Universidade Estadual de Campinas
Instituto de Computação

Marcelo Claudio Sousa Araújo

CMFog: Migração proativa de conteúdo em Computação em Névoa multi-nível

CAMPINAS
2024

Marcelo Claudio Sousa Araújo

CMFog: Migração proativa de conteúdo em Computação em Névoa multi-nível

Tese apresentada ao Instituto de Computação da Universidade Estadual de Campinas como parte dos requisitos para a obtenção do título de Doutor em Ciência da Computação.

Orientador: Prof. Dr. Luiz Fernando Bittencourt

Este exemplar corresponde à versão final da Tese defendida por Marcelo Claudio Sousa Araújo e orientada pelo Prof. Dr. Luiz Fernando Bittencourt.

CAMPINAS
2024

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca do Instituto de Matemática, Estatística e Computação Científica
Ana Regina Machado - CRB 8/5467

Ar15c

Araújo, Marcelo Claudio Sousa, 1991-
CMFog : migração proativa de conteúdo em computação em névoa multi-nível / Marcelo Claudio Sousa Araújo. – Campinas, SP : [s.n.], 2024.

Orientador: Luiz Fernando Bittencourt.
Tese (doutorado) – Universidade Estadual de Campinas, Instituto de Computação.

1. Computação em névoa. 2. Computação em nuvem. 3. Virtualização. 4. Internet das coisas. I. Bittencourt, Luiz Fernando, 1981-. II. Universidade Estadual de Campinas. Instituto de Computação. III. Título.

Informações Complementares

Título em outro idioma: CMFog : proactive content migration in multi-level fog computing
Palavras-chave em inglês:
Fog computing
Cloud computing
Virtualization
Internet of things
Área de concentração: Ciência da Computação
Titulação: Doutor em Ciência da Computação
Banca examinadora:
Luiz Fernando Bittencourt [Orientador]
Júlio Cezar Estrella
Fabio Luciano Verdi
Juliana Freitag Borin
Sandro Rigo
Data de defesa: 10-06-2024
Programa de Pós-Graduação: Ciência da Computação

Identificação e informações acadêmicas do(a) aluno(a)
- ORCID do autor: https://orcid.org/0009-0001-1960-1338
- Currículo Lattes do autor: https://lattes.cnpq.br/8755906380890769

Universidade Estadual de Campinas
Instituto de Computação

Marcelo Claudio Sousa Araújo

CMFog: Migração proativa de conteúdo em Computação em Névoa multi-nível

Banca Examinadora:

- Prof. Dr. Luiz Fernando Bittencourt (Presidente)
  IC/UNICAMP
- Prof. Dr. Júlio Cezar Estrella
  ICMC/USP
- Prof. Dr. Fabio Luciano Verdi
  DComp/UFSCar
- Profa. Dra. Juliana Freitag Borin
  IC/UNICAMP
- Prof. Dr. Sandro Rigo
  IC/UNICAMP

A ata da defesa, assinada pelos membros da Comissão Examinadora, consta no SIGA/Sistema de Fluxo de Dissertação/Tese e na Secretaria do Programa da Unidade.

Campinas, 10 de junho de 2024

Agradecimentos

Ao iniciar minha graduação em Ciência da Computação, a ideia de ingressar em um doutorado e desenvolver uma tese parecia algo distante ou até mesmo inalcançável. Hoje, olhando para trás, sinto uma mistura de nostalgia e realização por esta etapa que se conclui. Tenho carinho e gratidão pelos professores do meu período de graduação, que me mostraram que a área acadêmica poderia ser muito mais do que diplomas ou títulos. Qualquer pessoa que tenha passado por essas fases da vida acadêmica sabe o quanto esse período é cheio de altos e baixos, desde a frustração de momentos difíceis até a alegria das conquistas e do sucesso dos pequenos passos ao longo do caminho. É impossível citar e agradecer a todas as pessoas que foram importantes, mas acho importante destacar algumas delas.

Primeiramente, gostaria de agradecer ao meu orientador, *Prof. Dr. Luiz Fernando Bittencourt*. Muito obrigado pela confiança e oportunidade que me ofereceu desde a seleção para o mestrado até aqui. Obrigado pela dedicação, paciência e compreensão durante todos esses anos, que tanto contribuíram para meu crescimento acadêmico e pessoal. Foi uma honra ter contanto com sua orientação em cada etapa desse árduo caminho.

Durante o doutorado, tive a felicidade de realizar um período de intercâmbio que foi fundamental para o desenvolvimento da tese. Tenho que agradecer aos professores *Dra. Marilia Curado, Dr. Edmundo Monteiro e Dr. Bruno Sousa*, que me ajudaram e contribuíram nos passos iniciais do meu trabalho. Também agradeço pela ajuda e recepção que me proporcionaram durante minha estada no Departamento de Engenharia Informática da Universidade de Coimbra.

Gostaria também de agradecer aos meus pais, *Valéria* e *Antônio Carlos*. Obrigado pelos sacrifícios que fizeram durante toda a minha vida, foram as oportunidades que foram capazes de me oferecer que me levaram onde estou hoje. Também agradeço à minha irmã, *Rafaella*, pelo suporte e por sempre, mesmo sem saber, me mostrar um exemplo de dedicação e perseverança.

Agradeço ao *Prof. Dr. Ary Henrique Moraes de Oliveira*, por ter me guiado e aconselhado durante a minha graduação. Seus conselhos mostraram-me a importância de uma vida acadêmica com dedicação e disciplina. Obrigado por não ter me deixado desistir e por me incentivar a me inscrever no mestrado.

Também gostaria de agradecer aos meus pets, *Athos e Bebê*. Sei que vocês não são capazes de entender nada disso, mas obrigado pela companhia e por cada gesto inocente que, muitas vezes, foram a razão de eu me manter de pé. Perder vocês durante esse processo foi uma das maiores dificuldades que enfrentei.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de Financiamento 001 e MC1 Win The Market.

Resumo

A revolução digital, impulsionada pela difusão do conceito de Internet das Coisas (IoT), transformou radicalmente a interação da sociedade com o mundo digital. Esse novo cenário de computação trouxe consigo desafios e necessidades que evidenciaram as limitações do paradigma da Computação em Nuvem. Ao descentralizar os recursos computacionais e migrá-los para a borda da rede, o conceito de Computação em Névoa surge como uma alternativa para superar as limitações da Nuvem, aproximando a computação dos usuários e visando proporcionar uma infraestrutura capaz de atender às demandas de baixa latência e interações em tempo real. Entretanto, a mobilidade característica dos usuários e dos sensores da borda introduz desafios complexos que precisam ser superados para que a Névoa seja capaz de garantir uma baixa latência independentemente da localização do usuário. Nesse contexto, esta tese propõe o CMFog (*Content Migration in Fog Computing*), uma estratégia de migração proativa de conteúdo na infraestrutura de Névoa multi-nível, com o objetivo de otimizar a latência de comunicação. O CMFog integra a predição de mobilidade para adaptar dinamicamente a infraestrutura às necessidades dos usuários e busca explorar os diferentes níveis da Névoa para ampliar a flexibilidade e eficácia na hora de decidir o destino de migração do conteúdo do usuário. Ao utilizar a metodologia de Tomada de Decisão com Múltiplos Atributos (*Multiple Attributes Decision Making* - MADM), o CMFog oferece uma estratégia de tomada de decisão flexível na qual diversos fatores podem ser considerados e ponderados adequadamente para realizar decisões de migração que possam atender às necessidades dos usuários.

Abstract

The digital revolution, driven by the widespread adoption of the Internet of Things (IoT) concept, has radically transformed society's interaction with the digital world. This new computing landscape has brought challenges and needs that have highlighted the limitations of the Cloud Computing paradigm. By decentralizing computational resources and migrating them to the network edge, the concept of Fog Computing emerges as an alternative to overcome the limitations of the Cloud, bringing computation closer to users and aiming to provide an infrastructure capable of meeting the demands for low latency and real-time interactions. However, the inherent mobility of users and edge sensors introduces complex challenges that need to be addressed for the Fog to guarantee low latency regardless of the user's location. In this context, this thesis proposes CMFog, a proactive content migration strategy in multi-level Fog infrastructure, with the aim of optimizing communication latency. CMFog integrates mobility prediction to dynamically adapt the infrastructure to user needs and seeks to explore the different levels of the Fog to enhance flexibility and effectiveness in deciding the migration destination of user content. By using the Multiple Attributes Decision Making (MADM) methodology, CMFog offers a flexible decision-making strategy in which various factors can be considered and appropriately weighted to make migration decisions that meet user needs.

Lista de Figuras

2.1 Classificação dos tipos de hypervisor - Bare Metal vs Hosted. (adaptado de [36]). 20
2.2 Arquitetura da Computação em Névoa de três camadas: dispositivos, Névoa e Nuvem (Adaptada de [50]). 28

4.1 Diagrama de componentes e comunicação do CMFog$_H$. 42
4.2 Etapas seguidas pelo algoritmo de construção da cadeia n-MMC em um cenário de exemplo. 45
4.3 Etapas seguidas pelo algoritmo de tomada de decisão do componente DeMADM em um cenário de exemplo. 48
4.4 Esquematização da interação da Política 1 com um cenário de mobilidade do usuário sem migração de máquina virtual. 50
4.5 Esquematização da interação da Política 1 e 2 com um cenário de mobilidade do usuário com migração de máquina virtual. 51
4.6 Representação dos aspectos considerados pelo *trigger* baseado na proximidade do usuário com os limites da área de cobertura de uma cloudlet. . . . 52
4.7 Representação dos aspectos considerados pelo *trigger* baseado no tempo estimado de permanência do usuário em uma cloudlet. 52
4.8 Diagrama de componentes e respectivas entidades que compõem o simulador iFogSim. 54
4.9 Diagrama das principais entidades que compõem o simulador MyiFogSim. . 55
4.10 Rotina do usuário utilizada para gerar registros de mobilidade para experimentos do CMFog$_H$. 56
4.11 Diagrama de sequência que ilustra o funcionamento do CMFog$_H$. 58
4.12 Exemplo da distribuição das cloudlets no espaço de simulação dos experimentos. 59
4.13 Gráficos com os resultados dos experimentos de simulação das métricas de latência e migração. 61
4.14 Gráficos com os resultados dos experimentos de simulação das métricas de tuplas geradas e tuplas perdidas. 62
4.15 Gráficos com os resultados dos experimentos de simulação das métricas de latência e migração e com cloudlet T3. 64

5.1 Esquematização da interação da Política 4 com um cenário de mobilidade do usuário com migração vertical. 68
5.2 Representação dos aspectos considerados pelo *trigger* utilizado no processo de decisão de migração vertical. 69
5.3 Diagrama de componentes e comunicação do CMFog$_V$. 70
5.4 Diagrama de sequência que ilustra o funcionamento do CMFog$_V$. 74

5.5 Rotina do usuários utilizada para gerar registros de mobilidade para experimentos do CMFog$_V$. 76
5.6 Gráficos com os resultados dos experimentos de simulação dos cenários com políticas P3 e Sem migração. 78
5.7 Gráficos com os resultados dos experimentos de simulação dos cenários com CMFog$_V$ e $W = 50$. 80
5.8 Gráficos com os resultados dos experimentos de simulação dos cenários com CMFog$_V$ e $W = 100$. 81
5.9 Gráficos com os resultados dos experimentos de simulação dos cenários com CMFog$_V$ e $W = 200$ ou $W = 300$. 83
5.10 Gráficos com os resultados dos experimentos de simulação dos cenários com CMFog$_V$ e $T = 5$. 85
5.11 Gráficos com os resultados dos experimentos de simulação dos cenários com CMFog$_V$ e $T = 10$ ou $T = 20$. 86

Lista de Tabelas

3.1 Tabela comparativa das principais características do trabalhos similares encontrado na literatura e do CMFog. 39

4.1 Símbolos utilizados no modelo de formalização do CMFog. 41
4.2 Valores dos parâmetros de conexões entre as entidades da simulação. . . . 53
4.3 Parâmetros de conexão das tecnologias de acesso. 53
4.4 Fatores e variáveis de respostas para construção do fatorial de experimentos de simulação do CMFog_H. 60
4.5 Configurações dos tipos de cloudlets utilizadas nos experimentos adicionais. 64

5.1 Valores dos parâmetros de conexões entre as entidades da simulação do CMFog_V. 72
5.2 Fatores e variáveis de respostas para construção do fatorial de experimentos de simulação dos cenários de controle (sem migração vertical). 75
5.3 Fatores e variáveis de respostas para construção do fatorial de experimentos de simulação do CMFog_V. 76

Sumário

1 Introdução **13**

2 Conceitos **17**
2.1 Virtualização . . . 17
2.1.1 Maquinas Virtuais e VMMs . . . 19
2.1.2 Vantagens e Desvantagens . . . 20
2.1.3 Containers . . . 21
2.1.4 Migração de Máquinas Virtuais . . . 22
2.2 Computação em Nuvem . . . 23
2.2.1 Modelos de serviço . . . 25
2.2.2 Modelos de implantação . . . 26
2.3 Computação em Névoa . . . 26
2.3.1 Arquitetura . . . 27
2.3.2 Características e Vantagens . . . 28
2.3.3 Desafios . . . 29
2.3.4 Aplicações . . . 30
2.4 Predição de Mobilidade . . . 31
2.5 Teoria da decisão . . . 32

3 Revisão de Literatura **35**

4 CMFog: Migração de conteúdo na Computação em Névoa **40**
4.1 Descrição do modelo . . . 41
4.1.1 Modelo de Aplicação de Névoa . . . 41
4.1.2 Modelo de Infraestrutura . . . 42
4.2 Componentes do CMFog$_H$ . . . 42
4.2.1 DeTrigger . . . 43
4.2.2 Predmobi . . . 43
4.2.3 DeMADM . . . 46
4.3 Políticas de Migração . . . 49
4.3.1 Política 1: Sem migração de conteúdo . . . 49
4.3.2 Políticas 2 e 3: Migração proativa de conteúdo com *trigger* . . . 50
4.4 Simulação . . . 52
4.4.1 A Infraestrutura . . . 53
4.4.2 MyiFogSim . . . 54
4.4.3 Histórico de mobilidade . . . 56
4.4.4 Métricas . . . 57
4.4.5 O experimento . . . 57
4.4.6 Definições de simulação . . . 58

4.5 Resultados . . . 60
4.5.1 Experimento base . . . 60
4.5.2 Variação de largura de banda das cloudlets . . . 63
4.5.3 Discussão . . . 65

5 CMFog$_V$: Névoa multi-nível **67**
5.1 Modelo e Política . . . 68
5.2 Componentes . . . 69
5.3 Simulação . . . 71
5.3.1 Métricas . . . 72
5.3.2 Experimentos . . . 73
5.3.3 Definições de simulação . . . 74
5.4 Resultados . . . 76
5.4.1 Migração horizontal: P3$_1$ e P3$_2$ . . . 77
5.4.2 CMFog$_V$ . . . 79
5.4.3 Discussão . . . 86

6 Conclusões **88**
6.1 Trabalhos Futuros . . . 90

Referências Bibliográficas **92**

Capítulo 1

Introdução

A revolução digital, iniciada nas últimas décadas do século XX, foi marcada pelo surgimento de tecnologias como a computação pessoal, a Internet e, mais recentemente, a Internet das Coisas (*Internet of Things* - IoT). Atualmente, a sociedade está imersa em uma transformação digital impulsionada pela IoT, que transformou profundamente a maneira como nossa sociedade interage com o mundo digital, afetando desde ambientes residenciais até setores complexos da industria. Com bilhões de dispositivos interconectados, a IoT estabelece um ecossistema complexo, desafiando as abordagens convencionais de computação. Projeções indicam que até 2030, o número de dispositivos IoT conectados à Internet poderá superar a marca de 32 bilhões. [65].

As raízes da IoT remontam ao final do século XX, quando a visão de objetos interconectados começou a se formar no cenário computacional. No entanto, a verdadeira explosão da IoT ocorreu nas últimas duas décadas, impulsionada pela miniaturização de componentes eletrônicos, desenvolvimento de redes de comunicação e aprimoramento dos sensores. A IoT constitui uma rede dinâmica de objetos físicos incorporados com sensores, software e conectividade, possibilitando a coleta, transmissão e recepção de dados. A inteligência distribuída desses dispositivos cria um ambiente propício para automação e tomada de decisões baseadas em dados obtidos em tempo real.

A IoT, ao viabilizar a conectividade ubíqua, estabelece novos padrões tanto para a interação entre humanos e máquinas quanto para as interações entre as próprias máquinas. A diversidade de dispositivos, desde eletrodomésticos inteligentes até sensores industriais e dispositivos vestíveis, introduziu uma nova dinâmica no cenário computacional. A necessidade de processar e analisar dados próximo aos usuários tornou-se imprescindível para atender aos requisitos de baixa latência, comunicação em tempo real e eficiência no uso de recursos. Surge, assim, a necessidade de uma abordagem computacional mais distribuída e eficiente, capaz de realizar processamento próximo à fonte de geração desses dados.

Antes da difusão do conceito da IoT, o cenário de computação era caracterizado por uma abordagem mais isolada e centralizada. A conectividade limitada e a falta de capacidade para a coleta massiva de dados restringiam as possibilidades de inovação. A computação estava centralizada em sistemas tradicionais, como a Computação em Nuvem, com interações humanas mais diretas e uma menor capacidade de resposta em tempo real. Embora a computação em nuvem tenha se consolidado como a espinha dorsal da

infraestrutura digital, transformando a estratégia de provisionamento de recursos e o modelo de negócios da computação, o paradigma apresenta diversas limitações diante dos desafios gerados pela IoT.

A latência, medida pelo atraso na transmissão de dados, destaca-se como um dos principais obstáculos enfrentados pela Computação em Nuvem. A arquitetura centralizada tradicional da Nuvem demanda que os dados dos dispositivos IoT sejam enviados para os data-centers, gerando um atraso inerente devido à distância física entre os dispositivos localizados na borda da rede e os data-centers, localizados no núcleo da rede. Esse atraso pode ser inviável em aplicações que exigem respostas quase instantâneas, como monitoramento de pacientes em tempo real ou controle de tráfego urbano.

Outro desafio enfrentado pela Nuvem nesse contexto é o congestionamento de rede, decorrente do grande volume de dados gerados pela IoT. Sensores continuamente geram informações, criando um fluxo constante de dados. A transmissão desse volume significativo de dados pela rede em direção à nuvem pode levar a congestionamentos, resultando em atrasos e interrupções nas comunicações. Essa sobrecarga não apenas prejudica o funcionamento das redes e da Internet como um todo, mas também contribui para a deterioração da latência.

Diante desse cenário, a Computação em Névoa emerge como uma extensão natural da nuvem em direção à borda da rede. Essa abordagem descentralizada permite a execução de tarefas críticas mais próximas dos pontos de origem dos dados, reduzindo a latência e aprimorando a eficiência na comunicação. A névoa, ao descentralizar o processamento e armazenamento de dados, oferece maior flexibilidade para atender de forma eficiente às demandas dinâmicas e heterogêneas de um novo cenário de computação. Cidades inteligentes, saúde conectada, veículos autônomos e realidade aumentada são apenas alguns exemplos de tipos de aplicações que exigem respostas instantâneas e comunicação eficiente entre dispositivos dispersos na borda da rede.

A Computação em Névoa é uma arquitetura que estende as capacidades da Computação em Nuvem até a borda da rede, mais próxima de dispositivos e fontes de dados. Essa abordagem permite o processamento, armazenamento e análise de dados mais próximos de sua geração, resultando em menor latência e maior eficiência na comunicação de rede [24]. Essencialmente, a Névoa atua como uma extensão descentralizada da nuvem, oferecendo maior proximidade aos dispositivos IoT, escalabilidade, adaptabilidade e suporte à mobilidade, tornando-se fundamental para que os requisitos de qualidade de aplicações executadas na borda da rede sejam atendidos.

No entanto, a transição para a Computação em Névoa traz consigo desafios significativos, especialmente no que diz respeito à gestão dos recursos computacionais e à otimização da infraestrutura. Um dos obstáculos reside na heterogeneidade e distribuição dos recursos na borda da rede, onde os nós de Névoa oferecem uma quantidade limitada de recursos que deve ser gerenciada de forma eficiente para atender aos diversos dispositivos conectados à rede. Além disso, os nós de Névoa são capazes de cobrir apenas uma área limitada, exigindo que a infraestrutura seja capaz de lidar com a constante mobilidade dos dispositivos do usuário ao realizar a migração de dados entre os nós para manter a proximidade entre conteúdo e o usuário.

O desafio de migração de conteúdo é uma área de estudo na computação que já se mos-

trou desafiadora no processo de ampla difusão da Computação em Nuvem. No entanto, o desafio de migração na Computação em Névoa apresenta características que aumentam a complexidade do processo, como a heterogeneidade do hardware que compõe os nós. A capacidade de transferir dados de maneira eficiente nesse ambiente dinâmico é essencial para viabilizar a promessa de um ambiente computacional capaz de lidar com a diversidade de dispositivos e a complexidade das aplicações modernas.

Em particular, na Computação em Névoa, no qual a proximidade entre conteúdo e os dispositivos possui grande importância, a predição de mobilidade surge como uma vertente para otimizar o processo de migração de conteúdo. A capacidade de antecipar os padrões de mobilidade dos dispositivos na borda permite uma migração mais eficiente. Dessa forma, a integração da predição de mobilidade ao desafio de migração não apenas aprimora a eficácia do processo, mas também contribui para a infraestrutura de Névoa ser capaz de se adaptar à dinâmica da mobilidade dos dispositivos e atender aos requisitos de qualidade do usuário. Técnicas como algoritmos de aprendizado de máquina, análise de padrões de deslocamento e modelos probabilísticos são alguns dos exemplos que são utilizados para antecipar o movimento de dispositivos na borda da rede.

Nesse contexto, esta tese propõe o CMFog (*Content Migration in Fog Computing*) [4, 5], uma estratégia de migração proativa de conteúdo em uma infraestrutura de Névoa multi-nível. O CMFog tem como objetivo gerenciar o processo de decisão para migração de conteúdo na borda da rede, buscando maximizar a proximidade entre conteúdo e usuário e, assim, ser capaz de oferecer uma baixa latência de comunicação. Uma das principais características do CMFog é que a estratégia é baseada em migrações proativas, ou seja, realizar migrações antecipadamente e reduzir o impacto que o processo de migração exerce na experiência do usuário. A predição de mobilidade é realizada por meio de uma variação da Cadeia de Markov para construir uma estrutura capaz de inferir futuras localizações do usuário. O processo de decisão do CMFog é baseado na metodologia de otimização de Tomada de Decisão com Múltiplos Atributos (*Multiple Attribute Decision Making* - MADM), ao utilizar variáveis como custo de migração, distância, predição de mobilidade e latência como entrada para identificar os melhores destinos de migração, de modo a minimizar a latência de comunicação dos dispositivos do usuário.

O objeto de estudo desta tese será apresentado em dois momentos, seguindo uma abordagem *bottom-up*. Inicialmente, uma versão de maior abstração será apresentada com o objetivo de investigar o comportamento com um subconjunto minimizado dos problemas, ao qual nos referiremos como CMFog_H. O CMFog_H terá a infraestrutura de Névoa formada por apenas um nível e, portanto, será capaz de realizar apenas migrações horizontais. O CMFog_V será uma versão que introduzirá a Névoa multi-nível, permitindo à estratégia realizar migrações verticais, ou seja, eventos de migração poderão ser realizados entre dois níveis diferentes. Para ambas as versões do CMFog, serão realizados experimentos utilizando um simulador para investigar o comportamento da estratégia em cenários que buscam representar o mundo real e compreender como a abordagem proposta nesta tese é capaz de tomar decisões que reduzam a latência de comunicação dos dispositivos na borda da rede.

Esta tese está organizada da seguinte forma: o Capítulo 2 apresenta os principais conceitos utilizados neste trabalho. Alguns dos trabalhos relacionados com o contexto da

tese são discutidos no Capítulo 3. O Capítulo 4 apresenta informações de modelagem, experimentos e resultados de uma primeira versão do CMFog, o CMFog_H. Já o capítulo 5 mostra a evolução para a versão CMFog_V, uma versão final do CMFog que explorar migrações horizontais e verticais. As conclusões finais são apresentadas no Capítulo 6.

Capítulo 2

Conceitos

2.1 Virtualização

Ao longo do avanço das tecnologias de hardware, a capacidade de processamento dos computadores expandiu exponencialmente. No entanto, as estratégias de gerenciamento de recursos não acompanharam efetivamente esse rápido desenvolvimento, levando à subutilização desses recursos. A virtualização surge como uma solução para esse problema, proporcionando novas possibilidades em diversos aspectos da computação [51].

Na década de 1960, a IBM deu os primeiros passos na introdução de sistemas de máquinas virtuais, marcando o início da virtualização. Nos anos seguintes, a virtualização ganhou destaque com o projeto M44/44X, que explorou o conceito de *time-sharing*. Esse projeto permitia a execução simultânea de várias máquinas virtuais, cada uma recebendo uma parcela de tempo para processamento. Essa iniciativa evoluiu para o desenvolvimento do renomado sistema IBM VM/370.

Na década de 1970, os pesquisadores Popek e Goldberg desempenharam um papel significativo na teoria da virtualização [42]. No entanto, devido à rápida evolução do hardware e ao aumento dos recursos computacionais disponíveis, a virtualização gradualmente perdeu destaque nos anos 1980.

Nos anos 1990 e 2000, a indústria tinha acesso a hardware mais potentes, e servidores dotados de uma grande quantidade recursos computacionais eram comercializados e empregados em diversas áreas da tecnologia. Contudo, a falta de estratégias eficientes de gerenciamento levou a um crescente nível de subutilização de recursos, resultando em aumentos nos custos e redução da produtividade. Nesse cenário, a virtualização ressurgiu como solução pela capacidade de consolidar servidores. Múltiplas máquinas virtuais podiam ser executadas em paralelo, de forma isolada e com a capacidade de utilizar recursos de maneira customizável, proporcionando um nível superior de consolidação dos servidores. Nos anos seguintes, a virtualização seria empregada como o pilar central para o paradigma da Computação em Nuvem, revolucionando a forma como os recursos computacionais eram disponibilizados.

A virtualização, de forma genérica, refere-se ao processo de criar uma versão virtual de algo. Esse conceito é aplicado em diversas áreas da Tecnologia da Informação (TI), abrangendo aplicações, armazenamento, processos e hardware. A virtualização de máquina, em particular, diz respeito à divisão ou agrupamento de recursos computacionais

para criar um ambiente no qual um ou mais sistemas operacionais possam ser executados de maneira isolada e transparente [9].

O hipervisor, também conhecido como Monitor de Máquina Virtual (*Virtual Machine Monitor* - VMM), desempenha o importante papel de gerenciar a camada de virtualização. Ele abstrai a complexidade do hardware, oferecendo um conjunto de recursos computacionais virtualizados. O hipervisor é fundamental para a virtualização de sistemas, permitindo a execução simultânea de vários sistemas operacionais em um único hardware físico. Além disso, fornece isolamento entre as máquinas virtuais, assegurando que as operações em uma VM não impactem as outras. Os hipervisores também são responsáveis pela gestão eficiente da alocação de recursos, como CPU, memória e armazenamento, entre as VMs, otimizando assim a utilização desses recursos.

As máquinas virtuais representam uma abstração completa de ambientes computacionais. Uma VM pode ser definida como uma implementação de software de um ambiente de computação, semelhante a uma máquina física. Cada ambiente possui seu próprio sistema operacional, operando de forma isolada e concorrente, possivelmente em paralelo com outros ambientes [54].

Uma das grandes vantagens da virtualização reside na abstração, onde a camada de virtualização oculta as complexidades do hardware físico, proporcionando um ambiente computacional mais flexível e simplificando o gerenciamento de recursos. Conforme esse conceito se difundiu, surgiram novas abordagens, e recentemente, os containers despontaram como uma alternativa mais leve e com um nível mais elevado de abstração. Enquanto as máquinas virtuais abstraem o hardware físico, os containers exploram a virtualização do sistema operacional, consumindo menos recursos e oferecendo um isolamento mais eficiente.

A adoção da virtualização traz consigo inúmeros benefícios, destacando-se a redução de custos de infraestrutura e manutenção, um gerenciamento mais eficaz de recursos computacionais e a facilidade na migração de cargas de trabalho. Podemos citar dois benefícios fundamentais de qualquer tecnologia de virtualização [48]:

- *Compartilhamento de recursos:* Em um ambiente virtualizado, as VMs compartilham recursos (memória, armazenamento, redes, etc.) do host hospedeiro, ao contrário de um ambiente sem virtualização, onde todos os recursos são dedicados aos programas em execução.

- *Isolamento lógico:* Um fator crucial para o sucesso da virtualização é a capacidade de isolamento entre as máquinas virtuais em execução. Apesar do compartilhamento de recursos, as VMs operam sem conhecimento umas das outras. É importante notar que, devido ao compartilhamento do hardware subjacente, o isolamento lógico não garante o isolamento de desempenho entre as máquinas virtuais.

A virtualização possui um papel fundamental nos principais paradigmas de gerenciamento de recursos, como a Computação em Nuvem e a Computação em Névoa. Em ambos os casos, a virtualização simplifica e aprimora a eficiência do gerenciamento de recursos, proporcionando flexibilidade no desenvolvimento de soluções para desafios em diversas áreas da computação. Este papel central da virtualização continua a evoluir, adaptando-se aos avanços tecnológicos e às demandas crescentes da computação moderna.

2.1.1 Maquinas Virtuais e VMMs

A aplicação do conceito de virtualização em diversos subsistemas computacionais, incluindo discos de armazenamento, dispositivos de entrada/saída e memória, possibilita a combinação desses diferentes recursos virtualizados para a criação de uma máquina virtual. Na década de 70, quando os estudos nessa área ganhavam destaque, os renomados pesquisadores Gerald Popek e Robert Goldberg [42] definiram uma máquina virtual da seguinte forma:

> *"Uma máquina virtual é vista como uma duplicata eficiente e isolada de uma máquina real, tal abstração é construída por um 'monitor de máquina virtual' (Virtual Machine Monitor - VMM)."*

As VMs são representações abstratas de ambientes computacionais completos, encapsulando um sistema operacional e aplicativos. Elas desempenham um papel fundamental na virtualização, permitindo a criação e execução de diversos sistemas operacionais em um único hardware físico. Uma VM consiste em uma imagem virtual que simula uma máquina física, com recursos dedicados, como CPU, memória RAM, armazenamento e interfaces de rede.

Os dois principais tipos de VMs são de sistema (ou servidor) e de processo (ou aplicativo). As VMs de sistema são projetadas para executar sistemas operacionais completos, proporcionando isolamento e independência entre diferentes sistemas operacionais em um mesmo servidor físico. Essa abordagem é amplamente utilizada em ambientes de servidores e data centers, possibilitando a consolidação de hardware, otimização de recursos e maior flexibilidade operacional.

Por outro lado, as VMs de processo são focadas na execução de aplicativos específicos, sem necessariamente encapsular um sistema operacional completo. Elas são comumente empregadas para isolar aplicativos e garantir ambientes de execução independentes, contribuindo para a segurança e confiabilidade de sistemas distribuídos.

O *Virtual Machine Monitor*, também conhecido como hipervisor, é um dos principais conceitos relacionados às máquinas virtuais. O hipervisor é uma camada de software localizada entre o sistema visitante e o hardware, abstraindo os recursos utilizados pelas máquinas virtuais [46]. O VMM é responsável por criar um ambiente onde os sistemas visitantes podem executar, gerenciando todos os aspectos de hardware, como CPU, dispositivos de entrada e saída, memória, entre outros. Essa camada de abstração é crucial para o funcionamento eficiente das máquinas virtuais, permitindo a execução simultânea de múltiplos sistemas operacionais em um único hardware físico e garantindo o isolamento entre eles.

Goldberg [16] propôs uma classificação dos VMMs com base na plataforma na qual o hipervisor é executado. Segundo essa classificação, as VMMs podem ser categorizadas em dois grupos:

- **Tipo I:** também conhecido como *bare-metal*, é implementado diretamente sobre o hardware físico do servidor. Atua como um sistema operacional independente e não requer um sistema operacional host. Esse tipo de hipervisor possui acesso direto aos

recursos do hardware, resultando em melhor desempenho. É comumente utilizado em ambientes de produção e data centers para virtualização de servidores. A Figura 2.1a ilustra como o VMM do Tipo I está organizada em relação ao hardware, sistemas operacionais e aplicativos.

- **Tipo II:** também conhecido como *hosted*, é executado dentro de um sistema operacional host, conforme ilustrado na Figura 2.1b. Ele funciona como um aplicativo dentro do sistema operacional já em execução. Embora tenha um maior nível de abstração, o Tipo 2 normalmente apresenta um desempenho inferior em comparação com o Tipo 1, pois depende do sistema operacional host para acessar os recursos de hardware. Este tipo de hipervisor é frequentemente utilizado em ambientes de desenvolvimento e testes.

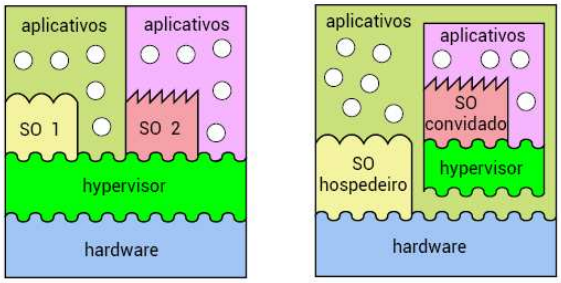


Figura 2.1: Classificação dos tipos de hypervisor - Bare Metal vs Hosted. (adaptado de [36]).

2.1.2 Vantagens e Desvantagens

A ascensão da virtualização representa uma alternativa ao convencional modelo de gestão de recursos computacionais, introduzindo novas perspectivas e desafios na área de Tecnologia da Informação. Os benefícios proporcionados pela virtualização abrangem desde a simplificação da infraestrutura de computação até o aprimoramento da qualidade de serviço. As vantagens da adoção da virtualização, conforme destacado por [48][64], englobam:

- Flexibilidade: a virtualização oferece um elevado grau de flexibilidade. Máquinas virtuais podem ser transferidas entre hosts, e suas especificações, como quantidade de RAM, armazenamento e vCPUs, podem ser modificadas sem a necessidade de reinicialização.

- Escalabilidade: com um esforço mínimo, uma infraestrutura de computação virtualizada pode expandir sua capacidade pela incorporação de novos hosts, sem interrupção dos serviços em execução.

- Consolidação: a ociosidade de recursos computacionais em grandes data centers se revelou como um desafio expressivo no gerenciamento de recursos. A virtualização possibilita o gerenciamento eficiente da carga de trabalho de um data center, otimizando a utilização de cada host.

- Encapsulamento: As máquinas virtuais operam de maneira isolada, sem conhecimento umas das outras, facilitando a migração entre hosts, desde que o host de destino forneça a mesma configuração do ambiente virtual anterior.

Contudo, apesar dessas vantagens, a virtualização também apresenta desvantagens e desafios que necessitam de consideração, visto que algumas delas introduzem compensações em relação aos benefícios obtidos:

- Overhead: a introdução de uma camada entre o hardware e o sistema operacional pode resultar em perda de desempenho em ambientes virtualizados. Todavia, pesquisas estão direcionadas a reduzir esse overhead, visando minimizar a diferença em relação a ambientes sem virtualização.

- Segurança: A segurança das máquinas virtuais está intrinsecamente ligada ao nível de segurança do sistema hospedeiro ou do VMM. Se a camada de controle do ambiente virtualizado for comprometida, todas as máquinas em execução sobre o sistema tornam-se vulneráveis.

- Ponto Único de Falha: A consolidação de servidores proporcionada pela virtualização pode concentrar vários serviços e máquinas virtuais em um único host, criando um Ponto Único de Falha. Logo, uma falha em um nó da infraestrutura de virtualização pode impactar diversos serviços.

2.1.3 Containers

A tecnologia de *containers* oferece virtualização no nível do sistema operacional, operando sobre o *kernel* para isolar e controlar recursos para um conjunto de processos [3]. Embora a virtualização ocorra em um nível diferente das máquinas virtuais, os *containers* herdam todas as propriedades e recursos oferecidos pelas VMs. A virtualização ao nível do sistema operacional permite uma redução significativa da sobrecarga de virtualização [62], realizando operações que demandam uma quantidade menor de recursos computacionais.

Os *containers* surgem como uma alternativa às populares máquinas virtuais, possibilitando o fornecimento de recursos virtualizados em ambientes com menor capacidade computacional do que os encontrados nos tradicionais datacenters. Além disso, os *containers* introduzem recursos interessantes para ambientes distribuídos, como: i) criação e inicialização rápidas de instâncias virtuais; ii) alta densidade de aplicações devido ao tamanho reduzido das imagens; e iii) isolamento entre instâncias em execução no mesmo host [57].

A mudança de paradigma promovida pela disseminação do conceito de mobilidade por meio da Computação em Névoa e IoT criou um cenário propício para a popularização das tecnologias de *containers*. Atualmente, várias ferramentas implementam as especificações da virtualização por *containers*, sendo as mais conhecidas o LXC e o Docker.

Assim como as máquinas virtuais, os *containers* também são representados por imagens. No entanto, as imagens são mais leves e contêm apenas binários e bibliotecas necessárias para sua execução, enquanto as imagens de VMs são grandes e incluem um sistema operacional completo.

2.1.4 Migração de Máquinas Virtuais

A migração de máquinas virtuais é um recurso essencial na administração de ambientes de virtualização, permitindo transferir uma VM de um host físico para outro, trazendo consigo benefícios como tolerância a falhas, portabilidade e otimização na utilização dos recursos [39]. Entretanto, a migração de VMs permanece um desafio complexo, podendo impactar negativamente no desempenho da infraestrutura e na experiência do usuário.

A difusão da virtualização impulsionou amplamente o uso de máquinas virtuais na estruturação de sistemas de computação. Como resultado, a gestão dessas máquinas tornou-se um dos principais desafios associados à virtualização. Por exemplo, a necessidade de movimentar VMs entre servidores devido a sobrecargas ou até mesmo falhas de hardware. Assim, a capacidade de migração de VMs entre servidores se mostrou uma recurso importante para permitir que o processo de gerenciamento de sistemas computacionais virtualizados fossem cada vez mais eficientes.

A migração de máquinas virtuais é uma prática essencial em ambientes virtualizados, permitindo transferir parte ou todos os dados de uma VM de um host físico para outro, trazendo consigo benefícios como tolerância a falhas, portabilidade e otimização na utilização dos recursos [39]. Apesar de fundamental para o gerenciamento de recursos virtualizados, esse recurso também introduz novos desafios. O processo de migração deve ser capaz de realizar uma cópia eficiente da VM, mas a replicação dos dados dos processos em execução não é trivial, especialmente quando estão alocados apenas na memória virtual ou no processador. Outro desafio relevante é a sobrecarga que o processo de migração pode impor à rede, uma vez que a transferência de dados e as mensagens necessárias para coordenar o processo podem sobrecarregar a rede e prejudicar o desempenho da infraestrutura computacional.

A migração de máquinas virtuais pode seguir duas abordagens: a *non-live migration* e a *live migration*. Na primeira, a VM é suspensa ou desligada antes do início do processo. Se a VM for apenas suspensa, os estados dos processos e da memória também devem ser copiados e transferidos, juntamente com as informações da VM, para o novo servidor de destino. O principal problema dessa abordagem é o tempo de indisponibilidade gerado pelo processo de migração [55]. Na segunda abordagem, a *live migration*, busca-se realizar a transferência da VM de maneira transparente para o usuário, com a menor interrupção possível do serviço. Ambas as abordagens têm usos distintos, sendo a *non-live migration* restrita a casos específicos nos quais os serviços ou dados das VMs não exigem um alto nível de disponibilidade.

A *live migration*, devido ao seu maior grau de complexidade para permitir uma migração com o mínimo de interrupção, utiliza métricas de desempenho para monitorar o processo, desde a preparação da VM no host de origem até a reinicialização no host de destino. Essas métricas são:

- Tempo de preparação: tempo entre o inicio da migração e o envio do estado do processador da VM para o host destino;
- Tempo de reinício: tempo entre reinício da VM e a finalização da migração;

- Páginas transferidas: quantidade de páginas de memória transferidas no processo de migração;
- Tempo de indisponibilidade: tempo que a máquina virtual está suspensa;
- Tempo total de migração: tempo entre o início de fim do processo de migração.

A abordagem *live migration* pode ser implementada por meio de duas técnicas: *pre-copy* e *pos-copy* [10]. A técnica *pre-copy* tem duas fases, a primeira conhecida como fase de *warm-up*, na qual são criadas cópias das páginas de memória da VM no host de origem e, em seguida, transferidas para o host de destino, sem interrupção da máquina virtual. Se houver alterações nas páginas de memória durante a transferência, novas cópias são feitas até que a diferença entre o host de origem e o destino esteja abaixo de um certo nível. A segunda fase, chamada de *stop-and-copy*, envolve a suspensão da VM no host de origem, transferência dos dados divergentes e reinicialização da VM no host de destino.

A técnica *pos-copy* suspende a execução da VM para a transferência das informações necessárias. Após a conclusão da transferência inicial e a obtenção do estado mínimo de execução no servidor de destino, as demais informações são migradas. A técnica *pre-copy* tem um baixo tempo de indisponibilidade, mas precisa transferir uma grande quantidade de páginas de memória para alcançar o estado mínimo necessário para a correta inicialização da VM. Já a *pos-copy* possui um maior tempo de indisponibilidade, mas realiza menos cópias e transferências de páginas de memória, suspendendo a VM no host de origem durante a fase de cópia. Cada técnica apresenta suas vantagens e desafios, sendo a escolha entre elas dependente dos requisitos específicos do ambiente e das aplicações envolvidas.

Devido à natureza mais leve e menos complexa dos *containers* em comparação com as VMs, o processo de migração também é mais simples. Se o *container* não estiver executando um serviço crítico, é possível encerrá-lo no host atual e apenas iniciar uma nova instância no destino desejado. Como o processo de instanciar um *container* é muito mais rápido e simples, essa operação não causa muitos problemas na disponibilidade do serviço. Contudo, como os *containers* são virtualizados em um nível de abstração mais elevado, é fundamental que os sistemas operacionais dos hosts envolvidos no processo possuam as mesmas bibliotecas e garantam a inexistência de conflitos para assegurar o seu funcionamento correto.

2.2 Computação em Nuvem

O surgimento e a popularização da Computação em Nuvem representaram uma verdadeira revolução no paradigma de gerenciamento e oferta de recursos computacionais no cenário de Tecnologia da Informação. Esse avanço tecnológico emergiu em resposta ao aumento da demanda por armazenamento e processamento de dados, que começou a superar a capacidade das infraestruturas tradicionais baseadas em servidores locais. O paradigma rapidamente ascendeu tornou-se um dos principais temas da comunidade científica de computação [12]. Antes da difusão da Computação em Nuvem, as organizações se viam diante de desafios complexos, que iam desde a manutenção de servidores até a gestão de

infraestrutura de rede, sem esquecer dos custos associados ao gerenciamento de recursos computacionais.

A Computação em Nuvem surge como uma solução para essas dificuldades, oferecendo um modelo flexível e escalável para a prestação de serviços de computação. Gigantes do setor, como Amazon, Google e Microsoft, lançaram plataformas que permitiam às organizações alugar recursos computacionais, armazenamento e serviços de rede. Em sua essência, a Computação em Nuvem disponibiliza recursos computacionais acessíveis a partir de qualquer dispositivo conectado à Internet. O modelo de negócios segue o princípio *"pay-as-you-go"*, onde os usuários consomem recursos computacionais como um serviço, pagando apenas pelo que efetivamente utilizam.

A diversidade de definições iniciais para o paradigma da Computação em Nuvem gerou um impasse na delimitação precisa do conceito e dos serviços abrangidos por ele. Apesar das várias contribuições de pesquisadores, a comunidade convergiu majoritariamente para a definição proposta pelo NIST (*National Institute of Standards and Technology*) [37]:

> *"A computação em nuvem é um modelo que possibilita o acesso pela rede de forma ubíqua, conveniente e sob demanda a um pool compartilhado de recursos computacionais configuráveis (por exemplo, redes, servidores, armazenamento, aplicações e serviços), que podem ser rapidamente provisionados e liberados com um esforço mínimo de gestão ou interação com o provedor de serviços."*

Essa definição destaca os principais atributos da Computação em Nuvem, enfatizando a acessibilidade e a capacidade de dimensionamento rápido dos recursos computacionais conforme a demanda. A noção de "*pool* compartilhado" evidencia a natureza centralizada do paradigma, enquanto a flexibilidade, expressada pelo termo "configuráveis", destaca a capacidade de ajuste dos recursos para atender às demandas específicas de cada usuário. Além disso, o "esforço mínimo de gestão ou interação" evidencia o auto nível de automação e a simplicidade no processo de gerenciamento de recurso.

A popularização da Computação em Nuvem coincidiu com um *boom* tecnológico em que vários segmentos, como dispositivos móveis, redes de alta velocidade e inteligência artificial. A combinação desses fatores, aliada à facilidade de acesso a recursos computacionais por meio da Nuvem, proporcionou um cenário propício para uma transformação no panorama da computação.

A computação em nuvem apresenta características que, em comparação com outras formas de provisionamento de recursos computacionais, possuem vantagens e desvantagens. É comum na literatura [44][13] encontrar comparações entre a nuvem e conceitos relacionados à computação em grade, o que pode tornar desafiador determinar a linha que separa ambos. Mell [37] destaca cinco características principais que diferenciam o modelo de computação em nuvem de qualquer outro modelo:

- Serviço sob demanda: a Nuvem é capaz pode provisionar recursos computacionais automaticamente, conforme a necessidade, sem exigir interação humana.

- Amplo acesso à rede: recursos estão disponíveis por meio da rede e podem ser acessados por plataformas de cliente sem exigir grande poder computacional.

- Disponibilização de conjunto de recursos: provedores computacionais oferecem um conjunto de recursos que podem não pertencer fisicamente à mesma infraestrutura, ou seja, recursos virtualizados que podem ser consumidos sob demanda.

- Elasticidade: recursos podem ser provisionados rapidamente, em alguns casos automaticamente, para adição ou liberação de recursos.

- Medição de serviço: os sistemas de computação em nuvem controlam, monitoram e otimizam automaticamente os recursos computacionais requisitados.

Os objetivos fundamentais da computação em nuvem, conforme identificados na literatura [13] são: aumento da confiabilidade, redução do custo de computação e aumento da flexibilidade. O aumento da confiabilidade está relacionado à condição de que os serviços fornecedores dos recursos são de responsabilidade do provedor de serviço, o que implica em um menor comprometimento com a manutenção dos recursos por parte dos usuários. A redução de custos é uma consequência da adotação do modelo de serviço *pay-as-you-go* que permite o fornecimento de recursos de computação de acordo com a demanda. A capacidade de escalar e desescalar recursos conforme a demanda é um fator crucial e fornece ao paradigma um alto nível de flexibilidade.

A virtualização é considerada o pilar central que sustenta o paradigma da Computação em Nuvem. Nesse contexto, a Nuvem pode ser visualizada como a virtualização fornecida como serviço, onde certas complexidades são abstraídas para oferecer recursos de maneira simplificada. Esse modelo possibilita que usuários com objetivos diversos, compartilhem recursos de um mesmo servidor físico, muitas vezes sem o conhecimento desse compartilhamento.

2.2.1 Modelos de serviço

Os ambientes de computação em nuvem oferecem diferentes modelos de serviço que definem a natureza dos serviços oferecidos aos usuários finais e influenciam diretamente na forma como as aplicações e os dados são gerenciados. Existem três modelos de serviço básicos: IaaS (Infraestrutura como Serviço), PaaS (Plataforma como Serviço) e SaaS (Software como Serviço).

- IaaS: o modelo é baseado em técnicas de virtualização de recursos computacionais. Ele fornece a base de infraestrutura necessária para os outros modelos, possibilitando maior controle sobre os recursos. O principal objetivo é simplificar o fornecimento de recursos, incluindo servidores, rede, switches, firewalls, balanceadores de carga, entre outros, na forma de serviço.

- PaaS: refere-se à entrega de uma plataforma de computação como serviço. Seu propósito é disponibilizar, manter, testar e escalar aplicações em um ambiente integrado de desenvolvimento e execução. Por meio do PaaS, os desenvolvedores podem direcionar os esforços para criação de aplicativos sem se preocupar com a complexidade da infraestrutura.

- SaaS: o modelo se beneficia da infraestrutura do IaaS e da plataforma do PaaS. No SaaS, as aplicações são disponibilizadas para uso por meio de interfaces de cliente, o que significa que não são executadas localmente. Assim, grande parte da complexidade de gerenciamento de recursos é transferida para a nuvem, onde os usuários consomem o serviço. Exemplos de aplicações mantidas no modelo SaaS incluem Dropbox, Google Docs, One Drive, entre outros.

2.2.2 Modelos de implantação

A computação em nuvem apresenta diversos modelos de implantação, e a escolha entre eles depende diretamente dos requisitos e objetivos do usuário que utilizará o serviço. Em relação ao modelo de implantação, uma nuvem pode ser classificada como pública, privada, híbrida e comunitária [37].

O modelo de Nuvem privada é gerenciada por uma única organização, geralmente com serviços de uso interno e total responsabilidade da própria organização. A construção de uma nuvem privada é justificada quando há necessidade de segurança ou privacidade nas informações contidas na nuvem. Por exemplo, em um cenário hipotético de uma empresa com vários departamentos, serviços podem ser disponibilizados exclusivamente para departamentos vinculados à organização.

Na nuvem pública, os serviços são oferecidos para para toda a Internet, compartilhando recursos entre usuários e diferentes organizações. Diversos fornecedores oferecem serviços na forma de infraestrutura, plataforma ou software. Algumas empresas notáveis nesse segmento incluem Amazon, Google, Microsoft e Salesforce [59].

A nuvem híbrida é adotada quando há a necessidade de uma nuvem combinar características de nuvens privadas e públicas. Isso pode ser motivado por questões de flexibilidade, redução de custos, entre outras. Em nuvens híbridas, alguns serviços menos críticos podem ser direcionados para a parte pública da nuvem, enquanto outros mais sensíveis são mantidos na parte privada. No entanto, a comunicação segura entre as duas partes da nuvem é uma questão importante e que requer estratégias cuidadosas para garantir a segurança das informações.

O modelo de nuvem comunitária caracteriza-se pelo compartilhamento de serviços entre diversas organizações que compartilham interesses comuns. A complexidade nesse modelo reside na necessidade de uma política de gerenciamento entre as organizações que compõem a nuvem

2.3 Computação em Névoa

A difusão do conceito de IoT e a crescente necessidade de uma infraestrutura computacional capaz de oferecer baixa latência mostraram algumas das limitações da Computação em Nuvem. A Computação em Névoa visa reduzir o volume de dados que trafegam na rede e criar uma infraestrutura capaz de reduzir a latência ao migrar parte dos recursos computacionais para a borda da rede.

Segurança, privacidade, consumo de energia, *offloading*, programabilidade e gerenciamento de recursos são alguns dos principais desafios da Computação em Névoa [7]. Os

desafios relacionados com a gestão de recursos e conteúdos ganham cada vez mais interesse da comunidade científica devido aos novos requisitos introduzidos pela crescente mobilidade dos usuários.

O surgimento da Computação em Névoa proporcionou novas possibilidades para cenários com requisitos de alta mobilidade e baixa latência. No entanto, o processo de decidir quando e onde os recursos da Névoa devem ser alocados ou migrados de acordo com a localização futura do usuário não é uma tarefa trivial.

A Computação em Névoa representa uma arquitetura descentralizada que aloca recursos computacionais na borda da rede, nas proximidades de dispositivos de usuários e sensores. Proposta inicialmente pela Cisco como uma alternativa à arquitetura centralizada da Computação em Nuvem, essa abordagem ganhou destaque com a popularização da Internet das Coisas e o surgimento de aplicações que demandam tempos de resposta reduzidos. A crescente demanda por baixa latência e a necessidade de lidar com volumes substanciais de dados evidenciaram as limitações da Nuvem, impulsionando a descentralização e distribuição de recursos computacionais em locais geograficamente próximos aos usuários finais.

Embora a Computação em Névoa tenha sido inicialmente concebida pela Cisco, diferentes perspectivas dos pesquisadores e da indústria resultaram em várias definições. De forma geral, há uma notável convergência entre essas definições, destacando a forte relação do conceito com a Nuvem e a alocação de recursos computacionais próximos aos dispositivos do usuário.

Atualmente, a definição proposta pelo NIST tem sido amplamente adotada. Segundo o instituto, a Computação em Névoa é uma arquitetura que estende as capacidades da Computação em Nuvem até a borda da rede, mais próxima de dispositivos e fontes de dados. Essa abordagem permite o processamento, armazenamento e análise de dados mais próximos de onde são gerados, resultando em menor latência e maior eficiência na comunicação de rede [24].

2.3.1 Arquitetura

Uma das características mais notáveis da Computação em Névoa é o alto nível de distribuição dos recursos computacionais, refletido diretamente em sua infraestrutura. A Figura 2.2 apresenta a arquitetura de Névoa, conforme definido pelo NIST, o paradigma pode ser organizado em três camadas:

- Camada de dispositivos: situada na base da arquitetura, esta camada consiste em um conjunto de dispositivos de usuário, incluindo sensores, smartphones, carros e computadores. Esses dispositivos atuam como pontos de coleta de informações e geram requisições que podem ser encaminhadas para as camadas superiores.

- Camada de Névoa: composta principalmente por hardware commodities, a camada lida com uma quantidade limitada de recursos computacionais e também com um alto nível de heterogeneidade. Os recursos computacionais são organizados em nós de Névoa, responsáveis por processar e responder requisições. Quando uma requi-

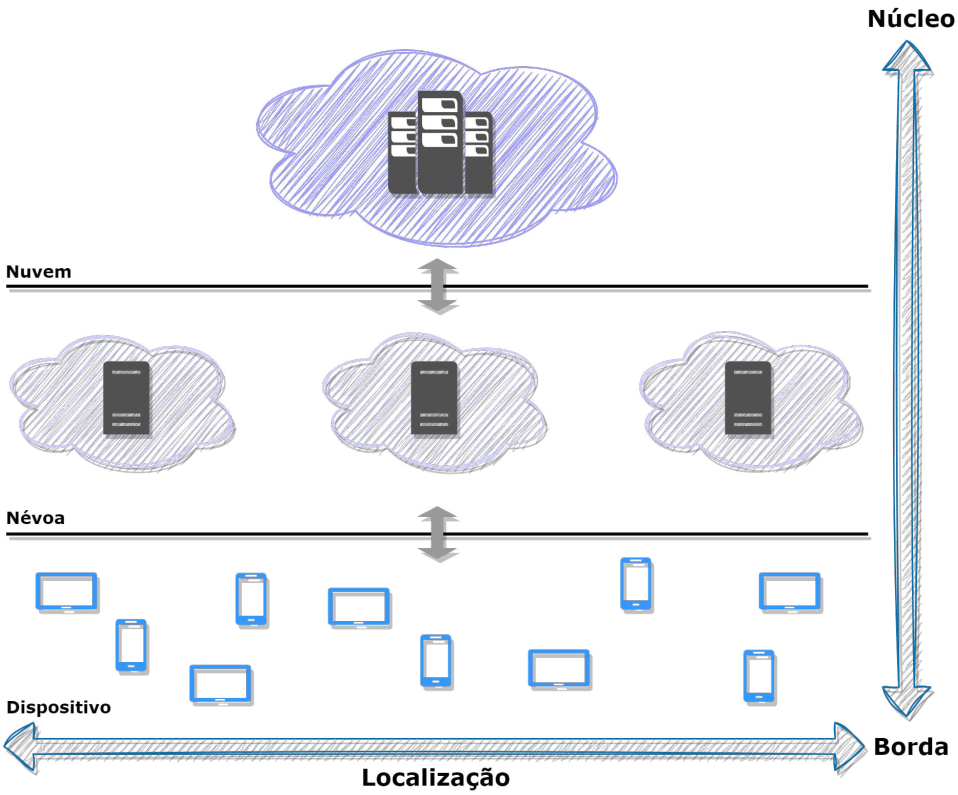


Figura 2.2: Arquitetura da Computação em Névoa de três camadas: dispositivos, Névoa e Nuvem (Adaptada de [50]).

sição ultrapassa a capacidade do nó de Névoa, ela pode ser encaminhada para a camada de Nuvem ao custo de uma maior latência de comunicação.

- Camada de Nuvem: corresponde a camada superior da arquitetura e é formada por data centers centralizados. Esta camada recebe requisições que não foram atendidas pela camada de Névoa, geralmente relacionadas a tarefas complexas que demandam elevado poder de processamento ou armazenamento.

Uma peculiaridade notável da camada de Névoa é a possibilidade de subdivisão em diferentes níveis. Os níveis inferiores consistem em nós de Névoa com menor área de cobertura e poucos recursos computacionais mas com capacidade de oferecer menores tempos de resposta. Em contrapartida, os níveis superiores são compostos por nós de Névoa com maior área de cobertura e com mais recursos, embora isso venha acompanhado de maiores tempos de resposta. A adoção da arquitetura de Névoa multi-nível proporciona à infraestrutura uma maior flexibilidade na alocação de recursos e, consequentemente, uma maior capacidade de adaptação às demandas variadas e dinâmicas dos usuários.

2.3.2 Características e Vantagens

A Computação em Névoa representa uma inovação no cenário da computação, propondo a descentralização de recursos computacionais ao aproximá-los da borda da rede. Esta abordagem visa atender às demandas específicas das aplicações modernas. Entre as principais características da Computação em Névoa podemos citar [22, 20, 15]:

- Baixa Latência: a proximidade geográfica dos recursos da Computação em Névoa com a borda da rede é fundamental para oferecer uma baixa latência de comunicação. Essa característica se revela essencial para aplicações como realidade virtual e saúde, proporcionando uma experiência responsiva e eficiente.

- Suporte à Mobilidade: uma característica importante da Névoa é a capacidade de migração de conteúdo entre os nós da camada, permitindo que aplicações mantenham a proximidade com os dispositivos de usuários. Esse recurso proporciona a possibilidade de atender os requisitos de qualidade mesmo durante o deslocamento dos usuários.

- Escalabilidade: a Computação em Névoa exibe notável escalabilidade, ajustando-se às variações na demanda dos dispositivos da borda da rede. O paradigma é capaz de fornecer recursos adicionais por meio da comunicação eficiente entre nós de Névoa ou, em último caso, com a Nuvem.

- Heterogeneidade: os nós de Névoa apresentam uma composição heterogênea de recursos computacionais virtualizados, provenientes de diversos hardwares na borda da rede. Essa diversidade permite a entrega de recursos distribuídos e adaptáveis a diferentes necessidades.

- Distribuição Geográfica: ao contrário da Nuvem, centralizada em data centers, a Computação em Névoa distribui recursos por toda a borda da rede. Essa distribuição geográfica confere flexibilidade no processo de alocação de recursos para aplicações e serviços, uma característica fundamental para ambientes que necessitam oferecer baixos níveis de latência.

2.3.3 Desafios

O paradigma da Computação em Névoa representa uma alternativa para superar as limitações identificadas na Computação em Nuvem, visando fornecer uma infraestrutura computacional capaz de atender às exigências das novas aplicações e dispositivos. No entanto, esse cenário não está isento de desafios que precisam ser cuidadosamente abordados para garantir o sucesso do paradigma em atender as demandas do atual cenário de computação. Podemos citar os seguintes desafios:

- Gerenciamento de recursos: a Névoa é composta principalmente por hardware commodities, o que limita a quantidade de recursos computacionais que um nó pode oferecer. Essa característica exige o desenvolvimento de estratégias eficazes para o gerenciamento e alocação de recursos de forma otimizada, atendendo às demandas dos dispositivos do usuário. Um outro fator importante é a dinâmica de conexão e desconexão contínua dos dispositivos, o que se apresenta como um grande desafio no processo de predição do consumo de recursos. Estratégias com alta adaptabilidade são essenciais para lidar com a natureza dinâmica e heterogênea da Névoa.

- Energia: além de recursos limitados, os nós de Névoa também precisam lidar com quantidade limitada de energia. Estratégias eficientes de gerenciamento de energia

são cruciais para garantir que a Névoa possa atender às demandas dos dispositivos, estabelecendo um equilíbrio entre desempenho e o consumo de energia.

- Privacidade e segurança: a natureza distribuída e heterogênea da Névoa aumenta consideravelmente a complexidade dos desafios relacionados com a segurança do paradigma. A diversidade de dispositivos e a frequência das trocas de mensagens são fatores que aumentam a vulnerabilidade, exigindo estratégias de segurança para mitigar os riscos. A Computação em Névoa herda e amplifica desafios tradicionais de privacidade e segurança encontrados na Computação em Nuvem, exigindo novas abordagens para proteger dados e comunicações dos usuários.

- Migração de conteúdo e serviços: atualmente uma das principais características dos dispositivos de usuário é o alto nível de mobilidade, o que exige que a infraestrutura de Névoa seja capaz de realizar migrações de conteúdo e serviço de forma eficaz, garantindo uma baixa latência ao maximizar a proximidade do serviço com o usuário. A eficácia da migração de conteúdo depende crucialmente de algoritmos preditivos capazes de antecipar os padrões de mobilidade dos usuários. Estratégias de aprendizado de máquina e algoritmos preditivos são essenciais para identificar destinos de migração próximos da localização atual do usuário, garantindo uma experiência ubíqua e que atenda aos requisitos de Qualidade de Serviço (*Quality of Service* - QoS).

2.3.4 Aplicações

A Computação em Névoa emerge como uma infraestrutura essencial para uma variedade de aplicações que exigem baixa latência de comunicação. Algumas dessas aplicações incluem:

- Cidades Inteligentes: cidades inteligentes são caracterizadas por uma extensa rede de sensores e aplicações de tempo real, demandando respostas quase instantâneas para garantir a ubiquidade ao usuário. A Névoa desempenha um papel crucial ao processar dados gerados pelos sensores na borda da rede, assegurando baixa latência e prevenindo congestionamentos no núcleo da rede.

- *Healthcare*: a Computação em Névoa possibilita o monitoramento em tempo real de indicadores de saúde e sinais vitais, tanto em ambientes hospitalares quanto em residências. A capacidade de reagir rapidamente a situações críticas é aprimorada, proporcionando eficácia no atendimento médico. Além disso, a Névoa abre novas possibilidades para o monitoramento contínuo de pacientes.

- Realidade Aumentada: a Névoa é essencial para aplicações de realidade aumentada, pois processa dados em nós próximos aos dispositivos. Esse processamento localizado reduz a latência, resultando em experiências mais imersivas. As aplicações abrangem diversas áreas, como entretenimento, treinamento, etc.

Essas aplicações destacam como a Computação em Névoa, ao oferecer uma infraestrutura descentralizada, pode não apenas atender às demandas específicas dessas áreas, mas

também impulsionar inovações e avanços em áreas importantes para o desenvolvimento tecnológico e social.

2.4 Predição de Mobilidade

A predição de mobilidade desempenha um importante papel na eficiência de sistemas urbanos inteligentes, oferecendo informações importantes para otimizar o planejamento urbano, melhorar o gerenciamento de tráfego ou facilitar a mobilidade de maneira sustentável. Dentro do cenário de computação, o estudo da mobilidade humana é uma área de pesquisa que vem ganhando cada vez mais importância. As pesquisas nessa área visam identificar padrões complexos de mobilidade, que se manifestam em ações recorrentes com intervalos de tempo constantes [69]. Esses movimentos são influenciados por uma gama de fatores, como contextos sociais, econômicos e culturais, tornando os modelos de mobilidade essenciais para compreender comportamentos em constante evolução.

Na literatura, diversos modelos foram propostos para compreender e representar a dinâmica da mobilidade humana. Abordagens baseadas em modelos de difusão, derivados de princípios químicos, fornecem uma representação eficaz da propagação de comportamentos e padrões de mobilidade em ambientes urbanos. A utilização de Redes de Mobilidade, que representam a cidade como uma complexa rede de interações espaciais, também se destaca na revelação de padrões de deslocamento e interconexões. Esses modelos de rede de mobilidade consideram os pontos de interesse como nós e as conexões entre eles como caminhos, proporcionando um melhor entendimento da dinâmica de funcionamento das atividades urbanas.

A mobilidade é caracterizada por padrões complexos e dinâmicos, variando ao longo do tempo e em diferentes localidades. A integração de dados de diversas fontes, como pessoas, sensores, dados de redes sociais aumenta a complexidade da análise. Fatores imprevisíveis, como eventos especiais e mudanças nas condições meteorológicas, contribuem para a incerteza na predição. A análise desses dados vai além da mera identificação de locais frequentados, proporcionando uma compreensão profunda de hábitos individuais e padrões de deslocamento. Técnicas avançadas, como aprendizado de máquina e análises preditivas, são cada vez mais empregadas para extrair informações do histórico de mobilidade dos usuários. [11, 58, 29].

Uma das áreas de pesquisa que a Computação em Névoa tem se concentrado corresponde as estratégias de migração proativa de conteúdo. Essa abordagem, aliada a técnicas sofisticadas de previsão de mobilidade, otimiza não apenas a entrega de serviços em tempo real, mas também antecipa as necessidades do usuário com base em seu histórico de mobilidade. Alguns do modelos bastante utilizados na predição de mobilidade são [26, 52, 67]:

- *Deep Learning*: demonstra eficácia na captura de padrões complexos e não lineares em dados de mobilidade urbana. A capacidade de representação hierárquica dessas redes permite a extração de características significativas, aprimorando a precisão das predições.

- Séries temporais: técnicas como modelos ARIMA (*AutoRegressive Integrated Moving Average*) e SARIMA (*Seasonal ARIMA*) oferecem uma visão aprofundada das tendências, sazonalidades e flutuações temporais nos dados de mobilidade. Essa abordagem é particularmente relevante para capturar variações temporais em padrões de deslocamento urbano.

- Redes Bayesianas: em certos momentos, a predição de mobilidade em ambientes urbanos exige uma abordagem mais probabilística. As redes Bayesianas oferecem uma representação formal de dependências entre variáveis, sendo aplicadas para modelar a complexidade e a incerteza associadas aos padrões de mobilidade.

- Cadeia de Markov: são modelos estocásticos que descrevem sequências de eventos, nos quais a probabilidade de transição entre estados depende exclusivamente do estado atual. No contexto da predição de mobilidade, os estados podem corresponder aos Pontos de Interesse (*Points of Interest* - POI), enquanto as transições refletem a probabilidade de um dispositivo mover-se entre dois POIs. A facilidade de associar o problema de predição com as Cadeias de Markov torna essa técnica uma das mais utilizadas no desafio da predição de mobilidade.

O esforço conjunto das áreas de estudo de dados de mobilidade, algoritmos preditivos e Computação em Névoa evidencia a complexidade no entendimento e exploração dos padrões de comportamento humano. Um outro fator de complicação é a necessidade de garantir a transparência nas práticas de coleta de dados para que a privacidade dos usuários seja garantida. Nesse cenário, as abordagens de análise de mobilidade precisam evoluir constantemente para lidar com o volume e a variedade dos dados disponíveis, garantindo que as aplicações relacionadas à mobilidade dos usuários atendam às demandas de latência e requisitos de qualidade de serviço.

2.5 Teoria da decisão

A Teoria da Decisão, uma área de estudo interdisciplinar, se concentra na compreensão dos desafios referentes ao processo de tomada de decisão, particularmente em contextos de incerteza e objetivos conflitantes [30, 40]. A necessidade de fazer escolhas é uma constante na experiência humana, os fundamentos da Teoria da Decisão remontam ao início do século XX, com notáveis contribuições de pesquisadores como Leonard J. Savage e John von Neumann.

O objetivo da Teoria da Decisão é fornecer uma base conceitual para analisar, modelar e otimizar escolhas em diversas áreas da vida. A análise de decisões pode abordar aspectos normativos, investigando como as decisões deveriam ser tomadas considerando fatores racionais, e descritivos, observando como as decisões são realmente tomadas, muitas vezes influenciadas por fatores não racionais.

Ao longo dos anos, diversos modelos de tomada de decisão foram propostos. O modelo da teoria da utilidade, por exemplo, oferece uma abordagem normativa para escolhas em condições de incerteza, quantificando as preferências de quem toma a decisão. Em outra linha, o modelo de decisão multicritério, proposto por H. A. Simon, lida com escolhas

envolvendo múltiplos critérios, realizando uma avaliação ponderada e classificando as alternativas.

Na Ciência da Computação, a Teoria da Decisão ganha destaque, sendo incorporada em sistemas de inteligência artificial e aprendizado de máquina para automatizar processos de tomada de decisão. A Tomada de Decisão com Múltiplos Critérios (MCDM), uma extensão da Teoria da Decisão, é especialmente relevante, uma vez que sistemas computacionais frequentemente precisam lidar com trade-offs, fazendo escolhas entre alternativas com critérios variados e, por vezes, conflitantes.

O processo de Tomada de Decisão com Múltiplos Critérios pode ser categorizado em: Tomada de Decisão com Múltiplos Atributos (*Multiple Attribute Decision Making* - MADM) e Tomada de Decisão com Múltiplos Objetivos (*Multiple Objective Decision Making* - MODM)[61]. A MADM é uma abordagem com foco em situações em que a decisão implica na escolha entre alternativas com base em múltiplos atributos. Nesse contexto, o método visa avaliar e classificar as alternativas considerando os diferentes atributos relevantes. Essa abordagem é particularmente útil quando se busca uma análise comparativa entre as opções disponíveis com base em critérios específicos.

Por outro lado, a MODM lida com cenários nos quais há a necessidade de otimizar simultaneamente múltiplos objetivos, frequentemente em conflito uns com os outros. Ao contrário do MADM, que se concentra na avaliação comparativa, o MODM busca encontrar soluções que representem um equilíbrio entre diversos objetivos, considerando os *trade-offs* entre eles.

A Tomada de Decisão com Múltiplos Atributos (MADM) é uma metodologia multidisciplinar para resolver problemas de tomada de decisão em diversas áreas do conhecimento, como ciências sociais, engenharia e economia [19]. No domínio da computação, a MADM destaca-se como uma solução eficaz para uma diversidade de problemas considerados *NP-Hard*, incluindo seleção de interface wireless, decisão de *handover* e seleção dinâmica de *web services*.

As estratégias de tomada de decisão fundamentadas na abordagem MADM têm como objetivo principal ordenar um conjunto de alternativas com base no conjunto de atributos previamente definidos para o problema em questão. Estes atributos podem ser agrupados em duas categorias principais: benefício e custo. O método busca atribuir pontuações a cada uma das alternativas, sendo as mais bem avaliadas aquelas em que as pontuações dos atributos de benefício são maximizadas e as dos atributos de custo são minimizadas. A modelagem do problema de decisão determina como esses atributos influenciam a pontuação final, podendo variar de acordo com a natureza do problema.

A aplicação prática da MADM na computação proporciona soluções eficientes para problemas complexos nos quais a análise de múltiplos critérios é essencial. A abordagem não apenas simplifica a complexidade computacional desses desafios, mas também oferece uma estrutura sistemática para a avaliação e escolha de alternativas em ambientes nos quais a otimização de benefícios e a minimização de custos são determinantes para solução do problema.

Na literatura, uma variedade de algoritmos MADM é encontrada, cada um adotando abordagens distintas para resolver problemas de tomada de decisão com múltiplos critérios. Novos algoritmos continuam surgindo, adaptados para uma ampla gama de contex-

tos e problemas. Em geral, esses algoritmos diferem na forma como lidam com critérios e alternativas, além de como calculam a distância entre as soluções e a solução ideal. Por exemplo, o algoritmo TOPSIS [23] (*Technique for Order of Preference by Similarity to Ideal Solution*) emprega a distância euclidiana, enquanto o DiA [60] (*Dominance in Analysis)* baseia-se na distância de Manhattan para determinar a proximidade com a solução ideal. Já o algoritmo VIKOR [53] (*VlseKriterijumska Optimizacija I Kompromisno Resenje*) utiliza um índice de similaridade para classificar as alternativas, buscando equilibrar custos e benefícios em cada alternativa.

Capítulo 3

Revisão de Literatura

A popularização da Internet das Coisas e a difusão de aplicações que demandam interação em tempo real revelaram algumas das limitações da Computação em Nuvem, evidenciando a necessidade de soluções ágeis e próximas aos dispositivos do usuário na borda da rede. A nuvem, embora eficaz para armazenamento e processamento em grande escala, muitas vezes enfrenta desafios ao lidar com a crescente quantidade de dados gerados por dispositivos IoT e as demandas por baixa latência dessas aplicações interativas.

A computação em névoa é uma arquitetura que migra parte da complexidade da rede para a borda, aproximando-os das fontes de dados. Essa abordagem visa superar as fragilidades da nuvem ao permitir o processamento, armazenamento e análise de dados de maneira mais próxima de sua origem, resultando em menor latência e maior eficiência na comunicação de rede [31, 47, 34, 38]. A capacidade de migrar conteúdo na Névoa torna-se fundamental para otimizar o desempenho, atendendo à expectativa de baixa latência exigida por aplicações sensíveis ao tempo e dispositivos IoT interativos. A migração de conteúdo na névoa emerge como uma ferramenta estratégica para oferecer uma experiência de usuário mais responsiva e eficaz no ambiente dinâmico da Computação em Névoa.

Na literatura, podemos encontrar diversos tipos de estratégias de migração que exploram as mais variadas técnicas e abordagens para transferência de conteúdo, seja de dados ou aplicações, na borda da rede. Essas estratégias visam otimizar o desempenho da computação em Névoa, considerando a diversidade de dispositivos, a dinamicidade do ambiente e a demanda por baixa latência. Puliafito et al. [43] apresentam uma arquitetura baseada no protocolo *Message Queue Telemetry Transport* (MQTT) para migrar serviços na Computação em Névoa, buscando aprimorar a QoS. Goudarzi et al. [17] propõem uma estratégia para gerenciar migrações distribuídas de aplicativos IoT na borda da rede. A decisão de migração é tomada em um modelo de custo ponderado, considerando tempo de resposta do aplicativo e consumo de energia do dispositivo IoT.

O trabalho de Alqam et. al [2] aborda desafios na migração de VMs destacando a heterogeneidade dos dispositivos e a importância da localização dos nós de Névoa. O trabalho propõe um sistema chamado EVM_MIG, baseado em Aprendizado por Reforço (*Reinforcement Learning* - RL), para otimizar a migração de VMs. O EVM_MIG considera a tipologia, função e localização dos nós de névoa, visando reduzir significativamente a latência de aplicações críticas no tempo. O algoritmo EVM_MIG utiliza RL para decidir sobre a migração de VMs, levando em consideração carga, distância do usuário e tipo

de nó de Névoa.

Roig et al. [45] apresenta um modelo algorítmico e algébrico para a migração de máquinas virtuais em uma topologia de datacenter *Leaf and Spine*, adequado para ambientes de Computação de Névoa onde dispositivos do usuário podem se mover. O estudo estende pesquisas anteriores relacionadas a uma arquitetura Fat Tree, focando na obtenção de um design mais eficiente para as demandas atuais nos datacenters. Ao considerar todas as trajetórias possíveis entre dois hosts, o modelo aborda cenários de hosts a uma ou duas conexões de distância, destacando a importância dos *switches Leaf* e *Spine*. Algoritmos são propostos para identificar dispositivos e portas nas trajetórias redundantes entre hosts.

No trabalho de He et al. [21] é proposto um framework para o planejamento e agendamento de migração múltipla em Computação de borda, abordando competição por recursos e restrições temporais. Utilizando algoritmos baseados em Conjuntos Independentes Máximos Iterativos (MIS) e Rede Definida por Software (*Sofware Defined Network* - SDN), o framework busca otimizar a eficiência em larga escala, simplificando a complexidade do problema com um grafo de dependência de recursos.

Devido à natureza dinâmica e imprevisível dos dispositivos na borda da rede, estratégias reativas frequentemente enfrentam dificuldades para realizar migrações dentro de um intervalo de tempo que mantenha a relevância da ação tomada. Por esse motivo, as estratégias proativas têm ganhado destaque, buscando antecipar momentos em que as migrações serão necessárias. Ao iniciar o processo de tomada de decisão com antecedência, essas estratégias podem realizar migrações de maneira mais eficaz, garantindo que os requisitos de qualidade de serviço sejam atendidos mesmo em cenários desafiadores e complexos.

O trabalho de Khaleel et al. [28] propõe um algoritmo de alocação de VMs em ambientes de Névoa chamado "E-PRWA" que busca otimizar a relação desempenho-energia. O modelo proposto possui quatro estágios: (a) detecção de nós sobrecarregados ou subutilizados; (b) seleção de VMs para migração; (c) desligamento de nós subutilizados selecionados; (d) implantação das VMs migradas com consideração de PPR, overhead de latência e custo computacional. O foco é reduzir o custo computacional da migração, abordando eficientemente a utilização de recursos na infraestrutura dos data-centers. O algoritmo E-PRWA busca otimizar o desempenho dos nós e reduzir o consumo de energia, contribuindo para a eficiência energética sem comprometer a qualidade do serviço.

Kaur et al. [27] propõe o modelo Proativo de Suporte à Mobilidade (PROMO) para o remanejamento de tarefas na Computação em Névoa, visando melhorar a QoS considerando a mobilidade do usuário. O modelo utiliza um mecanismo de classificação de múltiplos atributos e integra-se ao MobFogSim para validação, alcançando melhorias significativas em atraso e tempo de inatividade de migração/transição. Martin et al. [35] desenvolvem o framework MAMF, abordando a migração de contêineres que executam módulos de aplicativos do usuário em ambientes de computação em névoa. Utilizando o ciclo MAPE (Monitorar-Analisar-Planejar-Executar) e Algoritmo Genético, o MAMF garante a migração eficiente de conteiners com base na mobilidade do usuário, atendendo aos requisitos de QoS.

No trabalho de Tuli et. al [63] é apresentado o PreGAN, um modelo de inteligência artificial que utiliza uma Rede Adversária Generativa (GAN) para realizar decisões de

migração baseada na detecção proativa de falhas nos hosts. O PreGAN combina co-simulações com uma GAN para aprender um classificador de anomalias, proporcionando detecção, diagnóstico e classificação mais precisos de falhas. Além disso, o PreGAN utiliza modelos de *Graph Attention Network* e *Gated Recurrent Units* para extração de características para otimizar as decisões de migração, resultando em melhorias no consumo de energia, tempo de resposta e violações de SLA. Afrasiabi et al. [1] propõe uma estratégia de migração de componentes em um ambiente híbrido de Nuvem e Névoa baseado em Virtualização das Funções de Rede (NFV), considerando a mobilidade de usuários e nós de Névoa. Utilizando modelos de mobilidade específicos para cada elemento, Gauss-Markov para nós de névoa e passeio aleatório (*Random Walking*) para dispositivos de usuários, o problema é formulado matematicamente para minimizar a função ponderada de atraso e custo da aplicação. Diante da complexidade resultante da mobilidade simultânea, é apresentada uma abordagem de Aprendizado Profundo por Reforço para decisões rápidas e eficientes de migração de componentes da aplicação.

Taghizadeh et al. [56] apresentam um mecanismo de alocação de réplicas de dados baseado em otimização biogeográfica (BBO) para aplicações intensivas em dados na Névoa, propondo também um framework autônomo para a transferência de réplicas de dados entre dispositivos IoT e nós de névoa de armazenamento. Focado na resolução do desafio da alocação de réplicas de dados em meio à mobilidade de dispositivos IoT e nós de Névoa, o estudo aborda o problema como NP-difícil, optando por uma abordagem de aprendizado profundo para decisões rápidas. A estrutura proposta segue uma arquitetura de Névoa de três camadas, utilizando BBO como técnica de otimização para encontrar soluções eficientes em termos de custo, latência, confiabilidade e disponibilidade.

Explorando a arquitetura de Névoa com múltiplos níveis, Santos et al. [49] propõem o Fog4Video, um mecanismo orquestrador de conteúdo para serviços de Video-on-Demand (VoD) em ambientes de Computação em Névoa multi-nível. O Fog4Video visa aprimorar a Qualidade de Experiência (QoE) selecionando nós de Névoa apropriados com base em métricas de rede, nó de Névoa e usuário.

Em [6] Ashraf et al. aborda desafios na execução de aplicações baseadas em microsserviços em uma infraestrutura de Névoa de múltiplos níveis, visando atender aos requisitos de latência das aplicações. A proposta de scheduling, denominada *Dynamic Microservice Application Placement* (DMAP), otimiza a latência total da execução da aplicação ao identificar migrações adequadas de microsserviços dentro de uma infraestrutura de Névoa. O DMAP foca na otimização da latência e utilização de recursos ao alocar microsserviços em dispositivos IoT próximos, como laptops e smartphones, além de nós de Névoa e servidores em Nuvem. Ao lidar com o scheduling de microsserviços em uma infraestrutura de Névoa de dois níveis, a pesquisa busca destacar a escalabilidade das técnicas descentralizadas em comparação com abordagens centralizadas, especialmente em cenários de infraestrutura mais extensa ou com um maior número de serviços a serem alocados.

A Tabela 3.1 compara CMFog com uma coleção de trabalhos anteriores, destacando suas respectivas características. Enquanto inúmeros estudos propõem soluções para esse problema, apenas uma parte deles incorpora migrações de conteúdo proativas. Outra característica incomum é a utilização de uma arquitetura de Névoa multi-nível. Cada estudo

aborda o desafio de migração de conteúdo na Computação em Névoa com estratégias distintas. Em [17] os autores adotam um Modelo de Custo Ponderado. No trabalho [27], os autores apresentam sua solução chamada PROMO, também em [35] é utilizado Algoritmo Genético para o processo de tomada de decisão. Uma estratégia que utiliza o Processo Analítico Hierárquico é aplicada em [49].

O CMFog introduz uma estratégia inovadora de migração de conteúdo na Névoa, buscando explorar a flexibilidade proporcionada pela Névoa com múltiplos níveis, um aspecto ainda pouco explorado nas abordagens de migração na Nuvem. O CMFog também adota uma abordagem que incorpora migrações proativas, utilizando o método da Cadeia de Markov para construir uma estrutura de previsão de mobilidade. Em relação à tomada de decisão, o CMFog emprega a otimização MADM para reunir um conjunto de variáveis e identificar as melhores alternativas de migração. Ao explorar os trabalhos na literatura, diversas abordagens para o processo de decisão foram identificadas. No entanto, a abordagem MADM destaca-se como uma solução importante devido à sua flexibilidade em considerar múltiplos atributos para avaliação, sejam eles de custo ou benefício.

Tabela 3.1: Tabela comparativa das principais características do trabalhos similares encontrado na literatura e do CMFog.

Artigo	**Fator de decisão**	**Abordagem**	**Proativo**	**Ambiente**
[43]	Handoff de dispositivo	-		Computação em Névoa
[17]	Mobilidade, tempo de resposta de aplicação, consumo de energia	Weighted Cost Model		Computação em Névoa/Borda multi-nível
[2]	Utilização dos nós de Névoa e mobilidade dos dispositivos	Aprendizado por Reforço		Computação em Névoa
[45]	Path redudantes entre dois hosts	Topologia Leaf and Spine		Computação em Névoa
[21]	Mobilidade e latência	Conjuntos Independentes Máximos Iterativos		Computação em Borda
[28]	Consumo de energia, latência e utilização de CPU	Best Fit Decreasing	x	Computação em Névoa
[27]	Mobilidade, utilização de recurso	PROactive MObility-support model (PROMO)	x	Computação em Névoa
[35]	Mobilidade, utilização e disponibilidade de recursos	Algoritmo Genético	x	
[63]	Detecção de falhas em hosts	Redes Adversárias Generativas	x	Computação em Borda
[1]	Mobilidade, custo e acesso, latência	Otimização Biogeográfica	x	Computação em Névoa
[49]	Disponibilidade de banda, atraso e mobilidade	Análise Hierárquica de Processos (AHP)	x	Computação em Névoa multi-nível
[6]	Predição de mobilidade, latência, utilização de recursos	Dynamic Microservice Application Placement	x	Computação em Névoa multi-nível
CMFog_V	Predição de mobilidade, latência, custo de migração	Otimização por Tomada de Decisão com Multiplos Atributos (MADM)	x	Computação em Névoa multi-nível

Capítulo 4

CMFog: Migração de conteúdo na Computação em Névoa

Nos últimos anos, o cenário computacional tem sido transformado de maneira significativa pela ampla difusão da Internet das Coisas e pelo aumento exponencial de serviços orientados à latência. Essa transformação do cenário de computação tem impulsionado a descentralização dos recursos computacionais, migrando parte da complexidade para a borda da rede. Nesse contexto emergiu a Computação em Névoa, uma abordagem que busca superar as limitações da Computação em Nuvem ao oferecer poder computacional mais próximo do usuário, reduzindo consideravelmente a latência e proporcionando uma Qualidade de Serviço aprimorada.

A popularização de dispositivos IoT e a crescente demanda por serviços de alta sensibilidade à latência desencadearam uma redefinição nos paradigmas de computação. As cidades inteligentes, impulsionadas por sensores e dispositivos interconectados, são exemplos dessa transformação, requerendo soluções mais ágeis e próximas dos usuários. Enquanto a Nuvem desempenha um papel importante na hospedagem de serviços e dados em larga escala, sua infraestrutura centralizada e a dificuldade em lidar com a mobilidade dos usuários resultam em degradação da QoS e interrupções no serviço quando se trata de computação na borda da rede.

Para concretizar o objetivo e motivação da Computação em Névoa, é fundamental que conteúdo e serviços sejam dinamicamente migrados para os nós mais próximos da localização atual do usuário. Esta migração é importante para manter a acessibilidade ao conteúdo com uma latência adequada, alinhada aos requisitos de QoS do usuário.

Esta tese segue uma abordagem *bottom-up*, dividindo o desenvolvimento e experimentos do CMFog em dois momentos. A primeira versão, CMFog$_H$ [4], realiza apenas migrações horizontais, ou seja, entre cloudlets do mesmo nível na Névoa. No segundo momento, CMFog será expandido para CMFog$_V$ [5], incorporando migrações verticais entre cloudlets de diferentes níveis na Névoa. Este capítulo apresentará em detalhes o CMFog$_H$, incluindo seu modelo, estratégias de migração e experimentos para validar essa ferramenta. As particularidades do CMFog$_V$ e seus experimentos correspondentes serão discutidos no próximo capítulo.

4.1 Descrição do modelo

Esta seção tem como objetivo formalizar o modelo do CMFog, estabelecendo termos e fornecendo definições para a infraestrutura e aplicação. Para melhor clareza e organização das informações apresentadas neste trabalho, a Tabela 4.1 resume os símbolos e notações que serão encontrados ao longo da leitura.

Tabela 4.1: Símbolos utilizados no modelo de formalização do CMFog.

Símbolos	**Descrição**
AP	Lista de pontos de acesso
ap_i	Um ponto de acesso i qualquer
FN	Lista de nós da Névoa
fn_j^x	Um nó j do nível x na Névoa
id_j	Identificador único do nó de Névoa j
$fn'_j(fa)$	Nó da névoa j que armazena uma aplicação fa antes da migração
$fn''_j(fa)$	Nó da névoa j que armazena uma aplicação fa após a migração
m	Um dispositivo móvel
ma	Uma aplicação executando em um dispositivo móvel m
fa	Uma aplicação de névoa executando em uma máquina virtual
$d_{mk}(t)$	Distância 2D entre um dispositivo móvel m e o nó da névoa k no tempo t
C	Cloud Datacenter
MK	Uma cadeia de Markov
l	Latência
t_m	Tempo de migração

Devido à falta de uma nomenclatura padronizada para os componentes da infraestrutura de Computação em Névoa, optamos por utilizar o termo "cloudlet" para nos referirmos aos nós de Névoa ou qualquer conjunto de recursos computacionais na borda da rede. Essas cloudlets têm a capacidade de oferecer um ambiente de virtualização, permitindo a instância de máquinas virtuais e containers.

4.1.1 Modelo de Aplicação de Névoa

No CMFog, o conteúdo do usuário assume a forma de uma aplicação, que pode variar desde um serviço de interação em tempo real até um sistema de armazenamento de arquivos. O modelo de aplicação é formado como um par de componentes $[m_{app},\ f_{app}]$, em que m_{app} representa o lado cliente da aplicação, executando no dispositivo do usuário, e f_{app} corresponde ao componente executando nas cloudlets da infraestrutura da Névoa.

Similar à Nuvem, a Computação em Névoa fundamenta-se na virtualização. Para os propósitos deste trabalho, assumiremos que o componente f_{app} está sempre encapsulado em uma máquina virtual. Consequentemente, o evento de migração de uma VM implica automaticamente na migração do componente f_{app} correspondente da aplicação. Essa modelagem busca unificar a representação do conteúdo do usuário, proporcionado uma discussão mais clara.

4.1.2 Modelo de Infraestrutura

O modelo de infraestrutura do CMFog é representado por um 4-tupla de componentes [FN,AP,C,m]. O conjunto $FN = \{fn_1^x, fn_2^x, fn_3^x ... fn_j^x\}$ representa as cloudlets da Névoa, onde x representa o nível da Névoa no qual a cloudlet pertence. Essas entidades são responsáveis por armazenar as máquinas virtuais que encapsulam as aplicações de névoa f_{app}. O componente $AP = \{ap_1, ap_2, ap_3 ... ap_i\}$ representa o conjunto de pontos de acesso que funcionam como ponto de conexão entre os dispositivos móveis e a infraestrutura. O modelo assume que existe uma relação de 1:1 entre uma cloudlet fn_j e um ponto de acesso ap_i, isto é, cada cloudlet possui apenas um ponto de acesso capaz de oferecer conexão ao usuário e um AP só pode estar conectado à uma cloudlet.

O componente C corresponde à Nuvem, um datacenter centralizado, que, em conjunto com a camada de Névoa, fornece recursos computacionais aos usuários. Apesar da Nuvem C ser capaz de prover recursos para a execução de uma aplicação, supõe-se que qualquer aplicação f_{app} é sempre alocada em uma cloudlet na Névoa, visto que o foco principal é explorar a contribuição do CMFog para o processo de migração de conteúdo no ambiente de Computação em Névoa.

Um dispositivo móvel m é representado por [m_{app},at], onde m_{app} é o componente que representa o lado cliente da aplicação e ta corresponde a tecnologia de acesso que o usuário utiliza para estabelecer conexão com a Névoa. A conexão de um dispositivo m com uma cloudlet fn_j implica, por padrão, que essa conexão ocorre através do ponto de acesso ap_i associado à fn_j por meio de uma tecnologia de acesso at.

O processo de migração de uma máquina virtual pode ocorrer de duas formas: horizontal ou vertical. Na migração horizontal, uma VM é transferida entre cloudlets do mesmo nível. Isso pode ser formalizado ao definir que uma VM vm_i é transferida de uma cloudlet fc_i^x para outra fc_j^x, no qual ambas estão localizadas em um mesmo nível. Por outro lado, as migrações verticais envolvem a transferência de uma máquina virtual vm_i entre duas cloudlets, fc_i^x e fc_j^y, onde fc_i^x e fc_j^y pertencem a níveis diferentes. Em outras palavras, os níveis x e y das cloudlets representam níveis diferentes da camada da Névoa.

4.2 Componentes do CMFog$_H$

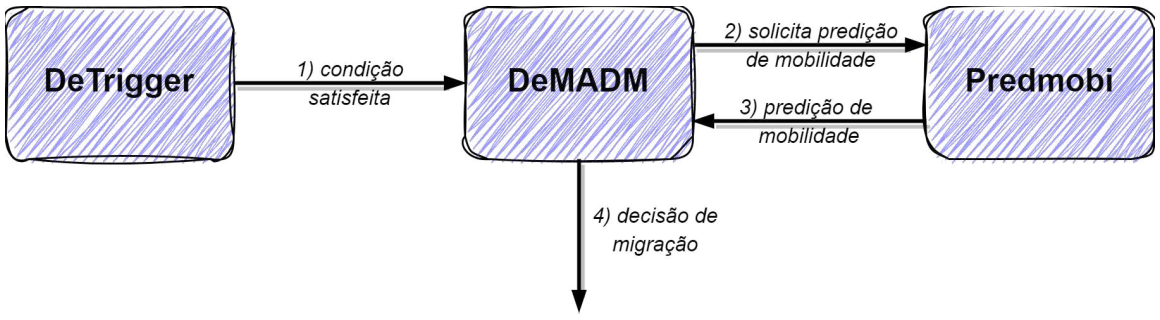


Figura 4.1: Diagrama de componentes e comunicação do CMFog$_H$.

O CMFog$_H$ é composto por três componentes principais: DeTrigger, DeMADM e

Predmobi. A interação entre esses componentes é ilustrada no diagrama apresentado na Figura 4.1. O DeTrigger desempenha o papel de identificar eventos ou condições que sinalizam a necessidade de migração de conteúdo. Em seguida, o DeMADM entra em ação, executando o algoritmo de otimização MADM para avaliar e classificar os possíveis destinos de migração. O componente, então, toma a decisão final sobre a cloudlet destino para o próximo evento de migração.

O Predmobi, como terceiro componente, contribui fornecendo informações de predição de mobilidade do usuário. Essas previsões são valiosas para o DeMADM durante o processo de decisão, influenciando diretamente nas escolhas de próximos destinos de migração das VMs.

4.2.1 DeTrigger

O componente DeTrigger assume a responsabilidade de monitorar os *triggers* e determinar o início de um processo de decisão de migração no contexto do CMFog$_H$. Em muitos casos, a literatura não oferece uma definição clara das condições necessárias para iniciar tal processo, o que torna estratégias capazes de identificar essas condições fundamentais para o funcionamento efetivo dessas políticas em cenários do mundo real. Por exemplo, é possível monitorar variáveis relacionadas aos parâmetros de qualidade de serviço, posição geográfica ou disponibilidade de recursos computacionais.

A ativação de um processo de decisão de migração está intrinsecamente ligada à identificação de um evento ou condição predefinida. O DeTrigger atua como um agente ativo que realiza monitoramento constante das condições especificadas pelos *triggers*. Assim que uma condição é detectada, o componente comunica ao DeMADM a necessidade de iniciar um processo de decisão para avaliar a viabilidade de executar uma migração entre cloudlets.

4.2.2 Predmobi

O componente Predmobi desempenha o papel fundamental de gerenciar o processo de predição de mobilidade no CMFog$_H$. Utilizando informações do histórico de mobilidade do usuário e sua localização atual, o componente tem a responsabilidade de estimar as próximas localizações do usuário em um futuro próximo.

O CMFog realiza a predição de mobilidade por meio de uma adaptação da Cadeia de Markov. A n-MMC é uma variação que adiciona memória à Cadeia de Markov para aprimorar a precisão na predição de mobilidade. Essa estrutura matemática realiza transições de estados com base em probabilidades, considerando os últimos n estados assumidos pelo sistema. No contexto da predição de mobilidade, a Cadeia de Markov é composta por:

- Um conjunto de estados $S = \{s_1, s_2...s_n\}$, onde cada estado corresponde a uma localização visitada ou um Ponto de Interesse (POI). Um POI é um conjunto de localizações próximas, abrangendo uma determinada área geográfica.

- Um conjunto de transições, onde uma transição $t_{i,j}$ representa um usuário movendo-se entre os POIs i e j.

A escolha de realizar a predição de mobilidade utilizando a cadeia de Markov está fortemente relacionada com a adequação deste modelo ao problema de predição de mobilidade, além de ser uma abordagem amplamente difundida nessa área de pesquisa. A cadeia de Markov oferece uma estrutura simples e de baixa exigência computacional, o que é crucial para garantir a eficiência operacional do CMFog.

Um aspecto importante a ser destacado é a flexibilidade do modelo de Markov, que pode se adaptar a novas informações de mobilidade do usuário sem exigir um grande esforço computacional. Essa característica é especialmente relevante no contexto em que o CMFog está inserido, uma vez que os usuários podem se deslocar de maneira imprevisível ou até mesmo mudar suas rotinas frequentemente.

Apesar do CMFog não incluir uma estratégia de retroalimentação das informações de mobilidade, a simplicidade e a adaptabilidade da cadeia de Markov permitem que em um trabalho futuro, o CMFog seja capaz de responder rapidamente às mudanças no comportamento dos usuários, mantendo a precisão na predição de mobilidade e, consequentemente, otimizando o processo de migração de conteúdo.

O Algoritmo 1 esboça a rotina para construção da cadeia n-MMC do CMFog$_H$, dividindo-a em três etapas. A primeira fase, linha 1, realiza a clusterização dos dados, que corresponde ao processo de identificação dos Pontos de Interesse (POIs) a partir das informações contidas no histórico de mobilidade. A identificação de POIs pode seguir várias abordagens, como por exemplo, por meio de pontos de concentração com base em informações coletivas ou por interesses individuais dos usuários. No contexto do CMFog, a abordagem para identificação de POIs é baseada na posição dos Pontos de Acesso, ou seja, cada AP corresponde a um POI no algoritmo.

A segunda etapa acontece entre as linhas 2 e 10. Nesta fase, o algoritmo itera sobre arquivos H que armazenam o histórico de mobilidade do usuário. O principal objetivo é identificar e criar os estados da cadeia n-MMC, que representam as cloudlets no qual o usuário se associou durante seu deslocamento. As linhas 3 a 5 são responsáveis por iterar sobre cada registro de mobilidade T e atribuir um rótulo r à cada coordenada t_i. Esse rótulo corresponde ao POI mais próximo da coordenada em questão. Em seguida, na linha 6, os registros de mobilidade em sequência que receberam o mesmo rótulo são agrupados e o resultado do processo gera um novo conjunto T'. Essa sequência de passos busca relacionar cada localização do dispositivo do usuário a uma possível cloudlet e, ao remover as redundâncias nas sequências de entrada com o mesmo rótulo, estabelece uma sequência de transições entre cloudlets diferentes, para garantir que as transições da cadeia ocorram sempre entre diferentes estados.

As linhas 7 a 9 do algoritmo são responsáveis pela criação do conjunto de estados E que compõe a Cadeia de Markov. As entradas do conjunto T' são iteradas para formar k subconjuntos de tamanho n. Cada um dos k subconjuntos corresponde a um estado e, onde $e = \{t'_{(i)}.r, t'_{(i+1)}.r, ..., t'_{(i+(n-1))}.r\}$. Cada estado criado representa um "momento" do deslocamento do dispositivo do usuário, formado com base em sua localização atual e nas últimas $n-1$ localizações anteriores.

A última etapa da construção da cadeia n-MMC acontece na linha 11. Após iterar por todos os registros de mobilidade e criar o conjunto de estados E é necessário calcular a probabilidade TP de transição entre quaisquer dois estados $i, j \in E$, de tal forma que

Algorithm 1: Algoritmo para construção da cadeia n-MMC.

Input: Conjunto H de histórico de mobilidade
Output: Cadeia de Markov MK

```
POI ← clusterizacao(H)
for cada T ∈ H do
    for cada t_i ∈ T do
        Defina o rótulo r para t_i correspondente ao POI mais próximo
    end
    T' ← Agrupe os registros em sequência com mesmo rótulo em uma única ocorrência
    for cada sequência t'_i,t'_{i+1}...t'_{i+n-1} ∈ T' do
        Adiciona um estado e_k = {t'_(i).r, t'_(i+1).r, ..., t'_(i+(n-1)).r} em E
    end
end
TP ← probabilidades de transição entre quaisquer dois estados i,j ∈ E |
    i_(n) = j_(1) | i_(n) ∈ i e j_(1) ∈ j
MK.add(S,TP)
return MK
```

n-ésimo rótulo do i seja igual ao primeiro de j. Após a conclusão dessa etapa a estrutura formada pela cadeia n-MMC é capaz de informar um conjunto de cloudlets que podem ser os possíveis próximos destinos de um usuário baseado nas suas últimas n localizações.

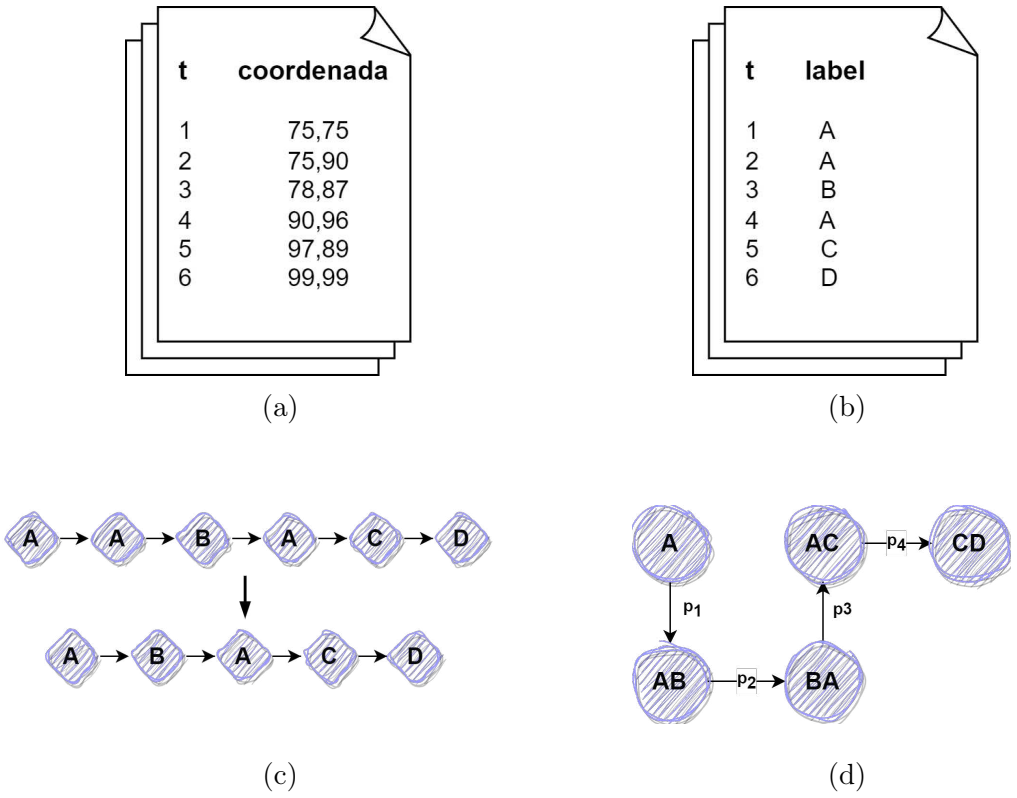


Figura 4.2: Etapas seguidas pelo algoritmo de construção da cadeia n-MMC em um cenário de exemplo.

A Figura 4.3 fornece um exemplo ilustrativo do funcionamento do algoritmo utilizado para construir a cadeia n-MMC, responsável pela predição de mobilidade no CMFog. É

importante destacar que os valores apresentados na ilustração são hipotéticos e destinam-se apenas para exemplificação.

Na Figura 4.2a, é apresentado um exemplo de um arquivo de histórico de mobilidade gerado pela ferramenta Hermoupolis, representando o histórico de coordenadas de um dispositivo ao longo do tempo t. O algoritmo inicia sua execução iterando sobre o arquivo, atribuindo um rótulo a cada coordenada registrada. Esses rótulos associam as coordenadas à cloudlet mais próxima. Neste exemplo, os rótulos possíveis são A, B, C ou D. Como mostrado na Figura 4.2b, cada uma das seis entradas recebe um desses rótulos.

O próximo passo é agrupar registros consecutivos que receberam o mesmo rótulo, conforme ilustrado na Figura 4.2c. Por exemplo, as entradas para $t = 1$ e $t = 2$ que receberam o rótulo "A" são agrupadas e formam uma única ocorrência. O passo subsequente consiste em armazenar essa informação em uma estrutura de fácil acesso, neste exemplo iremos utilizar com um grafo para melhor representação. O grafo da cadeia está representado na Figura 4.2d, no qual cada vértice do grafo é formado por dois rótulos X e Y, representando a cloudlet anterior X e a cloudlet atual Y ao qual o dispositivo esta associado. A formação dos vértices é realizada iterando sobre a sequência de cloudlets associadas pelo dispositivo ao longo do tempo t. Para $n = 2$, teremos os seguintes vértices formados: AB, BA, AC, CD. Um vértice especial formado apenas pelo rótulo "A" é utilizado para representar o estado inicial do dispositivo.

As arestas do grafo indicam transições entre os vértices e possuem um valor p, representando a probabilidade do dispositivo se deslocar entre as duas cloudlets que as vértices da aresta representam. O algoritmo executa esse procedimento para um conjunto de arquivos que contém histórico de mobilidade, identificando vértices que representam diferentes caminhos que o dispositivo pode percorrer durante seu deslocamento. A computação da probabilidade de transição entre quaisquer dois vértices da cadeia é realizada somente após a avaliação de todos os registros. Dessa forma, o algoritmo realiza uma análise detalhada dos padrões de mobilidade do dispositivo, construindo uma representação probabilística da dinâmica de suas transições entre cloudlets ao longo do tempo. Esse processo de construção é importante para extrair e identificar os padrões de mobilidade para que seja possível realizar a predição do comportamento de dispositivos móveis em ambientes de Computação em Névoa.

4.2.3 DeMADM

A ativação de um *trigger* indica a possibilidade de migração de uma VM, no entanto, a decisão final sobre a migração é delegada ao DeMADM, que avalia diversos fatores para determinar a real necessidade desse processo. Este componente realiza uma análise, considerando variáveis específicas, e, caso a migração seja necessária também identifica qual cloudlet é o destino mais adequado. A predição de mobilidade desempenha um papel vital nesse processo, permitindo ao DeMADM escolher o destino da migração considerando padrões de mobilidade dos usuários.

O DeMADM desempenha um papel fundamental na seleção da melhor cloudlet de destino para a migração de uma VM, visando maximizar a proximidade entre o conteúdo e o usuário. Este componente realiza essa tarefa por meio da análise de diversos fato-

res, envolvendo uma avaliação criteriosa de múltiplas variáveis. Para isso, o modelo de Otimização de Tomada de Decisão por Múltiplos Atributos foi adotado como estratégia pelo CMFog. Esse modelo considera várias variáveis de entrada, cada uma com seus pesos correspondentes, para calcular a pontuação de cada alternativa. O resultado é um ranking de destinos de migração, que o sistema utiliza para escolher a opção que visa minimizar a latência.

Existem vários algoritmos que podem ser empregados pelo DeMADM para classificar as alternativas de migração com base nos critérios e pesos atribuídos. Alguns exemplos incluem TOPSIS, DiA, MeTHODICAL e VIKOR. Estudos comparativos entre esses algoritmos em contextos de migração de conteúdo revelaram que o MeTHODICAL demonstrou melhor desempenho em uma variedade de cenários, destacando-se por sua flexibilidade e eficiência no processo de seleção de soluções. Portanto, o CMFog optou por adotar o MeTHODICAL como algoritmo padrão a ser utilizado pelo DeMADM. É importante ressaltar que o DeMADM foi desenvolvido de forma genérica, o que significa que sua operação não está vinculada a um algoritmo MADM específico, possibilitando a adição de algoritmos MADM adicionais.

Algorithm 2: Algoritmo de tomada de decisão do componente DeMADM.

Input: Usuário U, Peso W dos atributos, Peso BW atributos de beneficio, Peso CW atributos de custo
Output: Cloudlet destino M para migração

```
1  MOB ← predição de mobilidade do usuário U
2  B ← dados das variáveis de beneficio
3  C ← dados das variáveis de custo
4  RESULTADO ← executarMADM(MOB, B, C, BW, CW)
5  RANK ← ∅
6  for cada cloudlet c ∈ RESULTADO do
7  |  if atendeRequisitos(c,U) then
8  |  |  Adicione c à lista RANK
9  |  end
10 end
11 M ← cloudlet melhor classificada em RANK
12 return M
```

O Algoritmo 2 detalha as etapas seguidas pelo DeMADM para calcular a pontuação de cada possível cloudlet de destino durante o processo de decisão. Na linha 1, as informações de predição de mobilidade para o usuário U são obtidas por meio do componente *Predmobi*. Nas linhas 2 e 3, as variáveis de benefício e custo necessárias para o processo de decisão são coletadas. Em seguida, essas variáveis, seus pesos e as informações de predição de mobilidade são fornecidos como entrada para a execução do algoritmo de otimização MADM. A variável $RESULTADO$ armazena uma lista com os possíveis destinos de migração classificados de acordo com a pontuação gerada pelo algoritmo MADM. As linhas 6 a 10 iteram sobre a lista $RESULTADO$ para selecionar qualquer cloudlet c com pontuação maior que zero e com recursos suficientes para receber a VM do usuário, adicionando-as a uma nova lista $RANK$. Na linha 11, o algoritmo seleciona a cloudlet M de maior pontuação em $RANK$ como destino e inicia o processo de migração do conteúdo.

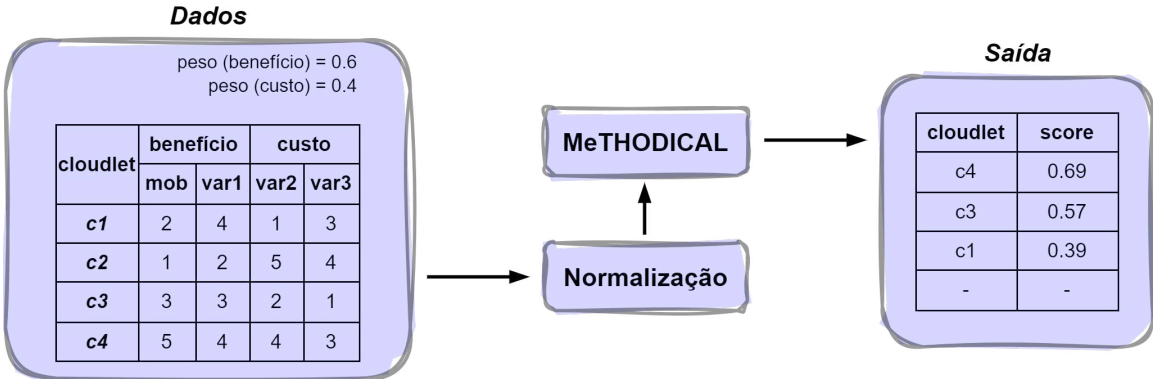


Figura 4.3: Etapas seguidas pelo algoritmo de tomada de decisão do componente DeMADM em um cenário de exemplo.

A Figura 4.3 ilustra o funcionamento do Algoritmo 2. O processo de tomada de decisão é iniciado quando o componente recebe as informações do usuário, incluindo o peso de cada variável assim como o peso de benefício e custo. Para este exemplo, os pesos de benefício e custo são considerados como 0.6 e 0.4, respectivamente.

Inicialmente, são solicitadas informações de mobilidade ao componente Predmobi para identificar os destinos potenciais e as probabilidades associadas a cada alternativa. A tabela na Figura 4.3 apresenta o resultado desse processo, onde as cloudlets $c1$, $c2$, $c3$ e $c4$ foram identificadas como possíveis destinos. Para complementar a coleta de informações, são obtidas informações das outras variáveis para cada cloudlet: $var1$, juntamente com a probabilidade de migração mob, para as variáveis de benefício; e $var2$ e $var3$ para as variáveis de custo. Os valores dessas variáveis são obtidos por meio de agentes que monitoram as entidades da infraestrutura de Névoa. Essas informações compõem o conjunto de parâmetros de entrada que serão utilizados no processo de tomada de decisão de migração.

O objetivo do processo de otimização MADM é ranquear as alternativas, maximizando os atributos de benefício e minimizando os de custo. Após a coleta das informações, os dados passam por um processo de normalização para assegurar que as grandezas de cada variável não distorçam os resultados. Posteriormente, os dados normalizados são submetidos a um algoritmo MADM, responsável pelo cálculo e ranqueamento das alternativas. O resultado é uma lista de cloudlets ordenadas, onde as primeiras representam as melhores escolhas. O componente itera sobre essa lista para verificar se as cloudlets têm capacidade para receber a VM do usuário. A cloudlet melhor ranqueada é então escolhida como destino de migração, sendo a $c4$ a selecionada para este exemplo. No cenário hipotético de nenhuma cloudlet atender aos critérios para seleção de um destino, a decisão de migração é postergada para a próxima iteração. Esse processo de tomada de decisão é executado iterativamente cada vez que o componente DeMADM é acionado, considerando não apenas a disponibilidade de recursos nas cloudlets, mas também outras métricas como as informações de mobilidade.

4.3 Políticas de Migração

A mobilidade dos usuários e a área de cobertura limitada das cloudlets exigem a migração constante dos conteúdos na borda da rede para garantir uma baixa latência. Para ser capaz de minimizar a latência fim-a-fim dos usuários, é essencial maximizar a proximidade entre conteúdo e usuário. Portanto, torna-se fundamental o desenvolvimento de políticas eficazes no processo de gerenciamento de migração de conteúdo na borda da rede, replicando a ubiquidade da Nuvem na Névoa.

O termo *latência fim-a-fim*, ou simplesmente *latência*, refere-se ao intervalo de tempo necessário para que uma requisição do usuário seja transmitida do dispositivo móvel, processada pela VM correspondente na Névoa e, finalmente, para que o dispositivo receba uma resposta à requisição. Como evidenciado na Fórmula 4.1, a latência é composta pela soma de diversos tipos de atrasos, onde: i) A_{AT} representa o atraso de comunicação gerado pela tecnologia de acesso que conecta o usuário à infraestrutura; ii) A_I corresponde à soma do tempo necessário para que uma requisição percorra a infraestrutura da Névoa até a VM de destino e retorne para a cloudlet em que o usuário se encontra associado; e iii) A_{FP} representa o tempo necessário para o processamento da requisição na máquina virtual.

$$L = A_{AT} + A_I + A_{FP} \tag{4.1}$$

A seguir, serão apresentadas as políticas de migração utilizadas pelo CMFog, cada uma com estratégias distintas para identificar a necessidade de migração e para determinar a cloudlet de destino para a transferência do conteúdo.

4.3.1 Política 1: Sem migração de conteúdo

Na Política 1, não ocorre migração de conteúdo durante a movimentação do usuário. Seu principal propósito é servir como uma base de comparação para avaliar os resultados das outras políticas que realizam algum tipo de migração de conteúdo.

A Figura 4.4 ilustra um cenário que representa o funcionamento da política. Nesse cenário, o usuário pode se associar a diferentes cloudlets à medida que se move entre as suas respectivas áreas de cobertura. Como não há a possibilidade de realizar migrações, o conteúdo permanece armazenado de maneira fixa em uma máquina virtual na cloudlet de origem fn_1. Enquanto o usuário estiver associado a esta cloudlet, não há necessidade de encaminhar a requisição entre outras na camada de Névoa, portanto, $A_I = 0$, e a latência é dada apenas por $L = A_{AT} + A_{FP}$.

Quando o usuário se desloca para uma localização fora da área de cobertura de fn_1, ou seja, nas áreas A_2 e A_3, as requisições precisarão ser encaminhadas por meio de outras cloudlets da infraestrutura para alcançar a que armazena a VM. Em cenários nos quais o usuário está fora da área de cobertura da cloudlet origem, a latência é calculada pela Equação 4.2, onde $A_I = A_{I(fn_i - fn_j)}$ corresponde ao tempo para a requisição ser encaminhada do nó fn_i para fn_j e para a resposta da requisição retornar a fn_i. Por exemplo, no cenário em que o usuário está associado a cloudlet da área A_2, a latência é dada por $L = A_{AT'} + A_{I(fn_2 - fn_1)} + A_{FP}$.

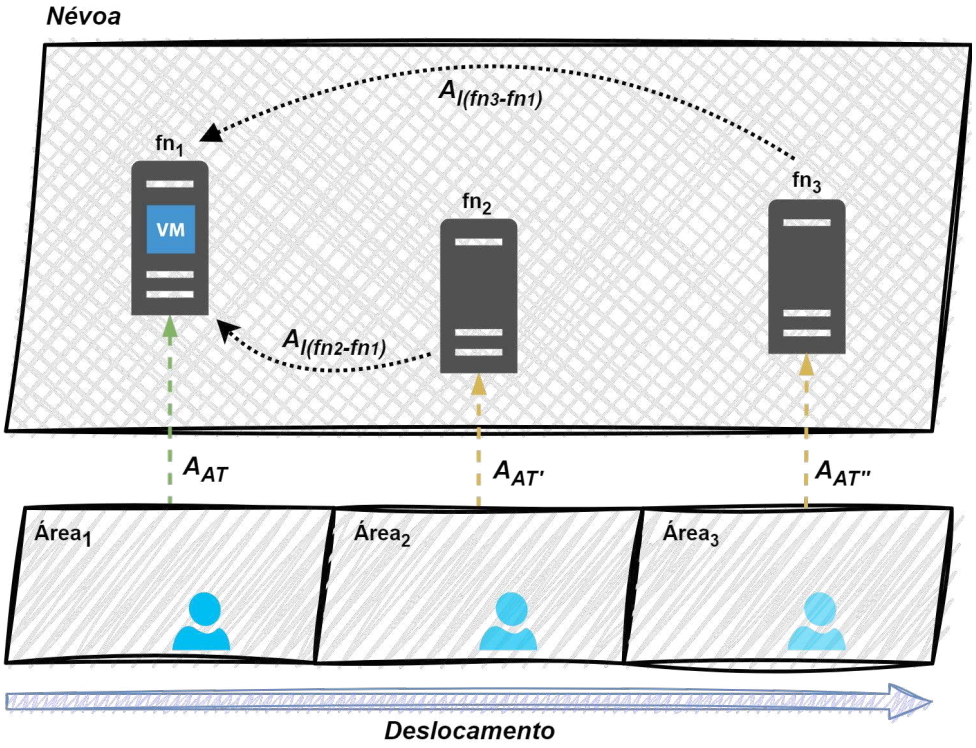


Figura 4.4: Esquematização da interação da Política 1 com um cenário de mobilidade do usuário sem migração de máquina virtual.

$$L = A_{AT} + A_{I(fn_i - fn_j)} + A_{FP} \tag{4.2}$$

Em cenários sem a possibilidade de migração de conteúdo, a latência pode ser drasticamente comprometida à medida que o usuário se afasta da área de cobertura da cloudlet que armazena sua VM. Além da degradação dos requisitos de QoS, o aumento das mensagens trocadas na infraestrutura de Névoa para manter a comunicação pode contribuir para um congestionamento da rede. As próximas políticas são capazes de realizar a migração do conteúdo para possibilitar que sua VM esteja o mais próximo possível, mesmo em cenários em que os usuários se desloquem por grandes distâncias, afastando-se de sua cloudlet origem.

4.3.2 Políticas 2 e 3: Migração proativa de conteúdo com *trigger*

Ao contrário da Política 1, nas Políticas 2 e 3, há a possibilidade de o conteúdo ser migrado para outras cloudlets. A Figura 4.5 ilustra um cenário em que, à medida que o usuário se desloca, o conteúdo é migrado para cloudlets mais próximas. Por exemplo, quando o usuário deixa a área de cobertura A_1 e entra na área A_2, um evento de migração é iniciado para que a VM seja migrada para a nova cloudlet à qual o usuário se associou.

Em um cenário hipotético em que a VM pode ser migrada com atraso mínimo, em qualquer instante de tempo, a latência pode ser calculada pela Equação 4.3. No entanto, em um cenário real, é preciso considerar que existe um atraso necessário para a migração da VM e para identificar o destino mais provável do usuário. O objetivo das políticas

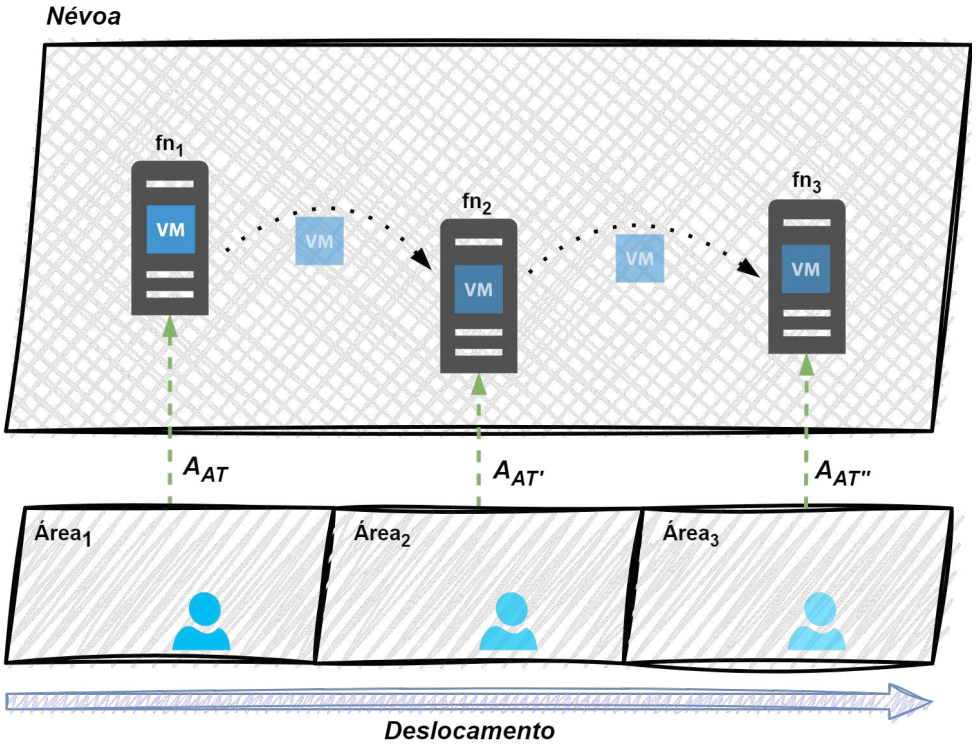


Figura 4.5: Esquematização da interação da Política 1 e 2 com um cenário de mobilidade do usuário com migração de máquina virtual.

descritas nesta seção é maximizar o tempo de proximidade entre o conteúdo e o usuário, em outras palavras, maximizar o tempo em que o cálculo da latência seja realizado através da Equação 4.3 em vez da Equação 4.2.

$$L = A_{AT} + A_{FP} \tag{4.3}$$

Um ponto de grande importância para políticas de migração é definir quando uma migração de conteúdo deve ser iniciada. O termo *trigger* será utilizado para se referir a um evento ou gatilho que sinalize para uma política a necessidade de que um processo de tomada de decisão para migração seja iniciado. As políticas possuem a capacidade de realizar a migração de VMs entre as cloudlets da Névoa, no entanto, fazem uso de *triggers* diferentes.

Política 2: limite de área de cobertura

A Política 2 utiliza um *trigger* baseado na distância entre o usuário e os extremos da área de cobertura da cloudlet à qual está atualmente associado. O *trigger* inicia um processo de tomada de decisão para migração quando o usuário se aproxima a uma distância Z do limite da área de cobertura de uma cloudlet. Essa condição é ilustrada na Figura 4.6.

Política 3: tempo estimado de migração

O *trigger* da Política 3 fundamenta-se em informações extraídas do histórico de mobilidade do usuário para determinar o momento adequado para iniciar um evento de migração. A

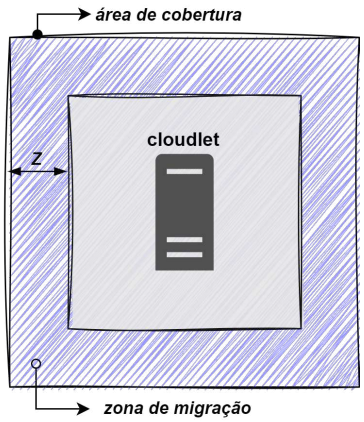


Figura 4.6: Representação dos aspectos considerados pelo *trigger* baseado na proximidade do usuário com os limites da área de cobertura de uma cloudlet.

Figura 4.7 esboça o funcionamento desse *trigger*, onde T_{PERM} denota uma estimativa do tempo médio que o usuário geralmente permanece associado a uma cloudlet, e T_{MIG} é a estimativa do tempo necessário para concluir a migração do conteúdo da VM. O instante de tempo M_{fim} assinala o término do último evento de migração e t_j o tempo estimado que o usuário irá transicionar para uma nova cloudlet. Baseado nesses marcos, o *trigger* identifica o instante de tempo t_{inicio_mig} para que um processo de decisão de migração seja iniciado, o momento é determinado pela diferença entre o tempo estimado de permanência do usuário e o tempo necessário para realizar a migração da VM.

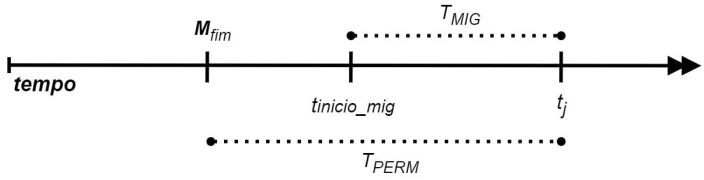


Figura 4.7: Representação dos aspectos considerados pelo *trigger* baseado no tempo estimado de permanência do usuário em uma cloudlet.

Essencialmente, um *trigger* é acionado com base em critérios temporais e na dinâmica de permanência do usuário em uma determinada cloudlet. Ao considerar o histórico de mobilidade, o CMFog utiliza essas estimativas para otimizar o início de eventos de migração, assegurando uma transição suave entre cloudlets e minimizando possíveis impactos na experiência do usuário.

4.4 Simulação

A avaliação do funcionamento de novas estratégias ou ferramentas no mundo real não é uma tarefa trivial, dada a diversidade de desafios, como custo elevado, dificuldade em criar ambientes controlados e interferência de fatores externos ou eventos imprevistos. Para superar essas limitações, simuladores surgem como ferramentas valiosas. Eles possibilitam a simulação de cenários do mundo real em um ambiente virtual, capaz de replicar diversas

características reais. Isso viabiliza a condução de experimentos de forma controlada, abstraindo, ao menos parcialmente, as complexidades do mundo real.

Nesta seção, serão abordados vários aspectos relacionados aos experimentos conduzidos para avaliar o CMFog$_H$. Apresentaremos as métricas de avaliação, os cenários utilizados, os parâmetros empregados na construção da infraestrutura computacional, e informações relevantes sobre a mobilidade dos usuários nos experimentos.

4.4.1 A Infraestrutura

A infraestrutura de rede utilizada nos experimentos do CMFog$_H$ segue o modelo de três camadas [18]. A camada superior representa a Nuvem, a do meio corresponde à Névoa, e a terceira é formada pelos dispositivos do usuário conectados à rede. Neste primeiro momento, a camada de Névoa é composta por apenas um nível. Os Pontos de Acesso atuam como gateways para que os usuários estabeleçam conexão com as cloudlets. A Tabela 4.2 apresenta os parâmetros para as conexões estabelecidas entre as entidades da infraestrutura.

Tabela 4.2: Valores dos parâmetros de conexões entre as entidades da simulação.

Entidade A - Entidade B	**Latência (ms)**	**Largura de Banda (Mbps)**
Ponto de Acesso - Cloudlets	1 - 3	280 - 320
Cloudlets - Nuvem	30 - 40	130 - 170

Diversas tecnologias de acesso permitem que dispositivos do usuário se associem à infraestrutura de rede. Para representar parte dessa diversidade nos experimentos de simulação do CMFog$_H$, foram selecionadas três das principais tecnologias utilizadas no cotidiano dos usuários: 4G, 5G e WiFi. As especificações dessas tecnologias de acesso foram obtidas por especificações na literatura [68][66] e são apresentadas na Tabela 4.3.

Tabela 4.3: Parâmetros de conexão das tecnologias de acesso.

Tecnologia	**Latência (ms)**	**Largura de Banda (Mbps)**
4G	8 - 12	8 - 12
WiFi	4 - 8	120 - 180
5G	1 - 4	80 - 120

Os experimentos de simulação utilizam diferentes tamanhos de máquinas virtuais para representar uma variedade de tipos de conteúdo, desde aplicações leves que rodam em pequenos containers até grandes volumes de dados que necessitam de grandes VMs. Os tamanhos das VMs foram definidos de forma arbitrária para cobrir um amplo espectro de requisitos computacionais. Quatro tamanhos distintos foram utilizados nos experimentos: pequeno (100-250 MBs), médio (400-500 MBs), grande (900-1000 MBs) e extra-grande (1900-2000 MBs). Essa diversidade de tamanhos de VMs permite avaliar como o CMFog se comporta em diferentes cenários de uso, garantindo que a estratégia de migração de conteúdo seja testada em condições variadas.

O modelo de Computação em Névoa é conhecido pela heterogeneidade das entidades que compõem o ambiente, no qual as cloudlets são formadas principalmente por hardware

commodities e os equipamentos dos usuários podem variar desde *wearables* até mesmo veículos. Para emular esse cenário nos experimentos, os valores de latência, largura de banda e tamanho das VMs são representados por variáveis aleatórias extraídas de um intervalo de números que seguem uma distribuição normal.

4.4.2 MyiFogSim

Um ambiente de Névoa é formado por uma complexa infraestrutura computacional com diversos dispositivos, provendo recursos computacionais para os dispositivos de usuário na borda da rede. A tarefa de analisar propostas de solução é um desafio complexo para ser realizada no mundo real devido aos diversos aspectos imprevisíveis que essas infraestruturas oferecem. Portanto, é comum que as soluções propostas sejam avaliadas por meio de simulações, onde é possível criar representações do mundo real em ambientes virtuais proporcionando um ambiente de avaliação mais controlado, de menor custo e esforço.

O iFogSim [33] é uma ferramenta de simulação que estende as capacidades do simulador de nuvem CloudSim [8] ao adicionar aspectos relacionados a Computação em Névoa. O simulador foi desenvolvido com a linguagem Java e fornece um conjunto de ferramentas capaz de modelar e simular infraestruturas de Névoa, além de permitir a coleta de dados referentes a latência de comunicação entre as entidades, consumo de energia e recursos, redes e diversos outros aspectos.

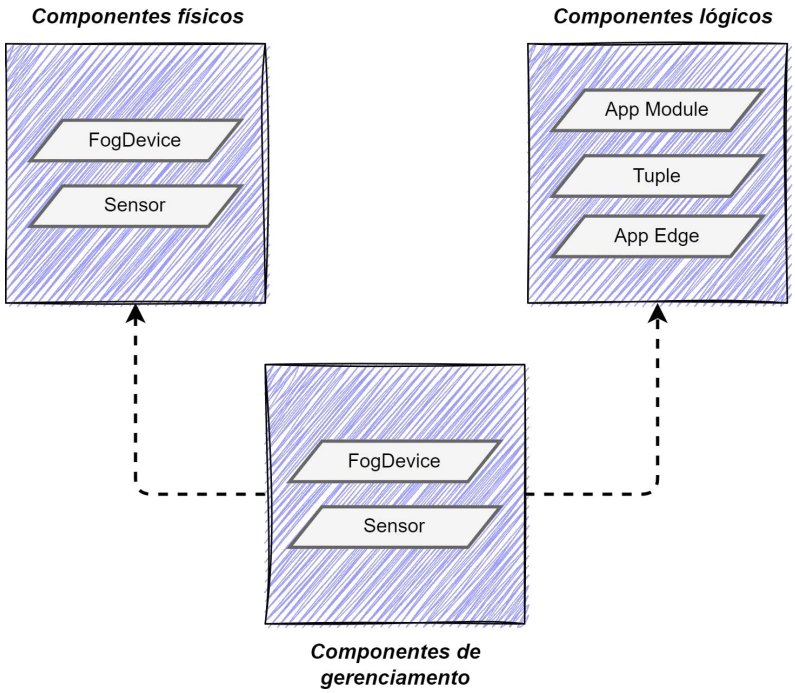


Figura 4.8: Diagrama de componentes e respectivas entidades que compõem o simulador iFogSim.

O iFogSim é uma das soluções de simulação para o ambiente de Névoa mais utilizadas atualmente, sendo diversos os estudos que utilizam a ferramenta para simular e avaliar a proposta de novas soluções. Conforme ilustrado na Figura 4.8, o simulador é formado por

três tipos de componentes [33]: físicos, lógicos e de gerenciamento. Os componentes físicos são representados pelas entidades de FogDevice e Sensors que correspondem às entidades que formam a infraestrutura de Névoa, como, por exemplo, sensores, smartphones, nós de Névoa e até mesmo data-centers da Nuvem.

Os componentes lógicos, por sua vez, são formados pelo Application Module, Application Edge e Tuples. As aplicações no simulador são representadas por grafos, onde os vértices representam módulos de aplicação representados pelo Application Module, e as arestas representam o fluxo de comunicação dos módulos pela aplicação pelo Application Edge. O componente Tuple representa as mensagens trocadas entre módulos de aplicação. Esses componentes lógicos adicionam uma camada de abstração essencial para a modelagem detalhada do comportamento das aplicações na infraestrutura de Névoa.

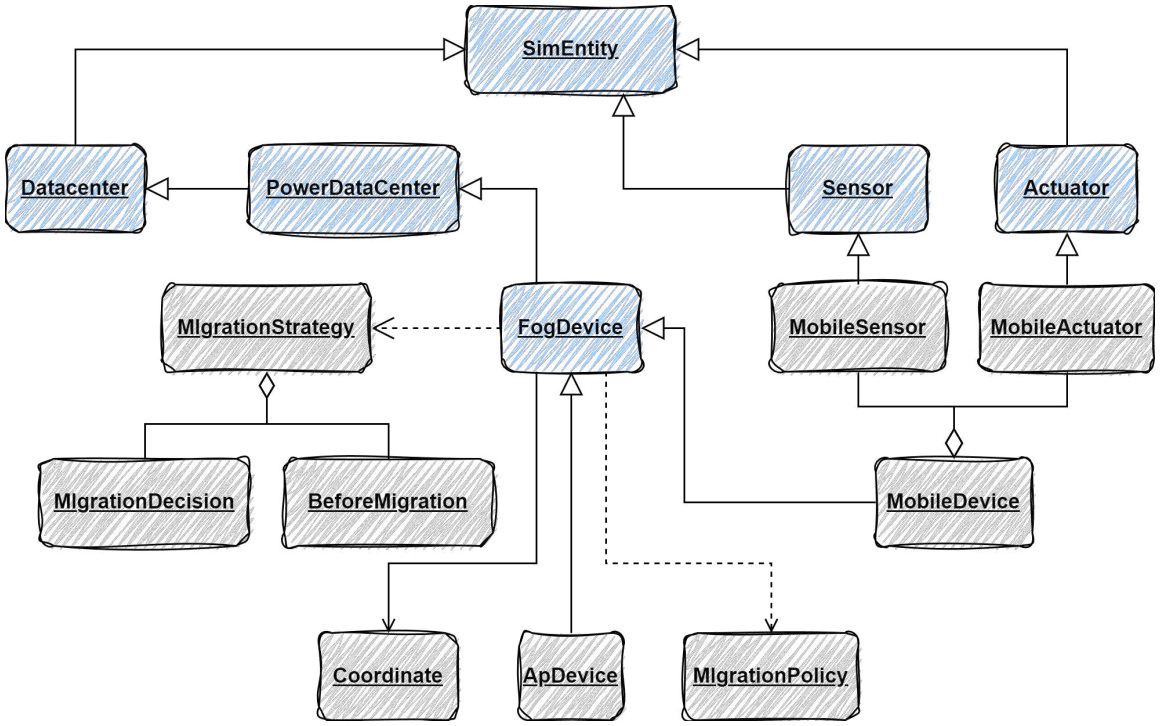


Figura 4.9: Diagrama das principais entidades que compõem o simulador MyiFogSim.

Os componentes de gerenciamento são responsáveis por coordenar componentes físicos e lógicos de simulação (Controller) e realizar o mapeamento de conexões e associações entre entidades (Mapping). Esse nível de gerenciamento adiciona a capacidade de controlar o fluxo da simulação, ajustando as interações entre entidades conforme necessário.

O MyiFogSim [32], uma extensão do iFogSim, foi projetado especialmente para simular políticas de migração em cenários com usuários móveis. A arquitetura do MyiFogSim, apresentada na Figura 4.9, adiciona aspectos de mobilidade às entidades base do iFogSim. O simulador introduz entidades como Migration Policy, que decide quando uma aplicação deve ser migrada, levando em conta aspectos de mobilidade do usuário e políticas de migração. O Migration Strategy é responsável por determinar como e para onde o processo de migração deve ocorrer. O MyiFogSim inclui ainda entidades como Access Point e Coordinate para incorporar a noção de posição geográfica na simulação.

O CMFog busca oferecer uma estratégia de migração para ambientes de Névoa, enfa-

tizando aspectos de mobilidade nos diversos dispositivos de borda. Após analisar recursos oferecidos por diversos simuladores para o contexto de Computação em Névoa, o simulador foi escolhido como a ferramenta de modelagem da infraestrutura e tomada de decisão de migração, proporcionando um ambiente virtual para avaliar o comportamento do CMFog em diferentes cenários. As entidades e políticas específicas do CMFog foram integradas ao MyiFogSim como um pacote de classes, funcionando como um módulo do simulador.

4.4.3 Histórico de mobilidade

Aspectos relacionados à privacidade tornam o acesso às informações de mobilidade dos usuários uma tarefa sensível e complexa. Embora existam conjuntos de dados disponibilizados em repositórios públicos que ofereçam históricos de mobilidade de forma anônima, para cumprir os objetivos específicos dos experimentos do CMFog$_H$, tornou-se necessário contar com dados de mobilidade que fossem capazes de representar o cotidiano real dos usuários.

Dada a complexidade e a sensibilidade no acesso aos históricos reais de mobilidade de usuários, optou-se por utilizar dados gerados pela ferramenta Hermoupolis [41]. Esta ferramenta, ao definir um mapa e personalizar diferentes parâmetros, pode gerar registros de mobilidade que emulam o comportamento de um usuário.

Uma rotina foi estabelecida para representar a mobilidade do usuário em seu dia a dia, iniciando em sua residência (H), seguindo para o trabalho (W), dirigindo-se à academia (G) e, no final do dia, retornando para casa. A Figura 4.10 ilustra graficamente um histórico de mobilidade gerado pelo Hermoupolis, no qual o usuário segue a rotina descrita anteriormente. Os dados de mobilidade foram gerados utilizando o mapa de uma porção da cidade de Atenas, Grécia.

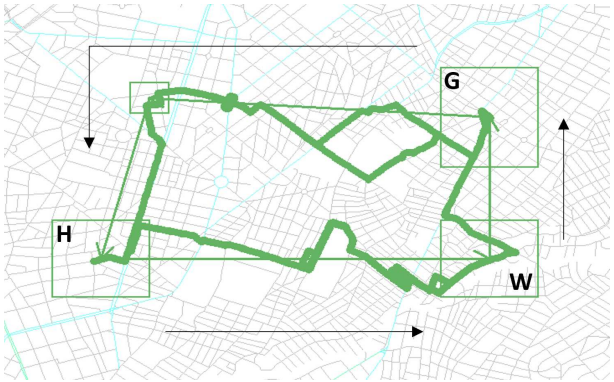


Figura 4.10: Rotina do usuário utilizada para gerar registros de mobilidade para experimentos do CMFog$_H$.

O Predmobi, componente responsável pela predição de mobilidade, utiliza os dados gerados pelo Hermoupolis para construir a cadeia n-MMC. Para evitar que a cadeia se comporte como um oráculo, um segundo conjunto de dados é utilizado para determinar o deslocamento do usuário durante a simulação. Ambos os conjuntos de dados foram gerados utilizando os mesmos parâmetros e o mesmo mapa.

Embora o conjunto de históricos de mobilidade usado para orientar os usuários durante a simulação seja gerado com base na rotina envolvendo os POIs: H, W, e G. O

caminho percorrido no mapa entre qualquer um desses pontos não é fixo e pode variar. Além disso, os traces de mobilidade usados para construir a cadeia de Markov para predição de mobilidade não são compartilhados com o subconjunto responsável por definir o deslocamento do usuário durante a simulação. Esses aspectos são cruciais para garantir que o caminho do usuário durante a simulação possa variar, introduzindo assim cenários de imprevisibilidade no processo de predição de mobilidade. Isso exige que o CMFog se adapte a situações que representem os movimentos imprevisíveis do usuário.

A cadeia n-MMC utilizada nos experimentos foi construída com o parâmetro $n = 2$, o que significa que a predição de mobilidade é realizada considerando as duas últimas localizações do usuário. Estudos demonstram que uma cadeia 2-MMC oferece níveis satisfatórios de precisão para previsão de mobilidade sem degradações significativas em desemepnho [14].

As cloudlets, que representam as entidades de processamento na borda da rede, possuem a mesma área de cobertura e são distribuídas uniformemente no espaço de simulação. Na etapa de agrupamento na construção da Cadeia de Markov, cada cloudlet é identificada como um POI, e os registros de mobilidade recebem o rótulo equivalente à cloudlet mais próxima da localização. Essa abordagem permite uma representação mais fiel do comportamento do usuário em relação às cloudlets distribuídas na área de cobertura.

4.4.4 Métricas

Para analisar os resultados dos experimentos, é fundamental monitorar variáveis que funcionem como indicadores de desempenho do CMFog$_H$. A avaliação dos experimentos levará em consideração um conjunto de quatro métricas: latência de aplicação, número de migrações, tuplas geradas e tuplas perdidas.

A métrica de latência de aplicação, ou simplesmente latência, monitora o tempo entre a saída da solicitação do usuário do dispositivo até o momento em que uma resposta é recebida da infraestrutura. A métrica de migração representa o número total de eventos de migração iniciados e concluídos durante a simulação. As duas últimas métricas, tuplas geradas e tuplas perdidas, correspondem ao número total de mensagens geradas e perdidas, respectivamente.

As métricas de latência e migração possuem um alto grau de correlação e, juntas, oferecem informações importantes para entender como as decisões tomadas pelo CMFog$_H$ impactam na proximidade entre o conteúdo e o usuário. As métricas de tuplas funcionam como indicadores adicionais, capazes de validar as conclusões derivadas das outras métricas, uma vez que expressam resultados diretamente relacionados ao tempo de atividade (*uptime*) e tempo de inatividade (*downtime*) das VMs.

4.4.5 O experimento

O CMFog$_H$ realiza exclusivamente migrações horizontais, ou seja, migrações entre cloudlets do mesmo nível. O diagrama na Figura 4.11 ilustra a sequência de eventos quando o CMFog$_H$ é iniciado. O processo começa com a identificação da ativação de um *trigger* pelo componente DeTrigger. Posteriormente, o DeMADM é acionado para iniciar o

processo de decisão de migração, solicitando as informações de predição de mobilidade ao PredMobi para a execução do algoritmo MADM. Se o resultado do MADM indicar que não há necessidade de migração, nenhuma ação é tomada, e o componente DeTrigger retorna ao estado de monitoramento. Se a saída do MADM indicar a necessidade de migrar uma VM, a cloudlet de destino é identificada e o processo de migração é iniciado.

Figura 4.11: Diagrama de sequência que ilustra o funcionamento do CMFog$_H$.

O experimento da primeira etapa visa avaliar o desempenho do CMFog em um ambiente de Névoa com apenas um único nível. A redução da complexidade tem o propósito de identificar aspectos positivos e negativos da estratégia em cenários mais simples, fornecendo informações que servirão como base de comparação para cenários mais complexos. Este enfoque mais simples facilita a análise, já que há um número menor de possibilidades e variáveis no processo de decisão.

4.4.6 Definições de simulação

As simulações ocorrem em um espaço imaginário com dimensões de $2500x2500$ metros. Na Figura 4.12, apresenta-se um exemplo do espaço de simulação, onde cada cloudlet possui uma área de cobertura c de 250 metros. Observam-se também regiões de sobreposição de áreas de cobertura o. As sobreposições foram incorporadas para ampliar as opções de cloudlets às quais os dispositivos dos usuários podem se conectar.

O processo de design de experimentos desempenha um papel fundamental na estruturação e análise de experimentos, sendo também importante para garantir a reprodutibilidade. De acordo com Jain [25], na área de sistemas de computação, o início de um experimento computacional corresponde à definição de um projeto experimental. Nesse

contexto, a construção do experimento envolve a definição de conjuntos de variáveis, onde as variáveis de controle são designadas como fatores, sendo manipuladas durante o experimento, enquanto os níveis representam os valores que essas variáveis podem assumir. Por sua vez, as variáveis de resposta indicam as métricas que serão observadas, fornecendo informações sobre a execução do experimento.

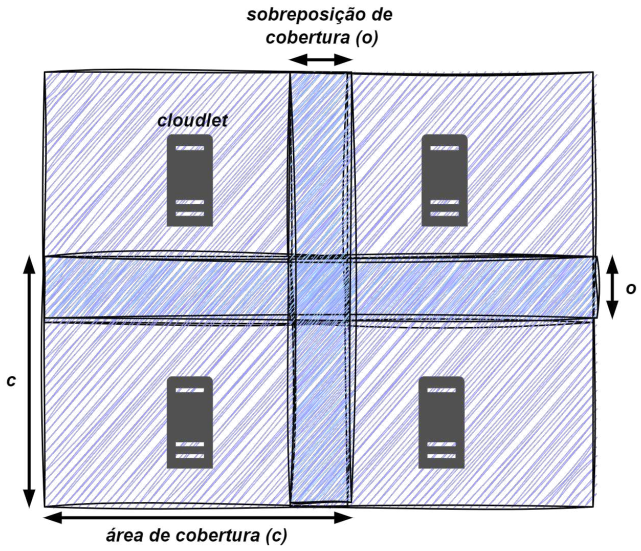


Figura 4.12: Exemplo da distribuição das cloudlets no espaço de simulação dos experimentos.

O objetivo de um projeto de experimentos é compreender como os diferentes valores dos fatores, ou seja, os níveis, influenciam as variáveis de resposta. Em um projeto simples, estuda-se o impacto dos valores de um único fator, mantendo os valores dos demais fixos. No extremo oposto, o projeto fatorial completo permite analisar a influência de todas as combinações de valores de todos os fatores nas variáveis de resposta. Uma abordagem intermediária é o projeto fatorial fracionário, onde alguns fatores permanecem fixos, enquanto outros são estudados em todas as combinações possíveis. Outro aspecto importante a ser considerado no design de experimentos é a replicação, que determina o número de repetições do experimento que serão executadas, aumentando a confiabilidade e reduzindo o impacto dos erros nas variáveis de resposta.

A Tabela 4.4 resume o projeto de experimento para o CMFog$_H$, apresentando os fatores e seus possíveis níveis, bem como as variáveis de resposta. O fator "Tecnologia de Acesso" terá três níveis: WiFi, 4G e 5G. Os parâmetros de largura de banda e latência das tecnologias já foram apresentados anteriormente. Os níveis do fator "Política" são compostos pelas políticas de migração P2 e P3, que utilizam gatilhos baseados no limite de cobertura e no tempo estimado de permanência, respectivamente. Além disso, para os experimentos, a política LL foi adicionada como um nível do fator "Política". Essa política de migração foi incluída para servir como uma base de comparação, uma vez que faz parte do pacote do MyiFogSim. A LL não realiza previsões de mobilidade e utiliza o mesmo gatilho da política P2, ou seja, determina o início de uma migração com base na proximidade com os limites da área de cobertura.

O experimento executado será do tipo fatorial completo, ou seja, serão formados cená-

rios de simulação através da combinação entre todos os níveis das três variáveis de fatores selecionadas para o experimento, resultando em 36 cenários. Cada cenário de simulação do experimento será executado dez vezes e as variáveis de respostas serão representadas por um média e um intervalo de confiança de 95%.

Tabela 4.4: Fatores e variáveis de respostas para construção do fatorial de experimentos de simulação do CMFog$_H$.

Fatores			**Variáveis de resposta**
Tamanho da VM	**Políticas**	**Tecnologia de Acesso**	
pequena	LL	WiFi	Latência
média	P2	4G	Migração
grande	P3	5G	Tuplas geradas
extra-grande			Tuplas perdidas

A aplicação utilizada nos experimentos de simulação do CMFog corresponde a um serviço de eco, no qual o componente m_{app} da aplicação será alocado nos dispositivos dos usuários, enquanto o componente f_{app} será encapsulado em uma VM, alocada nas cloudlets da camada de Névoa. Embora o CMFog seja uma proposta genérica para migração de conteúdo, optou-se pela aplicação simples de um servidor de eco para os experimentos, a fim de permitir uma avaliação constante da latência entre o usuário e sua respectiva VM.

Nos experimentos, cada usuário possui apenas uma VM, e cada VM pertence exclusivamente a um único usuário. Em outras palavras, os experimentos não consideram o compartilhamento de conteúdo entre os usuários. Essa configuração foi escolhida para simplificar a análise e focar na eficácia do CMFog na redução da latência individualmente.

4.5 Resultados

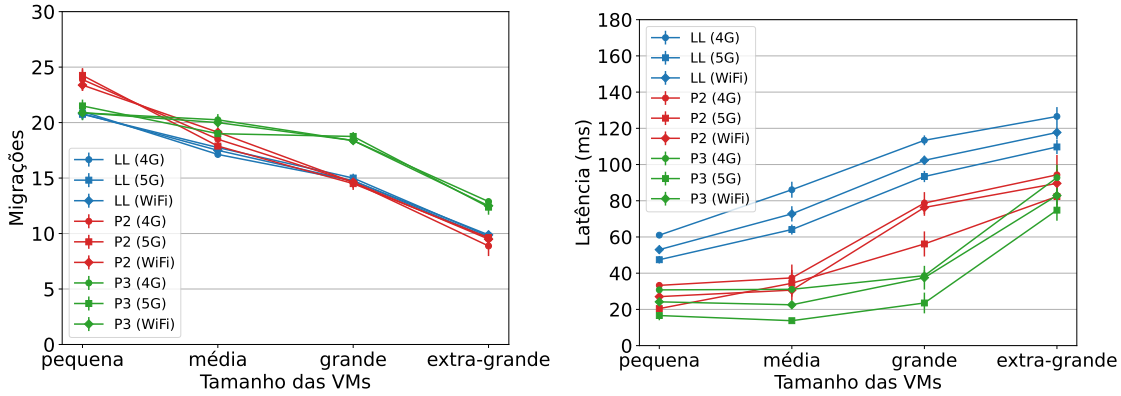
Os resultados obtidos pelo CMFog$_H$ serão apresentados em duas fases distintas. Na primeira fase, serão expostos os resultados referentes aos experimentos descritos nas seções anteriores. Em seguida, na segunda fase, serão apresentados resultados provenientes de uma análise adicional, mantendo todas as definições previamente estabelecidas, mas modificando alguns parâmetros para investigar detalhes adicionais.

4.5.1 Experimento base

Os resultados da média de migração do CMFog$_H$ são apresentados no gráfico da Figura 4.13a. Os cenários que empregam as políticas LL e P2 obtiveram médias de migração semelhantes em praticamente todos os cenários, exceto aqueles com VMs pequenas. O *trigger* de ambas as políticas monitoram a condição da proximidade do dispositivo com o limite da área de cobertura, o que aumenta a probabilidade de decisões semelhantes considerando condições parecidas.

Os experimentos com a política P3 mostraram um maior número de migrações em relação às outras duas políticas. O *trigger* baseado no tempo estimado de permanência do usuário permite que os eventos de migração sejam iniciados em um tempo proporcional

ao tamanho da VM, flexibilizando o início do processo e possibilitando um número maior de eventos de migrações.



(a) Média de migração (b) Média de latência

Figura 4.13: Gráficos com os resultados dos experimentos de simulação das métricas de latência e migração.

As médias de latência obtidas nos experimentos estão representadas no gráfico da Figura 4.13b. Observa-se que as maiores latências foram obtidas nos experimentos que utilizaram a política LL. A principal característica da LL é definir o destino de migração com base apenas na menor latência em um determinado instante de tempo. A política não considera a mobilidade do usuário, ou seja, não realiza predição de mobilidade, o que justifica as maiores médias de latência uma vez que as decisões são tomadas com foco apenas no momento.

Ao considerar a mobilidade do usuário, as políticas P2 e P3 realizam decisões que levam em conta as possíveis futuras localizações do usuário, visando selecionar uma cloudlet mais próxima e, consequentemente, proporcionar menor latência. De forma geral, os resultados dos cenários que adotam a política P3 demonstram menor latência em comparação com aqueles que utilizam a política P2. Esse desempenho superior da política P3 sugere que a estratégia empregada pelo trigger é mais eficaz na identificação dos momentos ideais para iniciar o processo de migração das VMs. No entanto, nos cenários com VMs de tamanho extra-grande, observa-se um aumento mais acentuado na latência, independentemente da política de migração utilizada

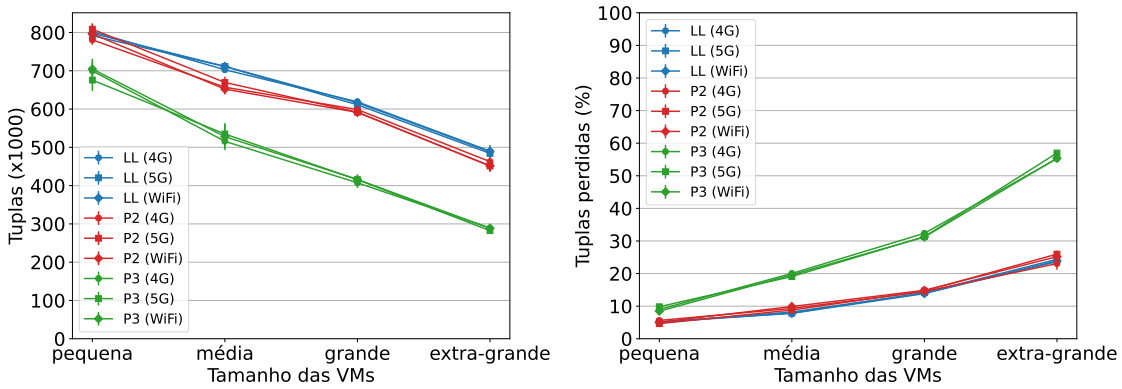
É importante observar que os eventos de migração desempenham um papel significativo na redução da latência percebida pelo usuário. A diminuição no número de migrações, resultante do aumento do tamanho das VMs, está diretamente correlacionada com o aumento acentuado nas médias de latência nos cenários com VMs extra-grandes. Para uma melhor compreensão, é importante considerar duas premissas fundamentais: i) o tempo de migração da VM é diretamente proporcional ao seu tamanho; e ii) o tempo de acesso à VM depende da proximidade do usuário, ou seja, latências menores serão alcançadas quando a VM estiver mais próxima do dispositivo do usuário.

O *trigger* utilizado pelas estratégias LL e P2 inicia o processo de tomada de decisão apenas quando o usuário entra na zona de migração, ou seja, quando ele se aproxima do limite da área de cobertura da cloudlet à qual está associado. Com o início fixo do processo

de tomada de decisão, a eficácia dessas estratégias pode começar a ser reduzida à medida que o tempo necessário para realizar a migração da VM aumenta, podendo até ultrapassar o tempo em que o usuário permanece associado à cloudlet. Em outras palavras, quando a migração é concluída, o usuário pode já ter realizado um novo *hand-off*, se afastando da cloudlet que recebeu a VM, o que reduz a eficácia do evento de migração.

Ao considerar o tempo médio de migração da VM e a estimativa de permanência do usuário em uma cloudlet específica, o *trigger* utilizado na Política P3 permite um início flexível do processo de tomada de decisão. Essa flexibilidade em determinar o momento de iniciar a migração contorna as limitações observadas nas estratégias LL e P2, reduzindo o impacto que o tamanho das VMs exerce na média de latência. No entanto, mesmo com a Política P3, nos cenários com VMs extra-grandes, a métrica de latência apresentou um aumento mais significativo em comparação com os outros tamanhos de VMs. A análise dos resultados revelou que, em cenários em que o tempo necessário para migrar o conteúdo se torna tão alto a ponto de exceder o tempo de permanência do usuário na cloudlet, as migrações perdem eficiência na tarefa de manter a proximidade entre conteúdo e usuário, resultando em um aumento na latência.

O número total de tuplas geradas é diretamente influenciado pelo tempo de *uptime* das VMs, ou seja, o período em que as VMs estão disponíveis e não estão em processo de migração. Por outro lado, o número total de migrações e o tempo de transferência das VMs são fatores que reduzem a disponibilidade das VMs, o que consequentemente diminui o número de mensagens trocadas entre as entidades da simulação.



(a) Média de tuplas geradas

(b) Média de tuplas perdidas

Figura 4.14: Gráficos com os resultados dos experimentos de simulação das métricas de tuplas geradas e tuplas perdidas.

A Figura 4.14a ilustra o total de tuplas geradas em cada cenário de simulação. Observou-se que os cenários que utilizam a política de migração P3 apresentaram os menores valores de tuplas geradas. Esse padrão ocorre devido ao fato de que esses cenários realizam mais migrações, o que implica em menos tempo de disponibilidade das VMs, resultando em uma redução no total de tuplas geradas. É válido ainda observar, que o tempo de migração de uma VM é composto apenas por atrasos provenientes das camadas de Névoa e Nuvem, e, portanto, a tecnologia de acesso não influencia os resultados do total de tuplas geradas.

O gráfico da Figura 4.14b exibe a porcentagem de tuplas perdidas nos experimentos realizados. Uma tupla é considerada perdida quando a requisição do usuário não consegue alcançar sua respectiva VM, possivelmente devido a indisponibilidade causada por um evento de migração, por exemplo. As políticas LL e P2 demonstram porcentagens semelhantes de tuplas perdidas, o que decorre do número muito próximo de migrações entre essas políticas. Por outro lado, a política P3 exibe as maiores médias de tuplas perdidas, resultado direto do menor tempo de disponibilidade das VMs nessa política.

Ao analisar os resultados, observa-se que as métricas de migração, tuplas geradas e tuplas perdidas foram pouco influenciadas pelas diferentes tecnologias de acesso utilizadas nos experimentos, mantendo-se, na maioria das vezes, dentro da margem de erro. Isso sugere que as tecnologias de acesso têm uma influência limitada nessas métricas, uma vez que as variações em seus parâmetros não afetam diretamente a infraestrutura de rede nas camadas de Névoa e Nuvem. No entanto, as tecnologias de acesso têm um impacto mais significativo na latência, já que os atrasos na comunicação são diretamente afetados pela latência do meio de comunicação entre o dispositivo e a infraestrutura de rede.

Os resultados de latência, conforme apresentados no gráfico 4.13b, destacam que, ao comparar os cenários com a mesma política, aqueles que utilizam 4G têm a maior latência, enquanto os que usam 5G têm a menor. Isso indica que as tecnologias de acesso têm influência nas métricas de latência, com o 4G apresentando o maior atraso e o 5G o menor. No entanto, é importante ressaltar que essas tecnologias não interferem nas demais métricas, já que estas monitoram variáveis afetadas apenas por agentes internos da infraestrutura de Névoa.

4.5.2 Variação de largura de banda das cloudlets

De maneira geral, as simulações dos cenários que utilizaram política P3 apresentaram as menores médias de latência. Isso ocorreu devido ao maior número de migrações, o que também resultou em menores médias de tuplas geradas e maiores médias de tuplas perdidas. No entanto, nos cenários com VMs de tamanho extra-grande, observou-se um aumento mais expressivo na métrica de latência em comparação com os aumentos observados nos outros tamanhos de VMs.

Após analisar os resultados dos experimentos, podemos concluir que o CMFog$_H$, independentemente da política de migração utilizada, mostra uma perda de eficácia em manter a proximidade entre conteúdo e usuário à medida que o tamanho da VM aumenta. Quando o tempo necessário para realizar uma migração excede o tempo em que o usuário permanece associado à cloudlet destino, os eventos de migração perdem a eficácia em manter o conteúdo próximo ao usuário, resultando em um aumento na média de latência.

Apesar da flexibilidade na determinação do início do processo de migração, os cenários com VMs extra-grandes que utilizam a política P3 apresentaram os maiores aumentos na média de latência. Para obter mais informações sobre esse comportamento identificado no primeiro conjunto de experimentos do CMFog$_H$, foram planejados novos cenários com a adição de configurações de cloudlet com diferentes larguras de banda de rede. Essa variação na largura de banda busca obter mais detalhes sobre como o CMFog é impactado

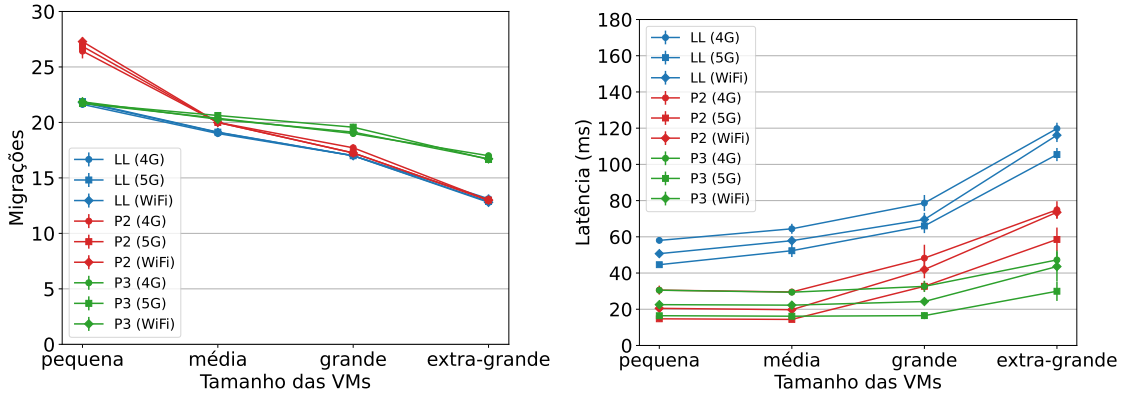
pelo aumento do tamanho da VMs.

A Tabela 4.5 apresenta os tipos de cloudlets com suas respectivas configurações usadas no novo conjunto de experimentos. A cloudlet do tipo T2 corresponde ao mesmo tipo usado nas simulações anteriores. As cloudlets T1 possuem uma largura de banda inferior, enquanto as T3 possuem largura de banda superior à cloudlet T2.

Tabela 4.5: Configurações dos tipos de cloudlets utilizadas nos experimentos adicionais.

Cloudlet	**Largura de banda (Mbps)**
Tipo 1 (T1)	40 - 80
Tipo 2 (T2)	130 - 170
Tipo 3 (T3)	240 - 280

Os resultados dos experimentos utilizando a cloudlet do tipo T3, com maior largura de banda de rede, estão representados nos gráficos da Figura 4.15. No Gráfico 4.15a, que mostra os resultados da métrica de migrações, observa-se que, apesar das variações numéricas, o comportamento entre os diferentes cenários é semelhante ao observado nos experimentos anteriores. Considerando os valores numéricos, nota-se um pequeno aumento na média de migração, principalmente nos cenários com VMs extra-grandes. Esse comportamento é consequência do menor tempo necessário para realizar as migrações, uma consequência direta do aumento da largura de banda.



(a) Média de eventos de migração

(b) Média de latência

Figura 4.15: Gráficos com os resultados dos experimentos de simulação das métricas de latência e migração e com cloudlet T3.

Os resultados para a métrica de latência nos experimentos com a cloudlet T3 mostram uma redução na média em quase todos os cenários, conforme mostrado na Figura 4.15b. A redução da latência observada nos cenários com VMs de tamanho pequeno foi menos expressiva devido ao pouco tempo necessário para realizar a migração dessas VMs. Por outro lado, os experimentos com VMs de tamanho médio, grande e extra-grande apresentaram reduções mais significativas na média de latência, independentemente da política de migração utilizada no cenário.

Após analisar os resultados dos experimentos anteriores, observou-se um aumento significativo na latência nos cenários com VMs extra-grandes. No entanto, ao introduzir um aumento na largura de banda de rede por meio das cloudlets T3, a latência observada

nesses cenários foi consideravelmente reduzida. A redução obtida nos experimentos com a política P2 foi de cerca de 10%, enquanto nos cenários com a política P3, a redução chegou a 40%.

Os resultados das métricas de tuplas indicam um aumento na quantidade de tuplas geradas e uma redução na porcentagem de tuplas perdidas. Esse comportamento pode ser atribuído ao aumento da largura de banda, que resultou em uma redução no tempo de migração das VMs e, consequentemente, aumentou o tempo de disponibilidade delas. Com o maior *uptime* das VMs, mais tuplas puderam ser geradas, ao mesmo tempo em que a porcentagem de tuplas perdidas foi reduzida.

Os experimentos adicionais forneceram uma compreensão mais detalhada do comportamento do CMFog$_H$. Ao comparar os resultados dos cenários com cloudlets T2, referentes às simulações iniciais, e T3, observamos que o aumento da largura de banda foi eficaz na redução da latência na maioria dos casos. Por outro lado, a redução da largura de banda nas cloudlets T1 evidenciou os problemas identificados nos experimentos com as cloudlets T2. Como os resultados das cloudlets T1 não trouxeram novas perspectivas, apenas reforçaram os comportamentos já identificados nos experimentos com as cloudlets T2, os gráficos correspondentes não foram incluídos na análise.

4.5.3 Discussão

Através da análise dos resultados dos experimentos com o CMFog$_H$, ficou evidente a importância da predição de mobilidade nas estratégias de migração de conteúdo na borda da rede. A Política LL, que não considera aspectos de mobilidade em seu processo de decisão, apresentou as maiores médias de latência em comparação com as outras políticas utilizadas nos experimentos. Em contrapartida, as políticas P2 e P3, que utilizam informações sobre as futuras localizações dos usuários, mostraram-se mais eficientes ao realizar migrações que promovem uma maior aproximação entre conteúdo e usuário.

Especificamente, a Política P3 obteve as menores médias de latência entre todas as políticas, exceto nos cenários com VMs extra-grandes, onde a Política P2 também alcançou resultados próximos. A eficácia da Política P3 pode ser atribuída ao seu *trigger*, baseado na estimativa do tempo de permanência dos usuários nas cloudlets. Esta abordagem se mostrou mais robusta e capaz de tomar decisões de migração de maneira mais proativa, aumentando o número de eventos de migração e, consequentemente, diminuindo a latência média ao aproximar mais eficientemente o conteúdo do usuário.

No entanto, a abordagem da Política P3 exige mais informações sobre a rotina do usuário, o que nem sempre é viável. A indisponibilidade de dados do usuário ou limitações relacionadas à privacidade podem impedir a coleta dessas informações detalhadas, restringindo a aplicação prática dessa estratégia em certos cenários.

Os experimentos também mostraram que o CMFog$_H$ enfrenta desafios em garantir a proximidade entre conteúdo e usuários. Essa dificuldade foi identificada em cenários nos quais o tempo necessário para realizar a migração do conteúdo é significativo a ponto de o usuário mudar de área de cobertura antes que o processo seja concluído, ou seja, o usuário já não está associado à cloudlet selecionada como destino da migração. Além disso, é importante destacar que, além de perder eficácia em garantir a proximidade do conteúdo

com o usuário, os recursos utilizados pela infraestrutura para realizar a transferência das VMs também são parcialmente desperdiçados.

Embora o aumento da largura de banda das cloudlets tenha mostrado uma redução na latência, em um cenário real não é possível garantir que a Névoa possa sempre oferecer recursos computacionais e largura de banda adicionais para atender aos requisitos dos usuários. Portanto, é importante que o CMFog seja capaz de se adaptar e oferecer alternativas para minimizar os efeitos de cenários que dificultem a manutenção da proximidade entre conteúdo e usuário. Nesse contexto, a próxima versão do CMFog, o CMFog$_V$, busca explorar os múltiplos níveis da camada de Névoa para proporcionar maior adaptabilidade no processo de decisão.

Capítulo 5

CMFog$_V$: Névoa multi-nível

Na busca pela melhoria das estratégias de migração de conteúdo na Computação em Névoa, este capítulo introduz o CMFog$_V$ [5], uma segunda versão do CMFog que incorpora um ambiente de Névoa multi-nível e inclui a possibilidade do CMFog realizar migrações verticais. Enquanto o CMFog$_H$ focava em estratégias horizontais, ou seja, migrações entre cloudlets do mesmo nível na Névoa, o CMFog$_V$ inova ao introduzir a capacidade de realizar migrações verticais, ou seja, entre diferentes níveis da arquitetura de Névoa. Essa evolução visa abordar os desafios identificados durante os experimentos com o CMFog$_H$, especialmente em situações em que, devido à velocidade do usuário ou ao tamanho da VM, a estratégia não conseguia manter conteúdo e usuário próximos o suficiente para garantir uma baixa latência de comunicação, resultando na degradação da qualidade de serviço.

A arquitetura de Computação em Névoa, conforme definida pelo modelo da NIST, é composta por três camadas: nuvem, névoa e dispositivos de usuário. A camada de Névoa, fundamental para a interação com dispositivos próximos aos usuários, pode ser ainda subdividida em múltiplos níveis. Os níveis mais próximos à borda da rede oferecem menor latência, porém com cobertura limitada, enquanto os níveis mais distantes proporcionam maior cobertura e recursos computacionais, mas com latências mais elevadas. Essa dinâmica entre os níveis da Névoa estabelece um *trade-off* entre latência e área de cobertura, no qual o CMFog$_V$ busca explorar por meio de migrações verticais.

Ao oferecer a capacidade de realizar migrações entre os diferentes níveis da Névoa, o CMFog$_V$ abre novas perspectivas para o processo de decisão de migração de conteúdo. A estrutura deste capítulo segue a seguinte organização: inicialmente, apresentamos a motivação para a evolução do CMFog$_H$ para o CMFog$_V$, destacando as limitações identificadas e os aspectos que buscamos explorar. Em seguida, descrevemos detalhadamente o funcionamento do CMFog$_V$, delineando como as migrações verticais são incorporadas no processo de decisão. Posteriormente, apresentamos os experimentos conduzidos para validar o desempenho do CMFog$_V$, comparando-o com o CMFog$_H$. Por fim, discutimos os resultados obtidos e suas implicações para aprimorar a eficácia das estratégias de migração na Computação em Névoa.

5.1 Modelo e Política

O modelo adotado no CMFog$_V$ permanece inalterado em relação ao utilizado no CMFog$_H$ e corresponde às definições já apresentadas na seção 4.1. Com a introdução da migração vertical no CMFog$_V$, uma nova política foi incorporada ao modelo do CMFog.

A Política 4 representa uma estratégia que realiza migrações verticais. A Figura 5.1 ilustra o cenário em que a migração realizada para acompanhar a mobilidade do usuário não acontece entre as cloudlets do mesmo nível, ou seja, a VM é transferida para uma cloudlet localizada em um nível da Névoa diferente do atual, nesse exemplo, um nível superior. As cloudlets dos níveis superiores são capazes de oferecer uma maior área de cobertura, e nesse exemplo, a área de cobertura A_4 corresponde à soma das áreas A_1, A_2, e A_3.

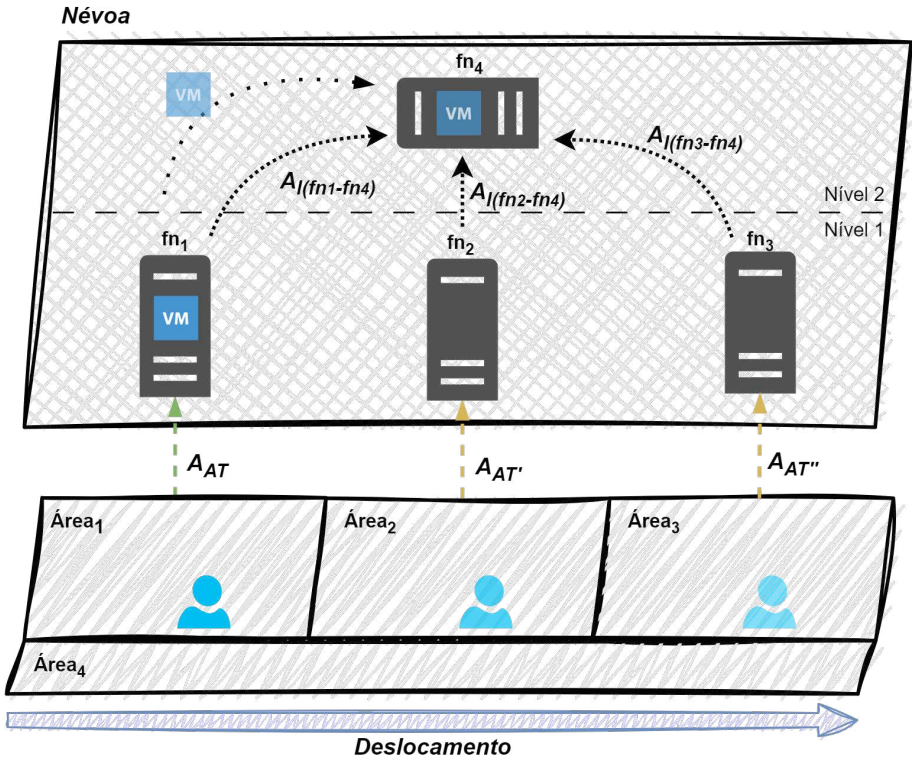


Figura 5.1: Esquematização da interação da Política 4 com um cenário de mobilidade do usuário com migração vertical.

Nas Políticas 2 e 3, a latência é representada pela Equação 5.1, onde $A_{I(fn_i - fn_j)}$ corresponde ao tempo para a requisição ser encaminhada de fn_i para fn_j e para a resposta da requisição retornar a fn_i. Para a Política 4, a mesma equação permanece válida. Entretanto, temos a restrição de que fn_j corresponde a um nó localizado em um nível superior da Névoa. Portanto, ao considerar que a VM do usuário foi migrada verticalmente para um nó fn qualquer, temos que, enquanto o usuário permanecer associado a qualquer uma das cloudlets, a latência L é dada pela Equação 5.2.

$$L = A_{AT} + A_{I(fn_i - fn_j)} + A_{FP} \tag{5.1}$$

$$L = A_{AT} + A_{I(fn_i - fn_j)} + A_{FP} \mid fn_j \text{ é um nó de um nível superior} \quad (5.2)$$

A Política 4 utiliza um *trigger* que monitora a variação da média de latência dentro de uma janela de tempo de tamanho W. A média de latência é calculada utilizando o modelo de Médias Móveis Exponencialmente Ponderadas (*Exponentially Weighted Moving Average* - EWMA). Este modelo emprega o método de suavização exponencial, permitindo ajustar o peso que os valores mais recentes exercem sobre o cálculo da média. Neste cenário, o modelo EWMA oferece uma boa caracterização do cálculo da média de latência, uma vez que, no contexto de predição de mobilidade, os valores mais recentes podem ser indicadores mais precisos em relação à tendência de deslocamentos dos usuários, embora os valores antigos também tenham sua importância.

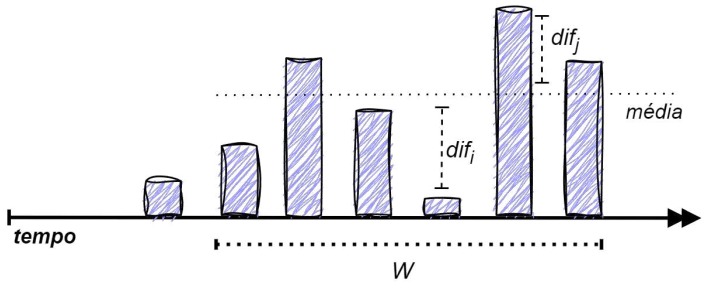


Figura 5.2: Representação dos aspectos considerados pelo *trigger* utilizado no processo de decisão de migração vertical.

A Figura 5.2 exemplifica o funcionamento do *trigger* da Política 4. Nessa abordagem, os últimos W valores da média EWMA são empregados para verificar se a variação nesse intervalo ultrapassa um determinado *threshold* T_L. Esse processo envolve o cálculo da média dos valores que compõem o intervalo W, seguido pela análise para identificar se alguma entrada de W possui uma diferença absoluta dif, em relação a média calculada, no qual $dif \geq T_L$. Caso essa condição seja atendida, é sinalizada a necessidade de iniciar uma migração vertical.

5.2 Componentes

O CMFog$_V$, uma extensão do CMFog$_H$, introduz um componente adicional chamado *V-Migration*, desempenhando um papel fundamental no gerenciamento do processo de decisão de migração vertical. A Figura 5.3 ilustra um diagrama de componentes, destacando a interação entre os componentes do CMFog$_V$. No diagrama, observamos como o novo componente interage com os componentes já existentes no CMFog, como *DeTrigger*, *DeMADM*, e *Predmobi*, os quais mantêm seus papéis inalterados.

A introdução do *V-Migration* confere ao CMFog$_V$ a capacidade de realizar migrações verticais, ou seja, migrar VMs entre os níveis da Névoa. Esse novo recurso é fundamental para estender a flexibilidade do CMFog e permitir que a estratégia tenha um leque maior de possibilidades para lidar com situações em que o CMFog$_H$ não conseguia tomar decisões que oferecessem níveis aceitáveis de latência. O *trigger* da migração vertical monitora

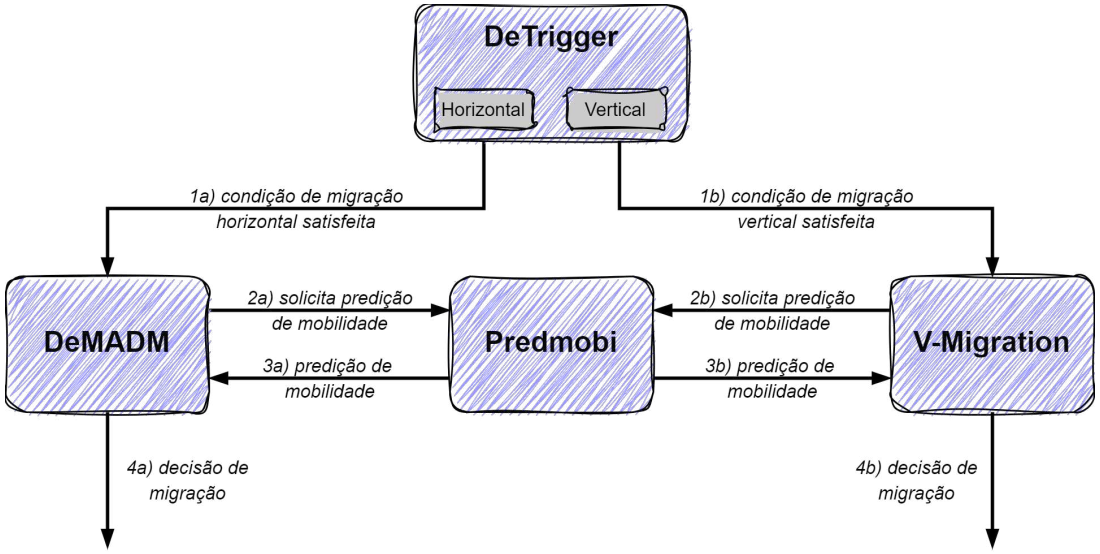


Figura 5.3: Diagrama de componentes e comunicação do CMFog_V.

variáveis específicas para identificar a necessidade de migrar o conteúdo entre os níveis da infraestrutura de Névoa.

Os eventos de migração desempenham um papel essencial na manutenção da proximidade do conteúdo à medida que os dispositivos do usuário se afastam de suas respectivas máquinas virtuais. Essas migrações envolvem a transferência de uma VM de uma cloudlet para outra e podem ser categorizadas como horizontais ou verticais. Migrações horizontais envolvem a movimentação de VMs entre cloudlets pertencentes ao mesmo nível dentro da Névoa.

Algorithm 3: Algoritmo do funcionamento do *trigger* para migração vertical.

Input: Usuário U, Tamanho da janela de monitoramento W, Registro de latência R_L, *Threshold* de latência T_L

```
1  VM ← máquina virtual do usuário U
2  R_W(L) ← últimas W entradas de R_L
3  D_L ← maior diferença entre os valores contidos em R_W(L)
4  migVertical ← false
5  if VM.cooldown <= 0 then
6  |  if (VM.nivel = 1) AND (D_L >= T_L) then
7  |  |  migVertical ← true
8  |  end
9  |  if (VM.nivel = 2) AND (D_L <= T_L/2) then
10 |  |  migVertical ← true
11 |  end
12 end
13 if migVertical = true then
14 |  VMigration(VM,U)
15 end
```

A abordagem de migração vertical é definida pela Política 4, que utiliza um *trigger* específico para a migração vertical. O Algoritmo 3 descreve a sequência de etapas execu-

tadas por esse novo *trigger*. Como entrada, o algoritmo recebe um usuário U, o tamanho da janela de monitoramento W, o registro do histórico da média de latência R_L e um limite T_L. Inicialmente, entre as linhas 1 e 4, o algoritmo obtém informações adicionais a partir dos parâmetros de entrada. Com base no tamanho da janela de monitoramento, apenas os últimos W registros de latência são selecionados do histórico, obtendo $R_W(L)$, e então é calculado D_L, que representa a maior diferença entre quaisquer dois registros de latência contidos em $R_W(L)$.

A sequência de verificações para determinar se as condições para iniciar um processo de tomada de decisão de migração vertical foram satisfeitas é realizada nas linhas 5 a 12. A primeira condição verifica se a VM em questão está fora do período de *cooldown*, ou seja, se a VM não realizou uma migração vertical recentemente. Se esta condição for atendida, com base no nível da Névoa onde a VM está alocada, é verificado se a relação entre D_L e o limite T_L caracteriza uma condição para iniciar o processo de decisão, seja para migração da VM para um nível inferior ou superior. Após as verificações, nas linhas 13 a 15, se a *flag migVertical* estiver verdadeira, o *trigger* realiza uma chamada para o componente V-Migration para iniciar o processo de decisão de migração. Se alguma das condições não for atendida, o *trigger* apenas retornará a um estado de monitoramento.

O Algoritmo 4 descreve as etapas realizadas pelo componente *V-Migration* para conduzir o processo de migração vertical. O primeiro passo, na linha 1, consiste em verificar se há alguma migração horizontal em andamento; caso afirmativo, o processo de decisão é adiado para o próximo intervalo de tempo. Se não houver migração horizontal em andamento, o componente determina se a migração vertical deve ocorrer para um nível superior ou inferior, como indicado nas linhas 2 a 8. Se a migração vertical for direcionada para um nível superior, o algoritmo seleciona a cloudlet "pai" de onde a VM está atualmente alocada como destino de migração. Por outro lado, se a migração for para um nível inferior, o V-Migration consulta o Predmobi para determinar uma cloudlet do mesmo *"branch"* da cloudlet à qual o usuário está associado. As linhas 9 a 11 verificam se a cloudlet selecionada é capaz de receber a VM; se for o caso, o processo de migração é iniciado.

Essas adições ao CMFog buscam ampliar sua capacidade de lidar com cenários dinâmicos, oferecendo uma abordagem mais sólida para manter a proximidade entre o conteúdo e o usuário. A combinação entre as migrações horizontais e verticais busca proporcionar uma estratégia mais flexível e adaptável na tentativa de melhorar a experiência do usuário, principalmente aspectos afetados pela latência.

5.3 Simulação

Os experimentos destinados a validar a estratégia do CMFog$_V$ também serão conduzidos por meio do simulador MyiFogSim. Os componentes adicionais específicos do CMFog$_V$ foram devidamente implementados e incorporados ao pacote já existente do CMFog no simulador. A Tabela 5.1 apresenta os parâmetros das conexões estabelecidas entre as entidades da infraestrutura dos experimentos. De maneira geral, os valores permanecem inalterados em relação aos experimentos anteriores com o CMFog$_H$, com a única exceção

Algorithm 4: Algoritmo do processo de tomada de decisão do componente V-Migration.

```
   Input: Usuário U, Máquina virtual VM
 1 if VM.mig = false then
 2 |   sentidoMig ← identifica sentido da migração vertical
 3 |   if sentidoMig = cima then
 4 |   |   migDest ← VM.cloudlet.pai
 5 |   end
 6 |   else
 7 |   |   migDest ← consulta Predmob para identificar cloudlet do nível VM.nivel − 1
   |   |     que pertence ao mesmo branch da cloudlet no qual U está associado
 8 |   end
 9 |   if migDest tem recursos computacionais para receber VM then
10 |   |   migrar(VM, migDest)
11 |   end
12 end
13 else
14 |   Posterga tomada de decisão para próxima unidade de tempo
15 end
```

sendo a adição de uma nova conexão para associar as cloudlets de diferentes níveis.

Tabela 5.1: Valores dos parâmetros de conexões entre as entidades da simulação do CMFog_V.

Entidade A - Entidade B	**Latência (ms)**	**Largura de Banda (Mbps)**
Ponto de Acesso - Cloudlets (Nível 1)	1 - 3	280 - 320
Cloudlet (Nível 1) - Cloudlet (Nível 2)	20 - 30	65 - 85
Cloudlet (Nível 2) - Nuvem	30 - 40	130 - 170

Para os experimentos conduzidos com o CMFog_V, optamos por utilizar exclusivamente a tecnologia de acesso 5G para conectar os dispositivos à infraestrutura. Essa decisão foi tomada com base na análise dos resultados obtidos em experimentos anteriores, nos quais foi identificada uma baixa influência das tecnologias de acesso no comportamento geral das estratégias de migração. Além disso, os tamanhos das máquinas virtuais permaneceram inalterados, mantendo os quatro tipos de VMs, são elas: pequeno (100-250 MBs), médio (400-500 MBs), grande (900-1000 MBs) e extra-grande (1900-2000 MBs).

5.3.1 Métricas

Os resultados dos experimentos com o CMFog_V serão avaliados com base em um conjunto de métricas cuidadosamente selecionadas para fornecer o máximo de informações sobre os experimentos realizados. As métricas escolhidas para esses experimentos são as seguintes: latência da aplicação, número de migrações, tempo de emparelhamento e tempo de segundo nível. Vale ressaltar que as duas primeiras métricas, latência e número de migrações, já foram utilizadas nos experimentos com o CMFog_H e, portanto, já foram definidas anteriormente.

Uma das novas métricas introduzidas é o tempo de emparelhamento, que quantifica a porcentagem de tempo durante o qual a VM do usuário permanece alocada no mesmo *"branch"* que a cloudlet à qual o dispositivo do usuário está associado. O termo *"branch"* denota que, se a VM estiver no primeiro nível da Névoa, ela estará alocada onde o usuário está conectado. Por outro lado, se a VM estiver no segundo nível, significa que ela está alocada em uma cloudlet "pai" em relação à qual o usuário está associado atualmente.

A segunda métrica adicional é o tempo no segundo nível, que representa a porcentagem do tempo durante o qual uma máquina virtual permanece alocada em uma cloudlet do segundo nível da camada de Névoa em relação ao tempo total da simulação.

As métricas de latência e número de migrações, que têm uma correlação significativa, fornecem informações cruciais sobre o impacto das decisões tomadas pelo CMFog$_V$ no processo de redução da latência da aplicação. As métricas de emparelhamento e tempo no segundo nível complementam essa análise, oferecendo informações sobre a proximidade entre conteúdo e usuário, bem como sobre o efeito das migrações verticais, introduzidas nesta versão do CMFog.

Através dessas quatro métricas, será possível obter um conjunto abrangente de dados que permitirá uma compreensão mais profunda do desempenho do CMFog$_V$ em diversos cenários de simulação. Essas métricas foram escolhidas estrategicamente para oferecer informações importantes relacionadas à eficácia e a adaptabilidade do CMFog.

5.3.2 Experimentos

O CMFog$_V$ representa a versão mais completa do objeto de pesquisa, consolidando todos os conceitos discutidos até esse ponto. O principal propósito é investigar os impactos das migrações verticais no processo de migração de conteúdo em um ambiente de Névoa multi-nível. Ao incorporar uma estratégia que possibilita a migração tanto horizontal quanto vertical de VMs, o CMFog$_V$ busca explorar o *trade-off* entre latência e área de cobertura entre os diferentes níveis da Névoa.

A presença de múltiplos níveis na camada de Névoa confere maior flexibilidade ao processo de decisão de migração. Isso permite explorar o equilíbrio entre as latências mais baixas nos níveis inferiores e a maior área de cobertura nas camadas superiores. Uma área de cobertura mais ampla possibilita uma redução na necessidade de migrações para manter o acesso ao conteúdo do usuário, ao custo de maior latência. No entanto, pode haver situações em que a volatilidade de migração nos níveis inferiores resulte em uma média não tão diferente das obtidas na camada superior. Ainda assim, esse *trade-off* oferecido pelos níveis da Névoa também aumenta a complexidade do processo de decisão de migração.

A segunda fase dos experimentos utiliza o CMFog$_V$, uma versão do modelo com modificações específicas para lidar com os níveis na camada de Névoa. Essas modificações incluem a introdução do componente V-Migration e um novo *trigger* para monitorar condições para migração vertical.

O funcionamento do CMFog$_V$ é esquematizado no diagrama da Figura 5.4. O lado direito do diagrama representa a sequência de eventos para o processo de decisão de migração horizontal, discutido previamente no Capítulo 4. Por outro lado, o lado esquerdo

do diagrama representa a sequência de eventos para a migração vertical. O processo de decisão é iniciado pelo monitoramento realizado pelo componente DeTrigger, que, ao detectar condições que indiquem a necessidade de iniciar um processo de tomada de decisão de migração vertical, aciona o componente V-Migration. A primeira ação do V-Migration é identificar a direção da migração, ou seja, se a VM deve ser migrada para um nível acima ou abaixo da cloudlet em que está atualmente alocada. Uma vez que o nível é identificado, com auxílio das informações de predição de mobilidade, o componente determina a cloudlet mais apropriada para a migração. Após a confirmação da necessidade de migração e a seleção da cloudlet de destino, o DeTrigger é colocado no modo de monitoramento e os procedimentos de transferência da VM são iniciados.

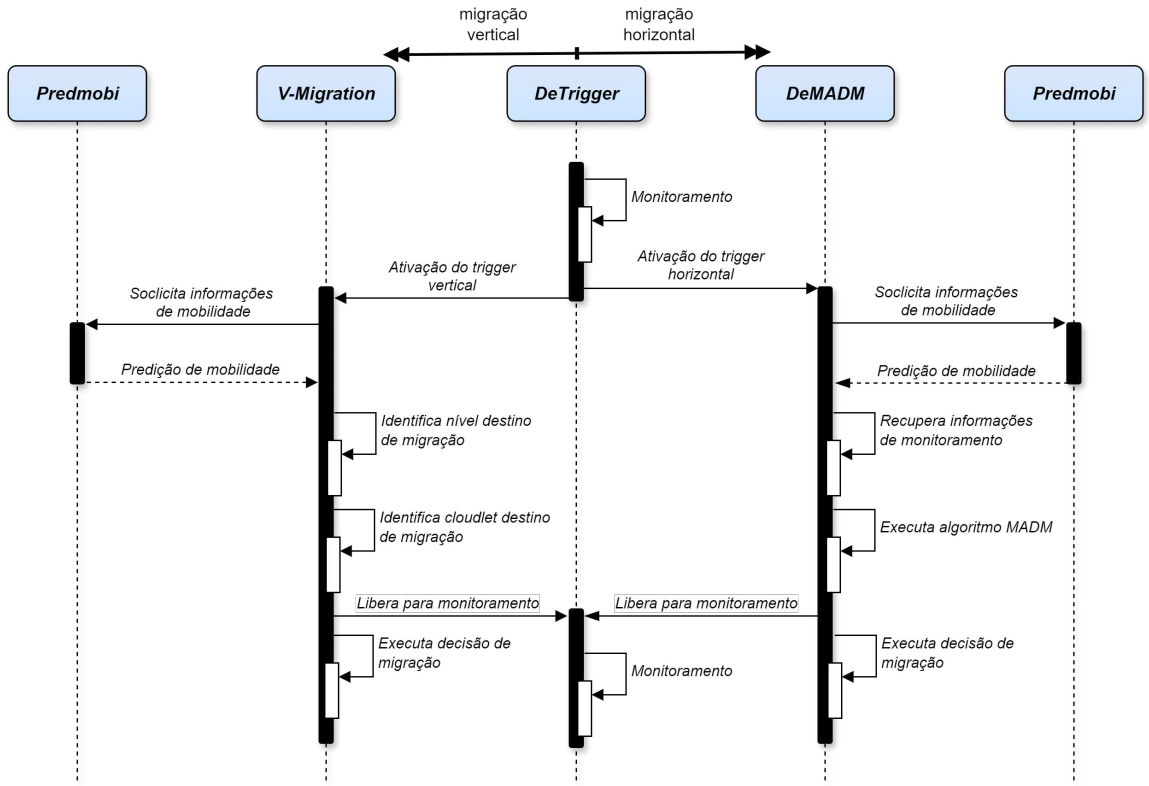


Figura 5.4: Diagrama de sequência que ilustra o funcionamento do CMFog_V.

Essas adições ao modelo visam especificamente lidar com as características únicas da migração vertical, oferecendo uma abordagem mais completa para o processo de decisão em um ambiente de Névoa multi-nível. O novo *trigger* e o componente V-Migration são elementos que desempenham papéis importante na identificação da necessidade de migração vertical e na execução efetiva desse processo, respectivamente.

5.3.3 Definições de simulação

Com o objetivo de comparar e validar o impacto da migração vertical no CMFog_V, é essencial planejar cenários de simulação que permitam uma análise clara e representem uma variedade de situações. Os experimentos do CMFog_V apresentam algumas diferenças em relação aos conduzidos anteriormente no CMFog_H.

Para garantir a comparação dos novos resultados com os anteriores, incluiremos cenários equivalentes no novo conjunto de experimentos. Para isso, conduziremos simulações em cenários que realizam apenas migrações horizontais em cada um dos dois níveis da Névoa. Para maior clareza, esses cenários serão denominados $P3_1$ para migração horizontal no primeiro nível e $P3_2$ para migração horizontal no segundo nível.

Os cenários que envolvem a política de migração vertical do $CMFog_V$ serão caracterizados por duas variáveis adicionais: o tamanho da janela de monitoramento W e o *threshold* de latência T_L. Essas duas variáveis irão assumir diferentes valores durante os experimentos. A variação dos valores busca investigar como o histórico de latência afeta o processo de decisão e como aplicações com diferentes níveis de tolerância à variação de latência são impactadas pelas decisões do $CMFog_V$.

As Tabelas 5.2 e 5.3 fornecem uma sumarização do planejamento para os experimentos do $CMFog_V$. O experimento será dividido em dois grupos, cada um com um planejamento distinto. O primeiro, apresentado na Tabela 5.2, refere-se aos experimentos de controle, cujo planejamento assemelha-se bastante ao $CMFog_H$, servindo principalmente como um grupo de referência para comparação. O fator "Políticas" neste conjunto incluirá as variações da Política P3 e uma política sem migração. O fator Tecnologia de Acesso será fixado apenas com a tecnologia 5G.

Tabela 5.2: Fatores e variáveis de respostas para construção do fatorial de experimentos de simulação dos cenários de controle (sem migração vertical).

Fatores			**Variáveis de resposta**
Tamanho da VM	**Políticas**	**Tecnologia de Acesso**	
pequena	Sem migração	~~WiFi~~	Latência
média	$P3_1$	~~4G~~	Migração
grande	$P3_2$	5G	Tempo de emparelhamento
extra-grande			Tempo de segundo nível

O segundo conjunto de experimentos, detalhado na Tabela 5.3, incorpora migrações verticais. Neste cenário, serão introduzidos dois novos fatores: W, que representa o tamanho da janela de monitoramento, e T_L, que denota o *threshold* de latência. O fator Tecnologia de Acesso permanecerá constante em 5G ao longo de todo o experimento. Em ambos os grupos de planejamento de experimentos, as variáveis de resposta serão latência, migração, tempo de emparelhamento e tempo de segundo nível. Cada cenário de simulação será repetido dez vezes, e as variáveis de resposta serão representadas por uma média acompanhada de um intervalo de confiança de 95%.

Durante a análise dos resultados dos experimentos com o $CMFog_H$, observou-se que os cenários que utilizaram a Política 3 apresentaram as menores médias de latência. Com base nesse achado, optamos por utilizar apenas a Política 3, que realiza migrações com base no tempo estimado de permanência do usuário, como estratégia para gerenciar migrações horizontais nos experimentos do $CMFog_V$.

A camada de Névoa será composta por um total de dois níveis, sendo o primeiro composto por cloudlets com as configurações já apresentadas na Tabela 4.5. O segundo nível da Névoa será composto por cloudlets com o dobro da área de cobertura (totalizando

Tabela 5.3: Fatores e variáveis de respostas para construção do fatorial de experimentos de simulação do CMFog$_V$.

Fatores					**Variáveis de resposta**
Tamanho da VM	**Políticas**	**Tecnologia de Acesso**	**W**	**T$_L$**	
pequena	P4	~~WiFi~~	100	5	Latência
média		~~4G~~	200	10	Migração
grande		5G	300	20	Tempo de emparelhamento
extra-grande					Tempo de segundo nível

500 metros) e também com o dobro da latência e largura de banda de rede.

Algumas alterações foram realizadas em relação aos experimentos do CMFog$_H$ para direcionar a investigação para pontos específicos identificados na fase de análise anterior. A inclusão de cenários que contemplam o mesmo tipo de configuração dos experimentos do CMFog$_H$ tem como objetivo permitir a comparação justa dos resultados, mesmo que os parâmetros de simulação tenham sido modificados pontualmente. Para garantir que os resultados dos experimentos anteriores não estejam condicionados a parâmetros ou cenários específicos, foi gerado um novo conjunto de registros de mobilidade, também gerado através da ferramenta Hermoupolis. O mapa utilizado também pertence à cidade de Atenas, mas de uma região diferente, conforme apresentado na Figura 5.5. Nenhuma modificação foi realizada em relação à rotina do usuário.

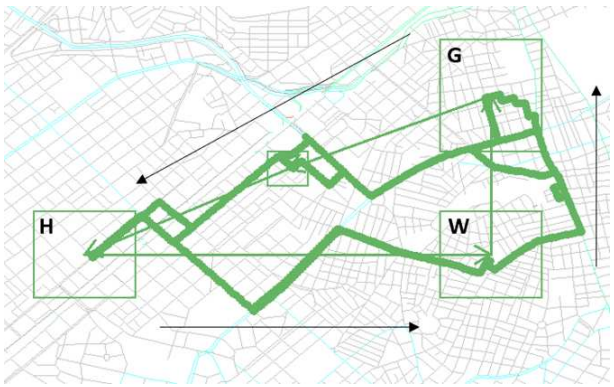


Figura 5.5: Rotina do usuários utilizada para gerar registros de mobilidade para experimentos do CMFog$_V$.

5.4 Resultados

Esta seção apresenta os resultados dos experimentos de simulação conduzidos para investigar o desempenho do CMFog$_V$ em cenários que replicam condições do mundo real. A análise dos resultados será dividida em duas partes. Iniciaremos examinando os resultados de cenários que envolvem apenas migrações horizontais, buscando compreender como cada nível da Névoa impacta o desempenho do CMFog$_V$. Nessa análise, será crucial observar como as configurações dos cloudlets nos diferentes níveis influenciam na latência.

Na segunda parte, analisaremos os resultados dos experimentos com migração vertical para entender como essa estratégia impacta a experiência do usuário. Será essencial avaliar como as migrações verticais contribuem para a otimização da latência em diferentes situações. Além disso, exploraremos como os parâmetros W e T_L, relacionados ao processo de tomada de decisão, influenciam o comportamento do $CMFog_V$. Essa análise proporcionará uma compreensão mais aprofundada de como a estratégia de migração vertical responde a diferentes configurações e demandas.

Em resumo, a análise detalhada desses dois aspectos dos experimentos permitirá uma compreensão abrangente do desempenho e da eficácia do $CMFog_V$ em cenários simulados que se aproximam das condições do mundo real. Esses resultados são fundamentais para validar a utilidade prática da abordagem proposta nesta tese.

5.4.1 Migração horizontal: $P3_1$ e $P3_2$

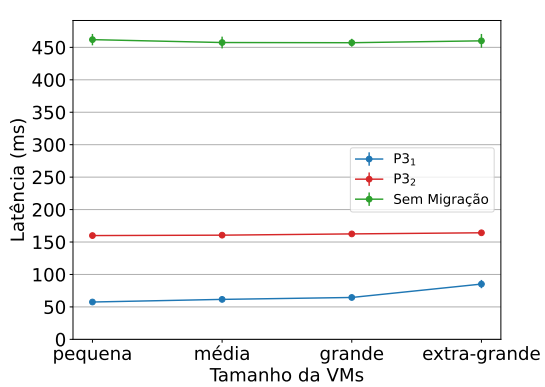
A análise dos resultados obtidos nos cenários com as políticas $P3_1$ e $P3_2$ é de extrema importância, uma vez que esses experimentos buscam identificar de que maneira cada nível da camada de Névoa é capaz de influenciar na qualidade do serviço provido ao usuário. Portanto, esses cenários, juntamente com a política Sem Migração, desempenham o papel de pontos de referência, fornecendo informações essenciais para uma análise mais precisa sobre o impacto do $CMFog_V$ na experiência do usuário.

A análise do *trade-off* entre os diferentes níveis da Névoa pode oferecer informações valiosas a partir desses resultados. Os rótulos $P3_1$ e $P3_2$ denotam cenários que envolvem exclusivamente migrações horizontais no primeiro e segundo nível da Névoa, respectivamente. Os resultados desses cenários são apresentados na Figura 5.6. O Gráfico 5.6a, que representa a métrica de latência, revela que $P3_1$ exibe médias consideravelmente inferiores em comparação com $P3_2$. Esse comportamento é justificado pela diferença na distância geográfica entre os níveis da Névoa e os dispositivos dos usuários.

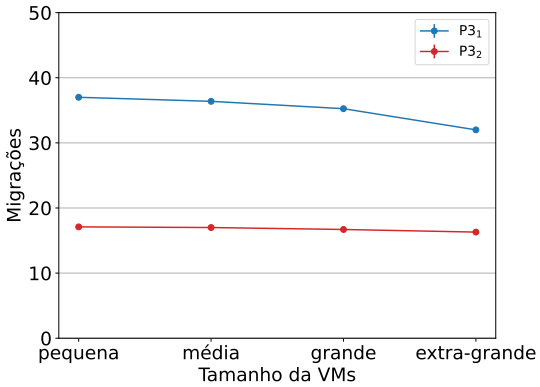
O gráfico 5.6b apresenta os resultados para a métrica de número de migrações. Ao analisar os resultados dos cenários com $P3_1$, destaca-se a diminuição no número de migrações à medida que os tamanhos das VMs aumentam, evidente especialmente em cenários com VMs extra-grandes. O aumento no tempo necessário para concluir uma migração com VMs maiores naturalmente tende a reduzir a frequência desses eventos, uma vez que o tempo de simulação é mesmo para todos os cenários.

Nos experimentos com $P3_2$, a redução na média de migrações ocorre com menor intensidade, quando comparados com experimentos com $P3_1$. Isso se deve à maior área de cobertura das cloudlets no segundo nível, o que resulta em uma menor necessidade de migrações. Como os usuários precisam percorrer distâncias maiores para sair da área de cobertura e desencadear um *handoff* entre cloudlets, o processo de migração de VM é menos frequente. Essas observações destacam como o impacto do tamanho das VMs no número total de migrações é reduzido no segundo nível da Névoa.

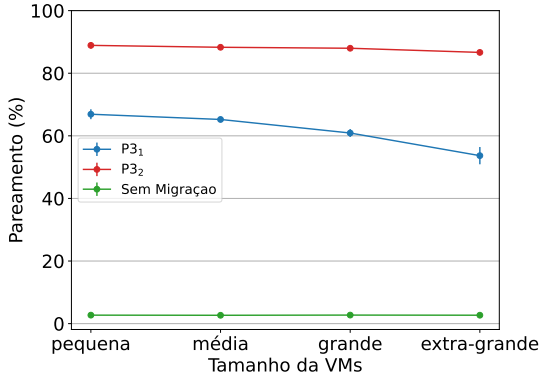
Os resultados da métrica de emparelhamento, apresentados no Gráfico 5.6c, indicam que os cenários com $P3_2$ proporcionam um maior período de emparelhamento. Como visto no gráfico de migração, a maior área de cobertura das cloudlets do segundo nível exige um menor número de eventos de migração. Somado a isso, a menor frequência de troca



(a) Latência



(b) Migração



(c) emparelhamento

Figura 5.6: Gráficos com os resultados dos experimentos de simulação dos cenários com políticas P3 e Sem migração.

de área de cobertura do usuário são as razões para maior período de emparelhamento nos cenários com $P3_2$.

Os diferentes tamanhos de VMs influenciam diretamente nos resultados obtidos nos vários cenários simulados. Ao examinar os gráficos de latência e migração das simulações com $P3_1$, fica evidente que, à medida que o tamanho da VM aumenta, a complexidade em gerir o conteúdo na Névoa também se intensifica. As médias de latência aumentam, enquanto o número de eventos de migração diminui. O desafio e custo associados à migração de VMs de maiores tamanhos impactam diretamente no tempo de emparelhamento, o que, consequentemente eleva as médias de latência.

Na análise dos cenários com VMs de tamanho extra-grande, identificamos a maior degradação de desempenho em comparação com outros tamanhos de VM em $P3_1$. Isso é uma consequência da maior complexidade associada à gestão de VMs maiores. No entanto, nos cenários com $P3_2$, devido à maior estabilidade e menor necessidade de eventos de migração no segundo nível da Névoa, é possível observar uma redução significativa do efeito das variações nos tamanhos das VMs quando comparado aos cenários $P3_1$. No entanto, apesar do efeito reduzido no P_1, na duas políticas podemos observar o impacto negativo do aumento de tamanho das VMs na latência.

Ao analisar os resultados dos experimentos com a política que não realiza migrações de VMs, é possível identificar um alta média de latência, com variações mínimas, mesmo

considerando diferentes tamanhos. Esse comportamento era esperado, pois a ausência de migração naturalmente resulta em médias de latência elevadas, e o tamanho da VM não afeta a latência na ausência de migrações. A métrica de emparelhamento, indicando uma porcentagem entre 1% e 2%, reflete os momentos iniciais da simulação nos quais o usuário já está pareado com sua respectiva VM. Embora os resultados para esses cenários sejam previsíveis, sua análise é importante para a compreensão dos benefícios que o CMFog$_V$ pode proporcionar tanto através das migrações verticais quanto das horizontais.

Ao comparar os resultados dos experimentos com as políticas P3$_1$ e P3$_2$, torna-se evidente o *trade-off* entre latência e área de cobertura, caracterizando os diferentes níveis da Névoa. Os resultados com P3$_1$ demonstraram uma latência menor em comparação com os experimentos com P3$_2$. Esse comportamento é justificado pela maior proximidade geográfica dos usuários com o primeiro nível da Névoa, o que naturalmente é capaz de oferecer menores latências. Em contrapartida, a maior área de cobertura das cloudlets do segundo nível permite que menos migrações sejam realizadas nos cenários com P3$_2$. Assim, o primeiro nível da Névoa pode oferecer uma menor latência a um custo de um maior número de migrações, enquanto o segundo nível da Névoa fornece maior disponibilidade de VMs e menor utilização de rede, com a desvantagem de uma latência mais alta. Esta análise dos resultados das políticas P3$_1$ e P3$_2$ solidifica a compreensão do impacto dos níveis na camada de Névoa, fornecendo uma base para as futuras análises do CMFog$_V$.

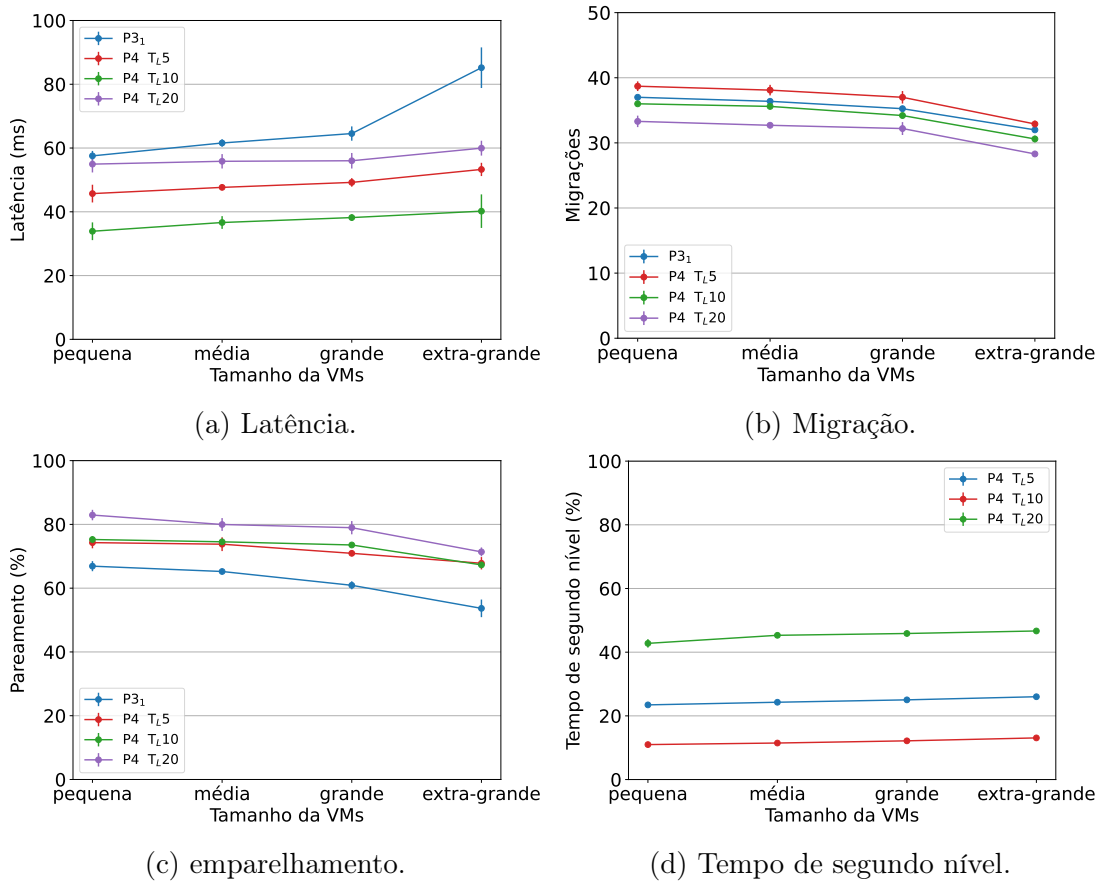
5.4.2 CMFog$_V$

A segunda fase da análise dos resultados se concentra no comportamento do CMFog$_V$ e na influência exercida pelas variáveis W e T$_L$ sobre o seu desempenho. Para facilitar a interpretação dos gráficos, devido à disparidade numérica, optamos por omitir os resultados das políticas P3$_2$ e da Sem Migração. A fim de manter clareza na discussão, os resultados são analisados separadamente para cada variável, começando por fixar os valores de W e variar T$_L$, seguido pela fixação de T$_L$ e variação de W. Essa abordagem visa proporcionar uma discussão mais clara e com diferentes perspectivas sobre como cada parâmetro influencia o comportamento do CMFog$_V$ nos diferentes cenários simulados.

Como lembrete, o parâmetro W representa o tamanho da janela de monitoramento, atuando como a memória do CMFog$_V$ e desempenhando um papel importante para definir a duração pela qual um evento específico influência o processo de tomada de decisão de migração vertical. Por outro lado, a variável T$_L$ representa um *threshold* para a métrica de latência, sendo valores menores mais propensos a ativar o mecanismo de migração, enquanto valores maiores dificultam a ativação.

Como mencionado anteriormente, a análise dos resultados do CMFog$_V$ será dividida em dois momentos partes: o primeiro avalia o impacto do *threshold* de latência, enquanto a segunda examina os efeitos do tamanho da janela de monitoramento. Ao considerar essas duas perspectivas, buscamos obter uma compreensão mais profunda de como o CMFog$_V$ é influenciado por cada uma dessas variáveis. Essa abordagem nos permitirá avaliar a estratégia de maneira mais eficaz.

Os gráficos apresentados nas Figuras 5.7, 5.8 e 5.9 mostram os resultados obtidos nos experimentos com diferentes valores para o fator tamanho de janela de monitoramento W.



(a) Latência.

(b) Migração.

(c) emparelhamento.

(d) Tempo de segundo nível.

Figura 5.7: Gráficos com os resultados dos experimentos de simulação dos cenários com CMFog$_V$ e $W = 50$.

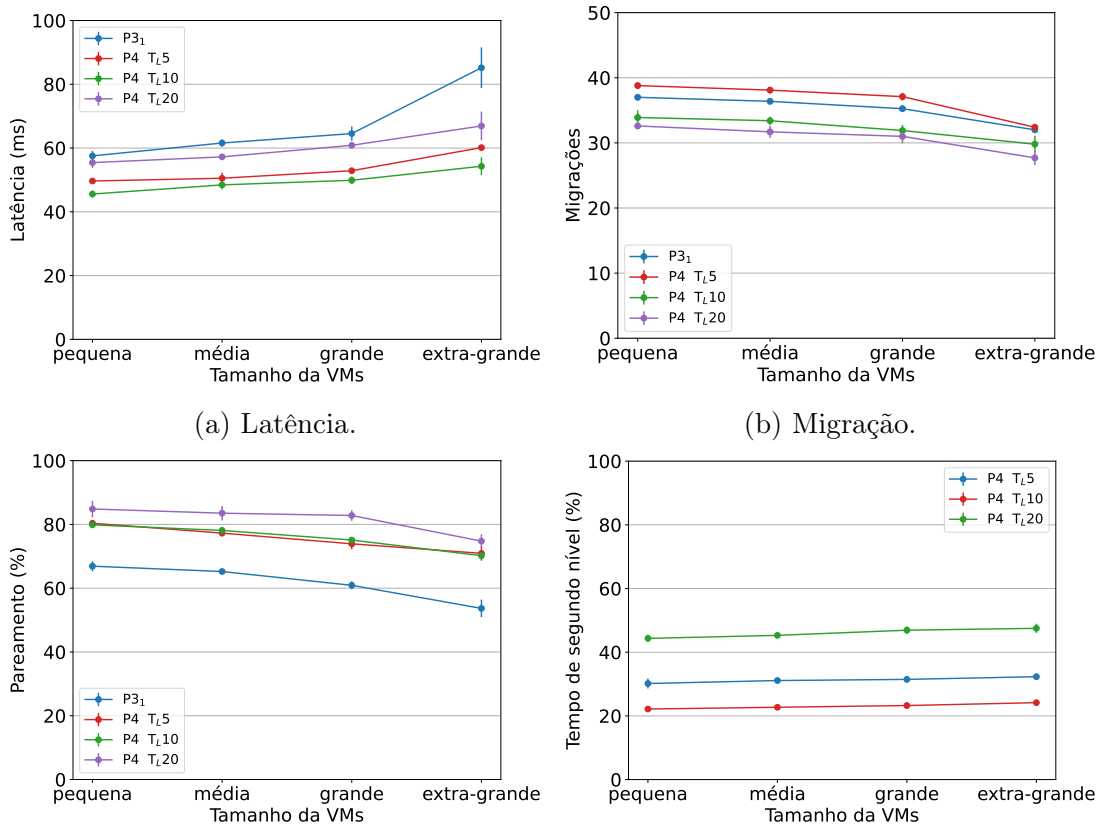
Nos cenários com $W = 50$, conforme ilustrado no gráfico 5.7a, observamos que todos os cenários com a política P4, independente dos valores de W e T_L, apresentaram médias de latência menores em comparação com cenários com a política P3$_1$. Ao considerar apenas as variações dos cenários com a política P4, aqueles com $T_L = 10$ registraram as menores médias, enquanto os com $T_L = 20$ apresentaram as maiores.

Na métrica de número de migrações, como evidenciado no gráfico 5.7b, os resultados dos experimentos nos cenários com $T_L = 5$ exibem um alto número de migrações, até mesmo superando os cenários com a política P3$_1$. Por outro lado, a menor média de migrações foi identificada nos cenários com $T_L = 20$. Um comportamento relevante é que, embora os cenários com $T_L = 5$ tenham apresentado as maiores médias de migração, eles também registraram uma latência superior em relação aos com $T_L = 10$. Esse padrão contradiz a tendência observada nos experimentos do CMFog$_H$, nos quais um maior número de migrações geralmente levava a latências mais baixas. No entanto, uma compreensão mais profunda da situação pode ser alcançada ao analisar os resultados das métricas de emparelhamento e tempo de segundo nível.

Nos cenários com $T_L = 5$, as VMs permaneceram por um período mais longo no segundo nível e ainda assim obtiveram uma menor porcentagem de emparelhamento quando comparadas com $T_L = 10$. Ao considerar os resultados de todas as métricas, percebemos que os cenários com $W = 50$ e $T_L = 5$ apresentaram um comportamento que foge das ten-

dências identificadas em análises anteriores. Sendo o menor valor de *threshold*, $T_L = 5$ é o cenário que oferece maior facilidade para realizar migrações verticais. No entanto, esses cenários também foram simulados com um tamanho de janela de monitoramento menor, o que atribui um peso muito maior às médias recentes. A combinação desses fatores pode ter gerado cenários mais voláteis a variações abruptas na latência. Para entender melhor esse comportamento, é importante analisar os resultados com outros tamanhos de janela de monitoramento, que serão discutidos em breve.

Os experimentos com $T_L = 20$ apresentaram as médias de latência mais altas, o menor número de migrações e o maior tempo de emparelhamento. Esse comportamento é resultado do maior tempo gasto no segundo nível da Névoa, como pode ser visto no gráfico da métrica de emparelhamento (5.7d). Já os cenários com $T_L = 10$ alcançaram as latências mais baixas, com número de migrações próximos aos dos cenários com $P3_1$ e com o menor tempo gasto no segundo nível da névoa.



(a) Latência.

(b) Migração.

(c) emparelhamento.

(d) Tempo de segundo nível.

Figura 5.8: Gráficos com os resultados dos experimentos de simulação dos cenários com $CMFog_V$ e $W = 100$.

Ao observar os resultados dos experimentos, é possível identificar que a variação do tamanho das VMs exerce uma influência semelhante nos experimentos com a política $P3_1$ e P4. O aumento do tamanho das VMs resulta no aumento na latência e na redução no número de migrações. É importante destacar que esse comportamento também foi observado nos experimentos realizados com o $CMFog_H$. Entretanto, os resultados da P4 mostram que esse impacto aconteceu de forma mais sutil, principalmente na métrica de

latência, onde o aumento da média ocorreu de forma mais sutil.

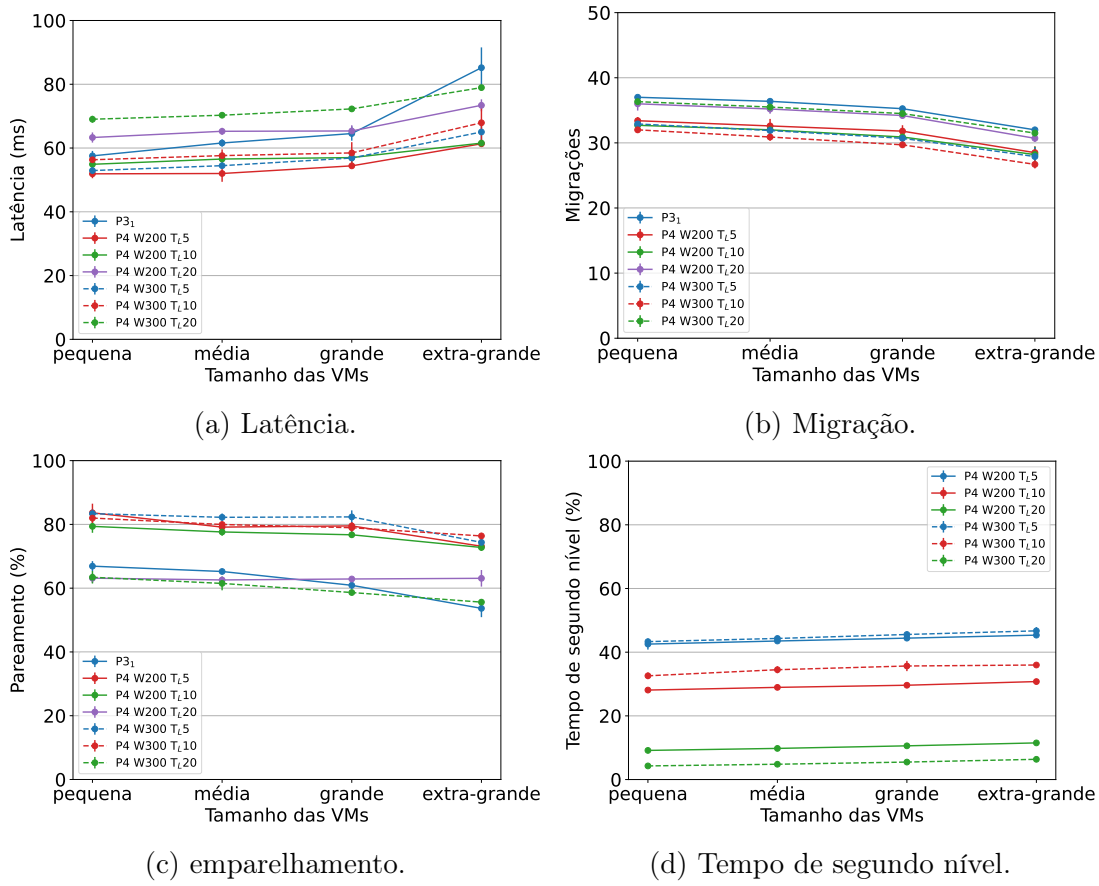
Nesse primeiro momento, a suavização do impacto do tamanho das VMs indica que a capacidade de migrar verticalmente o conteúdo pode contribuir para que o CMFog_V seja capaz de lidar de forma mais eficaz com esse tipo de cenário. Essa hipótese é reforçada ao observar as métricas de emparelhamento e tempo de segundo nível, nas quais os resultados mostram que à medida que o tamanho das VMs aumenta, o tempo que elas permanecem no segundo nível da névoa também cresce, refletindo também no aumento do período de emparelhamento entre conteúdo e usuário.

A Figura 5.8 apresenta os resultados dos experimentos realizados com $W = 100$. Esses resultados mostraram um comportamento semelhante nos cenários com a política P4 em comparação com os experimentos com $W = 50$. No entanto, uma diferença significativa entre os resultados de $W = 100$ e $W = 50$ é observada nas métricas de latência e tempo de segundo nível, onde ambas apresentaram um aumento nas médias. De maneira geral, os cenários com $T_L = 5$ e $T_L = 10$ destacaram-se com maior contraste em relação aos experimentos com $W = 50$. Considerando a latência, notamos que o cenário com $T_L = 5$ exibiu uma menor volatilidade e se aproximou da média dos cenários com $T_L = 10$. No que diz respeito ao tempo de segundo nível, também foi observada uma aproximação entre esses cenários. Com uma janela de monitoramento de $W = 100$, o impacto da média de cada instante de tempo tende a ser reduzido devido à diluição natural causada pelo aumento do tamanho da janela de monitoramento. Esse comportamento fez com que o *threshold* fosse atingido com menor frequência, resultando em uma redução no número de migrações verticais.

Os resultados dos experimentos com $W = 200$ são apresentados nos gráficos da Figura 5.9. Ao contrário do que observamos nos resultados dos cenários com $W = 100$, os experimentos com $W = 200$ apresentaram algumas mudanças importantes na relação entre os cenários que utilizam a Política P4. Um dos pontos de destaque pode ser observado no gráfico 5.9a, no qual alguns cenários com $T_L = 20$ ultrapassam a média de latência dos cenários com P3_1, o que não havia acontecido até então com nenhum cenário que utiliza a política P4. O único cenário de $\text{T}_L = 20$ que permanece com média de latência inferior são os com VM de tamanho extra-grande. Ao analisar outras métricas, também é possível observar outras mudanças, como a redução no tempo de segundo nível e o aumento do número de migrações. A proximidade dos resultados com o cenário com P3_1 mostra que os com P4, $W = 200$ e $T_L = 20$ realiza menos migrações verticais, resultado da combinação do aumento da janelas de monitoramento e de um maior *threshold* de latência.

É importante ressaltar que, apesar de alguns cenários apresentarem uma latência mais elevada, nos experimentos com VMs extra-grandes, a política P4 ainda conseguiu garantir uma latência inferior à da política P3_1. Esse comportamento é significativo, pois evidencia os benefícios da introdução da migração vertical, especialmente nos cenários em que não foi possível obter resultados satisfatórios nos experimentos do CMFog_H

Nos experimentos com $W = 200$, observamos uma dinâmica diferente em relação aos *threshold* de latência. Os cenários com $T_L = 5$ demonstraram oferecer uma latência menor do que aqueles com $T_L = 10$. Esse comportamento ocorreu em contraste com os resultados obtidos nos experimentos anteriores com $W = 50$ e $W = 100$. A menor latência associada aos cenários de $T_L = 5$ também foi acompanhada por uma média de migração mais alta,



(a) Latência.

(b) Migração.

(c) emparelhamento.

(d) Tempo de segundo nível.

Figura 5.9: Gráficos com os resultados dos experimentos de simulação dos cenários com CMFog$_V$ e $W = 200$ ou $W = 300$.

um aumento no emparelhamento e um maior tempo no segundo nível. Essa observação sugere que o aumento na janela de monitoramento desempenhou um papel importante ao aumentar o tamanho da amostra das médias, reduzindo a volatilidade observada nos cenários com $W = 50$ e $T_L = 5$. Por sua vez, isso controlou a frequência de ativação de eventos de migração, contribuindo para a redução da latência média. Por outro lado, o aumento da janela de monitoramento resultou em uma frequência reduzida de ativação do *threshold* $T_L = 10$, o que se refletiu em uma latência média mais alta em comparação com $T_L = 5$.

Ao analisar os resultados dos experimentos com $W = 300$, conforme mostrado na Figura 5.9, observamos que os cenários com $T_L = 5$ e $T_L = 10$ exibiram comportamentos semelhantes aos obtidos com $W = 200$. No entanto, os cenários com $T_L = 20$ apresentaram médias de latência mais altas do que os cenários P3$_1$, com exceção dos cenários com VMs extra-grandes. Nessas configurações, a latência média permaneceu mais próxima dos valores obtidos em outros cenários. Apesar do aumento na latência em alguns cenários, o CMFog$_V$ ainda conseguiu manter uma latência média mais baixa em comparação com experimentos com P3$_1$.

As Figuras 5.10 e 5.11 fornecem uma nova perspectiva sobre os resultados discutidos anteriormente, com os *thresholds* fixados para uma melhor compreensão do impacto dos diferentes tamanhos de janela de monitoramento nos resultados do CMFog$_V$. Ao analisar

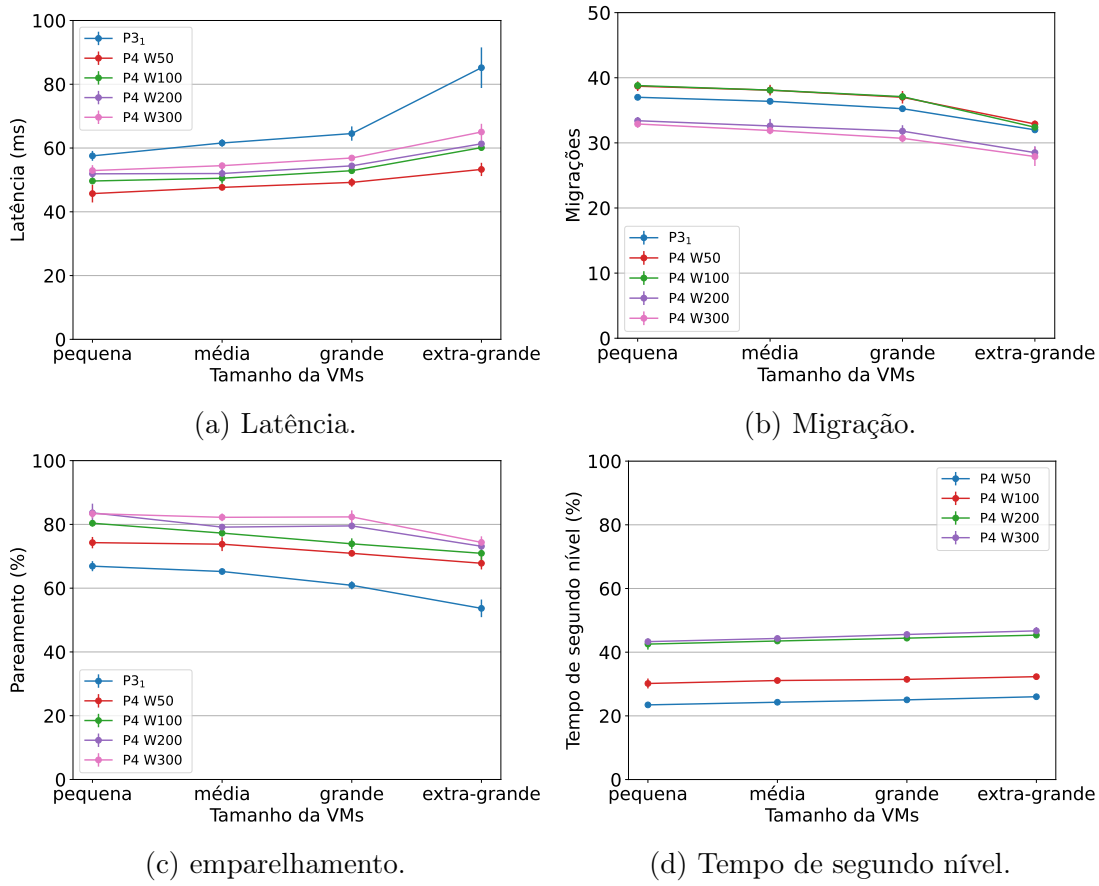
os gráficos 5.10a e 5.11a, que representam a métrica de latência, observamos que o aumento no tamanho da janela de monitoramento W está associado a um aumento na latência. Essa tendência foi consistente em todos os valores do *threshold* T_L, conforme evidenciado pelas médias obtidas nos experimentos.

Como observado nas análises anteriores, os cenários com a política P4 e $W = 100$, especialmente aqueles com $W = 50$, demonstraram comportamentos voláteis quando combinados com $\mathrm{T}_L = 5$. Essa volatilidade se torna mais evidente ao examinarmos a métrica de número de migrações, como mostrado nos gráficos 5.10b e 5.11b, onde a média superou aquela observada nos cenários da política $\mathrm{P3}_1$. Por outro lado, nos cenários com janelas de monitoramento maiores, como $W = 200$ ou $W = 300$, os resultados foram mais consistentes e as métricas começaram a convergir para uma mesma direção. No entanto, observamos divergências significativas, especialmente quando foram atribuídos valores baixos ou muito altos às variáveis W e T_L. Quando os valores eram baixos, cada média dentro da janela de monitoramento tinha uma influência maior, e o limite necessário para iniciar uma migração vertical também era baixo. Isso resultava em eventos de migração vertical baseado em picos ou aumentos momentâneos nas médias. Por outro lado, quando os valores eram altos, especialmente para T_L, as VMs permaneciam no segundo nível por um período excessivo, o que, apesar de aumentar a porcentagem de emparelhamento, também elevava a latência. Em certos cenários, isso resultava em uma média de latência superior àquela observada nos cenários com a política $\mathrm{P3}_1$. Esses resultados destacam a importância de selecionar valores adequados para as variáveis W e T_L para garantir um desempenho consistente e estável do CMFog_V.

A análise dos resultados revela como cada variável influencia a dinâmica do processo de migração no CMFog_V. O fator W, representando o tamanho da janela de monitoramento, desempenha o importante papel de determinar por quanto tempo um evento específico, representado pela média de um instante de tempo, influencia o processo de tomada de decisão. Janelas menores mantêm a relevância dos eventos por um curto período de tempo, pois cada valor dentro da janela tem um impacto mais significativo devido ao espaço amostral menor das médias. Por outro lado, janelas maiores estendem o período em que os eventos são considerados, reduzindo o impacto de cada evento individual na análise de variação devido ao aumento do espaço amostral das médias.

O fator T_L, que representa o limite de variação na janela de monitoramento para iniciar um evento de migração vertical, desempenha um papel crucial no processo. Valores baixos podem desencadear ativações frequentes ou prematuras, o que pode comprometer a média de latência e resultar no desperdício de recursos da infraestrutura de rede utilizados para as migrações. Por outro lado, valores excessivamente altos podem exigir um nível de variação muito alto, resultando em decisões tardias e possivelmente perda de eficiência gerenciamento de conteúdo na névoa.

Para otimizar o desempenho do CMFog_V, é essencial encontrar um equilíbrio adequado entre os parâmetros W e T_L. Enquanto o tamanho da janela de monitoramento W determina a influência das médias de latência em cada instante de tempo no processo de migração vertical, o *threshold* de latência T_L estabelece os limites necessários para desencadear decisões de migração. Ao analisar os diferentes cenários simulados nos experimentos, podemos identificar algumas tendências na relação entre esses dois fatores.



(a) Latência.

(b) Migração.

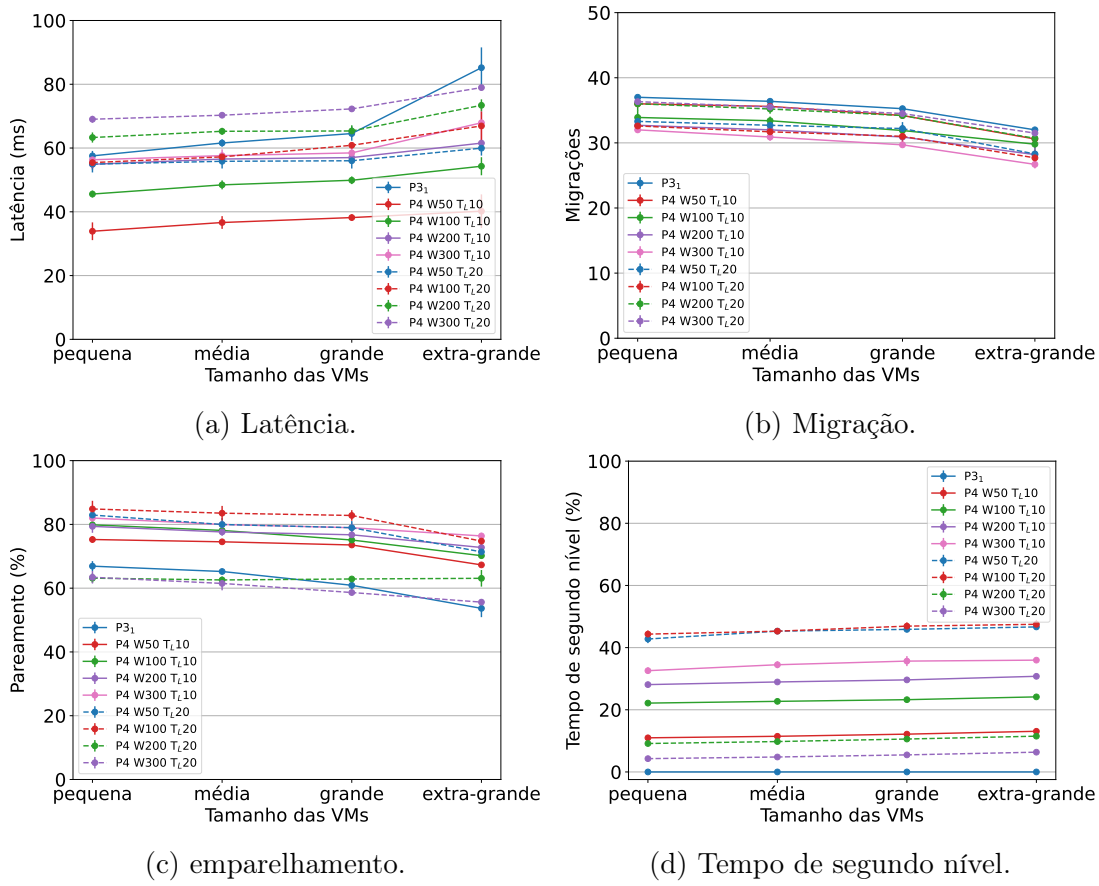
(c) emparelhamento.

(d) Tempo de segundo nível.

Figura 5.10: Gráficos com os resultados dos experimentos de simulação dos cenários com CMFog$_V$ e $T = 5$.

Nos experimentos com $W = 50$ e $T_L = 5$, observou-se uma considerável volatilidade nos resultados. Atribuir valores mínimos para cada um dos parâmetros resultou em cenários nos quais picos ou anomalias tinham uma grande influência no cálculo da variação, e o limite para iniciar uma migração vertical era baixo e facilmente atingido. Essa combinação de fatores criou um cenário volátil, no qual atribuir um peso excessivo a possíveis anomalias ou picos desencadeou migrações frequentes entre os níveis da névoa, comprometendo a efetividade do CMFog$_V$ no processo de tomada de decisão de migração de conteúdo na névoa.

Nos experimentos com $W = 300$ e $T_L = 20$, o alto limite de variação necessário para iniciar uma migração vertical tornou difícil para as VMs se deslocarem entre os níveis da névoa. Além disso, o tamanho da janela de monitoramento pode ter mascarado tendências do usuário ao considerar um intervalo muito grande no cálculo da variação da média. Isso comprometeu a capacidade do CMFog$_V$ de explorar efetivamente os diferentes níveis de computação em névoa, aproximando os resultados da política P4 dos resultados da política que não realiza migrações verticais. Em ambos os casos, fica evidente a importância de encontrar um equilíbrio entre a sensibilidade à variação de latência e a frequência de ativação das migrações verticais para garantir um desempenho eficaz do CMFog$_V$.



(a) Latência.

(b) Migração.

(c) emparelhamento.

(d) Tempo de segundo nível.

Figura 5.11: Gráficos com os resultados dos experimentos de simulação dos cenários com CMFog$_V$ e $T = 10$ ou $T = 20$.

5.4.3 Discussão

A implementação do CMFog$_V$, que introduz a migração vertical em uma infraestrutura de névoa multinível, mostrou avanços significativos em comparação com o CMFog$_H$. As análises dos experimentos realizados forneceram observações importantes sobre a eficácia das estratégias de migração de conteúdo na borda da rede.

Os experimentos do Projeto Experimental I (PE I) foram capazes de delinear o *trade-off* entre diferentes níveis da névoa, destacando as características específicas de cada nível. Mesmo ao utilizar um novo conjunto de dados de mobilidade, os resultados dos experimentos com a Política P3$_1$ corroboraram o comportamento observado no CMFog$_H$. Uma das novas métricas, o tempo de emparelhamento, foi capaz de confirmar a hipótese levantada anteriormente, mostrando uma queda na média de emparelhamento com o aumento do tamanho das VMs. Isso evidenciou a perda de eficácia do CMFog$_H$ em máquinas virtuais de tamanhos maiores.

Os resultados dos experimentos do PE II mostraram que a Política P4, que permite a migração de VMs entre níveis da Névoa, alcançou as menores latências em praticamente todos os cenários. Isso demonstra que, ao explorar múltiplos níveis, o CMFog$_V$ possui maior adaptabilidade em comparação com o CMFog$_H$. Além da latência, a métrica de tempo de emparelhamento também confirmou a eficácia dessa abordagem, no qual P4 mostrou as maiores médias de emparelhamento entre todos os cenários.

Os resultados dos experimentos com a Política P4 também indicaram uma redução no impacto que o tamanho das VMs exerce na proximidade entre conteúdo e usuário, e consequentemente, na latência. O CMFog$_H$ mostrava limitações em manter a eficácia em situações onde: i) o volume de dados das VMs era muito grande; ou ii) a velocidade de deslocamento do usuário era muito alta, impedindo a realização de migrações na velocidade necessária para acompanhar o usuário. O CMFog$_V$ superou essas limitações, mantendo-se relevante e apresentando bons resultados mesmo nesses cenários.

Os fatores W (tamanho da janela de monitoramento) e T$_L$ (*threshold* de latência) são fundamentais para definir quando o processo de migração vertical deve ser iniciado. Os experimentos utilizaram uma variedade de níveis desses fatores para avaliar seu impacto na redução da latência. Duas combinações se destacaram por degradarem significativamente a eficácia do CMFog:

- $W = 50$ com T$_L$=5: gerou um processo de decisão volátil, com facilidade em atingir as condições para migração vertical, resultando em um efeito "ping-pong" de migrações sucessivas.
- T$_L$=20 com $W = 200$ e $W = 300$: o alto *threshold* em conjunto com janelas de monitoramento maiores limitaram drasticamente as ativações da migração vertical.

Esses resultados destacam a importância do equilíbrio entre os parâmetros para garantir a proximidade entre conteúdo e usuário, e consequentemente, a eficiência do processo de migração.

Ao comparar os resultados entre as versões CMFog$_V$ e CMFog$_H$, fica evidente a melhora de desempenho alcançada com a introdução da migração vertical. O CMFog$_V$ demonstrou maior flexibilidade e adaptabilidade para lidar com uma ampla gama de cenários na Computação em Névoa. Notavelmente, o CMFog$_V$ apresentou desempenho superior em cenários com VMs extra grandes, evidenciando menor sensibilidade ao tamanho das VMs em comparação com o CMFog$_H$. A incorporação de migrações verticais pelo CMFog$_V$ proporciona soluções eficazes para cenários complexos, garantindo o emparelhamento do usuário e a disponibilidade de conteúdo.

Portanto, após analisar o conjunto de resultados dos experimentos, podemos identificar a viabilidade do CMFog$_V$ como uma estratégia promissora para aprimorar o desempenho e a eficiência do processo de migração de conteúdo na Computação em Névoa, sendo capaz de reduzir a latência e melhorar a qualidade do serviço na borda da rede.

Capítulo 6

Conclusões

A revolução digital desencadeada pelo surgimento e expansão da Internet das Coisas marcou uma nova era na qual bilhões de dispositivos interconectados coletam e trocam dados em tempo real. Esse fenômeno impacta todos os aspectos da vida moderna, desde ambientes residenciais até sistemas industriais complexos. Diante desse cenário, surgem desafios que demandam abordagens computacionais inovadoras capazes de lidar com questões como latência, eficiência na comunicação e gestão de recursos.

A Computação em Névoa surge como uma resposta a esses desafios, estendendo as capacidades da computação em nuvem até a borda da rede, onde os dispositivos IoT estão localizados. Essa abordagem descentralizada permite o processamento, armazenamento e análise de dados mais próximos de sua fonte de geração, reduzindo a latência e melhorando a eficiência na comunicação. No entanto, essa transição para a Computação em Névoa apresenta desafios significativos, incluindo a gestão eficiente dos recursos computacionais e a otimização da infraestrutura para lidar com a heterogeneidade e a mobilidade dos dispositivos.

A migração dinâmica de conteúdo e serviços para os nós mais próximos da localização atual do usuário é fundamental para garantir uma latência adequada e atender às demandas de Qualidade de Serviço. Neste contexto, esta tese propõe o CMFog, uma estratégia proativa de migração de conteúdo projetada para otimizar a latência de comunicação na infraestrutura de Névoa multi-nível. O CMFog integra a predição de mobilidade como parte do processo de decisão, permitindo que a infraestrutura de Névoa se adapte dinamicamente aos padrões de mobilidade dos dispositivos, maximizando a proximidade entre conteúdo e usuário. Essa abordagem proativa busca minimizar o impacto na experiência do usuário, antecipando as migrações de conteúdo com base na análise preditiva dos padrões de mobilidade.

Esta tese adotou uma abordagem *bottom-up*, dividindo o desenvolvimento e os experimentos do CMFog em duas fases distintas. Inicialmente, foi apresentada a versão CMFog_H, que se concentra em migrações horizontais entre cloudlets do mesmo nível na Névoa. No segundo momento, o CMFog foi expandido para CMFog_V, incorporando migrações verticais entre cloudlets de diferentes níveis na Névoa. Devido à dificuldade e ao custo de validar modelos no mundo real, o processo de análise de desempenho do CMFog foi realizado através da ferramenta de simulação MyiFogSim.

Os experimentos com o CMFog_H revelaram uma perda de efetividade em manter a

proximidade entre conteúdo e usuários à medida que o tamanho do conteúdo aumenta. Nos cenários formados por VMs de tamanho extra-grande, foram observados os piores resultados dos experimentos, revelando uma degradação significativa na média de latência experimentada pelos dispositivos dos usuários. Com o objetivo de obter mais informações sobre esse comportamento, novos experimentos foram realizados nos quais a largura de banda de rede da conexão entre as cloudlets era maior. Através desses novos experimentos, foi identificada uma curva de degradação mais suave com o aumento do tamanho da VMs. De forma geral, as políticas do CMFog$_H$ apresentaram resultados superiores à política LL; contudo, foi observado que o CMFog$_H$ possui dificuldade em garantir que as decisões de migrações permaneçam relevantes em certos cenários, como, por exemplo, quando o tempo necessário para realizar uma migração não é compatível com a velocidade de deslocamento do usuário, reduzindo drasticamente o impacto que a migração é capaz de realizar no processo de aproximação entre conteúdo e usuário.

Diante dessas limitações, o CMFog$_V$ foi desenvolvido para abordar esses desafios, incorporando migrações verticais entre diferentes níveis da infraestrutura de Névoa. Os experimentos conduzidos com o CMFog$_V$ demonstraram melhorias significativas em relação ao CMFog$_H$, especialmente em cenários com VMs extra grandes. A possibilidade de migrar VMs entre os níveis da Névoa é capaz de explorar uma maior gama de possibilidades para determinar destinos de migração com maior capacidade de oferecer menores médias de latência. Os parâmetros de controle de tamanho de janela de monitoramento e *threshold* de latência são fundamentais para representar e tornar o CMFog adaptável para diversas características de mobilidade e aplicações do usuário. Portanto, a estratégia do CMFog$_V$ mostrou-se mais adaptável e flexível, oferecendo soluções para cenários complexos e mantendo o emparelhamento entre usuário e conteúdo.

A análise dos resultados experimentais destacou a importância de considerar a dinâmica dos ambientes de Névoa, onde a mobilidade dos usuários e o tamanho das VMs podem impactar significativamente no desempenho das estratégias de migração de conteúdo. O CMFog$_V$, ao incorporar migrações verticais e explorar o *trade-off* entre latência e cobertura, oferece uma abordagem mais adaptável para otimizar a migração de conteúdo na Computação em Névoa.

Em um contexto de implantação do CMFog em um cenário real, é importante destacar que as escolhas das abordagens de predição de mobilidade e do processo de tomada de decisão permitem que a solução proposta tenha uma complexidade computacional e de execução reduzida. Diferente de abordagens mais complexas, relacionadas com Aprendizado de Máquina ou Inteligência Computacional, a cadeia de Markov e a Tomada de Decisão por Múltiplos Atributos (MADM) são métodos que não exigem alto poder computacional. Essa característica também permite que o CMFog seja executado em infraestruturas de redes baseadas em tecnologias móveis de comunicação, como 4G e 5G.

Nesse tipo de cenário, o CMFog poderia ser implementado no núcleo dessas redes, o que não exigiria muitos recursos e, ao mesmo tempo, permitiria o monitoramento de informações fundamentais para que a estratégia opere de acordo com o modelo proposto. A execução no núcleo das redes móveis oferece uma visão abrangente e centralizada do comportamento dos usuários e dos recursos de rede, possibilitando uma gestão mais eficiente e uma redução na latência de comunicação.

Em resumo, esta tese contribui para o avanço do campo da computação distribuída na borda da rede, resultando em duas publicações [4, 5], fornecendo soluções inovadoras para melhorar a eficiência e a qualidade de serviço em ambientes de Névoa multi-nível. Tanto o CMFog$_H$ quanto o CMFog$_V$ representam avanços significativos no campo, oferecendo soluções práticas para os desafios enfrentados na migração de conteúdo na Computação em Névoa. Ao integrar técnicas de migração proativa por meio de análises de padrões de mobilidade, o CMFog é capaz de oferecer uma solução adaptável e eficiente para o desafio de migração de conteúdo na borda da rede.

6.1 Trabalhos Futuros

Apesar dos avanços alcançados com o desenvolvimento e experimentos do CMFog, ainda existem diversas oportunidades para expandir e aprimorar essa abordagem de migração de conteúdo na Computação em Névoa. Abaixo, apresento algumas das possíveis direções para futuros trabalhos:

1. **Desenvolvimento de Políticas de Migração Adicionais:** É importante investigar e implementar novas políticas de migração que possam aprimorar ainda mais o desempenho do CMFog em diferentes cenários. Por exemplo, explorar políticas adaptativas que ajustem dinamicamente os parâmetros de migração com base nas condições da rede e nos padrões de mobilidade dos usuários.

2. **Consideração da Heterogeneidade de Recursos:** Ao tomar decisões de migração, é crucial levar em conta a heterogeneidade dos recursos disponíveis na infraestrutura de Névoa. Isso inclui não apenas a latência de comunicação, mas também outros fatores como capacidade de processamento, disponibilidade de energia e restrições de segurança. Adaptar o CMFog para considerar esses aspectos pode resultar em decisões mais eficientes e adaptadas às condições específicas de cada contexto.

3. **Combinação de Políticas de Migração:** Uma abordagem interessante seria associar estratégias de migração reativa e proativa dependendo do nível de informação de mobilidade do usuário disponível. Isso significa que o CMFog poderia alternar entre políticas de migração que respondem diretamente a eventos imediatos e aquelas que antecipam padrões de mobilidade dos usuários para tomar decisões mais assertivas.

4. **Avaliação em Ambientes do Mundo Real:** Para validar e calibrar o desempenho do CMFog em condições reais, é essencial realizar experimentos e avaliações em ambientes do mundo real. Isso pode envolver a implementação de protótipos em ambientes de teste controlados e apartir dessas avaliações, seria possível identificar desafios adicionais necessários para a aplicação prática do CMFog.

5. **Estratégia para determinar valores para W e T_L:** É importante desenvolver estratégias capazes de avaliar as particularidades de cada cenário e as necessidades individuais dos usuários para determinar os valores ideais das variáveis W e T_L, que

são utilizadas para gerenciar o processo de decisão de migração vertical. Os resultados destacaram a alta influência dessas variáveis na qualidade das decisões tomadas pelo CMFog. Portanto, uma estratégia eficaz que possa avaliar a infraestrutura e determinar valores apropriados para cada uma delas é fundamental para garantir a eficiência do CMFog.

6. **Comparação com outras soluções de migração:** Devido à diversidade de soluções propostas na literatura para o problema da migração de conteúdo na Computação em Névoa, é fundamental que esses trabalhos possam ser comparados com o objetivo de destacar os prós e contras entre as diversas abordagens. No entanto, devido à variedade de métodos e ferramentas de avaliação, é necessário realizar uma adaptação para que essas soluções possam ser comparadas em cenários justos. Isso facilita a identificação de áreas de melhoria e oferece um contexto mais amplo para futuros trabalhos na área.

Essas são apenas algumas das possibilidades para futuros trabalhos do CMFog. Ao explorar essas direções, é possível continuar avançando no desenvolvimento de soluções eficazes para otimizar a migração de conteúdo na Computação em Névoa e melhorar a qualidade de serviço para os usuários finais.

Referências Bibliográficas

[1] Seyedeh Negar Afrasiabi, Amin Ebrahimzadeh, Carla Mouradian, Sepideh Maleki-taji, and Roch H Glitho. Reinforcement learning-based optimization framework for application component migration in nfv cloud-fog environments. *IEEE Transactions on Network and Service Management*, 2022.

[2] Shahd Suleiman Alqam, Nasser Al Zeidi, Abderrezak Touzene, and Khaled Day. Location-aware reinforcement learning-based live virtual machine migration in fog computing. In *2023 16th International Conference on Advanced Computer Theory and Engineering (ICACTE)*, pages 242–248. IEEE, 2023.

[3] Marcelo Amaral, Jorda Polo, David Carrera, Iqbal Mohomed, Merve Unuvar, and Malgorzata Steinder. Performance evaluation of microservices architectures using containers. In *Network Computing and Applications (NCA), 2015 IEEE 14th International Symposium on*, pages 27–34. IEEE, 2015.

[4] Marcelo C Araújo, Bruno Sousa, Marilia Curado, and Luiz F Bittencourt. Cmfog: Proactive content migration using markov chain and madm in fog computing. In *2020 IEEE/ACM 13th International Conference on Utility and Cloud Computing (UCC)*, pages 112–121. IEEE, 2020.

[5] Marcelo C. Araújo and Luiz F. Bittencourt. Cmfogv: Proactive content migration for multi-level fog computing. *Pervasive and Mobile Computing - Elsevier*, 102:101933, 2024.

[6] Maria Ashraf, Muhammad Shiraz, Almas Abbasi, Omar Alqahtani, Gran Badshah, and Ayodele Lasisi. Microservice application scheduling in multi-tiered fog-computing-enabled iot. *Sensors*, 23(16):7142, 2023.

[7] Mohammad Irfan Bala and Mohammad Ahsan Chishti. Survey of applications, challenges and opportunities in fog computing. *International Journal of Pervasive Computing and Communications*, 2019.

[8] Rodrigo N Calheiros, Rajiv Ranjan, Anton Beloglazov, César AF De Rose, and Rajkumar Buyya. Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software: Practice and experience*, 41(1):23–50, 2011.

[9] Susanta Nanda Tzi-cker Chiueh and Stony Brook. A survey on virtualization technologies. *RPE Report*, pages 1–42, 2005.

[10] Christopher Clark, Keir Fraser, Steven Hand, Jacob Gorm Hansen, Eric Jul, Christian Limpach, Ian Pratt, and Andrew Warfield. Live migration of virtual machines. In *Proceedings of the 2nd Conference on Symposium on Networked Systems Design & Implementation-Volume 2*, pages 273–286. USENIX Association, 2005.

[11] Felipe Rocha de Araújo, Denis Lima Rosário, Kassio Machado, Eduardo Coelho Cerqueira, and Leandro Villas. Temmus: A mobility predictor based on temporal markov model with user similarity. In *Anais do XXXVII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, pages 594–607. SBC, 2019.

[12] Marios D Dikaiakos, Dimitrios Katsaros, Pankaj Mehra, George Pallis, and Athena Vakali. Cloud computing: distributed internet computing for it and scientific research. *Internet Computing, IEEE*, 13(5):10–13, 2009.

[13] Ian Foster, Yong Zhao, Ioan Raicu, and Shiyong Lu. Cloud computing and grid computing 360-degree compared. In *Grid Computing Environments Workshop, 2008. GCE'08*, pages 1–10. Ieee, 2008.

[14] Sébastien Gambs, Marc-Olivier Killijian, and Miguel Núñez del Prado Cortez. Next place prediction using mobility markov chains. In *Proceedings of the First Workshop on Measurement, Privacy, and Mobility*, page 3. ACM, 2012.

[15] Kaouther Gasmi, Selma Dilek, Suleyman Tosun, and Suat Ozdemir. A survey on computation offloading and service placement in fog computing-based iot. *The Journal of Supercomputing*, 78(2):1983–2014, 2022.

[16] Robert P Goldberg. Architectural principles for virtual computer systems. Technical report, DTIC Document, 1973.

[17] Mohammad Goudarzi, Marimuthu Palaniswami, and Rajkumar Buyya. A distributed application placement and migration management techniques for edge and fog computing environments. In *2021 16th Conference on Computer Science and Intelligence Systems (FedCSIS)*, pages 37–56. IEEE, 2021.

[18] OpenFog Consortium Architecture Working Group et al. Openfog reference architecture for fog computing. *OPFRA001*, 20817:162, 2017.

[19] Sarfaraz Hashemkhani Zolfani, Reza Maknoon, and Edmundas Kazimieras Zavadskas. Multiple attribute decision making (madm) based scenarios. *International Journal of Strategic Property Management*, 20(1):101–111, 2016.

[20] Abhishek Hazra, Pradeep Rana, Mainak Adhikari, and Tarachand Amgoth. Fog computing for next-generation internet of things: fundamental, state-of-the-art and research challenges. *Computer Science Review*, 48:100549, 2023.

[21] TianZhang He, Adel N Toosi, and Rajkumar Buyya. Efficient large-scale multiple migration planning and scheduling in sdn-enabled edge computing. *IEEE Transactions on Mobile Computing*, 2023.

[22] Maurice Herlihy. A methodology for implementing highly concurrent data objects. *ACM Trans. Program. Lang. Syst.*, 15(5):745–770, November 1993.

[23] Ching-Lai Hwang, Young-Jou Lai, and Ting-Yun Liu. A new approach for multiple objective decision making. *Computers & operations research*, 20(8):889–899, 1993.

[24] Michaela Iorga, Larry Feldman, Robert Barton, Michael J Martin, Nedim S Goren, and Charif Mahmoudi. Fog computing conceptual model. 2018.

[25] Raj Jain. *The art of computer systems performance analysis: techniques for experimental design, measurement, simulation, and modeling*, volume 1. Wiley New York, 1991.

[26] Morteza Karimzadeh, Zhongliang Zhao, Luuk Hendriks, Ricardo de O Schmidt, Sebastiaan la Fleur, Hans van den Berg, Aiko Pras, Torsten Braun, and Marius Julian Corici. Mobility and bandwidth prediction as a service in virtualized lte systems. In *2015 IEEE 4th International Conference on Cloud Networking (CloudNet)*, pages 132–138. IEEE, 2015.

[27] Navjeet Kaur, Ashok Kumar, and Rajesh Kumar. Promo: Proactive mobility-support model for task scheduling in fog computing. *International Journal of Computers and Applications*, 44(11):1092–1101, 2022.

[28] Mustafa I Khaleel and Michelle M Zhu. Adaptive virtual machine migration based on performance-to-power ratio in fog-enabled cloud data centers. *The Journal of Supercomputing*, 77(10):11986–12025, 2021.

[29] Hakima Khelifi, Senlin Luo, Boubakr Nour, Akrem Sellami, Hassine Moungla, and Farid Naït-Abdesselam. An optimized proactive caching scheme based on mobility prediction for vehicular networks. In *2018 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6. IEEE, 2018.

[30] Mykel J Kochenderfer. *Decision making under uncertainty: theory and application*. MIT press, 2015.

[31] Mohammed Laroui, Boubakr Nour, Hassine Moungla, Moussa A Cherif, Hossam Afifi, and Mohsen Guizani. Edge and fog computing for iot: A survey on current research activities & future directions. *Computer Communications*, 180:210–231, 2021.

[32] Márcio Moraes Lopes, Wilson A Higashino, Miriam AM Capretz, and Luiz Fernando Bittencourt. Myifogsim: A simulator for virtual machine migration in fog computing. In *Companion Proceedings of the10th International Conference on Utility and Cloud Computing*, pages 47–52. ACM, 2017.

[33] Redowan Mahmud and Rajkumar Buyya. Modelling and simulation of fog and edge computing environments using ifogsim toolkit. *Fog and edge computing: Principles and paradigms*, 1, 2019.

[34] Redowan Mahmud, Ramamohanarao Kotagiri, and Rajkumar Buyya. Fog computing: A taxonomy, survey and future directions. *Internet of Everything: Algorithms, Methodologies, Technologies and Perspectives*, pages 103–130, 2018.

[35] John Paul Martin, A Kandasamy, and K Chandrasekaran. Mobility aware autonomic approach for the migration of application modules in fog computing environment. *Journal of Ambient Intelligence and Humanized Computing*, 11:5259–5278, 2020.

[36] Carlos Alberto Maziero. Sistemas operacionais: Conceitos e mecanismos ix-virtualização.

[37] Peter Mell and Timothy Grance. The nist definition of cloud computing (draft). *NIST special publication*, 800(145):7, 2011.

[38] Ranesh Kumar Naha, Saurabh Garg, Dimitrios Georgakopoulos, Prem Prakash Jayaraman, Longxiang Gao, Yong Xiang, and Rajiv Ranjan. Fog computing: Survey of trends, architectures, requirements, and research directions. *IEEE access*, 6:47980–48009, 2018.

[39] Opeyemi Osanaiye, Shuo Chen, Zheng Yan, Rongxing Lu, Kim-Kwang Raymond Choo, and Mqhele Dlodlo. From cloud to fog computing: A review and a conceptual live vm migration framework. *IEEE Access*, 5:8284–8300, 2017.

[40] Giovanni Parmigiani and Lurdes Inoue. *Decision theory: Principles and approaches*. John Wiley & Sons, 2009.

[41] Nikos Pelekis, Christos Ntrigkogias, Panagiotis Tampakis, Stylianos Sideridis, and Yannis Theodoridis. Hermoupolis: a trajectory generator for simulating generalized mobility patterns. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 659–662. Springer, 2013.

[42] Gerald J Popek and Robert P Goldberg. Formal requirements for virtualizable third generation architectures. *Communications of the ACM*, 17(7):412–421, 1974.

[43] Carlo Puliafito, Antonio Virdis, and Enzo Mingozzi. The impact of container migration on fog services as perceived by mobile things. In *2020 IEEE International Conference on Smart Computing (SMARTCOMP)*, pages 9–16. IEEE, 2020.

[44] Thomas Robertazzi. Grid and cloud computing. In *Basics of Computer Networking*, pages 65–68. Springer, 2012.

[45] Pedro Juan Roig, Salvador Alcaraz, Katja Gilly, and Carlos Juiz. Modelling a leaf and spine topology for vm migration in fog computing. In *2020 24th International Conference Electronics*, pages 1–6. IEEE, 2020.

[46] Robert Rose. Survey of system virtualization techniques. 2004.

[47] H Sabireen and VJIE Neelanarayanan. A review on fog computing: Architecture, fog with iot, algorithms and research challenges. *Ict Express*, 7(2):162–176, 2021.

[48] Jyotiprakash Sahoo, Subasish Mohapatra, and Radha Lath. Virtualization: A survey on concepts, taxonomy and associated security issues. In *Computer and Network Technology (ICCNT), 2010 Second International Conference on*, pages 222–226. IEEE, 2010.

[49] Hugo Santos, Derian Alencar, Rodolfo Meneguette, Denis Rosário, Jéferson Nobre, Cristiano Both, Eduardo Cerqueira, and Torsten Braun. A multi-tier fog content orchestrator mechanism with quality of experience support. *Computer Networks*, 177:107288, 2020.

[50] Subhadeep Sarkar, Subarna Chatterjee, and Sudip Misra. Assessment of the suitability of fog computing in the context of internet of things. *IEEE Transactions on Cloud Computing*, 2015.

[51] Clark Scheffy. Virtualization for dummies: Amd special edition, 2007.

[52] Hongbo Si, Yue Wang, Jian Yuan, and Xiuming Shan. Mobility prediction in cellular network using hidden markov model. In *2010 7th IEEE Consumer Communications and Networking Conference*, pages 1–5. IEEE, 2010.

[53] Roman Slowinski. Multicriteria optimization and compromise solution. *European Journal of Operational Research*, 44(2):231–238, 1992.

[54] James Edward Smith and Ravi Nair. *Virtual machines: versatile platforms for systems and processes*. Elsevier, 2005.

[55] Gulshan Soni and Mala Kalra. Comparative study of live virtual machine migration techniques in cloud. *International Journal of Computer Applications*, 84(14), 2013.

[56] Jaber Taghizadeh, Mostafa Ghobaei-Arani, and Ali Shahidinejad. An efficient data replica placement mechanism using biogeography-based optimization technique in the fog computing environment. *Journal of Ambient Intelligence and Humanized Computing*, 14(4):3691–3711, 2023.

[57] Tarik Taleb, Sunny Dutta, Adlen Ksentini, Muddesar Iqbal, and Hannu Flinck. Mobile edge computing potential in making cities smarter. *IEEE Communications Magazine*, 55(3):38–43, 2017.

[58] Tarik Taleb, Adlen Ksentini, and Pantelis Frangoudis. Follow-me cloud: When cloud services follow mobile users. *IEEE Transactions on Cloud Computing*, 2016.

[59] O. Temitope. Cloud computing for business management. Master's thesis, Faculty Of Engineering, Leeds, 2010.

[60] Phuoc Nguyen Tran and Nadia Boukhatem. The distance to the ideal alternative (dia) algorithm for interface selection in heterogeneous wireless networks. In *Proceedings of the 6th ACM international symposium on Mobility management and wireless access*, pages 61–68. ACM, 2008.

[61] Evangelos Triantaphyllou and Evangelos Triantaphyllou. *Multi-criteria decision making methods*. Springer, 2000.

[62] Fan-Hsun Tseng, Ming-Shiun Tsai, Chia-Wei Tseng, Yao-Tsung Yang, Chien-Chang Liu, and Li-Der Chou. A lightweight auto-scaling mechanism for fog computing in industrial applications. *IEEE Transactions on Industrial Informatics*, 2018.

[63] Shreshth Tuli, Giuliano Casale, and Nicholas R Jennings. Pregan: Preemptive migration prediction network for proactive fault-tolerant edge computing. In *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*, pages 670–679. IEEE, 2022.

[64] Rich Uhlig, Gil Neiger, Dion Rodgers, Amy L Santoni, Fernando Martins, Andrew V Anderson, Steven M Bennett, Alain Kägi, Felix H Leung, and Larry Smith. Intel virtualization technology. *Computer*, 38(5):48–56, 2005.

[65] Statista Lionel Sujay Vailshery. Number of iot connections worldwide 2022-2033, 2024. Acesso em: 12 jun. 2024.

[66] Richard Van Nee, VK Jones, Geert Awater, Allert Van Zelst, James Gardner, and Greg Steele. The 802.11 n mimo-ofdm standard for wireless lan and beyond. *Wireless Personal Communications*, 37(3-4):445–453, 2006.

[67] Hongjun Wang, Zhen Yang, and Yingchun Shi. Next location prediction based on an adaboost-markov model of mobile users. *Sensors*, 19(6):1475, 2019.

[68] ITUR WP5D. Minimum requirements related to technical performance for imt-2020 radio interface (s), 2017.

[69] Xiao-Yong Yan, Wen-Xu Wang, Zi-You Gao, and Ying-Cheng Lai. Universal model of individual and population mobility on diverse spatial scales. *Nature communications*, 8(1):1639, 2017.