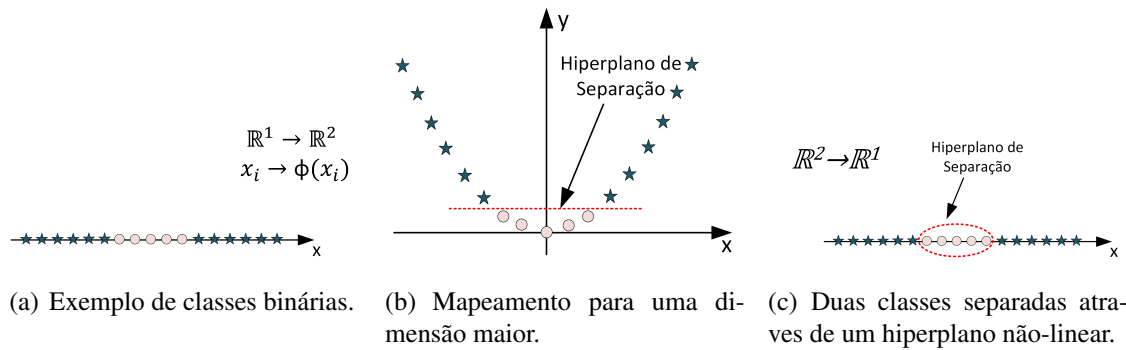


Figura 3.12. Estratégias para separação de classes de distribuição de dados não-linear: a) conjunto de dados original de duas classes com distribuição não-linear; b) mapeamento dos dados com acréscimo de dimensão que possibilite a separação com algoritmos lineares, como o PCA e c) uso de hiperplano não linear, como no algoritmo *Kernel* não-linear do PCA.

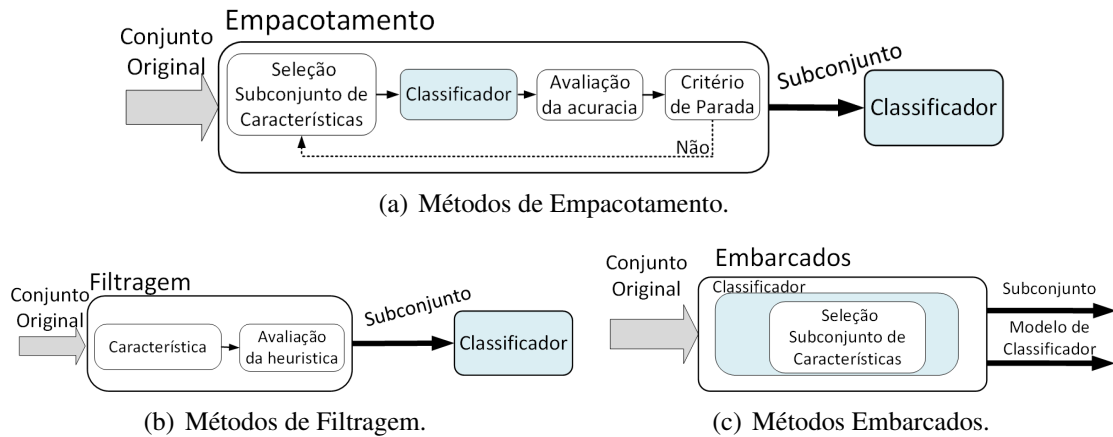


Figura 3.13. Algoritmos de Seleção de Características: a) métodos de empacotamento que utilizam um classificador para avaliar os subgrupos de características, b) métodos de filtragem que utilizam heurísticas para avaliar uma característica ou um subconjunto delas e c) métodos embarcados que utilizam um algoritmo específico de classificação para fazer a seleção naturalmente.

de características sem incorporar conhecimento sobre a estrutura específica da função de classificação. Assim, o método avaliará subconjuntos com base na acurácia do classificador. A avaliação é repetida para cada subconjunto até encontrar um conjunto que apresente um bom desempenho. Os métodos de empacotamento tendem ser mais acurados que os métodos de filtragem, porém apresentam um maior custo computacional.

Para diminuir o alto custo computacional da avaliação os diversos subconjuntos de características baseada nos classificadores foram implementados os métodos de filtragem. Os métodos de filtragem são chamados de métodos de ciclo aberto porque não possuem interação com o classificador. Ao invés de classificadores, o método usa heurísticas para avaliar a relevância da característica no conjunto de dados [Chandrashekar and Sahin 2014, Andreoni Lopez et al. 2017a]. Como seu nome indica, a característica que não supera o critério da heurística é filtrada. O ganho de informação, a distância, a consistência e a semelhança entre características, assim como medidas estatísticas, são algumas

das heurísticas mais utilizadas na avaliação da filtragem. Já que o processo de seleção é feito em uma etapa precedente à classificação, só depois que as melhores características são encontradas, os algoritmos de classificação podem usá-las. Esse método é rápido para selecionar característica, mas, por não ter interação com o classificador, o subconjunto de características selecionado pode não ter boa acurácia. Um dos algoritmos de filtragem mais populares é o Relief, no qual a pontuação das características é calculada como a diferença entre a distância do exemplo mais próximo da mesma classe e o exemplo mais próximo da classe diferente. A principal desvantagem deste método é que as classes dos dados devem ser rotuladas antecipadamente. O Relief é limitado a problemas com apenas duas classes, mas o ReliefF [Robnik-Šikonja and Kononenko 2003] é uma melhoria do método de Relief que trata de classes múltiplas usando a técnica dos k -vizinhos mais próximos. O ReliefF é um método supervisionado, no qual as rotulagens das classes devem ser conhecidas antes da aplicação do método. Em aplicações como o monitoramento de rede e a detecção de ameaças os fluxos de rede chegam aos classificadores sem rótulo. Logo, devem ser aplicados algoritmos não supervisionados. Andreoni *et al.* resolvem esse problema apresentando um algoritmo que executa uma seleção de características não supervisionada utilizando um método de filtragem [Andreoni Lopez et al. 2017a]. A correlação e a variância entre as características é usada para medir a quantidade de informação que cada característica representa em relação às outras independentemente da classe. Assim, o algoritmo rapidamente seleciona as características que apresentam maior informação.

Finalmente, os métodos embarcados são uma sub-classe dos métodos de empacotamento. Neste caso, é obtido o subconjunto de características assim como o modelo no qual foram selecionadas. Os métodos embarcados executam o processo de seleção de características durante a fase de aprendizagem de um classificador. Esses métodos selecionam características com base em critérios gerados durante o processo de aprendizagem de um classificador específico. Em contraste com os métodos de empacotamento, não separam a aprendizagem da seleção de características. Os métodos embarcados possuem um comportamento semelhante aos métodos de empacotamento, usando a acurácia de um classificador para avaliar a relevância da característica. No entanto, os métodos embarcados precisam modificar o algoritmo de classificação no processo de aprendizado.

Ao aplicar a redução de dimensionalidade e a redução de características a dados em fluxo, uma das abordagens é aplicar transformações nos dados quando chegam ao sistema de classificação. Assim, calcula-se um modelo *a priori*, com base em dados armazenados, e aplica-se aos dados que chegam em fluxo. O modelo pode ser tanto calculado para redução de dimensionalidade [Lobato et al. 2018], como para seleção de características [Andreoni Lopez et al. 2017a]. A aplicação da seleção de características sobre dados em fluxo usa técnicas como filtragem e rascunho (*sketching*) dos dados [Gaber et al. 2005, Gama and Rodrigues 2007].

3.4.2. Métricas de Avaliação de Desempenho de Classificadores

As principais métricas utilizadas para avaliar a qualidade de um classificador de detecção de intrusão são:

- **matriz de confusão** é forma tradicional de apresentar o desempenho de um mé-

todo de classificação, que informa a quantidade de Verdadeiros Positivos (VP), Falsos Positivos (FP), Verdadeiros Negativos (VN) e Falsos Negativos (FN) para cada classe. A partir destas informações pode-se derivar outras métricas que facilitam a comparação de diferentes métodos;

- **acurácia** de um método é a razão do total de amostras classificadas corretamente (VP + VN) dividido pelo número total de amostras (P+N). Para conjunto de dados não-balanceados, em que a proporção entre classes não é a mesma, a acurácia não é um indicador confiável de bom desempenho;
- **sensibilidade**, também conhecida como **revocação** (*recall*) ou **taxa de verdadeiros positivos**, é a razão da quantidade de amostras classificadas corretamente para a classe positiva (VP), dividido pelo total de amostras que pertencem a esta classe (VP + FN). O análogo para a classe negativa é chamado de **especificidade** ou **taxa de verdadeiros negativos**;
- **precisão** é a razão entre o número de amostras classificadas corretamente para a classe positiva (VP), dividido pelo total de amostras classificadas para esta classe (VP + FP). Há um compromisso natural entre a sensibilidade e a precisão de um algoritmo e equilibrar uma boa precisão com uma boa sensibilidade é uma tarefa difícil;
- **F1 score** é a média harmônica da precisão com a sensibilidade, expressa por

$$F1\ score = \frac{2}{\frac{1}{precisão} + \frac{1}{sensibilidade}}. \quad (1)$$

Esta métrica busca o compromisso da precisão e com a sensibilidade. Geralmente, quanto maior o F1 score, melhor a classificação.

- **área abaixo da curva ROC** (*Area Under the ROC Curve* - AUC) é uma métrica eficaz para quantificar o desempenho da classificação binária em conjuntos de dados não-balanceados. A curva de Característica de Operação do Receptor (*Receiver Operating Characteristic* - ROC) é um gráfico que representa o compromisso entre a taxa de verdadeiros positivos (sensibilidade) e a taxa de falsos positivos (1 – especificidade). A Figura 3.14(a) exemplifica a comparação das áreas abaixo da curva ROC de diferentes classificadores. Quanto maior a área AUC, melhor é o classificador.

A Figura 3.14(b) ilustra a relação das métricas de precisão, sensibilidade e acurácia com as estatísticas fundamentais da classificação binária. A classe positiva representa as amostras de tráfego normal e a classe negativa as ameaças. Além das métricas tradicionais, outras métricas podem ser utilizadas para complementar a avaliação e comparação dos métodos de classificação, como as medidas de tempo de treinamento e de tempo de predição, *Log Loss*, Coeficiente de Correlação de Matthews, Coeficiente de Cohen's Kappa entre outros [Lippmann et al. 2000b, Gaffney and Ulvila 2001, Gu et al. 2006, Cárdenas and Baras 2006]. É importante notar que o desempenho de um método de

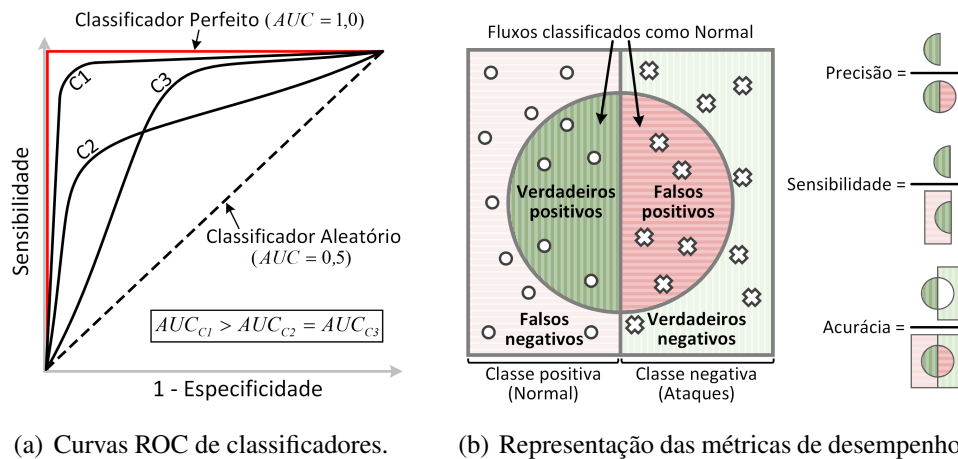


Figura 3.14. a) Curvas ROC de classificadores e comparação da área abaixo da curva ROC (AUC). b) Representação da precisão, sensibilidade e acurácia em um problema de classificação binária.

aprendizado de máquina está condicionado ao conjunto de características selecionadas, às técnicas de redução, seleção ou enriquecimento de características utilizada, à qualidade do conjunto de dados a ser classificado e à aplicabilidade do modelo ao problema.

O desempenho de um modelo representa o quão boa está a aproximação de sua função objetivo para generalizar a realidade. Dessa forma, o sobreajuste em aprendizado de máquina refere-se a uma função objetivo que modela demasiadamente bem os dados de treinamento, tão bem que prejudica a capacidade de generalização do modelo. Assim, o sobreajuste ocorre quando o modelo aprende com as flutuações e o ruído do conjunto de treinamento, o que piora o desempenho do modelo para novos conjuntos de dados. As principais causas do sobreajuste são conjuntos insuficientes de dados, alta dimensionalidade ou modelos com muitos graus de liberdade. O sobreajuste dos dados é mais comum em algoritmos não-paramétricos e não-lineares, em que a função objetivo tem mais flexibilidade de aprendizado. Por isso, esses algoritmos costumam ter mecanismos que procuram limitar e restringir o nível de detalhes do aprendizado.

Duas técnicas bastante empregadas para evitar o sobreajuste são a validação cruzada e o balanceamento. A validação cruzada *K-fold*, representada na Figura 3.15(a), realiza K iterações de treinamento nas frações dos dados e, a cada iteração, nas $K - 1$ frações dos dados restantes, realiza o teste de forma mutuamente exclusiva, sendo $K = 10$ o valor comumente usado. O balanceamento também evita o sobreajuste e é muito empregado em conjuntos de dados em que as quantidades de amostras de cada classe são desproporcionais. Nesse caso, as classes minoritárias não apresentam amostras suficientes para que o classificador aprenda adequadamente sobre elas de forma que o treinamento e, consequentemente, as métricas de avaliação de desempenho, se tornam tendenciosos para a classe dominante. No geral, as técnicas de balanceamento se resumem em replicar amostras da classe minoritária ou eliminar amostras da classe majoritária. Essas duas ações podem ser feitas por algoritmos específicos [Batista et al. 2004] ou de forma aleatória, que possui o risco de eliminar amostras importantes ou gerar mais sobreajuste nos dados. Por outro lado, o sobreajuste em aprendizado de máquina refere-se a um modelo que nem

modela bem o conjunto de treino nem generaliza bem para novos conjuntos de dados. Isso ocorre quando o classificador não é adequado para o problema e apresenta baixo desempenho na classificação. A solução nesse caso é a escolha de um algoritmo diferente de aprendizado. A Figura 3.10(b) ilustra o problema do sob ajuste e sobreajuste de um modelo de classificador.

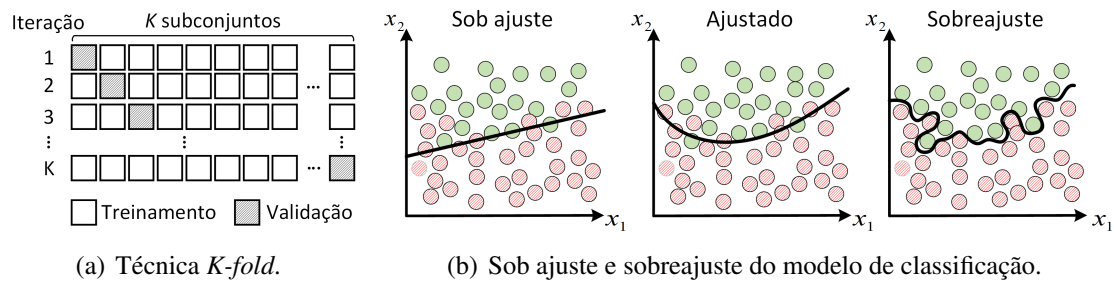


Figura 3.15. a) Técnica de validação cruzada *K-fold*, que particiona o conjunto de dados em *K* partições mutuamente exclusivas para obter uma medida mais confiável do desempenho do classificador. b) Ilustração do sob ajuste e sobreajuste de um modelo de classificação.

3.4.3. Detecção de Ameaças em Tempo Real a partir de Treinamento Prévio

A arquitetura kappa¹⁶, mostrada na Figura 3.16, possui um caminho direto da coleta, passando pelo processamento do fluxo e finalizando na visualização dos dados em tempo real. A kappa é voltada para aplicativos que demandem baixa latência sem a necessidade de armazenamento dos dados.

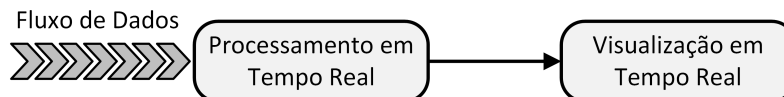


Figura 3.16. A arquitetura kappa provê um caminho direto com a coleta, o processamento do fluxo e a visualização dos dados em tempo real.

A arquitetura lambda¹⁷ combina o processamento de fluxo em tempo real com o processamento em lotes, para analisar grandes massas de dados (*Big Data*) [Marz and Warren 2013]. O processamento em lotes permite o uso de métodos acurados de classificação que necessitam de parâmetros calculados em tempo diferenciado. Assim, esta arquitetura possui um caminho em linha (*online*) de processamento de dados em movimento para obter resultados aproximados a baixa latência e um caminho em tempo diferenciado (*offline*) para processamento em lotes com resultados tardios, porém precisos.

Conforme mostrado na Figura 3.17, a arquitetura lambda possui três camadas: a camada de processamento de fluxo; a camada de processamento em lotes e a camada de serviço. A camada de processamento de fluxo lida com os dados recebidos em tempo real, objetivando a baixa latência no processamento. A camada de processamento em lotes possui uma visão global de uma grande quantidade de dados históricos e os analisa de

¹⁶Arquitetura kappa foi proposta pelo Jay Kreps co-criador do apache Kafka <https://www.oreilly.com/ideas/questioning-the-lambda-architecture>.

¹⁷A arquitetura lambda foi proposta por Nathan Marz criador da plataforma Storm.

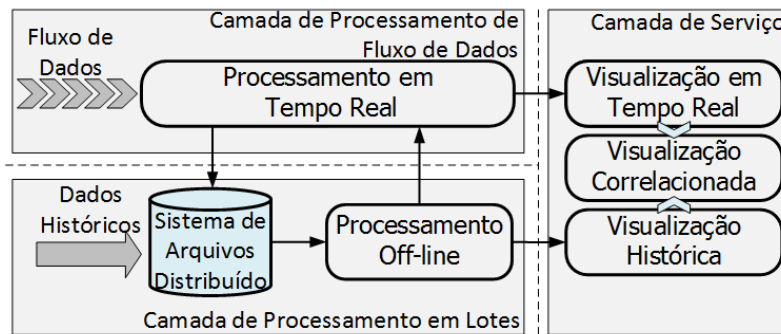


Figura 3.17. A arquitetura lambda que combina as três camadas de processamento de fluxo, de processamento em lotes e de serviço.

maneira distribuída através de técnicas como MapReduce. Finalmente, a camada de serviço combina a informação obtida das duas camadas anteriores para fornecer uma saída composta por dados analíticos para o usuário. Portanto, o objetivo da arquitetura lambda é fornecer análises precisas com base em dados históricos e, ao mesmo tempo, obter resultados em tempo real. Dessa forma, a arquitetura lambda combina o processamento em lotes tradicional em um banco de dados históricos com a análise de processamento de fluxo em tempo real. Isso permite implementar uma gama de aplicações em que uma resposta imediata seja necessária no menor tempo possível assim que novos dados chegam. Uma destas aplicações é a detecção de ameaças em tempo real a partir de modelos pré-treinados dos algoritmos de aprendizado de máquina implantados na camada de processamento de fluxo. Há um período de treinamento do algoritmo, antes de começar a classificar, para ele “aprender” o que é tráfego normal e tráfego malicioso. Esse treinamento, em outras palavras, significa encontrar os melhores parâmetros, no caso dos algoritmos paramétricos, ou encontrar o melhor modelo de classificação no caso de algoritmos não-paramétricos, a partir de dados de exemplo. Na arquitetura lambda, este treinamento é feito em tempo diferenciado na camada de processamento em lotes e o modelo ou parâmetros calculados são atualizados no processamento em fluxo eventualmente.

A Figura 3.18 ilustra como a arquitetura lambda pode ser aplicada no contexto de detecção de ameaças em linha (*online*). A captura em linha (*online*) de dados consiste em analisadores de tráfego e arquivos de registros (*logs*) que geram dados em linha a partir de múltiplas fontes e os enviam para um *buffer* de dados do processamento. Para poupar processamento, a informação gerada pode ser transformada no vetor de características da instância ainda na fonte. A camada de processamento de fluxo consiste em um *buffer* de dados e o módulo de detecção em linha. O *buffer* de dados é um mecanismo de segurança que garante que as instâncias não serão perdidas caso a taxa de chegada de informação seja maior que a capacidade de processamento. Já a detecção online consiste em um arcabouço de processamento de fluxo, que carrega um modelo de classificação previamente estabelecido e o aplica para cada amostra recebida. A camada de processamento em lotes, por sua vez, contém um banco de dados distribuído que armazena os conjunto de dados rotulados que serão usados para treinar o modelo de classificação. O treinamento neste caso é feito em tempo diferenciado (*offline*), e uma vez definido o modelo, seus parâmetros são atualizados. Como os dados que chegam em tempo real não possuem rótulos confiáveis, a atualização do modelo de classificação depende da adição de novos dados

rotulados à base histórica, ou de melhorias nos métodos de treinamento. Portanto, essa forma de detectar ameaças em linha (*online*) não é eficaz contra novos ataques, que ainda não constam da base histórica.

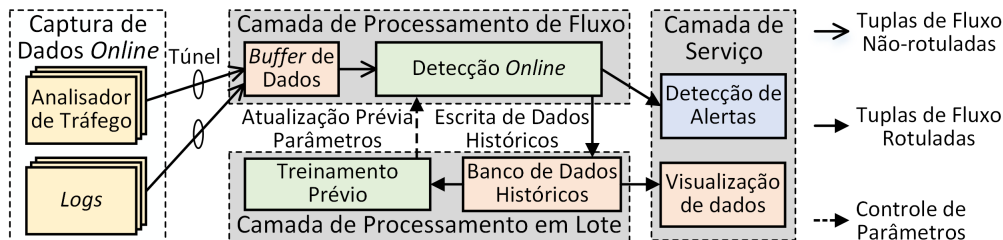


Figura 3.18. Aplicação da arquitetura lambda para a detecção em linha (*online*) de ameaças. O treinamento do classificador é feito em tempo diferenciado (*offline*), enquanto a classificação é em linha (*online*).

Uma grande vantagem desse tipo de arquitetura de classificação é a aplicação de modelos complexos aos dados em fluxo sem a adição do atraso da fase de treinamento. Essa arquitetura é particularmente eficiente para algoritmos que possuem um longo tempo de treinamento e que, uma vez construído o modelo, a predição da classe de saída é rápida. No contexto de detecção de intrusão na Internet, a velocidade em que os dados evoluem é alta e novas classes de ataques surgem a todo momento com a descoberta de vulnerabilidades, e ainda, o comportamento de tráfego benigno está em constante evolução com o surgimento de novas aplicações. Portanto, a eficácia desse tipo de arquitetura na detecção contra novas ameaças depende da frequente atualização dos parâmetros do modelo, o que não é feito de forma automática. A atualização pode ser feita de duas formas básicas, através do aperfeiçoamento dos algoritmos de aprendizado utilizados, como a alteração no métodos ou nas características utilizadas, ou através da adição de novos dados rotulados à base histórica para reforçar o treinamento.

O treinamento prévio do classificador é realizado em tempo diferenciado (*offline*) usando os algoritmos de aprendizado de máquina convencionais. Neste capítulo, o critério de seleção de quais algoritmos de aprendizado de máquina escolher para as aplicações de detecção de intrusão aqui descritas se apoia na lista dos algoritmos mais usados realizada por Buczak e Guven [Buczak and Guven 2015] e nos mais comumente encontrados na literatura de detecção de intrusão.

Os algoritmos de **árvore de decisão** (*Decision Tree* - DT) constroem uma árvore em que cada nó é responsável pelo teste de uma característica do conjunto de dados. Os valores de probabilidade de cada classe são armazenados naquele nó e servem como parâmetros para a tomada de decisão. Assim, os nós ramos representam condições baseadas no valor de uma das características de entrada e os nós folhas representam a classe final. A cada nova amostra desconhecida de entrada, o algoritmo percorre os ramos da árvore e avalia as respectivas características para estimar a probabilidade daquela amostra pertencer a uma determinada classe. Dessa forma, não é necessário percorrer todas as características da amostra para realizar a classificação, o que reduz o tempo de processamento na classificação.

As heurísticas de construção da árvore diferem quanto às diversas implementações do algoritmo existentes na literatura. Dentre as mais conhecidas encontram-se o

Iterative Dichotomiser 3 (ID3), o C4.5 e o *Classification And Regression Tree* (CART). A diferença principal entre as implementações são as métricas definidas para fazer a divisão entre os ramos e as técnicas para evitar sobreajuste. O algoritmo ID3 maximiza o ganho de informação e minimiza a entropia para escolher a ordem de divisão entre os ramos. O CART realiza divisões binárias utilizando a impureza de Gini e, nas suas primeiras versões, usava a redução da variância. O C4.5, por sua vez, é uma evolução do ID3 que permite tanto características categóricas quanto numéricas, além de resolver o problema de sobreajuste usando técnicas de poda.

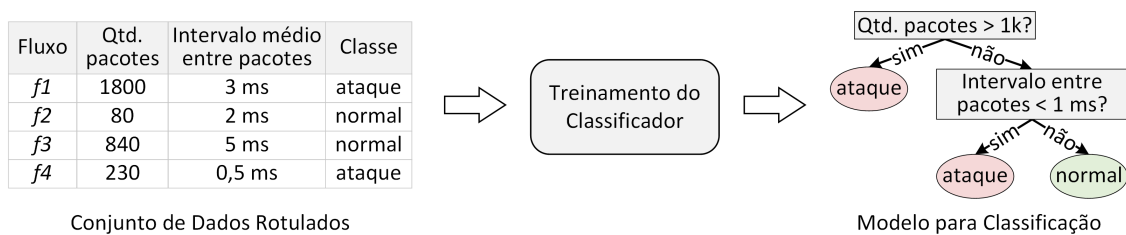


Figura 3.19. Exemplo do treinamento em tempo diferenciado (*offline*) de um classificador do tipo árvore de decisão para negação serviço por inundação. Neste classificador, a saída é um modelo de árvore de probabilidades do tipo *if-then-else*.

As árvores de decisão têm como vantagem a interpretação. Por serem um conjunto de regras do tipo “*if-then-else*”, é possível interpretá-las e até exibi-las graficamente para facilitar o entendimento das regras de decisão. A Figura 3.19 ilustra um treinamento de um classificador de árvore de decisão que, por ter um tamanho reduzido, pode ser facilmente interpretado. Além disso, as árvores são capazes de lidar com os dados numéricos e categóricos e apresentam um bom desempenho com conjunto de dados de grande tamanho. Entretanto, é comum a ocorrência de sobreajuste (*overfitting*) no conjunto de dados, exigindo a utilização de técnicas externas ao algoritmo como a poda (*pruning*).

As **redes neurais artificiais** (*Artificial Neural Networks* - ANN) são famílias de algoritmos inspiradas no funcionamento do cérebro humano, em que cada neurônio realiza uma pequena parte do processamento e transmite o resultado para o próximo neurônio. A transmissão, ou disparo, para o próximo neurônio, só ocorre caso o resultado de uma função de ativação sobre suas entradas seja maior que determinado limiar. As ANN são divididas em camadas, em que cada camada transfere sua saída para a seguinte até que a última camada produza o resultado de saída. A saída de uma ANN representa um grau associado à cada classe, em que o maior grau entre as saídas define a classe da amostra. O processo de treinamento calcula os vetores de peso w_n , que determinam o peso de cada conexão de neurônio. No treinamento, existem espaços de amostra de entrada e saída e dos erros causados por cada parâmetro, que são minimizados através do algoritmo de propagação direta (*forward-propagation*). Para determinar a classe de uma amostra de entrada, cada camada da rede neural calcula as equações

$$z_{(i)} = w_{(i)} \cdot y_{(i-1)} + b_{(i)} \quad (2) \quad y_{(i)} = \varphi(z_{(i)}) \quad (3) \quad \varphi(z) = \frac{1}{1 + e^{-z}}, \quad (4)$$

em que $y_{(i)}$ é o vetor que determina a saída da camada i , $w_{(i)}$ é o vetor de peso que leva a camada $i - 1$ para a camada i , $b_{(i)}$ é o vetor de parâmetros de polarização (*bias*) da

camada i e $z_{(i)}$ a função de transferência da camada i . A função $\varphi(z)$ é uma função de ativação, sendo geralmente utilizada a função *Sigmoid*, representada na Equação 4. A função de ativação tem como objetivo capturar uma relação não-linear entre as entradas e convertê-las em uma saída mais útil. Para valores elevados de z , $\varphi(z)$ retorna 1 e para valores baixos $\varphi(z)$ retorna 0. Portanto, a camada de saída fornece o grau de associação de cada classe, entre 0 e 1.

O **perceptron**, também chamado de neurônio artificial, é a forma mais simples de uma rede neural do tipo direta (*feed-forward*), sem realimentação, em que a informação sempre se move no mesmo sentido e não há laços (*loops*). Já o **perceptron multicamadas** (*Multi-Layer Perceptron* - MLP) é uma rede neural também do tipo sem realimentação (*feed-forward*) composta de múltiplas camadas de neurônios. Na maioria das aplicações, a função de ativação do MLP é a função *sigmoid* e o método de aprendizado é o de retropropagação (*backpropagation*). O método de retropropagação consiste em ajustar os pesos de cada neurônio da rede neural de acordo com o gradiente de uma função de perda (*loss function*) arbitrária. O objetivo é minimizar o erro de predição dos neurônios em relação à saída real para aumentar a acurácia da classificação. Esse processo é repetido inúmeras vezes até que o erro seja suficientemente pequeno. A função de perda mais comumente utilizada na literatura é a média da distância Euclidiana entre a saída real e a predita expressa por

$$E = \frac{1}{2n} \sum_x ||y(x) - y'(x)||^2, \quad (5)$$

em que n é o número de entradas no conjunto de treinamento, x é a entrada avaliada e $y(x)$ e $y'(x)$ são, respectivamente, os valores de saída real e preditos. O ajuste dos pesos de um neurônio j da camada i é feito pela minimização da função perda, geralmente através do método do gradiente descendente

$$\Delta w_{ij} = -\eta \frac{\partial E}{\partial w_{ij}}, \quad (6)$$

em que η é um parâmetro de ajuste do modelo que controla a velocidade de atualização dos pesos w_{ij} . Os erros E_i de cada camada são retropropagados e os pesos ajustados de acordo com as derivadas parciais em relação a cada neurônio.

A Figura 3.20 mostra um exemplo de uma rede neural artificial MLP, composta por uma camada de entrada de três neurônios, que representam as características, duas camadas ocultas de quatro neurônios cada e uma camada de saída de dois neurônios, que representam as classes normal e ameaça. As redes neurais artificiais (ANN) têm a vantagem de aprender e modelar relacionamentos não-lineares e complexos. Embora um único neurônio não pareça vantajoso, conectar muitos neurônios de maneira adequada pode criar um modelo de aprendizado altamente eficaz. O poder das ANN é baseado no Teorema da Aproximação Universal, que afirma que a partir de funções de ativação, como *Sigmoid*, Tangente Hiperbólica, Linear Retificada etc., uma ANN pode modelar qualquer função contínua dados os requisitos de tamanho e de estrutura.

A **máquina de vetores de suporte** (*Support Vector Machine* - SVM) é um classificador binário, baseado no conceito de um plano de decisão que define os limites de

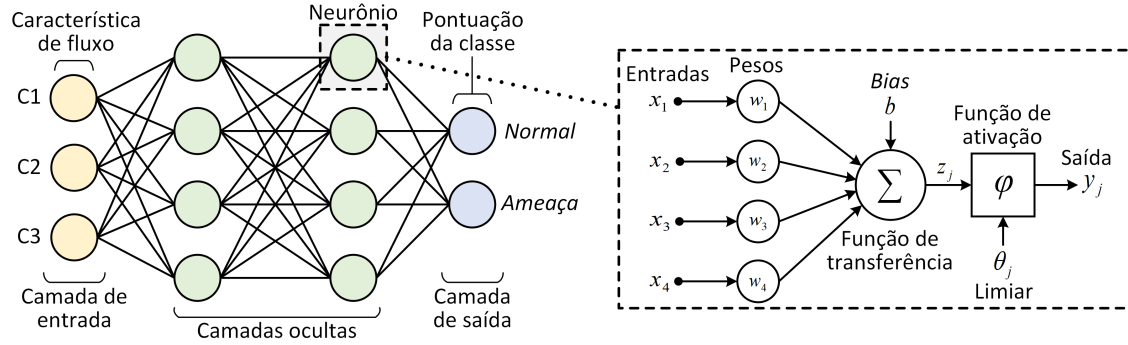


Figura 3.20. Rede neural artificial (ANN) com dois neurônios na camada de saída que representam as classes normal e ataque. Cada neurônio da rede neural realiza uma função, denominada função de ativação, sobre a soma ponderada de suas entradas. A saída desta função é propagada para os neurônios da camada seguinte.

decisão. Basicamente, a SVM classifica através da construção de um hiperplano em um espaço multidimensional que divide diferentes classes. Um algoritmo iterativo definido por uma função *kernel* minimiza uma função de erro, encontrando a melhor separação do hiperplano. Dessa forma, a SVM encontra o hiperplano de separação máxima, ou seja, o hiperplano que maximiza a distância entre as classes ou a margem. Uma ilustração desta separação é apresentada na Figura 3.21(a).

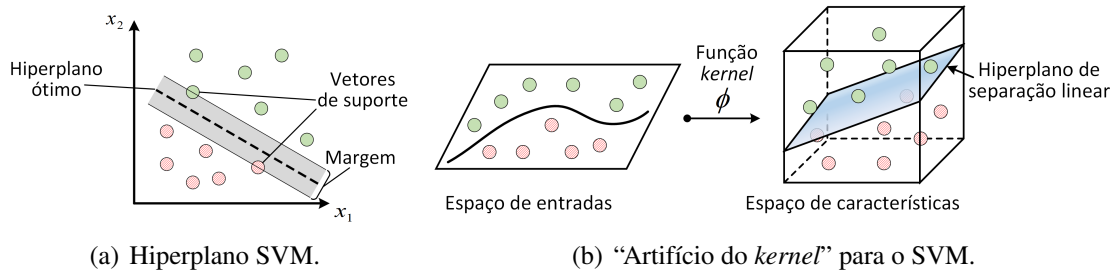


Figura 3.21. a) Definição do hiperplano de separação ótima entre classes binárias no algoritmo SVM. b) Um artifício para obter uma separação linear a partir de um conjunto não-linear consiste em usar uma função *kernel* que mapeia o espaço de entrada no espaço de características.

Para uma classificação multiclasse, divide-se o problema em subproblemas de classificação binária. Assumindo como exemplo um problema com três classes, normal, negação de serviço (DoS) e varredura, isto significa, encontrar os hiperplanos de separação ótima entre: normal e não normal; DoS e não-DoS; e varredura e não-varredura. Uma vez obtido o resultado de cada classe, escolhe-se aquela com maior pontuação. A pontuação do classificador de uma amostra x é a distância de x para os limites de decisão, que vai de $-\infty$ a $+\infty$. A classificação do classificador é dada por

$$f(x) = \sum_{j=1}^n \alpha_j y_j G(x_j, x) + b, \quad (7)$$

em que $(\alpha_1, \dots, \alpha_n)$ são os parâmetros estimados do algoritmo SVM e $G(x_j, x)$ é a função

kernel utilizada. Se não for possível separar os valores de entrada através de um hiperplano linear, uma estratégia é utilizar o “artifício de *kernel*” (*kernel-trick*), como mostra a Figura 3.21(b). O artifício de *kernel* utiliza uma função que leva o espaço de entrada dimensional n para um de dimensão maior m , tal que $m > n$, em que seja possível separar os valores usando um hiperplano. Os *kernels* mais utilizados são Polinomial, *Radial Basis Function* (RBF) e Sigmoid. A SVM é capaz de evitar o sobreajuste (*overfitting*). Uma vez que o hiperplano é encontrado, a maioria dos dados que não sejam os vetores de suporte, que são os pontos mais próximos do limite, tornam-se redundantes. Isso significa que mudanças pequenas nos dados não podem afetar o hiperplano.

A Tabela 3.4 mostra a acurácia de todos os métodos de classificação de ameaças com treinamento prévio apresentados. Além disso, essa tabela detalha a precisão e a sensibilidade do tráfego legítimo e malicioso. Como o principal objetivo da detecção de ameaças é acionar contramedidas, essas métricas mostram o número de alertas verdadeiros e falsos positivos que acionariam as medidas de segurança. Como mostra a Tabela 3.4, bons resultados para a classe normal foram obtidos, geralmente acima de 90%, tanto na precisão quanto na sensibilidade. Consequentemente, isso garante um baixo nível de falsos positivos que resultariam em bloqueio de tráfego legítimo. Em relação à classe de ameaça, para o conjunto de dados NetOp, os resultados de precisão são menores, porque o conjunto de dados é composto na maior parte de amostras legítimas. Portanto, ao avaliar o número absoluto de amostras normais classificadas como ameaças, elas têm um impacto negativo na precisão. No entanto, conforme mostrado nos resultados da classe normal, eles não resultam em muitos falsos positivos. Levando isso em consideração, a medida mais interessante para avaliar a classe de ameaça é a sensibilidade que mede a taxa de detecção de ameaças, ou seja, o número de ameaças classificadas corretamente entre todas as ameaças reais nos conjuntos de dados. Vale destacar que o SVM obteve a melhor taxa de detecção, obtendo uma sensibilidade na classe de ameaça acima de 87% nos dois conjuntos de dados. O SVM é um classificador robusto, que maximiza a margem para o hiperplano de decisão.

Tabela 3.4. Resumo da classificação de ameaças para os algoritmos Árvore de Decisão, Rede Neurais e SVM.

Algoritmo	Conj. Dados	Acurácia	Normal		Ameaça	
			Precisão	Sens.	Precisão	Sens.
Árvore de Decisão	GTA	80.6%	80.7%	94.6%	80.1%	48.8%
	NetOp	92.8%	97.3%	94.8%	51.5%	67.9%
Rede Neural	GTA	96.0%	94.8%	98.8%	97.9%	91.5%
	NetOp	95.1%	97.5%	97.2%	66.7%	69.2%
SVM	GTA	96.3%	95.4%	98.8%	98.0%	92.4%
	NetOp	95.8%	96.5%	98.9%	66.9%	87.2%

3.4.4. Detecção de Ameaças Inéditas em Tempo Real a partir de Treinamento Adaptativo

Os atacantes criam novos ataques (*zero-days attacks*) e alteram os ataques convencionais de forma a ludibriarem e passarem despercebidos por sistemas de detecção

de intrusão baseados em assinatura ou métodos de classificação em tempo diferenciado (*offline*). Para aumentar a robustez da detecção de ameaças, este capítulo apresenta um esquema adaptativo de coleta de dados capaz de aprender ataques novos e detectar ameaças que se modificam com o tempo [Lobato et al. 2018]. Este esquema de coleta de dados proposto garante adaptabilidade tanto para o comportamento legítimo como para o malicioso. Os métodos se adaptam a mudanças aceitáveis no uso da rede e aprendem novos ataques que são realizados nos *honeypots*. Assim, são propostos três métodos que adaptam seus parâmetros em tempo real à medida que os fluxos de dados chegam. Os dois primeiros algoritmos de detecção são algoritmos de classificação em linha (*online*) que visam classificar os ataques com base na captura de comportamento de ataques conhecidos pelo *honeypots*, enquanto o outro é um algoritmo de detecção de anomalia. Os três algoritmos implementados possuem a capacidade de detectar novas ameaças.

Os métodos de classificação em linha (*online*) requerem dados rotulados. Assim, no método de detecção proposto, todos os dados provenientes dos potes de mel (*honeypots*) são rotulados como ameaças, uma vez que não há nenhum serviço real sendo oferecido nos *honeypots* e todo o acesso é destinado a explorar uma vulnerabilidade intencionalmente instalada ou desconhecida *a priori*. Portanto, todos os fluxos que chegam aos *honeypots* serão usados pelos algoritmos para atualizar os parâmetros rotulados como ameaças. Essa realimentação de dados em tempo real garante comportamento adaptativo para a detecção de ameaças. Então, sempre que um atacante executa um novo ataque ou muda seu comportamento, os modelos de classificação são atualizados. Os métodos de classificação propostos assumem como tráfego benigno os fluxos recebidos nos sensores de tráfego na rede durante o tempo de configuração, que é um período em que o uso da rede é monitorado para garantir que todos os fluxos sejam legítimos. Após esse tempo, caso um fluxo proveniente dos sensores de rede seja classificado como ameaça, os parâmetros no algoritmo não são atualizados. Se o fluxo for considerado normal os parâmetros do algoritmo são atualizados adaptando-se às mudanças de comportamento normal da rede.

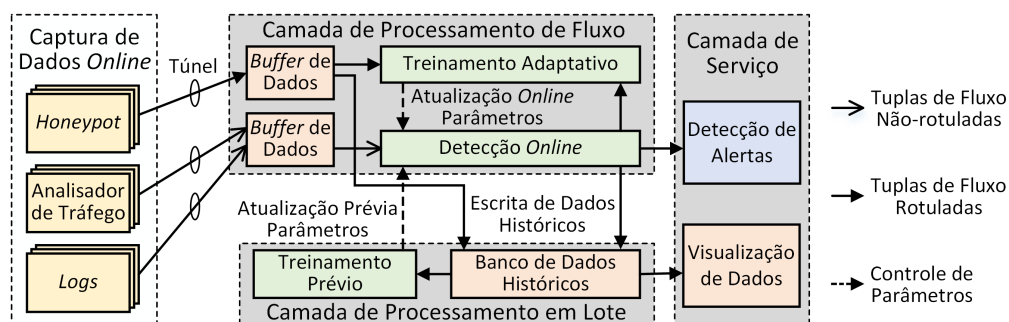


Figura 3.22. Arquitetura para a detecção em tempo real com treinamento adaptativo para proteção contra ataques desconhecidos.

A arquitetura mostrada na Figura 3.22 exemplifica um mecanismo de detecção de intrusão que processa dados em fluxo para adaptar os modelos treinados e aumentar a acurácia da detecção. Os dados da rede são coletados e enviados para um *buffer*. O algoritmo de aprendizado de máquina classifica o dado. Caso a amostra seja considerada como tráfego benigno é, então, usada como entrada na rotina de treinamento do algoritmo de aprendizado de máquina rotulada como benigna. Já as amostras de ameaças são coleta-

das em *honeypots* espalhados pela rede e são enviadas diretamente à rotina de treinamento rotuladas como ameaças. A cada amostra recebida, os algoritmos de aprendizado de máquina incrementais executam um passo de treinamento.

3.4.4.1. O Treinamento Adaptativo para Algoritmos de Detecção de Ameaças

O **algoritmo gradiente estocástico descendente com momento** é uma aproximação do algoritmo do gradiente estocástico descendente (*Stochastic Gradient Descent* - SGD), na qual o gradiente é aproximado por uma única amostra. Na aplicação de detecção de ameaças são consideradas duas classes: normal e ameaça. Portanto, a função Sigmoid, expressa por

$$h_{\theta}(\mathbf{x}) = \frac{1}{1 + e^{-\theta^T \mathbf{x}}} \quad (8)$$

é utilizada para executar a regressão logística. Na função Sigmoid, baixos valores de produto dos parâmetros θ^T vezes o vetor da característica da amostra $\mathbf{x}$ retornam 0, enquanto valores altos retornam 1. Quando uma nova amostra $\mathbf{x}_{(i)}$ chega, o método avalia a função Sigmoid e retorna um para $h_{\theta}(\mathbf{x}_{(i)})$ maiores que 0.5 e zero, caso contrário. Essa decisão apresenta um custo associado, com base na classe real da amostra $y_{(i)}$. A função de custo é definida na Equação 9. Essa função é convexa e o objetivo do algoritmo SGD é encontrar o seu mínimo, expresso por

$$J_{(i)}(\theta) = y_{(i)} \log(h_{\theta}(\mathbf{x}_{(i)})) + (1 - y_{(i)}) \log(1 - h_{\theta}(\mathbf{x}_{(i)})). \quad (9)$$

Quando uma nova amostra chega, o algoritmo dá um passo em direção ao mínimo custo com base no gradiente desta função.

Algoritmo 1: Gradiente Estocástico Descendente com Momento.

Entrada: Características de fluxo de entrada x , Classe y

Saída : Classe Prevista *predict*, Parâmetros do treinamento θ

Inicializar $\theta, \Delta\theta, \alpha, \beta$;

for $i \leftarrow 1$ **to** m **do**

$$h_{\theta}(x_{(i)}) = \frac{1}{1 + e^{-\theta^T x_{(i)}}};$$

$$predict = round(h_{\theta}(x_{(i)}));$$

if $predict == 1$ **and** $y_{(i)} == 0$ **then**

Envia Alerta;

else

$$\theta = \theta - \alpha \nabla J_{(i)}(\theta) + \beta \Delta\theta;$$

$$\Delta\theta = \alpha \nabla J_{(i)}(\theta);$$

end

end

O Algoritmo 1 mostra a implementação do algoritmo do SGD. A cada amostra recebida do vetor $\mathbf{x}_{(i)}$ o método determina a classe $y_{(i)}$ baseado no tipo da origem. Se

a amostra vier de um *honeypot* o rótulo é 1, enquanto se vier de uma sonda de tráfego o rótulo é 0. Se a amostra vem de um analisador de tráfego e o SGD prevê como uma ameaça, o algoritmo envia um alerta. Caso contrário, atualiza os parâmetros com base no gradiente da função custo. O termo $\Delta\theta$ é o momento e tem o valor da atualização anterior do parâmetro. No contexto do SGD, este termo considera o movimento passado ao atualizar os parâmetros θ . Os parâmetros α e β são periodicamente atualizados em tempo diferenciado, *offline*, com base no custo histórico de cada amostra.

A **máquina de vetores de suporte incremental** (*Incremental Support Vector Machine* - ISVM) é um classificador binário com base no conceito de plano de decisão que define os limites das classes. Um hiperplano construído em um espaço multidimensional divide os dados. O algoritmo SVM *online* usa uma aproximação de margem suave com a função convexa de perda de articulação (*hinge loss*), dada por

$$\max \{0, 1 - y\theta^T x\}. \quad (10)$$

O objetivo deste algoritmo é minimizar a função de perda. Assim como o algoritmo anterior, o método determina a classe $y_{(i)}$ com base na origem. Se vier de um *honeypot*, o rótulo é 1, enquanto que se ele vem de um sensor de rede, o rótulo é -1 . Novamente, quando um tráfego de uma sonda é classificado como ameaça, o método envia um alerta e não atualiza o modelo. O Algoritmo 2 mostra a implementação do SVM *online*. Os parâmetros α , λ são periodicamente atualizados com base na avaliação da função de perda sobre as amostras históricas.

Algoritmo 2: Máquina de Vetores de Suporte Incremental.

Entrada: Características de fluxo de entrada x , Classe y

Saída : Classe prevista *predict*, parâmetros do treinamento θ

Inicializar θ , α , λ ;

for $i \leftarrow 1$ **to** m **do**

$predict = \text{sign}(\theta^T x_{(i)});$

if $predict == 1$ **and** $y_{(i)} == -1$ **then**

 Envia Alerta;

else

if $y_{(i)} \theta^T x_{(i)} > 1$ **then**

$\nabla_{(i)} = \theta;$

else

$\nabla_{(i)} = -\lambda y_{(i)} x_{(i)};$

end

$\theta = \theta - \alpha \nabla_{(i)}$

end

end

O algoritmo de **detecção de anomalia pela distribuição normal** utiliza a distância da característica da amostra para a média de uma distribuição normal para detectar anomalias. Dessa forma, as anomalias são detectadas quando a distância de uma amostra normal do conjunto de dados de treinamento para a média é maior que um limiar vezes

a variância para pelo menos uma das características. A implementação em tempo real realiza a detecção de anomalia à medida que os dados de transmissão estão chegando. Portanto, uma anomalia é detectada se pelo menos uma das seguintes condições

$$X_j > \mu_j + \text{limiar} * \sigma_j^2 \quad (11) \quad X_j < \mu_j - \text{limiar} * \sigma_j^2 \quad (12)$$

é verdadeira para pelo menos uma característica j , levando em consideração os μ_j e as variâncias σ_j^2 calculadas no treinamento. Além disso, quando uma nova amostra chega e não é detectada como uma anomalia pelas condições (11) e (12), a média μ_j e a variância σ_j^2 de cada característica definida por

$$\mu_j = \frac{1}{N} \sum_{i=1}^N X_j \quad (13) \quad \sigma_j^2 = \frac{1}{N} \sum_{i=1}^N (X_j - \mu_j)^2 \quad (14)$$

são atualizados, considerando a nova amostra. Os valores atuais do total de amostras N e sua soma são armazenados na plataforma e incrementados quando uma nova amostra chega, considerando cada recurso X_j . Consequentemente, os parâmetros de distribuição normais são sempre atualizados por amostras consideradas legítimas, garantindo adaptabilidade.

O algoritmo de **detecção de anomalia pela série temporal de entropia** identifica uma anomalia pela análise do valor de entropia de uma janela deslizante de fluxos. A entropia de amostra, expressa por

$$H(X) = - \sum_{i=1}^N \left(\frac{n_i}{S} \right) \log_2 \left(\frac{n_i}{S} \right), \quad (15)$$

indica o grau de concentração ou dispersão de uma característica, em que S é o número total de observações, n_i é o número de observações dentro do intervalo i de valores e N é o número de intervalos. Quando todos os valores estão concentrados em um intervalo, $H(X)$ é igual a zero e quando cada valor está em uma faixa diferente i , o valor de $H(X)$ é $\log_2(N)$. Então, dada uma série de observações X , a entropia de amostra resume o nível de concentração em um único valor. Foi definida uma janela deslizante de 40 fluxos e calculado o valor de $H(X)$ para cada uma dessas janelas, gerando as séries temporais. Outro parâmetro determinado na fase de configuração é o intervalo contendo a maioria das amostras. Estes parâmetros são atualizados conforme novas amostras chegam. Os valores normais de entropia do tráfego tendem a ser concentrados e o valor mais frequente tende a estar no centro. Assim, o esquema de detecção de ameaça por anomalia proposto define um limiar que determina a distância aceita da entropia $H(X)$ para o intervalo mais frequente [Lobato et al. 2018].

3.4.4.2. Resultados de Detecção em Tempo Real com Treinamento Adaptativo

Os algoritmos de treinamento adaptativo e os algoritmos de detecção de anomalias foram avaliados para dois conjuntos de dados: o conjunto de dados desenvolvido no GTA/UFRJ e o conjunto de dados da operadora de telecomunicações (NetOp). A Figura 3.23 mostra o comportamento da precisão ao longo do tempo para cada amostra

recebida para a aplicação dos algoritmos de aprendizado de máquina com treinamento adaptativo, SGD e SVM. As Figuras 3.23(a) e 3.23(b) mostram os resultados de cada conjunto de dados. As precisões para os algoritmos implementados estabilizam em valores altos, sempre acima de 90%, mesmo quando ameaças desconhecidas aparecem nos conjuntos de dados. No início da análise, os métodos não conhecem ameaças. No entanto, à medida que os dados do pote de mel (*honeypot*) chegam, os métodos aprendem as ameaças e alcançam altas taxas de detecção de ameaças. Como os métodos de detecção não conhecem nenhuma das 16 ameaças no conjunto de dados do GTA/UFRJ e as 13 ameaças no conjunto de dados NetOp, essas ameaças são consideradas ameaças inéditas (*zero-day threats*) que são aprendidas e detectadas em linha (*online*) à medida que novas amostras chegam.

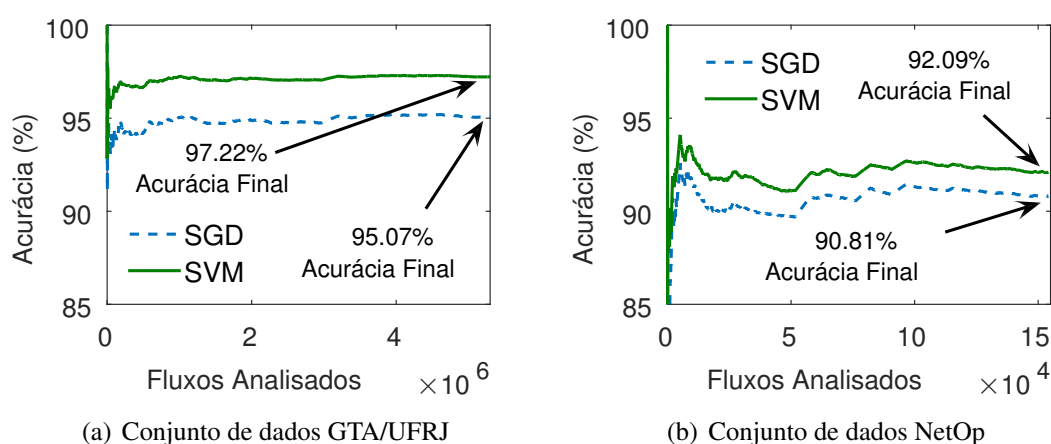


Figura 3.23. Acurácia para ambos os esquemas propostos conforme as amostras chegam. Em ambos os conjuntos de dados, a acurácia permanece estável mesmo com novos ataques e mudanças de comportamento de uso legítimo.

Como os dois esquemas de detecção de anomalia não são supervisionados e modelam o comportamento normal do usuário, são utilizados apenas os dados do *honeypot* para avaliar o desempenho do algoritmo. Os parâmetros são atualizados durante o tempo de configuração com base apenas nos dados capturados pelos analisadores de tráfego. Após esse período, se uma amostra for considerada normal, os parâmetros serão atualizados. Por outro lado, se for considerado malicioso, um alerta é enviado e os parâmetros dos algoritmos não são atualizados. Para avaliar os algoritmos, foi utilizado 70 % dos dados normais no tempo de *setup* e 30 % para avaliar os falsos positivos. A taxa de detecção de ataques é obtida com base em todas as ameaças nos conjuntos de dados.

A Figura 3.24 mostra os resultados do algoritmo de detecção de anomalia pela distribuição normal para diferentes valores limiares. Para o conjunto de dados do GTA/UFRJ, observa-se que selecionando um limiar de 2, o algoritmo apresenta um compromisso muito bom entre 96,4 % da taxa de detecção e 5,6 % da taxa de falsos positivos. O mesmo comportamento é apresentado para o conjunto de dados NetOp, com um bom compromisso com um limiar de 3, apresentando 88 % de taxa de detecção de ameaças e 6,7 % de taxa de falsos positivos. O parâmetro do limiar pode ser ajustado de acordo com a relação entre verdadeiros positivos e falso positivos desejada. Para redes com requisitos de segurança altos, esse limiar deve ser pequeno, resultando em alta taxa de detecção ao

custo de uma taxa de falsos positivos mais alta [Lobato et al. 2018].

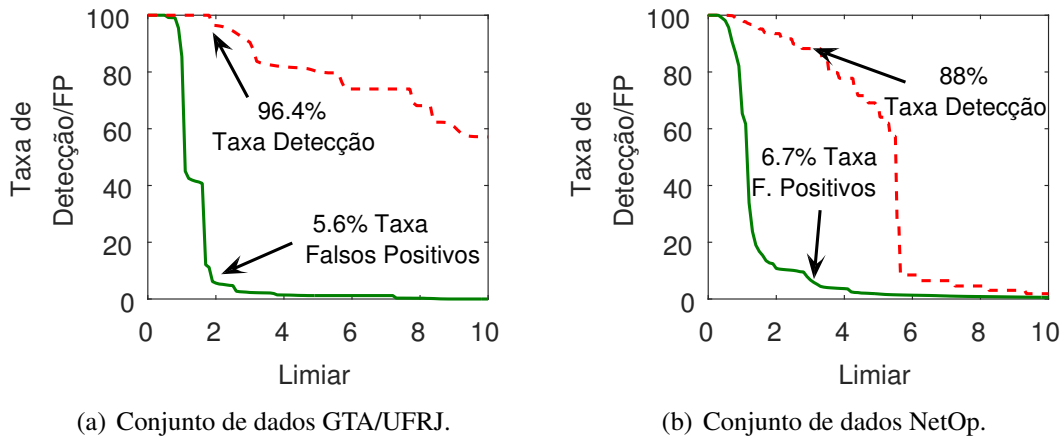


Figura 3.24. Detecção de anomalia pela distribuição normal: análise do compromisso entre a taxa de detecção e a taxa de falsos positivos ao variar o parâmetro de limiar.

A Figura 3.25 mostra os resultados da detecção de anomalia por série temporal de entropia para diferentes limiares. Para limiares pequenos, a taxa de detecção é muito alta, resultando também em uma alta taxa de falsos positivos. Para valores baixos, ocorre o inverso, uma baixa taxa de falso positivo a um custo também de baixa taxa de detecção. No entanto, existem valores limiares que apresentam um bom equilíbrio entre essas taxas. Para o conjunto de dados GTA/UFRJ, o limiar de 1,3 resulta em 86,8 % de taxa de detecção de ataques com apenas 2,8 % de taxa de falsos positivos, enquanto para o conjunto de dados NetOp, o limiar 3.0 obtém uma baixa taxa de falsos positivos de 4,7 % com uma taxa de detecção de ataque notavelmente alta de 97,1 %.

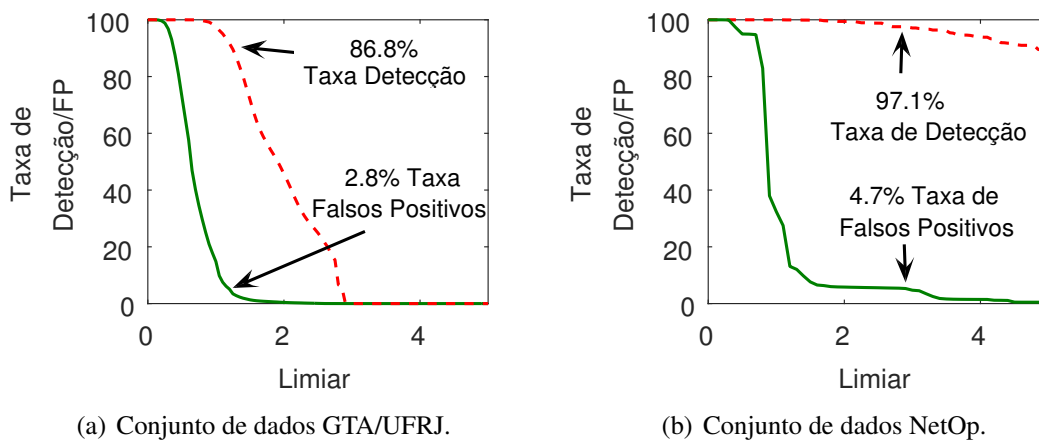


Figura 3.25. Detecção de anomalia por série temporal de entropia: análise do compromisso entre a taxa de detecção e a taxa de falsos positivos ao variar o parâmetro de limiar.

3.5. Considerações Finais e Problemas em Aberto

O uso de plataformas de processamento distribuídos de fluxo representa uma nova direção na pesquisa de detecção de ameaças. O paralelismo e a possibilidade de montar grafos de processamento permitem que métodos de detecção complexos rodem simultaneamente e de forma escalável no volume de grandes massas de dados e em tempo real. Este capítulo apresentou o problema de detecção de ameaça em tempo real baseada em aprendizado de máquina usando plataformas de processamento de fluxo de código aberto. As plataformas de processamento distribuído Apache Storm, Apache Flink e Apache Spark são apresentadas e o seu desempenho foi avaliado. Foram introduzidos conceitos de conjunto de dados e mostrados os conjuntos de dados usados nas avaliações de desempenho. Técnicas usadas para redução de dimensionalidade e seleção de características foram apresentadas assim como as principais métricas sistemas de aprendizado de máquina em detecção de intrusão. O foco do capítulo foram as arquiteturas de processamento distribuído de fluxo que seguem a arquitetura lambda, adaptativas e de processamento incremental.

A detecção de anomalias através do processamento distribuído de fluxos de dados é um desafio, pois os dados são gerados nas redes de forma contínua e com alta velocidade. Algoritmos que processam fluxos de dados proveem soluções aproximadas e uma resposta rápida usando poucos recursos de memória. A resposta aproximada pode ser bastante útil para uma primeira análise da anomalia, mas a medida em que se exige uma margem de erro menor, os recursos computacionais exigidos aumentam, assim como as dimensões das entradas de dados também aumentam. As tendências das pesquisas atuais em processamento de fluxos de dados envolvem a busca por algoritmos e estruturas de dados com limites estreitos de erro. Outro problema importante ainda em aberto envolve o processamento de fluxo sem invadir a privacidade dos usuários. Cada vez mais as aplicações usam protocolos seguros nos quais os dados são criptografados. Como fazer análises em dados criptografados ainda é um desafio. Há, ainda, questões semânticas em relação aos operadores aplicados sobre os dados.

A taxa de geração dos fluxos varia, podendo atingir elevados valores que viriam por estourar a fila de processamento dos sistemas de detecção de ameaças. Caso isso ocorra, ataques passariam despercebidos, o que representa uma ameaça real à rede e um problema em aberto. O uso de técnicas de mineração de dados em fluxo visa diminuir a quantidade de recursos necessários para o processamento de dados. Em cenários de sobrecarga, técnicas como amostragem, descarte de carga (*Load Shedding*), estruturas de dados de Sinopse (*Synopsis*), entre outras, podem melhorar a segurança, ao diminuir a representação dos dados ao invés de simplesmente descartar as amostras quando a fila estiver cheia [Gaber et al. 2005, Gama and Rodrigues 2007].

A arquitetura *lambda* mostrou-se acurada e rápida, permitindo que os dados em fluxo sejam tratados em tempo quase real, pois o processamento sobre cada amostra é reduzido, já que o treinamento e adaptação de modelos são realizados em tempo diferenciado. Por depender de treinamento prévio, a arquitetura lambda não é capaz de identificar ameaças inéditas. A principal vantagem da arquitetura de classificação de tráfego com aprendizado adaptativo é identificar ameaças inéditas (*zero-day threats*), permitindo um aperfeiçoamento gradual do modelo conforme novas amostras rotuladas surgem. Por

depende de dados rotulados recebidos em tempo real, usou-se potes de mel (*honeypots*) na rede para capturar o comportamento das ameaças. No entanto, introduz um custo maior de processamento por amostra ao obter um passo de adaptação para os algoritmos incrementais.

Há iniciativas tanto nacionais como internacionais em plataformas para detecção de ameaças usando aprendizado de máquina, tais como:

- CATRACA¹⁸ é uma ferramenta de processamento distribuído de fluxos de dados baseada na plataforma Spark Streaming que realiza detecção de ameaças em tempo real. A ferramenta é desenvolvida pelo Grupo de Teleinformática e Automação / UFRJ e aplica a arquitetura lambda de processamento de fluxo. Os pacotes na rede são abstraídos em fluxos identificados pela 5-tupla (IP de origem e destino, portas de origem e destino e protocolo de transporte) dentro de janelas saltitantes de 2 segundos. As estatísticas dos fluxos são carregadas no sistema de mensagens Kafka que as armazena até serem consumidas na forma de fluxos discretos (D-Streams) pela Spark Streaming. Na Spark Streaming, executam-se algoritmos de seleção de características, para fazer a filtragem das características mais importantes, e o algoritmo de classificação por árvores de decisão que se mostrou o mais adequado para a identificação de ameaças em rede [Andreoni Lopez et al. 2017d, Andreoni Lopez et al. 2017c]. Os fluxos de dados ainda são enriquecidos com informações de geolocalização dos IPs de origem e destino. A ferramenta conta ainda com uma interface de visualização dos resultados da classificação baseada na plataforma Kibana da pilha Elasticsearch¹⁹. Em caso de detecção de ameaça, a contramedida aplicada aos fluxos considerados maliciosos é o bloqueio imediato dos fluxos.
- Apache Metron²⁰ é um arcabouço de análise de segurança baseado no processamento de grandes massas de dados. Portanto, sua arquitetura consiste nas camadas de aquisição, consumo, processamento distribuído, enriquecimento, armazenamento e visualização dos dados. A ideia chave desse arcabouço é permitir a correlação de eventos de segurança originados de fontes distintas. Para tanto, o arcabouço emprega sensores na rede como fonte de dados, registros de ações (*logs*) de elementos ativos de segurança em rede e fontes de telemetria. O arcabouço se vale ainda de uma base histórica de ameaças de rede proveniente da Cisco;
- Apache Spot²¹ é um projeto semelhante ao Apache Metron ainda em incubação. O Apache Spot utiliza técnicas de telemetria e aprendizado de máquinas para análise de fluxo e pacotes a fim de detectar ameaças. Os criadores afirmam que a grande diferença com o Apache Metron é a capacidade de utilizar modelos comuns de dados abertos para rede;
- Stream4Flow²² utiliza o Apache Spark com a pilha ElasticStack para fazer monitoramento de redes. O protótipo serve como visualização dos parâmetros da rede.

¹⁸<https://www.gta.ufrj.br/catraca>.

¹⁹<https://www.elastic.co/>.

²⁰<http://metron.apache.org/>

²¹<http://spot.incubator.apache.org>

²²<https://github.com/CSIRT-MU/Stream4Flow>

Stream4Flow, no entanto, não possui nenhuma inteligência para realizar detecção de anomalias.

- Hogzilla²³ é um sistema de detecção de intrusão (IDS) com suporte para Snort, SFlows, GrayLog, Apache Spark, HBase e libnDPI, que provê detecção de anomalias em rede. Hogzilla permite também realizar a visualização do tráfego da rede.

Os algoritmos de aprendizado incremental atualizam as estatísticas nas quais baseiam a classificação com a chegada de cada nova amostra. Contudo, no caso de uma mudança no comportamento estacionário das estatísticas das características das amostras, a classificação pode apresentar baixa acurácia, chamado de *concept drift*. Um tema de pesquisa importante é o estudo e desenvolvimento de algoritmos de aprendizado de máquina resistentes ao *concept drift*.

Uma nova abordagem é de se realizar o aprendizado incremental por agregados de classificadores [Polikar et al. 2001], cuja a ideia central é realizar a classificação através de diversos classificadores de baixa acurácia e alta variabilidade e, então, realizar a classificação final através de um esquema de votação. A alta variabilidade dos classificadores permite que o agregado de classificação reaja mais prontamente a mudanças no processo de chegada de novas amostras. O aglomerado de classificadores é, então, um possível caminho futuro para o desenvolvimento de sistemas de detecção de intrusão baseados em aprendizado de máquina.

Referências

- [Abadi et al. 2005] Abadi, D. J., Ahmad, Y., Balazinska, M., Çetintemel, U., Cherniack, M., Hwang, J.-H., Lindner, W., Maskey, A., Rasin, A., Ryvkina, E., Tatbul, N., Xing, Y., and Zdonik, S. B. (2005). The Design of the Borealis Stream Processing Engine. *Cidr*, pages 277–289.
- [Akidau et al. 2013] Akidau, T., Balikov, A., Bekiroglu, K., Chernyak, S., Haberman, J., Lax, R., McVeety, S., Mills, D., Nordstrom, P., and Whittle, S. (2013). MillWheel: Fault-tolerant Stream Processing at Internet Scale. *Proc. VLDB Endow.*, 6(11):1033–1044.
- [Andreoni Lopez et al. 2016a] Andreoni Lopez, M., Ferrazani Mattos, D., and Duarte, O. C. M. B. (2016a). An elastic intrusion detection system for software networks. *Annales des Telecommunications/Annals of Telecommunications*, 71(11-12):595–605.
- [Andreoni Lopez et al. 2016b] Andreoni Lopez, M., Lobato, A. G. P., and Duarte, O. C. M. B. (2016b). A Performance Comparison of Open-Source Stream Processing Platforms. In *IEEE GLOBECOM*, pages 1–6, Washington, USA. IEEE.
- [Andreoni Lopez et al. 2016c] Andreoni Lopez, M., Lobato, A. G. P., and Duarte, O. C. M. B. (2016c). Monitoramento de Tráfego e Detecção de Ameaças por Sistemas Distribuídos de Processamento de Fluxos: uma Análise de Desempenho. *XXI Workshop de Gerência e Operação de Redes e Serviços (WGRS) do SBRC'2016*, pages 103–116.

²³<http://ids-hogzilla.org/>

- [Andreoni Lopez et al. 2018] Andreoni Lopez, M., Lobato, A. G. P., Duarte, O. C. M. B., and Pujolle, G. (2018). An evaluation of a virtual network function for real-time threat detection using stream processing. In *IEEE Fourth International Conference on Mobile and Secure Services (MobiSecServ)*, pages 1–5.
- [Andreoni Lopez et al. 2017a] Andreoni Lopez, M., Lobato, A. G. P., Mattos, D. M. F., Alvarenga, I. D., Duarte, O. C. M. B., and Pujolle, G. (2017a). Um Algoritmo Não Supervisionado e Rápido para Seleção de Características em Classificação de Tráfego. In *XXXV SBRC'2017*, Belém- Pará, PA,.
- [Andreoni Lopez et al. 2017b] Andreoni Lopez, M., Sanz, I. J., Ferrazani Mattos, D. M., Duarte, O. C. M. B., and Pujolle, G. (2017b). CATRACA: uma Ferramenta para Classificação e Análise Tráfego Escalável Baseada em Processamento por Fluxo. In *Salão de Ferramentas do XVII Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais - SBSeg'2017*, pages 788–795.
- [Andreoni Lopez et al. 2017c] Andreoni Lopez, M., Silva, R., Alvarenga, I. D., Mattos, D. M. F., and Duarte, O. C. M. B. (2017c). Coleta e Caracterização de um Conjunto de Dados de Tráfego Real de Redes de Acesso em Banda Larga. In *XXII WGRS'17*.
- [Andreoni Lopez et al. 2017d] Andreoni Lopez, M., Silva, R. S., Alvarenga, I. D., Rebello, G. A. F., Sanz, I. J., Lobato, A. G. P., Mattos, D. M. F., Duarte, O. C. M. B., and Pujolle, G. (2017d). Collecting and characterizing a real broadband access network traffic dataset. In *IEEE/IFIP 1st Cyber Security in Networking Conference (CSNet)*, pages 1–8.
- [Arasu et al. 2004] Arasu, A., Babcock, B., Babu, S., Cieslewicz, J., Datar, M., Ito, K., Motwani, R., Srivastava, U., and Widom, J. (2004). STREAM: The Stanford Data Stream Management System. Technical Report 2004-20, Stanford InfoLab.
- [ARMOR 2018] ARMOR (2018). The black market report 2018-03. <https://www.armor.com/app/uploads/2018/03/2018-Q1-Reports-BlackMarket-DIGITAL.pdf>. Acessado em 20/03/2018.
- [Balazinska et al. 2004] Balazinska, M., Balakrishnan, H., and Stonebraker, M. (2004). Load management and high availability in the Medusa distributed stream processing system. *ACM SIGMOD international conference on management of data*, pages 929–930.
- [Ballard et al. 2014] Ballard, C., Brandt, O., Devaraju, B., Farrell, D., Foster, K., Howard, C., Nicholls, P., Pasricha, A., Rea, R., Schulz, N., and others (2014). IBM InfoSphere Streams. *Accelerating Deployments with Analytic Accelerators, ser. Redbook. IBM*.
- [Batista et al. 2004] Batista, G. E. A. P. A., Prati, R. C., and Monard, M. C. (2004). A study of the behavior of several methods for balancing machine learning training data. *SIGKDD Explor. Newsl.*, 6(1):20–29.

- [Buczak and Guven 2015] Buczak, A. and Guven, E. (2015). A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection. *IEEE Communications Surveys Tutorials*, (99):1–26.
- [Carbone et al. 2015a] Carbone, P., Ewen, S., Haridi, S., Katsifodimos, A., Markl, V., and Tzoumas, K. (2015a). Apache Flink: Unified Stream and Batch Processing in a Single Engine. *Data Engineering*, pages 28–38.
- [Carbone et al. 2015b] Carbone, P., Fóra, G., Ewen, S., Haridi, S., and Tzoumas, K. (2015b). Lightweight Asynchronous Snapshots for Distributed Dataflows. *Computing Research Repository (CoRR)*, abs/1506.0.
- [Cárdenas and Baras 2006] Cárdenas, A. A. and Baras, J. S. (2006). Evaluation of classifiers: Practical considerations for security applications.
- [Cárdenas et al. 2013] Cárdenas, A. A., Manadhata, P. K., and Rajan, S. (2013). Big data analytics for security intelligence. *University of Texas at Dallas@ Cloud Security Alliance*.
- [Carney et al. 2002] Carney, D., Çetintemel, U., Cherniack, M., Convey, C., Lee, S., Seidman, G., Stonebraker, M., Tatbul, N., and Zdonik, S. (2002). Monitoring Streams: A New Class of Data Management Applications. In *28th International Conference on Very Large Data Bases*, pages 215–226.
- [Chandrasekar et al. 2017] Chandrasekar, K., Cleary, G., Cox, O., Lau, H., Nahorneyand, B., Gorman, B. O., O’Brien, D., Wallacen, S., Wood, P., and Wuees, C. (2017). Internet security threat report-symantec corporation,v22. <https://www.symantec.com/content/dam/symantec/docs/reports/istr-22-2017-en.pdf>. Acessado em 20/03/2018.
- [Chandrasekaran et al. 2003] Chandrasekaran, S., Cooper, O., Deshpande, A., Franklin, M. J., Hellerstein, J. M., Hong, W., Krishnamurthy, S., Madden, S. R., Reiss, F., and Shah, M. A. (2003). TelegraphCQ: continuous dataflow processing. In *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data*, page 668. ACM.
- [Chandrashekar and Sahin 2014] Chandrashekar, G. and Sahin, F. (2014). A survey on feature selection methods. *Computers & Electrical Engineering*, 40(1):16–28.
- [Chen et al. 2000] Chen, J., DeWitt, D. J., Tian, F., and Wang, Y. (2000). NiagaraCQ: A scalable continuous query system for internet databases. In *ACM SIGMOD Record*, volume 29, pages 379–390. ACM.
- [Cheng et al. 2010] Cheng, Z., Caverlee, J., and Lee, K. (2010). You Are Where You Tweet: A Content-based Approach to Geo-locating Twitter Users. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management, CIKM ’10*, pages 759–768. ACM.
- [Clay 2015] Clay, P. (2015). A modern threat response framework. *Network Security*, 2015(4):5–10.

- [Costa et al. 2012] Costa, L. H. M. K., de Amorim, M. D., Campista, M. E. M., Rubinstein, M., Florissi, P., and Duarte, O. C. M. B. (2012). Grandes Massas de Dados na Nuvem: Desafios e Técnicas para Inovação. In *SBRC 2012 - Minicursos*.
- [Demers et al. 2007] Demers, A. J., Gehrke, J., Panda, B., Riedewald, M., Sharma, V., White, W. M., and others (2007). Cayuga: A General Purpose Event Monitoring System. In *Proceedings of the Conference on Innovative Data Systems Research*, volume 7, pages 412–422.
- [Domingos 2012] Domingos, P. (2012). A few useful things to know about machine learning. *Commun. ACM*, 55(10):78–87.
- [Fontugne et al. 2010] Fontugne, R., Borgnat, P., Abry, P., and Fukuda, K. (2010). MAWILab: Combining diverse anomaly detectors for automated anomaly labeling and performance benchmarking. In *ACM CoNEXT '10*, Philadelphia, PA.
- [Franklin 2013] Franklin, M. (2013). The Berkeley Data Analytics Stack: Present and future. In *IEEE International Conference on Big Data*, pages 2–3. IEEE.
- [Gaber et al. 2005] Gaber, M. M., Zaslavsky, A., and Krishnaswamy, S. (2005). Mining data streams: A review. *SIGMOD Rec.*, 34(2):18–26.
- [Gaffney and Ulvila 2001] Gaffney, J. E. and Ulvila, J. W. (2001). Evaluation of intrusion detectors: a decision theory approach. In *Proceedings 2001 IEEE Symposium on Security and Privacy. S P 2001*, pages 50–61.
- [Gama and Rodrigues 2007] Gama, J. and Rodrigues, P. P. (2007). Data stream processing. In *Learning from Data Streams*, pages 25–39. Springer.
- [Garcia et al. 2014] Garcia, S., Grill, M., Stiborek, J., and Zunino, A. (2014). An empirical comparison of botnet detection methods. *computers & security*, 45:100–123.
- [Gluhak et al. 2011] Gluhak, A., Krco, S., Nati, M., Pfisterer, D., Mitton, N., and Razafindralambo, T. (2011). A survey on facilities for experimental Internet of things research. *IEEE Communications Magazine*, 49(11):58–67.
- [Gu et al. 2006] Gu, G., Fogla, P., Dagon, D., Lee, W., and Skorić, B. (2006). Measuring intrusion detection capability: An information-theoretic approach. In *Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security, ASIACCS '06*, pages 90–101, New York, NY, USA. ACM.
- [Haines et al. 2001] Haines, J. W., Lippmann, R. P., Fried, D. J., Zissman, M., and Tran, E. (2001). 1999 darpa intrusion detection evaluation: Design and procedures. Technical report, Massachusetts Inst Of Tech Lexington Lincoln Lab.
- [Heidemann and Papadopoulos 2009] Heidemann, J. and Papadopoulos, C. (2009). Uses and challenges for network datasets. In *Conference For Homeland Security, 2009. CATCH'09. Cybersecurity Applications & Technology*, pages 73–82. IEEE.

- [Henke et al. 2011] Henke, M., Santos, C., Nunan, E., Feitosa, E., dos Santos, E., and Souto, E. (2011). Aprendizagem de máquina para segurança em redes de computadores: Métodos e aplicações. *Minicursos do XI Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais (SBSeg 2011)*, 1:53–103.
- [ICA 2017] ICA, I. C. A. (2017). Assessing russian activities and intentions in recent us elections,ica 2017-01d. https://www.dni.gov/files/documents/ICA_2017_01.pdf. Acessado em 20/03/2018.
- [Jiang et al. 2010] Jiang, W., Ravi, V. T., and Agrawal, G. (2010). A Map-Reduce system with an alternate API for multi-core environments. In *Proceedings of the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, pages 84–93. IEEE Computer Society.
- [Kala Karun and Chitharanjan 2013] Kala Karun, A. and Chitharanjan, K. (2013). A review on Hadoop—HDFS infrastructure extensions. In *IEEE Conference on Information & Communication Technologies (ICT)*, pages 132–137. IEEE.
- [Kamburugamuve et al. 2013] Kamburugamuve, S., Fox, G., Leake, D., and Qiu, J. (2013). Survey of distributed stream processing for large stream sources.
- [Lee et al. 1999] Lee, W., Stolfo, S. J., and Mok, K. W. (1999). Mining in a data-flow environment: Experience in network intrusion detection. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 114–124. ACM.
- [Lippmann et al. 2000a] Lippmann, R. P., Fried, D. J., Graf, I., Haines, J. W., Kendall, K. R., McClung, D., Weber, D., Webster, S. E., Wyschogrod, D., Cunningham, R. K., and others (2000a). Evaluating intrusion detection systems: The 1998 DARPA off-line intrusion detection evaluation. In *Proceedings of DARPA Information Survivability Conference and Exposition. DISCEX'00.*, volume 2, pages 12–26. IEEE.
- [Lippmann et al. 2000b] Lippmann, R. P., Fried, D. J., Graf, I., Haines, J. W., Kendall, K. R., McClung, D., Weber, D., Webster, S. E., Wyschogrod, D., Cunningham, R. K., and Zissman, M. A. (2000b). Evaluating intrusion detection systems: the 1998 darpa off-line intrusion detection evaluation. In *DARPA Information Survivability Conference and Exposition, 2000. DISCEX '00. Proceedings*, volume 2, pages 12–26 vol.2.
- [Lobato et al. 2016] Lobato, A. G. P., Andreoni Lopez, M., and Duarte, O. C. M. B. (2016). Um Sistema Acurado de Detecção de Ameaças em Tempo Real por Processamento de Fluxos. In *SBRC'2016*, pages 572–585, Salvador, Bahia.
- [Lobato et al. 2017] Lobato, A. G. P., Andreoni Lopez, M., Rebello, G. A. F., and Duarte, O. C. M. B. (2017). Um Sistema Adaptativo de Detecção e Reação a Ameaças. In *Anais do XVII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais - SBSeg'17*, pages 400–413.
- [Lobato et al. 2018] Lobato, A. G. P., Andreoni Lopez, M., Sanz, I. J., Cárdenas, A., Duarte, O. C. M. B., and Pujolle, G. (2018). An adaptive Real-Time architecture for

- Zero-Day threat detection. In *to be published in IEEE ICC NGNI 2018*, Kansas City, USA.
- [Marz and Warren 2013] Marz, N. and Warren, J. (2013). *Big Data: Principles and Best Practices of Scalable Realtime Data Systems*. Manning Publications Co., Greenwich, CT, USA, 1st edition.
- [Mattos et al. 2018] Mattos, D. M. F., Velloso, P. B., and Duarte, O. C. M. B. (2018). Uma infraestrutura Ágil e efetiva de virtualização de funções de rede para a internet das coisas. In *XXXVI Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos - SBRC'2018*.
- [Mayhew et al. 2015] Mayhew, M., Atighetchi, M., Adler, A., and Greenstadt, R. (2015). Use of machine learning in big data analytics for insider threat detection. In *IEEE Military Communications Conference, MILCOM*, pages 915–922.
- [Mladenić 2006] Mladenić, D. (2006). Feature Selection for Dimensionality Reduction. In Saunders, C., Grobelnik, M., Gunn, S., and Shawe-Taylor, J., editors, *Subspace, Latent Structure and Feature Selection (SLSFS): Statistical and Optimization Perspectives Workshop*, pages 84–102. Springer Berlin Heidelberg, Bohinj, Slovenia.
- [Neumeyer et al. 2010] Neumeyer, L., Robbins, B., Nair, A., and Kesari, A. (2010). S4: Distributed Stream Computing Platform. In *IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 170–177. IEEE.
- [Polikar et al. 2001] Polikar, R., Upda, L., Upda, S. S., and Honavar, V. (2001). Learn++: an incremental learning algorithm for supervised neural networks. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 31(4):497–508.
- [Robnik-Šikonja and Kononenko 2003] Robnik-Šikonja, M. and Kononenko, I. (2003). Theoretical and Empirical Analysis of ReliefF and RReliefF. *Machine Learning*, 53(1/2):23–69.
- [Sanz et al. 2017] Sanz, J. I., Andreoni Lopez, M., Mattos, D. M. F., and Duarte, O. C. M. B. (2017). A Cooperation-Aware Virtual Network Function for Proactive Detection of Distributed Port Scanning. In *IEEE/IFIP 1st Cyber Security in Networking Conference (CSNet'17)*.
- [Schölkopf et al. 1999] Schölkopf, B., Smola, A. J., and Müller, K.-R. (1999). Kernel principal component analysis. In *Advances in kernel methods*, pages 327–352. MIT Press.
- [Shiravi et al. 2012] Shiravi, A., Shiravi, H., Tavallaee, M., and Ghorbani, A. A. (2012). Toward developing a systematic approach to generate benchmark datasets for intrusion detection. *Computers and Security*, 31(3):357 – 374.
- [Sommer and Paxson 2010] Sommer, R. and Paxson, V. (2010). Outside the closed world: On using machine learning for network intrusion detection. In *IEEE Symposium on Security and Privacy (SP)*, pages 305–316. IEEE.

- [Stonebraker et al. 2005] Stonebraker, M., Çetintemel, U., and Zdonik, S. (2005). The 8 requirements of real-time stream processing. *ACM SIGMOD Record*, 34(4):42–47.
- [Stonebraker and Kemnitz 1991] Stonebraker, M. and Kemnitz, G. (1991). The POSTGRES next generation database management system. *Communications of the ACM*, 34(10):78–92.
- [Tavallaee et al. 2009a] Tavallaee, M., Bagheri, E., Lu, W., and Ghorbani, A.-A. (2009a). A detailed analysis of the KDD CUP 99 data set. In *Proceedings of the Second IEEE Symposium on Computational Intelligence for Security and Defence Applications*. IEEE.
- [Tavallaee et al. 2009b] Tavallaee, M., Bagheri, E., Lu, W., and Ghorbani, A. A. (2009b). A detailed analysis of the kdd cup 99 data set. In *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, pages 1–6.
- [Toshniwal et al. 2014] Toshniwal, A., Taneja, S., Shukla, A., Ramasamy, K., Patel, J. M., Kulkarni, S., Jackson, J., Gade, K., Fu, M., Donham, J., Bhagat, N., Mittal, S., and Ryaboy, D. (2014). Storm@Twitter. In *ACM SIGMOD International Conference on Management of Data*, pages 147–156. ACM.
- [Van Der Maaten et al. 2009] Van Der Maaten, L., Postma, E., and den Herik, J. (2009). Dimensionality reduction: a comparative. *Journal of Machine Learning Research*, 10:66–71.
- [Widom 1992] Widom, J. (1992). The Starburst rule system: Language design, implementation, and applications. *IEEE Data Engineering Bulletin*.
- [Zaharia et al. 2013] Zaharia, M., Das, T., Li, H., Hunter, T., Shenker, S., and Stoica, I. (2013). Discretized streams: Fault-tolerant streaming computation at scale. In *XXIV ACM Symposium on Operating Systems Principles*, pages 423–438. ACM.
- [Zhai et al. 2014] Zhai, Y., Ong, Y.-S., and Tsang, I. W. (2014). The emerging "big dimensionality". *Comp. Intell. Mag.*, 9(3):14–26.
- [Zhao et al. 2015] Zhao, S., Chandrashekar, M., Lee, Y., and Medhi, D. (2015). Real-time network anomaly detection system using machine learning. In *11th International Conference on the Design of Reliable Communication Networks (DRCN)*, pages 267–270. IEEE.

Apêndice

A Ferramenta CATRACA e a Atividade Prática

Nesta seção é apresentada a ferramenta CATRACA, uma ferramenta baseada no processamento de fluxos que utiliza algoritmos de aprendizado de máquina para filtrar tráfego benigno de ameaças e caracterizar o comportamento da rede [Andreoni Lopez et al. 2017b]. Um processo de enriquecimento de contexto permite a visualização em tempo real do comportamento da rede assim como a origem e destino das ameaças. O CATRACA é uma função virtual de rede (*Virtual Network Function* - VNF), que permite o encadeamento com outro tipo de funções virtuais, como por exemplo um *firewall*, para a prevenção de ameaças. Outra facilidade provida pelo CATRACA é a capacidade de migração, que objetiva a sua localização o mais perto possível da origem da ameaça a fim de evitar retardos na detecção.

A arquitetura da ferramenta CATRACA, mostrada na Figura 3.26, é composta por três camadas: captura, processamento e visualização. A **camada de captura**, é responsável pela captura dos pacotes, através do espelhamento de tráfego, pela biblioteca *libpcap*. Uma aplicação, escrita em Python e baseada no *flowtbag*²⁴, abstrai os pacotes em fluxos, que são definidos como uma sequência de pacotes que possuem a mesma quintupla IP de origem, IP de destino, porta de origem, porta de destino e protocolo, durante uma janela de tempo. Ao todo, 46 características de cada fluxo são extraídas em tempo real do tráfego que atravessa a VNF e publicadas em uma fila de um serviço produtor/consumidor de mensagens *Kafka*, com baixa latência, para serem consumidos pela camada de processamento.

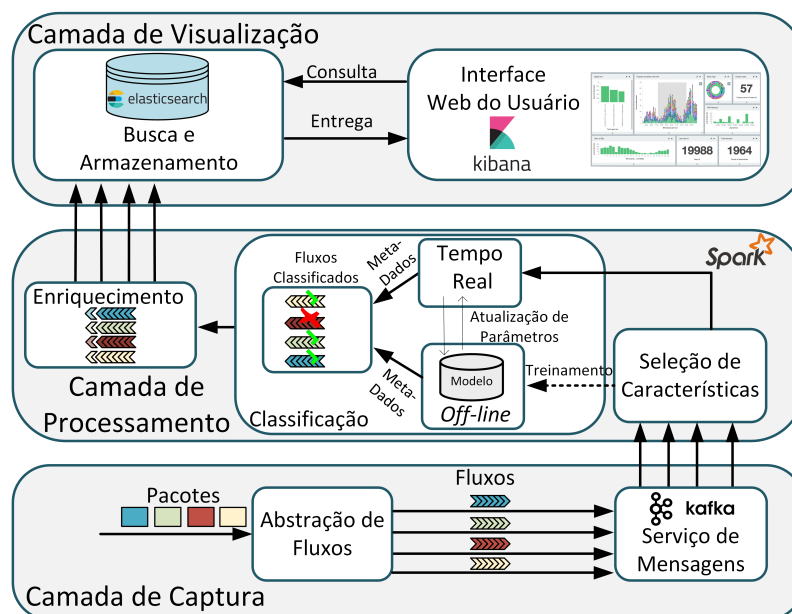


Figura 3.26. A arquitetura em camadas da ferramenta CATRACA: a camada de captura, a camada de processamento e a camada de visualização.

²⁴Estatísticas de fluxos *flowtbag*: <https://dan.arndt.ca/projects/flowtbag/>

A **camada de processamento** é instanciada em uma nuvem dedicada para a classificação que possui o *Spark* como núcleo principal de processamento. A plataforma *Spark* foi escolhida entre as diferentes plataformas de processamento de fluxos por possuir o melhor comportamento em relação a tolerância a falhas [Andreoni Lopez et al. 2016b], tornando a ferramenta CATRACA mais robusta em caso de falha de nós de processamento. O *Spark* é implementado em um aglomerado de máquinas no modelo mestre/escravo, em que os escravos possuem a capacidade de expansão e redução de recursos, tornando o sistema escalável. Uma vez que os fluxos chegam na camada de processamento, um algoritmo seleciona as características mais importantes para a classificação das ameaças [Andreoni Lopez et al. 2017a]. Na etapa de processamento, os dados são enriquecidos através de distintas informações como a localização geográfica dos IPs analisados. Em seguida, os fluxos são classificados em maliciosos ou bem-comportados através de um algoritmo de aprendizado de máquina.

Finalmente, a **camada de visualização** é implementada utilizando a pilha de *software Elastic*²⁵. A pilha *Elastic* permite a visualização personalizada de eventos em tempo real. Assim, a saída da camada de processamento é enviada ao componente *elasticsearch* que fornece um rápido serviço de busca e armazenamento. A interface do usuário, que é executada no componente *Kibana* da mesma pilha de *software*, comunica-se com *elasticsearch* através de consultas. O *Kibana* processa os resultados das consultas e gera a visualização final.

Toda a documentação e a instalação da ferramenta CATRACA encontra-se disponível no site <https://www.gta.ufrj.br/catraca>.

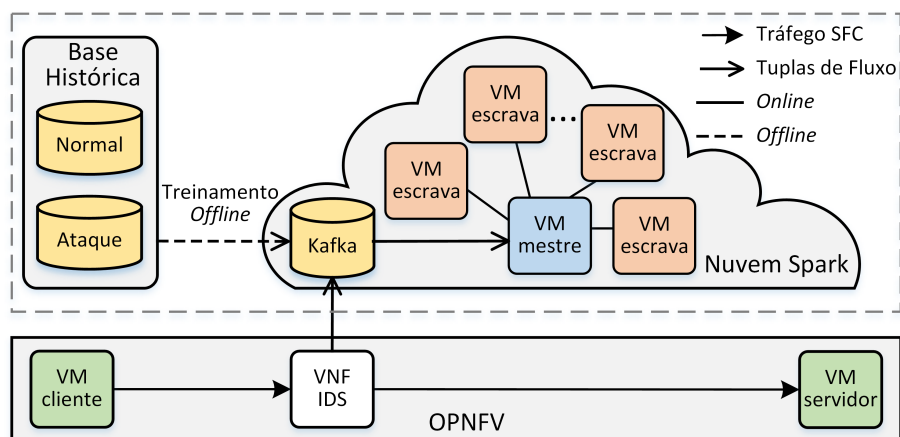


Figura 3.27. Demonstração dos experimentos Online e Offline.

A demonstração da ferramenta CATRACA será efetuada através de dois experimentos como mostra a Figura 3.27. Primeiramente será realizado um experimento *offline*. Para isso, um algoritmo de árvore de decisão é treinado de forma prévia por um conjunto de dados. A segunda etapa é a implementação do modelo de classificação gerado em fluxo e análise da classificação em tempo real do tráfego que atravessa uma máquina virtual, que será executada em um computador pessoal²⁶. A captura dos pacotes será enviada

²⁵Elastic Stack: <https://www.elastic.co/>

²⁶A demonstração requer um computador pessoal com acesso à Internet para ser configurado com os

ao nó central, nó *master*, localizado na nuvem OPNFV no Grupo de Teleinformática e Automação (GTA/UFRJ), no Rio de Janeiro. O nó localizado no GTA é responsável por receber os fluxos provenientes do *Kafka*, gerenciá-los e distribuí-los para os nós escravos do aglomerado *Spark*.

O objetivo da demonstração é evidenciar a acurácia, a agilidade e a escalabilidade da ferramenta na detecção de tráfego malicioso. Além disso, a ferramenta permite a visualização em tempo real de quando é capturado tráfego normal ou de quando é injetado tráfego malicioso através de uma interface de rede na máquina virtual monitorada. O classificador é previamente treinado com os parâmetros do modelo da árvore de decisão. Um conjunto de dados de uma operadora de telecomunicações [Andreoni Lopez et al. 2017c] da cidade do Rio de Janeiro é usado para gerar tráfego na máquina virtual.

Exemplo de Detecção de Ameaças em Tempo Diferenciado (*Offline*)

Conectar-se ao nó mestre do Spark

```
ssh -i cloud.pem ubuntu@<master-ip>,
```

carregar o conjunto de dados escolhido para o *Hadoop Distributed File System* (HDFS)

```
hdfs dfs -put dataset.csv /user/app
```

e submeter o programa Python `new-offline.py`²⁷ no aglomerado. O código `new-offline.py` é o aplicativo que executa a classificação sobre o Spark. Assim, é esse aplicativo carrega os dados do HDFS e realiza a classificação *offline* dos dados.

```
bin/spark-submit -master spark://master:7077
new-offline.py hdfs://master:9000/user/app/dataset.csv
```

Por padrão, o nó mestre do aglomerado responde pela porta 7077 e o arquivo carregado no HDFS se encontra na porta 9000. Os resultados da classificação serão exibidos na tela da seguinte forma:

```
Summary Stats:
Accuracy = 0.990038032436 Test Error = 0.00996196756424
Precision = 0.990038032436 Recall = 0.990038032436
F1 Score = 0.990038032436
```

Além disso, os resultados são armazenados no HDFS, para obtê-los em um arquivo `/tmp/Results.txt`:

```
hdfs dfs -getmerge hdfs://master:9000/user/app/Results_<inputfile>
/tmp/Results.txt
```

Exemplo de Classificação em Linha (*Online*) e Visualização

Nesse exemplo, são capturados dados de uma máquina virtual e o tráfego é espelhado para a nuvem Spark localizada no GTA/UFRJ. Na máquina virtual da qual serão capturados os pacotes executar:

softwares de teste.

²⁷código disponível <https://github.com/tinchoa/catraca/blob/master/processing-layer/new-offline.py>

```
python read_network.py <kafka topic> <kafka-server>
```

com os endereços Kafka-topic: Spark e Kafka-server: 10.10.10.3

conectar-se ao Spark Master:

```
ssh -i cloud.pem ubuntu@<master-ip> desde o $SPARK_HOME:
```

```
spark-submit -master spark://master:7077 -packages
<package elasticSearch> -jars <connector elastic search>,
<connector kafka> <python code> hdfs://master:9000/user/app/dataset
<kafka server> <kafka topic>
```

- package elasticSearch: TargetHolding:pyspark-elastic:0.4.2
- connector elastic search: /opt/spark/jars/elasticsearch-spark-20_2.10-5.5.1.jar, connector kafka: /opt/spark/jars/spark-streaming-kafka-0-8-assembly_2.11-2.1.1.jar
- código python: detection-with-elastic.py²⁸,
- kafka server: kafka01:2181,
- kafka topic: spark

para visualizar os dados de classificação e as métricas customizadas em tempo real, acessar a interface Kibana Dashboard, mostrada na Figura 3.28.

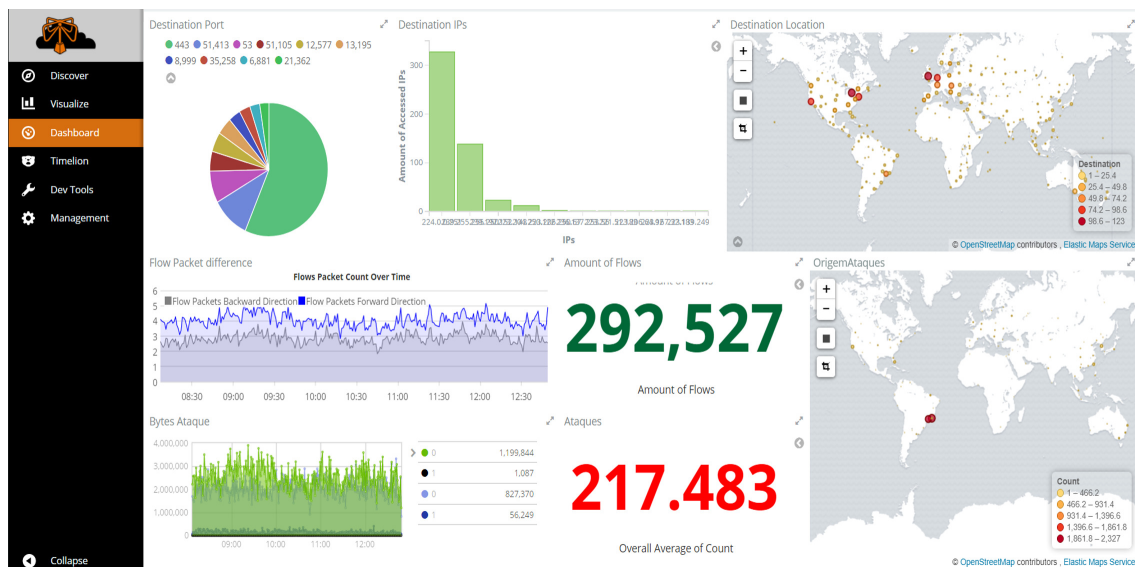


Figura 3.28. Visualização do painel de controle da ferramenta.

²⁸disponível em <https://github.com/tinchoa/catraca/blob/master/processing-module/detection-with-elastic.py>

Capítulo

4

Introdução à Linguagem P4 - Teoria e Prática

Luis Fernando Uria Garcia (UFES), Rodolfo S. Villaça (UFES), Moisés R. N. Ribeiro (UFES), Regis Francisco Teles Martins (UFSCar), Fábio Luciano Verdi (UFSCar), Cesar Marcondes (UFSCar)

Abstract

The goal of this short course is to present the main concepts related to network programmability, description of the data plane and packet forwarding to network devices using P4 language. This short course approaches this topic in a theoretical and practical way, presenting in the first part the PISA (Protocol-Independent Switch Architecture) model under which the P4 language works. The P4 language components used to process packets are also presented. Before the second part of this course, it presents a description of the compiler and the software switch native of P4, named bmv2. With all these concepts presented, the second part consists of practical exercises, which allow a better understanding of the concepts and elements initially presented. Finally, this short course also aims to show some of the most relevant projects that make use of P4 and in this way, perceive possible applications and motivate research topics in this area.

Resumo

O objetivo deste minicurso é apresentar os principais conceitos relacionados com a programabilidade de redes, descrição do plano de dados e encaminhamento de pacotes em dispositivos de rede por meio da linguagem P4. O minicurso aborda o tema de maneira teórica e prática, apresentando na primeira parte o modelo PISA (Protocol-Independent Switch Architecture), sob o qual a linguagem funciona. Apresenta os componentes da linguagem P4 pelos quais os pacotes são processados. Antes da segunda parte, apresenta uma descrição do compilador e do software switch nativo do P4, o bmv2. Com esses elementos, a segunda parte é constituída por exercícios práticos de implementação em laboratório que permitem uma melhor compreensão dos conceitos e elementos inicialmente apresentados. Por fim, este minicurso também tem como objetivo de mostrar alguns dos projetos mais relevantes que fazem uso do P4 e desta maneira, perceber possíveis aplicações e motivar temas de pesquisa nessa área.

1.1. Introdução

À medida que as redes evoluem e oferecem suporte para mais serviços, um grande volume de fluxo de dados é gerado e precisa ser transportado por tais redes. Assim, as redes atuais possuem a exigência de dar suporte para novas aplicações e serviços de natureza dinâmica surgindo a necessidade de tornar as redes flexíveis e programáveis, trazendo algumas mudanças de paradigma no que diz respeito à estrutura dessas redes.

Até pouco tempo atrás, devido à necessidade de processamento dos pacotes de forma rápida, essas funções foram encapsuladas nos ASICs (*Application Specific Integrated Circuits*) dos dispositivos de encaminhamento de rede. Como resultado, tem-se um plano de dados fixo e conseqüentemente engessado que impossibilitava a construção de redes de acordo com requisitos próprios, exclusivos de cada projeto. Ao fabricar dispositivos de encaminhamento com funções fixas, a questão é: o que fazer com as funções que o dispositivo não suporta e que os usuários podem necessitar? Nesse contexto, é requisito que o projetista do plano de controle possa informar ao dispositivo de encaminhamento como processar os pacotes no plano de dados.

Assim, as Redes Definidas por Software (*Software Defined Networking*, ou SDN) que, segundo a *Open Networking Foundation* (ONF), são definidas como uma arquitetura emergente, dinâmica, gerenciável e adaptável, permitem a separação entre o controle da rede e as funções de encaminhamento. Essa separação faz com que o controle da rede seja diretamente programável e a infraestrutura subjacente possa ser abstraída para o desenvolvimento de novas aplicações e serviços.

Um elemento importante dentro do paradigma SDN é o protocolo OpenFlow [McKeown et al. 2008], que fornece uma camada de abstração da rede física para o elemento de controle, permitindo assim que o plano de dados da rede seja configurado ou manipulado através de programação via software. Na sua versão atual, o OpenFlow especifica explicitamente os cabeçalhos de protocolo nos quais ele opera. Este conjunto cresceu de 12 a 41 campos em alguns anos, aumentando a complexidade da especificação porém ainda não oferece flexibilidade suficiente para se adicionar novos cabeçalhos e definir novas ações após um *flow matching*.

A linguagem P4 ajuda a superar essas limitações do OpenFlow, aumentando a programabilidade das Redes Definidas por Software. Neste minicurso pretende-se apresentar os conceitos gerais da linguagem P4 e como a linguagem permite descrever o plano de dados e o encaminhamento em dispositivos programáveis, tais como: software *switches*, *Smart NICs*¹, FPGAs (*Field Programmable Gate Array*) e *switches bare metal*. A linguagem P4 permite que o projetista recupere o controle do plano de dados e considere funções otimizadas para uma rede específica e possui três objetivos principais: Reconfigurabilidade, Independência do Protocolo e Independência de Alvo.

- Reconfigurabilidade: está relacionada com a habilidade do programador redefinir o processamento dos pacotes nos dispositivos de rede;
- Independência do Protocolo: define que os dispositivos de rede não devem ser amarrados a formatos de pacotes específicos;

¹<https://www.netronome.com/products/smartnic/overview/>.

- Independência do Alvo: relaciona-se com o fato de que o compilador da linguagem de programação de redes deve gerar uma descrição independente do dispositivo, porém levando em conta as capacidades do hardware alvo específico na hora de configurá-lo.

Com base neste contexto, vários benefícios da programabilidade do plano de dados podem ser percebidos, dentre eles:

- O controle total da programabilidade do plano de dados, de tal forma que o dispositivo de rede se comporta exatamente como desejado;
- Capacidade de adicionar novas funções, uma vez que o dispositivo é completamente programável;
- Exclusividade, porque os usuários têm a capacidade de implementar protocolos personalizados;
- Eficiência, uma vez que, considerando que os dispositivos atualmente têm capacidade limitada e funções fixas, muitas dessas não são usadas e, portanto, consomem recursos sem nenhuma finalidade;
- A confiabilidade constitui outro benefício, porque é possível não usar funções implementadas por terceiros;
- O monitoramento no plano de dados permite olhar para dentro do dispositivo. Por exemplo, é possível ver os cabeçalhos dos pacotes que estão sendo processados e exportar esses dados para obter indicadores de desempenho.

Em resumo, o minicurso se propõe a apresentar a linguagem P4 sob um ponto de vista teórico e prático. Na primeira parte do minicurso as bases conceituais da linguagem P4 e o modelo PISA (*Protocol-Independent Switch Architecture*), sob o qual a linguagem funciona, são apresentados. A segunda parte está constituída de exercícios práticos de implementação em laboratório que permitirão uma melhor compreensão dos conceitos e elementos inicialmente apresentados e, conseqüentemente, perceber possíveis aplicações da linguagem.

1.2. A Linguagem de Programação P4

Com base na seção anterior, que diz respeito aos benefícios da programabilidade do plano de dados, observa-se que para tornar o plano de dados programável precisamos de uma linguagem de alto nível. Por volta do ano 2013 surgiram as ideias iniciais sobre a programabilidade do plano de dados por meio de uma linguagem de domínio específico de alto nível. Assim, surgiu a denominação P4, acrônimo de *Programming Protocol-Independent Packet Processors*. A linguagem tomou forma e foi publicado o primeiro artigo no SIGCOMM em 2014 [Bosshart et al. 2014] quando surgiu a primeira versão, a $P4_{14}$, e foram lançadas as primeiras especificações oficiais. No ano de 2016 surgiu uma atualização mais relevante da linguagem, nomeada $P4_{16}$, que será trabalhada neste minicurso. O nome P4,

vem deste artigo, denominado *Programming Protocol-independent Packet Processors*, e que originou o acrônimo P4.

Pode-se observar que a especificação da linguagem possui diferentes versões, numeradas com três dígitos. Incrementos no dígito mais à direita indicam uma atualização ou adição ao último padrão. Um aumento no dígito do meio indica uma mudança maior e mais significativa. Todas as versões atuais começam com o dígito 1. Pode-se considerar o padrão 1.0.3 (lançado em 2016-11-03) como o padrão mais popular, mas o padrão 1.1.0 (lançado em 2016-01-27) e o padrão 1.2 (chamada agora $P4_{16}$) trouxeram mudanças importantes na linguagem. As principais características introduzidas em 1.1.0 foram: tipagem forte e a capacidade de indicar uma ordem na qual as ações devem ser executadas. As mesmas premissas sobre processamento dos pacotes no plano de dados foram mantidas, e tenta-se introduzir um modelo flexível visando a independência de alvos (ASICs, FPGA, NICs, software *switches*).

A linguagem P4 é uma linguagem que descreve como um pacote deve ser processado pelo plano de dados de um dispositivo programável de encaminhamento, que pode ser implementado em *switches* via hardware ou software, placas de rede, roteadores ou dispositivos programáveis do tipo FPGA. Inicialmente, a P4 foi desenhada para atuar em *switches* programáveis, no entanto, seu escopo acabou sendo estendido para cobrir uma ampla variedade de dispositivos, que no escopo da linguagem, passaram a ser chamados de "alvos"(ou *targets*).

A linguagem P4 é desenhada especificamente para descrever o plano de dados do alvo e também define a interface de comunicação com o plano de controle. No entanto, a linguagem P4 não pode ser usada para descrever as funções do plano de controle de um alvo. Por esse motivo, sempre que se refere a programar um alvo, entende-se programar o plano de dados. A Figura 1.1 exemplifica as principais distinções entre um *switch* tradicional e um *switch* programável P4.

1.2.1. Arquitetura de *Switches* Independentes de Protocolo (PISA)

Esta seção tem como objetivo apresentar a abstração do modelo de encaminhamento sobre o qual os pacotes são processados. Ou seja, por meio do modelo de encaminhamento abstrato consegue-se expressar como um dispositivo deve ser configurado e como os pacotes devem ser processados.

Esse modelo está baseado no OpenFlow e tem duas operações principais: Configurar e Preencher (*Configure and Populate*). A primeira basicamente define como o *parser* é programado, e a segunda operação adiciona entradas nas tabelas *match-action* que definem o comportamento do equipamento de rede. Na Figura 1.2 pode-se observar o *pipeline* dos pacotes iniciando-se no *parser*, passando pelas tabelas *match-action*, a execução das ações e, na última etapa, o mecanismo de processamento de filas para encaminhamento dos pacotes às portas de saída. É importante ressaltar que o reconhecedor (*parser*) pode ser programado por um modelo de máquinas de estados, conforme ilustra a figura, e que após uma ação, novos reconhecimentos podem ser programados, bem como novos *matching*, antes que um pacote seja encaminhado a uma fila de saída.

A arquitetura PISA possui os seguintes componentes principais:

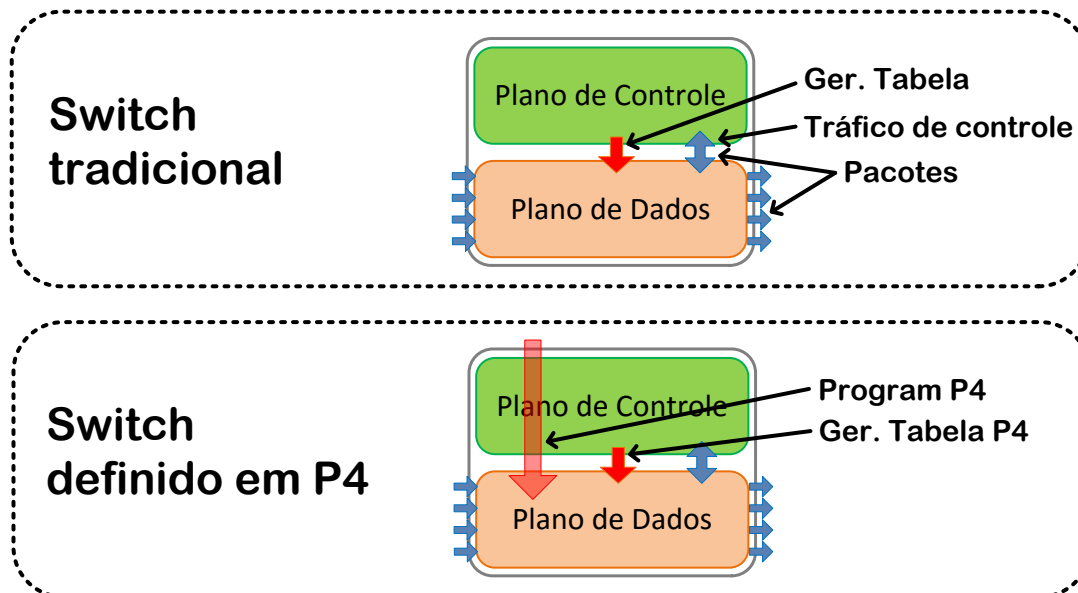


Figura 1.1. Comparação entre um *switch* tradicional e um *switch* programável P4.

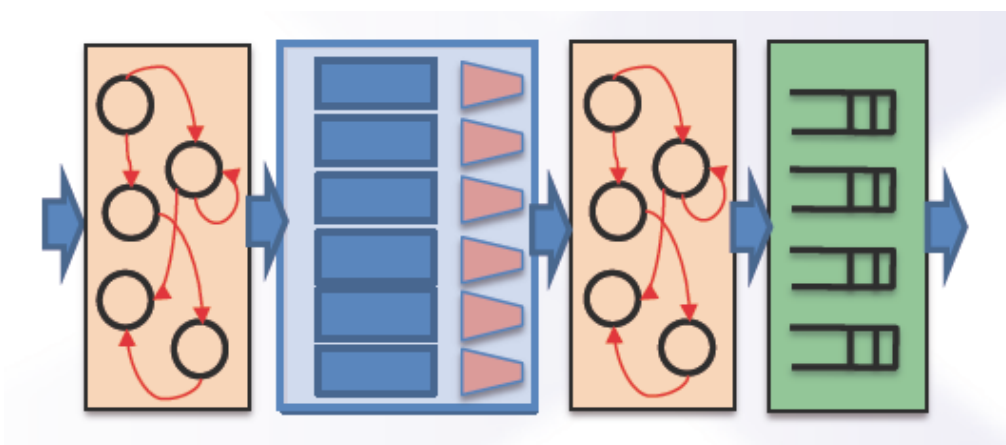


Figura 1.2. Pipeline entre os componentes da linguagem P4.

- *Parser Programável*: o *parser* tem por função identificar, em um *stream* de *bits*, os cabeçalhos que encontram-se presentes no pacote, ou seja, permite especificar o formato para processamento e identificar os protocolos contidos no pacote;
- *Match-Action*: nessa etapa faz-se um *match* e executa-se uma ação de acordo com as entradas que encontram-se nas tabelas *match-action*;
- *Deparser*: nessa etapa, monta-se de novo o pacote, para enviá-lo como um *stream* de *bits* na saída;
- Metadados: os metadados contém informações necessárias ao processamento mas que não estão no pacote, por exemplo, a identificação da porta de ingresso de um determinado pacote em um *switch*;

Pode-se observar, na Figura 1.3, o fluxo do processamento dos pacotes e as etapas do modelo PISA. O *parser* basicamente é uma máquina de estados finita que tem por função realizar a conversão do fluxo de *bits* que chega no dispositivo em um conjunto de cabeçalhos *headers*. Como exemplo, pode-se identificar e extrair primeiramente o cabeçalho Ethernet, olhar o identificador de VLAN e, na etapa seguinte do processamento no *pipeline*, os identificadores extraídos do cabeçalho identificado é repassado para as tabelas *match-action* onde para cada correspondência (*match*) executa-se uma ação (*action*) associada. Se temos pacotes L2 (*Layer 2*) pode-se fazer o *match* com o endereço MAC de destino e a ação correspondente seria encaminhar para uma porta de saída no *switch*. Em outro exemplo, se for um encaminhamento L3, com IPv4, o *match* poderia ser com o endereço IPv4 de destino e a ação poderia ser a substituição do endereço de origem pelo endereço do roteador, decrementar o TTL, e encaminhar. Cada vez que se faz uma nova ação, novos resultados serão gerados, alguns cabeçalhos podem ser modificados e portanto novos resultados intermediários podem ser produzidos exigindo novo processamento. Na última etapa haverá cabeçalhos modificados que precisam ser encaminhados à saída do dispositivo para remontagem do pacote. É nesse ponto que aparece o *deparser*, o qual monta os cabeçalhos modificados dos pacotes e os coloca nas filas de saída.



Figura 1.3. Fluxo dos pacotes dentro da Arquitetura PISA. Fonte: [Changhoon Kim 2016].

Os principais componentes da linguagem P4 associados ao modelo PISA estão descritos a seguir:

Headers: A definição do cabeçalho descreve a sequência e estrutura de uma série de campos de *bits* definidos pelo programador. Inclui especificações de largura, restrições de tipos e valores.

Parsers: A definição de *parser* especifica como identificar (reconhecer) cabeçalhos ou sequências de cabeçalho válidos nos pacotes.

Tables: As tabelas *match-action* são o mecanismo para realizar o processamento de pacotes, nas quais são associadas ações (*actions*) aos pacotes de acordo com o *matching* realizado nos *parsers*.

Actions: P4 suporta construção de ações complexas construídas usando primitivas simples e independentes de protocolo.

Em geral, o *parser* identifica os *headers* presentes em cada pacote ingressando no dispositivo. Cada tabela *match-action* realiza uma busca (*lookup*) em um subconjunto de campos de cabeçalho e aplica as ações correspondentes ao primeiro *matching* encontrado na tabela.

1.2.2. Elementos da Linguagem P4

Em $P4_{16}$ podem-se encontrar 5 (cinco) categorias principais de elementos da linguagem que permitem a programação do *pipeline* do plano de dados:

- Tipos de Dados: P4 tem um conjunto de tipos de dados básicos que servem para a construção de tipos mais complexos, tais como *arrays*, *headers* e *structures*;
- Expressões: constituídas por uma lista de operadores e operações básicas;
- Elementos de Fluxo de Controle: são elementos e estruturas para expressar o fluxo de dados entre as tabelas *match-action*;
- *Parsers*: elementos de programação de máquina de estados e extração de campos;
- *Externs*: conjunto de bibliotecas para suporte de componentes especializados.

1.2.2.1. Headers e Structures

O *Header* constitui o mais importante dos tipos de dados. Pode conter os tipos básicos *bit*, *int*, *varbit*. A declaração de um cabeçalho é dada pela sintaxe a seguir:

```
header HeaderName {
    fields.....
}
```

Por exemplo, pode-se definir o cabeçalho Ethernet da seguinte forma:

```
header Ethernet_h {
    bit<48> dstAddr;
    bit<48> srcAddr;
    bit<16> etherType;
}
```

A seguinte declaração de variável, ou instanciação, usa o tipo *Ethernet_h* definido anteriormente:

```
Ethernet_h ethernetHeader;
```

A *Struct* é uma estrutura composta. Nesse caso, a estrutura é formada por campos que são cabeçalhos (*headers*) definidos previamente. Um exemplo deste tipo de definição é o seguinte:

```
header Tcp_h { ... }
header Udp_h { ... }
struct Parsed_headers {
    Ethernet_h ethernet;
    Ip_h      ip;
    Tcp_h     tcp;
    Udp_h     udp;
}
```

1.2.2.2. Operadores e Expressões

P4 possui uma lista de operadores e um conjunto de expressões parecidas com outras linguagens de uso geral, como C, por exemplo. Os operadores básicos são: +, -, *, «, », &, |, &, &&, ||, !, <, <=, >, >=, ==, != (soma, subtração, multiplicação, deslocamento de *bits* à esquerda, deslocamento de *bits* à direita, "e" *bit* a *bit*, "ou" *bit* a *bit*, "ou exclusivo" *bit* a *bit*, negação, "e" lógico, "ou" lógico, negação e comparadores). Também suporta operadores ternários da forma: $(cond)?(true_{expr}) : (false_{expr})$.

1.2.2.3. Match-Actions

No pipeline, em uma operação *match-action* tem-se 3 elementos principais: Controles, Tabelas e Ações. Em P4, os controles são muito parecidos com funções em C, sem *loops* e recursão. Os blocos do *pipeline* seguem uma ordem sequencial e vão sempre para a frente com a possibilidade de uso de declarações *if* e/ou declarações do tipo *switch*. Ou seja, o corpo do bloco de controle é executado, semelhante à execução de um programa imperativo tradicional. Sintaticamente, um bloco de controle é declarado com um nome, parâmetros obrigatórios, opcionais e uma sequência de declarações de constantes, variáveis, ações, tabelas e outras instâncias. Dentro de um bloco de controle, utiliza-se a sentença *apply(nome_tabela)* para invocar as ações correspondentes declaradas na tabela.

```
control ingress {
    apply(nome_tabela);
}

control egress {
}
```

Parsers são funções especiais escritas em um estilo de máquina de estados finita que basicamente buscam converter ou mapear fluxos de *bits* em cabeçalhos. Têm uma sequência de sentenças que representam um conjunto de estados e as regras de como faz-se a transição entre eles. Todo *parser* tem três estados predefinidos. O primeiro é o estado *Start*, que é onde inicia-se o processo. O outro estado é o *Accept*, o qual é usado

para indicar que o pacote é aceito. O terceiro estado é o *Reject*, o qual é onde envia-se aqueles pacotes que não se quer tratar. A Figura 1.4 mostra a estrutura geral da máquina de estados do *parser*.

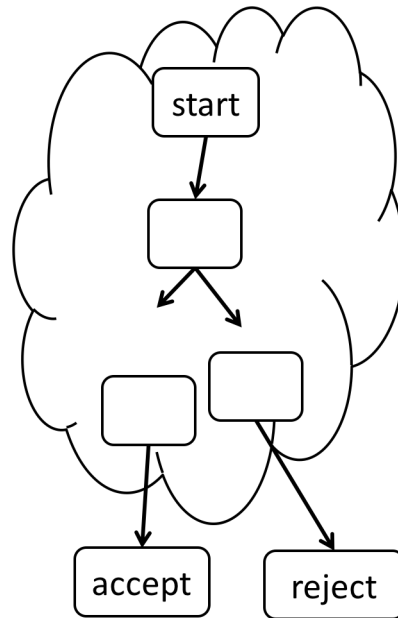


Figura 1.4. Estados predefinidos de um *parser*. Fonte: [Consortium 2017].

Os estados definidos pelo programador descrevem como o pacote será processado pelo *parser*. Como exemplo de declaração de uma estrutura de um *parser* pode-se observar a seguinte estruturação, que será detalhada na próxima seção.

```

parser start {
    return parse_ethernet;
}

parser parse_ethernet {
    extract (ethernet);
    return ingress;
}
  
```

Os objetos externos (*Externs*) são construções específicas da arquitetura que podem ser manipuladas por programas P4 através de APIs bem definidas, mas cujo comportamento interno é *hard-wired* (por exemplo, unidades de *checksum*) e, portanto, não programáveis usando P4.

1.2.3. Seções de um Programa P4

Um programa em P4 pode-se dividir em 3 principais seções, de acordo com o *pipeline* da Arquitetura PISA que os pacotes atravessam. Com base na seção anterior, onde apresentaram-se os elementos centrais da linguagem, a seguir descreve-se as seções de um programa em P4 procurando agrupar aqueles elementos em exemplos de declarações de cada seção.

1.2.3.1. Declaração de Dados

Basicamente, um programa P4 começa com a definição do cabeçalho. Por isso, o tipo de dado mais importante é o tipo *Header*. Na seção sobre elementos da linguagem já mostramos como fazer a declaração dos cabeçalhos. Com base nos tutoriais que encontram-se no repositório oficial da linguagem P4, especificamente o exercício chamado `basic.p4`, apresenta-se como exemplo a definição de um quadro Ethernet, onde tem-se especificado os campos endereço destino, endereço de origem e *ethertype*. Assim também define-se um cabeçalho IPv4.

```
typedef bit<9>  egressSpec_t;
typedef bit<48> macAddr_t;
typedef bit<32> ip4Addr_t;

header ethernet_t {
    macAddr_t dstAddr;
    macAddr_t srcAddr;
    bit<16>    etherType;
}

header ipv4_t {
    bit<4>    version;
    bit<4>    ihl;
    bit<8>    diffserv;
    bit<16>   totalLen;
    bit<16>   identification;
    bit<3>    flags;
    bit<13>   fragOffset;
    bit<8>    ttl;
    bit<8>    protocol;
    bit<16>   hdrChecksum;
    ip4Addr_t srcAddr;
    ip4Addr_t dstAddr;
}
```

1.2.3.2. Parser Programável

Uma vez definidos os *headers*, precisa-se expressar a lógica do *parser*, a mesma que será usada para identificar a transição da máquina de estados finita. Como exemplo, pode-se ver uma sequência de estados simples, desde o estado inicial, para extrair os cabeçalhos Ethernet e IPv4 em um pacote.

```
parser MyParser(packet_in packet,
                out headers hdr,
                inout metadata meta,
                inout standard_metadata_t standard_metadata) {

    state start {
        transition parse_ethernet;
    }

    state parse_ethernet {
```

```

        packet.extract(hdr.ethernet);
        transition select(hdr.ethernet.etherType) {
            TYPE_IPV4: parse_ipv4;
            default: accept;
        }
    }

    state parse_ipv4 {
        packet.extract(hdr.ipv4);
        transition accept;
    }
}

```

1.2.3.3. Tabelas

Nas tabelas, são encontradas as chaves com as quais procura-se fazer o *matching* assim como as ações associadas. As ações são basicamente funções compostas por ações primitivas. Como exemplo, pode-se observar a seguir, o código para uma tabela IPv4, onde o campo para realizar o *matching* é o endereço IP de destino sob o algoritmo de *Longest Prefix Matching* e as ações possíveis são *setnhop* (definir o próximo salto) e *drop* (descarte).

```

table ipv4_lpm {
    key = {
        hdr.ipv4.dstAddr: lpm;
    }
    actions = {
        ipv4_forward;
        drop;
        NoAction;
    }
    size = 1024;
    default_action = NoAction();
}

action ipv4_forward(macAddr_t dstAddr, egressSpec_t port) {
    standard_metadata.egress_spec = port;
    hdr.ethernet.srcAddr = hdr.ethernet.dstAddr;
    hdr.ethernet.dstAddr = dstAddr;
    hdr.ipv4.ttl = hdr.ipv4.ttl - 1;
}

```

Em outro exemplo, o bloco de controle a seguir considera a condição de verificação de validade do cabeçalho para aplicar a tabela *ipv4_lpm* por meio da sentença *apply*:

```

control MyIngress(inout headers hdr,
                  inout metadata meta,
                  inout standard_metadata_t standard_metadata) {
    apply {
        if (hdr.ipv4.isValid()) {
            ipv4_lpm.apply();
        }
    }
}

```

```
}

```

1.3. Compilando Programas na Linguagem P4

Esta seção apresenta uma introdução geral sobre o processo de compilação de programas na linguagem P4. Começamos destacando que a linguagem P4 é fruto de ampla discussão e trabalho colaborativo da comunidade de redes, trabalhando pela evolução da programabilidade do plano de dados. Nessa comunidade existe um processo de criação e refinamento do P4, onde a cada 2 anos são preparadas novas **especificações** da linguagem e da **implementação** de compilador de referência. Esses dois recursos, especificação e compilador de referência se inter-relacionam, e permitem a melhoria consecutiva de ambos, ao longo do tempo. A versão corrente da linguagem P4, considerada estável, é a **P4_16** que conta com um esforço colaborativo de dezenas de desenvolvedores, licença *open source* Apache, e milhares de linhas de código e teste. Dessa forma, destacamos que o P4 é um projeto maduro que pode inclusive ser usado em alguns cenários em modo de produção.

Arelada à linguagem, o desenvolvimento de um compilador para a linguagem P4, traz por si, uma série de desafios de projeto. Por exemplo, a compilação de um programa P4 precisa possuir suporte a versões mais antigas da linguagem, garantindo retro-compatibilidade, bem como permitir uma extensibilidade flexível para suportar versões futuras com diferentes sintaxes. Além dos desafios de análise sintática e semântica do código P4, ele precisa também, dentro da arquitetura geral, suportar um modelo de abstração *frontend* único e estável. Esse *frontend* deve ser de código aberto e não deve ser constantemente modificado por desenvolvedores.

Para ilustrar o fluxo de trabalho do compilador P4 desde a entrada do programa ".p4" até a geração de código específico do alvo, como o Verilog de código P4, ou mesmo código C, detalharemos a Figura 1.5. O primeiro trecho do fluxo do compilador é expressar o código P4 em um formato intermediário **único** (IR). Para tanto, é feito o uso de uma gramática que está especificada nos documentos de padronização, e permite fazer o segmentação precisa dos construtores da linguagem P4 (P4_16). Para manter a compatibilidade com versões antigas, o compilador possui dois segmentadores (*parsers*) baseados em duas gramáticas ligeiramente diferentes e ambas devem ser convertidas para uma mesma representação intermediária comum "segmentada" (primeira referência de IR), antes de serem processadas pelo *frontend*.

O primeiro código IR, com as transformações iniciais, é então processado pelo *frontend* e passa por um *pipeline* de processamento de transformação. Por exemplo, no P4_16 o *frontend* introduz mais de 25 passos de transformação sintática e semântica, de modo que seja colocado em um formato de uma plataforma abstrata P4 (expressa em classes de linguagem de C++) e portanto independente do hardware e software de baixo nível, gerando assim o segundo IR que aparece na Figura 1.5.

Esse IR genérico pode então passar por um estágio de transformação de baixo nível. Esse estágio já dependente da arquitetura, e focado para a preparação do processamento *midend* e depois *backend*. A diferença do *midend* e do *backend* é somente para fins de reaproveitamento de código em plataformas específicas que podem ser muito pró-

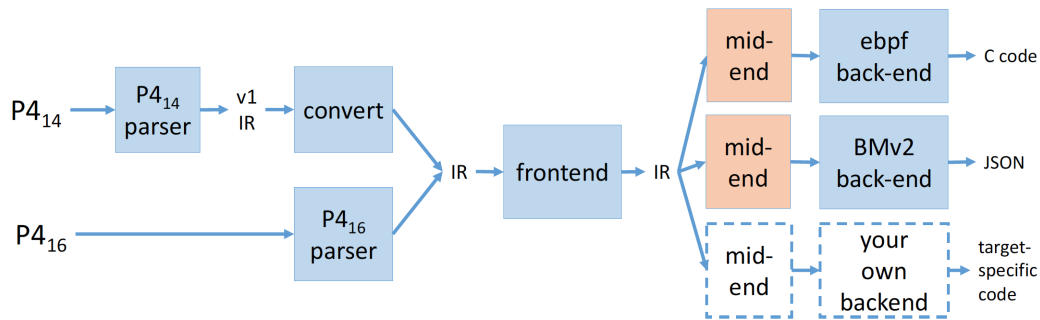


Figura 1.5. Fluxo do Compilador P4. Fonte: [Changhoon Kim 2016].

ximas. Finalizando o fluxo do compilador, é no *backend* que o compilador deverá suportar uma ampla variedade de arquiteturas **alvo** diferentes. Desse modo, o *backend* aumenta o ecossistema de plataformas específicas que suportam a possibilidade de reprogramação do plano de dados via P4. Essas plataformas específicas podem ser desde *chips* integrados (ASIC) baseados em PISA configuráveis como a plataforma Tofino² que tem capacidade de processar Terabits/s, passando por FPGA³ com seu modelo de referência NetFPGA-P4. Finalmente, até mesmo a conversão do programa P4 em plataformas puramente de software, como código binário de filtragem baseado no *kernel* eBPF (*extended Berkeley Packet Filter*)⁴ e código de *openvswitch* instrumentado e parametrizado para suportar a arquitetura PISA, como o projeto PISCES⁵. Esse poder de suportar uma variedade de plataformas é um objetivo importante, pois o compilador começa abstraindo o suficiente o processamento do plano de dados, e depois deixa os pontos específicos da plataforma real para esse estágio final de *backend* da compilação.

Dentre as plataformas usadas para testes e simulação para criar provas de conceito está o BMv2, que é a implementação do "*Behavioral Model*" da arquitetura PISA, e esse por sua vez implementa um modelo simples de *switch* no *backend* específico do compilador: *p4c-bm*. Uma vez feita a compilação do programa P4, o *backend* será o responsável pela programação ou configuração do plano de dados apropriado. Por exemplo, no caso do BMv2, o programa P4 será transformado em um descritivo JSON como saída do *backend*, conforme Figura 1.5. O arquivo JSON, por sua vez, será a entrada para o *switch* BMv2 que fará na entrada a segmentação dos pacotes, seguindo a orientação do que estiver especificado no arquivo de configuração JSON, e depois populará as tabelas do pipeline do *switch* PISA de referência. Todo esse processo é específico do BMv2. Na próxima sub-seção, apresentaremos o passo a passo de como compilar um programa P4 usando o BMv2.

Para outros tipos de *backend*, tais como as FPGAs, a saída final pode ser a geração de *soft cores* de FPGA, que são, em última instância, representados em linguagem Verilog ou VHDL e que permitem refazer o *pipeline* de baixo nível PISA, como é o caso do projeto NetFPGA-P4. Para outros *backend* baseados em *software* como o eBPF, a saída do

²<https://barefootnetworks.com/products/brief-tofino/>.

³<https://github.com/NetFPGA/P4-NetFPGA-public>.

⁴<http://prototype-kernel.readthedocs.io/en/latest/bpf/>.

⁵<http://piscs.cs.princeton.edu/>.

backend produz o equivalente a um *bytecode* de baixo nível que instrui a criação de regras de filtragem, de alto desempenho, dentro da pilha de redes do sistema operacional Linux. Após a instrução passo-a-passo de como compilar usando o *backend* BMv2, voltaremos em sub-seção subsequente a discutir detalhes de outro *backend*, o NetFPGA-P4, onde detalharemos o seu funcionamento interno.

1.3.1. Passo a Passo na Compilação de Programa P4 no *backend* BMv2

Conforme vimos no começo dessa seção, para implementar um programa em P4, precisa-se mapear a descrição de como serão processados os pacotes (em programa ".p4") em um dispositivo alvo (*target*) específico, seja hardware ou software. Esse é basicamente o trabalho do compilador.

O fluxo de etapas mostrado na Figura 1.6 representa uma configuração de ferramentas que permitem implementar programas escritos em P4 em um dispositivo alvo. Dentre essas ferramentas temos o compilador `p4c` e o *target* que nesse caso é o software *switch* nativo de referência, denominado BMv2. Nessa representação, pode-se observar que o programa escrito com código P4 é processado pelo `p4c` (compilador P4), para em seguida carregar a representação comportamental gerada pelo compilador no BMv2 software switch. É dessa forma que o dispositivo processará os pacotes no plano de dados da maneira desejada e descrita no programa. Além disso, pode-se observar como o plano de controle poderia interagir com o dispositivo *target* em tempo de execução por meio do *framework* chamado P4Runtime. No caso do BMv2, tem-se disponível uma ferramenta de tipo CLI (*Command Line Interface*) para preencher as tabelas do dispositivo alvo.

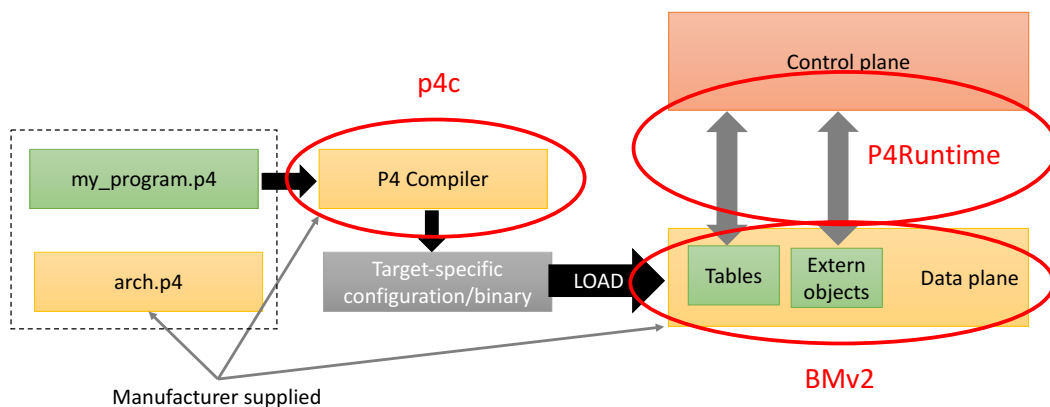


Figura 1.6. Fluxo para implementar um programa P4. Fonte: [Carmelo Cascone 2016].

1.3.2. Compilador P4C

Em um breve histórico, o `p4c-behavioral` foi o primeiro compilador P4 e era conhecido também como BMv1. Esse compilador gerava um código C para cada programa P4, que seria compilado para um executável usando o `gcc`, ou seja, para cada programa P4 precisava-se um novo executável.

Assim, mudou-se para o novo compilador *Behavioral Model version 2* (BMv2). O fluxo com esse novo compilador é simples: o programa P4 é primeiramente compi-

lado para uma representação JSON (*JavaScript Object Notation*) por meio do módulo chamado `p4c-bm`. Este JSON é então carregado no BMv2 e as estruturas de dados são inicializadas para refletir o comportamento de encaminhamento desejado.

Para executar seus próprios programas P4 no BMv2, primeiramente você precisa transformar o código P4 em uma representação JSON que pode ser consumida pelo software *switch*. Esta representação dirá ao BMv2 quais tabelas inicializar e como configurar o *parser*. Pode-se gerar o arquivo JSON a partir de um programa P4 da seguinte forma:

```
p4c-bm -json <path to JSON file> <path to P4 file>
```

O arquivo JSON agora pode ser usado como entrada para o *switch* BMv2. Como exemplo, para realizar esta ação usando um *simple_switch* como alvo a seguinte sequência de comandos é necessária:

```
sudo ./simple_switch -i 0@<iface0> -i 1@<iface1> <path to JSON file>
```

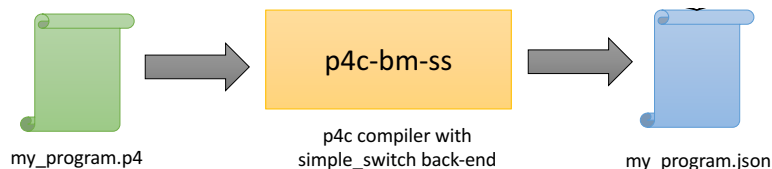


Figura 1.7. Fluxo de compilação. Fonte: [Carmelo Cascone 2016].

A última versão do compilador `p4c` aceita compilar programas tanto em $P4_{14}$ quanto $P4_{16}$, podendo então ser usado no lugar do `p4c-bm`. Com o `p4c-bm2-ss` compila-se diretamente para o alvo *simple_switch*, que é escrito usando o Behavioral Model (Ver Figura 1.7).

Desta forma, para compilarmos programas em $P4_{16}$ para o *simple_switch* (target BMv2) utiliza-se a seguinte linha de comando:

```
p4c-bmv2-ss -o program.json program.bmv2.p4
```

1.3.3. Detalhamento do Backend NetFPGA-P4

A plataforma de experimentação NetFPGA é uma das mais difundidas plataformas para ensino e pesquisa de novos projetos de dispositivos de rede. Com ela é possível projetar um equipamento de rede: a) completamente a partir de FPGA puro e; b) reutilizando módulos de alta qualidade escritos em projetos de referência. A plataforma possui diversos modelos de placas comercializados para fins acadêmicos, de um espectro de opções de placas com interfaces de 1G até 10G. Da família de placas NetFPGA, a mais sofisticada é a NetFPGA SUME com chip Xilinx Virtex-7 com capacidade de 690K células lógicas e capacidade de suportar projetos de grande porte, bancos de memória DRAM DDR3 SoDIMM com 4GB e SRAM QDRII+, 4 transceptores SFP+ com suporte a cabo direto e interfaces seriais de expansão QTH-DP de 8 x 12.5Gbps, e FMC HPC de 10 x 12.5 Gbps.

Nessa família de placas, para suportar a possibilidade de programação de projetos de alto desempenho usando a linguagem P4, foi feita uma parceria entre a empresa Xilinx e os gestores do projeto NetFPGA (Universidade de Stanford e Universidade de

Cambridge - UK) para a liberação de um *backend* apropriado para a NetFPGA SUME, o projeto P4-netFPGA [Sultana et al. 2017]. Além do *backend*, a comunidade e outros entusiastas de NetFPGA produziram parte do compilador P4 que suporta o fluxo de compilação para a NetFPGA.

A Figura 1.8 apresenta o fluxo de programação do plano de dados da NetFPGA à partir do programa P4. O programa p4 é observado no quadro à esquerda superior e ele é baseado em uma arquitetura de referência P4 para a NetFPGA, representada logo abaixo na mesma figura. Observe que a arquitetura de referência trata-se da arquitetura **SimpleSwitch**, muito parecida com a arquitetura vista com o BMv2, em termos de *pipelines Match* e *Actions*. Essa arquitetura **SimpleSwitch** é a forma abstrata de apresentar um modelo de *switch* compatível com PISA. Continuando no fluxo de programação, temos o compilador P4 que produz como saída: uma instanciização *runtime* do plano de dados e simultaneamente uma API (*Application Program Interface*) que permite a comunicação com registradores e configurações internas do hardware.

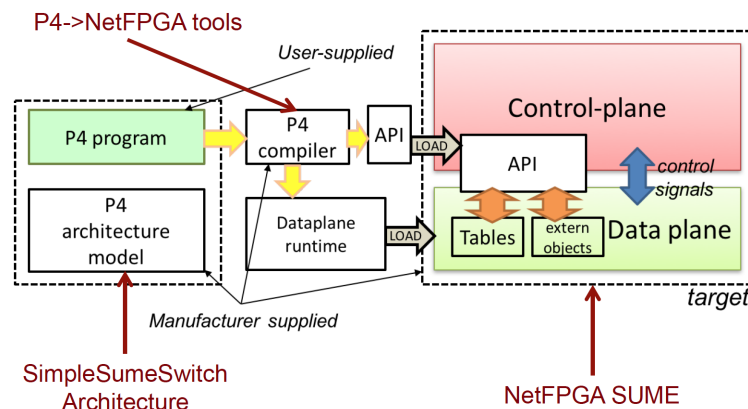


Figura 1.8. Fluxo de Programação P4 para o Alvo da placa netFPGA. Fonte: [Ibanez 2017].

Ambas essas saídas são carregadas na plataforma netFPGA para execução. O *runtime* é sintetizado, e o *bitstream* é carregado no chip Xilinx Virtex-7, enquanto que a API é carregada como um conjunto de *drivers*, programas em C e interfaces que permitem a posterior configuração do plano de dados através da Tabela *Match* e *Actions*. Também é carregado na plataforma *runtime*, código adicional em Verilog para algum tipo de processamento especial, acessível pela interface chamada *extern objects*.

O *extern objects* é a maneira pela qual a plataforma P4 pode ser estendida se o alvo permitir mudanças e inclusão de funções que não existem na arquitetura PISA, como por exemplo, funções *hashing* específicas, CRC (*Cyclic Redundance Checking*) ou outras que necessitam de processamento especial por pacote. Dessa forma, o fluxo na camada do plano de dados pode ser incrementado com processamento de funções matemáticas ou outros processos computacionais que seriam implementados na plataforma alvo. A plataforma alvo da NetFPGA é baseada em código Verilog, portanto, a adição de código FPGA específico para casos particulares podem ser feitos, configurados e manuseados diretamente na plataforma.

Outro destaque na Figura 1.8 são o conjunto de setas que apontam o material su-

portado pelo fabricante (*Manufacturer Supplied*), ou seja, o projeto tem partes em código aberto e partes que são ferramentas proprietárias da Xilinx. A figura a seguir, Figura 1.9, apresenta a cadeia de ferramentas para o processo de compilar alvos P4 em NetFPGA. Começando no topo, o programa P4 faz uso do compilador com partes proprietárias Xilinx P4_16. Essa ferramenta pode ser obtida por meio de solicitação diretamente para a Xilinx.

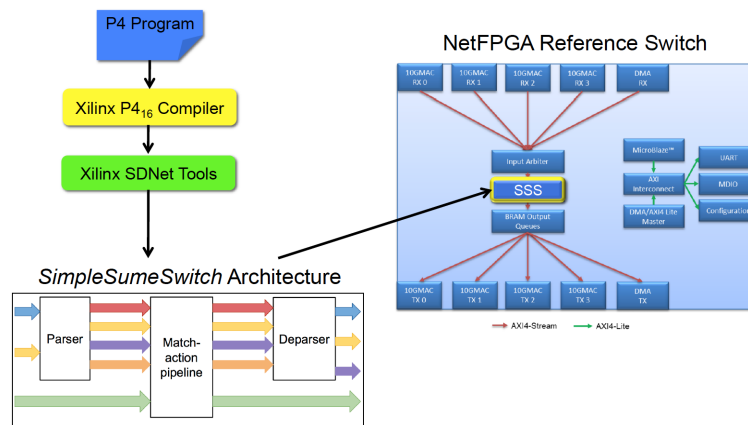


Figura 1.9. Visão Geral das Ferramentas Abertas e Proprietárias usados no P4-NetFPGA. Fonte: [Ibanez 2017].

Em seguida, a próxima ferramenta Xilinx SDNet Tools é um conjunto de *softcores* e bibliotecas proprietárias que são específicas para Redes Definidas por Software e que permitem a implementação eficiente de cadeias de *match* e *actions* como as usadas em OpenFlow e outras tecnologias. O resultado da aplicação das ferramentas em sequência permite especificar em Verilog, com detalhes, a arquitetura *SimpleSumeSwitch* (SSS) que basicamente contém os elementos da arquitetura PISA de segmentação (*parsing*), o conjunto de pipeline de ações de *match* e *action* e o serializador final (*deparsing*).

Esse elemento da arquitetura PISA então é embarcado (lado direito da Figura 1.9) no **NetFPGA Reference Switch**. Em outras palavras, todo o entorno do núcleo SSS reaproveita a implementação do *switch* de referência em NetFPGA, que foi escrito em Verilog. Com isso se reaproveita o código que permite lidar com a sincronização das filas TX e RX, o árbitro de entrada que detecta se um pacote está sendo enviado pela mesma porta de entrada e saída, o gerenciador de filas de saída que tem um mecanismo de controle de *buffers* em DRAM. Além disso, registradores também ficam expostos, e *soft* microprocessador MicroBlaze estão automaticamente integrados no projeto.

Outros detalhes dessa implementação P4-FPGA podem ser obtidos em [Sultana et al. 2017] e [Wang et al. 2017]. Por exemplo, para o aproveitamento completo de um desenvolvedor nesta plataforma é preciso ter noção dos aspectos específicos da linguagem P4, tais como: os metadados na arquitetura SSS que estão organizados na estrutura *sume_metad_a_t*. Também é preciso ter uma ideia do mapeamento dos metadados específicos da NetFPGA com os metadados nativos da linguagem P4. Outro aspecto bastante relevante é o uso das funções externas, que podem ser implementação caixa-preta de códigos em Verilog para melhorar a funcionalidade de um projeto. Essas funções externas permitem especificar coisas como variáveis de estado, e variáveis atômicas, bem como

funções de medição de latência feitas pela Xilinx.

O passo a passo da execução final de um projeto P4-NetFPGA passa pelos seguintes passos:

- Escrever um programa P4;
- Com o código do P4-NetFPGA baixado, executar o *script* `gen_testdata.py`;
- Na pasta do projeto P4-netFPGA compilar o programa P4 e gerar o código Verilog;
- Gerar as ferramentas de API e CLI, ambas com a de cima, com um único `make`;
- Executar o simulador SDNet com o comando `vivado_sim.bash`;
- Executar o simulador NetFPGA com o comando
`./nf_test sim -major switch -minor default`;
- Sintetizar o projeto e criar o bitstream com o comando `make`;
- Testar seu novo hardware.

Com isso, encerramos o detalhamento do processo interno de compilação de programas P4 e sua aderência para plataformas alvo variadas como BMv2 e NetFPGA, que permitem a construção de provas de conceito altamente sofisticadas. A seguir, seguiremos para os laboratórios P4 usando BMv2.

1.4. Laboratórios

A fim de entender melhor como funciona uma rede formada por dispositivos de encaminhamento programáveis em linguagem P4, este minicurso conta com uma parte prática onde os participantes, utilizando-se de uma máquina virtual fornecida e instalada previamente, poderão criar, compilar e implementar programas em linguagem P4.

A parte prática, ou de laboratório, conta com três atividades propostas, onde todo o ambiente já foi previamente configurado e disponibilizado⁶, cabendo ao participante complementar o código P4 de forma a cumprir com o objetivo de cada atividade.

1.4.1. Atividade 1 - Encaminhamento de Pacotes

O objetivo desta atividade é simplesmente encaminhar um pacote recebido de um *host* para outro. O diagrama apresentado na Figura 1.10 demonstra a topologia configurada para essa tarefa.

Para simplificar ainda mais, deve ser implementado somente a lógica de encaminhamento para o protocolo IPv4. Para isso, o elemento de encaminhamento ou *switch*, deverá desempenhar as seguintes ações para cada pacote recebido:

- Atualização do endereço MAC de origem e destino;

⁶https://www.dropbox.com/sh/r29oa0abzluj9n2/AABFsTTVZrh1yQ8WFK01_HRna?dl=0

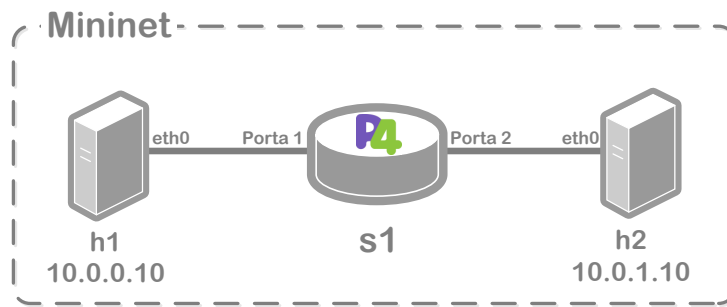


Figura 1.10. Topologia proposta para a Atividade 1.

- O decremento do valor do TTL (*Time To Live*) do pacote;
- Encaminhamento do pacote para a porta de saída apropriada.

O *switch* P4 contará com uma única tabela, a qual será populada com regras estáticas implementadas pelo plano de controle usando CLI. Cada uma das regras mapeia um endereço IP para o endereço MAC e a porta de saída para o próximo salto. Essas regras já foram previamente implementadas, então será necessário somente a implementação da lógica do plano de dados através do código P4.

1.4.1.1. Passo 1: Execução Inicial

O diretório `/home/student/labs/task_1/p4src` contém o código P4 `task_1.p4`, que pode ser compilado. No entanto, em seu estado atual ele simplesmente descarta todos os pacotes que chegam até o dispositivo. O objetivo dessa atividade é completar esse código inicial de forma que o *switch* P4 faça o encaminhamento do pacote ao *host* correto.

Primeiramente, compile o código de forma que está e execute o Mininet para testar o seu funcionamento.

Na linha de comando, dentro do diretório `/home/student/labs/task_1/` execute o *script*: `sudo ./run_task.sh`. Esse *script* irá compilar o programa `task_1.p4` e inicializar o Mininet com a topologia configurada para a atividade.

A instância do Mininet para essa atividade conta com um *switch* (`s1`) e dois *hosts* (`h1` e `h2`). Os *hosts* possuem os seguintes endereços IPs associados: `h1` - 10.0.0.10 e `h2` - 10.0.1.10.

Uma vez que a instância do Mininet esteja carregada, obtem-se o *prompt* `>`. Abra então terminais para `h1` e `h2`, respectivamente:

```
mininet> xterm h1 h2
```

Em cada um dos *hosts* existem aplicações baseadas em Python para enviar e receber mensagens. Inicie primeiramente a aplicação para receber mensagens no terminal do *host* 2 (`h2`).

```
sudo ./receive.py
```

Em seguida, inicie a aplicação para enviar mensagem a partir do *host* 1 (*h1*). A sintaxe do comando é `send.py <ip do receiver> <"mensagem">`:

```
sudo ./send.py 10.0.1.10 "Olá mundo P4!"
```

A mensagem não será recebida.

Saia do emulador digitando `exit` na linha de comando.

```
mininet> exit
```

A mensagem não foi entregue porque o *switch* está programado com o código P4 em `task_1.p4` original, que descarta os pacotes recebidos. O objetivo dessa tarefa é estender o código P4 para que os pacotes consigam alcançar o seu destino.

1.4.1.2. Plano de Controle

O programa em P4 define o *pipeline* de processamento dos pacotes, mas as regras em cada uma das tabelas são inseridas através do plano de controle. Quando uma regra encontra um pacote correspondente, sua ação é invocada com os parâmetros fornecidos pelo plano de controle.

Nessa atividade, a lógica do plano de controle já foi implementada. Assim que o Mininet é instanciado, as regras para o processamento dos pacotes são instaladas nas tabelas do *switch* P4. Essas regras estão definidas no arquivo `command_s1.txt`.

O código P4 também define a interface entre o *pipeline* do *switch* e o plano de controle. Os comandos CLI no arquivo `command_s1.txt` referem-se a tabelas, chaves e ações, especificadas pelo nome e qualquer alteração que ocorra no programa P4 que adicione ou renomeie uma tabela, chave ou ação, precisa ser refletida nestes comandos.

1.4.1.3. Passo 2: Implementação

O arquivo `task_1.p4` contém o código P4, com as principais partes da lógica a ser implementada, assinaladas com o comentários **PARA COMPLETAR**. A implementação a ser feita deverá seguir a estrutura proposta nesse arquivo, substituindo os comentários pelo código P4 necessário para que a implementação funcione.

O arquivo `task_1.p4` completo deve ter os seguintes componentes:

1. Definições de cabeçalho para as camadas *ethernet* (`ethernet_t`) e IPv4 (`ipv4_t`).
2. **PARA COMPLETAR:** Os *parsers* para *ethernet* e IPv4 que populam os campos de `ethernet_t` e `ipv4_t`.
3. Pelo menos uma ação para descartar os pacotes não reconhecidos usando `mark_to_drop()`.

4. **PARA COMPLETAR:** Uma ação (chamada `ipv4_forward`) que:
 - (a) Determina a porta de saída para o próximo salto;
 - (b) Atualiza o endereço *ethernet* de destino com o endereço do próximo salto;
 - (c) Atualiza o endereço *ethernet* de origem, com o endereço *ethernet* do *switch*;
 - (d) Decrementa o TTL.
5. **PARA COMPLETAR:** Um controle que:
 - (a) Defina a tabela para ler o endereço IPv4 de destino e invoque a ação `drop` ou a ação `ipv4_forward`;
 - (b) Defina um bloco `apply` que referencie a tabela.
6. **PARA COMPLETAR:** Defina um `deparser` que selecione a ordem que cada campo deve ser inserido no pacote que está saindo do *switch*;
7. Um instanciação do pacote composta por `parser`, `control` e `deparser`.

Eventualmente esse processamento também requer instâncias de verificação de *checksum* e controles. Isso não será necessário para essa tarefa e foram substituídas por controles vazios.

1.4.1.4. Passo 3: Execução Final

Siga as instruções do Passo 1 para testar. Após as alterações de código as mensagens do *host h1* deverão ser entregues ao *host h2*.

Reflexões e Desafios

Os programas utilizados para testar a sua implementação não são muito robustos e não suportam o envio de um tráfego mais elevado. Qual seria outra forma de testar a implementação? Além disso, outras questões podem ser consideradas: Como você poderia melhorar o seu código para suportar vários saltos? Essa implementação é suficiente para substituir um roteador comercial? O que estaria faltando?

Solução de Problemas

Existem vários problemas que podem acontecer durante o desenvolvimento da sua implementação:

1. O código `task_1.p4` pode falhar durante a compilação. Nesse caso o *script* `run_task_1.sh` apresentará na tela o erro de compilação e será terminado.
2. O código `task_1.p4` pode compilar, mas pode falhar ao carregar as regras do plano de controle contidas no arquivo `command_s1.txt`, que o *script* `run_task.sh` tentará instalar usando o CLI BMv2. Neste caso, o `run_task.sh` irá gerar mensagens de erro sobre a saída do CLI no diretório de *logs*. Utilize essa mensagem de erro para consertar a sua implementação.

3. O código `task_1.p4` pode compilar e as regras do plano de controle descritas no arquivo `command_s1.txt` podem ser carregadas sem problemas, mas ainda assim, o *switch* P4 pode não conseguir processar os pacotes corretamente. O arquivo `/tmp/p4s.s1.log` contém informações detalhadas descrevendo como o pacote foi processado pelo *switch*. Este *log* pode ajudar a encontrar erros de lógica na sua implementação.

Reiniciando o Mininet

Alguns casos de falha podem ser causados devido a inicialização do ambiente. Nestes casos, reiniciar o Mininet pode ajudar. Utilize o comando `mn -c` para terminar as instâncias que ficaram travadas.

1.4.2. Atividade 2 - Implementando uma Calculadora em P4

O objetivo desta atividade é implementar uma calculadora básica utilizando um cabeçalho customizado escrito em P4. O cabeçalho deve conter a operação a ser executada e dois operandos. Quando o *switch* receber o cabeçalho para calcular, ele deverá executar a operação com os operandos e retornar ao endereço de origem o resultado. Muito embora esta atividade pode não possuir uma aplicação efetivamente prática, ela é muito útil para demonstrar o potencial da linguagem P4 para definir novos cabeçalhos. O diagrama apresentado na Figura 1.11 demonstra a topologia configurada para essa tarefa.



Figura 1.11. Topologia proposta para a atividade 2.

1.4.2.1. Passo 1: Execução Inicial

O diretório `/home/student/labs/task_2/p4src` contém o código P4 `task_2.p4`, e pode ser compilado. No entanto, em seu estado atual ele simplesmente descarta os pacotes que chegam até o *switch*. O objetivo dessa atividade é completar esse código inicial de forma que o *switch* P4 faça o cálculo de acordo com as informações recebidas no cabeçalho e devolva o pacote à sua origem.

Primeiramente compile o código da forma que se encontra e execute o Mininet para testar o seu funcionamento.

Na linha de comando, dentro do diretório `/home/student/labs/task_2/` execute o *script*: `sudo ./run_task.sh`. Esse *script* irá compilar o programa

`task_2.p4` e inicializar o Mininet já com a topologia configurada para a atividade.

A instância do Mininet para essa atividade conta com um *switch* (*s1*) e um *host* (*h1*). O *host* tem o seguintes endereço IP associado: *h1* - 10.0.0.10

Para essa atividade um pequeno programa em Python foi escrito para testar o funcionamento da sua calculadora. O programa será carregado no *host h1* diretamente no *prompt* e permitirá a execução dos testes. O *script* proverá um novo *prompt*, no qual você poderá escrever expressões matemáticas básicas. Essa aplicação vai verificar a expressão e preparar um pacote com o operador e os operandos correspondentes, e então, enviará esse pacote para o *switch P4*. Quando o *switch* retornar o pacote com o resultado da operação, a aplicação imprimirá na tela o valor. Uma vez que a lógica em P4 ainda não foi implementada, você deve ver uma mensagem de erro na tela, conforme exemplo a seguir:

```
> 1+1
Didn't receive response
>
```

1.4.2.2. Passo 2: Implementação

Para a implementação da calculadora, será necessário definir um cabeçalho customizado (Figura 1.12 e definir a lógica no *switch P4* para verificar o cabeçalho, efetuar a operação indicada, escrever o resultado no cabeçalho e retornar o pacote ao seu emissor. O seguinte formato de cabeçalho foi utilizado para essa atividade:

- P é o caracter ASCII 'P' (0x50)
- 4 é o caracter ASCII '4' (0x34)
- Versão é atualmente a 0.1 (0x01)
- Op é a operação a ser efetuada:
- '+' (0x2b) Resultado = Operando A + Operando B
- '-' (0x2d) Resultado = Operando A - Operando B
- '&' (0x26) Resultado = Operando A & Operando B
- '|' (0x7c) Resultado = Operando A | Operando B
- '^' (0x5e) Resultado = Operando A ^ Operando B

Assume-se que o cabeçalho a ser calculado será carregado sobre o protocolo *ethernet*, e que será usado o tipo de *ethernet* 0x1234 para indicar a presença desse cabeçalho. Dado o que foi aprendido até agora, o objetivo é implementar o código P4 para fazer os cálculos. Não existe lógica do plano de controle, então só é necessário a implementação da lógica para o plano de dados.

0	1	2	3
P	4	Versão	Op
Operando A			
Operando B			
Resultado			

Figura 1.12. Cabeçalho customizado para a calculadora P4.

Uma implementação correta deverá verificar os cabeçalhos customizados, executar as operações matemáticas, escrever o resultado no campo apropriado e retornar o pacote ao emissor do mesmo.

1.4.2.3. Passo 3: Execução Final

Siga as instruções do Passo 1. Agora, ao invés de uma mensagem de erro, deverá aparecer o resultado correto da operação.

```
> 1+1
2
>
```

1.4.3. Atividade 3 - Implementação Inicial

O objetivo desta atividade é implementar um esquema de roteamento estabelecido na origem do pacote. Com o roteamento na origem, o *host* de origem especifica para qual porta cada *switch* na rede deverá encaminhar o pacote em cada salto. O *host* de origem estabelece uma série de portas de saída no cabeçalhos do pacote. Neste exemplo, será introduzida a lista das portas para encaminhamento após o cabeçalho *ethernet* e será estabelecido um *etherType* especial para indicar a presença dessa lista. Cada *switch* deverá tomar um item dessa lista e encaminhar o pacote de acordo com o número da porta especificada.

O *switch* P4 deve verificar a lista de roteamento de origem. Cada item tem um *bit* de fim de lista e o número da porta. O *bit* de fim lista será igual a 1 somente no último item da lista. Então, no ingresso do pacote, é necessário tomar um item da lista e configurar a porta de saída de acordo com o valor do item tomado. Note que no último salto o pacote deve ser revertido ao *etherType* `TYPE_IPV4` antes de ser entregue.

O diagrama apresentado na Figura 1.13 demonstra a topologia configurada para essa tarefa.

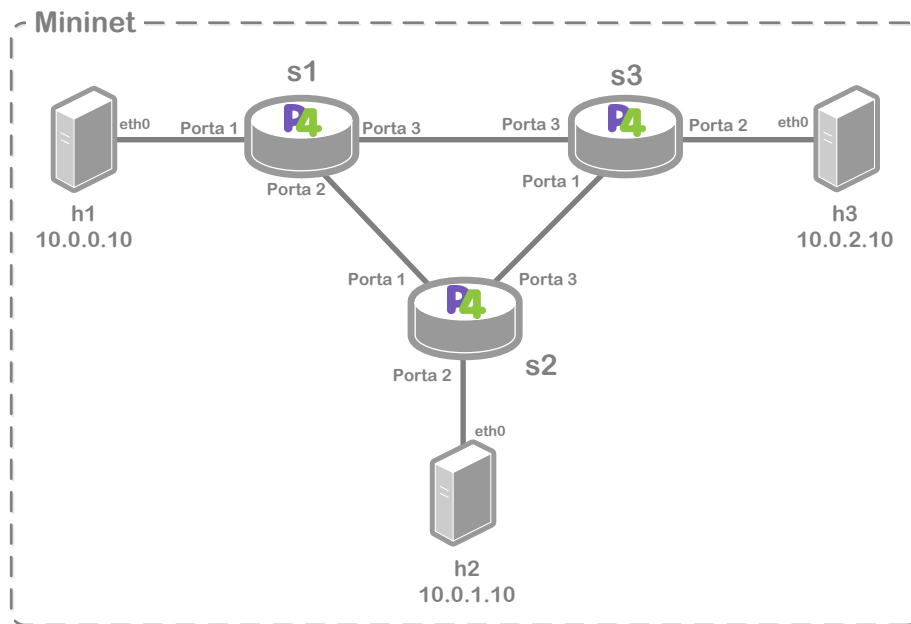


Figura 1.13. Topologia proposta para a atividade 3.

1.4.3.1. Passo 1: Execução

O diretório `/home/student/labs/task_3/p4src` contém o código P4 `task_3.p4`, que pode ser compilado, no entanto, em sua versão original, simplesmente descarta os pacotes que chegam até ele. O objetivo dessa atividade é completar esse código inicial, de forma que o *switch* P4 faça o encaminhamento do pacote ao *host* correto.

Primeiramente, compile o código de forma em que se encontra e execute o Mininet para testar o seu funcionamento.

Na linha de comando, dentro do diretório `/home/student/labs/task_3/` execute o *script*: `sudo ./run_task.sh`. Esse *script* irá compilar o código `task_3.p4` e inicializar o Mininet já com a topologia configurada para a atividade.

A instância do Mininet para essa atividade conta com três *switches* (`s1`, `s2` e `s3`) e três *hosts* (`h1`, `h2` e `h3`). Os *hosts* possuem os seguintes endereços IPs associados: `h1` - 10.0.0.10, `h2` - 10.0.1.10 e `h3` - 10.0.2.10.

Uma vez que a instância do Mininet esteja carregada, obtenha-se o *prompt* `>`. Abra então dois terminais, para `h1` e `h2`, respectivamente:

```
mininet> xterm h1 h2
```

Em cada um dos *hosts* existem aplicações baseadas em Python para enviar e receber mensagens. Inicie primeiramente a aplicação para receber mensagens no terminal do *host* 2 (`h2`).

```
sudo ./receive.py
```

Em seguida, inicie a aplicação para enviar mensagem a partir do *host* 1 (*h1*). A sintaxe do comando é `send.py <ip do receiver>`:

```
sudo ./send.py 10.0.1.10
```

Digite então a lista de roteamento de origem, digamos, 2 3 2 2 1. Essa lista deveria fazer com que o pacote percorra uma rota passando por: *h1*, *s1*, *s2*, *s3*, *s1*, *s2* e *h2*. Entretanto o pacote não será recebido por *h2*.

Saia do *prompt* do *script* Python digitando `q` e então digite `exit` para sair do *xterm* e voltar a linha de comando do Mininet. Então saia do Mininet digitando `exit` na linha de comando.

```
mininet> exit
```

A mensagem não foi entregue porque o *switch* está programado com o código P4 original em `task_3.p4`, que descarta todos os pacotes recebidos. O objetivo dessa tarefa é estender o código P4 para que os pacotes consigam alcançar o seu destino.

1.4.3.2. Passo 2: Implementação

O arquivo `task_3.p4` contém o código P4 com indicações das partes a serem implementadas, assinaladas com o comentários **PARA COMPLETAR**. A implementação a ser feita deverá seguir a estrutura proposta nesse arquivo, substituindo os comentários pelo código P4 necessário para que a implementação funcione.

O arquivo `task_3.p4` completo deve ter os seguintes componentes:

1. Definições de cabeçalho para as camadas *ethernet* (`ethernet_t`), IPv4 (`ipv4_t`) e o roteamento de origem (`srcRoute_t`).
2. **PARA COMPLETAR:** Os *parsers* para *ethernet*, IPv4 e roteamento de origem que populam os campos de `ethernet_t`, `ipv4_t` e `srcRoute_t`.
3. Pelo menos uma ação para descartar os pacotes, usando `mark_to_drop()`.
4. **PARA COMPLETAR:** Uma ação (chamada `srcRoute_nhop`) que irá:
 - (a) Determinar a porta de saída para o próximo salto;
 - (b) Remover o primeiro item da lista roteamento de origem.
5. Um controle com um bloco de aplicação que:
 - (a) Verifica a existência das rotas de origem.
 - (b) **PARA COMPLETAR:** Verifica se o `ethernet.etherType` deve ser alterado, caso o pacote esteja sendo enviado ao último *host*.
 - (c) **PARA COMPLETAR:** Chamar a ação `srcRoute_nhop`
6. Um *deparser* que selecione a ordem que cada campo deve ser inserido no pacote que está saindo do *switch*.

7. Um instanciação do pacote composta por `parser`, `control` e `deparser`.

Eventualmente esse processamento também requer instâncias de verificação de *checksum* e controles de recálculo. Isso não será necessário para essa tarefa e foram substituídas com controles vazios.

1.4.3.3. Passo 3: Execução Final

Siga as instruções do Passo 1. Agora as mensagens do *host h1* deverão ser entregues ao *host h2*.

Verifique o TTL do cabeçalho IP. A cada salto o TTL será decrementado. A sequência de portas 2 3 2 2 1 força o pacote e fazer uma volta a mais na rede (*loop*), então o TTL deverá ser 59 no *host h2*.

Reflexões

- É possível alterar a lógica do código P4 para manipular o encaminhamento IPv4 e o roteamento de origem ao mesmo tempo?
- Como você poderia melhorar seu código para permitir que o primeiro *switch* adicionasse a rota ao pacote, tornando o roteamento de origem transparente para todos os *hosts*?

Solução de Problemas

Existem vários problemas que podem acontecer durante o desenvolvimento da implementação:

1. O código `task_3.p4` pode falhar durante a compilação. Nesse caso o *script* `run_task_3.sh` apresentará na tela o erro de compilação e será terminado.
2. O código `task_3.p4` pode compilar, mas falhar ao carregar as regras do plano de controle descritas no arquivo `command_s<número_do_switch>.txt` que o *script* `run_task.sh` tentará instalar usando o CLI BMv2. Neste caso, o `run_task.sh` irá registrar a saída do CLI no diretório de *logs*. Utilize essa mensagem de erro para consertar a sua implementação.
3. O código `task_3.p4` pode compilar, e as regras do plano de controle descritas no arquivo `command_s1.txt` podem ser carregadas sem problemas, mas ainda assim, o *switch* P4 pode não conseguir processar os pacotes corretamente. O arquivo `/tmp/p4s.s1.log`, contém informações detalhadas descrevendo como o pacote foi processado pelo *switch*. Este *log* pode ajudar a encontrar erros de lógica na sua implementação. Um captura de pacotes também será gerada para as interfaces de entrada e saída de cada um dos *switchs* no formato `<switch-name>-<interface-name>.pcap`. Utilize o comando `tcpdump -r <filename> -xxx` e imprima a saída em hexadecimal (*hexdump*) dos pacotes capturados.

Reiniciando o Mininet

Alguns casos de falha podem ser causados devido a inicialização do ambiente. Nestes casos, reiniciar o Mininet pode ajudar. Utilize o comando `mn -c` para terminar as instâncias que ficaram travadas.

1.4.4. Respostas Sugeridas

Não existe somente uma resposta correta para resolver as atividades propostas. A seguir são apresentadas uma das possíveis soluções para cada uma das atividades.

1.4.4.1. Atividade 1 - Encaminhamento de Pacotes

```
/* -*- P4_16 -*- */
#include <core.p4>
#include <vlmodel.p4>

const bit<16> TYPE_IPV4 = 0x800;
*****
***** H E A D E R S *****
*****/

typedef bit<9>  egressSpec_t;
typedef bit<48> macAddr_t;
typedef bit<32> ip4Addr_t;

header ethernet_t {
    macAddr_t dstAddr;
    macAddr_t srcAddr;
    bit<16> etherType;
}

header ipv4_t {
    bit<4> version;
    bit<4> ihl;
    bit<8> diffserv;
    bit<16> totalLen;
    bit<16> identification;
    bit<3> flags;
    bit<13> fragOffset;
    bit<8> ttl;
    bit<8> protocol;
    bit<16> hdrChecksum;
    ip4Addr_t srcAddr;
    ip4Addr_t dstAddr;
}

struct metadata {
    /* empty */
}

struct headers {
    ethernet_t ethernet;
    ipv4_t ipv4;
```

```

}

/*****
***** P A R S E R *****/

parser MyParser(packet_in packet,
                 out headers hdr,
                 inout metadata meta,
                 inout standard_metadata_t standard_metadata) {

    state start {
        transition parse_ethernet;
    }

    state parse_ethernet {
        packet.extract(hdr.ethernet);
        transition select(hdr.ethernet.etherType) {
            TYPE_IPV4: parse_ipv4;
            default: accept;
        }
    }

    state parse_ipv4 {
        packet.extract(hdr.ipv4);
        transition accept;
    }

}

/*****
***** C H E C K S U M   V E R I F I C A T I O N *****/

control MyVerifyChecksum(inout headers hdr, inout metadata meta)
{
    apply { }
}

/*****
***** I N G R E S S   P R O C E S S I N G *****/

control MyIngress(inout headers hdr,
                  inout metadata meta,
                  inout standard_metadata_t standard_metadata) {

    action drop() {
        mark_to_drop();
    }

    action ipv4_forward(macAddr_t dstAddr, egressSpec_t port) {
        standard_metadata.egress_spec = port;
        hdr.ethernet.srcAddr = hdr.ethernet.dstAddr;
        hdr.ethernet.dstAddr = dstAddr;
        hdr.ipv4.ttl = hdr.ipv4.ttl - 1;
    }
}

```

```

    }

    table ipv4_lpm {
        key = {
            hdr.ipv4.dstAddr: lpm;
        }
        actions = {
            ipv4_forward;
            drop;
            NoAction;
        }
        size = 1024;
        default_action = NoAction();
    }

    apply {
        if (hdr.ipv4.isValid()) {
            ipv4_lpm.apply();
        }
    }
}

/***** E G R E S S   P R O C E S S I N G *****/

control MyEgress(inout headers hdr,
                 inout metadata meta,
                 inout standard_metadata_t standard_metadata) {
    apply { }
}

/***** C H E C K S U M   C O M P U T A T I O N *****/

control MyComputeChecksum(inout headers  hdr, inout metadata meta
) {
    apply {
        update_checksum(
            hdr.ipv4.isValid(),
            { hdr.ipv4.version,
              hdr.ipv4.ihl,
              hdr.ipv4.diffserv,
              hdr.ipv4.totalLen,
              hdr.ipv4.identification,
              hdr.ipv4.flags,
              hdr.ipv4.fragOffset,
              hdr.ipv4.ttl,
              hdr.ipv4.protocol,
              hdr.ipv4.srcAddr,
              hdr.ipv4.dstAddr },
            hdr.ipv4.hdrChecksum,
            HashAlgorithm.csum16);
    }
}

```

```

}

/*****
*****  D E P A R S E R  *****/
*****/

control MyDeparser(packet_out packet, in headers hdr) {
    apply {
        packet.emit(hdr.ethernet);
        packet.emit(hdr.ipv4);
    }
}

/*****
*****  S W I T C H  *****/
*****/

VlSwitch(
MyParser(),
MyVerifyChecksum(),
MyIngress(),
MyEgress(),
MyComputeChecksum(),
MyDeparser()
) main;

```

1.4.4.2. Atividade 2 - Implementando uma calculadora em P4

```

/* -*- P4_16 -*- */

/*
 * P4 Calculator
 *
 * This program implements a simple protocol. It can be carried
 * over Ethernet
 * (Ethertype 0x1234).
 *
 * The Protocol header looks like this:
 *
 *      0          1          2          3
 *  +-----+-----+-----+-----+
 *  |      P      |      4      | Version |      Op      |
 *  +-----+-----+-----+-----+
 *  |                                     Operand A      |
 *  +-----+-----+-----+-----+
 *  |                                     Operand B      |
 *  +-----+-----+-----+-----+
 *  |                                     Result           |
 *  +-----+-----+-----+-----+
 *
 * P is an ASCII Letter 'P' (0x50)
 * 4 is an ASCII Letter '4' (0x34)
 * Version is currently 0.1 (0x01)
 * Op is an operation to Perform:
 * '+' (0x2b) Result = OperandA + OperandB

```

```

*   '-' (0x2d) Result = OperandA - OperandB
*   '&' (0x26) Result = OperandA & OperandB
*   '|' (0x7c) Result = OperandA | OperandB
*   '^' (0x5e) Result = OperandA ^ OperandB
*
*   The device receives a packet, performs the requested operation
*   , fills in the
*   result and sends the packet back out of the same port it came
*   in on, while
*   swapping the source and destination addresses.
*
*   If an unknown operation is specified or the header is not
*   valid, the packet
*   is dropped
*/

#include <core.p4>
#include <vlmodel.p4>

/*
 * Define the headers the program will recognize
 */

/*
 * Standard ethernet header
 */
header ethernet_t {
    bit<48> dstAddr;
    bit<48> srcAddr;
    bit<16> etherType;
}

/*
 * This is a custom protocol header for the calculator. We'll use
 * ethertype 0x1234 for is (see parser)
 */
const bit<16> P4CALC_ETYPE = 0x1234;
const bit<8>  P4CALC_P      = 0x50;    // 'P'
const bit<8>  P4CALC_4      = 0x34;    // '4'
const bit<8>  P4CALC_VER    = 0x01;    // v0.1
const bit<8>  P4CALC_PLUS   = 0x2b;    // '+'
const bit<8>  P4CALC_MINUS  = 0x2d;    // '-'
const bit<8>  P4CALC_AND    = 0x26;    // '&'
const bit<8>  P4CALC_OR     = 0x7c;    // '|'
const bit<8>  P4CALC_CARET  = 0x5e;    // '^'

header p4calc_t {
    bit<8>  p;
    bit<8>  four;
    bit<8>  ver;
    bit<8>  op;
    bit<32> operand_a;
    bit<32> operand_b;
    bit<32> res;
}

```



```

/*
 * All headers, used in the program needs to be assembled into a
 * single struct.
 * We only need to declare the type, but there is no need to
 * instantiate it,
 * because it is done "by the architecture", i.e. outside of P4
 * functions
 */
struct headers {
    ethernet_t    ethernet;
    p4calc_t      p4calc;
}

/*
 * All metadata, globally used in the program, also needs to be
 * assembled
 * into a single struct. As in the case of the headers, we only
 * need to
 * declare the type, but there is no need to instantiate it,
 * because it is done "by the architecture", i.e. outside of P4
 * functions
 */

struct metadata {
    /* In our case it is empty */
}

/***** P A R S E R *****/
parser MyParser(packet_in packet,
                out headers hdr,
                inout metadata meta,
                inout standard_metadata_t standard_metadata) {

    state start {
        packet.extract(hdr.ethernet);
        transition select(hdr.ethernet.etherType) {
            P4CALC_ETYPE : check_p4calc;
            default      : accept;
        }
    }

    state check_p4calc {
        transition select(packet.lookahead<p4calc_t>().p,
            packet.lookahead<p4calc_t>().four,
            packet.lookahead<p4calc_t>().ver) {
            (P4CALC_P, P4CALC_4, P4CALC_VER) : parse_p4calc;
            default                          : accept;
        }
    }

    state parse_p4calc {
        packet.extract(hdr.p4calc);
    }
}

```

```

        transition accept;
    }
}

/***** C H E C K S U M   V E R I F I C A T I O N *****/
control MyVerifyChecksum(inout headers hdr,
                        inout metadata meta) {
    apply { }
}

/***** I N G R E S S   P R O C E S S I N G *****/
control MyIngress(inout headers hdr,
                 inout metadata meta,
                 inout standard_metadata_t standard_metadata) {

    action send_back(bit<32> result) {
        bit<48> tmp;

        /* Put the result back in */
        hdr.p4calc.res = result;

        /* Swap the MAC addresses */
        tmp = hdr.ethernet.dstAddr;
        hdr.ethernet.dstAddr = hdr.ethernet.srcAddr;
        hdr.ethernet.srcAddr = tmp;

        /* Send the packet back to the port it came from */
        standard_metadata.egress_spec = standard_metadata.
            ingress_port;
    }

    action operation_add() {
        send_back(hdr.p4calc.operand_a + hdr.p4calc.operand_b);
    }

    action operation_sub() {
        send_back(hdr.p4calc.operand_a - hdr.p4calc.operand_b);
    }

    action operation_and() {
        send_back(hdr.p4calc.operand_a & hdr.p4calc.operand_b);
    }

    action operation_or() {
        send_back(hdr.p4calc.operand_a | hdr.p4calc.operand_b);
    }

    action operation_xor() {
        send_back(hdr.p4calc.operand_a ^ hdr.p4calc.operand_b);
    }
}

```

```

    action operation_drop() {
        mark_to_drop();
    }

    table calculate {
        key = {
            hdr.p4calc.op          : exact;
        }
        actions = {
            operation_add;
            operation_sub;
            operation_and;
            operation_or;
            operation_xor;
            operation_drop;
        }
        const default_action = operation_drop();
        const entries = {
            P4CALC_PLUS : operation_add();
            P4CALC_MINUS: operation_sub();
            P4CALC_AND   : operation_and();
            P4CALC_OR    : operation_or();
            P4CALC_CARET: operation_xor();
        }
    }

    apply {
        if (hdr.p4calc.isValid()) {
            calculate.apply();
        } else {
            operation_drop();
        }
    }
}

/***** EGRESS PROCESSING *****/
control MyEgress(inout headers hdr,
                 inout metadata meta,
                 inout standard_metadata_t standard_metadata) {
    apply { }
}

/***** CHECKSUM COMPUTATION *****/

control MyComputeChecksum(inout headers hdr, inout metadata meta)
{
    apply { }
}

/*****

```

```

***** D E P A R S E R *****
*****/
control MyDeparser(packet_out packet, in headers hdr) {
    apply {
        packet.emit(hdr.ethernet);
        packet.emit(hdr.p4calc);
    }
}

/*****
***** S W I T T C H *****
*****/

VlSwitch(
MyParser(),
MyVerifyChecksum(),
MyIngress(),
MyEgress(),
MyComputeChecksum(),
MyDeparser()
) main;

```

1.4.4.3. Atividade 3 - Implementando roteamento na origem

```

/* -*- P4_16 -*- */
#include <core.p4>
#include <vlmodel.p4>

const bit<16> TYPE_IPV4 = 0x800;
const bit<16> TYPE_SRCROUTING = 0x1234;

#define MAX_HOPS 9

/*****
***** H E A D E R S *****
*****/

typedef bit<9> egressSpec_t;
typedef bit<48> macAddr_t;
typedef bit<32> ip4Addr_t;

header ethernet_t {
    macAddr_t dstAddr;
    macAddr_t srcAddr;
    bit<16> etherType;
}

header srcRoute_t {
    bit<1> bos;
    bit<15> port;
}

header ipv4_t {
    bit<4> version;
    bit<4> ihl;
}

```

```

    bit<8>    diffserv;
    bit<16>   totalLen;
    bit<16>   identification;
    bit<3>    flags;
    bit<13>   fragOffset;
    bit<8>    ttl;
    bit<8>    protocol;
    bit<16>   hdrChecksum;
    ip4Addr_t srcAddr;
    ip4Addr_t dstAddr;
}

struct metadata {
    /* empty */
}

struct headers {
    ethernet_t      ethernet;
    srcRoute_t[MAX_HOPS] srcRoutes;
    ipv4_t          ipv4;
}

/***** P A R S E R *****/

parser MyParser(packet_in packet,
                 out headers hdr,
                 inout metadata meta,
                 inout standard_metadata_t standard_metadata) {

    state start {
        transition parse_ethernet;
    }

    state parse_ethernet {
        packet.extract(hdr.ethernet);
        transition select(hdr.ethernet.etherType) {
            TYPE_SRCROUTING: parse_srcRouting;
            default: accept;
        }
    }

    state parse_srcRouting {
        packet.extract(hdr.srcRoutes.next);
        transition select(hdr.srcRoutes.last.bos) {
            1: parse_ipv4;
            default: parse_srcRouting;
        }
    }

    state parse_ipv4 {
        packet.extract(hdr.ipv4);
        transition accept;
    }
}

```

```

}

/***** C H E C K S U M       V E R I F I C A T I O N *****/
control MyVerifyChecksum(inout headers hdr, inout metadata meta)
{
    apply { }
}

/***** I N G R E S S       P R O C E S S I N G *****/
control MyIngress(inout headers hdr,
                  inout metadata meta,
                  inout standard_metadata_t standard_metadata) {

    action drop() {
        mark_to_drop();
    }

    action srcRoute_nhop() {
        standard_metadata.egress_spec = (bit<9>)hdr.srcRoutes[0].
            port;
        hdr.srcRoutes.pop_front(1);
    }

    action srcRoute_finish() {
        hdr.ethernet.etherType = TYPE_IPV4;
    }

    action update_ttl(){
        hdr.ipv4.ttl = hdr.ipv4.ttl - 1;
    }

    apply {
        if (hdr.srcRoutes[0].isValid()){
            if (hdr.srcRoutes[0].bos == 1){
                srcRoute_finish();
            }
            srcRoute_nhop();
            if (hdr.ipv4.isValid()){
                update_ttl();
            }
        }else{
            drop();
        }
    }
}

/***** E G R E S S       P R O C E S S I N G *****/

```

```

    *****/

    control MyEgress(inout headers hdr,
                    inout metadata meta,
                    inout standard_metadata_t standard_metadata) {
        apply { }
    }

    /***** C H E C K S U M      C O M P U T A T I O N      *****/
    *****/

    control MyComputeChecksum(inout headers  hdr, inout metadata meta
    ) {
        apply { }
    }

    /***** D E P A R S E R      *****/
    *****/

    control MyDeparser(packet_out packet, in headers hdr) {
        apply {
            packet.emit(hdr.ethernet);
            packet.emit(hdr.srcRoutes);
            packet.emit(hdr.ipv4);
        }
    }

    /***** S W I T C H      *****/
    *****/

    V1Switch(
    MyParser(),
    MyVerifyChecksum(),
    MyIngress(),
    MyEgress(),
    MyComputeChecksum(),
    MyDeparser()
    ) main;

```

1.5. Tendências e Conclusões

O P4 vem como uma segunda onda de transformação nas redes de computadores, segue a trilha aberta pela mudança de paradigma arquitetônico trazida pelo SDN. A separação entre o plano de controle e o plano de dados foi estabelecida pelo SDN, mas a programabilidade da rede ainda era papel quase exclusivo do plano de controle. Apesar do sucesso das implementações feitas pela primeira onda liderada pelo OpenFlow, a questão da programabilidade direta do plano de dados ainda não tinha sido resolvida de forma apropriada.

Sínteses das funções de rede no plano de dados vinham sendo implementadas por programas escritos em linguagens de alto nível, daí havia uma posterior compilação

em linguagem da plataforma alvo, como o Verilog e VHDL. Todavia, a expressividade requerida para a programabilidade de rede no plano de dados sofria com a falta de bibliotecas e módulos que combinassem as abstrações comuns desses módulos com objetos para desenvolver funcionalidades específicas. Some-se a isso fatores complicantes como a ausência de camadas intermediárias, e.g., sistema operacional e *drivers*, que nos habituamos em sistemas baseados em CPUs. Portanto, havia uma exposição de complexidade do hardware ao usuário dessas plataformas baseadas em FPGA, por exemplo. Isso criava uma barreira de entrada difícil de ser transposta para a implementação de funções de rede no plano de dados [Sultana et al. 2017].

É importante também entender que a fase de demonstração do potencial das novas funcionalidades de rede pelo uso de SDN/OpenFlow já foi vencida, e desempenho é o que se espera alcançar no momento. Busca-se também redução do tempo requerido prototipação de serviços de rede para acelerar os ciclos de inovação. Porém, nota-se a ausência de plataformas convenientes de *debug* para agilizar desenvolvimento de protótipos quando seguimos as soluções convencionais descritas acima.

Como uma proposta para preencher essas lacunas surge a linguagem P4 para, finalmente, habilitar programabilidade do plano de dados de dispositivos de rede. A baixa barreira de entrada e suas descrições de funcionalidades de rede independentes da plataforma alvo podemos resumir como os principais atrativos.

A linguagem P4 traz portanto benefícios para as inúmeras inovações funcionais já feitas em SDN que agora podem ser transferidas para o plano de dados de forma a melhorar desempenho. Todavia, a sua expressividade também viabiliza novas soluções para problemas clássicos. Por exemplo, medições ampliadas à toda rede valendo-se de monitoramento simplificado usando estruturas de dados probabilísticas (*i.e.*, *sketches*) [Liu et al. 2016] e balanceamento de carga sem manutenção de estados em *datacenters* [Olteanu et al. 2018].

Para analisarmos as tendências, primeiro temos que entender que o sucesso da linguagem P4 vai depender, naturalmente, de adoção ampla por fabricantes de equipamentos, de criação e da manutenção de comunidades para a difusão de soluções básicas e na criação de bibliotecas. Pela origem comum, acreditamos que os mesmos caminhos eficientes de difusão das soluções OpenFlow sejam empregados também para as desenvolvidas via linguagem P4.

Notamos um número crescente de compiladores para a linguagem P4, o que pode ser um importante sinal de interesse da comunidade e as variadas plataformas alvo existente ajudam muito na difusão da linguagem. Há inclusive propostas de criação de uma *benchmark suite* para permitir comparação entre as diversas soluções de compilação [Dang et al. 2017].

Outro fator importante é que o P4 vem gradualmente casando o seu nível de expressividade com o da linguagem nativa da plataforma alvo. Discussões sobre a tradução de P4 para descrição de hardware (VHDL) pode ser encontrada em [Benek et al. 2018]. O objetivo buscado é, naturalmente, o aumento de eficiência das implementações. Funcionalidades de alto desempenho (*i.e.*, em 100Gbps) podem ser daí geradas automaticamente para componentes principais de processamento de pacotes, como apresenta essa outra ini-

ciativa recente [Benáček et al. 2018].

Há ainda uma nova plataforma de alto desempenho que serve como alternativa ao uso de *switches bare metal* e FPGAs. A incorporação de programabilidade em P4 às interfaces de rede de servidores já é uma realidade via *SmartNICs*. Isso torna possível elevar implementações flexíveis de programação de rede, que são usualmente feitas com software *switches* dos servidores, ao patamar de alto desempenho - porém a custos bem reduzidos.

O espaço dos software *switches* com P4 também está bem ativo, afinal a expressividade permitida pelo P4 tem um papel importantíssimo no contexto de computação em nuvem, onde redes são cada vez mais compostas com elementos virtualizados. Neste cenário há trabalhos de extensão do OvS tradicional para a inclusão, por exemplo, de filtros com estados e funções de monitoramento em tempo de execução via P4 [Chaignon et al. 2018]. De outro lado, temos propostas como o PISCES [Shahbaz et al. 2016] que é um protótipo independente que pode ser programado a partir da linguagem P4.

Um outro sinal de vitalidade de novas propostas é o surgimento de replicas de ferramentas bem difundidas na comunidade de redes, e também o desenvolvimento de capacidade de se implementar testes, monitoramentos e validações para vencer as limitações de *debug* comentadas anteriormente. Um exemplo recente nessa linha é o *p4pktgen* [Nötzli et al. 2018], uma ferramenta automática de geração de casos de teste, e com capacidade de avaliação de códigos P4 usando execução simbólica.

O surgimento de novas soluções, antes limitadas pelo excesso de estados a serem mantido no plano de dados, dão prova do potencial futuro do P4 na resolução de problemas complexos [Jepsen et al. 2018]. Com capacidade de processamento em algumas aplicações já ultrapassando Terabits por segundo [Cabal et al. 2018], a utilização da linguagem P4 claramente eleva o patamar dos protótipos produzidos, facilitando a transição de soluções implementadas para a fase de pilotos e daí para produtos, reduzindo o tempo dos ciclos de inovação em redes de computadores. Isso permite avançar em diversas áreas de aplicação, como por exemplo naquelas pouco exploradas como a aviônica [Geyer and Winkel 2018].

A comunidade de redes deve atentar, entretanto, para o fato de que à medida que as soluções abertas aproximam-se dos níveis mais baixos, barreiras são erguidas por fornecedores de hardware com suas usuais licenças de utilização. E isso pode frustrar iniciativas de inovação lembrando-nos que a fronteira da *opensource* precisa avançar ainda mais.

Por último, não podemos deixar de comentar que há críticas apontadas à solução baseada em linguagem especializada e, portanto, engessada num paradigma específico de processamento de pacotes; como é o caso da linguagem P4. Argumenta-se que o baixo nível de abstração do P4 inibe o desenvolvimento de funcionalidades mais abrangentes. A plataforma *Emu* [Wang et al. 2017], por exemplo, alega desempenho similar ao P4 utilizando-se bibliotecas padronizadas para a criação ágil de funcionalidades de rede. Como no P4, as plataformas alvo do *Emu* podem ser CPUs, Mininet e FPGAs; porém desfrutando de ambiente de *debug* mais rico. Indicamos ainda a Tabela 1 de [Wang et al. 2017] para o leitor interessado numa discussão comparativa do *Emu* com o P4 e outras ferramentas que também habilitam serviços de rede implementados em hardware.

Referências

- [Benek et al. 2018] Benek, P., Pu, V., Kubtov, H., and ejka, T. (2018). P4-to-vhdl. *Microprocess. Microsyst.*, 56(C):22–33.
- [Benáček et al. 2018] Benáček, P., Puš, V., Kubátová, H., and Čejka, T. (2018). P4-to-vhdl: Automatic generation of high-speed input and output network blocks. *Microprocessors and Microsystems*, 56:22 – 33.
- [Bosshart et al. 2014] Bosshart, P., Daly, D., Gibb, G., Izzard, M., McKeown, N., Rexford, J., Schlesinger, C., Talayco, D., Vahdat, A., Varghese, G., and Walker, D. (2014). P4: Programming protocol-independent packet processors. *SIGCOMM Comput. Commun. Rev.*, 44(3):87–95.
- [Cabal et al. 2018] Cabal, J., Benáček, P., Kekely, L., Kekely, M., Puš, V., and Kořenek, J. (2018). Configurable fpga packet parser for terabit networks with guaranteed wire-speed throughput. In *Proceedings of the 2018 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, FPGA '18*, pages 249–258, New York, NY, USA. ACM.
- [Carmelo Cascone 2016] Carmelo Cascone (2016). P4 support in ONOS. <https://wiki.onosproject.org>.
- [Chaignon et al. 2018] Chaignon, P., Lazri, K., François, J., Delmas, T., and Festor, O. (2018). Oko: Extending open vswitch with stateful filters. In *Proceedings of the Symposium on SDN Research, SOSR '18*, pages 13:1–13:13, New York, NY, USA. ACM.
- [Changhoon Kim 2016] Changhoon Kim (2016). Programming the network dataplane. http://netseminar.stanford.edu/seminars/03_31_16.pdf.
- [Consortium 2017] Consortium, T. P. L. (2017). The P4 Language Specification - version 1.0.0. Specification, The P4 Language Consortium, <https://p4.org/p4-spec/docs/P4-16-v1.0.0-spec.pdf>.
- [Dang et al. 2017] Dang, H. T., Wang, H., Jepsen, T., Brebner, G., Kim, C., Rexford, J., Soulé, R., and Weatherspoon, H. (2017). Whippersnapper: A p4 language benchmark suite. In *Proceedings of the Symposium on SDN Research, SOSR '17*, pages 95–101, New York, NY, USA. ACM.
- [Geyer and Winkel 2018] Geyer, F. and Winkel, M. (2018). Towards Embedded Packet Processing Devices for Rapid Prototyping of Avionic Applications. In *9th European Congress on Embedded Real Time Software and Systems (ERTS 2018)*, Toulouse, France.
- [Ibanez 2017] Ibanez, S. (2017). P4 netfpga tutorial. https://github.com/NetFPGA/P4-NetFPGA-public/blob/master/slides/P4-NetFPGA-SIGCOMM-17/P4-NetFPGA_SIGCOMM_2017_v2.pdf. [Accesado 9-Nov-2017].

- [Jepsen et al. 2018] Jepsen, T., Moshref, M., Carzaniga, A., Foster, N., and Soulé, R. (2018). Life in the fast lane: A line-rate linear road. In *Proceedings of the Symposium on SDN Research, SOSR '18*, pages 10:1–10:7, New York, NY, USA. ACM.
- [Liu et al. 2016] Liu, Z., Manousis, A., Vorsanger, G., Sekar, V., and Braverman, V. (2016). One sketch to rule them all: Rethinking network flow monitoring with univ-mon. In *Proceedings of the 2016 ACM SIGCOMM Conference, SIGCOMM '16*, pages 101–114, New York, NY, USA. ACM.
- [McKeown et al. 2008] McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., and Turner, J. (2008). Openflow: Enabling innovation in campus networks. *SIGCOMM Comput. Commun. Rev.*, 38(2):69–74.
- [Nötzli et al. 2018] Nötzli, A., Khan, J., Fingerhut, A., Barrett, C., and Athanas, P. (2018). P4pktgen: Automated test case generation for p4 programs. In *Proceedings of the Symposium on SDN Research, SOSR '18*, pages 5:1–5:7, New York, NY, USA. ACM.
- [Olteanu et al. 2018] Olteanu, V., Agache, A., Voinescu, A., and Raiciu, C. (2018). Stateless datacenter load-balancing with beamer. In *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)*, pages 125–139. USENIX.
- [Shahbaz et al. 2016] Shahbaz, M., Choi, S., Pfaff, B., Kim, C., Feamster, N., McKeown, N., and Rexford, J. (2016). Pisces: A programmable, protocol-independent software switch. In *Proceedings of the 2016 ACM SIGCOMM Conference, SIGCOMM '16*, pages 525–538, New York, NY, USA. ACM.
- [Sultana et al. 2017] Sultana, N., Galea, S., Greaves, D., Wojcik, M., Shipton, J., Clegg, R., Mai, L., Bressana, P., Soulé, R., Mortier, R., Costa, P., Pietzuch, P., Crowcroft, J., Moore, A. W., and Zilberman, N. (2017). Emu: Rapid prototyping of networking services. In *2017 USENIX Annual Technical Conference (USENIX ATC 17)*, pages 459–471, Santa Clara, CA. USENIX Association.
- [Wang et al. 2017] Wang, H., Soulé, R., Dang, H. T., Lee, K. S., Shrivastav, V., Foster, N., and Weatherspoon, H. (2017). P4fpga: A rapid prototyping framework for p4. In *Proceedings of the Symposium on SDN Research, SOSR '17*, pages 122–135, New York, NY, USA. ACM.

Capítulo

5

Blockchain e a Revolução do Consenso sob Demanda

Fabíola Greve, Leobino Sampaio, Jauberth Abijaude, Antonio Coutinho, Ítalo Valcy, Sílvio Queiroz

Abstract

Blockchain introduces a new on-demand consensus paradigm where the set of P2P network nodes confirm the order in which the transaction blocks are aggregated into the chain of blocks, providing a reliable, secure, scalable, and immutable distributed environment for the transactions' execution on the Internet. The blockchain eliminates the need for a trusted third party and digitally creates a decentralized trust entity. The disruptive character of the technology is vast and applications are emerging in several areas: finance, health, arts, government, etc., beyond computing itself. A number of challenges exist and involve security, privacy, storage, availability, performance improvement, scalability, sustainability, etc.. This chapter presents the blockchain, its main elements, properties, models, algorithms and challenges. Special emphasis will be given to distributed consensus protocols for the maintenance of the replicated state machine, as well as the use of the technology in the context of computer networks, fog computing and IoT.

Resumo

A blockchain introduz o novo paradigma do consenso sob demanda, onde o conjunto de nós da rede P2P concorda com a ordem em que blocos de transações vão sendo agregados à corrente de blocos, proporcionando um ambiente distribuído confiável, seguro, escalável e imutável para a realização e armazenamento de transações na Internet. A blockchain elimina a necessidade de uma terceira parte confiável e cria digitalmente uma entidade de confiança descentralizada. O caráter disruptivo da tecnologia é imenso e aplicações estão surgindo em diversas áreas: finanças, saúde, artes, governo, etc., além da própria computação. Diversos desafios estão postos e envolvem segurança, privacidade, armazenamento, disponibilidade, melhoria de desempenho, crescimento em escala, sustentabilidade, etc.. Este capítulo apresenta a blockchain, seus principais elementos, propriedades, modelos, algoritmos e desafios. Especial ênfase será dada aos protocolos de consenso distribuído para manutenção da máquina de estados replicada, além do uso da tecnologia no contexto de redes de computadores, computação em névoa e IoT.

5.1. Introdução

Blockchain é uma tecnologia emergente que oferece suporte distribuído confiável e seguro para realização de transações entre participantes que não necessariamente têm confiança entre si e que estão dispersos em larga escala numa rede P2P. É considerada uma tecnologia disruptiva, pois cria digitalmente uma entidade de confiança descentralizada, eliminando a necessidade de uma terceira parte de confiança. Dessa forma, pode substituir entidades certificadoras e centralizadoras das transações de negócios, tais como bancos, governos, cartórios, etc. O potencial de transformação é imenso e aplicações estão surgindo a partir desta tecnologia em inúmeros setores: finanças, saúde, artes, governo, além da própria computação: protocolos de redes, nuvem e névoa, IoT, etc.

A blockchain implementa uma *máquina de estados replicada* para a manutenção consistente de um estado global compartilhado por um conjunto de pares distribuídos numa rede P2P. Todos os nós possuem e mantêm uma réplica do registro de transações efetuadas, materializado na forma de um livro-razão (*ledger*) distribuído, que é imutável, pode ser verificado e auditado, e está sempre disponível. O conjunto e a ordem em que as transações são executadas é acordada por todos os participantes da rede, através da realização de um protocolo de *consenso byzantino*, tolerante a ações de nós maliciosos, passíveis de subverter o sistema. O consenso é um elemento fundamental para o desenvolvimento de sistemas confiáveis e seguros, pois possibilita com que os participantes de uma computação concordem com as ações que serão realizadas, com o intuito de manter a consistência do sistema e de fazê-lo progredir. Mecanismos de *criptografia* são empregados para garantir a autoridade, autenticidade, não-repúdio, integridade das transações, bem como os requisitos de segurança de todo o sistema.

A blockchain é resultado de uma engenhosa combinação de técnicas robustas provenientes da computação distribuída confiável (tolerância a falhas byzantinas, sistemas P2P), criptografia (chave assimétrica, funções hash, desafios criptográficos) e teoria dos jogos (mecanismos de incentivos). Agrega elementos eficazes para a implementação de um sistema de acordo em escala global, trazendo respostas para importantes desafios computacionais, dentre os principais, destacamos: (i) realização de consenso numa rede aberta, com participantes desconhecidos, em escala planetária; (ii) tolerância a falhas byzantinas, com participantes anônimos; (iii) resistência ao ataque de duplo-gasto (*double-spending*), garantindo que ativos transacionados (como moedas digitais) não sejam gastos duplamente, para além do seu valor em posse; (iv) resistência a ataques Sybil, garantindo que usuários maliciosos não poderão se personificar em outros (a menos que tenham acesso a sua chave privada); (v) garantia da auditabilidade, autenticidade, não-repúdio e integridade em escala global de todas as transações validadas e armazenadas no livro-razão distribuído.

Na origem da blockchain, está o protocolo do Bitcoin, proposto por Satoshi Nakamoto [Nakamoto 2008], que entrou em operação em 2009. O artigo seminal propõe uma rede P2P onde transações com a criptomoeda (moeda digital) bitcoin, propostas por clientes, são recebidas por servidores (precisamente, mineradores), que irão decidir, através de um protocolo de consenso byzantino à base de desafios criptográficos, sobre a ordem em que serão realizadas e armazenadas permanentemente numa corrente de blocos (*block chain*), replicada em cada servidor. A prova de conceito oferecida pelo Bitcoin

foi suficiente para uma crescente adesão à rede de confiança digital descentralizada que se criou. Seu grande feito? Eliminar a terceira parte de confiança, necessária nas transações financeiras. Para além da tecnologia, a blockchain do Bitcoin traz uma ruptura nas transações de negócios, ao introduzir o mecanismo de incentivos digitais e criptomoedas em diversos níveis (sistemas, serviços, aplicações, etc.) e relações (econômicas, públicas, sociais, etc.). De fato, o consenso do Nakamoto (com sua mineração inovadora) e a geração da moeda (como mecanismo de incentivo) integram-se numa simbiose salutar para viabilizar o acordo entre pares que não se conhecem, e adicionalmente, a oferta de serviços de consenso sob demanda na Internet.

A blockchain original do Bitcoin incorpora uma máquina de estados bem simplificada, com elementos voltados essencialmente para transações com a moeda bitcoin. O projeto Ethereum, idealizado por Vitalik Buterin [Buterin 2014], adota o conceito de *contratos inteligentes*, que permite a execução de uma máquina de Turing completa. Os contratos inteligentes expressam uma lógica de transações mais sofisticada, possibilitando a implementação de aplicações descentralizadas e autônomas, em diversos níveis e escopos. Na evolução de projeto da blockchain, são então destacadas três fases [Bashir 2017]: a Blockchain 1.0 envolve o lançamento do Bitcoin em 2008, com as primeiras implementações das criptomoedas, e um ecossistema de aplicações e pagamentos com o ativo digital. A Blockchain 2.0 inicia-se com a proposta inovadora dos contratos inteligentes em 2013, e toda gama de aplicações financeiras possíveis. A Blockchain 3.0 caracteriza a adoção da tecnologia blockchain no benefício de aplicações em diversas áreas, para além da financeira: governo, comércio, artes, saúde, cidades digitais, etc..

Há diversos projetos, com forte investimento da indústria, visando a implementação de plataformas de blockchain para desenvolvimento de aplicações robustas e descentralizadas nos mais variados segmentos. Para além do Bitcoin e Ethereum, que oferecem uma *blockchain pública*, aberta, sem necessidade de controle dos participantes, há uma gama de setores que requerem participação controlada. As plataformas para *blockchain privada* ou *federada* atendem melhor a interesses corporativos e requerem a autenticação dos participantes na rede; dentre elas, destaca-se o Hyperledger Fabric, em estágio avançado de desenvolvimento. De fato, tal característica da blockchain, no que diz respeito ao tipo de composição e filiação dos membros à rede, vai nortear importantes aspectos da sua infra-estrutura (replicação máquina de estados, consenso, segurança, comunicação), bem como do projeto dos sistemas nas demais camadas (dados, contratos inteligentes, serviços, aplicações) e modelos de negócios associados.

5.1.1. Propriedades da Blockchain

As principais propriedades da tecnologia blockchain que contribuem de forma inovadora para o desenvolvimento de aplicações e sistemas são as seguintes:

- *Descentralização*: As aplicações e sistemas são executados de maneira distribuída, através do estabelecimento de confiança entre as partes, sem a necessidade de uma entidade intermediária confiável. Esse é o principal motivador para o crescente interesse na blockchain.
- *Disponibilidade e Integridade*: Todo o conjunto de dados e transações são replica-

dos em diferentes nós de maneira segura, de forma a manter o sistema disponível e consistente.

- *Transparência e Auditabilidade*: Todas as transações registradas no livro-razão são públicas, podendo ser verificadas e auditadas. Além disso, os códigos da tecnologia costumam ser abertos, passíveis de verificação.
- *Imutabilidade e Irrefutabilidade*: As transações registradas no livro-razão são imutáveis. Uma vez registradas não podem ser refutadas. Atualizações são possíveis a partir da geração de novas transações e realização de novo consenso.
- *Privacidade e Anonimidade*: É possível oferecer privacidade aos usuários sem que os terceiros envolvidos tenham acesso e controle dos seus dados. Na tecnologia, cada usuário gerencia suas próprias chaves e cada nó servidor armazena apenas fragmentos criptografados de dados do usuário. Transações são até certo ponto anônimas, com base no endereço dos envolvidos na blockchain.
- *Desintermediação*: A blockchain possibilita a integração entre diversos sistemas de forma direta e eficiente. Assim, é considerada um conector de sistemas complexos (sistemas de sistemas), permitindo a eliminação de intermediários de maneira a simplificar o projeto dos sistemas e processos [Xu et al. 2016].
- *Cooperação e Incentivos*: Oferta de modelo de negócios à base de incentivos, à luz da teoria dos jogos. O consenso sob demanda passa a ser oferecido como serviço em diversos níveis e escopos.

5.1.2. Organização do Capítulo

Neste capítulo, apresentamos a blockchain, seus principais conceitos, desafios e aplicações em alguns contextos. As pesquisas na área são recentes e a tecnologia emergente. Nosso foco, então, será de apresentação dos elementos fundamentais, com descrição dos modelos, algoritmos e mecanismos. Especial ênfase será dada aos protocolos de consenso distribuído para manutenção da máquina de estados replicada. Além disso, a blockchain pública do Bitcoin será exaurida. Como contraponto, apresenta-se o Hyperledger Fabric, um projeto promissor para blockchains privadas ou federadas. O consenso sob demanda oferecido pela blockchain abre perspectivas para desenvolvimento de sistemas robustos e seguros em diversos escopos da computação, particularmente, em redes de computadores, IoT e computação em névoa. Alguns resultados recentes nesses contextos serão apresentados. O capítulo dividi-se, então, em três partes: a primeira aborda os elementos fundamentais para o entendimento da blockchain: criptografia, consenso e livro-razão (Seção 5.2), bem como o consenso distribuído para blockchains públicas e privadas (Seção 5.3). A segunda parte apresenta as plataformas Bitcoin (5.4) e Hyperledger Fabric (5.5). A terceira parte explora exemplos de adoção da blockchain para as áreas de redes de computadores, IoT e névoa (Seções 5.6, 5.7, 5.8), e finalmente discute-se desafios e perspectivas na área (Seção 5.9).

5.2. Elementos Fundamentais (Criptografia, Consenso, Livro-Razão)

Nesta seção, serão apresentados os principais elementos da blockchain, que contribuem para suas propriedades: imutabilidade, integridade, atualidade, transparência, disponibilidade, descentralização, desintermediação.

5.2.1. Criptografia

A Blockchain apoia-se fortemente na Criptografia para satisfazer os requisitos de segurança do sistema e das aplicações. Dentre os recursos mais utilizados, destacam-se os *resumos criptográficos e as assinaturas digitais*. Nesta seção, elas serão definidas e, ao longo do capítulo, suas aplicações serão destacadas. Para maiores detalhes, o leitor deve reporta-se ao excelente livro [Narayanan et al. 2016].

5.2.1.1. Resumos Criptográficos

As *funções hash* incorporam algoritmos que fazem a dispersão de dados variáveis x em valores h com quantidade fixa de dígitos, tal que $h = \text{hash}(x)$. São excelentes artifícios de representação e transformação de dados de volume variável (x = arquivos, mensagens, etc.) num dado de valor único e com pequena quantidade de dígitos (e.g., 256 dígitos). Além disso, tais algoritmos devem ser eficientes computacionalmente, de tal forma que, se x é uma string de n bits, então $\text{hash}(x)$ tem complexidade $O(n)$.

As *funções hash criptográficas* (ou *resumos criptográficos*) são unidirecionais e dificilmente permitem a recuperação do valor original x , a partir do hash h . Assim, são funções hash que, adicionalmente, satisfazem as seguintes propriedades: (1) resistência à colisão (*collision resistance*) e (2) ocultação (*hiding*). Nos próximos parágrafos, vamos explicar cada uma dessas propriedades e suas abrangências.

Uma colisão irá ocorrer quando, ao aplicar a função hash para duas entradas distintas, x e y , o mesmo valor h for encontrado. Ou seja, dada a função hash H , então $H(x) = H(y) = h$. Como o universo de valores de saída h é muito menor do que o universo de valores de entrada, colisões evidentemente serão possíveis. Entretanto, o que se deseja é que elas sejam raras, que sejam evitadas quando as funções forem aplicadas.

Definição 1 Resistência à Colisão. *Um função hash H é resistente à colisão, quando for inviável achar dois valores x e y , tal que $x \neq y$ e $H(x) = H(y)$.*

Podemos deduzir que se a função *hash* H é resistente à colisão, então, dificilmente iremos encontrar dois valores x e y diferentes e com mesmo hash h . Logo, podemos utilizar tais funções para, por exemplo, representar adequadamente resumos de informações (ou *message digests*). Tais resumos podem ser usados em aplicações para atestar, por exemplo, a integridade dos arquivos que tenham sido armazenados ou transmitidos na rede. Assim, dado um arquivo A , e seu $h = H(A)$; podemos usar h como um resumo ímpar do arquivo A original. Como h tem tamanho fixo pequeno (e.g., 256 bits) e A costuma ter tamanho muito grande, em muitos contextos, costuma-se substituir A por h .

Uma característica importante que deve ser associada às funções *hash* é a necessidade de ocultação do valor original x . Ou seja, não seria viável deduzir o valor de x , dado

que o valor $h = H(x)$ é conhecido. Entretanto, é importante perceber que, se o universo de possibilidades de x for muito pequeno, ficaria muito fácil deduzir x . Por exemplo, se x tem apenas duas possibilidades (quente ou frio), bastaria um adversário aplicar *hash* nos dois valores para saber a qual h estaria associado. Dessa forma, espera-se que x seja escolhido de um conjunto de valores bastante aleatório.

Um truque que se costuma utilizar para que a ocultação seja possível, qualquer que seja a entrada x , é o de considerar um valor r , escolhido de um conjunto de valores bastante aleatório e, assim, aplicar a função hash $H(r||x)$ à concatenação ($||$) das duas entradas r e x . Ou seja, se r vem de uma distribuição, onde não há um valor específico provável de ocorrer, fica quase impossível chutar o valor de x , na aplicação da função H . Dessa maneira, a ocultação será possível.

Definição 2 Ocultação (Hiding). *Um função hash H está ocultando se: quando um valor secreto r é escolhido de um universo com distribuição aleatória muito espalhada, então, dado $H(r||x)$, é inviável identificar x .*

A ocultação é fundamental para estabelecer compromissos (*commitments*). Esses são uma forma de se comprometer digitalmente com uma mensagem *msg*, dado que a *msg* será oculta e, quando necessário, poderá ser revelada. A *msg* poderia ser associada a um contrato, que será colocado num envelope e lacrado. Uma vez firmado o compromisso (ou seja, o contrato), ele não poderá ser alterado. A posteriori, o conteúdo do envelope pode ser revelado e verificado.

Esquema 1 Compromisso (*Commitment*). *Consiste em dois algoritmos:*

- $com := commit(msg, nonce)$ – Função que recebe a *msg* e um valor randômico secreto *nonce* e retorna um valor *com* atestando o compromisso firmado. Em criptografia, o termo *nonce* designa um valor a ser utilizado apenas uma vez.
- $match := verify(com, msg, nonce)$ – Função que verifica se o compromisso *com* é válido. Ou seja, retorna um valor booleano *true* caso $com == commit(msg, nonce)$ e *false*, caso contrário.

Seguindo a analogia do contrato no envelope, a função *com* seria o lacre com o envelope, que todos têm acesso. A função *verify* corresponde a abertura do envelope para verificação. Para cada novo contrato, um *nonce* é utilizado. Como a *msg* deve ficar oculta através de *com*, não podemos recuperá-la, logo o compromisso deve *ocultar* a *msg*. Ao mesmo tempo, uma vez firmado o compromisso, não podemos alterá-lo, assim a *msg* não pode sofrer mudanças; logo deve-se *ligar* a *msg* ao compromisso, de tal forma que é impossível encontrar dois pares $(msg, nonce)$ e $(msg', nonce')$, tal que $msg \neq msg'$, de forma a satisfazer $commit(msg, nonce) == commit(msg', nonce')$. Desta maneira, os compromissos precisam assegurar essas duas propriedades de segurança: *ocultação* e *ligação*. Para implementar o esquema de compromisso, pode-se usar diretamente um resumo criptográfico H , onde *nonce* é valor aleatório de 256 bits. Assim:

$$commit(msg, nonce) := H(nonce||msg).$$

Como H satisfaz ocultação e é resistente à colisão, estaremos satisfazendo as propriedades do compromisso, conforme abaixo:

- *Ocultação*: Dado $H(\text{nonce}||\text{msg})$, é inviável encontrar msg .
- *Ligação*: É inviável encontrar dois pares $(\text{msg}, \text{nonce})$ e $(\text{msg}', \text{nonce}')$, tal que $\text{msg} \neq \text{msg}'$ e $H(\text{nonce}||\text{msg}) == H(\text{nonce}'||\text{msg}')$.

Como veremos, compromissos, resumos criptográficos e nonces serão fundamentais para assegurar as propriedades de segurança da blockchain.

5.2.1.2. Assinaturas Digitais

As assinaturas digitais devem atender às mesmas propriedades de assinaturas manuais em documentos. Assim, somente você pode assinar um documento, mas, qualquer um pode verificar a sua autenticidade. Além disso, não é possível forjar a sua assinatura, de tal forma a reutilizá-la em algum outro contexto. Ou seja, assinaturas devem ser irrefutáveis.

As assinaturas digitais são implementadas através de *criptografia de chave assimétrica*. Nesse caso, além da chave privada ou secreta (sk) usada para assinar os documentos, também será utilizada uma chave pública (pk). A chave privada é secreta e não deve ser revelada para ninguém. Já a chave pública deve ser revelada para qualquer um que queira atestar a autenticidade da assinatura.

Esquema 2 Assinatura Digital *Consiste em três algoritmos:*

- $(sk, pk) := \text{generateKeys}(\text{keysize})$ – O método *generateKeys()* recebe um tamanho de chave (*keysize*) na entrada e retorna um par de chaves pública (pk) e privada ou secreta (sk).
- $\text{sig} := \text{sign}(sk, \text{msg})$ – O método *sign()* recebe uma mensagem msg e uma chave secreta (sk) na entrada e retorna a assinatura sig daquela mensagem sob sk .
- $\text{isValid} := \text{verify}(pk, \text{msg}, \text{sig})$ – O método *verify* recebe uma chave pública (pk), uma mensagem msg e uma assinatura (sig) na entrada, e retorna um valor booleano: $\text{isValid} = \text{true}$ se sig é uma assinatura válida para msg sob pk ; $\text{isValid} = \text{false}$, caso contrário.

A assinatura digital deve satisfazer às seguintes propriedades:

- *Autenticidade*: É possível verificar assinatura válida, da seguinte maneira:

$$\text{verify}(pk, \text{msg}, \text{sign}(sk, \text{msg})) == \text{true}.$$

- *Existencialmente Infalsificável (Unforgeable)*: Assinatura não pode ser forjada.

A autenticidade garante a autoria da assinatura, que poderá ser verificada à posteriori. Como ela é infalsificável, qualquer alteração no documento torna a assinatura inválida, logo ela preserva a *integridade* do documento; também garante o *não repúdio*, e o assinante não pode negá-la, depois de produzida. Do ponto de vista prático, costuma-se assinar o resumo criptográfico H de uma mensagem msg , no lugar da mensagem propriamente dita. Isto porque mensagens costumam ser muito longas e resumos têm quantidade fixa de bits. Assim, adota-se $\text{sig} := \text{sign}(sk, H(\text{msg}))$. Como os hashes são resistentes à colisão, podemos usar $H(\text{msg})$ como um resumo singular para msg .

Os *endereços* na blockchain, importantes para identificar os nós nas transações, são resumos criptográficos das chaves públicas dos envolvidos. O Bitcoin e várias outras blockchains utilizam como resumo criptográfico a SHA256 (cuja saída tem 256 bits), aplicada duplamente *double-SHA256*. O Bitcoin utiliza um tipo especial de esquema de assinatura digital chamado *Elliptic Curve Digital Signature Algorithm* (ECDSA), tipo “secp256k1”, que provê 128 bits de segurança [Antonopoulos 2017, Bashir 2017].

5.2.1.3. Infraestrutura de Chave Pública

Uma *infraestrutura de chave pública* - ICP (*PKI - Public Key Infrastructure*) é uma cadeia hierárquica de confiança que viabiliza a emissão de certificados digitais para identificação virtual. Seu funcionamento baseia-se na derivação de confiança a partir do uso de assinaturas digitais. Portanto, caso confie-se em uma autoridade certificadora, todos os certificados emitidos pela instituição, que estejam válidos (dentro do período de validade, não foram revogados, etc.), serão também confiáveis. Uma ICP administra a geração, distribuição e revogação de chaves e certificados digitais.

O *certificado digital* é um documento eletrônico assinado digitalmente por uma autoridade certificadora, que contém diversos dados sobre o emissor e o seu titular. Sua função é vincular uma pessoa ou uma entidade a uma chave pública. No contexto da blockchain privada os certificados digitais estabelecem credenciais de usuário e assinaturas para as mensagens garantindo, assim, que a mensagem não tenha sido alterada.

Normalmente, uma ICP possui uma autoridade certificadora (*CA - Certificate Authority*), uma autoridade de Registro (*RA - Registration Authority*) e um banco de dados de certificados. A RA é responsável pela autenticação e confirmação da legitimidade dos dados, certificados ou outros artefatos utilizados para assegurar a solicitação de um usuário para um ou mais certificados que refletem a identidade ou propriedade do mesmo [Nguyen et al. 2018]. Uma CA, sob recomendação de uma RA, emite certificados digitais para usos específicos e é certificada diretamente ou hierarquicamente para uma CA raiz.

5.2.2. Consenso

Consenso é um problema fundamental em computação distribuída confiável, pois permite com que participantes distribuídos coordenem as suas ações, de forma a alcançar decisões comuns, e assim garantir a manutenção da consistência dos seus estados (*safety*) e o progresso do sistema (*liveness*), apesar da existência de falhas [Greve 2005]. O consenso fatoriza a classe de problemas em que existe necessidade de acordo. Nesse sentido, o consenso é seminal para a tecnologia blockchain, pois possibilita a obtenção do acordo sobre o próximo bloco a ser agregado à blockchain.

Falhas Byzantinas. Na blockchain, o consenso deve ser alcançado apesar da existência de processos maliciosos, que podem agir arbitrariamente de forma a subverter a computação. Ou seja, considera-se o modelo de falhas Byzantinas [Lamport et al. 1982], no qual, diferentemente de um processo *correto*, um processo *falho* pode exibir qualquer comportamento, podendo parar, omitir envios e entregas de mensagens, ou desviar arbitrariamente de sua especificação.

Definição 3 Consenso Bizantino. *No consenso, cada processo p_i propõe um valor v_i e todos os processos corretos devem “decidir” por um único valor v , dentre os propostos. Formalmente, ele é definido pelo seguinte conjunto de propriedades:*

- *Acordo:* dois processos corretos não decidem diferentes valores;
- *Terminação:* todo processo correto decide de maneira definitiva;
- *Validade:* se um processo correto decide o valor v , então v foi proposto por algum processo. Para a blockchain, especificamente, v será o *bloco de transações* a ser adicionado ao livro razão.

5.2.2.1. Replicação Máquina de Estados (RME)

Replicação Máquina de Estados (RME) ou *State Machine Replication (SMR)* é uma técnica de tolerância a falhas que permite manter um serviço replicado de forma consistente. Na RME a execução das diversas requisições, feitas por clientes aos servidores que mantêm as réplicas, é realizada deterministicamente, na mesma ordem total, de tal maneira que réplicas corretas irão passar pelo mesmos estados de transição, gerando por consequente os mesmos resultados a cada nova requisição. Importante observar que cada réplica mantém uma cópia completa do serviço replicado.

A RME deve assim atender aos seguintes requisitos [Schneider 1990]:

- *Estado inicial:* todas as réplicas corretas devem iniciar a partir de um mesmo estado;
- *Determinismo:* todas as réplicas corretas ao executarem a mesma operação sobre o mesmo estado produzem o mesmo resultado e mudança de estado subsequente;
- *Coordenação:* todas as réplicas corretas executam a mesma sequência de operações.

A coordenação da RME é assegurada por um protocolo de consenso, conhecido como *atomic broadcast* ou *difusão atômica* ou *difusão em ordem total* [Lamport 1998, Défago et al. 2004]. Ele assegura que as réplicas corretas realizarão o mesmo conjunto de requisições e na mesma ordem total. No modelo de falhas Bizantinas, em que até f réplicas podem se comportar maliciosamente, um cliente identifica que sua requisição foi executada com sucesso quando recebe $f + 1$ respostas idênticas dos servidores.

A blockchain implementa assim uma máquina de estados replicada para a manutenção consistente do livro razão distribuído em cada um dos servidores que compõem a rede P2P. Nesse caso, os blocos de transações são incorporados ao livro razão segundo uma ordem total, através de um protocolo de difusão atômica. Tal consenso é necessário para evitar que blocos concorrentes sejam agregados em ordens diferentes ao livro razão por cada um dos servidores, comprometendo assim a execução consistente das transações.

Resultado de Impossibilidade do Consenso. É importante destacar que numa rede clássica, cujos n participantes do sistema são identificados, e onde não há limites temporais para a transmissão de mensagens e realização das ações dos processos, ou seja, o sistema é *assíncrono*, o consenso não tem solução determinística, mesmo na presença de uma única falha benigna de processo: falha por parada (*crash*) [Fischer et al. 1985]. Este resultado é conhecido como FLP, devido ao nome dos autores: Fisher, Lynch e Paterson. Assim, resolvê-lo no contexto da blockchain, onde o conjunto de participantes

não é bem conhecido e estão sujeitos a falhas byzantinas, tem complexidade ainda maior. Este resultado fundamental provoca a impossibilidade de se resolver outros problemas, onde a necessidade de acordo é recorrente, como a própria *difusão atômica* [Chandra and Toueg 1996]. Na origem desses resultados negativos, encontramos a mesma causa: as incertezas decorrentes do modelo assíncrono no que diz respeito à detecção de falhas.

Incorporando Sincronia ao Sistema. É interessante notar que, desde que algum grau de sincronia é acrescentado ao sistema, vários desses problemas passam a ter solução determinística. Em particular, num sistema assíncrono estendido com detectores de falhas não-confiáveis, [Chandra and Toueg 1996] provaram que existe uma solução determinística ao problema do consenso com falhas por parada, desde que uma maioria de processos ($f < n/2$) seja correta. Além disso, a difusão atômica e o consenso são equivalentes em qualquer modelo de falhas, ou mais precisamente em todo modelo em que a difusão confiável pode ser resolvida [Chandra et al. 1996]. Isto significa que todo algoritmo para o consenso pode ser aplicado para resolver a difusão atômica e vice-versa.

5.2.2.2. Variações de Consenso

Paxos, introduzido por [Lamport 1998], é o algoritmo de consenso mais conhecido para implementação da RME. Ele propõe a extensão do sistema assíncrono com o oráculo de líder, uma abstração equivalente aos detectores de falhas [Chu 1998]. Um oráculo ou detector de líder da classe Ω (Ômega) satisfaz a propriedade de liderança definitiva, em que existe um instante após o qual, todo processo correto sempre confia no mesmo processo correto no sistema (i.e., o mesmo líder) [Lamport 2001]. Na resolução do consenso, o líder será fundamental para coordenar os demais na obtenção do acordo. A classe de detectores de falhas $\diamond\mathcal{S}$ (*eventually strong*) e o detector Ω possuem o mesmo poder computacional e reúnem as condições de sincronia mínimas para resolução do consenso em sistemas assíncronos, sujeitos a falhas por parada [Chandra et al. 1996].

O consenso byzantino, introduzido por [Lamport et al. 1982] sob a designação do **Problema dos Generais Byzantinos**, tem solução num sistema em que, no máximo, $f < n/3$ processos se comportam de maneira maliciosa. Lamport demonstra ser impossível atingir consenso byzantino se essa condição não é satisfeita. O modelo de sistema mais comumente usado na solução do consenso byzantino é o *parcialmente síncrono* de [Dwork et al. 1988]. Neste, existem limites temporais para a transmissão de mensagens na rede e para a realização de qualquer computação, no entanto, esses limites não são conhecidos e não se sabe quando passam a valer. Ele modela bem o comportamento da Internet, em que o sistema passa por períodos de estabilidade (quanto a computação converge dentro de limites temporais esperados), embora tenha comportamento assíncrono, com frequência.

Byzantine Fault Tolerance (BFT). Ao longo dos anos, um número bastante expressivo de protocolos de consenso foi proposto para tolerância a falhas por parada [Guerraoui and Raynal 2007] e falhas byzantinas [Guerraoui et al. 2008], além do desenvolvimento de RME [Charron-Bost et al. 2010]. Muitas soluções focam nos algoritmos e aspectos teóricos, sendo que apenas um pequeno grupo aborda as questões mais práticas. Mais recentemente, com a necessidade da incorporação de técnicas de tolerância a falhas no núcleo dos grandes serviços da Internet (nuvem, *storage*, etc.), oferecidos

pelos grandes players da indústria, um interesse crescente voltou-se para a sua engenharia. Diversos sistemas, como o Chubby da Google [Chandra et al. 2007] e o Apache Zookeeper [Hunt et al. 2010], apoiam-se fortemente em soluções à base do Paxos [Rao et al. 2011, Gafni and Lamport 2003] para prover uma plataforma de sincronização de estados em provedores de nuvens e *data centers* com redundância, alta disponibilidade, robustez e resiliência a falhas por parada.

PBFT (*Practical Byzantine Fault-Tolerant*) foi o primeiro consenso byzantino a abordar questões práticas e um dos mais conhecidos [Castro and Liskov 1999, Castro and Liskov 2002]. O BFT-Smart [Bessani et al. 2014] é um outro projeto promissor em bom estágio de maturidade. Como veremos na Seção 5.5, PBFT é oferecido pelo Hyperledger Fabric como camada de acordo (ordenação de transações). Além disso, o BFT-Smart [Sousa et al. 2018] também foi recentemente incorporado ao projeto.

Consenso Probabilístico. Uma outra alternativa, visando contornar o resultado de impossibilidade FLP, consiste em mudar a especificação do problema e enfraquecer as propriedades do consenso. Nessa linha, surgiram protocolos com garantias de acordo probabilísticas [Ben-Or 1983, Cachin et al. 2000]. Nesse caso, faz-se uso de um oráculo randômico: um *gerador de números aleatórios*. O oráculo randômico de [Ben-Or 1983] soluciona uma versão não-determinística do consenso, assegurando a obtenção de decisões com probabilidade 1, mesmo num cenário em que o adversário conhece tudo acerca do sistema. O consenso [Nakamoto 2008] da blockchain do Bitcoin provê acordo probabilístico. Durante um período de instabilidade, o sistema pode vir a decidir mais de um valor, ocasionando os chamados *forks*; porém, após um período de instabilidade na rede, a decisão tende a ser única.

Como a blockchain é um sistema de confiança per si, ela se beneficia diretamente de todo o avanço que a área de BFT teve ao longo das últimas décadas. Por outro lado, ela está promovendo um revival na área, e os protocolos e sistemas BFT passam a ser adaptados para esse novo e desafiante contexto da blockchain [Cachin and Vukolić 2017, Gramoli 2017, Bano et al. 2017].

5.2.2.3. Teorema CAP (*Consistency, Partition, Availability*)

Safety x Liveness. Importante notar que, para contornar o resultado FLP, há duas abordagens: ou enfraquece-se o modelo de sincronia ou enfraquece-se o problema. No primeiro caso, obtemos consenso com garantias de acordo determinísticas, já no segundo caso, teremos consenso com garantias probabilísticas. O consenso byzantino tradicional e a BFT oferecem garantias determinísticas e apoiam-se em redes parcialmente síncronas para assegurar o progresso; neste caso, enquanto a rede não oferece as condições de estabilidade (sincronia) para que se decida, a terminação (*liveness*) ficará comprometida, mas não o acordo (*safety*). Assim, o modelo BFT sempre assegura a consistência do sistema, mas, pode comprometer o seu progresso. Já o consenso randômico, incluindo o do Nakamoto, assegura a decisão (ordenação de blocos), garantindo a terminação; entretanto, pode comprometer o acordo (sobre a corrente de blocos) e assim a consistência do sistema.

Teorema CAP (*Consistency, Partition, Availability*). O antagonismo entre consistência e progresso nos sistemas tolerantes a falhas foi generalizado por [Brewer 2000] através do Teorema CAP (provado por [Gilbert and Lynch 2012]). De maneira geral, um

sistema distribuído apresenta um compromisso entre as seguintes propriedades:

- Consistência (*Consistency*): todas as requisições obtêm os dados mais recentes; ou seja, o estado do sistema é consistente;
- Disponibilidade (*Availability*): toda requisição obtém uma resposta; ou seja, o sistema está operante, executando as ações;
- Tolerância a Partições (*Partition tolerance*): o sistema continua em operação, apesar de particionamentos temporários da rede, devido a falhas.

O Teorema CAP estabelece que não é possível satisfazer plenamente as três propriedades ao mesmo tempo. Assim, alguns sistemas sacrificam uma delas em detrimento das outras duas. Sistemas baseados em BFT clássico, bem como as BFT Blockchain, privilegiam fortemente a consistência, em detrimento da disponibilidade. O mesmo acontece com bancos de dados não-estruturados (NoSQL), no relaxamento das propriedades ACID [Frank et al. 2014] e oferta de baixa disponibilidade. Já a Blockchain do Bitcoin sacrifica a consistência em prol da disponibilidade e tolerância a partições. No caso, ela oferece consistência após um tempo (*eventual consistency*), e permite a obtenção, de forma gradativa, ao longo do tempo, de um estado consistente.

5.2.3. Livro-Razão Distribuído (*Distributed Ledger*)

O livro-razão (*ledger*) é a estrutura de dados imutável, em que transações são registradas e o estado global do sistema é mantido. O *ledger* mantém-se completamente replicado em todos os nós da rede P2P. Logo, o livro-razão distribuído é replicado e imutável. Por sua vez, as transações devem obedecer as mesmas propriedades ACID de banco de dados. Nesse sentido, a criptografia e o consenso são os elementos chaves para manutenção da autenticidade, integridade, consistência e disponibilidade do livro-razão. Nessa seção, o livro-razão será detalhado [Narayanan et al. 2016, Antonopoulos 2017].

Apontador hash (*hash pointer*) é um apontador para aonde encontra-se armazenado o dado d e o seu hash criptográfico $H(d)$. Assim, além de permitir a recuperação do dado d , o apontador hash permite a sua verificação, através de $H(d)$. Eles podem ser usados para construir quaisquer estrutura de dados com encadeamento, como listas, árvores, grafos direcionados acíclicos (DAGs), etc.

Em particular, apontadores hash são empregados para criar uma estrutura de blocos encadeada, de nome **blockchain**, conforme Figura 5.1. Nesta, temos um conjunto de blocos de dados encadeados numa lista de hashes. Como a função H é resistente à colisão (Definição 1) e satisfaz ocultação (Definição 2), podemos usar H para verificar se o bloco b_k dentro da blockchain sofreu modificação. Se houver alguma mudança em b_k o apontador $H(b_k)$, presente no bloco b_{k+1} , estará inconsistente com o valor do dado armazenado em b_k . Assim, se o *header* $H()$ de toda estrutura é guardado de forma segura, pode-se verificar quando a blockchain sofreu violação, mesmo quando adversários tentam modificar todo o conjunto de apontadores armazenados na lista encadeada.

A tecnologia blockchain é assim formada por blocos encadeados por apontadores hashes numa lista. O primeiro bloco da lista é designado bloco *genesis*. Cada bloco contém um conjunto de transações. Estas, por sua vez, são estruturados numa estrutura de

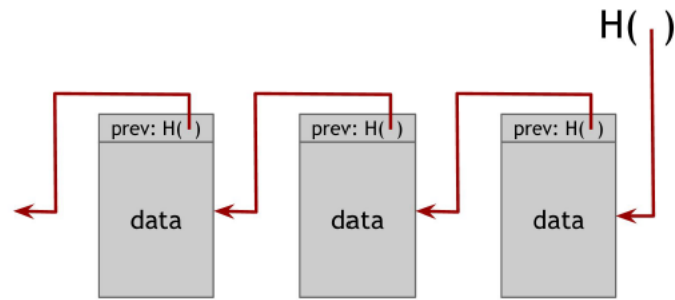


Figure 5.1. Estrutura de Blockchain com Apontadores Hash (fonte: [Narayanan et al. 2016])

árvore binária de apontadores hash. No caso da rede Bitcoin essa estrutura é a **árvore de Merkle**. A Figura 5.2 ilustra essa formação. Nesta árvore, no último nível, estão as folhas que contêm os dados (especificamente, apontadores para as transações propriamente ditas); no penúltimo nível, os pais possuem apontadores hash para esses dados. Em seguida, em cada nível, os pais apontadores são agrupados dois a dois, até que se alcance a raiz da árvore. O hash da raiz da árvore é então armazenado de forma segura dentro do *header* do bloco, em conjunto com demais informações. Pela propriedade dos apontadores hash, se houver qualquer modificação nos dados da árvore, ela será identificada pela verificação dos hashes. Além disso, o espaço necessário para o seu armazenamento é muito pequeno, comparativamente ao tamanho dos dados.

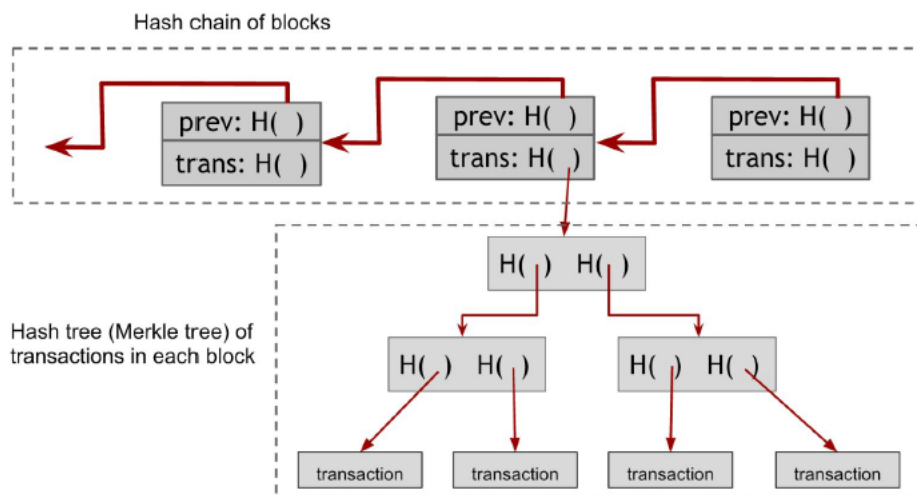


Figure 5.2. Blockchain com Árvore Binária de Merkle (fonte: [Narayanan et al. 2016])

Uma outra vantagem de uso da árvore de Merkle é a facilidade de prova de filiação (*proof of membership*), sendo possível verificar se uma transação t_i pertence a árvore T em tempo $O(\log(n))$, onde n é o conjunto de nós da árvore. Assim, para provar que $t_i \in T$, basta apresentar os hashes do caminho em que t_i é folha seguindo até a raiz da árvore. Esse caminho de autenticação ou *caminho de merkle* terá o hash do complemento dos pares, a cada nível. Na Figura 5.3, o caminho de merkle para se verificar que a transação

H_K pertence à árvore consiste nos quatro hashes seguintes: $H_L, H_{IJ}, H_{MNOP}, H_{ABCDEFGH}$. Dessa forma, qualquer nó da rede, de posse do caminho de merkle, poderá calcular os hashes complementares em $O(\log(n))$ operações.

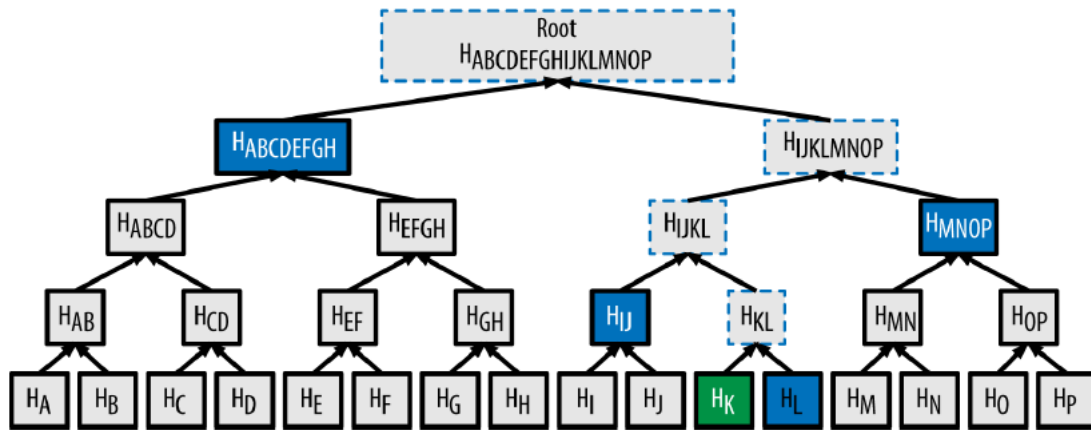


Figure 5.3. Caminho de Merkle (fonte: [Antonopoulos 2017])

5.2.3.1. Transações

Uma transação estabelece uma sequência de operações sobre estados. Ela incorpora uma transferência de ativo ou, de forma geral, um contrato inteligente. Num caso básico, a transação envolve uma assinatura digital do emissor (que detém o ativo) e o endereço do receptor, além de entradas e saídas com o valor transacionado. Se a transação refere-se a contratos inteligentes pode envolver a invocação de um método, com suas entradas; ou, se a intenção é o *deploy* do contrato, deve conter o código de inicialização.

A Figura 5.4 apresenta um exemplo simplificado de quatro transações envolvendo bitcoins (para facilitar, cada bloco apresenta uma única transação). Em cada transação, deve-se representar as entradas (INPUTS) e saídas (OUTPUTS), tal como num livro contábil. As entradas referenciam os hashes das transações anteriores que têm relação com a atual. No exemplo, a transação t_1 de geração da moeda, efetua a transferência de 25 bitcoins a Alice. A transação t_2 representa uma transferência de 17 bitcoins de Alice para Bob; para tanto, Alice deve assinar t_2 , indicando a sua autorização. Na entrada de t_2 , o valor 1[0] aponta para o hash de t_1 , indicando que o ativo deve ser transferido de t_1 , conforme saída 0. Na transação t_4 , Alice transfere 6 bitcoins para David, para tanto indica como entrada o hash da transação t_2 , conforme saída 1.

Validação de uma transação t : envolve sobretudo: (i) a verificação das assinaturas, (ii) a confirmação dos valores existentes a partir dos hashes das transações anteriores referenciadas e, ainda, (iii) a confirmação de que o valor não foi anteriormente gasto por nenhuma outra transação, ou seja, existem fundos para a transação; nesse caso, será necessário efetuar uma busca na blockchain entre o bloco da transação referenciada até o último bloco da estrutura. As transações são validadas, de forma independente, por cada nó da rede blockchain, e essa característica contribui para a descentralização do processo.

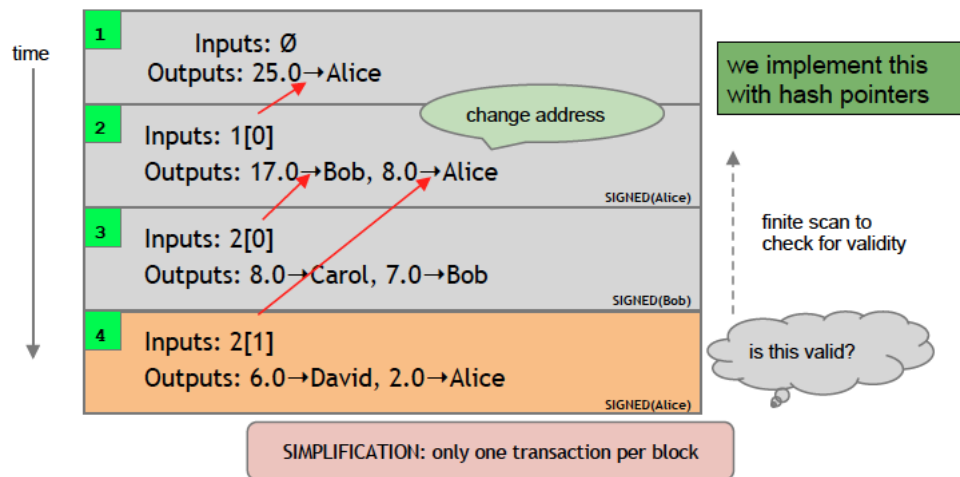


Figure 5.4. Esquema de Conjunto de Transações em Bitcoin (fonte: [Narayanan et al. 2016], slides: Montresor (UniTn))

5.2.3.2. Blocos

O bloco contém um *header* com informações necessárias para a manutenção da corrente e sua validação. A Figura 5.5 apresenta um esboço simplificado de bloco no Bitcoin. Os principais campos são: (i) *prev*: o apontador hash do bloco anterior da corrente, (ii) *mrkl_root*: o apontador hash da árvore de Merkle do conjunto de transações do bloco, (iii) *nonce*, um valor variável a ser obtido na resolução do enigma criptográfico durante o processo de consenso (via mineração), de tal maneira que o *hash* do bloco seja válido, i.e., seja menor que o *alvo* estipulado pela prova de trabalho. Para além desses campos (aqui representados), o header do bloco também possui um *timestamp*, que indica o tempo cronológico em que o bloco é acrescentado à blockchain, e a *dificuldade alvo*, estabelecido pela prova de trabalho.

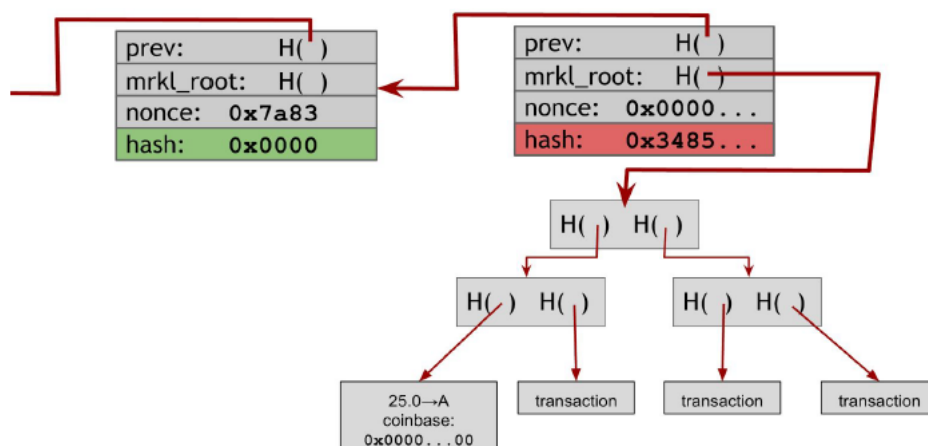


Figure 5.5. Bloco com Header Simplificado na Bitcoin (fonte: [Narayanan et al. 2016])

Além do header, o bloco também possui *metadados*, que ficam armazenados separadamente da blockchain e não são transmitidos na rede. São eles: o *hash do header do bloco*, contendo o resumo que identifica univocamente o bloco perante toda a rede; a *altura* ou posição do bloco na corrente. O primeiro bloco (genesis) tem altura 0, os demais terão a altura do bloco anterior acrescida de 1. Ou seja, sua altura estabelecerá a quantidade de blocos criados até então após o bloco genesis [Antonopoulos 2017].

Validação de um bloco b: consiste em verificar (i) se a sua estrutura é bem formada, (ii) o seu *hash* é válido (atende ao desafio), (iii) o seu tamanho está dentro do limite aceito pela rede, (iv) o conjunto de transações dentro do bloco é válido, (v) a primeira transação (e somente a primeira) é a *coinbase transaction* – que incorpora a geração de novas criptomoedas no sistema, além de atuar como mecanismo de recompensa. Os blocos são validados, de forma independente, por cada nó da rede blockchain, e essa característica contribui para a descentralização do processo.

5.3. Blockchain Pública (sem permissão) X Federada/Privada (com permissão)

As redes blockchain podem ser categorizadas em dois grupos: blockchain pública ou *permissionless* (sem permissão, de acesso aberto), e blockchain federada/privada ou *permissioned* (com permissão e acesso controlado).

Na *blockchain pública*, o conjunto de nós da rede P2P é desconhecido e sua composição (*membership*) é dinâmica, permitindo entradas e saídas aleatórias de nós. Como os nós não precisam de identificação, costumam ser anônimos. A blockchain pode atuar em escala planetária, sem controle dos seus participantes, que inclusive não se confiam mutuamente. Nesta categoria, encontra-se a rede do Bitcoin, do Ethereum e de diversas outras criptomoedas [Bashir 2017, Antonopoulos 2017].

A *blockchain federada* ou *privada* tem uma composição conhecida, formada por n processos, cujas entradas e saídas estão sujeitas a permissões. Os nós são identificados, autenticados e autorizados. A blockchain irá atender melhor a interesses corporativos ou privados, onde os participantes têm papéis bem definidos e podem inclusive se organizar em grupos. Nesta categoria, encontra-se o Hyperledger Fabric [Cachin et al. 2016] e alguns outros projetos [Cachin and Vukolić 2017].

Importante salientar que uma gama nova de consenso para a blockchain tem sido proposta, principalmente como suporte a criptomoedas. Entretanto, muitos têm negligenciado as dificuldades associadas à resolução do problema, decorrentes, inclusive, dos resultados de impossibilidade, e têm apresentado protocolos inconsistentes. A comunidade BFT vem alertando para tais distorções [Cachin and Vukolić 2017, Gramoli 2017]. Como veremos nesta seção, o consenso da blockchain é fortemente voltado ao tipo de rede a que se destina (com permissão ou sem permissão).

5.3.1. Consenso para Blockchain Pública

Devido às diversas incertezas em relação aos participantes da rede, as blockchains públicas costumam adotar consensos abertos baseados em mineração, onde os mineradores competem entre si pela liderança do consenso, a partir de poder computacional, poder

de posse sobre a criptomoeda ou outros poderes de relevância para a eleição e que não podem ser monopolizados (de tal forma que os mesmos nós sempre saem vitoriosos).

Como a rede é aberta e ad-hoc, costuma-se retribuir o trabalho dos mineradores com incentivos financeiros, que são concretizados através das criptomoedas. Os incentivos são fundamentais na estratégia montada para tolerar ataques byzantinos, como ataques em conluio para subverter a rede. A governança dessa rede costuma estabelecer um conjunto de regras mínimas para a sua existência, além de favorecimento mútuo, de quem usa e de quem serve.

O consenso à base de provas resolve o desafio fundamental de realização de acordo em escala global num ambiente sujeito a falhas byzantinas, com participantes desconhecidos e anônimos (sem necessidade de autenticação). Atualmente, a prova de trabalho (*proof of work*) é uma das poucos abordagens de consenso bem sucedidas e resilientes a ataques Sybil[Douceur 2002] (ataques de personificação, quando usuários maliciosos se fazem passar por outros).

5.3.1.1. Consenso do Nakamoto e Prova de Trabalho – *Proof of Work* (PoW)

Como a blockchain implementa uma máquina de estados replicada, todos os nós da rede P2P enviarão requisições de transações para serem ordenadas pela máquina ao longo do tempo. As transações serão agrupadas em blocos e um consenso será executado em *rodadas* para possibilitar a ordenação total dos blocos (conjunto de transações). O consenso do Bitcoin proposto pelo Nakamoto [Nakamoto 2008] utiliza um *oráculo randômico* para eleição de um líder, que irá coordenar o consenso naquela rodada. Assim, para se eleger, o nó precisa realizar uma *prova de trabalho* ou *proof of work* (PoW), resolvendo um desafio criptográfico, que exigirá bastante poder computacional. Essa eleição descentralizada no jargão do Bitcoin é chamada *mineração*, e os nós que a realizam, *mineradores*.

O nó vencedor da eleição randômica (a mineração) é aquele que primeiro vence o desafio. Quando isso ocorre, o vencedor se auto-declara líder para os demais, propagando o seu valor, ou seja o seu bloco. Esse bloco somente será aceito por cada nó se for válido. Assim, a rodada do consenso chega ao seu término. É importante observar que a rodada do consenso termina sem que os nós interajam para efetuar confirmação do valor proposto pelo líder. Eles simplesmente validam e agregam o bloco aceito ao livro-razão distribuído, seguindo para uma próxima rodada de consenso. Todo esse processo de eleição e de validação descentralizado é a chave para o sucesso do consenso "descentralizado" do Bitcoin. Um esquema geral do consenso Nakamoto é apresentado no Esquema3.

Esquema 3 Consenso Nakamoto (*Esquema do Algoritmo*)

1. Request: Clientes enviam transações para todos os nós da rede;
2. Collect: Cada nó p_i da rede, ao receber as transações, as adicionam a um bloco b_i ;
3. Election: Em cada rodada k do consenso, um *oráculo randômico* escolhe um nó líder p_l para propagar o seu bloco b_l aos demais;

4. **Validate:** Cada nó p_i aceita o bloco b_l se ele é *válido* e se as transações contidas em b_l são *válidas*. Ver procedimentos de validação nas Seções 5.2.3.1 e 5.2.3.2.
5. **Update:** O nó p_i ao aceitar o bloco b_l irá adicioná-lo ao final do livro-razão e finaliza a rodada k . A posteriori, irá agregar o hash $H(b_l)$ ao próximo bloco a ser criado, mantendo assim a estrutura de corrente criptográfica.

O coração do consenso Nakamoto é o *oráculo randômico*, que promove a eleição descentralizada, através da mineração. Tal como em protocolos BFT tradicionais, ele faz uso de um oráculo para contornar o resultado de impossibilidade FLP. Tal como no Paxos, o oráculo elege um líder para coordenar o processo. Mas, diferentemente do Paxos, que propõe um protocolo determinístico, ele apresenta um protocolo probabilístico, cujas garantias de terminação e, sobretudo consistência, estarão comprometidas, e só serão obtidas ao longo do tempo. Isso ocorre porque o processo de eleição randômica poderá derivar diversos líderes que poderão propor blocos a serem subsequentemente validados e aceitos para a blockchain. Nesse sentido, a corrente pode se bifurcar, ocasionando os chamados *forks*. A forma de implementação do oráculo é um dos elementos inovadores trazidos pelo Nakamoto e será apresentada em detalhes na Seção 5.4.1.

Um outro aspecto inovador do Bitcoin é o *mecanismo de incentivos*, nunca proposto dentro da BFT tradicional. Ele permite com que o sistema contorne duas grandes dificuldades de projeto: o consenso com participantes anônimos e a tolerância a falhas byzantinas nesse contexto de anonimidade. Graças a esse mecanismo, os nós maliciosos tendem a seguir a ordem instituída pelos nós honestos, pois os ganhos (financeiros) em agir honestamente, tendem a compensar atuações maliciosas. O incentivo também torna-se viável porque a geração da moeda bitcoin integra-se ao processo de eleição randômica. O nó que vence o desafio recebe bitcoins em troca do trabalho. Assim, consenso (e sua eleição) e geração da moeda (como incentivo) integram-se numa simbiose salutar para viabilizar o acordo e, adicionalmente, a oferta de serviços de acordo sob demanda na Internet. A Seção 5.4.3 discorre sobre o mecanismo de incentivos.

Finalmente, diferentemente da BFT tradicional, os nós da rede aberta (e do Bitcoin), não precisam se identificar, de maneira a serem autenticados. Mas, é preciso que a cada nó seja atribuído um *endereço*; é através dele que o nó é reconhecido nas transações do ativo, e todo o processo de validação pode ser realizado. Para tanto, o Bitcoin adota uma *gerência de identidades descentralizada*, que possibilita a *pseudo anonimidade* da rede. Esse é mais um aspecto inovador que será explorado na Seção 5.4.2.

5.3.1.2. Prova X (PoX)

Apesar das diversas inovações, o PoW do Bitcoin apresenta desvantagens evidentes. A primeira diz respeito à latência de decisão do bloco, que é relativamente alta, e em consequência a vazão (*throughput*) (quantidade de transações validadas no tempo), que é relativamente baixa. Uma outra, bem mais crítica, seria a sustentabilidade do processo de mineração, que exige alto poder computacional, gasto desmesurado de energia, sem retorno proporcional. Há ainda o risco de controle da rede por pool de mineradores que tenderiam a centralizar a decisão e realizar ataques diversos (*selfish mining attacks*).

Diversos trabalhos vêm sendo propostos no intuito de mitigar tais aspectos e favorecer decisões do consenso com base em outros critérios [Bonneau et al. 2015, Bano et al. 2017]. Eles são chamados *Prova X (PoX)*, pois propõem a substituição do desperdício computacional por trabalho válido, derivado de recursos alternativos e acessíveis. Eles advogam a adoção de uma "mineração virtual".

Prova de Posse – Proof of Stake (PoS). É uma das proposta promissoras de PoX. O princípio: no lugar de gastar tempo e recursos computacionais para sair vitorioso da eleição do consenso, estabelece-se critérios sustentáveis para a eleição, com base na posse (ou interesse) que cada nó tem na rede, mais especificamente, com base na quantidade de recursos em criptomonedas que cada nó possui. Nesse caso, os votos dos participantes para a eleição é ponderado pelo investimento já realizado na rede; o peso pode obedecer a vários critérios. Nota-se que a PoS mantém de certa forma a lógica da PoW, pois a posse de criptomonedas na rede é diretamente proporcional ao poder computacional na rede.

Um dos desafios da PoS é o acompanhamento da evolução do status de cada interessado (*stakeholder*) na rede. A Peercoin foi uma das primeiras moedas a propor a PoS e, atualmente, o Ethereum projeta migrar de PoW para PoS. Alguns protocolos são: Ouroboros [Kiayias et al. 2016], Snow-White [Bentov et al. 2016]. Variações da PoS são a *Prova de Idade (Proof-of-coin-age)*, quando a prova da posse é pelo tempo em que as moedas foram alocadas à rede; *Prova de depósito (Proof-of-deposit)*, quando os mineradores alocam uma qtde específica da moeda para a rede (sem possibilidade de gasto).

Prova de Capacidade – Proof of Capacity (PoC). Nesse caso, os votos dos participantes para a eleição são ponderados pela alocação de uma quantidade não negligenciável de espaço em disco. Um exemplo é o Permacoin [Miller et al. 2014] que usa a PoC para implementação de um storage distribuído robusto.

Prova de Tempo Decorrido – Proof of Elapsed Time (PoeT). O projeto Hyperledger Sawtooth propõe uma eleição à base de temporização a partir do *Trusted Execution Environment (TEE)* presente na arquitetura do Intel SGX. No caso, os participantes solicitam uma fatia de tempo para o seu enclave de confiança. O chip com menor tempo de espera recebido do enclave é eleito o líder. De certa maneira, a PoeT adota uma estratégia próxima a de consensos para ambientes síncronos, com uso de um modelo híbrido, à base de *wormholes*: um componente de confiança em HW.

5.3.2. Consenso para Blockchain Federada/Privada

Como a rede é controlada, com n participantes bem identificados pela federação e uma composição controlada, protocolos clássicos BFT e consenso byzantino determinístico podem ser adaptadas para a blockchain. Adicionalmente, não há necessidade de uso de incentivos para obtenção do acordo, dado que a federação de interessados (*stakeholders*) pode estabelecer o seu próprio modelo financeiro de retribuição. Os incentivos podem ser usados para outros fins, mas, diferentemente do consenso à base de provas, não são imprescindíveis ao consenso.

A principal vantagem de uso do consenso BFT para a blockchain é a latência de decisão (quase em tempo real, comparativamente ao consenso à base de provas) e a alta vazão de transações. Há algumas dificuldades inerentes aos protocolos byzantinos:

(i) eleição de um líder correto (não Bizantino) para guiar o acordo e mudança de visão do grupo, na ocorrência de falhas e sobretudo *churn* (entradas e saídas aleatórias de nós na rede, de forma intensa); (ii) adoção de requisitos de segurança custosos, como um cripto-sistema de emissão de chaves públicas para garantir autenticação (PKI); (iii) maior vulnerabilidade a ataques Sybil. Em [Cachin and Vukolić 2017, Bano et al. 2017] diversos desses protocolos são apresentados, com suas principais inovações e desafios. Nesta seção, destacamos os principais.

5.3.2.1. PBFT – Practical Byzantine Fault Tolerance

O clássico PBFT (*Practical BFT*) [Castro and Liskov 1999, Castro and Liskov 2002] é um modelo de referência para muitos dos protocolos BFT que vem sendo adaptados para a blockchain. Segundo as análises, o seu desempenho para implementação de uma máquina de estados replicada bizantina é similar ao de uma máquina em que réplicas falham por parada. Seu principal diferencial foi propor uma solução eficiente com artifícios de segurança mínimos, como a eliminação de assinaturas de chave pública, na autenticação de mensagens, e substituição por MAC (*message authentication code*).

O BFT-Smart [Bessani et al. 2014] é um outro projeto promissor para implementação da RME bizantina, adotando um consenso similar ao PBFT. Vem sendo considerado uma das implementações mais robustas e eficientes de BFT, tanto para redes pequenas como em larga escala. Além disso, assim como o PBFT, o BFT-Smart [Sousa et al. 2018] também foi recentemente incorporado ao projeto Hyperledger.

O consenso PBFT também é realizado em rodadas coordenadas por um líder. Os líderes se sucedem em sequências de visões. Dentro de cada visão, os nós monitoram o comportamento do líder, que pode ser honesto ou malicioso; assim, caso os nós suspeitem da falha do líder, seja por parada (comprometendo a *liveness* do protocolo), seja por comportamento Bizantino (comprometendo a *safety*), eles promovem uma mudança na visão do grupo (*view change*). Além de eleger novo líder, o protocolo de mudança de visão deve garantir acordo sobre mensagens entregues durante a visão anterior, ou seja, deve assegurar a *comunicação em sincronia com as visões* [Charron-Bost et al. 2010]. O PBFT adota como premissa o modelo *parcialmente síncrono* [Dwork et al. 1988], logo depende da satisfação de requisitos temporais para garantir a sua terminação. Ele apresenta complexidade de (n^2) mensagens para n processos (réplicas) e atende às propriedades do consenso bizantino, desde que $n > 3f$, onde f é o número máximo de processos maliciosos.

O Esquema 4 apresenta um esboço do funcionamento de uma rodada de protocolo à base do PBFT (PBFT-like) para o caso normal, quando o líder não é suspeito de falhar.

Esquema 4 PBFT-like (*Esquema do Algoritmo para o Caso Normal*)

1. Request. Clientes enviam requisições de transações para processo líder ou para demais processos que irão repassar ao líder;
2. Pre-Prepare. O processo líder p_l monta um bloco $b[k]$ com transações válidas recebidas. O bloco $b[k]$ irá apontar para o hash do último bloco $b[k - 1]$ agregado

à corrente de blocos. Em seguida, envia uma mensagem PRE-PREPARE para todos os demais processos propondo o acordo sobre $b[k]$;

3. **Validate.** Um processo p_i aceita receber PRE-PREPARE se a mensagem foi autenticada, a proposta do líder p_l é válida e o bloco $b[k]$ nunca foi decidido. Ou seja, se as transações têm assinaturas digitais válidas, não incorporam duplo-gasto e não foram confirmadas anteriormente (Ver validação nas Seções 5.2.3.1 e 5.2.3.2);
4. **Prepare.** Quando um processo p_i aceita o PRE-PREPARE, ele envia uma mensagem PREPARE contendo $(p_l, b[k])$ para todos os demais processos;
5. **Execute.** Quando um processo p_i recebe $\lceil (n + f)/2 \rceil$ mensagens PREPARE contendo $(p_l, b[k])$, então ele envia uma mensagem COMMIT contendo $(p_l, b[k])$ para todos os demais processos;
6. **Commit.** Quando um processo p_i recebe $\lceil (n + f)/2 \rceil$ mensagens COMMIT contendo $(p_l, b[k])$, então ele confirma o consenso sobre $(p_l, b[k])$, adiciona $b[k]$ ao final do livro-razão, e finaliza a rodada do consenso.

5.3.2.2. Outros Consensos BFT

Algumas propostas de consenso byzantino para a blockchain federada têm sido feitas com mecanismos e abstrações distintas do PBFT. Podemos classificá-las em dois grandes grupos: (1) aquelas apresentadas como *white papers*, sendo a base de diversas criptomoedas. Em sua grande maioria, como não fornecem os elementos completos para análise pública e verificação formal dos protocolos, não há como garantir que de fato implementam as propriedades do consenso byzantino e têm o desempenho anunciado; (2) propostas recentes apresentadas pela comunidade BFT e de segurança em forma de artigos científicos.

No grupo (1) encontramos protocolos bem inovadores, como o recente *Swirls Hashgraph* [Baird 2016], que propõe um consenso BFT probabilístico, para modelo assíncrono com $f < n/3$ participantes, cuja blockchain é armazenada num *DAG (Directed Acyclic Graph)*. Diferentemente da PoW do Bitcoin, os autores argumentam que resolvem o BFT com garantias de consistência forte, atingindo decisões com probabilidade 1 e resistência a ataques Sybil. A estrutura do Hashgraph admite bifurcações como mecanismo natural do seu crescimento. Vários blocos podem ser minerados e a latência de decisões é em tempo quase real. Diferentemente dos demais, ele não se fundamenta em eleição de líder e não utiliza o mecanismo de troca de mensagens com votos. De fato, não precisam encaminhar votos na rede porque propõem um protocolo de votos virtual. Em resumo, o *Swirls Hashgraph* apresenta muitas inovações, em diversos níveis, mas, é um protocolo proprietário e ainda não há implementações e análises independentes sobre seu funcionamento.

No grupo (2), alguns protocolos adotam o **Raft** [Ongaro and Ousterhout 2014], uma variação do clássico Paxos [Lamport 1998]. O Raft possui algumas implementações abertas, é adotado no projeto Quorum, do JPMorgan Chase, para desenvolvimento de uma blockchain à base do Ethereum, além dos projetos Juno e Tangaroa [Copeland and Zhong 2014]. [Crain et al. 2017] apresentam um consenso BFT diferenciado, que não se baseia em eleição de líder, nem em assinaturas, nem em randomização. Ele propõe uma

redução do consenso bizantino de valor múltiplo ao consenso bizantino binário, e com base em instâncias simultâneas de consenso binário converge a uma decisão, num modelo parcialmente síncrono.

5.4. Bitcoin e Mineração

A criptomoeda Bitcoin (www.bitcoin.org) é a prova de conceito seminal para o sucesso da blockchain. Nesta seção, serão explorados alguns conceitos técnicos associados ao protocolo original [Nakamoto 2008], com a apresentação dos elementos inovadores associados ao consenso PoW, descrito na Seção 5.3.1.1. Também serão abordadas as redes de mineradores e suas associações, além de possíveis oportunidades de ataques.

5.4.1. Mineração e Prova de Trabalho

No jargão do Bitcoin, o processo realizado pelos nós para vencer a eleição randômica do consenso é designado *mineração*. Os mineradores são computadores com hardware e software dedicados a resolver um desafio computacional (*hash puzzle*) que envolve essencialmente o uso de força bruta.

A mineração consiste em aplicar a função *hash* H sobre o *header do bloco* sucessivas vezes, pela variação do valor do *nonce*, até que o resultado do *hash* seja menor do que a dificuldade alvo imposta (*target*). No geral, o *hash* será aplicado aos campos descritos na Seção 5.2.3.2, a saber: o *nonce*, o apontador para o bloco anterior (*prev*), o hash da raiz da árvore Merkel (*mrkl_root*), além do *timestamp* e dificuldade alvo (*target*). Assim, espera-se determinar um *nonce* que satisfaça:

$$H(\textit{nonce} \parallel \textit{prev} \parallel \textit{mrkl_root} \parallel \textit{timestamp} \parallel \textit{target}) < \textit{target}$$

Como a função *hash* é resistente à colisão (Definição 1), a determinação do *nonce* que satisfaça a inequação não será nada trivial, necessitando um algoritmo força-bruta para que diversos valores sejam testados, um a um, até que se encontre o *nonce* adequado. A mineração consiste em aplicar este processo de tentativa e erro até que se ache um acerto. O nós minerados são aqueles que têm suficiente poder computacional para realizar esse jogo.

Dificuldade A dificuldade para resolver o *puzzle* aumenta com a diminuição do alvo. A intuição é a seguinte: Se a função H tem resultado de n bits, então ela pode obter qualquer valor dentre os 2^n possíveis. Resolver o *puzzle* consiste em encontrar um valor de saída que seja $< \textit{target}$ (esse alvo deve ser relativamente menor do que o espaço de 2^n soluções possíveis). Mas, se *target* é um valor alto, temos muitas possibilidades de escolha e o desafio é trivial; por outro lado, se tem um valor muito baixo, o *puzzle* é extremamente difícil. Assim, quanto menor o espaço de busca determinado pelo *target*, maior será a dificuldade para se achar o *hash* desejado, e a única possibilidade de acerto é variar o *nonce* um a um até que o hash atinja o valor alvo.

Em consequência, o tempo de mineração depende do alvo adotado. Pelas regras instituídas, a dificuldade de mineração muda a cada 2016 blocos. Como a rede está programada para minerar um bloco a cada 10 minutos, isso corresponde a cerca de 2 semanas

(2016 * 10 minutos). O alvo é ajustado com base na eficiência dos mineradores em encontrar o bloco, considerando-se o período passado de mineração para os 2016 blocos anteriores, de acordo com a fórmula:

$$next_target = (previous_target * 2016 * 10) / (time\ to\ mine\ the\ last\ 2016\ blocks)$$

Corrente de Blocos Mais Longa Os mineradores que conseguem efetuar a prova de trabalho, imposta pelo desafio, vencem a competição de mineração e são autorizados a publicar um bloco de transações pendentes na blockchain. Ao mesmo tempo, propagam o seu bloco vitorioso para os demais nós da rede P2P.

Como o processo é randômico, há a possibilidade de haver mais de um vencedor para a mesma rodada de eleição, além disso, a rede é assíncrona, logo, diversos blocos de mesma altura (relativos à mesma rodada de consenso) podem vir a serem adicionados à blockchain pelos nós da rede P2P, ocasionando uma "bifurcação" na corrente, o chamado *fork*. A Figura 5.6 ilustra esse processo, quando um bloco pai passa a ter mais de um filho. De fato, devido à bifurcação, não haverá um estado global único compartilhado, e diversas correntes (ou diversas visões da blockchain) passam a coexistir ao mesmo tempo.

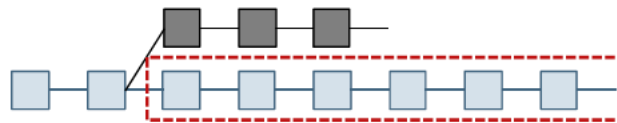


Figure 5.6. Bifurcação da Corrente de Blocos

Para resolver esta inconsistência eventual no estado da blockchain, os nós devem sempre estender a corrente mais longa (a que acumula maior altura) ou, em outras palavras, a que representa a maior prova de trabalho acumulada, ou o maior poder computacional empregado no sistema. Se todos nós adotam essa mesma estratégia, ao cabo de um tempo, a blockchain irá convergir para um estado global consistente. Assim, quanto mais antiga é uma decisão sobre um bloco na corrente, maior é a probabilidade de que esta decisão não seja demovida por outro líder minerador. *O acordo do consenso passa a ser contínuo no tempo.*

O intervalo de 10 minutos definido para mineração dos blocos é uma solução de compromisso entre o "tempo de confirmação das transações" x "a probabilidade de bifurcações". Assim, ao diminuir o intervalo, aumenta-se a vazão de transações confirmadas, mas, também aumenta-se a frequência de bifurcações. Inversamente, se o intervalo for maior, reduz-se a vazão e em consequência a quantidade de bifurcações. Na prática, estima-se que são necessários cerca de 6 rodadas de consenso (1 hora) para que a decisão tomada seja confiável.

5.4.2. Endereços e Pseudo Anonimidade

O Bitcoin adota uma *gerência de identidades descentralizada* para atribuição de *endereços* aos nós da rede. Assim, no lugar de existir uma única autoridade centralizadora,

responsável por emitir novas identidades (passíveis de autenticação), cada nó, independentemente, gera a sua própria identidade (no caso, endereço) e a anuncia para os demais nós da rede, quando pertinente. O endereço do Bitcoin será o hash H da chave pública pk , i.e., $H(pk)$, gerada pelo nó para participar de transações. Eles são criados livremente pelos nós, de acordo com a sua conveniência.

De certa maneira, a gerência de identidades descentralizada apresenta um mecanismo auto-imune aos ataques Sybil, já que, por projeto, a rede Bitcoin aceita diversas identidades (endereços) para um mesmo nó; portanto, se protege de ataques de personificação. Por outro lado, tal descentralização possibilita ao Bitcoin o oferecimento de uma rede cujas transações ocorrem de maneira pseudo-anônima. Ou seja, conhece-se os endereços dos emissores e receptores dos ativos, mas, não se sabe exatamente quem são, dado que os endereços não estão ligados a um nome universal ou endereço IP. Essa pseudo-anonimidade é um dos princípios basilares do Bitcoin.

5.4.3. Mecanismo de Incentivo e Geração da Criptomoeda

O incentivo no Bitcoin ocorre em dois contextos: blocos e transações. O primeiro acontece quando o minerador vence a eleição, quebrando o desafio proposto. Como recompensa, ganham uma quantidade de bitcoins recém-emitidos. O segundo ocorre na execução das transações propriamente ditas. Os nós, nesse caso, são recompensados por executarem as ações previstas nas transações.

Recompensa de Bloco. Na geração do bloco, o minerador tem autorização para incorporar uma transação especial, a *coinbase transaction*, que consiste na geração de novas criptomoedas (no caso, o bitcoin), segundo uma regra pre-estabelecida pela rede. Essa transação especial incorpora o endereço para o qual as moedas deverão ser destinadas. Assim, seguindo a lógica da recompensa, o endereço de destino será determinado pelo minerador.

Geração da Moeda. No artigo original, Nakamoto propõe um engenhoso mecanismo de geração da moeda, que obedece a uma série geométrica decrescente e, que, por coincidência, é próxima à curva de distribuição do ouro (de onde vem a expressão do bitcoin ser o "ouro digital"). Ele estabeleceu que, a cada 4 anos, a quantidade α de moedas gerada por bloco deveria ser dividida pela metade. Assim, nos períodos de 2009-2013, a recompensa por bloco minerado foi de $\alpha = 50$ bitcoins; de 2013-2017, $\alpha = 25$ bitcoins e, atualmente $\alpha = 12,5$ bitcoins. Ao mesmo tempo, se somarmos os valores da série geométrica, que é finita, teremos um total de 21 milhões de bitcoins, gerados ao longo dos anos. Assim, o artigo propõe criar uma moeda, com geração controlada, cuja emissão irá cessar no ano de 2140, de acordo com a série. Ou seja, o sistema está programado para emitir apenas 21 milhões de bitcoins! Este é, inclusive, um dos motivos pelos quais há uma forte especulação em torno da moeda.

Taxa de Transação. A taxa de transação pode ser estabelecida de forma a incentivar os mineradores a agregarem aquelas transações no seu bloco; ou, de modo geral, de forma a oferecer aos usuários demandantes uma qualidade de serviço desejada. Ela pode ser determinada, por ex., como uma taxa aplicada aos valores transacionados (dentro do bloco). De modo geral, ela equivale à subtração da soma dos valores de entrada menos os de saída. O mecanismo de compensação para as transações é mais sofisticado no projeto

no Ethereum, quando os demandantes costumam estimar valores de gás como forma de retribuição pela execução dos métodos dos contratos inteligentes. A recompensa é ainda uma área de pesquisa muito pouco estudada, exigindo elementos da teoria dos jogos para a sua apreensão e incorporação num modelo de negócios promissor.

O consenso PoW, juntamente com o mecanismo de recompensa, são a chave para tolerar ataques de agentes maliciosos na rede. Não é possível para uma minoria de mineradores manipular as transações, pois a rede como um todo não aceitaria transferência de ativos (ou pagamentos) que não foram autorizados pelo proprietário dos ativos (ou bitcoins). Além disso, os nós sempre estendem a corrente mais longa. Para um ataque cibernético, seria necessário que uma maioria (51%) de nós mineradores entrasse em conluio. Isso envolveria um custo computacional altíssimo [Nakamoto 2008].

5.4.4. Ataques e o Problema do Duplo Gasto

Nas moedas tradicionais, o problema do gasto duplo é inexistente, uma vez que são compostas de matéria e possuem propriedades físicas distintas. O bitcoin, no entanto, não possui tais características, sendo formado essencialmente por bits, cujo conceito é abstrato. Sendo assim, em princípio, seria simples duplicá-los e, por exemplo, pagar duas vezes com a mesma moeda em duas transações diferentes. Como não há um banco ou lastro confiável que possa impedir que os usuários gastem o mesmo dinheiro duas vezes, torna-se necessário criar uma solução adequada [Karame et al. 2012, Conti et al. 2017].

Um ataque abordando a possibilidade de duplo gasto explora o seguinte cenário: Um cliente desonesto C_d cria uma transação T entre ele e uma loja (L), em um tempo t usando um conjunto de bitcoins B_C para comprar um produto de L . C_d envia pela rede a mensagem T . Quase que instantaneamente, em t' , C_d cria e envia outra transação T' usando o mesmo conjunto de bitcoins B_C para outra loja L' ou outra carteira que esteja sob seu controle. Se a loja L entregar o produto, segundo este cenário, o duplo gasto pode se concretizar.

Na blockchain, o gasto duplo é evitado com a validação do bloco e das transações sempre que há uma decisão. Quando uma transação é incluída em um bloco, prova-se que nenhuma transação anterior gastou os mesmos bitcoins e que as transações futuras também não o farão (Seção 5.3.1.1). Assim, nas transações T e T' , o minerador estará apto a identificar que ambas as transações tentam usar a mesma entrada B_C durante a propagação da transação, fazendo com que uma seja rejeitada e outra aceita.

Entretanto, apesar do uso do PoW, do rigoroso ordenamento das transações, ainda assim, é possível ataque de duplo gasto, com base na exploração da bifurcação da corrente da blockchain, conforme o seguinte cenário: (i) parte da rede de mineradores aceita a transação T e envia a confirmação para a loja, que então entrega o produto ao cliente desonesto C_d ; (ii) ao mesmo tempo, outra parte da rede de mineradores aceita a transação T' e leva a blockchain a criar um *fork* na rede; (iii) a loja recebe a confirmação da transação T' depois de aceitar a transação T , então perde o produto, e (iv) a maioria dos mineradores decide pela transação T' como válida. Existem variantes deste tipo de ataque: (a) *Finney attack*; (b) *Vector 76*; (c) *Ballance*; (d) *Goldfinger* ou $>50\%$. Cada uma delas explora vulnerabilidades que permitem realizar o duplo gasto [Conti et al. 2017].

5.4.5. Associação de Mineradores

No contexto da mineração de criptomoedas, a associação de mineradores pode ser definida como um agrupamento de recursos computacionais dos mineradores, que compartilham seu poder de processamento em uma rede e dividem a recompensa igualmente, de acordo com a quantidade de trabalho contribuída por cada um para a probabilidade de encontrar um bloco. Em uma associação de mineradores pode haver centenas ou milhares de associados [Eyal 2015].

Este modelo de mineração foi motivado devido ao aumento de competição para geração de um bloco e obtenção de receitas sazonais. Como solução, os mineradores reuniram seus recursos de forma a gerar blocos mais rapidamente e, portanto, receber uma parte da recompensa do bloco ao invés de aleatoriamente em intervalos cada vez maiores [Eyal and Sirer 2014]. Embora ingressar em um pool não altere a receita esperada de um minerador, ele diminui a variação e torna as receitas mensais mais previsíveis. Foram propostos diferentes métodos para repartir as receitas, como por exemplo *Pay-per-Share*, *Proportional*, *Bitcoin Pooled Mining*, *Pay-per-last-N-shares*, *Geometric* e *Double Geometric* [Rosenfeld 2011].

Atualmente, as associações de mineradores são controladas por gerenciadores que encaminham unidades de trabalho não resolvidas para os membros. Os mineradores geram provas de trabalho parciais (PPoWs) e provas de trabalho completas (FPoWs), e as submetem ao gerente como ações. Uma vez que um mineiro descobre um novo bloco, ele é submetido ao gerente junto com o FPoW. O gerente transmite o bloco na rede Bitcoin para receber a recompensa de mineração e distribui a recompensa aos mineradores participantes com base na fração de ações contribuída quando comparada com os outros associados. Assim, os participantes são recompensados com base nos PPoWs, que não têm absolutamente nenhum valor no sistema Bitcoin [Conti et al. 2017].

5.4.5.1. Segurança e Ataques

Nos últimos anos, a exploração de vulnerabilidades e ataques às associações de mineradores aumentou, sendo possível classificá-los em ataques internos e ataques externos. Os ataques internos são aqueles onde um grupo de mineradores desonestos age de forma maliciosa com a associação, objetivando a coleta de mais recompensas do que lhe seria devido ou para interromper a funcionalidade da associação de forma a distanciar-se das tentativas de mineração bem sucedida. Nas investidas externas, os mineradores podem usar do seu poder computacional para executar fraudes como o duplo gasto.

Um tipo bastante popular de ataque é o *Block Discarding* (BD). Mineradores desonestos realizam a ocultação de informação, retendo um bloco minerado e realizando a sua revelação de maneira muito seletiva, visando: (a) obter uma recompensa injusta maior do que a sua quota de poder de computação gasto, e (b) confundir outros mineradores e levá-los a desperdiçar seus recursos em uma direção errada [Courtois and Bahack 2014].

O *Block Withholding* (BW) é um tipo ataque similar ao BD, onde um membro da associação nunca publica um bloco minado para sabotar a receita da associação. No entanto, ele submete ações consistindo em PPoWs, mas não em FPoWs. Dois cenários

de *Block Withholding* são destacados: a) “sabotagem” e b) “espera”. Em (a), o adversário não ganha nenhum bitcoin, apenas faz com que outros membros do pool percam. Em (b) o adversário realiza um bloqueio complexo que esconde um ataque semelhante ao *Block Discarding* [Rosenfeld 2011, Bag et al. 2017].

O *Fork after withholding* (FAW) [Kwon et al. 2017], proposto recentemente, mostra que a recompensa dos atacantes do BWH é o limite inferior dos atacantes do FAW. A recompensa extra por um ataque FAW ao operar em múltiplas associações de mineradores é cerca de 56% maior do que o ataque de BWH e chega a até quatro vezes ocorrências por associações do que o ataque do BWH.

O *Pool Hopping* (PH) usa as informações sobre o número de ações enviadas pela associação. Neste ataque, o minerador desonesto realiza uma análise contínua do número de ações submetidas por colegas mineradores ao administrador do grupo, a fim de descobrir um novo bloco [Rosenfeld 2011]. A ideia é que, se um grande número de ações já tiver sido submetido e nenhum novo bloco tiver sido encontrado até agora, o adversário receberá uma parcela muito pequena da recompensa, porque ela será distribuída com base nas ações enviadas. Portanto, em algum momento, o minerador egoísta pode decidir por mudar para outro grupo ou minerar de forma independente.

Por fim, o *Bribery* [Bonneau 2016] pode obter a maioria dos recursos de computação por um curto período de tempo via suborno. Discute-se três maneiras de introduzir suborno na rede: (i) O minerador desonesto paga diretamente ao proprietário pelos recursos de computação e estes proprietários mineram os blocos atribuídos ao minerador desonesto, (ii) O minerador desonesto forma uma associação pagando maior retorno pela mineração, e (iii) O atacante tenta subornar através do Bitcoin criando um *fork* contendo dinheiro de suborno disponível a qualquer minerador que adote a *fork* desonesta.

5.5. Hyperledger

O Hyperledger (www.hyperledger.org) é uma plataforma de código aberto, mantida pela Linux Foundation e diversas corporações, dentre elas a IBM, que propõe o Fabric [Cachin et al. 2016]. O Fabric é uma implementação de blockchain privada voltada para aplicações empresariais com objetivo de atender aos múltiplos e variados requisitos dos aplicativos de negócio, alavancar os estudos da tecnologia blockchain e permitir a escalabilidade de serviços nele baseados [Fabric 2017] [Nguyen et al. 2018]. Sua arquitetura modular oferece altos graus de confiabilidade, resiliência, flexibilidade e escalabilidade [Nguyen et al. 2017], possibilitando a criação de vários módulos conectáveis e viabilizando a implantação de serviços diferenciados, como consenso, armazenamento e filiação de membros [Fabric 2017] [Nguyen et al. 2018].

No Fabric cada participante registra sua participação utilizando o serviço de filiação a rede (*Membership Service Provider* - MSP) para obter acesso ao sistema [Nguyen et al. 2018], possibilitando, assim, o uso de consensos bizantinos e desobrigando o uso de consensos baseados em prova de trabalho (PoW) para validar as transações e proteger a rede, e provendo transações seguras, privadas e confidenciais [Fabric 2017].

O Fabric executa contratos inteligentes (*chaincodes*), disponibiliza o consenso PBFT [Castro and Liskov 1999], permite a um grupo de participantes criar um livro razão

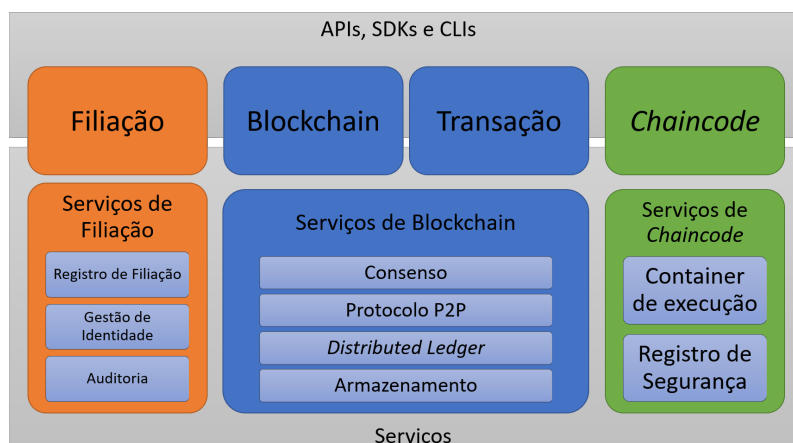


Figure 5.7. Arquitetura de Referência Hyperledger

separado das transações no qual apenas os participantes deste grupo possuem cópias do *ledger* através da criação de canais, permite persistência utilizando chave-valor facilitando as consultas através de um banco de dados não relacional, suporta eventos, abstrai da complexidade dos protocolos de comunicação através do *standard developed Kit* (SDK), promove segurança através de autoridades certificadoras e promove suporte a REST APIs e CLIs.

O Fabric diferencia os nós pela sua responsabilidade e os categoriza em: nós de validação e nós de não-validação. Os nós de validação são responsáveis pela execução de consenso, validação de transações e manutenção do ledger. Os nós de não-validação funcionam como um *proxy* para conectar clientes para os nós de validação (emitir transações) e podem validar transações. Um nó de não-validação não executa transações [Cachin et al. 2016]. A responsabilidade de cada nó é atribuída durante o processo de filiação à rede através do serviço de filiação. O serviço de filiação será apresentado posteriormente.

5.5.1. Arquitetura

O Fabric possui uma arquitetura de referência organizada em 3 categorias lógicas: Serviços de filiação ou *Membership*, Serviços de Blockchain e Serviços de *Chaincode*. A Figura 5.7 apresenta a organização destas categoria no Fabric.

5.5.1.1. Serviços de Filiação ou Membership

Os serviços de filiação (MSP- *Membership Service Provider*) consistem em uma infraestrutura composta por várias entidades que fornecem abstrações para operações de associação: registro e gerenciamento de identidade, privacidade, confidencialidade e auditoria na rede [Nguyen et al. 2018]. Estes serviços validam a identidade do usuário, registram-no no sistema e fornecem todas as credenciais necessárias para um participante na rede, seja ele um nó ativo ou uma aplicação capaz de invocar um *chaincode*.

O framework base do serviço de filiação é uma associação da criptografia de chave pública e o consenso. Através do uso da infraestrutura de chave pública, ICP, as mensagens podem ser trocadas de maneira segura e a identidade das partes podem ser ates-

tadas. Com o uso das entidades que compõem o serviço de gerenciamento de membros é possível identificar um usuário individual e permitir a execução de operações a partir das permissões atribuídas ao mesmo [Nguyen et al. 2018]. Com o uso deste mecanismo o Fabric classifica os nós do sistema, identifica e credencia usuários e permite a instalação e invocação de contratos inteligentes.

A abordagem adotada pelo Fabric para conciliar o gerenciamento de identidade com a privacidade de usuário implementa três artifícios: adicionar certificados a transação para implementar uma blockchain permissionada; utilizar um sistema com dois níveis certificados: certificados de inscrição estáticas (*static enrollment certificates* - ECerts) juntamente com certificados de transação (*transaction certificates* - TCerts); e oferecer mecanismos para ocultar o conteúdo das transações para membros não autorizados do sistema [Nguyen et al. 2018]. Os ECerts são certificados de longo prazo. Eles são emitidos para todos os papéis da rede: usuários, nós validadores e não-validadores [Nguyen et al. 2018]. Os TCerts são certificados de curto prazo para cada transação do usuário. Eles são emitidos pela Autoridade de Certificação de Transação (*Transaction Certificate Authority* - TCA) após a autenticação das credenciais de inscrição fornecidas pelo usuário. São emitidas exclusivamente para usuários e autorizam, de forma segura, uma transação. Podem ser configuradas para não revelar ou revelar seletivamente as identidades envolvidas na transação [Nguyen et al. 2018]. Desta forma, os TCerts possibilitam a utilização de pseudônimos para representar usuários inscritos na rede.

5.5.1.2. Serviços de Blockchain

O *ledger* no Fabric, conjunto de blockchain e logs de transações, encontra-se replicado na rede, isto é, cada participante possui sua própria cópia do *ledger*. Neste contexto, os serviços de blockchain gerenciam o estado do *ledger* distribuído entre os nós através de mensagens bidirecionais utilizando comunicação P2P sobre o http [Nguyen et al. 2018].

Como regra básica, as transações que compõem o blockchain devem ser escritas no *ledger* na ordem em que ocorrem, mesmo que possam estar entre diferentes conjuntos de participantes dentro da rede. Para que isso aconteça, a ordem das transações devem ser estabelecida e um método para rejeitar transações ruins que tenham sido inseridas no *ledger* por engano (ou maliciosamente) deve ser posto em prática [Nguyen et al. 2017].

O Fabric, por ser uma implementação de blockchain privada, permite a implementação de protocolos de consenso BFT, como por exemplo o PBFT [Castro and Liskov 1999], no qual fornece um mecanismo para que as réplicas do *ledger* se comuniquem entre si para manter cópias consistente, mesmo em caso de corrupção. Tal implementação é alavancada através dos serviços de blockchain, que efetuam a comunicação entre os nós através de 4 tipos de mensagem: mensagem de descoberta, mensagem de transação, mensagem de sincronização e mensagem de consenso.

As mensagens de descoberta são utilizadas como ponto de partida para descobrir todos os pares da rede e iniciar, caso o número do último bloco do nó emissor for menor que o do nó receptor, a sincronização do *ledger* com a rede. Todo nó possui pelo menos um endereço e porta de um participante da rede para efetuar a descoberta.

Após a descoberta, é iniciado o protocolo de sincronização, no qual um nó sin-

croniza seu ledger com os demais nós da rede. Algumas das mensagens usadas são: SYNC_GET_BLOCKS, SYNC_STATE_GET_SNAPSHOT ou SYNC_STATE_GET_DELTAS e recebe SYNC_BLOCKS, SYNC_STATE_SNAPSHOT ou SYNC_STATE_DELTAS, respectivamente. Na mensagem de sincronização o plug-in de consenso determina como a sincronização deverá ser realizada enviando sua respectiva mensagem.

As mensagens de transação manipulam o *ledger* e estão sempre associadas a um *chaincode* e um ambiente de execução (permissões e linguagem de execução). Estas mensagens podem ser de 3 tipos: *deploy*, *invoke* e *query*. Uma mensagem de *deploy* instala o *chaincode* especificado no blockchain, particularmente nos nós de não-validação, enquanto que *invoke* e *query* invocam e consultam um *chaincode* respectivamente [Nguyen et al. 2017]. As mensagens de consenso tratam as transações a serem incluídas no *ledger* convertendo uma mensagem CHAIN_TRANSACTION para CONSENSOS emitida para os nós de validação. O plug-in de consenso recebe esta mensagem e a processa de acordo com seu algoritmo. Na versão 1.1 do Fabric é disponibilizado o consenso PBFT através do pacote abcpbft.

O ciclo de vida de uma transação inicia-se nos nós de validação que as recebem e processam nas seguintes fases: Pré-validação: valida o certificado de transação na autoridade certificadora raiz, verifica a assinatura do certificado de transação incluído na transação e verifica se a transação não é repetida; Consenso: adicionam a transação em ordem total; Pré-execução: validam a validade do certificado de transação/inscrição em relação ao período de validade atual, descriptografam o corpo da mensagem e verificam se a mesma encontra-se bem formada; Execução: o *chaincode* é enviado ao container e executado; Confirmação: as atualizações advindas da transação são incluídas no *ledger* e o cliente que invocou a transação recebe a confirmação da transação

5.5.1.3. Serviços de Chaincode

Os serviços de *chaincode* fornecem uma maneira segura e otimizada para a execução de contratos inteligentes. É um container para execução do *chaincode*, com políticas de permissão e conta com SDK na linguagem GO para a execução. O *chaincode* é um código de nível de aplicativo implantado como uma transação para ser distribuído na rede e gerenciado por cada nó de validação.

A comunicação entre os nós de validação e seus *chaincodes* é baseada em um fluxo bidirecional de comunicação através da *shim layer*. A *shim layer* é uma camada entre o container de execução e o *chaincode* responsável pela manipulação do protocolo de comunicação entre os nós de validação e o *chaincode*. As mensagens trocadas são do tipo *ChaincodeMessage* e possuem os campos *type*, *payload*, *uuid*. *Type* indica o tipo da mensagem, *payload* possui informações importantes a comunicação e seu conteúdo depende do *type* da mensagem e *uuid* é o identificador único da mensagem.

Interface de Programação de Aplicativos (API) e Interface de Linha de Comando (CLI) As APIs e CLIs são interfaces que permitem aos aplicativos / desenvolvedores registrar usuários, consultar *chaincode* e emitir transações. Há um conjunto de comandos para interagir com os *chaincode*.

5.6. Aplicação em Redes de Computadores

O consenso sob demanda oferecido pela tecnologia blockchain abre perspectivas para os novos paradigmas e arquiteturas de redes recentemente discutidas na literatura [Xylomenos et al. 2014a, Mijumbi et al. 2016, Blenk et al. 2016]. Esta seção discute como alguns desses paradigmas e arquiteturas podem se beneficiar através de base de dados distribuídos dos blockchains [Sankar et al. 2017, Yin et al. 2018], apresenta aplicações de redes e, por fim, alguns trabalhos em andamento em cada uma das áreas citadas.

5.6.1. SDN e Controladores Distribuídos

Em redes implementadas através do paradigma das Redes Definidas por Software (do inglês, *Software-Defined Networking* – SDN) [Kreutz et al. 2015], o consenso da blockchain pode oferecer suporte à implementação de soluções com controladores distribuídos, que precisam compartilhar uma mesma visão, global e logicamente centralizada, da rede [Sharma et al. 2017a]. Ainda que em estágios iniciais, a literatura atual aponta para trabalhos que sugerem o uso de blockchains na implementação de planos de controle distribuídos e, assim, endereçar questões em aberto relacionadas à disponibilidade de controladores [Blenk et al. 2016, Mijumbi et al. 2016], sobretudo em cenários de IoT [Sharma et al. 2017b]. Um arquitetura SDN baseada no uso de blockchains pode estar menos vulnerável a ataques no plano de controle quando a base de informações sobre as regras de fluxos da rede está distribuída entre múltiplas entidades ao invés de estar associada uma autoridade central. Diversos trabalhos na literatura já apresentam soluções de tolerância a falhas em SDN através de planos de controle distribuídos [Blenk et al. 2016, Mijumbi et al. 2016, Kreutz et al. 2015], porém as mesmas ainda não exploram os benefícios de um sistema distribuído que faz o controle de transações, como na tecnologia blockchains. Numa discussão mais ampla, em [Bozic et al. 2016] os autores apresentam um conjunto de casos de uso de blockchains.

Além dos benefícios da implementação de soluções de tolerância a falhas, o uso de blockchains pode facilitar a instalação de regras de encaminhamento para um conjunto específico de comutadores (*switches*). Tais regras podem ser validadas de forma conjunta, evitando inconsistência ao longo do caminho e garantindo a lógica definida pelo controlador [Agarwal et al. 2014]. Apesar de algumas iniciativas recentes, estas são questões de pesquisa em aberto para as quais o consenso por blockchain pode trazer novas soluções.

5.6.2. Virtualização de Funções de Rede

Em anos recentes, a operação de redes de larga escala têm sido impulsionada na academia e indústria através de Virtualização de Funções de Rede (do inglês, *Network functions Virtualization* – NFV) [Mijumbi et al. 2016]. Trabalhos recentes têm sido propostos na implementação de arquiteturas de NFV, em que funções de rede (*Virtual Network Functions* – VNFs) são gerenciadas, configuradas e migradas de forma segura através de blockchains. Em [Alvarenga et al. 2018], os autores propõem uma plataforma de VNF para garantir imutabilidade e não-repúdio das funções, implantar mecanismos de auditoria a partir do histórico de configurações e prover integridade e consistência de informações armazenadas. Ainda relacionado a NFV, em [Bozic et al. 2017], os autores apresentam uma discussão sobre como blockchains pode ser utilizado para prover a orquestração de

operações de NFV em ambientes de nuvem. Neste trabalho é proposto um protocolo de autenticação que atua em um nó orquestrador de máquinas virtuais. O protocolo faz uso de blockchains para gerenciar requisições nas máquinas virtuais (criação, destruição, cópia, migração) como transações que ficam armazenadas no banco de dados compartilhado.

5.6.3. Redes Centradas na Informação

Diante das limitações impostas pelo IP, novos projetos para a chamada Internet do futuro têm sido frequentemente investigados pela academia nos últimos anos. Dentre esses projetos, a arquitetura de Redes Centradas na Informação (do inglês, *Information-Centric Networking* – ICN) consiste em um novo paradigma no qual, diferentemente da arquitetura IP, apresenta um modelo de consumo/requisição centralizado no detentor do conteúdo, em que o nó não precisa conhecer *a priori* a localização e identidade do produtor de um determinado conteúdo. Em arquiteturas ICN utiliza-se o nome do próprio conteúdo, ao invés do endereço IP, para realizar uma requisição [Jacobson et al. 2009, Xylomenos et al. 2014b, Zhang et al. 2014].

Em trabalhos recentes [Fotiou and Polyzos 2016a, Jin et al. 2017a] foi aplicado o conceito de ICN para prover suporte a tecnologia Blockchain. Apesar de Blockchain, como visto anteriormente, ser uma tecnologia robusta e altamente resiliente, esta ainda está sujeita às limitações impostas pela arquitetura IP atual no que se refere à roteamento, endereçamento e escalabilidade que introduzem problemas na performance da aplicação. Como consequência da ineficiência das redes IP nesse cenário é possível citar as conexões falhas e protocolos impróprios que contribuem para a geração de forks no blockchain [Jin et al. 2017b]. Outro problema apresentado por [Jin et al. 2017b] destaca que apesar da natureza distribuída do blockchain, o estabelecimento de conexões entre nós homogêneos é algo complexo. Para entender melhor esse problema segue um breve exemplo. Na descoberta de vizinhos no bitcoin, alguns endereços IP de possíveis voluntários são recomendados como opções de conexões para os usuários. Esses nós voluntários atuam como uma infraestrutura para ajudar usuários normais a se comunicarem. Entretanto, essa abordagem aumenta a probabilidade de existir supernodes e com isso fatores de insegurança como fraudes e pontos únicos de falha.

Já em [Fotiou and Polyzos 2016a] é proposto o inverso, no qual a tecnologia blockchain é utilizada para adicionar mais segurança a ICN. Essa camada adicional de segurança está presente na distribuição de conteúdos que, como apontado pelos autores, em ICN a informação é identificada pelo seu próprio nome, portanto, é natural considerar o nome como uma primitiva básica de segurança. Isso abre uma gama de possíveis superfícies de ataque uma vez que, diferente de das chaves públicas RSA, o nome está em texto claro e é humanamente legível o que o torna fácil de memorizar e distribuí-lo. Os nomes também são previsíveis tornando possível que um atacante crie um conteúdo falso, que ainda não foi concebido por seu produtor original, e o propague na rede. Por fim, os nomes podem ser hierárquicos o que expõe organizações e relações de negócios. Com base nesses problemas, os autores em [Fotiou and Polyzos 2016b] utilizaram o blockchain para permitir que qualquer “nó inscrito” possa checar a validade de um conteúdo com base em seu nome. No sistema proposto cada entidade compartilha informações públicas, a partir disso, o inscrito que conhecer essas informações pode verificar a assinatura da informação independente do provedor.

Redes Centradas na Informação provêm novas arquiteturas de comunicação da Internet em que projetistas podem se beneficiar do consenso oferecido pela tecnologia blockchains. Por exemplo, a proposta BlockNDN [Jin et al. 2017a], propõe a implementação de uma rede de dados nomeados a partir do uso de blockchains. Os autores propõem partes de uma arquitetura ICN, incluindo um novo esquema de nomeação baseada no uso de blocos e mecanismos de encaminhamento de interesses.

Além de novas arquiteturas, modelos de reputação podem ser construídos de forma a garantir apenas a entrada de produtores de conteúdos confiáveis na rede, um problema ainda com muitas questões em aberto e investigado na literatura recente [Salah and Strufe 2016]. Ao garantir a entrada apenas de nós confiáveis na rede, modelos de segurança podem ser criados para evitar ataques, tais como de negação de serviço [Dai et al. 2013, Compagno et al. 2013, Wang et al. 2014, Seixas et al. 2017]. Além disso, tais sistemas podem ajudar na manutenção de cache confiável entre os nós [Jin et al. 2017b, Tourani et al. 2017]. Ainda relacionado a segurança em ICN, o consenso por blockchains pode ser utilizado como estratégia de mitigação de ataques de negação de serviço através de técnicas de prova de trabalho nos clientes [Li and Bi 2014].

5.6.4. Segurança de Redes

No que diz respeito à segurança de redes, a aplicabilidade de blockchains começa a ser vista na implantação de modelos de confiança [Yang et al. 2017, Sharma et al. 2017a] para, dentre outras coisas, possibilitar autenticação descentralizada e anônima de dispositivos móveis em redes C-RAN [Yang et al. 2017] e a validação de anúncios de recursos de numeração (i.e., IPv4/v6, ASN) no roteamento BGP [Paillisse et al. 2017]. Nesta seção, algumas dessas aplicações serão apresentadas e discutidas.

5.6.4.1. Autenticação Anônima e Confiável Baseada em Blockchain para C-RAN

A arquitetura descentralizada e segura da tecnologia de blockchain pode beneficiar a confiabilidade na comunicação e armazenamento distribuído, agregando características-chaves como persistência, anonimidade e auditoria. Tais características mostram-se promissoras, por exemplo, no contexto de autenticação de termináveis móveis 5G em uma Rede de Acesso de Rádio em Nuvem (do inglês, *Cloud Radio Access Network* - C-RAN), cujo funcionamento originalmente baseia-se em uma autenticação centralizada no núcleo da rede móvel, gerando impactos na operação e no consumo dos recursos de rede. A partir do uso de um modelo de autenticação baseado em blockchain, é possível melhorar a a confiabilidade no acesso de clientes e com baixo custo de rede nesse ambiente.

[Yang et al. 2017] propõe uma estratégia de autenticação confiável baseada em blockchain (BTA, do inglês *Blockchain-based Trusted Authentication*) em C-RAN para 5G, incorporando um esquema de acesso anônimo e confiável baseado em blockchain. Para isso, os autores utilizam uma arquitetura integrada do controlador SDN com uma plataforma de blockchain. O papel do controlador SDN é realizar a orquestração da frequência de rádio, do espectro óptico e dos recursos de processamento das unidades de banda base (BBU). Já a blockchain consiste essencialmente de um livro-razão público, onde as identificações de acesso confirmadas são armazenadas na cadeia de blocos.

Na arquitetura proposta, o processo de autenticação e acesso anônimo de clientes

é executado de forma tripartite: pelo dispositivo cliente (*device*), pelo fabricante (*manufacturer*) e pelo operador (*operator*). O processo funciona conforme descrito a seguir. Para garantir a anonimidade do dispositivo, o fabricante primeiro cria e envia para o operador uma chave pública de proveniência comprovada. Ao conectar na rede, um dispositivo cliente requisita informações criptográficas do fabricante, utilizando técnicas de assinatura às cegas. Ao receber o material criptográfico do fabricante, o dispositivo utiliza-o para gerar sua chave secreta independente da identidade, que por sua vez é enviada ao operador para verificação de proveniência. A partir daí, o controlador SDN faz o setup do caminho entre o dispositivo e o BBU, considerando informações de utilização do rádio, dos canais ópticos e do BBU. Finalmente o dispositivo envia uma requisição de acesso para o operador, usando técnicas de desafio-resposta, e caso a validação seja bem sucedida, o cliente recebe acesso à C-RAN e é registrado na blockchain.

5.6.4.2. Validação de Recursos de Numeração de Rede (IPv4/v6)

O roteamento na Internet é baseado, principalmente, no protocolo BGP (do inglês *Border Gateway Protocol*), cujo funcionamento consiste, em linhas gerais, no anúncio de prefixos IP entre Sistemas Autônomos (AS) e em um conjunto de políticas de roteamento que influenciam na escolha dos caminhos. O BGP é comumente referenciado como a “cola que une a Internet”, habilitando a comunicação entre grandes redes operadas por diferentes organizações. Ao passo que o BGP desempenha um papel fundamental na comunicação da Internet, ele incrivelmente se mantém vulnerável à uma série de ataques. Grande parte dessas vulnerabilidades derivam do excesso de confiança entre os pares e da falta de mecanismos de checagem das informações trocadas. Diversas soluções foram propostas ao longo dos anos, geralmente baseadas em padrões de Infraestrutura de Chaves Públicas (ICP), que inclusive deram origem aos mecanismos atuais de segurança: RPKI (Resource Public Key Infrastructure) e BGP-SEC (BGP Security Extension). Estes mecanismos tratam-se de um conjunto de protocolos, padrões e sistemas que melhoram a segurança dos recursos de numeração da Internet como os endereços IPv4 e IPv6, possibilitando a verificação do direito de uso. Embora promissores, esses mecanismos ainda possuem baixa adoção pelos provedores. Por outro lado, a tecnologia de blockchain pode beneficiar ou complementar alguns dos desafios desse modelo centralizado das ICPs e incorporar melhorias para a segurança no roteamento interdomínio.

[Paillisse et al. 2017] analisa como a tecnologia de blockchain pode ser utilizada para adicionar segurança ao processo de alocação, delegação e mapeamento dos recursos de numeração (i.e. endereços IP), provendo um grau de segurança similar à arquitetura tradicional baseada em ICP. Em linhas gerais, criptomoedas e endereços IP compartilham características fundamentais que motivam o uso de blockchain para proteção: endereços IP são alocados de forma não ambígua para entidades (AS's); endereços IP podem ser transferidos entre entidades; um bloco de rede não pode ser alocado para duas entidades ao mesmo tempo; os blocos podem ser divididos em sub-blocos até um certo limite. A partir dessas propriedades em comum, é possível conceber uma blockchain que permite aos seus participantes (AS's) delegarem blocos de endereços IP de forma similar às transações de criptomoedas. Por exemplo, a IANA poderia escrever uma transação alocando endereços para os RIRs (do inglês, *Regional Internet Registries*), como o LACNIC por exemplo, que, por sua vez, poderiam lançar mão de uma outra transação alocando um sub-bloco para um

Registro Nacional de Internet (e.g. Registro.br) ou para um provedor diretamente, e assim sucessivamente. Em contrapartida, utilizar a tecnologia de blockchain nesse contexto adiciona algumas vantagens:

- Descentralização: não haveria uma entidade centralizadora controlando o blockchain, ele é compartilhado entre todos os participantes;
- Sem necessidade de certificados, ACs (Autoridades Certificadoras) e CRLs (do inglês, *Certificate Revocation List*);
- Resistência a censura: uma vez que o controle das transações depende do detentor da chave privada, a revocação ou mudança de uma alocação de IP sem a permissão do proprietário legítimo envolveria o comprometimento da chave privada do proprietário. Mesmo que a chave privada da entidade hierarquicamente superior na alocação do bloco IP seja comprometida, a confiabilidade da transação atual é preservada, o que não é válido no caso de comprometimento da chave privada de uma AC ou de ACs maliciosas no cenário com ICP.
- Auditoria: alocações e delegações podem ser rastreadas na blockchain para determinar a origem do proprietário legítimo de uma transação.

Um aspecto importante na modelagem desse problema em uma blockchain é a escolha do algoritmo de consenso. Em particular, algoritmos de prova de trabalho (PoW) apresentam algumas desvantagens, principalmente pelo fato de que o incentivo para um participante legítimo da blockchain de endereços IPs não é diretamente ligado ao seu poder computacional, ou seja, um atacante poderia investir em adquirir alto poder computacional para reescrever a blockchain com dados falsos. Já com os algoritmos de prova de posse (PoS) a capacidade de alteração da blockchain permaneceria no próprio proprietário do bloco. Tipicamente o AS que é proprietário do bloco de endereços possui negócios na Internet associados àquele bloco, de tal forma que ele possui claros incentivos em manter uma operação correta e segura das suas transações.

5.7. Contratos Inteligentes na Internet das Coisas

A Internet das Coisas (IoT), ao mesmo tempo em que possibilita uma gama de aplicações interessantes (cidades inteligentes, saúde, casas inteligentes, etc.), apresenta desafios, sobretudo em nível de segurança e privacidade, dado o grande número de dados gerados, compartilhados e a exposição a que os mesmos estão sujeitos. Nesse sentido, o casamento da IoT com a blockchain é celebrado por fornecer uma nova camada computacional para compartilhamento e análise segura dos dados, com garantias de privacidade. Assim, essa camada pode ser utilizada para autenticar, autorizar, controlar e auditar os dados gerados pelos objetos inteligentes [Christidis and Devetsikiotis 2016].

5.7.1. Introdução à IoT

A Internet das Coisas (do inglês *IoT*) nasceu do avanço de áreas como microeletrônica, sistemas embarcados, comunicação e sensoriamento [Santos et al. 2006] e hoje é uma realidade perene do cotidiano. A IoT é composta por Objetos Inteligentes (OI) capazes de enviar informações sobre si mesmo, sobre o lugar em que estão inseridos ou agir em determinadas situações, trazendo para o mundo real uma dinâmica nunca antes experimentada.

Esta definição gera implicações e cria conceitos inovadores como *weareable*, casas, carros e cidades inteligentes, *smart grids*, indústria 4.0, etc. Todos possuem como base um ou mais dispositivos IoT, os quais realizam quatro funções básicas: *sensoriamento*, *reação*, *coleta e comunicação* [Bashir 2017].

Os sistemas de IoT possuem, geralmente, cinco camadas: Aplicação, Serviços, Rede, Dispositivo e Física. Cada uma possui funções e componentes definidos. A primeira camada possui as aplicações de caráter financeiro, transporte, saúde, segurança, ambiental, etc.. Na camada de serviços estão tarefas como processamento de dados, gerenciamento de segurança e de fluxo de dados. As camadas de rede e dispositivos, tratam, respectivamente, da transmissão dos dados e dos sensores, atuadores e OIs. Por último, na camada física estão as pessoas, os carros, os animais, as salas e tudo mais que precise ser monitorado e controlado.

Como exemplo de tais sistemas, considere um cenário típico da cadeia de suprimentos (I) composta por: (a) Indústria; (b) Transporte até porto/aeroporto; (c) Transporte até um Centro de Distribuição; (d) Entrega em uma Loja e (e) Venda para um cliente. Este processo envolve diversos agentes desde (a) até (e). Cada um destes, possui um sistema e um banco de dados proprietários, onde guardam informações sobre os produtos. Para manter estes dados atualizados, uma equipe de pessoas trabalha, alimentando o sistema. Com o uso de IoT, todo processo é simplificado e os próprios OIs postam as informações sobre o seu status, local, data, etc. Adicionalmente, algumas categorias de OIs podem necessitar de atualização de software ou firmware (II). O próprio OI pode iniciar este processo ou ser acionado remotamente pelo fabricante para que receba a nova versão.

Esta é uma situação hipotética onde as próprias máquinas podem cuidar de si mesmas. No entanto, questões como confiança, segurança, transparência, auditabilidade, etc. estão carentes. Por consequência, atualmente, muitos autores defendem a inserção, nos sistemas de IoT, da solução de blockchain e de contratos inteligentes como forma de suprir estas lacunas [Brody and Pureswaran 2014].

No caso em tela, a blockchain proporciona um avanço. Cada um dos atores do cenário (I) e (II) enviam/recebem mensagens de forma segura, auditável e irrefutável. Os contratos convencionais são transformados em contratos inteligentes, predeterminando as condições e permitindo aos participantes autorizados o acesso às informações e o monitoramento eletrônico das transações. Especialmente no cenário (II), o fabricante pode implementar um SC que permita armazenar o hash da última atualização do firmware na rede. Os OIs ou possuem gravados ou descobrem por mecanismos adequados o endereço deste contrato Inteligente e podem verificar se há um novo firmware disponível, solicitando-o por meio de um novo hash, a um sistema de arquivos distribuído tipo P2P [Christidis and Devetsikiotis 2016].

5.7.2. Contratos Inteligentes

O termo *contrato inteligente* (*smart contract*) (SC) foi formalizado por Nick Szabo na década de 90, e significa: “um protocolo de transação informatizado que executa os termos de um contrato. Os objetivos gerais são satisfazer condições contratuais comuns (como condições de pagamento, ônus, confidencialidade e até mesmo cumprimento), minimizar as exceções maliciosas, acidentais e a necessidade de intermediários confiáveis. Os obje-

```

1  pragma solidity ^0.4.0;
2  contract TemperaturaIdeal{
3
4      int public temperatura;
5      address sensor;
6      address termostato;
7
8      event Instruction(address device, string instruction);
9      event reward(address device, string instruction);
10
11     function IoT (address sensor, address reg){
12         sensor = _sensor;
13         termostato=reg
14     }
15
16     function atualizaTemperatura (int temperatura){
17         if(msg.sender != sensor) throw;
18         temperatura = temp;
19         if(temperatura >=25)
20             Instruction(termostato, "Violação");
21         else reward(termostato,"Compliant")
22     }
23 }

```

Figure 5.8. Contrato Inteligente para Verificação de Temperatura.

tivos econômicos relacionados incluem redução de perdas por fraude, arbitragens e custos de execução e outros custos de transação ” [Szabo 1997].

Na blockchain, os SCs tomam a forma de scripts, armazenados com endereçamento exclusivo na própria blockchain. Aciona-se um contrato inteligente endereçando uma transação para ele. Em seguida, ele é executado de maneira independente e automática, conforme a prescrição, em todos os nós da rede, de acordo com os dados incluídos na transação. Isto significa que cada nó, habilitado por contrato inteligente, está executando uma máquina virtual (MV) e que a rede blockchain atua como uma MV distribuída. [Christidis and Devetsikiotis 2016]. Na IoT, a principal vantagem do uso dos SC deve-se ao gerenciamento das interações entre as entidades inteligentes envolvidas, podendo realizar uma das 4 funções básicas: sensoriamento, reação, coleta e comunicação.

A Figura 5.8 exemplifica um SC para o Ethereum, usando o Solidity (linguagem de programação para SC, similar ao JavaScript). Neste exemplo, quando uma certa temperatura não for mantida, é gerada uma violação. Se a temperatura estiver no intervalo esperado, o nó é recompensado na forma de moedas virtuais.

5.7.3. Aplicações na IoT

Há diversas aplicações possíveis no casamento da IoT com a Blockchain. Governo eletrônico e voto eletrônico, segurança interna, controle de fronteiras, serviços cartoriais, judiciais e fazendários, além de identidade digital são alguns pontos que podem ser atendidos pelos SCs [Ølne et al. 2017]. Infra-estrutura específica para a área de saúde e registros e acompanhamento médico em processos que exigem, dentre outras coisas, transparência, auditoria e controle são casos em que os SCs e IoT atuam [Salahuddin et al. 2017, Ekblaw et al. 2016]. Aplicações na área de finanças podem ser encontradas em [Peters and Panayi 2016] e para artes, música, cópias digitais e material multimídia em [Bhowmik and Feng 2017]. Nos serviços e infra-estrutura computacional, por exemplo, pode-se cobrar pelo uso de chamadas de API's ou uso de espaço em disco [Christidis and Devetsikiotis 2016]. A camada da blockchain pode possibilitar funcionalidades como os micro-pagamentos entre dispositivos digitalmente aprimorados, através de criptografia ultra-leve e de SCs. Na

área de rastreabilidade de produtos, em geral, as diversas partes envolvidas possuem bancos de dados próprios para controlar a mobilidade de insumos; com o uso da blockchain surge um banco de dados compartilhado, com recursos criptográficos e auditável.

5.7.4. Segurança em Contratos Inteligentes

Os SCs interpretam o código de maneira objetiva e implacável - "O Código é a lei". No entanto, um conjunto de contratos inteligentes (DAO) escritos para uma plataforma de investimentos continha um erro no código e foi alvo de um ataque em junho de 2006. Isto desviou cerca de 50 milhões de dólares e obrigou o Ethereum a realizar um "*hard fork*" para promover uma recuperação, quebrando este paradigma [Bashir 2017]. Mas, fazer isto não é uma operação trivial, portanto, as principais ameaças aos contratos inteligentes precisam ser estudadas e minimizadas.

As principais fragilidades dos SCs, com ênfase na plataforma Ethereum, podem ser agrupadas em três níveis: Linguagem Solidity, Máquina Virtual Ethereum (MVE) e Blockchain). Todas elas podem ser exploradas para ataques e roubos de dados e dinheiro dos contratos [Atzei et al. 2017]. Para melhor compreensão, são introduzidos os conceitos de MVE e Gas. A MVE é o nome dado ao ambiente que a plataforma Ethereum disponibiliza para a execução das aplicações e contratos inteligentes, garantindo que eles não tenham acesso aos estados uns dos outros e que a comunicação possa ser estabelecida sem interferências prejudiciais. O Gas é uma forma de desacoplar o custo das transações no Ethereum do câmbio flutuante da criptomoeda Éter, estabelecendo um custo para cada trabalho computacional.

De acordo com [Atzei et al. 2017], as principais ameaças existentes em relação ao uso do Solidity são: (a) **Primitivas call, send e delegatecall** - Usadas no Solidity para invocar funções e transferir dinheiro, podem ter efeito colateral de invocar a função fallback do destinatário; (b) **Exceptions** - Há diversas situações onde uma *exception* pode ser disparada e o Solidity não responde de modo padrão a todas elas; (c) **Gasless** - O uso da função *send* para transferir ether de um contrato com o saldo de *gas* zerado; (d) **Declaração de atributos** - O compilador Solidity pode detectar alguns erros, como por exemplo, atribuir um valor inteiro a uma variável do tipo string; (e) **Re-entrada** - A atomicidade e a sequencialidade das transações podem induzir os programadores a acreditar que, quando uma função não recursiva é invocada, ela não pode ser reinserida antes de sua finalização. No entanto, isso nem sempre é o caso, porque o mecanismo de fallback pode permitir que um invasor entre novamente na função do chamador. Isso pode resultar em comportamentos inesperados e possivelmente também em loops de invocações que eventualmente consomem todo o *gas* e (f) **Informações Sensíveis** - Os campos nos contratos podem ser públicos ou privados. Declarar um campo como privado não garante seu sigilo. Isso porque, para definir o valor de um campo, os usuários devem enviar uma transação adequada aos mineradores, que a publicarão no blockchain. Como o blockchain é público, todos podem inspecionar o conteúdo da transação e inferir o novo valor do campo. Portanto, o SC deve usar mecanismos de criptografia adequados.

Em relação à MVE, tem-se os seguintes riscos: (a) **Bugs imutáveis** - Um contrato após publicado, não pode ser modificado. O comportamento em tempo de execução é garantido pelos protocolos de consenso, portanto, caso haja um bug, ele vai permanecer;

(b) **Perda de ethers na transferência** - Ao enviar ethers de uma SC para outro, o endereço pode não estar mais associado ao SC de destino ou ao usuário, fazendo com que os ethers se percam para sempre e (c) **Tamanho da pilha de chamadas** - Cada vez que um contrato invoca outro contrato, a pilha de chamadas associada à transação cresce um quadro. O limite é 1024 quadros. Quando esse limite é atingido, uma outra chamada lançará uma exceção. Sem o tratamento adequado, um adversário poderá ter sucesso em um ataque.

Por fim, as ameaças para os SCs em relação à Blockchain: (a) **Estado do contrato** - O valor dos campos e saldo determinam o estado do SC. Um usuário, ao invocar algum SC, não tem certeza sob seu estado, pois outras transações podem modificá-lo ou pode ter ocorrido um *fork*. Em alguns casos, isto pode gerar vulnerabilidades e proporcionar o roubo de ethers; (b) **Geração aleatória** - A execução dos bytecodes da MVE é determinística e na ausência de mau comportamento, todos os mineradores que executam uma transação terão os mesmos resultados. No entanto, alguns contratos, como por exemplo, loterias, jogos, etc., geram números pseudo-aleatórios com a mesma semente para todos os mineradores. Isto permite que todos tenham a mesma visão da Blockchain, proporcionando a um minerador malicioso influenciar a rede; (c) **Restrições de Tempo** - Diversos aplicativos usam restrições temporais para determinar ações permitidas ou obrigatórias no instante t de tempo usando timestamps do bloco, que são acordados por todos os mineradores. Os SCs podem recuperar o registro de data e hora em que o bloco foi extraído e todas as transações dentro de um bloco compartilham o mesmo timestamp. O minerador que cria um novo bloco pode escolher o timestamp a ser usado com um certo grau de arbitrariedade, ou seja, uma tolerância de 900 segundos. Caso ele detenha uma participação em um contrato, poderá obter uma vantagem, ao escolher um registro de data e hora adequado para o bloco que está explorando.

5.8. Aplicação em Névoa

De maneira similar a blockchain, um ambiente de névoa provê uma plataforma virtualizada em uma escala amplamente distribuída que oferece serviços entre os dispositivos finais e os centros de dados convencionais da computação em nuvem. Nesta seção, vamos introduzir o ambiente de névoa e as vantagens tecnológicas com a integração desses dois paradigmas. Como exemplos de aplicação, serão apresentados trabalhos em andamento sobre arquiteturas de névoa que empregam a blockchain na implementação de seus modelos e de suas funcionalidades.

5.8.1. Ambientes de Computação em Névoa

Um modelo baseado em nuvem tornou-se a abordagem padrão para a IoT. Entretanto, propostas recentes estão promovendo uma grande mudança em direção a uma arquitetura descentralizada [Coutinho et al. 2016]. A computação em névoa (*fog computing*) e borda (*edge computing*) estendem os serviços de nuvem para as extremidades da rede buscando superar as limitações das arquiteturas IoT. Sua motivação está apoiada em aplicações que exigem latência mínima e o processamento de uma grande quantidade de dados, o que é difícil de se realizar através da rede.

A Figura 5.9 apresenta a arquitetura horizontal do sistema de névoa. Sua infraestrutura envolve a execução de serviços e aplicativos entre os dispositivos de borda

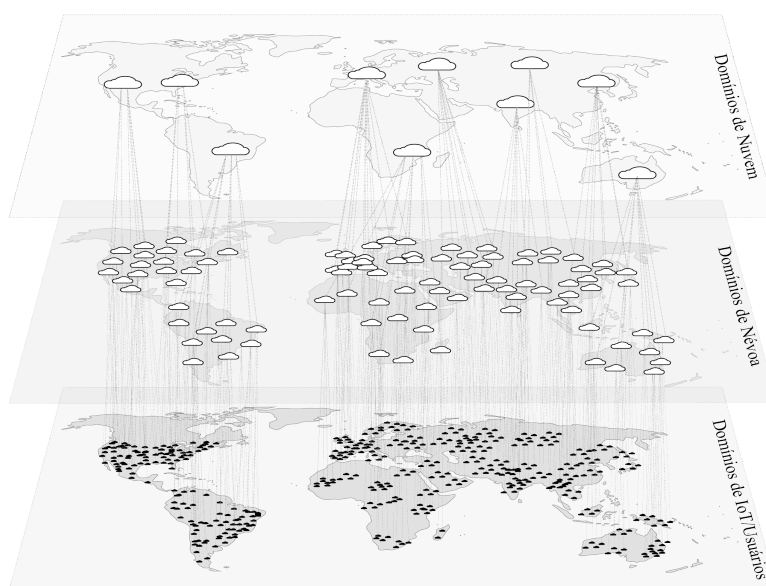


Figure 5.9. Arquitetura Horizontal do Sistema de Névoa.

e a nuvem. Eles são distribuídos hierarquicamente em uma infraestrutura de rede com pelo menos três camadas. Os componentes que podem fornecer recursos de processamento e armazenamento aos dispositivos de IoT são chamados nós de névoa. Eles ajudam no processamento de dados de aplicações de IoT e são responsáveis por filtrar e enviar periodicamente informações para a nuvem.

As aplicações podem usar recursos tanto da nuvem quanto da névoa para fornecer soluções inteligentes para usuários na borda da rede. O software de gerenciamento que coordena globalmente a infraestrutura de névoa é organizado em diferentes serviços que trabalham de forma distribuída e integrada. Seu principal objetivo é fornecer níveis aceitáveis de latência enquanto busca otimizar o desempenho das aplicações através de soluções eficientes que distribuem demandas de execução de tarefas em nós de névoa ótimos, bem localizados e disponíveis.

Espera-se que os ambientes de névoa suportem milhões de dispositivos de IoT e de usuários finais. Ambientes de névoa podem envolver um grande número de nós de névoa e aplicativos localizados em diferentes domínios de rede. Os sistemas de névoa precisam ser seguros, operacionais e escaláveis de forma elástica, ou seja, capazes de diminuir ou aumentar o número de recursos alocados quando necessário. Para atingir os objetivos das redes de IoT, faz-se necessária a existência de soluções autônomas e descentralizadas.

5.8.2. Integração Entre Sistemas de Névoa e Blockchain

A computação em névoa e a tecnologia blockchain são soluções independentes, voltadas à cenários de aplicação específicos. Entretanto, esses ambientes apresentam características fundamentais semelhantes: (i) são redes distribuídas em larga escala, que podem ser formadas por milhares de nós; (ii) precisam oferecer conectividade generalizada entre dispositivos geograficamente dispersos em uma região, país ou no mundo inteiro; (iii) os nós que formam a rede devem possuir recursos de processamento e de armazenamento para suporte à serviços e aplicações. (iv) são ambientes heterogêneos, onde os nós da rede podem apresentar diferentes capacidades, interfaces e configurações de hardware.

Existem diferentes motivações para aplicação da tecnologia blockchain no projeto e no desenvolvimento de sistemas de computação em névoa. Algumas das motivações identificadas em [Sharma et al. 2017a] são indicadas a seguir:

- *Descentralização*: Criando confiança sem a necessidade de uma entidade intermediária confiável, acredita-se que a blockchain promoverá a revisão dos sistemas de nuvem atuais e facilitará a implementação adequada de sistemas em névoa através de soluções autônomas e descentralizadas;
- *Acesso sobre demanda*: a blockchain pode facilitar o acesso sob demanda e a cobrança no uso de recursos utilizados. Por exemplo, executando o algoritmo de recursos sob demanda em um contrato inteligente, o pagamento ocorrerá automaticamente após a conclusão do serviço solicitado;
- *Redundância e disponibilidade*: de forma análoga aos sistemas P2P, a blockchain permite construir sistemas de névoa onde os dados e serviços da aplicação são distribuídos em dezenas de nós distintos, de maneira inteligente e bem localizada, tornando extremamente difícil causar interrupções significativas;
- *Privacidade e Integridade*: na tecnologia blockchain cada usuário gerencia suas próprias chaves e cada nó de bloco armazena apenas fragmentos criptografados de dados do usuário. É possível oferecer privacidade aos usuários sem que os terceiros envolvidos tenham acesso e controle dos dados;
- *Qualidade de Serviço*: usando a tecnologia blockchain, é possível oferecer rastreabilidade do uso de recursos visando a verificação adequada do contrato de nível de serviço tanto pelo cliente quanto pelo provedor de serviços;
- *Cooperação e Incentivos*: Os ambientes de névoa com suporte a blockchain podem fomentar novos modelos que ofereçam incentivos aos seus usuários e cooperação entre provedores. Neste ambiente, provedores independentes podem cooperar entre si de forma transparente para construir uma infraestrutura de TI escalável e oferecer sua capacidade computacional excedente aos usuários, enquanto os pagamentos são feitos através da blockchain;
- *Baixo custo*: Fazendo um paralelo entre as infraestruturas financeiras e as de computação em névoa, a tecnologia blockchain pode ser empregada para substituir os sistemas de nuvem existentes por sistemas que cobram taxas menores pelos serviços. Isso é possível devido à eficiência - sobre cada nó da névoa/blockchain incide custos que, quando somados, são menores do que os custos dos provedores de nuvem atuais, salientando que esses provedores ainda tem de realizar lucros consideráveis para seus investidores.

5.8.3. Soluções em Névoa Baseadas em Blockchain

Pesquisas atuais que propõem a integração dessas duas tecnologias tem como base às vantagens mútuas obtidas pela combinação de suas funcionalidades. Dependendo do escopo da pesquisa, as propostas podem ter como objetivo geral: o desenvolvimento da plataforma de névoa; ou o emprego da névoa/blockchain para aperfeiçoar modelos, sistemas

ou aplicações distribuídas existentes. A seguir, descrevemos os aspectos principais da aplicação da tecnologia blockchain na arquitetura de nuvem/névoa proposta por [Sharma et al. 2017a]. Neste trabalho, é empregado um modelo aberto na construção de um ambiente formado por diferentes provedores de serviço. Este modelo permite o amplo uso de uma infraestrutura de armazenamento distribuída pelos seus usuários.

Modelo de Negócio - O modelo proposto em [Sharma et al. 2017a] reúne dados de fornecedores e consumidores de recursos, consistindo nas quatro etapas a seguir: (i) seleção de fornecedores de recursos, (ii) prestação de serviços, (iii) registro de transações e pagamento. Na primeira etapa, o usuário seleciona o fornecedor dos recursos a partir de uma pesquisa sobre os provedores de serviço disponíveis na névoa baseada em blockchain. Após o processo de alocação, o provedor fornece os recursos requisitados para o usuário. Após fornecer os recursos, o provedor registra a transação na blockchain compartilhada entre todos os provedores de serviço de mesmo nível. Por fim, o usuário recompensa o provedor pelos serviços prestados e utilização dos recursos.

Protocolo de Consenso - De maneira diferente de uma blockchain convencional como o Bitcoin, que depende de um protocolo baseado em PoW para validar suas transações, a névoa baseada em blockchain proposta em [Sharma et al. 2017a] utiliza um protocolo de consenso nomeado como prova de serviço (*Proof-of-Service*) para validar as contribuições e permitir que as transações associadas possam ser incluídas na blockchain. Uma contribuição é uma ação que ocorre fora da blockchain como o processamento de uma computação, a transferência de um arquivo ou o fornecimento de dados. Isso levará à ocorrência de transações entre os membros da rede. Este protocolo usa a técnica de 2-hop blockchain proposta por [Duong et al. 2016] que combina os mecanismos de PoW e PoS. A segurança desta blockchain depende se o participante legítimo controla a maior parte dos recursos totais, que incluem tanto as participações quanto o poder de computação.

Alocação de Recursos: No projeto de sistemas de névoa, o algoritmo de alocação de recursos é um elemento essencial para garantir a qualidade dos serviços prestados [Coutinho et al. 2016]. O modelo proposto em [Sharma et al. 2017a] vincula uma solicitação de recurso a uma oferta de recurso utilizando contratos inteligentes. Um contrato de alocação descreve os requisitos para executar a tarefa como o ambiente de virtualização esperado, o tempo de execução da GPU, o espaço em disco necessário, etc. Um contrato de alocação pode estar relacionado à diferentes algoritmos de alocação de recursos.

Existem trabalhos que não possuem como objetivo geral o projeto de uma plataforma de névoa, mas que utilizam o modelo de névoa para propor sistemas inovadores baseados em blockchain. Por exemplo, em [Huang et al. 2018] o objetivo do trabalho é aprimorar o sistema tradicional de caixa eletrônico que requer uma autoridade confiável, ou seja, um banco para gerar o *token* de pagamento. Os usuários do sistema (*outsourcers*) podem fazer diretamente uma transação para os nós de névoa (*workers*). Nesta proposta, é apresentado um esquema de pagamento justo baseado em bitcoin entre *outsourcers* e *workers* que ofertam seus recursos para tarefas de computação distribuídas. No protocolo proposto, os *outsourcers* e os *workers* não confiam uns nos outros. Foram utilizados contratos inteligentes para garantir que, independente do quão malicioso seja o comportamento de um *outsourcer*, um *worker* honesto que executa a tarefa sempre receberá o pagamento. Diferentemente de trabalhos anteriores, não existe uma entidade centrali-

zadora envolvida no pagamento. O protocolo proposto foi mais eficiente em termos de processamento e comunicação em relação às soluções vigentes analisadas.

5.9. Desafios e Perspectivas

Os desafios para a consolidação e melhoria da blockchain, tanto científicos quanto tecnológicos, são inúmeros e diversos. Os principais envolvem sustentabilidade, escalabilidade [Eyal et al. 2016], desempenho [Dinh et al. 2017b], adaptabilidade, armazenamento [Dinh et al. 2017a], segurança, privacidade, regulação, ordenação das transações [Natoli and Gramoli 2016], etc. Muitos desafios foram apresentados ao longo de todo o texto. Nessa seção, faremos um breve panorama de alguns deles, considerando, sobretudo, o serviço de coordenação e consenso.

Na blockchain pública os maiores desafios são a sustentabilidade, escalabilidade e desempenho da rede. Alguns trabalhos interessantes avaliam o impacto desses fatores para a adoção abrangente da tecnologia [Dinh et al. 2017b, Vukolic 2015, Croman et al. 2016]. Na Bitcoin, a sustentabilidade é crítica devido ao alto gasto de energia, com exigência de poder computacional cada vez mais crescente. As propostas de consenso PoX trazem uma resposta a esse ponto. A escalabilidade está diretamente relacionada ao desempenho, no que diz respeito à latência (tempo de confirmação de uma transação) e vazão (*throughput*) (taxa de confirmação de transações no tempo). A vazão depende diretamente do tamanho máximo do bloco e do intervalo de tempo entre blocos. Na Bitcoin, esse valor é ainda muito baixo, comparativamente às demandas de negócios; atualmente, a latência é de 10 min. e a vazão máxima é de 3 – 7 tx/seg, enquanto sistemas de transações financeiros, como operadoras de cartão de crédito, exigem para além de 10^3 tx/seg. Já a Ethereum apresenta melhor vazão (blocos são decididos a cada 12 – 15 seg), porém o aumento da frequência de decisões proporciona maior número de *forks* na rede.

Alguns trabalhos recentes foram propostos para mitigar o baixo desempenho da Bitcoin. O *Bitcoin-NG* [Eyal et al. 2016] advoga a realização do consenso em duas fases distintas: (i) eleição do líder e (ii) serialização de transações, de tal forma que, durante um período estabelecido (*epoch*), o líder teria direito a serializar transações, até que um novo líder seja selecionado; as análises de desempenho apresentadas são favoráveis à estratégia. O *Bitcoin lightning network* [Poon and Dryja 2016] propõe a realização de uma parte das transações off-blockchain. As transações ocorreriam em rede de canais de micro-pagamento e a transferência de valor passaria a ser realizada off-line.

A blockchain privada/federada tem excelentes taxas de desempenho: a latência é instantânea, proporcional à latência da rede, enquanto a vazão de confirmação é da ordem de 10^4 tx/seg. Entretanto, diferentemente da PoW, os protocolos de consenso BFT, adequados para as redes controladas não escalam com o aumento do número de nós; seu bom desempenho fica restrito a um conjunto limitado de 20 – 30 nós.

[Vukolić 2017] destaca algumas limitações de projeto da blockchain privada, que permitem aperfeiçoamento da tecnologia: (i) *Execução sequencial*: a execução em série das transações e contratos inteligentes, após a realização do consenso, é bastante limitante, contribuindo negativamente para a melhoria da vazão. Pesquisas nas áreas de computação paralela e fragmentação de dados (*data sharding*) são um caminho para superar esse aspecto; (ii) *Execução não-determinística*: para preservação da consistência do *ledger*, é

importante que a execução da máquina de estados replicada ativamente seja determinística, de tal maneira a forçar com que todas as réplicas executem exatamente os mesmos passos, na mesma ordem. Ocorre que os sistemas de execução e as próprias linguagens de programação de propósito geral incorporam elementos de não determinismo. O desafio consiste em conciliar o não determinismo inerente aos sistemas com um modelo de falhas e confiança realista para as aplicações na blockchain; (iii) *Execução em todos os nós da rede*: o padrão de execução dos contratos inteligentes em todos os nós da rede é o mais comum, sobretudo em redes abertas. Mas, em redes controladas esse padrão pode ser enfraquecido, de tal forma a restringir a execução dos contratos a apenas um subconjunto de nós. A definição de um modelo adequado para estabelecimento do conjunto de nós, da sua quantidade e tipo de interação para essa execução é um desafio interessante de projeto; (iv) *Modelo de confiança flexível*: deve-se flexibilizar o modelo, de tal forma a separar o modelo de execução do consenso do modelo de execução dos contratos inteligentes. As restrições impostas para a camada inferior de coordenação e consenso (como a limitação de $f < n/3$ nós maliciosos na rede) não precisam ser as mesmas para execução dos contratos inteligentes (cujas aplicações podem exigir limitações diferenciadas). (v) *Consenso hard-coded*: a grande totalidade das plataformas de blockchain adota consensos cujo código é imbricado de tal forma que a sua manutenção, evolução e substituição é muito dificultada. Como não há protocolos únicos para todas as aplicações, o desejado para uma plataforma de blockchain genérica seria prover um consenso modular, passível de substituição, tal como se observa no projeto Hyperledger.

A aplicação da tecnologia blockchain na promoção de uma infra-estrutura de confiança abrange diversas áreas da computação. Neste capítulo, abordamos algumas aplicações emergentes em redes de computadores, computação em névoa e IoT. Em redes, ela é vislumbrada na implementação de planos de controles mais robustos por meio de controlares de distribuídos de redes definidas por software. Além disso, blockchains têm sido utilizadas no suporte à implementação e uso de funções de redes virtualizadas. Por fim, é preciso destacar o uso da tecnologia em paradigmas de redes recentes, tais como as redes centradas na informação e na promoção da segurança em redes.

Trabalhos recentes têm proposto uma arquitetura de referência e componentes de uma plataforma de névoa. No entanto, as soluções propostas continuam a ser desenvolvidas como prova de conceito, implementadas em ambientes limitados e condições restritas [Coutinho et al. 2018]. Neste sentido, a blockchain poderá ser o arcabouço que dará suporte ao processamento e coordenação das transações entre os elementos da névoa, tornando viável a operação em larga escala de aplicações voltadas à IoT. Ao mesmo tempo, a névoa pode oferecer um ambiente estruturado para implementação de diferentes soluções blockchain. Para a IoT, a blockchain vem ofertar uma nova camada computacional para compartilhamento e análise segura dos dados, com garantias de privacidade. Assim, essa camada pode ser utilizada para autenticar, autorizar, controlar e auditar os dados gerados pelos objetos inteligentes [Christidis and Devetsikiotis 2016].

References

- [Agarwal et al. 2014] Agarwal, K., Rozner, E., Dixon, C., and Carter, J. (2014). Sdn traceroute: Tracing sdn forwarding without changing network behavior. In *3rd Workshop on Hot Topics in SDN*, HotSDN '14, pages 145–150, New York, NY, USA. ACM.

- [Alvarenga et al. 2018] Alvarenga, I. D., Rebello, G. A. F., and Duarte, C. (2018). Securing configuration management and migration of virtual network functions using blockchain. In *IEEE/IFIP Network Operations and Management Symposium (NOMS)*.
- [Antonopoulos 2017] Antonopoulos, A. (2017). *Mastering Bitcoin: Programming the Open Blockchain*. O'reilly, 2nd edition.
- [Atzei et al. 2017] Atzei, N., Bartoletti, M., and Cimoli, T. (2017). A survey of attacks on ethereum smart contracts (sok). In Maffei, M. and Ryan, M., editors, *Principles of Security and Trust*, pages 164–186, Berlin, Heidelberg. Springer Berlin Heidelberg.
- [Bag et al. 2017] Bag, S., Ruj, S., and Sakurai, K. (2017). Bitcoin block withholding attack: Analysis and mitigation. *IEEE Transactions on Information Forensics and Security*, 12(8):1967–1978.
- [Baird 2016] Baird, L. (2016). The swirlds hashgraph consensus algorithm: Fair, fast, byzantine fault tolerance. Technical Report SWIRLDS-TR-2016-01, SWIRLDS Project.
- [Bano et al. 2017] Bano, S., Sonnino, A., Al-Bassam, M., Azouvi, S., McCorry, P., Meiklejohn, S., and Danezis, G. (2017). Sok: Consensus in the age of blockchains. In arxiv.org/abs/1711.03936.
- [Bashir 2017] Bashir, I. (2017). *Mastering Blockchain: Distributed ledgers, decentralization and smart contracts explained*. Packt Publishing.
- [Ben-Or 1983] Ben-Or, M. (1983). Another advantage of free choice: Completely asynchronous agreement protocols (extended abstract). In *Proc. of 2nd ACM Symp. on Principles of Distributed Computing*, pages 27–30.
- [Bentov et al. 2016] Bentov, I., Pass, R., and Shi, E. (2016). Snow white: Provably secure proofs of stake. *IACR Cryptology ePrint Archive*, 2016:919.
- [Bessani et al. 2014] Bessani, A. N., Sousa, J., and Alchieri, E. A. P. (2014). State machine replication for the masses with bft-smart. *44th IEEE/IFIP Int. Conf. on Dependable Systems and Networks*, pages 355–362.
- [Bhowmik and Feng 2017] Bhowmik, D. and Feng, T. (2017). The multimedia blockchain: A distributed and tamper-proof media transaction framework. In *Digital Signal Processing (DSP), 2017 22nd International Conference on*, pages 1–5. IEEE.
- [Blenk et al. 2016] Blenk, A., Basta, A., Reisslein, M., and Kellerer, W. (2016). Survey on network virtualization hypervisors for software defined networking. *IEEE Communications Surveys Tutorials*, 18(1):655–685.
- [Bonneau 2016] Bonneau, J. (2016). Why buy when you can rent? In *International Conference on Financial Cryptography and Data Security*, pages 19–26. Springer.
- [Bonneau et al. 2015] Bonneau, J., Miller, A., Clark, J., Narayanan, A., Kroll, J. A., and Felten, E. W. (2015). Sok: Research perspectives and challenges for bitcoin and cryptocurrencies. *IEEE Symposium on Security and Privacy*, pages 104–121.

- [Bozic et al. 2016] Bozic, N., Pujolle, G., and Secci, S. (2016). A tutorial on blockchain and applications to secure network control-planes. In *3rd Smart Cloud Networks Systems (SCNS)*, pages 1–8.
- [Bozic et al. 2017] Bozic, N., Pujolle, G., and Secci, S. (2017). Securing virtual machine orchestration with blockchains. In *1st Cyber Security in Networking Conference (CSNet)*, pages 1–8.
- [Brewer 2000] Brewer, E. A. (2000). Towards robust distributed systems (abstract). In *Symp. on Principles of Distributed Computing (PODC)*, pages 7–7. ACM.
- [Brody and Pureswaran 2014] Brody, P. and Pureswaran, V. (2014). Device democracy: Saving the future of the internet of things. *IBM, September*.
- [Buterin 2014] Buterin, V. (2014). A next-generation smart contract and decentralized application platform. Technical report, White paper.
- [Cachin et al. 2016] Cachin, C., Caro, A. D., Christidis, K., and Yellick, J. (2016). Architecture of the hyperledger blockchain fabric. Technical report, IBM Research - Zurich.
- [Cachin et al. 2000] Cachin, C., Kursawe, K., and Shoup, V. (2000). Random oracles in constantinople: Practical asynchronous byzantine agreement using cryptography. *Journal of Cryptology*, 18:219–246.
- [Cachin and Vukoli’c 2017] Cachin, C. and Vukoli’c, M. (2017). Blockchains consensus protocols in the wild. In *IBM Research – Zurich*.
- [Castro and Liskov 1999] Castro, M. and Liskov, B. (1999). Practical byzantine fault tolerance. In *3rd Symp. on Operating Systems Design and Implementation, OSDI’99*, pages 173–186, Berkeley, CA, USA. USENIX Association.
- [Castro and Liskov 2002] Castro, M. and Liskov, B. (2002). Practical Byzantine fault-tolerance and proactive recovery. *ACM Transactions on Computer Systems*, 20(4):398–461.
- [Chandra and Toueg 1996] Chandra, T. and Toueg, S. (1996). Unreliable failure detectors for reliable distributed systems. *Journal of the ACM*, 43(2):225–267.
- [Chandra et al. 2007] Chandra, T. D., Griesemer, R., and Redstone, J. (2007). Paxos made live: an engineering perspective. In *Symp. on Principles of Distributed Computing (PODC)*, pages 398–407.
- [Chandra et al. 1996] Chandra, T. D., Hadzilacos, V., and Toueg, S. (1996). The weakest failure detector for solving consensus. *Journal of the ACM*, 43(4):685–722.
- [Charron-Bost et al. 2010] Charron-Bost, B., Pedone, F., and Schiper, A. e. (2010). *Replication Theory and Practice*, volume 5959. Springer.
- [Christidis and Devetsikiotis 2016] Christidis, K. and Devetsikiotis, M. (2016). Blockchains and smart contracts for the internet of things. *IEEE Access*, 4:2292–2303.

- [Chu 1998] Chu, F. (1998). Reducing ω to $\diamond s$. *Inf. Process. Lett.*, 67(6):289–293.
- [Compagno et al. 2013] Compagno, A., Conti, M., Gasti, P., and Tsudik, G. (2013). Poseidon: Mitigating interest flooding DDoS attacks in Named Data Networking. In *38th IEEE Conf. on Local Computer Networks*, pages 630–638.
- [Conti et al. 2017] Conti, M., E, S. K., Lal, C., and Ruj, S. (2017). A survey on security and privacy issues of bitcoin. *CoRR*, abs/1706.00916.
- [Copeland and Zhong 2014] Copeland, C. T. and Zhong, H. (2014). Tangaroa: a byzantine fault tolerant raft.
- [Courtois and Bahack 2014] Courtois, N. T. and Bahack, L. (2014). On subversive miner strategies and block withholding attack in bitcoin digital currency. *CoRR*, abs/1402.1718.
- [Coutinho et al. 2018] Coutinho, A., Greve, F., Prazeres, C., and Cardoso, J. (2018). Fogbed: A rapid-prototyping emulation environment for fog computing. In *(ICC) Int. Conf. on Communications*. IEEE.
- [Coutinho et al. 2016] Coutinho, A. A. T. R., Carneiro, E. O., and Greve, F. (2016). Computação em névoa: Conceitos, aplicações e desafios. In *Minicursos do XXXIV Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, chapter 6, pages 266–315. SBC.
- [Crain et al. 2017] Crain, T., Gramoli, V., Larrea, M., and Raynal, M. (2017). (leader/randomization/signature)-free byzantine consensus for consortium blockchains. *CoRR*, abs/1702.03068.
- [Croman et al. 2016] Croman, K., Decker, C., Eyal, I., Gencer, A. E., Juels, A., Kosba, A. E., Miller, A., Saxena, P., Shi, E., Sirer, E. G., Song, D. X., and Wattenhofer, R. (2016). On scaling decentralized blockchains. In *Proc. 3rd Workshop on Bitcoin and Blockchain Research*.
- [Dai et al. 2013] Dai, H., Wang, Y., Fan, J., and Liu, B. (2013). Mitigate DDoS attacks in NDN by interest traceback. In *IEEE Conf. on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 381–386.
- [Défago et al. 2004] Défago, X., Schiper, A., and Urban, P. (2004). Total order broadcast and multicast algorithms: Taxonomy and survey. *ACM Computing Surveys*, 4:372–421.
- [Dinh et al. 2017a] Dinh, T. T. A., Liu, R., Zhang, M., Chen, G., Ooi, B. C., and Wang, J. (2017a). Untangling blockchain: A data processing view of blockchain systems. *IEEE Transactions on Knowledge and Data Engineering*, pages 99–.
- [Dinh et al. 2017b] Dinh, T. T. A., Wang, J., Chen, G., Liu, R., Ooi, B. C., and Tan, K.-L. (2017b). Blockbench: A framework for analyzing private blockchains. In *SIGMOD Conference*.

- [Douceur 2002] Douceur, J. (2002). The sybil attack. In *Proceedings of the 1st International Workshop on Peer-to-Peer Systems*, pages 251–260. Springer-Verlag.
- [Duong et al. 2016] Duong, T., Fan, L., and Zhou, H.-S. (2016). 2-hop blockchain: Combining proof-of-work and proof-of-stake securely. Technical report, Cryptology ePrint Archive, Report 2016/716, 2016. <https://eprint.iacr.org/2016/716>.
- [Dwork et al. 1988] Dwork, C., Lynch, N. A., and Stockmeyer, L. (1988). Consensus in the presence of partial synchrony. *Journal of ACM*, 35(2):288–322.
- [Ekblaw et al. 2016] Ekblaw, A., Azaria, A., Halamka, J. D., and Lippman, A. (2016). A case study for blockchain in healthcare: “medrec” prototype for electronic health records and medical research data. In *IEEE Open & Big Data Conference*, volume 13, page 13.
- [Eyal 2015] Eyal, I. (2015). The miner’s dilemma. In *Security and Privacy (SP), 2015 IEEE Symposium on*, pages 89–103. IEEE.
- [Eyal et al. 2016] Eyal, I., Gencer, A. E., Sirer, E. G., and van Renesse, R. (2016). Bitcoin-ng: A scalable blockchain protocol. In *Proc. of the 13th Usenix Conf. on Networked Systems Design and Implementation (NSDI)*, pages 45–59.
- [Eyal and Sirer 2014] Eyal, I. and Sirer, E. G. (2014). Majority is not enough: Bitcoin mining is vulnerable. In *Int. Conf. on financial cryptography and data security*, pages 436–454. Springer.
- [Fabric 2017] Fabric, H. (2017). What is hyperledger fabric.
- [Fischer et al. 1985] Fischer, M. J., Lynch, N. A., and Paterson, M. D. (1985). Impossibility of distributed consensus with one faulty process. *Journal of ACM*, 32(2):374–382.
- [Fotiou and Polyzos 2016a] Fotiou, N. and Polyzos, G. C. (2016a). Decentralized name-based security for content distribution using blockchains. In *IEEE Conf. on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 415–420.
- [Fotiou and Polyzos 2016b] Fotiou, N. and Polyzos, G. C. (2016b). Decentralized name-based security for content distribution using blockchains. In *Conf. on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 415–420. IEEE.
- [Frank et al. 2014] Frank, L., Pedersen, R. U., Frank, C. H., and Larsson, N. J. (2014). The cap theorem versus databases with relaxed acid properties. In *8th Int. Conf. on Ubiquitous Information Management and Communication (ICUIM)*, pages 1–7. ACM.
- [Gafni and Lamport 2003] Gafni, E. and Lamport, L. (2003). Disk paxos. *Distributed Computing*, 16(1):1–20.
- [Gilbert and Lynch 2012] Gilbert, S. and Lynch, N. A. (2012). Perspectives on the cap theorem. *IEEE Computer*, 45:30–36.
- [Gramoli 2017] Gramoli, V. (2017). From blockchain consensus back to byzantine consensus. *Future Generation of Computer Systems - available on line Sep/2017*.

- [Greve 2005] Greve, F. (2005). Protocolos fundamentais para o desenvolvimento de aplicações robustas. In *Minicursos do Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, chapter 5, pages 330–398. SBC.
- [Guerraoui et al. 2008] Guerraoui, R., Knezevic, N., Quéma, V., and Vukolic, M. (2008). The next 700 bft protocols. *ACM Trans. Comput. Syst.*, 32:12:1–12:45.
- [Guerraoui and Raynal 2007] Guerraoui, R. and Raynal, M. (2007). The alpha of indulgent consensus. *Comput. J.*, 50(1):53–67.
- [Huang et al. 2018] Huang, H., Chen, X., Wu, Q., Huang, X., and Shen, J. (2018). Bitcoin-based fair payments for outsourcing computations of fog devices. *Future Generation Computer Systems*, 78:850–858.
- [Hunt et al. 2010] Hunt, P., Konar, M., Junqueira, F. P., and Reed, B. (2010). Zookeeper: Wait-free coordination for internet-scale systems. In *USENIX Annual Technical Conference*, pages 11–11.
- [Jacobson et al. 2009] Jacobson, V., Smetters, D. K., Thornton, J. D., Plass, M. F., Briggs, N. H., and Braynard, R. L. (2009). Networking named content. In *5th Int. Conf. on Emerging networking experiments and technologies*, pages 1–12. ACM.
- [Jin et al. 2017a] Jin, T., Zhang, X., Liu, Y., and Lei, K. (2017a). Blockndn: A bitcoin blockchain decentralized system over named data networking. In *9th Int. Conf. on Ubiquitous and Future Networks (ICUFN)*, pages 75–80.
- [Jin et al. 2017b] Jin, T., Zhang, X., Liu, Y., and Lei, K. (2017b). Blockndn: A bitcoin blockchain decentralized system over named data networking. In *9th Int. Conf. on Ubiquitous and Future Networks (ICUFN)*, pages 75–80. IEEE.
- [Karame et al. 2012] Karame, G. O., Androulaki, E., and Capkun, S. (2012). Double-spending fast payments in bitcoin. In *ACM Conf. on Computer and communications security*, pages 906–917. ACM.
- [Kiayias et al. 2016] Kiayias, A., Konstantinou, I., Russell, A., David, B. M., and Oliynykov, R. (2016). A provably secure proof-of-stake blockchain protocol. *IACR Cryptology ePrint Archive*, 2016:889.
- [Kreutz et al. 2015] Kreutz, D., Ramos, F. M., Verissimo, P. E., Rothenberg, C. E., Azodolmolky, S., and Uhlig, S. (2015). Software-defined networking: A comprehensive survey. *Proceedings of the IEEE*, 103(1):14–76.
- [Kwon et al. 2017] Kwon, Y., Kim, D., Son, Y., Vasserman, E., and Kim, Y. (2017). Be selfish and avoid dilemmas: Fork after withholding (faw) attacks on bitcoin. In *ACM SIGSAC Conf. on Computer and Communications Security*, pages 195–209. ACM.
- [Lamport 1998] Lamport, L. (1998). The part-time parliament. *ACM Transactions Computer Systems*, 16(2):133–169.
- [Lamport et al. 1982] Lamport, L., Shostak, R., and Pease, M. (1982). The Byzantine generals problem. *ACM Trans. on Programming Languages and Systems*, 4(3):382–401.

- [Lampson 2001] Lampson, B. (2001). The abcd's of paxos. In *Proceedings of the 20th annual ACM Symposium on Principles of Distributed Computing*, page 13.
- [Li and Bi 2014] Li, Z. and Bi, J. (2014). Interest cash: an application-based counter-measure against interest flooding for dynamic content in named data networking. In *9th Int. Conf. on Future Internet Technologies*, page 2. ACM.
- [Mijumbi et al. 2016] Mijumbi, R., Serrat, J., Gorricho, J. L., Bouten, N., Turck, F. D., and Boutaba, R. (2016). Network function virtualization: State-of-the-art and research challenges. *IEEE Communications Surveys Tutorials*, 18(1):236–262.
- [Miller et al. 2014] Miller, A., Juels, A., Shi, E., Parno, B., and Katz, J. (2014). Permacoin: Repurposing bitcoin work for data preservation. *2014 IEEE Symposium on Security and Privacy*, pages 475–490.
- [Nakamoto 2008] Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system. Technical report, Bitcoin Org.
- [Narayanan et al. 2016] Narayanan, A., Bonneau, J., Felten, E., Miller, A., and Goldfeder, S. (2016). *Bitcoin and cryptocurrency technologies*. Princeton University Press.
- [Natoli and Gramoli 2016] Natoli, C. and Gramoli, V. (2016). The blockchain anomaly. *IEEE 15th Int. Symp. on Network Computing and Applications (NCA)*, pages 310–317.
- [Nguyen et al. 2017] Nguyen, B. Q., Androulaki, E., and et. al., A. D. C. (2017). Hyperledger: Blockchain. Hyperledger Org.
- [Nguyen et al. 2018] Nguyen, B. Q., Androulaki, E., and et. al., A. D. C. (2018). Hyperledger: Protocol specification. Hyperledger Org.
- [Ongaro and Ousterhout 2014] Ongaro, D. and Ousterhout, J. K. (2014). In search of an understandable consensus algorithm. In *USENIX Annual Technical Conference*, pages 305–320.
- [Paillisse et al. 2017] Paillisse, J., Rodriguez-Natal, A., Ermagan, V., Maino, F., and Cabellos, A. (2017). An analysis of the applicability of blockchain to secure ip addresses allocation, delegation and bindings. In *Internet draft (draft-paillisse-sidrops-blockchain-01)*. IETF.
- [Peters and Panayi 2016] Peters, G. W. and Panayi, E. (2016). Understanding modern banking ledgers through blockchain technologies: Future of transaction processing and smart contracts on the internet of money. In *Banking Beyond Banks and Money*, pages 239–278. Springer.
- [Poon and Dryja 2016] Poon, J. and Dryja, T. (2016). The bitcoin lightning network: Scalable off-chain instant payments. Technical report, Lightning Network.
- [Rao et al. 2011] Rao, J., Shekita, E. J., and Tata, S. (2011). Using paxos to build a scalable, consistent, and highly available datastore. *Proc. of the VLDB Endowment (PVLDB)*, 4:243–254.

- [Rosenfeld 2011] Rosenfeld, M. (2011). Analysis of bitcoin pooled mining reward systems. *CoRR*, abs/1112.4980.
- [Salah and Strufe 2016] Salah, H. and Strufe, T. (2016). Evaluating and mitigating a collusive version of the interest flooding attack in ndn. In *IEEE Symp. on Computers and Communication (ISCC)*, pages 938–945.
- [Salahuddin et al. 2017] Salahuddin, M. A., Al-Fuqaha, A., Guizani, M., Shuaib, K., and Sallabi, F. (2017). Softwarization of internet of things infrastructure for secure and smart healthcare. *Computer*, 50(7):74–79.
- [Sankar et al. 2017] Sankar, L. S., Sindhu, M., and Sethumadhavan, M. (2017). Survey of consensus protocols on blockchain applications. In *4th Int. Conf. on Advanced Computing and Communication Systems (ICACCS)*, pages 1–5.
- [Santos et al. 2006] Santos, B. P., Silva, L. A. M., and Celes, C. S. F. S. e. a. (2006). Internet das coisas: da teoria á prática. In SBC, editor, *Minicursos XXXIV Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuidos*, pages 1–50. SBC.
- [Schneider 1990] Schneider, F. B. (1990). Implementing fault-tolerant service using the state machine aproach: A tutorial. *ACM Computing Surveys*, 22(4):299–319.
- [Seixas et al. 2017] Seixas, N. F. S., Ribeiro, A., and Sampaio, L. (2017). Um modelo de rede centrada na informação resiliente a ataques de negação de serviços por inundação de interesses. In *XXXV Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC)*, Belém.
- [Sharma et al. 2017a] Sharma, P. K., Chen, M. Y., and Park, J. H. (2017a). A software defined fog node based distributed blockchain cloud architecture for iot. *IEEE Access*, PP(99):1–1.
- [Sharma et al. 2017b] Sharma, P. K., Singh, S., Jeong, Y. S., and Park, J. H. (2017b). Distblocknet: A distributed blockchains-based secure sdn architecture for iot networks. *IEEE Communications Magazine*, 55(9):78–85.
- [Sousa et al. 2018] Sousa, J., Bessani, A., and Vukolic, M. (2018). A byzantine fault-tolerant ordering service for the hyperledger fabric blockchain platform. In *DSN’18: The IEEE/IFIP Int. Conf. on Dependable Systems and Networks*.
- [Szabo 1997] Szabo, N. (1997). Formalizing and securing relationships on public networks. *First Monday*, 2(9).
- [Tourani et al. 2017] Tourani, R., Misra, S., Mick, T., and Panwar, G. (2017). Security, privacy, and access control in information-centric networking: A survey. *IEEE Communications Surveys & Tutorials*.
- [Vukolic 2015] Vukolic, M. (2015). The quest for scalable blockchain fabric: Proof-of-work vs. bft replication. In *Open Problems in Network Security (iNetSeC)*, volume 9591, pages 112–125. Lecture Notes in Computer Science - Springer.

- [Vukolić 2017] Vukolić, M. (2017). Rethinking permissioned blockchains: Invited paper. In *1st ACM Workshop on Blockchains, Cryptocurrencies and Contracts (BCC)*.
- [Wang et al. 2014] Wang, K., Zhou, H., Qin, Y., and Zhang, H. (2014). Cooperative-filter: countering interest flooding attacks in named data networking. *Soft Computing*, 18(9):1803–1813.
- [Xu et al. 2016] Xu, X., Pautasso, C., Zhu, L., Gramoli, V., Ponomarev, A., Tran, A. B., and Chen, S. (2016). The blockchain as a software connector. *13th Working IEEE/IFIP Conf. on Software Architecture (WICSA)*, pages 182–191.
- [Xylomenos et al. 2014a] Xylomenos, G., Ververidis, C. N., Siris, V. A., Fotiou, N., Tsilopoulos, C., Vasilakos, X., Katsaros, K. V., and Polyzos, G. C. (2014a). A survey of information-centric networking research. *IEEE Communications Surveys & Tutorials*, 16(2):1024–1049.
- [Xylomenos et al. 2014b] Xylomenos, G., Ververidis, C. N., Siris, V. A., Fotiou, N., Tsilopoulos, C., Vasilakos, X., Katsaros, K. V., and Polyzos, G. C. (2014b). A survey of information-centric networking research. *IEEE Communications Surveys & Tutorials*, 16(2):1024–1049.
- [Yang et al. 2017] Yang, H., Zheng, H., Zhang, J., Wu, Y., Lee, Y., and Ji, Y. (2017). Blockchain-based trusted authentication in cloud radio over fiber network for 5g. In *16th Int. Conf. on Optical Communications and Networks (ICOON)*, pages 1–3.
- [Yin et al. 2018] Yin, H., Guo, D., Wang, K., Jiang, Z., Lyu, Y., and Xing, J. (2018). Hyperconnected network: A decentralized trusted computing and networking paradigm. *IEEE Network*, 32(1):112–117.
- [Zhang et al. 2014] Zhang, L., Afanasyev, A., Burke, J., Jacobson, V., Crowley, P., Papadopoulos, C., Wang, L., Zhang, B., et al. (2014). Named data networking. *ACM SIGCOMM Computer Communication Review*, 44(3):66–73.
- [Ølnes et al. 2017] Ølnes, S., Ubacht, J., and Janssen, M. (2017). Blockchain in government: Benefits and implications of distributed ledger technology for information sharing. *Government Information Quarterly*, 34(3):355 – 364.