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

# Explorando Estratégias baseadas em Invariantes de Grafos para o Posicionamento de Servidores de Fog

Palton Lima Alves

Dissertação apresentada como requisito parcial para
conclusão do Mestrado em Informática

Orientador
Prof. Dr. Marcelo Antonio Marotta

Brasília
2025

## Ficha Catalográfica de Teses e Dissertações

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

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

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

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

# Explorando Estratégias baseadas em Invariantes de Grafos para o Posicionamento de Servidores de Fog

Palton Lima Alves

Dissertação apresentada como requisito parcial para
conclusão do Mestrado em Informática

Prof. Dr. Marcelo Antonio Marotta (Orientador)
PPGI/UnB

Prof. Dr. Cristiano Bonato Both
Unisinos

Prof. Dra.Aletéia Patrícia Favacho de Araújo von Paumgartten
PPGI/UnB

Prof.a Dr.a Cláudia Nalon
Coordenadora do Programa de Pós-graduação em Informática

Brasília, 01 de novembro de 2025

# Dedicatória

Dedico este trabalho à minha mãe, que nunca desistiu de me incentivar a estudar e a seguir em frente, mesmo diante das dificuldades. Esta conquista é nossa.

# Agradecimentos

Gostaria de agradecer, antes de tudo, à minha mãe, por sempre me incentivar a valorizar os estudos e a persistir diante das dificuldades.

Agradeço também a todos os professores que fizeram parte da minha trajetória na Universidade de Brasília, desde a graduação até a conclusão desta dissertação de mestrado. Tive a imensa sorte de aprender com profissionais inspiradores, que me ajudaram a crescer muito além do que eu poderia imaginar e me ensinaram a acreditar em objetivos maiores do que eu julgava possíveis.

Expresso meu especial agradecimento ao meu orientador, professor Marcelo Antonio Marotta, pelo apoio constante, pela confiança depositada em mim, mesmo nos momentos em que duvidei de mim mesmo, e pela paciência nas inúmeras vezes em que pensei em desistir. Espero, um dia, poder retribuir sendo para meus futuros alunos a mesma luz que o professor foi em meu caminho.

Agradeço à minha esposa, companheira de todas as horas, pelo amor, pela compreensão e pelo apoio incondicional durante este processo, tantas vezes difícil e desafiador.

Por fim, agradeço a Deus por me conceder forças para seguir em frente. Que eu continue encontrando coragem para realizar o que desejo e transformar as dificuldades em combustível para o meu crescimento.

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

# Resumo

O paradigma de *fog computing* busca reduzir a latência e ampliar a qualidade dos serviços ao aproximar os servidores das aplicações dos usuários finais. Contudo, sua implementação em redes *cloud* envolve custos significativos e requer a escolha eficiente do posicionamento dos nós de *fog*. Esta dissertação propõe e compara diferentes métodos para resolver esse problema, incluindo um modelo exato baseado em programação linear inteira mista (MILP) e heurísticas fundamentadas em invariantes de grafos, como excentricidade e conectividade. As abordagens foram avaliadas em múltiplas topologias reais e sintéticas sob métricas de latência média e número de nós implantados. Os resultados indicam que as heurísticas atingem desempenho próximo ao ótimo com menor custo computacional, oferecendo alternativas viáveis para cenários de aplicações sensíveis à latência e restritas por capacidade.

**Palavras-chave:** *Cloud computing*, *Fog computing*, Posicionamento de *fog*, Latência, Capacidade, Heurísticas, Otimização.

# Abstract

The fog computing paradigm aims to reduce latency and enhance service quality by bringing application servers closer to end users. However, its implementation in cloud-based networks entails significant costs and requires an efficient strategy for positioning fog nodes. This dissertation proposes and compares different methods to address this problem, including an exact model based on Mixed Integer Linear Programming (MILP) and heuristics grounded in graph invariants such as eccentricity and connectivity. The approaches were evaluated on multiple real and synthetic topologies using metrics of average latency and number of deployed nodes. The results show that the heuristic methods achieve performance close to the optimal solution with lower computational cost, providing viable alternatives for latency-sensitive and capacity-constrained application scenarios.

**Keywords:** Cloud computing, Fog computing, Fog placement, Latency, Capacity, Heuristics, Optimization.

# Sumário

# Lista de Figuras

# Lista de Tabelas

# Capítulo 1

# Introdução

*Cloud computing* consolidou-se como um dos modelos mais transformadores da tecnologia da informação, ao viabilizar o fornecimento de recursos computacionais — como processamento, armazenamento e software — sob demanda pela Internet, de forma escalável e independente de infraestrutura local [1]. Diferentemente dos modelos tradicionais, nos quais cada organização precisava adquirir e manter seus próprios servidores e aplicações, a *cloud* permite o acesso remoto a grandes *datacenters* compartilhados, que operam com elasticidade, provisionamento automático e cobrança no modelo *pay-per-use* [2]. Essa abordagem tornou-se o alicerce de inúmeras aplicações modernas, sustentando serviços amplamente utilizados no cotidiano, como redes sociais, e-mails, plataformas bancárias e ambientes colaborativos, além de possibilitar avanços no desenvolvimento de novas tecnologias digitais. As características centrais da *cloud* — elasticidade, alta disponibilidade, atualizações automáticas e baixo custo de entrada — impulsionaram sua adoção em escala global, com crescimento exponencial em diversos setores industriais e acadêmicos [2]. Contudo, justamente por concentrar o processamento em grandes centros de dados, a *cloud* traz consigo limitações intrínsecas de latência e eficiência para aplicações geograficamente dispersas e sensíveis ao tempo, o que motiva a busca por paradigmas complementares, como *fog computing* [3].

Apesar dos avanços da *cloud computing*, sua natureza fortemente centralizada introduz limitações estruturais significativas, sobretudo em termos de latência e congestionamento dos enlaces de *backbone*, que comprometem aplicações sensíveis ao tempo e com elevado volume de dados. Como resposta a essas limitações, surgiu o paradigma de *fog computing*, concebido para estender as funcionalidades da *cloud* até a borda da rede por meio da inserção de nós intermediários próximos aos usuários finais [3]. Essa descentralização permite processar dados e fornecer serviços mais próximos da origem do tráfego, reduzindo os atrasos de comunicação e aliviando o tráfego direcionado aos *datacenters* centrais. Além disso, o *fog computing* possibilita maior capacidade percebida pelos clientes, pois encurta

os caminhos de transmissão e melhora o aproveitamento da largura de banda disponível. Assim, ao aproximar os recursos computacionais da borda da rede, o paradigma *fog* emerge como uma arquitetura complementar à *cloud*, voltada a superar seus limites de latência e escalabilidade e a viabilizar novas classes de aplicações distribuídas, como sistemas industriais de IoT (Internet of Things), veículos conectados e *streaming* em tempo real.

A implementação de nós de *fog* em uma rede *cloud* é um processo que deve então ser pensado de forma planejada e estratégica para solucionar os problemas de distribuição dos serviços com o cumprimento dos requisitos associados às aplicações. Requisitos associados a qualidade de serviço das aplicações atendidas estão relacionados ao fornecimento de baixa latência e alta capacidade. Dessa forma, um modelo de posicionamento de nós de *fog* deverá ter esses dois requisitos como parâmetros para determinação da melhor solução. Esse problema então poderá ser abordado de diferentes formas.

Tomando como base a topologia onde flui o tráfego de dados entre a *cloud* e os clientes, as abordagens podem ser realizadas de diferentes formas. Uma solução seria levar em consideração uma melhor distribuição geográfica dos nós dessa topologia, o que leva a abordagens relacionadas a excentricidade. Uma diferente abordagem poderia considerar os nós mais conectados e ao mesmo tempo excêntricos à *cloud*. No entanto, é possível também determinar a solução ótima para o problema do posicionamento de nós de *fog* com base em uma modelagem analítica.

Para resolver o problema de posicionamento de nós de *fog*, modelos de otimização analíticos buscam soluções ótimas, seja por meio de formulações multiobjetivo que consideram o custo computacional [3] ou por meio de programas não lineares que minimizam os custos explícitos de implantação [4]. Embora esses modelos apresentem bons resultados, seu custo computacional proibitivo muitas vezes impede sua aplicabilidade em topologias de grande escala. Por outro lado, abordagens heurísticas fornecem soluções escaláveis, geralmente baseadas em algoritmos de agrupamento. Essa escalabilidade, no entanto, é frequentemente alcançada à custa da simplificação excessiva do problema, seja priorizando a otimização da latência em detrimento dos custos diretos de implantação [5, 6], seja tratando o dimensionamento e o posicionamento como etapas sequenciais e desacopladas [7]. Como resultado, mesmo em estudos que comparam soluções ótimas e heurísticas [6], uma estrutura que aborda a minimização dos custos de implantação (representados pelo número de nós) como um objetivo central sob restrições simultâneas de latência e capacidade permanece uma lacuna de pesquisa.

Este trabalho tem como objetivo avaliar a viabilidade de métodos heurísticos baseados em invariantes de grafos para equilibrar, de forma eficiente, a latência, a capacidade e o custo de implantação de servidores fog em uma infraestrutura cloud. Mais especificamente, investiga-se em que medida heurísticas fundamentadas em conectividade e

excentricidade são capazes de aproximar a solução ótima, analisando seu desempenho como alternativas práticas aos modelos ótimos baseados em MILP, cuja obtenção pode demandar elevado tempo de processamento e que, em certos casos, sequer garantem uma solução viável para todas as instâncias do problema. Por meio de experimentos em três topologias de rede representativas (NSFNET [8], GEANT2 [9] e SYNTH50BW[10]) e quatro cenários de aplicação (IoT Industrial - IIoT, Streaming de Vídeo 4K, Jogos Online e Videoconferência HD), realiza-se uma comparação abrangente entre soluções ótimas e heurísticas. Os resultados mostram que a heurística de excentricidade baseada em latência atinge um desempenho quase ótimo, igualando a solução ótima baseada em uma modelagem Programação Linear Inteira Mista (*Mixed-Integer Linear Programming* - MILP), para 11 dos 12 cenários (91,7% dos casos). A análise revela que a heurística mantém uma diferença de custo médio robusta de apenas 1,7% e uma diferença de latência média de apenas 4,5% em comparação com a linha de base ideal, alcançando essa qualidade com uma fração do custo computacional. Essas descobertas demonstram que heurísticas baseadas em invariantes de grafo podem fornecer soluções escaláveis, econômicas e de alto desempenho para o planejamento de infraestrutura de cenários de *fog computing*.

Para apresentar esta pesquisa e os resultados atingidos, esta dissertação foi organizada, além deste capítulo introdutório, em mais cinco capítulos, os quais são:

- No Capítulo 2, apresenta-se uma revisão sistemática e comparativa do posicionamento de nós de *fog* em arquiteturas *cloud*/borda, organizada por domínios de aplicação, classes de métodos e critérios técnico-operacionais. São examinados modelos exatos e formulações multiobjetivo que tratam, de forma conjunta, a latência e o tráfego para a *cloud*; estudos de migração e controle em SDN, com decisões incrementais de posicionamento e seus efeitos ao longo do tempo; e trabalhos que integram *fog* e CDN por meio de MINLP, com decisão conjunta de localização e entrega de conteúdo. A revisão abrange ainda abordagens voltadas a IoT/IIoT e SDN, baseadas em MILP e heurísticas. Tabelas de síntese reúnem o escopo, as métricas avaliadas e o repertório metodológico predominante. A partir desse panorama, evidencia-se a lacuna que motiva esta dissertação: a necessidade de uma avaliação unificada que compare soluções ótimas (MILP) e meta-heurísticas baseadas em invariantes de grafo sob requisitos simultâneos de latência e capacidade, minimizando o número de nós de *fog* e caracterizando o *trade-off* por meio da frente de Pareto em múltiplas topologias (reais e sintéticas) e contextos (IoT, CDN e jogos online).

- No Capítulo 3, é apresentada a formulação completa do sistema para o posicionamento ótimo de nós de *fog* e as heurísticas baseadas em invariantes de grafos. O texto inicia pela parametrização do modelo (notações, matrizes de conectividade, latência

e capacidade, requisitos por nó e vínculos *cloud–fog*), segue para as variáveis de decisão que codificam a escolha de locais candidatos e a associação cliente–servidor, e então detalha as restrições que asseguram consistência lógica e operacional (atendimento exclusivo, limites de latência e de capacidade, e condições específicas do enlace *cloud–fog*). Em seguida, é apresentada a função multiobjetivo que equilibra dois alvos conflitantes, minimizar o número de nós de *fog* e a latência média da rede, por meio de um parâmetro de ponderação $\alpha$ e de fatores de normalização que tornam comparáveis as escalas dos termos e preservam a linearidade (MILP). O capítulo conclui com os métodos heurísticos, sendo apresentado um fluxograma que descreve o ciclo de seleção incremental de nós de *fog* (elegibilidade *cloud–fog*, cálculo de pesos e atualização do atendimento), e detalha as regras de escolha fundamentadas em excentricidade e conectividade;

- No Capítulo 4, é delineada a arquitetura geral da solução e o plano de investigação: são definidos o objetivo e as contribuições—comparar, em cenários representativos (IIoT, Videoconferência em HD, *streaming* 4K e jogos *on-line*), heurísticas baseadas em invariantes de grafos frente à solução ótima (MILP)—e é apresentada a visão operacional do método, em que o posicionamento incremental de nós de *fog* busca restaurar requisitos de latência e capacidade quando a *cloud* centralizada se mostra insuficiente. É descrito o protótipo experimental e são detalhados os elementos de reprodutibilidade (código, *datasets*, pós-processamento), com especificação das topologias e do procedimento de extração das matrizes de latência e capacidade. Além disso, são estabelecidos os critérios de avaliação e é apresentada a construção da frente de Pareto para mapear o *trade-off* entre número de nós de *fog* e latência média, e permitir a comparação direta de cada heurística com a fronteira eficiente do MILP. Por fim, são explicitadas limitações práticas (dependência das condições de rede, possíveis soluções inviáveis e impacto na estimativa de latência média);

- No Capítulo 5, é apresentada a avaliação empírica dos métodos nas três topologias e nas quatro aplicações, com métricas centrais de número de nós de *fog* e latência média. É descrita a montagem experimental—incluindo tempos de execução por método, repetição de ensaios e intervalos de confiança de 90%—e é estabelecido o procedimento de estimação por médias a partir de 320 repetições por configuração, após análise de convergência do erro. São reportadas curvas de Pareto que ancoram a comparação direta entre heurísticas e solução ótima (MILP), destacando, por cenário e topologia, ganhos e custos marginais ao longo do *trade-off* infraestrutura-desempenho. É discutido o comportamento por aplicação (IIoT, 4K, jogos e HD), com sínteses de quais heurísticas se aproximam do joelho da curva e em que condições

a capacidade deixa de ser fator limitante. Também são apresentados experimentos de sensibilidade (variação de requisitos de latência e capacidade) e superfícies de solução que evidenciam a predominância da latência nas decisões de posicionamento. Por fim, são expostos limites práticos — incluindo dispersão de resultados e casos de inviabilidade por restrições *cloud–fog* — e são indicadas implicações operacionais quando a simples adição de nós de *fog* não assegura o atendimento dos requisitos;

- No Capítulo 6, a conclusão e trabalhos futuros são apresentados como uma síntese das principais constatações sobre as heurísticas avaliadas, indicando, de modo objetivo, os contextos de melhor adequação de cada método e os respectivos benefícios no *trade-off* entre reduzir o número de nós de *fog* e diminuir a latência média da rede. Além disso, são explicitadas também as limitações observadas nas soluções e nos *datasets*, destacando-se como essas restrições podem comprometer análises mais aprofundadas de latência e capacidade e, em certos cenários, inviabilizar soluções. Encerrando, são delineadas direções de pesquisa para o aperfeiçoamento dos métodos propostos.

# Capítulo 2

# Trabalhos Relacionados

No início deste capítulo, a análise da literatura sobre o posicionamento de nós de *fog* em redes é apresentada. Foi realizado um levantamento das principais abordagens desenvolvidas para resolver esse problema. Por fim, a lacuna de pesquisa é apresentada por meio de uma tabela, e discussões a respeito da mesma são tecidas.

No artigo de [3], os autores propõem um modelo matemático para o planejamento e o *design* de redes com *fog*, determinando simultaneamente a posição, a capacidade e a quantidade de nós, bem como a interconexão desses nós com a *cloud*. O objetivo é minimizar a latência média da rede e o tráfego encaminhado à *cloud*. Para lidar com o caráter multiobjetivo do problema, o estudo compara três estratégias: a técnica de soma de pesos, que agrega os objetivos por meio de vetores de ponderação; o método hierárquico, que estabelece uma ordem de prioridades entre os objetivos; e a abordagem de *trade-off*, que otimiza um objetivo enquanto trata os demais como restrições. Essa formulação sustenta a análise comparativa desenvolvida na sequência.

O problema atacado por [3] é central em aplicações sensíveis à latência em *fog*/*edge*, nas quais o desenho da topologia impacta diretamente latência média e descarga de tráfego na *cloud*. Em vez de discutir a literatura de forma ampla, o artigo oferece um comparativo entre diferentes modelos de otimização para o mesmo problema, permitindo observar como escolhas de modelagem e de método influenciam as métricas-alvo. A contribuição central do artigo consiste na avaliação multiobjetivo realizada sob condições controladas. Ao aplicar três estratégias, soma de pesos, método hierárquico e abordagem de *trade-off*, a um mesmo modelo, os autores demonstram que a escolha metodológica induz soluções com ótimos distintos para latência e volume de tráfego, além de custos computacionais diferenciados (medidos em tempo de CPU). O estudo, assim, delineia um quadro de compromissos entre latência, tráfego e custo computacional específico a cada técnica, oferecendo base criteriosa para a seleção do método mais adequado às prioridades do sistema.

O artigo [3] guarda pertinência direta com este trabalho ao fornecer um referencial comparativo entre técnicas voltadas ao mesmo problema decisório. Embora nesta pesquisa sejam também exploradas meta-heurísticas, serão adotados critérios de avaliação compatíveis com os de [3], tais como latência média e capacidade, a fim de assegurar a comparabilidade entre soluções ótimas e heurísticas. Desse modo, os resultados desta investigação poderão ser interpretados nos mesmos eixos de compromisso destacados em [3], permitindo uma análise alinhada e cumulativa.

No artigo de [11], os autores abordam o problema de posicionamento dinâmico de serviços em infraestruturas de *fog computing* voltadas a aplicações em tempo real. O trabalho formula o *Service Placement Problem* como um problema de otimização multiobjetivo, no qual se busca simultaneamente minimizar o tempo de resposta das aplicações e maximizar a utilização dos recursos computacionais disponíveis nos nós de fog. Para isso, é apresentada uma modelagem analítica baseada em Programação Linear Inteira Mista (MILP), que considera restrições de *CPU*, memória e prazos de execução (*deadlines*) dos serviços. A partir dessa formulação, os autores propõem um algoritmo heurístico denominado *Multi-objective Dynamic Service Placement* (moDSP), capaz de tomar decisões de alocação em tempo real, selecionando dinamicamente o nó mais adequado com base nos objetivos conflitantes definidos.

A relevância do trabalho de [11] decorre do foco explícito em aplicações sensíveis à latência no contexto de *IoT*, nas quais decisões estáticas de posicionamento tendem a ser ineficientes frente à dinâmica da demanda e à heterogeneidade dos recursos de fog. Diferentemente de abordagens que tratam isoladamente métricas como latência ou utilização de recursos, o estudo integra ambos os critérios em uma mesma função de decisão, evidenciando o *trade-off* inerente entre qualidade de serviço e eficiência do uso da infraestrutura. A avaliação experimental, conduzida por meio do simulador *iFogSim*, demonstra que a estratégia proposta supera métodos de referência ao reduzir o tempo de resposta médio e melhorar o balanceamento de carga entre os nós, sem incorrer no elevado custo computacional associado à resolução exata do modelo MILP. Assim, o artigo contribui ao mostrar que heurísticas multiobjetivo bem fundamentadas podem oferecer ganhos práticos relevantes em cenários dinâmicos.

O trabalho de [11] relaciona-se diretamente com esta dissertação ao explorar uma formulação multiobjetivo que combina latência e capacidade como critérios centrais de decisão. Embora o foco do artigo esteja no posicionamento dinâmico de serviços, e não no planejamento estrutural do posicionamento de nós de fog, ambos compartilham a preocupação em caracterizar e analisar os compromissos entre desempenho e uso eficiente dos recursos. Nesta pesquisa, métricas como latência média e capacidade também são adotadas como eixos de avaliação, o que permite estabelecer paralelos conceituais entre

os resultados obtidos por abordagens heurísticas e soluções ótimas baseadas em MILP. Dessa forma, o artigo fornece um referencial metodológico complementar, especialmente no que diz respeito à análise de *trade-offs* multiobjetivo em ambientes de *fog computing*.

No estudo [10], os autores propõem um modelo analítico para orientar a migração de redes legadas para redes definidas por software (do inglês, Software-Defined Network), com ênfase na localização ótima de controladores ao longo de uma implementação necessariamente gradual. O modelo é confrontado com soluções subótimas fundamentadas em invariantes de grafos, notadamente excentricidade e conectividade dos nós, o que permite representar o problema de maneira abrangente e comparar, em cenários de transição, estratégias de posicionamento que equilibram custo computacional e qualidade da solução.

A relevância dessa agenda decorre do fato de que, embora SDN ofereça ganhos expressivos de flexibilidade e desempenho nas arquiteturas contemporâneas, os custos de adoção plena frequentemente inviabilizam migrações instantâneas em ambientes legados. Por isso, a literatura tem convergido para rotas incrementais de migração, buscando decisões progressivas de posicionamento de controladores que preservem desempenho e disponibilidade ao longo do processo. Nessa linha, [10] distingue-se por tratar o planejamento como um problema de otimização voltado ao estado final plenamente SDN, isto é, por priorizar a qualidade global da solução ao término da transição.

Do ponto de vista metodológico, grande parte dos trabalhos correlatos concentra-se em otimizar métricas específicas (custo, latência, resiliência) em etapas pontuais da transição [10], sem modelar explicitamente as interdependências entre tempo, quantidade e localização de controladores. Em contraste, a contribuição central de [10] está em incorporar a dimensão temporal ao planejamento, isto é, ao discretizar a evolução da rede, o modelo explicita os *trade-offs* entre o momento de implantação, o número de controladores e suas posições, sob restrições operacionais (latência controlador–*switch*, carga de controle). Com isso, torna-se possível perseguir uma solução globalmente ótima mesmo quando o percurso de evolução não é totalmente previsível.

Ancorado nesse arcabouço, este trabalho de dissertação adota um desenho experimental análogo para o problema de posicionamento de nós de *fog* desenvolvido em [10]. Neste trabalho serão comparadas soluções otimizadas e heurísticas baseadas em invariantes de grafos, como o nó mais conectado (do inglês, Most Connected Node) e o de maior excentricidade (do inglês, Highest Eccentricity Node), de modo a capturar tanto cenários planejados quanto decisões induzidas por políticas práticas. O objetivo é mapear os *trade-offs* entre posicionamento e requisitos de rede (latência e capacidade) em distintas topologias, produzindo evidências sobre a eficiência operacional e a viabilidade de implementação de cada abordagem.

No estudo [4], os autores formulam o problema de posicionamento de nós de *fog* para

distribuição de conteúdos como um programa inteiro não linear que integra *fog computing* e redes de distribuição de conteúdo (do inglês, Content Delivery Network). A função objetivo minimiza o custo total de implantação dos nós e de entrega dos conteúdos, sujeita a restrições de Qualidade de Serviço (do inglês, Quality of Service); a resolução é realizada pelo algoritmo JFnP-CDA[4], concebido para decidir conjuntamente onde posicionar nós de *fog* e como alocar réplicas de conteúdo. A avaliação empírica, baseada em dados reais de pontos de acesso Wi-Fi (OPWAP) [4] em regiões como Delhi e Nova York, considera dimensões como tamanho de réplicas, frequência de atualização e atraso, evidenciando reduções de custo acompanhadas de melhorias de QoS em cenários realistas.

A motivação para tal abordagem decorre de limitações estruturais das arquiteturas tradicionais de CDN, ancoradas em servidores de borda e de origem [12–14]. Em aplicações sensíveis à latência, como o *streaming* de vídeo, a distância geográfica e o congestionamento dos enlaces tornam soluções exclusivamente em *cloud* insuficientes; mesmo estratégias *multicloud*, embora ampliem disponibilidade, não resolvem plenamente a latência [15]. Nesse quadro, o *fog computing* oferece um caminho promissor ao deslocar capacidade computacional para pontos intermediários próximos ao usuário final, reduzindo atraso e elevando QoS [16–18]. O ponto crítico, contudo, é o custo da implantação e o desenho ótimo do posicionamento—desafio diretamente enfrentado por [4] ao priorizar, na função objetivo, a relação custo-efetividade sem sacrificar escalabilidade e QoS.

No panorama da literatura, há propostas que tratam o posicionamento de nós de *fog* sob diferentes prismas de heurísticas em contextos SDN [6] a modelagens voltadas a fábricas inteligentes [19], mas ainda são relativamente escassas as contribuições que atacam o problema no contexto específico de CDNs, no qual custo, disponibilidade e QoS interagem de modo particularmente estreito. Nesse sentido, Yadav et al. (2024) [4] se destacam ao demonstrar, por meio de avaliações com dados do mundo real, que seu algoritmo supera *baselines* em custo-efetividade, ao mesmo tempo em que oferece garantias práticas de QoS e de escalabilidade.

Ancorado nesse referencial, este trabalho dialoga diretamente com [4], mas desloca o foco de otimização, ou seja, em vez de internalizar explicitamente os custos na função objetivo, modelou-se esses efeitos indiretamente ao otimizar o número de nós de *fog* necessários para satisfazer requisitos de latência e de capacidade. Em termos metodológicos, isso se traduz em comparar soluções ótimas da modelagem proposta com meta-heurísticas baseadas em invariantes de grafos, preservando o vínculo com cenários práticos e políticas de decisão. Assim, enquanto [4] explicita custos e disponibilidade como variáveis de decisão e termos de QoS, a formulação apresentada nesta dissertação, absorve essas dimensões por meio de restrições operacionais (capacidade e latência), resultando em um modelo mais parcimonioso, com menos variáveis e, potencialmente, mais generalizável a

distintas topologias e contextos de implantação.

O trabalho [6] aborda o problema de posicionamento de nós de *fog* (FNPP) no contexto de aplicações intensivas de IoT em cenários de redes definidas por software (SDN). A proposta inclui a formalização do problema como um modelo de programação linear inteira mista (MILP) e a implementação de soluções baseadas tanto em algoritmos heurísticos quanto em métodos exatos. O foco principal é otimizar a latência e a QoS, considerando restrições de capacidade dos nós de *fog* e da infraestrutura de rede. Os autores comparam suas soluções com outros trabalhos semelhantes, demonstrando a eficiência da abordagem em diferentes configurações de rede, incluindo cenários de SDN e IIoT.

A motivação de Herrera et al. [6] decorre das exigências estritas de latência e QoS em cenários industriais e de grande escala, típicos do ecossistema IoT [20]. Abordagens que dependem exclusivamente de processamento centralizado ou de *cloud* tendem a sucumbir à alta densidade de dispositivos e à limitação de recursos, razão pela qual a integração entre SDN e *fog* se apresenta como via de orquestração eficiente na borda. Nesse enquadramento, o estudo busca o equilíbrio entre viabilidade prática e eficiência: reduzir atrasos fim a fim, acomodar variações de demanda e respeitar capacidades de enlaces e nós, sem comprometer a escalabilidade operacional.

Os resultados empíricos reportados por Herrera et al. [6] mostram que a heurística proposta atinge latências muito próximas às soluções ótimas, com tempos de execução substancialmente menores que os de abordagens concorrentes. A avaliação, conduzida em múltiplas topologias, incluindo cenários SDN e IIoT, considera capacidade de *links*, distribuição de tráfego e escalabilidade, evidenciando cumprimento de requisitos estritos de QoS mesmo em ambientes complexos e restritivos. Com isso, o artigo consolida um arcabouço metodológico que combina desempenho e custo computacional, habilitando aplicações IoT quase em tempo real com maior previsibilidade.

O trabalho de Herrera et al. [6] apresenta um estudo relacionado ao problema de posicionamento de nós de *fog*, com foco na otimização da latência e na comparação entre soluções ótimas e meta-heurísticas. A pesquisa se concentra no contexto de redes SDN e IoT, avaliando a latência média da rede e o tempo necessário para determinar as soluções propostas por diferentes métodos. Por outro lado, o estudo desenvolvido neste trabalho adota uma abordagem distinta ao otimizar o número de nós de *fog*, considerando restrições de latência e capacidade. Isso permite uma análise mais ampla e precisa dos requisitos de aplicações reais, já que a capacidade dos nós é incorporada como uma restrição central no problema de posicionamento. Além disso, o trabalho de Herrera et al [6] contribui com levantamentos importantes para avaliar o desempenho de soluções subótimas em comparação com as soluções ótimas. A proposta de análise baseada na latência média dos dispositivos é algo que pode ser estendido como parâmetro de comparação entre

as diferentes abordagens, tanto para latência média quanto para capacidade média. Esse aspecto é relevante, pois, além da redução dos custos de implementação e cumprimento de requisitos, as soluções encontradas podem ser avaliadas também em relação às melhorias globais de desempenho da rede.

No artigo de [21], os autores investigam o problema de posicionamento de aplicações em ambientes *cloud–fog*, propondo uma política de decisão baseada em técnicas de *Multi-Criteria Decision Making* (MCDM). O trabalho combina o *Analytic Hierarchy Process* (AHP) e o método *Technique for Order of Preference by Similarity to Ideal Solution* (TOPSIS) para selecionar os nós de fog mais adequados à execução de módulos de aplicações *IoT*. A abordagem considera múltiplos critérios heterogêneos, incluindo capacidade de CPU, memória, largura de banda de subida e descida, e latência, cujos pesos relativos são determinados via AHP e posteriormente utilizados pelo TOPSIS para ranquear os nós candidatos. A política proposta é avaliada por meio de simulações no *iFogSim*, em um cenário de aplicações sensíveis a atraso, demonstrando melhorias no tempo de posicionamento e na latência *end-to-end*.

A importância do trabalho de [21] reside na adoção explícita de um arcabouço multicritério para lidar com a heterogeneidade dos recursos e requisitos em infraestruturas de *fog computing*. Em contraste com modelos de otimização exata ou heurísticas focadas em uma única métrica, o estudo enfatiza o processo decisório como um problema de ordenação e escolha entre alternativas viáveis, incorporando critérios potencialmente conflitantes de forma estruturada. Os resultados experimentais indicam que a política AHP–TOPSIS (A-TAP) apresenta desempenho comparável ou superior a abordagens baseadas em lógica *fuzzy* e em versões modificadas do TOPSIS, sobretudo em termos de tempo de posicionamento e manutenção da latência em níveis aceitáveis à medida que a carga de aplicações cresce. Assim, o artigo evidencia que métodos MCDM podem oferecer soluções eficazes e computacionalmente leves para ambientes dinâmicos e distribuídos.

O trabalho de [21] relaciona-se com esta dissertação ao compartilhar o interesse na análise de *trade-offs* entre latência e capacidade em decisões de posicionamento no contexto de *fog computing*. Embora o foco do artigo esteja no posicionamento de módulos de aplicações, e não no planejamento estrutural do número e da localização dos nós de fog, ambos convergem na utilização de métricas de latência e disponibilidade de recursos como critérios centrais de avaliação. Nesta dissertação, esses mesmos parâmetros são empregados dentro de uma formulação *multiobjective* e na comparação entre soluções ótimas e heurísticas, permitindo que os resultados sejam analisados sob eixos conceituais compatíveis. Dessa forma, o artigo fornece um referencial complementar, especialmente no que diz respeito ao uso de métodos multicritério como alternativa prática às formulações de otimização clássicas em cenários de *fog computing*.

No artigo [5], os autores tomam *gateways* como candidatos a nós de *fog* e os "elevam" ao o que denominam como *Fog Smart Gateways* (FSG) para processar tráfego IoT com VMs na borda. O problema é modelado em grafo e formalizado para, dado um número $f$ de nós de *fog*, minimizar a latência total da rede. A busca considera combinações de posicionamento e emprega variantes de *k-means* para selecionar posições iniciais e refinar centróides até a convergência, escolhendo cada centróide final como local de implantação do nó de *fog*. Em síntese, o trabalho busca otimizar o número de nós de *fog* implantados em *gateways* de forma que minimize a latência total de toda a rede e compara os resultados aos modelos exclusivamente baseados em *cloud*.

O trabalho de Maiti et al. (2018)[5] mostra-se particularmente oportuno na medida em que, em aplicações sensíveis a atraso, as arquiteturas centradas em *cloud* enfrentam problemas de desempenho; ao relocar capacidade para a borda, o *fog* reduz o trajeto dos dados e o processamento intermediário, com ganhos de QoS. O artigo explicita um arcabouço em camadas e define métricas de QoS, em particular a decomposição do atraso de serviço em componentes de transmissão e de processamento ao longo do caminho CD $\rightarrow$ SSGW $\rightarrow$ IGW $\rightarrow$ FSG, estabelecendo a base para avaliar o impacto do posicionamento na experiência fim a fim.

Nos experimentos com o simulador iFogSim [22], os autores variam de 32 a 512 *gateways* e mostram forte queda de latência até cerca de 6 nós de *fog*, com benefícios marginais a partir desse número; o *random placement* é claramente inferior, e as melhores inicializações para $k$-means são *Mid Point* e *Partition Mean*. O estudo, portanto, fornece um procedimento reprodutível (algoritmos e critérios) e uma regra prática de dimensionamento ($\approx$ 6 nós como ponto de rendimentos decrescentes) para guiar decisões em topologias IoT densas.

Em consonância com esse artigo, este trabalho compartilha o objetivo fundamental de minimizar o número de nós de *fog* na rede, simultaneamente à redução da latência agregada. Diverge, contudo, nos critérios de restrição: enquanto o artigo [5] considera exclusivamente a latência, esta dissertação incorpora, de forma conjunta, latência e capacidade. No que tange ao contexto de aplicação, o artigo pressupõe o posicionamento de nós de *fog* em *gateways*, limitando a um cenário típico de IoT com alta densidade de dispositivos. Já esta dissertação investiga topologias geograficamente mais extensas e menos densas, buscando avaliar os métodos em múltiplos contextos de aplicação.

No artigo [7], os autores enfrentam o problema de dimensionamento e posicionamento de nós de *fog* em redes *Device-to-Device* (D2D) [23]. O arcabouço metodológico procede em três etapas: estima a carga de requisições nos *clusters* D2D por estação-base; dimensiona o número de nós de *fog* necessário para satisfazer um limiar de qualidade de serviço por meio de filas M/M/m (tempo de resposta abaixo de um *threshold*); e posiciona os nós

dimensionados em estações-base reais utilizando *K-medoids*, minimizando a distância de acesso entre *clusters* D2D e os pontos de processamento na borda, tomada como *proxy* de latência.

Aplicações sensíveis a atraso dependem de capacidade computacional próxima ao usuário final, de modo que tanto a quantidade de nós de *fog* quanto sua localização impactam diretamente o atraso de acesso e o cumprimento de requisitos de QoS. Ao integrar, em um único processo, o dimensionamento orientado por métricas de desempenho (via M/M/m) e a alocação geográfica sobre pontos reais da infraestrutura (via *K-medoids*), o estudo oferece um procedimento reprodutível para o projeto de redes *fog* em cenários densos de D2D, com potencial de reduzir tráfego encaminhado à *cloud* e melhorar a experiência fim a fim.

As principais contribuições incluem: a formulação explícita do objetivo de minimizar a distância de acesso sujeito a restrições estruturais de associação entre estações-base e nós *fog*; o uso de um modelo M/M/m para determinar o número de servidores na borda a partir de metas de tempo de resposta (incluindo a expressão de Erlang C); e a adoção de *K-medoids* para garantir que os medoides coincidam com locais efetivamente implantáveis. Na avaliação empírica com um conjunto de 125 estações-base (região central de Melbourne), observa-se que o aumento do número de nós de *fog* reduz o tempo de resposta e que o *K-medoids* supera uma implantação aleatória ao diminuir as distâncias médias e evitar regiões descobertas; como ilustração, para uma taxa agregada de chegada de 126 requisições por milissegundo, aproximadamente quatro nós atendem um limiar de 4,0 ms de tempo de resposta.

Este estudo dialoga diretamente com esta dissertação ao compartilhar o objetivo de reduzir latência por meio do posicionamento de nós de *fog*, mas difere na formulação do problema e no escopo experimental. Enquanto Hamadani e Borcoci (2021) [7] dimensionam o número de servidores a partir de metas de tempo de resposta e otimizam distâncias de acesso em cenários D2D densos, este trabalho de dissertação internaliza simultaneamente restrições de latência e de capacidade, e busca minimizar o número de nós sujeitos a esses requisitos.

A Tabela 2.1 oferece uma visão comparativa das linhas de pesquisa que se aproximam deste estudo, ela está organizada por eixos que permitem uma visão direta sobre os temas comuns de todos os trabalhos selecionados e este trabalho de dissertação. A tabela se organiza primeiramente sobre o domínio de aplicação (fog, integração com CDNs, presença de SDN e aplicação em IoT), depois por classe de métodos (modelagem exata e heurísticas/meta-heurísticas), estrutura de objetivos (multiobjetivo ou não), os critérios técnicos adotados como métricas e/ou restrições (latência, capacidade, tráfego para núvem e custo), as características operacionais (centralização e descentralização) e o tipo de

validação predominante (dados reais ou uso de simuladores/ferramentas).

Analisando os domínios de aplicação, observa-se que grande parte dos estudos levantados explora o uso de *fog computing* no contexto de Internet das Coisas (IoT), e todos os trabalhos que integram *fog* e IoT consideram a métrica de latência como parâmetro central de avaliação [3, 4, 6, 7]. Embora muitos desses estudos também utilizem a capacidade como métrica ou requisito complementar para os métodos propostos, a latência se mostra predominante, sobretudo devido ao foco das pesquisas em integrar *fog* e IoT com o objetivo de reduzir o tempo de resposta da rede. Isso ocorre porque, em aplicações IoT, a principal motivação para o emprego de serviços de *fog* é justamente a diminuição da latência na comunicação entre dispositivos e servidores. Nessas arquiteturas, os dados gerados pelos dispositivos são processados mais próximos da borda da rede, o que acelera a obtenção das respostas quando comparado ao tempo de envio e retorno das informações para a *cloud* centralizada [6].

Observando a relação entre os métodos, nota-se que a maioria dos estudos analisados adota abordagens baseadas em heurísticas ou meta-heurísticas, enquanto apenas uma parcela menor utiliza métodos exatos como referência comparativa para avaliação de desempenho. Além disso, verifica-se que grande parte das pesquisas faz uso de ferramentas ou simuladores para a geração de dados de teste, em vez de empregar dados provenientes de redes reais. Essa opção pelo uso de dados simulados, muitas vezes inspirados em topologias reais, confere maior flexibilidade aos experimentos, permitindo a criação de cenários mais desafiadores e de topologias mais densas, de modo a avaliar o comportamento e os limites de desempenho dos modelos propostos.

A Tabela 2.2 sintetiza os principais métodos e técnicas empregados nos trabalhos analisados, permitindo visualizar as abordagens predominantes em cada estudo. Observa-se que alguns métodos se repetem entre diferentes pesquisas, destacando-se os modelos baseados em programação linear inteira mista (do inglês, Mixed-Integer Linear Programming) e os algoritmos de agrupamento *k-means* e *k-medoids*.

Em síntese, mesmo em estudos que comparam soluções ótimas e heurísticas [6], um modelo que aborde a minimização dos custos de implantação (representados pelo número de nós) como um objetivo central sob restrições simultâneas de latência e capacidade, com bom desempenho computacional e boa convergência, permanece sendo uma lacuna de pesquisa. Neste trabalho, propõe-se a utilização de modelos exatos baseados em programação linear inteira mista, e técnicas multiobjetivo com enfoque no balanço entre minimizar o número de nós de *fog* da rede versus a redução do atraso, sem comprometer a capacidade necessária para diversas aplicações. A partir dos resultados deste modelo, comparam-se métodos heurísticos fundamentados em invariantes de grafos para alcançar melhores desempenhos de processamento sem comprometer a qualidade dos resultados do

Tabela 2.1: Comparação dos trabalhos relacionados.

| **Trabalho (título completo e ano)** | **Fog** | **CDN** | **SDN** | **IoT** | **Modelagem exata** | **Heurísticas / Meta-heurísticas** | **Multiobjetivo** | **Latência (méd/máx/restr.)** | **Tráfego para *cloud*** | **Capacidade (nó/link)** | **Custo** | **Descentralizado** | **Validação (dados reais)** | **Ferramenta / Simulador** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| On the Planning and Design Problem of *fog* Computing Networks [3] | X | | | X | X | | X | X | X | X | X | | | X |
| On the Transition of Legacy Networks to SDN — An Analysis on the Impact of Deployment Time, Number, and Location of Controllers [10] | | | X | | X | X | | X | | X | | | | X |
| Efficient Content Distribution in Fog-Based CDN: A Joint Optimization Algorithm for *fog*-Node Placement and Content Delivery [4] | X | X | | | X | X | | X | | | X | | X | X |
| QoS-Aware *fog* Node Placement for Intensive IoT Applications in SDN-Fog Scenarios [6] | X | | X | X | X | X | | X | | X | | | | X |
| QoS-aware *fog* Nodes Placement [5] | X | | | X | | X | | X | | | | | | X |
| Fog Nodes Placement for D2D Networks [7] | X | | | X | | X | | X | | X | | | X | X |
| Multi-objective Optimization for Dynamic Service Placement Strategy for Real-Time Applications in *Fog* Infrastructure (2023) [11] | X | | | X | X | X | X | X | | X | | X | | X |
| Enhancing Application Placement in *Cloud-Fog* Environment Based on Multicriteria Decision Making (2024) [21] | X | | | X | | X | X | X | | X | | | | X |
| Este trabalho | X | X | | X | X | X | X | X | | X | X | | X | X |

planejamento. Esta melhora é identificada comparando-se soluções ótimas com aquelas derivadas de estratégias heurísticas utilizando invariantes de grafo.

No que se refere às aplicações, o trabalho contempla um espectro abrangente de cenários, incluindo redes de distribuição de conteúdo (CDNs), Internet das Coisas (IoT) e jogos *online*. Quanto aos dados utilizados, a abordagem combina topologias reais e uma topologia sintética de alta densidade de nós, possibilitando avaliar o desempenho tanto em contextos práticos quanto em ambientes simulados e mais desafiadores. Dessa forma, o estudo contribui para preencher uma lacuna na literatura ao abordar o problema de

Tabela 2.2: Síntese dos métodos/técnicas utilizados em cada trabalho.

| **Método / Técnica** | **Haider 2021** | **Pontes 2021** | **Yadav & Kar 2024** | **Herrera 2022** | **Maiti** | **Hamadani** | **Este trabalho** |
|---|---|---|---|---|---|---|---|
| Modelo exato *MILP* | X | | | X | | | X |
| Modelo exato *MINLP* | | | X | | | | |
| Multiobjetivo — soma ponderada | X | | | | | | |
| Multiobjetivo — hierárquico | X | | | | | | |
| Multiobjetivo — *trade-off* (objetivo + restrições) | X | | | | | | X |
| Ranqueamento Pareto — *fuzzy* / hipervolume | X | | | | | | |
| Heurística por invariantes de grafo (MCN/HEN/HLN) | | X | | | | | X |
| $k$-means (variações/initializações) | | | X | | X | | |
| $k$-medoids | | | | X | | X | |

posicionamento de nós *fog* com base em requisitos de latência e capacidade, comparando o desempenho de modelos heurísticos baseados em invariantes de grafos com soluções ótimas obtidas via MILP. A avaliação é conduzida por meio de uma curva de Pareto que expressa o *trade-off* entre a minimização do número de nós de *fog* implantados e a redução da latência média da rede.

# Capítulo 3

# Modelagem do Problema

Este capítulo tem como objetivo apresentar a modelagem do sistema proposta para a obtenção da solução ótima baseada em MILP e os métodos heurísticos baseados em invariantes de grafos para resolver o problema de posicionamento de nós *fog*. Na seção "Modelagem do Problema"é apresentado o modelo ótimo, realizando a parametrização do modelo, com a definição das notações e dos parâmetros utilizados na formulação matemática do problema. Em seguida, são descritas as variáveis de decisão, que representam o processo de escolha dos nós candidatos a servidores de *fog* e a forma como essas decisões são expressas no modelo. Na sequência, são detalhadas as restrições e a função objetivo, que compõem a modelagem matemática do problema, incorporando tanto os requisitos de latência e capacidade quanto o *trade-off* entre o número de servidores implantados e o desempenho da rede. Além disso, é apresentada a definição da latência média da rede, explicando-se sua formulação e a importância desse indicador para a análise de desempenho. Na seção "Modelos Baseados em Invariantes de Grafos", apresenta-se um fluxograma que sintetiza visualmente o processo de seleção dos nós destinados à implementação dos servidores de *fog*. Além disso, são descritas as diferentes formas de cálculo dos pesos utilizados na função de seleção gradual desses servidores, fundamentadas em quatro invariantes distintas da topologia.

## 3.1 Modelagem de sistema

Esta seção tem como objetivo tratar o modelo implementado para obter a solução ótima para o problema de posicionamento de nós *fog*. A Tabela 3.1 apresenta todas as notações utilizadas nessa seção para modelar o problema. O modelo é apresentado como meio de determinar os melhores pontos para posicionamento de nós *fog* em uma topologia envolvendo *datacenter* de *cloud* e nós adjacentes. Tendo como premissa os requisitos máximos de latência e mínimos de capacidade, o modelo apresenta a melhor maneira

de posicionar os nós *fog* em uma determinada topologia, de maneira a cumprir todos os requisitos e minimizar o número total de nós *fog* para a topologia.

Tabela 3.1: Notações utilizadas no modelo de otimização.

| **Notação** | **Parâmetro** |
|---|---|
| $N$ | Número de nós |
| $\eta$ | Conjunto de nós |
| $x$ | Vetor com as posições dos servidores de *fog* |
| $x_\theta$ | Indica o nó *cloud*, sempre ativo ($x_\theta = 1$) |
| $A_{n\times n}$ | Matriz de conexões dos nós na topologia |
| $L_{N\times N}$ | Matriz de latência entre os nós da topologia |
| $L_{i\theta}$ | Latência entre o nó $i$ e o nó *cloud* $\theta$ |
| $C_{N\times N}$ | Matriz de capacidade entre os nós da topologia |
| $U_i$ | Requisito de capacidade mínima |
| $\Theta_i$ | Requisito de latência máxima |
| $\phi$ | Requisito de latência máxima entre um servidor de *cloud* e um servidor de *fog* |
| $\xi$ | Capacidade mínima entre um servidor de *cloud* e um servidor de *fog* |
| $y_{ij}$ | Variável binária: 1 se o nó $i$ é atendido pelo nó $j$; 0 caso contrário |
| $\theta$ | Posição do nó de *cloud* |
| $\alpha$ | Fator de ponderação do trade-off entre número de servidores de *fog* e latência média |
| $\beta$ | Fator de normalização da latência média da rede |

A topologia da rede é composta por um conjunto de nós, sendo um deles o nó *cloud*, enquanto os demais representam os nós adjacentes. Formalmente, define-se:

$$\eta = \{1, 2, \ldots, N\}, \quad \theta \in \eta \tag{3.1}$$

onde $\eta$ é o conjunto de todos os nós da topologia e $\theta$ indica a posição do nó *cloud*. Para modelar as conexões entre esses nós, utiliza-se a matriz de conexão $A_{N\times N}$, uma matriz binária que representa os *links* presentes na topologia:

$$A_{ij} = \begin{cases} 1, & \text{se o nó } i \text{ se conecta ao nó } j, \\ 0, & \text{caso contrário.} \end{cases} \tag{3.2}$$

A modelagem do problema tem como objetivo, dada uma topologia de rede, determinar as posições ideais para os nós *fog*, de forma a atender aos requisitos de latência e capacidade, ao mesmo tempo em que se minimiza o número total desses nós na rede. Os nós *fog* podem ser alocados em qualquer posição da topologia, exceto no nó de *cloud*. As decisões de posicionamento dos nós *fog* são representadas por um vetor binário:

$$x = (x_1, x_2, \ldots, x_N) \tag{3.3}$$

Em que cada elemento $x_i$ indica se o nó $i$ foi selecionado como servidor *fog*. Formalmente, essa decisão é expressa como:

$$x_i = \begin{cases} 1, & \text{se o nó } i \text{ é um servidor } \textit{fog} \\ 0, & \text{caso contrário} \end{cases}, \quad i = 1, 2, \ldots, N \tag{3.4}$$

Para simplificar o modelo, o vetor que representa os nós *fog* também inclui a posição do nó de *cloud* como um servidor. Dessa forma, a relação é garantida como:

$$x_\theta = 1 \tag{3.5}$$

Isso assegura que o nó de *cloud* seja sempre considerado como um servidor na rede, mantendo a consistência do modelo. Os nós *cloud* e *fog* são responsáveis por fornecer serviços aos demais nós da topologia. Inicialmente, todos os nós são atendidos exclusivamente pelo nó de *cloud*. À medida que os servidores *fog* são posicionados, o atendimento é redistribuído, de modo que cada nó passe a ser atendido pelo servidor, *cloud* ou *fog*, que melhor satisfaça os requisitos de latência e capacidade.

Para formalizar qual servidor atende cada nó, utiliza-se uma matriz binária de atendimento $y_{ij}$, definida como

$$y_{ij} = \begin{cases} 1, & \text{se o nó } i \text{ é atendido pelo nó } j \\ 0, & \text{caso contrário.} \end{cases} \tag{3.6}$$

### 3.1.1 Restrições

Esta seção tem como objetivo apresentar as restrições formuladas para a modelagem do problema de posicionamento de servidores de *fog* com base no método de otimização MILP. As restrições definem os vínculos lógicos e operacionais entre as variáveis de decisão, assegurando que o modelo represente de forma consistente o comportamento da rede. Elas abrangem aspectos como a associação entre nós clientes e servidores, os limites de latência e capacidade, e as condições específicas de conectividade entre os nós *cloud* e *fog*, garantindo que a solução obtida atenda aos requisitos de desempenho.

Para modelar o problema, foram definidas restrições que estabelecem a natureza das variáveis e os vínculos entre elas. Dessa forma, o problema é regido pelas seguintes condições:

- Inicialmente, o nó de *cloud* provê serviços para todos os nós da topologia. À medida que os servidores de *fog* são posicionados, o atendimento é redistribuído, garantindo que cada nó cliente seja atendido pelo servidor, *cloud* ou *fog*, que melhor satisfaz os requisitos de latência e capacidade. Dessa forma, a matriz de atendimento $y_{ij}$

assegura que cada nó cliente esteja associado a apenas um servidor. A restrição correspondente é:

$$y_{ij} - x_j \leq 0, \quad \forall i \in \eta, \, \forall j \in \eta \tag{3.7}$$

- Os servidores de *fog*, ao serem alocados em um nó da topologia, atendem diretamente os usuários desse nó e também fornecem serviços aos nós adjacentes. Essa autossuficiência garante que as demandas locais sejam plenamente atendidas. A restrição correspondente é dada por:

$$y_{ii} = x_i, \quad \forall i \in \eta \tag{3.8}$$

- Cada nó da topologia pode ser atendido por qualquer servidor da rede, seja ele de *cloud* ou *fog*. Durante a alocação dos servidores de *fog*, cada nó cliente é associado ao servidor que proporciona a menor latência e maior capacidade. No entanto, cada cliente deve ser atendido por apenas um servidor. Formalmente, essa restrição é expressa como:

$$\sum_{j=1}^{N} y_{ij} = 1, \quad \forall i \in \eta \tag{3.9}$$

- A latência entre um nó cliente e o servidor que o atende é um critério essencial no problema de otimização. Para garantir que esse requisito seja cumprido, utiliza-se a matriz de latências $L_{ij}$ juntamente com a matriz de atendimento $y_{ij}$. A multiplicação $y_{ij} \cdot L_{ij}$ retorna a latência efetiva entre o nó $i$ e o servidor $j$, a qual deve ser menor ou igual à latência máxima permitida $\Theta_i$. Dessa forma, a restrição é formalmente expressa por:

$$y_{ij} \cdot L_{ij} \leq \Theta_i, \quad \forall i \in \eta, \, \forall j \in \eta \tag{3.10}$$

- A capacidade do *link* entre um nó cliente e o servidor que o atende é outro requisito essencial no problema de otimização. Para garantir seu cumprimento, é necessário que a capacidade disponível entre o nó $i$ e o servidor $j$ seja igual ou superior ao valor mínimo exigido $U_j$. Essa restrição é modelada utilizando a matriz de atendimento $y_{ij}$ em conjunto com a matriz de capacidades $C_{ij}$, resultando na seguinte expressão:

$$U_j \cdot y_{ij} \leq C_{ij}, \quad \forall i \in \eta, \, \forall j \in \eta \tag{3.11}$$

- O problema de otimização busca posicionar os nós *fog* de forma estratégica, garantindo que eles forneçam serviços com latência e capacidade adequadas para diferentes contextos de aplicação, cada um com requisitos específicos. Dessa forma, nós que inicialmente não atendiam aos requisitos passam a ser cobertos à medida que novos nós *fog* são alocados em posições mais próximas.

  Na arquitetura *cloud*-fog, os nós *fog* devem estar conectados ao nó de *cloud*, respeitando requisitos mínimos de latência e capacidade para assegurar o funcionamento adequado da rede. Entretanto, para viabilizar as soluções de otimização, é necessário que a comunicação entre a *cloud* e os nós *fog* tenha requisitos mais flexíveis. Essa flexibilidade permite posicionar nós *fog* mais distantes da *cloud*, atendendo melhor os servidores e clientes situados em regiões mais afastadas.

  Exigir os mesmos padrões de comunicação entre *cloud*-fog e os clientes finais poderia inviabilizar algumas soluções, restringindo as posições disponíveis para os nós *fog* e tornando a implementação pouco prática em topologias médias e grandes. Por isso, são definidos requisitos específicos de latência e capacidade para os *links* entre a *cloud* e os nós *fog*, garantindo uma implementação mais eficiente e escalável da rede. Assim, as seguintes relações matemáticas estabelecem essas restrições:

$$x_i L_{\theta i} \leq \phi, \quad \forall i \in \eta \tag{3.12}$$

$$x_i \xi \leq C_{\theta i}, \quad \forall i \in \eta \tag{3.13}$$

### 3.1.2 Problema de Otimização

Diante das restrições estabelecidas, o posicionamento dos nós *fog* deve assegurar o atendimento simultâneo aos requisitos de latência e capacidade. Entretanto, a introdução de novos servidores acarreta custos adicionais de implementação, tornando necessário um equilíbrio entre desempenho e eficiência. Nesse sentido, considera-se um *trade-off* entre dois objetivos conflitantes:

- **Minimização do número de servidores de *fog***: reduzir a quantidade de nós implementados, de modo a conter os custos de implantação e operação da infraestrutura;
- **Redução da latência média da rede**: melhorar o desempenho global, garantindo menor tempo de resposta entre os dispositivos finais e os servidores de *fog*.

A latência média da rede é definida como o valor médio da latência entre cada nó da topologia e o respectivo servidor que o atende. A matriz $y_{ij}$ indica essa relação de atendimento, assumindo o valor 1 quando o nó $i$ é atendido pelo nó $j$; e 0 caso contrário. Já a matriz $L_{ij}$ contém os valores de latência entre cada par de nós da topologia, conforme os dados fornecidos pelo *dataset*. Assim, o produto $y_{ij} \cdot L_{ij}$ representa a latência efetiva entre o nó $i$ e o servidor $j$ responsável por seu atendimento, resultando em zero para todas as demais combinações. A latência média da rede é, portanto, obtida pela soma desses produtos para todos os pares de nós, dividida pelo número total de nós $N$. Dessa forma, a latência média de toda a rede é calculada pela Equação 3.14:

$$\frac{1}{N} \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} y_{ij} \cdot L_{ij} \tag{3.14}$$

A minimização do número de servidores de *fog* tem como objetivo reduzir a quantidade total de nós implantados na topologia, buscando uma configuração mais eficiente e de menor custo. O vetor $x = (x_1, x_2, x_3, \ldots, x_N)$ representa as posições dos servidores de *fog*, em que cada elemento assume o valor $x_i = 1$ quando há um servidor de *fog* alocado na posição $i$, e $x_i = 0$ caso contrário. Assim, o número total de servidores de *fog* implantados na rede é obtido pela soma de todos os valores desse vetor, conforme expresso na Equação 3.15:

$$\sum_{i=0}^{N-1} x_i \tag{3.15}$$

Assim, a formulação do problema de otimização multiobjetivo que combina a minimização da latência média e a minimização do número de servidores de *fog* é uma combinação das duas Equações 3.14 e 3.15, ponderadas por um fator de ponderação $\alpha$, os seus valores variam linearmente entre 0 e 1, no qual 0 significa considerar apenas o objetivo de minimizar o número de *fogs*, e 1 minimizar apenas a latência média:

$$\alpha \in [0, 1] = \alpha \in \mathbb{R} \mid 0 \leq \alpha \leq 1, \tag{3.16}$$

Para além do fator de ponderação, é necessários que ambas as funções objetivo sejam normalizadas, por conta disso, considera-se o fator de normalização para o número de servidores de *fog* como sendo:

$$\frac{1}{N} \tag{3.17}$$

O fator de normalização da latência média foi definido com base na latência média de todos os nós e a *cloud*, definido como $\beta$, é calculado pela Equação 3.18:

$$\beta = \frac{1}{N} \sum_{i=0}^{N-1} L_{i\theta} \tag{3.18}$$

Dessa forma, a equação que define o multiobjetivo entre minimizar a latência média da rede e minimizar o número de servidores de *fog* implantados na rede, utilizando um *tradeoff* entre os dois objetivos é calculada pela Equação 3.19:

$$\min \left( \frac{1-\alpha}{N} \sum_{i=0}^{N-1} x_i + \frac{\alpha}{\beta} \cdot \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} y_{ij} \cdot L_{ij} \right) \tag{3.19}$$

*s.t.*

$$(3.7), (3.8), (3.9), (3.10), (3.11), (3.12), (3.13), (3.14) \; e \; (3.15).$$

Como pode ser observado, a linearidade do novo problema é conservada, tornando a solução ótima possível de ser obtida a partir do uso de algoritmos típicos de otimização linear, como *simplex*, *branch and bound* e *branch and cut*. Estes algoritmos são comumente utilizados por bibliotecas como CPLEX [24] e *OR-Tools* [25]. É importante mencionar que a implementação e solução desse algoritmo pode ser encontrada no Anexo I deste documento, e também como no código fonte disponibilizado no Github [26]. Todavia, vale ressaltar que o problema acima depende do completo conhecimento das condições da rede e não é capaz de resolver o problema sem essas informações. Além disso, os algoritmos de programação linear consomem demasiado processamento, possuem convergência não polinomial e não garantem convergência da solução, o que torna necessária a exploração de estratégias que podem consumir menor quantidade de processamento e que garantam a solução.

## 3.2 Modelos Baseados em Invariantes de Grafos

Esta seção tem como objetivo apresentar as abordagens heurísticas desenvolvidas com base em invariantes de grafos para o problema de posicionamento de nós *fog*. Diferentemente do modelo exato de otimização, essas heurísticas buscam oferecer soluções aproximadas mais simples e de menor custo computacional, explorando propriedades estruturais da topologia da rede. São propostas estratégias que utilizam métricas de excentricidade e conectividade para identificar nós candidatos à alocação de servidores de *fog*. A seção

descreve a lógica geral dos algoritmos, ilustrada por meio de um fluxograma, e detalha o funcionamento de cada abordagem, explicando como os critérios de latência e capacidade são incorporados no processo de seleção dos nós.

Resolver o problema de posicionamento de nós *fog* de forma ótima oferece a melhor solução em termos de minimizar o número de nós necessários na topologia. Entretanto, em cenários reais, modelar e implementar a abordagem ideal pode não ser prático e viável [27]. Como alternativa, é possível adotar estratégias mais diretas e intuitivas, como a identificação de pontos de maior excentricidade ou de maior conectividade. A fim de avaliar essas abordagens em relação à solução ótima, foram desenvolvidos algoritmos que exploram essas características de excentricidade e conectividade dos nós. Todos os algoritmos propostos seguem uma estratégia comum, representada no diagrama da Figura 3.1.

A execução inicia-se com a leitura dos dados de entrada, que incluem as matrizes de conexões, latência e capacidade da topologia, os requisitos de cada nó e servidor de *fog*, além da localização do nó de *cloud*. Em seguida, o algoritmo verifica se todos os nós da topologia já atendem aos requisitos de latência e capacidade. Caso todos os critérios sejam satisfeitos, o processo é encerrado; caso contrário, inicia-se o *loop* de posicionamento dos servidores de *fog*.

O *loop* começa pela identificação dos nós elegíveis para se tornarem servidores de *fog*. Essa elegibilidade é determinada com base nas restrições definidas nas Equações 3.12 e 3.13, que asseguram que os parâmetros de latência e capacidade entre o nó candidato e o nó de *cloud* estejam dentro dos limites estabelecidos. Após a seleção dos nós elegíveis, o algoritmo calcula um peso associado a cada nó candidato, de acordo com um critério específico. Esse critério varia conforme a heurística adotada e é responsável por definir a prioridade de seleção dos nós, sendo detalhado nas subseções seguintes. O nó que obtiver o maior peso é então promovido a servidor de *fog*. A partir dessa atualização, a matriz de atendimento $y_{ij}$ é recalculada, de modo que cada nó cliente seja atendido pelo servidor, de *fog* ou *cloud*, que ofereça o melhor desempenho em termos de latência e capacidade. Por fim, o algoritmo retorna à etapa de verificação para avaliar se todos os requisitos da topologia foram atendidos com a nova alocação. Caso positivo, o processo é finalizado; caso contrário, o *loop* é repetido até que todos os requisitos sejam satisfeitos ou até que não restem mais nós candidatos elegíveis para a implantação de servidores de *fog*.

### 3.2.1 Abordagem Baseada no Cálculo da Excentricidade

Esta seção apresenta a abordagem baseada na métrica de excentricidade para o posicionamento de servidores de *fog*, cujo objetivo é identificar os nós mais afastados em relação à *cloud* e aos demais servidores já posicionados, promovendo assim uma distribuição mais

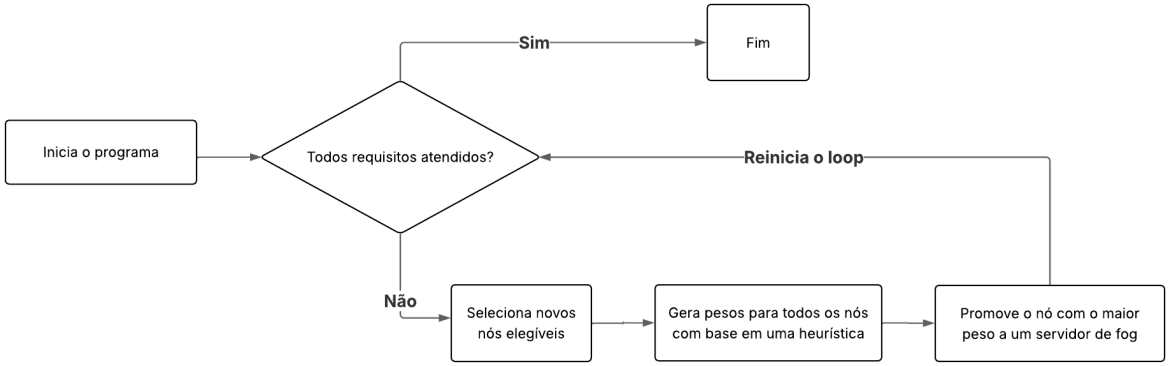



Figura 3.1: Diagrama da lógica de funcionamento das heurísticas utilizadas.

equilibrada e eficiente da infraestrutura na rede. A excentricidade é utilizada como métrica para quantificar o grau de afastamento de um nó em relação a um ou mais nós de referência, servindo como critério para selecionar os pontos mais estratégicos de alocação dos servidores de *fog*. No contexto deste trabalho, essa medida é adaptada para refletir as características de desempenho da rede, podendo ser calculada com base na latência ou na capacidade de comunicação entre os nós.

### Cálculo da Excentricidade Com Base na Latência

A latência representa o tempo de atraso necessário para que uma informação percorra o caminho entre dois pontos da rede. À medida que a distância entre os pontos aumenta, a latência tende a crescer, podendo, portanto, ser utilizada como parâmetro de medida de distância. Dessa forma, é possível determinar a excentricidade de cada nó a partir da latência em relação aos demais nós da topologia e à *cloud*, por meio da Equação 3.20:

$$E_j = \sum_{i=0}^{N} L_{ij}^2 \cdot x_i \tag{3.20}$$

Em que $E_j$ representa a excentricidade do nó $j$, $L_{ij}$ corresponde à latência entre os nós $i$ e $j$, e $x_i$ indica a presença de um servidor de *fog* ou *cloud* no nó $i$. Assim, a excentricidade de um nó é obtida pela soma quadrática das latências entre esse nó e todos os demais.

### Cálculo da Excentricidade com Base na Capacidade

A capacidade de comunicação entre dois nós de uma rede, representada pela matriz $C_{ij}$, é geralmente inversamente proporcional à distância física ou lógica entre eles, isto é, quanto maior a distância, menor tende a ser a capacidade do enlace. Assim, diferentemente da latência, que aumenta com a distância, a capacidade diminui, de modo que a excentricidade deve ser modelada de forma inversamente proporcional aos valores de capacidade.

Considerando essa relação inversa, a excentricidade de um nó $j$ pode ser expressa como a soma ponderada do inverso do quadrado da capacidade entre $j$ e os demais nós da topologia, conforme a Equação 3.21:

$$E_j = \sum_{i=0}^{N} \frac{1}{C_{ij}^2} \cdot x_i \tag{3.21}$$

Em que $E_j$ representa a excentricidade do nó $j$ com base na capacidade, $C_{ij}$ é a capacidade do enlace entre os nós $i$ e $j$, e $x_i$ indica a presença de um nó no ponto $i$. Dessa forma, nós situados em regiões com menor capacidade média em relação aos demais tendem a apresentar maior excentricidade, o que permite identificar pontos mais isolados para posicionamento de servidores de *fog*.

### 3.2.2 Abordagem Baseada no Cálculo da Conectividade

Outro algoritmo empregado baseia-se no posicionamento do nó de *fog* a partir do nó mais conectado da topologia. Essa abordagem parte da premissa de que um nó com muitas conexões pode atender eficientemente diversos nós vizinhos. Isso permite uma prestação de serviços mais próxima e com menor latência para grande parte da rede. Assim, a escolha do nó mais conectado busca maximizar a cobertura e o alcance dos serviços de *fog*.

No entanto, essa estratégia apresenta limitações que comprometem sua eficácia. É comum que vários nós possuam o mesmo número de conexões, gerando empates. Essa situação dificulta a seleção precisa do nó mais adequado para receber o servidor de *fog*. Portanto, é necessário um critério adicional para orientar a decisão final.

Para resolver esse impasse, utiliza-se um fator de desempate baseado na excentricidade. A excentricidade é calculada em função da latência, conforme Equação 3.20, ou da capacidade, conforme Equação 3.21, em relação à *cloud* e aos demais servidores de *fog*. Esse critério complementa a análise da conectividade ao incorporar a posição relativa do nó na topologia. Dessa forma, garante-se uma escolha mais estratégica e equilibrada.

O método final combina os critérios de conectividade e excentricidade. Essa combinação aproveita o alcance dos nós mais conectados e a posição estratégica dos mais excêntricos. Assim, os servidores de *fog* são alocados em pontos que maximizam a eficiência da rede. Com isso, espera-se uma distribuição mais robusta e funcional na topologia em relação ao cumprimento dos requisitos e melhoria das características da rede.

# Capítulo 4

# Proposta

Este capítulo apresenta os objetivos e contribuições do estudo, descrevendo o modelo analítico usado como referência e as heurísticas baseadas em invariantes de grafos comparadas à solução ótima obtida por programação inteira mista. O capítulo detalha o funcionamento geral da solução, a estrutura experimental e os *datasets* utilizados, bem como os critérios de seleção de dados, ferramentas e bibliotecas empregadas para assegurar reprodutibilidade e rigor metodológico. Além disso, define o plano de avaliação baseado na frente de Pareto entre número de servidores e latência média, destacando as métricas e requisitos de desempenho de cada aplicação. Por fim, discute as limitações do modelo e as condições em que certas topologias ou aplicações podem gerar soluções inviáveis, contextualizando os resultados dentro dos limites técnicos e práticos da abordagem proposta.

## 4.1 Objetivo e Contribuição

Este trabalho investiga o desempenho de heurísticas baseadas em invariantes de grafos para o posicionamento de nós *fog* em redes *cloud* que suportam IIoT, jogos *on-line*, *streaming* 4K e Videoconferência em HD. O objetivo é comparar essas heurísticas à solução ótima obtida por modelagem analítica, avaliando o trade-off entre (i) o número de nós *fog* instalados e (ii) a latência média da rede. A principal contribuição é um procedimento de avaliação que permite identificar, para cada contexto de aplicação, quais métodos sub-ótimos fornecem a melhor relação custo-desempenho em relação à frente de Pareto da solução ótima, oferecendo um guia prático de seleção de estratégias conforme os requisitos de latência e capacidade.

## 4.2 Visão Geral da Solução

Os modelos propostos possuem um objetivo amplo de possibilitar uma visão sobre os benefícios de cada método heurístico em cada contexto. Os experimentos, no entanto, possuem soluções para o posicionamento dos nós *fog* que são concretas. Essas soluções têm como base posicionar nós *fog* em uma rede *cloud* com problemas de cumprimento de requisitos de latência e capacidade para certos usuários finais da rede. Com isso. a solução visa garantir o comprimento dos requisitos pelo posicionamento de nós *fog*.

A latência entre dois pontos de uma rede é altamente prejudicada pela distância que esses dois pontos se separam. Em uma rede *cloud*, o servidor de *cloud* idealmente se encontra em uma posição geograficamente e topologicamente central, mas ainda assim as distâncias que se estendem por um país ou continente podem ser altas o suficientes para que a latência de comunicação entre a *cloud* e o usuário final seja inviável a depender dos requisitos da aplicação executada pelo usuário final.

Além do fator da distância geográfica, uma rede *cloud* é composta de nós de rede que intermediam a conexão entre a *cloud* e os usuários finais. Os dados trocados entre a *cloud* e o usuário final trafegam entre vários nós intermediários até chegar em seu destino. Os requisitos de latência e capacidade sofrem de maneiras diferentes com o tamanho e complexidade da rede, e também com a qualidade dos *links* de conexão entre cada nó intermediário da rede. Com isso, a qualidade dos requisitos é também uma função da estrutura da topologia e qualidade dos seus *links* de conexão e a implementação dos nós *fog* deve levar isso em conta.

A Figura 4.1 ilustra um diagrama de funcionamento do processo de implementação de nós *fog*. Essa figura é dividida em duas partes, onde a primeira apresenta um cenário de rede *cloud* onde não existem nós *fog* implementados, e a segunda um cenário no qual os nós *fog* foram implementados em nós específicos da rede.

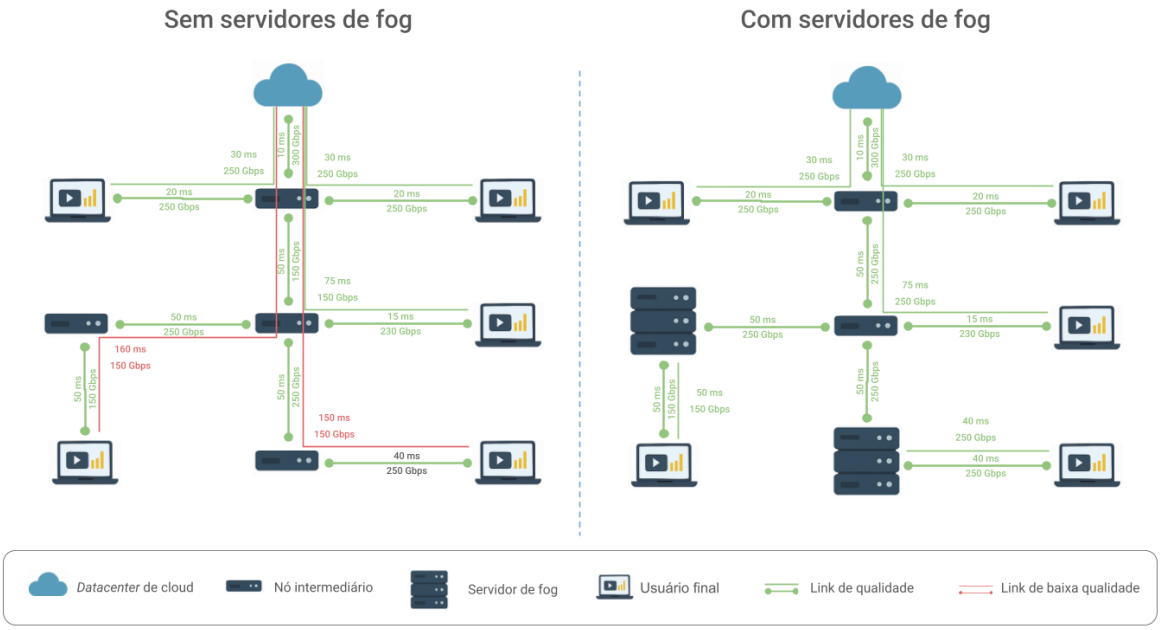



Figura 4.1: Diagrama simplificado da implementação dos nós *fog* em uma rede *cloud*.

Nesse diagrama, é ilustrado uma rede *cloud* simplificada contendo o servidor de *fog* e quatro nós intermediários. Na primeira metade do diagrama, a topologia possui usuários finais cujos seus requisitos de aplicação não são atendidos satisfatoriamente, esses *links* de baixa qualidade são ilustrados na cor vermelha. Também são ilustrados *link* de conexão em que os requisitos são satisfatoriamente cumpridos, estes estão na cor verde.

Na topologia sem nós *fog*, os serviços são fornecidos unicamente pela *cloud* e, dessa forma, os requisitos de conexão são os resultantes da comunicação do usuário final até a *cloud*, intermediada por todos os seus nós adjacentes. O intermédio dos nós entre a *cloud* e o usuário final é um fator a se considerar quando se avalia o desempenho da rede em cumprir requisitos de latência e capacidade.

A capacidade dos *links* de conexão da *cloud* para o usuário final não pode exceder o limite de nenhum *link* intermediário. Com isso, a capacidade de conexão entre a *cloud* e qualquer outro nó da rede é sempre definida como a menor capacidade entre todos os nós no caminho entre esses dois nós da rede (Figura 4.2).

A latência nesse contexto é o requisito mais prejudicado, dado que o tempo em que um dado é transmitido entre um ponto e outro é a soma dos tempos de cada caminho. Dessa forma, a latência final entre o usuário e a *cloud* é a soma das latências entre todos os nós intermediários, como ilustrado na Figura 4.2.

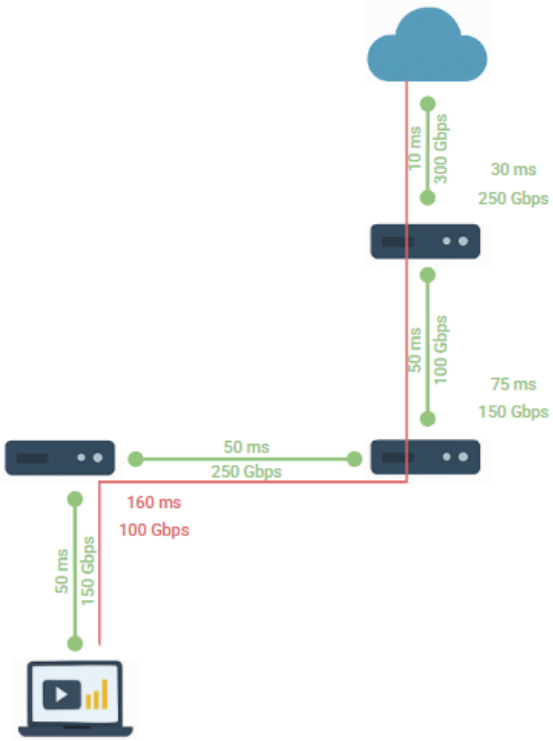



Figura 4.2: Diagrama da latência e capacidade reais entre um usuário e a *cloud*.

A solução estudada neste trabalho visa então posicionar nós *fog* próximos aos usuários finais. Essa implementação é realizada de maneira estratégica de forma a posicionar os servidores em locais onde possam fornecer os serviços da *cloud* ao usuários finais de maneira mais próxima e, com isso, com melhor qualidade em relação à latência e à capacidade, como apresentado na Figura 4.3.

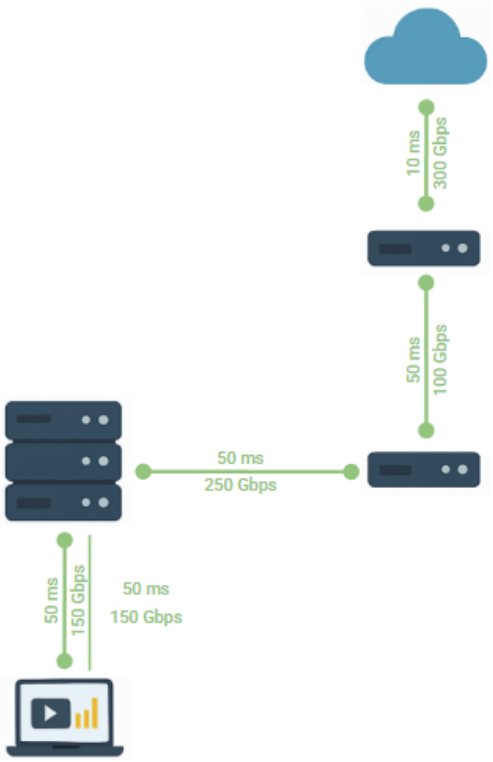



Figura 4.3: Diagrama de implementação de um nós *fog* para melhoria da latência e capacidade da rede.

## 4.3 Protótipo

Esta seção descreve o cenário experimental adotado para a validação dos modelos e heurísticas propostos. Para isso, são apresentados o ambiente de execução, as bibliotecas e ferramentas utilizadas, bem como a estrutura do código-fonte disponibilizado publicamente para assegurar a reprodutibilidade dos resultados. O objetivo é detalhar as condições sob as quais os experimentos foram conduzidos, garantindo transparência metodológica e permitindo a replicação dos testes por outros pesquisadores. Além disso, esta seção introduz os *datasets* empregados, as topologias de rede utilizadas e os critérios de seleção e preparação dos dados, estabelecendo a base para a análise de desempenho e a comparação entre as abordagens de otimização e heurísticas desenvolvidas.

### 4.3.1 Cenário Experimental

O código-fonte dos experimentos está disponibilizado em repositório público no GitHub [26]. O repositório inclui: (i) instruções de instalação das dependências (`requirements.txt`);

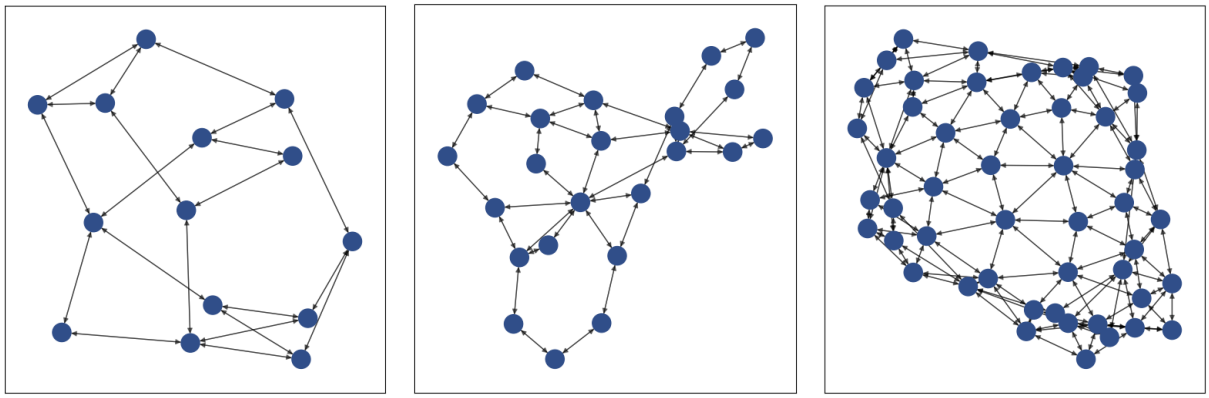
(ii) guia para *download* e organização dos *datasets*; (iii) *scripts* de execução com parâmetros padronizados para cada topologia e aplicação; e (iv) rotinas para pós-processamento e geração automática das tabelas; e (v) as figuras apresentadas nesta dissertação. Para assegurar reprodutibilidade, o repositório fornece arquivos de configuração com *seed* fixo.

A reprodutibilidade dos experimentos por meio da configuração de *seeds* garante que todas as operações aleatórias realizadas durante a execução dos algoritmos produzam sempre os mesmos resultados, independentemente do ambiente ou do momento em que o experimento é executado. Isso significa que qualquer pessoa, utilizando o mesmo código e os mesmos dados, poderá refazer os experimentos e obter resultados idênticos, facilitando a validação científica, auditoria dos métodos e comparação justa entre diferentes abordagens.

Os experimentos foram implementados em Python e fazem uso de um conjunto de bibliotecas amplamente consolidadas na comunidade científica para apoiar as tarefas de modelagem, execução e análise dos resultados. As bibliotecas *NumPy* [28] e *Pandas* [29] foram utilizadas para o processamento e a manipulação vetorial e tabular dos dados, enquanto a *Matplotlib* [30] foi empregada na geração dos gráficos. A biblioteca *SciPy* [31] foi utilizada para o cálculo do erro padrão da média e dos intervalos de confiança de 90% aplicados aos resultados experimentais, e a *NetworkX* [32] foi responsável pela representação e análise dos grafos de topologia de rede. Para a solução ótima, foi utilizada a biblioteca *OR-Tools* do Google [25], por meio do módulo `pywraplp`, para a construção e resolução do modelo de programação inteira mista. Além disso, o acesso estruturado aos *datasets* de rede foi realizado por meio da biblioteca *datanetAPI*, desenvolvida pelo Barcelona Neural Networking Center (BNN/UPC)[33], que fornece as matrizes de desempenho e tráfego de cada topologia.

### 4.3.2 *Datasets*

Com o objetivo de analisar o posicionamento de nós *fog* para o provisionamento de serviços em *cloud*, como *streaming*, jogos *online*, IoT e transmissão em 4K, foram selecionados três conjuntos de dados baseados em topologias de grande escala: NSFNET [8], GEANT2 [9] e SYNTH50BW [10]. Essas topologias representam redes reais históricas ou sintéticas de médio e grande porte, frequentemente empregadas como referência em pesquisas sobre modelagem e otimização de redes. A Figura 4.4 ilustra as três topologias utilizadas nos experimentos.



(a) NSFNET (b) GEANT2 (c) SYNTH50BW

Figura 4.4: Topologias utilizadas.

A NSFNET (National Science Foundation Network), com sua topologia ilustrada na Figura 4.4 (a), foi uma rede criada em 1985 pela National Science Foundation dos Estados Unidos para interligar centros de supercomputação e redes regionais acadêmicas, atuando como um *backbone* nacional e servindo como elo de transição entre a ARPANET e a internet comercial. Essa rede introduziu o uso de roteadores modernos, adotou os protocolos TCP/IP e foi pioneira na aplicação de Ethernet sobre linhas telefônicas, impulsionando a padronização de tecnologias que permitiram a interconexão aberta de diferentes sistemas. Embora tenha sido desativada em 1995, a NSFNET deixou como legado um modelo de rede de redes e avanços técnicos que possibilitaram a expansão da internet em escala global [8].

A GEANT2, com sua topologia ilustrada na Figura 4.4 (b), foi a segunda geração da rede pan-europeia de pesquisa e educação, operada pela associação GEANT e cofinanciada pela Comissão Europeia e pelas redes nacionais de ensino e pesquisa (NRENs) dos países participantes. Implementada a partir de 2005, ela interligava universidades e centros de pesquisa em mais de 30 países por meio de um *backbone* óptico de altíssima capacidade baseado em tecnologia DWDM, permitindo conexões dedicadas e de baixa latência para aplicações científicas de grande escala. Essa infraestrutura fomentou a interoperabilidade entre redes nacionais e consolidou um modelo federado de conectividade acadêmica na Europa. A GEANT2 serviu de base para as gerações subsequentes da rede GEANT, que hoje constituem a espinha dorsal das comunicações científicas europeias [9].

A SYNTH50BW, com sua topologia ilustrada na Figura 4.4 (c), é uma topologia sintética de 50 nós utilizada em experimentos e *benchmarks* de redes para avaliar algoritmos de modelagem e posicionamento em cenários controláveis e reproduzíveis [34]. Esse conjunto de dados foi gerado por simulações em nível de pacotes com o OMNeT++ [35], inclui a largura de banda de cada enlace como atributo e oferece volume amostral suficiente para treinar e avaliar modelos sob diferentes perfis de tráfego. Em trabalhos recentes de mo-

delagem com GNNs, a SYNTH50BW é empregada ao lado de NSFNET e GEANT2 para testar generalização entre topologias, servindo como ponte entre redes reais e ambientes experimentais [36]. A sua inclusão neste estudo complementa as topologias históricas ao permitir análises sistemáticas de latência, capacidade e conectividade em uma rede de médio porte.

O acesso aos conjuntos de dados pode ser feito por meio do site oficial do projeto *Knowledge-Defined Networking (KDN)* [37], mantido por pesquisadores da *Universitat Politècnica de Catalunya (BNN/UPC)* [33]. Esse portal disponibiliza diversos *datasets* voltados à modelagem de redes, destinados a pesquisas em aprendizado de máquina aplicado a redes (como SDN, otimização, GNNs e DRL). Após o *download*, o acesso estruturado aos dados é feito por meio da biblioteca *datanetAPI* do *Barcelona Neural Networking Center (BNN/UPC)* [33], conforme a documentação oficial [37].

Cada *dataset*, organizado por topologia, contém cerca de 500 amostras independentes. Cada amostra é representada por um dicionário Python que reúne diversos conjuntos de informações sobre a topologia de rede, os fluxos de tráfego e os resultados de desempenho obtidos na simulação. Essa estrutura inclui: (i) campos globais com métricas gerais da execução; (ii) a `performance_matrix`, com os indicadores de desempenho por par de nós e por fluxo; (iii) a `traffic_matrix`, com os parâmetros e medições do tráfego gerado; (iv) a `routing_matrix`, que define os caminhos de roteamento adotados para cada par; (v) o grafo `topology_object`, que representa a topologia da rede com seus nós e enlaces; e (vi) campos auxiliares como `_flowresults_line` e `_results_line`, usados para registrar os resultados em formato CSV. Os campos globais reúnem métricas gerais da simulação, válidas para toda a rede ou amostra:

- `data_set_file`: caminho do arquivo `.tar.gz` do qual a amostra foi extraída;
- `global_delay`: atraso médio por pacote na rede (*time units*), calculado sobre todos os caminhos e fluxos;
- `global_losses`: perdas agregadas por unidade de tempo (*packets/time unit*);
- `global_packets`: número de pacotes gerados ou transmitidos por unidade de tempo (*packets/time unit*);
- `maxAvgLambda`: intensidade máxima de tráfego usada na geração da matriz de tráfego (*bits/time unit*).

O campo `performance_matrix` é uma matriz $N \times N$ na qual cada entrada `[src,dst]` é um dicionário contendo métricas de desempenho do par de nós `src` e `dst`, tanto de forma agregada quanto por fluxo. Cada entrada possui a chave `AggInfo`, que reúne métricas

como `PktsDrop` (perdas agregadas), `AvgDelay` (atraso médio), percentis `p10...p90` da distribuição do atraso e `Jitter` (variância do atraso). Também há a chave `Flows`, uma lista de dicionários, um para cada fluxo entre `src` e `dst`, contendo as mesmas métricas em nível individual.

O campo `traffic_matrix` também é uma matriz $N \times N$, em que cada entrada `[src,dst]` descreve a parametrização e as medições de tráfego do par de nós. A chave `AggInfo` inclui métricas como `AvgBw` (taxa média de transmissão em *bits/time unit*), `PktsGen` (taxa média de geração de pacotes) e `TotalPktsGen` (contagem total de pacotes). A chave `Flows` armazena uma lista de dicionários, um por fluxo, com parâmetros do gerador estocástico (`ToS`, `TimeDist`, `SizeDist`) e suas configurações detalhadas, além das mesmas métricas do tráfego por fluxo.

Além disso, cada amostra inclui o campo `routing_matrix`, uma matriz $N \times N$ (`dtype=object`) na qual cada entrada é uma lista de IDs de nós representando o caminho dirigido entre `src` e `dst`; todos os fluxos de um mesmo par compartilham o mesmo caminho. O campo `topology_object` é um grafo dirigido `networkx.MultiDiGraph` que representa a topologia, com atributos como largura de banda (*bits/tempo*), `port` e `weight` associados às arestas, acessíveis por `G.nodes`, `G.edges` e `G[u][v][k]`. Campos auxiliares como `_flowresults_line` e `_results_line` armazenam os resultados em formato CSV.

As seguintes convenções se aplicam:

- **Valores faltantes**: valores `-1` indicam métricas não medidas ou não aplicáveis (por exemplo, pares sem tráfego);
- **Tipos numéricos**: valores podem aparecer como tipos NumPy (por exemplo, `np.float64`), mas semanticamente representam números de ponto flutuante;
- **Unidades**: *bits/time unit* (largura de banda), *packets/time unit* (taxas/perdas), *time units* (atraso) e *time units*$^2$ (*jitter*).

Nos experimentos realizados, foram extraídas de cada amostra duas matrizes principais: a de latência e a de capacidade. As latências vieram de `performance_matrix` (`AggInfo.AvgDelay`) e as de capacidades de `traffic_matrix` (`AggInfo.AvgBw`) de cada par `[src,dst]`. Para os elementos da diagonal principal (`[i,i]`), foram atribuídos manualmente valores fixos de 10 ms de latência e $10^6$ bits/tempo de capacidade, a fim de evitar valores nulos ou inconsistentes na auto-referência de cada nó.

### 4.3.3 Seleção dos Dados

A seleção dos dados utilizados nos experimentos foi efetuada de maneira a garantir a uniformidade na aplicação dos métodos, utilizando os mesmos conjuntos de dados pro-

venientes das três topologias. Para tanto, os dados foram extraídos dos *datasets* e armazenados em variáveis, de forma que todos os experimentos de uma mesma topologia compartilhassem os mesmos dados.

Para avaliar a melhor forma de variar as configurações das topologias, adotou-se o critério de selecionar aleatoriamente o servidor de *cloud* para cada experimento, mantendo, contudo, a posição do servidor consistente para cada heurística. Dessa forma, garantiu-se a execução das heurísticas com os mesmos dados e a mesma localização do servidor de *cloud*. Essas escolhas foram feitas de maneira a assegurar uma comparação justa entre as diferentes heurísticas.

### 4.3.4 Desempenho e Complexidade

Para a solução ótima, o algoritmo que constrói o modelo para ser resolvido pela ferramenta de otimização possui complexidade $O(N^2)$, onde $N$ é o número de nós da topologia. Esse algoritmo, no entanto, apenas constrói o modelo para que a ferramenta possa executar o processo de otimização e, dessa forma, a complexidade é exponencial no pior caso devido à natureza do *simplex*.

Os algoritmos de todos os métodos heurísticos desenvolvidos possuem a mesma lógica de execução descrita na Figura 3.1, de forma que o único passo que diferencia os algoritmos são o cálculo dos pesos para seleção dos nós elegíveis para terem nós *fog* implementados. Dessa forma, os algoritmos, compartilhando a mesma lógica de funcionamento, possuem uma mesma ordem de complexidade.

Em relação às heurísticas, de maneira geral, os algoritmos realiza iterações com laço `while` em que em casa iteração, o código reavalia os requisitos de todos os nós e reconstrói integralmente a matriz de atendimento $y_{ij}$, operações que percorrem estruturas de tamanho $N \times N$ e custam $\mathcal{O}(N^2)$ por iteração. Como o algoritmo pode executar até $N$ iterações, adicionando um novo nó *fog* a cada passo, o custo acumulado atinge $N \times \mathcal{O}(N^2) = \mathcal{O}(N^3)$. O passo de seleção com `sorted` contribui com $\mathcal{O}(N^2 \log N)$ ao longo da execução, mas esse valor é assintoticamente menor e não altera o termo dominante $\mathcal{O}(N^3)$.

A partir dessa estrutura comum analisada, é possível constatar que o tempo de execução dos algoritmos heurísticos crescem de forma cúbica em relação ao número de nós $N$, resultando em uma complexidade de $\mathcal{O}(N^3)$ no pior caso.

## 4.4 Planos de Avaliação

Esta seção tem como objetivo apresentar as aplicações consideradas na avaliação experimental e os respectivos requisitos de desempenho adotados como base para os testes. As aplicações foram selecionadas de forma a representar diferentes perfis de uso de rede,

abrangendo cenários com demandas variadas de latência e capacidade, como IIoT, vídeo conferência em HD, jogos *online* e *streaming* em 4K.

### 4.4.1 Aplicações

Os experimentos foram planejados para contemplar cenários com diferentes requisitos de latência e capacidade. Para isso, foram selecionadas aplicações com exigências específicas para esses dois parâmetros. Para representar cenários com requisitos mais rigorosos de latência, foi selecionada a aplicação de *Industrial Internet of Things* (IIoT). Em ambientes industriais, aplicações IIoT exigem tempos de resposta extremamente baixos para garantir a sincronização adequada entre sensores, atuadores e controladores em linhas de produção, de modo que qualquer atraso na comunicação possa comprometer a eficiência e a segurança do sistema. De acordo com [38], a latência constitui um dos principais fatores de desempenho em sistemas IIoT, sendo determinante para a confiabilidade das operações em tempo real. O estudo demonstra que a latência tende a aumentar proporcionalmente ao número de unidades IoT conectadas, ressaltando a necessidade de arquiteturas de rede otimizadas e de tecnologias capazes de manter atrasos mínimos na transmissão de dados. Além disso, por se tratarem de aplicações que transmitem pacotes curtos e frequentes, as redes IIoT não demandam altas capacidades de banda, priorizando, portanto, a baixa latência em detrimento da largura de banda.

No contexto de aplicações de vídeo conferência em alta definição (HD), o requisito de latência é importante, mas não tão rigoroso quanto em aplicações críticas em tempo real. De acordo com [39], serviços multimídia, como *video streaming*, *video on demand* e transmissões em HD ou até 4K, exigem elevadas taxas de *throughput*, da ordem de gigabits por segundo, para garantir qualidade de vídeo contínua e sem interrupções. Entretanto, esses serviços toleram atrasos moderados, desde que a latência permaneça abaixo de um limiar aceitável, tipicamente inferior a 0,1 segundo, o que assegura uma boa *Quality of Experience* (QoE) para o usuário final. Assim, as aplicações de vídeo em HD demandam alta capacidade de rede e estabilidade de conexão, priorizando o aumento da largura de banda em detrimento da minimização extrema da latência, ao contrário de cenários industriais ou de controle em tempo real, onde cada milissegundo é crítico para o desempenho do sistema [39].

No contexto de jogos *online*, os requisitos de latência e capacidade se equilibram em níveis igualmente elevados. Conforme destacado em [40], aplicações de jogos massivos e interativos, como os MMOs (*Massively Multiplayer Online games*), são altamente sensíveis à latência, pois qualquer atraso perceptível na comunicação entre jogador e servidor pode comprometer significativamente a *Quality of Experience* (QoE) e até levar à desconexão do usuário. Além disso, esses sistemas envolvem intensa troca de dados em tempo real,

incluindo atualização de estados de jogo, renderização, física e inteligência artificial, o que impõe elevadas demandas de *throughput* e largura de banda. Dessa forma, aplicações de jogos *online* exigem simultaneamente baixas latências e alta capacidade de rede, justificando o uso de arquiteturas baseadas em *fog computing* para aproximar os servidores dos usuários e reduzir atrasos de comunicação [40]. A Tabela 4.1 apresenta os cenários experimentais e seus respectivos requisitos.

Tabela 4.1: Requisitos das aplicações.

| **Aplicação** | **Latência máxima (ms)** | **Mínima capacidade (MBps)** |
|---|---|---|
| IIoT | 15 | 0.1 |
| Vídeo conferência (HD) | 100 | 1.5 |
| Streaming (4K) | 70 | 3.2 |
| Jogos online | 20 | 4.0 |

### 4.4.2 Frente de Pareto

A avaliação das heurísticas baseadas em invariantes de grafos foi realizada em comparação com a solução ótima obtida pelo modelo MILP. Para garantir consistência, foram consideradas duas métricas principais de qualidade da rede: (i) o número de nós *fog* necessários para atender aos requisitos de latência e capacidade e (ii) a latência média da rede após a implantação. Essas métricas permitem comparar diretamente o desempenho das heurísticas em relação à solução ótima.

A análise dos resultados baseou-se no *trade-off* entre essas duas métricas, representado pela frente de Pareto derivada da solução ótima (Equação 3.19). Cada ponto da curva corresponde a uma solução eficiente, em que qualquer melhoria em um dos critérios implica a deterioração do outro, evidenciando o compromisso entre custo de infraestrutura e desempenho. A Figura 4.5 apresenta a forma típica da curva de Pareto obtida nas simulações, em que cada ponto representa uma configuração associada a um valor distinto do parâmetro $\alpha$, que pondera a importância relativa entre os objetivos. À medida que $\alpha$ aumenta, as soluções passam a priorizar o objetivo 1 em detrimento do objetivo 2; já valores menores de $\alpha$ produzem o efeito inverso. As setas indicam essas direções de priorização e destacam o ponto de inflexão, ou "joelho" da curva, que comumente representa o equilíbrio mais vantajoso entre os dois critérios.

## 4.5 Limitações do Modelo

A abordagem experimental utilizada se baseia no uso de um conjunto dados de rede de uma mesma topologia para executar repetidos experimentos em cada configuração. Essa

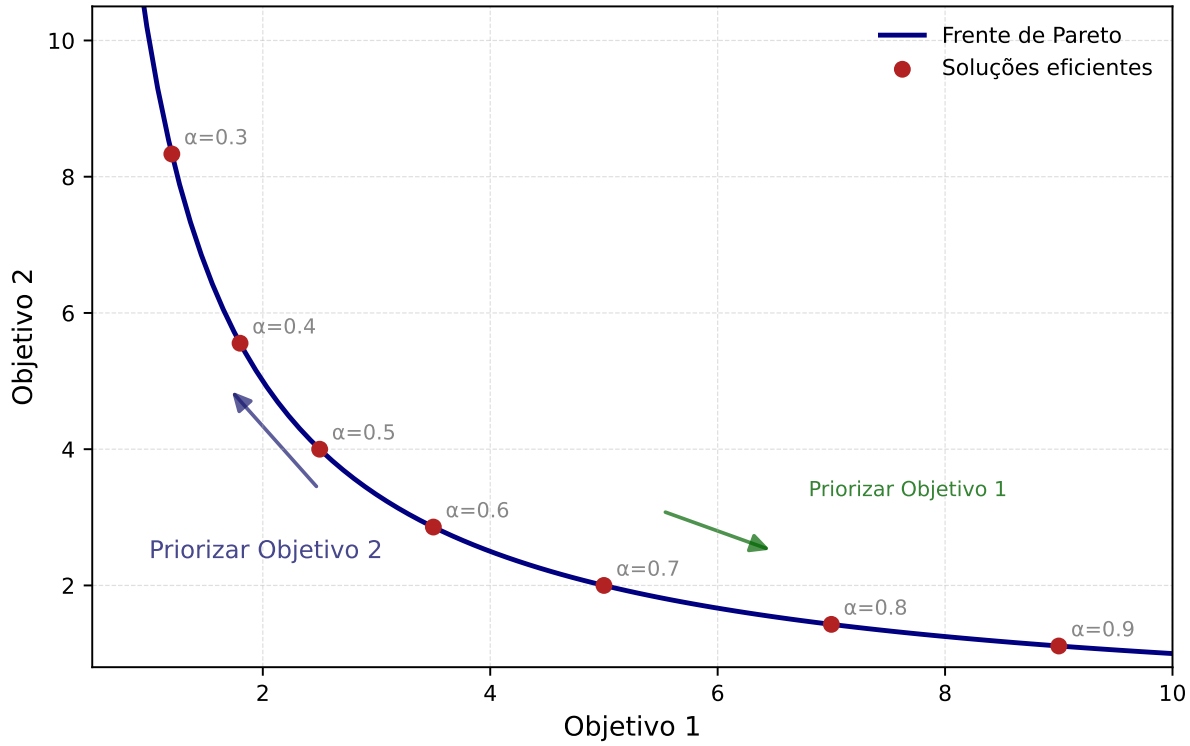



Figura 4.5: Exemplo de uma Frente de Pareto.

estrategia permite obter uma boa estimativa para o número de nós *fog* implantados para cada configuração e uma margem de erro associada com o intervalo de confiança de 90%. No entanto, em muitas soluções as condições da rede viabilizam apenas as piores soluções possíveis, onde são posicionados nós em todos os nós da topologia. Essa abordagem traz uma grande desvantagem para a determinação da latência média da rede, pois nesses casos todos os requisitos de latência cumpridos são relativos ao requisito de latência entre nós *fog* e *cloud*, que são requisitos mais flexíveis, e não os requisitos reais das aplicações. Por tanto, a inviabilidade de certas soluções faz com que as estimativas obtidas para as soluções baseada em requisitos de latência e capacidade da aplicação selecionada não sejam cumpridos totalmente.

Esses problemas trazem então uma piora da estimativa da latência média, o que faz com que a latência média entre muitas soluções de uma mesma topologia e aplicação sejam superiores a latência mínima estabelecida pelos requisitos. Este problema então é uma limitação da abordagem utilizada. A Figura 5.12 ilustra um experimento executado com a topologia GEANT2, com a heurística que é calculada com a excentricidade baseada na latência e mostra como soluções podem não estar definidas dentro do domínio dos requisitos da aplicação de Videoconferência em HD.

É fundamental considerar que, dependendo da topologia e das condições de rede, o posicionamento dos nós *fog* pode não garantir que, mesmo com o maior número possí-

vel de servidores, todos os requisitos sejam atendidos. Isso leva à conclusão de que, em determinadas situações e aplicações, a implantação de nós *fog* não é a única alternativa para atender aos requisitos necessários. Nesse contexto, outras abordagens para a melhoria das condições da rede tornam-se essenciais para atender aos requisitos de latência e capacidade. Dessa forma, a abordagem adotada é capaz de fornecer uma solução que atende aos requisitos estabelecidos, até onde valem as seguintes relações:

$$L_{\theta i} \leq \Theta_i \tag{4.1}$$

$$C_{\theta i} \geq U_i \tag{4.2}$$

# Capítulo 5

# Resultados

Para avaliar o desempenho das heurísticas propostas foram utilizados três conjuntos de dados com diferentes escalas topológicas: NSFNet (14 nós), GEANT2 (24 nós) e SYNTH50 (50 nós), cujas estruturas estão ilustradas na Figura 4.4. A escolha desses *datasets* visa representar cenários com distintos níveis de complexidade, permitindo analisar a eficácia das abordagens tanto em topologias menores e menos conectadas quanto em redes maiores e mais conectadas.

Os experimentos foram realizados em quatro cenários de aplicação com características variadas: IoT Industrial, Videoconferência em HD, Jogos *Online* e *Streaming* de Vídeo 4K. Cada cenário possui requisitos específicos de latência máxima e largura de banda mínima, conforme detalhado na Tabela 4.1. A diversidade desses contextos permite avaliar como as heurísticas se comportam frente a diferentes restrições de qualidade de serviço (QoS), testando sua adaptabilidade em distintos perfis de tráfego e exigências da rede.

Em todos os experimentos, as estimativas para tanto as heurísticas quanto para a solução ótima foram determinadas com base em repetições de experimentos. Alguns testes de consistência foram desenvolvidos para garantir estimativas seguras das soluções. Todas as estimativas foram obtidas a partir de médias de repetições dos experimentos. Além disso, o erro associado aos pontos que caracterizam o desempenho das heurísticas foi estimado com base em um intervalo de confiança de 90

As análises foram desenvolvidas, principalmente, avaliando as heurísticas em comparação com a soluções ótimas determinadas analiticamente pelo modelo de otimização desenvolvido nesse trabalho. Para fim de comparação, a solução ótima foi traçada com base em uma curva de Pareto determinada pelo *trade off* entre o número de nós *fog* e a latência média de toda a rede. A avaliação das heurísticas é feita então avaliando, em diferentes aplicações, a posição das heurísticas em relação a curva de Pareto.

## 5.1 Montagem dos Experimentos

### 5.1.1 Tempo de Execução

Embora a complexidade dos algoritmos das heurísticas cresça com uma ordem de $N^3$, esse aumento não representa uma limitação significativa para o tempo de execução, uma vez que as topologias analisadas neste trabalho não excedem $N = 50$ nós. Dessa forma, o tempo necessário para construir e resolver o modelo permanece baixo em relação à complexidade das topologias estudadas. A Tabela 5.1 apresenta os tempos de execução de cada método ótimo e heurístico, obtidos pela média de 100 execuções, com os erros indicados correspondendo aos intervalos de confiança de 90%. Esses resultados mostram que, na prática, até mesmo a abordagem exata apresenta um tempo de execução suficientemente curto para as dimensões do problema, permitindo a realização de múltiplos experimentos e fornecendo estimativas confiáveis sobre o desempenho dos métodos estudados.

Tabela 5.1: Tempo médio de execução (em segundos) para cada método e topologia.

| **Método** | **nsfnetbw** | **geant2bw** | **synth50bw** |
|---|---|---|---|
| Excentricidade (Lat. max.) | $0.00038 \pm 0.00004$ | $0.00141 \pm 0.00008$ | $0.00686 \pm 0.00051$ |
| Excentricidade (Cap. min.) | $0.00096 \pm 0.00008$ | $0.00576 \pm 0.00020$ | $0.0373 \pm 0.0017$ |
| Conectividade (Lat. max.) | $0.00020 \pm 0.00003$ | $0.00120 \pm 0.00006$ | $0.00298 \pm 0.00023$ |
| Conectividade (Cap. min.) | $0.00022 \pm 0.00003$ | $0.00076 \pm 0.00004$ | $0.00332 \pm 0.00024$ |
| Ótima | $0.0127 \pm 0.0003$ | $0.0299 \pm 0.0004$ | $0.268 \pm 0.009$ |

Mesmo que, nos cenários avaliados, a abordagem exata apresente tempos de execução baixos e, portanto, pareça viável do ponto de vista computacional, é fundamental considerar que modelos baseados em MILP não garantam a convergência da solução, sobretudo quando se aumenta a escala ou se impõem restrições mais rígidas. Assim, torna-se necessário investigar métodos heurísticos que, além de manterem tempo de execução reduzido, priorizem a obtenção de soluções viáveis e permitam equilibrar, de forma consistente, a redução de custos de implantação e a diminuição da latência média da rede, ao mesmo tempo em que respeitam os requisitos de latência e capacidade.

### 5.1.2 Repetições e Análise de Erros

Os primeiros experimentos mostraram uma grande dispersão nos resultados para diferentes dados dentro de um mesmo *dataset*. Como apresentado na Tabela 5.1, tanto o método ótimo quanto as heurísticas obtiveram tempos de execução bastante eficientes. Para cada um dos três *datasets* utilizados, foram avaliadas 500 configurações distintas de latência e

capacidade. Devido ao baixo custo computacional dos algoritmos, tornou-se viável repetir os experimentos diversas vezes para cada configuração, possibilitando o cálculo das médias. Essa estratégia garantiu estimativas mais robustas e precisas, reduzindo o impacto de variações aleatórias nos resultados.

Para determinar a margem de erro experimental na estimativa dos valores computados de número de nós *fog* e latência média da rede, foi utilizado o intervalo de confiança de 90%, no entanto. A margem de erro se tornou excessiva e por conta disso mais experimentos foram realizados a fim de reduzir a margem de erro e permitir uma comparação mais justa entre os diferentes métodos.

Dessa forma, seria necessário determinar o número de experimentos necessários para que a margem de erro se estabilize. Para isso, foi realizado uma experimento com as diferentes topologias para determinar a forma de convergência do intervalo de confiança de 90% a medida com que são realizados mais experimentos para as estimativas de número de nós *fog* necessários e latência média da rede. Para esses experimentos foram então utilizadas as três topologias, aplicando os requisitos da aplicação de IIoT. Os gráficos das Figuras 5.1 e 5.2 ilustram a convergência do erro associado ao intervalo de confiança de 90%.

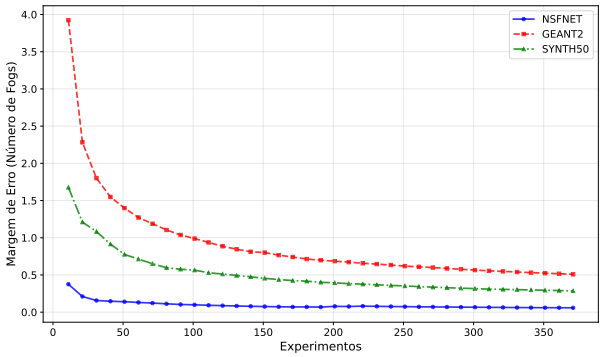



Figura 5.1: Convergência do Erro - Número de *fogs* (Intervalo de Confiança 90%).

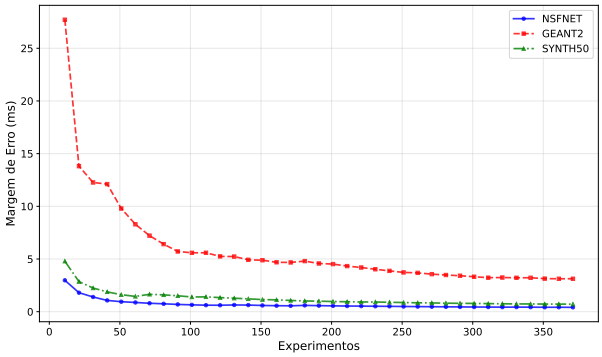



Figura 5.2: Convergência do Erro - Latência Média (Intervalo de Confiança 90%).

A análise das Figuras 5.1 e 5.2 mostra que a margem de erro das soluções se estabiliza quando o número de experimentos ultrapassa aproximadamente 320. Portanto, com base nesse experimento, em todos os métodos analisados, as estimativas médias das soluções foram calculadas a partir da execução de 320 experimentos, garantindo consistência e reduzindo a necessidade de repetições excessivas de experimentos.

## 5.2 Internet das Coisas Industriais (IIoT)

As aplicações de Internet das Coisas Industriais (IIoT) caracterizam-se por operar em ambientes com baixa largura de banda. Todavia, o requisito determinante para sua viabilidade é a latência, dado que tais sistemas dependem de respostas rápidas para manter níveis aceitáveis de confiabilidade e desempenho. Considerando que o limite prático situa-se em torno de 15 ms, o principal objetivo das heurísticas avaliadas consiste na minimização da latência média, mesmo que à custa de um número adicional de nós *fog*.

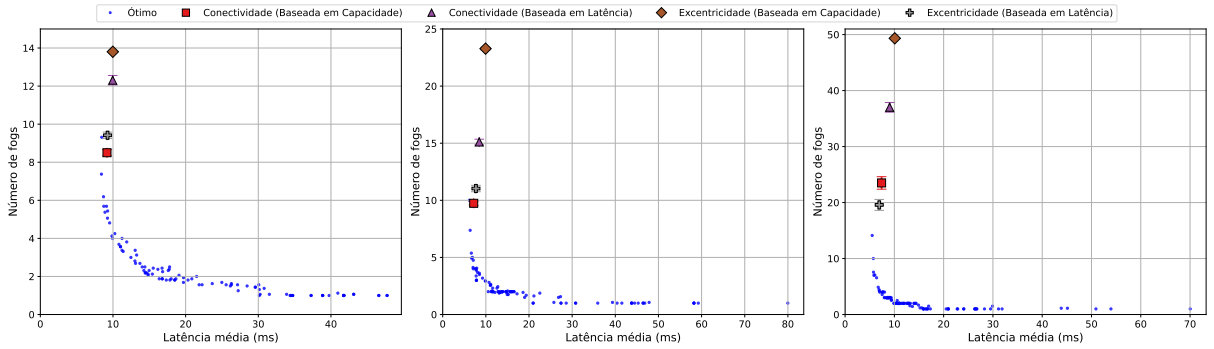



Figura 5.3: Curva de Pareto comparando a solução ótima e os resultados obtidos por heurísticas aplicadas ao problema de posicionamento de nós *fog* em cenários industriais de IoT.

### 5.2.1 Topologia NSFNETBW

Analisando a Figura 5.3.a, nas soluções para a topologia NSFNETBW mantiveram-se em torno de $\approx 10ms$, com o número de nós *fog* implantados variando entre 8 e 14. O algoritmo que apresentou o melhor desempenho foi o de conectividade com base na capacidade, alcançando soluções médias entre 8 e 9 nós *fog*. Em segundo lugar, destacou-se o método de excentricidade baseada na latência, que obteve em média entre 9 e 10 nós *fog* implantados.

Por outro lado, os métodos de conectividade baseada na latência e excentricidade baseada na capacidade apresentaram os piores resultados. As suas soluções oscilaram entre 12 e 14 nós *fog* implantados, oferecendo apenas leve piora na média da latência em

comparação com os demais métodos. Em especial, o método de excentricidade baseada na capacidade resultou em médias próximas de $\approx 14$ nós *fog* implantados, evidenciando que, em quase todos os experimentos, a sua performance aproximou-se da pior solução possível.

### 5.2.2 Topologia GEANT2

Na topologia GEANT2, os resultados obtidos mostraram-se semelhantes aos da topologia NSFNETBW. Nesse cenário, o método de conectividade baseada na capacidade apresentou o melhor desempenho, exigindo em média 10 nós *fog* para alcançar uma latência média de aproximadamente $\approx 8ms$. Em segundo lugar, destacou-se o método de conectividade baseada na latência, que necessitou de cerca de 11,5 nós *fog*, mantendo a mesma latência média de $\approx 8ms$.

Já o método de excentricidade baseada na latência apresentou desempenho intermediário, demandando em torno de 15 nós *fog* para alcançar a mesma latência média obtida pelos demais métodos.

Por fim, o método de excentricidade baseada na capacidade obteve os piores resultados. Apesar de suas soluções atingirem uma latência próxima à dos outros métodos ($\approx 10ms$), o número médio de nós *fog* implantados foi de cerca de 24, mais do que o dobro do requerido pelos dois melhores métodos. Esses resultados evidenciam que, em quase todos os experimentos, tal abordagem configurou-se como a solução menos eficiente.

### 5.2.3 Topologia SYNTH50

Na topologia SYNTH50, os resultados apresentaram um padrão semelhante ao observado nas demais topologias, mas com uma diferença importante: o método de excentricidade baseada na latência obteve o melhor desempenho. Nesse caso, foram necessários aproximadamente 20 nós *fog* para alcançar uma latência média de $\approx 8ms$.

Em seguida, destacou-se o método de conectividade baseada na capacidade, que demandou cerca de 24 nós *fog* para manter a mesma latência média de $\approx 8ms$.

Já o método de conectividade baseada na latência obteve desempenho inferior, exigindo a implantação de aproximadamente 37 nós *fog*, resultando em uma latência média ligeiramente superior, de $\approx 9ms$.

Por fim, o método de excentricidade baseada na capacidade apresentou o pior resultado. Para cumprir os requisitos de latência, foi necessário implantar o número máximo de nós *fog*, configurando-se como a solução menos eficiente entre todas as analisadas.

De modo geral, a análise das três topologias avaliadas evidencia que, embora todas as heurísticas tenham sido capazes de atender ao requisito de latência das aplicações in-

dustriais de IoT (em torno de 15 ms), seu desempenho variou significativamente quanto ao número de nós *fog* necessários. O método de conectividade baseada na capacidade mostrou-se consistentemente eficiente nas topologias NSFNETBW e GEANT2, alcançando latências próximas a 8–10$ms$ com um número relativamente reduzido de nós *fog*. Já na topologia SYNTH50, o melhor desempenho foi obtido pelo método de excentricidade baseada na latência, ainda que com maior demanda de nós em comparação às demais topologias. Por outro lado, o método de excentricidade baseada na capacidade apresentou desempenho insatisfatório em todos os cenários, pois, apesar de manter latências comparáveis às outras heurísticas, exigiu a implantação de um número excessivo de nós *fog*, chegando ao máximo permitido em alguns experimentos.

## 5.3 *Streaming* 4K

As aplicações de *streaming* em resolução 4K apresentam requisitos distintos quando comparadas a cenários de IoT industrial. O fator determinante para a qualidade do serviço é a largura de banda (3.2MBps), necessária para suportar o tráfego de vídeo em alta definição sem perdas perceptíveis. Embora a latência também seja um parâmetro relevante (70$ms$), ela não exerce a mesma criticidade que em aplicações sensíveis ao tempo real. Nesse contexto, a principal heurísticas consiste em dimensionar adequadamente a quantidade de nós *fog*, de forma a garantir baixa latência e alta capacidade de transmissão.

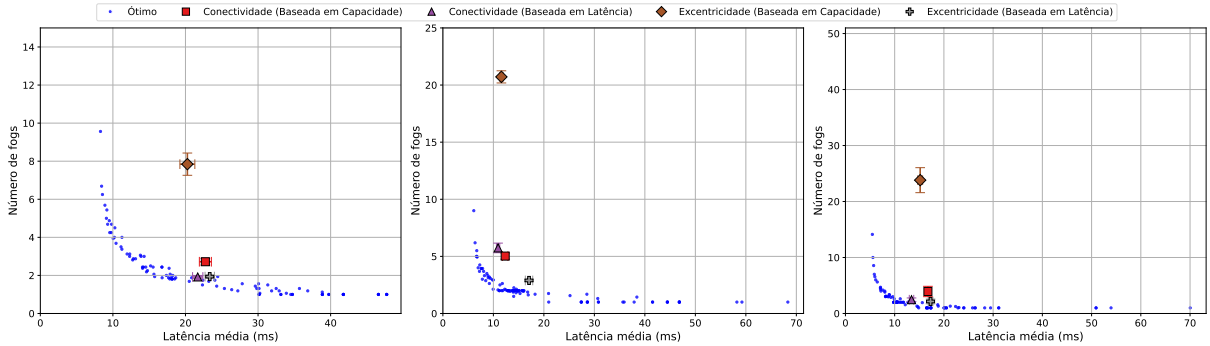



Figura 5.4: Curva de Pareto comparando a solução ótima e os resultados obtidos por heurísticas aplicadas ao problema de posicionamento de nós *fog* em cenários de streaming em 4k.

### 5.3.1 Topologia NSFNET

Analisando a Figura 5.4.a, nas soluções para a topologia NSFNETBW, todas as soluções atenderam aos requisitos da aplicação analisada e apresentaram tendência de agrupamento em pontos próximos à curva de Pareto. Os métodos de excentricidade baseada na latência

e de conectividade baseada na latência e na capacidade obtiveram resultados bastante próximos, tanto em termos de latência média (cerca de $\approx 22ms$) quanto no número de nós *fog* implantados (aproximadamente $\approx 2$). Ainda assim, foi possível distinguir seus desempenhos em relação aos dois eixos de análise. Nesse contexto, o método de conectividade baseada na latência alcançou o melhor resultado: com o mesmo número de nós *fog* implantados, obteve uma latência média ligeiramente inferior à do método de excentricidade baseada na latência e superou a conectividade baseada na capacidade em ambos os critérios avaliados.

Por outro lado, o método de excentricidade baseada na capacidade apresentou desempenho significativamente inferior. Embora tenha proporcionado uma leve melhoria na latência média da rede, demandou a implantação de aproximadamente 8 nós *fog*, ou seja, quatro vezes mais do que a média dos demais métodos, o que o torna a abordagem menos eficiente neste cenário.

### 5.3.2 Topologia GEANT2

Os métodos heurísticos aplicados à topologia GEANT2 apresentaram desempenho próximo à curva de Pareto, mantendo em todos os casos uma latência média inferior ao requisito estabelecido pela aplicação. Os resultados obtidos pelos métodos de excentricidade baseada na latência e de conectividade baseada na latência e na capacidade foram tão semelhantes que apenas um critério de priorização, entre latência média e número de nós *fog*, poderia servir como parâmetro para definir qual deles se destacou.

O método de excentricidade baseada na latência apresentou o melhor resultado em termos de economia de recursos, necessitando apenas de $\approx 3$ nós *fog*. Contudo, sua latência média foi a mais elevada entre os quatro métodos analisados ($\approx 17ms$), embora ainda esteja bem abaixo do limite de $70ms$ imposto pela aplicação. Já o método de conectividade baseada na latência alcançou o melhor desempenho em relação à latência, registrando média de $\approx 11ms$. Entretanto, essa leve vantagem exigiu a implantação de cerca de $\approx 6$ nós *fog*, o que representa um custo de implementação maior.

O método de conectividade baseada na capacidade obteve resultados intermediários, com uma latência média de $\approx 12ms$ e, aproximadamente, $\approx 5$ nós *fog* implantados. Essa configuração se aproxima do chamado joelho da curva de Pareto, por combinar desempenho razoável em ambos os eixos. No entanto, não necessariamente representa a melhor solução, já que a adição de dois nós *fog* extras pode não justificar uma redução de apenas 7% na latência em relação ao requisito que já vinha sendo plenamente atendido por métodos mais econômicos. Esse cenário poderia mudar, caso o limite de latência fosse mais restritivo, contexto em que o método de excentricidade baseada na latência poderia se tornar a opção preferencial.

Por fim, o método de excentricidade baseada na capacidade apresentou o pior desempenho, exigindo aproximadamente $\approx$ 21 nós *fog* para alcançar uma latência média de $\approx 12ms$, configurando-se como a solução menos eficiente entre as analisadas.

### 5.3.3 Topologia SYNTH50

Na topologia SYNTH50BW, caracterizada por elevada complexidade, os métodos de excentricidade baseada na latência e conectividade baseada na latência apresentaram desempenhos praticamente idênticos, indistinguíveis dentro das margens de erro. Ambos necessitaram da implantação de aproximadamente 13 nós *fog*, alcançando uma latência média de $\approx 8ms$. Essas soluções configuraram-se como as mais eficientes, tanto em termos de latência quanto no número de nós *fog* requeridos.

O método de conectividade baseada na capacidade apresentou desempenho intermediário, com latência média levemente superior ($\approx 9ms$), à custa da implantação de cerca de 16 nós *fog*.

Por outro lado, o método de excentricidade baseada na capacidade manteve-se como a abordagem menos eficiente. Em média, foram necessários 49 nós *fog* para alcançar uma latência média de $\approx 10ms$, resultado significativamente inferior ao dos demais métodos.

## 5.4 Jogos *Online*

Os jogos online em tempo real demandam baixa latência para que as ações do jogador sejam refletidas instantaneamente no ambiente virtual, além de uma capacidade mínima de transmissão suficiente para troca de pacotes de dados de forma contínua e confiável. Assim, qualquer solução que ultrapasse 20 ms de latência já compromete significativamente a experiência do usuário.

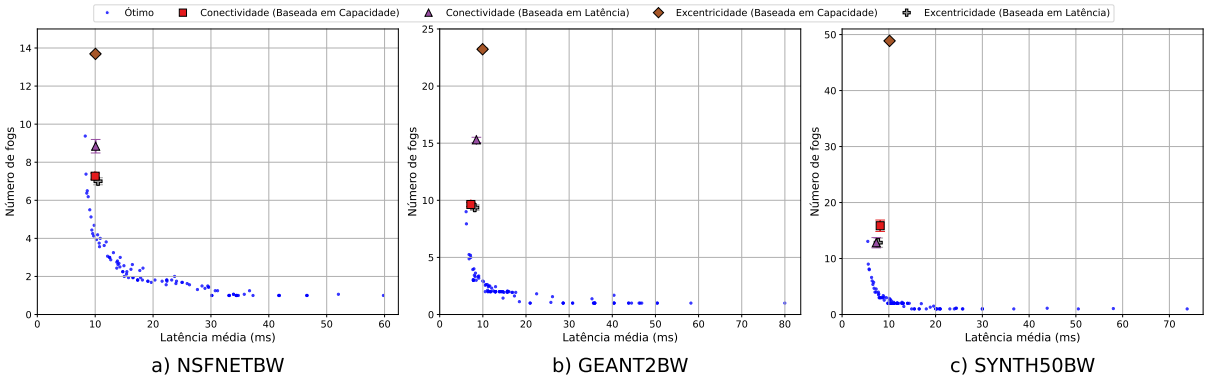



Figura 5.5: Curva de Pareto comparando a solução ótima e os resultados obtidos por heurísticas aplicadas ao problema de posicionamento de nós *fog* em cenários de jogos *online*.

### 5.4.1 Topologia NSFNET

Analisando a Figura 5.5.a, nas soluções para a topologia NSFNETBW, todos os métodos analisados apresentaram uma latência média em torno de $\approx 10ms$, atendendo de forma satisfatória aos requisitos da aplicação. Os métodos de excentricidade baseada na latência e de conectividade baseada na capacidade obtiveram os melhores resultados, alcançando valores muito próximos tanto no número de nós *fog* implantados ($\approx 7$) quanto na latência média.

O método de conectividade baseada na latência apresentou desempenho intermediário, exigindo aproximadamente 9 nós *fog* para atingir a mesma latência média de $\approx 10ms$. Embora não seja a solução mais eficiente, seus resultados permaneceram próximos aos dos melhores métodos.

Por fim, o método de excentricidade baseada na capacidade configurou-se como a pior alternativa, necessitando em média 14 nós *fog* para cumprir os requisitos de latência. Assim, em praticamente todos os experimentos, esse método representou a solução menos eficiente entre as analisadas.

### 5.4.2 Topologia GEANT2

A topologia GEANT2 apresentou resultados bastante semelhantes aos observados na topologia anterior, com a latência média variando entre $\approx 8ms$ e $\approx 10ms$ em todos os métodos. Os melhores desempenhos foram obtidos pelos métodos de excentricidade baseada na latência e de conectividade baseada na capacidade, ambos necessitando de aproximadamente 9 nós *fog* para atender plenamente aos requisitos.

O método de conectividade baseada na latência alcançou desempenho intermediário, demandando cerca de 15 nós *fog* para atingir a mesma faixa de latência. Já o método de excentricidade baseada na capacidade manteve-se como a solução menos eficiente, exigindo em média 24 nós *fog* para cumprir os requisitos estabelecidos.

### 5.4.3 Topologia SYNTH50

Na topologia SYNTH50BW, os resultados apresentaram um comportamento ligeiramente distinto. Os métodos de excentricidade baseada na latência e de conectividade baseada na latência tiveram desempenhos praticamente equivalentes, indistinguíveis dentro da margem de erro das estimativas. Ambos alcançaram uma latência média de, aproximadamente, $\approx 9ms$, com a necessidade de implantar cerca de 13 nós *fog*.

O método de conectividade baseada na capacidade obteve resultados muito próximos, mas ainda assim diferenciáveis. Embora tenha mantido a mesma latência média da rede,

exigiu a implantação de aproximadamente 16 nós *fog*, configurando-se como uma solução ligeiramente menos eficiente em termos de custo de implantação.

Por fim, o método de excentricidade baseada na capacidade permaneceu como a abordagem de pior desempenho. Em média, foi necessário posicionar o número máximo de nós *fog* para atender aos requisitos da aplicação, alcançando uma latência média levemente superior à dos demais métodos ($\approx 10ms$).

## 5.5 Videoconferência em HD

As aplicações de Videoconferência em alta definição possuem requisitos específicos que as diferenciam de outros cenários avaliados. Neste caso, o sistema deve manter a latência abaixo de 100 ms, ao mesmo tempo em que assegura uma capacidade mínima de 1,5 Mbps por conexão. Assim, trata-se de uma aplicação que tolera latências moderadas, mas que exige garantia de banda suficiente para suportar fluxos contínuos de áudio e vídeo em tempo real.

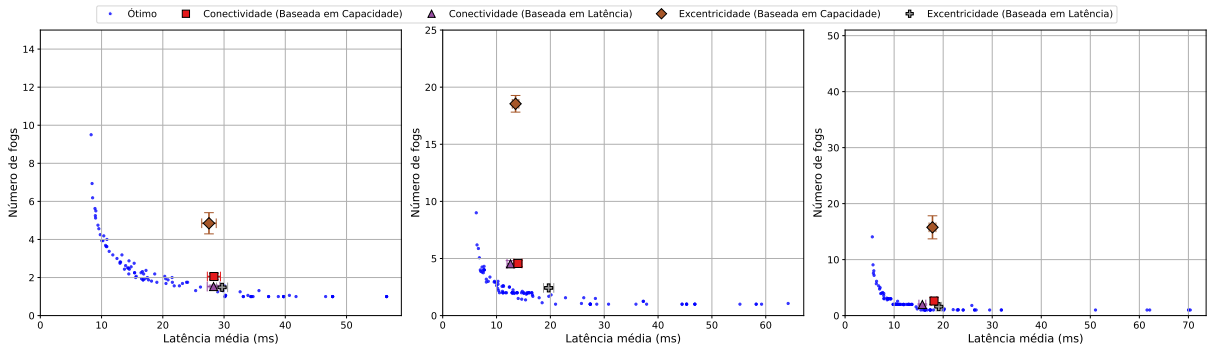



Figura 5.6: Curva de Pareto comparando a solução ótima e os resultados obtidos por heurísticas aplicadas ao problema de posicionamento de nós *fog* em cenários de Videoconferência em HD.

### 5.5.1 Topologia NSFNET

Analisando a Figura 5.6.a, nas soluções para a topologia NSFNETBW, foi possível observar que todos os métodos tiveram, em média, resultados que atendem aos requisitos de latência e ao mesmo tempo tiveram baixa necessidade de implementação de nós *fog*.

Os métodos de excentricidade baseada em latência e conectividade baseada em latência e capacidade tiveram resultados muito próximos e quase indiferenciáveis pelas suas margens de erros. Os métodos de excentricidade baseada em latência e conectividade baseada em latência tiveram os melhores resultados, tendo em média a necessidade de

implementação de $\approx 1.5$ nós *fog* para atender aos requisitos e manter uma latência média de $\approx 28ms$.

O método de excentricidade baseada em capacidade teve a pior solução, mas ainda assim teve necessitou de uma média de 5 nós *fog* para atender aos mesmos requisitos e atingir a mesma latência média.

### 5.5.2 Topologia GEANT2

A topologia GEANT2 apresentou resultados que seguiram paralelos à curva de Pareto. O método de excentricidade baseada em latência obteve o melhor desempenho, necessitando de aproximadamente $\approx 2.5$ nós *fog* para atender aos requisitos e alcançar uma latência média de $\approx 20, ms$.

Os métodos de excentricidade baseada em latência e capacidade apresentaram resultados semelhantes, exigindo cerca de $\approx 4$ nós *fog* para atingir uma latência média de $\approx 13, ms$. Embora representem uma melhora razoável em relação à latência, esses métodos não superam a solução baseada apenas em latência, pois o ganho adicional não compensa o aumento no número de nós *fog*, dado que o requisito de latência já é satisfeito de forma adequada.

Por outro lado, o método de excentricidade em relação à latência apresentou o pior desempenho, demandando aproximadamente $\approx 18$ nós *fog* para alcançar uma latência média de $\approx 13, ms$.

### 5.5.3 Topologia SYNTH50

Na topologia SINTH50 foi possível observar que as soluções tiveram bons desempenhos, mesmo tendo em vista a complexidade da topologia, o que se deve ao requisito de latência de $100ms$. Os métodos de excentricidade baseada na latência e conectividade baseada na capacidade tiveram resultados muito próximos, tendo em média $\approx 2$ nós *fog* necessários para atingir uma latência média de $\approx 18ms$. O método de conectividade baseada em latência teve um resultado próximo, necessitando do mesmo número de nós *fog* ao mesmo tempo que gerou uma latência média de $\approx 16ms$, sendo então a solução com melhor desempenho nesse cenário. A solução de excentricidade baseada em capacidade ainda teve a pior solução, sendo que por mais que tivesse uma latência média de $\approx 18ms$, necessitou de $\approx 16$ nós *fog* para atingir tal desempenho.

## 5.6 Análise Comparativa

### 5.6.1 Análise da Latência

A fim de avaliar de maneira mais precisa o desempenho dos métodos frente ao requisito de latência, foi realizado um experimento no qual se executou todos os métodos em um total de 320 repetições em uma amostragem de requisitos de latência que variaram desde $10ms$ até $200ms$, realizando experimentos com variação de $5ms$ entre cada requisito. Com isso, foi obtido gráfico da Figura 5.7.

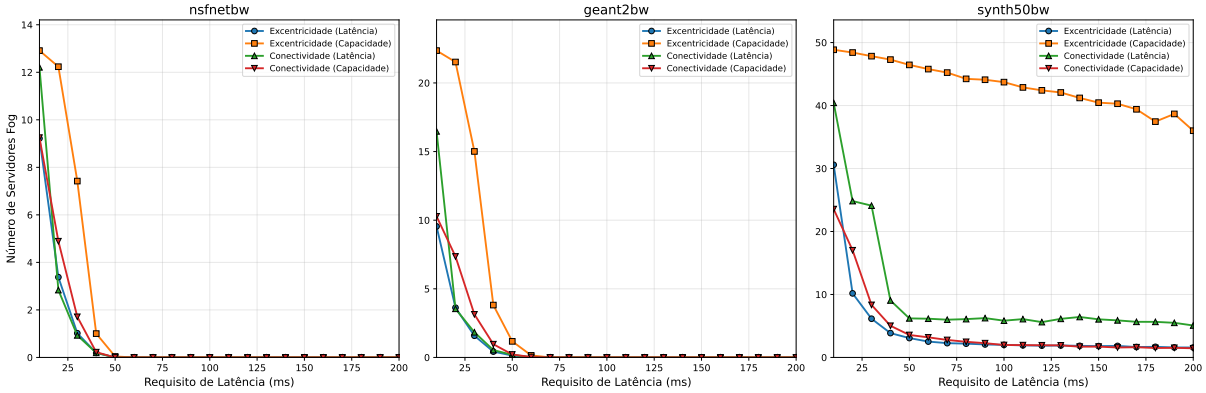



Figura 5.7: Experimento de análise comparativa dos diferentes métodos ao variar o requisito de latência entre $10ms$ e $200ms$.

Analisando a Figura 5.7 é possível observar que, para o requisito de latência superior a $50ms$, nas topologias NSFNET e GEANT2, os métodos mantiveram resultados muito próximos e iguais a solução com nenhum servidor de *fog* implementado. Já na topologia SINTH50 é possível observar que houve uma clara divisão entre os desempenhos dos 4 métodos. Neste cenário o método de de excentricidade baseada na latência teveo melhor desempenho, seguido do método de conectividade baseada na capacidade, conectividade baseada na latência e por fim o pior desempenho se manteve com o método de excentricidade baseada na capacidade.

Abaixo do requisito de $50ms$ é possível observar uma competitividade entre os três melhores métodos. A fim de analisar este intervalo precisamente, foi executado um experimentos variando os requisitos de latência de $10ms$ até $50ms$ com incremento de $1ms$ entre cada experimento. O resultado obtido está ilustrado na Figura 5.8.

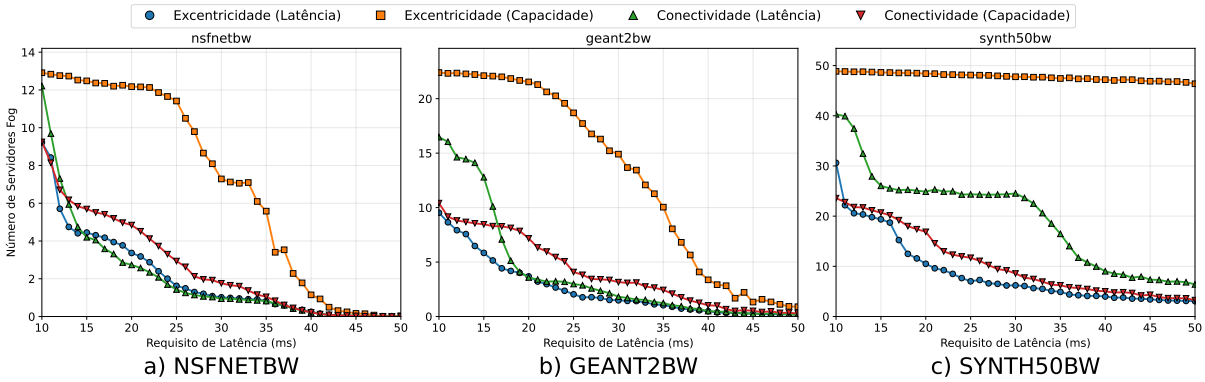



Figura 5.8: Experimento de análise comparativa dos diferentes métodos ao variar o requisito de latência entre $10ms$ e $50ms$.

Na Figura 5.8 é possível observar então que os três melhores métodos tiveram resultados levemente diferentes em certos intervalos do domínio do requisito de latência máxima. Na topologia NSFNET o método de conectividade baseada na latência teve o melhor resultado em quase todo o experimento, tendo uma piora nos resultados em relação aos outros métodos quando o requisito de latência média deduziu para abaixo de $14ms$. O método de excentricidade baseado na latência teve um resultado muito próximo do método anterior, se tornando a melhor solução apenas quando o requisito de latência se torna inferior a $14ms$. O método de conectividade em relação à capacidade foi o método que teve o terceiro melhor desempenho, superando o método de conectividade baseada na latência para o requisito de latência inferior a $13ms$ e se igualando a melhor solução de excentricidade baseada na latência para o requisito de latência inferior a $12ms$. O pior resultado ficou com o método de excentricidade baseada na capacidade, onde se mantede como pior resultado em todo o domínio do requisitos de latência.

Na topologia GEANT2, o método de excentricidade com base na latência tece o melhor desempenho em todo o intervalo. O método de conectividade baseada na latência teve um resultado muito próximo do melhor resultado quando o requisito de latência estava entre $20ms$ e $50ms$, tendo uma piora significativa para os valores inferiores a esse intervalo. O método de conectividade baseada na capacidade se manteve com o terceiro pior resultado durante quase todo o domínio, se tornando melhor que o método de conectividade baseada na latência apenas quando o requisito de latência é inferior a $17ms$. O método de excentricidade baseada na capacidade se manteve com o pior resultado.

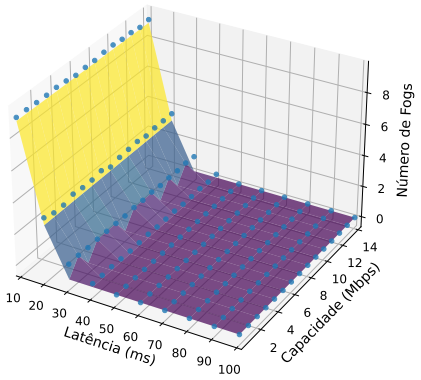
Na topologia SYNTH50 é possível observar uma diferença consistente entre todos os métodos. O método de excentricidade baseada na capacidade se manteve com o melhor resultado, sendo superado pelo método de conectividade baseada na capacidade apenas quando o requisito de latência se reduz a $11ms$. O método de conectividade baseada na latência teve o terceiro melhor resultado, mesmo que distante e apresentou, assim como

nas demais topologias, uma piora considerável nos resultados comforme o requisito de latência se reduz aos seus valores mais críticos. Por fim, o método de excentricidade baseado na capacidade se manteve como o pior resultado

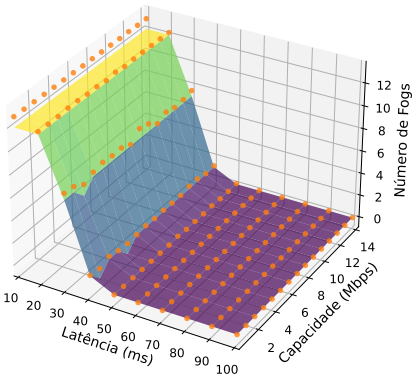
### 5.6.2 Análise da Capacidade

Durante as análises, o estudo buscou analisar o desempenho com base nos requisitos de latência e capacidade, os resultados das análises buscaram analisar preferencialmente o requisito de latência. Essa escolha foi realizada devido ao fato de que os dados coletados com base nas topologias estudadas possuem valores de capacidade que cumprem os requisitos de maneira satisfatória a ponto de não estressar os modelos neste requisito de capacidade. Com isso, quaisquer análises elaboradas para comparar os diferentes métodos com base na capacidade fizeram com que as análises fossem indistinguíveis e por esse motivo usou-se prioritariamente análises voltadas ao requisito latência.

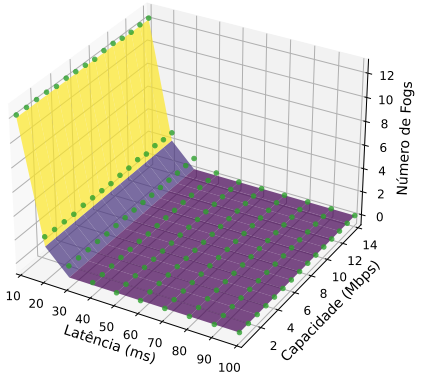
As Figuras 5.9, 5.10 e 5.11 apresentam experimentos realizados com a variação dos requisitos de latência e de capacidade em valores próximos aos intervalos demandados pelas aplicações selecionadas neste trabalho. A latência foi ajustada entre $10, ms$ e $100, ms$, enquanto a capacidade variou de $0.1, MBps$ a $14, MBps$.



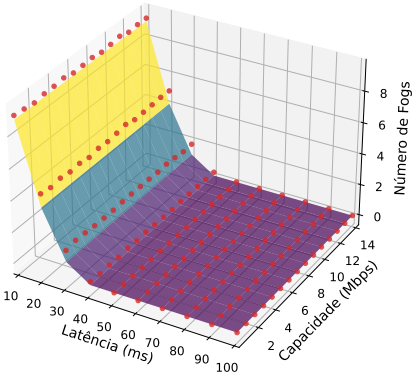

a) Excentricidade baseada em latência




b) Excentricidade baseada em capacidade

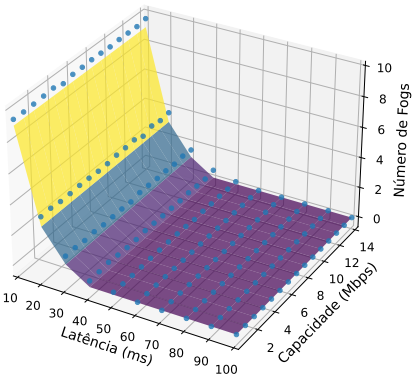



c) Conectividade baseada em latência

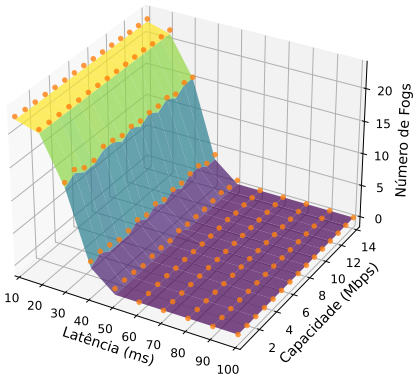



d) Conectividade baseada em capacidade

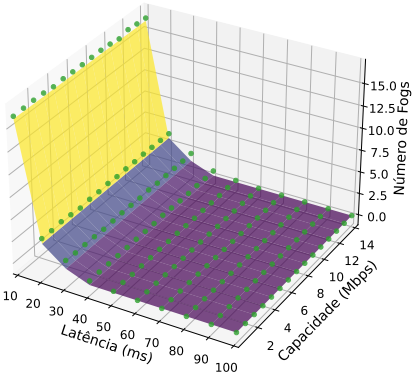
Figura 5.9: Superfície de solução das heurísticas na topologia NSFNET.



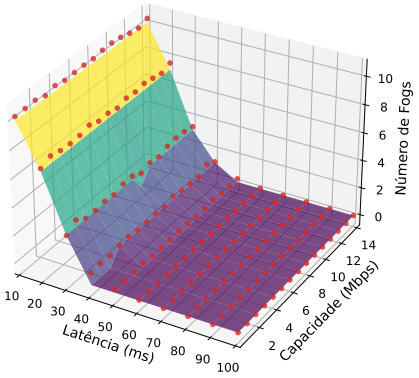

a) Excentricidade baseada em latência




b) Excentricidade baseada em capacidade

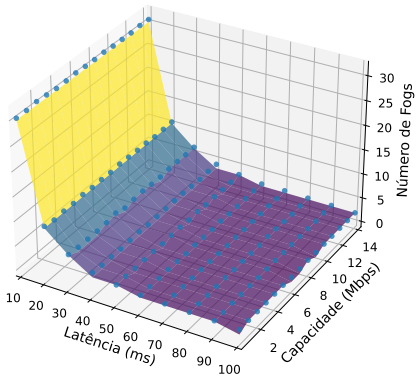



c) Conectividade baseada em latência

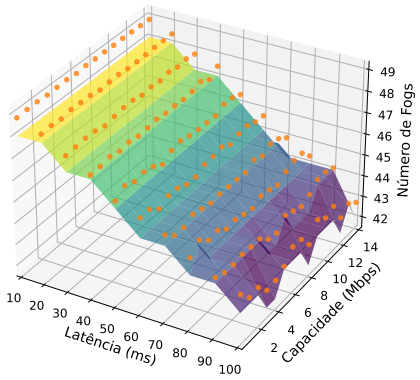



d) Conectividade baseada em capacidade

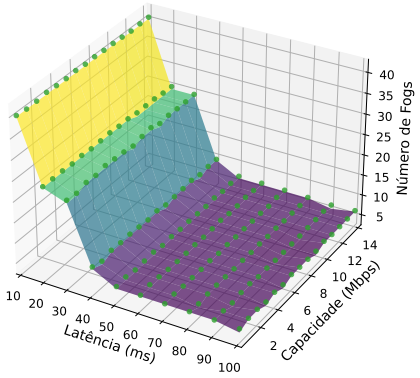
Figura 5.10: Superfície de solução das heurísticas na topologia GEANT2.



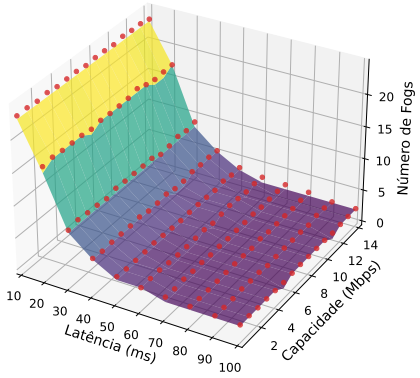

a) Excentricidade baseada em latência




b) Excentricidade baseada em capacidade




c) Conectividade baseada em latência




d) Conectividade baseada em capacidade

Figura 5.11: Superfície de solução das heurísticas na topologia SYNTH50.

Analisando as curvas de nível dessas superfícies em caminhos no qual se varia a capacidade, é possível notar que a variação da capacidade interfere minimamente no número de nós *fog* a serem implantados. Isso se deve ao fato de que, para os *datasets* analisados e para a escala dos requisitos de capacidade das aplicações selecionadas, a capacidade dos *links* de conexão são suficientes para cumprir os requisitos de forma satisfatória e com isso não apresentam nenhum estresse para os modelos de posicionamento de nós *fog*. Diferentemente do que se mostrou o requisito de latência, o qual se mostrou muito mais marcante em relação à tomadas de decisão sobre o posicionamento de nós *fog*.

## 5.7 Limites da Solução

A fim de analisar precisamente a dispersão das soluções de cada método que geram as médias analisadasm executou-se um experimento utilizando o método de excentricidade baseada em latência em um contexto onde, para a aplicação, o requisito de latência é $13ms$

e de capacidade $10MBps$ e para o nós *fog* com a *cloud* o requisito de latência é $25ms$ e de capacidade $2Mbps$. Nesse experimento foram plotados a dispersão dos resultados de 320 experimentos em um plano bidimensional, no qual o eixo vertical é representado pelo número de nós *fog*, e o horizontal é representado pela latência média da rede em cada solução. Para além disso foi utilizado um gráfico de calor para representar a capacidade média da topologia em cada experimento. Esse gráfico está ilustrado na Figura 5.12.

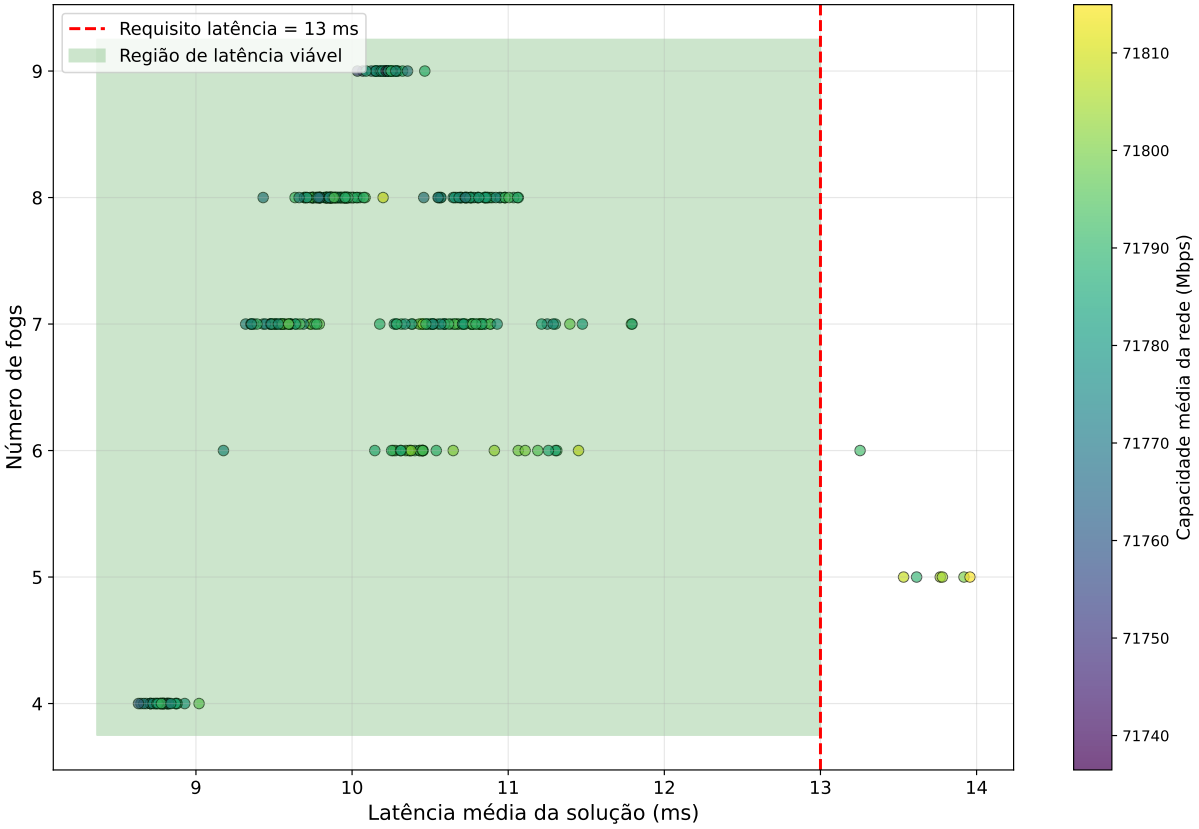



Figura 5.12: Relação das soluções do método de posicionamento de nós *fog* calculado com a excentricidade baseada na latência no contexto de aplicações de jogos *online*.

Esse gráfico evidencia os limites da solução de posicionamento de nós *fog* para o atendimento dos requisitos da rede. Observa-se que, em certos casos, determinadas soluções implementam poucos nós *fog*, mas ainda assim não cumprem o requisito de latência, representado pela faixa vermelha tracejada. Essa limitação decorre diretamente da restrição de latência entre *cloud* e *fog*, que, embora mais flexível que o requisito das aplicações, restringe o conjunto de nós elegíveis para se tornarem nós *fog*.

Esse resultado demonstra que, dependendo dos requisitos da aplicação e da qualidade dos enlaces da topologia analisada, a simples adição de nós *fog* pode não ser uma medida suficiente para garantir o desempenho esperado. Assim, reforça-se a ideia de que a melhoria da qualidade de serviço em redes *cloud* não deve ser pensada exclusivamente em termos

da implementação de nós *fog*, mas deve ser pensada aliada a estratégias complementares de aprimoramento da infraestrutura. Isso inclui, por exemplo, a melhoria da latência e da capacidade entre determinados nós. É importante destacar que outros fatores também devem ser considerados na melhoria da qualidade da rede, uma vez que este trabalho se restringiu à análise dos efeitos dos requisitos de latência e capacidade. Dessa forma, um trabalho mais abrangente envolvendo outros aspectos da rede, como perda de pacotes e congestionamento da rede, podem salientar demais aspectos a se melhorar em uma rede *cloud* a fim de cumprir requisitos mais desafiadores.

Em síntese, observando-se os resultados criados a partir do uso da otimização, foi possível verificar como seriam os diferentes interesses de um operador de rede em relação a uma frente de Pareto. Ao mesmo tempo, em comparação a essa frente, uma estratégia baseada em observar apenas a latência e considerar a maior excentricidade do grafo, mostrou-se extremamente promissora como forma de solução. Sendo que esta estratégia demanda um esforço de monitoramento extremamente baixo para os operadores de rede, pois já pode ser empregada com o simples uso de uma técnica de *traceroute* ou *ping* para cada um dos nós da rede, podendo ser utilizados sem nenhum monitoramento prévio. Além disso, essa técnicas não demandam instalação de novos softwares ou análises profundas da rede, como aquelas que são utilizadas para estimar a capacidade da rede, as quais demandariam colocações de sondas na rede ou instalação de aplicações nos nós para aferição da capacidade, por exemplo, a partir do uso de *iperf*.

# Capítulo 6

# Conclusão e Trabalhos Futuros

Este trabalho avaliou heurísticas baseadas em invariantes de grafos para o posicionamento de nós *fog* em arquiteturas *cloud–fog*, tomando como referência a solução ótima obtida por modelagem analítica ao longo da frente de Pareto entre dois objetivos conflitantes: minimizar o número de nós de *fog* e reduzir a latência média da rede. A avaliação contemplou três topologias (NSFNET, GEANT2 e SYNTH50BW) e quatro perfis de aplicação (IIoT, jogos *on-line*, Videoconferência HD e *streaming* 4K), com estimativas obtidas por repetição e intervalos de confiança de 90%.

De forma consistente, a heurística de excentricidade por latência reduziu o número de nós *fog* necessários sem violar os requisitos temporais, tanto em cenários mais rígidos (IIoT e jogos on-line) quanto nos mais tolerantes (Videoconferência HD e streaming 4K). Esse resultado indica melhor relação entre desempenho e custo de implantação, ao preservar a qualidade de serviço com menor quantidade de infraestrutura.

As heurísticas baseadas em conectividade apresentaram latências estáveis, e em casos específicos se aproximaram ou foram até melhores que as soluções do método de excentricidade baseada em latência. De maneira geral, dentre as soluções calculadas com conectividade, o método baseado em latência teve resultados melhores quando os requisitos foram mais flexíveis e o método baseado em capacidade teve melhores resultados quando o requisito de latência foi inferior a $15ms$. Para além disso, o método de conectividade baseado em capacidade apresentou os melhores resultados em relação a redução da latência média da rede nas aplicações de *Streaming* em $4k$ e em Videoconferência em HD, mesmo que ao custo de alguns nós de *fog* adicionais. O método de excentricidade baseado em capacidade mostrou desempenho sistematicamente inferior, produzindo soluções mais onerosas e sem vantagens em latência em todos os experimentos realizados. Em síntese, entre as estratégias analisadas, a excentricidade por latência foi a alternativa mais equilibrada nos diferentes cenários de aplicação.

Em relação à melhoria da latência média da rede, é importante notar que as heurísticas,

ao contrário do modelo ótimo, não otimizam diretamente a latência média. O uso da frente de Pareto como referência permitiu investigar ganhos diretos com o número de nós de *fog* implementados e ganhos indiretos com a melhoria da latência média da rede. Em todos os experimentos, não se observaram melhorias relevantes de latência média atribuíveis às heurísticas. Ainda assim, a proximidade da excentricidade por latência às regiões eficientes da frente de Pareto, com menor número de nós *fog*, a torna uma candidata preferencial quando o objetivo é equilibrar qualidade de serviço e custo.

A análise da dimensão da capacidade dos experimentos mostrou que, para o experimentos e *datasets* utilizados, a capacidade não se mostrou como um requisito tão impactante quanto ao número necessário de nós de *fog* a serem implantados, como mostrados nas Figuras 5.9, 5.10 e 5.11. No entanto, a utilização de estratégias baseadas na capacidades dos links se mostrou uma estratégia razoável, tanto em relação a invariante de grafo de excentricidade quanto à conectividade. Isso mostra que, mesmo que o requisito de capacidade não tenha sido significativo para o posicionamento dos nós de *fog*, abordagens que levem em conta esse parâmetro da rede podem ser muito benéficas para a redução de custos e melhoria da qualidade da rede. Para além disso, cabe, em trabalhos futuros, o questionamento sobre como a capacidade pode ser ainda mais decisiva no posicionamento de nós de *fog* em contextos onde este requisito seja mais significativo.

Embora as soluções propostas possam não oferecer sempre uma solução dentro dos requisitos do domínio, é importante considerar que, dependendo das topologias e das condições da rede, o posicionamento de nós de *fog* pode não garantir que uma topologia específica seja capaz de atender a todos os requisitos necessários. Isso levanta a questão de que, em determinados contextos e aplicações, a implantação de nós de *fog* pode não ser a solução única capaz de cumprir todos os requisitos exigidos para o funcionamento otimizado da rede. Além disso, a complexidade das redes e as variações nos ambientes de aplicação podem exigir que outras abordagens sejam consideradas, como o aprimoramento de protocolos de comunicação, a utilização de diferentes arquiteturas de rede ou até mesmo melhorias físicas de infraestrutura. Portanto, para garantir o atendimento completo dos requisitos, é essencial adotar uma visão mais holística e integrada, explorando diversas alternativas além do posicionamento de nós de *fog*.

Por fim, esse trabalho mostrou que as heurísticas baseadas em invariantes de grafos, quando utilizadas no contexto de posicionamento de nós de *fog*, podem produzir resultados muito próximos de soluções ótimas. Nesse contexto, as melhores soluções se concentraram na invariante de grafo em relação à latência, onde foram obtidos os melhores resultados em relação a redução do número de nós de *fog* necessários. Os métodos baseados na invariante de grafo de conectividade se mostraram relativamente eficazes, tanto quando o cálculo se tomou em relação à latência quanto em relação à capacidade. Em alguns cenários esses

dois métodos tiveram variações significativas, se distanciando muito da curva de Pareto ou tendo melhor latência média para o mesmo número de nós de *fog* que as melhores soluções. O método de excentricidade em relação a capacidade, no entanto, apresentou resultados que embasam a ideia de que esse método não seria viável em nenhum contexto ( Figura 5.8).

Dessa forma, conclui-se que métodos de posicionamento de nós de *fog* baseados em invariantes de grafos podem constituir uma alternativa viável para soluções otimizadas, destacando-se, em especial, a excentricidade associada à latência. Contudo, em determinados contextos, abordagens baseadas na conectividade, considerando simultaneamente latência e capacidade, podem apresentar resultados próximos. É importante ressaltar, entretanto, que em projetos voltados à melhoria da qualidade de serviço em ambientes de *cloud*, os benefícios do posicionamento de nós de *fog* tendem a diminuir à medida que os requisitos de latência e capacidade se tornam mais rigorosos. Nesses casos, a solução não deve se restringir à simples adição de nós de *fog*, mas também contemplar parâmetros complementares da rede, como melhorias na infraestrutura, nos protocolos de comunicação e na própria arquitetura da rede.

# Anexo I - Algoritmo de Otimização

O código implementa, por meio da biblioteca OR-Tools, o modelo de alocação de servidores de fog descrito nesta dissertação: dado um grafo de latências e capacidades entre nós, determina-se quais vértices da topologia serão promovidos a servidores. Inicialmente, importa-se o módulo `pywraplp` e instancia-se um *solver* de programação inteira mista (CBC). Em seguida, são definidas configurações destinadas a tornar os resultados reprodutíveis (`randomSeed=42`) e a habilitar o log de execução. Na sequência, define-se um fator de escala `uni = 0.025/2`, empregado para converter as latências e respectivos limites (`L_max`, `L_cloud_fog`) para a unidade de medida adotada nos experimentos. Os parâmetros de capacidade mínima cliente–servidor (`C_min`) e cloud–fog (`C_cloud_fog`) são inicializados a partir dos valores de entrada, ao passo que `N = len(latencias)` define o número de nós na topologia.

Na etapa seguinte, o código define as variáveis de decisão do modelo. Primeiramente, constrói-se a matriz booleana `y[i][j]`, em que cada variável indica se o nó $i$ é atendido pelo nó $j$. Em seguida, cria-se o vetor booleano `x[i]`, em que cada posição indica se o nó $i$ é selecionado como servidor (fog ou cloud). A partir dessas variáveis são impostas as restrições do modelo. A primeira restrição assegura que cada nó seja atendido por exatamente um servidor: para cada $i$, a soma das variáveis `y[i][j]` é forçada a ser igual a 1. A segunda impõe que, se um nó $j$ atende algum nó $i$ (`y[i][j] = 1`), então $j$ deve necessariamente ser marcado como servidor (`x[j] = 1`), o que é traduzido pela desigualdade $y\_ij - x\_j \leq 0$. A terceira restrição garante que todo servidor atenda a si próprio, impondo $x\_i - y\_ii = 0$, de modo que `y[i][i]` coincida sempre com `x[i]`.

Na sequência, o código incorpora os requisitos de qualidade de serviço. As restrições de latência cliente–servidor são formuladas de modo que, se `y[i][j] = 1`, a latência `latencias[i][j]` deva ser menor ou igual a `L_max`, uma vez que a desigualdade `latencias[i][j] * y[i][j] <= L_max` somente se torna ativa quando $j$ efetivamente atende $i$. De maneira análoga, as restrições de capacidade cliente–servidor utilizam `C_min * y[i][j] <= capacidades[i][j]`: se $j$ for servidor de $i$, o enlace entre eles deve possuir capacidade pelo menos igual a `C_min`. No que se refere à relação cloud–fog, são impostas duas famílias de restrições: `latencias[cloud][i] * x[i] <= L_cloud_fog`

assegura que somente podem ser selecionados como nós de fog aqueles cuja latência até a cloud seja aceitável, enquanto `C_cloud_fog * x[i] <= capacidades[cloud][i]` garante capacidade mínima suficiente no enlace cloud–fog. Por fim, fixa-se `x[cloud] = 1`, de modo que o nó que representa a cloud seja sempre considerado um servidor ativo.

A função objetivo é definida como uma combinação de dois critérios: o número de servidores e a latência média. Inicialmente, calcula-se `normalizacao_latencia` como a média das latências entre cada nó e a cloud, a qual é utilizada como fator de normalização. Em seguida, para cada nó $i$, atribui-se ao coeficiente de `x[i]` na função objetivo o valor $(1-\alpha)/N$, o que representa a contribuição associada à minimização do número de nós de fog, normalizada pelo número total de nós. Para cada par $(i, j)$, o coeficiente de `y[i][j]` é definido como $\alpha \cdot$ `latencias[i][j]`$/($`normalizacao_latencia`$\cdot N)$, correspondendo à contribuição normalizada da latência média. Dessa forma, para $\alpha \in [0, 1]$, o modelo minimiza $(1-\alpha)\times$(número de servidores)$+\alpha\times$(latência média), refletindo explicitamente o *trade-off* explorado nas curvas de Pareto. Em seguida, especifica-se que o objetivo deve ser minimizado e o *solver* é invocado por meio de `solver.Solve()`.

Uma vez obtida uma solução, o código extrai os valores das variáveis `x[i]` em uma lista `X_solution`, convertendo-os para inteiros (0 ou 1) e, em seguida, marca explicitamente o nó cloud com o valor 2 (`X_solution[cloud] = 2`), a fim de diferenciá-lo visualmente dos nós de fog e dos clientes na etapa de visualização ou análise dos resultados. Para avaliar o desempenho em termos de qualidade de serviço, o algoritmo percorre todas as variáveis `y[i][j]` e, sempre que identifica uma variável "ativa" (valor $>$ 0,5, indicando que $j$ de fato atende $i$), acumula a latência `latencias[i][j]` e incrementa o contador de nós atendidos, calculando, ao final, a latência média observada na solução. Caso, por alguma anomalia, nenhum nó seja atendido e a média resulte igual a 0, esse valor é substituído por `L_cloud_fog` como um valor de referência coerente com a escala do problema. Por fim, a função retorna uma tupla contendo o valor ajustado da função objetivo (`Objective().Value() - 1`, em que o termo `-1` é utilizado para desconsiderar a cloud na contagem de nós de fog), o vetor `X_solution`, que resume o papel de cada nó na topologia, e a latência média calculada.

```python
from ortools.linear_solver import pywraplp

def solveProblem(latencias, capacidades,L_max, C_min, L_cloud_fog, C_cloud_fog,
    cloud, alpha=0.5):
    solver = pywraplp.Solver('Simple_lp_program',
        pywraplp.Solver.CBC_MIXED_INTEGER_PROGRAMMING)

    # Configurações para determinismo no OR-Tools
    solver.SetSolverSpecificParametersAsString('randomSeed=42')
    solver.EnableOutput()  # Habilita a saída de log do solver

    # fator de escala para transformar os valores de latência em segundos
    uni = 0.025/2

    L_max = uni*L_max # 10ms (tempo real) 80ms (jogos) 200ms (IoT)
    L_cloud_fog = uni*L_cloud_fog # 200ms
    C_min = C_min
    C_cloud_fog = C_cloud_fog

    N = len(latencias)

    ## define as variaveis de decisao
    # y representa se um nó i é servido por um nó j
    y = []
    for i in range(N):
        y.append([])
        for j in range(N):
            y[i].append(solver.BoolVar("y_"+str(i)+"_"+str(j)))

    # x representa se um nó i é um servidor
    x = []
    for i in range(N):
        x.append(solver.BoolVar("x_"+str(i)))

    ## define as restricoes

    # cada nó deve ser servido por um servidor
    for i in range(N):
        ct = solver.Constraint(1, 1)
        for j in range(N):
```

```
39              ct.SetCoefficient(y[i][j], 1)
40
41      # se um nó é servido por outro, então o nó que serve deve ser um servidor
42      for i in range(N):
43          for j in range(N):
44              ct = solver.Constraint(-solver.infinity(), 0)
45              ct.SetCoefficient(y[i][j], 1)
46              ct.SetCoefficient(x[j], -1)
47
48      # se o nó é um servidor ele deve ser servidor por ele mesmo
49      # x[i] - y[i][i] = 0 para todo i pertencente a n
50      for i in range(N):
51          ct = solver.Constraint(0, 0)
52          ct.SetCoefficient(x[i], 1)
53          ct.SetCoefficient(y[i][i], -1)
54
55      # a latencia de um nó i para um nó j deve ser menor que L_max
56      for i in range(N):
57          for j in range(N):
58              ct = solver.Constraint(-solver.infinity(), L_max)
59              ct.SetCoefficient(y[i][j], float(latencias[i][j]))
60
61      # a capacidade de um nó i para um nó j deve ser maior que C_min
62      for i in range(N):
63          for j in range(N):
64              ct = solver.Constraint(-solver.infinity(),
                ↪  float(capacidades[i][j]))
65              ct.SetCoefficient(y[i][j], C_min)
66
67      # a latência entre um nó de serivdor e o cloud deve ser menor que
        ↪  L_cloud_fog
68      for i in range(N):
69          ct = solver.Constraint(-solver.infinity(), L_cloud_fog)
70          ct.SetCoefficient(x[i], float(latencias[cloud][i]))
71
72      # a capacidade entre um nó de servidor e o cloud deve ser maior que C_min
73      for i in range(N):
74          ct = solver.Constraint(-solver.infinity(),
            ↪  float(capacidades[cloud][i]))
75          ct.SetCoefficient(x[i], C_cloud_fog)
```

```
76
77      # x[cloud] = 1
78      ct = solver.Constraint(1, 1)
79      ct.SetCoefficient(x[cloud], 1)
80
81      # define a função objetivo com dois critérios: número de fogs e latência
        ↪ média
82      objective = solver.Objective()
83      # define a normalização da latência média como a latência entre cada nó e a
        ↪ cloud
84      normalizacao_latencia = sum([float(latencias[i][cloud]) for i in
        ↪ range(N)])/N
85      for i in range(N):
86          objective.SetCoefficient(x[i], (1-alpha)/N)  # Minimiza número de fog
            ↪ nodes
87          for j in range(N):
88              objective.SetCoefficient(y[i][j], alpha *
                ↪ float(latencias[i][j])/(normalizacao_latencia*N))  # Minimiza
                ↪ latência média
89
90      objective.SetMinimization()
91
92      # resolve o problema
93      solver.Solve()
94
95      # resgata a solução para x[]
96      X_solution = []
97      for i in range(N):
98          X_solution.append(int(x[i].solution_value()))
99      # se i for igual ao cloud coloca o valor 2
100     X_solution[cloud] = 2
101
102     # calcula a latência média da rede com base na solução encontrada
103     total_latencia = 0
104     total_atendidos = 0
105     for i in range(N):
106         for j in range(N):
107             if y[i][j].solution_value() > 0.5:
108                 total_latencia += float(latencias[i][j])
109                 total_atendidos += 1
```

```
110     latencia_media = total_latencia / total_atendidos if total_atendidos > 0
        ↪ else 0
111
112     # se a latência média for 0, então não tem solução e então coloca
        ↪ L_cloud_fog
113     if latencia_media == 0:
114         latencia_media = L_cloud_fog
115         #print("Solução não encontrada, latência média é 0")
116
117     # imprime o resultado
118     return  solver.Objective().Value() -1, X_solution, latencia_media
```

# Anexo II - Artigos Submetidos

Este anexo apresenta informações referentes ao artigo científico derivado desta pesquisa que foi submetida à *IEEE/IFIP Network Operations and Management Symposium (NOMS 2026)*, evento internacional de referência na área de gestão e operação de redes.

## Artigo 1 – An Exploration of Graph Invariant-Based Strategies for Fog Server Placement

**Título:** *An Exploration of Graph Invariant-Based Strategies for Fog Server Placement*

**Autores:** Palton Lima Alves; Gustavo Hermínio Araújo; Lucas Bondan; Marcos F. Caetano; Marcelo Antonio Marotta.

**Afiliações:** Universidade de Brasília (UnB); Rede Nacional de Ensino e Pesquisa (RNP).

**Evento:** *IEEE/IFIP Network Operations and Management Symposium (NOMS 2026)*

**Local e Data:** Roma, Itália, 18–22 de maio de 2026.

**Situação:** Artigo submetido em 10 de novembro de 2025; em processo de avaliação na data de depósito desta dissertação.

**Resumo:** Fog computing tem emergido como uma abordagem promissora para reduzir latência e aumentar a capacidade de transmissão em arquiteturas distribuídas. Entretanto, o posicionamento ótimo de servidores de *fog* permanece um problema desafiador devido ao custo de implantação e às restrições de desempenho. Este trabalho compara um modelo ótimo de Programação Linear Inteira Mista (MILP) com heurísticas baseadas em invariantes de grafos (excentricidade e conectividade). Os resultados experimentais mostram que a heurística baseada em excentricidade de latência atinge desempenho próximo

ao ótimo em 91,7% dos cenários avaliados, apresentando lacunas médias inferiores a 5%. Conclui-se que heurísticas baseadas em invariantes topológicos constituem alternativas eficientes e escaláveis para o planejamento de infraestruturas *fog*.

# Referências


[1] Prajapati, Amit Gyandev, Shankarlal Jayantilal Sharma e Vishal Sahebrao Badgujar: *All about cloud: A systematic survey.* Em *2018 International Conference on Smart City and Emerging Technology (ICSCET)*, páginas 1–6, 2018. 1

[2] Lee, Young Choon, Youngjin Kim, Hyuck Han e Sooyong Kang: *Fine-grained, adaptive resource sharing for real pay-per-use pricing in clouds.* Em *2015 International Conference on Cloud and Autonomic Computing*, páginas 236–243, 2015. 1

[3] Haider, Faisal, Decheng Zhang, Marc St-Hilaire e Christian Makaya: *On the planning and design problem of fog computing networks.* IEEE Transactions on Cloud Computing, 9(2):724–736, 2021. 1, 2, 6, 7, 14, 15

[4] Yadav, Prateek e Subrat Kar: *Efficient content distribution in fog-based cdn: A joint optimization algorithm for fog-node placement and content delivery.* IEEE Internet of Things Journal, 11(9):16578–16590, 2024. 2, 8, 9, 14, 15

[5] Maiti, Prasenjit, Jaya Shukla, Bibhudatta Sahoo e Ashok Kumar Turuk: *Qos-aware fog nodes placement.* Em *2018 4th International Conference on Recent Advances in Information Technology (RAIT)*, páginas 1–6, 2018. 2, 12, 15

[6] Herrera, Juan Luis, Jaime Galán-Jiménez, Luca Foschini, Paolo Bellavista, Javier Berrocal e Juan M. Murillo: *Qos-aware fog node placement for intensive iot applications in sdn-fog scenarios.* IEEE Internet of Things Journal, 9(15):13725–13739, 2022. 2, 9, 10, 14, 15

[7] Khaleel Hamadani, Mustafa e Eugen Borcoci: *Fog nodes placement for d2d networks.* Em *2021 44th International Conference on Telecommunications and Signal Processing (TSP)*, páginas 152–156, 2021. 2, 12, 13, 14, 15

[8] Goldstein, Phil: *What is nsfnet: How this old tech sparked scientific research opportunities*, novembro 2016. `https://fedtechmagazine.com/article/2016/11/nsfnet-served-precursor-internet-helped-spur-scientific-research`, Acessado em: 16 set. 2025. 3, 32, 33

[9] GÉANT Association: *GÉant network*, 2025. `https://network.geant.org/`, Acessado em: 16 set. 2025. 3, 32, 33

[10] Pontes, Diogo Ferreira Thé, Marcos Fagundes Caetano, Geraldo Pereira Rocha Filho, Lisandro Zambenedetti Granville e Marcelo Antonio Marotta: *On the transition of legacy networks to sdn - an analysis on the impact of deployment time, number, and*

*location of controllers*. Em *2021 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, páginas 367–375, 2021. 3, 8, 15, 32

[11] Trabelsi, Mayssa, Nadjib Mohamed Mehdi Bendaoud e Samir Ben Ahmed: *Multi-objective optimization for dynamic service placement strategy for real-time applications in fog infrastructure*. Em *2023 IEEE Symposium on Computers and Communications (ISCC)*, páginas 607–612, 2023. 7, 15

[12] Song, Youmei, Tianyu Wo, Renyu Yang, Qi Shen e Jie Xu: *Joint optimization of cache placement and request routing in unreliable networks*. Journal of Parallel and Distributed Computing, 157:168–178, 2021. 9

[13] Alghamdi, Fatimah, Saoucene Mahfoudh e Ahmed Barnawi: *A novel fog computing based architecture to improve the performance in content delivery networks*. Wireless Communications and Mobile Computing, 2019(1):7864094, 2019.

[14] Li, Yuanzhe e Shangguang Wang: *An energy-aware edge server placement algorithm in mobile edge computing*. Em *2018 IEEE International Conference on Edge Computing (EDGE)*, páginas 66–73, 2018. 9

[15] Stocker, Volker, Georgios Smaragdakis, William Lehr e Steven Bauer: *The growing complexity of content delivery networks: Challenges and implications for the internet ecosystem*. Telecommunications Policy, 41(10):1003–1016, 2017. 9

[16] Firouzi, Farshad, Shiyi Jiang, Krishnendu Chakrabarty, Bahar Farahani, Mahmoud Daneshmand, Jaeseung Song e Kunal Mankodiya: *Fusion of iot, ai, edge–fog–cloud, and blockchain: Challenges, solutions, and a case study in healthcare and medicine*. IEEE Internet of Things Journal, 10(5):3686–3705, 2023. 9

[17] Alharbi, Hatem A. e Mohammad Aldossary: *Energy-efficient edge-fog-cloud architecture for iot-based smart agriculture environment*. IEEE Access, 9:110480–110492, 2021.

[18] Mansouri, Yaser e M Ali Babar: *A review of edge computing: Features and resource virtualization*. Journal of Parallel and Distributed Computing, 150:155–183, 2021. 9

[19] Wang, Juan, Di Li e Yueming Hu: *Fog nodes deployment based on space–time characteristics in smart factory*. IEEE Transactions on Industrial Informatics, 17(5):3534–3543, 2021. 9

[20] Xu, Hansong, Wei Yu, David Griffith e Nada Golmie: *A survey on industrial internet of things: A cyber-physical systems perspective*. IEEE Access, 6:78238–78259, 2018. 10

[21] Bochra, Arichi e Amraoui Asma: *Enhancing application placement in cloud-fog environment based on multicriteria decision making*. Em *2024 3rd International Conference on Advanced Electrical Engineering (ICAEE)*, páginas 1–6, 2024. 11, 15

[22] Bala, Mohammad Irfan e Mohammad Ahsan Chishti: *Offloading in cloud and fog hybrid infrastructure using ifogsim*. Em *2020 10th International Conference on Cloud Computing, Data Science  Engineering (Confluence)*, páginas 421–426, 2020. 12

[23] Sanchez Moya, Fernando, Venkatkumar Venkatasubramanian, Patrick Marsch e Ali Yaver: *D2d mode selection and resource allocation with flexible ul/dl tdd for 5g deployments*. Em *2015 IEEE International Conference on Communication Workshop (ICCW)*, páginas 657–663, 2015. 12

[24] IBM: *IBM ILOG CPLEX Optimization Studio*, 2024. `https://www.ibm.com/products/ilog-cplex-optimization-studio`, acesso em 2025-10-15, Software de otimização de decisão para modelagem e resolução de problemas complexos. 23

[25] Perron, Laurent e Frédéric Didier: *Cp-sat*. `https://developers.google.com/optimization/cp/cp_solver/`. 23, 32

[26] Palton-s: *Posicionamento de servidores de fog*. `https://github.com/Palton-s/Posicionamento-de-servidores-de-fog`, 2025. Repositório de código em Python. Commit 4f1a2f0. Acesso em: 18 set. 2025. 23, 31

[27] Dasari, Rajasekhar e Sanjeet Kumar Nayak: *A game-theoretic approach for cost-effective and reliable task offloading in fog-enabled iot networks*. Em *2025 17th International Conference on COMmunication Systems and NETworks (COMSNETS)*, páginas 817–819, 2025. 24

[28] Harris, Charles R., K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke e Travis E. Oliphant: *Array programming with NumPy*. Nature, 585(7825):357–362, setembro 2020. `https://doi.org/10.1038/s41586-020-2649-2`. 32

[29] team, The pandas development: *pandas-dev/pandas: Pandas*, fevereiro 2020. `https://doi.org/10.5281/zenodo.3509134`. 32

[30] Hunter, J. D.: *Matplotlib: A 2d graphics environment*. Computing in Science & Engineering, 9(3):90–95, 2007. 32

[31] Virtanen, Pauli, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt e SciPy 1.0 Contributors: *SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python*. Nature Methods, 17:261–272, 2020. 32

[32] Hagberg, Aric, Pieter Swart e Daniel S Chult: *Exploring network structure, dynamics, and function using networkx*. Relatório Técnico, Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008. 32

[33] BNN-UPC: *datanetapi*. `https://github.com/BNN-UPC/datanetAPI`, acesso em 2025-09-18. 32, 34

[34] Suárez-Varela, J., S. Carol-Bosch, K. Rusek, P. Almasan, M. Arias, P. Barlet-Ros e A. Cabellos-Aparicio: *Challenging the generalization capabilities of graph neural networks for network modeling (sigcomm 2019 demo) — demo-routenet*. GitHub repository, 2019. `https://github.com/knowledgedefinednetworking/demo-routenet`, BSD-3-Clause license. Acessado em: 16 set. 2025. 33

[35] Abid, Mohamed Amine e Abdelfettah Belghith: *Modeling efficiency of mobile ad hoc networks in omnet++*. Em *2011 4th Joint IFIP Wireless and Mobile Networking Conference (WMNC 2011)*, páginas 1–5, 2011. 33

[36] Ding, Shengyi, Jin Li, Yonghan Wu, Danshi Wang e Min Zhang: *Transnet: A high-accuracy network delay prediction model via transformer and gnn in 6g*. Em *2024 IEEE Wireless Communications and Networking Conference (WCNC)*, páginas 1–6, 2024. 34

[37] Cabellos, A.: *Knowledge-defined networking training datasets*. `https://knowledgedefinednetworking.org/`, 2024. [Online; accessed 19-Dezembro-2024]. 34

[38] Kenitar, Soukaina Bakhat, Mounir Arioua e Mohammed Yahyaoui: *A novel approach of latency and energy efficiency analysis of iiot with sql and nosql databases communication*. IEEE Access, 11:129247–129257, 2023. 37

[39] Santos, Fillipe, Roger Immich e Edmundo Madeira: *Multimedia microservice placement in hierarchical multi-tier cloud-to-fog networks*. Em *2021 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, páginas 1044–1049, 2021. 37

[40] Tsipis, Athanasios, Vasileios Komianos e Konstantinos Oikonomou: *A cloud gaming architecture leveraging fog for dynamic load balancing in cluster-based mmos*. Em *2019 4th South-East Europe Design Automation, Computer Engineering, Computer Networks and Social Media Conference (SEEDA-CECNSM)*, páginas 1–6, 2019. 37, 38