Universidade Estadual de Campinas
Instituto de Computação

Fillipe dos Santos Silva

Algoritmo para Posicionamento de Serviços Multimídia em Ambientes Hierárquicos Nuvem-Névoa

CAMPINAS
2021

Fillipe dos Santos Silva

Algoritmo para Posicionamento de Serviços Multimídia em Ambientes Hierárquicos Nuvem-Névoa

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

Orientador: Prof. Dr. Edmundo Roberto Mauro Madeira
Coorientador: Prof. Dr. Roger Kreutz Immich

Este exemplar corresponde à versão final da Dissertação defendida por Fillipe dos Santos Silva e orientada pelo Prof. Dr. Edmundo Roberto Mauro Madeira.

CAMPINAS
2021

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

Si38a

Silva, Fillipe dos Santos, 1994-
Algoritmo para posicionamento de serviços multimídia em ambientes hierárquicos nuvem-névoa / Fillipe dos Santos Silva. – Campinas, SP : [s.n.], 2021.

Orientador: Edmundo Roberto Mauro Madeira.
Coorientador: Roger Kreutz Immich.
Dissertação (mestrado) – Universidade Estadual de Campinas, Instituto de Computação.

1. Computação em nuvem. 2. Computação em névoa. 3. Serviços multimídia. I. Madeira, Edmundo Roberto Mauro, 1958-. II. Immich, Roger Kreutz. III. Universidade Estadual de Campinas. Instituto de Computação. IV. Título.

Informações para Biblioteca Digital

Título em outro idioma: Multimedia services placement algorithm for cloud-fog hierarchical environments
Palavras-chave em inglês:
Cloud computing
Fog computing
Multimedia services
Área de concentração: Ciência da Computação
Titulação: Mestre em Ciência da Computação
Banca examinadora:
Edmundo Roberto Mauro Madeira [Orientador]
Denis Lima do Rosário
Alex Borges Vieira
Data de defesa: 07-01-2021
Programa de Pós-Graduação: Ciência da Computação

Identificação e informações acadêmicas do(a) aluno(a)
- ORCID do autor: 0000-0002-3698-2304
- Currículo Lattes do autor: http://lattes.cnpq.br/2137828208226681

Universidade Estadual de Campinas
Instituto de Computação

Fillipe dos Santos Silva

Algoritmo para Posicionamento de Serviços Multimídia em Ambientes Hierárquicos Nuvem-Névoa

Banca Examinadora:

- Prof. Dr. Edmundo Roberto Mauro Madeira
  Instituto de Computação - UNICAMP
- Prof. Dr. Denis Lima do Rosário
  Instituto de Ciências Exatas e Naturais - Universidade Federal do Pará (UFPA)
- Prof. Dr. Alex Borges Vieira
  Departamento de Ciência da Computação - Universidade Federal de Juiz de Fora (UFJF)

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, 07 de janeiro de 2021

O nêgo é cabeça de gelo. (Rasta, Cleiton.)

Agradecimentos

Agradeço primeiro a Deus por ter me mantido na trilha certa durante este projeto de pesquisa com saúde e forças para chegar até o final.

À minha mãe, pai e irmãos deixo um agradecimento especial, por todas as lições de amor, dedicação e compreensão. Sinto-me orgulhoso e privilegiado por ter uma família tão especial.

Agradeço aos meus orientadores Prof. Dr. Edmundo Roberto Mauro Madeira e Prof. Dr. Roger Kreutz Immich por aceitarem conduzir o meu trabalho de pesquisa. Também agradeço aos professores Prof. Dr. Evangelos Kranakis e Prof. Dr. Michel Barbeau pela curta orientação durante o estágio na Universidade de Carleton, Canadá. Foi muito especial.

Deixo também um agradecimento especial aos amigos com quem convivi ao longo desses anos. Também agradeço a todos que fazem parte do Instituto de Computação da UNICAMP pelo suporte durante essa pesquisa de mestrado.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de Financiamento 001. Esse projeto também faz parte do INCT denominado Internet do Futuro para Cidades Inteligentes (CNPq 465446 / 2014-0, CAPES 88887.136422 / 2017-00 e FAPESP 2014 / 50937-1).

Por fim, a todos aqueles que contribuíram para a realização desta dissertação, o meu sincero agradecimento.

Resumo

A demanda por serviços multimídia em redes móveis têm aumentado nos últimos anos. A grande quantidade de usuários, tanto consumindo quanto produzindo esses serviços de e para a Nuvem, pode ultrapassar a capacidade de largura de banda disponível da rede e resultar em baixa qualidade de experiência. Apesar dos muitos benefícios que a Computação em Nuvem oferece, como alta disponibilidade e escalabilidade, os serviços multimídia exigem fluxo constante e contínuo de pacotes com baixa latência, requisitos que a Computação em Nuvem ocasionalmente pode, em determinadas situações, não fornecer. A Computação em Névoa, em conjunto com a Nuvem, apresentam-se como uma solução para atender esses serviços e aplicações sensíveis a latência, onde a gestão de todos os recursos acontece de forma hierárquica e coordenada, desde a Nuvem até os dispositivos finais. Esses serviços podem ser alocados em nós névoas que são capazes de virtualizar suas funções e migrar os serviços de acordo com as requisições. A natureza hierárquica, distribuída e heterogênea das instâncias computacionais torna o posicionamento desses serviços nesse ambiente uma tarefa desafiadora. Nesta dissertação de mestrado, inicialmente é proposto um método para a criação de ambientes hierárquicos Nuvem-Névoa. O método proposto utiliza uma abordagem *bottom-up*, iniciando-se a partir de um conjunto de estações base e organiza novos nós hierarquicamente em camadas, produzindo um ambiente hierárquico Nuvem-Névoa. Além disso, é proposto um algoritmo denominado **SMART-FL** para o problema de posicionamento de serviços multimídia em ambientes hierárquicos Nuvem-Névoa modelado como um Problema de Localização de Facilidades Capacitadas. O objetivo é encontrar o menor conjunto de nós considerando suas capacidades de armazenamento para fornecer serviços multimídia de forma que a latência seja minimizada. Para melhorar ainda mais a entrega desses serviços, o volume de tráfego da rede é predito. O objetivo é reservar um conjunto de nós para alocar esses serviços através do volume de tráfego predito. A avaliação de desempenho foi realizada considerando um mês do volume de tráfego real da rede móvel de Milão, Itália. Os resultados são comparados considerando seis estratégias de posicionamento de serviços e avaliados em termos da latência, pacotes entregues, requisições atendidas e uso da rede. Os resultados mostram que o algoritmo proposto posiciona os serviços multimídia em nós com capacidade de armazenamento adequada, próximos aos usuários e com latência média inferior a todas as estratégias. Devido ao volume de tráfego predito, o posicionamento torna-se ainda mais eficiente em razão do armazenamento dos nós reservados previamente. O processamento e armazenamento perto da fonte de dados, sem a necessidade do envio de todos esses serviços para a Nuvem centralizada, reduz o uso da rede total em $\approx 52\%$, pois menos canais para transmissão dos dados são utilizados, diferentemente quando os serviços estão posicionados na Nuvem. Além disso, utilizando as informações obtidas nesse trabalho, pode-se implementar uma estratégia para o desligamento dos servidores na Nuvem a fim de economizar energia.

Abstract

The demand for multimedia services in mobile networks has increased in the last years. The high quantity of users' mobiles, both consuming and producing multimedia content to and from the Cloud, can outpace the available bandwidth capacity and incur low Quality of Experience (QoE). Despite the many benefits that Cloud Computing offers, such as high availability and scalability, multimedia services require a constant and continuous flow of packets with low latency, requirements that Cloud Computing, in certain situations, can not provide adequately. These services require a constant and continuous flow of packages with low latency. Furthermore, using Fog Computing, it is possible to improve on the issues mentioned above, being especially useful in latency-sensitive applications such nodes are physically much closer to devices than centralized data centers. According to requests, these services can be allocated on fog nodes that can virtualize their functions and migrate services. The hierarchical, distributed, and heterogeneous nature of computational instances makes these services' positioning in this environment a challenging task. Therefore, this dissertation proposes a method to design/create a hierarchical multi-tier Cloud-to-Fog network. The proposed method uses a bottom-up approach, starting from a set of base stations, and arranges new nodes hierarchically, from Edge to Cloud. Moreover, it introduces a novel multimedia service placement algorithm, named **SMART-FL**, for multi-tier Cloud-Fog environments modeled as a Capacitated Facility Location Problem. The goal is to select the minimum number of nodes, considering their hardware capacities for providing multimedia services, so that the latency for servicing all the demands is minimized. To further improve these service delivery, two models are considered for traffic flow prediction. The goal is to predict future demand and reserve the storage capacity of nodes to improve multimedia services' positioning. The performance assessment was composed of one month of real-world mobile network traffic data from Milan, Italy. The results are compared considering six multimedia services placement strategies and evaluated in terms of latency, package delivery, requests attempted, and network usage. The results show that our algorithm can achieve the right balance among the Fog nodes' geographical location along with their hardware capacity and the users' location. Hence, this solution enhances the quality of experience since the response time is lower in comparison to the Cloud tier, reducing the latency. Due to data traffic volume prediction, positioning becomes even more efficient due to previously reserved nodes' storage. Processing and storage near the data source improve the services delivered to end-users. For example, a fog node can be responsible for the video stream, which is quicker than sending the Cloud's request for centralized processing. Furthermore, using the information obtained in this work, it is possible to implement a strategy for shutting down servers in the Cloud to save energy.

Lista de Figuras

Lista de Tabelas

Lista de Abreviações e Siglas

ARIMA	Auto-Regressivo Integrado de Médias Móveis
ARIMA-PRED	Auto-Regressivo Integrado de Médias Móveis-prediction
C-RAN	Cloud Radio Access Network
CDR	Registro de Detalhe da Chamada
CFLP	Capacitated Facility Location Problem
CloudSim	Cloud Simulator
FLP	Facility Location Problem
GRU	Gated Recurrent Unit
IFogSim	IoT and Fog Simulator
IoT	Internet das Coisas
LSTM	Long Short-Term Memory
LSTM-PRED	Long Short-Term Memory-prediction
MAE	Erro Médio Absoluto
MIPS	Memória de Acesso Aleatório
MSP	Multimedia Service Placement
MultiTierFogSim	MultiTier Cloud, Fog, and Edge Simulator
PLI	Programação Linear Inteira
QoE	Qualidade de Experiência
QoS	Qualidade de Serviço
RAM	Memória de Acesso Aleatório
RAN	Rede de Acesso Via Rádio
RMSE	Raiz Quadrada do Erro Médio
RNA	Redes Neurais Artificiais
RNN	Redes Neurais Recorrentes

TCS	Sistema de Controle de Tráfego
UFLP	Uncapacitated Facility Location Problem

Sumário

Capítulo 1

Introdução

Nos últimos anos, houve uma rápida proliferação de uma ampla gama de serviços multimídia, como vídeo sob demanda, videoconferência, transmissão de ambientes 3D interativos, vídeos com alta definição, *streaming* de vídeo com resolução 4k/8k (*Ultra-high-definition video* - UHD), dentre outros [37, 38]. Esses serviços já representam a maior parte do tráfego global e até 2021 inundarão as redes móveis exigindo alta velocidade e baixa latência sem precedentes [6, 58].

De acordo com relatórios técnicos disponibilizados pela empresa Cisco Systems [25], 73% de todo o tráfego IP global gerado na Internet em 2019 foi referente a tráfego de vídeo sobre IP e 1% referente a tráfego de *gaming*, com projeções de que esses percentuais saltem para 82% e 4%, respectivamente, para o ano de 2021. A Figura 1.1 ilustra esse crescimento. Aliás, a adoção da 5ª geração de sistemas sem fio (5G) permitirá que esse crescimento seja ainda maior devido à sua alta capacidade de largura de banda e baixa latência.

Os estudos mais recentes sobre os hábitos do consumidor durante a pandemia, e o que pode permanecer depois, mostra que o tráfego de serviços multimídia atingiu um pico de ≈ 60% maior do que os níveis de janeiro, quando iniciou o *lockdown* em alguns países [24, 48]. Essa alta demanda terá impacto na Qualidade de Serviço (QoS) e na Qualidade de Experiência (QoE). A degradação da QoS e QoE é ainda maior quando as requisições são realizadas por usuários em veículos com alta mobilidade [39].

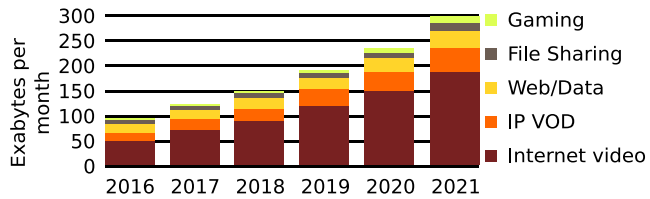


Figura 1.1: Tráfego IP global por categoria de aplicação. Dados de 2016 e projeções para 2021. Fonte: Cisco Systems (2017b).

Apesar dos muitos benefícios que a Computação em Nuvem oferece, como alta disponibilidade, escalabilidade e interoperabilidade, os serviços multimídia exigem fluxo constante e contínuo de pacotes com baixa latência [36], requisitos que a Computação em Nuvem ocasionalmente não fornece [86]. Adaptar esses serviços nesse ambiente é uma tarefa não

trivial [8]. A propósito, mesmo com as melhorias nas tecnologias sem fio oferecidas pela rede 5G, a entrega de vídeo confiável e de alta qualidade ainda impõe vários desafios, por exemplo, como lidar com um grande número de dispositivos heterogêneos e atender aos requisitos cada vez maiores dos usuários [61]. Para superar esses e outros problemas, é desejável o uso de uma arquitetura distribuída que armazena e processa os serviços de forma lógica: entre a Nuvem e a fonte de dados.

A Computação em Névoa (*Fog Computing*) e Borda (*Edge Computing*) apresentam-se como uma solução em conjunto para atender esses serviços sensíveis a latência, onde a gestão de todos os recursos acontece de forma coordenada e em camadas, desde a Nuvem até os dispositivos finais. O principal objetivo é alocar recursos da Nuvem fisicamente próximos aos usuários finais [73]. Por um lado, a Computação de Borda executa todo o processamento ou grande parte nos dispositivos finais, conhecidos como dispositivos de borda. Por outro lado, a Computação em Névoa estende a borda da rede e consiste na alocação do poder de processamento próximo do limite da rede, entre a Nuvem e a borda. Ambas as arquiteturas compartilham benefícios semelhantes em comparação a Computação na Nuvem, incluindo uma redução na latência para milissegundos, diminuição no congestionamento da rede e o reconhecimento da localização geográfica dos usuários em tempo real [49]. Essa proximidade com os usuários garante uma conexão mais consistente e alta largura de banda, facilitando a implementação de novos serviços que a Computação em Nuvem não suporta, particularmente, aqueles que exigem garantias de QoE.

Os serviços são fornecidos através de nós névoas e bordas que são capazes de virtualizar suas funções e migrar os serviços de acordo com as requisições. Esses nós possuem capacidades computacionais para se comunicarem com uma variedade de dispositivos, sensores e atuadores, oferecendo serviços baseados em dados coletados e, eventualmente, processados/filtrados localmente. A distribuição dos serviços ao longo da rede e próximo aos usuários através desses nós permite o processamento e armazenamento perto da fonte de dados, sem a necessidade do envio de todos esses serviços para a Nuvem remota ou para outros sistemas centralizados [68]. Juntamente com algumas características da rede, como a qualidade do enlace e carga da célula, esses serviços podem ser posicionados de forma adequada. Com isso, é possível reduzir ainda mais a latência, além de fornecer alta disponibilidade e resiliência [72]. A natureza hierárquica, distribuída e heterogênea das instâncias computacionais torna o posicionamento desses serviços nesses ambientes uma tarefa desafiadora [26, 63]. Por exemplo, posicionar servidores névoas de forma a otimizar a entrega de conteúdo é um desafio em aberto [75].

Existem várias estratégias de posicionamento de serviços multimídia. Dentre elas, destacam-se a *first fit* e *best fit*. Ambas estratégias posicionam serviços a partir das condições da rede por meio da Rede de Acesso Via Rádio (*Radio Access Network* - RAN) bem como da popularidade do conteúdo, localização geográfica do usuário e até mesmo do volume do tráfego da rede. Algoritmos baseados na estratégia *first fit* sempre iniciam a busca por recursos selecionando nós pertencentes a camada mais próxima do usuário. Caso não encontre recursos suficientes na camada atual, a camada superior é considerada. Caso contrário, a busca é encerrada e os serviços são posicionados na camada atual. Esses algoritmos são eficientes e consomem menos recursos computacionais para a busca. Entretanto, não exploram o uso eficiente dos recursos da rede e dos nós, como a latência

mínima necessária para atender o serviço e/ou a localização geográfica dos nós. Algoritmos baseados na estratégia *best fit* selecionam o conjunto de nós mais apropriados independente da camada levando em consideração as características da rede, como a qualidade do enlace, capacidade de armazenamento, latência dos nós, dentre outros. Em contrapartida, esses algoritmos apresentam complexidade computacional superior aos baseados em *first-fit*.

Esses algoritmos podem ser avaliados em ambientes hierárquicos Nuvem-Névoa. Nesses ambientes, os nós são organizados hierarquicamente em camadas, desde a Borda até a Nuvem. Por um lado, nós que pertencem a mesma camada possuem recursos de redes (latência, taxa de *download* e *upload*,...) e computacionais (armazenamento, processamento, ...) semelhantes. Por outro lado, nós de diferentes camadas possuem tais recursos dessemelhante. A não disponibilidade desses ambientes torna-se a avaliação desses algoritmos um desafio.

Inicialmente, nesta dissertação de mestrado é proposto um método para a criação de ambientes hierárquicos Nuvem-Névoa. Esse método utiliza uma abordagem *bottom-up*, iniciando-se a partir de um conjunto $BS = \{bs_1, bs_2, ..., bs_{bs}\}$ de estações base e organiza novos nós hierarquicamente em camadas, produzindo um ambiente hierárquico Nuvem-Névoa [64]. Além do mais, é proposto um algoritmo denominado *be**S**t fit **M**ultimedi**A** se**R**vice placemen**T** with **F**acility **L**ocation* (**SMART-FL**) para o problema de posicionamento de serviços multimídia (*Multimedia Service Placement* - MSP) em ambientes hierárquicos Nuvem-Névoa modelado como um Problema de Localização de Facilidades Capacitadas (*Capacitated Facility Location Problem* - CFLP). A solução é implementada como Programação Linear Inteira (*Integer Linear Programming* - PLI) e o objetivo é encontrar o menor conjunto de nós considerando suas capacidades de armazenamento para fornecer serviços multimídia de forma que a latência seja minimizada. Dentre as principais vantagens de modelar problemas como PLI é a garantia de obter a melhor solução possível de problemas considerados complexos computacionalmente em tempo razoável.

Para melhorar ainda mais a entrega desses serviços, são considerados dois modelos, a saber Auto-Regressivo Integrado de Médias Móveis-*prediction* (ARIMA-PRED) e *Long Short-Term Memory-prediction* (LSTM-PRED), baseados nos modelos Auto-Regressivo Integrado de Médias Móveis (ARIMA) e *Long Short-Term Memory* (LSTM), respectivamente, para a predição do volume de tráfego da rede da cidade de Milão. O objetivo é reservar um conjunto de nós para alocar esses serviços através do volume de tráfego predito. Ambos são analisados em relação a vários parâmetros para se aproximar da otimalidade. Em seguida, são avaliados em relação as métricas Erro Médio Absoluto (Mean Absolute Error - MAE) e Raiz Quadrada do Erro Médio (Root-Mean-Square Error - RMSE). Ambos apresentam resultados satisfatórios. A escolha desses métodos deve-se ao fato da ampla utilização para previsões em tempo real, tais como para previsões de fluxos de tráfego de rede, sendo utilizados em diversos trabalhos [52].

O algoritmo **SMART-FL** é avaliado em um ambiente hierárquico Nuvem-Névoa utilizando o simulador **MultiTier** Cloud, **Fog**, and Edge **Sim**ulator (MultiTierFogSim). Nesse ambiente, os nós são organizados hierarquicamente em quatro camadas: **Nuvem**, **Nuvem Regional**, **Cloudlets** e **Estação Base**. Esse ambiente de simulação considera recursos relacionados a Computação em Nuvem-Névoa, como localização geográfica das requisições, migrações de serviços de e para qualquer camada, suporte a mobilidade dos

usuários, dentre outros. Esse simulador é uma extensão do *MobFogSim* e também é uma contribuição desta dissertação.

Os resultados são avaliados em termos da latência, pacotes entregues, requisições atendidas e o uso da rede em termos de (a) volume total de dados transmitidos durante a migração e (b) uso do enlace. De acordo com os resultados, o algoritmo **SMART-FL** posiciona os serviços multimídia em nós com capacidade de armazenamento adequada, próximos aos usuários e com latência média inferior a todas as estratégias. Devido ao volume de tráfego predito, o posicionamento torna-se ainda mais eficiente em razão dos nós previamente reservados. Essa distribuição dos serviços ao longo da rede e próximos aos usuários permite o processamento e armazenamento perto da fonte de dados, sem a necessidade do envio para a Nuvem remota ou para outros sistemas centralizados. Consequentemente, o uso da rede total também é reduzido, pois menos canais de transmissão de dados são utilizados, diferentemente quando os serviços estão posicionados na Nuvem. Vale a pena mencionar que, o algoritmo **SMART-FL** pode ser adaptado, por exemplo, para serviços de Sistema de Controle de Tráfego (Traffic Control System - TCS), Internet das Coisas (Internet of Things - IoT), realidade aumentada e outros que também aproveitam das vantagens oferecidas pelo ambiente hierárquico Nuvem-Névoa. A avaliação da QoE consta como trabalhos futuros.

1.1 Objetivo

A rede 5G é impulsionada pela evolução de serviços mais exigentes da atualidade, dentre eles, os multimídia, que exigem alta velocidade de processamento/armazenamento e baixa latência. O objetivo dessa dissertação de mestrado é propor um algoritmo para o problema de posicionamento de serviços multimídia em ambientes hierárquicos Nuvem-Névoa. A distribuição desses serviços ao longo da rede e próxima aos usuários concebe vários benefícios, dentre eles, a redução da latência e uso da rede. Um método para a criação de ambientes hierárquico Nuvem-Névoa também é desenvolvido. Além disso, dois modelos para a predição do volume de tráfego são considerados. O objetivo é reservar um conjunto de nós para alocar esses serviços através do volume de tráfego predito. Um simulador para avaliar serviços, incluindo os multimídias, nesses ambientes também é desenvolvido.

1.1.1 Objetivos específicos

Para alcançar o objetivo geral foram definidos os seguintes objetivos específicos:

1. **Definir um ambiente hierárquico Nuvem-Névoa.** O ambiente é definido a partir do método proposto que tem como entrada um conjunto de estações base e organiza novos nós hierarquicamente em camadas, produzindo um ambiente hierárquico Nuvem-Névoa.

2. **Determinar quais os requisitos que influenciam na decisão do posicionamento dos serviços multimídia em ambientes Nuvem-Névoa.** Dentre os

requisitos, pode-se citar a carga da célula, capacidade de armazenamento dos nós, latência, *bit rate*, taxas de perdas e erros, assim como a localização do usuário. Neste trabalho, somente alguns desses requisitos são considerados com base nos trabalhos relacionados [39, 40].

3. **Projetar e implementar um algoritmo para o posicionamento de serviços multimídia em ambientes hierárquicos Nuvem-Névoa.** O objetivo é encontrar o menor conjunto de nós considerando suas capacidades de armazenamento para prover tais serviços de forma que a latência seja minimizada.

4. **Definir um modelo para a predição do volume de tráfego da rede a fim de melhorar o posicionamento dos serviços multimídia.** Diversos modelos estão disponíveis na literatura para a predição do volume de tráfego da rede celular. O objetivo é reservar um conjunto de nós para alocar esses serviços através do volume de tráfego predito.

5. **Desenvolver/Estender um simulador para avaliar serviços, incluindo os multimídias, em ambientes hierárquicos Nuvem-Névoa.** As ferramentas e simuladores disponíveis já implementam alguns componentes importantes para simular e avaliar esses serviços em ambientes de Nuvem, Névoa ou Borda. No entanto, nenhum permite a avaliação desses serviços em ambientes hierárquicos Nuvem-Névoa. Além disso, alguns não suportam migrações de serviços e mobilidade dos usuários. Portanto, torna-se necessário implementar/estender um simulador para superar essas e outras limitações.

6. **Analisar e comparar diferentes estratégias de posicionamentos dos serviços multimídia.** O objetivo é analisar o quão eficiente são as estratégias **SMART-FL** e **SMART-FL** ciente da predição do volume de tráfego (**SMART-FL + predição**) em comparação com algumas encontradas na literatura.

1.2 Organização

Esta dissertação de mestrado está organizada da seguinte forma:

- **Capítulo 2 - Fundamentação Teórica:** Apresenta os conceitos fundamentais utilizados para o desenvolvimento deste trabalho.

- **Capítulo 3 - Trabalhos Relacionados:** Apresenta os trabalhos disponíveis na literatura relacionados ao tema de pesquisa deste trabalho dividido em duas seções. A primeira seção apresenta propostas de soluções para o problema de posicionamento de serviços multimídia em ambientes hierárquicos Nuvem-Névoa. A segunda seção discute várias propostas de arquiteturas hierárquicas baseadas nesses ambientes.

- **Capítulo 4 - Posicionamento de Serviços Multimídia em Ambientes Hierárquicos Nuvem-Névoa:** Apresenta a modelagem do cenário base utilizado,

assim como o método proposto para a criação de ambientes hierárquicos Nuvem-Névoa e a formulação, modelado em PLI, para o problema de posicionamento de serviços multimídia. Também descreve o processo da modelagem da série temporal e os parâmetros utilizados para a predição do volume de tráfego, assim como o simulador estendido e o ambiente simulado.

- **Capítulo 5 - Resultados:** Apresenta e discute os resultados alcançados.
- **Capítulo 6 - Conclusão:** Por fim, as propostas são revisadas, resumindo as contribuições e indicando direções futuras para os problemas abordados.

Capítulo 2

Fundamentação Teórica

Este capítulo apresenta os conceitos fundamentais para o desenvolvimento deste trabalho, divididos em três seções. A Seção 2.1 introduz o tema sobre Computação em Nuvem e Névoa. A Seção 2.2 apresenta conceitos básicos sobre otimização e a modelagem dos problemas de Localização de Facilidades Capacitadas e não Capacitadas, que trata-se da ideia base utilizada neste trabalho para modelar o algoritmo **SMART-FL**. Finalmente, a Seção 2.3 introduz brevemente conceitos de séries temporais e apresenta dois modelos de predição, ARIMA e LSTM, utilizados como base para os dois modelos analisados, ARIMA-PRED e LSTM-PRED, para a predição do volume de tráfego da rede.

2.1 Computação em Nuvem e Névoa

A Computação em Nuvem é caracterizada pela oferta de recursos, tais como armazenamento e processamento para aplicações e serviços por meio da Internet e entregues conforme a demanda [3]. Tais recursos podem ser provisionados e liberados de forma transparente aos usuários, podendo ser acessados de qualquer lugar do mundo, a qualquer hora, com esforço mínimo de gerenciamento. O acesso é realizado remotamente através da Internet - daí a alusão à nuvem. Com o rápido crescimento do número de dispositivos conectados à Internet com restrições à latência, torna-se muito difícil para a Nuvem acomodá-los de forma eficiente. A abordagem centralizada adotada pela sua arquitetura apresenta problemas de desempenho para atender aplicações sensíveis à latência [28]. Uma solução é a aproximação desses recursos para a borda da rede como tentativa de amenizar os danos causados a essas aplicações sensíveis a latência [10].

A Computação em Névoa surge como uma tecnologia promissora que envolve a transferência de recursos da Nuvem em direção aos dispositivos finais [49]. O objetivo é diminuir a latência, tendo em vista que os dados não precisam ser enviados dos dispositivos para um sistema de processamento central (Nuvem) e, em seguida, para os dispositivos. É uma arquitetura descentralizada onde os dados, processamentos, comunicações, armazenamentos, medições, aplicações e gerenciamentos são distribuídos no local mais lógico e eficiente: entre a fonte de dados e a Nuvem [10]. Em certas ocasiões, o processamento ocorre diretamente nos dispositivos finais aos quais os sensores estão conectados ou em um dispositivo de *gateway* que está fisicamente próximo dos sensores [15]. Esse cenário

também é comumente chamado de Computação em Borda. Essa tecnologia é apresentada como uma extensão à arquitetura de Computação em Nuvem [10, 19], pois, ao processar dados localmente e acelerar os fluxos de dados, a Computação em Névoa reduz a quantidade de dados transmitidos na rede e também a complexidade computacional necessária na Nuvem.

No modelo teórico da Computação em Névoa, entidades de rede, como *gateways*, roteadores, *switches* e servidores são conhecidas como nós névoas e são usadas para fins computacionais. A gestão de todos os recursos acontece de forma coordenada e em camadas (desde a Nuvem até a Borda), propiciando serviços colaborativos baseado em *clusters* de recursos e compartilhamento de informações. Baseado em um ambiente virtualizado, esses nós são capazes de executar serviços virtualizados baseados em *containers* [32]. Isso remove as dependências de infraestruturas subjacentes e facilita a migração dos serviços entres os nós névoas, o que reduz a complexidade de lidar com diferentes plataformas. Dessa forma, os serviços podem ser executados de forma distribuída ao longo da rede e próximo aos usuários [12]. Conforme ilustrado na Figura 2.1, a Computação em Névoa atua entre a Nuvem e os dispositivos finais, incluindo a fonte dos dados.

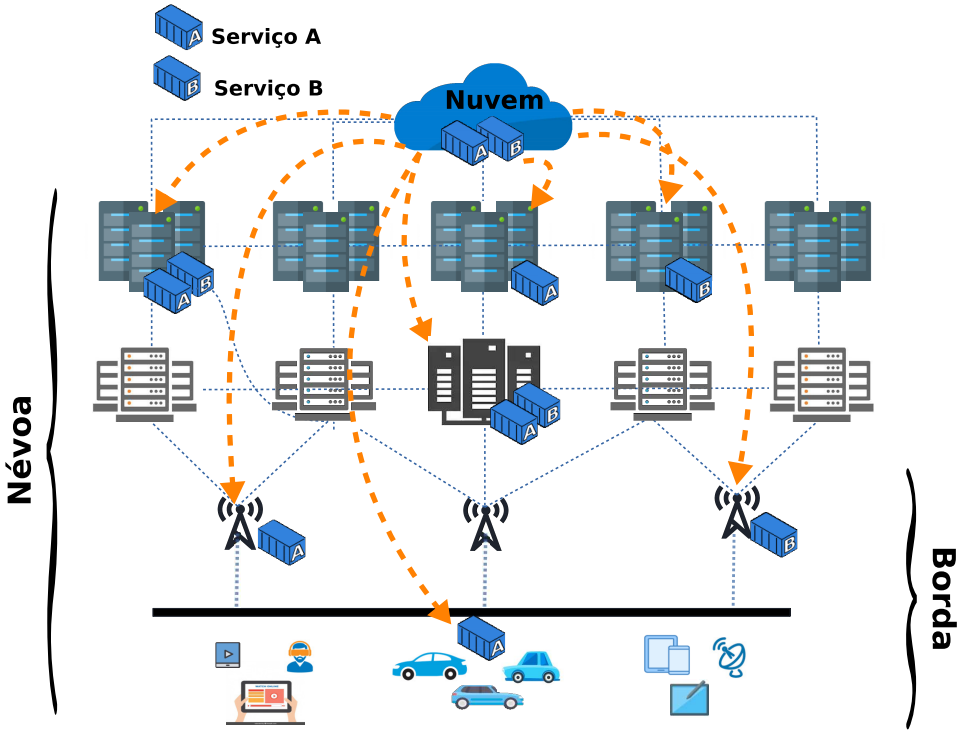


Figura 2.1: Arquitetura de Névoa multicamada.

A migração de serviços da Nuvem para os nós névoas e entre os nós névoas pode ser ativada pelos padrões de solicitações das requisições dos usuários finais. A natureza hierárquica, distribuída e heterogênea das instâncias computacionais torna o posicionamento de serviços para diversas aplicações na Nuvem uma tarefa desafiadora.

2.2 Otimização combinatória

Os problemas em otimização combinatória, em geral, se resumem em encontrar, dentre todos os possíveis subconjuntos, aquele de valor ótimo, que pode ser de valor mínimo, caso o problema seja de minimização, ou máximo, caso o problema seja de maximização. Como exemplos clássicos, destacam-se o problema do caixeiro viajante, da mochila e cobertura mínima por conjuntos [77]. Uma forma de resolver é simplesmente enumerar todas as soluções possíveis e guardar aquela de valor ótimo. Entretanto, para qualquer problema de um tamanho minimamente interessante, este método torna-se impraticável, já que o número de soluções possíveis são computacionalmente intratáveis. Estratégias que apresentam métodos exatos e eficientes de resolução e tem tido sucesso no tratar destes problemas são os Algoritmos Probabilísticos, Algoritmos de Grafos e PLI [56].

Dentre esses métodos, a PLI consiste em expressar um problema em termos de variáveis contínuas e um conjunto de restrições lineares sobre essas variáveis [65]. Modelada uma função objetivo que descreve como é calculado os custos a ser minimizado e as restrições, implementa-se um algoritmo que resolve o problema de forma eficiente. Dessa forma, problemas de como determinar o corte mais adequado de placas de modo que o desperdício seja minimizado ou determinar quais os melhores locais para instalação de fábricas de forma que todos os clientes sejam atendidos a um custo mínimo, podem ser modelados de forma a encontrar uma solução exata através do método de PLI.

A Subseção 2.2.1 detalha o problema de Localização de Facilidades modelado como PLI, que trata-se da ideia base utilizada neste trabalho para modelar o algoritmo **SMART-FL**.

2.2.1 Localização de facilidades

Os Problemas de Localização de Facilidades (Facility Location Problem - FLP) tratam de decisões sobre um conjunto de clientes que precisam ser atendidos e um conjunto de possíveis locais para instalação de facilidades (fábricas/armazéns). O objetivo é determinar quais são os melhores locais para instalação das facilidades de forma que todos os clientes sejam atendidos a um custo mínimo [23]. As facilidades podem ou não ter capacidades limitadas de atendimento, que classificam o problema em duas variantes: capacitadas e não capacitadas [1]. O problema de localização de facilidades capacitadas é a base de muitos problemas práticos de otimização, onde a demanda total que cada instalação pode atender é limitada. No entanto, para um modelo de FLP sem capacidade (Uncapacitated Facility Location Problem - UFLP), é assumido que a demanda produzida e enviada de cada instalação é ilimitada [74].

Por exemplo, considere uma empresa com três possíveis locais para instalar seus armazéns e atender cinco pontos de demandas. Cada armazém tem um custo anual de ativação f_i, isto é, uma despesa anual de aluguel incorrida para usá-lo, independentemente do volume que atende. A variável contínua $y_{ij} \leq 0$ define o volume atendido pelo armazém i ao ponto de demanda j; b_j é o volume solicitado pelo ponto de demanda j; e u_i quantidade máxima de volume que pode ser produzido pela instalação i. Além disso, há um custo de transporte c_{ij} por unidade atendida do armazém i até o ponto de demanda

j. O objetivo do problema é minimizar a soma dos custos de ativação dos armazéns e os custos de transporte para que todos os pontos de demandas sejam atendidos. A Figura 2.2 ilustra esse exemplo. Nesse caso, a escolha dos armazéns A e C minimizam a soma dos custos para atender todos os clientes.

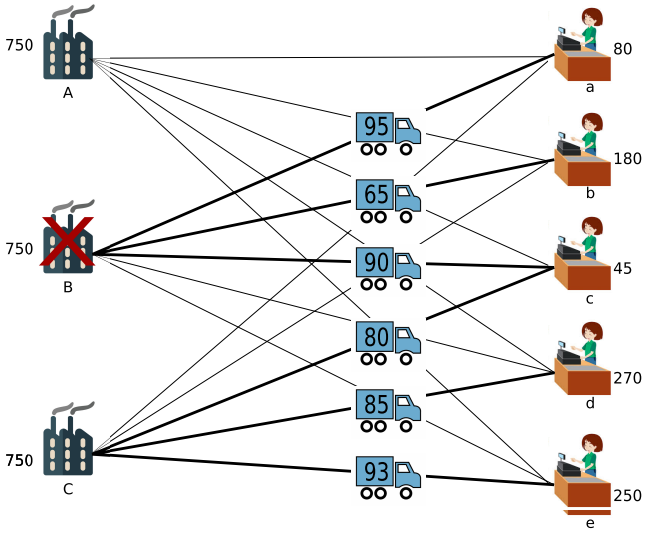


Figura 2.2: Localização de Facilidades

Um modelo de PLI para o **CFLP** é então dado por

Minimiza

$$\sum_{i=1}^{n} f_i \cdot x_i + \sum_{i=1}^{n} \sum_{j=1}^{m} c_{ij} \cdot y_{ij} \tag{2.1}$$

sujeito a

$$\sum_{i=1}^{n} y_{ij} = b_j \qquad \forall j = 1, ..., m \tag{2.2}$$

$$\sum_{j=1}^{m} b_j \cdot y_{ij} \leq u_i \cdot x_i \qquad \forall i = 1, ..., n \tag{2.3}$$

$$y_{ij} \geq 0 \qquad \forall i = 1, ..., n \; and \; \forall j = 1, ..., m \tag{2.4}$$

$$x_i \in \{0, 1\} \qquad \forall i = 1, ..., n \tag{2.5}$$

onde, a variável binária x_i é 1 se o armazém i estiver aberto ou x_i é 0, caso contrário. A restrição 2.2 exige que a demanda de cada ponto de demanda j seja satisfeita. A restrição 2.3 garante que se $x_i = 0$, ou seja, o armazém i não estiver aberto, então $y_{ij} = 0$ para todos os j, ou seja, nenhuma demanda para qualquer cliente pode ser atendida a partir do armazém i.

Para o **UFLP**, a restrição 2.3 é substituída por

$$\sum_{j=1}^{m} b_j \cdot y_{ij} \leq M \cdot x_i \; \forall i = 1, ..., n \tag{2.6}$$

$$\tag{2.7}$$

onde, M é uma constante. A Tabela 2.1 resume as notações utilizadas para ambos os modelos **CFLP** e **UFLP**.

Tabela 2.1: Notação utilizada para os modelos CFLP e UFLP.

Parâmetros de entrada	
Notação	**Descrição**
f_i	custo anual de ativação do armazém i.
c_{ij}	custo de transporte do armazém i para o ponto de demanda j.
u_i	volume máximo produzido pela armazém i.
b_j	volume solicitado pelo ponto de demanda j.
M	constante relativamente grande.
n	quantidades de armazéns.
m	quantidades de pontos de demanda.
Variáveis de decisão	
y_{ij}	volume atendido pelo armazém i ao ponto de demanda j.
x_i	**1** indica se o armazém i está aberto; **0** caso contrário.

2.3 Séries temporais

Série temporal é uma coleção de observações feitas sequencialmente ao longo de intervalos, geralmente uniformes. Por exemplo, as taxas de desemprego mensais para os últimos cinco anos, produção diária em uma fábrica durante um mês ou a quantidade de usuários conectados por minuto a uma estação base durante um ano [22].

A análise de séries temporais tem como objetivo identificar padrões não aleatórios de uma variável de interesse. A observação deste comportamento passado pode permitir fazer previsões sobre o futuro, orientando a tomada de decisões. A maneira tradicional de realizar essa análise é através da sua decomposição em três componentes [27], os quais são:

- **Tendência**: indica o seu comportamento de longo prazo, isto é, se ela cresce, decresce ou permanece estável, e qual a velocidade destas mudanças.
- **Ciclo**: são caracterizados pelas oscilações de subida e de queda nas séries, mais ou menos regulares, relacionadas a uma sequencia de tendencia.
- **Sazonalidade**: é a identificação de padrões regulares da série de tempo, isto é, às oscilações de subida e de queda que sempre ocorrem em um determinado período do ano, do mês, da semana ou do dia.

A Figura 2.6 ilustra esses três componentes.

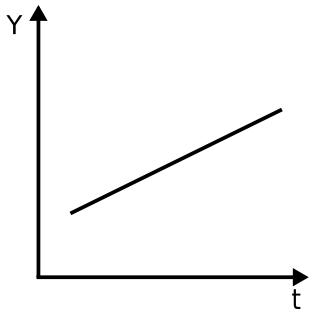


Figura 2.3: Tendência a longo prazo.

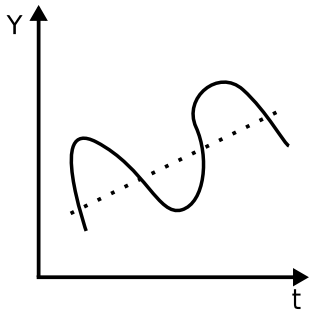


Figura 2.4: Tendência a longo prazo e movimento cíclico.

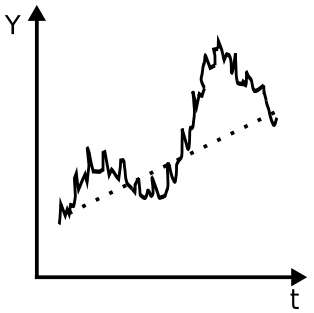


Figura 2.5: Tendência a longo prazo, movimento cíclicos e por estações.

Figura 2.6: Tendência, Ciclo e Sazonalidade.

Assim, pela identificação desses componentes, a predição para períodos de tempo subsequentes ao observado pode ser desenvolvida.

A maioria dos métodos de predição de séries temporais se baseiam na suposição de que as observações passadas contém todas as informações sobre o padrão de comportamento da série temporal e esse padrão é recorrente no tempo. Na literatura, há inúmeros métodos que descrevem o comportamento de uma série temporal e eles são divididos em duas abordagens: clássica e de aprendizado [45]. Por um lado, a abordagem clássica pertencente aos métodos de Suavização Exponencial e ARIMA, que utilizam da estatística paramétrica para transformar seu conjunto de dados numa distribuição de probabilidade conhecida, e, portanto, necessitam conhecer o comportamento da série temporal [27]. Por outro lado, as abordagens de aprendizado pertencentes às Redes Neurais LST, Unidades Recorrentes Bloqueadas (Gated Recurrent Unit - GRU) e Redes Neurais Recorrentes (Recurrent Neural Network - RNN), em oposição aos modelos clássicos, não dependem do conhecimento prévio das propriedades da série temporal [2]. As Redes Neurais são mais simples de serem ajustadas e demostram considerável desempenho mesmo quando aplicados às séries complexas e altamente não lineares.

As subseções seguintes descrevem dois métodos de ambas abordagens, o ARIMA e a Rede Neural, em especial, a rede LSTM, utilizadas neste trabalho como modelos bases para predição de volume de tráfego da rede.

2.3.1 ARIMA

O modelo ARIMA é utilizado na compreensão de séries temporais ou predição de um ponto no futuro. Qualquer série temporal que exiba padrões e não seja um ruído branco aleatório pode ser modelada com os modelos ARIMA [84].

A parte auto-regressiva (**AR**) do método indica que a variável de interesse sofre uma regressão em seus valores anteriores. A parte integrada (**I**) indica que os valores de dados foram substituídos com a diferença entre seus valores atuais e anteriores, tornando a série estacionária (este processo pode ser realizado mais de uma vez). A parte de média móvel (**MA**) indica que o erro de regressão é uma combinação linear dos termos de erro aplicado a observações passadas. O propósito de cada componente é fazer o modelo se ajustar aos dados da melhor forma possível [20].

Os modelos ARIMA não sazonais são geralmente denotados como ARIMA**(p, q, d)**, onde:

- **p**: é o número de defasagens do modelo auto-regressivo.
- **d**: é o número de vezes em que os dados tiveram valores passados subtraídos.
- **q**: é a ordem do modelo de médias móveis.

Geralmente, os modelos ARIMA apresentam melhores resultados quando a série é relativamente longa e bem comportada. Se a série é muito irregular, os resultados são, geralmente, inferiores aos obtidos por outros métodos, como as redes neurais recorrentes.

2.3.2 Redes neurais artificiais

As Redes Neurais Artificiais (RNAs) são modelos matemáticos inspirados na estrutura neural de organismos inteligentes com capacidade de adquirir conhecimento através da experiência. Tal como humanos aplicam o conhecimento adquirido de experiências passadas para novos problemas ou situações, uma rede neural utiliza exemplos resolvidos previamente para construir um sistema de neurônios que toma novas decisões, realizando classificações ou previsões [45].

Uma RNA é composta por várias unidades de processamento, conectados entre si por canais de comunicação que são associados a um determinado peso. Cada unidade, ou neurônio, realiza operações apenas sobre seus dados locais, que são entradas recebidas pelas suas conexões. O comportamento inteligente vem das interações entre as unidades de processamento. A Figura 2.7 ilustra uma RNA com três, dois e um neurônio na camada de entrada, intermediária e saída, respectivamente.

A principal característica das RNAs é a capacidade de aprender, ajustando os pesos das interconexões entre as camadas. As respostas que a rede produz são comparadas com as respostas corretas, e cada vez os pesos das conexões são ajustados na direção das respostas corretas. Por um lado, as RNAs são utilizadas para resolver problemas que são complexos computacionalmente, pois são efetivas quanto a qualidade dos resultados. Por outro lado, as RNAS não apresentam a capacidade de processar e armazenar informações temporais e sinais sequenciais. Redes Neurais Recorrentes resolvem esse problema através de *loops*, permitindo que as informações persistam.

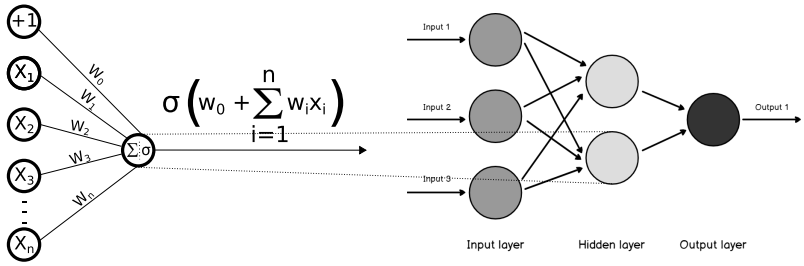


Figura 2.7: Exemplo de Rede Neural Artificial.

Redes Neurais Recorrentes

As RNN são classes de Redes Neurais especialmente úteis para processar dados sequenciais. Essas redes dispõem como entrada não apenas a entrada atual, mas também o que perceberam anteriormente no tempo. Isto é, a resposta obtida em $t-1$ afeta a decisão que alcançará um momento mais tarde em t, como ilustrado na Figura 2.8. Assim, as RNNs têm duas fontes de entrada, o presente e o passado recente, que se combinam para determinar como respondem aos novos dados [2].

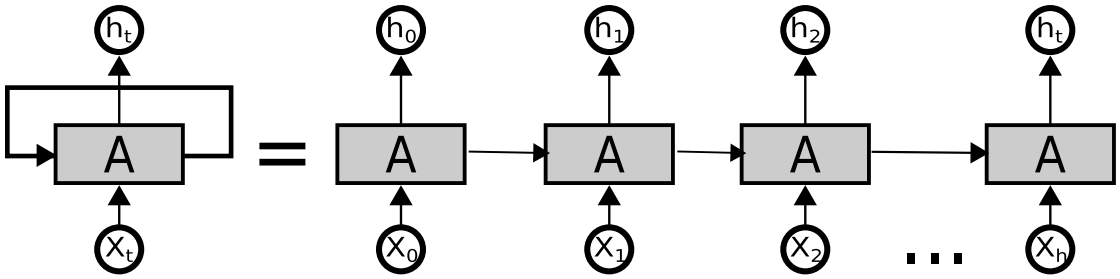


Figura 2.8: Rede Neural Recorrente.

Apesar do conceito de reciclar informações anteriores para serem usadas no processo corrente ser muito atrativo, à medida que observações históricas crescem, as RNNs tornam-se incapazes de aprender a conectar as informações. As Redes LSTM são um tipo especial de RNN capaz de aprender dependências de longo prazo.

Redes LSTM

As redes LSTM são um subconjunto de RNAs projetadas para reconhecerem padrões e dependências de longo prazo, ideais para classificar, processar e prever séries temporais com intervalos de tempo de duração desconhecida. Isso é possível devido a capacidade de remover ou adicionar informações ao estado da célula, reguladas por estruturas chamadas portões [34].

O estado da célula, em teoria, atua como uma via que transporta informações relevantes por toda a cadeia de sequência. As informações são adicionadas ou removidas ao estado da célula através dos portões, que decidem quais informações são permitidas no estado da célula. Eles aprendem quais informações são relevantes para manter ou esquecer durante o treinamento. Em geral, uma rede LSTM possui três portões:

- **Forget Gate**: remove as informações que não são úteis ao estado da célula.

- **Input Gate**: adiciona informações úteis ao estado da célula.
- **Output Gate**: Extrai informações úteis do estado da célula atual para decidir qual deve ser o próximo estado oculto.

A Figura 2.9 ilustra como os dados fluem através de uma célula de memória e são controlados por seus portões.

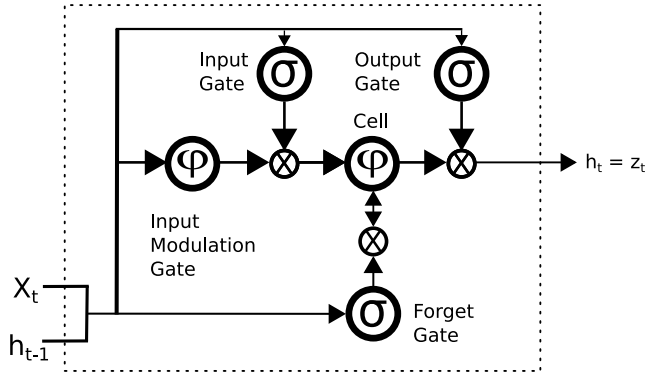


Figura 2.9: Célula de uma rede LSTM.

As redes LSTMs podem ser aplicadas a uma variedade de tarefas de aprendizado profundo, que incluem principalmente predição com base em informações anteriores. Os exemplos incluem predição de texto, ações comerciais e volume de tráfego de rede.

Capítulo 3

Trabalhos Relacionados

Este capítulo apresenta os trabalhos da literatura relacionados ao tema de pesquisa deste trabalho. A Seção 3.1 apresenta propostas de soluções para o problema de posicionamento de serviços multimídia em ambientes Nuvem-Névoa. A Seção 3.2 discute várias propostas de métodos para a criação de ambientes hierárquicos baseados nesses ambientes.

3.1 Posicionamento de serviços multimídia

Proporcionar QoE à entrega de vídeo em redes sem fio têm atraído pesquisadores ao longo dos anos [35, 47, 62]. Diferentes arquiteturas foram propostas para a entrega de vídeos, desde adaptações na arquitetura em Nuvem [86], armazenamento em cache D2D, SBS, MBS [17], até o uso de nós névoas e bordas para diminuir a latência e melhorar a QoE [63].

O posicionamento ideal de serviços de distribuição de vídeo assistido pela Nuvem é abordada pelos autores em [30]. O objetivo é minimizar conjuntamente o custo operacional (aluguel por recursos da Nuvem) e a latência. O problema de minimização conjunta é resolvido por um algoritmo *offline* projetado com base na solução de Barganha de *Nash*. O algoritmo considera a demanda predita do usuário final em intervalos de tempo futuros. Um algoritmo online também foi projetado para minimizar o custo operacional e a probabilidade de subprovisionamento de recursos. A cada intervalo de tempo, a demanda predita do usuário final é usada para acionar o ajuste ou reimplementação de recursos nos *data centers*. Os resultados mostram que os algoritmos propostos podem alcançar um bom equilíbrio entre vários objetivos e otimizar efetivamente o custo operacional e a experiência do usuário. No entanto, é notado que a Computação em Nuvem não oferece uma qualidade de experiência adequada em áreas com alta demanda por esses serviços [63]. Em nosso trabalho, em vez de concentrar dados e computação na Nuvem, consideramos que partes ou todos os serviços na Nuvem devem migrar para nós névoas localizados próximo ao usuário, atendendo às suas necessidades em relação à latência. Este é um dos fatores que constituem a principal motivação deste trabalho.

Uma solução para a alocação de serviços em ambientes de várias camadas é proposto por [69]. A estratégia combina nós nuvem e névoas para fornecer baixa latência para os serviços solicitados. A hierarquia das camada é determinada pela capacidade, vizinhança e acessibilidade para os usuários finais. O problema é modelado com PLI para otimização

da latência. Os resultados comprovam os benefícios da distribuição de serviços entre vários nós névoas, evitando o acesso de alto atraso nas camadas superiores. Dessa forma, algumas solicitações de usuários podem ser atendidas diretamente por nós névoas próximos a borda da rede. Isso diminui significativamente o custo da largura de banda na entrega de *streaming* adaptável de provedores de conteúdo para usuários finais. Além disso, os autores discutem a vantagem da distribuição de serviços entre os nós névoas com diferentes capacidades de computação e armazenamento. No entanto, o algoritmo proposto tem um tempo adicional considerável para a distribuição dos serviços em horário de pico. A predição do volume de tráfego minimizaria esse tempo de alocação.

O problema de armazenamento em cache em Redes de Acesso a Rádio baseadas em Computação em Névoa é abordado em [44]. Os autores desenvolveram estratégias de posicionamento de cache com reconhecimento de transmissão centralizada e distribuída para minimizar o atraso médio do *download* dos usuários, sujeito às restrições de capacidade de armazenamento dos nós. No modelo centralizado, o problema de colocação de cache é modelado como um problema de maximização submodular com restrição de matróides, e um algoritmo de aproximação é proposto para encontrar uma solução dentro de um fator constante ao ideal. No modo distribuído, é proposto um algoritmo distribuído baseado em propagação de crença para fornecer uma solução abaixo do ideal, com atualizações iterativas em cada estação base com base nas informações coletadas localmente. Os resultados mostraram que ambos os algoritmos propostos podem não apenas melhorar a probabilidade de acertos dos usuários ao cache, mas também fornecer oportunidades de transmissão cooperativa mais flexíveis para os usuários. No entanto, devido às restrições de capacidade de armazenamento dos nós névoas próximos aos usuários finais, o conteúdo armazenado em um único nó não é suficiente para atender a todas as requisições em horários de picos. O posicionamento desses serviços distribuídos ao longo da rede em nós Nuvem-Névoa é mais apropriado [75].

Um algoritmo para aprimoramento da entrega dos serviços de *streaming* multimídia para usuários com alta mobilidade é proposto por [39]. O ambiente é modelado em várias camadas com nós nuvem e névoa. O objetivo é reduzir a latência e minimizar o consumo da largura de banda. A primeira camada é composta por nós móveis, ou seja, os veículos que são capazes de processar e armazenar os serviços. A segunda camada é formada por alguns nós névoas fixos em alguns pontos específicos próximos à área dos veículos que requisitam os serviços de *streaming*. A solicitação tenta ser atendida por nós na primeira camada, seguida pelas demais na hierarquia. Se o serviço solicitado não estiver disponível na primeira camada, o serviço é solicitado na segunda camada. Se não estiver lá, será solicitada na Nuvem. O principal fator considerado pelos autores para aprimoramento é a QoS e a QoE do fluxo de multimídia. Os resultados obtidos mostram que os nós da camada secundária e primária localizadas próximo a borda da rede são considerados os mais adequados para a alocação dos serviços de *streaming* multimídia. No entanto, os autores consideram poucos veículos requisitantes e que a capacidade de hardware e os recursos de redes dos nós não variam com o tempo de simulação. Portanto, a arquitetura proposta pode não representar um modelo de rede do mundo real.

Uma arquitetura de rede ciente do volume de tráfego predito para o provisionamento e entrega de conteúdos multimídia a partir de *data centers* na Nuvem é proposta em [41]. Os

autores também apresentam dois algoritmos para a entrega desses conteúdos. Em linhas gerais, por meio de informações de monitoramento e de dados históricos sobre demandas por conteúdos multimídia realizados no passado, a arquitetura proposta utiliza recursos de predição para prever demandas futuras sobre tais conteúdos. As informações obtidas são utilizadas, em especial, para alocações de recursos de banda e para a seleção dos métodos de distribuição do conteúdo multimídia ao longo da rede. Os resultados mostram uma redução no congestionamento e uma taxa de 80% de sucesso da transmissão dos serviços oferecidos. O mecanismo de predição é preciso e, em geral, o processo de entrega de conteúdo obtém benefícios com a utilização do modelo de predição e dos algoritmos. No entanto, os serviços são oferecidos por nós na Nuvem, na qual, como já discutido, a Computação em Nuvem não oferece uma QoE satisfatória em áreas com alta demanda por esses serviços. Além disso, novos modelos de predição surgiram, como as redes LSTM que podem apresentar maior acurácia na predição de volume de tráfego da rede do que os modelos analisados pelo autor.

Um modelo de processo de decisão de *Markov* para posicionar dinamicamente os serviços de vídeo na Nuvem em vários *data centers* geograficamente distribuídos é proposto por [85]. O objetivo é maximizar os lucros médios para o provedor de serviços de vídeo a longo prazo e introduzir um critério de desempenho médio que reflete o custo e a experiência do usuário em conjunto. É proposto um algoritmo ideal com base na análise de sensibilidade e na iteração de política baseada em histórico para obter o posicionamento mais adequado de vídeo e estratégia de envio de solicitação. Além disso, é demostrado a otimalidade do algoritmo proposto com prova teórica e a viabilidade prática do algoritmo. Os resultados demostram que a estratégia pode efetivamente reduzir o custo total e garantir a qualidade da experiência dos usuários. No entanto, o posicionamento de serviços na Nuvem, mesmo que de forma ótima, não oferece uma qualidade de experiência adequada [63]. Além disso, a predição de dados utilizando o modelo de *Markov* torna-se exponencialmente custoso computacionalmente ao predizer instante de tempos maiores que $t + 1$, onde t é o instante de tempo atual. É visto ainda que, esses modelos de predição se baseiam em fortes suposições que nem sempre são verdadeiras (as transições de estado dependem apenas do estado atual, não de dados históricos). Isso diminui ainda mais a acurácia dos resultados [76].

O posicionamento de serviços e aplicações em nós névoas utilizando lógica fuzzy é proposto em [50]. O objetivo é melhorar a qualidade da experiência medida por vários parâmetros, como taxa de acesso aos serviços e sensibilidade ao atraso no processamento de dados. A lógica fuzzy é utilizada para priorizar diferentes requisições de posicionamento das aplicações e serviços, considerando os recursos computacionais do nós. Os resultados mostram melhorias nas condições da rede e na qualidade dos serviços oferecidos. No entanto, uma solução incoerente de posicionamento dessas aplicações e serviços para um grande volume de dados na arquitetura descentralizada na Computação em Névoa pode causar congestionamento na rede. A proposta de um algoritmo de posicionamento dessas aplicações e serviços em ambientes hierárquicos Nuvem-Névoa torna-se mais eficiente [54].

Um sistema para entrega de *streaming* para usuários móveis usando a Computação em Névoa é proposto por [66]. Os nós névoas são distribuídos na borda da rede, entre a Nuvem e os usuários móveis. Esses nós recebem as requisições dos usuários e respondem a cada

solicitação imediatamente, caso o nó disponha o serviço. Caso contrário, esses serviços são migrados da Nuvem para os nós névoas. Os resultados mostram que a Computação em Névoa fornece baixo tempo de resposta, ao contrário da Computação em Nuvem. Além disso, os autores analisam o consumo de energia dos usuários móveis nos quatro modos diferentes, modo de baixa energia, modo de transmissão, modo de recepção e CPU quando eles se conectam diretamente aos servidores de névoa. Os autores concluem que o sistema proposto oferece boa qualidade de experiência na entrega dos serviços e que o consumo de energia em todos os modos é minimizado. Por um lado, armazenamento prévio na cache desempenha um papel vital na Computação em Névoa, pois atende as requisições com baixa latência. Por outro lado, implementar mecanismos inteligentes que avaliam as condições da rede e dos nós para o posicionamento desses serviços de forma dinâmica torna-se mais adequado [54].

Nesta dissertação de mestrado é proposto um algoritmo de otimização modelado como PLI para o problema de posicionamento de serviços multimídia em ambientes hierárquicos Nuvem-Névoa, modelado como um **CFLP**. O objetivo é encontrar o menor conjunto de nós considerando suas capacidades de armazenamento para prover tais serviços de forma que a latência seja minimizada. Os resultados mostram que o algoritmo proposto posiciona os serviços multimídia em nós com capacidade de armazenamento adequada, próximos aos usuários e com latência média inferior a todas as estratégias. O posicionamento desses serviços torna-se ainda mais eficiente devido ao conjunto de nós reservados, através da predição do volume de tráfego da rede. Essa distribuição dos serviços ao longo da rede e próximo aos usuários permite o processamento e armazenamento perto da fonte de dados, sem a necessidade do envio de todos esses serviços para a Nuvem remota ou para outros sistemas centralizados.

A Tabela 3.1 lista os trabalhos relacionados e classifica suas contribuições com relação a cinco características. A primeira coluna representa os trabalhos relacionados. A segunda coluna descreve o domínio (Nuvem e/ou Névoa) dos trabalhos analisados. Os trabalhos com domínio em Nuvem posicionam os serviços considerando um conjunto de nós nuvens disponíveis. Diferentemente, os domínios Névoa ou Nuvem/Névoa alocam os serviços distribuídos ao longo da rede e próximos aos usuários. A terceira coluna apresenta o número de nós habilitados na avaliação para o posicionamento dos serviços multimídia. A quarta coluna refere-se aos trabalhos que consideram a capacidade de armazenamento dos nós. Por fim, a quinta acoluna refere-se aos trabalhos que consideram predições do volume de tráfego para posicionar os serviços. Células preenchidas com o símbolo "?" (interrogação) são valores não relatados pelos autores, $\times$ e $\bullet$ são contribuições não consideradas e consideradas pelos autores, respectivamente.

Tabela 3.1: Comparação dos trabalhos relacionados.

Trabalhos relacionados	Domínio	Número de nós	Capacidade de armazenamento dos nós	Múltiplas requisições instâtaneas	Predição do volume de tráfego
Jian He et al. [30]	Nuvem	30	•	×	•
Souza el at. [69]	Nuvem/Névoa	7	•	×	×
Liu et al. [44]	Névoa	5	•	×	×
Kharel et al. [39]	Nuvem/Névoa	84	•	×	×
Kryftis et al. [41]	Nuvem	?	×	×	•
Zhang et al. [85]	Nuvem	?	•	•	•
Mahmud et al. [50]	Névoa	4-10	•	•	×
Sheltami et al. [66]	Nuvem/Névoa	4	•	•	×
SMART-FL + predição	Nuvem/Névoa	1160	•	•	•

3.2 Ambientes hierárquicos Nuvem-Névoa

Ambientes hierárquicos Nuvem-Névoa também vem sendo propostos para otimizar diversos problemas relacionados a esse domínio, desde identificar o agrupamento mais adequado de estações base para compartilhar recursos da Nuvem ou até minimizar a distância entre os servidores e os pontos de acesso em toda a cidade [5, 16].

Uma solução para o problema de agrupamento de estações base para compartilhar recursos da Nuvem Rede de Acesso via Rádio (Cloud Radio Access Network - C-RAN) é proposta em [16]. A solução visa agrupar as estações rádio-base vizinhas com padrões de tráfego complementares, de forma que o volume de tráfego processado na C-RAN seja balanceada, exigindo menos recursos. Dessa forma, cada grupo ou partição encontrada constituem um nível no ambiente hierárquico. Os resultados mostram que este esquema de agrupamento reduz 12,88% de custo de implantação. O conjunto de dados utilizado foi disponibilizado pela Telecom Itália [5].

Um *framework* para particionar um conjunto de estações base em grupos e processar os dados em um data center compartilhado é proposto em [7]. Assim como descrito no trabalho anterior, cada partição encontrada constituem um nível no ambiente hierárquico. Essa estrutura de particionamento e agendamento economiza até 19% dos recursos de computação para uma probabilidade de falha de um em 100 milhões. No entanto, a adoção de somente um data center pode resultar em atrasos entre as estações base distantes e o data center.

Uma proposta de posicionamento de nós névoas para reduzir os custos associados à sua implantação e manutenção considerando demandas variáveis no tempo é proposto em [12]. O conjunto de nós selecionados formam o ambiente hierárquico. Os autores consideram a mobilidade do usuário, o que causa variações na demanda ao longo do

tempo em diferentes regiões. A solução é modelada como programação linear inteira com vários critérios. Os resultados, baseados em dados reais mostram que há uma melhoria no atendimento ao usuário final que pode ser obtida em conjunto com a minimização dos custos com a implantação de um número menor de servidores na infraestrutura. Além disso, os custos podem ser reduzidos ainda mais se um bloqueio limitado de solicitações for tolerado.

O posicionamento de servidores de bordas considerando restrições de capacidade é modelado como um problema de localização-alocação capacitadas por [42]. O objetivo é minimizar a distância entre os servidores e os pontos de acesso em toda a cidade. O desempenho do algoritmo é avaliado com diferentes parâmetros e em um conjunto de dados do mundo real em áreas centrais e suburbanas. Os resultados mostram que o algoritmo proposto é capaz de fornecer posicionamentos ideais que minimizam as distâncias e fornecem carga de trabalho equilibrada com compartilhamento de acordo com as restrições de capacidade dos nós.

Uma proposta de localização de nós névoas com suporte a usuários móveis com bateria limitada capazes de processar altas demandas com restrições de baixa latência é proposto por [21]. A abordagem favorece locais onde o descarregamento do volume de tráfego na Névoa reduz a energia consumida pelos dispositivos do usuário final. A solução é modelada como PLI, bem como uma solução heurística para resolver problemas de grande escala. A conclusão é que a solução heurística produz resultados precisos quando comparados aos dados pela PLI, permitindo assim uma economia significativa de energia para os usuários finais.

Neste trabalho, também é proposto um método para a criação de ambientes hierárquicos Nuvem-Névoa, apresentado na Seção 4.1. O método proposto utiliza uma abordagem *bottom-up*, iniciando-se a partir de um conjunto de estações base e organiza novos nós hierarquicamente em camadas, produzindo um ambiente hierárquico Nuvem-Névoa. O ambiente resultante do método proposto é simulado no **MultiTierFogSim**, no qual se beneficia de várias vantagens relacionadas a Nuvem e Névoa, como reconhecimento de localização, capacidade de análise para o processamento, bem como migrações de serviços de e para qualquer camada. Esse ambiente também é utilizado para a avaliação do algoritmo **SMART-FL** A avaliação desse ambiente real consta como trabalhos futuros devido ao escopo do projeto.

Capítulo 4

Posicionamento de Serviços Multimídia em Ambientes Hierárquicos Nuvem-Névoa

Este capítulo apresenta as principais contribuições desta dissertação de mestrado. A primeira contribuição é apresentada na Seção 4.1 e descreve o método proposto para a criação de ambientes hierárquicos Nuvem-Névoa. A segunda contribuição é apresentada na Seção 4.2 e descreve a solução proposta para o posicionamento de serviços multimídia em ambientes hierárquicos Nuvem-Névoa. A terceira contribuição é apresentada na Seção 4.3 que descreve dois modelos propostos, a saber ARIMA-PRED e LSTM-PRED, para a predição do volume de tráfego da rede celular da cidade de Milão. A Seção 4.4 descreve a modelagem do posicionamento dos serviços multimídia ciente do volume de tráfego predito. Finalmente, a quarta contribuição é apresentada na Seção 4.5 e descreve o simulador estendido para avaliar o trabalho proposto, assim como os parâmetros do ambiente de simulação.

4.1 Método para a criação de ambientes hierárquicos Nuvem-Névoa

Esta Seção apresenta o método para a criação de ambientes hierárquicos Nuvem-Névoa. O método proposto utiliza uma abordagem *bottom-up*, iniciando-se a partir de um conjunto $BS = \{bs_1, bs_2, ..., bs_{bs}\}$ de estações base. Inicialmente, é necessário definir o tipo de ligação Ξ entre as estações bases e a condição de parada definida pelo o número de subgrafos μ da penúltima camada. Para Ξ, pode-se considerar a distância r, força de sinal, similaridade do tráfego, dentre outros. O Método 1 e a Figura 4.1 descrevem todos os passos.

Método 1: Método para a criação de ambientes hierárquicos Nuvem-Névoa

Entrada: BS, Ξ, μ

Saída: $\mathcal{G}$

Condição de parada: Número de subgrafos μ.

Condição de parada alcançada: Adicionar um nó de camada superior que se conecta aos nós da camada inferior.

Passos:

1 - Definir um grafo não ponderado e não dirigido $\mathcal{G} = (BS, \mathcal{E})$.

2 - Detectar comunidades em $\mathcal{G}$.

3 - Para cada comunidade detectada, um nó de camada superior é adicionado em $\mathcal{G}$ que se comunica com todos os nós das estações base que pertence a comunidade. Essa etapa é finalizada com a remoção das arestas entre todas as estações base.

4 - Adicionar arestas entre todos os nós da camada atual.

5 - Detectar comunidades e remover arestas entre nós de comunidades distintas, formando subgrafos.

6 - Para cada subgrafo, adicionar um nó de camada superior com arestas entre o nó e o subgrafo.

7 - Volte ao passo 4.

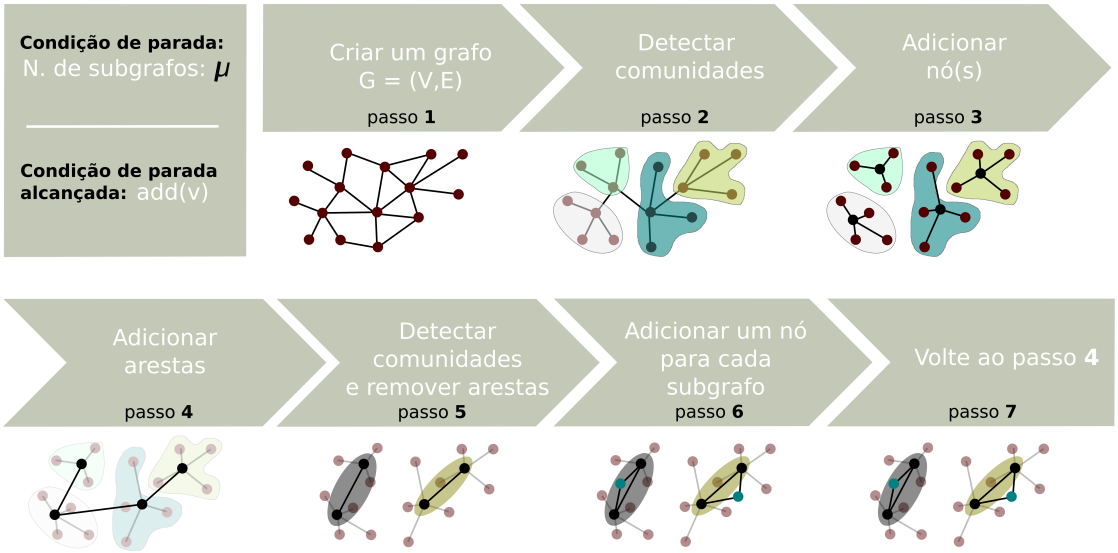


Figura 4.1: Método para a criação de ambientes hierárquicos Nuvem-Névoa.

A **Condição de parada** deve ser observada a cada passo. Considerando o **passo 1**, nós de outros provedores podem ser adicionados como um vértice em $\mathcal{G}$. Os novos nós organizados hierarquicamente podem ser considerados também como provedores de serviços e pode ser uma solução para o problema de planejamento de rede. A posição geográfica desses nós não se limitam a área da região estudada. Isto é, os novos nós hierarquicamente adicionados podem estar posicionados geograficamente em outra localidade.

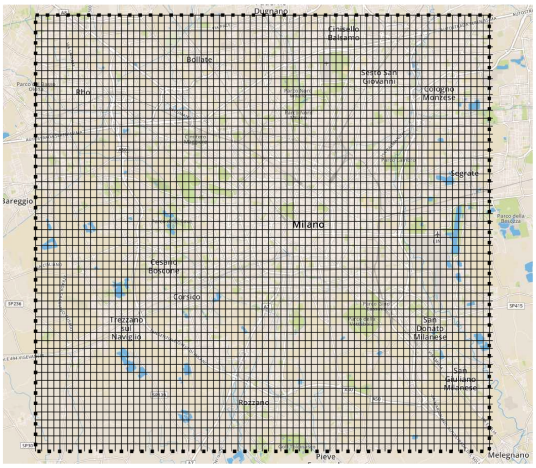
As comunidades detectadas nos **passos 2,5** são representadas pelo conjunto $S = \{s_1, s_2, ..., s_s\}$ e podem ser detectadas por qualquer estratégia [9, 78]. Comunidades, também chamadas de *clusters* ou agrupamentos, são grupos de vértices que tem grande probabilidade de compartilhar propriedades comuns ou tem papéis semelhantes no grafo. A avaliação desse ambiente consta como trabalhos futuros devido ao escopo do projeto.

4.1.1 Aplicação do método proposto

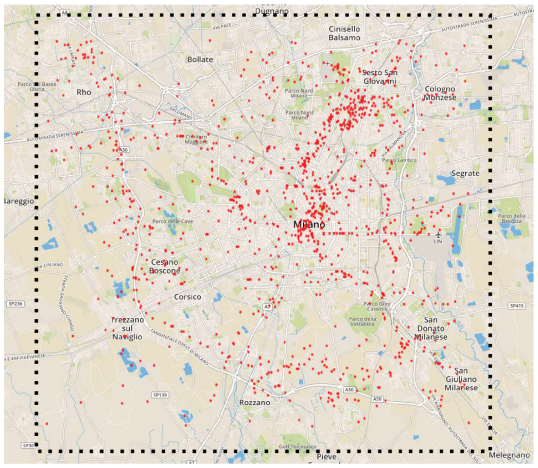
Esta Seção apresenta a aplicação do método proposto em um cenário real disponibilizado pela Telecom Itália [5]. Esse cenário é composto por dados de telecomunicação, temperatura, notícias, redes sociais e dados de eletricidade da cidade de Milão e da província de Trento. A composição única de múltiplas fontes de dados o torna um conjunto de dados ideal para analisar vários problemas, incluindo planejamento do consumo de energia em grandes centros, análise da mobilidade urbana, análise do volume de tráfego das redes celulares, dentre outros. Como os dados foram coletados por várias empresas que adotaram padrões diferentes para agregar as informações, a irregularidade na distribuição espacial é agregada em malhas, de acordo com o padrão WGS84 (EPSG:4326). Isso permite comparações entre diferentes áreas e facilita o gerenciamento geográfico dos dados. Assim, a área de Milão e Trento são compostas por uma malha com 1.000×1.000 e 6.575×6.575 células de $235m^2$ cada, respectivamente.

O cenário utilizado neste trabalho é baseado nos dados reais de telecomunicação observado durante dois meses de Novembro/2013 até Dezembro/2013 na região de Milão. Para cada célula, é armazenado a cada 10 minutos a quantidade de Registro de Detalhe

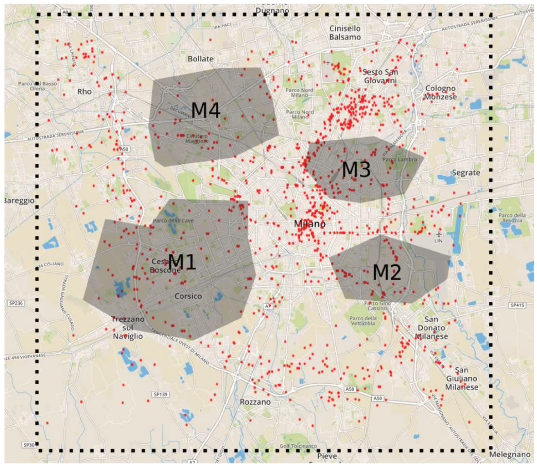
da Chamada (Call Detail Record - CDR) que é gerado sempre que um usuário móvel realiza uma interação de telecomunicações com uma estação base de rádio. Considera-se que existe uma requisição por serviços multimídia em uma célula g no tempo t se a quantidade de CDRs em g for maior do que a média total de CDRs do tempo t [18, 81]. O posicionamento realístico das estações base foi gerado através da ferramenta CellMapper[1], que consiste nas localizações e áreas de cobertura das estações base ativas observadas durante o período analisado. As células e a quantidade de CDRs foram mapeadas para as áreas de cobertura das estações base, considerando um raio de 3km. A Figura 4.2 apresenta três imagens do cenário utilizado. A Figura 4.2(a) ilustra a região de Milão composta por uma malha com 1.000 × 1.000 células. A Figura 4.2(b) ilustra o posicionamento real das estações base (em vermelho). A Figura 4.2(c) ilustra um cenário em 10 de Novembro de 2013, no qual as regiões M1, M2, M3 e M4 representam áreas demanda por serviços multimídia.



(a) Área de Milão.



(b) Posicionamento real das estações base.



(c) Áreas com requisições: M1, M2, M3 e M4.

Figura 4.2: Visualização do cenário considerado.

Nesse ambiente, são considerados requisições por serviços pertencentes a duas classes:

[1]https://www.cellmapper.net/map

multimídia e concorrente. Por um lado, os serviços que pertencem a classe multimídia são os serviços de vídeo sob demanda, videoconferência, transmissão de ambientes 3D interativos, vídeos com alta definição, *streaming* de vídeo com resolução 4k/8k, dentre outros. Por outro lado, os serviços pertencentes a classe concorrente são os oferecidos em TCS ou IoT. Considera-se que a quantidade de requisições por esses serviços em uma célula g no tempo t é o número de CDRs em g. Dessa forma, os recursos providos pela rede e por nós são disputados por serviços de ambas as classes.

Os requisitos dos serviços de ambas as classes são modelados em termos de três parâmetros, a saber (**i**) latência máxima de atendimento, (**ii**) quantidade máxima de memória RAM para o armazenamento temporário e (**iii**) quantidade máxima de MIPS. Inicialmente, seja $V_t = \{v_1, v_2, v_3, ..., v_g\}$ o conjunto que representa o volume de tráfego da rede no tempo t gerado por cada célula g. Assim, o armazenamento máximo necessário para executar qualquer serviço, isto é, da classe multimídia ou concorrente, requisitado no tempo t solicitado na célula g é v_g^t, onde $v_g^t \in V_t$. Dessa forma, o armazenamento necessário para executar quaisquer serviço varia de acordo com o volume de tráfego da rede, exibindo periodicidade.

Seja $J = \{j_1, j_2, j_3, ...j_r\}$ o conjunto que representa os serviços da classe concorrente. Os conjuntos $Lat^J = \{lat_1^j, lat_2^j, lat_3^j, ..., lat_r^j\}$ e $Mip^J = \{mip_1^j, mip_2^j, mip_3^j, ..., mip_r^j\}$ representam os valores de latência e MIPS para cada serviço concorrente $j \in J$, baseados em [70]. Portanto, para cada $j \in J$

$$j_i = \{lat^j, mip^j, v_g^t\} \tag{4.1}$$

onde

$$0 \leq i \leq r \tag{4.2}$$
$$lat^j \in Lat^J \tag{4.3}$$
$$mip^j \in Mip^J \tag{4.4}$$
$$v_g^j \in V_t \tag{4.5}$$

onde, lat^j representa a latência máxima para atender o serviço concorrente j, v_g^t é a quantidade máxima de memória RAM para armazenar o serviço concorrente j e mip^j a quantidade máxima de MIPS para processar o serviço concorrente j.

Seja $W = \{w_1, w_2, w_3, ...w_z\}$ o conjunto que representa os serviços da classe multimídia. Os conjuntos $Lat^W = \{lat_1^w, lat_2^w, lat_3^w, ..., lat_z^w\}$ e $Mip^W = \{mip_1^w, mip_2^w, mip_3^w, ..., mip_z^w\}$ representam os valores de latência e MIPS para cada serviço multimídia $w \in W$, baseados em [29, 57, 67]. Portanto, para cada $w \in W$

$$w_i = \{lat^w, mip^w, v_g^t\} \tag{4.6}$$

onde

$$0 \leq i \leq n \tag{4.7}$$
$$lat^w \in Lat^W \tag{4.8}$$
$$mip^w \in Mip^W \tag{4.9}$$
$$v_g^t \in V_t \tag{4.10}$$

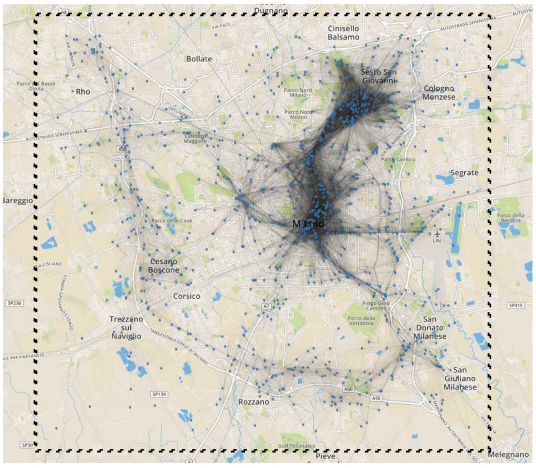
onde, lat^w representa a latência máxima para atender o serviço multimídia w, v_g^t é a quantidade máxima de memória RAM para armazenar o serviço multimídia w e mip^w a quantidade máxima de MIPS para processar o serviço multimídia w.

A aplicação do método proposto inicia-se com a definição do conjunto BS e os parâmetros Ξ e μ. BS representa as estações bases do cenário descrito anteriormente; Ξ é definido como o raio $r = 3km$; e μ igual a dois. A Figura 4.3 ilustra o desenvolvimento do cenário hierárquico resultante de cada passo executado. A Figura 4.3(a) ilustra o grafo $\mathcal{G} = (BS, \mathcal{E})$ modelado a partir do passo (**1**). Como Ξ é definido como o raio $r = 3km$, isso significa que, existe uma aresta entre bs_i e $bs_j \in BS$, se e somente se, a distância entre bs_i e bs_j é de $3km$. Vale a pena mencionar que, nesse passo, outras formas de ligação é possível, como a força do sinal ou similaridade do tráfego.

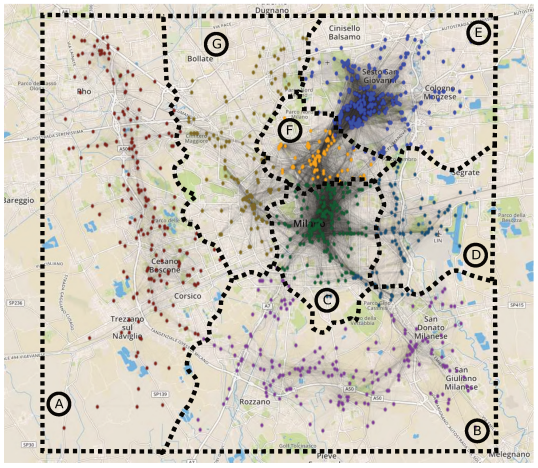
A Figura 4.3(b) ilustra as comunidades detectadas em $\mathcal{G}$, considerando as estações base (pontos coloridos) a partir do passo (**2**). Nesse cenário, foram detectadas setes comunidades representadas pelas letras de A a G. As comunidades detectadas compõe áreas urbanas (G, F, D e C) e suburbanas (A, B e E). Isso pode indicar que as estações base pertencentes a mesma comunidade são potencialmente complementares devido aos padrões de tráfego da rede, que pode estar associado à distância entre as estações base. Para encontrar o conjunto de comunidades, utilizou-se a heurística de Louvain de complexidade computacional $O(|BS| + |\mathcal{E}| \cdot log|BS| + |\mathcal{E}|)$, onde $|BS|$ e $|\mathcal{E}|$ são os números de vértices e arestas, respectivamente. Esse método tem como objetivo encontrar partições (estruturas compostas de comunidades) que maximizem a densidade de conexões intragrupo quanto à densidade de conexões intergrupos e, assim, encontrar subgrafos ótimos em grafos densos [9].

O ambiente derivado dos passos (**3,4**) adição de um nós para cada comunidade detectada e arestas entre elas, passo (**5**) detecção das comunidades, remoção das arestas entre nós de comunidades distintas e detecção dos subgrafos é ilustrado na Figura 4.3(c). O passo (**6**) consiste da adição de um nó para cada subgrafo encontrado e a ligação entre o nó e o subgrafo, ilustrado na Figura 4.3(d). A condição de parada é monitorada a cada passo é atingida quando μ é dois. Nesse caso, a condição de parada é alcançada, finalizando com a adição de um nó de camada superior que se conecta aos dois nós pertencentes a camada inferior, conforme ilustrado na Figura 4.3(e). A Figura 4.3(f) ilustra o cenário final do ambiente hierárquico Nuvem-Névoa. Neste trabalho, os nós de cada camada a partir do passo **(2)** são rotulados de *cloudlet* (**CL1**, **CL2**, **CL3**, **CL4**, **CL5**, **CL6** e **CL7**), nuvem regional (**RC1** e **RC2**) e nuvem (**CL**). A Tabela 4.1 mostra o número de nós e a área de cobertura média por camada.

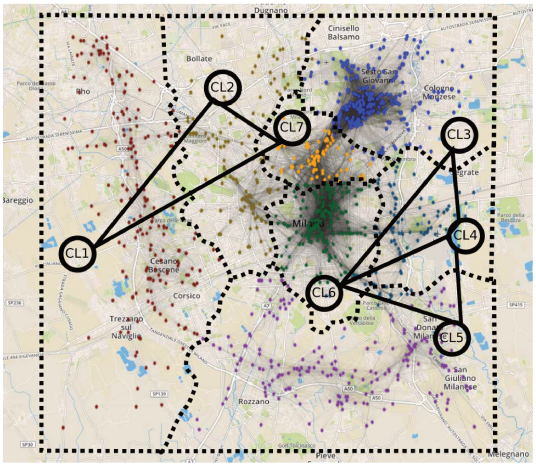
Os nós capazes de armazenar e processar os serviços são modelados em termos de seis parâmetros, a saber (**i**) *mips_disp* quantidade de MIPs disponíveis; (**ii**) *stor_disp* quan-



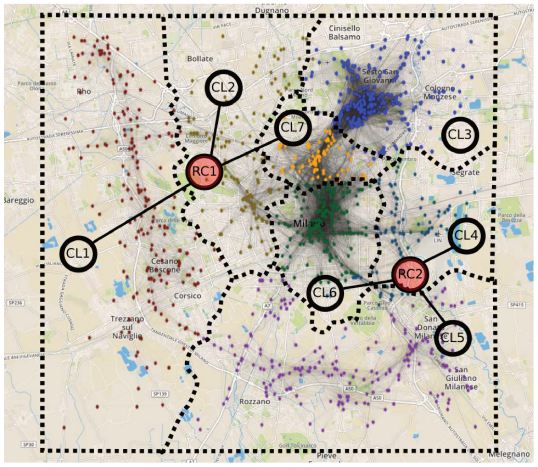
(a) Cenário modelado como um grafo G. Estações base em azul.



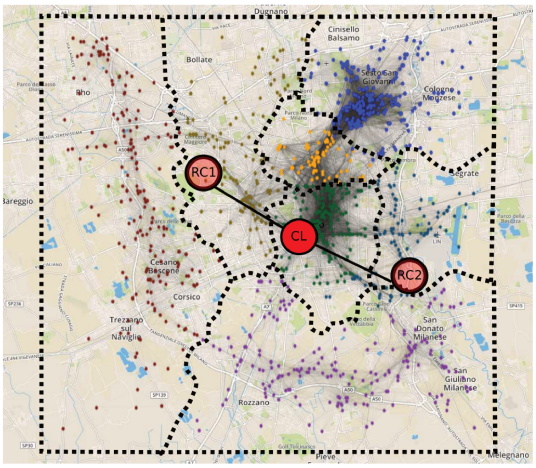
(b) Detecção das comunidades utilizando o algoritmo de agrupamento de *Louvain*.



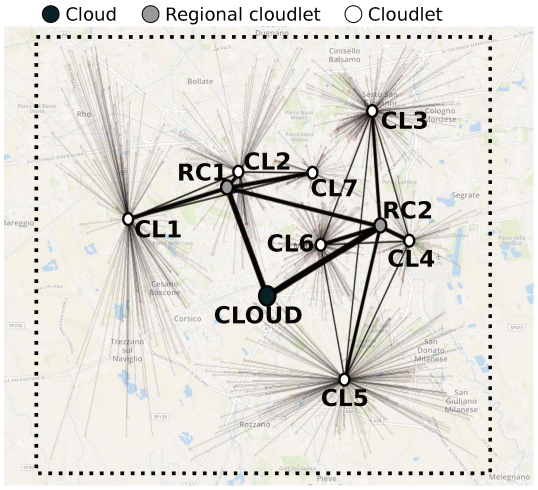
(c) Cenário com sete nós cloudlet adicionados.



(d) Cenário com dois nós de nuvens regionais adicionados.



(e) Cenário com um nó cloud adicionado.



(f) Visualização do mapa da topologia do cenário final.

Figura 4.3: Visualização do cenário resultante do método proposto.

Tabela 4.1: Número de nós e área de cobertura por camada.

Nós	**Número de nós**	**Área de cobertura (m^2 por nó)**
Estação base	1150	492.46
Cloudlet	7	9072.67
Nuvem regional	2	31754.37
Nuvem	1	552250

tidade de armazenamento disponível em GB; (**iii**) ram_disp memória RAM disponível em GB; (**iv**) e_prod o consumo de energia produzido; e (**v**) up_rate e (**vi**) $down\_rate$ são taxas de *uplink* e *downlink* oferecido, respectivamente, ambas em Mb. Assim, seja $\mathcal{L} = \{ell_1, ell_2, ell_3, ...ell_x\}$ o conjunto de nós capazes de processar e armazenar os serviços multimídia. Portanto, para cada $ell \in \mathcal{L}$:

$$\ell = \{mips\_disp,\ stor\_disp,\ ram\_disp,\ e\_prod,\ up\_rate,\ down\_rate\} \tag{4.11}$$

Vale a pena mencionar que todos os parâmetros variam dinamicamente entre um intervalo pré-definido. A justificava é que o cenário seja mais realístico possível e que os serviços sejam alocados em diferentes nós, mesmo quando houver áreas com demandas semelhantes. Além disso, nós pertencentes a camadas superiores possuem maior capacidade de *hardware* e rede. Em contrapartida, as camadas inferiores oferecem suporte a serviços e aplicações sensíveis à latência, não totalmente contemplados pela Computação em Nuvem. Os valores são baseados em [60] e apresentados na Tabela 4.2. A Tabela 4.3 resume as notações utilizadas nesta seção. Esse ambiente foi implementado e analisado utilizando o simulador **MultitierFogSim**, descrito na Seção 4.5.

Tabela 4.2: Faixa de valores utilizados para modelar os nós por camadas.

Parâmetros	**Valores (por camada)**			
	1°	2°	3°	4°
$mips\_disp$	[2.8 - 5.3]	[5.3-7.8]	[7.8-10.2]	[10.2-20.5]
$stor\_disp$	100^2	200^2	400^2	1000^2
ram_disp	25	40	60	100
e_prod	100	300	500	1000
up_rate & $down\_rate$	300	500	800	2000

Tabela 4.3: Resumo das notações utilizadas.

Notação	**Descrição**
W	conjunto de serviços da classe multimídia.
J	conjunto de serviços da classe concorrente.
Lat^W	conjunto que representa a latência dos serviços multimídia.
Lat^J	conjunto que representa a latência dos serviços concorrentes.
Mip^W	conjunto que representa a quantidade máxima de MIPs para processar os serviços multimídia.
Mip^J	conjunto que representa a quantidade máxima de MIPs para processar os serviços concorrentes.
V_t	conjunto do volume de tráfego da rede no tempo t gerado por cada célula g.
$[A, ..., G]$	conjunto que representa as setes comunidades detectadas em $\mathcal{G}$.
[CL1, CL2, CL3, CL4, CL5, CL6, CL7, RC1, RC2, CL]	conjunto dos nós considerados em $\mathcal{G}$ a partir do passo **(2)**.
BS	conjunto de estações base.
S	conjunto de comunidades detectadas.
$\mathcal{G}$	grafo considerado.
$\mathcal{E}$	conjunto de arestas entre as estações base.
$\|BS\|$	quantidade de vértices.
$\|\mathcal{E}\|$	quantidade de arestas.
Ξ	tipo de ligação entre as estações base.
μ	número de subgrafos.

4.2 Algoritmo SMART-FL

A solução proposta para o posicionamento de serviços multimídia em ambientes hierárquicos Nuvem-Névoa é modelada como um **CFLP**, na qual (**i**) os nós são potenciais locais das instalações dos serviços oferecidos, (**ii**) as requisições dos serviços multimídia são as demandas, (**iii**) a capacidade de armazenamento dos nós e a demanda dos usuários fazem parte do conjunto de restrições, e (**iv**) os serviços multimídia correspondem ao tipo de serviço considerado.

Seja $G_t = \{g_1^{w_1} g_2^{w_2}, ..., g_k^{w_z}\}$ o conjunto de serviços multimídia $w \in W$ requisitado no tempo t pertencente a célula g; $\mathcal{L}$ o conjunto de nós capazes de processar e armazenar os serviços. Isso inclui os nós nuvem, nuvens regionais e *cloudlets*, cada um com capacidade $c_{max(\ell)}$, onde $\ell \in \mathcal{L}$; CLO, RC, CL e BS os conjuntos que representam os nós nuvens, nuvens regional, *cloudlets* e estações base, respectivamente; $lat(\ell, g^w, t)$ a latência do nó ℓ

para atender o serviço w na célula g no tempo t, onde $g^w \in G_t$. A variável $x(\ell, g^w, t) \geq 0$ representa o serviço g^w a que o nó ℓ processa iniciado no tempo t, onde $g^w \subseteq G_t$. Assim, um algoritmo de otimização modelado como PLI para o problema de posicionamento de serviços multimídia pode ser especificado da seguinte maneira:

Minimiza

$$\sum_{\ell \in \mathcal{L}} y(\ell, w, t) + \sum_{\ell \in \mathcal{L}} \sum_{g^w \in G_t} lat(\ell, g^w, t) \cdot x(\ell, g^w, t) \tag{4.12}$$

Sujeito a

$$\sum_{\ell \in \mathcal{L}} x(\ell, g^w, t) = g^w \qquad \forall g^w \in G_t \tag{4.13}$$

$$\sum_{g^w \in G_t} x(\ell, g^w, t) \leq c_{max}(\ell, t) \cdot y(\ell, w, t) \qquad \forall \ell \in \mathcal{L} \tag{4.14}$$

$$x(\ell, g^w, t) \geq 0 \qquad \forall \ell \in \mathcal{L} \; and \; \forall g^w \in G_t \tag{4.15}$$

$$y(\ell, w, t) \in \{0, 1\} \qquad \forall \ell \in \mathcal{L} \tag{4.16}$$

$$w \in W \tag{4.17}$$

$$t \in [0, max\_simulation\_time] \tag{4.18}$$

A variável binária $y(\ell, w, t) = 1$ indica se o serviço multimídia w está instalado em ℓ no tempo t; $y(\ell, w, t) = 0$, caso contrário. A função objetivo 4.12 é composta por duas partes. A primeira, seleciona os nós que minimizam o custos associados. A segunda, associa os custos da latência do nó ℓ para atender o serviço w na célula g no tempo t e a demanda d_g que o nó ℓ atende iniciada no tempo t. A restrição na Equação 4.13 exige que o serviço w requisitado pelo usuário pertencente a célula g seja atendido. A capacidade de cada nó $\ell \in \mathcal{L}$ é limitada pela restrição na Equação 4.14. Isto é, se o nó ℓ é selecionado, os serviços processados e armazenados por ele não podem ultrapassar sua capacidade máxima de armazenamento. Finalmente, as restrições das Equações 4.15-4.18 definem os valores mínimos para as variáveis de decisão. A Tabela 4.4 resume as notações utilizadas.

Tabela 4.4: Resumo das notações utilizadas.

Parâmetros de entrada	
Notação	**Descrição**
CLO	Conjunto de nós nuvens.
RC	Conjunto de nós nuvens regionais.
CL	Conjunto de nós *cloudlets*.
BS	Conjunto de estações base.
$\mathcal{L}$	Conjunto de nós no qual os serviços podem ser alocados.
G_t	conjunto de serviço multimídia $w \in W$ requisitado no tempo t pertencente a célula g.
$c_{max}(\ell, t)$	Capacidade de armazenamento do nó ℓ no tempo t, onde $\ell \in \mathcal{L}$.
$lat(\ell, g^w, t)$	Latência do nó ℓ para atender o serviço w na célula g no tempo t.
Variáveis de decisão	
$y(\ell, w, t)$	**1** indica se o serviço multimídia w está instalado em ℓ no tempo t; **0** caso contrário.
$x(\ell, g^w, t)$	Demanda d_g que o nó ℓ atende iniciada no tempo t, onde $d_g \subseteq G_t$.

Esse modelo de Programação Linear foi codificado utilizando o *Gurobi Optimizer* [53]. *Gurobi* é um *solver* de programação matemática comercial, mas gratuita na versão estudante. É possível implementar o paralelismo de memória compartilhada, capaz de explorar simultaneamente qualquer número de processadores e núcleos por processador. *Gurobi* utiliza processos iterativos para convergir para uma solução ideal.

Esse modelo de PLI é baseado, logicamente, em equações lineares. Isto significa que, se a capacidade de armazenamento do nó ell_x aumenta em 100% enquanto tudo permanece constante, é possível atender o dobro de serviços em ell_x. Entretanto, algumas variáveis de decisão tem um efeito não linear. Por exemplo, a QoE. Se a latência oferecida por ell_x é 50% menor em relação aos outros nós, isso não significa que a QoE aumentará em 50% se o serviço for oferecido por ell_x. A QoE não se relaciona a efeitos lineares. Uma alternativas à esse modelo é a Programação por Metas, que considera variáveis não lineares.

4.3 Desenvolvimento dos modelos de predição

Esta seção discorre sobre o desenvolvimento dos modelos ARIMA-PRED e LSTM-PRED que tem como objetivo predizer o volume de tráfego da rede para que os serviços multimídia sejam alocados em um conjunto de nós reservados previamente. A escolha desses métodos deve-se ao fato da ampla utilização para previsões em tempo real, tais como para previsões de fluxos de tráfego de rede, sendo utilizados em diversos trabalhos [52].

Na prática, o desenvolvimento de cada modelo regressivo difere (**i**) na modelagem da série temporal para um conjunto apropriado ao modelo e (**ii**) a escolha dos parâmetros

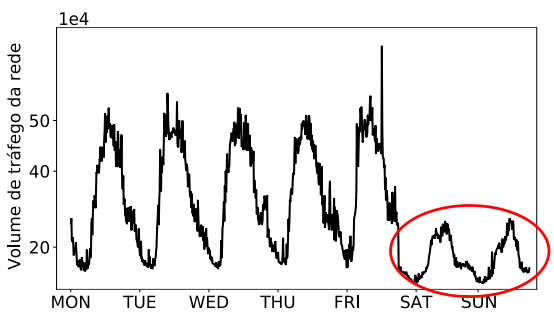
mais adequados para cada tipo de abordagem. A modelagem da série temporal para ambos os modelos é discutida nesta seção. Já os parâmetros utilizados nos modelos ARIMA-PRED e LSTM-PRED são separadamente discutidos nas Seções 4.3.1 e 4.3.2, respectivamente.

O conjunto de dados utilizado é o volume de tráfego gerado em cada célula do cenário descrito no começo deste capítulo. A Tabela 4.5 mostra os cinco primeiros valores do volume de tráfego da rede na célula #1. A primeira coluna (*timestamp*) representa uma variável temporal, iniciando em 2013-11-01 00:10:00 e finalizando em 2014-01-01 23:40:00. A segunda coluna (*traffic volume*) representa a quantidade de CDR gerados. Diante da organização temporal, esses dados podem ser modelados e avaliados como uma série temporal, onde as observações vizinhas são dependentes e o interesse é analisar e modelar essa dependência.

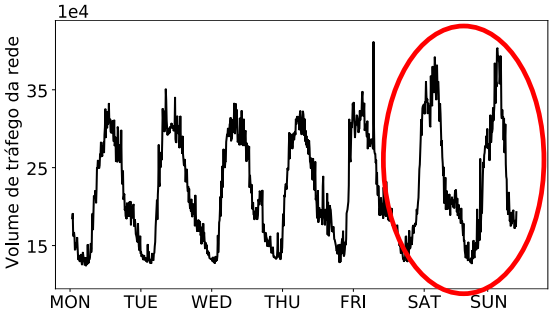
Tabela 4.5: Estrutura do conjunto de dados.

	timestamp	**traffic volume**
0	2013-11-01 00:00:00	11.028366381681
1	2013-11-01 00:10:00	11.1271008756737
2	2013-11-01 00:20:00	10.8927706027911
3	2013-11-01 00:30:00	8.62242459098975
4	2013-11-01 00:40:00	8.00992746244576
...	...	...

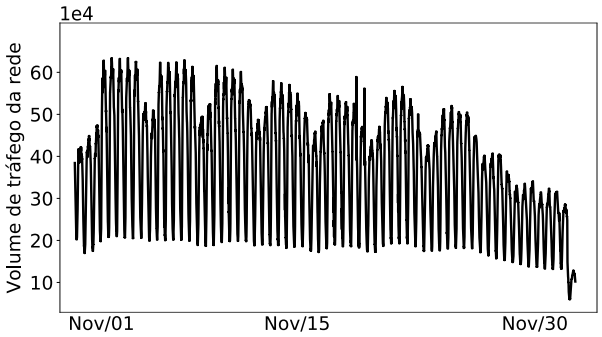
A Figura 4.4 ilustra o volume de tráfego da rede modelado como uma série temporal. As Figuras 4.4(a)-4.4(b) ilustram o volume médio do tráfego da rede durante uma semana baseado na área de cobertura de todas as estações base posicionadas nas regiões centrais e periféricas do cenário, respectivamente. É possível notar claramente que o volume de tráfego exibe certa periodicidade (nos padrões diários e semanais) como resultado de horários de trabalho regulares (elipse em vermelho). Por um lado, as regiões centrais apresentam volume de tráfego menor nos finais de semana do que durante a semana, devido a mobilidade urbana e horários de trabalho regulares. Por outro lado, as regiões periféricas apresentam volume de tráfego regular durante os sete dias da semana. A Figura 4.4(c) mostra um trecho do volume de tráfego durante o mês de Novembro da célula #3.500.



(a) Volume do tráfego durante uma semana (regiões centrais).



(b) Volume do tráfego durante uma semana (regiões periféricas).



(c) Volume do tráfego durante um mês.

Figura 4.4: Volume do tráfego.

Com base nessas recorrências, diversos métodos e modelos de predição podem ser empregados como modelos base, tais como Filtro de Kalman, Redes Neurais Recorrentes, ARIMA ou até mesmo lógica Fuzzy. Dentre tais métodos e modelos, LSTM e ARIMA são considerados neste trabalho como modelos bases para a predição do volume de tráfego da rede. A escolha desses métodos como modelos bases para predição deve-se ao fato da ampla utilização para previsões em tempo real, tais como para previsões de fluxos de tráfego de rede, sendo utilizados em diversos trabalhos [52, 82, 83]. Ao final do processo avaliativo discutido nas próximas seções, somente um modelo é empregado. Para ambos os modelos, os dados não precisam ser divididos, apenas organizados no formato de um *array* unidimensional.

O processo de criação para ambos os modelos segue a metodologia de *Montgomery* [51]. Resumidamente, os passos são enumerados a seguir.

1°. Determinar o escopo da predição. Nesse caso, o volume de tráfego.

2°. Obter o conjunto de dados que inclua informações temporais.

3°. Gerar gráficos que representem as séries temporais para inspeção visual a fim de interpretar padrões reconhecíveis. Assim com analisar a correlação entre os dados e remover possíveis *outliers*.

4°. Definir o(s) modelo(s) de predição e ajustar os parâmetros. Essa etapa também relaciona-se com a terceira.

5°. Validar com dados que não fazem parte do treinamento.

6°. Disponibilizar o modelo com fácil acesso e interpretação.

7°. Monitorar e adaptar o modelo às mudanças dos dados em produção, para garantir o bom desempenho do sistema.

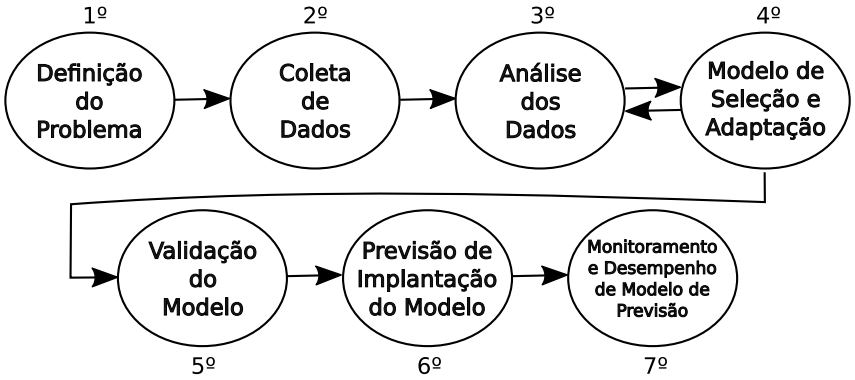


Figura 4.5: Processo de predição (adaptado de [51])

Os dados foram separados em conjunto de treinamento e teste. Para ambos os modelos, o conjunto de treinamento é formado considerando os primeiros 40 dias *dataset* (entre 1 de novembro de 2013 e 10 de dezembro de 2013). O conjunto de teste é coletado nos últimos 22 dias (entre 21 de dezembro de 2013 e 1 de janeiro de 2014). Antes do treinamento, todos os dados são normalizados de forma que todos os valores tenham média zero e variância unitária, ilustrado na Figura 4.6.

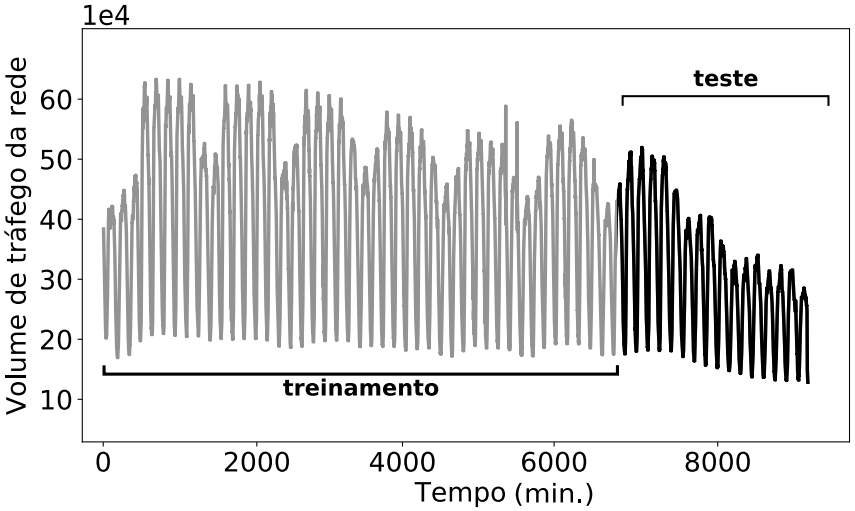


Figura 4.6: Separação dos dados em conjunto de treino e teste.

Para comparar os resultados entre os modelos, são utilizadas métricas de regressões para quantificar a acurácia das previsões. Essas métricas consistem em avaliar a extensão do erro da predição, ou seja, mensurar o distanciamento dos valores preditos pelo modelo em relação aos dados reais observados. Dentre as métricas utilizadas neste trabalho

estão o MAE e RMSE, os quais são amplamente utilizados nos trabalhos encontrados na literatura, principalmente em predição de tráfego móvel [33, 80, 81]. O MAE calcula a média da diferença entre duas variáveis contínuas, isto é, a média da diferença entre o valor real e o predito. O RMSE calcula a média quadrática dessas diferença, isto é, cada erro na RMSE é proporcional ao tamanho do erro quadrado. Ambas as métricas variam de 0 a ∞ e são indiferentes à direção dos erros. Isto é, valores próximos de zero são melhores (quase nunca alcançado na prática). Obter a raiz quadrada da média dos erros tem algumas implicações interessantes. Como os erros são elevados ao quadrado antes de serem calculados, o RMSE atribui um peso relativamente alto aos erros grandes. Dessa forma, grandes diferenças entre os valores preditos e reais tem maiores efeitos sobre RMSE do que no MAE.

Ambos os modelos foram desenvolvidos na linguagem de programação Python 3[2], por meio do uso, principalmente, das bibliotecas *sklearn* e *statsmodels*, assim como as métricas utilizadas para avaliar os modelos.

As seções seguintes descrevem os parâmetros, algoritmo e modelagem do conjunto de treinamento para ambos os modelos. A Seção 4.3.1 descreve a implementação do modelo ARIMA-PRED. A Seção 4.3.2 descreve a implementação do modelo LSTM-PRED. Finalmente, a Seção 4.3.3 discute sobre a avaliação de ambos os modelos.

4.3.1 Implementação do modelo ARIMA-PRED

Esta seção apresenta o método de predição ARIMA-PRED, baseada no modelo ARIMA. Nesse modelo, o valor futuro de uma variável é assumido como uma função linear de várias observações anteriores e erros aleatórios. O modelo pode ser escrito da seguinte forma:

$$y_t = \alpha + \phi_1 y_{t-1} + \phi_2 y_{t-2} + ... + \phi_p y_{t-p} + \varepsilon t - \theta_1 \varepsilon_{t-1} - \theta_2 \varepsilon_{t-2} - ... - \theta_q \varepsilon_{t-q}, \qquad (4.19)$$

onde α é uma constante, ϕ é um coeficiente estimado e y trata de entradas passadas (*lags*). *theta* é um coeficiente estimado e ε trata-se dos erros associados as predições regressivas passadas.

Modelo ARIMA-PRED em palavras:

> y_t(predito) = constante + defasagens linear de y (até p defasagens) + combinação linear de erros de predição defasados (até q defasagens).

A Figura 4.7 resume as etapas da implementação do modelo. A primeira etapa está relacionada com a modelagem da série temporal, considerando o conjunto de dados descrito nas seções anteriores. A segunda e terceira etapa são referentes à modelagem do conjunto de treinamento, teste e definição dos parâmetros (p, q e d). O modelo ARIMA-PRED gera previsões através de informações contidas na própria série cronológica, através dos ajustes ideias desses parâmetros e determinar-lós é uma tarefa não trivial [11]. No entanto, a função $auto\_arima()$ em Python encontra automaticamente esses valores ideais.

[2]https://www.python.org/

Em geral, $p + q <= 2$ [84]. A Tabela 4.6 mostra as quatro melhores combinações desses parâmetros geradas em relação as métricas MAE e RMSE.

Tabela 4.6: Combinações dos parâmetros (p,q,d) em relação as métricas MAE e RMSE.

(p,q,d)	**MAE**	**RMSE**
(1,1,1)	1.092	3.645
(1,1,0)	1.108	3.985
(0,1,0)	1.175	4.732
(1,0,0)	1.190	4.355

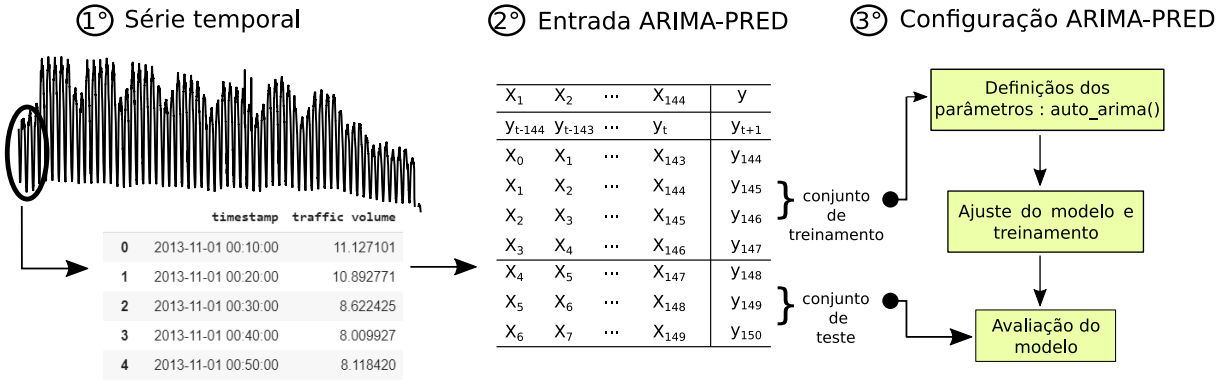
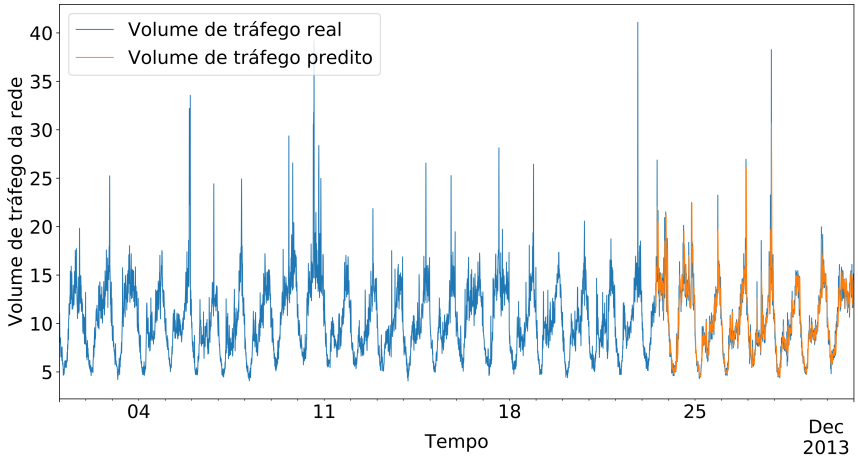
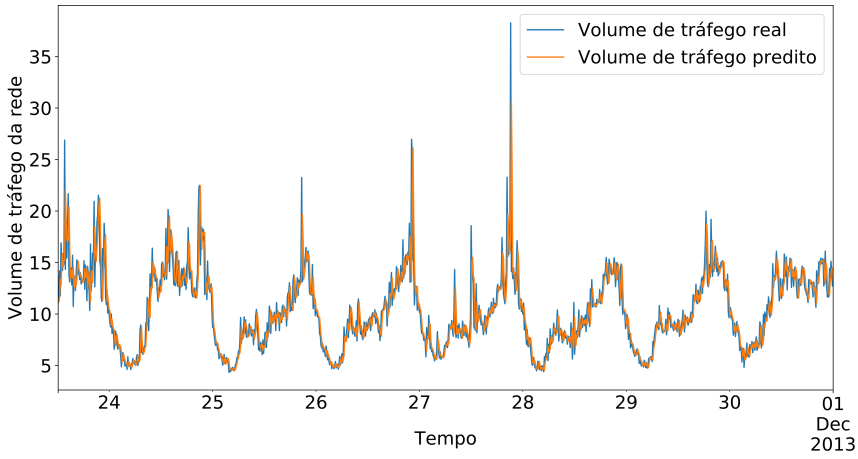


Figura 4.7: Processo de implementação do modelo ARIMA-PRED.

Portanto, o modelo é configurado para a fase de testes com os parâmetros $(1, 1, 1)$ para (p, q, d), respectivamente, devido apresentar valores ideais em relação as métricas MAE e RMSE na fase de treinamento. Para o auxílio da análise gráfica, a Figura 4.8 mostra o comparativo entre o volume de tráfego real e predito utilizando o modelo ARIMA-PRED. A Figura 4.8(a) ilustra o volume de tráfego real (em azul) e predito (em laranja). A Figura 4.8(b) ilustra o conjunto de teste (em azul) e predito (em laranja).



(a) Conjunto de treinamento (azul). Valores preditos (laranja).



(b) Comparação entre o volume de tráfego real e predito.

Figura 4.8: Visualização da predição utilizando o modelo ARIMA-PRED.

A complexidade de tempo associada a fase de treinamento e predição em tempo real é $O(n)$ e $O(1)$, respectivamente.

4.3.2 Implementação do modelo LSTM-PRED

Esta seção apresenta o método de predição LSTM-PRED, baseada no modelo LSTM. As etapas da implementação do modelo LSTM-PRED são ilustradas na Figura 4.9.

A primeira etapa está relacionada com a modelagem da série temporal, baseando-se no conjunto de dados descrito nas seções anteriores. A segunda etapa refere-se à modelagem do conjunto de treinamento e teste. A geração desses conjuntos baseia-se em uma técnica chamada janela deslizante, comumente adotada em [4]. Nessa técnica, o conjunto de treinamento expande-se à medida que o modelo é treinado. Isto é, utiliza-se o valor do tempo t, bem como valores de tempos anteriores $[t-1, t-2, ..., t-L)$ como variáveis de entrada para a predição do valor no tempo $(t+1)$. Neste trabalho, o tamanho da janela deslizante é $L = 144$ incluindo timestamp atual. Esse valor é baseado na periodicidade diária observada na série.

Finalmente, a terceira etapa está relacionada com a definição dos hiperparâmetros,

a fase de treinamento, finalizando com a avaliação do modelo. Para encontrar os hiperparâmetros mais adequados para o modelo LSTM, utilizou-se uma técnica de pesquisa exaustiva *GridSearchCV*, disponibilizada pela biblioteca *sklearn*. O objetivo dessa técnica é testar todas as combinações possíveis dos hiperparâmetros que lhes foram passados para encontrar a configuração mais adequada para o modelo em questão. A Tabela 4.7 mostra os valores dos hiperparâmetros estimados e que otimizam o modelo. A segunda coluna refere-se aos valores estimados baseados em [59,80]. A terceira coluna refere-se aos valores que otimizam o modelo.

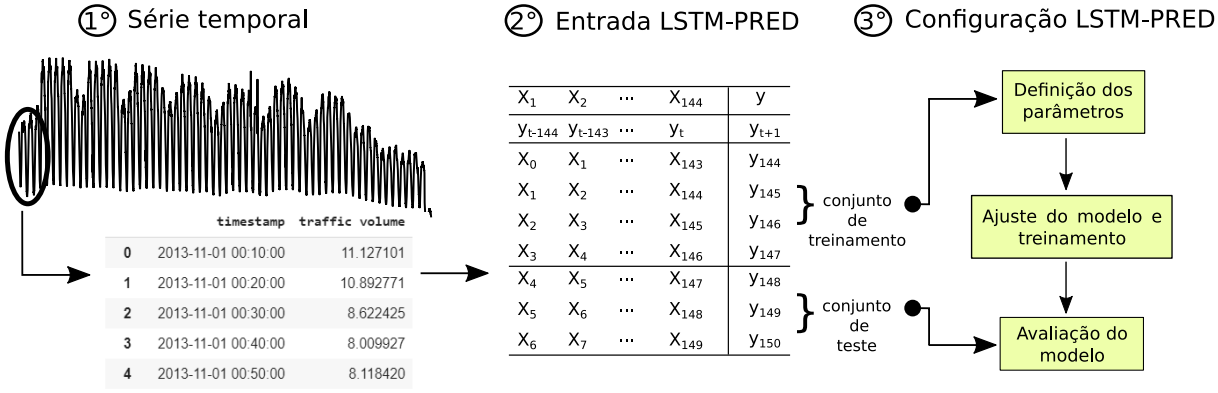
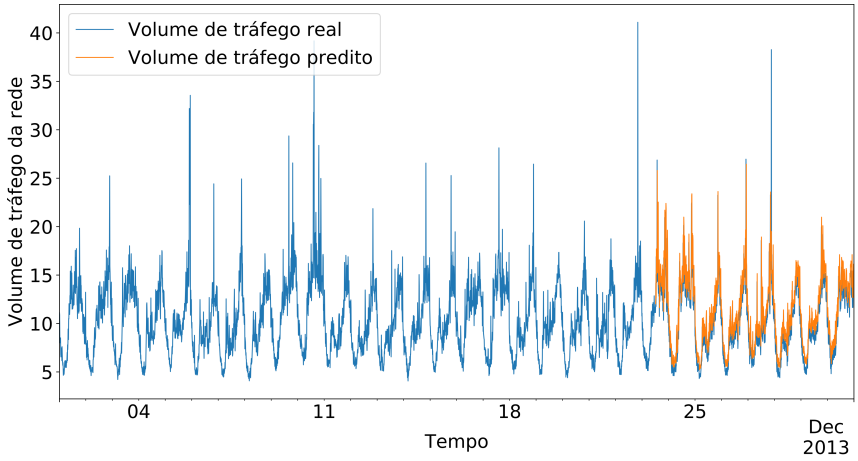


Figura 4.9: Processo de implementação do modelo LSTM-PRED.

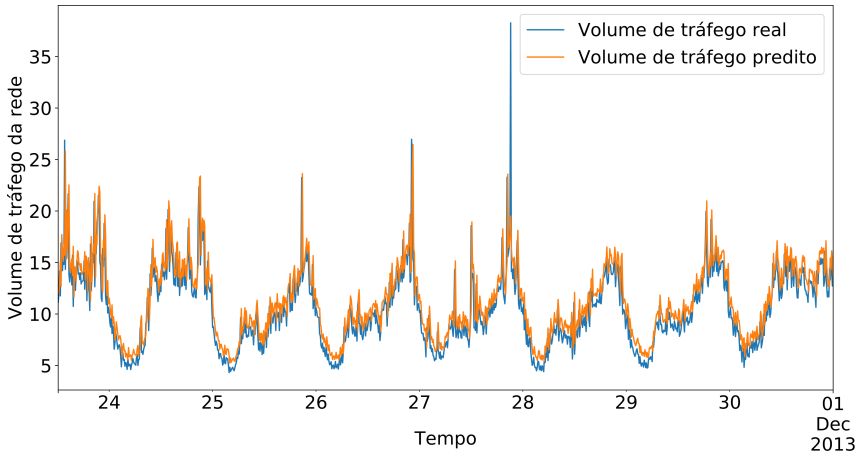
Tabela 4.7: Configurações dos hiperparâmetros.

Hiperparâmetros	**Valores estimados**	**Valores que otimizam o modelo**
Épocas	[1500, 1700, 1800]	1800
Taxa de aprendizagem	[0.001,0.01,0.1,0.0001]	0.01
Optimizador	[Nadam, Adam, RMSProp]	adam
Função perda	[logcosh, mae, mse, hinge, squared_hinge]	mae
Função de ativação	[relu, linear, sigmoid, hard_sigmoid, tanh]	sigmoid
Número de camada oculta	[1,2,3]	2
Dimensão da camada oculta	[200, 400, 600]	200
Tamanho da janela deslizante	144	

Após a fase de definição dos hiperparâmetros e treinamento, o modelo é testado e avaliado. Os valores médios para a MAE e RMSE considerando os hiperparâmetros que otimizam o modelo são 0.97 e 1.35, respectivamente. Para o auxílio da análise gráfica, a Figura 4.10 mostra o comparativo entre o volume de tráfego real e predito utilizando o modelo LSTM-PRED. A Figura 4.10(a) ilustra o volume de tráfego real (em azul) e predito (em laranja). A Figura 4.10(b) ilustra o conjunto de teste (em azul) e predito (em laranja).



(a) Conjunto de treinamento (azul). Valores preditos (laranja).



(b) Comparação entre o volume de tráfego real e predito.

Figura 4.10: Visualização da predição utilizando o modelo LSTM-PRED.

Por meio dos resultados numéricos e ilustrativos, o modelo LSTM-PRED também mostrou-se capaz de estimar o volume de tráfego durante a fase de testes. A complexidade de tempo associada a fase de treinamento e predição em tempo real é $O(n_c \times (n_c + n_o))$ e $O(1)$, onde n_c e n_o é o número de células de memória e saída, respectivamente. A próxima seção apresenta uma breve discussão de ambos os modelos a fim de definir somente um para utilizar na fase de aplicação.

4.3.3 Definição do modelo a ser utilizado na fase de aplicação

Essa seção apresenta e compara os resultados das métricas MAE e RMSE em relação aos modelos ARIMA-PRED e LSTM-PRED com o objetivo de definir o modelo que será utilizado na fase de aplicação. A Tabela 4.8 exibe os valores relativos a essas métricas para ambos os modelos.

Tabela 4.8: Comparação das métricas MAE e RMSE para ambos os modelos.

Modelo/Métrica	**MAE**	**RMSE**
ARIMA	1.092	3.645
LSTM	0.97	1.35

Como notado, ambos produzem resultados satisfatórios considerando a predição de um mês do volume de tráfego da rede de Milão. Entretanto, devido a irregularidade da série temporal, os resultados dos modelos baseados em ARIMA, geralmente, são inferiores aos obtidos por outros métodos, como os baseados em LSTM. Os modelos ARIMA apresentam melhores resultados quando a série é relativamente longa e bem comportada, mas dependem de um maior conhecimento dos componentes da série temporal. Já as RNNs, são capazes de se adaptar melhor para aprender as dependências temporais do contexto sem muito conhecimento prévio dos componentes da série. Assim, de acordo com a análise e os resultados apresentados o modelo utilizado na fase de aplicação é o LSTM-PRED.

4.4 Algoritmo SMART-FL + tráfego predito

Considere a performance do algoritmo **SMART-FL** apresentado na Seção 4.2 e o tempo atual $t(n)$. Logo, seja $A_{t(n)}$ o conjunto de nós selecionados para alocar os serviços multimídia no tempo $t(n)$; $P(A_{t(n+1)})$ o conjunto de nós selecionados em $t(n)$ considerando o volume de tráfego predito para alocar os serviços multimídia no tempo $t(n+1)$; e $A_{t(n+1)}$ o conjunto de nós selecionados em $t(n+1)$ para alocar os serviços multimídia no tempo $t(n+1)$. Assim, o conjunto $Y = A_{t(n)} \cap P(A_{t(n+1)})$ contém os nós reservados para alocar os serviços multimídia no tempo $t(n+1)$, caso $Y \neq \{\emptyset\}$. Isso significa que, os recursos disponíveis por nós que executam serviços multimídia e que serão finalizados em $t(n)$ pertencentes ao conjunto Y são reservados para alocarem os serviços multimídia no tempo $t(n+1)$. Dessa forma, os serviços multimídia que serão alocados em Y requisitados em $t(n+1)$ não disputarão recursos com serviços concorrentes em $t(n+1)$. Então, $\Gamma = Y \cup P(A_{t(n+1)})$ contém os nós mais adequados para alocar os serviços multimídia no tempo $t(n+1)$. A vantagem é que, os nós em Γ oferecem latência inferior do que os nós em $A_{t(n+1)}$.

A Figura 4.11 ilustra um exemplo considerando dois cenários com a mesma demanda, com e sem predição do volume de tráfego. Para o cenário sem predição, os nós selecionados em $t(0)$ e $t(1)$ são $A_{t(0)} = \{E, C\}$ e $A_{t(1)} = \{B, C, D\}$, respectivamente. Nesse caso, não há nenhum nó reservado e o serviço concorrente, requisitado em $t(1)$, é executado em E. Para o cenário com predição do volume de tráfego, os nós selecionados são $A_{t(0)} = \{E, C\}$, $A_{t(1)} = \{B, C, D\}$ e $P(A_1) = \{E, C, D\}$ para o tempo $t(0)$, $t(1)$ e $t(1)$ predito, respectivamente. Assim, $\Gamma = \{E, C, D\}$. Considerando que os serviços multimídia posicionados em $A_{t(0)}$ serão finalizados em $t(0)$ e $Y = \{E, C\}$ contém os nós reservados para alocar serviços multimídia em $t(1)$, o serviço concorrente que seria executado em E no tempo $t(1)$ é executado em B. Dessa forma, os nós em Γ oferecem latência inferior do que os nós em $A_{t(1)}$. A Tabela 4.9 resume as notações utilizadas nesta seção.

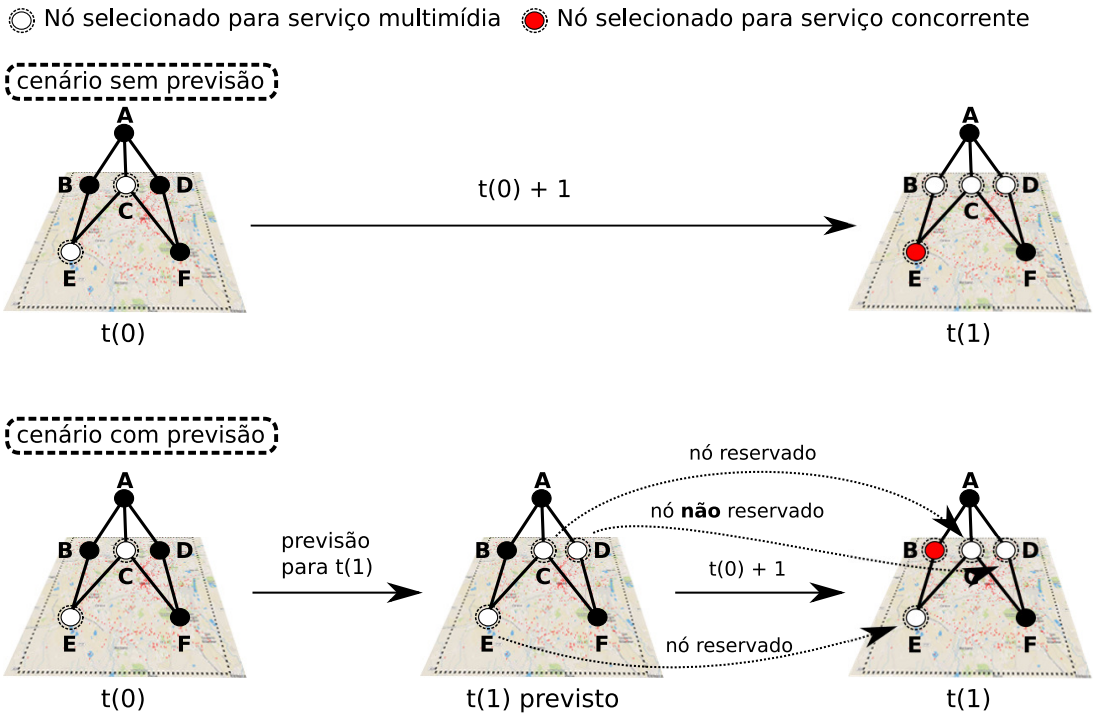


Figura 4.11: Posicionamento dos serviços multimídia ciente do volume de tráfego predito.

Tabela 4.9: Resumo das notações utilizadas.

Notação	Descrição
$A_{t(n)}$	conjunto de nós selecionados no tempo $t(n)$.
$A_{t(n+1)}$	conjunto de nós selecionados para o tempo $t(n+1)$.
$P(A_{t(n+1)})$	conjunto de nós selecionados para o tempo $t(n+1)$ considerando o volume de tráfego predito.
Y	conjunto de nós reservados para o tempo $t(n+1)$.
Γ	conjunto de nós mais adequados para alocar os serviços multimídia no tempo $t(n+1)$.

Os serviços concorrentes são migrados para nós de diferentes camadas se, e somente se, a capacidade de armazenamento e a latência oferecida são adequadas para tais serviços. Caso contrário, os nós em Y não são reservados. Os custos de migrações, alocações e reservas por recursos são considerados pelo simulador e não são avaliados neste trabalho.

4.5 MultiTierFogSim

Embora existam simuladores para avaliar o comportamento e o desempenho de aplicações e serviços em ambientes de Computação em Nuvem, Névoa e Borda [14,60], nenhum permite a avaliação do posicionamento de serviços multimídia em ambientes de Computação em Nuvem-Névoa com suporte a migrações e políticas relacionadas a esse ambiente. Motivados por essas carências, é estendido um simulador denominado **MultiTierFogSim**

baseados nos simuladores de eventos Cloud Simulator (CloudSim), IoT and Fog Simulator (IFogSim)[3], e Simulation of Mobility and Migration for Fog Computing (MobFogSim)[4].

O CloudSim é um simulador extensível desenvolvido por uma equipe de pesquisadores no Laboratório de Computação em Nuvem e Sistemas Distribuídos (CLOUDS) da Universidade de Melbourne. Através dele, é possível modelar e simular características relacionadas a infraestruturas e serviços de aplicações e serviços baseados em Nuvem [13]. O IFogSim é um simulador que extende o Cloudsim é foi desenvolvido para avaliar e quantificar o desempenho das políticas de gerenciamento de recursos em ambientes de Computação em Névoa e os dispositivos IoT. O MobFogSim é um simulador baseado no IFogSim, projetado para permitir a modelagem da mobilidade dos usuários e a migração de serviços na Computação em Névoa. A migração das máquinas virtuais ou containers entre os nós névoas leva em consideração a localização geográfica dos nós névoas, a direção e a velocidade do usuário e as características da rede. O MobFogSim foi avaliado através de resultados de simulações comparados com os obtidos em um banco de testes real, onde os serviços da Névoa foram alocados em containers. Os resultados experimentais levaram em consideração vários padrões de mobilidade de um usuário, derivados do Luxembourg SUMO Traffic (LuST). Os experimentos demonstraram que o MobFogSim é útil para avaliar ambientes baseado em Computação em Névoa, assim como para a avaliação das aplicações e serviços nas quais são solicitadas por usuários móveis, onde é necessário migrar os dados entre os nós névoas.

Os simuladores descritos anteriormente já implementam componentes importantes para simular ambientes baseados em Computação em Nuvem, Névoa e Borda. No entanto, não modelam **i**) serviços multimídia; **ii**) ambientes Nuvem-Névoa-Borda; **iii**) migração de serviços em ambientes Nuvem-Névoa; **iv**) avaliação da migração de serviços e políticas relacionadas em ambientes hierárquicos Nuvem-Névoa. O **MultiTierFogSim** foi projetado durante o mestrado para superar essas limitações.

O **MultiTierFogSim** é uma extensão do MobFogSim que implementa suporte de simulação para ambientes hierárquicos Nuvem-Névoa. Além disso, oferece suporte a simulação e avaliação desses ambientes, definindo conexões entre dispositivos de borda, névoa (*cloudlets* ou *gateways* de névoa) e data centers na Nuvem. O **MultiTierFogSim** manteve a implementação principal do CloudSim para realizar o processamento de eventos entre os componentes da Nuvem, Névoa e Borda. Além disso, foram adicionadas/estendidas novas classes e métodos para suportar avaliações nesses ambientes. As principais classes são brevemente descritas a seguir.

- **RegionalCloud & Cloudlet**: estende *FogDevice* e representa a clouds regionais e cloudlets, respectivamente.
- **MigrationStrategyBtwTiers**: implementa as estratégias de migrações dos serviços entra os nós nuvens, névoas e bordas.
- **ManagerMultitier**: coordena as aplicações e recursos em ambientes hierárquicos. Além disso, gerencia os mecanismos de transferência e conexões/desconexões dos

[3]https://github.com/Cloudslab/IFogSim
[4]https://github.com/diogomg/MobFogSim

dispositivos finais.

- **MultimediaApplication**: representa, de forma abstrata, os serviços multimídia.

A Figura 4.12 mostra uma visão geral dessas modificações.

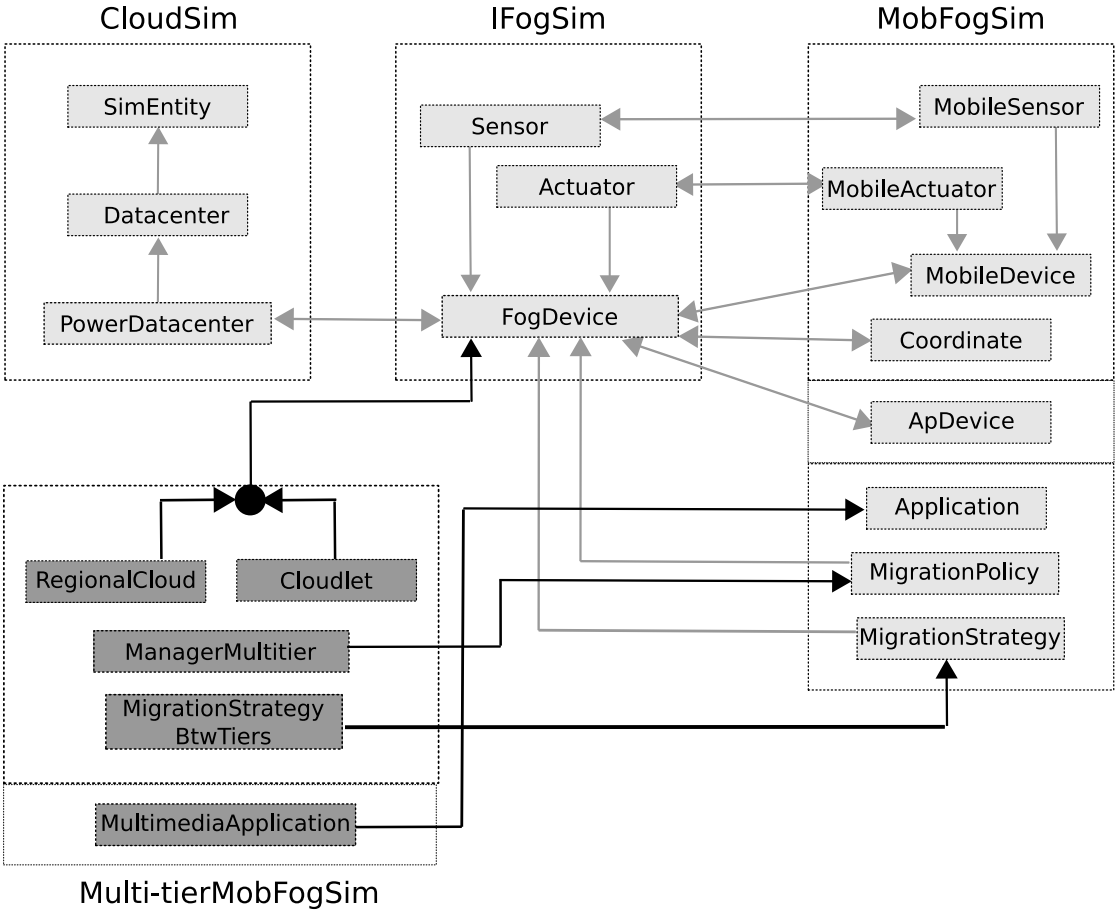


Figura 4.12: Visão geral do MultiTierFogSim como uma extensão de CloudSim, IFogSim e MobFogSim.

A parte mais à esquerda mostra as classes CloudSim que são importantes para criar as entidades relevantes para o IFogSim, MobFogSim e **MultiTierFogSim**. A classe *FogDevice* (estende o *PowerDatacenter* do CloudSim), que representa os recursos de *hardware* de um nó névoa ou IoT, simboliza uma das principais classes de todos os simuladores. Os principais atributos dessa classe são Memória de Acesso Aleatório (*Random Access Memory* - RAM), Milhões de Instruções Por Segundo (*Million of Instructions Per Second* - MIPS), armazenamento dos nós e largura de banda (*uplink* e *downlink*). Os métodos dessa classe executam tarefas específicas de um dispositivo de névoa ou borda para processar as requisições recebidas. A parte mais à direita mostra as principais classes introduzidas com o MobFogSim. Nesse simulador, é possível implementar a mobilidade dos dispositivos, *handoff*/*handover* e migração de máquina virtual ou container entre os nós névoas.

As classes contidas em **MultiTierFogSim** implementam suporte a aplicações multimídias, coordenação das aplicações e serviços em ambientes hierárquicos e suporte a

migrações entre nós na Nuvem, Névoa e Borda. As aplicações e serviços multimídia são modeladas pela classe *MultimediaApplication*. Essa classe modela essas aplicações e serviços como um grafo acíclico direcionado. Os vértices são os módulos de execução de chegada de uma tupla. As arestas são dependências de dados entre os módulos (vértices). A migração dessas aplicações e serviços é gerenciada por objetos da classe *Manager-Multitier* e *MigrationStrategyBtwTiers*. Esses objetos são responsáveis pela ativação da migração e a instanciação dos serviços. Nós das camadas superiores as estações base são representados pela classes RegionalCloud e *Cloudlet*. As métricas disponíveis para a avaliação das aplicações são tempo de migração, tempo de inatividade, bits transferidos durante as migrações, número de migrações, tempo médio de cada migração, latência, pacotes entregues e perdidos, dentre outras.

O ambiente e o algoritmo proposto são simulados e avaliados utilizando o **MultitierFogSim** considerando valores realísticos para as condições da rede e dos serviços oferecidos, baseados nos trabalhos [46, 79]. A simulação é executada 30 vezes com as condições da rede variando dinamicamente durante toda a simulação. Os parâmetros da simulação são descritos na Tabela 4.10.

Tabela 4.10: Parâmetros do simulador.

Parâmetro	**Valor**
Virtualização	Container
Migração	*Live migration*
Tamanho do container	128MB
Área de cobertura (por estações base)	500m
Tempo max. de simulação	40min.
Densidade de nós névoas por pontos de acesso	164:1

As simulações terminam quando todas as migrações são finalizadas ou o tempo máximo de simulação é alcançado. O destino dos serviços no processo de migração é escolhido com base nos algoritmos de posicionamento. Os custos associados a migração e alocação de recursos são considerados pelo o simulador. A estratégia de migração assumida nesta avaliação seleciona os nós que minimizam a latência entre um conjunto de nós candidatos que estão presentes no cenário.

Capítulo 5

Resultados

Este capítulo avalia os resultados em duas seções. A Seção 5.1 apresenta a avaliação de desempenho em seis perfis de tráfego selecionados em dias e horas específicos baseados na intensidade do volume de tráfego. A Seção 5.2 apresenta a avaliação de desempenho considerando um mês do volume de tráfego predito da rede móvel. Seção 5.3 discute os impactos dos resultados alcançados.

5.1 Avaliação considerando a intensidade do volume de tráfego

Nesta seção, são analisados seis perfis de tráfego selecionados em dias e horas específicos com base na intensidade do volume de tráfego da rede para avaliar o desempenho do **SMART-FL** em diferentes circunstâncias. O objetivo é analisar o posicionamento dos nós selecionados em relação a sua capacidade de armazenamento, a latência oferecida e a localização geográfica das requisições. Para agrupar a intensidade do volume de tráfego, é aplicado o método **K-means**, um algoritmo de agrupamento não hierárquico que objetiva particionar n observações dentre k grupos onde cada observação pertence ao grupo mais próximo da média. Para a utilização desse método, o analista deve especificar somente o número de grupos k, que nesse caso são três. Assim, a intensidade do volume de tráfego da rede é particionada em **baixa**, **média** e **alta**. Nesse cenário, os serviços multimídia também são classificados de acordo com a latência e armazenamento necessário para atendimento e relacionados aos grupos de intensidade do volume de tráfego. De forma geral, o volume de tráfego da rede foi agrupado como baixo $[0, \approx 10]$, médio $(\approx 10, \approx 14]$ e alto $(\approx 14, +\infty)$. A Figura 5.1 ilustra uma parte do conjunto de treinamento durante a primeira semana de Novembro.

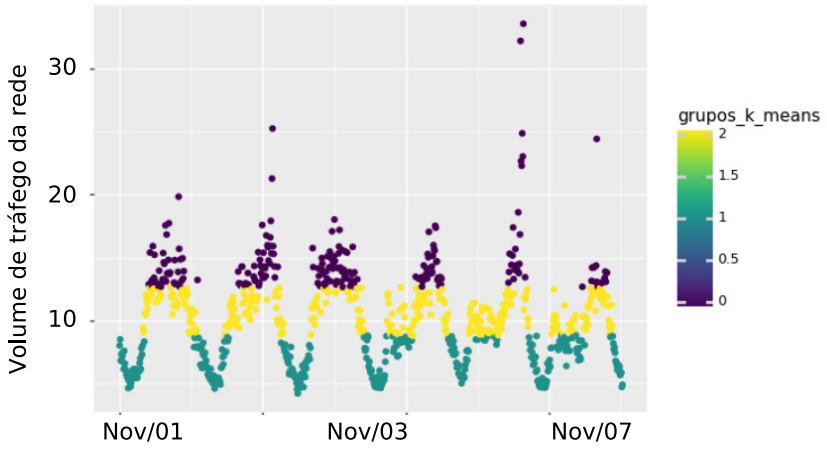


Figura 5.1: Agrupamento do volume de tráfego.

As Figuras 5.2-5.7 mostram as áreas com requisições de serviços multimídia e os atributos dos nós na região de Milão para cada perfil de tráfego. As Figuras 5.2(a)-5.7(a) ilustram o cenário com as requisições por serviços multimídia, bem como os nós selecionados para fornecer esses serviços. Nesse caso, as regiões em cinza correspondem ao conjunto G_t. Os círculos representam os nós habilitados para fornecer os serviços multimídia. As Figuras 5.2(b)-5.7(b) mostram a capacidade de armazenamento dos nós (eixo x) e latência (eixo y) naquele momento. Essa análise oferece uma ideia aproximada, mas intuitiva, dos nós selecionados em diferentes possibilidades.

A Figura 5.2(a) mostra o primeiro perfil de tráfego (17/11/2013 às 06:00). É associado ao grupo de baixa intensidade que ocorre durante o amanhecer nos finais de semana. Assim, os nós selecionados são **CL1**, **CL2**, **CL3** e **CL5**. A Figura 5.2(b) mostra que esses nós possuem capacidade de armazenamento apropriada e a menor latência para atender a essa demanda. Além disso, eles estão posicionados geograficamente próximos as regiões de demanda, reduzindo a latência e melhorando a experiência do usuário.

A Figura 5.3(a) mostra o segundo perfil de tráfego (10/12/2013 às 09:30). É associado ao grupo de média intensidade que ocorre em alguns momentos no período da manhã. Neste caso, todos os nós apresentam um atraso máximo aceitável para prover os serviços multimídia, ou seja, menor que 0.1 segundos. Além disso, todos os nós *cloudlets* estão com baixa capacidade de armazenamento (devido a execução e armazenamento de serviços concorrentes, por exemplo). Portanto, os nós selecionados são **CLOUD**, **RC1** e **RC2**. A Figura 5.3(b) mostra a capacidade de armazenamento e latência dos nós no momento analisado.

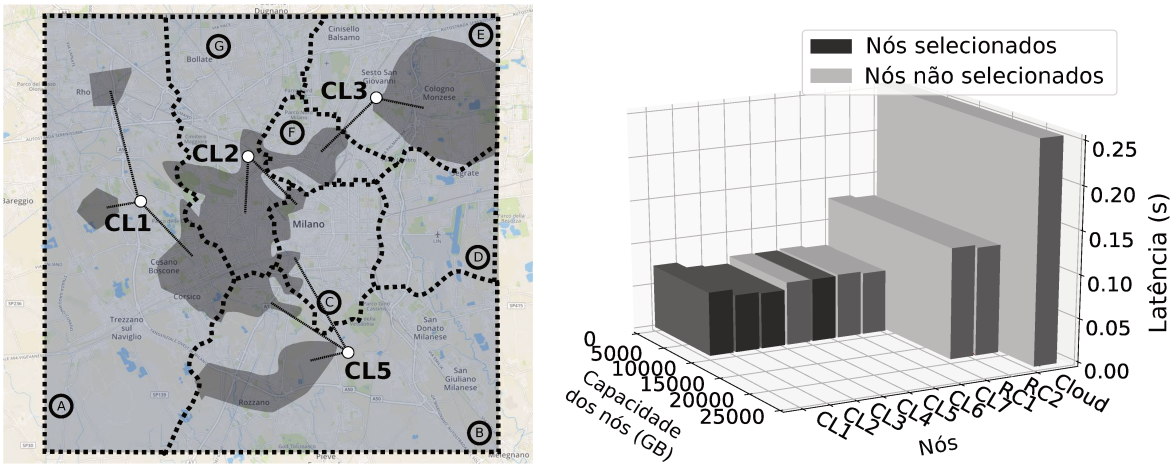
A Figura 5.4(a) mostra o terceiro perfil de tráfego (20/10/2013 às 11:30). É associado ao grupo de média intensidade que ocorre durante as manhãs nos finais de semana. Novamente, todos os nós possuem o atraso máximo aceitável para fornecer os serviços multimídia. Conforme mostrado na Figura 5.4(b), apenas o nó nuvem **CL** e alguns nós *cloudlets* possuem capacidade de armazenamento para atender a essa demanda. Portanto, o nós selecionados são **CLOUD**, **CL2**, **CL3**, **CL4** e **CL5**.

A Figura 5.5(a) mostra o quarto perfil de tráfego (10/11/2013 às 5:00). É associado ao grupo de média intensidade que ocorre durante o amanhecer específico de um final de semana. Com base na Figura 5.5(b), o nó nuvem **CL** oferece latência maior do que a

máxima desejável e os nós **CL1**, **CL5** e **CL6** possuem baixa capacidade de armazenamento para atender a essa demanda. Portanto, com base na localização geográfica, capacidade de armazenamento e latência dos nós, os nós selecionados são **RC1**, **RC2**, **CL2**, **CL3** e **CL4**.

A Figura 5.6(a) mostra o quinto perfil de tráfego (20/12/2013 às 13:40). É associado ao grupo de alta intensidade que ocorre durante o horário de trabalho às sextas-feiras. A Figura 5.6(b) mostra que todos os nós possuem latência inferior ao atraso máximo aceitável e capacidade de armazenamento adequada. Mais uma vez, com base na localização geográfica das requisições, capacidade de armazenamento e latência dos nós, os nós selecionados são **CLOUD**, **RC1**, **RC2**, **CL1**, **CL2**, **CL4** e **CL5**.

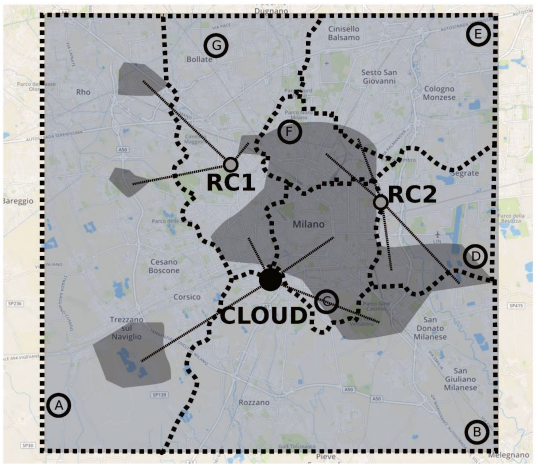
Finalmente, a Figura 5.7(a) mostra o sexto perfil de tráfego (27/12/2013 às 14:30). Mais uma vez, é associado ao grupo de alta intensidade que ocorre durante o horário de trabalho às sextas-feiras. A Figura 5.7(b) mostra que todos os nós possuem latência inferior ao atraso máximo aceitável, mas baixa capacidade de armazenamento. Neste caso especial, todos os nós são selecionados para atender o máximo possível das requisições. Dessa forma, algumas regiões não serão atendidas ou alguns usuários terão sua taxa de vídeo adaptada, afetando a qualidade de experiência devido à baixa capacidade de armazenamento dos nós.



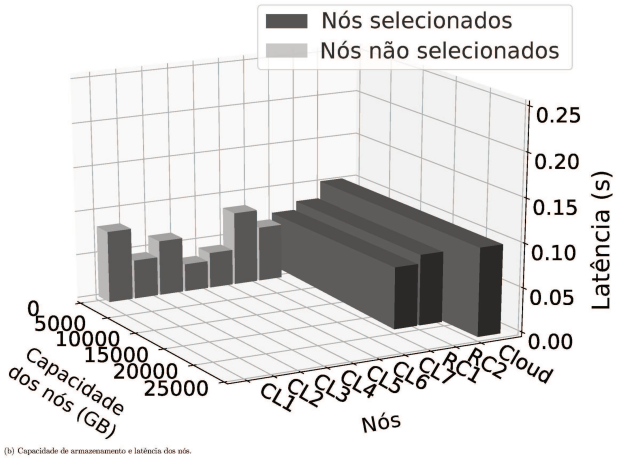
(a) Áreas com requisições multimídia e nós selecionados.

(b) Capacidade de armazenamento e latência dos nós.

Figura 5.2: Intensidade de tráfego baixa.

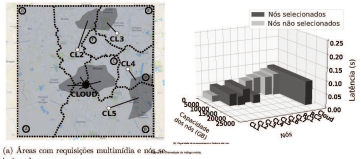


(a) Áreas com requisições multimídia e nós selecionados.

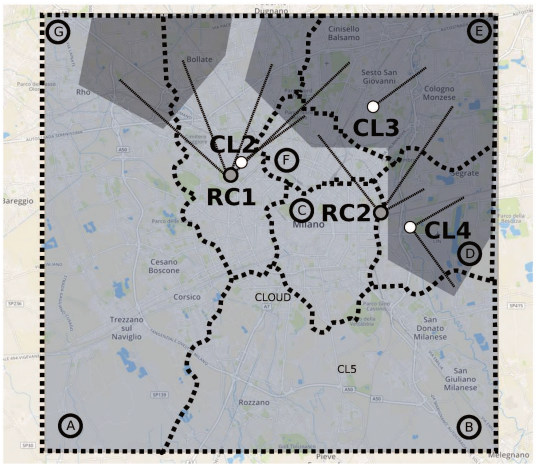


(b) Capacidade de armazenamento e latência dos nós.

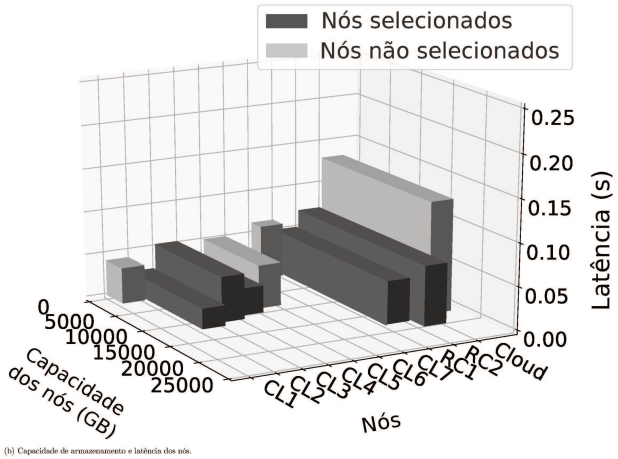
Figura 5.1: Intensidade de tráfego média.



(a) Áreas com requisições multimídia e nós selecionados.



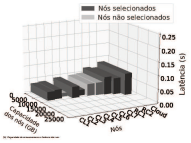
(a) Áreas com requisições multimídia e nós selecionados.

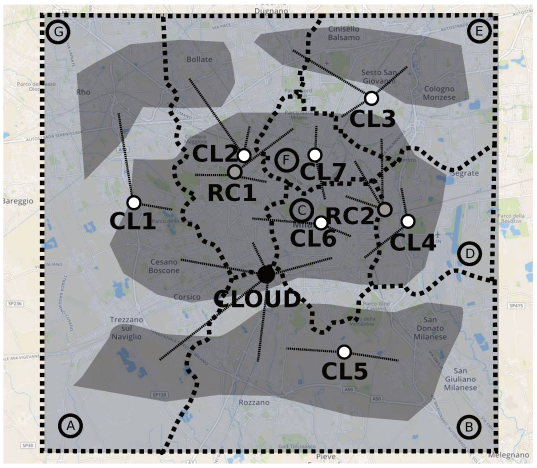


(b) Capacidade de armazenamento e latência dos nós.

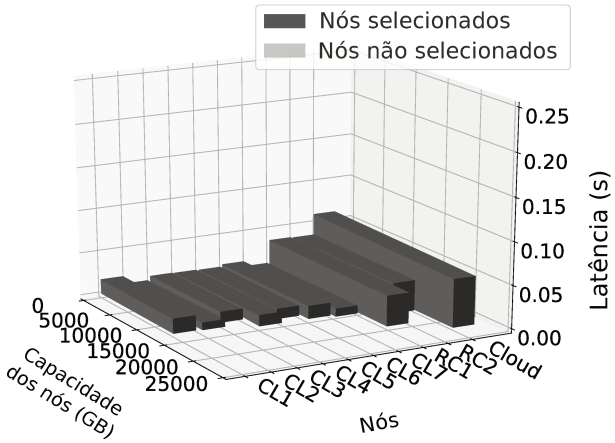
Figura 5.1: Intensidade de tráfego média.

(a) Áreas com requisições multimídia e nós selecionados.





(a) Áreas com requisições multimídia e nós selecionados.



(b) Capacidade de armazenamento e latência dos nós.

Figura 5.7 – Intensidade de tráfego alta.

5.2 Avaliação considerando a predição do volume de tráfego

Nesta seção, a avaliação de desempenho é realizada considerando um mês do volume de tráfego predito da rede móvel em Milão - Itália. Os resultados são analisados em termos da (**i**) latência; (**ii**) pacotes entregues; (**iii**) requisições atendidas; (**iv**) uso da rede em termos do (a) volume total de dados transmitidos durante a migração e (b) uso do enlace, considerando somente os serviços da classe multimídia; e são comparados considerando seis estratégias de posicionamento:

1. **Tier 1:** Todos os serviços multimídia são posicionados na Nuvem;
2. **Tier 2:** Todos os serviços multimídia são posicionados na Nuvem regional;
3. **Tier 3:** Todos os serviços multimídia são posicionados nas *cloudlets*;
4. **Ajuste dinâmico:** Os serviços multimídia são posicionados nas camadas próximas aos usuários.
5. **SMART-FL:** Os serviços multimídia são posicionados de acordo com o algoritmo **SMART-FL**;
6. **SMART-FL + predição:** Os serviços multimídia são posicionados de acordo com o algoritmo **SMART-FL** ciente do volume de tráfego predito.

O atraso máximo aceitável para fornecer os serviços multimídia é inferior a 0,1 s [43]. Para todas as condições, a falta de recursos dos nós acarreta no não atendimento dos serviços. O intervalo de confiança considerado para todos os resultados é de 95%. As discussões a seguir são baseadas na Figuras 5.8-5.10, que ilustram a latência, uso da rede, requisições

atendidas e pacotes entregues, respectivamente, para as seis estratégias de posicionamento de serviços multimídia.

A Figura 5.8 exibe o percentual da taxa de requisições atendidas e pacotes entregues para todas as estratégias de posicionamentos considerando somente as requisições por serviços multimídia.

Devido a alta capacidade de armazenamento e processamento, a estrategia Tier 1 atende $\approx$ 100% das requisições. Em contrapartida, somente $\approx$ 80% dos pacotes são entregues. Nessa camada, os usuários podem armazenar com eficiência os conteúdos multimídias de qualquer tipo e tamanho sem dificuldades. Além disso, é possível processá-los de forma eficiente, já que esses conteúdos exigem tempo de computação mais complexo em *hardware* [86]. Por um lado, a estratégia Tier 2 atende $\approx$ 93% das requisições com $\approx$ 82% dos pacotes entregues. Por outro lado, a estratégia Tier 3 atende somente $\approx$ 50% das requisições com $\approx$ 100% dos pacotes entregues. É notado que, a taxa de requisições atendidas relaciona-se com a capacidade de armazenamento dos nós. Isso significa que, camadas superiores tendem a atender mais serviços do que as camadas inferiores, devido a restrição da capacidade. Em contrapartida, as camadas inferiores oferecem serviços com alta velocidade e conexões com enlaces mais confiáveis, reduzindo a taxa de pacotes perdidos, diferentemente de quando os serviços estão posicionados na Nuvem. A Tier 3 é a camada mais próxima do usuário e, consequentemente, com poucos recursos computacionais (processamento, armazenamento, ...).

Devido a capacidade de localizar recursos em camadas superiores, a estratégia Ajuste dinâmico atende $\approx$ 100% das requisições com $\approx$ 90% dos pacotes entregues. Como discutido anteriormente, essa estratégia inicia a busca por recursos selecionando nós pertencentes a camada mais próxima do usuário, seguida pelas demais na hierarquia. Caso não encontre recursos suficientes na camada atual, a camada superior é considerada. No entanto, essa estratégia não explora os benefícios do posicionamento dos serviços em camadas distintas, o que pode aumentar ainda mais a taxa de requisições atendidas e pacotes entregues. O armazenamento e a computação dos dados distribuídos de forma mais lógica e eficiente, próximos aos usuários, é performado pelas estratégias SMART-FL e SMART-FL + predição.

Ambas as estratégias apresentam vantagens superiores a todas discutidas até então. A SMART-FL atende $\approx 100\%$ das requisições com $\approx 96\%$ dos pacotes entregues. A SMART-FL + predição atende $\approx 100\%$ das requisições com $\approx 98\%$ dos pacotes entregues, sendo superior a todas as estratégias. Isso deve-se ao fato do volume de tráfego predito, pois o posicionamento torna-se ainda mais adequado em razão do conjunto de nós alocados em Γ.

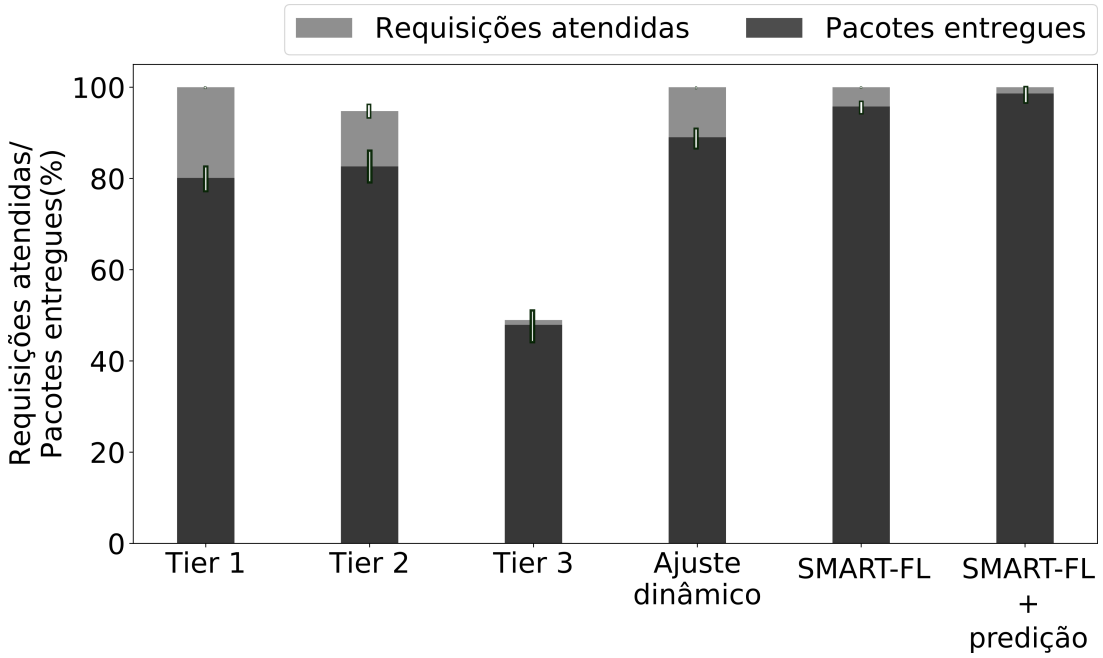


Figura 5.8: Comparativo da taxa de requisições atendidas e pacotes entregues.

A Figura 5.9 exibe a latência média para todas as estratégias de posicionamentos considerando somente as requisições por serviços multimídia.

Os serviços posicionados considerando a estratégia Tier 1 são oferecidos com latência média superior a máxima aceitável, isto é, superior a 0,1 s. É visto que, a distância física aumenta inevitavelmente a latência. Em contrapartida, esses serviços posicionados considerando as estratégias Tier 2 e Tier 3 são oferecidos com latência média inferior a máxima aceitável. Ambas apresentam vantagens em relação a Tier 1, como latência inferior a máxima aceitável (Tier 2 e Tier 3) e taxas de pacotes entregues superiores (Tier 2). Entretanto, ambas apresentam altas taxas de requisições não atendidas e pacotes perdidos. Todos esses fatores aumentam o atraso e entregam tais conteúdos com baixa QoE. Soluções baseados nas estratégias **first fit** e **best fit** tornam-se mais eficientes ao combinarem os recursos providos por ambientes hierárquicos Nuvem-Névoa.

A estratégia Ajuste dinâmico apresenta latência média inferior a máxima aceitável. Além disso, apresenta taxas de requisições e pacotes entregues superiores as estratégias Tier 1, Tier 2 e Tier 3, conforme discutido anteriormente. Entretanto, essa estratégia não explora os benefícios da distribuição dos serviços ao longo da rede, o que pode reduzir ainda mais a latência. As estratégias SMART-FL e SMART-FL + predição atendem esses requisitos juntamente com a seleção de nós que minimizam a latência para entregar esses serviços. Ambas estratégias oferecem serviços com latência média inferior a máxima aceitável e menor que as estratégias Tier 1, Tier 2 e Ajuste dinâmico, exceto em comparação com a estratégia Tier 3, que em contrapartida atende somente $\approx$ 50% das requisições devido a restrição de armazenamento.

A estratégia SMART-FL seleciona nós distribuídos ao longo da rede e próximos aos usuários, diminuindo a quantidade de saltos e, consequentemente, diminuindo a latência para entregar esses serviços. Em comparação, a estratégia SMART-FL + predição oferece a menor latência média dentre todas as estratégias, exceto pela Tier 3, que apresenta

valores já justificados. A razão é que, os nós em Γ estão em camadas mais próximas dos usuários, e consequentemente com latência inferior do que os nós em $A_{t(n+1)}$.

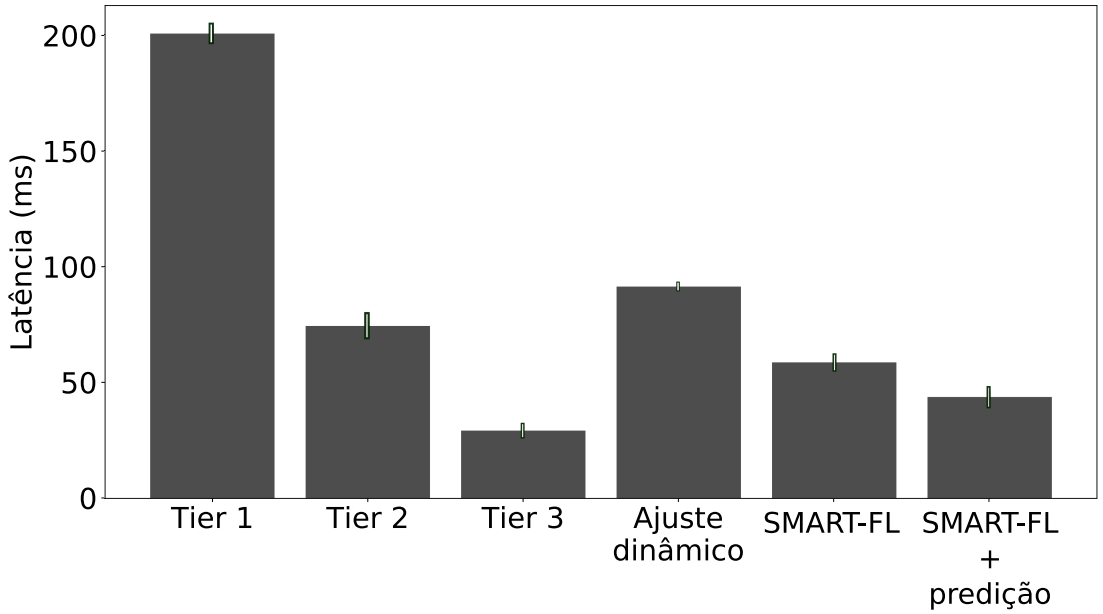


Figura 5.9: Comparativo da latência.

A Figura 5.10 exibe o uso da rede e das migrações por todas as estratégias de posicionamentos considerando somente as requisições por serviços multimídia.

A estratégia Tier 1 apresenta uso da rede maior do que todas as estratégias de posicionamento. Isso deve-se ao fato da alta quantidade de enlaces utilizados entre a Nuvem e os usuários para fornecer tais serviços. Apesar dos muitos benefícios oferecidos, como alta disponibilidade, processamento e armazenamento, essa estratégia apresenta latência superior a máxima aceitável, problemas de enlaces não confiáveis com perdas de pacotes e alto uso da rede. Os serviços multimídia exigem fluxo constante e contínuo de pacotes com baixa latência, requisitos que essa estratégia ocasionalmente não fornece.

A aproximação desses serviços para a borda da rede reduz a necessidade geral do uso da rede, pois menos canais para transmissão dos dados são utilizados, diferentemente quando os serviços estão posicionados na Nuvem. Dessa forma, as estratégias Tier 2 e Tier 3 apresentam uso da rede médio inferior a Tier 1. Entretanto, como já discutido, ambas apresentam altas taxas de requisições não atendidas e pacotes perdidos.

A estratégia Ajuste dinâmico apresenta uso da rede superior a Tier 3 devido a alta taxa de requisições atendidas e, consequentemente, maior uso dos enlaces e inferior a Tier 1 e Tier 2 devido ao posicionamento mais adequado dos serviços. Em comparação com as abordagens anteriores, é acrescentado um custo de $\approx$ 29391592.17 bits por segundo (bps) devido às migrações dos serviços multimídia para as camadas mais adequada mediante as variações dos recursos da rede, localização da requisição e capacidade de processamento e armazenamento dos nós. Esse custo torna-se mínimo em relação as vantagens apresentadas e comparadas com as estratégias Tier 1, Tier 2 e Tier 3. Contudo, como mencionado, os benefícios da distribuição dos serviços ao longo da rede não é explorado, o que pode reduzir ainda mais o uso da rede.

A estratégia SMART-FL apresenta uso da rede inferior ao Tier 1, Tier 2 e Ajuste dinâmico e superior a Tier 3. Conforme já discutido, a estratégia Tier 3 apresenta baixos valores relativos a latência e o uso da rede, mas, em contrapartida atende somente $\approx$ 50% das requisições devido a restrição de armazenamento. Igualmente, é acrescentado um custo de $\approx$ 31267651.24 bts devido às migrações dos serviços multimídia para os nós mais adequados mediante as variações dos recursos da rede, localização da requisição e capacidade de processamento e armazenamento dos nós. Esse custo torna-se mínimo mediante as vantagens apresentadas. Essa estratégia apresenta latência inferior a mínima aceitável, taxas adequadas de requisições atendidas e pacotes entregues e uso da rede relativamente baixo.

A estratégia SMART-FL + predição apresenta valores ainda mais satisfatórios. O uso da rede é inferior a todas as estratégias de posicionamento, exceto em comparação com a estratégia Tier 3, por motivos já discutidos anteriormente. Do mesmo modo, é acrescentado um custo de $\approx$ 34176269.96 bts devido às migrações dos serviços multimídia por motivos similares às estratégias anteriores. Esse custo também é mínimo mediante as vantagens apresentadas. Os nós reservados em Γ apresentam vantagens em relação aos nós em $A_{t(n+1)}$. Portanto, dentre todas as estratégias, essa é a que mais se beneficia dos recursos disponíveis.

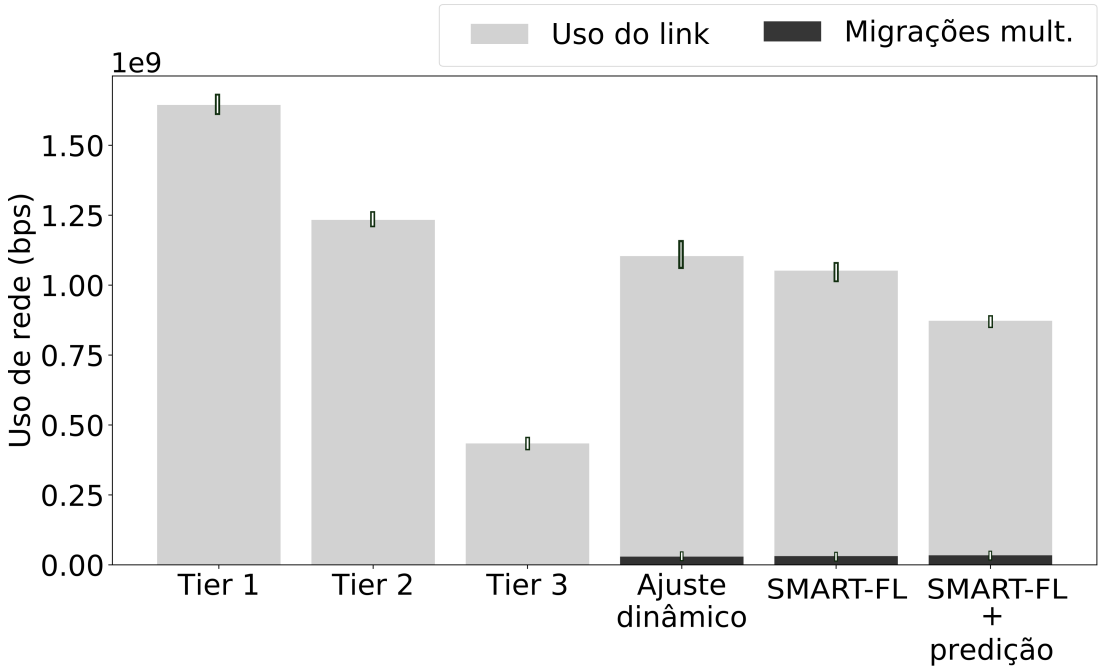


Figura 5.10: Comparativo do uso da rede.

5.3 Considerações Finais

Apesar das vantagens fornecidas pela Computação em Nuvem, como alta disponibilidade e elasticidade, os serviços multimídia exigem fluxo constante e contínuo de pacotes com baixa latência [71]. Embora a sua função permaneça inalterada, a combinação da arquitetura Nuvem-Névoa fornece uma hierarquia de poder de computação que pode armazenar

e processar esses serviços distribuídos próximos aos usuários. Essa combinação oferece várias vantagens, dentre elas a redução da latência, pacotes perdidos e uso da rede.

A distribuição dos serviços ao longo da rede e próximos aos usuários permite o processamento e armazenamento perto da fonte de dados, sem a necessidade do envio de todos esses serviços para a Nuvem remota ou para outros sistemas centralizados. Manter essa análise próxima da fonte de dados, especialmente para serviços sensíveis a latência onde cada milissegundo é importante, além das vantagens apresentadas neste trabalho melhora a experiência do usuário e reduz a sobrecarga na Nuvem como um todo [10]. Como os serviços podem ser processados localmente, sem serem enviados para a Nuvem, o uso da rede total também é reduzido. Com o número cada vez maior de usuários móveis e dispositivos IoT, todos gerando e consumindo dados, essa economia é considerável.

A propósito, a predição do volume de tráfego de rede aprimora de forma relevante o fornecimento dos serviços. A predição inicial antes da necessidade real fornece ao algoritmo SMART-FL a capacidade de impor ações de gerenciamento para manter a latência inferior à máxima e de todas as estratégias apresentadas (exceto pela Tier 3, que apresenta valores inadequados). Além disso, devido a eficiência da alocação de serviços ciente do volume de tráfego predito, há uma redução no uso da rede e pacotes perdidos. A eficácia do SMART-FL em termos de redução da latência, uso da rede, requisições atendidas e pacotes entregues é superior a todas comparadas neste trabalho.

Essa estratégia alcança um bom equilíbrio entre os nós selecionados, sua capacidade de armazenamento e a latência oferecida. Aliás, a latência é uma das métricas que mais afetam a qualidade dos serviços multimídia e a QoE [86]. Vale a pena mencionar que, o algoritmo SMART-FL pode ser adaptado para outros serviços, por exemplo, serviços de TCS, IoT, realidade aumentada e outros que também aproveitam das vantagens oferecidas pelo ambiente hierárquico Nuvem-Névoa.

Capítulo 6

Conclusão

Espera-se que uma ampla gama de aplicações e serviços multimídia seja oferecida a usuários móveis através de várias redes de acesso sem fio. No geral, esse crescimento pode ser atribuído ao rápido avanço das tecnologias de rede e ao crescimento do mercado de dispositivos inteligentes. A transmissão desses conteúdos considerando uma adequada QoE em infraestruturas de redes móveis sem fio é uma questão crítica tanto na comunidade acadêmica quanto industrial. Esses serviços exigem técnicas e abordagens específicas quanto ao seu processo de comunicação.

Com o advento do paradigma de Computação em Névoa, na qual dados, processamento e armazenamento são distribuídos próximos aos usuários finais, a latência oferecida e até mesmo os riscos de segurança podem ser reduzidos, ao mesmo tempo em que otimiza o uso da rede. Quando combinadas, Computação em Nuvem e Névoa proporcionam recursos computacionais eficientes para o atendimento de serviços sensíveis a latência.

Nesta dissertação de mestrado, é proposto um método para a criação de ambientes hierárquicos Nuvem-Névoa e um algoritmo denominado SMART-FL para o problema de posicionamento de serviços multimídia. O objetivo é encontrar o menor conjunto de nós considerando suas capacidades de armazenamento para prover tais serviços de forma que a latência seja minimizada. A extensão de um simulador denominado **MultiTierFogSim** é proposta para avaliar diversas estratégias de posicionamento de serviços multimídia considerando o ambiente desenvolvido.

Os resultados mostram que o algoritmo **SMART-FL + predição** posiciona os serviços multimídia em nós com capacidade de armazenamento adequada, próximos aos usuários e com latência média inferior a todas as estratégias. O posicionamento desses serviços torna-se ainda mais eficiente devido ao conjunto de nós reservados, através da predição do volume de tráfego da rede. Essa distribuição dos serviços ao longo da rede e próximos aos usuários permite o processamento e armazenamento perto da fonte de dados, sem a necessidade do envio de todos esses serviços para a Nuvem remota ou para outros sistemas centralizados.

De forma geral, os resultados apresentados nesta dissertação de mestrado avançam o estado da arte em relação ao posicionamento de serviços multimídia em ambientes hierárquicos Nuvem-Névoa e a modelagem desses ambientes. O algoritmo pode ser adaptado para outros serviços (IoT, TCS, ...) e o ambiente hierárquico também pode oferecer suporte a serviços de cidades inteligentes, como localização de eventos e diminuição do

tráfego veicular.

6.1 Contribuição

As contribuições são resumidas em quatro itens.

1. **Ambientes hierárquicos Nuvem-Névoa:** É proposto um método na Seção 4.1 para a criação de ambientes hierárquicos Nuvem-Névoa. Esse método utiliza uma abordagem *bottom-up*, iniciando-se a partir de um conjunto $BS = \{bs_1, bs_2, ..., bs_{bs}\}$ de estações base, sendo possível aplicá-lo a qualquer cenário com um conjunto semelhante. Considerando o cenário de Milão, os nós são organizados hierarquicamente em quatro camadas: **Nuvem**, **Nuvem Regional**, **Cloudlets** e **Estação Base**. O ambiente proposto pode ser simulado no **MultiTierFogSim**, onde é beneficiado por várias vantagens relacionadas a Nuvem e Névoa, como reconhecimento de localização, capacidade de análise para o processamento, bem como migrações de serviços de e para qualquer camada.

2. **Algoritmo para o problema de posicionamento de serviços multimídia:** É proposto na Seção 4.2 um algoritmo de otimização denominado **SMART-FL** implementado como PLI para o problema de posicionamento de serviços multimídia em ambientes hierárquicos Nuvem-Névoa modelado como um **CFLP**. O objetivo é encontrar o menor conjunto de nós considerando suas capacidades de armazenamento para prover tais serviços de forma que a latência seja minimizada. Os resultados mostraram que o algoritmo **SMART-FL** é capaz de selecionar os nós mais próximos dos usuários e atender as requisições. Além disso, com a redução da demanda dos serviços alocados na Nuvem é possível desligar os servidores e economizar energia.

3. **Predição do volume de tráfego:** São considerados na Seção 4.3 dois modelos, a saber ARIMA-PRED e LSTM-PRED, para a predição do volume de tráfego da rede celular da cidade de Milão. Ambos os modelos foram analisados em relação a vários parâmetros para se aproximar da otimalidade. Em seguida, foram avaliados em relação as métricas MAE e RMSE, apresentando resultados satisfatórios. Entretanto, devido a irregularidade da série, os resultados do modelo ARIMA-PRED foram inferiores aos obtidos pelo modelo LSTM-PRED. Isso deve-se ao fato do modelo LSTM-PRED extrair com mais precisão as características espaço-temporais dos dados e melhorar a precisão da predição. Através do tráfego da rede predito, um conjunto de nós são reservados para alocar os serviços multimídias. Os resultados mostram um aumento de $\approx 3\%$ da taxa de pacotes entregues e uma redução de $\approx 15\%$ da latência média e $\approx 13\%$ do uso da rede em comparação com o algoritmo SMART-FL.

4. **MultiTierFogSim:** Finalmente, é proposto na Seção 4.5 uma extensão do simulador *MobFogSim* para avaliar aplicações e serviços em ambientes hierárquicos Nuvem-Névoa. Esse ambiente é definido através de conexões entre dispositivos de borda, névoa (*cloudlets* ou *gateways* de névoa) e data centers na Nuvem. O simulador

também permite migrações de máquinas virtuais ou containers entre esses dispositivos assim como suporte a mobilidade do usuário. As métricas disponíveis para a avaliação são latência, uso da rede, pacotes entregues e perdidos, tempo de migração, tempo de inatividade, dentre outras. O simulador é extensível e disponível em `https://github.com/fillipesansilva/MultiTierFogSim`.

6.2 Limitações e trabalhos futuros

Os constantes avanços da tecnologia de comunicação e a grande disponibilidade de serviços multimídia trazem a necessidade de novos métodos que garantam a qualidade de experiência aos usuários finais. Neste sentido, o presente trabalho contribui para esse cenário com um algoritmo de posicionamento desses serviços em ambientes hierárquicos Nuvem-Névoa.

Pode-se considerar a avaliação do algoritmo **SMART-FL** em um ambiente capaz de produzir tráfego VoD com comportamento igual ao tráfego produzido por servidores do mundo real. Nesse cenário, os usuários devem ser capazes de consumir este serviço e gerar estatísticas de interesse. Além disso, a mobilidade do usuário pode ser avaliada considerando uma camada específica entre a borda e nuvem para posicionar esses serviços. Para tal, pode-se considerar os custos financeiros referentes a alocação de recurso e o tempo de migração para esses serviços.

Em relação ao método para a criação de ambientes hierárquicos proposto, mais especificamente no passo (**1**), pode-se considerar o agrupamento das estações base em relação aos padrões de tráfego complementares para nós névoas semelhantes. Ao projetar um perfil de tráfego para cada estação base e construir um modelo para representar a complementaridade entre elas, é possível encontrar um agrupamento ideal. Esse agrupamento pode ser através de vários métodos, entre eles a Clusterização Espacial Baseada em Densidade de Aplicações com Ruído (Density Based Spatial Clustering of Application with Noise - DBSCAN), o qual é significativamente efetivo para identificar agrupamentos de diferentes formatos e tamanhos, assim como a identificação e separação de ruídos, sem qualquer informação preliminar sobre os grupos.

Vale a pena mencionar que a análise e modelagem apresentada dos algoritmos de predição de tráfego celular em problemas de predição de tráfego em outros contextos semelhantes também é uma direção de pesquisa muito válida. Entretanto, a modelagem da predição de tráfego precisa considerar não apenas o uso de redes mais sofisticadas para extrair recursos, mas também a análise e introdução de dados externos, como o uso de dados proveniente de redes sociais [31]. Ainda mais, devido a similaridade entre o tráfego da rede celular de diferentes regiões da cidade, é possível utilizar uma estratégia conhecida como *transfer learning* para melhorar a reutilização do conhecimento e, consequentemente, reduzir o tempo de treinamento da rede neural. Entretanto, a eficácia dessa técnica para a predição do volume de tráfego da rede é questionável devido a complexidade de encontrar conjuntos de estações bases com padrões de tráfegos similares [55]. A janela futura para minimizar o custo das migrações é avaliada como trabalhos futuros.

Referências Bibliográficas

[1] Alex Anas. *Residential location markets and urban transportation. Economic theory, econometrics and policy analysis with discrete choice models.* Number Monograph. 1982.

[2] Robert Andrews, Joachim Diederich, and Alan B Tickle. Survey and critique of techniques for extracting rules from trained artificial neural networks. *Knowledge-based systems*, 8(6):373–389, 1995.

[3] L Arockiam, S Monikandan, and G Parthasarathy. Cloud computing: a survey. *Int. J. Internet Comput*, 1(2):26–33, 2011.

[4] Oresti Banos, Juan-Manuel Galvez, Miguel Damas, Hector Pomares, and Ignacio Rojas. Window size impact in human activity recognition. *Sensors*, 14(4):6474–6499, 2014.

[5] Gianni Barlacchi, Marco De Nadai, Roberto Larcher, Antonio Casella, Cristiana Chitic, Giovanni Torrisi, Fabrizio Antonelli, Alessandro Vespignani, Alex Pentland, and Bruno Lepri. A multi-source dataset of urban life in the city of milan and the province of trentino. *Scientific data*, 2:150055, 2015.

[6] Jordi Mongay Batallfa, Piotr Krawiec, Constandinos X Mavromoustakis, George Mastorakis, Naveen Chilamkurti, Daniel Negru, Joachim Bruneau-Queyreix, and Eugen Borcoci. Efficient media streaming with collaborative terminals for the smart city environment. *IEEE Communications Magazine*, 55(1):98–104, 2017.

[7] Sourjya Bhaumik, Shoban Preeth Chandrabose, Manjunath Kashyap Jataprolu, Gautam Kumar, Anand Muralidhar, Paul Polakos, Vikram Srinivasan, and Thomas Woo. Cloudiq: A framework for processing base stations in a data center. In *Proceedings of the 18th annual international conference on Mobile computing and networking*, pages 125–136, 2012.

[8] Luiz Bittencourt, Roger Immich, Rizos Sakellariou, Nelson Fonseca, Edmundo Madeira, Marilia Curado, Leandro Villas, Luiz DaSilva, Craig Lee, and Omer Rana. The internet of things, fog and cloud continuum: Integration and challenges. *Internet of Things*, 3-4:134 – 155, 2018.

[9] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008, 2008.

[10] Flavio Bonomi, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli. Fog computing and its role in the internet of things. In *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, pages 13–16. ACM, 2012.

[11] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[12] Rodrigo A C da Silva and Nelson L S da Fonseca. On the location of fog nodes in fog-cloud infrastructures. *Sensors*, 19(11):2445, 2019.

[13] 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.

[14] Gustavo Carneiro. Ns-3: Network simulator 3. In *UTM Lab Meeting April*, volume 20, pages 4–5, 2010.

[15] Hyunseok Chang, Adiseshu Hari, Sarit Mukherjee, and TV Lakshman. Bringing the cloud to the edge. In *Computer Communications Workshops (INFOCOM WKSHPS), 2014 IEEE Conference on*, pages 346–351. IEEE, 2014.

[16] Longbiao Chen, Linjin Liu, Xiaoliang Fan, Johnthan Li, Cheng Wang, Gang Pan, Jérémie Jakubowicz, et al. Complementary base station clustering for cost-effective and energy-efficient cloud-ran. In *2017 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*, pages 1–7. IEEE, 2017.

[17] Min Chen, Yixue Hao, Meikang Qiu, Jeungeun Song, Di Wu, and Iztok Humar. Mobility-aware caching and computation offloading in 5g ultra-dense cellular networks. *Sensors*, 16(7):974, 2016.

[18] Claudio Cicconetti, Marco Conti, and Andrea Passarella. Low-latency distributed computation offloading for pervasive environments. In *2019 IEEE International Conference on Pervasive Computing and Communications (PerCom*, pages 1–10. IEEE, 2019.

[19] Fog Computing. the internet of things: Extend the cloud to where the things are. *Cisco White Paper*, 2015.

[20] Javier Contreras, Rosario Espinola, Francisco J Nogales, and Antonio J Conejo. Arima models to predict next-day electricity prices. *IEEE transactions on power systems*, 18(3):1014–1020, 2003.

[21] R. A. C. da Silva and N. L. S. da Fonseca. Location of fog nodes for reduction of energy consumption of end-user devices. *IEEE Transactions on Green Communications and Networking*, 4(2):593–605, 2020.

[22] Walter Enders. *Applied econometric time series*. John Wiley & Sons, 2008.

[23] Reza Zanjirani Farahani and Masoud Hekmatfar. *Facility location: concepts, models, algorithms and case studies*. Springer, 2009.

[24] Thomas Favale, Francesca Soro, Martino Trevisan, Idilio Drago, and Marco Mellia. Campus traffic and e-learning during covid-19 pandemic. *Computer Networks*, page 107290, 2020.

[25] GMDT Forecast. Cisco visual networking index: global mobile data traffic forecast update, 2017–2022. *Update*, 2017:2022, 2019.

[26] Eduardo S. Gama, Roger Immich, and Luiz F. Bittencourt. Towards a multi-tier fog/cloud architecture for video streaming. In *2018 IEEE/ACM International Conference on Utility and Cloud Computing Companion (UCC Companion)*, pages 13–14, 2018.

[27] Clive William John Granger and Paul Newbold. *Forecasting economic time series*. Academic Press, 2014.

[28] Robert L Grossman. The case for cloud computing. *IT professional*, 11(2):23–27, 2009.

[29] Harshit Gupta, Amir Vahid Dastjerdi, Soumya K Ghosh, and Rajkumar Buyya. ifogsim: A toolkit for modeling and simulation of resource management techniques in the internet of things, edge and fog computing environments. *Software: Practice and Experience*, 47(9):1275–1296, 2017.

[30] Jian He, Di Wu, Yupeng Zeng, Xiaojun Hei, and Yonggang Wen. Toward optimal deployment of cloud-assisted video distribution services. *IEEE transactions on circuits and systems for video technology*, 23(10):1717–1728, 2013.

[31] Jingrui He, Wei Shen, Phani Divakaruni, Laura Wynter, and Rick Lawrence. Improving traffic prediction with tweet semantics. In *Twenty-Third International Joint Conference on Artificial Intelligence*, 2013.

[32] Saiful Hoque, Mathias Santos de Brito, Alexander Willner, Oliver Keil, and Thomas Magedanz. Towards container orchestration in fog computing infrastructures. In *2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC)*, volume 2, pages 294–299. IEEE, 2017.

[33] C. Huang, C. Chiang, and Q. Li. A study of deep learning networks on mobile traffic forecasting. In *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pages 1–6, 2017.

[34] Kenneth J Hunt, D Sbarbaro, R Żbikowski, and Peter J Gawthrop. Neural networks for control systems—a survey. *Automatica*, 28(6):1083–1112, 1992.

[35] R. Immich, E. Cerqueira, and M. Curado. Towards the enhancement of uav video transmission with motion intensity awareness. In *2014 IFIP Wireless Days (WD)*, pages 1–7, Nov 2014.

[36] R. Immich, E. Cerqueira, and M. Curado. Adaptive qoe-driven video transmission over vehicular ad-hoc networks. In *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 227–232, April 2015.

[37] R. Immich, L. Villas, L. Bittencourt, and E. Madeira. Multi-tier edge-to-cloud architecture for adaptive video delivery. In *2019 7th International Conference on Future Internet of Things and Cloud (FiCloud)*, pages 23–30, Aug 2019.

[38] Roger Immich, Eduardo Cerqueira, and Marilia Curado. Efficient high-resolution video delivery over vanets. *Wireless Networks*, Feb 2018.

[39] Jeevan Kharel and Soo Young Shin. Multimedia service utilizing hierarchical fog computing for vehicular networks. *Multimedia Tools and Applications*, pages 1–24, 2018.

[40] Jan Koum and Brian Acton. Multimedia transcoding method and system for mobile devices, April 18 2017. US Patent 9,628,831.

[41] Yiannos Kryftis, George Mastorakis, Constandinos X Mavromoustakis, Jordi Mongay Batalla, Joel JPC Rodrigues, and Ciprian Dobre. Resource usage prediction models for optimal multimedia content provision. *IEEE Systems Journal*, 11(4):2852–2863, 2017.

[42] Tero Lähderanta, Teemu Leppänen, Leena Ruha, Lauri Lovén, Erkki Harjula, Mika Ylianttila, Jukka Riekki, and Mikko J Sillanpää. Edge server placement with capacitated location allocation. *arXiv preprint arXiv:1907.07349*, 2019.

[43] Eirini Liotou, Dimitris Tsolkas, Nikos Passas, and Lazaros Merakos. Quality of experience management in mobile cellular networks: key issues and design challenges. *IEEE Communications Magazine*, 53(7):145–153, 2015.

[44] Juan Liu, Bo Bai, Jun Zhang, and Khaled B Letaief. Cache placement in fog-rans: From centralized to distributed algorithms. *IEEE Transactions on Wireless Communications*, 16(11):7039–7051, 2017.

[45] Weibo Liu, Zidong Wang, Xiaohui Liu, Nianyin Zeng, Yurong Liu, and Fuad E Alsaadi. A survey of deep neural network architectures and their applications. *Neurocomputing*, 234:11–26, 2017.

[46] 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, 2017.

[47] Ping Lu, Quanying Sun, Kaiyue Wu, and Zuqing Zhu. Distributed online hybrid cloud management for profit-driven multimedia cloud computing. *IEEE Transactions on Multimedia*, 17(8):1297–1308, 2015.

[48] Andra Lutu, Diego Perino, Marcelo Bagnulo, Enrique Frias-Martinez, and Javad Khangosstar. A characterization of the covid-19 pandemic impact on a mobile network operator traffic. In *Proceedings of the ACM Internet Measurement Conference*, pages 19–33, 2020.

[49] Redowan Mahmud, Ramamohanarao Kotagiri, and Rajkumar Buyya. Fog computing: A taxonomy, survey and future directions. In *Internet of everything*, pages 103–130. Springer, 2018.

[50] Redowan Mahmud, Satish Narayana Srirama, Kotagiri Ramamohanarao, and Rajkumar Buyya. Quality of experience (qoe)-aware placement of applications in fog computing environments. *Journal of Parallel and Distributed Computing*, 132:190–203, 2019.

[51] Douglas C Montgomery, Cheryl L Jennings, and Murat Kulahci. *Introduction to time series analysis and forecasting*. John Wiley & Sons, 2015.

[52] Attila M Nagy and Vilmos Simon. Survey on traffic prediction in smart cities. *Pervasive and Mobile Computing*, 50:148–163, 2018.

[53] Gurobi Optimization. Inc.,"gurobi optimizer reference manual," 2015, 2014.

[54] 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.

[55] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

[56] Christos H Papadimitriou and Kenneth Steiglitz. *Combinatorial optimization: algorithms and complexity*. Courier Corporation, 1998.

[57] Sai Saketh Nandan Perala, Ioannis Galanis, and Iraklis Anagnostopoulos. Fog computing and efficient resource management in the era of internet-of-video things (iovt). In *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–5. IEEE, 2018.

[58] Flavia Pisani, Fabiola de Oliveira, Eduardo S Gama, Roger Immich, Luiz F Bittencourt, and Edson Borin. Fog computing on constrained devices: Paving the way for the future iot. *Advances in Edge Computing: Massive Parallel Processing and Applications*, 35:22, 2020.

[59] Peter Prettenhofer and Gilles Louppe. Gradient boosted regression trees in scikit-learn. 2014.

[60] Carlo Puliafito, Diogo M Gonçalves, Márcio M Lopes, Leonardo L Martins, Edmundo Madeira, Enzo Mingozzi, Omer Rana, and Luiz F Bittencourt. Mobfogsim: Simulation of mobility and migration for fog computing. *Simulation Modelling Practice and Theory*, 101:102062, 2020.

[61] C. Quadros, E. Cerqueira, A. Neto, A. Riker, R. Immich, and M. Curado. A mobile qoe architecture for heterogeneous multimedia wireless networks. In *2012 IEEE Globecom Workshops*, pages 1057–1061, Dec 2012.

[62] Sepehr Rezvani, Saeedeh Parsaeefard, Nader Mokari, Mohammad R Javan, and Halim Yanikomeroglu. Delivery-aware cooperative joint multi-bitrate video caching and transcoding in 5g. *arXiv preprint arXiv:1805.07132*, 2018.

[63] Denis Rosário, Matias Schimuneck, João Camargo, Jéferson Nobre, Cristiano Both, Juergen Rochol, and Mario Gerla. Service migration from cloud to multi-tier fog nodes for multimedia dissemination with qoe support. *Sensors*, 18(2):329, 2018.

[64] Fillipe Santos, Roger Immich, and Edmundo Madeira. Multimedia microservice placement in hierarchical multi-tier cloud-to-fog networks (submitted). *1st IFIP/IEEE International workshop on Fully-Flexible Internet Architectures and Protocols for the Next-Generation Tactile Internet - FlexNGIA*, 2021.

[65] Alexander Schrijver. *Theory of linear and integer programming*. John Wiley & Sons, 1998.

[66] Tarek R Sheltami, Essa Q Shahra, and Elhadi M Shakshuki. Fog computing: Data streaming services for mobile end-users. *Procedia computer science*, 134:289–296, 2018.

[67] Mluleki Sinqadu and Zelalem Sintayehu Shibeshi. Performance evaluation of a traffic surveillance application using ifogsim. In *International Conference on Wireless Intelligent and Distributed Environment for Communication*, pages 51–64. Springer, 2020.

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

[69] Vitor Barbosa C Souza, Wilson Ramírez, Xavier Masip-Bruin, Eva Marín-Tordera, G Ren, and Ghazal Tashakor. Handling service allocation in combined fog-cloud scenarios. In *2016 IEEE international conference on communications (ICC)*, pages 1–5. IEEE, 2016.

[70] Gang Sun, Victor Chang, Muthu Ramachandran, Zhili Sun, Gangmin Li, Hongfang Yu, and Dan Liao. Efficient location privacy algorithm for internet of things (iot) services and applications. *Journal of Network and Computer Applications*, 89:3–13, 2017.

[71] Tarik Taleb, Marius Corici, Carlos Parada, Almerima Jamakovic, Simone Ruffino, Georgios Karagiannis, and Thomas Magedanz. Ease: Epc as a service to ease mobile core network deployment over cloud. *IEEE Network*, 29(2):78–88, 2015.

[72] 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.

[73] Tarik Taleb, Adlen Ksentini, and Abdellatif Kobbane. Lightweight mobile core networks for machine type communications. *IEEE Access*, 2:1128–1137, 2014.

[74] Shikha Tayal, PK Garg, and Sandip Vijay. Optimization models for selecting base station sites for cellular network planning. In *Applications of Geomatics in Civil Engineering*, pages 637–647. Springer, 2020.

[75] Karima Velasquez, David Perez Abreu, Marcio RM Assis, Carlos Senna, Diego F Aranha, Luiz F Bittencourt, Nuno Laranjeiro, Marilia Curado, Marco Vieira, Edmundo Monteiro, et al. Fog orchestration for the internet of everything: state-of-the-art and research challenges. *Journal of Internet Services and Applications*, 9(1):14, 2018.

[76] Alexander Waibel, Toshiyuki Hanazawa, Geoffrey Hinton, Kiyohiro Shikano, and K Lang. Phoneme recognition: neural networks vs. hidden markov models vs. hidden markov models. In *ICASSP-88., International Conference on Acoustics, Speech, and Signal Processing*, pages 107–108. Citeseer, 1988.

[77] Laurence A Wolsey and George L Nemhauser. *Integer and combinatorial optimization*, volume 55. John Wiley & Sons, 1999.

[78] Jierui Xie and Boleslaw K Szymanski. Community detection using a neighborhood strength driven label propagation algorithm. In *2011 IEEE Network Science Workshop*, pages 188–195. IEEE, 2011.

[79] Hong Yao, Changmin Bai, Deze Zeng, Qingzhong Liang, and Yuanyuan Fan. Migrate or not? exploring virtual machine migration in roadside cloudlet-based vehicular cloud. *Concurrency and Computation: Practice and Experience*, 27(18):5780–5792, 2015.

[80] C. Zhang, H. Zhang, J. Qiao, D. Yuan, and M. Zhang. Deep transfer learning for intelligent cellular traffic prediction based on cross-domain big data. *IEEE Journal on Selected Areas in Communications*, 37(6):1389–1401, 2019.

[81] Chaoyun Zhang and Paul Patras. Long-term mobile traffic forecasting using deep spatio-temporal neural networks. In *Proceedings of the Eighteenth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pages 231–240, 2018.

[82] Chuanting Zhang, Haixia Zhang, Jingping Qiao, Dongfeng Yuan, and Minggao Zhang. Deep transfer learning for intelligent cellular traffic prediction based on cross-domain big data. *IEEE Journal on Selected Areas in Communications*, 37(6):1389–1401, 2019.

[83] Chuanting Zhang, Haixia Zhang, Dongfeng Yuan, and Minggao Zhang. Citywide cellular traffic prediction based on densely connected convolutional neural networks. *IEEE Communications Letters*, 22(8):1656–1659, 2018.

[84] G Peter Zhang. Time series forecasting using a hybrid arima and neural network model. *Neurocomputing*, 50:159–175, 2003.

[85] Zheng-Huan Zhang, Xiao-Feng Jiang, and Hong-Sheng Xi. Optimal content placement and request dispatching for cloud-based video distribution services. *International Journal of Automation and Computing*, 13(6):529–540, 2016.

[86] Wenwu Zhu, Chong Luo, Jianfeng Wang, and Shipeng Li. Multimedia cloud computing. *IEEE Signal Processing Magazine*, 28(3):59–69, 2011.