Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Uma Proposta para a Descoberta e para a Alocação de Recursos Computacionais em Fog Computing

João Bachiega Júnior

Tese apresentada como requisito parcial para
conclusão do Doutorado em Informática

Orientadora
Prof.a Dr.a Aletéia Patrícia Favacho de Araújo von Paumgartten

Brasília
2025

Ficha Catalográfica de Teses e Dissertações

Está página existe apenas para indicar onde a ficha catalográfica gerada para dissertações de mestrado e teses de doutorado defendidas na UnB. A Biblioteca Central é responsável pela ficha, mais informações nos sítios:

http://www.bce.unb.br
http://www.bce.unb.br/elaboracao-de-fichas-catalograficas-de-teses-e-dissertacoes

Esta página não deve ser inclusa na versão final do texto.

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Uma Proposta para a Descoberta e para a Alocação de Recursos Computacionais em Fog Computing

João Bachiega Júnior

Tese apresentada como requisito parcial para
conclusão do Doutorado em Informática

Prof.a Dr.a Aletéia Patrícia Favacho de Araújo von Paumgartten (Orientadora)
PPGI/UnB

Prof. Dr. Daniel Henriques Moreira
Loyola University Chicago

Prof. Dr. Alfredo Goldman vel Lejbman
IME/USP

Prof.a Dr.a Lucia Maria de Assumpção Drummond
IC/UFF

Prof.a Dr.a Maristela Terto de Holanda
PPGI/UnB

Prof. Dr. Rodrigo Bonifácio Almeida
Coordenador do Programa de Pós-graduação em Informática

Brasília, 28 de março de 2025

Dedicatória

Para Gustavo, Cristiane, Alzira e João, por acreditarem que concluir esta etapa seria possível antes que eu mesmo acreditasse.

Agradecimentos

Seria injusto acreditar que uma tese de doutorado é construída por um só autor. Trago aqui algumas das pessoas que estiveram comigo nesta longa jornada e gostaria que não apenas elas, mas todas as outras que um dia participaram um pouquinho desta caminhada se sentissem abraçadas e aceitassem o meu agradecimento.

Em 1998 eu concluí o ensino fundamental e ingressaria no ensino médio. As condições da época me fizeram escolher pelo curso Técnico de Processamento de Dados. Minha mãe e meu pai, na simplicidade de suas sabedorias, se esforçaram e investiram em um microcomputador para que eu pudesse estudar melhor as disciplinas do curso. Como troca, eles me pediram apenas uma coisa: estude. Mãe e pai, eu espero ter cumprido o que vocês me pediram e confiaram que eu faria. Estudei, estudei muito. Como gratidão e reconhecimento ao esforço deles, não me sentiria realizado se não entregasse ao menos este título de doutorado em "coisas de computador". Então, meus primeiros agradecimentos são para eles, que foram uma motivação para que este trabalho fosse concluído.

Quase que na mesma época conheci alguém que mudaria minha vida. Cris, ter tido você do meu lado desde que éramos apenas dois adolescentes apaixonados me fez sentir confortável em arriscar, em ter coragem quando tudo parecia perdido e a saber que mesmo que se nada desse certo, você estaria comigo e isto já me bastaria. Obrigado por acreditar comigo. Obrigado por abster-se de fins de semana e de momentos de descontração para que eu pudesse estudar. Obrigado por não ter me deixado desistir. Sei o quanto se orgulha dessa nossa conquista. Sem você, isto não seria possível. Nunca esqueça o quanto eu amo você.

Guga, que pena que o papai vai se limitar em só um parágrafo para te agradecer. Você é um anjo que Deus pôs em nossas vidas num desafio constante de te educar e de aprender contigo. Obrigado pela paciência, pelo entendimento, pela compreensão e pela maturidade que com seus poucos anos de vida você já demonstra. Estude, filho. Estude. Nunca no mundo haverá algo que substituirá a educação. Se o papai foi capaz de obter este título de doutorado em Informática, você é capaz de chegar muito mais longe. Uma das grandes motivações de buscar esta minha formação é mostrar para você que estudar é bom. Faça isso, sempre.

Ainda existem muitas outras pessoas para agradecer. Às minhas irmãs, Denise e Adriana, que apoiaram desde sempre e fizeram de tudo para que eu tivesse disponibilidade para estudar. À minha sogra, Fátima, e ao meu sogro, Valmir (*in memorian*), que não mediram esforços para que o difícil se tornasse fácil. Aos colegas de trabalho que se sacrificaram de algum modo para que eu pudesse conciliar os compromissos do trabalho com os estudos. Aos colegas de doutorado, Breno, Leonardo e Michel. Juntos formamos um time forte e alegre, capaz de discutir assuntos com profundidade e leveza.

À minha orientadora, Aletéia, agradeço pelos ensinamentos, pela orientação, pelo cuidado e pela dedicação. Sempre peço à Deus que coloque pessoas boas em minha vida e eu tenho certeza que ela é uma dessas pessoas. Obrigado por ter acreditado em mim, por ter se preocupado comigo durante todos estes mais de oito anos que sou seu aluno. Você sempre será lembrada com muito carinho por toda minha família.

Por fim, agradeço à Deus e à Nossa Senhora Aparecida, com a convicção que este é o agradecimento mais importante, pois sem eles nada disso seria possível. Eles me escutaram nas horas de angústia, orientaram-me quando tudo parecia escuro, e me abriram portas que nós homens, somos incapazes de compreender com a nossa simples razão. Eles me indicaram o caminho e me deram forças o tempo todo para não vacilar na caminhada. A eles confidenciei silenciosamente alegrias e sofrimentos, pedi muita ajuda e me escutaram. Nunca foi sorte, sempre foi Deus.

Este estudo foi parcialmente financiado pelo projeto BioCloud2, aprovado sob Chamada CNPq/AWS nº 64/2022, processo CNPq nº 421828/2022-6.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES), por meio do Acesso ao Portal de Periódicos.

Resumo

A *fog computing* (computação em névoa) é um paradigma que permite o provisionamento de recursos e serviços computacionais na borda da rede, mais próximo dos dispositivos finais e usuários, com menor latência, complementando a *cloud computing* (computação em nuvem). A heterogeneidade e o grande número de dispositivos são desafios a serem superados pelo gerenciamento de recursos neste ambiente, que tem o objetivo de garantir que os recursos necessários estejam disponíveis no momento certo e no local adequado para que as tarefas sejam concluídas com sucesso. Assim sendo, entre as principais funcionalidades do gerenciamento de recursos estão a etapa de Descoberta que visa encontrar os recursos adequados; e a etapa de Alocação que objetiva selecionar, reservar e garantir o uso dos melhores recursos para a execução de uma dada carga de trabalho. Contudo, observou-se que as soluções propostas até o momento na literatura são incompletas, pois não consideram concomitantemente as perspectivas do usuário e do provedor, uma vez que elas possuem objetivos diferentes. Além disso, as soluções da literatura não levam em consideração que um gerenciamento de recursos eficiente em ambiente de *fog computing* deve considerar as capacidades computacionais e as características comportamentais e, portanto, devem ser tratados de forma diferenciada. Assim sendo, nesta tese é apresentada uma proposta para as etapas de Descoberta e Alocação de recursos que fazem parte do serviço de gerenciamento em um ambiente de *fog computing*. A proposta considera as capacidades computacionais e as características comportamentais, a partir das perspectivas do provedor e do usuário final. A validação das propostas foi realizada em ambientes reais e simulados. A proposta de descoberta de recursos trouxe resultados até 33% melhores quando comparada com outras existentes na literatura em relação ao tempo de execução. Já a proposta de alocação de recursos em *fog computing* foi capaz de gerar resultados até 47% melhores em relação ao custo.

Palavras-chave: Fog Computing, Alocação de Recursos, Descoberta de Recursos, Gerenciamento de Recursos

Abstract

Fog computing is a paradigm that allows the provisioning of computing resources and services at the edge of the network, closer to end devices and users, with lower latency, complementing cloud computing. Heterogeneity and the large number of devices are challenges to be overcome by resource management in this environment, which aims to ensure that the necessary resources are available at the right time and in the right place so that tasks are completed successfully. Therefore, among the main functionalities of resource management are the Discovery stage, which aims to find the appropriate resources; and the Allocation stage, which aims to select, reserve, and ensure the use of the best resources for the execution of a given workload. However, it was observed that the solutions proposed so far in the literature are incomplete, as they do not simultaneously consider the perspectives of the user and the provider, since they have different objectives. Furthermore, the solutions in the literature do not take into account that efficient resource management in a fog computing environment must consider computational capabilities and behavioral characteristics and, therefore, must be treated differently. Therefore, this thesis presents a proposal for the Resource Discovery and Allocation steps that are part of the management service in a fog computing environment. The proposal considers computational capabilities and behavioral characteristics, from the perspectives of the provider and the end user. The proposals were validated in real and simulated environments. The resource discovery proposal brought results up to 33% better when compared to others in the literature concerning execution time. The resource allocation proposal in fog computing generated results up to 47% better concerning cost.

Keywords: Fog Computing, Resource Allocation, Resource Discovery, Resource Management

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

A *cloud computing* é considerada um paradigma maduro porque está em vigor desde 2006 [14]. Esse paradigma oferece recursos virtualizados, criados em uma infraestrutura compartilhada, que são consumidos pelos clientes em um modelo de negócio do tipo *pay-per-use* [15], com uma variedade de opções de custo [16, 17], e escalabilidade automática para seus serviços [18]. A *cloud computing* é baseada em grandes *datacenters*, distribuídos ao redor do mundo, que são colocados no centro da rede e acessados por meio da Internet. Estes *datacenters* são compostos por milhares de servidores físicos, ricos em recursos que são interconectados por uma rede redundante e estável [19].

Nos últimos anos, com os avanços tecnológicos nos recursos computacionais, tais como processadores, memória e comunicações, uma infinidade de objetos passaram a ter alguma capacidade computacional, dando forma ao que é conhecido como era da Internet das Coisas (do inglês, *Internet of Things* (IoT)) [20]. Dispositivos que medem parâmetros relacionados à saúde, como *smartbands*, relógios inteligentes, dispositivos para construções inteligentes, redes elétricas e semáforos de cidades inteligentes, são exemplos dessa nova área [21]. Neste novo cenário, surge a necessidade por respostas rápidas das aplicações, exigindo uma comunicação de baixa latência [22]. Devido à isso, a *cloud computing* não atende adequadamente a essas novas demandas computacionais, caracterizadas pela necessidade de um paradigma computacional mais próximo do usuário final e das "coisas".

Nesse cenário, a *fog computing* surgiu como uma solução promissora para atender a essa demanda crescente de expandir a capacidade de processamento, rede e armazenamento para mais perto dos usuários finais, complementando assim a fragilidade da *cloud computing* [23]. As características essenciais da *fog computing* são baixa latência, grande distribuição geográfica, heterogeneidade, interoperabilidade, interações em tempo real e escalabilidade [24]. No entanto, pesquisas sobre o uso desse recente paradigma computacional ainda estão em desenvolvimento, e diversas questões estão em aberto [22].

Um ponto relevante da *fog computing* refere-se ao *fog node*. Ele pode ser qualquer

dispositivo de hardware que possua capacidade computacional, de comunicação e de virtualização, e que esteja em um ambiente de *fog computing* [9]. Nesse cenário, um importante desafio a ser tratado é o adequado gerenciamento destes *fog nodes*, uma vez que em um ambiente de *fog computing* predomina o uso de contextos dinâmicos, atrelado a competição pelo compartilhamento de recursos computacionais já alocados. Assim, isso pode levar o ambiente a eventos imprevisíveis, tais como indisponibilidade de serviço, alto tempo de resposta e confiabilidade reduzida [25].

Dessa forma, embora o escopo do processo de gerenciamento de recursos possa ser utilizado de uma forma bem abrangente, incluindo etapas de estimativa, escalonamento, orquestração, monitoração, entre outros, para o desenvolvimento deste trabalho serão consideradas as etapas de descoberta e de alocação de recursos. Considerando a restrição do poder computacional existente em um ambiente de *fog*, estas etapas são fundamentais para a melhor utilização dos recursos disponíveis. Isto porque, enquanto a descoberta visa encontrar os *fog nodes* disponíveis no ambiente, a alocação foca na melhor escolha entre os *fog nodes* disponíveis para a execução de uma aplicação ou serviço.

1.1 Motivação

A *fog computing*, embora tenha sido apresentada em 2012 [23], ainda é um paradigma que está em desenvolvimento na academia e na indústria [19]. Além disso, por trazer uma abordagem diferente das demais tecnologias existentes, tais como a *cloud computing*, as funções até então existentes para o gerenciamento de recursos não podem ser plenamente reutilizadas, exigindo o desenvolvimento de novas propostas.

Ao longo dos anos, algumas pesquisas foram publicadas sobre o processo completo de gerenciamento de recursos para *fog computing* [3, 26, 27] e, de forma mais específica, sobre as etapas de descoberta [28, 29] ou alocação de recursos [30, 31]. No entanto, após uma detalhada análise de 108 artigos, foi possível perceber uma fragilidade nas propostas ao não levarem em consideração, simultaneamente, as perspectivas do usuário e do provedor. O usuário busca sempre obter os melhores recursos disponíveis, o provedor visa a entrega de um conjunto mínimo de recursos para evitar custos desnecessários ou situações que envolvam indisponibilidade de recursos. Também deve ser considerado ainda que, diferentemente de outros paradigmas computacionais, tais como a *cloud computing*, os recursos computacionais na *fog computing* possuem capacidades computacionais e comportamentais [32]. As capacidades computacionais são aqueles referentes a itens de hardware, tais como CPU, armazenamento, memória e outras capacidades computacionais [9]. Já as características comportamentais referem-se àquelas que são desejáveis, como disponibilidade e confiabilidade [32]. Embora elas sejam desejáveis de serem consideradas, sua degradação

não é considerada como um impedimento para que a aplicação seja executada. Por outro lado, o mesmo não ocorre com as capacidades computacionais, uma vez que eles impactam diretamente a performance da aplicação e, portanto, o atendimento dos requisitos mínimos exigidos pelo demandante para que cada atributo seja obedecido.

Para lidar com essas limitações é necessário que o gerenciamento de recursos em *fog computing* garanta um uso eficiente e eficaz dos recursos na borda, a fim de diminuir a carga com o envio das aplicações para serem processadas ou armazenadas na *cloud computing*. Para isso, este trabalho focará nas etapas de descoberta e alocação de recursos, pois são as etapas mais críticas no serviço de gerenciamento dos recursos.

1.2 Hipótese

Este trabalho defende a hipótese de que um serviço eficiente de gerenciamento de recursos em um ambiente de *fog computing* está diretamente relacionado as etapas de descoberta e alocação de recursos, as quais devem levar em consideração as perspectivas do usuário e do provedor, assim como tratar concomitantemente as capacidades computacionais e as características comportamentais.

Além disso, a complexidade inerente dessas etapas exige a adoção de abordagens que conciliem a otimização do desempenho com a satisfação dos usuários quanto a maximização da utilização da infraestrutura disponível e do cumprimento dos requisitos junto ao provedor. Para isso, é fundamental considerar as perspectivas do usuário e do provedor, tanto em relação à capacidade computacional quanto as características comportamentais.

1.3 Objetivos deste Trabalho

O objetivo geral deste trabalho é desenvolver uma solução para as funcionalidades de descoberta e alocação de recursos computacionais em *fog computing*, que sejam capazes de se adaptar às perspectivas do usuário e do provedor de serviços. A solução também deve considerar que as capacidades computacionais são requisitos mínimos de hardware e devem ser obrigatoriamente atendidos, enquanto as características comportamentais são apenas desejáveis e não possuem esta mesma obrigatoriedade, pois não impactam diretamente a performance da aplicação. Assim, para que o objetivo geral deste trabalho seja cumprido, faz-se necessário atender aos seguintes objetivos específicos:

- Analisar a literatura existente com foco nos trabalhos mais relevantes sobre gerenciamento de recursos para *fog computing*;

- Revisar sistematicamente as propostas existentes na literatura sobre as funcionalidades de alocação e descoberta de recursos para *fog computing*;
- Propor métricas e técnicas de alocação de recursos em um ambiente de *fog computing*;
- Construir um modelo para a etapa de descoberta que seja capaz de catalogar os recursos, considerando as especificidades das capacidades computacionais e das características comportamentais;
- Desenvolver um algoritmo para alocação de recursos que considere ambas as capacidades computacionais e as características comportamentais do *fog node*, além das perspectivas do usuário final e dos provedores de serviço.

1.4 Contribuições desta Tese

Durante o período de desenvolvimento desta tese, contribuições científicas foram publicadas ou estão em análise para publicação futura. Elas estão sumarizadas na Tabela 1.1 e são apresentadas nos anexos desta tese. Assim sendo, os artigos produzidos nesta tese contam com um total de 227 citações[1], demonstrando uma clara contribuição deste trabalho com o estado da arte.

1.5 Organização desta Tese

Este trabalho está dividido em mais cinco capítulos além desta introdução. O Capítulo 2 traz uma contextualização sobre os aspectos relevantes a serem tratados no desenvolvimento deste trabalho, tais como os conceitos de *cloud computing*, *fog computing* e demais

[1]O número de citações foi coletado do Google Scholar em 27/01/25.

Tabela 1.1: Produção científica derivada desta tese.

Anexo	Tipo	Local	Nome	Ano	Tema da Publicação
1	A	Conf	CLOSER	2022	Fog Computing e Paradigmas Relacionados
2	A	Conf	ICOMP	2021	Fog Node
3	A	Per	JPDC	2025	Descoberta de Recursos na Fog
4	A	Per	ACM Computing Surveys	2023	Alocação de Recursos em Fog
5	A	Conf	Mobiquitous	2023	Ciclo de Vida dos Dados de Observabilidade
6	C	Liv	Handbook of Data Engineering	2024	Arquitetura de sistemas de observabilidade
7	A	Per	Computer Networks	2022	Taxonomia de monitoração em Fog
8	A	Per	ACM Computing Surveys	2022	Orquestração em Fog
9	A	Per	ICICT	2025	Descoberta de Recursos na Fog
10	A	Per	UCC	2024	Alocação de Recursos na Fog

Abreviações – **A** = Artigo; **C** = Capítulo; **Conf** = Conferência; **Per** = Periódico; **Liv** = Livro;

paradigmas relacionados. O Capítulo 3 apresenta a perspectiva computacional de um *fog node* e, ainda, apresenta os conceitos sobre o processo de gerenciamento de recursos, detalhando as etapas de estimativa, descoberta, alocação, monitoração e orquestração. No Capítulo 4 serão apresentados os resultados de uma revisão sistemática da literatura realizada especificamente sobre os processos de descoberta e de alocação de recursos para *fog computing*. A proposta de um algoritmo para a descoberta de recursos na *fog computing* baseada em qualidade de serviço e de experiência é apresenta no Capítulo 5. Já a proposta de um algoritmo para a alocação de recursos, incluindo a modelagem do sistema, os testes realizados e os resultados coletados até o momento é apresentada no Capítulo 6. Por fim, algumas considerações finais e os trabalhos futuros resultantes deste trabalho são apresentados no Capítulo 7. Os onze anexos deste documento trazem os artigos resultantes do trabalho desenvolvido para esta tese e que foram publicados ou estão em análise em periódicos e congressos.

Capítulo 2

Referencial Teórico

Neste capítulo as definições de *cloud computing* são apresentadas na Seção 2.1 e as definições de *sky computing* são descritas na Seção 2.2. A arquitetura e as características essenciais da *fog computing* são apresentadas na Seção 2.3. Por fim, alguns outros paradigmas computacionais relacionados são apresentados na Seção 2.4.

2.1 *Cloud Computing*

O conceito inicial de *cloud computing* foi introduzido em 1961 por John McCarthy quando disse que "algum dia a computação pode ser organizada como um serviço público assim como o sistema telefônico é um serviço público" [33]. A evolução da computação e da comunicação, atrelado a crescente demanda por recursos computacionais para a execução de aplicações que requerem cada vez mais capacidade de processamento e de armazenamento, permitiu a expansão do paradigma da *cloud computing* e, atualmente, essa plataforma é amplamente utilizada pela academia e pela indústria.

Fox *et al.* [18] citam que na *cloud computing* os recursos de tecnologia são fornecidos como serviços, permitindo aos usuários acessarem os serviços sob demanda, conforme sua necessidade e, sem precisar ter conhecimento sobre a tecnologia utilizada, fornecendo as características de disponibilidade e de escalabilidade.

Para Vaquero *et al.* [15] a *cloud computing* pode ser definida como um grande conjunto de recursos virtualizados e compartilhados (hardware, plataformas de desenvolvimento ou software) que podem ser facilmente acessados e utilizados. Ainda de acordo com este autor, esses recursos podem ser dinamicamente reconfigurados para se ajustarem a uma carga variável de trabalho, permitindo uso otimizado dos mesmos. Este conjunto de recursos é tipicamente explorado por um modelo de pagamento por utilização chamado de *pay-per-use* no qual as garantias são oferecidas pelo provedor de infraestrutura por meio de contratos de serviço personalizados, chamados de *Service Level Agreement* (SLA).

De acordo com o *National Institute of Standards and Technology* (NIST) [14], definição mais utilizada na literatura, a *cloud computing* é um modelo computacional que possibilita acesso, de modo conveniente e sob demanda, a um conjunto de recursos computacionais configuráveis (por exemplo, redes, servidores, armazenamento, aplicações e serviços) que podem ser rapidamente adquiridos e liberados com mínimo esforço gerencial ou interação com o provedor de serviços. Assim, a *cloud computing* apresenta cinco características essenciais [14]:

- Serviço sob demanda: capacidade de provisionar os serviços de forma direta, sem a necessidade de interação do usuário com o provedor do serviço;
- Acesso amplo a rede: capacidade de acesso aos serviços por meio de diversos dispositivos;
- Agrupamento de recursos: capacidade de agrupamento dos serviços de forma que o mesmo possa atender a diferentes demandas de usuários;
- Elasticidade: capacidade de adquirir e configurar os recursos automaticamente de forma rápida, conforme a demanda do usuário;
- Serviço medido: capacidade de medição e de controle dos serviços oferecidos, conforme a necessidade do usuário, fazendo com que o mesmo pague somente pelo serviço que efetivamente tiver usado.

Além disso, o NIST [14] descreve que o paradigma de *cloud computing* é composto por três modelos de serviços e quatro modelos de implantação, que são descritos nas próximas seções.

2.1.1 Modelos de Serviços

De acordo com NIST [14], a *cloud computing* pode ser entregue em três modelos de serviços:

- Software como Serviço (SaaS – *Software as a Service*): este modelo proporciona que os usuários tenham acesso aos serviços de software, por meio de diversos dispositivos, em qualquer lugar e a qualquer momento, por meio da Internet. Alguns exemplos deste modelo de serviço são: Microsoft Office 365, Google Apps e Dropbox;
- Plataforma como Serviço (PaaS – *Platform as a Service*): este modelo proporciona que os usuário tenham acesso aos serviços de plataforma, fornecendo ao usuário um ambiente de desenvolvimento de aplicações. Alguns desses serviços são: Heroky, Google App Engine e AWS Beanstalk;

- Infraestrutura como Serviço (IaaS – *Infrastructure as a Service*): este modelo garante que os usuários tenham acesso aos serviços de infraestrutura que dará suporte para ao SaaS e ao PaaS, proporcionando acesso aos recursos, tais como servidores, rede e armazenamento, baseado em técnicas de virtualização de recursos de computação. Assim, são exemplos de IaaS, Amazon EC2, Google Compute Engine (GCE) e Microsoft Azure VMs.

Neste ponto é importante salientar que os provedores de *cloud computing* têm amadurecido constantemente seus serviços, inclusive promovendo integração entre eles. À medida que a oferta de serviços cresce, as fronteiras previstas pelo NIST [14] vão sendo redefinidas. Muitos provedores têm renomeado seus serviços, em especial os SaaS, visto que o termo "software" é algo muito amplo. Assim, o termo "XaaS" (*Everthing as a Service*) tem sido usado para definir uma diversidade de diferentes modelos de serviço em *cloud computing* [34].

2.1.2 Modelos de Implantação

A *cloud computing* oferece quatro modelos de implantação para os seus serviços, sendo eles [14]:

- Privado: o serviço de infraestrutura é disponibilizado especificamente para um público com total restrição de acesso e, exclusivamente, operado pelo mesmo;
- Público: a infraestrutura é provisionada para atender ao público em geral. Ela pode ser provida por empresas, pelo governo ou por centros acadêmicos;
- Comunitário: o serviço de infraestrutura é compartilhado para um público específico, com interesses comuns;
- Híbrido: a infraestrutura é composta por, pelo menos, dois dos modelos citados anteriormente (privada, comunitária e pública). A *cloud computing* permanece como uma única plataforma, permitindo a troca de dados e aplicações.

Além destes modelos apresentados pelo NIST [14], existe também a implantação de maneira federada. Uma federação de nuvens é um sistema cooperativo de dois ou mais provedores de *cloud computing*. Esta cooperação é interessante por quebrar a dependência de um único provedor, melhorando assim a disponibilidade de serviço da federação, reduzindo custos, e também aumentando o número de combinações de instâncias [35]. Para este modelo tem sido adotado a definição de *Sky computing* [36], apresentado a seguir.

2.2 *Sky Computing*

O mercado de provedores públicos de *cloud computing* tornou-se mais competitivo ao longo do tempo. Atualmente, há uma profusão de serviços oferecidos por diferentes provedores. Muitos deles oferecem produtos semelhantes, mas usando abordagens diferentes. Nesse cenário, cresceu uma implantação *multicloud* dentro das organizações, na qual os recursos oferecidos por diferentes provedores compõem a estrutura de suporte tecnológico a ser utilizado. Essa abordagem foi denominado de *Sky computing* [36] e pode ser definida como um nível acima da *cloud computing*, pois seus recursos são provisionados dinamicamente por diferentes provedores. Assim, o *Sky computing* consiste em uma camada de gerenciamento de ambientes em *cloud computing*, oferecendo capacidade de armazenamento variável com suporte dinâmico para demandas em tempo real.

O *Sky computing* também pode ser encontrada sob as denominações de *multicloud* [37], *cross-cloud* [38], nuvens federadas [39] ou *inter-clouds* [40]. Essa diferença de nomenclatura pode estar relacionada à forma como as arquiteturas lidam com o escalonador de recursos, pois esse componente pode ser intrínseco ao provedor, ou um serviço externo fornecido por um *middleware*, por exemplo. Existem também diferenças quanto à forma de integração entre as nuvens participantes da camada de *Sky computing*, bem como quanto ao conhecimento da existência dessa camada pelos recursos em seus respectivos provedores.

Assim sendo, uma característica importante de um ambiente de *Sky computing* é a consolidação de recursos, cuja finalidade é oferecer uma visão única de todos os elementos de cada provedor de *cloud computing* no *pool*. Para atuar nesse processo contínuo de integração, descoberta e consolidação de recursos, é possível utilizar ferramentas denominadas orquestradores de *cloud computing* (do inglês, *Cloud Orchestrators*) [41] como: Terraform [42], Cloudify [43] e Heat [44], por exemplo.

Nesse contexto, nota-se que a *cloud computing* e, consequentemente, a *Sky computing* são modelos de entrega de poder computacional em forma de serviço que possuem características que favorecem o processamento de grandes volumes de dados, tais como a elasticidade para o provisionamento de recursos; o alto poder computacional obtidos por meio do compartilhamento de recursos; o amplo acesso que permite uma rápida comunicação; a obtenção de recursos de acordo com a demanda; e, por fim, a tarifação baseada apenas nos recursos que foram utilizados.

Todavia, ambos os conceitos também possuem algumas limitações. A principal delas está relacionada com a conectividade entre a *cloud computing* e os dispositivos finais dos usuários. Isso ocorre porque os provedores de *cloud computing* pública são suportados por grandes *datacenters* em todo o mundo, mas não distribuídos o suficiente para estar perto do usuário final. Essa realidade resulta na degradação da Qualidade de Serviço (QoS), que

não é adequada para solicitações de serviço do tipo "*time-critical*", os quais são sensíveis à latência [45]. Isso ocorre porque a largura de banda da rede tem se tornado o gargalo da *cloud computing* dada a enorme quantidade de dados que precisa trafegar entre os grandes *datacenters* e o usuário final [46].

Assim, superar as limitações da *cloud computing* beneficiará casos de uso específicos, como veículos conectados, cidades inteligentes, Internet das Coisas (IoT) e realidade virtual [4]. Para tanto, alguns paradigmas e tecnologias foram propostos nos últimos anos, conforme será apresentado nas próximas seções.

2.3 *Fog Computing*

O termo *fog computing* surgiu a partir da analogia com o termo usado na meteorologia no qual a névoa é uma nuvem próxima ao solo. Assim sendo, a *fog computing* foi apresentada como um paradigma capaz de realizar tarefas mais próxima dos usuários e dispositivos finais [45] parte das tarefas que até então eram executadas exclusivamente na *cloud computing*.

A primeira definição para *fog computing* foi apresentada em Bonomi *et al.* [23], como sendo "uma plataforma altamente virtualizada que fornece serviços de computação, armazenamento e rede entre dispositivos finais e *datacenters* de *cloud computing* tradicionais, tipicamente, mas não exclusivamente localizados na borda da rede".

A definição inicial de *fog computing* foi, posteriormente, expandida e revisada por vários pesquisadores. Para Vaquero *et al.* [47] e Yi *et al.* [48], a *fog computing* é um cenário onde um grande número de dispositivos descentralizados e heterogêneos se comunicam, e cooperam entre si e com a rede para realizar tarefas (tais como, armazenamento e processamento) sem intervenções de terceiros. Essas tarefas podem oferecer suporte a funções básicas de rede ou serviços, e aplicativos executados em um ambiente de área restrita.

Na definição apresentada por Dastjerdi *et al.* [25], a *fog computing* é um paradigma de computação distribuída que estende fundamentalmente os serviços fornecidos pela *cloud computing* até a borda da rede. Ele suporta mobilidade, recursos de computação, protocolos de comunicação, heterogeneidade de interface, integração em *cloud computing* e análise de dados distribuída para atender aos requisitos de aplicativos que precisam de baixa latência com uma distribuição geográfica ampla e densa.

Para Naha *et al.* [4] a *fog computing* é uma plataforma distribuída na qual os dispositivos de ponta ou finais, que podem ser virtualizados ou não, farão a maior parte do processamento. Ele reside entre a *cloud computing* e os usuários e a *cloud computing* fará o armazenamento de longo prazo e o processamento não dependente de latência.

Na indústria, a Cisco [49] define que a *fog computing* estende a *cloud computing* para estar mais perto das coisas que produzem e atuam nos dados de IoT. Para a IBM [50], significa a operação nas extremidades da rede em vez de hospedar e trabalhar a partir de uma *cloud computing* central. O Open Fog Consortium [6] afirma que a *fog computing* é uma infraestrutura de computação descentralizada que considera o melhor lugar entre a fonte de dados e a *cloud computing* para distribuir armazenamento, processamento e aplicativos, sendo, portanto, uma complementação e uma extensão dos modelos tradicionais baseados em *cloud computing*.

Finalmente, o NIST [24] define a *fog computing* como um modelo em camadas para permitir o acesso ubíquo a um ambiente compartilhado de recursos de computação escaláveis. O modelo de *fog computing* facilita a implantação de aplicativos e serviços distribuídos, e sensíveis à latência. A *fog computing* minimiza o tempo de resposta de solicitação e fornece, para os dispositivos finais, recursos de computação locais e, quando necessário, conectividade de rede para serviços centralizados.

Em todas as definições é possível notar que a *fog computing* está intimamente ligada à existência da *cloud computing*, uma vez que a *fog computing* por si só não pode substituir a *cloud computing* completamente, pois ela é necessária para lidar com problemas de dados grandes ou complexos [51]. Assim sendo, a *fog computing* é adequada para ser usada quando a *cloud computing* não atende ao limite de tempo, limitações de largura de banda ou requisitos de latência.

Considerando as similaridades e as diferenças dos conceitos apresentados, nota-se que ainda há divergências na literatura [4,6,25]. Assim, a seguinte definição de *fog computing*, proposta neste trabalho, será adotada no desenvolvimento do mesmo: *"Fog computing é uma arquitetura distribuída que utiliza os recursos computacionais de dispositivos localizados entre os usuários finais e a cloud computing, para otimizar o processamento e reduzir o tempo de resposta das aplicações, atendendo demandas que até então não eram possíveis."*

2.3.1 Arquitetura

A arquitetura desempenha um papel central na *fog computing* [46], e o modelo em camadas (ou hierárquico) é o mais amplamente encontrado na literatura por ser considerado a melhor maneira de representá-la [4]. Contudo, existem outras propostas tais como a arquitetura do Open Fog Consortium [6], a denominada Cisco-Bonomi [52], além de outras indicadas para casos de uso específicos, tais como sistemas de monitoramento de aplicações de saúde [53], cidades inteligentes [54], veículos conectados [55], energia [56] e redes, sendo que esta última possui diversas variações, tais como *Fog RAN* (F-RAN) [45], *Wireless*

Sensor Networks (WSN) [57], *Software Defined Networking* (SDN) [58] e *Network Function Virtualization* (NFV) [59].

Ainda assim, a arquitetura de três camadas é a mais utilizada para representar a *fog computing* [60–62]. Contudo, também é possível encontrar trabalhos que apresentem arquiteturas com quatro [63], cinco [64] ou até seis camadas [65], como sintetizado na Figura 2.1.

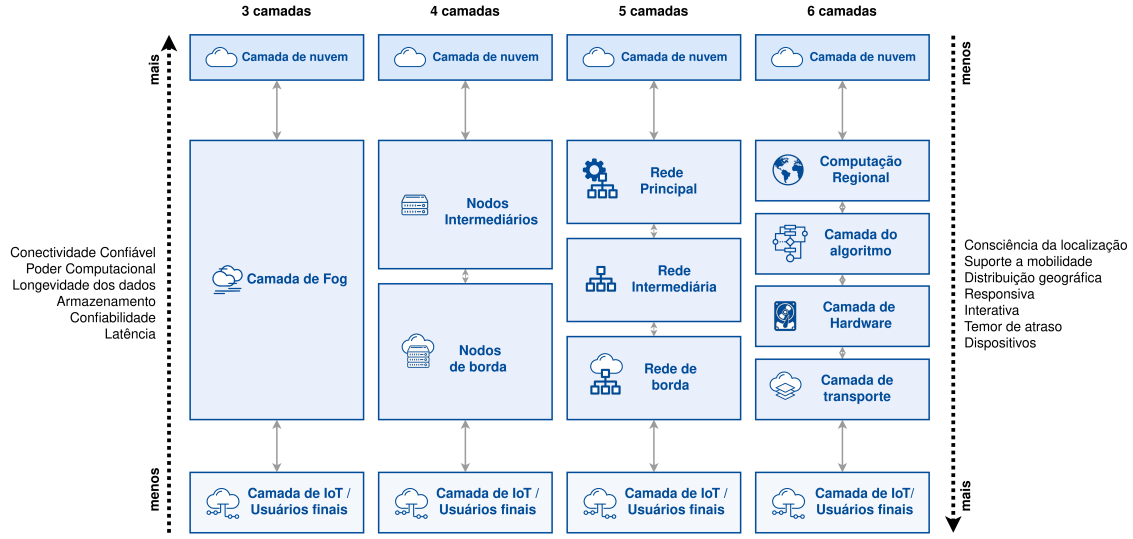

Figura 2.1: Variação no número de camadas da arquitetura de *fog computing* [1].

Como apresentado na Figura 2.1, embora exista uma variação no número de camadas, o que ocorre, na verdade, é apenas uma separação mais detalhada dos componentes da Camada de *Fog*, sendo que as Camadas de Dispositivos/Usuários Finais e *cloud computing* estão presentes em todas as propostas apresentadas. Isso reforça que a arquitetura de três camadas é a mais aderente para representar a *fog computing* [66].

Diante do exposto, para o desenvolvimento deste trabalho também será utilizada uma arquitetura de três camadas, conforme apresentado na Figura 2.2. No entanto, diferentemente de outras propostas que utilizaram arquiteturas hierárquicas com bordas rígidas delimitando cada uma das camadas, este trabalho utiliza bordas suaves, uma vez que os dispositivos da *fog computing* podem ser encontrados não apenas na borda, próximos aos dispositivos IoT, mas também próximos aos dispositivos de *cloud computing*, como roteadores e *switches* de telecomunicações, por exemplo.

A Camada de IoT/Usuários Finais é considerada a base da arquitetura e representa todos os dispositivos e usuários que requisitam os serviços a serem processados nas camadas superiores, ou seja, Camada de *Fog* ou Camada de *Cloud*. Já a Camada de *Fog* é intermediária entre a Camada de IoT/Usuários Finais e a Camada de *Cloud*, agregando funcionalidades tais como processamento e filtragem antes dos dados serem transferidos

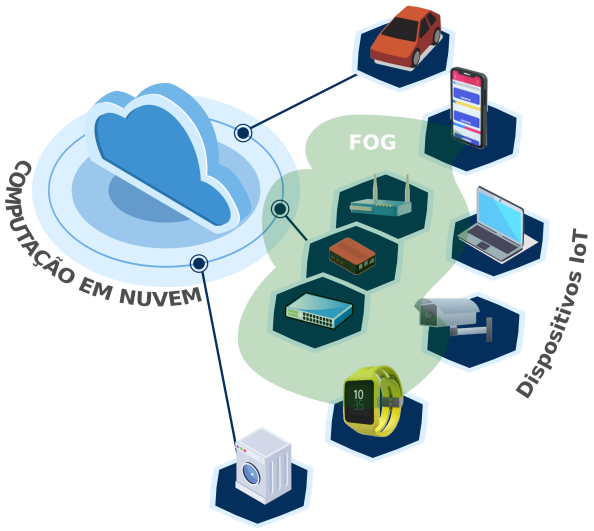

Figura 2.2: Visão geral da arquitetura de *fog computing* [1].

para a *cloud computing* [67]. Os dispositivos que compõem a Camada de *Fog* são comumente denominados *fog nodes*, que serão analisados mais detalhadamente na Seção 3.1.

Por fim, a Camada de *Cloud* é aquela que possui um alto poder computacional para executar todas as requisições que não foram possíveis de serem concluídas pela Camada de *Fog*, devolvendo a reposta para as camadas inferiores. A Camada de *Cloud* é composta por serviços oferecidos pelos provedores de *cloud computing*, os quais podem ser públicos, privados, comunitários ou híbridos.

2.3.2 Características Essenciais

Bonomi *et al.* [23], em 2012, apresentou dez características para o paradigma de *fog computing*, conforme descrito a seguir: i. Localização de borda, local conhecido e baixa latência; ii. Distribuição geográfica; iii. Redes de sensores em grande escala; iv. Grande número de *fog nodes*; v. Apoio à mobilidade; vi. Interações em tempo real; vii. Predomínio do acesso wireless; viii. Heterogeneidade; ix. Interoperabilidade e federação; x. Análises *on-line* e interação com a *cloud computing*.

Todavia, com o crescimento das publicações sobre o tema, estas características vêm sendo revisadas por diversos autores [45, 46]. De todo modo, uma publicação do NIST em 2018 [24] consolidou seis características que são consideradas essenciais para a *fog computing*, a saber:

- **Baixa Latência:** como a implementação ocorre na borda da rede, isto permite que os dados sejam processados em dispositivos mais próximos ao usuário final, ao invés

de enviar dados brutos para serem processados todos na *cloud computing*, acelerando o processamento destes dados, e resultando em redução da latência;

- **Distribuição Geográfica:** os recursos de computação, armazenamento e rede estão localizados perto de coletores de dados, que são distribuídos geograficamente em uma ampla área na borda da rede;
- **Heterogeneidade:** a *fog computing* é amplamente heterogênea tanto nos próprios *fog nodes* quanto em sua infraestrutura de rede. Além disso, os dados são adquiridos de diferentes fontes e formas;
- **Interoperabilidade:** é essencial especificar quais decisões devem ser tomadas pelos *fog nodes* na borda da rede, e quais devem ser tomadas pela *cloud computing*. Além disso, o suporte contínuo de determinados serviços requer a cooperação de diferentes provedores;
- **Interações *real-time*:** as aplicações da *fog computing* envolvem majoritariamente interações em tempo real ao invés de processamento em lote;
- **Escalabilidade:** é necessário ser adaptável e dar suporte a elasticidade dos recursos computacionais, bem como possuir um conjunto de recursos e saber lidar com alterações de carga de trabalho e variações nas condição de rede, por exemplo;
- **Recursos limitados:** deve ser considerado que os componentes da *fog computing*, são, em sua maioria, detentores de baixo poder computacional.

Além destas características, o NIST [24] também indica a predominância da comunicação sem fio entre os componentes e o suporte à mobilidade como características adicionais aos ambientes de *fog computing*. Para [27], a mobilidade pode ser vista sob dois aspectos. Do ponto de vista do cliente, ele precisa ter uma propensão a ter acesso a serviços de qualquer lugar, a qualquer momento, sem qualquer limitação. Já do ponto de vista do *fog node*, a mobilidade indica que eles podem sair da área de cobertura do ambiente da *fog computing*, exigindo a capacidade de reconfigurar rapidamente os recursos e a realizar a migração de objetos de um local para outro. Contudo, a demanda crescente pela computação mais próxima dos usuários e dispositivos fez com que alguns outros paradigmas computacionais, além da *fog computing*, surgissem nos últimos anos, os quais serão apresentados na próxima seção.

2.4 Paradigmas Computacionais Relacionados

Nesta seção são apresentados diversos paradigmas computacionais que têm surgido nos últimos anos, e que estão relacionados a *cloud computing* e a *fog computing*. Além disso,

será apresentada uma comparação entre a *fog computing* e outros paradigmas relacionados, destacando semelhanças e diferenças entre eles, com base nas principais características de *fog computing* (Seção 2.3.2).

2.4.1 Computação de Borda (*Edge Computing*)

A *fog computing* é muitas vezes confundida erroneamente com a Computação de Borda (do inglês, *Edge Computing*), mas existem diferenças importantes entre estes dois conceitos. Enquanto a *fog computing* executa aplicativos em uma arquitetura multicamada, a computação de borda executa aplicativos específicos em um local fixo, ou seja, nos dispositivos de borda.

Além disso, pode-se considerar que a computação de borda deve ser limitada a um pequeno número de dispositivos do usuário final, enquanto a *fog computing* possui um número maior de dispositivos periféricos com arquitetura hierárquica. Portanto, é possível ver que a computação de borda é mais restrita que a *fog computing* [19].

Assim, na computação de borda os dispositivos produzem e consomem dados, participando do processamento, do armazenamento de dados, do processamento e do armazenamento. O dispositivo de borda também é capaz de distribuir solicitações e fornecer serviços em *cloud computing* aos usuários [68].

Além disso, na computação de borda, os dispositivos produzem e consomem dados, no entanto, eles não são capazes de implementar múltiplas aplicações IoT, pois há recursos limitados e isso pode resultar em contenção de recursos e aumentar a latência de processamento. Por outro lado, a *fog computing* pode superar essas limitações integrando perfeitamente dispositivos de borda e recursos de *cloud computing* [25].

Por fim, para o escopo deste trabalho, será considerado que a computação de borda se concentra no nível de dispositivos finais, enquanto a *fog computing* se concentra no nível de uma infraestrutura intermediária mais abrangente entre os dispositivos finais e a *cloud computing*, conforme apresentado na Figura 2.2.

2.4.2 Computação de Borda Multi-acesso (*Multi-access Edge Computing*)

A Computação de Borda Multi-acesso (do inglês, *Multi-access Edge Computing*) que também é referenciada por alguns autores como "Computação Móvel de Borda" (*Mobile Edge Computing*) [69], pode ser definida como uma implementação da computação de borda para trazer capacidades computacionais e de armazenamento para dentro da rede denominada *Radio Access Network* (RAN) [70] e, desta forma, reduzir a latência e melhorar a experiência dos usuários [71].

De acordo com Beck *et al.* [72], a Computação de Borda Multi-acesso pode ser vista como uma implantação colaborativa de telecomunicações e redes computacionais, uma vez que é uma evolução das estações base móveis. Este paradigma opera na borda da rede e permanece funcional, mesmo sem conectividade com a Internet [71].

A Computação de Borda Multi-acesso fornece vários serviços, incluindo IoT, serviços de localização, realidade aumentada, serviço de cache, análise de vídeo e distribuição de conteúdo local [19]. Ela também pode fornecer acesso de baixa latência em tempo real ao conteúdo local, ou armazenando o conteúdo em cache em servidores alocados na própria rede.

No entanto, alguns pontos de atenção não podem ser ignorados, como a necessidade de instalação de um servidor dedicado a este serviço. Com isso, o aumento da demanda de recursos ao longo do tempo também pode se tornar um grande problema de escala [73].

2.4.3 Computação em Nuvem Móvel (*Mobile Cloud Computing*)

O conceito de computação móvel foi proposto por Satyanarayanan em 1996 [74], e representa a computação realizada por meio de dispositivos móveis, portáteis, como *laptops*, *tablets* ou telefones celulares. No entanto, a evolução dos requisitos de dispositivos conectados fez com que a computação móvel por si só não fosse suficiente para enfrentar alguns desafios de computação de nossos dias [19].

Neste contexto, a computação móvel ganhou um complemento valioso à medida que a *cloud computing* foi amadurecendo. Com isso, a combinação da *cloud computing*, da computação móvel e da comunicação sem fio resultou na Computação em Nuvem Móvel [60], melhorando a Qualidade de Experiência (QoE) dos usuários móveis, uma vez que o armazenamento e o processamento de dados ocorrem fora do dispositivo móvel [27].

A computação móvel requer mudanças em algumas características da *cloud computing*, como a existência de uma camada intermediária de baixa latência, a otimização da infraestrutura de *cloud computing* para execução de aplicativos móveis e o descarregamento e execução remota contínuos. A viabilidade da Computação em Nuvem Móvel é baseada em uma rede confiável de ponta a ponta, com largura de banda alta, e isso pode ser alcançado usando máquinas virtuais que devem estar localizados mais próximos dos dispositivos móveis [74].

2.4.4 Computação em Nuvem Móvel *Ad hoc* (*Mobile Ad hoc Cloud Computing*)

O conceito de Computação em Nuvem Móvel, apresentado anteriormente, tem uma natureza abrangente, mas há cenários em que ele pode não ser adequado, por exemplo, onde não há *cloud computing* centralizada ou a infraestrutura é insuficiente [19]. Uma rede móvel *Ad hoc* consiste em *fog nodes* que formam uma rede temporária e dinâmica por meio de protocolos de roteamento e transporte, construindo uma forma descentralizada de rede [75].

Assim sendo, a Computação em Nuvem Móvel *Ad hoc* consiste em um conjunto de dispositivos com alta capacidade computacional que estão mais próximos do usuário, conectados em uma rede local [76]. Este ambiente computacional de baixo custo é implantado em uma rede onde todos os nós mantêm cooperativamente a rede.

Uma das principais motivações para a Computação em Nuvem Móvel *Ad hoc* é abordar situações na Computação em Nuvem Móvel convencional para as quais a conectividade com ambientes de *cloud computing* não é viável, como uma conexão de rede intermitente ou a completa ausência de uma conexão [77]. Além disso, o hardware utilizado, o método de acesso ao serviço, e a distância dos usuários também são outras diferenças entre os dois paradigmas. Por fim, definiu-se a Computação em Nuvem Móvel *Ad hoc* como um conjunto de dispositivos próximos aos demandantes implementada por meio de uma rede dinâmica, endereçando situações no MCC para as quais a conectividade com ambientes de *cloud computing* não é possível.

2.4.5 Computação em Neblina (*Mist Computing*)

A Computação em Neblina (do inglês, *Mist Computing*) pode ser definida como uma forma leve e rudimentar de *fog computing* [78]. Considerando a arquitetura apresentada na Figura 2.2 (Seção 2.3), a computação em neblina aproxima a Camada de *Fog* aos dispositivos finais inteligentes, usando recursos de hardware para isso, tais como microcomputadores e microcontroladores, residindo diretamente na borda da rede [24].

Em dispositivos móveis, a transferência de dados pode utilizar muita energia da bateria e, por este motivo, é desejável que os dados possam ser processados, pré-condicionados e otimizados antes de serem transferidos e armazenados. O paradigma de computação em neblina pode aumentar a autonomia das soluções móveis, resultando em uma transferência de dados muito menor, consumindo menos energia [79] além de diminuir ainda mais a latência [80].

Para Yousefpour *et al.* [19], a Computação em Neblina também pode ser referenciada como "*IoT computing*" ou ainda "*things computing*". Para o desenvolvimento deste tra-

balho, a Computação em Neblina será indicada como o primeiro local onde a computação ocorre na integração entre IoT, a *fog* e a *cloud computing*.

2.4.6 *Cloudlet*

Em 2009, a Carnegie Mellon University introduziu o conceito de "*datacenter in a box*", denominado *Cloudlet* [81]. Eles são compostos de infraestrutura de Internet descentralizada e amplamente dispersa, na qual os equipamentos próximos podem aproveitar seus ciclos de computação e recursos de armazenamento [82].

O dispositivo móvel, agindo como um *thin client*, pode realizar tarefas computacionais por meio de uma rede sem fio para um *cloudlet*, implantado a um salto de distância [83], ou seja, na próxima camada de rede. No entanto, a mobilidade pode ser um problema, uma vez que se um dispositivo sair do alcance de um *cloudlet*, ele deve alternar para a *cloud computing* distante, ou no pior cenário, confiar apenas em seus recursos [84].

Outra característica importante dos *cloudlets* é que eles suportam serviços locais para clientes móveis dividindo tarefas entre nós de *cloudlet* próximos aos dispositivos móveis [85]. Por outro lado, a *fog computing* oferece uma alternativa mais genérica que suporta nativamente grandes quantidades de tráfego, e permite que os recursos estejam em qualquer lugar entre as camadas da arquitetura (Figura 2.2) [86].

2.4.7 Computação de Orvalho (*Dew Computing*)

No espectro da *cloud computing*, o paradigma de organização de software x hardware de computador local é conhecido como computação de orvalho. Isto ocorre porque os recursos computacionais locais fornecem funcionalidades independentes, ou seja, que funcionam perfeitamente com ou sem uma conexão com a Internet, além de ser colaborativa com serviços em *cloud computing* [87]. Com isso, estas aplicações trabalham de forma independente e, quando uma conexão com a Internet fica disponível, ele sincroniza dados com a *cloud computing*.

Definimos, portanto, a Computação de Orvalho como o paradigma computacional no qual os recursos computacionais locais fornecem funcionalidades independentes, trabalhando com ou sem conexão com a internet, aproveitando plenamente o potencial dos recursos computacionais locais e serviços em *cloud computing*. A natureza dos aplicativos da Computação de Orvalho pode ser descrita pela independência, que indica que seus aplicativos são inerentemente distribuídos; e pela colaboração, que indica que os aplicativos são inerentemente conectados [88].

A Computação de Orvalho leva os conceitos de serviço, armazenamento e rede, e vai além, definindo uma subplataforma, baseada em microsserviços e distribuindo vertical-

mente sua hierarquia computacional [89]. O paradigma de Computação de Orvalho facilita o uso de recursos de hardware, pois eles estão conectados a uma rede, abrangendo uma ampla gama de tecnologias [90].

2.4.8 Comparação entre *Fog* e Paradigmas Relacionados

Com base nas principais características da *fog computing*, na Seção 2.3.2, uma comparação dos paradigmas mencionados é apresentada na Tabela 2.1. A ideia é destacar as diferenças e as semelhanças entre os paradigmas e ajudar a esclarecer a forma como eles se comparam à *fog computing*.

Tabela 2.1: Comparação entre a *fog computing* e os paradigmas relacionados [2].

Paradigma	**Baixa Latência**	**Distribuição Geográfica**	**Heterogeneidade**	**Interoperabilidade**	**Tempo Real**	**Escalabilidade**
Sky	-	-	✓	✓	-	✓
Cloud	-	-	✓	✓	-	✓
Fog	✓	✓	✓	✓	✓	✓
Borda	✓	✓	✓	✓	✓	✓
Móvel	✓	✓	-	✓	✓	-
Borda Multi-Acesso	-	-	✓	✓	-	-
Borda ad hoc	-	✓	-	✓	-	-
Neblina	-	✓	✓	-	✓	-
Cloudlet	✓	✓	-	✓	✓	✓
Orvalho	-	-	-	✓	✓	-

Analisando a Tabela 2.1, é possível notar que a Computação de Borda é o paradigma mais próximo da *fog computing*. Por outro lado, a Computação de Nuvem Móvel, a Computação de Nuvem Móvel Ad hoc e a Computação de Orvalho, compartilham poucas características da *fog*, afastando-se do escopo desse paradigma.

Por fim, a Figura 2.3 mostra uma visão deste trabalho sobre a localização desses paradigmas em relação à arquitetura de três camadas, apresentada na Seção 2.3.1. Para esta proposta, foi considerada a integração entre IoT e *Cloud*. A *fog computing* está localizada entre a *cloud computing* e a Camada de Dispositivos/Usuários Finais e, embora os outros paradigmas descritos sejam adequados para alguns casos de uso específicos, a *fog computing* tem sido vista como uma forma mais geral de computação devido ao seu escopo de definição abrangente [19].

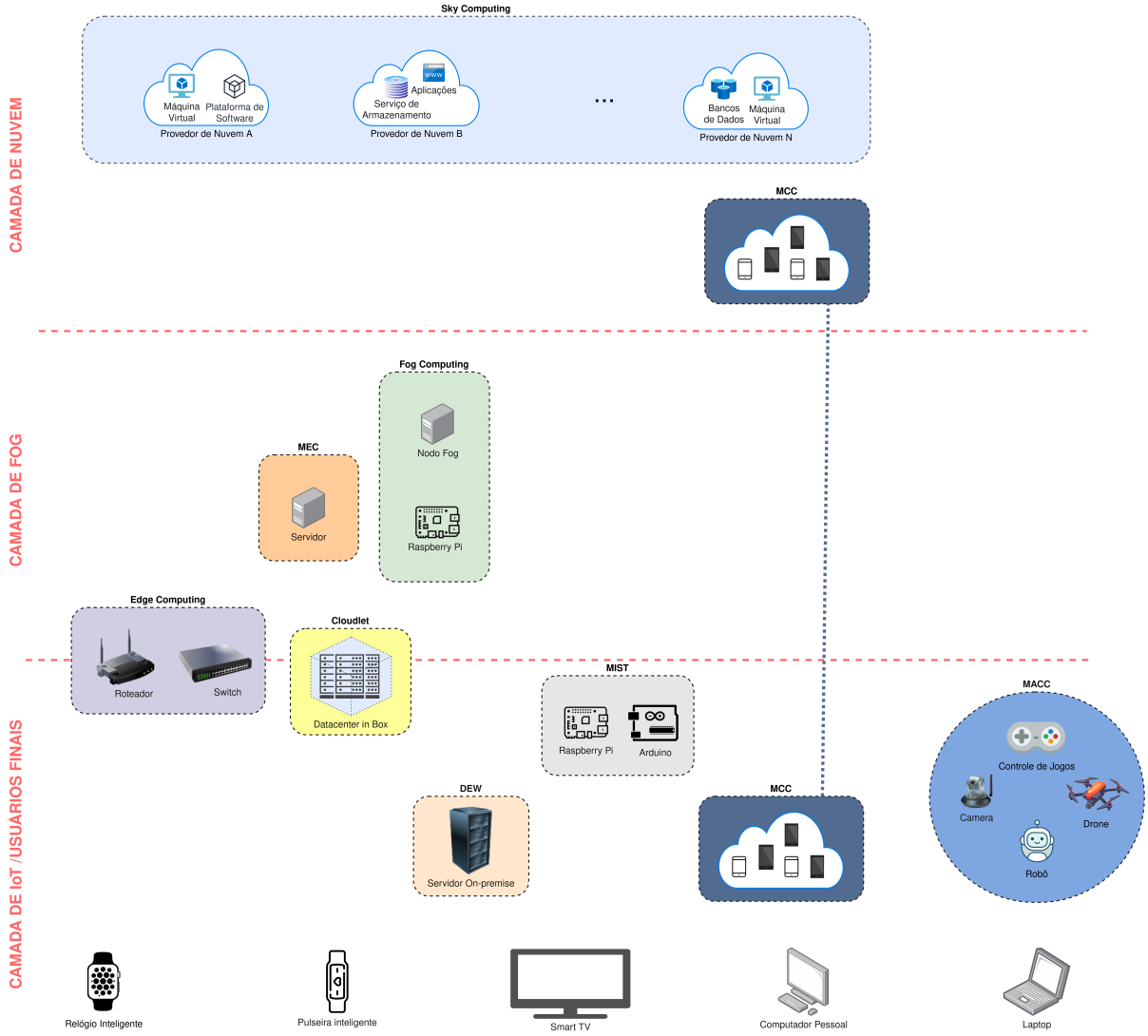

Figura 2.3: Localização dos paradigmas computacionais na integração IoT-*Cloud* [2].

O resultado desta análise entre a *fog computing* e os demais paradigmas computacionais relacionados gerou o artigo entitulado "*From the Sky to the Ground: Comparing Fog Computing with Related Distributed Paradigms*" [2], publicado no *12th International Conference on Cloud Computing and Services Science (CLOSER 2022)*, a qual é um evento com Qualis A2, em Abril de 2022 (Anexo I).

2.5 Considerações Finais

Este capítulo abordou o paradigma *Fog Computing* como uma evolução de *Cloud Computing*, proporcionando a descentralização dos recursos computacionais ao aproximá-los dos usuários na borda da rede. Foram analisados a arquitetura de *Fog Computing* e suas características essenciais. Também foram apresentadas as distinções entre *fog* e ou-

tros paradigmas computacionais, como *Sky Computing*, *Edge*, MEC, MCC, MACC, *Mist*, *Cloudlet* e *Dew Computing*.

A comparação entre paradigmas similares ao *fog* é essencial para expandir as possibilidades de pesquisa, pois a adaptação de soluções entre paradigmas com características semelhantes, como *Edge* e *MEC*, tende a ser mais acessível. Por outro lado, compreender as diferenças significativas entre esses paradigmas evidencia os desafios na implementação de soluções dentro do contexto de *Fog Computing*.

Capítulo 3

Gerenciamento de Recursos em Fog Computing

Este capítulo apresenta os principais aspectos sobre o gerenciamento de recursos em *fog computing*. A Seção 3.1 traz informações sobre os recursos computacionais deste paradigma, enquanto a Seção 3.2 detalha as etapas de estimativa, descoberta, alocação, monitoração e orquestração que compõem o gerenciamento de recursos.

3.1 Recursos na *Fog Computing*

Manvi e Shyam [91], consideram que "um recurso é qualquer componente físico ou virtual disponível em um sistema de computação". Assim, são exemplos de recursos computacionais: CPU, memória, dados, elementos de rede, sensores, sistemas operacionais, sistemas de virtualização, entre outros [92]. Especificamente para a *fog computing*, é comum a utilização do termo *fog node* para indicar um conjunto de recursos computacionais, conforme apresentado na Tabela 3.1.

Conforme pode ser observado nas definições apresentadas na Tabela 3.1, em *fog computing* os dispositivos, de um modo geral, possuem menor capacidade computacional, são mais heterogêneos [19] e estão geograficamente distribuídos [3]. Estas características a diferenciam de outros paradigmas computacionais, como por exemplo a *cloud computing*, na qual os recursos computacionais possuem alta capacidade de processamento e de armazenamento, além de serem mais lineares em relação à arquitetura e compatibilidade, e ainda estarem localizados em *datacenters* instalados em pontos específicos do mundo [96].

Na literatura é possível encontrar publicações que fizeram uma revisão sistemática sobre diversos aspectos da *fog computing*, incluindo aqueles que se aprofundaram na análise dos *fog nodes*. Entre estes trabalhos, para Hong e Varghese [3] os recursos são classificados de maneira macro em hardware e software. Os recursos de hardware são compostos por

Tabela 3.1: Definições de *fog node*.

Autores	Definição de fog node
Yi *et al.* [48]	Instalações ou infraestruturas que podem fornecer recursos para serviços na borda da rede.
Devi *et al.* [93]	Um dispositivo físico que fornece um auxílio na implantação de *fog computing*.
Cisco Corp. [94]	Dispositivo físico no qual o paradigma *fog computing* é implantado.
Sarabia *et al.* [95]	É o componente da infraestrutura de *fog computing*. Dispositivos de rede (por exemplo, *gateways*) que, além da função de rede simples, têm recursos de armazenamento e processamento.
Marin *et al.* [7]	Entidades de *fog computing* distribuídas que permitem a implantação de serviços de *fog* e formados por pelo menos um ou mais dispositivos físicos com recursos de processamento e detecção (por exemplo, computador, telefone celular, dispositivo de borda inteligente, carro, sensores de temperatura, etc.).
NIST [24]	Componentes físicos ou virtuais que são fortemente acoplados aos dispositivos dos usuários finais ou redes de acesso, e fornecem recursos de computação para esses dispositivos.

dispositivos de computação e dispositivos de rede, enquanto os recursos de software são compostos por sistemas de virtualização e por redes virtualizadas, conforme apresentado na Figura 3.1. Com isso, Hong e Varghese [3] deixam evidente a necessidade dos dispositivos da *fog* terem não apenas recursos físicos disponíveis, mas também recursos virtuais que permitam a operar e gerenciar os recursos de hardware. Já o trabalho apresentado por Naha *et al.* [4] traz uma taxonomia que classifica os dispositivos em três categorias, sendo: dispositivos de IoT, dispositivos de processamento e dispositivos *gateway*, conforme apresentado na Figura 3.2.

O trabalho de Naha *et al.* [4] se torna mais abrangente por, além de apresentar uma classificação para os recursos computacionais da *fog computing*, também apresentar requisitos de infraestrutura (processamento, armazenamento, rede e memória) e de rede (conexão e mobilidade), conforme exibido na Figura 3.2. Esta mesma classificação, ainda que não tão detalhada, foi também proposta por Buyya *et al.* [60]. No artigo [7], os *fog nodes* são classificados apenas em duas categorias: *smart*, para aqueles que possuem alguma capacidade de processamento e armazenamento; e *dumb* para os dispositivos que possuem capacidades computacionais limitadas, tais como sensores e atuadores.

De toda forma, ainda não é encontrada na literatura uma definição bem aceita de quais são, de fato, os dispositivos que compõem a *fog computing*, uma vez que todas as definições apresentadas na Tabela 3.1 são evasivas neste sentido. Para contornar esta situação, uma solução foi buscar por esta definição em abordagens similares. Com isso, foram encontradas definições no âmbito da IoT que podem ser aplicadas também à *fog*

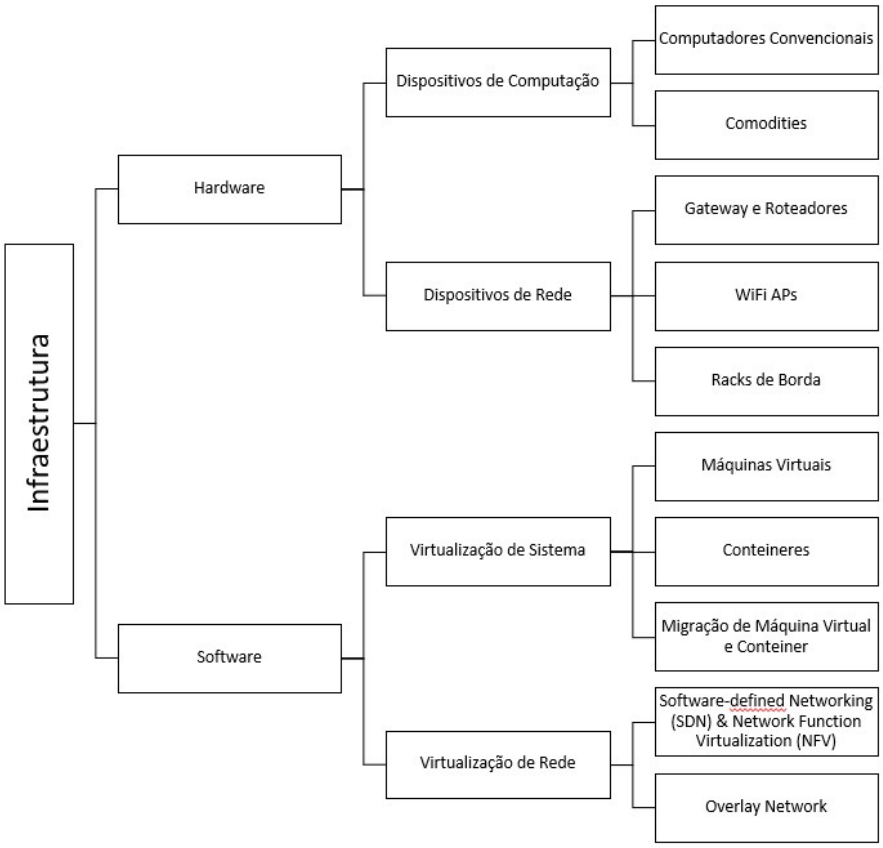

Figura 3.1: Taxonomia de *fog nodes* proposta por Hong e Varghese [3].

computing. Ao longo do tempo, algumas organizações buscaram elaborar definições e arquiteturas de referência para IoT, incluindo classificações quanto aos dispositivos computacionais que a compõem, tais como *European Telecommunications Standards Institute Technical Committee* (ETSI TC) [97], *Reference Architecture Model Industrie 4.0* (RAMI 4.0) [98] e *International Telecommunication Union* (ITU-T) [5].

No universo de IoT, alguns dispositivos coletam informações e as fornecem às redes de informação e comunicação para processamento posterior, enquanto outros são capazes de executar operações com base nas informações recebidas das redes de informação e comunicação. Baseado nisso, os dispositivos são classificados quanto a capacidade de comunicação (como os dispositivos podem se conectar às redes de comunicação) e de processamento (como os dispositivos podem realizar tarefas computacionais e executar algoritmos). A arquitetura de referência apresentada pela ITU-T [5], por exemplo, utiliza estes atributos para criar a classificação apresentada na Figura 3.3, a qual é dividida em quatro categorias:

- **Baixo Processamento e Baixa Conectividade (BPBC):** este tipo de dispositivo não possui recursos de processamento suficientes para tomar decisões ou exe-

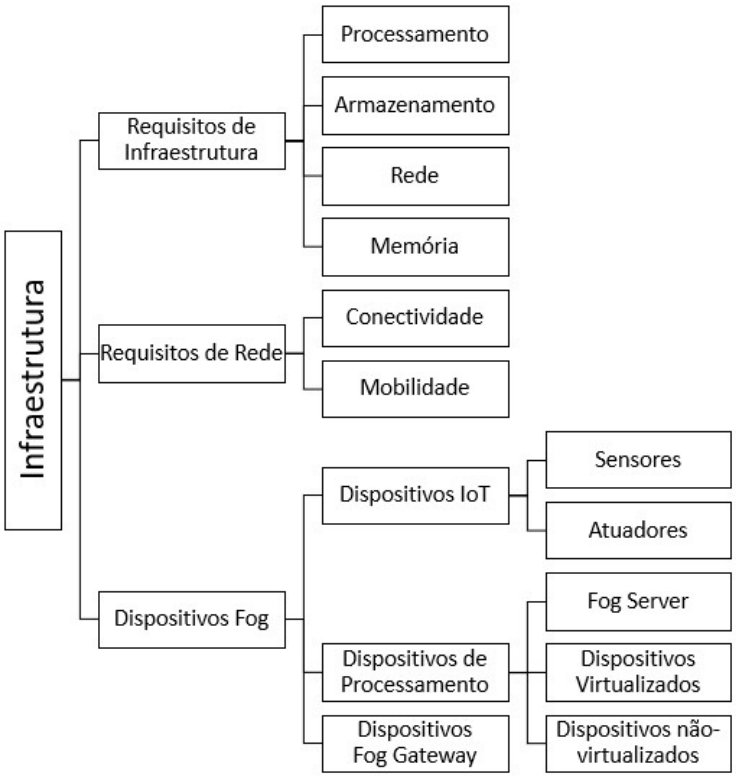

Figura 3.2: Taxonomia de *fog nodes* proposta por Naha *et al.* [4].

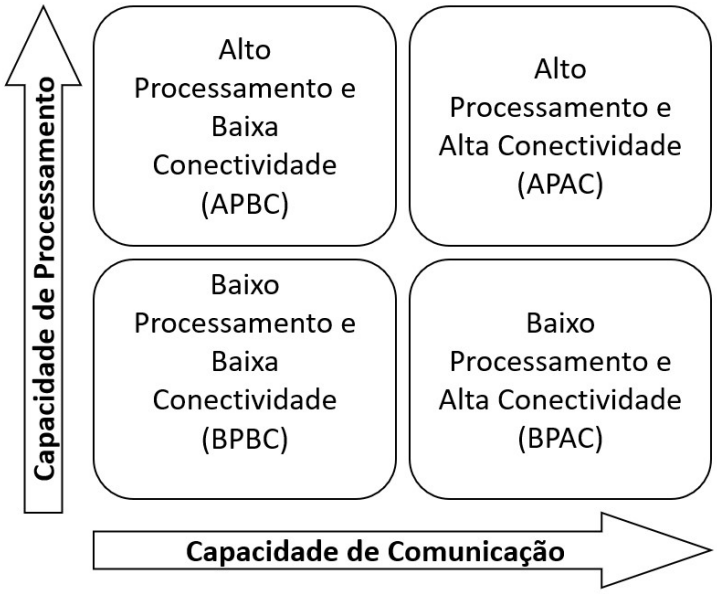

Figura 3.3: Classificação dos dispositivos de acordo com as definições da ITU-T [5].

cutar algoritmos complexos e também não se conecta diretamente a uma rede de comunicação. Esses dispositivos devem contar com outro elemento de rede [99];

- **Baixo Processamento e Alta Conectividade (BPAC):** embora estes dispositivos também tenham pouco poder de processamento, eles podem se comunicar diretamente com aplicativos ou serviços em *cloud computing* pela Internet. Roteadores, *gateways*, *set-top boxes* e *access points* são exemplos desse tipo de dispositivo [100];
- **Alto Processamento e Alta Conectividade (APAC):** além de ter a capacidade de executar aplicativos e algoritmos mais complexos, e ter uma conexão direta com a Internet, eles também podem coordenar diretamente outros dispositivos. Alguns exemplos incluem nós coletores (por exemplo, Raspbery Pi), dispositivos computacionais de baixo custo (por exemplo, *smartphones*) e dispositivos computacionais de ponta (por exemplo, computadores pessoais) [99];
- **Alto Processamento e Baixa Conectividade (APBC):** a combinação de alto processamento e baixa conectividade não é um cenário usual, uma vez que um dispositivo que possui alto poder de processamento também terá alta capacidade de conectividade [5].

Além dos recursos de processamento e de comunicação, é necessário indicar quais são os requisitos funcionais esperados para um *fog node*. Nesse sentido, no documento proposto pelo NIST [24], os *fog nodes* precisam oferecer suporte a, pelo menos, um dos seguintes atributos:

- **Autonomia:** os *fog nodes* podem operar de forma independente, tomando decisões locais;
- **Heterogeneidade:** os *fog nodes* possuem diferentes formatos e podem ser implantados em uma ampla variedade de ambientes;
- **Agrupamento hierárquico**: os *fog nodes* suportam estruturas hierárquicas, com diversas camadas fornecendo diferentes subconjuntos de funções de serviço;
- **Gerenciabilidade:** os *fog nodes* são gerenciados e orquestrados por sistemas que podem executar rotinas automaticamente;
- **Programabilidade:** os *fog nodes* são programáveis em vários níveis, por vários intervenientes - como operadores de rede, especialistas de domínio, fornecedores de equipamento ou usuários finais.

Já no documento de referência de arquitetura proposto pelo OpenFog Consortium [6], um *fog node* é composto por oito aspectos, conforme apresentado na Figura 3.4. As características essenciais de um *fog node*, propostas por [6], e retratadas na Figura 3.4, são descritas a seguir:

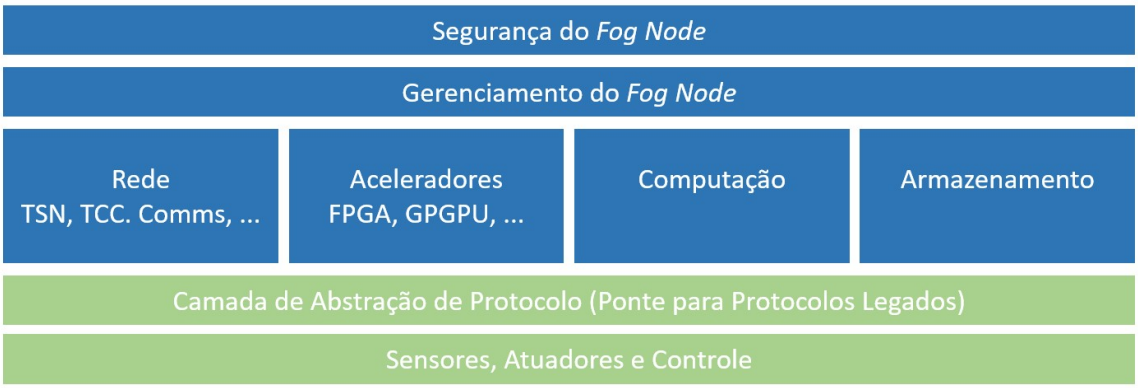

Figura 3.4: Estrutura do *fog node* proposta pelo OpenFog Consortium [6].

- **Segurança:** a segurança do *fog nodes* é essencial para a segurança geral do sistema. Isso inclui proteção para interfaces, computação, software, etc. Em muitos casos, um *fog node* atuará como um *gateway* para sensores e atuadores legados para funções de *fog computing* de nível superior e, portanto, pode atuar, também, como um *gateway* de segurança. É importante observar que a segurança do *fog node* é mostrada tanto como uma perspectiva vertical quanto como um requisito horizontal para esta visualização. Este é um conceito importante, pois a segurança deve ser considerada em todos os níveis, do hardware ao software;
- **Gerenciamento:** um *fog node* deve suportar as interfaces de gerenciamento, fornecidas pelo *fog node* que está sendo gerenciado. As interfaces de gerenciamento permitem que os agentes do sistema de nível superior vejam e controlem o *fog node* de nível mais baixo. O mesmo protocolo de gerenciamento pode ser usado em muitas interfaces físicas diferentes;
- **Rede:** deve fornecer escalabilidade, disponibilidade e flexibilidade exigidas pelo QoS além de priorizar dados críticos ou sensíveis à latência. Dependendo do cenário de implantação, os *fog nodes* provavelmente serão o próprio elemento de rede, como um ponto de acesso, um *gateway* ou um roteador;
- **Aceleradores:** alguns *fog nodes* podem ser alocados para análises aprimoradas, que não podem ser atendidos por uma CPU convencional. Nesses casos, os módulos aceleradores serão configurados próximos aos módulos do processador (ou integrados a eles) para fornecer rendimento computacional suplementar. Assim, são exemplos de aceleradores: *Graphics Processing Unit* (GPUs) e *Field Programmable Gate Arrays* (FPGAs);
- **Computação:** um *fog node* deve ter recursos de computação de propósito geral para permitir um nível mais alto de interoperabilidade;

- **Armazenamento:** muitos tipos de armazenamento serão necessários em *fog nodes*, à medida que a *fog computing* continua a se desenvolver. Com isso, camadas de armazenamento normalmente vistas em *datacenters*, tais como *RAM Arrays*, *Solid State Drives* e *Fixed Spinning Disks* devem ser suportados em um *fog node*;
- **Sensores e Atuadores:** esses dispositivos baseados em hardware ou software são considerados os elementos de nível mais baixo em IoT. Pode haver várias centenas ou mais destes dispositivos associados a um único *fog node*. Alguns deles são dispositivos "burros", sem qualquer capacidade de processamento significativa, enquanto outros podem ter algumas funções básicas de computação. Esses elementos, geralmente, têm alguma conectividade e incluem protocolos com ou sem fio, como I2C, GPIO, SPI, BTLE, ZigBee, USB e Ethernet, etc;
- **Protocolo de Abstração:** atualmente, muitos dos sensores e atuadores no mercado não são capazes de interagir diretamente com um *fog node*. A camada de protocolo de abstração torna logicamente possível colocar esses elementos sob a supervisão de um *fog node*, para que seus dados possam ser utilizados para análises e funções de sistema de nível superior.

Considerando as definições apresentadas até agora nesta seção, é possível sintetizar que um *fog node* é composto por uma Camada de Hardware na qual são localizados CPU, memória, interface de rede, entre outros componentes físicos; e também por uma Camada de Sistema Operacional que é necessária para a abstração do hardware e execução das aplicações [101], conforme apresentado na Figura 3.5(a).

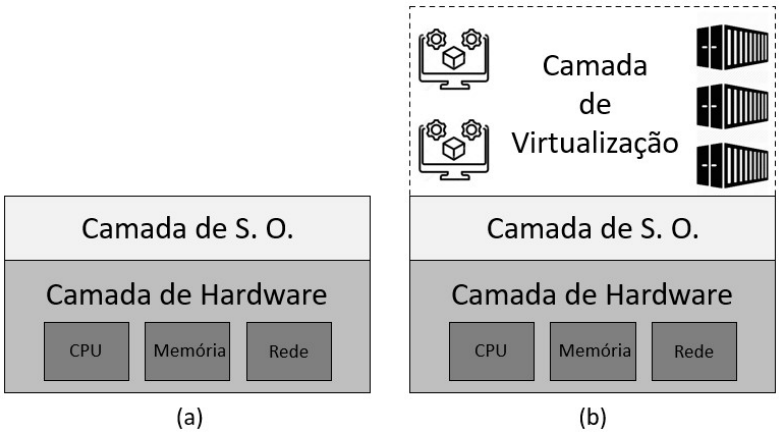

Figura 3.5: Representação para um *fog node* básico (a), e um *fog node* com virtualização (b). Adaptada de [7].

No entanto, uma funcionalidade desejável para o *fog node* é a capacidade de virtualização [102]. A virtualização pode ser definida como "uma abstração de software com a

aparência de um hardware de sistema de computador" [103]. Além disso, os mecanismos de virtualização baseados em hardware estão disponíveis em quase todos os hardwares de processador que seriam usados para implementar plataformas de *fog computing* [6]. Neste sentido, para um melhor aproveitamento dos recursos de hardware disponíveis no "*fog node* básico" (Figura 3.5(a)), é desejável a existência de uma Camada de Virtualização [66, 104], conforme apresentado na Figura 3.5(b).

Todavia, embora a maneira mais comum de entrega de recursos virtuais seja por meio de máquinas virtuais [105], mais recentemente isso tem migrado para o uso de contêineres. Isso porque eles oferecem um mecanismo de isolamento mais aderente a um ambiente de *fog computing* [6] justamente por realizarem a virtualização na Camada de Sistema Operacional e não mais na Camada de Hardware, como era utilizado por outros modelos de virtualização [106]. Outras formas de virtualização, tais como o *unikernel* [107], que é um tipo de virtualização em nível de aplicação ainda mais enxuto que os contêineres, também tem surgido em publicações recentes sobre *fog computing* [108–110]. Um diagrama que compara as três técnicas de virtualização (máquinas virtuais, contêineres e *unikernel*) é apresentado na Figura 3.6.

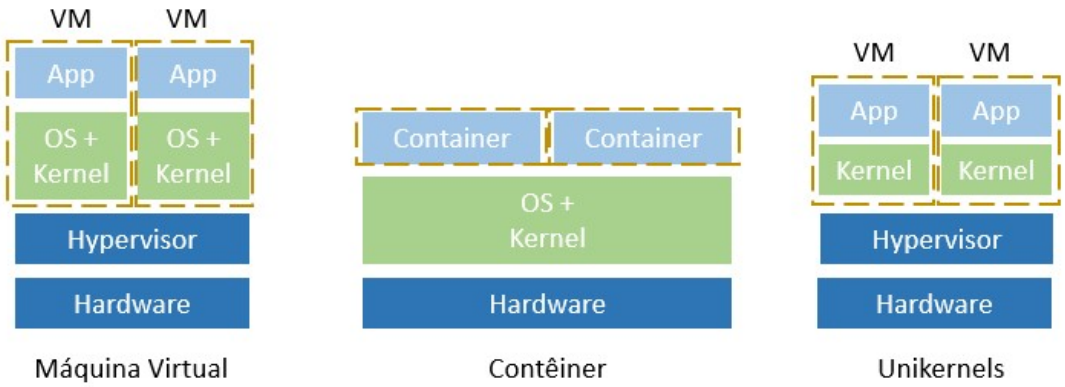

Figura 3.6: Representação de arquitetura de máquina virtual, contêiner e *unikernel*. Adaptada de [8] [9].

Por fim, considerando a abrangência e a diversidade de dispositivos que podem fazer parte de uma ambiente de *fog computing* (conforme apresentado nesta seção), e por ainda não haver uma padronização adotada pela academia, a seguinte definição de *fog node* é proposta nesta tese: "*fog node é qualquer dispositivo de hardware que possua capacidades computacionais, de comunicação e de virtualização que esteja em um ambiente de fog computing*". Entretanto, do ponto de vista computacional, a capacidade de virtualização é considerada importante para viabilizar a operacionalização do recurso computacional, permitindo que a execução de aplicações ocorra mesmo na Camada de *Fog*.

Com base na definição de *fog node* proposta e considerando uma abordagem mínima do ponto de vista computacional, a Figura 3.7 apresenta uma classificação proposta neste

trabalho. Essa classificação é apresentada por meio de um diagrama de cebola no qual é possível notar que o núcleo é composto por sensores e atuadores que fazem parte da Camada Dispositivos/Usuários Finais (conforme mostrado na Figura 2.2). A medida que as camadas da cebola se expandem, elas agregam capacidades de hardware, que devem ser, pelo menos, CPU, memória e rede; também podem incluir aceleradores como GPU, capacidades de software, como sistema operacional; e, finalmente, capacidades de virtualização, tais como máquina virtual, contêiner ou *unikernel*. Com isso, tem-se que quanto mais camadas o *fog node* abranger, mais favorável será para aplicações e serviços de computação. Assim, é possível atender a diversos outros requisitos, tais como segurança, gerenciamento e programabilidade.

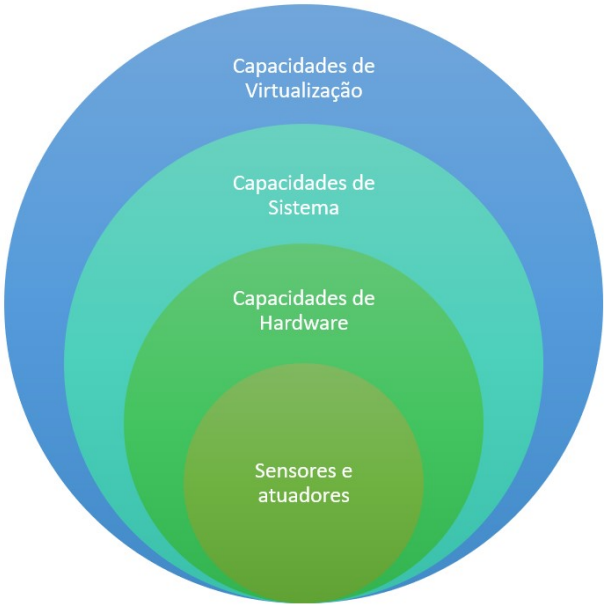

Figura 3.7: Classificação do *fog node* sob a perspectiva computacional [9].

Além das capacidades computacionai (tais como CPU, armazenamento, e memória), os *fog nodes* também possuem características comportamentais (por exemplo, disponibilidade, mobilidade e confiabilidade) [32]. Em um ambiente dinâmico como o de *fog computing*, estas características comportamentais são relevantes uma vez que, embora eles não sejam impeditivos para a execução da aplicação, eles podem impactar no seu desempenho (de tempo, de custo, etc.).

A análise de *fog node* sob a perspectiva computacional, apresentada nesta seção, resultou em um artigo publicado no *22nd International Conference on Internet Computing and IoT (ICOMP 21)*, em julho de 2021, com o título "*Computational Perspective of the Fog Node*" [9] (Anexo II).

3.2 Gerenciamento de Recursos

O gerenciamento de recursos é uma disciplina relevante em diversas áreas de pesquisa porque visa o uso otimizado dos recursos disponíveis [111]. Este tópico também é amplamente discutido na área de Computação, pois os recursos computacionais são muitas vezes limitados e, portanto, devem ser bem utilizados. Nos últimos anos, o paradigma *fog computing* tem sido amplamente estudado tanto pela academia quanto pela indústria [4] e, por isso, é possível encontrar algumas publicações de revisão de literatura sobre este tema. Enquanto algumas dessas pesquisas visam apresentar uma visão geral da computação em *fog computing*, com conceitos e definições mais abrangentes [19,48,61], outras têm um escopo mais limitado. Assim, esta seção tem como objetivo apresentar os principais trabalhos publicados nos últimos anos, especificamente, para o tema de gerenciamento de recursos em *fog computing*. Uma análise comparativa dos trabalhos apresentados está resumida na Tabela 3.2, na qual é possível notar que não há padronização para o processo de gerenciamento de recursos, e que cada autor considera diferentes abordagens e passos para contextualizá-lo.

Tabela 3.2: Trabalhos sobre gerenciamento de recursos em *fog computing* [1].

Artigo	**Ano**	Etapas																								
		Estimativa	**Descoberta**	**Alocação**	**Placement**	**Migração**	**Escalonamento**	**Compartilhamento**	**Otimização**	**Provisionamento**	**Caching**	**Offloading**	**Pré-processamento**	**Coordenação**	**Balanceamento**	**Benchmarking**	**Modeling**	**Monitoração**	**Composição**	**Gerenciamento**	**Detecção**	**Seleção**	**Mapeamento**	**Distribuição**	**Atribuição**
[60]	2017	✓		✓										✓												
[22]	2018	✓	✓	✓	✓	✓	✓	✓	✓																	
[112]	2018	✓		✓																						
[100]	2018				✓	✓		✓		✓	✓	✓	✓	✓												
[26]	2019			✓	✓		✓			✓		✓			✓											
[3]	2019		✓		✓										✓	✓										
[113]	2020		✓																							
[114]	2020		✓		✓										✓	✓										
[115]	2020	✓	✓	✓													✓	✓								
[116]	2020			✓			✓			✓					✓											
[117]	2020						✓			✓	✓	✓			✓											
[118]	2020			✓	✓		✓			✓		✓			✓											
[119]	2020			✓			✓			✓		✓							✓	✓						
[120]	2020			✓			✓			✓					✓			✓			✓	✓	✓			
[121]	2020			✓																						
[27]	2020			✓							✓	✓												✓	✓	
[122]	2020						✓			✓					✓			✓								
[123]	2020			✓					✓																	
[124]	2021			✓	✓		✓		✓	✓																
[125]	2021		✓	✓	✓							✓			✓											
[126]	2021		✓				✓																			
[127]	2021			✓			✓																			
[128]	2021			✓			✓			✓		✓			✓											
[129]	2021			✓					✓																	
[130]	2021						✓																			
Total		4	7	17	8	2	13	2	4	10	3	8	1	2	10	2	1	3	1	1	1	1	1	1	1	1

Ao detalhar as definições de cada uma das publicações para todas as etapas, é possível encontrar um sombreamento entre os conceitos atribuídos a cada uma delas. Por exem-

plo, para Nath *et al.* [112] a alocação de recursos é quando "diferentes recursos devem ser alocados adequadamente para diferentes dispositivos", mas Mijuskovic *et al.* [125] consideram isso como a etapa de *Placement*. Infelizmente, isso ocorre com a maioria das definições. Para resolver essa confusão entre os conceitos e as diferentes etapas que cada autor atribuiu ao processo de gerenciamento de recursos, foi realizada uma análise detalhada entre os 25 artigos apresentados na Tabela 3.2. Com esta análise é possível agrupar todas as 24 etapas apresentadas na Tabela 3.2 em apenas cinco etapas, a saber: Descoberta, Estimativa, Alocação, Monitoração e Orquestração, e o escopo completo do gerenciamento de recursos em *fog computing* pode ser representado por elas, conforme apresentado na Figura 3.8.

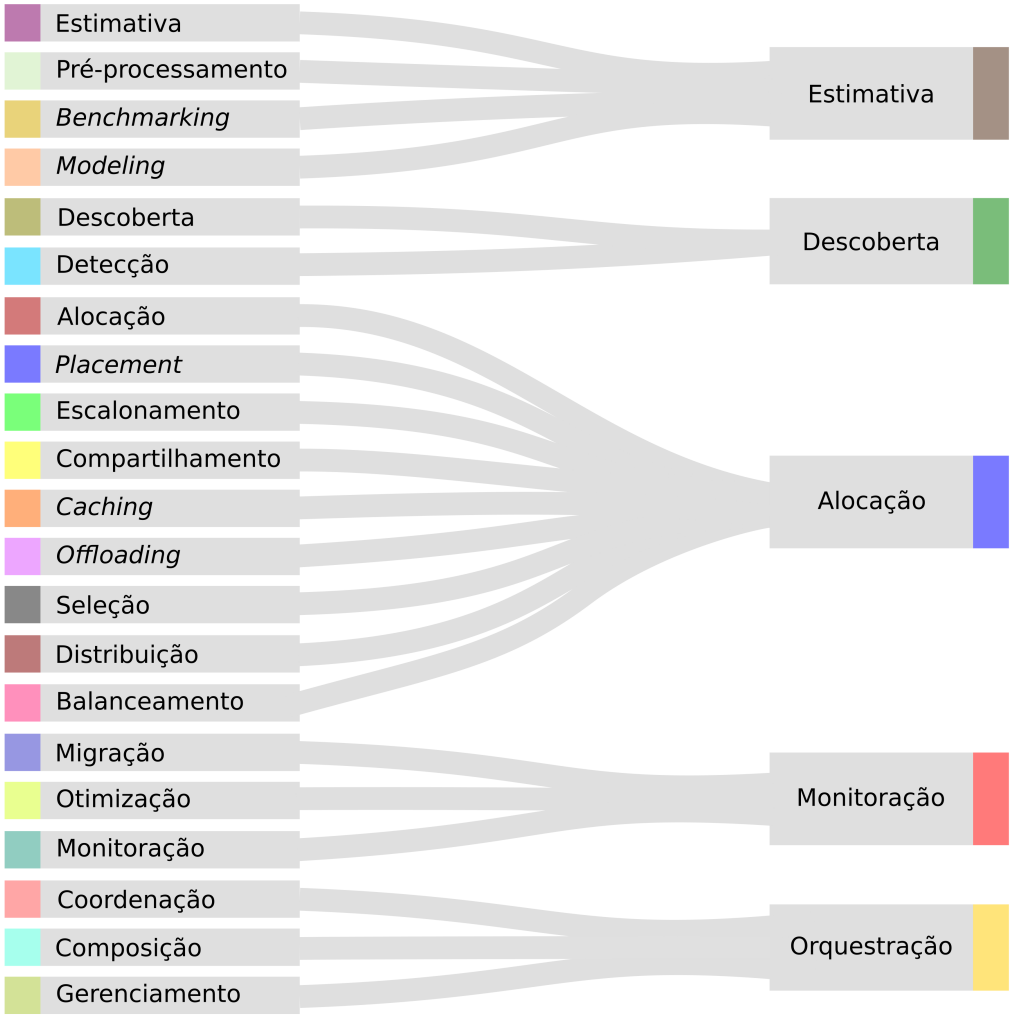

Figura 3.8: Análise de etapas de gerenciamento de recursos [1].

Com base nesta proposta, o autor desta tese acredita que o serviço de gerenciamento é realizado completamente pelas cinco etapas. A etapa de Descoberta visa encontrar os re-

cursos disponíveis no ambiente da *fog computing*. A etapa de Estimativa define o número de recursos que serão necessários para a execução da carga de trabalho [122]. A etapa de Alocação seleciona os recursos que atendem aos requisitos definidos na etapa Estimativa, reservando-os e entregando-os para realizar as tarefas, atendendo aos critérios de QoS definidos anteriormente [114]. Quando a etapa de Alocação estiver concluída, o processo de Monitoração será iniciado. Ele considera aspectos como elasticidade, balanceamento de carga, tolerância a falhas, verificação de integridade, etc. [131]. O processo de Orquestração percorre todo esse processo, sendo fundamental para acompanhar, por exemplo, a desalocação dos recursos e disponibilizá-los novamente para uma nova alocação. Cada uma das cinco etapas é detalhada nas próximas seções.

3.2.1 Estimativa de Recursos

Um dos principais requisitos no gerenciamento de recursos computacionais é a capacidade de estimar quantos recursos serão necessários para a execução de uma tarefa [22]. Para Manvi e Shyam [91] este processo trata-se de "uma estimativa aproximada dos recursos reais necessários para a execução de uma aplicação, geralmente com algum pensamento ou cálculo envolvido". Aprofundando um pouco mais esta definição, Mahmud *et al.* [60] indica que a estimativa é um processo que auxilia na alocação de recursos computacionais apropriados, de acordo com algumas políticas e/ou critérios, objetivando atingir o nível de qualidade de serviço determinado.

O planejamento da capacidade necessária para *fog computing* pode envolver outras perspectivas além dos recursos computacionais [19], tais como a precificação [132, 133] e o consumo de energia [134]. Além disso, a estimativa de recursos também depende do tipo de dispositivo, da mobilidade, da energia disponível, dos tipos de dados a serem gerados ou processados, do método de comunicação, das medidas de segurança adotadas e, ainda, do comportamento do usuário [115].

Desta forma, dada a restrição de capacidades computacionais dos *fog nodes*, o processo de estimativa desempenha papel fundamental para a alocação e o uso otimizado dos recursos [19] em *fog computing*. Em [122] são apresentadas três técnicas para estimativa de recursos em *fog computing*, conforme apresentado na Figura 3.9, e descritas na sequência:

- **Perfil:** é utilizada quando um número limitado de *fog nodes* residem em um ambiente de *fog computing* e as especificações das aplicações são estáticas [135];
- **Preditiva:** com base nos padrões de execuções anteriores, os recursos apropriados para a execução de uma aplicação são determinados [115, 136];
- **Sob demanda:** os recursos são estimados com base nas expectativas dos usuários e em sua demanda [137].

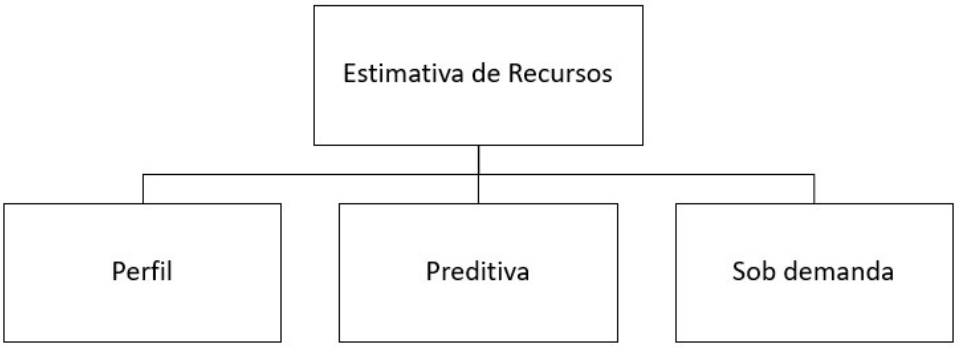

Figura 3.9: Técnicas para estimativa de recursos.

Além disso, a estimativa deve ser capaz de lidar com as flutuações na demanda de recursos tanto no lado do provedor quanto do lado do usuário final. Isso porque os recursos podem ser móveis e, portanto, se tornam rapidamente inacessíveis, o que os torna menos confiáveis do que os recursos da *cloud computing*, por exemplo. Atrelado a isso, deve ser considerada também a mobilidade do demandante, o que implica em uma rotatividade repentina de usuários, resultando em solicitações dinâmicas [22]. Dessa forma, a etapa de estimativa de recursos deve ser realizada com sobrecarga mínima e alta precisão [112].

Justamente por esta complexidade, a estimativa de recursos em *fog computing* é considerada um desafio a ser superado [138]. Em [139] outros desafios inerentes a esse processo são elencados, entre eles: o tipo do equipamento requisitante, que impacta no tipo de recurso ou serviço que deve ser entregue; a mobilidade, como já abordado, que impacta na velocidade e na dinamicidade da entrega; questões relacionadas ao consumo e a disponibilidade de energia nos equipamentos, uma vez que muitos deles são sustentados por baterias; o tipo da aplicação a ser entregue, considerando que algumas podem requerer mais processamento enquanto outras podem demandar por mais disco, por exemplo; a segurança exigida, uma vez que dados sensíveis não podem ser armazenados e/ou processados em determinados equipamentos ou regiões; a confiabilidade e fidelidade do cliente possibilitando, em alguns casos, a análise histórica da utilização para uma alocação mais ajustada.

Após analisar todos os artigos apresentados na Tabela 3.2 e considerando todos os termos similares usados que podem ser relacionados à estimativa de recursos, conforme resumido na Figura 3.8, a definição proposta para esta etapa é: "*A estimativa de recursos desempenha um papel essencial no processo de gerenciamento de recursos e refere-se ao cálculo da quantidade de recursos computacionais e do tempo necessário para realizar tarefas em ambientes de fog computing*".

3.2.2 Descoberta de Recursos

Manvi e Shyam [91] definem que o processo de descoberta é a identificação da lista de recursos computacionais autenticados que estão disponíveis para envio de trabalhos, e a escolha do melhor entre eles. Já para Singh e Chana [140], a descoberta é um processo de identificação dos dispositivos disponíveis para os quais é gerada uma lista dos recursos identificados para posterior seleção. Com isso, a descoberta de recursos abrange as ações de localização e divulgação das informações dos recursos, sendo essencial para explorar totalmente todos os recursos distribuídos no ambiente [28].

Considerando as características de *fog computing*, tais como a mobilidade, a alta distribuição geográfica, e também a heterogeneidade, o processo de descoberta de recursos é considerado um grande desafio e fundamental para o ambiente [141]. No artigo [92] o problema de descoberta de recursos em *fog computing* é declarado como uma forma de projetar uma solução para encontrar recursos pertencentes a componentes dispostos a se juntar a uma ambiente de *fog computing*. Tal solução deve considerar as diferentes características inerentes ao paradigma de *fog computing*, como por exemplo a mobilidade ou os modelos colaborativos. A Tabela 3.3 traz as definições sobre a etapa da Descoberta de recursos utilizadas pelos autores das publicações sobre gerenciamento de recursos, apresentados anteriormente na Tabela 3.2 (Seção 3.2).

Tabela 3.3: Definições para a etapa de descoberta de recursos.

Artigo	Definição
[3]	Identificar os recursos na borda para os quais as aplicações podem ser alocadas.
[22]	Identificar os recursos disponíveis, onde estão localizados e por quanto tempo.
[113]	Detecção para fornecer uma coleção de recursos e atributos disponíveis.
[114]	Encontrar os recursos de borda para implantar as cargas de trabalho da *cloud computing* ou dos usuários.
[115]	Identificar os recursos necessários para atender às solicitações de aplicativos.

Diante disso, a proposta desta tese para a definição da etapa de Descoberta de recursos é a seguinte: "*A descoberta é a tarefa do serviço de gerenciamento de recursos que visa encontrar os recursos disponíveis no ambiente de fog computing, mantendo o catálogo de recursos atualizado*". Uma vez que o processo de descoberta tenha registrado com sucesso o *fog node* no catálogo, seu monitoramento subsequente ocorre nas etapas de monitoramento e orquestração. Portanto, é essencial que a solução de descoberta de recursos esteja estreitamente alinhada com os demais processos do gerenciamento de recursos para garantir que os critérios de entrada sejam atendidos, e os resultados esperados sejam gerados, contribuindo assim para uma gestão eficaz [10].

Embora, sob uma perspectiva superficial, o objetivo da descoberta de recursos em *fog computing* possa parecer simples, envolvendo principalmente a identificação e o registo

dos *fog nodes* em um catálogo de recursos, uma análise mais profunda revela uma série de requisitos e desafios que precisam ser abordados adequadamente para lidar de forma mais eficaz com as peculiaridades do ambiente de *fog computing*. Assim sendo, as soluções de descoberta de recursos devem atender a novos requisitos que não estão presentes em outros paradigmas computacionais estabelecidos, como por exemplo a *cloud computing*, implicando na necessidade de abordagens específicas. A Figura 3.10 estabelece uma relação entre as características essenciais de *fog computing*, conforme definidas em [24], e os requisitos esperados que devem ser atendidos em soluções de descoberta de recursos. Esses requisitos são brevemente explicados abaixo.

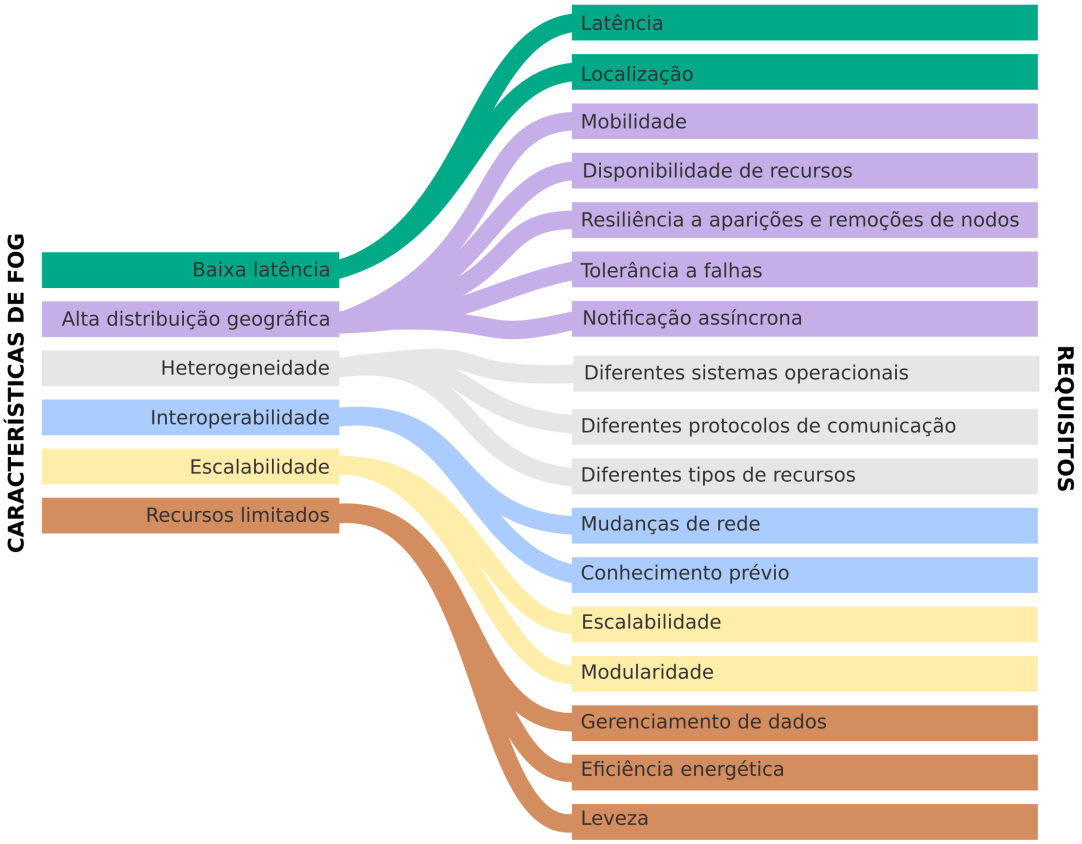

Figura 3.10: Características de *fog computing* versus requisitos e desafios da etapa de descoberta de recursos.

- **Latência**: a implantação na borda da rede permite o processamento de dados em dispositivos próximos aos usuários finais, em vez de enviar dados brutos para processamento na *cloud computing*. Isto resulta na aceleração do processamento e, consequentemente, na redução da latência [142];

- **Localização**: a solução de descoberta deve estar posicionada o mais próximo possível dos recursos, garantindo eficiência na alocação de recursos [143];
- **Mobilidade**: a mobilidade de dispositivos de borda da rede, tal como *smartphones*, é uma característica inerente de *fog computing*. Portanto, a solução de descoberta deve ser capaz de lidar com dispositivos móveis de forma eficaz [142];
- **Disponibilidade de Recursos**: devido à mobilidade e à natureza heterogênea dos *fog nodes*, é fundamental considerar a disponibilidade intermitente dos dispositivos, bem como a possibilidade destes dispositivos serem desligados em determinados momentos [144];
- **Resiliência a Aparições e Remoções de Nós**: a solução de descoberta deve ser projetada para acomodar dispositivos temporários, que podem entrar e sair do ambiente de *fog computing* sem contribuir significativamente para a alocação de serviços [143];
- **Tolerância a falhas**: construir sistemas tolerantes a falhas é imperativo devido à distribuição de serviços em múltiplos nós [143];
- **Notificação Assíncrona**: a utilização de serviços assíncronos na solução de descoberta é uma estratégia importante para superar problemas de sincronização de informações e falhas de comunicação [142];
- **Diferentes Protocolos de Comunicação**: a solução de descoberta deve ser flexível no que diz respeito aos esquemas de identificação, permitindo a descoberta independente das tecnologias e protocolos de comunicação adotados pelos dispositivos [142];
- **Diferentes Tipos de Recursos**: a presença de diferentes tipos de *fog nodes*, tais como *switches*, roteadores, *notebooks* e *smartphones*, cada um com arquiteturas diferentes, deve ser considerada na solução de descoberta [9];
- **Diferentes Sistemas Operacionais**: por se tratarem de equipamentos distintos e com funções diferentes, é comum a existência de sistemas operacionais diferentes nos *fog nodes* que compõem um ambiente de *fog computing* [9]. Assim, uma solução de descoberta multiplataforma é desejável;
- **Mudanças na Rede**: a heterogeneidade das arquiteturas de rede, juntamente com as mudanças ao longo do ciclo de vida do ambiente *fog*, exigem soluções de descoberta que se adaptem a diferentes contextos de rede [143];

- **Conhecimento Prévio**: o entendimento prévio da infraestrutura subjacente e da distribuição dos componentes de serviço é essencial para garantir que um *fog node* atenda aos requisitos estabelecidos de QoS [145];
- **Escalabilidade**: o número cada vez maior de *fog nodes* requer soluções de descoberta que possam escalar operações para lidar com a crescente demanda [144];
- **Modularidade**: adaptabilidade e parametrização são essenciais, permitindo que a solução de descoberta seja adaptada para atender cenários de uso específicos como *Industrial IoT* (IIoT) e casos de uso de gerenciamento de tráfego, que podem apresentar diferentes funcionalidades e estabilidade de rede [143];
- **Gerenciamento de Dados**: a solução de descoberta deve ser projetada para evitar sobrecargas na rede e no ambiente de *fog computing*, reduzindo o volume de informações transmitidas e, por sua vez, preservando o desempenho [143];
- **Eficiência Energética**: a busca pela eficiência energética é um grande desafio em *fog computing*, dada a limitação de energia de muitos dispositivos operados por bateria [144];
- **Leveza**: para evitar impactos significativos no desempenho do ambiente de *fog computing*, a solução de descoberta deve ser projetada de forma leve, adequando-se aos recursos limitados dos *fog nodes* [144].

A consideração desses requisitos e desafios desempenha um papel fundamental no processo de projeto e implementação de soluções eficazes de descoberta de recursos no ambiente de *fog computing*, permitindo o desenvolvimento de sistemas eficientes e flexíveis, além de contribuir para uma gestão otimizada e adaptável dos recursos. Uma análise sistemática da literatura existente sobre o domínio da descoberta de recursos, que será apresentada na Seção 4.1, permitirá esclarecer se esses requisitos estão sendo atendidos e como os desafios estão sendo abordados. Os resultados desta pesquisa sobre descoberta de recursos resultaram em um artigo entitulado "Resource Discovery in Fog Computing: A Review, Taxonomy, and Future Directions" que está em processo de revisão pelo *Journal of Parallel and Distributed Computing*, cujo Qualis é A1 (Anexo III).

3.2.3 Alocação de Recursos

Considerando as etapas apresentadas na Tabela 3.2 e agrupadas na Figura 3.8, a etapa de Alocação desempenha um papel essencial no processo de gerenciamento de recursos. Isso é identificado pelo fato de ser a etapa que abrange o maior número de termos, e é o foco da maior parte dos autores pesquisados. Em Nath *et al.* [112] e em Luo *et*

al. [128] a alocação de recursos deve entregar o dispositivo mais adequado, considerando o conhecimento obtido nas etapas anteriores, como a Estimativa de recursos. Ambos, Mahmud *et al.* [60] e Mijuskovic *et al.* [125], afirmam que a alocação de recursos é uma técnica usada para otimizar a utilização de recursos.

No contexto de *fog computing*, a alocação de recursos visa atender a um conjunto de tarefas $T = \{T_1, T_2, .., T_n\}$ que possuem diferentes requisitos de qualidade de serviço (tais como custo e tempo de execução), em um conjunto de recursos $R = \{R_1, R_2, ..., R_m\}$, que possuem diferentes capacidades computacionais (como por exemplo, processamento e armazenamento), utilizando uma função objetivo que pode adotar diferentes critérios, tais como minimizar custo, maximizar uso, etc. [127].

Os autores em Ghobaei *et al.* [26] afirmam que o problema de alocação de recursos é diferente em ambientes de *cloud computing* e *fog computing*, uma vez que neste último, deve-se alocar eficientemente um conjunto de *fog nodes* geograficamente distribuídos para serviços de IoT concorrentes com diferentes requisitos de QoS, enquanto na *cloud computing* este ambiente é mais homogêneo. Geralmente, os principais recursos computacionais envolvidos na pesquisa atual sobre alocação são recursos de computação, comunicação e armazenamento [127]. A etapa de alocação de recursos está representada na Figura 3.11.

Assim, a alocação de recursos pode ser vista como a etapa que recebe as informações de estimativa e faz a efetiva reserva dos recursos computacionais para que a tarefa possa ser executada no ambiente de *fog computing*, de acordo com os requisitos de QoS. Como em qualquer processo, a alocação de recursos tem entrada e saída. A entrada vem da etapa anterior, que em nossa proposta de gestão de recursos é a etapa de Estimativa. Essas entradas indicam métricas de QoS, bem como outros requisitos, por exemplo, computacionais. A saída da alocação de recursos é composta pela técnica de alocação, método de virtualização e as camadas de arquitetura a serem cobertas. Uma visão geral deste fluxo da alocação de recursos é apresentada na Figura 3.12.

Considerando as Figuras 3.11 e 3.12 nesta tese é proposta a definição de alocação de recursos como sendo "*a etapa do processo de gerenciamento de recursos que visa selecionar, reservar e usar os melhores recursos disponíveis para executar uma carga de trabalho no ambiente de fog computing, respeitando a adesão aos parâmetros de QoS*". De toda forma, para uma melhor contextualização, especificamente sobre a etapa de alocação de recursos, a seguir serão apresentadas algumas revisões de literatura que já foram publicadas sobre o tema.

Em Patil *et al.* [146] é feita uma revisão sobre a alocação de recursos em *fog computing* com 17 artigos, apresentando os objetivos e o escopo de cada método. Todavia, nesse trabalho não foram informados dados sobre a metodologia de revisão utilizada, e os desafios para o problema de alocação de recursos. Ahmed e Zeebaree [30] forneceram uma análise

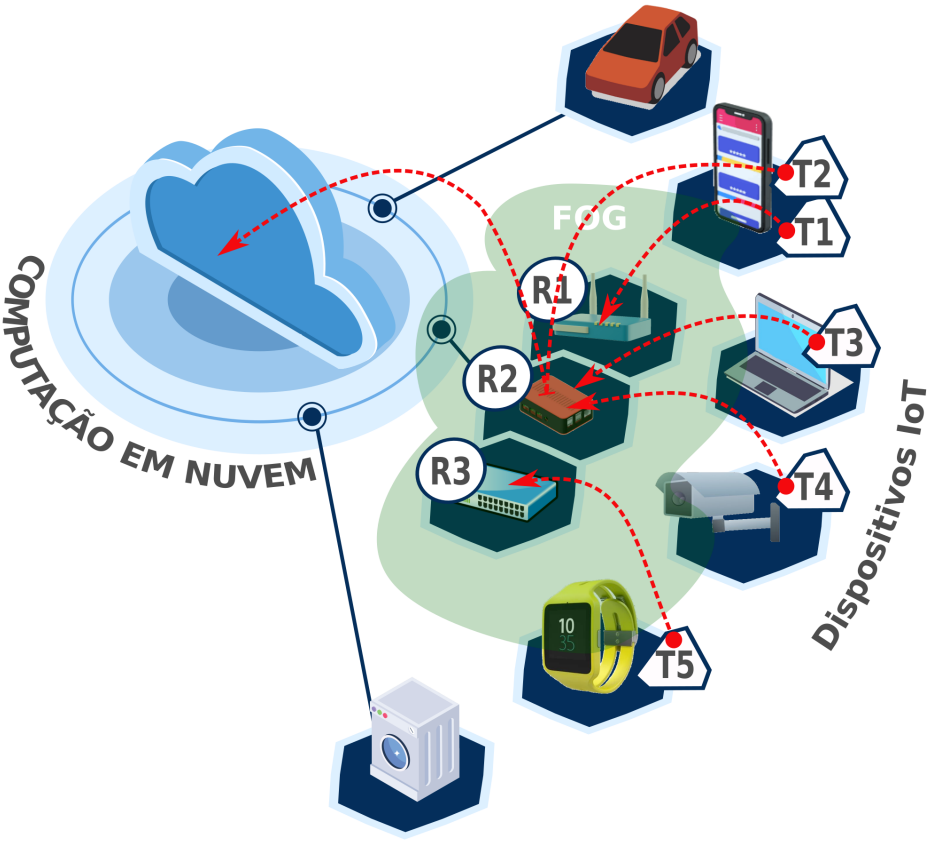

Figura 3.11: Fluxo da alocação de recursos na *fog computing* [1].

sistemática de *fog computing* com foco em modelos de sistema e alocação de recursos. O artigo é sustentado por três questões de pesquisa que objetivaram responder a relevância, as métricas e os objetivos da etapa de alocação de recursos, por meio da análise de 25 artigos selecionados.

O artigo [147] apresentou uma revisão sobre alocação de recursos em *fog* e *edge*, com foco no *design* e na implantação de sistemas corporativos, como medicina, veículos e construções inteligentes. No entanto, não foi fornecida nenhuma descrição sobre o método de pesquisa, bem como os detalhes sobre o número de artigos analisados. Os autores Rahul *et al.* [148] apresentaram uma revisão de 10 publicações sobre o problema de alocação de recursos em *fog computing*, destacando as métricas de desempenho, as metas e o ambiente de simulação de cada artigo analisado. Contudo, não foram apresentados detalhes sobre a metodologia utilizada para realizar a revisão.

Por fim, Jamil *et al.* [31] forneceram uma pesquisa considerando a etapa de alocação de recursos em *fog computing* com foco em algoritmos dinâmicos baseados em Aprendizado de Máquina. No entanto, não foi fornecida nenhuma descrição sobre o método de pesquisa, faltando informações sobre o número e o período dos artigos analisados.

Ao analisar os pontos fortes e as fragilidades dessas publicações, foi possível elaborar

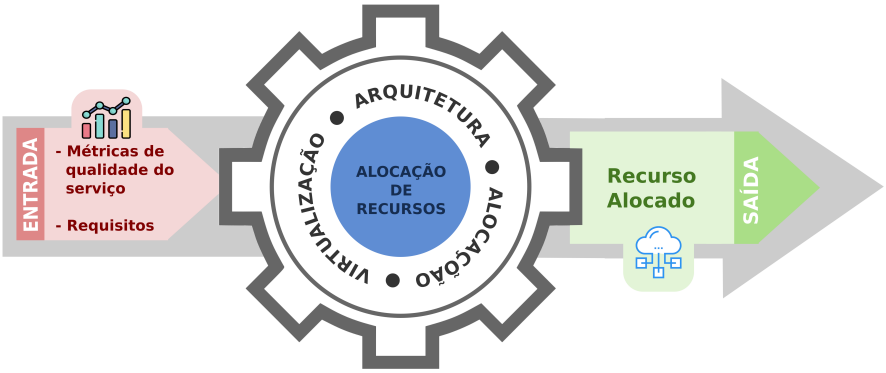

Figura 3.12: Visão geral do fluxo da alocação de recursos [1].

uma análise detalhada dos artigos disponíveis na literatura sobre a alocação de recursos em *fog computing*, relacionando-os com os tópicos abordados no fluxo apresentado na Figura 3.12. O resultado dessa análise será apresentado na Seção 4.2.

Uma revisão da literatura sobre a etapa de alocação de recursos em *fog computing* foi publicada pelo autor desta tese no artigo entitulado "*Computational Resource Allocation in Fog Computing: A Comprehensive Survey*", no periódico *ACM Computing Surveys (CSUR)*, cujo Qualis é A1, em julho de 2023 [1] (Anexo IV).

3.2.4 Monitoração de Recursos

Considerando as características de *fog computing*, os sistemas que são executados neste ambiente terão uma alta distribuição de seus componentes, carga variável e imprevisível, ocasionada pela heterogeneidade de dispositivos, *links* de comunicação e falhas. Assim, torna-se difícil prever como esses sistemas se comportarão ao longo do tempo [149]. A monitoração de infraestrutura é a base para dar suporte a vários objetivos, tais como: fazer uso eficiente de recursos, medir o desempenho de recursos e serviços, gerar faturas precisas [150], e implementar processos de tolerância a falhas.

Um serviço de monitoração pode ser estruturado como uma composição de três funções distintas: 1. observação dos recursos e serviços monitorados; 2. processamento de dados; e 3. exposição de dados [151,152]. Observação significa a aquisição de *status* atualizados de uso de recursos (por exemplo, carga e latência) ou desempenho do serviço (por exemplo, tempo de resposta). O processamento está relacionado aos ajustes necessários e à transformação exigida nos dados, como filtragem e agregação, criação e gerenciamento de eventos e notificações derivadas de regras e limites pré-configurados. A exposição está relacionada ao local onde os dados gerados são armazenados (por exemplo, em um banco de dados local, arquivos JSON), e como podem ser acessados por um sistema de gestão.

Além disso, a monitoração possui três domínios de instrumentação [153]: métricas, *logs* e *traces*. Cada domínio possui características próprias e suporta diferentes processos de tomada de decisão, que podem ocorrer antes, durante e após o início da coleta de dados de um determinado objeto monitorado. Uma métrica é uma medida em um determinado ponto no tempo. Ela é representada por um nome, um valor (a medida), um carimbo de data/hora (*timestamp*), e outros dados de contexto associados (opcionais). Um *log* é uma coleção de *strings* não estruturadas ou semi-estruturadas. Eles trazem informações detalhadas e contexto adicional. Por fim, um *trace* é a representação de uma única operação, uma requisição de um usuário dentro de um serviço, por exemplo, mostrando todo o caminho de execução feito do início ao fim da requisição [153].

Usando terminologia criada no domínio de teste de software, a monitoração pode ser dividida em caixa preta e caixa branca [152]. Caixa preta é o nome atribuído à monitoração feita a partir da interface pública do objeto monitorado, e tem como objetivo responder se o objeto está disponível/funcionando, ou seja, identificar se há algum problema ou não. A caixa branca se baseia na obtenção de informação detalhada sobre o funcionamento dos processos internos do objeto. Ela tem como objetivo responder porque o objeto não está disponível ou porque não está funcionando corretamente, ou seja, permite uma análise de causa raiz [154]. As métricas estão mais relacionadas à monitoração de caixa preta, enquanto os *logs* e os *traces* estão mais relacionados à monitoração de caixa branca.

A observabilidade é um conceito emergente que tem sido usado para fazer referência a funções avançadas de monitoração, no contexto de aplicativos baseados em microsserviços. A observabilidade às vezes é considerada um superconjunto de monitoração, pois visa atender aos mesmos propósitos e outros adicionais, estendendo o conceito de monitoração e aplicando técnicas de análise de dados nos dados de monitoração [155]. A observabilidade está mais relacionada à monitoração do tipo caixa branca, e à geração e consumo de rastros como principal base de informação e tomada de decisão.

Em relação ao tema de observabilidade, em novembro de 2023 foi apresentado um artigo entitulado "*Achieving Observability on Fog Computing with the use of open-source tools*" na conferência "EAI MobiQuitous 2023 - 20th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services" (Anexo V). Também foi submetido e aceito para publicação um capítulo de livro no *Springer Handbook of Data Engineering* em 2025 (Anexo VI).

Ao analisar a Tabela 3.2, as etapas de migração e otimização têm uma abordagem semelhante à monitoração e, por isso, foram agrupadas em nossa proposta apresentada na Figura 3.8. Uma revisão de literatura e uma taxonomia para monitoração de recursos em *fog computing* é apresentada no trabalho [143]. Com isso, nesta tese definiu-se a etapa

de monitoração de recursos como: "*A monitoração é a etapa do serviço de gerenciamento de recursos que deve coletar informações de status atualizadas sobre fog nodes e links de comunicação, e enviá-las para o orquestrador, que pode tomar as devidas providências para garantir os SLAs*".

Como resultado da análise da literatura sobre a monitoração de recursos em *fog computing*, foi proposta uma taxonomia que o autor publicou no artigo "*Monitoring fog computing: A review, taxonomy and open challenges*", no periódico *Computer Networks*, que tem Qualis Capes A1, em julho de 2022 [143] (Anexo VII).

3.2.5 Orquestração

O dicionário Cambridge [156] define orquestração como "um arranjo cuidadoso de algo para alcançar um resultado particular", e o dicionário Merriam-Webster [157] a define como "organização harmoniosa". Esses dicionários trazem os seguintes sinônimos para orquestração: gerenciar, organizar, coordenar, reorganizar e reestruturar.

O artigo [158] afirma que orquestração, geralmente, refere-se a "um gráfico descrevendo relacionamentos entre elementos de software ou processos". Essa definição também é utilizada no trabalho [159]. Viejo e Sánchez [160] definem orquestração como a seleção de nós que participarão da execução de um serviço ou aplicativo. Uma vez selecionados, esses nós se comunicam por meio de um protocolo seguro que garante a privacidade da troca de informações. Assim, considera-se que a entidade demandante, neste caso um dispositivo IoT solicitando serviços da camada superior, será sempre estática e com uma associação fixa com os mesmos *fog nodes*, como em uma estrutura de fábrica. Esses artigos citados usaram orquestração como um gráfico de dependência ou composição. Nesse cenário, todas as entidades envolvidas têm participação obrigatória. Esta utilização de "orquestração" é usual no contexto de Arquitetura Orientada a Software (SOA) e Web Services [161].

O artigo [162] foi um dos primeiros artigos focados em orquestração em ambientes de *fog computing*. Segundo ele, "orquestração é um conceito chave dentro de sistemas distribuídos, permitindo o alinhamento das aplicações implantadas com os interesses de negócios dos usuários". Em sua visão, o orquestrador deve "prever, detectar e resolver problemas relativos a gargalos de escalabilidade que podem surgir com o aumento da escala de aplicativos". Nas palavras de Jiang *et al.* [163], orquestração "refere-se aos processos de gerenciamento e coordenação dos recursos computacionais físicos fornecidos pela infraestrutura subjacente para servir as aplicações". De acordo com o artigo [164], "hoje em dia, orquestração é uma palavra muito usada; é apresentada em diferentes cenários para indicar a gestão do ciclo de vida de um ou mais componentes distribuídos que juntos entregam um serviço ou uma funcionalidade".

Considerando a diversidade de conceitos apresentados, é necessário atribuir uma definição para o termo "orquestração" no contexto de *fog computing*, e por ainda não haver uma padronização adotada pela academia, no artigo [10], de autoria do mesmo autor desta tese, foi proposta a seguinte definição para a etapa de orquestração: "*A orquestração na fog computing é uma função de gerenciamento responsável pelo ciclo de vida do serviço. Para fornecer os serviços solicitados ao usuário e garantir os SLAs, deve-se monitorar a infraestrutura subjacente, reagir em tempo hábil às suas mudanças e cumprir as regras de privacidade e segurança*".

Bonomi *et al.* [52] propuseram uma arquitetura de software para execução de serviços de *fog computing*, a qual é apresentada na Figura 3.13. Eles mostram uma camada de orquestração de *fog computing*, estruturada como um *loop* de controle Monitorar-Analisar-Planejar-Executar (do inglês, *Monitor-Analise-Plan-Execute* - MAPE), que é responsável por fornecer gerenciamento de ciclo de vida de serviços de *fog* de maneira distribuída. Para ilustrar a malha de controle é interessante começar na fase Monitorar. Nesta fase, a orquestração deve coletar o *status* atualizado sobre cada recurso gerenciado e os serviços em execução. A partir da análise (fase Analisar) dos dados de monitoração, pode-se construir uma visão atualizada e abrangente do ambiente de *fog computing* e planejar (fase Planejar) as mudanças necessárias para manter os serviços dentro dos limites de SLAs e QoS, além de fornecer novos serviços solicitados. A execução (fase Executar) dessas mudanças planejadas liberará e alocará recursos adequados, fornecendo serviços próximos aos usuários finais.

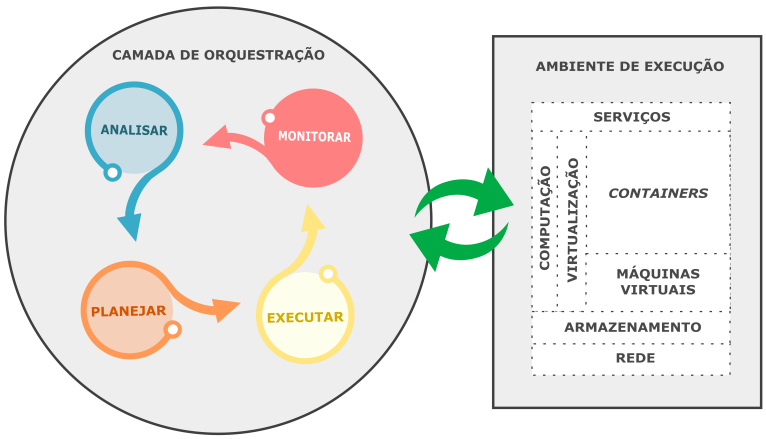

Figura 3.13: Camada de orquestração em *fog computing* [10].

Uma revisão da literatura sobre a etapa de orquestração em *fog computing* foi publicada pelo autor desta tese no trabalho entitulado "*Orchestration in Fog Computing: A*

Comprehensive Survey", no periódico *ACM Computing Surveys (CSUR)*, cujo Qualis é A1, em janeiro de 2022 [10] (Anexo VIII).

3.3 Considerações Finais

Neste capítulo foram apresentados os conceitos mais relevantes sobre o gerenciamento de recursos em *fog computing*. O conceito de fog node foi detalhado, trazendo a fundamentação sobre as suas capacidades computacionais e suas características comportamentais. Também foram analisadas as cinco etapas que compõem o gerenciamento de recursos: estimativa, descoberta, alocação, monitoração e orquestração.

A descoberta tem como objetivo encontrar novos recursos no ambiente dinâmico de *Fog*, tornando-os disponíveis para uso após a atualização do Catálogo de Recursos. A estimativa é o cálculo da quantidade de recursos computacionais e do tempo necessário para executar um conjunto de tarefas. Já a alocação busca selecionar os melhores recursos catalogados para executar uma carga de trabalho no ambiente de *Fog*, de acordo com os critérios dos usuários. A monitoração coleta dados sobre o status dos nós de *Fog*, seus recursos e links de comunicação, mantendo atualizado o Catálogo de Recursos e fornecendo informações relevantes ao orquestrador para tomada de decisão. Por fim, a Orquestração é uma função de gerenciamento responsável pelo ciclo de vida do serviço, garantindo a entrega dos serviços solicitados ao usuário e assegurando o Acordo de Nível de Serviço (SLA).

Capítulo 4

Trabalhos Relacionados

Como o escopo desta tese abrange as etapas de descoberta e de alocação de recursos, uma análise da literatura sobre essas duas etapas será apresentada neste capítulo. Para isso, serão apresentados os procedimentos das revisões sistemáticas da literatura realizadas. Os resultados obtidos sobre a descoberta de recursos constam na Seção 4.1. Já os resultados para a alocação de recursos é apresentada na Seção 4.2.

4.1 Descoberta de Recursos

Nos últimos anos, foram publicadas algumas pesquisas específicas sobre o processo de descoberta de recursos. Porém, pela pesquisa realizada, não foi encontrado nenhum trabalho específico que aborde o processo de descoberta de recursos para o ambiente de *fog computing*. No entanto, dado o contexto da *fog computing* estar relacionado à IoT ou ainda a outros paradigmas distribuídos tais como a *Mobile Computing* ou a *Edge Computing*, grande parte desta pesquisa sobre descoberta de recursos é direcionada para essas outras abordagens, o que foi usado para contornar a ausência de artigos específicos sobre *fog computing* para desenvolver esta seção de trabalhos relacionados.

Uma pesquisa abrangente e uma taxonomia sobre o processo de descoberta de recursos para sistemas de computação distribuída são apresentados em [28], agrupando aspectos inerentes ao tema em três grandes grupos. Os *aspectos estruturantes* contêm as variáveis que devem ser analisadas e configuradas antes da efetiva realização da descoberta. Entre esses aspectos estão as informações de recursos em termos arquitetônicos sobre o ambiente de computação, os modelos e as linguagens usados para descrever os recursos e as abordagens para agrupar os recursos. Os *aspectos de design* incluem técnicas, estratégias e métodos importantes e relevantes que podem ser usados para conduzir consultas em entidades distribuídas para localizar e descobrir recursos, entre elas estão: algoritmos de busca, mecanismos de propagação de pacotes, estratégias de consulta, entrega de in-

formações de recursos, sincronização de dados entre provedores de recursos distribuídos e técnicas de resumo de informações. Finalmente, os *aspectos de avaliação* abrangem todos os conceitos e termos que expressam, demonstram ou avaliam os resultados esperados ou desejados do procedimento de descoberta de recursos. Dependendo do ambiente, das aplicações e dos objetivos específicos para os quais um protocolo de descoberta foi projetado, o número de determinadas funcionalidades, recursos e fatores de desempenho podem se tornar extremamente importantes para serem alcançados, alguns exemplos são: escalabilidade, eficiência, confiabilidade, flexibilidade e heterogeneidade. Embora bastante completo, o trabalho apresentado é antigo (publicado em 2017), e não trata de especificidades de ambientes altamente distribuídos, como o *fog computing*.

Além disso, foram apresentados alguns trabalhos para sistemas distribuídos baseados em comunicação Peer-to-Peer (P2P). Os autores do [165] apresentam uma taxonomia baseada em oito domínios, a saber: escala, tipo, arquitetura, busca, escopo, propagação de pacotes, camada OSI e posição. Em [166], alguns artigos são comparados por alguns fatores para projetar técnicas de descoberta de recursos eficientes e aplicáveis. Por fim, o artigo [167] classifica os mecanismos de descoberta de recursos em sistemas P2P em cinco grupos: meta-heurísticas, informadas, baseadas em grupos, híbridas e bio-inspiradas. A limitação destes artigos é o fato de serem projetados especificamente para comunicação P2P, o que não atende plenamente todas as características e ambientes de *fog computing*.

Mais recentemente, com o avanço da Internet das Coisas, foram apresentadas algumas publicações sobre descoberta de recursos neste domínio. Em [142] os autores classificam as técnicas de descoberta de recursos IoT como baseadas em dados e em objetos. Além disso, também propuseram uma classificação em quatro domínios (escopo, metodologia, arquitetura e abordagens) para os principais protocolos de comunicação no contexto da IoT. No artigo [29], os autores apresentam uma taxonomia para classificar métodos de descoberta de recursos em um ambiente envolvendo *cloud computing* e IoT em quatro pilares: arquitetura, algoritmo, *middleware* e protocolo. Por fim, em [144] os autores apresentam uma revisão sistemática que classifica as técnicas de descoberta de recursos em quatro domínios: contexto, energia, qualidade de serviço e semântica.

Finalmente, os autores do [168] conduziram uma revisão da literatura sobre descoberta e alocação de recursos. Especificamente para a etapa de descoberta, foram considerados cinco requisitos: localização, contexto, dispositivo, correspondência e desempenho. Diante desta análise, os artigos foram categorizados em uma taxonomia de sete domínios, a saber: *Broadcasting* baseado em *beacon* Wi-Fi 802.11; Transporte de Telemetria de Enfileiramento de Mensagens (MQTT); Baseado em Wi-Fi; Baseado na comunidade DNS e BGP; Baseado em P2P; Baseado em ARP; e outros métodos de exploração. Porém, embora os autores tenham realizado pesquisas sobre os cinco requisitos listados, alguns

Tabela 4.1: Análise dos trabalhos relacionados à descoberta de recursos.

Artigo	Ano	Ambiente	Sistemática	Taxonomia	Domínios
[165]	2008	P2P	Não	Sim	8
[166]	2015	P2P	Não	Não	–
[28]	2017	Sist. Distribuídos	Não	Sim	3
[167]	2020	P2P	Não	Sim	5
[142]	2020	IoT	Não	Sim	2
[144]	2020	IoT	Sim	Não	–
[29]	2022	IoT	Sim	Sim	4
[168]	2022	Fog	Sim	Sim	5
Nosso trabalho	2023	Fog	Sim	Sim	10

outros aspectos relevantes não foram abordados, tais como a descrição e o catálogo de recursos, tornando sua revisão incompleta quando comparada à revisão da literatura realizada.

A Tabela 4.1 fornece um resumo dos trabalhos analisados nesta seção, comparando-os com trabalho proposto. Como pode ser observado na Tabela 4.1, a maior parte dos trabalhos relacionados foi escrita sem questões de pesquisa definidas ou uma forma sistemática de seleção dos artigos a serem analisados. Ressalta-se também que o artigo proposto pelo autor desta tese apresentou uma taxonomia com o maior número de domínios analisados em relação aos demais. Além disso, apenas um dos trabalhos relacionados focou especificamente em *fog computing*. Portanto, a partir da identificação da lacuna existente na literatura, esta tese tem uma clara contribuição com o estado da arte.

4.1.1 Metodologia de Pesquisa da Literatura

Nesta seção é descrito o método utilizado na revisão sistemática da literatura sobre descoberta de recursos em *fog computing*. O protocolo de revisão foi modelado a partir de abordagens previamente adotadas em [11] e [12]. Este processo incluiu as seguintes etapas: 1. formulação das Questões de Pesquisa (QP); 2. seleção das bases de dados de pesquisa; 3. elaboração de uma *string* de busca composta por termos-chave relevantes; 4. coleta de resultados; 5. aplicação de critérios de inclusão e exclusão; 6. filtragem dos estudos com base em termos-chave, título e resumo; e 7. análise e avaliação crítica dos demais estudos. Assim sendo, as seguintes questões de pesquisa orientaram a revisão sistemática:

- **QP1 - Quais são os atributos relevantes de uma solução apropriada de descoberta de recursos?**
  A resposta a esta questão será sistematizada como uma nova taxonomia, facilitando aos investigadores a identificação das características relevantes que uma solução de descoberta de recursos em *fog computing* deve abranger;
- **QP2 - Qual é o nível de maturidade e abrangência das soluções atuais para descoberta de recursos em *fog computing*?**

A resposta a esta questão será obtida utilizando a taxonomia para classificar as propostas selecionadas por esta revisão sistemática da literatura;

- **QP3 - Quais são os desafios que ainda exigem atenção da comunidade acadêmica?**
  A resposta a esta questão pode servir de guia para pesquisadores que queiram direcionar seus esforços em futuras investigações nesta área.

Na busca por responder estas questões de pesquisa, foram utilizadas como fontes de pesquisa as seguintes bases de dados: Scopus[1], Web of Science[2], ACM Digital Library[3] e IEEE Xplore Library[4]. Uma *string* de pesquisa básica foi formulada como "(fog OR edge) AND (discover* OR detect*)". Além disso, foram realizadas pesquisas adicionais utilizando terminologias associadas a outros paradigmas distribuídos, conforme listado na Seção 2.4.

Os critérios de inclusão adotados compreenderam: artigos primários submetidos à revisão por pares; artigos escritos em inglês; data de publicação entre 2017 e julho de 2024 (momento da realização desta pesquisa) para garantir a análise apenas das propostas atuais; e trabalhos que apresentem soluções, modelos arquitetônicos, técnicas ou métodos aplicados ao campo de descoberta em *fog computing*. Após a realização das buscas e consolidação dos resultados obtidos, as duplicatas foram eliminadas, resultando em 97 estudos restantes. Estes foram então submetidos a um processo de filtragem baseado em palavras-chave, títulos e resumos, resultando num conjunto final de 31 artigos. Por fim, após a análise completa do conteúdo dos demais estudos, foram selecionados 16 trabalhos que serão amplamente discutidos nas próximas seções, fornecendo a base necessária para responder às questões de pesquisa propostas.

4.1.2 Taxonomia para Descoberta de Recursos em *Fog*

Para responder ao QP1, que aborda as características mais pertinentes de uma solução adequada para descoberta de recursos em *fog computing*, esta seção introduz uma nova taxonomia baseada em uma revisão sistemática da literatura. A taxonomia proposta consolida domínios e categorias cruciais em uma solução de descoberta de recursos de próxima geração para *fog computing*.

Para desenvolver esta taxonomia foi realizada uma análise das taxonomias existentes e uma revisão sistemática da literatura. Também foi levado em consideração os desafios e requisitos associados à descoberta de recursos (conforme detalhado na Seção 3.2.2) e as

[1] scopus.com
[2] webofknowledge.com
[3] dl.acm.org
[4] ieeexplore.ieee.org

características específicas de *fog computing*. Esses domínios e categorias estão resumidos na Figura 4.1, e uma descrição detalhada de cada um é apresentada nas seções seguintes.

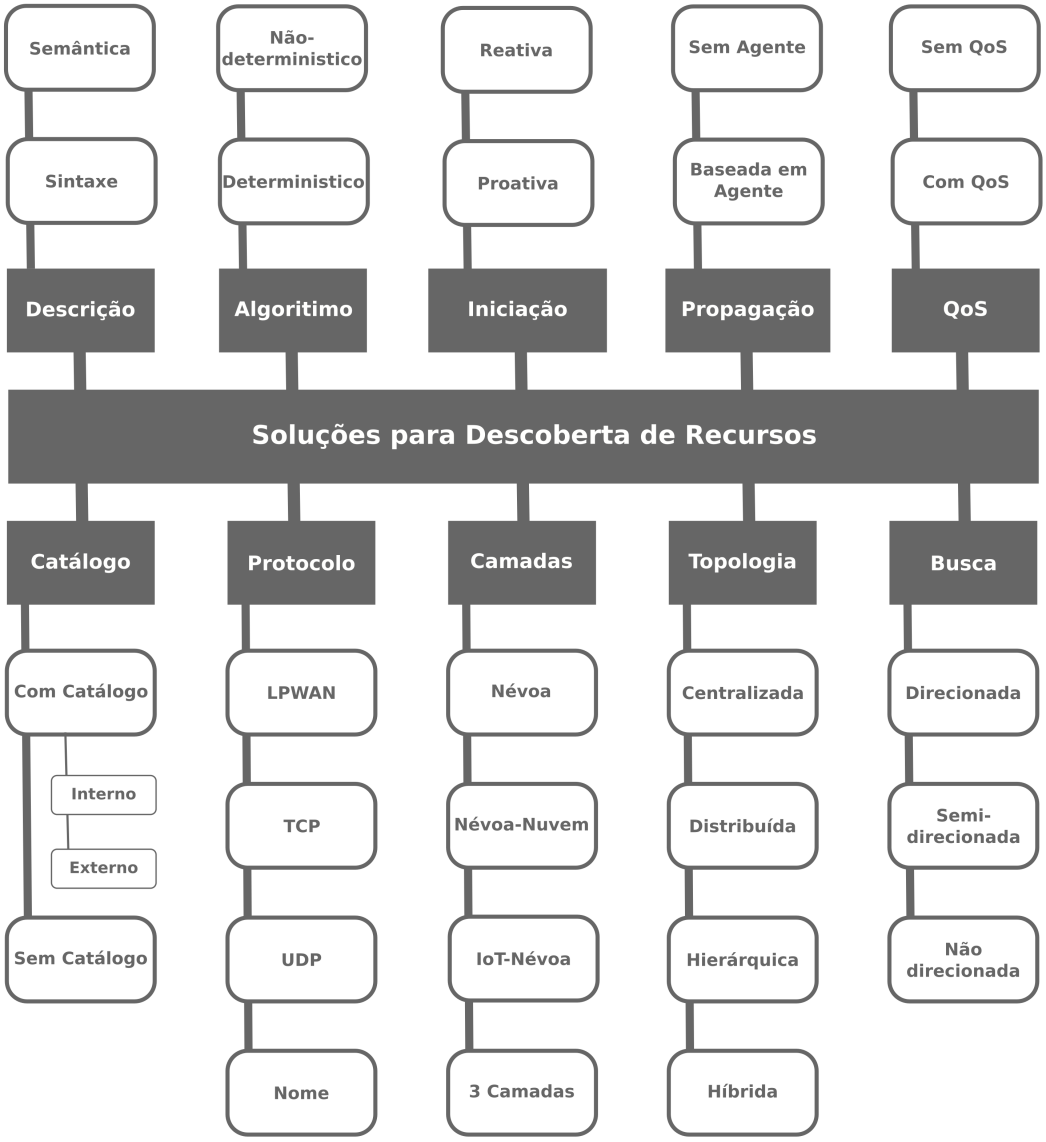

Figura 4.1: Proposta de taxonomia de descoberta de recursos em *fog computing*.

4.1.2.1 Algoritmo

Os algoritmos são de grande importância nas soluções de descoberta de recursos, pois são a base da técnica empregada nesta área [28]. O paradigma *fog computing* é um ambiente complexo, caracterizado pela sua variabilidade, que vai desde configurações limitadas de recursos, um número variável deles, além da heterogeneidade. Neste contexto, a tarefa de identificar e classificar algoritmos apropriados para descoberta de recursos pode ser considerada um desafio.

A literatura existente utiliza diferentes abordagens para categorizar algoritmos neste domínio. Por exemplo, num estudo realizado por Pourghebleh *et al.* [144], os métodos são agrupados em categorias que levam em consideração conhecimento de contexto, eficiência energética, QoS e semântica. Em contraste, Zarrin *et al.* [28] propõe uma taxonomia que classifica algoritmos com base em critérios como algoritmos informados versus não-informados, síncronos versus assíncronos, determinísticos versus não-determinísticos e algoritmos bio-inspirados versus não inspirados na natureza.

Porém, nos últimos anos, houve uma evolução significativa no desenvolvimento de algoritmos em todas as áreas da Computação, incluindo soluções para descoberta de recursos em *fog computing*. Esta evolução é evidenciada, por exemplo, pela progressão das técnicas de Aprendizagem de Máquina e de Inteligência Artificial. Consequentemente, as abordagens de categorização propostas pelos autores anteriores já não cobrem totalmente as diversas soluções presentes na literatura atual. Por esse motivo, optou-se por classificar os algoritmos de descoberta de recursos em *fog computing* em apenas dois grupos distintos: determinísticos e não-determinísticos. Esta simplificação reflete a necessidade de adaptar a categorização às tendências emergentes na área pesquisada, conforme descrito a seguir:

- **Determinístico**: são aqueles que a cada execução trazem sempre o mesmo resultado para as mesmas entradas de dados. Como em um ambiente de *fog computing* é esperada uma grande dinamicidade entre os *fog nodes*, o uso de métodos determinísticos é limitado a situações de ambientes controlados e conhecidos [169];
- **Não-Determinístico**: são aqueles que, para uma mesma entrada, podem produzir saídas diferentes em execuções diferentes. Algoritmos não-determinísticos podem mostrar comportamentos diferentes para a mesma entrada em execuções diferentes, e há um certo grau de aleatoriedade nisso. Devido a estas características, este tipo de algoritmo pode não fornecer soluções ótimas globais, garantindo, por vezes, apenas soluções ótimas locais, que são necessárias para resolver o problema em tempo hábil [170].

4.1.2.2 Inicialização

Inicialização é a primeira etapa do processo de descoberta, que ocorre quando um novo *fog node* se conecta ao ambiente e a solução de descoberta procura novos *fog nodes*. Com base na pesquisa realizada, é possível identificar três abordagens diferentes de inicialização:

- **Proativa**: neste tipo de inicialização, a solução de descoberta procura novos recursos constantemente ou em intervalos regulares, sem qualquer intervenção externa, como a de um agente. Este tipo de inicialização pode ser positiva para ambientes

grandes, pois ajuda a manter o catálogo de recursos atualizado. No entanto, se a frequência não for ajustada adequadamente, poderá sobrecarregar os recursos da rede;

- **Reativa**: em contraste com a abordagem proativa, esse tipo de inicialização apenas procura um novo *fog node* quando solicitado por outro processo. Isto pode não ser ideal para manter o catálogo de recursos atualizado, mas é benéfico em termos de conservação dos recursos utilizados para a pesquisa.

4.1.2.3 Camadas

É essencial analisar onde as estratégias de descoberta de recursos em *fog computing* procuram recursos, considerando a arquitetura de três camadas discutida anteriormente na Seção 2.3. Esta análise é crucial devido à natureza do problema de descoberta de recursos em *fog computing*, que pode ser interpretado como um problema de dupla correspondência, conforme mencionado por Ghobaei *et al.* [26].

As abordagens desenvolvidas para resolver esse problema podem variar em escopo e considerar diferentes combinações de camadas da arquitetura. Essas considerações podem envolver exclusivamente a camada *Fog*, a interação entre duas camadas específicas como IoT e *Fog*, *Fog* e *Cloud*, ou ainda abranger todas as três camadas da arquitetura, ou seja, IoT, *Fog* e *Cloud*, conforme descrito abaixo:

- ***Fog***: utiliza apenas os recursos disponíveis na camada *Fog*;
- ***Fog - Cloud***: utiliza os recursos disponíveis nas duas camadas superiores da arquitetura;
- **IoT - *Fog***: utiliza os recursos disponíveis nas duas camadas inferiores da arquitetura;
- **Três Camadas**: usa os recursos disponíveis em todas as camadas da arquitetura.

Existem diversas abordagens que consideram diferentes configurações das camadas da arquitetura de *fog computing* para lidar com a descoberta de recursos. Uma dessas abordagens concentra-se na conexão direta entre os dispositivos do usuário final na camada IoT e os dispositivos na camada de *Fog*. Uma vez estabelecido esse *link*, eles formam um único grupo de recursos e, posteriormente, se conectam à camada de *Cloud*. Essas abordagens, embora reconheçam a relação entre a camada *Fog* e a camada *Cloud*, buscam resolver todos os desafios de descoberta de recursos exclusivamente na camada de *Fog*, evitando a necessidade de encaminhar solicitações para a camada de *Cloud*.

Uma alternativa é aproveitar a conexão entre a camada de *Fog* e a camada de *Cloud*, onde os recursos disponíveis podem ser utilizados pelos usuários finais para realizar suas

tarefas. Também é possível considerar apenas a camada de *Fog*, na qual todas as solicitações são tratadas pelos recursos disponíveis naquele ambiente. No entanto, esta abordagem não é comum, uma vez que o *fog computing* é concebido como um complemento à *cloud computing*, e não como um substituto.

Finalmente, outra alternativa é usar as três camadas da arquitetura de *fog computing* juntas. Neste cenário, a camada de *Fog* é vista como uma camada intermediária que contribui para o alcance dos objetivos estabelecidos, como a melhoria da QoS. As solicitações originam-se na camada IoT e são atendidas de forma abrangente, envolvendo não apenas a camada de *Fog*, mas também a camada de *Cloud*. É razoável adotar esta abordagem, uma vez que a *fog computing*, conforme descrito nas definições fornecidas na Seção 2.3, objetiva complementar a *cloud computing*. Assim, quando todas as três camadas são gerenciadas de maneira integrada, surge o ambiente computacional conhecido como *continuum*, que neste caso seria o *continuum Edge/IoT - Fog - Cloud* [171].

4.1.2.4 Propagação

A propagação refere-se à forma como o processo de descoberta encontra recursos [165]. Entre os modelos mais comuns estão: i) *unicast*, que envolve apenas um emissor e um receptor; ii) *broadcast*, que é um modelo de busca um-para-todos; iii) *multicast*, usado em comunicações de modelo um-para-muitos. Porém, para extrair as informações dos *fog nodes*, as soluções podem exigir a existência de um agente. Por esta razão, a propagação em soluções de descoberta pode ser classificada da seguinte forma:

- **Baseada em Agente**: um agente é distribuído e instalado nos *fog nodes* e tem como missão coletar informações sobre o nó e transmiti-las ao catálogo de recursos. Este tipo de solução, geralmente, é capaz de obter dados mais completos sobre o *fog node*. Além disso, o agente pode ser necessário para outros processos de gestão de recursos, como monitoramento e orquestração;
- **Sem Agente**: em soluções deste tipo, não há necessidade de instalar nenhum agente no *fog node* descoberto. As informações são obtidas diretamente e tendem a ser menos completas do que as obtidas pelo modelo baseado em agentes. Estas soluções são mais leves e podem ser uma boa solução para nós com pouca capacidade computacional.

4.1.2.5 Protocolo

O protocolo desempenha um papel importante na solução de descoberta de recursos, pois serve como meio de comunicação com o recurso identificado. De acordo com [172], "um protocolo de rede é um conjunto de regras e procedimentos de comunicação usados em

ambos os lados por todas as estações que trocam dados pela rede". Ao longo do tempo, diversos protocolos foram desenvolvidos, cada um com diferentes métodos de comunicação, com o objetivo de atender regras gerais ou específicas de acordo com as necessidades. Uma revisão abrangente sobre os protocolos de comunicação no campo de *fog computing* pode ser encontrada em [173].

O paradgima *fog computing* desempenha um papel intermediário entre a *cloud computing* e os dispositivos do usuário em um ambiente IoT. Por esse motivo, as soluções de descoberta de recursos para *fog computing* devem ser capazes de lidar com protocolos de rede compatíveis com ambos os ambientes. Isso se deve ao fato de que a *cloud computing* muitas vezes depende de protocolos mais tradicionais, como HTTP ou HTTPS, enquanto a arquitetura IoT utiliza protocolos que visam eficiência, com baixo consumo de energia, custo reduzido e baixa taxa de bits [174]. Vale ressaltar que diversas propostas de protocolos foram publicadas por organizações renomadas, como a Internet Engineering Task Force (IETF), o World Wide Web Consortium (W3C), o Instituto de Engenheiros Elétricos e Eletrônicos (IEEE), a União Internacional de Telecomunicações (UIT) e o Instituto Europeu de Normas de Telecomunicações (ETSI) [172].

Com base nessas propostas e considerando os requisitos e desafios de descoberta de recursos apresentados, os protocolos utilizados em soluções de *fog computing* podem ser agrupados em quatro categorias distintas: *Low Power Wide Area Network* (LPWAN), Protocolo de Controle de Transmissão (TCP), Protocolo de Datagrama de Usuário (UDP) e Protocolo baseado em Nome. Cada uma dessas categorias será detalhada a seguir.

i) Protocolos LPWAN

De acordo com o modelo de arquitetura de quatro camadas de IoT da ITU [5], as "coisas" em um ambiente IoT podem ser diferenciadas de acordo com o tipo de interação com o ambiente, que pode ser classificado como alarme, medição e controle, ou uma combinação entre eles, e de acordo com o intervalo (curto e longo).

Os protocolos de curto alcance incluem RFID [175], Bluetooth [176], ZigBee [177] e Wifi [178]. Os protocolos de longo alcance são divididos em dois grupos: Padrões do Projeto de Parceria de Terceira Geração (3GPP) e Padrões Não-3GPP. Os padrões 3GPP incluem LTE-M [179], EC-GSM [180], NB-IoT [181] e 5G [182]. Os não padrões incluem LoRaWAN [181] e Sigfox [183].

Os requisitos da LPWAN buscam atender algumas especificidades e impactos no planejamento da rede, como bateria de longa duração, baixo custo, fácil implantação e suporte para um grande número de dispositivos.

ii) Protocolos Baseados em TCP

TCP é um padrão que define como estabelecer e manter a comunicação em rede,

permitindo que aplicações troquem dados [184]. Algumas pesquisas específicas sobre protocolos de rede usados em ambientes de *fog computing* listam os seguintes protocolos baseados em TCP [172, 173]:

- **Hyper Text Transport Protocol (HTTP)**: a comunicação ocorre com base em mensagens de solicitação/resposta, em um modelo tradicional cliente-servidor. Recentemente, este protocolo foi associado à Transferência de Estado Representacional (REST) permitindo uma troca de informações ainda mais completa [174];
- **Message Queue Telemetry Transport Protocol (MQTT)**: baseado no paradigma de publicação/assinatura, este protocolo leve é projetado para redes restritas [185]. O corretor recebe informações dos usuários e as entrega aos assinantes sobre um tópico. O MQTT é executado no protocolo TCP e possui um pequeno cabeçalho de mensagem que o qualifica para uso em ambientes de *fog computing*;
- **Advanced Message Queuing Protocol (AMQP)**: também baseado no paradigma de publicação/assinatura, é um protocolo de padrão aberto voltado à interoperabilidade em grandes redes [186]. Essa facilidade de trabalhar em ambientes heterogêneos qualifica o AMQP para ser utilizado em *fog computing*, principalmente em sistemas que não possuem restrições de largura de banda e latência [173];
- **Extensible Messaging and Presence Protocol (XMPP)**: um protocolo baseado em texto suportado por *Extensible Markup Language* (XML) que usa paradigmas de publicação/assinatura e cliente-servidor [187]. É focado em interações de mensagens instantâneas, com boa compatibilidade com outros protocolos, e também oferece alguns outros recursos relevantes como autenticação e criptografia [173];
- **Simple Service Discovery Protocol (SSDP)**: geralmente incluído no protocolo *Universal Plug and Play* (UPnP), o SSDP [188] é um protocolo usado para descobrir outras máquinas na mesma rede onde o protocolo está operando. Isso torna viável a comunicação de vários dispositivos diferentes, sem a necessidade de configurações complexas.

iii) Protocolos Baseados em UDP

O UDP é caracterizado por sua simplicidade e natureza sem conexão [189]. Ao contrário dos protocolos orientados à conexão, tal como o TCP, o UDP não incorpora serviços de verificação e recuperação de erros. Consequentemente, não introduz a sobrecarga associada à abertura, manutenção e fechamento de conexões. No UDP os dados são enviados ao destinatário sem garantia de entrega, o que significa que nenhum mecanismo é implementado para garantir que os dados sejam recebidos pelo destinatário [189]. No contexto

das soluções de descoberta de recursos de *fog computing*, os principais protocolos baseados em UDP são:

- **Constrained Application Protocol (CoAP)**: proposto pela IETF [190], este protocolo é baseado no conceito RESTFull, utilizando verbos HTTP básicos (GET, POST, DELETE e PUT) no processo de descoberta. Não é adequado para uso em dispositivos restritos e com capacidades computacionais limitadas, o que o qualifica para uso em ambientes de *fog computing*, mas apenas nas camadas superiores (*Fog* e *Cloud*);
- **Data Distribution Service (DDS)**: opera sobre protocolos TCP e UDP, e é suportado por um padrão de interoperabilidade centrado em dados em tempo real em um paradigma de publicação/assinatura. Outra característica que permite sua utilização em ambientes como a *fog computing* é a sua escalabilidade, uma vez que suporta descoberta dinâmica.

iv) Protocolos Baseados em Nome

Este estudo amplia o escopo das pesquisas existentes na literatura sobre os protocolos utilizados em soluções de descoberta de recursos em *fog computing*. Uma das contribuições inovadoras deste artigo é a introdução de protocolos baseados em nomes como uma alternativa viável para facilitar a comunicação neste contexto. Um exemplo notável de protocolo pertencente a esta categoria é *Named Data Networking* (NDN), um projeto que começou em 2010 com o objetivo de projetar uma nova arquitetura para a Internet [191].

O NDN difere substancialmente dos paradigmas de comunicação tradicionais, pois emprega conceitos de "interesse" e "dados", ao invés dos convencionais "pacotes de solicitação" e "pacotes de resposta". Em seu processo, o consumidor emite inicialmente um pacote de interesse, identificando com precisão os dados que deseja obter. Posteriormente, quando esse interesse chega a um nó que contém os dados solicitados, um pacote de dados é gerado e transmitido de volta ao consumidor. Este pacote de dados contém informações sobre o nome associado aos dados e o próprio conteúdo dos dados.

4.1.2.6 Qualidade de Serviço (QoS)

Ao considerar os desafios no desenvolvimento de soluções de descoberta de recursos para ambientes de *fog computing*, conforme apresentado na Seção 3.2.2, é possível perceber a necessidade de tornar esse processo mais alinhado às necessidades de QoS, evitando desperdício de recursos e otimizando tempo e custo. Contudo, ainda é possível encontrar na literatura algumas propostas de descoberta de recursos que não levantam esta preocupa-

ção, procurando recursos de forma aleatória e sem critério. Por este motivo, em termos de QoS, as soluções de descoberta de recursos podem ser classificadas em:

- **Com QoS**: eficiência, escalabilidade, confiabilidade, flexibilidade, dinamicidade, e autonomia são alguns exemplos de características de soluções de descoberta desenvolvidas com foco em QoS [28]. Além disso, outras características ainda mais específicas também podem ser exigidas, como latência, distribuição de carga, capacidade computacional, arquitetura etc.;
- **Sem QoS**: embora menos frequente, este tipo de solução tem apenas a simples missão de encontrar recursos, sem qualquer critério de seleção. Porém, este tipo de solução pode ser utilizada em ambientes complexos e desconhecidos para permitir um primeiro mapeamento e posterior seleção.

4.1.2.7 Catálogo de Recursos

Uma vez descobertos os recursos e obtidas as suas descrições, é comum agregar esta informação num Catálogo de Recursos compartilhado para disponibilizar a utilização em outras atividades de gestão de recursos, tais como alocação e monitoração.

O *fog computing* é caracterizado por um grande número de dispositivos, o que pode representar um desafio significativo em termos de catalogação. Goodchild *et al.* [192] identifica que esses desafios estão relacionados à diversidade do sistema de informação, ao aumento da base de usuários e ao volume de dados. A diversidade dos sistemas de informação diz respeito à heterogeneidade dos *fog nodes* e à complexidade da informação a ser registada. O crescimento da base de usuários refere-se à necessidade de manter um banco de dados consistente, que pode ser impactado pelo crescimento contínuo de novos recursos e pela forma como os recursos são catalogados. Dentro deste contexto, propostas de soluções de descoberta de recursos em *fog computing* podem ser classificadas da seguinte forma:

- **Com Catálogo:** nesta solução, há um catálogo de recursos que oferece suporte às atividades de descoberta. Assim, o catálogo de recursos pode ser implementado de duas maneira diferentes:
  - *Interno:* nesta proposta o catálogo de recursos é mantido internamente na solução, seja na memória cache ou em variáveis temporárias;
  - *Externo:* as informações dos recursos são armazenadas em bancos de dados externos à aplicação, suportados por sistemas gerenciadores de banco de dados (SGDB).

- **Sem Catálogo:** também podem existir soluções que não são suportadas por nenhum tipo de catálogo porque tomam decisões com base em informações locais e temporárias.

4.1.2.8 Descrição do Recurso

Uma das características fundamentais da descoberta de recursos é fornecer uma descrição do recurso identificado de forma compreensível para as etapas subsequentes de gerenciamento, permitindo o uso eficaz desses recursos, conforme destacado em [193]. Adicionalmente, num ambiente de *fog computing*, espera-se que haja interação entre diferentes atores, tais como provedores e usuários. A utilização de uma descrição de recursos compreensível para todos estes agentes, ou seja, uma descrição comum, facilitará a troca de informações, aumentando a eficiência global do sistema. Contudo, como observado em [28], esta tarefa não é trivial quando se trata de um ambiente altamente distribuído como o *fog computing*. Isso se deve à alta dinâmica do ambiente e à constante mudança no *status* dos recursos, o que pode levar a inconsistências nas informações fornecidas ou resultar em uma sobrecarga significativa na aplicação para manter os dados sempre atualizados. As propostas existentes na literatura para a descrição de recursos na fase de descoberta podem ser agrupadas em dois grupos principais [28, 144, 193]: baseadas em sintaxe (ou atributos) e baseadas em semântica, conforme descritas a seguir:

- **Baseado em Sintaxe:** geralmente são implementados usando pares de atributos chave-valor, com o processo de descoberta suportado pela correspondência de palavras-chave. Exemplos desse modelo de descoberta incluem a *Web Services Description Language* (WSDL) [194], a *Universal Description, Discovery and Integration* (UDDI) [195] e a *Extensible Markup Language* (XML) [196]. Os autores do artigo [193] destacam algumas limitações dos modelos baseados em sintaxe em ambientes altamente distribuídos e heterogêneos, como *fog computing*. Essas limitações incluem a dificuldade de estabelecimento de acordos entre produtores e consumidores quanto às palavras-chave a serem utilizadas, e a limitada capacidade computacional;
- **Baseado em Semântica:** uma abordagem para superar as limitações baseadas em sintaxe é a adoção de modelos baseados em semântica, que focam na descrição colaborativa e abrangente de recursos, tornando a descoberta e alocação mais eficientes [28]. As ontologias desempenham um papel fundamental nos modelos de base semântica, buscando padronizar a representação dos recursos. Alguns exemplos de ontologias baseadas em semântica incluem a ontologia *Semantic Sensor Network* (SSN) [197], a arquitetura IoT-A [198], o *Resource Description Framework* (RDF) [199], e a Ontology Inference Layer (OIL) [200]. É importante notar que

a escalabilidade destas soluções pode ser um desafio num ambiente de *fog computing*, devido à elevada distribuição geográfica e ao grande número de dispositivos envolvidos [28].

4.1.2.9 Busca

A essência da descoberta de recursos está no processo de busca e pode ser afetada pelo algoritmo (Seção 4.1.2.1), e também pela existência ou ausência de critérios de QoS (Seção 4.1.2.6). Num cenário de *fog computing*, três tipos de pesquisa são possíveis:

- **Direcionada**: a solução de descoberta é clara sobre o que está sendo pesquisado, as variáveis de ambiente e as informações que irá obter. *Best-first* [201] é um exemplo de método usado neste tipo de pesquisa;
- **Semi-direcionada**: quando apenas se conhece alguma característica do recurso procurado, como a faixa de IP ou o nome do *host*, por exemplo. Alguns algoritmos de busca gulosos podem ser listados nos exemplos desse tipo de busca;
- **Não direcionada**: é usado quando nada se sabe sobre os *fog nodes*, nem o número de nós que podem ser encontrados, nem o escopo do ambiente. *Breadth search* [202], *first search* [203], e *flooding* [204] são exemplos neste grupo.

4.1.2.10 Topologia

A topologia de descoberta refere-se à configuração estrutural das soluções de descoberta de recursos em relação à distribuição de seus componentes e ao fluxo de dados, sendo considerada um aspecto extremamente relevante por diversos autores [28, 142, 143]. As arquiteturas de descoberta de recursos podem ser categorizadas em três tipos principais: centralizadas, distribuídas e hierárquicas, conforme ilustrado na Figura 4.2. Além destas abordagens, surge também um modelo híbrido que aparece quando duas ou mais das arquiteturas acima mencionadas são usadas em combinação, conforme indicado em [142].

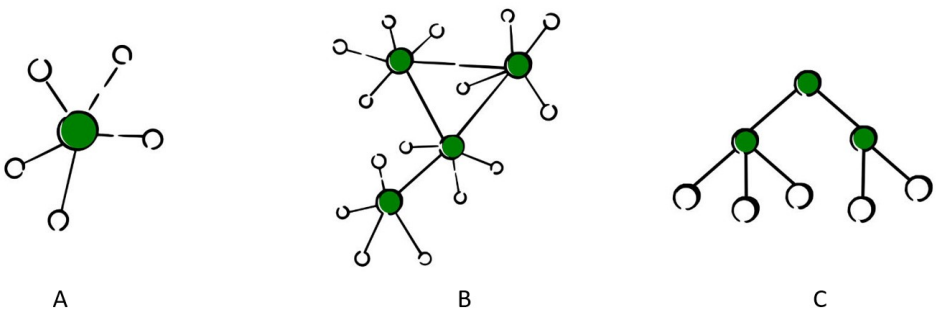

Figura 4.2: Topologia: a) Centralizada; b) Distribuída; c) Hierárquica.

- **Centralizada**: neste contexto, toda a informação relevante é armazenada centralmente, acessível por todos os nós. Este modelo tem a vantagem de ter sempre a fonte da informação identificável e de permitir uma realocação de serviços ou recursos com relativa facilidade. No entanto, esta centralização também pode ser vista como um potencial gargalo e um ponto único de falha [28];
- **Distribuída**: nesta abordagem, os nós não precisam contar com uma fonte central para análise das consultas; em vez disso, eles obtêm informações diretamente sobre os recursos necessários. Isso torna o sistema mais resiliente a falhas e notavelmente escalonável. Contudo, dependendo do cenário de uso, um alto grau de distribuição pode resultar rapidamente em sobrecarga de comunicações [22];
- **Hierárquica**: a propagação de informações relevantes ocorre hierarquicamente por cada camada estar conectada à camada abaixo dela. Este método evita a necessidade de disseminação de informações pela rede, reduzindo a sobrecarga de comunicação associada às arquiteturas distribuídas. No entanto, esta abordagem pode introduzir atrasos devido ao roteamento de mensagens, resultando em níveis mais baixos acessando informações mais tarde do que níveis mais altos [28];
- **Híbrida**: ocorre quando dois ou mais dos modelos anteriores são usados simultaneamente [29].

4.1.3 Análise dos Artigos sobre Descoberta de Recursos

Esta seção visa responder ao QP2 desta pesquisa, que trata do nível de maturidade e completude em que se encontram as soluções atuais para descoberta de recursos em *fog computing*. A Tabela 4.2 apresenta uma análise dos 14 artigos selecionados pelos critérios apresentados na Seção 4.1.1 em relação à taxonomia proposta. Antes de entrar nos detalhes de cada trabalho analisado, é importante apontar algumas informações que podem ser extraídas a partir da Tabela 4.2:

- **Algoritmo**: o uso de algoritmos não determinísticos é mais frequente. Dentre eles, os mais comumente encontrados são algoritmos heurísticos e meta-heurísticos, bem como algoritmos genéticos;
- **Inicialização**: iniciar a solução de forma reativa é o método adotado por metade das propostas analisadas;
- **Camadas**: as soluções que se baseiam nas três camadas da arquitetura (*Cloud*, *Fog* e IoT) são as mais adotadas, embora algumas soluções já estejam sendo projetadas para utilizar apenas a camada de *Fog*, que pode ser muito positivo para aliviar a camada de *Cloud* e reduzir o consumo de recursos;

Tabela 4.2: Análise de soluções de descoberta de recursos em *fog computing* com base na taxonomia proposta.

Artigo	Ano	Algoritmo	Inicialização	Camadas	Propagação	Protocolo	QoS	Catálogo	Descrição	Busca	Topologia
[205]	2017	D	Proativa	1 camada	Com Agente	UDP	Sim	Sim	Sintaxe	Direcionada	Distribuída
[206]	2018	D	Proativa	3 camadas	Sem Agente	TCP	Sim	Sim	Sintaxe	Não-direcionada	Centralizada
[207]	2018	D	Reativa	IoT-*Fog*	Sem Agente	TCP	Sim	Sim	Sintaxe	Direcionada	Distribuída
[208]	2019	D	Reativa	1 camada	Sem Agente	UDP	Sim	Não	Sintaxe	Não-direcionada	Distribuída
[209]	2019	ND	Proativa	3 camadas	Sem Agente	TCP	Sim	Sim	Sintaxe	Semi	Distribuída
[210]	2020	D	Reativa	1 camada	Com Agente	TCP	Sim	Sim	Sintaxe	Semi	Centralizada
[211]	2020	D	Reativa	3 camadas	Com Agente	UDP	Sim	Sim	Sintaxe	Não-direcionada	Híbrida
[212]	2021	ND	Reativa	IoT-*Fog*	Sem Agente	Name	Sim	Não	Semântica	Não-direcionada	Distribuída
[213]	2021	ND	Proativa	3 camadas	Sem Agente	LPWAN	Sim	Sim	Semântica	Semi	Hierárquica
[214]	2021	ND	Proativa	3 camadas	Sem Agente	TCP	Sim	Não	Sintaxe	Não-direcionada	Distribuída
[215]	2021	ND	Reativa	3 camadas	Com Agente	TCP	Sim	Não	Sintaxe	Direcionada	Centralizada
[216]	2021	ND	Reativa	IoT-Fog	Sem Agente	TCP	Sim	Sim	Sintaxe	Não-direcionada	Centralizada
[217]	2022	ND	Proativa / Reativa	Fog-Cloud	Sem Agente	Nome	Sim	Não	Sintaxe	Semi	Hierárquica
[218]	2022	ND	Reativa	3 camadas	Com Agente	TCP	Sim	Não	Semântica	Não-direcionada	Distribuída
[219]	2023	D	Proativa	3 camadas	Sem Agente	UDP	Não	Sim	Semântica	Semi	Distribuída
[220]	2024	ND	Proativa	IoT-Fog	Sem Agente	TCP	Sim	Não	Semântica	Não-direcionada	Distribuída

D = Determinística ND = Não-determinística

- **Propagação**: o uso de agentes foi descartado em 64% das soluções. Essa tem sido uma tendência na computação, pois um agente sempre gera uma sobrecarga no recurso que o hospeda, ainda mais em *fog nodes* que possuem restrições de recursos computacionais;
- **Protocolo**: Coap [190], MQTT [186] e variações do protocolo HTTP como Chord [221], Pastry [222] e Kademlia [223] são comumente encontrados nas soluções analisadas. Em menor escala estão os protocolos baseados em nomes, como NDN [191];
- **QoS**: todas as propostas apresentam soluções de descoberta com critério de QoS definido, sendo a latência o principal critério. Isso é compatível com *fog computing* que tem como objetivo dar suporte a aplicativos que exigem baixa latência e não podem ser atendidos pela *cloud computing*;
- **Catálogo de Recursos**: mais da metade (57%) das propostas contam com um catálogo de recursos para manter dados sobre os recursos descobertos;
- **Descrição dos Recursos**: a maioria das propostas (78%) utiliza uma descrição sintática dos recursos. O método mais utilizado para isso é o *Web Services Description Language* (WSDL) [194];
- **Search**: o método de busca não direcionada é utilizado em 50% das soluções, enquanto os métodos direcionado e semi-direcionado foram adotados em 21% e 29%, respectivamente. De certa forma, isso também é muito consistente com o modelo de

fog computing, pois é um ambiente altamente heterogêneo e distribuído, não sendo possível, em sua maioria, conhecer todo o ambiente a ser pesquisado;

- **Topologia**: 50% das propostas utilizam arquitetura distribuída. Isso está bastante aderente ao ambiente de *fog computing*, que possui alta distribuição geográfica e, consequentemente, a arquitetura distribuída como majoritária.

Além das informações apresentadas na Tabela 4.2 e desta análise que sumariza cada atributo avaliado, os 14 trabalhos serão detalhados a seguir.

No trabalho apresentado por Tanganelli *et al.* [205], os autores usam uma arquitetura distribuída para fornecer descoberta de recursos para aplicações IoT com base no diretório de recursos CoRE e no protocolo CoAP. Além disso, esse trabalho utilizou a federação de recursos por meio de uma sobreposição P2P implementada por uma *eXtensible Metadata Hash Table* (XMHT) estendendo a *Distributed Hash Table* (DHT), com o objetivo de favorecer o processo de identificação de recursos. A avaliação da proposta foi realizada em ambientes controlados e também em simuladores, indicando uma redução na latência para aplicações que foram executadas em *fog computing* quando comparada à *cloud computing*.

Em [206] os conceitos de DHT e P2P também foram utilizados para automatizar o processo de descoberta de recursos em IoT. Na proposta deles, um módulo de serviço de descoberta armazena as informações necessárias para provisionamento na tabela DHT. Para isso, são utilizados *templates* contendo informações sobre as possíveis capacidades computacionais de alocação para cada nó, seja ele na *fog computing* ou na *cloud computing*. Dessa forma, cada nó conhece a quantidade de recursos computacionais alocados para uma determinada aplicação e tem uma visão aproximada da quantidade de recursos ainda disponíveis para provisionamento de outros serviços. Os testes simulados da proposta foram realizados considerando ambientes de cidades inteligentes, e demonstraram taxa de assertividade na alocação do recurso adequado acima de 99%.

Em [207] os autores usam uma linguagem de modelagem de API RESTful para descrever recursos e um protocolo de modelo de especificação *Open Communication Foundation* (OCF) para comunicação entre *fog nodes*. Além disso, eles contam com um catálogo de recursos do servidor para armazenar informações a serem consultadas em etapas posteriores de gerenciamento de recursos.

O uso de DHT e P2P também foi proposto por [208], acrescentando uma funcionalidade para gerar identificadores que visam garantir a privacidade dos *fog nodes*. Além disso, os autores apresentaram um modelo de propagação de endereços utilizando busca local em vez de busca global, reduzindo o *overhead* da rede de acordo com os testes realizados.

Os autores do [210] trazem propostas para a etapa de descoberta de recursos para *fog computing* baseada no conceito de *Sieve, Process, and Forward* (SPF) [224]. A proposta

é composta por quatro etapas principais para a descoberta e gerenciamento centralizado do recurso computacional. Para isso, utilizaram o protocolo MQTT [185] para troca de informações, com o objetivo de indicar, em tempo de execução, quais nós são mais adequados para a execução de uma tarefa. Para a aplicação da proposta foram utilizados cenários de assistência humana e recuperação de desastres.

Murturi *et al.* [209, 214] apresentam uma proposta de descoberta contínua de recursos por meio da ampliação da área coberta pela implantação de nós, de forma descentralizada, uma vez que os próprios nós trocam informações por meio de metadados sobre os recursos disponíveis em seu escopo utilizando DHT e P2P. Essas informações trocadas entre nós compreendem identificação, conectividade, capacidade, acessibilidade, informações de saída, localização e domínio. Com isso, os autores criaram o que chamam de "vizinhança", permitindo que os próprios nós indiquem o melhor recurso para a execução de uma tarefa.

No artigo [211], os autores propõem uma solução distribuída de descoberta de recursos para plataformas de *fog computing* usando o diretório de recursos de ambientes RESTful restritos (CoRE) [225]. Além disso, também são utilizados microsserviços alocados em containers, bem como o protocolo CoAP [186] para permitir a alocação sob demanda, contando com uma arquitetura híbrida, ou seja, que utiliza recursos de arquiteturas centralizadas e distribuídas. As avaliações foram realizadas em cenários simulados e reais, mostrando redução na latência das consultas realizadas.

Kondo *et al.* [212] trouxe a aplicação do NDN [191], indicando que este protocolo é adequado para descoberta de recursos em ambientes IoT e *fog computing*. Considerando que outras propostas podem direcionar a utilização de recursos que já estão ocupados por outras tarefas, os autores utilizam NDN para direcionar tarefas apenas para nós que podem, de fato, executar a tarefa. Para isso, o próprio protocolo traz informações sobre o recurso, evitando sobrecarga na arquitetura para obtenção dessas informações e, consequentemente, reduzindo o tempo de busca pela indicação do nó mais adequado.

Uma estrutura para a integração entre IoT, *fog computing* e *cloud computing* é apresentada em [213]. Nesta estrutura, o processo de descoberta de recursos é apenas um componente de várias outras funcionalidades, como o reagendamento. A descoberta de recursos proposta pelos autores utiliza um algoritmo genético que visa apenas encontrar novos dispositivos que possam ter entrado no ambiente e registrá-los em um diretório de recursos, ou seja, é uma abordagem ativa de busca e registro de recursos.

Para apoiar um caso de uso específico para cuidados de saúde, os autores do artigo [215] adotam um algoritmo heurístico para obter resultados locais ideais na descoberta de recursos, com a latência como um requisito de qualidade de serviço. A descrição do recurso é baseada em arquivos no formato JSON. A arquitetura é distribuída entre as camadas

de *Cloud*, *Fog* e IoT e a proposta é baseada em uma busca reativa com distribuição de um agente nos *fog nodes*.

A descrição dos recursos é feita por WSDL e um algoritmo meta-heurístico que busca *fog nodes* com tempo de resposta, função de aptidão e critérios de custo [216]. O trabalho [217] utiliza os benefícios do protocolo NDN [191] para apresentar uma proposta que seja adaptável a modelos de inicialização proativos e reativos. Finalmente, o artigo [218] usa um algoritmo genético denominado *KM-gossip* para procurar recursos para reduzir a sobrecarga ambiental.

Uma abordagem de descoberta de recursos baseada em redes P2P é apresentada em [219], usando o *fog computing* como uma camada para reduzir o tempo de descoberta pela implementação DHT do tipo Pastry, e um modelo IoT semântico para descrever os recursos. Mais recentemente, no artigo [220] é apresentada uma solução de descoberta de recursos focada no provisionamento de serviços para o futuro 6G *Edge-Cloud-Continuum*, usando um zoneamento de sobreposição dinâmica para otimizar o processo de descoberta de recursos com base no aproveitamento de algoritmos de Aprendizagem de Máquina para antecipar o comportamento do usuário e do processo para instanciar o serviço.

Por fim, ao concluir a análise dos artigos selecionados é possível concluir qual é o nível de maturidade e a completude em que se encontram as soluções atuais para descoberta de recursos em *fog computing*. Para tanto, com os trabalhos analisados, é possível projetar uma solução genérica de descoberta de recursos que é baseado em uma solução de topologia distribuída que considera as três camadas (*Cloud* - *Fog* - IoT), que utiliza um catálogo de recursos com descrição sintática de recursos, iniciado de forma reativa, conduzido por um protocolo baseado em TCP, e com um algoritmo determinístico que procura recursos com base em um critério de QoS. A busca é feita de forma não direcionada e sem a necessidade de um agente no *fog node* a ser descoberto.

4.1.4 Desafios para Descoberta de Recursos

Esta seção responde à terceira questão de pesquisa desta revisão, a qual é "QP3 - Quais são os desafios que ainda precisam de atenção na academia?". Nos próximos parágrafos serão discutidos alguns desses desafios, relacionando-os com as categorias da taxonomia e apresentando algumas possíveis direções para superá-los.

Assim, existem várias considerações e desafios significativos relacionados à implementação de soluções de descoberta de recursos na *fog computing*. Alguns desses desafios incluem:

- **Segurança e privacidade**: a segurança arquitetônica é uma preocupação crítica, abrangendo aspectos como criptografia e canais de comunicação seguros. No

entanto, a limitação de recursos nos *fog nodes* pode restringir técnicas de segurança utilizáveis, facilitando potencialmente as invasões. Aspectos como controle de acesso, criptografia de dados, integridade contextual e mecanismos de isolamento de dados sensíveis são essenciais para evitar violações de segurança [226]. É importante notar que estas medidas de segurança podem ser herdadas de outros processos de gestão de recursos, como monitoramento ou orquestração [143];

- **Gerenciamento de dados armazenados**: lidar com dados coletados na descoberta de recursos é um grande desafio. Armazenar todos esses dados localmente pode sobrecarregar a rede e os servidores, inviabilizando a abordagem devido a restrições de armazenamento. Portanto, estratégias de gestão de dados devem ser implementadas para equilibrar riscos e benefícios [150, 227];
- **Heterogeneidade e alta distribuição**: o *fog computing* envolve uma grande variedade de dispositivos, desde *Single Board Computer*, como Raspberry Pi, até dispositivos mais potentes. A falta de padronização de hardware em *fog computing* pode aumentar os desafios de compatibilidade de pilhas de software com diferentes dispositivos [228, 229]. A técnica *over-the-air* (OTA) [230] é uma solução possível para implantar e atualizar software em tempo real, conforme necessário. Além disso, em implantações em larga escala, a sincronização de relógios entre dispositivos pode ser um desafio significativo [231, 232];
- **Alta diversidade de dados**: a heterogeneidade dos dispositivos se reflete nos dados gerados, que podem apresentar variedade de formatos e falta de padronização. Assim sendo, lidar com essa diversidade de dados é um desafio crítico que muitas das propostas não abordam de forma abrangente;
- **Falta de ferramentas de simulação abrangentes**: atualmente, a falta de simuladores abrangentes para apoiar o desenvolvimento de soluções de descoberta de recursos em *fog computing* é uma limitação. Muitos simuladores existentes concentram-se em categorias básicas, enquanto as soluções podem exigir recursos mais específicos. Isto requer o uso de vários simuladores ou o desenvolvimento de extensões personalizadas, o que pode ser demorado e exigir esforço adicional [233–235];
- **Dinamicidade**: os requisitos de recursos dos usuários em *fog computing* podem mudar ao longo do tempo, e muitas propostas não consideram essa dinamicidade. Soluções preditivas baseadas em Aprendizado de Máquina e Inteligência Artificial podem ser uma forma de enfrentar esse desafio [129];
- **Poder computacional limitado**: a limitação do poder computacional em *fog nodes* é um aspecto frequentemente esquecido. Muitas propostas só podem ser

executadas em servidores poderosos na camada de *Cloud*, o que pode gerar uma sobrecarga significativa;

- **Conexão Limitada**: a suposição de conexões de alta velocidade pode não refletir a realidade em todas as situações, sendo necessário considerar cenários com conexões mais limitadas.

Estes desafios exigem soluções criativas e abordagens específicas para o desenvolvimento de soluções eficazes de descoberta de recursos em *fog computing*. No Capítulo 5 será apresentada uma proposta com este objetivo.

4.2 Alocação de Recursos

Para a análise das publicações sobre a alocação de recursos para *fog computing*, foi realizada uma revisão da literatura baseada nos trabalhos [11] e [12]. As etapas para o desenvolvimento deste trabalho são apresentadas na Figura 4.3, e serão detalhadas nas próximas subseções.

Figura 4.3: Etapas da metodologia de revisão sistemática da literatura [11] e [12].

4.2.1 Metodologia de Pesquisa

Com base em nossa definição sobre alocação de recursos, e entendendo que é necessário conhecer as propostas existentes na literatura sobre este tema, foi possível determinar as Questões de Pesquisa (QP) a serem respondidas por este trabalho. Isso é importante para ajudar a determinar o que está sendo pesquisado, quais resultados devem ser alcançados, e orientar a revisão sistemática da literatura. Para atingir esse objetivo foram formuladas seis QPs, a saber:

- **QP1: Quais são as métricas utilizadas?** - esta questão visa encontrar as principais métricas utilizadas na literatura até o momento, e as respostas auxiliarão na análise da relevância de cada uma delas para a melhoria do processo de alocação de recursos;

- **QP2: Quais são as técnicas utilizadas?** - a resposta a esta pergunta mostrará as técnicas mais comuns utilizadas na literatura até o momento, e indicará possíveis tendências e lacunas a serem estudadas;
- **QP3: Quais camadas de arquitetura são consideradas?** - considerando a arquitetura de *fog computing* apresentada na Figura 2.2, as respostas a esta pergunta mostrarão as tendências e a relevância de cada uma delas nas abordagens de alocação de recursos;
- **QP4: Quais são as técnicas de virtualização utilizadas?** - esta questão de pesquisa ajudará a categorizar as propostas de acordo com as técnicas de virtualização mais usadas na literatura;
- **QP5: Como são avaliadas as abordagens propostas?** - esta questão visa apresentar as formas pelas quais as propostas de alocação de recursos estão sendo avaliadas, atualmente, pelos trabalhos da literatura;
- **QP6: Quais são os casos de uso mais usados?** - esta pergunta ajudará a identificar quais são os domínios mais comuns nos quais a *fog computing* está sendo aplicada atualmente, e destacar novas áreas a serem estudadas.

A primeira questão de pesquisa (QP1) refere-se às métricas e requisitos passados como entrada. As técnicas (QP2), as camadas cobertas na arquitetura (QP3) e o modelo de virtualização (QP4), formam o núcleo da etapa de alocação de recursos, e são essenciais para entregar o recurso alocado como saída. Por fim, QP5 e QP6 foram escolhidas para apresentar as principais ferramentas de avaliação utilizadas para validar as propostas de alocação de recursos e os casos de uso aplicados mais comuns, respectivamente.

O processo de pesquisa incluiu a seleção das bases de busca a serem utilizadas, o período e a elaboração da *string* de busca. Para esta pesquisa, as bases de dados Scopus[5], Web of Science[6], ACM Digital Library[7] e IEEE XploreLibrary[8] foram utilizadas como fontes de pesquisa. Essas bases foram escolhidas por trazerem publicações de diferentes editoras, como Elsevier, IEEE, ACM, MDPI, etc. Também foi especificado o período entre 2012 (ano da primeira publicação sobre *fog computing* [23]) e agosto de 2022 (momento da realização desta pesquisa).

A seguinte *string* de pesquisa foi usada "*TITLE-ABS-KEY((fog) AND (resource) AND (management OR allocation OR placement OR scheduling sharing OR caching OR offloading OR selection OR "load balancing" OR provisioning))*". Esta *string* foi construída

[5]scopus.com
[6]webofknowledge.com
[7]dl.acm.org
[8]ieeexplore.ieee.org

com o objetivo de trazer resultados aderentes às questões de pesquisa, apresentadas na Seção 4.2.1. Conforme apresentado na Seção 3.2, fica claro que as publicações usam termos diferentes para descrever os mesmos conceitos. Isso justifica porque foram usados outros termos ao invés de apenas "alocação de recursos" na *string* de busca.

Após a utilização da *string* de busca nas bases de dados relacionadas, foi necessário aplicar os critérios de inclusão e exclusão. Assim, os critérios de inclusão definidos para esta pesquisa foram publicações redigidas em inglês; publicações em sites públicos da Internet; e que apresentassem modelos arquitetônicos, técnicas ou métodos aplicados à alocação de recursos computacionais, especificamente, em *fog computing*. Da mesma forma, foram definidos como critérios de exclusão aqueles que não atendessem a nenhum dos critérios de inclusão listados acima; bem como artigos duplicados nas bases de dados; artigos que não estivessem relacionados à *fog computing*; publicações que não fossem orientadas a processos de alocação de recursos computacionais; ou que contivessem apenas conceitos teóricos.

Inicialmente, a pesquisa realizada nas duas bases de dados retornou um total de 1.182 publicações. Após os critérios de inclusão e exclusão serem aplicados, e considerando os conteúdos encontrados nos títulos e resumos dessas publicações, esse número foi reduzido para 203 publicações. Um segundo filtro foi aplicado considerando a adesão ao texto completo do artigo, resultando em 108 publicações a serem analisadas.

Assim, é importante notar que a revisão sistemática realizada neste trabalho analisou um valor substancialmente superior aos demais publicados na literatura, pois o artigo que mais analisou foi o publicado por Jamil *et al.* [31] com 49 artigos, e o segundo foi a publicação de Fahimullah *et al.* [236] com 34 artigos.

Considerando as 108 publicações analisadas, uma informação relevante é que considerando apenas o título do artigo, o termo "alocação de recursos" foi o mais encontrado quando comparado com todos os demais termos pesquisados na *string* de busca, conforme apresentado na Figura 4.4. Isso mostra mais uma vez que essa etapa é amplamente utilizada na literatura.

Uma vez contextualizado o processo de pesquisa dos trabalhos relacionados, a próxima seção apresentará os resultados obtidos e uma análise sobre os artigos em relação às questões de pesquisa elaboradas.

4.2.2 Resultados da Revisão da Literatura sobre Alocação de Recursos

Esta seção apresenta os resultados obtidos a partir das 108 publicações selecionadas. A Figura 4.5 apresenta uma visão geral das questões de pesquisa a serem discutidas nesta

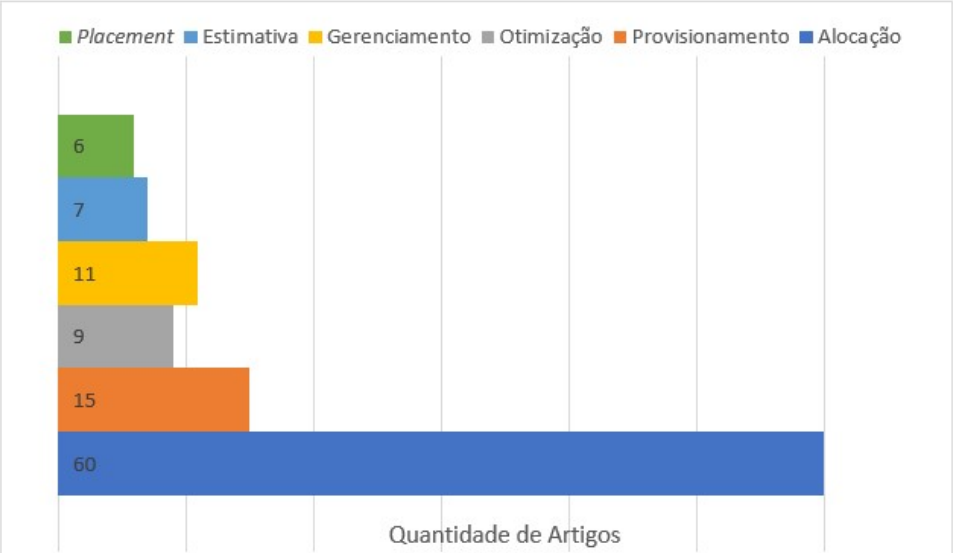

Figura 4.4: Termos encontrados nos títulos dos artigos.

seção.

4.2.2.1 Métricas

A primeira questão de pesquisa (QP1) a ser respondida refere-se às métricas mais utilizadas para alocação de recursos em um ambiente de *fog computing*. Uma métrica está quase sempre vinculada a um critério, dando origem ao objetivo de alocação, que por sua vez é indicado em termos de QoS [124]. Por exemplo, a métrica "custo" está vinculada ao critério de "redução", gerando o objetivo de "minimizar o custo".

Analisando as publicações selecionadas, foram observadas as seguintes métricas de alocação de recursos: utilização de recursos, custo, latência, energia, experiência do usuário e tempo de execução. Esses trabalhos estão detalhados na Tabela 4.3, que também fornece a porcentagem de artigos analisados que cada métrica representa.

A partir da análise das publicações apresentadas na Tabela 4.3 e respondendo QP1, foi possível identificar que a métrica mais abordada na alocação de recursos é o custo, pois foi usada em 31 das 108 publicações (28,7%). Porém, outras métricas também são utilizadas, como latência, em 24,1% dos trabalhos; e tempo de execução em 22,2% deles. Nota-se também que alguns trabalhos aparecem em duas classificações diferentes [240, 247, 251, 254, 261, 264, 271, 272, 274, 275, 279, 284, 288, 291–294, 308] ou até em três delas [252], o que é possível quando os autores desses artigos combinam diferentes métricas em suas propostas.

Além das métricas listadas nesta seção, algumas outras métricas podem ser propostas. O tempo de alocação, por exemplo, é considerado uma métrica relevante. Isso porque

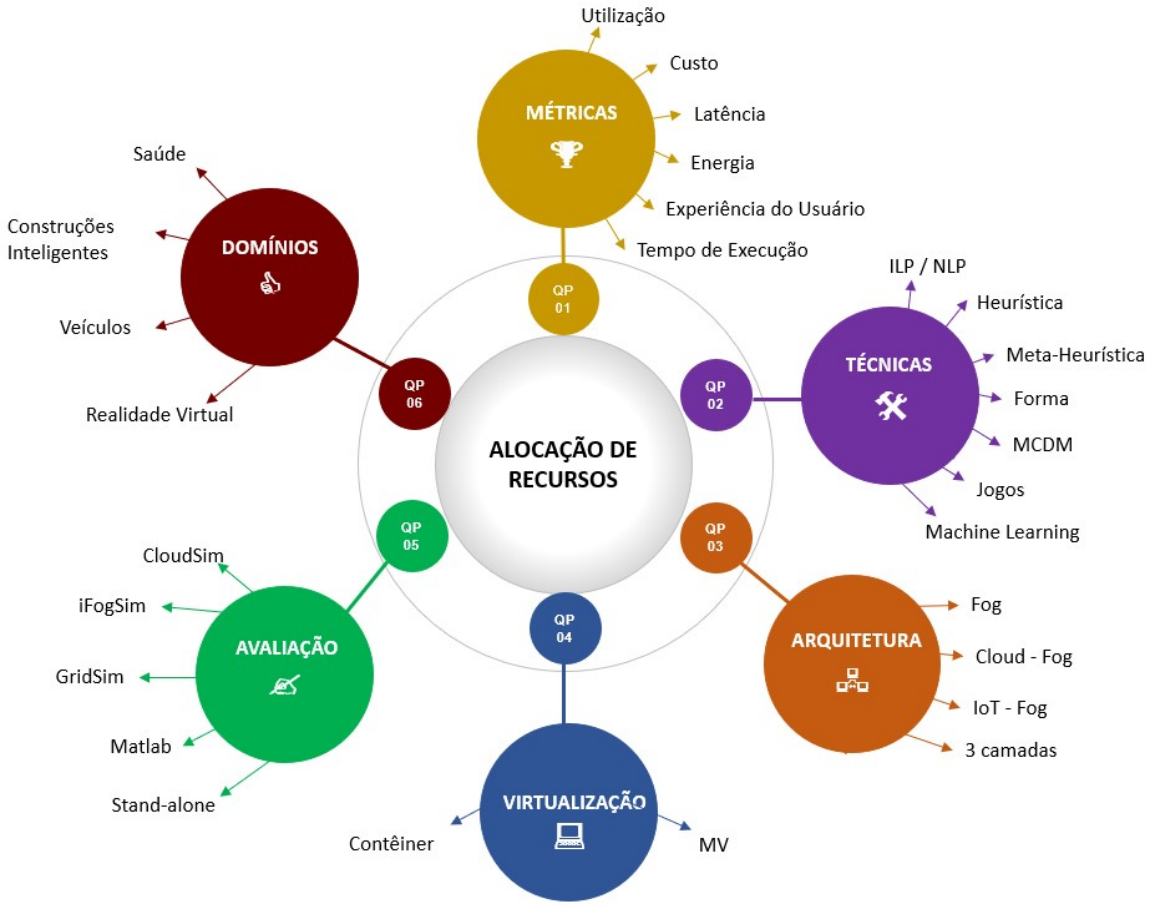

Figura 4.5: Visão geral da revisão sistemática sobre alocação de recursos [1].

essa métrica se refere ao tempo necessário para receber as informações da carga de trabalho, estimar e verificar os recursos necessários, e efetivar a alocação. Considerando a mobilidade e dinâmica das aplicações em ambiente de *fog computing*, um tempo de alocação otimizado desempenha um papel fundamental para garantir uma alta experiência do usuário, e alcançar a qualidade de serviço exigida [27].

Outras métricas que podem ser utilizadas estão relacionadas ao tempo e ao número de migrações entre os *fog nodes* e a *cloud computing*, pois uma alocação eficiente deve manter a quantidade de migrações a mais baixa possível. Assim, propõe-se o uso de algoritmos preditivos, pois eles podem avaliar a estabilidade dos recursos alocados, visando garantir que os recursos estarão disponíveis até o final da execução da carga de trabalho.

4.2.2.2 Técnicas

A QP2 investiga quais são as técnicas mais utilizadas nos trabalhos analisados para alocação de recursos em *fog computing*. A Tabela 4.4 mostra as técnicas utilizadas nos artigos analisados, agrupadas em Programação Linear Inteira (ILP)/Programação Não-Linear

Tabela 4.3: Métricas de alocação de recursos.

Métrica	Artigos	Descrição	%
Utilização de Recursos	[237–257]	Maximizar o uso e a disponibilidade dos recursos, considerando o mapeamento adequado desses recursos e a correta distribuição e alocação, evitando assim o desperdício de poder computacional [131].	19,4%
Custo	[56, 133, 247, 252, 258–284]	Minimizar o custo de alocação de recursos considerando a implementação da infraestrutura, os custos de operação e o custo de alocação das instâncias [122].	28,7%
Latência	[102, 206, 240, 252, 275, 284–304]	Reduzir a latência é um dos objetivos mais significativos de *fog computing* em comparação com outros paradigmas, como a *cloud computing* [23]. Refere-se ao atraso de comunicação para transferir dados entre recursos somado ao tempo necessário para que a tarefa seja processada [31].	24,1%
Energia	[134, 274, 288, 292–294, 305–311]	Os dispositivos de *fog computing* podem usar energia renovável e não renovável. Portanto, a eficiência energética é uma das principais métricas neste contexto, e muitas pesquisas têm sido feitas para atingir essa métrica [122]. Por esse motivo, as técnicas de alocação de recursos com foco na eficiência energética visam minimizar o consumo de energia.	12,0%
Experiência do Usuário	[132, 251, 312–322]	Considera que as demandas podem mudar durante a execução das tarefas, e que os níveis de qualidade do serviço devem ser garantidos ao longo de todo o ciclo de vida do serviço [122]. É afetada por capacidades computacionais (como processador e memória) e características comportamentais (ou seja, confiabilidade e segurança).	12,0%
Tempo de Execução	[254, 261, 264, 271, 272, 279, 291, 308, 323–328, 328–337]	A métrica de tempo de execução está vinculada ao objetivo de reduzir o tempo de entrega e cumprir o prazo especificado para execução da carga de trabalho [122]. Também pode ser referenciado como *makespan*, *runtime*, *throughput* ou *deadline* [31]. Em *fog computing* muitas das abordagens que visam reduzir o tempo de execução estão ligadas à redução da latência ou ao consumo da energia.	22,2%

(NLP), Heurísticas, Meta-heurísticas, Abordagens Baseadas em Forma, *Multiple-Criteria Decision-Making* (MCDM), Abordagens Baseadas em Jogos e Aprendizado de Máquina.

A Programação Linear consiste em um método para resolver problemas de otimização que possuem algumas restrições (injunções), sendo a função objetivo linear em relação às variáveis de controle. O domínio dessas variáveis é composto por um sistema de desigualdades lineares [341]. A principal vantagem da Programação Linear é a flexibilidade para analisar problemas complexos [342]. Uma abordagem de Programação Linear foi usada nos artigos [56, 206, 248, 269, 273, 277, 289, 290, 297, 300, 305–307, 320, 328, 332, 335], que representam 15,7% do total de artigos analisados. Em outros três artigos (2,7% dos artigos analisados), os quais são [134, 241, 251] a Programação Linear Inteira Mista (MILP), que

Tabela 4.4: Técnicas de alocação de recursos.

Técnica	Artigos	Descrição	%
ILP NLP	[56, 134, 206, 241, 248, 251, 262, 267, 269, 273, 277, 289, 290, 297, 300, 305–307, 320, 328, 328, 332, 335]	A ILP resolve problemas de otimização que possuem restrições, com função objetivo linear em relação às variáveis de controle. A NLP resolve um problema de otimização na qual algumas das restrições ou a função objetivo são não lineares [338].	21.3%
Heurística	[102, 132, 133, 239, 249, 260, 262–264, 270, 271, 276, 278, 280, 281, 283, 285, 286, 291, 299, 301, 308, 310–317, 319, 329, 330, 334]	Algoritmos heurísticos são aqueles que não garantem encontrar a solução ótima para um problema, mas são capazes de retornar uma solução de qualidade em tempo adequado para a aplicação.	31.5%
Meta-heurística	[244, 250, 266, 274, 287, 293, 294, 307, 309, 324]	Visa encontrar soluções ótimas ou quase ótimas para o problema de alocação de recursos em *fog computing* dentro de um período de tempo razoável.	9.2%
Forma	[267, 272, 295, 296, 333]	Algoritmos baseados em forma são aqueles que direcionam soluções para o parâmetro definido em sua programação, como melhor alocação, melhor ajuste, tarefa mais curta ou primeiro ajuste [339].	4.6%
MCDM	[237, 246, 253–257, 284, 303, 304, 322, 325, 336, 337]	É composto por métodos de apoio à decisão para diversos critérios, indicando modelos adaptativos para tarefas de escalonamento, seleção e alocação de recursos, atribuindo pesos aos seus critérios como forma de escolher os recursos mais coerentes com a QoS requerida.	12,9%
Jogos	[238, 245, 259, 265, 268, 279, 318, 321, 326, 327, 331]	Usa modelos matemáticos para tomar decisões ótimas em condições de conflito. Cada jogador tem interesse ou preferências para cada situação do jogo [340].	10.2%
Aprendizado de Máquina	[240–243, 247, 252, 258, 261, 275, 282, 288, 292, 298, 302, 323, 328]	Tem como objetivo explorar, analisar e encontrar significado em conjuntos de dados complexos. Eles abrangem outras técnicas, como *Deep Learning* e *Reinforcement Learning.*	14.8%

é uma extensão da ILP para quando algumas variáveis de decisão não são discretas, foi utilizada para resolver o problema de alocação de recursos.

A Programação Não Linear, por outro lado, é o processo de resolver um problema de otimização definido por um sistema de igualdades e desigualdades, chamado de restrições, sobre um conjunto de variáveis reais cujos valores são desconhecidos, com uma função objetivo a ser maximizada ou minimizada, e onde algumas das restrições ou a função objetivo são não lineares [338]. Esse tipo de abordagem foi usado para tratar o problema de alocação de recursos em *fog computing* nas sete publicações: [248, 262, 269, 300, 307, 328, 332].

As heurísticas foram as técnicas mais utilizadas nos artigos analisados. Nesse tipo de solução, as decisões são baseadas apenas nas informações disponíveis, sem se preocupar com os efeitos futuros de tais decisões, fazendo assim a escolha localmente ótima em cada

etapa da execução. O objetivo é encontrar uma solução global boa, e não necessariamente ótima. Portanto, são considerados fáceis de implementar e eficientes [343]. Nos artigos analisados sobre alocação de recursos em *fog computing*, alguns autores (12%) propuseram novos algoritmos heurísticos, tais como em [133, 249, 264, 286, 299, 301, 308, 310, 311, 315, 329,330,334]. Também foram utilizados alguns métodos heurísticos conhecidos, tais como as abordagens baseadas em preço (10,1% do total) [132, 260, 270, 281, 283, 285, 313, 314, 316, 317, 319], Algoritmos Gulosos (2.7%) [263, 280, 312], a abordagem de otimização de Lyapunov (2.7%) [102, 278, 291] e o Algoritmo Húngaro (0.92%) [239].

Da mesma forma, as técnicas de meta-heurística combinam heurísticas básicas em um nível de estrutura mais alto, mas também com o objetivo de encontrar uma solução ótima ou quase ótima em um tempo de execução limitado [344]. Dentro desta categoria estão os Algoritmos Evolutivos, que se baseiam nos princípios da evolução natural, mantendo uma população de candidatos à solução ao longo da pesquisa [345]. Após a inicialização, novas soluções são geradas iterativamente, selecionando boas soluções da população, cruzamento e mutação. Novos indivíduos também são avaliados e inseridos na população, geralmente substituindo as piores soluções. O algoritmo normalmente é interrompido após um certo número de iterações, retornando a melhor solução encontrada naquele período [345]. Algoritmos evolutivos para alocação de recursos em ambientes de *fog computing* foram usados em oito artigos [244, 250, 266, 274, 287, 293, 294, 324]. Além disso, outras meta-heurísticas foram encontradas nos artigos revisados, como o algoritmo *Particle Swarm Optimization* usado em [294] e o *Ant Colony Optimization* adotado em [309].

Considerando as abordagens baseadas em Forma, o algoritmo *First-Fit* foi utilizado em três artigos [295, 296, 333]. Nesta técnica, o problema de alocação é resolvido entregando o primeiro recurso que satisfaz aos parâmetros solicitados, independentemente de haver opções melhores. Nesse sentido, o algoritmo *Shortest Job First* utilizado no artigo [272], prioriza a alocação das menores requisições. Apenas o algoritmo *Best-fit*, apresentado no artigo [267], busca a melhor alocação considerando as entradas e os recursos disponíveis.

Com relação às técnicas de MCDM, no artigo [237] os autores utilizaram o método PROMETHEE (do inglês, *Preference Ranking Organization Method for Enrichment Evaluation*), enquanto em [284] e [255] os autores utilizaram o método ELECTRE (do inglês, *Elimination Et Choice Translating Reality*). O método chamado AHP (do inglês, *Analytic Hierarchy Process*) foi usado nos artigos [303, 304, 322, 336]. Embora tenham conseguido atender a alguns critérios de QoS estabelecidos, o método não é capaz de garantir que os requisitos mínimos sejam atendidos. O mesmo ocorre com o método conhecido como *Technique for Order of Preference by Similarity to Ideal Solution* (TOPSIS) que foi usado nos artigos [246, 256, 257, 337], também limitado em sua capacidade de alcançar uma qualidade de entrega refinada.

A abordagem baseada em jogos usa modelos matemáticos para tomar decisões ótimas em condições de conflito. Com isso, um elemento básico é o conjunto de jogadores que dele participam, e cada jogador possui estratégias. A escolha de uma estratégia reflete uma situação entre todas as situações possíveis. Cada jogador tem um interesse ou preferência para cada situação do jogo [340]. Entre as publicações analisadas, a teoria dos jogos de Stackelberg [346], em que o líder se move primeiro e os demais jogadores se movem em sequência foi a mais utilizada (2,7% do total de trabalhos) [268, 279, 327].

Por fim, as técnicas de Aprendizado de Máquina envolvem algoritmos que visam aprender sobre o ambiente de *fog computing*, os usuários e os comportamentos de alocação de recursos para prever novas solicitações. Nesse contexto, existem técnicas de *Deep Learning* [240, 258], *Deep Reinforcement Learning* [242, 275, 282, 298], *Bayesian learning* [247, 252], e *Fuzzy Logic* [288, 302].

4.2.2.3 Camadas da Arquitetura Utilizadas

A análise de quais camadas de arquitetura são consideradas nas abordagens de alocação de recursos em *fog computing* é necessária para responder à questão de pesquisa QP3. A maioria dos trabalhos analisados considerou a arquitetura de *fog computing* dividida em três camadas (IoT, *Fog* e *Cloud*), conforme apresentado na Seção 2.3. Portanto, a alocação de recursos em *fog computing* pode ser considerada como um problema de dupla correspondência [26]. As abordagens para resolver este problema podem envolver apenas a Camada de *Fog*, ou a comunicação entre duas (IoT x *Fog* ou *Fog* x *Cloud*), ou mesmo três camadas (IoT x *Fog* x *Cloud*) da arquitetura de *fog computing*. Assim sendo, as publicações analisadas foram classificadas nesses cenários, conforme apresentado na Tabela 4.5.

Existem abordagens que consideram apenas o *link* formado entre os dispositivos do usuário final, localizados na Camada IoT, e os dispositivos da Camada de *Fog*. Depois que esse *link* for estabelecido, eles se conectarão à Camada de *Cloud* como um único grupo de recursos. Embora essas abordagens não desconsiderem a existência e o relacionamento da Camada de *Fog* com a Camada de *Cloud*, elas buscam alternativas para resolver todos os problemas de alocação de recursos na Camada de *Fog*, evitando o encaminhamento de solicitações para a Camada de *Cloud*.

Algumas publicações analisadas focaram na conexão formada entre os dispositivos da Camada de *Fog* e os da Camada de *Cloud*. Os recursos disponibilizados por meio desse relacionamento permitiram que os usuários finais executassem suas cargas de trabalho neles. Os trabalhos neste contexto consideram que os serviços requeridos pelos usuários finais, que estão na Camada IoT, já foram distribuídos na Camada de *Fog* e, com base

Tabela 4.5: Camadas da arquitetura utilizadas.

Camadas	Artigos	Descrição	%
Fog	[102, 237, 249–251, 255, 258, 263, 273, 275, 276, 284, 285, 294, 297, 304, 320, 324, 326, 328, 330, 335]	Usa apenas os recursos disponíveis na Camada de *Fog*.	20.4%
Cloud - *Fog*	[206, 241, 261, 264, 268, 271, 272, 287, 292, 313, 331]	Usa os recursos alocados nas duas camadas superiores na arquitetura de três camadas.	10.2%
IoT - *Fog*	[277–280, 283, 298, 300, 309, 321, 328]	Usa os recursos alocados nas duas camadas inferiores na arquitetura de três camadas.	9.3%
Três Camadas	[132, 242, 244–248, 252–254, 256, 257, 266, 269, 274, 281, 282, 288–291, 293, 295, 296, 299, 301–303, 308, 310, 311, 316, 317, 319, 322, 329, 332–334, 336, 337]	Usa os recursos disponíveis em todas as camadas na arquitetura de três camadas.	49.1%

nessa premissa, precisam ser provisionados utilizando os recursos disponíveis nessa camada e na Camada de *Cloud*.

Há também alguns trabalhos que aplicam suas propostas considerando apenas a Camada de *Fog*. Nesse cenário, todas as requisições devem ser solucionadas pelos recursos disponíveis no ambiente de *fog computing*. No entanto, isso não é usual, pois a *fog computing* é complementar à *cloud computing* e não um paradigma para substituí-la.

Por fim, a maioria dos trabalhos analisados nesta pesquisa tiveram suas propostas abordadas utilizando todas as três camadas da arquitetura *fog computing*. Essa é a abordagem mais comum, pois considera que a Camada de *Fog* é apenas uma camada intermediária para atingir os objetivos definidos (por exemplo, melhorar a QoS). As requisições são geradas na Camada IoT e são totalmente atendidas utilizando não apenas a Camada de *Fog*, mas também na Camada de *Cloud*. No total, 53 das 108 publicações analisadas (49%) consideraram as três camadas para desenvolverem suas propostas.

4.2.2.4 Modelos de Virtualização

A QP4 visa identificar os modelos de virtualização utilizados em abordagens de alocação de recursos em *fog computing*. É importante enfatizar que em *fog computing* a disponibilidade de recursos (como processamento, memória, armazenamento e rede) é essencial. Ao contrário da *cloud computing*, na qual os recursos estão sempre disponíveis, na *fog computing* há uma forte restrição de recursos, pois muitas vezes são dispositivos com baixa capacidade computacional. Um *switch*, por exemplo, tem a função principal de gerenciar

conexões de rede, mas é utilizado pela Camada de *Fog* para fornecer seus recursos computacionais não utilizados para processamento, armazenamento, etc. Diante do exposto, os modelos de virtualização utilizados nos estudos analisados podem ser agrupados em duas categorias, conforme apresentado na Tabela 4.6, e discutido a seguir.

Tabela 4.6: Modelos de virtualização.

Virtualização	Artigos	Descrição	%
MV	[56, 132, 244, 246, 247, 249, 252, 254, 262, 264–267, 269, 272, 277, 281, 286, 288, 290–294, 307, 308, 310, 314, 316, 317, 319, 325, 329, 334]	Explora a virtualização no nível do hardware para que vários sistemas operacionais possam ser executados independentemente em um único recurso físico.	31.5%
Contêiner	[102, 238, 240, 242, 249, 251, 253, 274, 286, 297, 313]	Isola os processos apenas com os pacotes de aplicativos necessários.	10.2%

O conceito de Máquina Virtual (MV) é amplamente utilizado, pois explora a virtualização no nível do hardware para que vários sistemas operacionais possam ser executados de forma independente em um único recurso físico. As instâncias de MVs são executadas em uma camada de abstração, chamada *hypervisor*, que permite o compartilhamento de hardware entre diferentes instâncias [122]. O contêiner é um tipo de virtualização mais leve quando comparado às máquinas virtuais, e oferece virtualização no nível operacional [122]. Os contêineres isolam os processos apenas com os pacotes de aplicativos necessários, e são altamente portáteis em vários *fog nodes*. No artigo [249] os autores apresentam algumas vantagens no uso de contêineres quando comparados a máquinas virtuais, como segue: contêineres iniciam mais rápido que MVs porque não são necessários *hypervisors*; contêineres são melhores que MVs em termos de desempenho; e quanto maior o número de MVs implantadas em um servidor, maior é a degradação do desempenho do servidor.

A utilização de um modelo de virtualização adequado é fundamental para o desempenho da aplicação e o alcance dos objetivos indicados na QoS [9]. O uso de contêineres é mais aderente ao paradigma de *fog computing*, pois é mais leve e dinâmico em relação às máquinas virtuais, favorecendo a mobilidade e se adaptando melhor às restrições de recursos, características deste modelo computacional [7].

4.2.2.5 Avaliação das Propostas

Ferramentas e modelos de simulação são usados para avaliar os trabalhos propostos. Um modelo é uma representação de um sistema real ou planejado [347]. Simuladores são usados para estudar o comportamento do sistema e entender os fatores que afetam seu

desempenho à medida que ele evolui ao longo do tempo [348]. As estruturas de simulação fornecem soluções nos casos em que as técnicas de modelagem matemática são difíceis ou impossíveis de aplicar devido à escala, complexidade e heterogeneidade de um sistema de *fog computing* [347]. A simulação é uma forma de imitar o funcionamento de sistemas reais, com a liberdade de modificar as entradas e modelar uma série de características, analisar sistemas existentes ou apoiar o projeto de novos [349].

Alguns simuladores, que já eram utilizados para validar estudos em *cloud computing*, foram adaptados para *fog computing*. Além disso, novos simuladores foram projetados especificamente para atender a demanda de *fog computing*. Uma análise detalhada de vários simuladores para *fog computing* foi apresentada em [348]. Assim, esta seção visa responder a QP5, que aborda a forma como as abordagens propostas foram avaliadas. Uma análise dos simuladores que foram utilizados nas publicações selecionadas é apresentada na Tabela 4.7.

Tabela 4.7: Simuladores utilizados.

Ferramenta	**Artigo**	**Descrição**	**%**
CloudSim	[132, 133, 244, 264, 291, 309, 310, 314, 316, 317, 319, 325, 334]	Simulador criado inicialmente para simular serviços de *cloud computing* [350].	12.0%
iFogSim	[56, 242, 245, 247, 252, 273, 292–295, 329, 336]	Uma extensão do CloudSim especificamente para *fog computing* [233].	11.1%
GridSim	[243]	Permite a modelagem e a simulação de modelos de aplicação para computação em grade [351].	0.9%
Numerical (Matlab)	[239, 246, 248, 250, 254–257, 260, 265–267, 269, 272, 276–281, 283, 284, 288, 290, 298–300, 302–304, 306–308, 318, 320–322, 326, 328, 328, 330–332, 335, 337]	Usado para estudar o comportamento de sistemas cujos modelos matemáticos são muito complexos para fornecer soluções analíticas.	41.7%
Específico	[102, 134, 237, 240, 241, 249, 251, 258, 261–263, 268, 270, 271, 274, 285–287, 289, 296, 297, 305, 311–313, 315, 324, 327, 333]	Ambiente de hardware e conjuntos de dados sintéticos projetados especificamente para a execução de testes.	26.9%

A maioria dos trabalhos analisados utilizou simuladores numéricos para validar suas propostas. Este tipo de simulação é usado para estudar o comportamento de sistemas cujos modelos matemáticos são muito complexos para fornecer soluções analíticas, como em muitos sistemas não lineares. Essa é a situação encontrada em muitas propostas que abordaram a alocação de recursos em *fog computing*. O simulador mais comum foi o Matlab [352], utilizado por 45 do total de 108 publicações, ou seja, cerca de 42%.

O simulador CloudSim [350] foi proposto para simular serviços de *cloud computing*. É uma biblioteca para simulação de *cloud computing* desenvolvida em linguagem Java, na qual cada entidade é representada como uma classe. Portanto, a maioria dos trabalhos que utilizou o CloudSim, apresentou uma solução com integração com ambientes em *cloud computing*, justificando a utilização deste simulador. Uma extensão do simulador CloudSim é o iFogSim [233]. Este simulador permite modelar ambientes IoT e *fog* para medir o impacto das técnicas propostas para gerenciamento de recursos em termos de latência, congestionamento de rede, consumo de energia e custo. Considerando que só foi apresentado em 2017 [233], e que o tempo necessário para seu amadurecimento e maior utilização, este simulador passou a ser utilizado mais recentemente por acadêmicos.

De forma menos representativa, alguns estudos analisados utilizaram outros simuladores para avaliar suas propostas. O simulador GridSim [351], que permite a modelagem e a simulação de modelos de aplicação para computação em grade, foi utilizado no artigo [243]. Por fim, cerca de 27% das publicações foram avaliadas em ambientes de teste construídos especificamente para validar a proposta do artigo. Nesse tipo de teste, todo o software e o hardware são configurados de forma autônoma, utilizando conjuntos de dados sintéticos.

4.2.2.6 Domínios da *Fog Computing*

Dos 108 artigos analisados, apenas 20 deles (18%) indicaram um domínio específico ao qual suas propostas se aplicam. Assim, para endereçar a QP6, esses domínios serão detalhados na Tabela 4.8.

Tabela 4.8: Domínios da *fog computing*.

Domínio	**Artigos**	**Descrição**	**%**
Saúde	[262, 302]	Sistemas médicos ciber-físicos permitem interação contínua e inteligente entre elementos computacionais e dispositivos médicos que precisam de abordagens de baixa latência.	1.9%
Construções inteligentes	[206, 239, 249, 251, 269, 272, 294, 297]	Como o *fog computing* está mais próximo dos dispositivos IoT, ela é amplamente usada em cidades inteligentes, edifícios e projetos industriais [19], processando dados e fornecendo resposta rápida.	7.4%
Veículos	[240, 246, 285, 298–300, 323, 324]	Os veículos devem ser capazes de calcular, armazenar e comunicar com outros veículos ou dispositivos, melhorar a segurança, a conveniência e a satisfação da condução [353].	7.4%
Realidade Virtual	[253, 301]	É uma tecnologia de interface avançada entre um usuário e o computador, criando um ambiente mais próximo da realidade da pessoa com efeitos visuais, sonoros e táteis.	1.9%

Nos últimos anos, tem havido crescente atenção aos sistemas que suportam o desenvolvimento de redes veiculares. Isso ocorre porque os veículos estão cada vez mais equipados com poderosos computadores de bordo, unidades de armazenamento de dados de grande capacidade, e módulos de comunicação mais avançados para melhorar a segurança, conveniência e satisfação ao dirigir [353]. Estes veículos devem ser capazes de calcular, armazenar e comunicar com outros veículos ou dispositivos. Os recursos e benefícios da *fog computing* são totalmente aderentes a esse tipo de serviço.

Uma tendência na área da saúde é a utilização de MCPS, sigla para o termo *Medical Cyber-Physical Systems*, que permitem uma interação contínua e inteligente entre elementos computacionais e dispositivos médicos (por exemplo, monitores de frequência cardíaca) [262]. No entanto, considerando a complexidade e a alta qualidade dos serviços necessários, esses dispositivos precisam de baixa latência e outros critérios para comunicação com a plataforma de *cloud computing*. Portanto, a *fog computing* é uma abordagem promissora para o uso desses recursos.

Como o *fog computing* está mais próxima dos dispositivos IoT, ele é amplamente usado em cidades inteligentes, edifícios e projetos industriais [19]. Uma construção inteligente é aquele que responde às exigências dos ocupantes, das organizações e da sociedade. Ele também precisa ser sustentável (consumo de energia e água), saudável (bem-estar das pessoas que vivem e trabalham nele), e funcional (necessidades do usuário) [354]. Um caso de uso de construção inteligente foi utilizado nos artigos [246, 251, 272, 294] para abordar a proposta de alocação de recursos. Assim como a construção inteligente, a fabricação inteligente também é um caso de uso que pode aproveitar os benefícios da *fog computing*.

Por fim, a Realidade Virtual foi utilizada em dois artigos analisados para explicar o uso da alocação de recursos. À medida que a necessidade desses aplicativos que exigem baixa latência aumenta, espera-se que novos casos de uso para *fog computing* apareçam nos próximos anos.

4.2.3 Análise dos Artigos sobre Alocação de Recursos

Ao analisar as publicações apresentadas nesta seção sobre a alocação de recursos para a *fog computing*, é possível notar que um conceito relevante em um ambiente da *fog computing* e na alocação de recursos é o *fog node*, conforme apresentado na Seção 3.1. Além de capacidades computacionais, tais como CPU, armazenamento, memória e outras capacidades computacionais [9], eles também possuem algumas características comportamentais, como disponibilidade e confiabilidade [32]. Desta forma, para obter um gerenciamento adequado deste recurso na *fog computing*, é necessário considerar ambos.

Todavia, embora as características comportamentais sejam desejáveis de serem considerados, sua degradação não é considerada como um impedimento para que a aplicação seja executada. O mesmo não ocorre com as capacidades computacionais, uma vez que eles impactam diretamente a performance da aplicação e, portanto, o atendimento dos requisitos mínimos exigidos pelo demandante para cada atributo deve ser atendido na íntegra. No entanto, esta diferenciação foi encontrada apenas em três artigos [244,255,355], sendo isto uma oportunidade para propor melhorias.

Além disso, para realizar o gerenciamento de recursos em *fog computing*, duas perspectivas devem ser consideradas: a perspectiva do usuário final e a perspectiva do provedor de serviços. Atender a essas duas necessidades é um desafio porque ambos, naturalmente, têm interesses divergentes. Por um lado, os usuários finais sempre desejam obter os melhores recursos disponíveis, aqueles com maior poder computacional, por exemplo, com o menor custo. Por outro lado, os provedores de serviços estão interessados em entregar o conjunto mínimo de recursos para evitar custos desnecessários, ou situações que envolvam indisponibilidade de recursos para outros usuários. Entre os 108 trabalhos analisados, nenhum deles considerou estas duas perspectivas, gerando uma outra oportunidade de melhoria para ser considerada na proposta apresentada nesta tese.

Baseado neste contexto, a decisão de alocar o melhor *fog node* para uma determinada execução de aplicação pode ser considerada como um problema de decisão do tipo MCDM [356]. De acordo com Zavadskas *et al.* [357], "os problemas de decisões multicritérios referem-se a tomar decisões na presença de múltiplos e, geralmente, conflitantes, critérios". Esta abordagem foi utilizada em algumas propostas avaliadas na Seção 4.2.2.2, porém elas não são abrangentes como a solução apresentada nesta tese para alocação de recursos, que será apresentada no Capítulo 6, uma vez que elas não consideram as capacidades computacionais e as características comportamentais, bem como as perspectivas do usuário final e do provedor de serviços em uma única solução.

4.3 Considerações Finais

Este capítulo apresentou os resultados de duas revisões sistemáticas da literatura que foram realizadas sobre os tópicos de descoberta e de alocação de recursos computacionais em *fog computing*. Os resultados obtidos nas revisões realizadas e apresentados neste capítulo serviram para o desenvolvimento das propostas de descoberta e de alocação que serão apresentados nos capítulos seguintes.

Capítulo 5

Proposta para Descoberta de Recursos em *Fog Computing*

Este capítulo apresenta uma abordagem eficiente para descoberta de recursos que considera as capacidades computacionais e as características comportamentais no processo de descoberta, capaz de selecionar e registrar no Catálogo de Recursos apenas dispositivos que atendem a todos os critérios de entrada definidos pelo usuário e/ou provedor. Na Seção 5.1 são apresentados o modelo do sistema, as definições do problema e a proposta. A descrição dos ambientes utilizados, dos testes realizados e dos resultados obtidos é apresentada na Seção 5.2. Por fim, a Seção 5.3 traz uma comparação da nossa proposta com os trabalhos similares existentes na literatura e também com a Taxonomia sobre descoberta de recursos apresentada no Capítulo 2.

5.1 Descoberta de Recursos Baseada em Capacidades Computacionais e Características Comportamentais

Na descoberta de recursos em um ambiente de *fog computing*, o objetivo central é identificar e mapear os recursos disponíveis no ambiente de forma eficiente e dinâmica. Esta etapa é essencial para garantir que as aplicações possam acessar os recursos necessários de forma oportuna e otimizada. A Tabela 5.1 mostra todas as notações utilizadas nesta seção, que apresenta uma proposta para descoberta de recursos em *fog computing*.

Tabela 5.1: Notação utilizada na proposta.

Notação	Parâmetro
$\mathcal{F}$	Conjunto de *fog nodes*
f	*Fog node*
F	Número de *fog nodes*
i	Índice do *fog node* em $\mathcal{F}$
$\mathcal{C}$	Capacidade Computacional
$\mathcal{B}$	Comportamento
c_{ij}	Capacidade do *fog node* i em relação ao atributo j
b_{ij}	Comportamento do *fog node* i em relação ao atributo j
C	Número de Atributos de Capacidade
B	Número de Atributos de Comportamento
A	Matriz "A"
a	elemento "a" da Matriz "A"
$\mathcal{V}$	Vetor de Valores Solicitados
n	Elementos
$\mathcal{O}$	Catálogo de Recursos

5.1.1 Definição do Problema

Em um ambiente de *fog computing*, há um conjunto de *fog nodes* f, onde cada $f_i \in \mathcal{F} = \{f_1, f_2, ..., f_F\}$. Os atributos de capacidades computacionais ($\mathcal{C}$) são aqueles que representam requisitos computacionais mínimos para execução de tarefas, como o número de CPUs, a quantidade de memória, disco, etc. Eles são representados pelo vetor $\mathcal{C} = \{c_1, c_2, ..., c_C\}$, onde C é o cardinal do conjunto. Em contraste, características comportamentais ($\mathcal{B}$) podem impactar significativamente a experiência do usuário e o desempenho da aplicação, mas não impedem a execução de tarefas. Segurança, mobilidade, escalabilidade e confiabilidade são exemplos de características comportamentais. Eles são representados por $\mathcal{B} = \{b_1, b_2, ..., b_B\}$, onde B é o cardinal do conjunto. Levando em consideração ambos os tipos de atributos (computacionais e comportamentais), cada *fog node* é determinado como uma tupla $f_i = \{\mathcal{C}_i \cup \mathcal{B}_i\}$; ou seja, $f_i = \{c_{i1}, ..c_{iC}, b_{i1}, ..., b_{iB}\} \mid i \in \{1, 2, ..., F\}$.

Com isso, considerando que cada coluna representa um atributo (a, seja computacional ou comportamental), o Catálogo de Recursos ($\mathcal{O}$) pode ser denotado por uma matriz de atributos (A), onde $\{a_{in} \in A \mid 0 < i \leq F; 0 < n \leq C + B\}$. Portanto, o Catálogo de Recursos pode ser representado como:

$$A = \begin{vmatrix} c_{11} & c_{12} & ... & c_{1C} & b_{11} & b_{12} & ... & b_{1B} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ c_{F1} & c_{F2} & ... & c_{FC} & b_{F1} & b_{F2} & ... & b_{FB} \end{vmatrix}$$

O objetivo da descoberta de recursos na computação em *fog computing* é encontrar novos *fog nodes* disponíveis no ambiente e registrar suas capacidades computacionais ($\mathcal{C}$) e características de comportamento ($\mathcal{B}$) no Catálogo de Recursos ($\mathcal{O}$).

Assim, para selecionar e registrar efetivamente no Catálogo de Recursos apenas *fog nodes* que atendem aos critérios computacionais e comportamentais, um vetor $\mathcal{V} =$

$\{v_1, v_2, ..., v_{C+B}\}$ traz os valores mínimos necessários para cada um dos atributos. Portanto, a função objetivo pode ser definida conforme apresentada na Equação 5.1, para cada $f_i \in \mathcal{F}$:

$$\mathcal{O}_i = \sum_{n=1}^{C+B}(a_{in} \geq v_n) \tag{5.1}$$

Desta forma, para cada *fog node* encontrado, a variável $\mathcal{O}$ aumenta com cada critério atendido (um atributo igual ou maior que seu valor mínimo definido), e o registro no Catálogo de Recursos só ocorrerá se os atributos respeitarem os limites definidos $\mathcal{V}$. Portanto, as restrições para esse problema são que todos os recursos computacionais necessários ($\mathcal{C}$) e características comportamentais ($\mathcal{B}$) devem ser fornecidos pelo *fog node* que está sendo avaliado após a descoberta. O valor total de $\mathcal{O}$ para cada *fog node* deve ser a soma do número de atributos indicados pela Equação 5.2.

$$\mathcal{O}_i = C + B \tag{5.2}$$

5.1.2 Proposta

Considerando as características essenciais da *fog computing* e também a análise do processo de descoberta de recursos (Capítulos 2 e 4), esta seção apresenta uma nova proposta de descoberta que leva à consideração de capacidades computacionais e características comportamentais ao selecionar os *fog nodes*. Neste contexto, os seguintes parâmetros foram utilizados:

- **CPU:** o número de núcleos de CPU disponíveis para uso exclusivo deve ser igual ou maior do que o número de CPUs solicitadas pelo usuário;
- **CPU *Clock*:** frequência na qual o *clock* de um processador pode gerar pulsos, que são usados para sincronizar as operações de seus componentes. É importante definir a velocidade do processador, e ela é medida em Megahertz (Mhz);
- **Memória:** o requisito mínimo de RAM (em GB) disponível para uso exclusivo no *fog node* é a quantidade de RAM solicitada pelo usuário;
- **Armazenamento:** a quantidade de armazenamento livre (em GB) disponível para uso exclusivo no *fog node* deve ser no mínimo a quantidade de armazenamento livre solicitada pelo usuário;

- **Latência:** o atraso (em milissegundos) para que uma requisição seja transferida deve ser igual ou menor do que a quantidade de atraso solicitada pelo usuário;
- **Número de Saltos:** define a localidade e estima a distância entre o *fog node* e o servidor, facilitando a localização, por exemplo, dos *fog nodes* mais próximos;
- **Disponibilidade:** considerando o comportamento histórico, essa característica avalia o quanto do recurso está disponível para alocação. Usando o AWS SLA[1] como ponto de comparação, quando a disponibilidade do *fog node* é maior do que 99%, ele tem o valor 5. Para disponibilidade menor que 95%, o valor 1 é atribuído;
- **Escalabilidade:** a capacidade do *fog node* de lidar com uma carga de trabalho crescente, considerando tanto o comportamento histórico quanto a quantidade de recursos livres. O valor 5 é atribuído aos recursos capazes de expandir seu poder de computação em até 10 vezes. O valor 4 é atribuído àqueles com 8 vezes, e sucessivamente até o valor 1;
- **Confiabilidade:** esse requisito avalia o nível em que o *fog node* é confiável para concluir a execução da tarefa, considerando o comportamento histórico. O valor 1 é atribuído aos *fog nodes* que sempre abortam uma execução. Por outro lado, o valor 5 é atribuído aos *fog nodes* que, em pelo menos as últimas 10 vezes, não abortaram uma execução;
- **Mobilidade:** quão móvel é o *fog node*. É uma métrica proporcionalmente inversa, na qual quanto mais estático for um *fog node*, maior será o valor de mobilidade. No artigo [132], é apresentada uma escala para classificar a mobilidade do dispositivo, atribuindo valores diferentes a dispositivos móveis grandes (tais como *tablets* e *laptops*), dispositivos móveis pequenos (por exemplo, *smartphones*) e também dispositivos estáticos (por exemplo, *desktops*). Esta escala foi adaptada para determinar os valores entre 1 a 5 a serem designados neste atributo.

Os parâmetros de entrada são o intervalo de IPs a serem pesquisados, os critérios computacionais e os critérios comportamentais. O processo é iniciado em um servidor centralizado pela busca dos *hosts* na rede indicada na entrada usando o protocolo NMAP [358]. Esse protocolo foi escolhido para essa tarefa porque ele já pode fornecer informações relevantes para outras decisões de algoritmo, como latência e o endereço MAC do *fog node* encontrado. Então, ele é comparado com os registros no Catálogo de Recursos por meio do MAC Address. Se for um *fog node* desconhecido, o processo continua coletando informações sobre os parâmetros computacionais. Para obter essas informações, o *host* é acessado via protocolo SSH.

[1] https://aws.amazon.com/compute/sla/

```
[{"hostnames": [{"name": "", "type": ""}], "addresses":
{"ipv4": "192.168.2.100", "mac": "08:00:27:48:27:80"},
"vendor": {"08:00:27:48:27:80": "Oracle VirtualBox virtual
NIC"}, "status": {"state": "up", "reason": "arp-response"},
"memory": "5900", "storage": "2944464", "cpu": "2",
"scalability": 5, "reliability": 5, "availability": 5},
{"hostnames": [{"name": "", "type": ""}], "addresses":
{"ipv4": "192.168.2.102", "mac": "08:00:27:73:08:94"},
"vendor": {"08:00:27:73:08:94": "Oracle VirtualBox virtual
NIC"}, "status": {"state": "up", "reason": "arp-response"},
"memory": "6437", "storage": "2098628", "cpu": "2",
"scalability": 5, "reliability": 5, "availability": 5},
{"hostnames": [{"name": "", "type": ""}], "addresses":
{"ipv4": "192.168.2.103", "mac": "08:00:27:14:1E:C4"},
"vendor": {"08:00:27:14:1E:C4": "Oracle VirtualBox virtual
NIC"}, "status": {"state": "up", "reason": "arp-response"},
"memory": "6219", "storage": "2097776", "cpu": "2",
"scalability": 5, "reliability": 5, "availability": 5},
{"hostnames": [{"name": "", "type": ""}], "addresses":
{"ipv4": "192.168.2.105", "mac": "08:00:27:0B:3B:49"},
"vendor": {"08:00:27:0B:3B:49": "Oracle VirtualBox virtual
NIC"}, "status": {"state": "up", "reason": "arp-response"},
"memory": "6329", "storage": "2098268", "cpu": "2",
"scalability": 5, "reliability": 5, "availability": 5}]
```


Figura 5.1: Extrato do Catálogo de Recursos.

Uma vez que os valores das capacidades computacionais e características comportamentais foram obtidos, a validação é realizada. O *fog node* somente é registrado no Catálogo de Recursos quando for capaz de atender a todos os critérios. E uma vez registrado no Catálogo de Recursos, o *fog node* estará disponível para ser usado por outros processos de gerenciamento de recursos, tais como alocação e monitoramento. Assim sendo, para esta proposta, o Catálogo de Recursos é um arquivo no formato JSON, pois entende-se que esse tipo de arquivo favorece a troca de mensagens e é comumente utilizado em diferentes aplicações. A Figura 5.1 mostra um extrato de um arquivo de Catálogo de Recursos. A partir deste recorte do Catálogo de Recursos é possível observar que os campos "*memory*", "*storage*" e "*cpu*" correspondem aos atributos c_1, c_2 e c_3, respectivamente. Os campos "*scalability*", "*reliability*" e "*availability*" representam os atributos b_1, b_2 e b_3.

Um tempo de recorrência é fornecido para reiniciar o processo para garantir a dinamicidade necessária em um ambiente de *fog*. No entanto, também é possível alterar seu método de *bootstrapping* de proativo para reativo com apenas um pequeno ajuste no algoritmo que suporta a proposta. Como resultado, a varredura por novos *hosts* ocorreria apenas quando solicitada, permitindo que a sobrecarga de consumo de recursos computacionais e de rede do servidor de descoberta fosse ainda mais reduzida.

Uma das novidades desta proposta é que ela permite filtrar os *fog nodes* de interesse por critérios computacionais e comportamentais. Se um *fog node* não atender aos critérios mínimos fornecidos logo após ser descoberto, ele será ignorado e não registrado no Catálogo de Recursos. Outro ponto que vale destacar sobre esta proposta é que ela não

requer a instalação prévia de um agente no *fog node*. Isso é importante porque os recursos computacionais de um ambiente de *fog computing* tendem a ter baixa capacidade, e a instalação de um agente pode ter um impacto negativo. Por fim, este algoritmo tem uma complexidade $O(n)$, ou seja, é uma complexidade linear baseada na quantidade de *fog nodes* encontrados e inseridos no Catálogo de Recursos.

5.2 Avaliação

Um ambiente de experimento real foi configurado para dar suporte à avaliação da proposta, simulando os parâmetros e comportamentos esperados para um cenário de *fog computing*. Este ambiente consistia em diferentes tipos de dispositivos IoT: quatro unidades Raspberry Pi, cada uma com 4 GB de RAM, 2 GB de tamanho de disco e 4 CPUs; e quatro máquinas virtuais com 8 GB de RAM, 4 GB de tamanho de disco e 2 vCPUs. Todos eles têm o Ubuntu 22.04 LTS como sistema operacional. O servidor era um dispositivo quad-core de 16 GB de RAM. Além disso, com o objetivo de simular um ambiente real de *fog computing*, vários outros dispositivos foram adicionados ao ambiente, tais como *Smart TVs*, *smartphones*, *WiFi mesh*, assistentes virtuais, etc. No total, o ambiente de experimento real consistiu em 32 dispositivos. Os dispositivos IoT e o servidor estavam no mesmo intervalo de rede, com base em uma conexão sem fio. O número de saltos entre o servidor e os *fog nodes* foi o mesmo para todos os experimentos realizados para garantir uma execução justa e a proximidade e baixa latência esperadas de um ambiente de *fog computing*. Assim, nesta tese esse ambiente de teste será chamado de "Real Lab".

Para o escopo deste trabalho, o algoritmo proposto foi implementado em Python e está disponível no gitbub[2]. Nesta versão, ele pode trazer informações de *fog nodes* que utilizam o sistema operacional Linux e, para isso, foi necessário um processo de *bootstrap* prévio em cada *host*, fornecendo as credenciais de segurança necessárias para executar os comandos Linux pela aplicação de descoberta, melhorando a segurança e a privacidade. Desta forma, o sistema operacional também é considerado um critério da capacidade computacional a ser atendida. Ainda, com o mesmo objetivo de aumentar a segurança, a porta padrão do serviço SSH foi alterada. Como todos os comandos são considerados nativos, a instalação de pacotes adicionais do *host* foi desnecessária. No lado do servidor, foi necessário instalar o pacote NMAP[3].

[2]https://github.com/leonardoreboucas/simple-network-discover/tree/main
[3]https://NMAP.org/download#linux-rpm

5.2.1 Testes e Resultados

Esta seção apresenta os experimentos e os resultados obtidos pelo algoritmo proposto para descoberta de recursos em um ambiente real de *fog computing*. Os testes realizados objetivaram validar a capacidade do algoritmo de encontrar os *fog nodes* no ambiente e aplicar as restrições de inclusão no Catálogo de Recursos, bem como validar sua performance referente ao tempo de execução e ao consumo de rede, que são fatores relevantes em um ambiente de *fog computing*.

5.2.1.1 Teste 1 - Completude da Descoberta

O primeiro experimento envolveu a capacidade do algoritmo de encontrar todos os *fog nodes* disponíveis e elegíveis no ambiente. Para este experimento, os critérios computacionais e comportamentais foram configurados abaixo das capacidades existentes dos dispositivos do *testbed* para permitir a descoberta, e o sistema operacional necessário foi definido como Linux. Todos os *fog nodes* foram ligados e o algoritmo de descoberta foi iniciado. O algoritmo conseguiu descobrir todos os *fog nodes* que atendiam aos requisitos e trazer todos os dados descobertos para o Catálogo de Recursos.

É importante notar que depois que todos os *fog nodes* foram encontrados na primeira execução, as próximas execuções não trouxeram novos resultados até que outros novos *fog nodes* estivessem disponíveis na rede. Outra possibilidade é que um ou mais *fog node* seja removido do Catálogo de Recursos (por exemplo, pelo processo de monitoramento devido a *timeouts* em mensagens de *heartbeat*) ou quando fica indisponível e, algum tempo depois, se torna acessível novamente. Isso mostra que o algoritmo pode comparar os *fog nodes* encontrados com aqueles já registrados no Catálogo de Recursos e evitar duplicação de registros, inconsistência de dados e esforço computacional para o registro desnecessário de um *fog node* que já estava registrado. Para garantir isso, o endereço MAC do *fog node* é considerado a chave primária.

Este experimento também foi realizado iniciando o algoritmo de descoberta sem nenhum *fog node* conectado e ir conectando-os um a um, ainda com os critérios computacionais e comportamentais inferiores aos existentes nos *hosts*. Para esta etapa, o tempo de recorrência do algoritmo foi definido como 1 minuto. Após a execução do experimento, foi possível confirmar que os *fog nodes* são descobertos na rede assim que são conectados, conforme apresentado na Figura 5.2.

5.2.1.2 Teste 2 - Restrições de Capacidade Computacional

Para continuar os testes, o critério *memória* foi ajustado para 3000 MB, filtrando a descoberta apenas para *fog nodes* com memória livre maior do que esse número. A Tabela

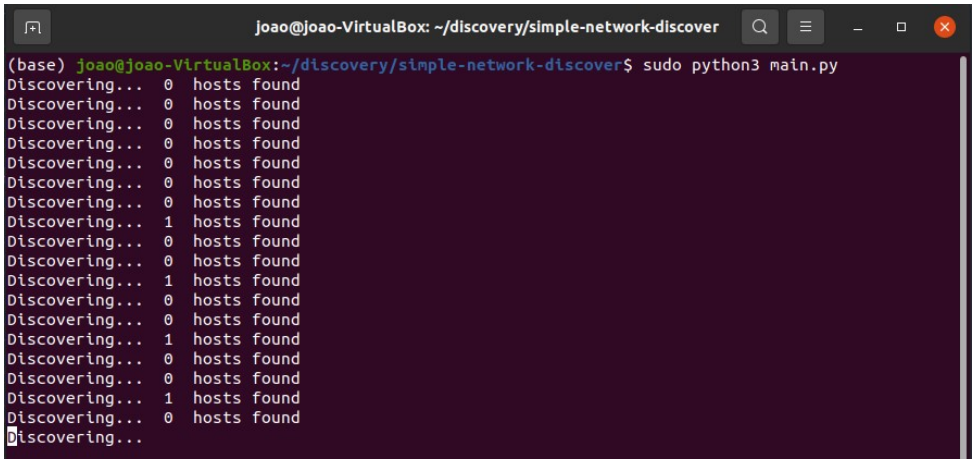

Figura 5.2: Efetividade da pesquisa com intervalos.

Tabela 5.2: Valores de memória livre.

Fog Node	**FN01**	**FN02**	**FN03**	**FN04**	**FN05**	**FN06**	**FN07**	**FN08**
Memória Livre (MB)	3569	3123	2681	2983	6852	6230	5877	6541

5.2 apresenta o valor de memória livre para cada um dos *fog nodes* que são Raspberry Pis e máquinas virtuais no momento dos experimentos.

Na primeira execução, apenas dois *fog nodes* não atenderiam aos critérios de seleção, e apenas os *fog nodes* FN03 e FN04 não foram encontrados. Em uma segunda execução, o critério foi ajustado para 3500 MB e, novamente, o algoritmo foi assertivo, resultando na exclusão do FN02.

Essa funcionalidade do algoritmo também é inovadora, e é considerada importante para garantir a busca por *fog nodes* que sejam viáveis para o caso de uso, evitando registros desnecessários no Catálogo de Recursos, e otimizando o tempo de registro no processo de descoberta e busca para outras etapas do gerenciamento de recursos que usarão o Catálogo de Recursos.

5.2.1.3 Teste 3 - Tempo de Execução

Outro experimento foi realizado para mensurar o tempo de execução referente ao número de *fog nodes* a serem descobertos. Entretanto, com a execução dos experimentos, percebeu-se que um fator que impacta no tempo de execução do algoritmo é o intervalo de IPs de rede que está sendo pesquisado. O intervalo de rede é um parâmetro definido como entrada para o algoritmo, conforme indicado na Seção 6.2. Ao considerar o intervalo 192.168.2.0/28 como entrada, por exemplo, foram verificados 13 IPs, de 192.168.2.2 a 192.168.2.14, pois foram utilizados 3 IPs para rede (192.168.2.0), *gateway* (192.168.2.1)

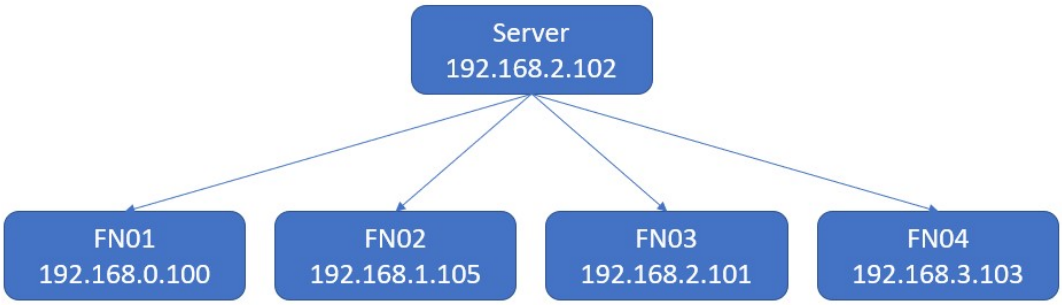

Figura 5.3: *Fog nodes* em sub-redes diferentes.

e *broadcast* (192.168.2.15). Com isso, no intervalo 192.168.2.0/28, serão verificados 1021 IPs.

Entretanto, o crescimento não é linear com o número de IPs verificados. Enquanto um intervalo /28 é verificado em 3 segundos, um /22 precisa de 38 segundos. Este é um ponto a ser observado, pois um ambiente de *fog computing* deve ter muitas redes diferentes e, consequentemente, um grande intervalo de IPs a serem verificados. Uma maneira de resolver essa situação foi executar o algoritmo em paralelo, limitando o intervalo de IPs a um tempo razoável para a solução.

Para isso, os IPs de alguns dispositivos foram ajustados para estarem em sub-redes diferentes, conforme mostrado na Figura 5.3. O *script* que contém as chamadas paralelizadas para o algoritmo teve um tempo médio de execução de 22 segundos e, no final, foi capaz de encontrar todos os *fog nodes* existentes. Para este experimento, o parâmetro de intervalo de rede foi definido como /24 em quatro tarefas, cada uma pesquisando em uma sub-rede. Assim, supondo que essa ação de execuções de paralelização em intervalos de rede limitados não tivesse sido feita. O tempo médio necessário para descobrir os *fog nodes* em quatro sub-redes diferentes teria sido de 41 segundos, quase o dobro do tempo. Nesse caso, um intervalo /22 no modelo sem alterações no algoritmo teria levado quase o dobro do tempo.

Para simular um ambiente de *fog computing* com alta heterogeneidade de recursos de forma ainda mais precisa, vários dispositivos não elegíveis ou que não escolheram participar do ambiente de *fog computing* foram registrados na mesma rede WiFi e na mesma sub-rede. Esses dispositivos incluiram celulares pessoais, roteadores, *smart* TVs, etc. Nesse cenário, foi possível identificar que após o mapeamento do *NMAP* encontrar algum dispositivo, foi feita uma tentativa de conexão a ele para extrair suas informações computacionais usando o comando Linux *ssh*. Esse processo requer tempo até que o *host* de destino negue a conexão, causando uma execução mais lenta em intervalos de rede com muitos dispositivos não elegíveis como *fog nodes*. Assim sendo, o processo de busca do comando NMAP foi melhorado para retornar apenas dispositivos com uma porta

Tabela 5.3: Tempo de execução no ambiente de testes.

Intervalo IP	**Total Dispositivos**	**Possíveis *Fog Nodes***	**Tempo Antigo (segundos)**	**Novo Tempo (segundos)**
192.168.0.1-64	29	8	1032	43
192.168.0.65-128	3	0	364	12
192.168.0.129-192	0	0	3	3
192.168.0.193-254	0	0	3	3

específica aberta para resolver essa situação. Essa porta é configurada no *fog node* durante o processo de *bootstrapping*. Portanto, a lista de *hosts* encontrados é limitada a potenciais candidatos pertencentes ao ambiente de *fog computing*.

A Tabela 5.3 apresenta o tempo de execução do algoritmo antes e depois da implementação da melhoria. Também são mostrados o número de dispositivos disponíveis no intervalo de IP e quantos eram elegíveis para pertencer ao ambiente de testes. Analisando a Tabela 5.3 é possível observar uma redução significativa no tempo de execução do algoritmo nos dois primeiros intervalos de IP, nos quais alguns dispositivos não eram *fog nodes* elegíveis para descoberta. Para os dois últimos intervalos de IP, não houve alteração no tempo de execução, pois nenhum dispositivo estava disponível. Com isso, foi possível identificar que a alteração feita no parâmetro de busca NMAP foi suficiente para melhorar o desempenho do tempo de execução do algoritmo.

A redução no tempo de execução permitiu que a solução proposta nesta tese tivesse um desempenho mais próximo da proposta apresentada por Jin e Kim [207]. No entanto, embora nosso tempo de descoberta por dispositivo tenha sido superior ao trabalho comparado, é importante destacar que o escopo do trabalho é diferente e que esta proposta analisa as capacidades computacionais e características comportamentais que não foram realizadas por eles.

Considerando a diferença de escopo entre o trabalho proposto e o artigo [215], também foi possível comparar os tempos de descoberta, que no artigo referido leva aproximadamente de 6 a 17 segundos quando executado sem conteinerização. Em contraste, na solução proposta, considerando o ambiente de experimento apresentado, leva cerca de 4 segundos para encontrar cada dispositivo. Isso é, pelo menos, uma melhoria de 33% considerando seu melhor caso (6 segundos).

5.2.1.4 Teste 4 - Consumo de Rede

Por fim, foi considerado o consumo de rede do algoritmo de descoberta. Esta é uma preocupação relevante em um ambiente de *fog computing* potencialmente conectado por conexões instáveis causadas por múltiplas tecnologias e mobilidade de dispositivos. Para essa medida, foi utilizada a execução do comando *ifstat -t* no servidor, que mostra os

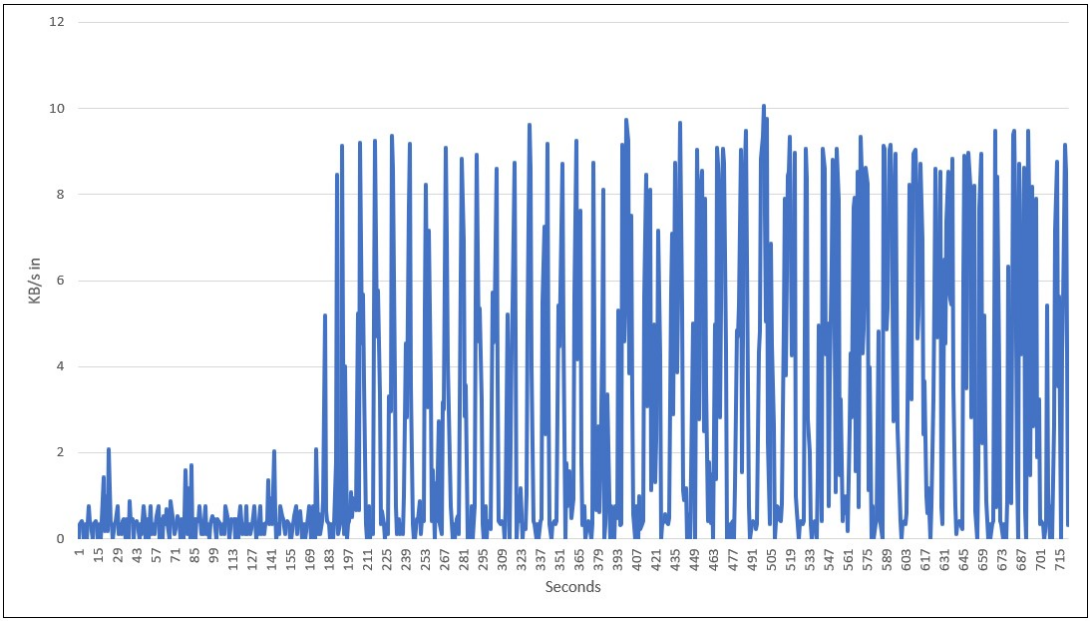

Figura 5.4: Consumo de rede (KB/s) para pacotes de entrada.

pacotes de entrada e de saída ao longo do tempo[4]. Os resultados apresentados representam o consumo de rede ao executar o algoritmo de descoberta em que um *fog node* foi adicionado a cada 1 minuto. A Figura 5.4 mostra os valores de entrada dos pacotes (KB/s in), enquanto a Figura 5.5 mostra os valores de saída dos pacotes por segundo (KB/s out) do servidor. O processo foi iniciado 30 segundos após o início da coleta de dados pelo comando.

Os fluxos de entrada e de saída têm comportamento semelhante. O consumo de rede aumenta quando o primeiro *fog node* começa a reportar na rede, permanecendo constante mesmo com novos *fog nodes* adicionados ao longo do tempo. Assim, o consumo de rede não é afetado pelo número de *fog nodes* descobertos e disponíveis. Ao longo da janela de coleta de 716 segundos, o consumo máximo de entrada foi de 11,36 KB/s, com uma média de 2,55 KB/s de consumo. Em relação à saída, o pico foi de 7,91 KB/s, e a média foi de 1,94 KB/s, mostrando que o algoritmo proposto demanda pouca saída de pacotes de rede no lado do servidor.

5.3 Análise da Proposta

A proposta para a descoberta de recursos em *fog computing* foi comparada com trabalhos similares existentes na literatura, e também foi classificada com a taxonomia para descoberta de recursos apresentada no Capítulo 4, conforme descrito a seguir.

[4]https://linux.die.net/man/1/ifstat

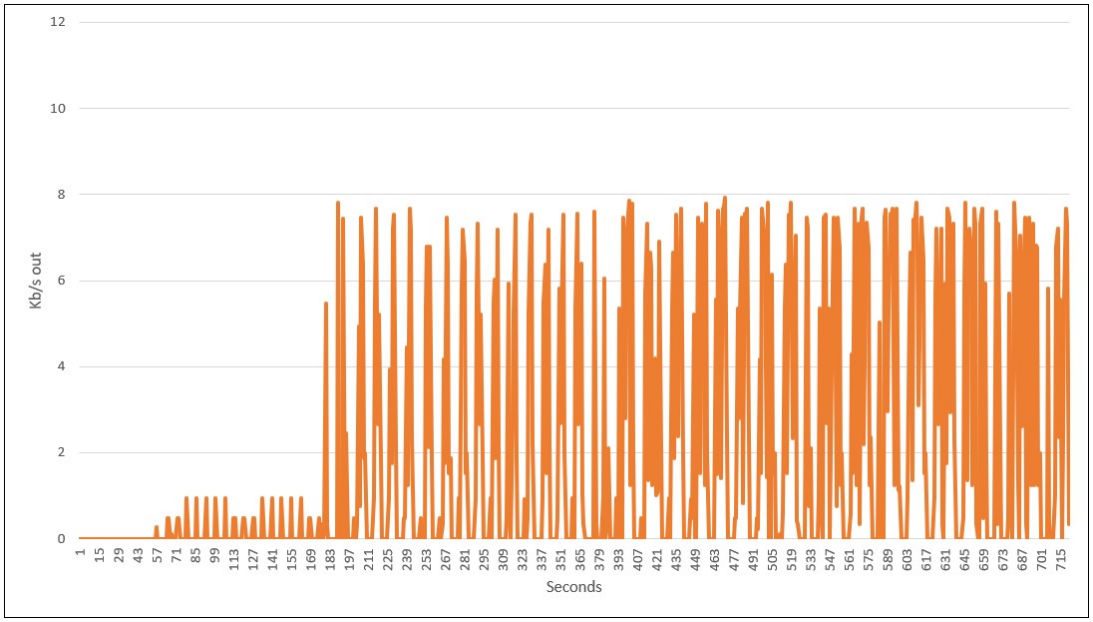

Figura 5.5: Consumo de rede (KB/s) para pacotes de saída.

5.3.1 Comparação com Trabalhos Relacionados

Esta seção apresenta os trabalhos relacionados mais relevantes na descoberta de recursos para ambientes de *fog computing*. Eles estão indicados na Tabela 5.4 e foram selecionados da Seção 4.1.3 por estarem mais próximos do escopo de descoberta proposto e, desta forma, possibilitarem a comparação entre eles e a proposta apresentada neste capítulo. Os trabalhos foram comparados em relação ao paradigma computacional utilizado, o tratamento de capacidades computacionais e de características comportamentais, a existência de um Catálogo de Recursos, a forma de realização dos testes em ambientes reais ou simulados, a quantidade de dispositivos utilizados nos testes e, por fim, se houve comparação da proposta com outras existentes na literatura.

No trabalho de Jin *et al.* [207], uma linguagem de modelagem de API RESTful é usada para articular recursos, e um protocolo de modelo de especificação *Open Communication Foundation* (OCF) é usado para facilitar a comunicação entre *fog nodes*. Além disso, os autores utilizam um Catálogo de Recursos no servidor para armazenar informações, que podem ser posteriormente consultadas durante processos subsequentes do gerenciamento de recursos.

No artigo de Sattari *et al.* [211], é sugerida uma solução distribuída de descoberta de recursos para plataformas de *fog computing*, aproveitando o diretório de recursos de ambientes RESTful. Os autores incorporam microsserviços implantados em contêineres e empregam o protocolo CoAP para permitir a alocação sob demanda, adotando uma arquitetura híbrida que combina recursos de arquiteturas centralizadas e distribuídas. As

Tabela 5.4: Análise dos trabalhos relacionados sobre descoberta de recursos.

Artigo	Ano	Ambiente	Computacionais	Comportamentais	Catálogo	Testes	Dispositivos	Comparação
[207]	2018	IoT	Sim	Não	Sim	Real	2	Não
[211]	2020	Mist	Sim	Não	Não	Real	2	Não
[212]	2021	Edge	Sim	Não	Não	Simulado	–	Não
[215]	2021	Edge	Sim	Não	Não	Real	7	Não
[216]	2022	Edge	Sim	Não	Não	Simulado	–	Sim
Nossa proposta	2024	Fog	Sim	Sim	Sim	Real	32	Sim

avaliações incluem cenários simulados e do mundo real, revelando uma redução na latência de consulta.

Kondo *et al.* [212] explorou a aplicação do protocolo *Named Data Networking* (NDN), destacando sua adequação para descoberta de recursos em IoT e ambientes de *fog computing*. Em contraste com outras propostas que podem direcionar tarefas para recursos já ocupados por outras atividades, os autores escolheram o NDN para direcionar tarefas exclusivamente para nós capazes de executá-las de forma eficaz. O próprio protocolo fornece informações sobre o recurso, eliminando a necessidade de sobrecarregar a arquitetura para obter esses dados e, consequentemente, reduzindo o tempo necessário para localizar o *fog node* mais adequado.

Baseado em um caso de uso específico da área da saúde, os autores do artigo [215] empregam um algoritmo heurístico para alcançar resultados locais ideais na descoberta de recursos, considerando a latência como um requisito crucial de qualidade de serviço. A descrição do recurso é delineada por meio de arquivos no formato JSON. A arquitetura é distribuída entre as camadas de *Cloud*, *Fog* e IoT, e a proposta é baseada em uma busca reativa, com distribuição de um agente pelos *fog nodes*. A descrição dos recursos é realizada por meio de WSDL, juntamente com um algoritmo meta-heurístico que procura *fog nodes* tendo como critérios o tempo de resposta e o custo [216]. Essa abordagem é baseada principalmente na resolução de descoberta para dispositivos IoT e não especificamente para *fog nodes*.

A Tabela 5.4 resume os trabalhos relacionados analisados nesta seção. É possível identificar que nenhum dos trabalhos apresentados forneceu soluções de descoberta de recursos especificamente para *fog computing*, embora *Edge* e *Mist computing* também sejam paradigmas distribuídos. Além disso, é possível identificar uma deficiência na realização de experimentos em ambientes reais. Os que avaliaram seus trabalhos em *testbeds* do

mundo real [207, 211, 215] usaram um número de dispositivo muito inferior aos usados neste trabalho. Também é possível observar a ausência de propostas considerando capacidades computacionais e características comportamentais, e registrar o resultado em um Catálogo de Recursos. Outro destaque deste estudo é que ele é um dos poucos estudos que comparam os resultados obtidos com outro trabalho disponível na literatura, mesmo considerando as diferenças de escopo.

5.3.2 Classificação em Relação à Taxonomia Proposta

No Capítulo 4 foi apresentada uma taxonomia para o processo de descoberta de recursos em *fog computing*. Dessa forma, esta seção apresenta o enquadramento da solução proposta em relação à essa taxonomia. A Tabela 5.5 traz um resumo desta análise.

Tabela 5.5: Classificação da solução proposta em relação à taxonomia apresentada.

Item Taxonomia	**Valor**
Algoritmo	Determinístico
Inicialização	Proativa ou Reativa
Camadas	IoT - *Fog*
Propagação	Sem Agente
Protocolo	TCP e UDP
QoS	Sim
Catálogo	Interno
Descrição	Semântica
Busca	Direcionada
Topologia	Centralizada

Portanto, ao comparar a avaliação da solução proposta em relação à taxonomia apresentada, e também com os trabalhos considerados na Seção 4.1.3, é possível notar que nenhum dos trabalhos apresenta o conjunto de atributos igual proposto nesta tese. Além disso, a proposta de descoberta de recursos em *fog computing* também busca superar alguns desafios indicados na Seção 4.1.4, tais como a dinamicidade, o poder computacional limitado característico do ambiente de *fog computing* e as limitações de conexão.

5.4 Considerações Finais

Este capítulo apresentou uma proposta para a descoberta de recursos que está aderente ao ambiente de *fog computing*, e que tem como diferencial considerar as capacidades computacionais e as características comportamentais no processo de descoberta. Além disso, a solução sugerida é capaz de selecionar e registrar no Catálogo de Recursos apenas dispositivos que atendem a todos os critérios de entrada definidos pelo usuário. Ademais,

também foram apresentados os testes realizados e os resultados obtidos, demonstrando a escalabilidade da solução.

Um artigo contendo a proposta de descoberta de recursos, apresentada neste capítulo, foi apresentado no *10th International Congress on Information and Communication Technology* que foi realizado em Londres, em fevereiro de 2025 (Anexo IX, reproduzido com a permissão da *Springer Nature*).

Capítulo 6

Proposta para Alocação de Recursos em *Fog Computing*

Este capítulo apresenta uma proposta para a alocação de recursos em *fog computing*, baseado no modelo de decisão multicritério MCDM. A Seção 6.1 apresenta o modelo do sistema, as definições do problema, as restrições e a função objetivo. A Seção 6.2 traz o algoritmo proposto para alocação de recursos. A solução aplicada para tratar requisições múltiplas é apresentada na Seção 6.3. Um caso de uso para demonstrar a aplicação da proposta é apresentado na Seção 6.4. Por fim, a Seção 6.5 apresenta os testes realizados e os resultados obtidos.

6.1 Modelo para Alocação de Recursos

Esta seção apresenta as definições das variáveis adotadas para o método de alocação de recursos em *fog computing* proposto neste trabalho. A Tabela 6.1 reúne todas as notações usadas nesta seção. Em um ambiente de *fog computing*, há um conjunto de *fog nodes*, onde cada $f_i \in \mathcal{F} = \{f_1, f_2, ..., f_F\}$. As capacidades computacionais ($\mathcal{C}$) representam requisitos computacionais mínimos para execução de tarefas, como CPU, memória, disco, etc. Eles são representados pelo vetor $\mathcal{C} = \{c_1, c_2, ..., c_C\}$, onde C é o cardinal definido. Em contraste, as características comportamentais ($\mathcal{B}$) são características que o usuário deseja, mas não é obrigado a operar. Não atender às características comportamentais não impede que o aplicativo seja executado, mas pode reduzir significativamente a experiência do usuário. Segurança, mobilidade, escalabilidade e confiabilidade são exemplos de características comportamentais. Elas são representadas por $\mathcal{B} = \{b_1, b_2, ..., b_B\}$, onde B é o cardinal do conjunto.

Levando em consideração ambos os tipos de atributos, cada *fog node* é determinado como uma tupla $f_i = \{\mathcal{C}_i \cup \mathcal{B}_i\}$; ou seja, $f_i = \{c_{i1}, ..., c_{iC}, b_{i1}, ..., b_{iB}\} \mid i \in \{1, 2, ..., F\}$.

Tabela 6.1: Notação utilizada nesta seção.

Notação	Parâmetro
$\mathcal{F}$	Conjunto de *fog nodes*
f	*Fog node*
F	Número total *fog nodes*
m	Um *fog node* em F
$\mathcal{C}$	Vetor capacidade
$\mathcal{B}$	Vetor comportamento
c_{ij}	Capacidade do *fog node* i em relação ao atributo j
b_{ij}	Comportamento do *fog node* i em relação ao atributo j
C	Número de Atributos de capacidade
B	Número de Atributos de comportamento
N	Cardinal dos total de atributos
A	Matriz A
a	Elemento a na Matriz A
$\mathcal{V}$	Vetor de valores da requisição
$\mathcal{W}$	Vetor de pesos da requisição
$\mathcal{R}$	Vetor da requisição
Υ	Fator de normalização
P	Valor normalizado
Ω	Soma dos valores normalizados dos atributos de capacidade
Ψ	Soma dos valores normalizados dos atributos de comportamento
ζ	Soma dos valores normalizados absolutos dos atributos de comportamento
α	Fator de ponderação
w	Peso
v	Valor
n	Elementos
$\mathcal{Q}$	Custo do *fog node*
q	Atributo de custo
Q	Cardinal dos atributos de custo
$price$	Máximo preço a ser pago
$Cost$	Matriz de Custo
r	Quantidade de *fog nodes* pedidos

O número total de atributos (capacidades e características comportamentais) de cada *fog node* é representado por $N = C + B$. Considerando que cada coluna representa um atributo, uma matriz de *fog nodes* pode ser denotada por $\{a_{mn} \in A \mid 1 \leq m < F; 1 \leq n < N\}$. Portanto, a Matriz A pode ser representada como:

$$A = \begin{vmatrix} c_{11} & c_{12} & ... & c_{1C} & b_{11} & b_{12} & ... & b_{1B} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ c_{F1} & c_{F2} & ... & c_{FC} & b_{F1} & b_{F2} & ... & b_{FB} \end{vmatrix}$$

Além disso, a Matriz A pode ser contraída para:

$$A = \begin{vmatrix} \mathcal{C}_1 & \cup & \mathcal{B}_1 \\ \mathcal{C}_2 & \cup & \mathcal{B}_2 \\ \vdots & \vdots & \vdots \\ \mathcal{C}_F & \cup & \mathcal{B}_F \end{vmatrix}$$

E, finalmente, para:

$$A = \left| \begin{array}{c} f_1 \\ f_2 \\ \vdots \\ f_F \end{array} \right|$$

Como cada atributo em ($\mathcal{C}$) e em ($\mathcal{B}$) pode ter unidades e escalas diferentes, é necessário normalizar a Matriz A para permitir comparações e operações entre seus elementos. Primeiro, usando uma técnica de normalização vetorial adaptada [359], o valor do fator (Υ) para cada atributo é calculado pela Equação 6.1:

$$\Upsilon_n = \sqrt{\left(\sum_{m=1}^{F} (a_{mn})^2 \right)} \tag{6.1}$$

Além disso, cada *fog node* deve ter um custo financeiro atribuído ($\mathcal{Q}$). Para isso, outra matriz é construída:

$$Cost = \left| \begin{array}{c} q_{f_1} \\ q_{f_2} \\ \vdots \\ q_{f_F} \end{array} \right|$$

Do lado do solicitante, os valores ($\mathcal{V}$) são usados para indicar a quantidade necessária de recursos para cada atributo (por exemplo, o valor "8" é necessário para vCPUs), e os pesos ($\mathcal{W}$) são usados para representar a importância que o usuário definiu para cada atributo em forma de percentil, como "15%" para memória e "30%" para armazenamento. Um *Mean Opinion Score* (MOS) [360] de 1 a 5 indica o valor das características comportamentais, no qual 1 representa a menor importância e 5 é a maior importância. Ao solicitar um recurso, o usuário informa os valores desejados ($\mathcal{V}$) e os pesos ($\mathcal{W}$) das capacidades computacionais e características comportamentais, bem como o custo máximo que ele ou ela está disposto a pagar pela alocação de recursos (*price*). A quantidade de *fog nodes* necessária também é informada, representada por r. Consequentemente, a requisição do usuário é definida como $\mathcal{R} = \mathcal{V} \cup \mathcal{W} \cup price \cup r$ onde $\mathcal{V} = \{v_1, v_2, ..., v_N\}$, $\mathcal{W} = \{w_1, w_2, ..., w_N\}$, *price* traz o custo máximo aceitável do recurso e r indica o número de *fog nodes* necessários. Semelhante ao que ocorre na *cloud computing*, para esta proposta, foi considerado que uma vez alocado o recurso, ele permanece disponível para o solicitante até que este o libere. Por fim, como propriedade dos dados, é essencial garantir que a soma de todos os pesos informados seja igual a 100%, então $\sum \mathcal{W} = 1$.

Levando em consideração os valores ($\mathcal{V}$) e os pesos ($\mathcal{W}$) informados pelo solicitante, e

utilizando o fator de normalização Υ apresentado na Equação 6.1, é executada a normalização de cada atributo da Matriz A, gerando um valor P, conforme calculado na Equação 6.2.

$$P_{mn} = \left(\frac{a_{mn} - v_n}{\Upsilon n \quad w_n}\right) \tag{6.2}$$

A soma dos valores normalizados obtidos com a Equação 6.2 para $\mathcal{C}$ cria uma variável chamada Ω, apresentada na Equação 6.3:

$$\Omega_m = \sum_{n=1}^{C} P_{mn} \tag{6.3}$$

A soma dos valores normalizados obtidos com a Equação 6.2 para $\mathcal{B}$ cria uma variável chamada Ψ, obtida com a Equação 6.4:

$$\Psi_m = \sum_{n=C+1}^{N} P_{mn} \tag{6.4}$$

Como os valores normalizados Υ de $\mathcal{B}$ podem ser negativos, para estimar a distância ao valor informado pelo usuário, a soma do módulo dos valores normalizados obtidos com a Equação 6.2 para $\mathcal{B}$ cria uma variável chamada ζ, apresentada na Equação 6.5:

$$\zeta_m = \sum_{n=C+1}^{N} |P_{mn}| \tag{6.5}$$

Neste ponto, duas perspectivas podem ser desenvolvidas. A primeira, denominada *Solução USR*, considera a perspectiva do usuário, na qual se busca o melhor *fog node*, ou seja, aquele que possui os valores máximos de cada atributo e que possui o menor custo, garantindo que o custo seja menor do que o custo informado pelo usuário. A perspectiva do provedor é a segunda, denominada *Solução PRV*. Ou seja, o *fog node* selecionado é aquele que, ao atender a solicitação do usuário, possui os valores mínimos disponíveis para seus atributos e, além disso, possui o maior custo, porém obedecendo o limite informado pelo usuário. O provedor visa atingir o maior lucro possível, atendendo a todos os requisitos do usuário.

6.1.1 Definição do Problema

A alocação de recursos é uma etapa do gerenciamento de recursos que busca os melhores recursos computacionais disponíveis necessários para executar uma aplicação no ambiente de *fog computing*, visando atender a Qualidade de Serviço (QoS) [361]. Neste trabalho, o recurso é um *fog node* (f), e é composto por capacidades computacionais ($\mathcal{C}$) e características comportamentais ($\mathcal{B}$), além de custo financeiro ($\mathcal{Q}$). Quando um usuário faz uma requisição, espera-se que, pelo menos, um *fog node* (f) atenda aos requisitos mínimos de capacidade computacional ($v_1..v_C$), além de ser mais barato que o limite de custo. É aceitável que o *fog node* (f) disponível seja o mais próximo possível de atender a todas as características comportamentais necessárias ($v_{C+1}..v_N$), mesmo que não as atenda totalmente. Portanto, considerou-se que atender aos requisitos de capacidades computacionais ($v_1..v_C$) e respeitar o limite de custo informado ($price$) são essenciais para rodar uma aplicação. Entretanto, atender aos requisitos de características comportamentais ($v_{C+1}..v_N$) é apenas desejável.

Com isso, o objetivo é encontrar, dentre os *fog nodes* disponíveis na Matriz A, aqueles que atendem a todos os requisitos de capacidades, ao limite de custo e, dentro desse subconjunto, o *fog node* que mais se aproxima do atendimento aos requisitos de características comportamentais, considerando os pesos informados para cada atributo. Portanto, as restrições para esse problema são que todos os atributos de capacidades ($\mathcal{C}$) devem atender aos requisitos do usuário, conforme indicado pela Equação 6.6. Para modelar essa seleção, x_m é uma variável binária que obterá o valor 1 quando o *fog node* m for selecionado e o valor 0, caso contrário.

$$\begin{gathered} x_m \leftarrow f_{mn} \geq \mathcal{V}_n \\ 1 \leq n \leq C;\ 1 \leq m \leq F \end{gathered} \tag{6.6}$$

Quando um atributo de capacidade deve ser menor do que o valor solicitado pelo usuário (por exemplo, a latência é considerada melhor quando está no menor valor), o inverso da Equação 6.6 é executado.

Além disso, o custo financeiro ($\mathcal{Q}$) também deve ser igual ou menor do que o solicitado pelo usuário, conforme indicado na Equação 6.7:

$$\begin{gathered} q_{f_m} \leq price \\ 1 \leq m \leq F \end{gathered} \tag{6.7}$$

Outra restrição é que o número de *fog nodes* selecionados não pode ser maior do que a quantidade solicitada, conforme indicado pela Equação 6.8:

$$\sum_{m=1}^{F} x_m = r \tag{6.8}$$

$$x_m = \{0, m\} \mid \forall m \in F.$$

6.1.2 Função Objetivo

Conforme mencionado, uma das diferenças desta proposta é que ela possui duas soluções distintas para as perspectivas do usuário e do provedor, denominadas *USR Solution* e *PRV Solution*, respectivamente. Considerando que o usuário sempre pretende obter o melhor recurso disponível e que tenha a melhor relação custo-benefício, a *USR Solution* é definida conforme indicado na Equação 6.9:

$$min \sum_{m=1}^{F} \left(\frac{(\Omega_m + \Psi_m) x_m}{q_m}\right) \tag{6.9}$$

$$\text{s.t. } (6.6), (6.8)$$

Do lado do provedor, o objetivo é encontrar o *fog node* que entrega todos os valores solicitados, mas considera um conjunto mínimo de recursos, evitando desperdício. Assim, a *PRV Solution* é definida como mostrado na Equação 6.10:

$$max \sum_{m=1}^{F} \left(\frac{(\Omega_m + \zeta_m) x_m}{q_m}\right) \tag{6.10}$$

$$\text{s.t. } (6.6), (6.8)$$

Na Equação 6.9, que representa a perspectiva do usuário, o recurso a ser selecionado será aquele com a menor relação custo-benefício, cujos atributos de capacidades atendam a todos os requisitos do usuário, e cujas características comportamentais sejam as mais adequadas possíveis, com valores máximos para cada um, dados os pesos informados pelo usuário. Por outro lado, na Equação 6.10, que representa a perspectiva do provedor, o recurso a ser selecionado será aquele com a maior relação custo-benefício, cujos atributos de capacidades atendam a todos os requisitos do usuário e também cujas características comportamentais sejam as mais adequadas possíveis, dados os pesos atribuídos pelo usuário, com valores mínimos para cada atributo. Da mesma forma, a escolha mais justa é

garantida, ou seja, o recurso com a menor distância da requisição do usuário entre todos os *fog nodes* disponíveis.

6.2 Proposta

Ao solicitar um *fog node* f em um ambiente de *fog computing*, o usuário informa os valores ($\mathcal{V}$) que precisa para capacidades computacionais ($\mathcal{C}$) e características comportamentais ($\mathcal{B}$), bem como o valor máximo que está disposto a pagar pelo recurso (*price*). O número desejado de *fog nodes* (r) também é informado como entrada. Neste ponto, pesos ($\mathcal{W}$) devem ser atribuídos a cada atributo. A função de peso direciona precisamente a escolha para o atributo mais relevante da perspectiva do usuário.

Nesta proposta, a perspectiva do usuário visa selecionar o melhor recurso disponível com o maior poder computacional e menor custo. Da perspectiva do provedor, o recurso a ser entregue deve ser o mais próximo possível dos valores solicitados, com valores mínimos para cada atributo entre os recursos disponíveis e um custo maior.

Portanto, é proposta uma solução baseada no método de Tomada de Decisão por Múltiplos Critérios (MCDM), conforme indicado no Algoritmo 1, que escolhe o melhor *fog node* entre todos os disponíveis que atendem aos requisitos do usuário, considerando as perspectivas do usuário final e do provedor de serviços.

O algoritmo recebe os parâmetros $\mathcal{C}$, $\mathcal{B}$, Υ A e $Cost$, $\mathcal{V}$, $\mathcal{W}$, *price* e r como entradas. Como o vetor do fator de normalização depende unicamente dos valores de atributo da Matriz A, ele é pré-calculado (e atualizado quando necessário) antes de qualquer solicitação. Inicialmente, o algoritmo define o vetor X como TRUE, o que significa que todos os *fog nodes* podem atender à solicitação, mas essa variável binária será definida como FALSE no caso de qualquer atributo não atender à solicitação do usuário (linhas 5-7) ou se o custo do *fog node* for maior que o solicitado (linhas 9-11). Os valores de atributo normalizados P são calculados (linha 4), conforme definido na Equação 6.2. Após isso, os valores de Ω, Ψ e ζ também são calculados (linhas 13-15), permitindo a avaliação da relação custo-benefício. Então, o próximo passo é determinar os melhores *fog nodes* para a *USR Solution* (f_u, na linha 18) e para a *PRV Solution* (f_p, na linha 21) dentro do limite de *fog nodes* solicitados pelo usuário final (r). Por fim, este algoritmo tem uma complexidade $O(n)$, ou seja, a complexidade apresentada é linear, baseada na quantidade de *fog nodes* disponíveis no Catálogo de Recursos.

Algorithm 1: *USR Solution* e *PRV Solution* para a alocação de *fog nodes*

Data: $\mathcal{C}$, $\mathcal{B}$, Υ, Matrix A, Matrix $Cost$, $\mathcal{V}$, $\mathcal{W}$, $price$, r
Result: f_u, f_p

$X \leftarrow TRUE$
for $i \leftarrow 1$ **to** F **do**
  for $j \leftarrow 1$ **to** $(C + B)$ **do**
    $P[i][j] \leftarrow (A[i][j] - \mathcal{V}[j])/(\Upsilon[j] * \mathcal{W}[j])$;
    if $A[i][j] < V[j]$ AND $j \leq C$ **then**
      $X[i] \leftarrow FALSE$
    end
  end
  if $Cost[i] > price$ **then**
    $X[i] \leftarrow FALSE$
  end
end
calculate Ω;
calculate Ψ;
calculate ζ;
for $i \leftarrow 1$ **to** F **do**
  if $X[i]$ AND $f_u \leq r$ **then**
    $f_u \leftarrow min((\Omega[i] + \Psi[i])/Cost[i])$;
  end
  if $X[i]$ AND $f_p \leq r$ **then**
    $f_p \leftarrow max((\Omega[i] + \zeta[i])/Cost[i])$;
  end
end

6.3 Requisições Múltiplas

Um dos desafios de *fog computing* em relação à outros paradigmas computacionais já mais consolidados, tais como a *cloud computing*, é a alta distribuição geográfica que exige uma solução para a alocação de recursos que seja capaz de tratar um grande número de requisições em um pequeno intervalo de tempo, conforme apresentado na pesquisa da literatura (Seção 4.2). Assim, para que a proposta apresentada neste capítulo seja capaz de atender à este requisito, a sua arquitetura foi construída baseada em um serviço de mensageria, conforme apresentada na Figura 6.1.

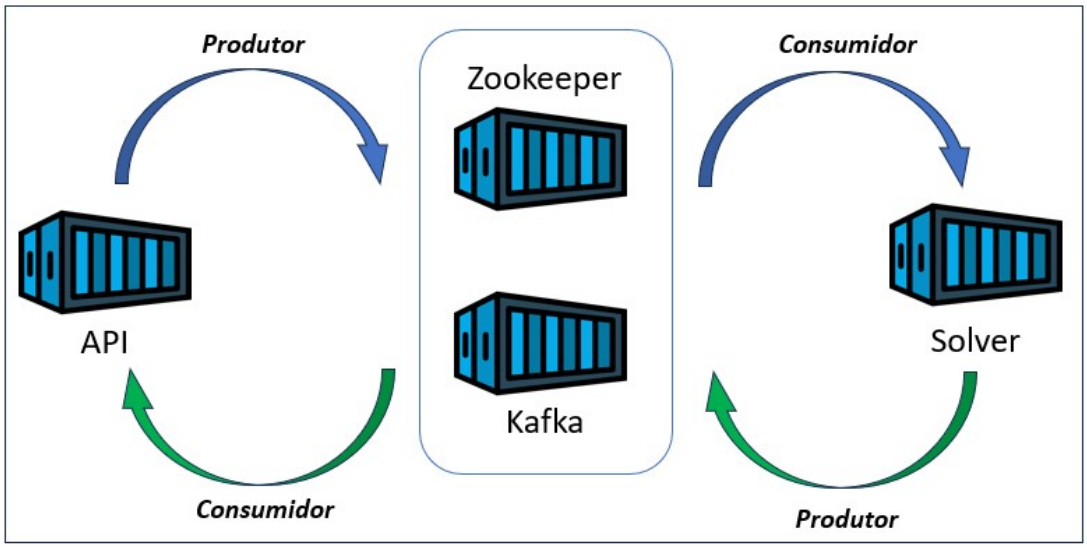

Figura 6.1: Visão geral da arquitetura da solução.

Para suportar esta arquitetura foi utilizado o Apache Kafka [362] que é um sistema de mensageria com código aberto capaz de manipular dados em tempo real, e que também tem baixa latência mesmo para tarefas com alto volume de dados. No Kafka ocorre o envio de dados, também denominados *mensagens* por um Publicador e essa mensagem é consumida por um receptor chamado de Consumidor. Para que seja possível a troca de mensagens entre publicadores e consumidores, elas são organizadas em Tópicos (ou Filas) e são controladas por um componente chamado *Broker*. Para que o funcionamento da proposta nesta arquitetura seja possível, foram criados quatro contêineres cada um com uma função específica, conforme segue:

- API: este componente é baseado em um serviço REST [363] que recebe a requisição do demandante e a publica no *broker* Kafka para ser consumida. Ao mesmo tempo, ela tem a função de consumidor do tópico Kafka da resposta a ser disponibilizada pela solução;

- Kafka: é o *broker* da solução, centralizando as publicações tanto do componente API quanto do componente *Solver*, organizando o consumo das mensagens;
- *Solver*: é neste componente que está o algoritmo apresentado na Seção 6.2. Ele desempenha a função de consumidor ao buscar as mensagens produzidas pelo componente API e, também, é um produtor ao disponibilizar as respostas para as requisições;
- Zookeeper: é responsável por manter informações de configurações e nomenclaturas entre os serviços do kakfa, sincronizando as configurações entre os diversos *clusters*.

Um projeto desenvolvido em Python está no *github*[1] foi criado para disponibilizar esta arquitetura e permitir a reprodução da solução proposta. É importante salientar, no entanto, que o objetivo é apenas criar um ambiente mínimo para que a arquitetura possa ser avaliada, permitindo analisar a viabilidade do uso de um sistema de mensageria para lidar com as múltiplas requisições de um ambiente de *fog computing*. A avaliação da escalabilidade da solução, bem como as avaliações de desempenho serão tratadas em trabalhos futuros. Na página do *github* é possível encontrar as configurações utilizadas em cada um dos componentes, bem como os requisitos necessários para que os testes sejam realizados. Ao realizar esses procedimentos, o ambiente será construído conforme apresentado na Figura 6.2.

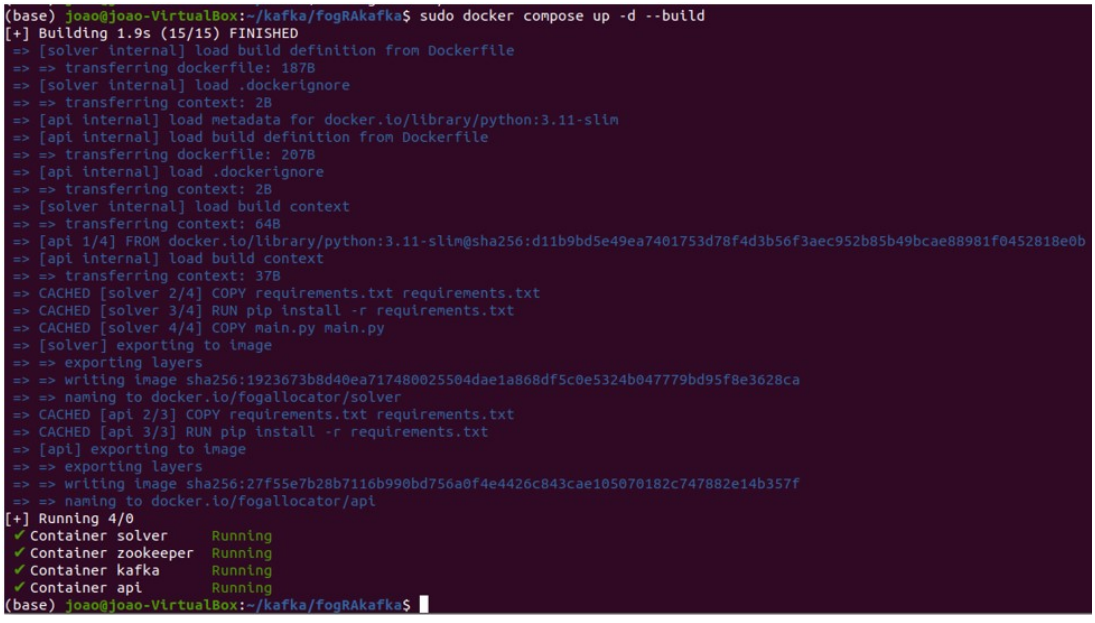

Figura 6.2: Ambiente com quatro contêineres ativados.

Uma vez que o ambiente está em funcionamento, é possível fazer requisições conforme as apresentadas na Figura 6.3, e receber a resposta de quais *fog nodes* foram selecionados.

[1] https://github.com/unb-fog/fogRAkafka

Um mesmo Catálogo de Recursos é utilizado para todas as requisições e, conforme os *fog nodes* são disponibilizados, eles são marcados como "em uso" no Catálogo de Recursos, evitando que sejam alocados em novas requisições. Quando uma requisição não pode ser atendida por não haverem máquinas disponíveis com a quantidade de recursos solicitados, uma mensagem de "Máquinas insuficientes" é apresentada, indicando a necessidade de buscar recursos em outros ambientes, tal como na *cloud computing*, por exemplo. Esta funcionalidade de acionar outros serviços para a obtenção de máquinas não atendidas não está implementada e deverá ser abordada em trabalhos futuros.

Figura 6.3: Exemplo de chamada via *REST Client*.

6.4 Caso de Uso

O paradigma *fog computing* é indicado para suprir casos de uso nos quais a *cloud computing* é insuficiente, como assistência médica, edifícios inteligentes, redes veiculares e processamento de fluxo de dados [364]. Aqui, um cenário envolvendo um ambiente de *fog computing* dando suporte a um aplicativo de casa inteligente é apresentado para fins de demonstração. Nesse caso, o sistema deve selecionar um *fog node* para executar uma tarefa relacionada ao processamento de imagens de circuitos de segurança. Portanto, com base no método SMART [365], os seguintes atributos foram usados:

- **CPU:** o número de núcleos de CPU disponíveis para uso exclusivo deve ser igual ou maior do que o número de CPUs solicitado pelo usuário;
- **Memória:** o requisito mínimo de RAM (em GB) disponível para uso exclusivo no *fog node* é a quantidade de memória RAM solicitada pelo usuário;
- **Armazenamento:** a quantidade de armazenamento livre (em GB) disponível para uso exclusivo no *fog node* deve ser, pelo menos, a quantidade de armazenamento livre solicitada pelo usuário;
- **Latência:** o atraso (em milissegundos) para uma solicitação ser transferida deve ser igual ou menor do que a quantidade de atraso solicitada pelo usuário;

- **Disponibilidade:** considerando o comportamento histórico do quanto o recurso está disponível para alocação. Usando o AWS Service Level Agreement (SLA)[2] como ponto de comparação, quando a disponibilidade do *fog node* é maior do que 99%, ele tem o valor cinco. Para disponibilidade menor do que 95%, o valor um é atribuído;
- **Escalabilidade:** a capacidade do *fog node* de lidar com uma carga de trabalho crescente, considerando tanto o comportamento histórico quanto a quantidade de recursos livres. O valor cinco é atribuído a recursos capazes de expandir seu poder de computação em até 10 vezes. O valor quatro é atribuído àqueles com oito vezes, e sucessivamente até o valor um;
- **Confiabilidade:** o nível em que o *fog node* é confiável para concluir a execução da tarefa, considerando o comportamento histórico. O valor um é atribuído aos *fog nodes* que sempre abortam uma execução. Por outro lado, o valor cinco é atribuído aos *fog nodes* que, em pelo menos as últimas 10 vezes, não abortaram uma execução;
- **Mobilidade:** mensura o quanto móvel é o *fog node*. É uma métrica proporcionalmente inversa, na qual quanto mais estático um *fog node* for, maior será o valor de mobilidade. Em [132], é apresentada uma escala para classificar a mobilidade do dispositivo, atribuindo valores diferentes a dispositivos móveis grandes (como *tablets* e *laptops*), dispositivos móveis pequenos (por exemplo, *smartphones*) e também dispositivos estáticos (por exemplo, *desktops*). Essa escala foi adaptada para determinar os valores entre um e cinco a serem atribuídos nesse atributo.

Considerando que a precificação em *fog computing* ainda é um assunto em discussão no meio acadêmico [366], dois cenários devem ser considerados. No primeiro cenário, o custo do *fog node* é proporcional à quantidade de recursos (CPU e Memória) disponíveis. Ou seja, quanto maior o poder computacional, maior o custo. Outra possibilidade seria um preço único para todos os *fog nodes*, independentemente de sua configuração. Assim sendo, a Tabela 6.2 detalha as necessidades de um potencial solicitante. Portanto, são considerados dados de entrada na proposta apresentada. Neste caso, os pesos são distribuídos igualmente entre os atributos. A Tabela 6.3 apresenta valores hipotéticos com valores proporcionais para todos os *fog nodes* disponíveis neste caso de uso de casa inteligente. Ela é considerada a Matriz A para análise da proposta.

Ao executar o algoritmo com os valores apresentados na Tabela 6.3, para a *USR Solution*, encontrou-se o FN12 com valor de custo-benefício calculado de 0,68. Ao analisar esse *fog node*, é possível notar que ele atende a todos os requisitos solicitados e tem

[2]https://aws.amazon.com/compute/sla/

Tabela 6.2: Valores de entrada.

	Capacidades				**Comportamentos**			
	CPU	**Memória**	**Armazenamento**	**Latência**	**Disponibilidade**	**Escalabilidade**	**Confiabilidade**	**Mobilidade**
Valores	4	16	2000	4000	5	2	1	3
Pesos	12,5%	12,5%	12,5%	12,5%	12,5%	12,5%	12,5%	12,5%
Preço	1,50							

um valor bem abaixo dos demais, favorecendo a tomada de decisão com base no custo-benefício. Da mesma forma, ao escolher a *PRV Solution*, pondera-se o FN18, com valor de custo-benefício de 3,28 e custo de 0,43.

Já se todos os *fog nodes* tiverem o mesmo custo, a eficiência do algoritmo proposto fica ainda mais evidente. Para esse segundo cenário, foi considerado que todos os *fog nodes* têm um único valor de $ 1,00. Todos os outros valores da Tabela 6.3 foram mantidos. Após a execução do algoritmo foi escolhido o *fog node* FN08 para a *USR Solution*, que possui os melhores valores de atributos. Para a *PRV Solution*, o FN14 foi selecionado como o *fog node* com a menor diferença entre o que foi solicitado e o que está sendo entregue em todos os atributos.

Outra diferença na utilização de uma metodologia baseada em critérios de múltipla escolha é a possibilidade de alterar os pesos entre as variáveis, favorecendo a escolha dos *fog nodes* com mais recursos em um dado atributo. Diferentemente do exemplo anterior, no qual os pesos foram distribuídos igualmente, caso o solicitante precise distribuí-los de forma diferente entre os atributos, o algoritmo proposto pode lidar com isso. Para ilustrar esse cenário, a mesma Matriz A será usada, os mesmos valores de atributos do solicitante serão usados e os pesos serão alterados para 47% para CPU, e 1% para memória, armazenamento e latência. As características comportamentais foram mantidas em 12,5% cada. O cenário de todos os *fog nodes* tendo o preço exato será mantido para demonstrar a eficiência do algoritmo. Neste caso, o *fog node* selecionado em *USR Solution* é FN08, o *fog node* com o valor máximo para CPU na Matriz A. Em *PRV Solution*, o *fog node* selecionado é FN07, com cinco CPUs entregues, apenas uma CPU a mais do que o solicitado. FN07 é o *fog node* na Matriz A com o menor valor para o atributo CPU. As diferenças entre as escolhas com a variação de peso são mostradas na Figura 6.4.

Tabela 6.3: Matriz A.

Fog node (f)	**Capacidades** ($\mathcal{C}$)				**Comportamentos** ($\mathcal{B}$)				**Custo** (q)
	CPU	**Memória**	**Armazenamento**	**Latência**	**Disponibilidade**	**Escalabilidade**	**Confiabilidade**	**Mobilidade**	
FN01	32	204	4438	941	4	3	2	3	0,24
FN02	32	331	1251	403	3	3	4	4	0,36
FN03	38	357	2730	114	1	5	1	3	0,29
FN04	36	454	2922	367	4	2	1	1	0,49
FN05	25	249	8270	3257	1	2	3	1	0,27
FN06	42	245	7648	930	5	1	2	4	0,28
FN07	5	233	3858	631	5	2	1	2	0,23
FN08	46	223	8184	903	4	1	1	1	0,27
FN09	10	376	4099	490	1	2	3	4	0,39
FN10	12	268	3737	624	1	3	3	1	0,28
FN11	14	434	7298	1775	4	3	1	2	0,44
FN12	33	58	8408	261	4	2	2	1	0,09
FN13	40	86	5071	3148	5	1	3	4	0,12
FN14	22	172	3406	2604	4	2	2	4	0,19
FN15	42	371	6057	1418	5	2	1	2	0,41
FN16	31	100	2567	689	4	5	2	3	0,13
FN17	48	119	7237	2619	5	2	4	1	0,16
FN18	32	403	5540	3856	4	4	3	3	0,43
FN19	35	225	7043	4725	5	3	3	4	0,25
FN20	20	447	1369	427	2	2	4	5	0,47

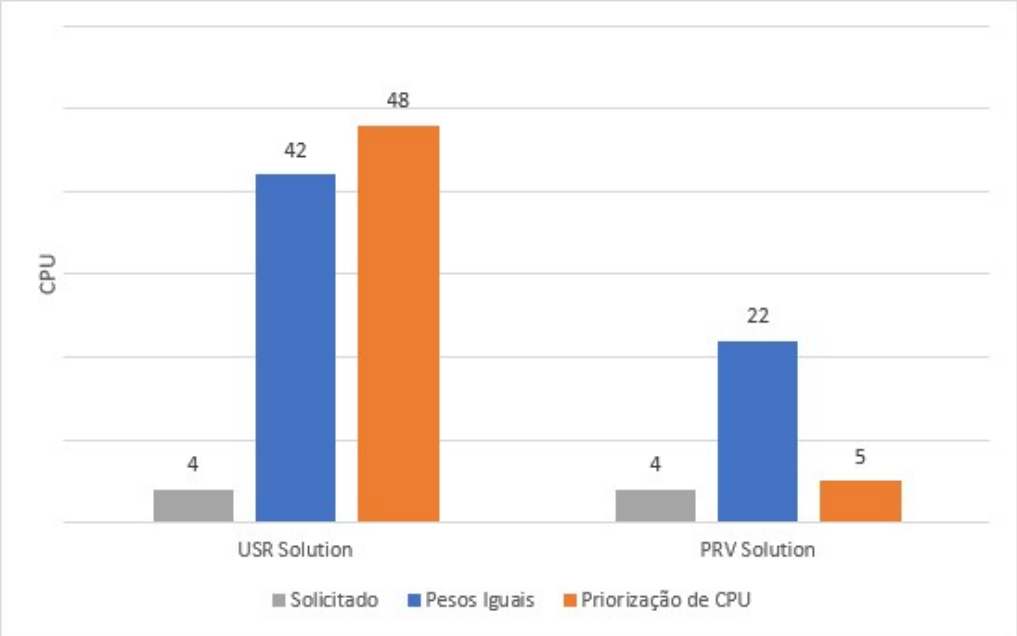

Figura 6.4: Valores de CPU para pesos diferentes.

6.5 Resultados e Discussão

Nesta seção são apresentadas as configuração do ambiente de testes, bem como a avaliação do desempenho da solução proposta em diferentes cenários. Assim, o desempenho desta

Tabela 6.4: Matriz de *fog nodes* do ambiente de teste real.

Fog node (f)	**Capacidades** ($\mathcal{C}$)				**Comportamentos** ($\mathcal{B}$)				**Cost** (q)
	CPU	**Memória**	**Armazenamento**	**Latência**	**Disponibilidade**	**Escalabilidade**	**Confiabilidade**	**Mobilidade**	
FN01	4	4	2048	510	3	4	5	4	0,08
FN02	4	4	2048	563	3	4	5	4	0,08
FN03	4	4	2048	423	3	4	5	4	0,08
FN04	4	4	2048	601	3	4	5	4	0,08
FN05	2	8	1024	300	5	4	5	4	0,10
FN06	4	4	2048	230	5	4	5	4	0,08
FN07	4	8	4096	213	5	4	5	4	0,12
FN08	2	2	2048	360	5	4	5	4	0,04

proposta para alocação de recursos em *fog computing* foi avaliada em um ambiente de testes real e em um outro simulado, conforme descritos a seguir.

6.5.1 Ambiente de Testes Real

Um ambiente de teste real foi construído para dar suporte à avaliação da proposta. Esse ambiente consistiu em diferentes dispositivos IoT, como *Smart TVs*, *smartphones*, *notebooks*, *WiFi mesh*, assistentes virtuais, etc. Além disso, os *fog nodes* foram criados por meio de quatro unidades Raspberry Pi e quatro máquinas virtuais. O servidor foi um Quad-core de 16 GB de RAM. No total, o ambiente de teste real consistiu em 27 dispositivos. Os dispositivos IoT, os *fog nodes* e o servidor estavam no mesmo intervalo de rede com base em uma conexão sem fio. A Tabela 6.4 apresenta as capacidades computacionais e as características comportamentais de cada *fog node* no ambiente. Os custos de cada *fog node* foram calculados em proporção às suas capacidades computacionais.

Para realizar os testes, os *fog nodes* listados na Tabela 6.4 foram registrados em um Catálogo de Recursos, um arquivo externo ao algoritmo. Para permitir que a alocação fosse realizada com a saída do algoritmo, foi criado um *script* no servidor que leu os endereços IPs indicados como selecionados pela aplicação, e fez uma conexão via comando SSH com os *fog nodes* indicados.

Nos testes realizados foram utilizados os valores apresentados na Tabela 6.5. Ao executar os testes com os parâmetros de entrada, considerando que praticamente todos os parâmetros são os mesmos, inclusive o custo, o *fog node* escolhido para a *USR Solution* foi o FN08, o qual atende a todos os requisitos e tem o menor custo. Já para a *PRV Solution*, o *fog node* escolhido foi o FN05, que entrega os requisitos muito próximos do que foi solicitado e também tem um bom custo, o que é mais favorável para o provedor.

Tabela 6.5: Valores de entrada para o ambiente real.

	Capacidades				**Comportamentos**			
	CPU	**Memória**	**Armazenamento**	**Latência**	**Disponibilidade**	**Escalabilidade**	**Confiabilidade**	**Mobilidade**
Valores	2	2	1000	600	5	5	5	5
Pesos	12,5%	12,5%	12,5%	12,5%	12,5%	12,5%	12.5%	12,5%
Preço	0,50							

6.5.2 Ambiente Simulado

Embora a realização de testes em um ambiente real seja essencial para a validação da proposta, as limitações no número de recursos podem influenciar, de certo modo, a análise do desempenho do algoritmo proposto. Assim, outro cenário de teste foi projetado em um ambiente simulado usando a ferramenta *iFogSim* [233], que foi usada para gerar os *fog nodes*. O principal objetivo do teste no ambiente simulado é estimar o desempenho do algoritmo conforme o número de *fog nodes* no Catálogo de Recursos varia. Para isso, foi considerado o tempo necessário para encontrar aquele que deve ser alocado ao solicitante entre todos os *fog nodes* no Catálogo.

Para permitir uma comparação entre a solução proposta com outras na literatura para o problema de alocação de recursos, o mesmo ambiente e variáveis foram submetidos para execução usando outro algoritmo. Para tanto, o TOPSIS [367] foi selecionado, pois tem sido usado em muitos trabalhos relacionados e é usado principalmente para resolver problemas de MCDM na literatura. Esta comparação é apresentada na Figura 6.5.

Ao analisar a Figura 6.5 é possível observar que o algoritmo proposto varia de acordo com o número de *fog nodes* disponíveis no Catálogo de Recursos. No entanto, ele se mostra adequado para *fog computing* porque sempre consegue encontrar uma solução viável entre as opções disponíveis. Isso significa que o algoritmo garantiu encontrar a solução em 100% das requisições.

Assim, embora o desempenho do TOPSIS tenha sido melhor para cenários com poucos *fog nodes* (menos de 3.000 *fog nodes*), a solução proposta se mostrou melhor em relação à consistência temporal e ao crescimento do número de *fog nodes* no ambiente. Isso é importante porque espera-se que um ambiente de *fog computing* seja composto por muitos dispositivos e, portanto, ter desempenho adequado para um volume maior de *fog nodes* é essencial para garantir uma boa execução da aplicação de alocação de recursos. Isso mostra que a solução proposta nesta tese é escalável, pois mantém uma variação muito baixa no tempo para encontrar a solução viável em *fog computing* com até 6.000 *fog nodes*.

Outra comparação essencial com esse outro algoritmo é com relação ao custo-benefício

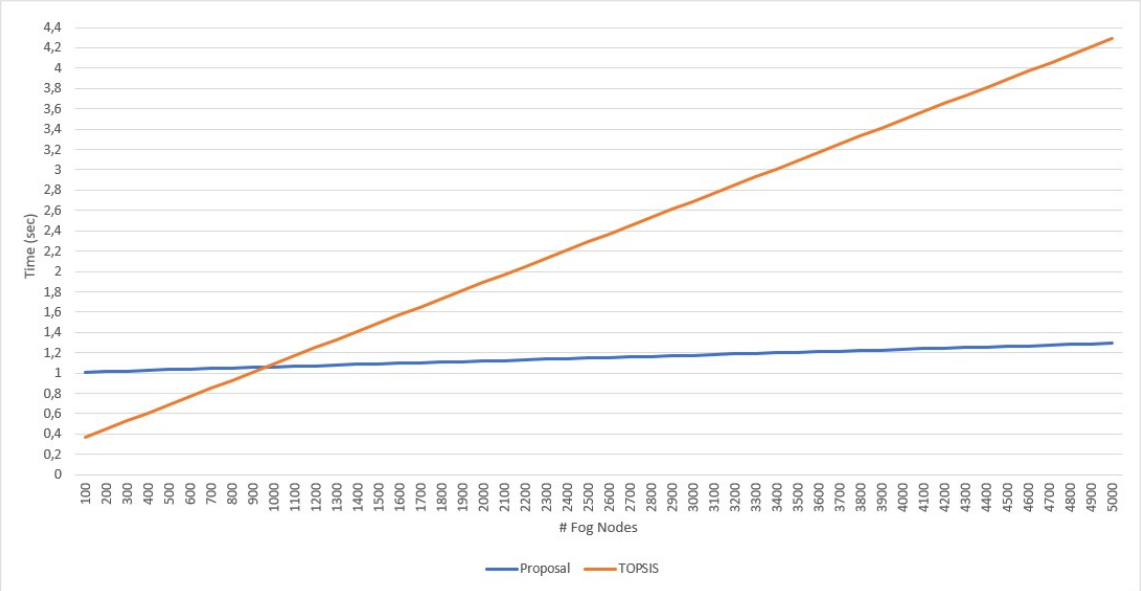

Figura 6.5: Tempo de execução do algoritmo proposto e o TOPSIS [13].

para o provedor e para o usuário. Especificamente, o TOPSIS não utiliza critérios para oferecer duas soluções diferentes, pois o objetivo principal do algoritmo é encontrar apenas a melhor solução, aquela com os maiores valores. Para ilustrar isso, considerou-se que neste ambiente simulado, seria executada uma aplicação por 1, 5, 10 e 60 minutos. Para isso, foi considerado que se tem disponíveis os *fog nodes* apresentados na Tabela 6.3 e as entradas de demanda são aquelas fornecidas na Tabela 6.2. Neste caso, a *USR Solution* selecionou o FN12, que tem um custo de 0,09 por minuto; para a *PRV Solution*, foi escolhido o FN18 com um custo de 0,43; caso contrário, usando o TOPSIS, o *fog node* selecionado foi o FN03, que tem um custo de 0,29 por minuto.

Os resultados das execuções dos algoritmos avaliados são apresentados na Figura 6.6. Note que, embora o TOPSIS tenha um tempo de execução menor para cenários com poucos *fog nodes*, ele não seleciona o *fog node* mais adequado para o usuário ou para o provedor ao considerar a relação custo-benefício. Isso fica ainda mais evidente quando o tempo de execução da aplicação aumenta, mostrando que o TOPSIS sempre se mantém na linha do meio entre o melhor custo-benefício para o usuário e o melhor custo-benefício para o provedor. Dessa forma, fica evidente a eficiência do algoritmo proposto em maximizar os resultados para os provedores ou, ao mesmo tempo, entregar as melhores opções para os usuários, reduzindo os custos de uso dos recursos.

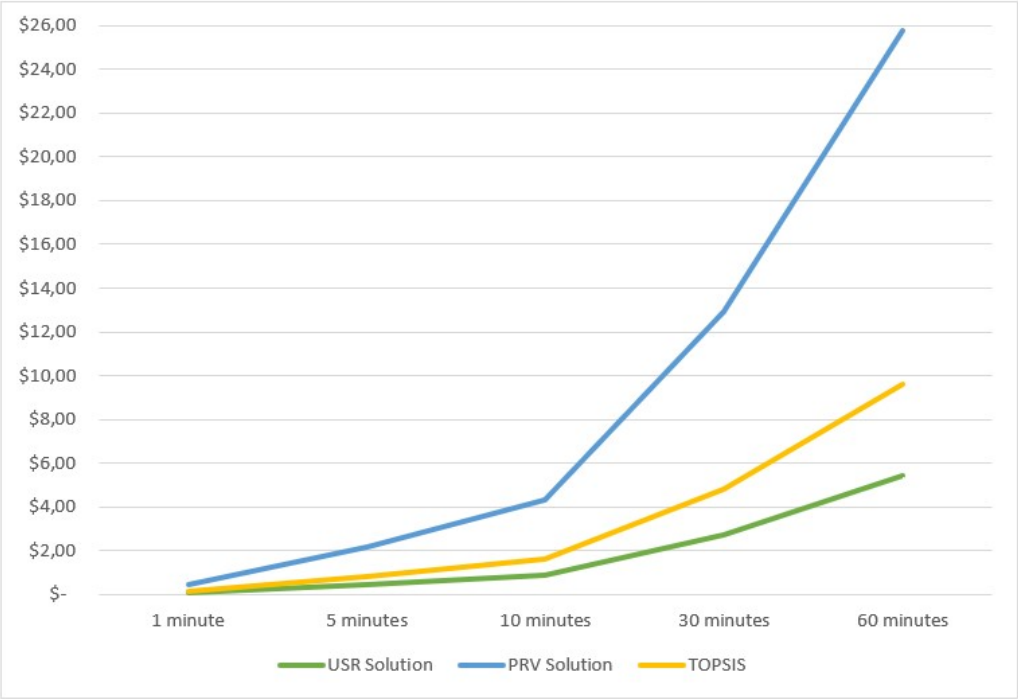

Figura 6.6: Comparação de custos dos algoritmos [13].

6.6 Trabalhos Relacionados

Um resumo dos trabalhos relacionados sobre alocação de recursos para ambientes de *fog computing* com base em capacidades computacionais ou características comportamentais é mostrado na Tabela 6.6. Como não há consenso na literatura sobre o escopo da computação *fog* e *edge* [2], ambos os paradigmas foram considerados.

No artigo [325], uma arquitetura de delegação de serviços de IoT e alocação de recursos foi proposta considerando a colaboração entre *fog computing* e *cloud computing* usando as regras de decisão de um algoritmo de árvore de decisão linear com base no tamanho do serviço, tempo de conclusão e capacidade da máquina virtual. Os autores visavam equilibrar a carga de trabalho e melhorar a alocação de recursos de forma eficiente, considerando tanto o SLA quanto o QoS. No entanto, apenas a perspectiva do provedor foi abordada, e apenas os atributos de capacidades foram considerados na análise.

No trabalho [368], os autores apresentaram um planejamento de capacidade para computação de borda que considera os requisitos de QoS e custo, visando decidir se a tarefa deve ser executada em dispositivos disponíveis na borda ou se deve ser executada em recursos disponíveis na *cloud computing*. Um algoritmo de *benchmarking* é usado para calcular os requisitos de recursos e, em seguida, decidir se o processamento remoto ocorreria na borda ou na *cloud*. Embora a proposta se concentre na perspectiva do usuário, apenas os atributos de capacidades são abordados, e nenhuma característica comportamental é

Tabela 6.6: Trabalhos relacionados à etapa de alocação em *fog computing* [13].

Artigo	Ano	Método	Paradigma	Características	Comportamentos	Requisitos Mínimos	Evita Desperdício	Perspectiva Usuário	Perspectiva Provedor	Trata atributos diferentemente	Custo
[325]	2016	Linearized Decision Tree	Fog	✓		✓	✓		✓		
[368]	2017	Benchmarking Algorithm	Edge	✓		✓		✓			✓
[355]	2018	Regression Markov Chain	Edge	✓	✓	✓	✓	✓			
[369]	2019	FCAP	Fog	✓		✓		✓			
[287]	2019	WSGA / NSGA-II / MOEA/D	Fog	✓					✓		
[244]	2019	Genetic Algorithm	Fog	✓	✓			✓			✓
[237]	2019	PROMETHEE-II	Fog	✓				✓			
[284]	2019	ELECTRE	Fog	✓					✓		✓
[255]	2019		Edge	✓	✓	✓		✓			
[256]	2018	TOPSIS	Fog	✓	✓	✓		✓			✓
[246]	2019		Fog	✓	✓				✓		
[337]	2020		Edge	✓				✓			
[257]	2020		Fog	✓		✓	✓		✓		✓
[304]	2018	AHP	Fog	✓					✓		
[336]	2020		Fog	✓					✓		
[322]	2020		Edge	✓			✓	✓			✓
[303]	2020		Fog	✓				✓			
[370]	2021	KL	Fog		✓	✓		✓			
[254]	2022	Monarch-dragon Algorithm	Fog	✓				✓			
[371]	2024	FAHP and FTOPSIS	Fog	✓		✓	✓				
This work	2024	Adapted MCDM	Fog	✓	✓	✓	✓	✓	✓	✓	✓

considerada.

Li *et al.* [355] apresentou um método para agendamento de recursos de *edge computing* baseado em atributos de capacidades e características comportamentais, mas considerando apenas a perspectiva do usuário. Ele usa um grupo de matriz de incidência e classifica os recursos disponíveis. Quando recebe uma solicitação para executar aplicativos, ele seleciona os recursos mais adequados para atender aos seus requisitos. Além disso, um preditor de cadeia de regressão de Markov foi usado para atualizar o *status* do recurso selecionado como um primeiro passo, reduzindo a probabilidade de recursos insuficientes para a continuidade da execução devido à mobilidade e à mudança constante na carga de trabalho. Ao tratar atributos de capacidades e características comportamentais igualmente, ou seja, sem a possibilidade de que estas últimas sejam menores do que o requerido pelo requerente, a seleção do melhor recurso pode ser ponderada.

Características comportamentais e formas de evitar desperdício de recursos não foram consideradas em [369] e [287]. No primeiro, artigo [369] foi proposto um algoritmo que indicava as preferências do usuário para atributos de capacidades, como armazenamento, processamento e quantidades de memória. No último artigo [287] foram propostos e comparados algoritmos genéticos para alocação de recursos em *fog computing*. O algoritmo genético para ordenação proposto obteve melhores resultados na alocação de recursos com

base nos atributos de capacidades, enquanto um algoritmo baseado em decomposição teve melhor desempenho na redução dos tempos de execução de tarefas.

Algoritmos genéticos também foram usados em [244] para alocar máquinas virtuais e tarefas no ambiente de *fog computing*. Além disso, eles também usam pesos para representar o custo e garantir a QoS, mas não conseguem garantir a conformidade com os requisitos mínimos solicitados.

Métodos bem conhecidos foram usados para abordar o problema do MCDM para propor modelos adaptativos para agendamento, seleção e alocação de recursos, atribuindo pesos aos seus critérios para escolher os recursos que melhor correspondiam às suas necessidades de QoS. Em [237], os autores usaram o método PROMETHEE (*Preference Ranking Organization Method for Enrichment Evaluation*), enquanto em [284] e em [255] os autores usaram o método ELECTRE (*Elimination Et Choice Translating Reality*). Nenhum deles propôs uma maneira de evitar o desperdício de recursos ou considerou tratar atributos de capacidades e características comportamentais de forma diferente, como neste artigo.

O método chamado *Analytic Hierarchy Process* (AHP) foi usado em [336], [304], [303] e [322]. Embora tenham atendido a alguns critérios de QoS estabelecidos, o método não pode garantir que os requisitos mínimos exigidos sejam atendidos. Isso também ocorre com o processo conhecido como *Technique for Order of Preference by Similarity to Ideal Solution* (TOPSIS) que foi usado em [256], [246], [337] e [257]. Ainda assim, ele também vai contra as limitações do método para atingir uma qualidade de entrega mais refinada.

Um fator de ponderação para preferência do usuário é usado em [370], para obter o QoS necessário e selecionar o serviço mais adequado para dispositivos IoT. No entanto, o trabalho é focado na recomendação de serviço de *fog nodes* com base na confiabilidade dos usuários e usando um método chamado divergência Kullback–Leibler (KL). Com isso, apenas características comportamentais são cobertas.

Em [254], os autores usaram um novo algoritmo híbrido e, também, o algoritmo de otimização em sua estratégia de alocação, considerando apenas restrições de atributos de capacidades como pontuação de credibilidade, simultaneidade, acessibilidade de preço e computação de tempo de tarefa. Eles não consideraram características comportamentais.

Finalmente, no trabalho [371] os autores usaram dois algoritmos adaptados do AHP e do TOPSIS. Os autores testaram atributos de capacidades computacionais como CPU, memória e latência. No entanto, nenhuma qualidade comportamental foi considerada. Diferentes perspectivas sobre as demandas de usuários e provedores também não foram consideradas.

Pela análise da Tabela 6.6 é possível observar que a maioria dos trabalhos se concentra em encontrar o melhor *fog node* com base apenas em um tipo de atributo (por

exemplo, atributos de capacidades ou características comportamentais), e sem garantir que os requisitos computacionais mínimos sejam atendidos. Além disso, nenhum artigo aborda atributos de capacidades e características comportamentais de forma diferente, ou seja, o mesmo método é usado para tratar ambos os atributos. Isso pode levar a erros e falha em atender aos requisitos dos usuários. Além disso, nenhum artigo apresenta uma solução que considere as perspectivas do usuário e do provedor em profundidade. Isso significa que os recursos computacionais podem ser desperdiçados por serem subutilizados ou superutilizados, o que não é prudente em um ambiente de *fog computing*.

Por fim, ao confrontar as características desta proposta de alocação de recursos para *fog computing* com a revisão da literatura apresentada no Capítulo 4, tem-se que: as *métricas* utilizadas são referentes ao custo e à experiência do usuário; a *técnica* é apoiada em um algoritmo de múltiplo critério de decisão (MCDM); a solução é aplicada nas duas *camadas* inferiores da arquitetura de três camadas, ou seja, nas camadas de IoT e *Fog*; o *modelo de virtualização* é predominantemente por meio de máquinas virtuais; a *avaliação* foi feita em ambientes simulados e em ambientes reais; e, finalmente, o *domínio* que serviu como base para a avaliação de cenários hipotéticos foram as construções inteligentes.

6.7 Considerações Finais

Este capítulo apresentou uma proposta para a alocação de recursos em *fog computing*, baseado no modelo de decisão multicritério e também considerando as capacidades computacionais e as características comportamentais dos *fog nodes*, bem como as perspectivas do usuário e do provedor de serviços. Também foram evidenciados, por meio de um caso de uso, os testes realizados e os resultados obtidos, demonstrando a escalabilidade da solução e sua aderência a um ambiente de *fog computing*.

Esta proposta de alocação de recursos foi apresentada no *17th IEEE/ACM International Conference on Utility and Cloud Computing* (UCC 2024), ocorrido em dezembro de 2024 nos Emirados Árabes Unidos [13], que é Qualis A3 (Anexo X).

Capítulo 7

Conclusão

A pesquisa descrita visou apresentar uma solução para a funcionalidade de alocação de recursos em *fog computing*. O objetivo inicial foi analisar as pesquisas existentes na literatura sobre o processo de gerenciamento de recursos para *fog computing*. Para que esta análise pudesse ser feita de forma consistente foi necessário realizar estudos complementares, como a investigação sobre os paradigmas computacionais relacionados à *fog computing*, bem como a conceituação, por uma perspectiva computacional, dos componentes denominados como *fog nodes*. Tudo isso possibilitou a definição de que o processo de gerenciamento de recursos é composto pelas etapas de estimativa, descoberta, alocação, monitoração e orquestração.

Para a etapa de descoberta de recursos foi elaborado um referencial teórico baseado em publicações que, por vezes, extrapolam os ambientes de *fog computing* e avançam para temas correlatos, principalmente, a *edge computing* e IoT. Neste sentido, foi realizada uma revisão sistemática da literatura, conduzida com foco em responder três questões de pesquisas e que resultou na proposta de uma nova taxonomia para o tema, na contextualização dos artigos analisados em relação à taxonomia proposta. Além disso, a partir deste estudo foram indicados desafios que devem ser considerados pelos pesquisadores para direcionar futuros estudos nesta área.

Entre os desafios citados, estava a necessidade da criação de uma solução de descoberta de recursos que fosse baseada na qualidade de serviço e de experiência. Esta foi a motivação para a criação de uma proposta que atendesse essa deficiência, por meio de um algoritmo que conseguisse obter as informações dos *fog nodes*, e registrar estes dados em um Catálogo de Recursos para ser consumido pelas outras etapas do gerenciamento de recursos em *fog computing*. Assim, com a descrição dos testes realizados e dos resultados obtidos em um ambiente real de testes, foi possível identificar que a solução proposta é capaz de encontrar, de forma eficiente, os *fog nodes* disponíveis e que estejam dentro dos parâmetros esperados, bem como registrá-los no Catálogo de Recursos com pouco esforço

computacional e baixo consumo de rede.

Para a etapa de alocação de recursos, uma pesquisa consistente foi realizada e apresentada. Inicialmente, foi conduzida uma análise dos trabalhos existentes na literatura que visam justamente fazer uma revisão sistemática das publicações sobre o tema, observando os pontos positivos de cada uma delas, mas também enxergando as oportunidades de melhoria. Feito isso, uma extensiva pesquisa foi realizada, resultando na análise de 108 artigos, os quais foram avaliados sobre seis questões de pesquisas elaboradas com o objetivo de extrair informações relevantes sobre o tema.

Entre os resultados obtidos, foi possível identificar um baixo número de propostas para o processo de alocação de recursos que utilizam métodos de MCDM. Além disso, as características de *fog computing* atreladas ao conceito de gerenciamento de recursos utilizado neste trabalho indicaram que este método seria adequado para o tratamento do processo de alocação de recursos. Assim sendo, foi apresentada a proposta de um modelo de alocação de recursos que considera as capacidades computacionais e as características comportamentais do *fog node*, incluindo as perspectivas do usuário final e dos provedores de serviço, permitindo ponderar e balancear esses parâmetros para criar um *ranking* de *fog nodes* disponíveis que atendam à solicitação do usuário. Também foi apresentado um protótipo de sistema que utiliza este novo modelo de forma eficiente, sendo escalável em termos de número de *fog nodes* no ambiente, e aderente às características de *fog computing*.

Por fim, foi apresentada uma extensiva análise experimental da proposta de alocação de recursos, indicando os parâmetros do ambiente de testes, e validando que o modelo do sistema para resolver o problema de alocação em um ambiente de *fog computing* é eficiente quando comparado a soluções similares.

7.1 Trabalhos Futuros

A presente tese avançou o estado-da-arte ao fazer a análise sistemática da literatura sobre a descoberta de recursos e também sobre a alocação de recursos em *fog computing*, e apresentar propostas de soluções para ambas as etapas. No entanto, existem ainda diversas questões em aberto, que merecem investigação em trabalhos futuros.

Com relação ao processo de descoberta, além dos desafios indicados no Capítulo 4, também há avanços a serem conduzidos na proposta apresentada no Capítulo 5. Uma destas melhorias está na extração de informações de *fog nodes* que possuem um sistema operacional diferente do Linux, ampliando a capacidade da solução em registrar mais *fog nodes* no Catálogo de Recursos.

Ainda sobre a proposta de descoberta de recursos, há avanços necessários no tratamento de diversos tipos e topologias de rede, principalmente, com o objetivo de contornar

os conflitos de gerenciamento de tráfego, comumente utilizado pelas operadoras de telecomunicações. Sendo assim, é eminente a utilização de casos de uso em ambiente reais para conhecer e tratar estas situações. Além disso, há possibilidade do algoritmo de descoberta proposto ser executado em estruturas de software de *fog computing*, como FogBus2 [213] para avaliação de ponta a ponta em um ambiente de *fog computing* real para diferentes classes de aplicativos de IoT. Para tanto, é esperado explorar problemas de segurança, o que pode ser crítico para aplicações de IoT sensíveis.

Para a proposta de alocação de recursos apresentada no Capítulo 6, embora a solução apresentada já direcione a utilização de uma arquitetura escalável baseada no Apache Kafka, é indicado como trabalho futuro a avaliação da escalabilidade e do desempenho quando a solução for submetida em situações de testes de *stress*, por exemplo. Uma das indicações para encaminhamento deste trabalho futuro está na distribuição do Catálogo de Recursos em detrimento à implementação atual que é centralizada, mitigando o risco desta arquitetura se tornar um gargalo. Uma outra indicação para tabalhos futuros está na utilização de um Catálogo de Recursos que não seja em forma de matriz, uma vez que isto pode impactar na escalabilidade da solução. Outro ponto que pode ser conduzido como trabalho futuro está na funcionalidade de acionar outros serviços para a obtenção de máquinas não atendidas pelo *fog computing*. Neste cenário, é esperado que a *cloud computing* sejam utilizada para a alocação de recursos não encontrados no ambiente de *fog computing*.

Por fim, é recomendado o avanço nas realizações de testes direcionados para as características de mobilidade e de escalabilidade dos ambientes de *fog computing*, incluindo, quando possível, a realização de testes em cenários e casos de uso reais. Além disso, conforme avançar o desenvolvimento e a utilização da *fog computing* e também houver a oferta deste serviço por provedores públicos, será possível avançar na análise da proposta de alocação de recursos com valores reais tanto para as configurações dos *fog nodes* como também para os preços cobrados no mercado.

Referências

[1] Bachiega, Joao, Breno Costa, Leonardo R. Carvalho, Michel J. F. Rosa e Aleteia Araujo: *Computational resource allocation in fog computing: A comprehensive survey.* ACM Computing Surveys, 55(14s), jul 2023, ISSN 0360-0300. `https://doi.org/10.1145/3586181`. xiii, xv, 12, 13, 31, 32, 40, 41, 70

[2] Bachiega Jr., João, Breno Costa, Leonardo Carvalho, Victor Hugo Oliveira, William Santos, Maria Clícia S. de Castro e Aleteia Araujo: *From the sky to the ground: Comparing fog computing with related distributed paradigms.* Em *Proceedings of the 12th International Conference on Cloud Computing and Services Science (CLOSER 2022)*, páginas 158–169, 2022. xiii, xv, 19, 20, 113

[3] Hong, Cheol Ho e Blesson Varghese: *Resource management in fog/edge computing: A survey on architectures, infrastructure, and algorithms.* ACM Computing Surveys (CSUR), 52(5):97, 2019. xiii, 2, 22, 23, 24, 31, 35

[4] Naha, Ranesh Kumar, Saurabh Garg, Dimitrios Georgakopoulos, Prem Prakash Jayaraman, Longxiang Gao, Yong Xiang e Rajiv Ranjan: *Fog computing: Survey of trends, architectures, requirements, and research directions.* IEEE Access, 6:47980–48009, 2018. xiii, 10, 11, 23, 25, 31

[5] Union, International Telecommunication: *ITU-T y.4460 Recommendation - Architectural reference models of devices for internet of things applications. Released 06/2019*, 2019. xiii, 24, 25, 26, 54

[6] Group, OpenFog Consortium Architecture Working: *Openfog reference architecture for fog computing.* Relatório Técnico, OpenFog Consortium, 2017. xiii, 11, 26, 27, 29

[7] Marín-Tordera, Eva, Xavi Masip-Bruin, Jordi García-Almiñana, Admela Jukan, Guang Jie Ren e Jiafeng Zhu: *Do we all really know what a fog node is? current trends towards an open definition.* Computer Communications, 109:117–130, 2017. xiii, 23, 28, 76

[8] Unikernel: *Unikernel and immutable infrastructures*, 2020. `https://github.com/cetic/unikernels`, acesso em 15/02/2020. xiii, 29

[9] Bachiega, Joao, Breno Gustavo Soares da Costa e Aleteia P. F. Araujo: *Computational perspective of the fog node.* ACSE, 2021. xiii, 2, 29, 30, 37, 76, 79

[10] Costa, Breno, Joao Bachiega, Leonardo Rebouças de Carvalho e Aleteia P. F. Araujo: *Orchestration in fog computing: A comprehensive survey.* ACM Computing Surveys, 55(2), jan 2022, ISSN 0360-0300. `https://doi.org/10.1145/3486221`. xiii, 35, 44, 45

[11] Kitchenham, Barbara, O Pearl Brereton, David Budgen, Mark Turner, John Bailey e Stephen Linkman: *Systematic literature reviews in software engineering–a systematic literature review.* Information and software technology, 51(1):7–15, 2009. xiii, 48, 66

[12] Petersen, Kai, Robert Feldt, Shahid Mujtaba e Michael Mattsson: *Systematic mapping studies in software engineering.* Em *Ease*, volume 8, páginas 68–77, 2008. xiii, 48, 66

[13] Bachiega, Joao, Breno Costa, Michel J. F. Rosa, Leonardo R. Carvalho, Marcelo A. Marotta, Aleteia Araujo e Rajkumar Buyya: *A cost-efficient resource allocation for fog computing with users and providers perspective.* Em *Proceedings of the 2024 IEEE/ACM 17th International Conference on Utility and Cloud Computing (UCC).* IEEE/ACM, 2024. xiv, xv, 112, 113, 114, 116

[14] Mell, P: *The NIST definition of cloud computing.* Recommendations of the National Institute of Standards and Technology, 2011. 1, 7, 8

[15] Vaquero, Luis M, Luis Rodero-Merino, Juan Caceres e Maik Lindner: *A break in the clouds: towards a cloud definition.* ACM SIGCOMM Computer Communication Review, 39(1):50–55, 2008. 1, 6

[16] Costa, Breno GS, Marco Antonio Sousa Reis, Aletéia PF Araújo e Priscila Solis: *Performance and cost analysis between on-demand and preemptive virtual machines.* Em *Proceedings of the 8th International Conference on Cloud Computing and Services Science (CLOSER 2018)*, páginas 169–178, 2018. 1

[17] Junior, João Bachiega, Marco Antonio Sousa Reis, Aleteia PF de Araujo e Maristela Holanda: *Cost optimization on public cloud provider for big geospatial data.* Em *Proceedings of the 7th International Conference on Cloud Computing and Services Science (CLOSER 2017)*, páginas 82–90, 2017. 1

[18] Fox, Armando, Rean Griffith, Anthony Joseph, Randy Katz, Andrew Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica *et al.*: *Above the clouds: A berkeley view of cloud computing.* Dept. Electrical Eng. and Comput. Sciences, University of California, Berkeley, Rep. UCB/EECS, 28(13):2009, 2009. 1, 6

[19] Yousefpour, Ashkan, Caleb Fung, Tam Nguyen, Krishna Kadiyala, Fatemeh Jalali, Amirreza Niakanlahiji, Jian Kong e Jason P. Jue: *All one needs to know about fog computing and related edge computing paradigms: A complete survey.* Journal of Systems Architecture, (December 2018), 2019, ISSN 13837621. 1, 2, 15, 16, 17, 19, 22, 31, 33, 78, 79

[20] Taivalsaari, Antero e Tommi Mikkonen: *A roadmap to the programmable world: software challenges in the IoT era.* IEEE Software, 34(1):72–80, 2017. 1

[21] Gubbi, Jayavardhana, Rajkumar Buyya, Slaven Marusic e Marimuthu Palaniswami: *Internet of things (IoT): A vision, architectural elements, and future directions*. Future Generation Computer Systems (FGCS), 29(7):1645–1660, 2013. 1

[22] Toczé, Klervie e Simin Nadjm-Tehrani: *A Taxonomy for Management and Optimization of Multiple Resources in Edge Computing*. Wireless Communications and Mobile Computing, 2018, 2018. 1, 31, 33, 34, 35, 60

[23] Bonomi, Flavio, Rodolfo Milito, Jiang Zhu e Sateesh Addepalli: *Fog computing and its role in the internet of things*. Em *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*, MCC '12, páginas 13–16, New York, NY, USA, 2012. ACM, ISBN 978-1-4503-1519-7. `http://doi.acm.org/10.1145/2342509.2342513`. 1, 2, 10, 13, 67, 71

[24] Iorga, Michaela, Larry Feldman, Robert Barton, Michael Martin, Nedim Goren e Charif Mahmoudi: *The nist definition of fog computing*. Relatório Técnico, National Institute of Standards and Technology, 2018. 1, 11, 13, 14, 17, 23, 26, 36

[25] Dastjerdi, Amir Vahid, Harshit Gupta, Rodrigo Neves Calheiros, Soumya K. Ghosh e Rajkumar Buyya: *Fog computing: Principles, architectures, and applications*. CoRR, abs/1601.02752, 2016. 2, 10, 11, 15

[26] Ghobaei-Arani, Mostafa, Alireza Souri e Ali A Rahmanian: *Resource management approaches in fog computing: a comprehensive review*. Journal of Grid Computing, páginas 1–42, 2019. 2, 31, 39, 52, 74

[27] Habibi, Pooyan, Mohammad Farhoudi, Sepehr Kazemian, Siavash Khorsandi e Alberto Leon-garcia: *Fog Computing : A Comprehensive Architectural Survey*. IEEE Access, PP:1, 2020. 2, 14, 16, 31, 70

[28] Zarrin, Javad, Rui L Aguiar e João Paulo Barraca: *Resource discovery for distributed computing systems: A comprehensive survey*. Journal of parallel and distributed computing, 113:127–166, 2018. 2, 35, 46, 48, 50, 51, 57, 58, 59, 60

[29] Goudarzi, Parisa, Amir Masoud Rahmani e Mohammad Mosleh: *Resource discovery approaches in cloudiot: a systematic review*. The Journal of Supercomputing, 78(15):17202–17230, 2022. 2, 47, 48, 60

[30] Ahmed, Kosrat Dlshad e Subhi RM Zeebaree: *Resource allocation in fog computing: A review*. International Journal of Science and Business, 5(2):54–63, 2021. 2, 39

[31] Jamil, Bushra, Humaira Ijaz, Mohammad Shojafar, Kashif Munir e Rajkumar Buyya: *Resource allocation and task scheduling in fog computing and internet of everything environments: A taxonomy, review, and future directions*. ACM Computing Surveys (CSUR), 54(11s):1–38, 2022. 2, 40, 68, 71

[32] Sengupta, Souvik: *Adaptive learning-based resource management strategy in fog-to-cloud*. 2020. 2, 30, 79

[33] Foster, Ian, Yong Zhao, Ioan Raicu e Shiyong Lu: *Cloud computing and grid computing 360-degree compared*. Em *Proceedings of the 2008 grid computing environments workshop*, páginas 1–10. IEEE, 2008. 6

[34] Duan, Yucong, Guohua Fu, Nianjun Zhou, Xiaobing Sun, Nanjangud C Narendra e Bo Hu: *Everything as a service (XaaS) on the cloud: origins, current and future trends*. Em *2015 IEEE 8th International Conference on Cloud Computing*, páginas 621–628. IEEE, 2015. 8

[35] Celesti, Antonio, Francesco Tusa, Massimo Villari e Antonio Puliafito: *How to enhance cloud architectures to enable cross-federation*. Em *Proceedings of the 2010 IEEE 3rd international conference on cloud computing*, páginas 337–345. IEEE, 2010. 8

[36] Keahey, Katarzyna, Maurício Tsugawa, Andréa Matsunaga e José A. B. Fortes: *Sky computing*. Em *IEEE Internet Computing*, página 43–51. IEEE Computer Society, 2009. 8, 9

[37] Kritikos, K. e D. Plexousakis: *Multi-cloud application design through cloud service composition*. Em *2015 IEEE 8th Int. Conf. on Cloud Computing*, páginas 686–693, June 2015. 9

[38] Elkhatib, Y.: *Defining cross-cloud systems*. CoRR, abs/1602.02698, 2016. `http://arxiv.org/abs/1602.02698`, acesso em 05/07/2020. 9

[39] Paraiso, F., N. Haderer, P. Merle, R. Rouvoy e L. Seinturier: *A federated multi-cloud PaaS infrastructure*. Em *Proceedings of the 2012 IEEE 5th Int. Conf. on Cloud Computing*, páginas 392–399, June 2012. 9

[40] Grozev, Nikolay e Rajkumar Buyya: *Inter-cloud architectures and application brokering: taxonomy and survey*. Software: Practice and Experience, 44(3):369–390, 2014. 9

[41] Carvalho, Leonardo Rebouças de e Aleteia Patricia Favacho de Araujo: *Performance comparison of terraform and cloudify as multicloud orchestrators*. Em *Proceedings of the 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID)*, páginas 380–389. IEEE, 2020. 9

[42] Shirinkin, Kirill: *Getting Started with Terraform*. Packt Publishing Ltd, 2017. 9

[43] Cloudify: *Getting started*, 2021. `https://cloudify.co/getting-started/`, acesso em 2021/11/29. 9

[44] Michelino, D, JC Leon e LF Alvarez: *Implementation and testing of openstack heat*, 2013. 9

[45] Mukherjee, Mithun, Lei Shu e Di Wang: *Survey of fog computing: Fundamental, network applications, and research challenges*. IEEE Communications Surveys and Tutorials, 20(3):1826–1857, 2018. 10, 11, 13

[46] Hu, Pengfei, Sahraoui Dhelim, Huansheng Ning e Tie Qiu: *Survey on fog computing: architecture, key technologies, applications and open issues*. Journal of Network and Computer Applications, 98(April):27–42, 2017. http://dx.doi.org/10.1016/j.jnca.2017.09.002. 10, 11, 13

[47] Vaquero, Luis M. e Luis Rodero-Merino: *Finding your way in the fog: Towards a comprehensive definition of fog computing*. Special Interest Group on Data Communication (SIGCOMM), 44(5):27–32, outubro 2014, ISSN 0146-4833. 10

[48] Yi, Shanhe, Cheng Li e Qun Li: *A Survey of Fog Computing*. Proceedings of the 2015 Workshop on Mobile Big Data - Mobidata '15, páginas 37–42, 2015. 10, 23, 31

[49] Corp, Cisco: *Fog computing and the internet of things: Extend the cloud to where the things are*. https://www.cisco.com/c/dam/en-us/solutions/trends/iot/docs/computing-overview.pdf, acesso em 07/11/2021. 11

[50] Corp, IBM: *What is fog computing?* https://www.ibm.com/blogs/cloud-computing/2014/08/25/fog-computing/, acesso em 07/11/2021. 11

[51] Gill, Sukhpal Singh, Shreshth Tuli, Minxian Xu, Inderpreet Singh, Karan Vijay Singh, Dominic Lindsay, Shikhar Tuli, Daria Smirnova, Manmeet Singh, Udit Jain, Haris Pervaiz, Bhanu Sehgal, Sukhwinder Singh Kaila, Sanjay Misra, Mohammad Sadegh Aslanpour, Harshit Mehta, Vlado Stankovski e Peter Garraghan: *Transformative effects of IoT, Blockchain and Artificial Intelligence on cloud computing: Evolution, vision, trends and open challenges*. Internet of Things, 8(October):100118, 2019. 11

[52] Bonomi, Flavio, Rodolfo Milito, Preethi Natarajan e Jiang Zhu: *Fog computing: A platform for internet of things and analytics*. Em *Big data and internet of things: A roadmap for smart environments*, páginas 169–186. Springer, 2014. 11, 44

[53] Rahmani, Amir M, Tuan Nguyen Gia, Behailu Negash, Arman Anzanpour, Iman Azimi, Mingzhe Jiang e Pasi Liljeberg: *Exploiting smart e-health gateways at the edge of healthcare internet-of-things: A fog computing approach*. Future Generation Computer Systems (FGCS), 78:641–658, 2018. 11

[54] Tang, Bo, Zhen Chen, Gerald Hefferman, Tao Wei, Haibo He e Qing Yang: *A hierarchical distributed fog computing architecture for big data analysis in smart cities*. Em *Proceedings of the ASE BigData & SocialInformatics 2015*, páginas 1–6. 2015. 11

[55] Huang, Cheng, Rongxing Lu e Kim Kwang Raymond Choo: *Vehicular fog computing: architecture, use case, and security and forensic challenges*. IEEE Communications Magazine, 55(11):105–111, 2017. 11

[56] Sun, Huaiying, Huiqun Yu, Guisheng Fan e Liqiong Chen: *Energy and time efficient task offloading and resource allocation on the generic IoT-fog-cloud architecture*. Peer-to-Peer Networking and Applications, páginas 1–16, 2019. 11, 71, 72, 76, 77

[57] Hung, Shao Chou, Hsiang Hsu, Shao Yu Lien e Kwang Cheng Chen: *Architecture harmonization between cloud radio access networks and fog networks*. IEEE Access, 3:3019–3034, 2015. 12

[58] Huo, Ru, Fei Richard Yu, Tao Huang, Renchao Xie, Jiang Liu, Victor CM Leung e Yunjie Liu: *Software defined networking, caching, and computing for green wireless networks*. IEEE Communications Magazine, 54(11):185–193, 2016. 12

[59] Stojmenovic, Ivan: *Fog computing: A cloud to the ground support for smart things and machine-to-machine networks*. Em *Proceedings of the 2014 Australasian telecommunication networks and applications conference (ATNAC)*, páginas 117–122. IEEE, 2014. 12

[60] Mahmud, Md e Rajkumar Buyya: *Fog Computing: A Taxonomy, Survey and Future Directions*. novembro 2016, ISBN 978-981-10-5861-5. 12, 16, 23, 31, 33, 39

[61] Mouradian, Carla, Diala Naboulsi, Sami Yangui, Roch H. Glitho, Monique J. Morrow e Paul A. Polakos: *A Comprehensive Survey on Fog Computing: State-of-the-Art and Research Challenges*. IEEE Communications Surveys and Tutorials, 20(1):416–464, 2018. 12, 31

[62] Fan, Chih Tien, Zong You Wu, Che Pin Chang e Shyan Ming Yuan: *Web resource cacheable edge device in fog computing*. Proceedings of the 15th International Symposium on Parallel and Distributed Computing, ISPDC 2016, (November 2010):432–439, 2017. 12

[63] Tang, Bo, Zhen Chen, Gerald Hefferman, Tao Wei, Haibo He e Qing Yang: *A Hierarchical Distributed Fog Computing Architecture for Big Data Analysis in Smart Cities*. Proceedings of the ASE BigData & SocialInformatics 2015, (October):28, 2015. 12

[64] Naas, Mohammed Islam, Philippe Raipin Parvedy, Jalil Boukhobza e Laurent Lemarchand: *IFogStor: An IoT Data Placement Strategy for Fog Infrastructure*. Proceedings of the 2017 IEEE 1st International Conference on Fog and Edge Computing, ICFEC 2017, (May 2018):97–104, 2017. 12

[65] Fan, Yaoling, Qiliang Zhu e Yang Liu: *Cloud/fog computing system architecture and key technologies for south-north water transfer project safety*. Wireless Communications and Mobile Computing, 2018, 2018. 12

[66] Mann, Zoltán Ádám: *Notions of architecture in fog computing*. Computing, 103(1):51–73, 2021. 12, 29

[67] Al-Doghman, F., Z. Chaczko, A. R. Ajayan e R. Klempous: *A review on fog computing technology*. Em *Proceedings of the 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, páginas 001525–001530, Oct 2016. 13

[68] Avasalcai, Cosmin, Ilir Murturi e Schahram Dustdar: *Edge and fog: A survey, use cases, and future challenges*. Fog Computing: Theory and Practice, páginas 43–65, 2020. 15

[69] Zishu, LI, XIE Renchao e SUN Li: *A survey of mobile edge computing*. Telecommunications Science, 34(1):87, 2018. 15

[70] Gudipati, Aditya, Daniel Perry, Li Erran Li e Sachin Katti: *Softran: Software defined radio access network*. Em *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*, páginas 25–30, 2013. 15

[71] Dolui, Koustabh e Soumya Kanti Datta: *Comparison of edge computing implementations: Fog computing, cloudlet and mobile edge computing*. Proceedings of the GIoTS 2017 - Global Internet of Things Summit, 2017. 15, 16

[72] Beck, Michael Till, Martin Werner, Sebastian Feld e S Schimper: *Mobile edge computing: A taxonomy*. Em *Proceedings of the 6th International Conference on Advances in Future Internet*, páginas 48–55. Citeseer, 2014. 16

[73] Cui, Mengmeng, Yiming Fei e Yin Liu: *A survey on secure deployment of mobile services in edge computing*. Security and Communication Networks, 2021, 2021. 16

[74] Satyanarayanan, M.: *Fundamental challenges in mobile computing*. Proceedings of the Annual ACM Symposium on Principles of Distributed Computing, páginas 1–7, 1996. 16

[75] Hubaux, J P, Thomas Gross, J Y Le Boudec e Martin Vetterli: *Toward self-organized mobile ad hoc networks: the terminodes project*. IEEE Communications Magazine, 39(1):118–124, 2001. 17

[76] Balasubramanian, Venkatraman e Ahmed Karmouch: *An infrastructure as a service for mobile ad-hoc cloud*. Em *Proceedings of the 2017 IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC)*, páginas 1–7. IEEE, 2017. 17

[77] Yaqoob, Ibrar, Ejaz Ahmed, Abdullah Gani, Salimah Mokhtar, Muhammad Imran e Sghaier Guizani: *Mobile ad hoc cloud: A survey*. Wireless Communications and Mobile Computing, 16(16):2572–2589, 2016. 17

[78] Ranaweera, Pasika, Anca Delia Jurcut e Madhusanka Liyanage: *Survey on multi-access edge computing security and privacy*. IEEE Communications Surveys & Tutorials, 23(2):1078–1124, 2021. 17

[79] Silva, Pedro M Pinto, Joao Rodrigues, Joaquim Silva, Rolando Martins, Luís Lopes e Fernando Silva: *Using edge-clouds to reduce load on traditional wifi infrastructures and improve quality of experience*. Em *Proceedings of the 2017 IEEE 1st International Conference on Fog and Edge Computing (ICFEC)*, páginas 61–67. IEEE, 2017. 17

[80] Preden, Jürgo S, Kalle Tammemäe, Axel Jantsch, Mairo Leier, Andri Riid e Emine Calis: *The benefits of self-awareness and attention in fog and mist computing*. Computer, 48(7):37–45, 2015. 17

[81] Satyanarayanan, Mahadev, Victor Bahl, Ramón Caceres e Nigel Davies: *The case for vm-based cloudlets in mobile computing*. IEEE pervasive Computing, 2009. 18

[82] Alli, Adam A e Muhammad Mahbub Alam: *The fog cloud of things: A survey on concepts, architecture, standards, tools, and applications.* Internet of Things, 9:100177, 2020. 18

[83] Riaz, Nida, Saad Qaisar, Mudassar Ali e Muhammad Naeem: *Node selection and utility maximization for mobile edge computing–driven IoT.* Transactions on Emerging Telecommunications Technologies, página e3704, 2019. 18

[84] Bilal, Kashif, Osman Khalid, Aiman Erbad e Samee U. Khan: *Potentials, trends, and prospects in edge technologies: Fog, cloudlet, mobile edge, and micro data centers.* Computer Networks, 130(2018):94–120, 2018, ISSN 13891286. 18

[85] Li, Yujin e Wenye Wang: *Can mobile cloudlets support mobile applications?* Em *Proceedings of the IEEE Conference on Computer Communications (IEEE INFOCOM 2014)*, páginas 1060–1068. IEEE, 2014. 18

[86] Cui, Yong, Jian Song, Kui Ren, Minming Li, Zongpeng Li, Qingmei Ren e Yangjun Zhang: *Software defined cooperative offloading for mobile cloudlets.* IEEE/ACM Transactions on Networking (TON), 25(3):1746–1760, 2017. 18

[87] Ray, Partha Pratim: *An introduction to dew computing: Definition, concept and implications.* IEEE Access, 6:723–737, 2017. 18

[88] Wang, Yingwei: *Definition and categorization of dew computing.* Open Journal of Cloud Computing (OJCC), 3(1):1–7, 2016. 18

[89] Wang, Yingwei: *Cloud-dew architecture.* International Journal of Cloud Computing, 4(3):199–210, 2015. 19

[90] Skala, Karolj, Davor Davidovic, Enis Afgan, Ivan Sovic e Zorislav Sojat: *Scalable distributed computing hierarchy: Cloud, fog and dew computing.* Open Journal of Cloud Computing (OJCC), 2(1):16–24, 2015. 19

[91] Manvi, Sunilkumar S. e Gopal Krishna Shyam: *Resource management for Infrastructure as a Service (IaaS) in cloud computing: A survey.* Journal of Network and Computer Applications, 41(1):424–440, 2014. 22, 33, 35

[92] Masip, Xavi, Eva Marín, Jordi Garcia e Sergi Sànchez: *Collaborative mechanism for hybrid fog-cloud scenarios.* Fog and Fogonomics: Challenges and Practices of Fog Computing, Communication, Networking, Strategy, and Economics, páginas 7–60, 2020. 22, 35

[93] Devi, Anshu, Ramesh Kait e Virender Ranga: *Security challenges in fog computing.* Em *Handbook of Research on the IoT, Cloud Computing, and Wireless Network Optimization*, páginas 148–164. IGI Global, 2019. 23

[94] Solutions, Cisco Fog Computing: *Unleash the power of the internet of things.* Cisco Systems Inc, 2015. 23

[95] Sarabia-Jácome, David, Regel Gonzalez-Usach e Carlos E Palau: *Iot big data architectures, approaches, and challenges: A fog-cloud approach*. Em *Handbook of Research on Big Data and the IoT*, páginas 125–148. IGI Global, 2019. 23

[96] Singh, Aditya Narayan e Shiva Prakash: *Challenges and opportunities of resource allocation in cloud computing: A survey*. Em *Proceedings of the 2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom)*, páginas 2047–2051. IEEE, 2015. 22

[97] Besen, Stanley M: *The european telecommunications standards institute: A preliminary analysis*. Telecommunications policy, 14(6):521–530, 1990. 24

[98] Hankel, Martin e Bosch Rexroth: *The reference architectural model industrie 4.0 (rami 4.0)*. Zvei, 2(2):4–9, 2015. 24

[99] Perera, Charith, Yongrui Qin, Julio C Estrella, Stephan Reiff-Marganiec e Athanasios V Vasilakos: *Fog computing for sustainable smart cities: A survey*. ACM Computing Surveys (CSUR), 50(3):1–43, 2017. 25, 26

[100] Li, Chao, Yushu Xue, Jing Wang, Weigong Zhang e Tao Li: *Edge-oriented computing paradigms: A survey on architecture design and system management*. ACM Computing Surveys (CSUR), 51(2):1–34, 2018. 26, 31

[101] Tanenbaum, Andrew S e Maarten Van Steen: *Distributed systems: principles and paradigms*. Prentice-Hall, 2007. 28

[102] Abouaomar, Amine, Soumaya Cherkaoui, Abdellatif Kobbane e Oussama Abderrahmane Dambri: *A resources representation for resource allocation in fog computing networks*. Em *2019 IEEE Global Communications Conference (GLOBECOM)*, páginas 1–6. IEEE, 2019. 28, 71, 72, 73, 75, 76, 77

[103] Goldberg, Robert P: *Survey of virtual machine research*. Computer, 7(6):34–45, 1974. 29

[104] Tiburski, Ramao Tiago, Carlos Roberto Moratelli, Sergio F Johann, Marcelo Veiga Neves, Everton de Matos, Leonardo Albernaz Amaral e Fabiano Hessel: *Lightweight security architecture based on embedded virtualization and trust mechanisms for IoT edge devices*. IEEE Communications Magazine, 57(2):67–73, 2019. 29

[105] Merkel, Dirk: *Docker: lightweight linux containers for consistent development and deployment*. Linux journal, 2014(239):2, 2014. 29

[106] Maenhaut, Pieter Jan, Bruno Volckaert, Veerle Ongenae e Filip De Turck: *Resource management in a containerized cloud: Status and challenges*. Journal of Network and Systems Management, 28(2):197–246, 2020. 29

[107] Madhavapeddy, Anil e David J Scott: *Unikernels: the rise of the virtual library operating system*. Communications of the ACM, 57(1):61–69, 2014. 29

[108] Wu, Song, Chao Mei, Hai Jin e Duoqiang Wang: *Android unikernel: Gearing mobile code offloading towards edge computing*. Future Generation Computer Systems (FGCS), 86:694–703, 2018. 29

[109] Cozzolino, Vittorio, Jörg Ott, Aaron Yi Ding e Richard Mortier: *Ecco: Edge-cloud chaining and orchestration framework for road context assessment*. Em *Proceedings of the 2020 IEEE/ACM 5th International Conference on Internet-of-Things Design and Implementation (IoTDI)*, páginas 223–230. IEEE, 2020. 29

[110] Davoli, Gianluca, Davide Borsatti, Daniele Tarchi e Walter Cerroni: *Forch: An orchestrator for fog computing service deployment*. Em *Proceedings of the 2020 IFIP Networking Conference (Networking)*, páginas 677–678. IEEE, 2020. 29

[111] Katoh, Naoki e Toshihide Ibaraki: *Resource allocation problems*. Em *Handbook of combinatorial optimization*, páginas 905–1006. Springer, 1998. 31

[112] Nath, Shubha Brata, Harshit Gupta, Sandip Chakraborty e Soumya K Ghosh: *A survey of fog computing and communication: current researches and future directions*. arXiv preprint arXiv:1804.04365, 2018. 31, 32, 34, 38

[113] Javadpour, Amir, Guojun Wang e Samira Rezaei: *Resource management in a peer to peer cloud network for IoT*. Wireless Personal Communications, 115(3):2471–2488, 2020. 31, 35

[114] Sudhakara, M, K Dinesh Kumar, Ravi Kumar Poluru, R Lokesh Kumar e S Bharath Bhushan: *Towards efficient resource management in fog computing: A survey and future directions*. Em *Architecture and Security Issues in Fog Computing Applications*, páginas 158–182. IGI Global, 2020. 31, 33, 35

[115] Bhajantri, Lokesh B e Gangadharaiah S: *A comprehensive survey on resource management in internet of things*. Journal of Telecommunications and Information Technology, 2020. 31, 33, 35

[116] Rahimi, Morteza, Maryam Songhorabadi e Mostafa Haghi Kashani: *Fog-based smart homes: A systematic review*. Journal of Network and Computer Applications, 153:102531, 2020. 31

[117] Malik, Swati, Kamali Gupta e Malvinder Singh: *Resource management in fog computing using clustering techniques: A systematic study*. Annals of the Romanian Society for Cell Biology, páginas 77–92, 2020. 31

[118] Apat, Hemant Kumar, Prasenjit Maiti, Punyaban Patel *et al.*: *Review on qos aware resource management in fog computing environment*. Em *Proceedings of the 2020 IEEE International Symposium on Sustainable Energy, Signal Processing and Cyber Security (iSSSC)*, páginas 1–6. IEEE, 2020. 31

[119] Haghi Kashani, Mostafa, Amir Masoud Rahmani e Nima Jafari Navimipour: *Quality of service-aware approaches in fog computing*. International Journal of Communication Systems, 33(8):e4340, 2020. 31

[120] Bendechache, Malika, Sergej Svorobej, Patricia Takako Endo e Theo Lynn: *Simulating resource management across the cloud-to-thing continuum: A survey and future directions*. Future Internet, 12(6):95, 2020. 31

[121] Moura, Jose e David Hutchison: *Fog computing systems: State of the art, research issues and future trends, with a focus on resilience*. Journal of Network and Computer Applications, página 102784, 2020. 31

[122] Mahmud, Redowan, Kotagiri Ramamohanarao e Rajkumar Buyya: *Application Management in Fog Computing Environments: A Taxonomy, Review and Future Directions*. 1(1), 2020. 31, 33, 71, 76

[123] Bellendorf, Julian e Zoltán Ádám Mann: *Classification of optimization problems in fog computing*. Future Generation Computer Systems (FGCS), 107:158–176, 2020. 31

[124] Ogundoyin, Sunday Oyinlola e Ismaila Adeniyi Kamil: *Optimization techniques and applications in fog computing: An exhaustive survey*. Swarm and Evolutionary Computation, 66:100937, 2021. 31, 69

[125] Mijuskovic, Adriana, Alessandro Chiumento, Rob Bemthuis, Adina Aldea e Paul Havinga: *Resource management techniques for cloud/fog and edge computing: An evaluation framework and classification*. Sensors, 21(5):1832, 2021. 31, 32, 39

[126] Islam, Mir Salim Ul, Ashok Kumar e Yu Chen Hu: *Context-aware scheduling in fog computing: A survey, taxonomy, challenges and future directions*. Journal of Network and Computer Applications, página 103008, 2021. 31

[127] Matrouk, Khaled e Kholoud Alatoun: *Scheduling algorithms in fog computing: A survey*. International Journal of Networked and Distributed Computing, 9(1):59–74, 2021. 31, 39

[128] Luo, Quyuan, Shihong Hu, Changle Li, Guanghui Li e Weisong Shi: *Resource scheduling in edge computing: A survey*. IEEE Communications Surveys & Tutorials, 2021. 31, 39

[129] Sharghivand, Nafiseh, Farnaz Derakhshan e Nazli Siasi: *A comprehensive survey on auction mechanism design for cloud/edge resource management and pricing*. IEEE Access, 9:126502–126529, 2021. 31, 65

[130] Laroui, Mohammed, Boubakr Nour, Hassine Moungla, Moussa A Cherif, Hossam Afifi e Mohsen Guizani: *Edge and fog computing for IoT: A survey on current research activities and future directions*. Computer Communications, 2021. 31

[131] Mustafa, Saad, Babar Nazir, Amir Hayat, Atta Ur Rehman Khan e Sajjad A. Madani: *Resource management in cloud computing: Taxonomy, prospects, and challenges*. Computers and Electrical Engineering, 47:186–203, 2015. 33, 71

[132] Aazam, Mohammad e Eui Nam Huh: *Fog computing micro datacenter based dynamic resource estimation and pricing model for IoT*. Proceedings of the International Conference on Advanced Information Networking and Applications, AINA, 2015-April(March):687–694, 2015. 33, 71, 72, 73, 75, 76, 77, 84, 107

[133] Battula, Sudheer Kumar, Saurabh Garg, Ranesh Kumar Naha, Parimala Thulasiraman e Ruppa Thulasiram: *A Micro-Level Compensation-Based Cost Model for Resource Allocation in a Fog Environment*. Sensors, 19(13):2954, 2019. 33, 71, 72, 73, 77

[134] Li, Qiuping, Junhui Zhao, Yi Gong e Qingmiao Zhang: *Energy-efficient computation offloading and resource allocation in fog computing for internet of everything*. China Communications, 16(3):32–41, 2019. 33, 71, 72, 77

[135] Auluck, Nitin, Akramul Azim e Kaneez Fizza: *Improving the schedulability of real-time tasks using fog computing*. IEEE Transactions on Services Computing, 2019. 33

[136] Concone, Federico, Giuseppe Lo Re e Marco Morana: *A fog-based application for human activity recognition using personal smart devices*. ACM Transactions on Internet Technology (TOIT), 19(2):1–20, 2019. 33

[137] Benblidia, Mohammed Anis, Bouziane Brik, Leila Merghem-Boulahia e Moez Esseghir: *Ranking fog nodes for tasks scheduling in fog-cloud environments: A fuzzy logic approach*. Em *Proceedings of the 2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC)*, páginas 1451–1457. IEEE, 2019. 33

[138] Pereira, Paulo, Jean Araujo, Matheus Torquato, Jamilson Dantas, Carlos Melo e Paulo Maciel: *Stochastic performance model for web server capacity planning in fog computing*. The Journal of Supercomputing, 76(12):9533–9557, 2020. 34

[139] Aazam, Mohammad, Marc St-Hilaire, Chung Horng Lung, Ioannis Lambadaris e Eui Nam Huh: *IoT resource estimation challenges and modeling in fog*. Springer, 2018. 34

[140] Singh, Sukhpal e Inderveer Chana: *Cloud resource provisioning: survey, status and future research directions*. Knowledge and Information Systems, 49(3):1005–1069, 2016. 35

[141] Datta, Soumya Kanti, Rui Pedro Ferreira Da Costa e Christian Bonnet: *Resource discovery in internet of things: Current trends and future standardization aspects*. Em *Proceedings of the 2015 IEEE 2nd World Forum on Internet of things (WF-IOT)*, páginas 542–547. IEEE, 2015. 35

[142] Khalil, Kasem, Khalid Elgazzar, Mohamed Seliem e Magdy Bayoumi: *Resource discovery techniques in the internet of things: a review*. Internet of Things, 12:100293, 2020. 36, 37, 47, 48, 59

[143] Costa, Breno, João Bachiega, Leonardo Rebouças Carvalho, Michel Rosa e Aleteia Araujo: *Monitoring fog computing: A review, taxonomy and open challenges*. Computer Networks, página 109189, 2022, ISSN 1389-1286. `https://www.sciencedirect.com/science/article/pii/S1389128622002845`. 37, 38, 42, 43, 59, 65

[144] Pourghebleh, Behrouz, Vahideh Hayyolalam e Amir Aghaei Anvigh: *Service discovery in the internet of things: review of current trends and research challenges*. Wireless Networks, 26(7):5371–5391, 2020. 37, 38, 47, 48, 51, 58

[145] Xiao, Zheng, Pijun Liang, Zhao Tong, Kenli Li, Samee U Khan e Keqin Li: *Self-adaptation and mutual adaptation for distributed scheduling in benevolent clouds*. Concurrency and Computation: Practice and Experience, 29(5):e3939, 2017. 38

[146] Patil-Karpe, Sharmila, SH Brahmananda e Shrunoti Karpe: *Review of Resource Allocation in Fog Computing*. Springer, 2020. 39

[147] Dumitru, Marian Cosmin, Mihnea Alexandru Moisescu e Radu Pietraru: *A review of dynamic resource allocation in edge and fog oriented enterprise systems*. Em *Proceedings of the 2021 23rd International Conference on Control Systems and Computer Science (CSCS)*, páginas 295–301. IEEE, 2021. 40

[148] Rahul, Satyakam e Rajni Aron: *Fog computing architecture, application and resource allocation: A review*. Proceedings of the CEUR Workshop, 2889:31–42, 2021, ISSN 16130073. 40

[149] Arpaci-Dusseau, Remzi H, Andrea Arpaci-Dusseau e Venkat Venkataramani: *Cloud-native file systems*. Em *Proceedings of the 10th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 18)*, 2018. 41

[150] Syed, Hassan Jamil, Abdullah Gani, Raja Wasim Ahmad, Muhammad Khurram Khan e Abdelmuttlib Ibrahim Abdalla Ahmed: *Cloud monitoring: A review, taxonomy, and open research issues*. Journal of Network and Computer Applications, 98:11–26, 2017. 41, 65

[151] Abderrahim, Mohamed, Meryem Ouzzif, Karine Guillouard, Jerome Francois e Adrien Lebre: *A holistic monitoring service for fog/edge infrastructures: a foresight study*. Em *Proceedings of the 2017 IEEE 5th International Conference on Future Internet of Things and Cloud (FiCloud)*, páginas 337–344. IEEE, 2017. 41

[152] Brandón, Álvaro, María S Pérez, Jesus Montes e Alberto Sanchez: *Fmone: A flexible monitoring solution at the edge*. Wireless Communications and Mobile Computing, 2018, 2018. 41, 42

[153] Karumuri, Suman, Franco Solleza, Stan Zdonik e Nesime Tatbul: *Towards observability data management at scale*. ACM SIGMOD Record, 49(4):18–23, 2021. 42

[154] Ewaschuk, Rob e Betsy Beyer: *Monitoring distributed systems. site reliability engineering: How google runs production systems, chapter 6*, 2016. 42

[155] Marie-Magdelaine, Nicolas, Toufik Ahmed e Gauthier Astruc-Amato: *Demonstration of an observability framework for cloud native microservices*. Em *Proceedings of the 2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, páginas 722–724. IEEE, 2019. 42

[156] *ORCHESTRATION|meaning*. `https://dictionary.cambridge.org/dictionary/english/orchestration`, acesso em 17/02/2022. 43

[157] *Orchestration | Definition of Orchestration*. `https://www.merriam-webster.com/dictionary/orchestration`, acesso em 17/02/2021. 43

[158] Nguyen, Phu, Nicolas Ferry, Gencer Erdogan, Hui Song, Stéphane Lavirotte, Jean Yves Tigli e Arnor Solberg: *Advances in deployment and orchestration approaches for iot-a systematic review*. Em *Proceedings of the 2019 IEEE International Congress on Internet of Things (ICIOT)*, páginas 53–60. IEEE, 2019. 43

[159] Ferry, Nicolas, Phu Nguyen, Hui Song, Pierre Emmanuel Novac, Stéphane Lavirotte, Jean Yves Tigli e Arnor Solberg: *Genesis: Continuous orchestration and deployment of smart iot systems*. Em *Proceedings of the 2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC)*, volume 1, páginas 870–875. IEEE, 2019. 43

[160] Viejo, Alexandre e David Sánchez: *Secure and privacy-preserving orchestration and delivery of fog-enabled iot services*. Ad Hoc Networks, 82:113–125, 2019. 43

[161] Peltz, C: *Web services orchestration and composition*. Computer, 36(10):46–52, 2003. 43

[162] Wen, Zhenyu, Renyu Yang, Peter Garraghan, Tao Lin, Jie Xu e Michael Rovatsos: *Fog orchestration for internet of things services*. IEEE Internet Computing, 21(2):16–24, 2017. 43

[163] Jiang, Yuxuan, Zhe Huang e Danny HK Tsang: *Challenges and solutions in fog computing orchestration*. IEEE Network, 32(3):122–129, 2017. 43

[164] Brito, Mathias Santos de, Saiful Hoque, Thomas Magedanz, Ronald Steinke, Alexander Willner, Daniel Nehls, Oliver Keils e Florian Schreiner: *A Service Orchestration Architecture for Fog-enabled Infrastructures*. páginas 127–132. IEEE, 2017, ISBN 978-1-5386-2859-1. 43

[165] Meshkova, Elena, Janne Riihijärvi, Marina Petrova e Petri Mähönen: *A survey on resource discovery mechanisms, peer-to-peer and service discovery frameworks*. Computer networks, 52(11):2097–2128, 2008. 47, 48, 53

[166] Jafari Navimipour, Nima e Farnaz Sharifi Milani: *A comprehensive study of the resource discovery techniques in peer-to-peer networks*. Peer-to-Peer networking and applications, 8:474–492, 2015. 47, 48

[167] Khatibi, Elahe e Mohsen Sharifi: *Resource discovery mechanisms in pure unstructured peer-to-peer systems: a comprehensive survey*. Peer-to-Peer Networking and Applications, 14:729–746, 2021. 47, 48

[168] Bukhari, Afnan, Farookh Khadeer Hussain e Omar K Hussain: *Fog node discovery and selection: A systematic literature review*. Future Generation Computer Systems (FGCS), 2022. 47, 48

[169] Breiman, Leo e Adele Cutler: *A deterministic algorithm for global optimization*. Mathematical Programming, 58(1-3):179–199, 1993. 51

[170] Floyd, Robert W: *Nondeterministic algorithms*. Journal of the ACM, 14(4):636–644, 1967. 51

[171] Arzovs, Audris, Janis Judvaitis, Krisjanis Nesenbergs e Leo Selavo: *Distributed learning in the iot–edge–cloud continuum*. Machine Learning and Knowledge Extraction, 6(1):283–315, 2024. 53

[172] Al-Fuqaha, Ala, Mohsen Guizani, Mehdi Mohammadi, Mohammed Aledhari e Moussa Ayyash: *Internet of things: A survey on enabling technologies, protocols, and applications*. IEEE communications surveys & tutorials, 17(4):2347–2376, 2015. 53, 54, 55

[173] Dizdarević, Jasenka, Francisco Carpio, Admela Jukan e Xavi Masip-Bruin: *A survey of communication protocols for internet of things and related challenges of fog and cloud computing integration*. ACM Computing Surveys (CSUR), 51(6):1–29, 2019. 54, 55

[174] Elhadi, Sakina, Abdelaziz Marzak, Nawal Sael e Soukaina Merzouk: *Comparative study of IoT protocols*. Proceedings of the Smart Application and Data Analysis for Smart Cities (SADASC'18), 2018. 54, 55

[175] Weinstein, Ron: *Rfid: a technical overview and its application to the enterprise*. IT professional, 7(3):27–33, 2005. 54

[176] Bisdikian, Chatschik: *An overview of the bluetooth wireless technology*. IEEE Communications magazine, 39(12):86–94, 2001. 54

[177] Ergen, Sinem Coleri: *Zigbee/ieee 802.15. 4 summary*. UC Berkeley, September, 10(17):11, 2004. 54

[178] Ma, Yongsen, Gang Zhou e Shuangquan Wang: *Wifi sensing with channel state information: A survey*. ACM Computing Surveys (CSUR), 52(3):1–36, 2019. 54

[179] Borkar, Suresh R: *Long-term evolution for machines (LTE-M)*. Elsevier, 2020. 54

[180] Lippuner, Stefan, Benjamin Weber, Mauro Salomon, Matthias Korb e Qiuting Huang: *EC-GSM-IoT network synchronization with support for large frequency offsets*. Em *Proceedings of the 2018 IEEE Wireless Communications and Networking Conference (WCNC)*, páginas 1–6. IEEE, 2018. 54

[181] Sinha, Rashmi Sharan, Yiqiao Wei e Seung Hoon Hwang: *A survey on LPWA technology: LoRa and NB-IoT*. ICT Express, 3(1):14–21, 2017. 54

[182] Chettri, Lalit e Rabindranath Bera: *A comprehensive survey on internet of things (IoT) toward 5G wireless systems.* IEEE Internet of Things Journal, 7(1):16–32, 2019. 54

[183] Lavric, Alexandru, Adrian I Petrariu e Valentin Popa: *Sigfox communication protocol: The new era of IoT?* Em *Proceedings of the 2019 international conference on sensing and instrumentation in IoT Era (ISSI)*, páginas 1–4. IEEE, 2019. 54

[184] Forouzan, Behrouz A: *TCP/IP protocol suite.* McGraw-Hill Higher Education, 2002. 55

[185] Soni, Dipa e Ashwin Makwana: *A survey on mqtt: a protocol of internet of things (iot).* Em *International Conference On Telecommunication, Power Analysis And Computing Techniques (ICTPACT-2017)*, volume 20, páginas 173–177, 2017. 55, 63

[186] Naik, Nitin: *Choice of effective messaging protocols for iot systems: Mqtt, coap, amqp and http.* Em *2017 IEEE international systems engineering symposium (ISSE)*, páginas 1–7. IEEE, 2017. 55, 61, 63

[187] Saint-Andre, Peter, Kevin Smith e Remko Tronçon: *XMPP: the definitive guide.* " O'Reilly Media, Inc.", 2009. 55

[188] Lee, Jonghyeok, Jungdo Han e Jaesang Cha: *A study on ssdp protocol based iot/iol device discovery algorithm for energy harvesting interworking smart home.* International Journal of Internet, Broadcasting and Communication, 10(2):7–12, 2018. 55

[189] Postel, Jon: *RFC0768: User datagram protocol.* Relatório Técnico, 1980. 55

[190] Joshi, Manveer e Bikram Pal Kaur: *Coap protocol for constrained networks.* International journal of wireless and microwave technologies, 5(6):1–10, 2015. 56, 61

[191] Zhang, Lixia, Deborah Estrin, Jeffrey Burke, Van Jacobson, James D Thornton, Diana K Smetters, Beichuan Zhang, Gene Tsudik, Dan Massey e Christos Papadopoulos: *Named data networking (NDN) project.* Relatório Técnico NDN-0001, Xerox Palo Alto Research Center-PARC, 157:158, 2010. 56, 61, 63, 64

[192] Goodchild, Andrew: *An overview of catalog design problems in resource discovery.* Internet Research, 6(1):33–43, 1996. 57

[193] Zorgati, Hela, Raoudha Ben Djemaa e Ikram Amous Ben Amor: *Service discovery techniques in internet of things: a survey.* Em *Proceedings of the 2019 IEEE international conference on systems, man and cybernetics (SMC)*, páginas 1720–1725. IEEE, 2019. 58

[194] Curbera, Francisco, Matthew Duftler, Rania Khalaf, William Nagy, Nirmal Mukhi e Sanjiva Weerawarana: *Unraveling the web services web: an introduction to soap, wsdl, and uddi.* IEEE Internet computing, 6(2):86–93, 2002. 58, 61

[195] Newcomer, Eric: *Understanding Web Services: XML, Wsdl, Soap, and UDDI*. Addison-Wesley Professional, 2002. 58

[196] Deutsch, Alin, Mary Fernandez, Daniela Florescu, Alon Levy e Dan Suciu: *A query language for XML*. Computer networks, 31(11-16):1155–1169, 1999. 58

[197] Compton, Michael, Payam Barnaghi, Luis Bermudez, Raul Garcia-Castro, Oscar Corcho, Simon Cox, John Graybeal, Manfred Hauswirth, Cory Henson e Arthur Herzog: *The ssn ontology of the W3C semantic sensor network incubator group*. Journal of Web Semantics, 17:25–32, 2012. 58

[198] Bauer, Martin e Joachim W Walewski: *The iot architectural reference model as enabler*. Enabling Things to Talk: Designing IoT solutions with the IoT Architectural Reference Model, páginas 17–25, 2013. 58

[199] Pan, Jeff Z: *Resource description framework*. Em *Handbook on ontologies*, páginas 71–90. Springer, 2009. 58

[200] Fensel, Dieter, Frank Van Harmelen, Ian Horrocks, Deborah L McGuinness e Peter F Patel-Schneider: *Oil: An ontology infrastructure for the semantic web*. IEEE intelligent systems, 16(2):38–45, 2001. 58

[201] Korf, Richard E: *Linear-space best-first search*. Artificial intelligence, 62(1):41–78, 1993. 59

[202] Zhou, Rong e Eric A Hansen: *Breadth-first heuristic search*. Artificial Intelligence, 170(4-5):385–408, 2006. 59

[203] Yoo, Andy, Edmond Chow, Keith Henderson, William McLendon, Bruce Hendrickson e Umit Catalyurek: *A scalable distributed parallel breadth-first search algorithm on bluegene/l*. Em *SC'05: Proceedings of the 2005 ACM/IEEE Conference on Supercomputing*, páginas 25–25. IEEE, 2005. 59

[204] Arango, Jesus, Mikael Degermark, Alon Efrat e Stephen Pink: *An efficient flooding algorithm for mobile ad-hoc networks*. Em *Proc. of WiOpt*, páginas 1–7, 2004. 59

[205] Tanganelli, Giacomo, Carlo Vallati e Enzo Mingozzi: *Edge-centric distributed discovery and access in the internet of things*. IEEE Internet of Things Journal, 5(1):425–438, 2017. 61, 62

[206] Santos, José, Tim Wauters, Bruno Volckaert e Filip De Turck: *Towards dynamic fog resource provisioning for smart city applications*. Em *Proceedings of the 2018 14th International Conference on Network and Service Management (CNSM)*, páginas 290–294. IEEE, 2018. 61, 62, 71, 72, 75, 78

[207] Jin, Wenquan e Dohyeun Kim: *Consistent registration and discovery scheme for devices and web service providers based on RAML using embedded RD in OCF IoT network*. Sustainability, 10(12):4706, 2018. 61, 62, 90, 92, 93, 94

[208] Kamel, Mohammed BM, Bruno Crispo e Peter Ligeti: *A decentralized and scalable model for resource discovery in IoT network*. Em *Proceedings of the 2019 international conference on wireless and mobile computing, networking and communications (WiMob)*, páginas 1–4. IEEE, 2019. 61, 62

[209] Murturi, Ilir, Cosmin Avasalcai, Christos Tsigkanos e Schahram Dustdar: *Edge-to-edge resource discovery using metadata replication*. Em *Proceedings of the 2019 IEEE 3rd International Conference on Fog and Edge Computing (ICFEC)*, páginas 1–6. IEEE, 2019. 61, 63

[210] Campioni, Lorenzo, Niccolò Fontana, Alessandro Morelli, Niranjan Suri e Mauro Tortonesi: *A federated platform to support IoT discovery in smart cities and HADR scenarios*. Em *Proceedings of the 2020 15th Conference on Computer Science and Information Systems (FedCSIS)*, páginas 511–519. IEEE, 2020. 61, 62

[211] Sattari, Arash, Rouhollah Ehsani, Teemu Leppänen, Susanna Pirttikangas e Jukka Riekki: *Edge-supported microservice-based resource discovery for mist computing*. Em *Proceedings of the 2020 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*, páginas 462–468. IEEE, 2020. 61, 63, 92, 93, 94

[212] Kondo, Daishi, Thomas Ansquer, Yosuke Tanigawa e Hideki Tode: *Resource discovery for edge computing over named data networking*. Em *Proceedings of the 2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC)*, páginas 552–559. IEEE, 2021. 61, 63, 93

[213] Deng, Qifan, Mohammad Goudarzi e Rajkumar Buyya: *Fogbus2: a lightweight and distributed container-based framework for integration of iot-enabled systems with edge and cloud computing*. Em *Proceedings of the International Workshop on Big Data in Emergent Distributed Environments*, páginas 1–8, 2021. 61, 63, 119

[214] Murturi, Ilir e Schahram Dustdar: *A decentralized approach for resource discovery using metadata replication in edge networks*. IEEE Transactions on Services Computing, 15(5):2526–2537, 2021. 61, 63

[215] Islam, Johirul, Tanesh Kumar, Ivana Kovacevic e Erkki Harjula: *Resource-aware dynamic service deployment for local IoT edge computing: Healthcare use case*. IEEE Access, 9:115868–115884, 2021. 61, 63, 90, 93, 94

[216] Wang, Ronghan e Junwei Lu: *Qos-aware service discovery and selection management for cloud-edge computing using a hybrid meta-heuristic algorithm in IoT*. Wireless Personal Communications, 126(3):2269–2282, 2022. 61, 64, 93

[217] Torres, George, Reza Tourani, Abderrahmen Mtibaa, Diana Stelmakh, Satyajayant Misra, Srikathyayani Srikanteswara, Yi Zhang e Sanzida Hoque: *udiscover: User-driven service discovery in pervasive edge computing using NDN*. Em *Proceedings of the 2022 IEEE International Conference on Edge Computing and Communications (EDGE)*, páginas 77–82. IEEE, 2022. 61, 64

[218] Sim, Kwang Mong: *Cooperative and parallel fog discovery and pareto optimal fog commerce bargaining*. IEEE Transactions on Computational Social Systems, 9(4):1112–1121, 2022. 61, 64

[219] Zorgati, Hela, Raoudha Ben Djemaa e Ikram Amous: *Efficient iot resource discovery approach based on P2P networks and fog computing*. Internet of Things, 24:100954, 2023. 61, 64

[220] Farhoudi, Mohammad, Masoud Shokrnezhad, Tarik Taleb, Richard Li e JaeSeung Song: *Discovery of 6g services and resources in edge-cloud-continuum*. IEEE Network, 2024. 61, 64

[221] Stoica, Ion, Robert Morris, David Liben-Nowell, David R Karger, M Frans Kaashoek, Frank Dabek e Hari Balakrishnan: *Chord: a scalable peer-to-peer lookup protocol for internet applications*. IEEE/ACM Transactions on networking, 11(1):17–32, 2003. 61

[222] Lu, Tianxiang, Stephan Merz e Christoph Weidenbach: *Towards verification of the pastry protocol using tla+*. Em *International Conference on Formal Methods for Open Object-Based Distributed Systems*, páginas 244–258. Springer, 2011. 61

[223] Brunner, René e E Biersack: *A performance evaluation of the kad-protocol*. Institut Eurecom, France, 2006. 61

[224] Tortonesi, Mauro, James Michaelis, Alessandro Morelli, Niranjan Suri e Michael A Baker: *Spf: An sdn-based middleware solution to mitigate the iot information explosion*. Em *2016 IEEE Symposium on Computers and Communication (ISCC)*, páginas 435–442. IEEE, 2016. 62

[225] Group, CoRE Working *et al.*: *Constrained application protocol (coap) rfc 7252*. Retrieved, 20:1–9, 2014. 63

[226] Petrakis, Euripides GM, Stelios Sotiriadis, Theodoros Soultanopoulos, Pelagia Tsiachri Renta, Rajkumar Buyya e Nik Bessis: *Internet of things as a service (ITAAS): Challenges and solutions for management of sensor data on the cloud and the fog*. Internet of Things, 3:156–174, 2018. 65

[227] Grossmann, Marcel e Christian Schenk: *A comparison of monitoring approaches for virtualized services at the network edge*. Em *Proceedings of the 2018 International Conference on Internet of Things, Embedded Systems and Communications (IINTEC)*, páginas 85–90. IEEE, 2018. 65

[228] Grossmann, Marcel e Clemens Klug: *Monitoring container services at the network edge*. Em *Proceedings of the 2017 29th International Teletraffic Congress (ITC 29)*, volume 1, páginas 130–133. IEEE, 2017. 65

[229] Babu, R, K Jayashree e R Abirami: *Fog computing qos review and open challenges*. Em *Research Anthology on Architectures, Frameworks, and Integration Strategies for Distributed and Cloud Computing*, páginas 1147–1157. IGI Global, 2021. 65

[230] Chandra, Hans, Erwin Anggadjaja, Pranata Setya Wijaya e Edy Gunawan: *Internet of things: Over-the-air (ota) firmware update in lightweight mesh network protocol for smart urban development*. Em *Proceedings of the 2016 22nd Asia-Pacific Conference on Communications (APCC)*, páginas 115–118. IEEE, 2016. 65

[231] Mourlin, Fabrice e Charif Mahmoudi: *Monitoring architecture for fog and mobile cloud*. Em *Proceedings of the 2018 17th International Symposium on Parallel and Distributed Computing (ISPDC)*, páginas 109–117. IEEE, 2018. 65

[232] Mansouri-Samani, Masoud e Morris Sloman: *Monitoring distributed systems: A survey*. Citeseer, 1992. 65

[233] Gupta, Harshit, Amir Vahid Dastjerdi, Soumya K Ghosh e Rajkumar Buyya: *iFogSim: A toolkit for modeling and simulation of resource management techniques in the internet of things, edge and fog computing environments*. Software: Practice and Experience, 47(9):1275–1296, 2017. 65, 77, 78, 111

[234] Alwasel, Khaled, Devki Nandan Jha, Fawzy Habeeb, Umit Demirbaga, Omer Rana, Thar Baker, Scharam Dustdar, Massimo Villari, Philip James, Ellis Solaiman *et al.*: *IoTSim-Osmosis: A framework for modeling and simulating IoT applications over an edge-cloud continuum*. Journal of Systems Architecture, 116:101956, 2021. 65

[235] Mahmud, Redowan, Samodha Pallewatta, Mohammad Goudarzi e Rajkumar Buyya: *Ifogsim2: An extended ifogsim simulator for mobility, clustering, and microservice management in edge and fog computing environments*. Journal of Systems and Software, 190:111351, 2022. 65

[236] Fahimullah, Muhammad, Shohreh Ahvar, Mihir Agarwal e Maria Trocan: *Machine learning-based solutions for resource management in fog computing*. Multimedia Tools and Applications, 83(8):23019–23045, 2024. 68

[237] Mishra, Manoj Kumar, Niranjan Kumar Ray, Amulya Ratna Swain, Ganga Bishnu Mund e Bhabani Sankar Prasad Mishra: *An adaptive model for resource selection and allocation in fog computing environment*. Computers & Electrical Engineering, 77:217–229, 2019. 71, 72, 73, 75, 77, 114, 115

[238] Kaur, Kuljeet, Tanya Dhand, Neeraj Kumar e Sherali Zeadally: *Container-as-a-service at the edge: Trade-off between energy efficiency and service availability at fog nano data centers*. IEEE wireless communications, 24(3):48–56, 2017. 71, 72, 76

[239] Wang, Tian, Yuzhu Liang, Weijia Jia, Muhammad Arif, Anfeng Liu e Mande Xie: *Coupling resource management based on fog computing in smart city systems*. Journal of Network and Computer Applications, 135:11–19, 2019. 71, 72, 73, 77, 78

[240] Yan, Liangliang, Min Zhang, Chuang Song, Danshi Wang, Jin Li e Luyao Guan: *Deep learning-based containerization resource management in vehicular fog computing*. Em *Asia Communications and Photonics Conference*, páginas M4A–213. Optical Society of America, 2019. 69, 71, 72, 74, 76, 77, 78

[241] Asensio, A, X Masip-Bruin, RJ Durán, I de Miguel, G Ren, S Daijavad e A Jukan: *Designing an efficient clustering strategy for combined fog-to-cloud scenarios*. Future Generation Computer Systems (FGCS), 2020. 71, 72, 75, 77

[242] Gowri, AS *et al.*: *Fog resource allocation through machine learning algorithm*. Em *Architecture and Security Issues in Fog Computing Applications*, páginas 1–41. IGI Global, 2020. 71, 72, 74, 75, 76, 77

[243] Mostafa, Nour, Ismaeel Al Ridhawi e Moayad Aloqaily: *Fog resource selection using historical executions*. Em *Proceedings of the 2018 Third International Conference on Fog and Mobile Edge Computing (FMEC)*, páginas 272–276. IEEE, 2018. 71, 72, 77, 78

[244] Akintoye, Samson Busuyi e Antoine Bagula: *Improving quality-of-service in cloud/-fog computing through efficient resource allocation*. Sensors, 19(6):1267, 2019. 71, 72, 73, 75, 76, 77, 80, 114, 115

[245] Kochar, Vrinda: *Real Time Resource Allocation on a Dynamic Two Level Symbiotic Fog Architecture*. Proceedings of the 2016 6th International Symposium on Embedded Computing and System Design (ISED), páginas 49–55. 71, 72, 75, 77

[246] Bashir, Hayat, Seonah Lee e Kyong Hoon Kim: *Resource allocation through logistic regression and multicriteria decision making method in iot fog computing*. Transactions on Emerging Telecommunications Technologies, página e3824, 2019. 71, 72, 73, 75, 76, 77, 78, 79, 114, 115

[247] Skarlat, Olena, Stefan Schulte, Michael Borkowski e Philipp Leitner: *Resource provisioning for IoT services in the fog*. Em *Proceedings of the 2016 IEEE 9th international conference on service-oriented computing and applications (SOCA)*, páginas 32–39. IEEE, 2016. 69, 71, 72, 74, 75, 76, 77

[248] Lan, Yanwen, Xiaoxiang Wang, Dongyu Wang, Zhaolin Liu e Yibo Zhang: *Task caching, offloading, and resource allocation in d2d-aided fog computing networks*. IEEE Access, 7:104876–104891, 2019. 71, 72, 75, 77

[249] Yin, Luxiu, Juan Luo e Haibo Luo: *Tasks scheduling and resource allocation in fog computing based on containers for smart manufacturing*. IEEE Transactions on Industrial Informatics, 14(10):4712–4721, 2018. 71, 72, 73, 75, 76, 77, 78

[250] Wu, Chu ge, Wei Li, Ling Wang e Albert Y Zomaya: *An evolutionary fuzzy scheduler for multi-objective resource allocation in fog computing*. Future Generation Computer Systems (FGCS), 117:498–509, 2021. 71, 72, 73, 75, 77

[251] Santos, José, Tim Wauters, Bruno Volckaert e Filip De Turck: *Towards end-to-end resource provisioning in fog computing over low power wide area networks*. Journal of Network and Computer Applications, 175:102915, 2021. 69, 71, 72, 75, 76, 77, 78, 79

[252] Etemadi, Masoumeh, Mostafa Ghobaei-Arani e Ali Shahidinejad: *Resource provisioning for iot services in the fog computing environment: An autonomic approach*. Computer Communications, 161:109–131, 2020. 69, 71, 72, 74, 75, 76, 77

[253] Alencar, Derian, Cristiano Both, Rodolfo Antunes, Helder Oliveira, Eduardo Cerqueira e Denis Rosário: *Dynamic microservice allocation for virtual reality distribution with qoe support.* IEEE Transactions on Network and Service Management, 2021. 71, 72, 75, 76, 78

[254] Harika, Sonti e B Chaitanya Krishna: *Multi-objective optimization-oriented resource allocation in the fog environment: A new hybrid approach.* International Journal of Information Technology and Web Engineering (IJITWE), 17(1):1–25, 2022. 69, 71, 72, 75, 76, 77, 114, 115

[255] Lin, Mingwei, Zheyu Chen, Huchang Liao e Zeshui Xu: *Electre ii method to deal with probabilistic linguistic term sets and its application to edge computing.* Nonlinear Dynamics, 96(3):2125–2143, 2019. 71, 72, 73, 75, 77, 80, 114, 115

[256] Bangui, Hind, Said Rakrak, Said Raghay e Barbora Buhnova: *Moving towards smart cities: A selection of middleware for fog-to-cloud services.* Applied Sciences, 8(11):2220, 2018. 71, 72, 73, 75, 77, 114, 115

[257] Baranwal, Gaurav, Ravi Yadav e Deo Prakash Vidyarthi: *Qoe aware iot application placement in fog computing using modified-topsis.* Mobile Networks and Applications, 25(5):1816–1832, 2020. 71, 72, 73, 75, 77, 114, 115

[258] Zhang, F., Z. Tang, M. Chen, X. Zhou e W. Jia: *A dynamic resource overbooking mechanism in fog computing.* Em *Proceedings of the 15th IEEE International Conference on Mobile Ad Hoc and Sensor Systems, MASS 2018*, páginas 89–97, 2018. 71, 72, 74, 75, 77

[259] Zhang, Huaqing, Yanru Zhang, Yunan Gu, Dusit Niyato e Zhu Han: *A Hierarchical Game Framework for Resource Management in Fog Computing.* IEEE Communications Magazine, 55(8):52–57, 2017. 71, 72

[260] Nguyen, Duong Tung, Long Bao Le e Vijay K Bhargava: *A market-based framework for multi-resource allocation in fog computing.* IEEE/ACM Transactions on Networking, 2019. 71, 72, 73, 77

[261] Tadakamalla, Uma e Daniel A Menascé: *Autonomic resource management using analytic models for fog/cloud computing.* Em *2019 IEEE International Conference on Fog Computing (ICFC)*, páginas 69–79. IEEE, 2019. 69, 71, 72, 75, 77

[262] Gu, Lin, Deze Zeng, Song Guo, Ahmed Barnawi e Yong Xiang: *Cost efficient resource management in fog computing supported medical cyber-physical system.* IEEE Transactions on Emerging Topics in Computing, 5(1):108–119, 2015. 71, 72, 76, 77, 78, 79

[263] Lu, Shuaibing, Jie Wu, Yubin Duan, Ning Wang e Zhiyi Fang: *Cost-efficient resource provisioning in delay-sensitive cooperative fog computing.* Em *Proceedings of the 2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS)*, páginas 706–713. IEEE, 2018. 71, 72, 73, 75, 77

[264] Kumar, Ranesh, Saurabh Garg, Andrew Chan e Sudheer Kumar: *Deadline-based dynamic resource allocation and provisioning algorithms in Fog-Cloud environment.* Future Generation Computer Systems (FGCS), 104:131–141, 2020. 69, 71, 72, 73, 75, 76, 77

[265] Jia, Boqi, Honglin Hu, Yu Zeng, Tianheng Xu e Yang Yang: *Double-matching resource allocation strategy in fog computing networks based on cost efficiency.* Journal of Communications and Networks, 20(3):237–246, 2018. 71, 72, 76, 77

[266] Arshad, Hafsa, Munam Ali Shah, Hasan Ali Khattak, Zoobia Ameer, Assad Abbas e Samee Ullah Khan: *Evaluating bio-inspired optimization techniques for utility price estimation in fog computing.* Em *Proceedings of the 2018 IEEE International Conference on Smart Cloud (SmartCloud)*, páginas 84–89. IEEE, 2018. 71, 72, 73, 75, 76, 77

[267] Yao, Jingjing e Nirwan Ansari: *Fog resource provisioning in reliability-aware iot networks.* IEEE Internet of Things Journal, 6(5):8262–8269, 2019. 71, 72, 73, 76, 77

[268] Jie, Yingmo, Mingchu Li, Cheng Guo e Ling Chen: *Game-theoretic online resource allocation scheme on fog computing for mobile multimedia users.* China Communications, 16(3):22–31, 2019. 71, 72, 74, 75, 77

[269] Arkian, Hamid Reza, Abolfazl Diyanat e Atefe Pourkhalili: *MIST: Fog-based data analytics scheme with cost-efficient resource provisioning for IoT crowdsensing applications.* Journal of Network and Computer Applications, 82(August 2016):152–165, 2017, ISSN 10958592. 71, 72, 75, 76, 77, 78

[270] Ni, Lina, Jinquan Zhang e Jiguo Yu: *Priced timed petri nets based resource allocation strategy for fog computing.* Em *Proceedings of the 2016 International Conference on Identification, Information and Knowledge in the Internet of Things (IIKI)*, páginas 39–44. IEEE, 2016. 71, 72, 73, 77

[271] Souza Toniolli, Jean Lucas de e Brigitte Jaumard: *Resource allocation for multiple workflows in cloud-fog computing systems.* Em *Proceedings of the 12th IEEE/ACM International Conference on Utility and Cloud Computing Companion*, páginas 77–84, 2019. 69, 71, 72, 75, 77

[272] Javaid, Sakeena, Nadeem Javaid, Sahrish Khan Tayyaba, Norin Abdul Sattar, Bibi Ruqia e Maida Zahid: *Resource allocation using fog-2-cloud based environment for smart buildings.* Em *Proceedings of the 2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC)*, páginas 1173–1177. IEEE, 2018. 69, 71, 72, 73, 75, 76, 77, 78, 79

[273] Skarlat, Olena, Matteo Nardelli, Stefan Schulte e Schahram Dustdar: *Towards qos-aware fog service placement.* Em *Proceedings of the 2017 IEEE 1st international conference on Fog and Edge Computing (ICFEC)*, páginas 89–96. IEEE, 2017. 71, 72, 75, 77

[274] Natesha, BV e Ram Mohana Reddy Guddeti: *Adopting elitism-based genetic algorithm for minimizing multi-objective problems of iot service placement in fog computing environment.* Journal of Network and Computer Applications, 178:102972, 2021. 69, 71, 72, 73, 75, 76, 77

[275] Khumalo, Nosipho N, Olutayo O Oyerinde e Luzango Mfupe: *Reinforcement learning-based resource management model for fog radio access network architectures in 5g.* IEEE Access, 9:12706–12716, 2021. 69, 71, 72, 74, 75

[276] Lu, Shuaibing, Jie Wu, Yubin Duan, Ning Wang e Juan Fang: *Towards cost-efficient resource provisioning with multiple mobile users in fog computing.* Journal of Parallel and Distributed Computing, 146:96–106, 2020. 71, 72, 75, 77

[277] Farooq, Muhammad Junaid e Quanyan Zhu: *Qoe based revenue maximizing dynamic resource allocation and pricing for fog-enabled mission-critical iot applications.* IEEE Transactions on Mobile Computing, 2020. 71, 72, 75, 76, 77

[278] Chang, Zheng, Liqing Liu, Xijuan Guo e Quan Sheng: *Dynamic resource allocation and computation offloading for IoT fog computing system.* IEEE Transactions on Industrial Informatics, 17(5):3348–3357, 2020. 71, 72, 73, 75, 77

[279] Hazra, Abhishek, Mainak Adhikari, Tarachand Amgoth e Satish Narayana Srirama: *Stackelberg game for service deployment of IoT-enabled applications in 6G-aware fog networks.* IEEE Internet of Things Journal, 8(7):5185–5193, 2020. 69, 71, 72, 74, 75, 77

[280] Hosseinpour, Farhoud, Ahmad Naebi, Seppo Virtanen, Tapio Pahikkala, Hannu Tenhunen e Juha Plosila: *A resource management model for distributed multi-task applications in fog computing networks.* IEEE Access, 9:152792–152802, 2021. 71, 72, 73, 75, 77

[281] Luong, Nguyen Cong, Yutao Jiao, Ping Wang, Dusit Niyato, Dong In Kim e Zhu Han: *A machine-learning-based auction for resource trading in fog computing.* IEEE Communications Magazine, 58(3):82–88, 2020. 71, 72, 73, 75, 76, 77

[282] Goudarzi, Mohammad, Marimuthu Palaniswami e Rajkumar Buyya: *A distributed deep reinforcement learning technique for application placement in edge and fog computing environments.* IEEE Transactions on Mobile Computing, 22(5):2491–2505, 2021. 71, 72, 74, 75

[283] Jain, Vibha e Bijendra Kumar: *Auction based cost-efficient resource allocation by utilizing blockchain in fog computing.* Transactions on Emerging Telecommunications Technologies, 33(7):e4469, 2022. 71, 72, 73, 75, 77

[284] Lera, Isaac, Carlos Guerrero e Carlos Juiz: *Analyzing the applicability of a multi-criteria decision method in fog computing placement problem.* Em *Proceedings of the 2019 4th International Conference on Fog and Mobile Edge Computing (FMEC)*, páginas 13–20. IEEE, 2019. 69, 71, 72, 73, 75, 77, 114, 115

[285] Zhou, Zhenyu, Pengju Liu, Junhao Feng, Yan Zhang, Shahid Mumtaz e Jonathan Rodriguez: *Computation resource allocation and task assignment optimization in vehicular fog computing: A contract-matching approach.* IEEE Transactions on Vehicular Technology, 68(4):3113–3125, 2019. 71, 72, 73, 75, 77, 78

[286] Luo, Juan, Luxiu Yin, Jinyu Hu, Chun Wang, Xuan Liu, Xin Fan e Haibo Luo: *Container-based fog computing architecture and energy-balancing scheduling algorithm for energy IoT.* Future Generation Computer Systems (FGCS), 97:50–60, 2019. 71, 72, 73, 76, 77

[287] Guerrero, Carlos, Isaac Lera e Carlos Juiz: *Evaluation and efficiency comparison of evolutionary algorithms for service placement optimization in fog architectures.* Future Generation Computer Systems, 97:131–144, 2019. 71, 72, 73, 75, 77, 114

[288] Reddy, D Arunkumar e P Venkata Krishna: *Feedback-based fuzzy resource management in IoT using fog computing.* Evolutionary Intelligence, páginas 1–13, 2020. 69, 71, 72, 74, 75, 76, 77

[289] Souza, Vitor Barbosa C, Wilson Ramírez, Xavier Masip-Bruin, Eva Marín-Tordera, G Ren e Ghazal Tashakor: *Handling service allocation in combined fog-cloud scenarios.* Em *Proceedings of the 2016 IEEE international conference on communications (ICC)*, páginas 1–5. IEEE, 2016. 71, 72, 75, 77

[290] Mukherjee, Mithun, Suman Kumar, Mohammad Shojafar, Qi Zhang e Constandinos X Mavromoustakis: *Joint Task Offloading and Resource Allocation for Delay-sensitive Fog Networks.* Proceedings of the ICC 2019 - 2019 IEEE International Conference on Communications (ICC), páginas 1–7, 2020. 71, 72, 75, 76, 77

[291] Li, Lei, Quansheng Guan, Lianwen Jin e Mian Guo: *Resource allocation and task offloading for heterogeneous real-time tasks with uncertain duration time in a fog queueing system.* IEEE Access, 7:9912–9925, 2019. 69, 71, 72, 73, 75, 76, 77

[292] Silva, Rodrigo AC da e Nelson LS da Fonseca: *Resource allocation mechanism for a fog-cloud infrastructure.* Em *Proceedings of the 2018 IEEE International Conference on Communications (ICC)*, páginas 1–6. IEEE, 2018. 69, 71, 72, 75, 76, 77

[293] Taneja, Mohit e Alan Davy: *Resource aware placement of IoT application modules in fog-cloud computing paradigm.* Em *Proceedings of the 2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, páginas 1222–1228. IEEE, 2017. 69, 71, 72, 73, 75, 76, 77

[294] Gill, Sukhpal Singh, Peter Garraghan e Rajkumar Buyya: *Router: Fog enabled cloud based intelligent resource management approach for smart home iot devices.* Journal of Systems and Software, 154:125–138, 2019. 69, 71, 72, 73, 75, 76, 77, 78, 79

[295] Nguyen, Quang Hung e Thanh An Truong Pham: *Studying and developing a resource allocation algorithm in fog computing.* Em *Proceedings of the 2018 International Conference on Advanced Computing and Applications (ACOMP)*, páginas 76–82. IEEE, 2018. 71, 72, 73, 75, 77

[296] Souza, Vitor Barbosa, Xavier Masip-Bruin, Eva Marín-Tordera, Sergio Sànchez-López, Jordi Garcia, Guang Jie Ren, Admela Jukan e A Juan Ferrer: *Towards a proper service placement in combined fog-to-cloud (f2c) architectures*. Future Generation Computer Systems (FGCS), 87:1–15, 2018. 71, 72, 73, 75, 77

[297] Santos, José, Tim Wauters, Bruno Volckaert e Filip De Turck: *Towards network-aware resource provisioning in kubernetes for fog computing applications*. Em *Proceedings of the 2019 IEEE Conference on Network Softwarization (NetSoft)*, páginas 351–359. IEEE, 2019. 71, 72, 75, 76, 77, 78

[298] Lee, Seung seob e SuKyoung Lee: *Resource allocation for vehicular fog computing using reinforcement learning combined with heuristic information*. IEEE Internet of Things Journal, 7(10):10450–10464, 2020. 71, 72, 74, 75, 77, 78

[299] Tang, Chaogang, Chunsheng Zhu, Xianglin Wei, Huaming Wu, Qing Li e Joel JPC Rodrigues: *Intelligent resource allocation for utility optimization in rsu-empowered vehicular network*. IEEE Access, 8:94453–94462, 2020. 71, 72, 73, 75, 77, 78

[300] Zhang, Kecheng, Mugen Peng e Yaohua Sun: *Delay-optimized resource allocation in fog-based vehicular networks*. IEEE Internet of Things Journal, 8(3):1347–1357, 2020. 71, 72, 75, 77, 78

[301] Hassan, Syed Rizwan, Ishtiaq Ahmad, Ateeq Ur Rehman, Seada Hussen e Habib Hamam: *Design of resource-aware load allocation for heterogeneous fog computing environments*. Wireless Communications and Mobile Computing, 2022(1):3543640, 2022. 71, 72, 73, 75, 78

[302] Garg, Kanika, Naveen Chauhan e Rajeev Agrawal: *Optimized resource allocation for fog network using neuro-fuzzy offloading approach*. Arabian Journal for Science and Engineering, 47(8):10333–10346, 2022. 71, 72, 74, 75, 77, 78

[303] Varshney, Shefali, Rajinder Sandhu e PK Gupta: *Qoe-based multi-criteria decision making for resource provisioning in fog computing using ahp technique*. International Journal of Knowledge and Systems Science (IJKSS), 11(4):17–30, 2020. 71, 72, 73, 75, 77, 114, 115

[304] Eswaran, Subha P, Sridhar Sripurushottama e Manoj Jain: *Multi criteria decision making (mcdm) based spectrum moderator for fog-assisted internet of things*. Procedia computer science, 134:399–406, 2018. 71, 72, 73, 75, 77, 114, 115

[305] Chen, Xincheng, Yuchen Zhou, Bintao He e Lu Lv: *Energy-efficiency fog computing resource allocation in cyber physical internet of things systems*. IET Communications, 13(13):2003–2011, 2019. 71, 72, 77

[306] Yu, Y., X. Bu, K. Yang e Z. Han: *Green fog computing resource allocation using joint benders decomposition, dinkelbach algorithm, and modified distributed inner convex approximation*. Em *Proceedings of the 2018 IEEE International Conference on Communications (ICC)*, páginas 1–6, May 2018. 71, 72, 77

[307] Li, Xi, Yiming Liu, Hong Ji, Heli Zhang e Victor CM Leung: *Optimizing resources allocation for fog computing-based internet of things networks*. IEEE Access, 7:64907–64922, 2019. 71, 72, 76, 77

[308] Xavier, Tiago CS, Igor L Santos, Flavia C Delicato, Paulo F Pires, Marcelo P Alves, Tiago S Calmon, Ana C Oliveira e Claudio L Amorim: *Collaborative resource allocation for cloud of things systems*. Journal of Network and Computer Applications, 159:102592, 2020. 69, 71, 72, 73, 75, 76, 77

[309] Zhuang, Yan e Hui Zhou: *A hyper-heuristic resource allocation algorithm for fog computing*. Em *Proceedings of the 2020 the 4th International Conference on Innovation in Artificial Intelligence*, páginas 194–199, 2020. 71, 72, 73, 75, 77

[310] Rehman, Anees Ur, Zulfiqar Ahmad, Ali Imran Jehangiri, Mohammed Alaa Ala'Anzy, Mohamed Othman, Arif Iqbal Umar e Jamil Ahmad: *Dynamic energy efficient resource allocation strategy for load balancing in fog environment*. IEEE Access, 8:199829–199839, 2020. 71, 72, 73, 75, 76, 77

[311] Gai, Keke, Xiao Qin e Liehuang Zhu: *An energy-aware high performance task allocation strategy in heterogeneous fog computing environments*. IEEE Transactions on Computers, 70(4):626–639, 2020. 71, 72, 73, 75, 77

[312] Bi, Fan, Sebastian Stein, Enrico Gerding, Nick Jennings e Thomas La Porta: *A truthful online mechanism for resource allocation in fog computing*. Em Nayak, Abhaya C. e Alok Sharma (editores): *PRICAI 2019: Trends in Artificial Intelligence*, páginas 363–376, Cham, 2019. Springer International Publishing. 71, 72, 73, 77

[313] Jiao, Yutao, Ping Wang, Dusit Niyato e Kongrath Suankaewmanee: *Auction mechanisms in cloud/fog computing resource allocation for public blockchain networks*. IEEE Transactions on Parallel and Distributed Systems, 30(9):1975–1989, 2019. 71, 72, 73, 75, 76, 77

[314] Aazam, Mohammad e Eui Nam Huh: *Dynamic resource provisioning through fog micro datacenter*. Em *Proceedings of the 2015 IEEE international conference on pervasive computing and communication workshops (PerCom workshops)*, páginas 105–110. IEEE, 2015. 71, 72, 73, 76, 77

[315] Zhang, Lei e Jiangtao Li: *Enabling robust and privacy-preserving resource allocation in fog computing*. IEEE Access, 6:50384–50393, 2018. 71, 72, 73, 77

[316] Aazam, Mohammad, Khaled A Harras e Sherali Zeadally: *Fog computing for 5G tactile industrial internet of things: Qoe-aware resource allocation model*. IEEE Transactions on Industrial Informatics, 15(5):3085–3092, 2019. 71, 72, 73, 75, 76, 77

[317] Aazam, Mohammad, Marc St-Hilaire, Chung Horng Lung e Ioannis Lambadaris: *Mefore: QoE based resource estimation at fog to enhance QoS in IoT*. Em *Proceedings of the 2016 23rd International Conference on Telecommunications (ICT)*, páginas 1–5. IEEE, 2016. 71, 72, 73, 75, 76, 77

[318] Kim, Sungwook: *Novel resource allocation algorithms for the social internet of things based fog computing paradigm.* Wireless Communications and Mobile Computing, 2019(1):3065438, 2019. 71, 72, 77

[319] Aazam, Mohammad, Marc St-Hilaire, Chung Horng Lung e Ioannis Lambadaris: *Pre-fog: Iot trace based probabilistic resource estimation at fog.* Em *Proceedings of the 2016 13th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, páginas 12–17. IEEE, 2016. 71, 72, 73, 75, 76, 77

[320] Tong, Shiyuan, Yun Liu, Mohamed Cheriet, Michel Kadoch e Bo Shen: *Ucaa: User-centric user association and resource allocation in fog computing networks.* IEEE Access, 8:10671–10685, 2020. 71, 72, 75, 77

[321] Yang, Lichao, Ming Li, Heli Zhang, Hong Ji, Mingyan Xiao e Xi Li: *Distributed resource management for blockchain in fog-enabled iot networks.* IEEE Internet of Things Journal, 8(4):2330–2341, 2020. 71, 72, 75, 77

[322] Du, Ruizhong, Kunqi Xu e Xiaoyan Liang: *Multiattribute evaluation model based on the ksp algorithm for edge computing.* IEEE Access, 8:146932–146943, 2020. 71, 72, 73, 75, 77, 114, 115

[323] Chen, X., S. Leng, K. Zhang e K. Xiong: *A machine-learning based time constrained resource allocation scheme for vehicular fog computing.* China Communications, 16(11):29–41, 2019. 71, 72, 78

[324] Pereira, Rickson S, Douglas D Lieira, Marco AC da Silva, Adinovam HM Pimenta, Joahannes BD da Costa, Denis Rosário e Rodolfo I Meneguette: *A novel fog-based resource allocation policy for vehicular clouds in the highway environment.* Em *Proceedings of the 2019 IEEE Latin-American Conference on Communications (LATINCOM)*, páginas 1–6. IEEE, 2019. 71, 72, 73, 75, 77, 78

[325] Alsaffar, Aymen Abdullah, Hung Phuoc Pham, Choong Seon Hong, Eui Nam Huh e Mohammad Aazam: *An architecture of IoT service delegation and resource allocation based on collaboration between fog and cloud computing.* Mobile Information Systems, 2016(1):6123234, 2016. 71, 72, 76, 77, 113, 114

[326] Wang, Haoyu, Lina Wang, Zhichao Zhou, Xueqiang Tao, Giovanni Pau e Fabio Arena: *Blockchain-based resource allocation model in fog computing.* Applied Sciences, 9(24):5538, 2019. 71, 72, 75, 77

[327] Zhang, Huaqing, Yong Xiao, Shengrong Bu, Dusit Niyato, F Richard Yu e Zhu Han: *Computing resource allocation in three-tier iot fog networks: A joint optimization approach combining stackelberg game and matching.* IEEE Internet of Things Journal, 4(5):1204–1215, 2017. 71, 72, 74, 77

[328] Fan, Qiang, Jianan Bai, Hongxia Zhang, Yang Yi e Lingjia Liu: *Delay-aware resource allocation in fog-assisted iot networks through reinforcement learning.* IEEE Internet of Things Journal, 9(7):5189–5199, 2021. 71, 72, 75, 77

[329] Jana, Gopal Chandra e Sudatta Banerjee: *Enhancement of QoS for fog computing model aspect of robust resource management.* Em *Proceedings of the 2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT)*, páginas 1462–1466. IEEE, 2017. 71, 72, 73, 75, 76, 77

[330] Gülpınar, Nalan, Ethem Çanakoğlu e Juergen Branke: *Heuristics for the stochastic dynamic task-resource allocation problem with retry opportunities.* European Journal of Operational Research, 266(1):291–303, 2018. 71, 72, 73, 75, 77

[331] Gu, Yunan, Zheng Chang, Miao Pan, Lingyang Song e Zhu Han: *Joint radio and computational resource allocation in IoT fog computing.* IEEE Transactions on Vehicular Technology, 67(8):7475–7484, 2018. 71, 72, 75, 77

[332] Vu, Thai T, Diep N Nguyen, Dinh Thai Hoang e Eryk Dutkiewicz: *Qos-aware fog computing resource allocation using feasibility-finding benders decomposition.* Em *Proceedings of the 2019 IEEE Global Communications Conference (GLOBECOM)*, páginas 1–6. IEEE, 2019. 71, 72, 75, 77

[333] Rakshith, G, MV Rahul, GS Sanjay, BV Natesha e G Ram Mohana Reddy: *Resource provisioning framework for IoT applications in fog computing environment.* Em *Proceedings of the 2018 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*, páginas 1–6. IEEE, 2018. 71, 72, 73, 75, 77

[334] Mani, Sathish Kumar e Iyapparaja Meenakshisundaram: *Improving quality-of-service in fog computing through efficient resource allocation.* Computational Intelligence, 36(4):1527–1547, 2020. 71, 72, 73, 75, 76, 77

[335] Yin, Chao, Tongfang Li, Xiaoping Qu e Sihao Yuan: *An optimization method for resource allocation in fog computing.* Em *Proceedings of the 2020 International Conferences on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress on Cybermatics (Cybermatics)*, páginas 821–828. IEEE, 2020. 71, 72, 75, 77

[336] Mishra, Suchintan, Manmath Narayan Sahoo, Sambit Bakshi e Joel JPC Rodrigues: *Dynamic resource allocation in fog-cloud hybrid systems using multicriteria ahp techniques.* IEEE Internet of Things Journal, 7(9):8993–9000, 2020. 71, 72, 73, 75, 77, 114, 115

[337] Huang, Hualong, Kai Peng e Xiaolong Xu: *Collaborative computation offloading for smart cities in mobile edge computing.* Em *Proceedings of the 2020 IEEE 13th International Conference on Cloud Computing (CLOUD)*, páginas 176–183. IEEE, 2020. 71, 72, 73, 75, 77, 114, 115

[338] Bertsekas, Dimitri P, WW Hager e OL Mangasarian: *Nonlinear programming.* Athena Scientific Belmont, MA, 1998. 72

[339] Baker, Brenda S: *A new proof for the first-fit decreasing bin-packing algorithm.* Journal of Algorithms, 6(1):49–70, 1985. 72

[340] Abapour, Saeed, Morteza Nazari-Heris, Behnam Mohammadi-Ivatloo e Mehrdad Tarafdar Hagh: *Game theory approaches for the solution of power system problems: a comprehensive review.* Archives of Computational Methods in Engineering, 27(1):81–103, 2020. 72, 74

[341] Zill, Dennis, Warren S Wright e Michael R Cullen: *Advanced engineering mathematics.* Jones & Bartlett Learning, 2011. 71

[342] Santos, José, Tim Wauters, Bruno Volckaert e Filip De Turck: *Resource provisioning in fog computing: From theory to practice.* Sensors, 19(10):2238, 2019. 71

[343] Brassard, Gilles e Paul Bratley: *Fundamentals of algorithmics*, volume 524. Prentice Hall Englewood Cliffs, 1996. 73

[344] Desale, Sachin, Akhtar Rasool, Sushil Andhale e Priti Rane: *Heuristic and meta-heuristic algorithms and their relevance to the real world: a survey.* Int. J. Comput. Eng. Res. Trends, 351(5):2349–7084, 2015. 73

[345] Eiben, Agoston E, James E Smith *et al.*: *Introduction to evolutionary computing*, volume 53. Springer, 2003. 73

[346] Nie, Pu yan e Pei ai Zhang: *A note on stackelberg games.* Em *Proceedings of the 2008 Chinese Control and Decision Conference*, páginas 1201–1203. IEEE, 2008. 74

[347] Svorobej, Sergej, Patricia Takako Endo, Malika Bendechache, Christos Filelis-Papadopoulos, Konstantinos M Giannoutakis, George A Gravvanis, Dimitrios Tzovaras, James Byrne e Theo Lynn: *Simulating fog and edge computing scenarios: An overview and research challenges.* Future Internet, 11(3):55, 2019. 76, 77

[348] Margariti, Spiridoula V, Vassilios V Dimakopoulos e Georgios Tsoumanis: *Modeling and simulation tools for fog computing—a comprehensive survey from a cost perspective.* Future Internet, 12(5):89, 2020. 77

[349] Banks, Jerry, John S CARSON II e L Barry: *Discrete-event system simulationfourth edition.* Relatório Técnico, 2005. 77

[350] Buyya, Rajkumar, Rajiv Ranjan e Rodrigo N Calheiros: *Modeling and simulation of scalable cloud computing environments and the cloudsim toolkit: Challenges and opportunities.* Em *Proceedings of the 2009 international conference on high performance computing and simulation*, páginas 1–11. IEEE, 2009. 77, 78

[351] Buyya, Rajkumar e Manzur Murshed: *Gridsim: A toolkit for the modeling and simulation of distributed resource management and scheduling for grid computing.* Concurrency and computation: practice and experience, 14(13-15):1175–1220, 2002. 77, 78

[352] Shampine, Lawrence F e Mark W Reichelt: *The matlab ode suite.* SIAM journal on scientific computing, 18(1):1–22, 1997. 77

[353] Lai, Yongxuan, Fan Yang, Lu Zhang e Ziyu Lin: *Distributed public vehicle system based on fog nodes and vehicular sensing*. IEEE Access, 6:22011–22024, 2018. 78, 79

[354] Clements-Croome, Derek: *Sustainable intelligent buildings for people: A review*. Intelligent Buildings International, 3(2):67–86, 2011. 79

[355] Li, Guangshun, Jianrong Song, Junhua Wu e Jiping Wang: *Method of resource estimation based on QoS in edge computing*. Wireless Communications and Mobile Computing, 2018(1):7308913, 2018. 80, 114

[356] Opricovic, Serafim e Gwo Hshiung Tzeng: *Compromise solution by mcdm methods: A comparative analysis of vikor and topsis*. European journal of operational research, 156(2):445–455, 2004. 80

[357] Zavadskas, Edmundas Kazimieras, Zenonas Turskis e Simona Kildienė: *State of art surveys of overviews on mcdm/madm methods*. Technological and economic development of economy, 20(1):165–179, 2014. 80

[358] Orebaugh, Angela e Becky Pinkard: *Nmap in the enterprise: your guide to network scanning*. Elsevier, 2011. 84

[359] Vafaei, Nazanin, Rita A Ribeiro e Luis M Camarinha-Matos: *Normalization techniques for multi-criteria decision making: analytical hierarchy process case study*. Em *doctoral conference on computing, electrical and industrial systems*, páginas 261–269. Springer, 2016. 98

[360] Streijl, Robert C, Stefan Winkler e David S Hands: *Mean opinion score (MOS) revisited: methods and applications, limitations and alternatives*. Multimedia Systems, 22(2):213–227, 2016. 98

[361] Mukherjee, Anwesha, Debashis De e Rajkumar Buyya: *Resource Management in Distributed Systems*. Springer, 2024. 100

[362] Garg, Nishant: *Apache kafka*. Packt Publishing Birmingham, UK, 2013. 104

[363] Richardson, Leonard e Sam Ruby: *RESTful web services*. " O'Reilly Media, Inc.", 2008. 104

[364] Buyya, Rajkumar e Satish Narayana Srirama: *Fog and edge computing: principles and paradigms*. John Wiley & Sons, 2019. 106

[365] Drucker, Peter: *The practice of management*. Routledge, 2012. 106

[366] Srirama, Satish Narayana: *A decade of research in fog computing: relevance, challenges, and future directions*. Software: Practice and Experience, 54(1):3–23, 2024. 107

[367] Papathanasiou, Jason, Nikolaos Ploskas, Jason Papathanasiou e Nikolaos Ploskas: *TOPSIS*. Springer, 2018. 111

[368] Noreikis, Marius, Yu Xiao e Antti Ylä-Jaäiski: *QoS-oriented capacity planning for edge computing.* Em *Proceedings of the 2017 IEEE International Conference on Communications (ICC)*, páginas 1–6. IEEE, 2017. 113, 114

[369] Li, Guangshun, Yuncui Liu, Junhua Wu, Dandan Lin e Shuaishuai Zhao: *Methods of resource scheduling based on optimized fuzzy clustering in fog computing.* Sensors, 19(9):2122, 2019. 114

[370] Hallappanavar, Vijay L e Mahantesh N Birje: *Prediction of quality of service of fog nodes for service recommendation in fog computing based on trustworthiness of users.* Journal of Reliable Intelligent Environments, 8(2):193–210, 2022. 114, 115

[371] Gad-Elrab, Ahmed AA, Almohammady S Alsharkawy, Mahmoud E Embabi, Ahmed Sobhi e Farouk A Emara: *Adaptive multi-criteria-based load balancing technique for resource allocation in fog-cloud environments.* IJCNC, 16(01), 2024. 114, 115

Anexo I

Artigo publicado no CLOSER 2022 (primeira página)

From the Sky to the Ground: Comparing Fog Computing with Related Distributed Paradigms

Joao Bachiega Jr.[1], Breno Costa[1], Leonardo R. Carvalho[1], Victor H. C. Oliveira[1], William X. Santos[1], Maria Clicia S. de Castro[2] and Aleteia Araujo[1]

[1]*Department of Computer Science - University of Brasília (UnB), Brazil*

[2]*Department of Infomatics and Computer Science, State University of Rio de Janeiro (UERJ), Rio de Janeiro, Brazil*

joao.bachiega.jr@gmail.com, brenogcosta@gmail.com, leouesb@gmail.com, victoroliveira89@live.com, wilxavier@me.com, clicia@ime.uerj.br, aleteia@unb.br

Keywords: Edge Computing, Fog Computing, Cloud Computing, Sky Computing.

Abstract: Fog computing is a paradigm that enables provisioning resources and services at the network edge, closer to end devices, complementing cloud computing and recently it's being embraced by sky computing. Beyond this computational paradigm, there are many other related technologies which also make use of distributed computing to improve the quality of service delivered to the end-users. The main contribution of this paper is to present a comparison between fog computing and nine other relevant related paradigms, namely Sky Computing, Cloud Computing, Edge Computing, Mobile Edge Computing, Mobile Cloud Computing, Mobile Ad hoc Cloud Computing, Mist Computing, Cloudlet Computing, and Dew Computing, highlighting the similarities and differences between them based on the main fog characteristics. A graphical characterization for each paradigm, highlighting its computational power, communication type and position in a three-layers architecture (Cloud-Fog-IoT) and some relevant challenges in this area are also presented.

1 Introduction

Over the years, computing paradigms have been evolving. Since the beginning of the modern computer era until about 1985, computers were large and expensive. Moreover, there was no connectivity between them and they operated independently of each other. With the development of powerful microprocessors and high-speed computer networks, it was possible to change this scenario (Tanenbaum and Van Steen, 2007).

After that, computing technologies evolved from distributed, parallel, clustering, peer-to-peer (P2P), and grid until the cloud computing. There is no doubt that cloud computing is one of the most important computational paradigms, playing an essential role in almost every aspect of our everyday life. It brought the impression of infinite computational resources (e.g. storage, processing) provided in a pay-per-use basis, allowing fast scalability at a reasonable cost. Nevertheless, as a centralized computing model, computation happens in the cloud data centers, located in specific places around the world, and far from the end-users (Mukherjee et al., 2018).

As in a cycle, nowadays is the time when distributed computing paradigms are back in the spotlight, aiming to overcome the limitations of centralized cloud computing processes. Internet of Things (IoT) development demands this (Yousefpour et al., 2019). Thus, new concepts were born ranging from *Sky Computing* (Keahey et al., 2009), which is a level above cloud computing and uses resources and services from several providers simultaneously, until *Fog Computing* (Bonomi et al., 2012), which is a cloud close to the ground, addressing several issues of connected devices, like high-bandwidth, geographical dispersion, and low latency needs. Fog is both complementary to, and an extension of, traditional cloud-based models.

Although fog computing is a recent paradigm, in the last 3 years it was the subject of approximately 20% of all publications related to computational paradigms, including Sky Computing, Edge Computing, Mobile Edge Computing, Mobile Cloud Computing, Mobile Ad hoc Cloud Computing, Mist Computing, Cloudlet Computing and Dew Computing.

The main contribution of this paper is to present a comparison between fog computing and other related paradigms, highlighting similarities and differences

Anexo II

Artigo publicado no ICOMP 2021 (primeira página)

Computational Perspective of the Fog Node

João Bachiega Jr, Breno Costa, and Aletéia P. F. Araujo
Department of Computer Science - University of Brasília (UnB) - Brasília - DF - Brazil
Email: joao.bachiega.jr@gmail.com, brenogscosta@gmail.com, aleteia@unb.br

***Abstract*—Fog computing is a recent computational paradigm that was proposed to solve some weaknesses in cloud-based systems. For this reason, this technology has been extensively studied by several technology areas. It is still in a maturing stage, so there is no consensus in academia about its concepts and definitions, and each area adopts the ones that are convenient for each use case. This article proposes a definition and a classification relying on a computational perspective for the "fog node", which is a fundamental element in a fog computing environment. In addition, the main challenges related to the fog node are also presented.**

***Index Terms*—fog node, fog computing, edge computing, computational resource**

I. Introduction

Fog computing is a paradigm that enables provisioning of resources and services at the network edge, closer to the end devices. It is both complementary to, and an extension of, the traditional cloud-based model, addressing several issues of connected devices, such as high bandwidth, geographical dispersion, and low latency needs [1].

The advantages of a model like fog computing drew the attention not only of the computing area, but also of several other ones, such as telecommunications and electrical engineering. In this perspective, the development of this computational paradigm, as well as the maturation of the concepts and definitions that surround it, are strongly linked to the different use cases of each study area.

Among these concepts is the term "fog node", which is a fundamental element in fog computing architecture. The concept of a fog node is wide and although some works have recently been published proposing taxonomies and definitions for it [2], [3], there is no work, to the best of our knowledge, that has contextualized the fog node in a computational perspective.

Based on an analysis of some publications currently in the academy, this paper aims to contextualize the function of a computational resource in fog computing, as well as to present a definition and a classification for the fog node. So, Section II contextualizes fog computing, presenting the architecture's main characteristics. Section III takes an approach to the computational resource, presenting the desirable characteristics of a fog node and its forms of delivery, as well as presenting a definition to be adopted. Related works are presented and compared with our proposal in Section IV. The challenges inherent to fog computing and fog nodes are presented in Section V. Finally, Section VI brings the conclusions of this article and the future work that will be done next.

II. Fog Computing

Fog Computing is a distributed computing paradigm, integrated into the cloud, whose processing is done at the network edge. It provides computing resources for applications that cannot perform properly with the high latency provided by cloud-only environments [2].

Bonomi et al. [4] presented the first definition of fog computing stating that it is a highly virtualized platform that provides computing, storage, and networking services among many computing data centers or end-devices. These components may or may not be at the edge of the network.

Several researchers have expanded and revised this initial definition of fog computing. Yi et al. [5] consider fog computing as a scenario, composed of a high number of decentralized and heterogeneous devices, where they communicate and cooperate among themselves and with the network to perform data processing and storage without third-party interventions. Services, applications, or basic network functions that run in a sandboxed environment, can be supported by the data processing and storage.

For Dastjerdi et al. [6], fog computing is considered a distributed computing paradigm. In this paradigm, the services provided by the cloud are essentially extended to the network edge. Fog computing addresses application requirements that need low latency with a huge and dense geographical distribution. Therefore, fog computing supports computing resources, different communication protocols, mobility, interface heterogeneity, integration with the cloud, and distributed data analytics.

For Naha et al. [2], fog computing is a distributed platform where the edge or end devices, that can be virtualized or not, will do the majority of processing. It resides in between the cloud and users and the cloud will do long-term storage and non-latency-dependent processing.

In the industry point of view, Cisco [7] defines that the fog extends the cloud to be closer to the things that produce and act on Internet of Things (IoT) data. These devices, called fog nodes, can be any device with computing, storage, and network connectivity and are deployed anywhere with a network connection. For IBM [8], the term "fog computing" and "edge computing" carry the same meaning. They both mean operation on network ends rather than hosting and working from a central cloud. They represent the scenario where processes and resources are located at the edge of the cloud and do not establish any channel for cloud utilization.

The Open Fog Consortium [9], which since January 2019 has merged with the Industrial Internet Consortium, states that

Anexo III

Artigo submetido ao JPDC em 2024 (primeira página)

Resource Discovery in Fog Computing: A Review, Taxonomy, and Future Directions

Joao Bachiega Jr.[a,*], Breno Costa[a], Leonardo R. Carvalho[a], Aleteia Araujo[a,b], Rajkumar Buyya[b]

[a] *University of Brasilia, Department of Computer Science, Brasilia, Brazil*
[b] *University of Melbourne, Cloud Computing and Distributed Systems (CloudS) Labs, Melbourne, Australia*

Abstract

Fog Computing is an emerging paradigm that has gained attention for its ability to provide access to computing resources and services near the edge of the network, close to end users. This paradigm complements traditional Cloud Computing, providing significant advantages in terms of latency, bandwidth, and efficiency in distributed scenarios. The heterogeneous nature and ever-increasing number of connected devices present a substantial obstacle to finding the best resource discovery solutions in this ever-changing and widely distributed environment. This article presents a systematic literature review, specifically on the resource discovery process for Fog Computing systems. The results provide a novel taxonomy of Fog discovery solutions. Additionally, the main challenges in this field are identified, providing a clear overview of future directions and areas that require further investigation.

Keywords: fog Computing, resource discovery, resource management, taxonomy

1. Introduction

Fog Computing emerges as a promising solution to address the growing demand to expand computational, network, and storage capacity close to end users, thus complementing some limitations of Cloud Computing [1]. However, given its status as an emerging technological paradigm, this domain

*Corresponding author

Anexo IV

Artigo publicado na ACM Computing Surveys em 2023 (primeira página)

Computational Resource Allocation in Fog Computing: A Comprehensive Survey

JOAO BACHIEGA JR., BRENO COSTA, LEONARDO R. CARVALHO, MICHEL J. F. ROSA, and ALETEIA ARAUJO, University of Brasilia

Fog computing is a paradigm that allows the provisioning of computational resources and services at the edge of the network, closer to the end devices and users, complementing cloud computing. The heterogeneity and large number of devices are challenges to obtaining optimized resource allocation in this environment. Over time, some surveys have been presented on resource management in fog computing. However, they now lack a broader and deeper view about this subject, considering the recent publications. This article presents a systematic literature review with a focus on resource allocation for fog computing, and in a more comprehensive way than the existing works. The survey is based on 108 selected publications from 2012 to 2022. The analysis has exposed their main techniques, metrics used, evaluation tools, virtualization methods, architecture, and domains where the proposed solutions were applied. The results show an updated and comprehensive view about resource allocation in fog computing. The main challenges and open research questions are discussed, and a new fog computing resource management cycle is proposed.

CCS Concepts: • **General and reference** → **Surveys and overviews**; • **Computer systems organization** → **Cloud computing**; **Distributed architectures;**

Additional Key Words and Phrases: Fog computing, resource allocation, resource management, resource provisioning

ACM Reference format:
Joao Bachiega Jr., Breno Costa, Leonardo R. Carvalho, Michel J. F. Rosa, and Aleteia Araujo. 2023. Computational Resource Allocation in Fog Computing: A Comprehensive Survey. *ACM Comput. Surv.* 55, 14s, Article 336 (July 2023), 31 pages.
https://doi.org/10.1145/3586181

1 INTRODUCTION

Fog computing has emerged as a promising solution to meet the growing demand for expansion of the processing, network, and storage capacity closer to end users, thus complementing the fragility of cloud computing [33]. However, as this is an emerging paradigm, there are several open research questions, with many challenges to be overcome [178]. Among these challenges is computational resource allocation, which aims to provide, in an appropriate manner, the computational resources

The authors would like to thank CAPES, a Brazilian institution that funds research and publishing, for facilitating the access to many publications through CAPES' journal portal (https://www.periodicos.capes.gov.br).
Authors' address: J. Bachiega Jr., B. Costa, L. R. Carvalho, M. J. F. Rosa, and A. Araujo, University of Brasilia, Department of Computer Science, Campus Darcy Ribeiro, Asa Norte, Brasilia, DF, 70910-900, Brazil; emails: {joao.bachiega.jr, brenogcosta, leouesb, micheljunioferreira}@gmail.com, aleteia@unb.br.

0360-0300/2023/07-ART336 $15.00
https://doi.org/10.1145/3586181

Anexo V

Artigo publicado no Mobiquitous 2023 (primeira página)

Achieving Observability on Fog Computing with the Use of Open-Source Tools

Breno Costa[1,2(✉)], Abhik Banerjee[2], Prem Prakash Jayaraman[2], Leonardo R. Carvalho[1], João Bachiega Jr.[1], and Aleteia Araujo[1]

[1] Department of Computer Science, University of Brasília (UnB), Brasília, DF, Brazil
brenogscosta@gmail.com, aleteia@unb.br
[2] School of Science, Computing and Engineering Technologies, Swinburne University of Technology, Melbourne, VIC, Australia
{abanerjee,pjayaraman}@swin.edu.au

Abstract. Fog computing can provide computational resources and low-latency communication at the network edge. But with it comes uncertainties that must be managed in order to guarantee Service Level Agreements. Service observability can help the environment better deal with uncertainties, delivering relevant and up-to-date information in a timely manner to support decision making. Observability is considered a superset of monitoring since it uses not only performance metrics, but also other instrumentation domains such as logs and traces. However, as Fog Computing is typically characterised by resource-constrained nodes and network uncertainties, increasing observability in fog can be risky due to the additional load injected into a restricted environment. There is no work in the literature that evaluated fog observability. In this paper, we first outline the challenges of achieving observability in a Fog environment, based on which we present a formal definition of fog observability. Subsequently, a real-world Fog Computing testbed running a smart city use case is deployed, and an empirical evaluation of fog observability using open-source tools is presented. The results show that under certain conditions, it is viable to provide observability in a Fog Computing environment using open-source tools, although it is necessary to control the overhead modifying their default configuration according to the application characteristics.

Keywords: Observability · Fog Computing · Edge Computing · Metrics · Logs · Traces

1 Introduction

Fog computing is a computing model that brings computation and data storage closer to where it is needed, typically at the edge of the network. The main characteristics of Fog Computing are its distributed and decentralised nature,

A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 594, pp. 319–340, 2024.
https://doi.org/10.1007/978-3-031-63992-0_21

Anexo VI

Capítulo de Livro no Springer Handbook of Data Engineering em 2024 (primeira página)

Chapter 1

Observability in Fog Computing

Abstract - Fog Computing provides computational resources close to the end user, supporting low-latency and high-bandwidth communications. It supports IoT applications, enabling real-time data processing, analytics, and decision-making at the edge of the network. However, the high distribution of its constituent nodes and resource-restricted devices interconnected by heterogeneous and unreliable networks makes it challenging to execute service maintenance and troubleshooting, increasing the time to restore the application after failures and not guaranteeing the service level agreements. In such a scenario, increasing the observability of Fog applications and services may speed up troubleshooting and increase their availability. An observability system is a data-intensive service, and Fog Computing could have its nodes and channels saturated with an additional load. In this work, we detail the three pillars of observability (metrics, log, and traces), discuss the challenges, and clarify the approaches for increasing the observability of services in Fog environments. Furthermore, the system architecture that supports observability in Fog, related tools, and technologies are presented, providing a comprehensive discussion on this subject. An example of a solution shows how a real-world application can benefit from increased observability in this environment. Finally, there is a discussion about the future directions of Fog observability.

Keywords - Observability, Fog Computing, Edge Computing, Metrics, Logs, Traces

Abbreviation	Meaning
CNCF	Cloud Native Computing Foundation
eBPF	enhanced Berkeley Packet Filter
GCT	Garbage Collection Truck
IoT	Internet of Things
MEC	Mobile Edge Computing
ODLC	Observability Data Life Cycle
OTel	OpenTelemetry
P2P	Peer-to-Peer
SLA	Service Level Agreement
SNMP	Simple Network Management Protocol
SPOF	Single Point of Failure
TSDB	Time Series Database
WMI	Windows Management Instrumentation

Anexo VII

Artigo publicado na Computer Networks em 2022 (primeira página)

Computer Networks 215 (2022) 109189

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Review article

Monitoring fog computing: A review, taxonomy and open challenges

Breno Costa *, João Bachiega Jr., Leonardo Rebouças Carvalho, Michel Rosa, Aleteia Araujo

Department of Computer Science - University of Brasilia, Brazil

ARTICLE INFO

Keywords:
Monitoring
Orchestration
Fog computing
Taxonomy
Fog monitoring

ABSTRACT

Fog computing is a distributed paradigm that provides computational resources in the users' vicinity. Fog orchestration is a set of functionalities that coordinate the dynamic infrastructure and manage the services to guarantee the Service Level Agreements. Monitoring is an orchestration functionality of prime importance. It is the basis for resource management actions, collecting status of resource and service and delivering updated data to the orchestrator. There are several cloud monitoring solutions and tools, but none of them comply with fog characteristics and challenges. Fog monitoring solutions are scarce, and they may not be prepared to compose an orchestration service. This paper updates the knowledge base about fog monitoring, assessing recent subjects in this context like observability, data standardization and instrumentation domains. We propose a novel taxonomy of fog monitoring solutions, supported by a systematic review of the literature. Fog monitoring proposals are analyzed and categorized by this new taxonomy, offering researchers a comprehensive overview. This work also highlights the main challenges and open research questions.

1. Introduction

Fog computing is a computational paradigm that complements cloud computing, providing computational resources on the network edge, closer to the users. As a distributed infrastructure, fog computing must deal with heterogeneity of network links and processing capacity of its composing nodes [1]. These characteristics bring complexity to fog management, and it is addressed by the orchestration of services and resources. Orchestration is a management function, composed of several complementary functionalities. It is responsible for dealing with infrastructure dynamicity, for taking timely actions and for assuring that Service Level Agreements (SLAs) are respected [2]. There are several proposals of fog service orchestration in the literature, although most of them, only conceptual.

Monitoring is a functionality of prime importance and it is crucial to properly orchestrate fog services [3]. It collects updated status information about fog nodes and communication links and send them to the orchestrator. With an updated view of fog infrastructure and service execution, the orchestrator can take proper actions to guarantee the SLAs, e.g., offloading a service to a resource richer node and optimizing service placement according to historic data about node failures [4]. Besides the heterogeneity of nodes being monitored, there are other related concerns about frequency, topology, and communication model. There is a trade-off between the frequency of information updates and the overhead to the nodes and to the orchestrator related to generating, transmitting and processing status data. In such a dynamic scenario, adaptability of monitoring parameters can play an important role. In our previous work [4], we did a systematic literature review of fog service orchestration and analyzed 50 proposals. Most of them (40 out of 50) highlighted monitoring as a relevant process, but they frequently assumed that a fog monitoring solution would be available to deliver the information they needed, without presenting either implementation methods or insightful information on the subject.

Monitoring is not only about reporting availability, i.e. the capacity to answer the question of whether a node or a service is online and working properly. It is also about the capacity to explain why a node or service stopped working properly. The former is achieved by monitoring metrics, e.g. service response time. The latter is achieved by monitoring logs, i.e. unstructured strings of text, and traces, i.e. records of requests made by an user in a service. Metrics, logs and traces form what is called Instrumentation Domains of monitoring [5]. Different instrumentation domains can be used simultaneously by a monitoring solution to get different perspectives of a service. In such a scenario, there would be more capacity for decision-making on the server-side, but at the cost of increasing the complexity of monitoring, since their specific characteristics (e.g. life-cycle, data volume) would be managed accordingly. Another emergent concept that is being applied to monitoring microservices is Observability. It is referenced as a superset of monitoring that uses data analytics techniques on the collected

* Corresponding author.
E-mail addresses: brenogscosta@gmail.com (B. Costa), joao.bachiega.jr@gmail.com (J. Bachiega Jr.), leouesb@gmail.com (L.R. Carvalho), micheljunioferreira@gmail.com (M. Rosa), aleteia@unb.br (A. Araujo).

https://doi.org/10.1016/j.comnet.2022.109189
Received 5 April 2022; Received in revised form 16 June 2022; Accepted 13 July 2022
Available online 19 July 2022

Anexo VIII

Artigo publicado na ACM Computing Surveys em 2022 (primeira página)

Orchestration in Fog Computing: A Comprehensive Survey

BRENO COSTA, JOAO BACHIEGA JR., LEONARDO REBOUÇAS DE CARVALHO, and ALETEIA P. F. ARAUJO, University of Brasilia, Brazil

Fog computing is a paradigm that brings computational resources and services to the network edge in the vicinity of user devices, lowering latency and connecting with cloud computing resources. Unlike cloud computing, fog resources are based on constrained and heterogeneous nodes whose connectivity can be unstable. In this complex scenario, there is a need to define and implement orchestration processes to ensure that applications and services can be provided, considering the settled agreements. Although some publications have dealt with orchestration in fog computing, there are still some diverse definitions and functional intersection with other areas, such as resource management and monitoring. This article presents a systematic review of the literature with focus on orchestration in fog computing. A generic architecture of fog orchestration is presented, created from the consolidation of the analyzed proposals, bringing to light the essential functionalities addressed in the literature. This work also highlights the main challenges and open research questions.

CCS Concepts: • **General and reference** → **Surveys and overviews**; • **Computer systems organization** → **Cloud computing**; **Distributed architectures;**

Additional Key Words and Phrases: Fog computing, orchestration, monitoring, resource management

ACM Reference format:
Breno Costa, Joao Bachiega Jr., Leonardo Rebouças de Carvalho, and Aleteia P. F. Araujo. 2022. Orchestration in Fog Computing: A Comprehensive Survey. *ACM Comput. Surv.* 55, 2, Article 29 (January 2022), 34 pages.
https://doi.org/10.1145/3486221

1 INTRODUCTION

Cloud computing is already a mature paradigm that has been in place since 2006 [83]. It offers virtualized resources, created upon a huge shared infrastructure, that are consumed by the customers on a pay-per-use business model [139] and with a variety of cost options [19, 62]. Cloud computing is based on dozens of large datacenters, distributed around the world, that are placed in the center of the network and accessed by the Internet. Cloud computing also provides automatic scalability to its services.

Cloud computing datacenters are composed of thousands of resource-rich homogeneous physical servers that are interconnected by a redundant and stable network [157]. To optimize the infrastructure in use and comply with service and quality agreements made with the customers, a resource orchestration framework is in place.

Authors' address: B Costa, J. Bachiega Jr., L. R. de Carvalho, and A. P. F. Araujo, University of Brasilia, Department of Computer Science, Campus Darcy Ribeiro, Asa Norte, Brasilia, DF, 70910-900, Brazil; emails: {brenogcosta, joao.bachiega.jr, leouesb}@gmail.com, aleteia@unb.br.

0360-0300/2022/01-ART29 $15.00
https://doi.org/10.1145/3486221

Anexo IX

Artigo Apresentado no ICICT em 2025 (primeira página)

An Effective Resource Discovery Strategy for Fog Computing Driven by Computational Capabilities and Behavioral Characteristics

Joao Bachiega Jr.[1], Breno Costa[1] Leonardo R. Carvalho[1], Aleteia Araujo[1,2], and Rajkumar Buyya[2]

[1] University of Brasilia, Brazil
[2] The University of Melbourne, Australia
{joao.bachiega.jr, brenogcosta, leouesb}@gmail.com, aleteia@unb.br, rbuyya@unimelb.edu.au

Abstract. The fog computing paradigm allows for the distribution of computing resources and services at the edge of the network, close to end users, complementing cloud computing. Due to the dynamicity of fog computing environments, resource discovery is a key process that aims to find new computational resources that are available to integrate into it. These resources compose fog nodes, devices considering computational capabilities (such as CPU, memory, and disk) and behavioral characteristics (such as availability, scalability, and mobility). Performing an optimized resource discovery with all those attributes is still a challenge. This article proposes an efficient approach to resource discovery in fog computing that considers the computational capability and behavioral characteristics to select fog nodes. The results show that it is at least 33% more efficient than a similar solution found in the literature.

Keywords: resource discovery, fog computing, behavioral characteristics, computational capability

1 Introduction

Fog Computing has emerged as a promising solution to meet the growing demand to expand the capacity of computational, network, and storage resources closer to end users, compensating for some limitations of cloud Computing [17]. Among the current challenges, the effective discovery of computational resources is crucial. It occurs because in a fog computing environment characterized by the prevalence of dynamic contexts, such as the Internet of Things (IoT), and by competition for allocated computing resources, unpredictable events can arise, such as unavailability of services and devices, long response times, high latency, and reduced reliability [26]. To increase the challenge, applying resource discovery strategies from other consolidated computing paradigms, such as cloud computing, may present additional obstacles, as fog has computational resource constraints and more significant heterogeneity between devices. These points

Anexo X

Artigo Apresentado no UCC em 2024 (primeira página)

A Cost-Efficient Resource Allocation for Fog Computing with Users and Providers Perspective

1st Joao Bachiega Jr.
University of Brasilia, Brazil
joao.bachiega.jr@gmail.com

2nd Breno Costa
University of Brasilia, Brazil
brenogcosta@gmail.com

3rd Michel J. F. Rosa
University of Brasilia, Brazil
michelj@gmail.com

4th Leonardo R. Carvalho
University of Brasilia, Brazil
leouesb@gmail.com

5th Marcelo A. Marotta
University of Brasilia, Brazil
marcelo.marotta@unb.br

6th Aleteia Araujo
University of Brasilia, Brazil
The University of Melbourne, Australia
aleteia@unb.br

7th Rajkumar Buyya
The University of Melbourne, Australia
rbuyya@unimelb.edu.au

***Abstract*—Fog computing paradigm allows allocating computational resources and services at the edge of the network, closer to the end devices and users, complementing Cloud computing. Fog nodes comprise computational capabilities (such as CPU, memory, and disk) and behavioral characteristics (such as availability, scalability, and mobility). It should also be considered that while the end-user's goal is to obtain the best available resources, the service provider's concern is meeting user requests using minimal resources. Furthermore, it is crucial to consider the financial cost of each resource in both scenarios. Taking all of this into account, obtaining a cost-effective optimized resource allocation is viewed as a challenge. In this article, we propose an efficient algorithm for resource allocation, considering both the provider's and the end-user's perspectives, exploiting computational capabilities, behavioral characteristics, and the financial cost. The tests were carried out in both real and simulated environments and demonstrated that our proposal complies with the resource allocation needs for Fog computing and has a better performance compared to a similar solution.**

***Index Terms*—Fog computing, resource allocation, provider's perspective, end-user's perspective, cost-efficiency**

I. Introduction

Fog computing has emerged as a promising solution for Cloud computing bottlenecks with regard to high latency and response time [1]. To address these bottlenecks, Fog computing is designed as a highly geographically distributed and heterogeneous network composed of low computational capacity devices with different setups closer to end-users. The devices can remotely process the workloads initially meant for the Cloud with smaller response time and latency. This network has been used in dynamic contexts, with devices sharing and competing for computational resources, such as the Internet of Things (IoT). This competition creates uncertainty for resource management in Fog computing environments, leading to possible overloaded or underused nodes, causing unnecessary power consumption, service unavailability, high response time, and decreased reliability [2]. In such a scenario, resource allocation to determine suitable resources that satisfy a required workload remains a challenge [3].

A relevant concept in a Fog computing environment is the Fog node, defined as any computational device with system and hardware resources added to high communication capability [4]. A fog node has computational capabilities, such as CPU, storage, memory, and other hardware attributes that can be measured. In addition, reflecting the dynamicity and unpredictability of Fog computing, the Fog node is also made up of some behavioral characteristics, such as availability, interoperability, and mobility [5].

The end-user and service provider perspectives must be considered when implementing Fog computing management. On the one hand, end-users always want to obtain the best resources available, those with greater computational power and higher values for behavioral characteristics. On the other hand, service and infrastructure providers are interested in delivering minimal resources to avoid unnecessary costs or situations where resources are unavailable to other users [6]. Although it is desirable to consider the behavioral characteristics, their degradation is not an obstacle to the execution of the application. In this sense, most Fog computing resource allocation solutions focus exclusively on hardware attributes without considering behavioral characteristics [7] [8]. Despite this trend, some publications consider only the user's perspective [9] [10]. To the best of our knowledge, considering both perspectives, i.e., balancing the needs of end-users and providers with the computational capabilities and behavioral characteristics of the Fog node, is a matter of investigation [6].

This work presents an algorithm for resource allocation in Fog computing, considering hardware attributes and behavioral characteristics. Thus, the main contributions of this work are:

- A Fog node cost-efficient decision-making algorithm that considers both hardware attributes and behavioral characteristics of available nodes, considering the perspectives of end-users and providers. This model allows parameters to be weighted and balanced to create a ranking of available Fog nodes that comply with the user's request;
- A prototype of a system that efficiently uses the proposed algorithm, being scalable in terms of the number of nodes in the environment and adherent to the Fog aspects;
- An analysis of the proposed algorithm efficiency performed in a real test environment, validating the system