Modelagem e Algoritmos para Computação de Borda e Nuvem em Cidades Inteligentes

Bruno L. R. e Silva, Viviane C. Santos, Alex V. Bastos, Samuel M. A. Araújo

***Resumo*— Este artigo aborda o problema de alocação de recursos em Cidades Inteligentes, para atender às demandas de dispositivos de Internet das Coisas, considerando servidores de computação de borda e em nuvem. Propõe-se um modelo de Programação Linear Inteira, que dá origem a um algoritmo exato, e a um algoritmo guloso. Ambos os algoritmos buscam minimizar os custos de operação. Os experimentos são conduzidos em cenários realistas. São analisados os *trade-offs* entre os algoritmos e a adoção de tecnologias como 4G, 5G e 6G. Constatou-se que o tempo de execução do algoritmo exato é da ordem de minutos, o que inviabiliza sua adoção prática. Por outro lado, o algoritmo guloso, apesar de gerar custos até $\approx 26\%$ maiores, possui uma taxa de rejeição e atrasos de atendimento próximos ao algoritmo exato, e um tempo de execução na ordem de milissegundos, proporcionando uma tomada de decisão rápida, ideal para aplicações de tempo real.**

***Palavras-Chave*— Cidades Inteligentes, Computação de Borda, Computação em Nuvem, Programação Linear Inteira.**

***Abstract*— This article addresses the resource allocation problem in smart cities, aiming to meet the demands of Internet of Things devices, considering edge and cloud computing servers. An Integer Linear Programming model is proposed, which gives rise to an exact algorithm and a greedy algorithm. Both algorithms aim to minimize operational costs. The experiments are conducted in realistic scenarios. The trade-offs between the algorithms and the adoption of technologies such as 4G, 5G, and 6G are analyzed. It was found that the exact algorithm runtime is on the order of minutes, making its practical adoption unfeasible. On the other hand, the greedy algorithm, although generating operational costs up to approximately 26% higher, achieves rejection rates and service delays comparable to the exact algorithm, with runtime on the order of milliseconds, enabling fast decision-making, which is ideal for real-time applications.**

***Palavras-Chave*— Smart Cities, *Edge Computing*, *Cloud Computing*, Integer Linear Programming.**

I. Introdução

O conceito de cidade inteligente emergiu na década de 90, período em que as cidades começaram a implementar tecnologias para enfrentar os desafios urbanos. Desde então, tal conceito tem sido objeto de investigações, com um aumento significativo de estudos na última década [1], [2]. Complementarmente a esse cenário, observa-se um crescimento acelerado da população urbana, cuja proporção passou de 30% em 1950 para 55% em 2018, com projeções de alcançar 68% até 2050 [3]. Ressalta-se que as regiões mais urbanizadas no mundo incluem a América do Norte e a Latina, com, respectivamente, 82% e 81% da população vivendo em cidades. Adicionalmente, projeta-se que, até 2030, o mundo terá 43 megacidades com mais de 10 milhões de habitantes cada [3].

Dado o crescimento urbano, aliado à crescente adoção de soluções digitais no cotidiano das pessoas, tem impulsionado o conceito de Cidades Inteligentes. Tal conceito tem ganhado destaque como uma solução para otimizar a infraestrutura urbana, melhorar a qualidade dos serviços, aumentar a eficiência, reduzir custos e mitigar impactos ambientais, promovendo cidades mais sustentáveis, competitivas e conectadas [4], [5]. Neste meio, a aplicação de tecnologias baseadas na Internet das Coisas (*Internet of Things*, IoT) [6] se torna uma realidade.

A IoT é um paradigma tecnológico que conecta dispositivos à Internet, permitindo a coleta, troca e análise de dados sem intervenção direta humana. Ela se destaca por viabilizar a coleta, análise e transmissão de dados, promovendo a automação de processos e a melhoria da qualidade de vida [7]. Diversas cidades ao redor do mundo já apresentam um alto nível de desenvolvimento de aplicações de IoT, sendo respectivamente, Zurique e Oslo[1] as cidades mais desenvolvidas neste aspecto.

A. Background

A expansão da adoção da IoT nas Cidades Inteligentes gera desafios ligados à alta demanda por largura de banda, baixos atrasos, altas capacidades de processamento e armazenamento, além de questões relacionadas à segurança e privacidade dos dados. Nesse contexto, a integração entre Computação de Borda (*Edge Computing*, EC) e Computação em Nuvem (*Cloud Computing*, CC) surge como uma solução promissora, ao equilibrar o processamento distribuído e o atendimento em tempo quase real [4]. Entretanto, fatores como escalabilidade, custos, complexidade e o cumprimento de métricas de Qualidade de Serviço (*Quality of Service*, QoS) tornam a alocação de recursos um tema central nas pesquisas da área [2], [6].

Apesar dos benefícios, a integração entre EC e CC apresenta desafios consideráveis [7]. Enquanto a EC contribui para a redução de atrasos ao aproximar os recursos computacionais dos dispositivos IoT, a CC assegura escalabilidade e desempenho diante de grandes volumes de dados. Essa integração demanda uma alocação eficiente de recursos, em um baixo tempo computacional, a fim de minimizar custos e atender às necessidades dos usuários em tempo real (ou quase real) [8].

Todos os autores são da Universidade Federal de São João del-Rei (UFSJ) – Ouro Branco, MG, Brasil. Bruno L. R. e Silva e Viviane C. Santos são discentes do curso de Engenharia Mecatrônica ({brunoluizrsilva, vivianecostasantos18}@aluno.ufsj.edu.br). Alex V. Bastos e Samuel M. A. Araújo são professores adjuntos do Departamento de Tecnologia em Engenharia Civil, Computação, Automação, Telemática e Humanidades (DTECH) ({alexvbh, sabreu}@ufsj.edu.br). Os autores agradecem à UFSJ, com menção aos programas PIDAC-Af/UFSJ e PIBIC/UFSJ, que viabilizaram esta pesquisa por meio de bolsas de Iniciação Científica.

[1]Dados do *Institute for Management Development*. *Ranking* das cidades mais inteligentes do mundo. Disponível em: https://abrir.me/Siecp.

Além da infraestrutura, o desempenho adequado das aplicações IoT depende de redes com alta taxa de transferência e baixos atrasos [4]. As Tabelas I e II ilustram algumas características da evolução das tecnologias móveis de rede. A Tabela I mostra a evolução das tecnologias móveis e contextualiza as aplicações que podem ser empregadas em cada tecnologia. Contudo, apesar dos avanços nas tecnologias, ainda existem desafios. O 2G e 3G, por exemplo, apresentam limitações, como baixa taxa de transferência, aspecto restritivo para aplicações modernas. Já o 6G, por sua vez, oferece maior taxa de transferência, mas a redução do alcance limita a cobertura em comparação com tecnologias anteriores.

TABELA I
EVOLUÇÃO DAS APLICAÇÕES

Geração	Aplicações e serviços
1G	Chamadas de voz analógicas, transmissão de voz simples
2G	SMS, chamadas digitais, MMS (Multimedia Messaging Service), e-mails
3G	Navegação web, e-mails, vídeo chamadas, TV móvel, chamadas de vídeo
4G	Streaming HD, redes sociais, jogos, chamadas de vídeo em alta definição
5G	IoT quase em tempo real, automação industrial, Cidades Inteligentes, RA e RV
6G[b]	IoT em tempo real, veículos autônomos, comunicações holográficas

TABELA II
EVOLUÇÃO DAS TAXAS DE TRANSFERÊNCIA[a,c]

Geração	Taxa de transferência	Cobertura
1G	2.4 kbps	20 KM
2G	64 kbps	10 KM
3G	2 Mbps	5 KM
4G	100 Mbps - 1 Gbps	3 KM
5G	1 - 10 Gbps	0.6 KM
6G[b]	100 Gbps - 1 Tbps	0.32 KM

[a] A taxa de transferência e a cobertura são influenciadas por diversos fatores, incluindo frequência e interferências. A versão completa da Tabela e a referência dos valores utilizados estão disponíveis em [9]. [b] Os valores para 6G são teóricos; os valores reais ainda dependem da implementação e da adoção prática no mercado.

B. Problema Abordado e Contribuições

No problema abordado neste artigo, de modo semelhante a [4], [7], as demandas geradas por dispositivos e aplicações de IoT requerem processamento em servidores da infraestrutura de rede. Similarmente a [4], [8], neste trabalho, o processamento das demandas ocorre por meio de uma infraestrutura de rede composta por servidores distribuídos entre as camadas de borda e de nuvem. Os dispositivos IoT conectam-se inicialmente a servidores da EC, com base na disponibilidade de recursos e no raio de cobertura da tecnologia, enviando seus dados para processamento. Caso os servidores de EC não disponham de recursos suficientes ou quando for necessário — ou vantajoso — otimizar métricas como custo, a solicitação pode ser redirecionada para os servidores de CC.

Como contribuição, este trabalho apresenta um modelo em Programação Linear Inteira (*Integer Linear Programming*, ILP) e um algoritmo exato baseado nessa formulação, denominado $\texttt{ILP}_{opt}$. O problema de alocação de recursos abordado possui características da classe NP-difícil [10], levantando a hipótese de que o tempo de resolução do algoritmo exato seja elevado em cenários de maior escala. Para lidar com essa complexidade, propõe-se, adicionalmente, um algoritmo de baixo tempo computacional, denominado *Greedy Resource Allocation* (GRA). Nesse contexto, investiga-se se o GRA pode gerar soluções de qualidade em comparação ao algoritmo exato. Além disso, este trabalho analisa os *trade-offs* entre tempo de execução, custos, atrasos e taxa de aceitação. Os algoritmos propostos têm como objetivo resolver a alocação de recursos, maximizando o número de demandas de IoT atendidas e minimizando os custos operacionais da rede.

Para tal, são realizados experimentos por meio de simulações computacionais baseadas em cenários realistas, com o objetivo de comparar os diferentes algoritmos. Os experimentos consideram a variação na cobertura das tecnologias de rede, abrangendo do 4G ao 6G (Tabela II). O restante do artigo está organizado da seguinte forma: a Seção II revisa os trabalhos da literatura; o modelo é apresentado na Seção III e o algoritmo de GRA na Seção IV; as métricas, os cenários de simulação e os experimentos são abordados na Seção V; e as conclusões, assim como os trabalhos futuros, são discutidos na Seção VI.

II. REFERENCIAL TEÓRICO

Dahmane *et al.* [2] revisam serviços e soluções em Cidades Inteligentes, analisando tecnologias de IoT, EC e CC. Os autores destacam benefícios em áreas como transporte, saúde e segurança, além de desafios relacionados à densidade populacional, custos e gestão de dados. Além disso, identificam lacunas na literatura, incluindo a necessidade de otimização na alocação de recursos e melhorias na eficiência energética, enfatizando a importância da interoperabilidade e de abordagens centradas no usuário. Por outro lado, Zarpellon *et al.* [1] analisam o crescente uso de tecnologias IoT em universidades, impulsionado pela demanda por maior conectividade e automação no meio acadêmico. No entanto, os autores de [1] apontam desafios de infraestrutura, dificuldades na compatibilidade entre dispositivos e problemas no gerenciamento de dados. Para enfrentar essas questões, o artigo propõe uma arquitetura flexível para IoT, permitindo ajustes conforme as necessidades das universidades e promovendo melhorias na conectividade, segurança e integração entre dispositivos.

Khan *et al.* [11] apresentam um levantamento sobre a evolução das pesquisas relacionadas às Cidades Inteligentes, com tecnologias de CC e EC. Os autores destacam problemas do setor e deixam como recomendações para pesquisas o desenvolvimento de modelos eficientes para integração de EC em cenários reais. Aspectos estes abordados neste artigo, que propõe gerar modelos e analisar os *trade-offs* envolvidos nos cenários avaliados. Segundo Khan *et al.*, a adoção de EC em Cidades Inteligentes envolve diversos desafios, sendo um deles abordado neste estudo, mas que não é discutido em trabalhos da literatura [4], [7], tal como a análise de algoritmos em conjunto com as tecnologias de rede emergentes, como o 6G.

Li *et al.*[10] exploram estratégias para alocação de recursos na camada de EC, visando atender às demandas de IoT e maximizar a satisfação do usuário. O estudo aborda um problema de otimização pertencente à classe NP-difícil, similar ao tratado neste artigo, com foco na maximização da satisfação do usuário com base no atraso. Neste trabalho, aspectos relacionados ao atraso também são analisados, considerando comparações entre a evolução das tecnologias de rede, bem como atrasos no processamento e na transmissão de dados.

III. Modelagem Matemática do Problema com ILP

Em um problema geral de ILP, busca-se otimizar uma função objetivo linear sujeita a um conjunto de restrições lineares [12], na qual as variáveis de decisão devem ser inteiras. Neste contexto, para modelar o problema abordado, a Tabela III apresenta os conjuntos, parâmetros e variáveis utilizadas. Em seguida, são descritos os conjuntos de restrições (Equações (1) a (6)) e a função objetivo adotada (Equação (7)).

TABELA III
Conjuntos, Variáveis e Parâmetros do Modelo

	Descrição dos Conjuntos
D	Conjunto de aplicações IoT a serem processadas
E	Conjunto de servidores de EC disponíveis
C	Conjunto de servidores de CC disponíveis
$V = E \cup C$	Conjunto de todos os servidores disponíveis na rede
$A^d \subseteq V$	Conjunto de servidores de EC que cobrem o dispositivo $d \in D$[1], ou de CC que podem ser acessados através da EC
	Descrição das Variáveis
w^d	0 se a aplicação $d \in D$ foi atendida (1, caso contrário)
x_i^d	1 se a aplicação $d \in D$ foi processada no servidor $i \in V$
z_i	1 se o servidor $i \in V$ está ativo
	Descrição dos Parâmetros
c^d	Custo de não atendimento (€) da aplicação $d \in D$
b^d	Demanda de largura de banda (Mbps) da aplicação $d \in D$
m^d	Demanda de memória (GB) da aplicação $d \in D$
p^d	Demanda de processamento (GHz) da aplicação $d \in D$
nc^d	Demanda de número de *cores* pela aplicação $d \in D$
st^d	Demanda de armazenamento (GB) da aplicação $d \in D$
s^d	Quantidade de dados enviados (Mbps) por $d \in D$
OC_i	Custo de operação (€) do servidor $i \in V$
B_i	Largura de banda oferecida (Mbps) pelo servidor $i \in V$
M_i	Oferta de memória (GB) no servidor $i \in V$
P_i	Oferta de processamento (GHz) no servidor $i \in V$
NC_i	Número *cores* disponíveis no servidor $i \in V$
ST_i	Oferta de armazenamento (GB) no servidor $i \in V$
t_i^p	Atraso de processamento (ms) no servidor $i \in V$
t_i^d	Atraso de comunicação (ms) entre $d \in D$ e $i \in V$[1]

[1]Definido pela fórmula de Haversine.
[2]Considera-se o atraso de transmissão e de propagação no meio [13].

$$\sum_{i \in V | i \in A^d} x_i^d = (1 - w^d), \forall d \in D \tag{1}$$

$$\sum_{d \in D | i \in A^d} b^d x_i^d \leq B_i z_i, \forall i \in V \tag{2}$$

$$\sum_{d \in D | i \in A^d} m^d x_i^d \leq M_i z_i, \forall i \in V \tag{3}$$

$$\sum_{d \in D | i \in A^d} nc^d x_i^d \leq NC_i z_i, \forall i \in V \tag{4}$$

$$\sum_{d \in D | i \in A^d} p^d x_i^d \leq P_i z_i, \forall i \in V \tag{5}$$

$$\sum_{d \in D | i \in A^d} st^d x_i^d \leq ST_i z_i, \forall i \in V \tag{6}$$

As Restrições (1) definem o local de atendimento das demandas. Elas garantem que, se uma demanda $d \in D$ for atendida em um servidor $i \in V$, w^d receberá o valor 0. Os conjuntos de Restrições (2) a (6) garantem, respectivamente, que a soma das demandas de recursos, em termos de largura de banda, memória, *cores*, processamento e armazenamento, não ultrapassem as capacidades oferecidas nos servidores.

A função objetivo do modelo minimiza os custos de operação, considerando o custo de ativação dos servidores e o custo associado às demandas não atendidas (Equação (7)). Tal função induz a variável w^d a assumir o valor 0 para que o custo c^d não seja contabilizado, cumprindo aspectos associados a um acordo de nível de serviço por demanda não atendida.

$$\min \left(\sum_{i \in V} OC_i z_i + \sum_{d \in D} w^d c^d \right) \tag{7}$$

IV. Algoritmo *Greedy Resource Allocation* (GRA)

Um algoritmo *greedy* adiciona elementos à solução do problema com base na melhor opção disponível no momento [12]. O algoritmo escolhe a solução localmente ótima, esperando que isso leve a uma solução globalmente ótima. Embora não garanta uma solução ótima global, ele busca encontrar uma boa solução em um tempo computacional baixo. Para o entendimento do algoritmo proposto, adote: DC como o conjunto de dispositivos cobertos por pelo menos um servidor $i \in E$. Considere $d.LC$ como o conjunto de servidores de EC que cobrem o dispositivo d; e $d.lp$ como o servidor específico responsável pelo processamento das demandas $d \in DC$.

Algoritmo 1 *Greedy Resource Allocation* (D, V)

1: $DC \leftarrow \emptyset$
2: **para cada** $d \in D$ **faça**
3: $\quad d.LC \leftarrow$ distancia_de_haversine(d, E)
4: $\quad$ **se** $d.LC > 0$ **então**
5: $\quad\quad d.LC \leftarrow d.lc \cup C$
6: $\quad\quad d.LC \leftarrow$ ordena_por_custo_servidores$(d.LC)$
7: $\quad\quad DC \leftarrow DC \cup \{d\}$
8: $\quad$ **fim se**
9: **fim para**
10: $DC \leftarrow$ ordena_menor_demanda_$s^d(DC)$
11: **para cada** $d \in DC$ **faça**
12: $\quad d.lp \leftarrow$ define_local_processamento$(d.LC, V)$
13: $\quad V \leftarrow$ decrementa_recursos(d, V)
14: **fim para**

O Algoritmo de GRA inicia sua execução definindo DC como vazio. Na linha 2, percorre-se todos os dispositivos $d \in D$, e, para cada dispositivo d, calcula-se a distância entre ele e todos os $i \in E$ por meio da Equação de *Haversine* (linha 3). Dessa forma, os servidores que estão dentro do raio de cobertura são inseridos no respectivo conjunto $d.LC$. Caso exista pelo menos um servidor cobrindo d (linha 4), então $d.LC$ possui uma opção de conexão, processamento e/ou roteamento. Nesse caso, os dados também podem ser enviados para serem processados em servidores de $i \in C$. Assim, C é adicionado ao conjunto $d.LC$ correspondente (linha 5).

Posteriormente, os elementos de $d.LC$ são ordenados de maneira crescente com base no custo dos servidores, que representam potenciais locais de processamento (linha 6). Após, o dispositivo d é incluído no conjunto DC (linha 7). Na linha 10, o conjunto DC é então ordenado, de forma crescente, pela quantidade de dados enviados s^d. Essa abordagem baseia-se na premissa de maximizar a taxa de aceitação, começando a atender primeiro quem solicita menos recursos.

Para cada dispositivo presente em DC, é executada uma tentativa de mapeamento, na qual $d.lp$ é definido. Essa definição é feita pela função *define_local_processamento()* (linha 12). Neste caso, tal função define o local de processamento com base no primeiro dispositivo de menor custo de $d.LC$ com recursos residuais disponíveis para o processamento. Caso não existam opções válidas, com recursos disponíveis em $d.LC$, o local de processamento é definido como nulo, e a demanda não processada. Caso a demanda possa ser processada, os recursos computacionais por ela demandados são decrementados do servidor alocado para o processamento (linha 13).

V. Experimentos Computacionais

Os experimentos foram conduzidos em um computador AMD Ryzen 5 3600, com 32GB DDR4, e sistema operacional Ubuntu 24.04.1 LTS. O simulador e os algoritmos foram implementados em C++. O *solver* CPLEX V12.6.3, parametrizado com uma *thread*, foi utilizado para a execução do algoritmo $\texttt{ILP}_{opt}$. Devido à complexidade do problema, o *solver* foi configurado com um tempo de execução máximo de 20 minutos. Todas as soluções geradas são factíveis.[2] Por se tratarem de algoritmos determinísticos e os cenários serem estáticos, não são necessárias repetições. Cenários dinâmicos serão explorados em trabalhos futuros.

A. Ambiente de simulação

1) Topologia de rede física: de forma similar a [4], a topologia de rede utilizada é inspirada na cidade de Módena, Itália. Ela possui, respectivamente, 100 e 5 servidores de EC e CC. Os servidores de CC estão localizados na cidade de Ohio, Estados Unidos, e podem receber dados de qualquer servidor de EC. Os dados utilizados para parametrizar os servidores de EC e CC estão detalhados em [9]. A cobertura e a taxa de transferência da rede são definidas com valores da Tabela II.

2) Dispositivos de IoT: os valores utilizados para p^d, m^d, st^d e s^d foram retirados de [7]. Os custos associados às demandas e demais parâmetros estão detalhados em [9].

3) Métricas: a) tempo de execução do algoritmo (segundos, s); *b*) taxa de rejeição, dada pela soma dos percentuais de Rejeição por Falta de Cobertura (RFC) e Rejeição por Falta de Recursos (RFR); *c*) custos (Equação 7); e *d*) atraso, definido como o tempo máximo necessário para atender a todas as demandas, considerando os tempos de processamento, propagação e transmissão dos dados [13].

B. Análise de Desempenho

O algoritmo exato, baseado no modelo matemático e denominado $\texttt{ILP}_{opt}$, gera soluções ótimas quando executado até o final, podendo ser utilizado como *baseline* para as comparações. Contudo, devido ao limite de tempo atribuído ao solver, podem ser geradas soluções viáveis, mas não necessariamente ótimas. Os experimentos que resultaram em soluções sem garantias de otimalidade são indicados nos gráficos com um asterisco (*). Em cada gráfico apresentado, observa-se no eixo y a métrica em análise e, no eixo x, a variação do número de dispositivos ($\beta \in \{500, 600, 700\}$) e do algoritmo utilizado ($\lambda \in \{\texttt{ILP}_{opt}, \texttt{GRA}\}$), representados pela tupla $[\beta, \lambda]$.

Ao se analisar o tempo de execução (Figura 1), o algoritmo $\texttt{ILP}_{opt}$ apresenta um tempo elevado e crescente, restringido no limite definido, o que era esperado devido à sua natureza exata e à complexidade exponencial do problema. Contudo, o algoritmo GRA apresenta um tempo constante e baixo, devido à sua baixa complexidade. Em comparação ao $\texttt{ILP}_{opt}$, observa-se que o baixo tempo de execução do GRA (ordem de milissegundos), o torna indicado para uso em cenários reais, onde é necessária uma tomada de decisão rápida.

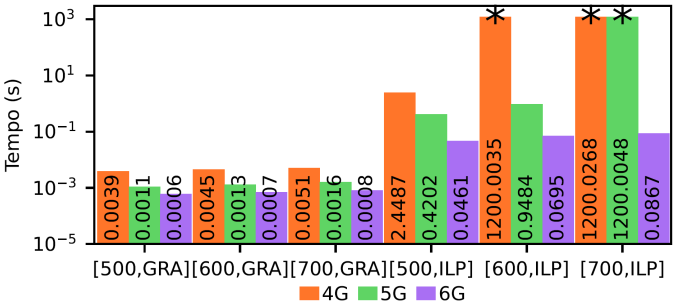

Fig. 1. Tempo de execução dos algoritmos.

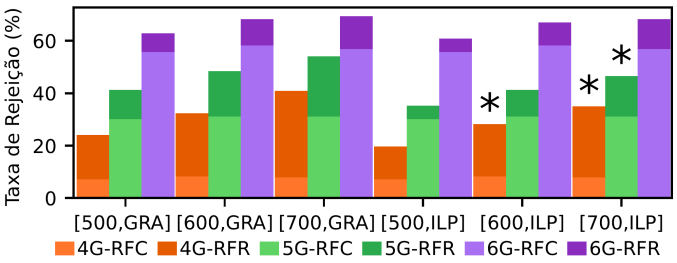

Fig. 2. Taxa de rejeição, estratificada por RFC e RFR.

Comparando o tempo de execução com a taxa de rejeição (Figuras 1 e 2), percebe-se que, ao adotar o 6G, o tempo de execução em todos os experimentos é menor, justamente por haver uma maior taxa de rejeição em relação ao 5G e ao 4G. Nesses casos, há menos servidores aptos a atender às demandas dos dispositivos, devido ao menor raio de cobertura, o que reduz o espaço de solução a ser investigado e, consequentemente, diminui o tempo de execução.

Ainda observando a Figura 2, percebe-se que, com o uso da tecnologia 4G, a taxa de rejeição é menor em relação ao 5G e ao 6G em todos os experimentos. Tal comportamento se deve à baixa cobertura dos dispositivos 6G, que não conseguiram atender às demandas por estarem fora do raio de alcance de comunicação. Nesses casos, a RFC é significativamente mais alta que a RFR. Esse fator indica que a topologia de rede adotada deve ser repensada para aplicações de IoT com tecnologias de comunicação emergentes, como o 5G e o 6G. Observa-se ainda que a taxa de rejeição do algoritmo GRA é próxima ao $\texttt{ILP}_{opt}$, mostrando que o princípio adotado é benéfico em termos de manter uma alta taxa de atendimento.

Ao comparar a taxa de rejeição e os custos (Figuras 2 e 3), percebe-se que, embora em termos de rejeição o algoritmo GRA esteja próximo do $\texttt{ILP}_{opt}$, essa proximidade não se mantém em relação aos custos. Nesse caso, a diferença foi de $\approx 26\%$, considerando o cenário com 500 dispositivos IoT,

[2]Código e instâncias disponíveis em https://abrir.link/JXNHY

e esse comportamento se repete nos demais experimentos. Esse resultado ocorre porque o algoritmo $\texttt{ILP}_{opt}$ explora completamente o espaço de soluções em busca da melhor solução possível (menor custo). Em contrapartida, o algoritmo GRA adota um princípio heurístico promissor para reduzir a rejeição, mas que não se mostrou tão eficaz na minimização dos custos. Observa-se que, apesar do baixo tempo de execução e taxa de rejeição, o GRA pode ser aprimorado para gerar custos menores, beneficiando os provedores envolvidos.

Nos cenários avaliados (Figura 2), observa-se que, ao considerar as tecnologias 6G para conexão dos dispositivos de IoT, o custo aumenta significativamente em relação às outras tecnologias. Esse fator pode ser explicado pela modelagem do problema associada, que impõe um custo, acordado em Acordo de Nível de Serviço, por cada demanda rejeitada (Equação 7). Nesse contexto, quanto maior o número de rejeições (Figura 2), maior o custo gerado pelo não atendimento.

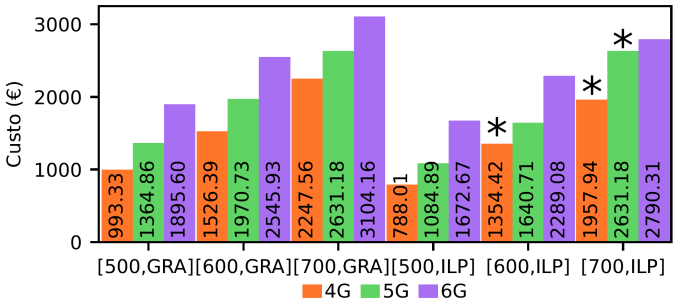

Fig. 3. Custos operacionais.

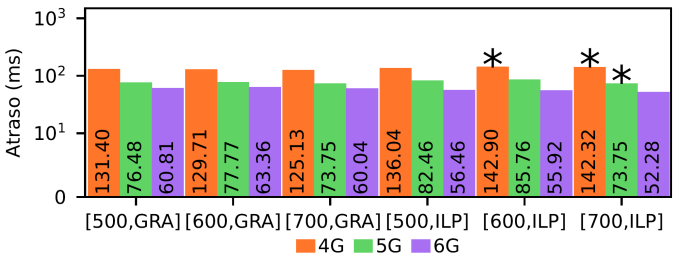

Fig. 4. Atraso de atendimento, considerando processamento, propagação e transmissão dos dados.

Por fim, ao se analisar o atraso de atendimento (Figura 4), observa-se que, em todos os experimentos, esse atraso é maior nas tecnologias 4G, em comparação ao 5G, e, de forma semelhante, o 5G apresenta valores superiores ao 6G. Esse comportamento pode ser atribuído não apenas ao tempo de processamento que depende da disponibilidade de recursos computacionais nas camadas de EC e CC, mas também aos atrasos de propagação e transmissão. A tecnologia 6G apresenta atrasos significativamente menores nesses dois aspectos, em relação às gerações anteriores. Isso se deve, principalmente, à maior taxa de transferência e ao uso de frequências mais elevadas, que permitem uma transmissão de dados mais rápida e eficiente. Dessa forma, o uso do 6G tende a aprimorar a responsividade de sistemas baseados em EC e CC, embora ainda demande melhorias na infraestrutura de rede para garantir uma cobertura adequada, provendo ubiquidade e confiabilidade nas comunicações, reduzindo a RFC (Figura 2).

VI. Conclusões

Este trabalho apresentou um modelo em ILP, que fundamentou o desenvolvimento de um algoritmo exato, denominado $\texttt{ILP}_{opt}$, e de um algoritmo de *greedy*, denominado GRA. Ambos os algoritmos buscam resolver o problema de alocação eficiente de recursos computacionais em Cidades Inteligentes, minimizando custos, e no contexto de estruturas de EC e CC. A partir de simulações em cenários realistas, com diferentes gerações de tecnologias de comunicação móvel (4G, 5G e 6G), foi possível realizar uma análise comparativa dos algoritmos propostos, considerando os principais *trade-offs* entre tempo de execução, taxa de rejeição de demandas, custos operacionais e atraso no atendimento das aplicações.

Os resultados indicam que o algoritmo GRA é promissor para aplicações em tempo real, apresentando tempos de execução significativamente menores e taxas de rejeição comparáveis ao algoritmo exato ($\texttt{ILP}_{opt}$). No entanto, o GRA apresentou maiores custos operacionais, evidenciando a necessidade de melhorias futuras visando este aspecto. Observou-se que o uso de tecnologias emergentes como o 6G, embora beneficie a redução dos atrasos, demanda repensar a infraestrutura de cobertura para garantir eficiência na alocação de recursos.

Como trabalhos futuros, pretende-se explorar aprimoramentos no algoritmo *GRA*, incluindo integrações com metaheurísticas e métodos baseados em aprendizado de máquina para aprimorar a tomada de decisão. Além disso, serão considerados cenários dinâmicos, com mobilidade dos dispositivos.

Referências

[1] B. O. Zarpellon, L. de Oro Arenas, E. Paciência Godoy, F. Pinhabel Marafão, and H. K. Morales Paredes, "Design and implementation of a smart campus flexible internet of things architecture on a brazilian university," *IEEE Access*, vol. 12, pp. 113705–113725, 2024.

[2] W. M. Dahmane, S. Ouchani, and H. Bouarfa, "Smart cities services and solutions: A systematic review," *Data and Info. Management*, 2024.

[3] United Nations: Department of Economic and Social Affairs, "Revision of world urbanization prospects," 2018. Acessado em: 20 mar. 2025, Disponível em: https://abrir.link/CklNA.

[4] T. A. d. Queiroz, C. Canali, M. Iori, and R. Lancellotti, *An Optimization View to the Design of Edge Computing Infrastructures for IoT Applications*, pp. 1–30. Cham: Springer International Publishing, 2022.

[5] V. K. Quy, D. C. Nguyen, D. Van Anh, and N. M. Quy, "Federated learning for green and sustainable 6g iiot applications," *Internet of Things*, vol. 25, p. 101061, 2024.

[6] S. N. Srirama, "A decade of research in fog computing: Relevance, challenges, and future directions," *Software: Practice and Experience*, vol. 54, no. 1, pp. 3–23, 2024.

[7] J. Santos, T. Wauters, B. Volckaert, and F. De Turck, "Towards end-to-end resource provisioning in fog computing over low power wide area networks," *Journal of Net. and Computer Applications*, vol. 175, 2021.

[8] M. Alsabah, M. A. Naser, B. M. Mahmmod, S. H. Abdulhussain, M. R. Eissa, A. Al-Baidhani, N. K. Noordin, S. M. Sait, K. A. Al-Utaibi, and F. Hashim, "6G Wireless Communications Networks: A Comprehensive Survey," *IEEE Access*, vol. 9, 2021.

[9] B. L. R. Silva, V. C. Santos, A. V. Bastos, and S. M. A. Araújo, "Anexos," 2025. Disponível em: https://abrir.link/kwaou.

[10] J. Li, W. Liang, W. Xu, Z. Xu, X. Jia, W. Zhou, and J. Zhao, "Maximizing user service satisfaction for delay-sensitive iot applications in edge computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 5, pp. 1199–1212, 2022.

[11] L. U. Khan, I. Yaqoob, N. H. Tran, S. M. A. Kazmi, T. N. Dang, and C. S. Hong, "Edge-Computing-Enabled Smart Cities: A Comprehensive Survey," *IEEE Internet of Things Journal*, vol. 7, no. 10, 2020.

[12] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. Massachusetts: MIT Press, 3rd ed., 2009.

[13] A. S. Tanenbaum and D. Wetherall, *Computer networks, 5th Edition*. Pearson, 2011.