Algorithm for 5G Resource Management Optimization in Edge Computing

Douglas D. Lieira, Matheus S. Quessada, Andre L. Cristiani and Rodolfo I. Meneguette

***Abstract*—The Internet of Things (IoT) brings new applications and challenges related to cloud computing. The service distribution challenge is becoming more evident and a need for better service options is emerging. The focus of the work is to optimize issues related to the allocation of resources in Edge Computing, improving the quality of service (QoS) with new methodologies. An algorithm based on a bio-inspired model was developed. This algorithm is based on the behavior of gray wolves and it is called Algorithm for 5G Resource Management Optmization in Edge Computing (GROMEC). The algorithm uses the metaheuristic technique applied to Edge Computing, to result in a better allocation resources through user equipment (UE). The resources considered for allocation in that work are processing, memory, time and storage. Two genetic algorithms were used to define the fitness of an Edge in relation to the resource. Two other algorithms that use traditional techniques in the literature, the Best-First and AHP methods, were considered in the evaluation and comparison with the GROMEC. In the function used to calculate fitness during the simulation made with the GROMEC, the proposed algorithm had a lower number of denied services, presented a low number of blocks and was able to meet the largest number of UEs allocating on average up to 50% more in relation to the Best and 5.25% in relation to Nancy.**

***Index Terms*—optimization, resource allocation, 5G, edge computing.**

I. Introdução

O setor de telecomunicações vive uma explosão de avanços que envolve múltiplas áreas tecnológicas, como: a rede celular de quinta geração (5G), a computação em borda e a inteligência artificial. Essas tecnologias, quando trabalham em cooperação, buscando conectar diferentes tipos de dispositivos, recebem o nome de Internet das Coisas, IoT do inglês "Internet of Things". A IoT está cada vez mais presente em nossas atividades diárias, conectando à internet os mais variados tipos de dispositivos físicos presentes ao nosso redor, alterando a maneira como as pessoas se comunicam, utilizam conteúdos e interagindo com o meio ambiente [1]–[3].

O 5G promete fornecer à sociedade um novo tipo de conexão, que pode romper as atuais limitações de tempo e espaço, para criar conexões entre diferentes tipos de dispositivos, sendo a forma de conexão de bilhões de sensores com a internet [4]. Estima-se que cerca de 100 bilhões de dispositivos se comunicarão por meio de redes 5G e isso fará parte de diversos aspectos da vida social das pessoas, como: entretenimento, negócios, educação, cuidados de saúde, redes sociais, entre outros [5]. Devido à esta demanda crescente, cuja previsão é que dobre a cada ano, novos desafios vem surgindo, como: prover maior eficiência espectral, lidar com conectividade massiva e baixa latência [6], [7].

Para isso, os dispositivos utilizam serviços de computação em nuvem, CC do inglês *"Cloud Computing"*, onde é possível armazenar e compartilhar seus recursos. A CC é disponibilizada por provedores, por meio de tecnologias de internet, que mantém uma grande infraestrutura para armazenar e gerenciar recursos dos dispositivos de um modo mais dinâmico, seguro, confiável e econômico [8]. A comunicação dos dados pode ser feita de dispositivo para infraestrutura, D2I do inglês *"device-to-infrastructure"* ou de dispositivo para dispositivo, D2D do inglês *"device-to-device"* [9]. No entanto, com o aumento expressivo do número de dispositivos que utilizam CC, surge a necessidade de conseguir atender toda a demanda com os serviços de modo eficiente e com baixa latência.

Buscando solucionar parcialmente esses problemas, foi introduzido um conceito denominado computação de borda, EC do inglês *"Edge Computing"*, para lidar com os dispositivos de uso intensivo de recursos computacionais. A EC fornece uma camada adicional de infraestrutura que distribui parte das máquinas virtuais dos *data centers* para as extremidades (*edges*) da rede, desta forma, o poder computacional é parcialmente alocado nas bordas da rede, permitindo que os recursos sejam processados em pontos mais próximos dos equipamentos dos usuários, UE do inglês *"User Equipament"* [10]. De acordo com [11], os cinco principais objetivos da EC são: i) melhorar o gerenciamento de dados para lidar com grandes quantidades de dados potencialmente sensíveis a atrasos gerados por UEs, que devem ser processados em tempo real; ii) melhorar a qualidade de serviço para atender a vários tipos de requisitos rigorosos de qualidade de serviço que buscam melhorar a qualidade da experiência [12], ajudando a oferecer suporte a aplicativos de próxima geração, como aplicativos altamente interativos e serviços sob demanda; iii) prever a demanda de rede para estimar os recursos necessários para atender à demanda de rede em uma proximidade local, fornecendo uma alocação de recursos ideal suprir esta demanda, ajudando a decidir se a demanda será tratada localmente, na borda ou na nuvem; iv) gerenciar e ter conhecimento das localizações das *edges* para habilitar geograficamente servidores de ponta distribuídos para induzir suas localizações e rastrear UEs para fornecer suporte com base em serviços de localização, auxiliando na busca de informações sobre pontos de interesse próximos, e v) melhorar a gestão de recursos para otimizar o uso de recursos de rede devido à quantidade limitada de recursos de rede disponíveis nas bordas das nuvens em comparação com a nuvem, melhorando o desempenho da rede.

Embora esta metodologia traga diversos ganhos quanto à latência, ela deu origem a um novo desafio, que é lidar com a

Douglas D. Lieira, Universidade Estadual Paulista, São José do Rio Preto, São Paulo, Brasil, e-mail: (douglas.lieira@unesp.br)

Matheus S. Quessada, Universidade Estadual Paulista, São José do Rio Preto, São Paulo, Brasil, e-mail: (matheus.quessada@unesp.br)

Andre L. Cristiani, Universidade Federal de São Carlos, São Carlos, São Paulo, Brasil, e-mail: (andre.cristiani@estudante.ufscar.br)

Rodolfo I. Meneguette, Universidade de São Paulo, São Carlos, São Paulo, Brasil, e-mail: (meneguette@icmc.usp)

alocação dos recursos disponíveis nas bordas da rede [13]. As *edges* devem ser equipadas com mecanismos capazes de prover um gerenciamento e compartilhamento eficiente de recursos para alcançar uma alocação de recursos globalmente eficaz [14].

Na prática, problemas de otimização são complexos e exigem muito esforço computacional para serem resolvidos. Métodos clássicos de resolução exata encontram soluções ideais para problemas de otimização, porém, podem ser inúteis em cenários do mundo real por conta do tempo computacional que levam para chegar ao resultado. Neste caso, técnicas meta-heurísticas podem resolver problemas de otimização eficientemente e com baixo custo computacional, encontrando soluções de boa qualidade em um baixo tempo de execução [15].

Diante disso, este artigo apresenta um algoritmo para otimização de gerenciamento de recursos 5G em EC denominado GROMEC (*5**G** **R**es**O**urce **M**anagement Optmization in **E**dge **C**omputing*). Nossa proposta é baseada no algoritmo meta-heurístico Otimização do Lobo Cinzento, GWO do inglês *Grey Wolf Optimizator* para selecionar o melhor *edge* para processar a requisição de um UE. Por meio de simulações, nossa abordagem foi comparada com outras técnicas que realizam alocação de recursos em EC e demonstrou resultados promissores.

Dessa forma, o GROMEC apresenta como principais contribuições:

1) Um mecanismo para otimização no gerenciamento do processo de alocação de recursos em uma infraestrutura de computação em borda para maximizar o atendimento dos usuários;
2) A aplicação de uma técnica meta-heurística para adaptar o algoritmo de decisão de alocação ao cenário real;
3) Experimentos para simular o processo de alocação, onde o GROMEC mostrou ser mais eficiente no atendimento dos dispositivos que solicitaram acesso aos recursos da rede, melhorando a experiência do usuário.

O restante do artigo está organizado da seguinte maneira: A Seção II apresenta os principais trabalhos relacionados com essa pesquisa. A Seção III apresenta o modelo do sistema e descreve detalhadamente o algoritmo proposto. A Seção IV apresenta o cenário, a metodologia adotada para avaliar o algoritmo e os resultados obtidos. Finalmente, a Seção V as conclusões e direcionamentos para trabalhos futuros.

II. Trabalhos relacionados

Nesta seção listamos alguns trabalhos recentes que apresentam técnicas de alocação de recursos para redes 5G e também apresentamos como os algoritmos meta-heurísticos estão sendo utilizados em serviços de redes.

Em [4], os autores propuseram um esquema para minimizar o consumo de energia em um sistema de descarga em MEC (*Mobile Edge computing*), buscando otimizar as decisões de descarregamento e a alocação de recursos de rádio, onde o custo da energia de computação e da transmissão foram levados em consideração para as decisões. Em [16], foi proposto um modelo matemático baseado em QoE para analisar o impacto na estimativa de recursos em *Fogs Computing*, considerando aplicações móveis 5G voltadas ao setor de internet das coisas industrial, IIoT do inglês *"Industrial Internet of Things"*.

Os algoritmos meta-heurísticos demonstram ser eficientes quando aplicados para solucionar diversos problemas em redes 5G. No trabalho de [17], utilizou-se o algoritmo meta-heurístico Jaya para fazer um híbrido com o algoritmo de evolução diferencial auto-adaptativa jDE, criando o Jaya-jDE, para reduzir o consumo de energia respondendo à demanda de taxa de bits instantânea pelo usuário. A solução demonstrou ser eficiente quando comparada a outros algoritmos no consumo de energia para configurações de redes 4G e, principalmente, 5G.

Em [18] é apresentado um algoritmo meta-heurístico para reduzir o custo computacional, tempo de conclusão e consumo de energia em ECs, utilizando a otimização por enxame de partículas, denominado JROPSO. O algoritmo é inicializado com soluções candidatas aleatórias (partículas). A cada iteração, a posição das partículas no enxame muda dentro do espaço de busca afim de encontrar a melhor solução de acordo com a melhor posição individual e de todas as partículas. As simulações realizadas pelos autores confirmaram a eficiência de métodos meta-heurísticos para problemas de otimização quando comparado a outros métodos.

Seguindo a mesma linha, [19] apresenta um algoritmo meta-heurístico para escalonamento de tarefas em ECs, baseado no algoritmo de otimização GWO, denominado IBGWO. Os resultados demonstraram que o IBGWO obteve resultados superiores ao algoritmo de morcego binário e a otimização de enxame de partículas, sendo esses dois outros algoritmos meta-heurísticos disponíveis na literatura.

Os autores em [20] também buscaram minimizar o consumo de energia durante o processamento em agendamento de tarefas em serviços de computação em nuvem. Para isso, desenvolveram um algoritmo híbrido de Otimização e Cultura de Ervas Daninhas Invasivas (IWO-CA). A solução apresentou resultados melhores quando comparados com algoritmos heurísticos e outros tradicionais.

Em [21], os autores propuseram um sistema de gerenciamento de replicação dinâmico, um planejador específico de baixo custo para colocação de dados, um sistema de verificação e algumas ferramentas de segurança de dados para computação de borda colaborativa e sistemas de computação em nuvem. Consideraram problemas como dependência de tarefas, confiabilidade no compartilhamento de dados, agendamento de dados para fluxo de trabalho como um problema de computação inteiro. Por fim, propuseram também um algoritmo meta-heurístico mais rápido para solução desse problema. O algoritmo proposto apresentou uma performance superior a outros considerados tradicionais, o qual é capaz de reduzir os custos totais de acesso a dados.

A Tabela I descreve os principais trabalhos da literatura. Alguns trabalhos utilizam algoritmos matemáticos considerados complexos [4] [16], outros apresentam a eficiência da utilização de algoritmos meta-heurísticos (indicados na tabela como meta) em situações aplicadas à computação em nuvem [17]–[21], mas se limitando a alguns recursos específicos da computação em nuvem, como o consumo de energia em Fogs ou ECs. A EC tem a responsabilidade de disponibilizar os recursos de um servidor de rede mais próximos do usuário, com o objetivo principal de diminuir o tempo de respostas durante a comunicação entre os dispositivos e a nuvem. No entanto, uma desvantagem encontrada em uma *Edge* é que, ao deslocar os serviços em pontos mais próximos aos usuários,

a nuvem fica com menos recursos. Neste artigo, propomos o GROMEC, um algoritmo de otimização para melhorar a alocação dos recursos dos UEs nas ECs. O GROMEC foi desenvolvido com base no algoritmo meta-heurístico GWO, buscando alocar os recursos nas ECs eficientemente e com baixo custo computacional.

TABELA I
COMPARAÇÃO DOS TRABALHOS RELACIONADOS.

Trabalhos	Método	Estrutura	Problema
[4]	heurístico	MEC	Consumo de Energia
[16]	prioridade	Fog	Estimativa de recursos
[17]	híbrido (Jaya-jDE)	Edge	Consumo de energia
[18]	meta (PSO)	Edge	Consumo de energia
[19]	meta (GWO)	Edge	Escalonamento de tarefas
[20]	meta (IWO-CA)	Edge	Consumo de energia
[21]	meta (PSO)	Edge	Custos de acesso aos dados
GROMEC	meta (GWO)	Edge	Alocação de recursos

TABELA II
LISTA DE NOTAÇÕES IMPORTANTES

Termo	Descrição
EC	Computação em Borda (Edge Computing)
UE	Equipamentos de usuários (User Equipment)
i	Identificação dos UEs
MEC	Computação de borda móvel
RSU	Roadside Unit
AHP	Analytic Hierarchy Process
$proc$	Capacidade de processamento
arm	Capacidade de armazenamento
tem	Tempo de utilização dos recursos
mem	Memória disponível
λ_x	Taxa de distribuição Poisson
ECD	Distância da EC
V	Valor dos recursos da EC
V_d	Valor dos recursos do UE
$coef1, coef2$	Coeficientes para movimentação
a	Iteração para aproximar do melhor fitness
$UE\_Entrada$	Entrada de dispositivo do usuário
UE_Saida	Saída de dispositivo do usuário
$conj\_recursos$	Conjunto de recursos
$Func\_Escolhe\_EC$	Função que Escolhe Edge Computing
$UE\_Recursos$	Recursos do dispositivo do usuário
$EC\_Recursos$	Recursos da EC
$provEC$	Provável EC

III. PROBLEMA

Nesta seção é descrita a solução proposta para o problema de gerenciamento de recursos em EC, apresentando o modelo de sistema, a técnica meta-heurística utilizada e o algoritmo desenvolvido.

A. Modelo de Sistema

A Fig. 1 ilustra um exemplo de cenário onde o GROMEC pode operar. Neste exemplo, foi considerado que a região possui 4 RSUs (*Roadside Units*) instalados que realizam a comunicação entre os UEs e os serviços de EC, através de rede sem fio, utilizando comunicação 5G. As *edges* recebem solicitações dos UEs e verificam a disponibilidade de alocação dos recursos. A Tab. II apresenta as principais notações utilizadas.

Considerou-se que as *Edges* são compostas por um conjunto de equipamentos computacionais com capacidade de gerenciar e atender os serviços requisitados pelos UEs. Os UEs são dispositivos IoT, tais como usuários portando

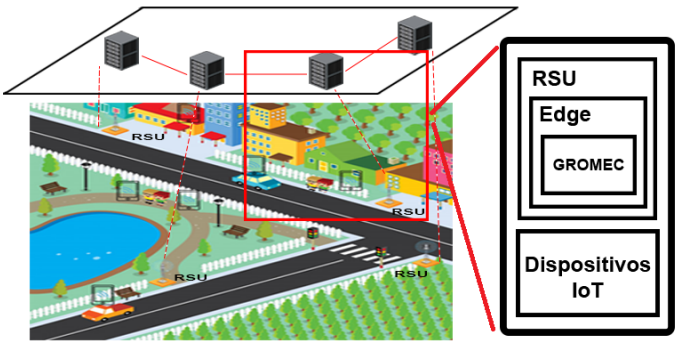


Fig. 1. Arquitetura de sistema do GROMEC.

dispositivos móveis, veículos conectados à internet, sensores, entre outros, que são denotados como UE_i, tendo cada UE uma identificação individual ($i \in [1, n]$). Além disso, cada UE possui um conjunto de recursos $UE_i = (mem_i, arm_i, tempo_i, proc_i)$, que correspondem aos recursos de capacidade de memória, processamento, tempo e armazenamento. Quando uma *Edges* recebe uma requisição de um dispositivo, o algoritmo GROMEC é executado para selecionar a EC que irá atender o serviço. Além disso, é utilizada a distribuição de *Poisson* para as entradas e saídas de dispositivos nos serviços das *Edges*, com uma taxa de λ_x, para que a simulação seja executada de forma aleatória.

B. Algoritmo Meta-Heurístico

Métodos meta-heurísticos são técnicas para problemas de otimização que se inspiram em fenômenos da natureza, como comportamentos de animais ou conceitos evolutivos [22], [23]. Em ciência da computação e otimização matemática, essas técnicas são definidas como métodos de alto nível para encontrar, gerar ou selecionar uma heurística, que pode proporcionar resultados satisfatórios e otimizados [24]. Uma das vantagens de utilizar algoritmos inspirados na natureza é sua simplicidade, que auxilia outros pesquisadores a entenderem mais rapidamente técnicas meta-heurísticas e aplica-las em seus problemas [23].

O algoritmo meta-heurístico de otimização que será adaptado ao cenário de alocação de recursos em ECs é o algoritmo de Otimização do Lobo Cinzento (GWO de *Grey Wolf Optimizer*). Esse algoritmo foi utilizado por ter uma metodologia semelhante ao cenário considerado para este trabalho, onde tem-se um conjunto de ECs e é preciso selecionar qual é a melhor EC para alocar o dispositivo que está requisitando serviços. Assim, faz-se uma analogia, onde as ECs representam os lobos. Assim como a alcateia dos lobos cinzentos possui um líder, é necessário selecionar qual é a EC representará o alpha e será responsável por alocar os recursos do UE que está

realizando a solicitação. A seguir, é descrito como é feito o processo de escolha da melhor EC.

A alcateia dos lobos cinzentos possui uma hierarquia social predominante. Essa hierarquia, definida em quatro níveis, é liderada pelo alpha α, que é considerado o lobo que está no topo da pirâmide. Ele é o responsável por tomar as decisões de caça do grupo. No segundo nível, logo abaixo do topo, temos o beta β, que é o lobo que auxilia o líder nas tomadas de decisões da alcateia. Abaixo, no terceiro nível da pirâmide, temos o delta δ, que é considerado o responsável pela segurança da alcateia e é subordinado dos lobos acima da hierarquia. Os demais lobos da alcateia são considerados os ômegas ω e ficam no último nível da pirâmide [23].

O modelo matemático para otimização é apresentado considerando as seguintes etapas utilizadas pela alcateia dos lobos cinzentos [23]:

1) localizar e perseguir o alvo;
2) perseguir e rodear o alvo;
3) atacar o alvo;

A primeira etapa do algoritmo é identificar quais são as três melhores ECs para guiar a alcateia na busca pela melhor *Edge* para armazenar o recurso. Portanto, considera-se a melhor EC como o alpha (α). Em seguida, verifica-se qual será definido com beta (β) e qual será o delta (δ). A última EC será definida como Ômega. Para a implementação do modelo do algoritmo de otimização GWO, a caça é guiada pelas soluções α, β e δ.

As equações 1 e 2 foram propostas por [23], e adaptadas para o GROMEC, para modelar matematicamente o comportamento dos lobos durante a perseguição à presa.

$$ECD = |coef2 * V_d(x) - V(x)| \tag{1}$$

$$V(r+1) = V_d(x) - coef1 * ECD \tag{2}$$

Onde, x representa a iteração atual, V_d é a variável de valores do recurso do UE, V indica os valores de recursos que a EC possui e $coef1$ e $coef2$ são os coeficientes utilizados para movimentação na fórmula. Assim, é feita a relação de distância entre a presa (recurso) e o lobo (*Edge*), que diminui a cada iteração.

As variáveis $coef1$ e $coef2$ são calculadas seguindo os modelos das equações 3 e 4, respectivamente:

$$coef1 = 2a * s_1 - a \tag{3}$$

$$coef2 = 2 * s_2 \tag{4}$$

Onde a é decrementado de 2 até 0 durante a iteração e s_1 e s_2 são variáveis aleatórias com valores entre 0 e 1. O a é responsável por fazer o lobo se aproximar da presa em cada iteração.

A próxima etapa é atacar a presa. Como normalmente não se tem o posicionamento ótimo da presa, para simular matematicamente o comportamento de caça, considera-se que os três primeiros tenham conhecimento do posicionamento da presa. Para isso, [23] propôs as equações 5, 6 e 7. Assim, são selecionadas as três melhores *edges* de acordo com seu *fitness* e definido qual será a alpha, sendo a de melhor solução. Também é definido quem será o beta e delta. O último é o ômega e tem sua posição atualizada de acordo com as três melhores.

$$fitness_1 = fitness_{-\alpha}(t) - coef1_1 * ECD\_\alpha \tag{5}$$

$$fitness_2 = fitness_{-\beta}(t) - coef1_2 * ECD\_\beta \tag{6}$$

$$fitness_3 = fitness_{-\delta}(t) - coef1_2 * ECD\_\delta \tag{7}$$

Onde calcula-se os *fitness* dos três líderes da alcateia. Os ECD_α, ECD_β e ECD_δ são definidos, respectivamente, pelas equações 8, 9 e 10:

$$ECD\_\alpha = |coef2 * fitness_{-\alpha}(x) - fitness(x)| \tag{8}$$

$$ECD\_\beta = |coef2 * fitness_{-\beta}(x) - fitness(x)| \tag{9}$$

$$ECD\_\delta = |coef2 * fitness_{-\delta}(x) - fitness(x)| \tag{10}$$

A melhor EC é atualizada com a soma da média das posições das três melhores, conforme a equação 11.

$$fitness(x+1) = \frac{fitness_1 + fitness_2 + fitness_3}{3} \tag{11}$$

Após conseguir fazer com que a presa pare de se movimentar, a alcateia de lobos cinzentos parte para o ataque. Isto é feito com a atualização do coeficiente a, que é decrementado para encontrar a melhor tarefa. Matematicamente, esta etapa é feita com a atualização de a, como apresentado na equação 12:

$$a = 2 - (\frac{a}{num\_iteracao}) \tag{12}$$

Onde a é decrementado de [2-0] e $num\_iteracao$ corresponde ao número máximo de iterações definido no algoritmo. Assim, com a atualização de a a cada iteração, o valor de $coef1$ também é atualizado, fazendo com que os lobos se aproximem da melhor posição da presa.

C. Algoritmo GROMEC

O algoritmo 1 apresenta a abstração do GROMEC. Inicialmente ele verifica se o recurso da UE é de entrada ($UE\_Entrada$) ou saída (UE_Saida) (linhas 1 e 5), para acrescentar ou retirar, respectivamente, no conjunto de recursos da EC ($conj\_recursos$). Assim, se tiver alguma requisição, é executada a função para selecionar a Edge ($Func\_Escolhe\_EC$) (linha 9). A função recebe como parâmetros os dados de recursos de um UE ($UE\_Recursos$) e as informações das ECs ($EC\_Recursos$), e verifica qual a provável EC que irá alocar os recursos da UE, considerando o algoritmo GWO que foi adaptado para selecionar a melhor EC de acordo com a sua capacidade de recursos no momento da solicitação. Assim, é verificado se há recursos disponíveis que atendam a solicitação da UE. Se tiver, o UE é alocado por meio da função $UE\_Alocado()$ na Edge (linha 12). Caso contrário, é registrado como um recurso bloqueado por meio da função $UE\_Bloqueado()$ (linha 14). Se o UE tiver recursos bloqueados em todos as ECs, será negada a alocação por meio da função $UE\_Negado()$ (linha 17).

O Algoritmo 2 apresenta a $Func\_Escolhe\_EC$, utilizada para selecionar a provável Edge ($provEC$) para alocar os recursos solicitados. O algoritmo recebe como parâmetros os dados de recursos do UE e das Edges, e tem como objetivo

Algoritmo 1 Algoritmo do GROMEC

```
1: se (UE_Entrada) então
2:    conj_recursos ← conj_recursos + UE_Recursos
3:    requisicao = true
4: fim se
5: se (UE_Saida) então
6:    conj_recursos ← conj_recursos - UE_Recursos
7: fim se
8: se (requisicao) então
9:    provEC ← Func_Escolhe_EC(UE_Recursos,EC_Recursos)
10:   enquanto (provEC_Proximo ≤ Numero_EC) faça
11:      se (provEC_recursos ≤ requisicao_Recursos) então
12:         retorna UE_Alocado()
13:      senão
14:         retorna UE_Bloqueado()
15:      fim se
16:   fim enquanto
17:   retorna UE_Negado()
18: fim se
```

selecionar a Edge com o melhor fitness, utilizando o modelo do algoritmo GWO.

O *fitness* é a busca pelo indivíduo (EC) mais apropriado para alocar os recursos do UE. Para calcular o *fitness* é aplicado um algoritmo genético (GA de *Generic Algorithm*). No GROMEC, foram utilizados dois GAs buscando encontrar o melhor a ser utilizado no cenário apresentado.

A primeira função utilizada para calcular o *fitness* no GROMEC é a função Rosenbrock (13), que é uma função teste convencional para muitos algoritmos de otimização [25].

$$Rosenbrock(x) = \sum_{i=1}^{d-1} \left[100\left(x_{i+1} - x_t^2\right)^2 + (x_i - 1)^2\right] \tag{13}$$

Onde x é o conjunto de recursos disponíveis na EC, e i corresponde ao recurso específico da iteração que o algoritmo está executando.

A segunda função utilizada para comparação de resultados é a função de Ackley (14), que é uma função clássica de otimização estática.

$$\begin{aligned} Ackley(p) = & -20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}) \\ & - \exp(\frac{1}{n}\sum_{i=1}^{n} cos(2\pi x_i)) \\ & + 20 + e \end{aligned} \tag{14}$$

Onde, n corresponde à dimensão de recursos disponíveis nas *edges*, x corresponde ao recurso específico da iteração que o algoritmo está executando e e foi considerado um valor exponencial (1). Definindo assim, durante a estrutura de repetição, o fitness de cada EC.

Após aplicar as equações 13 e 14 dentro da função $calculaFitness$ e definir o *fitness* da *edge* da iteração atual (linha 3), é verificado se ela é a melhor EC até o momento. Caso seja, na linha 5 é atualizada ($atualiza\_Alpha()$) a melhor EC (EC_Alpha). Se não, é verificado se o *fitness* calculado é melhor que o do EC_Beta e, caso verdadeiro, atualiza o beta ($atualiza\_Beta()$) (linha 7). Se não entrar em nenhuma das condições anteriores, ele verifica se é melhor que o EC_Delta e, se for, atualiza o delta ($atualiza\_Delta()$) (linha 9).

Então, o algoritmo executa a etapa do ataque à presa. Primeiro é atualizada a variável responsável por fazer o avanço no ataque a (linha 9). Depois, na estrutura de repetição, são atualizados os coeficientes $coef1$ e $coef2$ (linha 16), conforme equações 3 e 4, respectivamente. Além de executar as equações que definem as posições de movimentação dos lobos (linhas 17, 18 e 19). Assim, após executar o máximo de iterações $num\_iteracao$ definido no algoritmo, é retornado o EC_Alpha (linha 22).

Algoritmo 2 Func_Escolhe_EC

```
1: enquanto num_iteracao faça
2:    enquanto Numero_EC faça
3:       fitness ← calculaFitness(Numero_EC, :)
4:       se fitness < EC_Alpha então
5:          atualiza_Alpha()
6:       senão se fitness < EC_Beta então
7:          atualiza_Beta()
8:       senão se fitness < EC_Delta então
9:          atualiza_Delta()
10:      fim se
11:   fim enquanto
12: fim enquanto
13: atualiza(a)
14: enquanto (provEC_Proximo ≤ Numero_EC) faça
15:   enquanto tamanho faça
16:      atualiza_as_variáveis(coef1, coef2)
17:      executa_as_equações(8, 9, 10)
18:      executa_as_equações(5, 6, 7)
19:      executa_as_equações(11)
20:   fim enquanto
21: fim enquanto
22: retorna EC_Alpha
```

IV. Avaliação e Resultados

Esta seção descreve o cenário, a metodologia de avaliação e os resultados do algoritmo de alocação de recurso para dispositivos 5G em EC proposto, comparando as duas configurações feitas para o algoritmo meta-heurístico e, posteriormente, duas técnicas de alocação da literatura.

A. Cenário e Metodologia

Para simulação, considerou-se como cenário a região do Parque de Ibirapuera, localizado na cidade de São Paulo, que é considerado o parque mais visitado do Brasil. O parque tem 1.58km^2 e seu horário de funcionamento em dias da semana é das 5h às 24h [26]. O cenário será coberto por 4 RSUs, para atender o perímetro do parque. As RSUs são instaladas nas extremidades do parque e interligadas através de uma conexão cabeada para fazer o gerenciamento de recursos entre si. Cada RSU possui uma EC que tem as especificações de área de cobertura de comunicação de redes 5G, como faixa de frequência no canal de comunicação, sem interferência de equipamentos e condições mínimas de segurança e qualidade. A linguagem de programação Python (versão 3.6) foi utilizada para realizar a simulação.

O cenário foi iniciado com um fluxo pequeno de UEs evoluindo para um cenário com um fluxo maior, o que pode ocorrer durante os horários de maiores movimentos no parque. Assim, foram considerados 199, 397, 654, 863 e 1095 UEs na simulação. As quantidades foram definidas de forma aleatória

durante a simulação utilizando a distribuição de Poisson, determinando os eventos de entrada e saída. As ECs iniciam cada simulação com 100% de cada recurso (memória, armazenamento, tempo e processamento) e, em cada alocação de recursos de um UE, é subtraída a quantidade que o dispositivo precisa, da capacidade atual da EC selecionada. Para definir a quantidade máxima de recursos de cada serviço, foram analisados e considerados serviços de consumo baixo, como mensagens multimídia, onde pode-se incluir textos ilimitados e um arquivo de som, vídeo ou foto. Quando o processo é de saída de um UE, os recursos do dispositivo são somados na EC, ficando disponíveis novamente. Além disso, cada simulação foi executada 33 vezes para obter um intervalo de confiança de 95%.

B. Técnicas de Comparação

Para análise da eficiência do algoritmo proposto, nossa proposta foi comparada com outras técnicas heurísticas tradicionalmente utilizadas por pesquisadores.

A *best-first* é uma técnica heurística, que utiliza o conceito de fila prioritária, seguindo uma regra específica [27]. Neste trabalho a técnica foi implementada para alocar os recursos dos dispositivos na EC que tem menor disponibilidade de um recurso. Como regra de prioridade na fila, foi considerada a disponibilidade do recurso de memória das ECs para o critério de seleção do *Best-first*. Assim, o algoritmo foi desenvolvido para percorrer todas as *edges* do cenário analisando a quantidade do recurso de memória disponível e selecionar para armazenar os recursos, a *edge* que está mais próxima de completar 100% de sua capacidade.

O *Nancy* [28] utiliza o método AHP (*Analytic Hierarchy Process*), que calcula um fator de influência de cada parâmetro utilizado no algoritmo, fornecendo uma estrutura técnica para tomadas de decisão em problemas que envolvem critérios múltiplos. Desta forma, o algoritmo ajustar o peso atribuído aos seus parâmetros, através da comparação de pares entre os valores numéricos de cada parâmetro e os graus de importância atribuídos a eles. O parâmetro com maior peso é considerado o mais importante para o critério.

C. Resultados

Para avaliação do GROMEC foram consideradas duas configurações e três métricas. Para cada métrica foram realizadas duas simulações do GROMEC para cálculo do *fitness*. Na configuração 1, o GROMEC utilizou para cálculo do *fitness* a função Ackley e na configuração 2 utilizou-se a função Rosenbrock. Nas Tabelas III, IV e V são apresentadas as quantidades médias de UEs atendidos, negados e bloqueados, respectivamente, para cada função em relação à quantidade de dispositivos considerados na simulação (199, 397, 654, 863 e 1095). Esta média foi calculada considerando os 33 resultados obtidos para cada algoritmo durante a simulação.

TABELA III
FUNÇÕES PARA CÁLCULO DE *fitness* - ATENDIDOS

Algoritmo Genético	199	397	654	863	1095
Ackley	155.06	256.42	388.85	510.24	628.48
Rosenbrock	155.21	255.60	387.72	510.09	630.12

TABELA IV
FUNÇÕES PARA CÁLCULO DE *fitness* - NEGADOS

Algoritmo Genético	199	397	654	863	1095
Ackley	33.93	130.57	255.15	342.75	456.51
Rosenbrock	33.78	131.39	256.27	342.90	454.54

TABELA V
FUNÇÕES PARA CÁLCULO DE *fitness* - BLOQUEADOS

Algoritmo Genético	199	397	654	863	1095
Ackley	210.27	724.12	1396.0	1916.42	2504.78
Rosenbrock	204.48	730.87	1404.39	1930.72	2526.90

Nota-se que os resultados, tanto na configuração 1 do GROMEC, que utilizou a função Ackley para cálculo do fitness, quanto na configuração 2, que utilizou a função de Rosenbrock, tiveram pequenas variações de valores em relação aos serviços atendidos, negados e bloqueados. Na soma dos resultados dos 5 cenários considerados na simulação, a função Ackley apresentou uma pequena vantagem nas 3 métricas. Assim, para comparação com as políticas de alocação de recursos *Best-first* e *Nancy*, foi utilizada a configuração 1.

A Fig. 2 apresenta o número de serviços atendidos, onde considera todos os UEs que tiveram seus serviços alocados nas ECs. Pode-se observar que o GROMEC apresentou um resultado melhor que o *Best-first*, com mais de 50% a partir do terceiro cenário, com uma média de 387 UEs atendidos. Já em relação ao *Nancy*, o GROMEC também conseguiu atender mais UEs, alocando aproximadamente 5.25% a mais de recursos em média considerando todos os cenários. Nesta configuração do GROMEC, o método de seleção de Edge por melhor *fitness* e aproximação do melhor EC, apresentou ser mais eficiente que os outros.

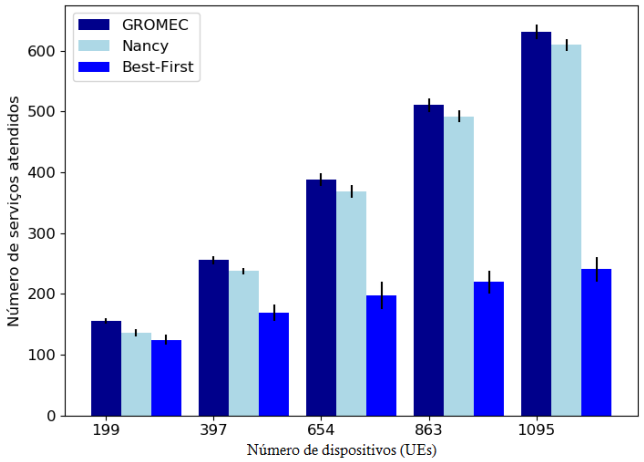


Fig. 2. Número de dispositivos atendidos GROMEC - Configuração 1.

Na Fig. 3 pode-se observar os serviços negados, ou seja, os UEs que não conseguiram ser alocados em nenhuma das Edges por falta de disponibilidade de recursos. O GROMEC teve menos serviços recusados, obtendo em média 33,93 serviços negados, na simulação com menor fluxo de serviços, enquanto o *Best-first* e o *Nancy* tiveram 54,36 e 43,30, respectivamente.

O GROMEC também recusou menos serviços no cenário com maior fluxo de UEs, recusando em média 456,15 dispositivos. Já o *Nancy* e o *Best-first* negaram, em média, a entrada de 465,87 e 834,36 UEs, respectivamente. A escolha das três melhores Edges durante o processo de ”caça”do GROMEC, mostrou ser mais eficiente, pois já pode indicar a segunda e terceira opção de EC, caso a primeira não tenha disponibilidade.

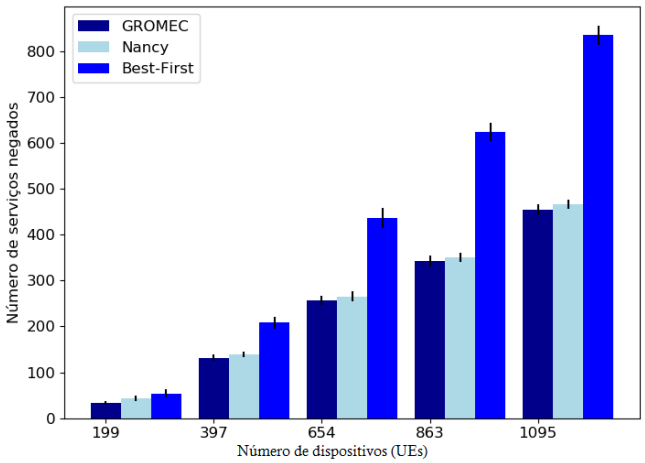


Fig. 3. Número de dispositivos negados GROMEC - Configuração 1.

Na Fig. 4 é possível observar os recursos bloqueados, que mostra a quantidade de vezes que a política precisa aguardar até encontrar uma EC. O GROMEC apresentou vantagem em relação ao *Nancy* e ao *Best-first*, tendo em média 210,27 e 724,12 bloqueios, nos dois primeiros cenário com menor fluxo de dispositivos. Já o *Best-first* teve a média de 379,87 e 1053,21, bloqueios, enquanto o *Nancy* teve 282,87 e 755,54. No entanto, nos cenários com mais de 654 dispositivos, o GROMEC apresentou uma pequena desvantagem de aproximadamente 6.2% em relação do *Nancy*, mas continuou apresentando vantagem em relação ao *Best-first*. O GROMEC mostrou ser mais rápido que o *Best-first* para encontrar uma Edge com disponibilidade, mesmo sendo um algoritmo que precisa de um laço de iterações na seleção das ECs.

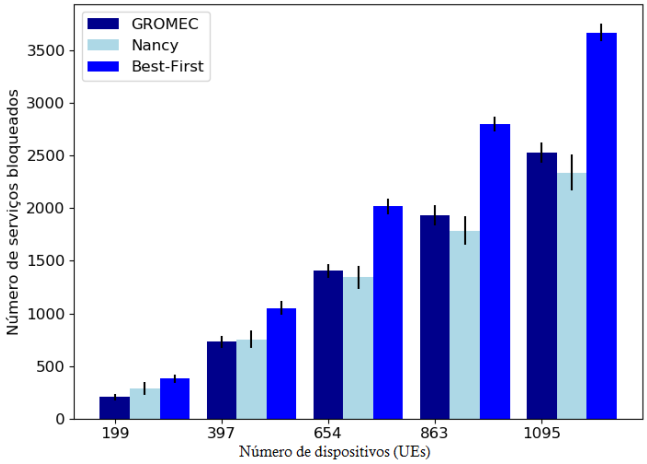


Fig. 4. Número de recursos de dispositivos bloqueados GROMEC - Configuração 1.

A Fig. 5 apresenta a curva de convergência das iterações do GROMEC. Os dados foram obtidos através da média das simulações. Pode-se observar que o algoritmo aproxima-se de um caminho de convergência para encontrar a *edge* ideal a partir da décima iteração, chega próximo do melhor *fitness* após a décima quinta execução e termina as iterações com o a média de *fitness* em 0,17.

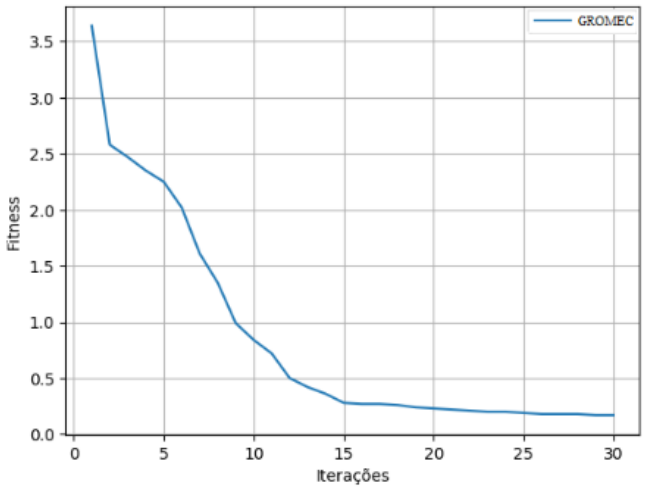


Fig. 5. Convergência das iterações - Configuração 1.

A solução proposta neste artigo demonstrou ser mais eficiente que o método tradicional *Best-first* e também apresentou vantagem em relação ao *Nancy*, que utiliza o método AHP. O GROMEC conseguiu atender a maior quantidade de UEs, reduzindo a quantidade de serviços recusados e apresentando um número baixo de bloqueios durante a procura por uma EC.

V. Conclusão

A IoT é considerada uma revolução tecnológica que irá mudar a vida das pessoas. Para acompanhar tal evolução, é preciso melhorar as estruturas e serviços que são disponibilizados. A *Edge Computing* é uma solução promissora que direciona os recursos para a borda das redes, deixando mais próximo dos UEs, diminuindo o problema de latência [8]. Além disso, cada vez mais temos dispositivos IoT compartilhando e precisando de recursos disponibilizados nas redes. Assim, novas técnicas de alocação de recursos surgem para melhorar os métodos de alocação e gerenciamento desses recursos.

Neste trabalho buscou-se utilizar uma técnica de inteligência bioinspirada na natureza para alocação de recursos. Para isso, utilizou-se como base a movimentação de caça da alcateia de lobos cinzentos para desenvolver o GROMEC. O algoritmo utiliza a lógica meta-heurística do GWO para selecionar as melhores *edges* durante a simulação de alocação de recursos de UEs em ECs. O método mostrou-se mais eficiente que as outras políticas bastante utilizadas para alocação de recursos na literatura e usadas na comparação, conseguindo atender mais UEs e minimizando os serviços recusados.

A utilização da técnica meta-heurística apresenta uma perspectiva interessante, visto que a inteligência utilizada por animais difere de métodos matemáticos que são utilizados tradicionalmente em alocação de recursos. A técnica se favorece por poder ser adaptada a diversos cenários. No GROMEC foi

considerado um cenário com 4 *edges* durante a simulação, no entanto, a abordagem pode ser aplicada em diversos tipo de cenários, com configurações diferentes, independentemente da quantidade de ECs e UEs.

Para trabalhos futuros iremos fazer a comparação do GRO-MEC com outros algoritmos meta-heurísticos, considerar a predição de recursos nas ECs, além de utilizar um software que simule um ambiente real.

REFERÊNCIAS

[1] IBM. (2020) Telecom's 5g future - creating new revenue streams and services with 5g, edge computing, and ai. IBM Corporation. [Online]. Available: https://www.ibm.com/industries/telecom-media-entertainment/resources/5g-revolution/res/Telecom_s_5G_future_RESEARCH_INSIGHTS_v2.pdf

[2] G. P. Rocha Filho, R. I. Meneguette, G. Maia, G. Pessin, V. P. Gonçalves, L. Weigang, J. Ueyama, and L. A. Villas, "A fog-enabled smart home solution for decision-making using smart objects," *Future Generation Computer Systems*, vol. 103, pp. 18–27, 2020.

[3] D. J. Freitas, T. B. Marcondes, L. H. V. Nakamura, and R. I. Meneguette, "A health smart home system to report incidents for disabled people," in *2015 International Conference on Distributed Computing in Sensor Systems*, 2015, pp. 210–211.

[4] K. Zhang, Y. Mao, S. Leng, Q. Zhao, L. Li, X. Peng, L. Pan, S. Maharjan, and Y. Zhang, "Energy-efficient offloading for mobile edge computing in 5g heterogeneous networks," *IEEE Access*, 2016.

[5] J. Fu and C. L. Y. Chen, "The contribution and prospect of 5g technology to china's economic development," *Journal of Economic Science Research— Volume*, vol. 3, no. 03, 2020.

[6] J. Liu, Y. Kawamoto, H. Nishiyama, N. Kato, and N. Kadowaki, "Device-to-device communications achieve efficient load balancing in lte-advanced networks," *IEEE Wireless Communications*, 2014.

[7] R. I. Meneguette, "Software defined networks: Challenges for sdn as an infrastructure for intelligent transport systems based on vehicular networks," in *2020 16th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, 2020, pp. 205–212.

[8] D. C. Marinescu, *Cloud Computing: Theory and Practice*, second edition ed., J. Grover, P. Vinod, and C. Lal, Eds. Morgan Kaufman - Elsevier, 2018.

[9] D. M. Soleymani, M. R. Gholami, J. Mueckenheim, and A. Mitschele-Thiel, "Dedicated sub-granting radio resource in overlay d2d communication," in *IEEE 30th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, 2019, pp. 1–7.

[10] S. Li, N. Zhang, S. Lin, L. Kong, A. Katangur, M. K. Khan, M. Ni, and G. Zhu, "Joint admission control and resource allocation in edge computing for internet of things," *IEEE Network*, pp. 72–79, 2018.

[11] N. Hassan, K. A. Yau, and C. Wu, "Edge computing in 5g: A review," *IEEE Access*, vol. 7, pp. 127 276–127 289, 2019.

[12] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing-a key technology towards 5g," in *ETSI white paper*, vol. 11, no. 11, 2015, pp. 1–16.

[13] X. Li, Y. Liu, H. Ji, H. Zhang, and V. C. M. Leung, "Optimizing resources allocation for fog computing-based internet of things networks," *IEEE Access*, vol. 7, pp. 64 907–64 922, 2019.

[14] J. Xu, B. Palanisamy, H. Ludwig, and Q. Wang, "Zenith: Utility-aware resource allocation for edge computing," in *2017 IEEE International Conference on Edge Computing (EDGE)*, 2017, pp. 47–54.

[15] S. Nesmachnow, "An overview of metaheuristics: accurate and efficient methods for optimisation," *International Journal of Metaheuristics*, vol. 3, no. 4, pp. 320–347, 2014.

[16] M. Aazam, K. A. Harras, and S. Zeadally, "Fog computing for 5g tactile industrial internet of things: Qoe-aware resource allocation model," *IEEE Transactions on Industrial Informatics*, pp. 3085–3092, 2019.

[17] S. K. Goudos, M. Deruyck, D. Plets, L. Martens, K. E. Psannis, P. Sarigiannidis, and W. Joseph, "A novel design approach for 5g massive mimo and nb-iot green networks using a hybrid jaya-differential evolution algorithm," *IEEE Access*, vol. 7, pp. 105 687–105 700, 2019.

[18] L. N. Huynh, Q.-V. Pham, X.-Q. Pham, T. D. Nguyen, M. D. Hossain, and E.-N. Huh, "Efficient computation offloading in multi-tier multi-access edge computing systems: A particle swarm optimization approach," *Applied Sciences*, vol. 10, no. 1, p. 203, 2020.

[19] K. Jiang, H. Ni, P. Sun, and R. Han, "An improved binary grey wolf optimizer for dependent task scheduling in edge computing," in *21st Intern. Conference on Advanced Communication Technology*, 2019.

[20] P. Hosseinioun, M. Kheirabadi, S. R. Kamel Tabbakh, and R. Ghaemi, "A new energy-aware tasks scheduling approach in fog computing using hybrid meta-heuristic algorithm," *Journal of Parallel and Distributed Computing*, vol. 143, pp. 88 – 96, 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S074373152030023X

[21] Y. Shao, C. Li, Z. Fu, L. Jia, and Y. Luo, "Cost-effective replication management and scheduling in edge computing," *Journal of Network and Computer Applications*, vol. 129, pp. 46 – 61, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1084804519300013

[22] P. Pol and V. K. Pachghare, "of meta-heuristic optimization approaches: in virtue of grey wolf optimization," in *2019 Global Conference for Advancement in Technology (GCAT)*, 2019, pp. 1–7.

[23] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, pp. 46 – 61, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0965997813001853

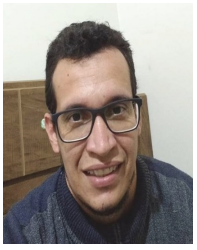
[24] A. A. Jha, S. Jain, and S. Thenmalar, "Survey on grey wolf algorithm in resource allocation," *International Journal of Pure and Applied Mathematics - Special Issue*, vol. 118, pp. 201 – 206, 2018.

[25] S. K.Valluru and M. Singh, "Multi-objective genetic and adaptive particle swarm optimization algorithms: A performance analysis with benchmark functions," in *2nd IEEE International Conference on Power Electronics, Intelligent Control and Energy Systems*, 2018.

[26] P. I. Conservação. (2020, jul) Visitando o parque ibirapuera: Sobre o parque. [Online]. Available: https://parqueibirapuera.org/parque-ibirapuera/parque-ibirapuera/

[27] M. Fickert, "A novel lookahead strategy fordelete relaxation heuristics in greedy best-first search," in *Thirtieth International Conference on Automated Planning and Scheduling (ICAPS 2020)*, 2020.

[28] R. S. Pereira, D. D. Lieira, M. A. C. da Silva, A. H. de Macedo Pimenta, J. B. D. da Costa, D. Rosário, and R. I. Meneguette, "A novel fog-based resource allocation policy for vehicular clouds in the highway environment," in *proceeding of the 11th Latin-American Conference on Communications (LATINCOM)*, 2019, pp. 1–6.



Douglas Dias Lieira tecnólogo em Informática Para Negócios (FATEC) em 2011, Especialista em Desenvolvimento Web (Centro Universitário Claretiano) em 2013. Atualmente é professor no Instituto Federal de São Paulo (IFSP) e mestrando em Ciência da Computação no Programa de Pós-Graduação da Universidade Estadual Paulista - Júlio de Mesquita Filho (UNESP), com pesquisas nas áreas de redes veiculares, gerenciamento de recursos em *Cloud/Fog/Edge Computing* e algoritmos meta-heurísticos.

Matheus Sanches Quessada tecnólogo em Análise e Desenvolvimento de Sistemas pelo Instituto Federal de Educação, Ciência e Tecnologia de São Paulo (IFSP) em 2019. Atualmente é aluno do Programa de Pós-Graduação em Ciência da Computação pela Universidade Estadual Paulista Júlio de Mesquita Filho (UNESP). Possui experiência em desenvolvimento de software, aplicações móveis, aplicações web. Sua linha de pesquisa é em redes veiculares, gerenciamento de recursos e sistemas de transporte inteligentes.

André Luis Cristiani graduado em Análise e Desenvolvimento de Sistemas pelo Instituto Federal de São Paulo campus Catanduva (2018). Atualmente é aluno de mestrado em Ciência da Computação na Universidade Federal de São Carlos (UFSCar). Possui experiência na área de Ciência da Computação, com ênfase em desenvolvimento de software, aprendizado de máquina, big data, detecção de anomalias e sistemas de transporte inteligentes.

Rodolfo Ipolito Meneguette Bacharel em Ciência da Computação pelo Universidade Paulista (UNIP) em 2006. Mestre em Ciência da Computação pela Universidade Federal de São Carlos (UFSCar) em 2009. Doutor em Ciência da Computação pela Universidade Estadual de Campinas (UNICAMP) em 2013. Pós-Doutorando pela Universidade de Ottawa (UOttawa) em 2017. Atualmente é professor do Instituto De Ciências Matemáticas e de Computação (ICMC) da Universidade de São Paulo (USP). Líder do Grupo de pesquisa internet das coisas com foco em computação urbana. Sua linha de pesquisa é em sistema de transporte inteligente, redes veiculares, nuvens, gerenciamento de mobilidade.