UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

Alocação de Recursos em Nuvens Veiculares baseada em Teoria dos Jogos

Aguimar Ribeiro Júnior

Dissertação de Mestrado do Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional (PPG-CCMC)

ICMC USP
SÃO CARLOS

Aguimar Ribeiro Júnior

Alocação de Recursos em Nuvens Veiculares baseada em Teoria dos Jogos

Dissertação apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP, como parte dos requisitos para obtenção do título de Mestre em Ciências – Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientador: Prof. Dr. Rodolfo Ipolito Meneguette

USP – São Carlos
Fevereiro de 2024

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

R484a

Ribeiro Júnior, Aguimar
Alocação de Recursos em Redes Veiculares baseada em Teoria dos Jogos / Aguimar Ribeiro Júnior; orientador Rodolfo Ipolito Meneguette. -- São Carlos, 2023.
77 p.

Dissertação (Mestrado - Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2023.

1. VANET. 2. Nuvens Veiculares. 3. Alocação de Recursos. 4. Teoria dos Jogos. 5. Shapley Values. I. Ipolito Meneguette, Rodolfo, orient. II. Título.

Bibliotecários responsáveis pela estrutura de catalogação da publicação de acordo com a AACR2:
Gláucia Maria Saia Cristianini - CRB - 8/4938
Juliana de Souza Moraes - CRB - 8/6176

Aguimar Ribeiro Júnior

Game Theory-Based Resource Allocation for Vehicular Clouds

Dissertation submitted to the Instituto de Ciências Matemáticas e de Computação – ICMC-USP – in accordance with the requirements of the Computer and Mathematical Sciences Graduate Program, for the degree of Master in Science. *FINAL VERSION*

Concentration Area: Computer Science and Computational Mathematics

Advisor: Prof. Dr. Rodolfo Ipolito Meneguette

USP – São Carlos
February 2024

AGRADECIMENTOS

Aos meus pais, alicerces da minha jornada. Tudo o que conquistei até aqui é um reflexo do incansável conjunto de cuidados, dedicação e exemplos que recebi de vocês. Cada palavra, olhar e conselho foi um apoio constante em todas as escolhas que fiz. Amo vocês profundamente.

Ao meu orientador, o Prof. Dr. Rodolfo Ipolito Meneguette, pela orientação inestimável e apoio constante ao longo da minha trajetória acadêmica. Você foi um guia paciente, inspirador e um parceiro incansável nesta jornada. Agradeço imensamente por sua contribuição para esta conquista. Obrigado.

“Conheça todas as teorias,
domine todas as técnicas,
mas ao tocar uma alma humana,
seja apenas outra alma humana.”
(Carl Jung)

RESUMO

RIBEIRO JR., A. **Alocação de Recursos em Nuvens Veiculares baseada em Teoria dos Jogos**. 2024. 77 p. Dissertação (Mestrado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2024.

A alocação de recursos em tempo real tornou-se uma tarefa cada vez mais complexa nas redes veiculares, à medida que precisam atender a diversas solicitações de serviços. Essa complexidade, muitas vezes associada a ambientes com recursos limitados, tem sua origem principalmente no contínuo crescimento do número de veículos conectados às redes, como a Internet. Além disso, o surgimento de aplicações com restrições de tempo para sua execução contribui para tornar essas redes mais desafiadoras. Essas aplicações estão se tornando progressivamente complexas e requerem recursos computacionais adicionais.

Neste estudo apresenta-se uma abordagem para resolver o problema de alocação de recursos em redes veiculares. A abordagem fundamentada em Teoria dos Jogos emprega um jogo de coalizão que visa maximizar e equilibrar a utilização de recursos entre as várias nuvens veiculares (VC). A solução heurística proposta adota o uso de *Shapley Values* para estabelecer as sequências de tarefas e VCs a serem seguidas durante o processo de alocação. Mais especificamente, modelou-se o problema como um Jogo de Mercado especialmente desenhado para resolver o problema de alocação de recursos em nuvens veiculares dinâmicas. Por fim, uma análise comparativa foi realizada entre o desempenho da solução proposta e outras soluções relevantes encontradas na literatura. Essa análise foi conduzida em cenários com diferentes restrições, como diferentes taxas de serviços, alcance de comunicação e quantidade de recursos oferecidos por cada veículo. Essa abordagem permitiu avaliar a eficácia e a adaptabilidade da solução diante de uma variedade de condições.

Palavras-chave: VANET, Nuvens Veiculares, Alocação de Recursos, Teoria dos Jogos, Shapley Values.

ABSTRACT

RIBEIRO JR., A. **Game Theory-Based Resource Allocation for Vehicular Clouds**. 2024. 77 p. Dissertação (Mestrado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2024.

Real-time resource allocation has become increasingly complex in vehicular networks as they must address various service requests. This complexity, often associated with resource-constrained environments, primarily stems from the continuous growth in the number of vehicles connected to networks, such as the Internet. Furthermore, the emergence of applications with time constraints on their execution adds to the challenges in these networks. These applications are becoming progressively intricate and demand additional computational resources. This study presents an approach to solving the resource allocation problem in vehicular networks. The approach, grounded in Game Theory, uses a coalition game to maximize and balance resource utilization among multiple vehicular clouds (VC). The proposed heuristic solution employs *Shapley Values* to establish the task sequences and VCs to be followed during the allocation process. More precisely, we modeled the problem as a Market Game designed to address the resource allocation problem in dynamic vehicular clouds. Lastly, we conducted a comparative analysis of the performance of the proposed solution against other relevant solutions found in the literature. We conducted this analysis in scenarios with different constraints, such as varying service rates, communication ranges, and the quantity of resources each vehicle offers. This approach enabled us to evaluate the solution's effectiveness and adaptability across various conditions.

Keywords: VANET, Vehicular Clouds, Resource Allocation, Game Theory, Shapley Values.

LISTA DE ILUSTRAÇÕES

LISTA DE QUADROS

LISTA DE ALGORITMOS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

A3C	*Advantage Actor-Critic*
AHP	*Analytic Hierarchy Process*
BM-BFO	*Brownian motion-centered Bacteria Foraging Optimization*
CGAME	Jogo de Coalizão
CM-CSO	*Crossover and Mutation-centered Chicken Swarm Optimization*
DEC	Distribuição Equilibrada de Carga
DFAC	Diferentes Faixas de Alcance de Comunicação
DRL	*Deep Reinforcement Learning*
EC	*Edge Computing*
MEC	*Mobile Edge Computing*
NSGA-II	*Non-dominated Sorting Genetic Algorithm II*
OBU	*On Board Unit*
PSO	*Particle Swarm Optimization*
QoS	Qualidade de Serviços
RL	*Reinforcement Learning*
RSU	*Roadside Unit*
SUMO	*Simulation of Urban Mobility*
TraCI	*Traffic Control Interface*
TVS	Taxas Variáveis de Solicitações de Serviços
V2I	*Vehicle-to-Infrastructure*
V2P	*Vehicle-to-Pedestrian*
V2V	*Vehicle-to-Vehicle)*
VANET	*Vehicular Ad Hoc Networks*
VC	*Vehicular Cloud*
VCC	*Vehicular Cloud Controller*

SUMÁRIO

CAPÍTULO

1

INTRODUÇÃO

O setor automotivo passa por uma significativa revolução tecnológica, marcada por avanços recentes que se concentram na melhoria da segurança e na aprimoração da experiência dos passageiros durante viagens rodoviárias (MENEGUETTE; GRANDE; LOUREIRO, 2018). Essas inovações geram desafios e uma mudança notável de paradigma, especialmente para os usuários e proprietários de veículos inteligentes, tais como veículos com conectividade à Internet, autônomos e veículos elétricos (QUALCOMM, 2020; DANQUAH; ALTILAR, 2020; MENEGUETTE; BOUKERCHE, 2017).

A CISCO (2020) estima que até o final de 2025, aproximadamente dois bilhões de veículos estarão conectados à Internet em todo o mundo. Para atender a essa expansão notável, a indústria automobilística investe substancialmente em soluções inovadoras para otimizar o aproveitamento dos recursos computacionais presentes nesses veículos. Além disso, direciona esforços para o desenvolvimento de novos sensores e *chipsets* capazes de proporcionar mais capacacidade de processamento e comunicação. (QUALCOMM, 2020; MASCHI *et al.*, 2018).

Conforme mencionam Meneguette, Grande e Loureiro (2018), essa conectividade possibilita o provisionamento de serviços por meio do compartilhamento de recursos e da troca de informações entre os agentes participantes da rede, contribuindo para a resolução de problemas relacionados à mobilidade urbana.

A comunicação é facilitada por meio de uma rede veicular Ad Hoc (VANET - *Veicular Ad Hoc Networks*) (MENEGUETTE; BOUKERCHE, 2017) que permite a comunicação direta entre veículos (V2V - *Vehicle-to-Vehicle)*, entre veículos e a infraestrutura (V2I - *Vehicle-to-Infrastructure*) (AL-SULTAN *et al.*, 2014; MENEGUETTE; GRANDE; LOUREIRO, 2018) e também entre veículos e pedestres (V2P - *Vehicle-to-Pedestrian*) (QUALCOMM, 2020).

A comunicação em VANETs inaugura uma mudança paradigmática que viabiliza a integração da computação em nuvem no contexto das redes de veículos. Os veículos são agrupados em nuvens veiculares (VC - *Vehicular Cloud*) criando um *pool* de recursos que pode ser

compartilhado tanto entre os participantes da VC quando para atender requisições de serviços fora desse grupo. É introduzido também o conceito de Controlador de Nuvem Veicular (VCC - *Vehicular Cloud Controller*) (PAUL *et al.*, 2016), que é uma entidade localizada na borda da rede responsável pela criação e manutenção das VCs e pelo gerenciamento dos recursos computacionais disponibilizados pelos veículos.

Nesse ambiente, os desafios associados à Qualidade de Serviços (QoS) no atendimento de requisições sob demanda e a escalabilidade nesse atendimento ganham relevância (MENEGUETTE; GRANDE; LOUREIRO, 2018; ZHANG; GRANDE; BOUKERCHE, 2015; DANQUAH; ALTILAR, 2020; MENEGUETTE; BOUKERCHE; GRANDE, 2016; MENEGUETTE; MADEIRA; BITTENCOURT, 2012). Consequentemente, torna-se crítica a necessidade de algoritmos que gerenciem o compartilhamento desses recursos para melhor atender essas solicitações (ZHANG; SHEN; YANG, 2021; DANQUAH; ALTILAR, 2020).

Em uma vasta gama de serviços oferecidos pelas VCs, para alcançar o efetivo gerenciamento de grandes *pools*, faz-se o uso da Computação de Borda (EC - *Edge Computing*) (GAI *et al.*, 2016; PEREIRA *et al.*, 2019). As abordagens associadas ao paradigma EC são adequadas para implementações agregadoras de recursos computacionais que necessitem de rápida alocação de recursos e de ajustes finos ao atender requisições de serviços em uma nuvem (LIU *et al.*, 2018; WU *et al.*, 2019).

A criação de aplicações seguras e a disponibilização de serviços em nuvens veiculares representam desafios significativos, especialmente quando relacionados à segurança, ao controle de tráfego e ao entretenimento em ambientes de alta mobilidade veicular (MENEGUETTE *et al.*, 2021; ARTHURS *et al.*, 2022). Para atender a esses serviços de forma eficaz, é fundamental que as decisões relativas à alocação de recursos sejam tomadas de maneira ágil e em tempo real. A literatura oferece uma gama de modelos para alocar recursos em VCs, muitos deles baseados em métodos como algoritmos gulosos (YANG *et al.*, 2017), meta-heurísticas (LIEIRA *et al.*, 2020), otimização combinatória (COSTA *et al.*, 2020), otimização multiobjetivo (WEI *et al.*, 2021), programação dinâmica e aprendizado por reforço (PENG; SHEN, 2020).

No entanto, poucos são os que consideram um importante aspecto: a natureza do objetivo de cada veículo ao participar de uma VC. Esse objetivo pode ser colaborativo, ou seja, buscar a cooperação entre todos os participantes da VC em prol de maximizar um interesse comum; ou competitivo, onde cada participante tenta maximizar seu próprio interesse de forma egoísta. Essa natureza é particularmente relevante em ambientes onde diversas requisições de serviços, muitas vezes conflituosas, como por exemplo *throughput* e *delay*; e utilização de recursos, consumo de energia e segurança.

Assim, garantir o balanceamento entre requisições conflitantes impacta diretamente no processo de tomada de decisão ao alocar recursos na rede. Em particular, impacta em como agregar os veículos em VCs e como gerenciar os recursos compartilhados para atender os diferentes objetivos de cada participante.

1.1 Motivação

Neste trabalho investigou-se o papel da Teoria dos Jogos na alocação de recursos compartilhados pelas VCs para atender diferentes solicitações de serviços em uma VANET. A Teoria dos Jogos é uma abordagem que utiliza ferramentas matemáticas para modelar e analisar situações envolvendo tomadas de decisão interativas.

Essas situações frequentemente incluem diversos tomadores de decisão, conhecidos como jogadores — que neste trabalho são as VCs —, cada um com objetivos distintos, nos quais as decisões individuais afetam o resultado para todos os envolvidos. Essa característica de interatividade diferencia a Teoria dos Jogos da teoria de decisão convencional, que geralmente lida com um único tomador de decisão. Assim, a Teoria dos Jogos tenta prever o comportamento dos jogadores e, por vezes, oferecer orientações aos tomadores de decisão sobre estratégias para atingir seus objetivos (MASCHLER; SOLAN; ZĀMÎR, 2020).

1.2 Objetivos

O objetivo geral deste trabalho foi modelar e desenvolver uma solução para o problema de alocação de recursos em redes veiculares utilizando ferramentas matemáticas de Teoria dos Jogos. A solução implementada realiza a alocação de forma eficiente ao mesmo tempo que promove a distribuição das solicitações de serviço entre um maior número de VCs a fim de distribuir a carga de requisições sem causar gargalos na rede e excesso de recursos ociosos nas VCs. Para atingir esse objetivo, os seguintes objetivos específicos foram propostos e implementados:

- Modelagem do problema de alocação de recursos como um jogo de coalizão, mais especificamente um Jogo de Mercado. Nesse jogo os jogadores, aqui chamados de produtores, encontram a melhor forma de produzir ganhos para o mercado como um todo a partir da troca de mercadorias entre eles;

- Aplicação de *Shapley Values* para determinar a ordem em que as tarefas devem ser atendidas para maximizar a alocação de recursos na rede veicular;

- Aplicação de *Shapley Values* para determinar a ordem em que as VCs devem ser utilizadas a fim de distribuir a alocação de tarefas entre um maior número de VCs, distribuindo a carga, e ao mesmo tempo maximizar a alocação de recursos na rede veicular; e

- Extensiva simulação dos modelos propostos utilizando *traces* reais, diferentes raio de comunicação, diferentes taxas de serviço e diferentes quantidades de recursos compartilhados pelos veículos.

1.3 Organização do Trabalho

Esta dissertação está organizada da forma que se segue. No Capítulo 2 são apresentados os conceitos de Jogo de Coalizão, Mercado, Jogo de Mercado e *Shapley Values* que são fundamentais para o melhor entendimento dos capítulos seguintes. No Capítulo 3 os principais trabalhos correlatos são apresentados e em seguida comparados à solução implementada neste trabalho.

Em seguida, no Capítulo 4 é apresentado o HARMONIC, a solução final modelada e implementada neste trabalho. São apresentadas também as fases de desenvolvimento do projeto até se chegar à implementação final. No Capítulo 5 são apresentados o cenário, as métricas utilizadas para avaliar os modelos desenvolvidos e os resultados obtidos. Por fim, no Capítulo 6 apresentam-se as conclusões finais deste trabalho, suas principais contribuições e trabalhos futuros.

CAPÍTULO

2

FUNDAMENTAÇÃO TEÓRICA

Neste capítulo são apresentadas as definições formais de Jogo de Coalizão, Mercado, Jogo de Mercado e *Shapley Values*. Esses conceitos formam a base para o desenvolvimento dos modelos matemáticos propostos neste trabalho.

2.1 Jogo de Coalizão

Um Jogo de Coalizão (CGAME) é um típico jogo cooperativo que tem como objetivo modelar situações onde os jogadores cooperam em si para atingir seus objetivos (MASCHLER; SOLAN; ZĀMÎR, 2020). Assume-se que os jogadores podem formar coalizões livremente e fazer acordos a fim de gerar determinado ganho. O único requisito necessário é que os jogadores sejam capazes de chegar a decisões em comum e que se comprometam a cumpri-las. O valor agregado gerado por uma coalizão com utilidades transferíveis é o ganho máximo que se pode gerar por meio da cooperação entre os seus participantes. Diferentes disciplinas utilizam o conceito de CGAMEs e seus teoremas para modelagem de problemas (SAFDARIAN *et al.*, 2021).

A Definição 1 descreve formalmente um Jogo de Coalizão (MASCHLER; SOLAN; ZĀMÎR, 2020). A coalizão formada por todos os jogadores é conhecida como grande coalizão. O número real $v(S)$ é o valor agregado gerado pela coalizão S e o resultado produzido por essa coalizão é independente das ações dos jogadores que não são membros de S.

Definição 1 (Jogo de Coalizão)**.** Um Jogo de Coalizão com utilidade transferível é um par (N, v) tal que:

- $N = 1, 2, ..., n$ é um conjunto finito de jogadores. Um subconjunto de N é chamado de coalizão. A coleção de todas as coalizões é dada por 2^N.
- $v : 2^N \to \mathbb{R}$ é uma função que associa toda coalizão S a um número real $v(S)$, satisfazendo $v(\emptyset) = 0$. Essa função é chamada de função característica ou função de coalizão do jogo.

2.2 Mercado

Maschler, Solan e Zāmîr (2020) definem Mercado como um conjunto composto de produtores $N = \{1,2,\ldots,n\}$ que negociam l mercadorias, sendo o conjunto delas dado por $L = \{1,2,...,l\}$. Essas mercadorias podem ser combinadas pelos produtores para produzir bens de diferentes tipos, por exemplo, equipamentos eletrônicos e produtos alimentícios. Assim, sejam os números reais não-negativos $\mathbb{R}_+ := [0,\infty)$, um vetor de mercadorias é representado por $x = (x_j)_{j=1}^{L} \in \mathbb{R}_+^L$ e será chamado de pacote de mercadorias. O pacote do produtor i é dado por x_i e a quantidade de cada mercadoria j representada por $x_{i,j}$.

Cada produtor tem a sua disposição uma tecnologia de produção representada por uma função de produção $u_i : \mathbb{R}_+^L \to \mathbb{R}$: se $x_i \in \mathbb{R}_+^L$ é o pacote de mercadorias do produtor i que pode então produzir uma soma de valor igual a $u_i(x_i)$. As funções de produção podem ser diferentes entre produtores, portanto a quantidade de valor que uma unidade da mercadoria j gera, pode variar de um produtor para outro. Assume-se que todo produtor i tem uma dotação inicial de mercadorias $a_i \in \mathbb{R}_+^L$ e que os produtores podem negociar entre eles.

Se um mercado S é formado, seus membros negociam mercadorias entre si com o objetivo de maximizar o valor que produzirão. O total das dotações iniciais de mercadorias disponíveis é dado por $a(S) := \sum_{i\in S} a_j \in \mathbb{R}_+^L$. Assim, aloca-se para cada um dos integrantes um pacote de mercadorias $x_i \in \mathbb{R}_+^L$, sujeito à restrição imposta pela Equação 2.1. Com a realocação das mercadorias no mercado, os membros produzem um valor total igual a $\sum_{i\in S} u_i(x_i)$.

$$x(S) = \sum_{i\in S} x_i = \sum_{i\in S} a_i = a(S) \tag{2.1}$$

Portanto, formalmente, um mercado é descrito por um vetor $(N, L, (a_i, u_i)_{i\in N})$ tal que:

- $N = \{1,2,...,n\}$ é o conjunto de produtores;
- $L = \{1,2,...,l\}$ é o conjunto de mercadorias;
- $\forall i \in N$, $a_i \in \mathbb{R}_+^L$ é o vetor de dotações iniciais do produtor i;
- $\forall i \in N$, $u_i : \mathbb{R}_+^L \to \mathbb{R}$ é a função de produção do produtor i.

Ao assumir que $a_i \in \mathbb{R}_+^L$ para todo $i \in N$, implica na existência de uma quantidade finita de cada mercadoria no Mercado. Por fim, considera-se que uma alocação para a coalizão S é a coleção de pacotes de mercadorias $(x_i)_{\in S}$, onde $x_i \in \mathbb{R}_+^L$ para todo produtor $i \in N$ satisfazendo $x(S) = a(S)$.

2.3 Jogo de Mercado

Todo Mercado pode ser associado a um Jogo de Coalizão (MASCHLER; SOLAN; ZĀMÎR, 2020), onde o conjunto de jogadores é o conjunto de produtores $N = \{1, 2, ..., n\}$ e o valor agregado de cada coalizão não vazia $S \subseteq N$ é dado pela Equação 2.2.

$$v(S) = \max \left\{ \sum_{i \in S} u_i(x_i) : x = (x_i)_{i \in S} \in X^S \right\} \tag{2.2}$$

Ou seja, o valor agregado de uma coalizão S é o máximo de valor que pode ser produzido por seus membros quando negociam mercadorias entre si.

O Jogo de Coalizão $(N; v)$ é definido como sendo um Jogo de Mercado derivado do Mercado $(N, L, (a_i, u_i)_{i \in N})$. Formalmente, um Jogo de Coalizão $(N; v)$ é um Jogo de Mercado se existe um número positivo l, um pacote inicial de mercadorias $a_i \in \mathbb{R}^L_+$ para todo jogador $i \in N$ e uma função de produção $u_i : \mathbb{R}^L_+ \to \mathbb{R}$ côncava e contínua tal que a Equação 2.2 seja satisfeita para toda coalizão $S \in P(N)$.

2.4 Shapley Values

Shapley value (SHAPLEY, 1953) é um solução conceitual única para Jogos de Coalizão que, considerando que a grande coalizão N seja formada, responde como o valor agregado $v(N)$ será dividido entre seus membros. A noção de contribuição marginal, ou seja, a quantidade com a qual a participação de cada membro incrementa o valor agregado, é central para a definição de *Shapley value*. De forma axiomática é definido como uma solução conceitual única que satisfaz as seguintes propriedades (MASCHLER; SOLAN; ZĀMÎR, 2020):

- Simetria: dados os jogadores i e j cujas contribuições marginais sejam iguais para toda as coalizões que não os contenham, adicionar o jogador i à coalizão é equivalente a adicionar o jogador j à coalizão. O axioma da simetria garante que jogadores tratados de forma idêntica pela função de coalizão recebam o mesmo valor ao final na distribuição dos ganhos;

- Jogador nulo: se um jogador não contribui para coalizão alguma, ele é um jogador nulo e não recebe nada do valor agregado final gerado pela coalizão;

- Aditividade: o valor agregado para a soma de dois jogos independentes deve ser igual à soma dos valores agregados gerados por cada um deles separadamente; e

- Eficiência: assume que quando a grande coalizão N é formada, o valor agregado final é inteiramente distribuído entre seus membros.

Seja i um jogador e S uma coalizão arbitrária que não inclua o jogador i, a Equação 2.3 calcula o *Shapley value* de i. É a soma das contribuições marginais de i para todas as possíveis coalizões formadas pelas permutação dos outros jogadores.

$$Sh_i(N;v) = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|! \times (n - |S| - 1)!}{n!} (v(S \cup \{i\}) - v(S)) \tag{2.3}$$

O cálculo exato de *Shapley Values* é um problema $\mathscr{NP}$-difícil e que, geralmente, necessita de métodos de aproximação para modelagens não-triviais (MITCHELL *et al.*, 2022). A complexidade de seu cálculo $O(2^n)$ para n jogadores, torna o tempo de execução intratável quando existem muitos jogadores. Assim, diversos métodos de aproximação baseados em amostragem tem sido propostos. Neste trabalho utilizou-se o método proposto por Mitchell *et al.* (2022), pois provê soluções rápidas com uma margem de erro aceitável. Todas as simulações realizadas utilizaram o tamanho de amostra igual a 1000.

CAPÍTULO

3

TRABALHOS RELACIONADOS

Diversos estudos têm sido propostos para alocar tarefas de forma eficiente em ambientes de nuvens veiculares. No trabalho de Luo et al. (LUO *et al.*, 2021), os autores examinaram o atraso de processamento e o custo monetário associados à alocação de tarefas em VCs. Nesse trabalho, foi desenvolvido um quadro de agendamento que leva em consideração a comunicação e a computação em VCs, considerando tarefas com diferentes requisitos. Assim, um problema de otimização multiobjetivo foi formulado com o objetivo de minimizar o atraso e o custo monetário. Para alcançar isso, foi proposto um algoritmo de agendamento com base na Otimização por Enxame de Partículas (PSO - *Particle Swarm Optimization*) para obter soluções Pareto ótimas. No entanto, devido à abordagem bioinspirada do algoritmo, o tempo de convergência pode afetar o desempenho global da solução. Além disso, é importante notar que os autores não consideraram a mobilidade dos veículos como requisito no processo de agendamento.

Wu et al. investigaram o problema de agendamento de tarefas e otimização de alocação de recursos, levando em consideração o efeito da mobilidade dos veículos no ambiente VEC (WU *et al.*, 2020). Especificamente, os autores formularam o problema de otimização em uma perspectiva Min-Max para reduzir a latência geral no agendamento de tarefas. Além disso, considerando os padrões de movimento relativamente estáveis dos veículos em um curto período, foi contemplada a previsão de mobilidade para projetar um esquema baseado em previsão de mobilidade, visando obter melhores resultados. No entanto, o modelo de mobilidade é considerado irrealista, uma vez que os veículos precisam de aceleração constante durante o agendamento de tarefas e alocação de recursos.

Pereira et al. (PEREIRA *et al.*, 2020) propuseram o algoritmo RELIABLE para alocação de recursos com base no Processo Analítico Hierárquico (AHP - *Analytic Hierarchy Process*). O AHP é um método matemático de múltiplos critérios que auxilia na tomada de decisões complexas. O RELIABLE aplica um fator de influência e utiliza uma matriz de decisão para determinar o melhor local para executar um serviço. O algoritmo utiliza características de Computação em Borda Móvel (MEC - *Mobile Edge Computing*) como parâmetros, calcula um

fator de influência para cada parâmetro e, em seguida, usa a matriz de decisão para selecionar o melhor MEC. Os autores aplicaram o RELIABLE em um ambiente rodoviário com infraestruturas rodoviárias com capacidade de processamento chamadas de Unidades de Acostamento (RSU - *Roadside Units*), cada uma formando um MEC. Cada MEC possui três características: previsão de mobilidade, largura de banda e tempo de serviço. O RELIABLE é executado quando um veículo precisa de recursos de névoa e, com base no número de recursos de cada MEC, decide se o serviço pode ser atendido e onde alocar os recursos. No entanto, a chegada de tarefas no sistema é baseada no momento em que o veículo entra na área de cobertura da RSU. Além disso, a mobilidade dos veículos segue o modelo *Random Way Point*. A mobilidade dos veículos e as taxas de chegada de tarefas não seguem modelos realistas.

Da Costa et al. apresentaram um mecanismo de agendamento de tarefas para VCs baseado em otimização combinatória (da Costa *et al.*, 2020). Após o processo de formação de VC, um controlador localizado em um nível superior da rede recebe os pedidos de recursos e os agenda para processamento nas VCs disponíveis usando um algoritmo pseudo-polinomial para o problema da mochila 0/1. No entanto, os autores não consideram aspectos contextuais no processo de tomada de decisão de agendamento. Ou seja, o impacto da mobilidade veicular e os requisitos de tempo das tarefas são desconsiderados.

Wei et al. (WEI *et al.*, 2021) formularam a alocação de recursos nem nuvens veiculares como um problema multiobjetivo para reduzir a probabilidade de bloqueio e o custo conjuntamente. Os autores propuseram uma versão aprimorada do algoritmo genético de classificação não dominada II (NSGA-II - *Non-dominated Sorting Genetic Algorithm II*), chamada AC-INSGA-II. O AC-INSGA-II melhora a população inicial, as probabilidades de mutação e cruzamento para evitar otimização local e aumentar a diversidade. A solução proposta concentra-se em utilizar os recursos com equilíbrio de carga de forma eficiente. O AC-INSGA-II foi implementado considerando uma arquitetura de três camadas com uma nuvem central, RSUs e nuvens veiculares. Portanto, a solução considera recursos de nuvem tradicionais em centros de dados e aqueles fornecidos por veículos na rede veicular. A RSU recebe uma solicitação e decide se ela será processada pela nuvem central ou veicular, levando em consideração os requisitos. Além disso, a solução não utiliza computação em borda, o que pode aumentar a latência nas solicitações de serviços.

Lee et al. (LEE; LEE, 2020) abordaram o problema de alocação de recursos considerando a mobilidade dos veículos, especialmente durante as horas de pico. Devido às restrições de recursos de hardware/software da névoa veicular, apenas um número limitado de veículos pode utilizar os recursos de computação em névoa. O estudo considera veículos de movimentação lenta e veículos estacionados como recursos disponíveis, que podem ser usados para processar tarefas com o objetivo de obter uma resposta em tempo real. Assim, as soluções propostas visam minimizar a latência na execução das tarefas. Nos casos em que os cenários e padrões de tráfego carecem de veículos lentos e estacionados, as soluções propostas utilizam as RSUs disponíveis

para executar as tarefas, e para tarefas computacionalmente intensivas, a nuvem central também é utilizada. Os autores propuseram uma heurística baseada em Aprendizado por Reforço (RL - *Reinforcement Learning*) para a alocação de recursos. O RL é utilizado para entender o padrão de mobilidade a fim de alocar recursos de forma eficiente, considerando a mobilidade dos veículos. O mecanismo de alocação de recursos proposto é baseado em computação em névoa para redes veiculares, utiliza RL e um algoritmo heurístico para melhorar a eficiência na alocação de recursos. No entanto, o modelo considera recursos computacionais de nuvens remotas e não utiliza apenas os recursos compartilhados pelos veículos em nuvens veiculares.

Tang et al. (TANG *et al.*, 2020) propuseram um algoritmo de estratégia gananciosa que considera uma função de utilidade para a alocação de tarefas em VCs, chamado DbHA. Esse algoritmo leva em consideração os recursos de computação e comunicação necessários pelas tarefas para o processo de tomada de decisão. Os autores calculam a matriz Hessiana de uma função com tais variáveis e obtêm os valores ótimos de computação e comunicação quando o valor máximo da função de utilidade é alcançado. Após esse processo, a distância euclidiana entre os valores ótimos e os valores de cada solicitação é calculada e alimentada a uma *heap* para obter a distância mínima de forma eficiente. O mecanismo de alocação de tarefas é baseado em um algoritmo genético e em PSO. O mecanismo é comparado a outras abordagens em relação ao desempenho da rede e eficiência na utilização de recursos. Além disso, o mecanismo utiliza recursos de centros de dados para alocação de recursos, o que naturalmente pode introduzir um atraso indesejado no sistema.

Marques et al. (MARQUES; MENEGUETTE, 2021) propuseram uma política de alocação de recursos baseada na Teoria dos Jogos para maximizar a utilização de recursos em VCs, analisando o equilíbrio de Nash do jogo proposto. Os autores formulam o problema como um jogo não cooperativo em que os jogadores, neste caso, os veículos, têm estratégias para oferecer ou não um recurso e consumir ou não um serviço. A ideia principal do modelo é que tanto os consumidores, quanto os provedores de recursos tenham algum ganho. No entanto, um equilíbrio de Nash não é necessariamente um ótimo de Pareto (SUN *et al.*, 2021), o que pode levar a soluções ineficientes dependendo do perfil de estratégias escolhido. Além disso, o trabalho também não considera variações nas taxas de serviços na rede.

Yu et al. estudaram o compartilhamento e gerenciamento da largura de banda para aplicativos móveis em ambientes de nuvens veiculares (YU *et al.*, 2015). Eles propuseram um modelo de jogo de coalizão para apoiar a cooperação entre provedores de serviços para compartilhar seus recursos ociosos. O estudo considerou diferentes taxas de serviços, e os resultados mostraram uma utilização de recursos computacionais acima de 75% em comparação com cenários sem cooperação. Para criar um mecanismo cooperativo de alocação de recursos, os veículos se comunicam entre si e com estações base para compartilhar informações sobre suas necessidades de recursos. Essa comunicação visa melhorar a alocação de recursos de rede, como largura de banda, memória e capacidade de processamento. No entanto, este trabalho também

utiliza recursos externos, o que pode não ser viável em alguns cenários.

Da Costa et al. (COSTA *et al.*, 2020) propuseram um mecanismo que otimiza a alocação de tarefas em nuvens veiculares. Denominado MORFEU, este algoritmo utiliza técnicas de otimização combinatória para determinar a melhor alocação de tarefas entre os veículos, levando em consideração a carga de processamento dos veículos, sua localização e a conectividade com outros veículos. Para avaliar o desempenho, o mecanismo foi comparado com outros métodos existentes na literatura e os resultados mostraram que o MORFEU pode reduzir significativamente o tempo de processamento e melhorar a eficiência na alocação de tarefas. No entanto, esse trabalho não considera diferentes taxas de serviços e raios de comunicação.

Fan et al. (FAN *et al.*, 2023) propuseram um esquema de alocação de tarefas e recursos com o objetivo de minimizar o atraso total no processamento das tarefas de todos os veículos. A solução categoriza os veículos em 4 grupos, dependendo se eles solicitam recursos individuais para realizar suas tarefas ou compartilham recursos com outros. Os autores formulam a alocação de recursos como um problema de otimização e desenvolvem um algoritmo com base em técnicas de Decomposição de Benders Generalizada e Reformulação de Linearização para resolver o problema de otimização de programação não linear inteira mista de maneira ideal. Contudo, uma vez que este problema é $\mathscr{NP}$-difícil, eles também criaram uma solução heurística sub-ótima de baixa complexidade computacional. O processamento das tarefas pode ocorrer localmente no veículo ou com o auxílio de recursos provenientes dos servidores de borda localizados nas RSUs. Portanto, o modelo considera não apenas os recursos computacionais disponíveis nos veículos, mas também os recursos oferecidos por servidores remotos.

Liu et al. (LIU *et al.*, 2023) investigaram o processamento colaborativo de tarefas e a alocação de recursos sob demanda. O modelo proposto pelos autores considera camadas de cooperação onde os usuários com recursos limitados têm a capacidade de determinar onde as tarefas serão executadas: localmente, nos servidores de borda ou na nuvem computacional. Os autores também introduzem a noção de que cada tarefa, além dos recursos necessários para sua execução, possui um valor de ganho associado. Esse valor é utilizado pelo algoritmo A3C (*Advantage Actor-Critic*), que é um método de aprendizado por reforço profundo (DRL - *Deep Reinforcement Learning*) utilizado para encontrar a solução ótima. O processamento é efetuado de forma descentralizada, permitindo que a atualização dos gradientes da função de perda da rede neural seja realizada de maneira independente por cada um dos agentes, sem afetar a rede de forma global. Mais uma vez, é importante salientar que o modelo proposto leva em consideração não apenas os recursos computacionais disponíveis nos veículos, mas também os recursos oferecidos pelos servidores, abrangendo servidores na borda da rede e também em nuvens remotas.

Kouser e Manikandan (KOUSER; MANIKANDAN, 2023) desenvolveram um algoritmo bioinspirado denominado CM-CSO (*Crossover and Mutation-centered Chicken Swarm Optimization*) com o propósito de realizar a alocação de recursos disponíveis nas nuvens veiculares

formadas. O algoritmo proposto recupera informações dos veículos que, fazendo uso de outro algoritmo bioinspirado chamado BM-BFO (Brownian motion-centered Bacteria Foraging Optimization), extrai as características que melhor representam o conjunto de veículos. Com base nas características selecionadas, os veículos são agrupados em VCs por meio de um algoritmo *K-means* modificado. Em seguida, a solução coleta informações de recursos de servidores localizados nas nuvens computacionais tradicionais. O CM-CSO utiliza todas essas informações para efetuar a alocação de recursos na rede. Destaca-se que a solução não apenas utiliza recursos distintos dos presentes nos veículos por meio do uso de servidores remotos, mas também opta por não utilizar *traces* reais de mobilidade em suas simulações.

Sun et. al. (SUN *et al.*, 2023) apresentaram um framework hierárquico baseado na Teoria de Jogos com o objetivo de otimizar a alocação de recursos entre veículos na borda da rede e o equilíbrio de carga entre servidores na borda e servidores em nuvens tradicionais. Esse framework promove a colaboração entre os veículos, os servidores na borda e o servidor na nuvem. A solução heurística, denominada BARGAIN-MATCH, que faz uso de recursos que vão além daqueles compartilhados pelos veículos, aborda a alocação de recursos entre servidores na borda da rede (colaboração horizontal) por meio de um método de barganha para precificar dinamicamente o valor recebido por atender uma tarefa, e entre servidores e a nuvem computacional (colaboração vertical), empregando um método de pareamento entre oferta e demanda por recursos.

Neste contexto, este trabalho modela, apresenta e implementa uma solução heurística denominada **HARMONIC** (s*H*apley v**A**lues in ma**R**ket ga**M**es for res**O**urce allocatio**N** in veh**I**cular **C**louds), que organiza as VCs como um único mercado, estabelecendo um Jogo de Mercado com o objetivo de otimizar a utilização de recursos disponíveis. O HARMONIC leva em consideração Taxas Variáveis de Solicitações de Serviços (TVS) e Diferentes Faixas de Alcance de Comunicação (DFAC) em cenários dinâmicos. Além disso, o método utiliza *Shapley Values* para determinar a ordem de atendimento às solicitações de serviços com foco exclusivo nos recursos compartilhados pelos veículos. A ordem é determinada respeitando as propriedades elencadas na Seção 2.4 do Capítulo 2 que proporciona uma Distribuição Equilibrada de Carga (DEC) entre as VCs.

A Tabela 1 resume esta seção e apresenta os seguintes critérios para comparação e contraste dos trabalhos relacionados e a solução implementada: **Arquitetura** indica se a tomada de decisão no processo de alocação de tarefas é feita de forma centralizada ou descentralizada; **Recursos Externos** indica se recursos alheio àqueles compartilhados pelos veículos são utilizados; **Agregação Prévia** indica se os recursos compartilhados pelos veículos são agregados antes mesmo que existam solicitações de serviços na rede; **TVS** indica se taxas variáveis de solicitações de serviços são consideradas; **DFAC** indica se diferentes faixas de alcance (raio) de comunicação são consideradas ao determinar quais serão as nuvens veiculares criadas; **DEC** indica se a solução leva em consideração o balanceamento de cargas na rede veicular; e ***Trace***

Real indica se foi utilizado *Trace* de mobilidade dinâmico e real durante as simulações.

Tabela 1 – Resumo dos trabalhos relacionados

Trabalho	Características							
	Arquitetura	Recursos Externos	Agregação Prévia	TVS	DFAC	DEC	*Trace* Real	Método
Luo et al. (LUO *et al.*, 2021)	Descentralizada	✓	✓					PSO
Pereira et al. (PEREIRA *et al.*, 2020)	Descentralizada		✓		✓	✓		AHP
Da Costa et al. (da Costa *et al.*, 2020)	Centralizada						✓	Otimização combinatória
Wu et al. (WU *et al.*, 2020)	Centralizada	✓	✓					Otimização minimax
Wei et al. (WEI *et al.*, 2021)	Descentralizada	✓	✓					Algoritmo genético
Lee et al. (LEE; LEE, 2020)	Centralizada	✓	✓	✓			✓	RL
Tang et al. (TANG *et al.*, 2020)	Centralizada	✓	✓					Algoritmo genético
Marques et al. (MARQUES; MENEGUETTE, 2021)	Centralizada		✓					Teoria dos Jogos
Yu et al. (YU *et al.*, 2015)	Centralizada	✓		✓		✓		Teoria dos Jogos
Da Costa et al. (COSTA *et al.*, 2020)	Centralizada						✓	Otimização linear
Fan et al. (FAN *et al.*, 2023)	Descentralizada	✓				✓		Otimização não linear
Liu et al. (LIU *et al.*, 2023)	Descentralizada	✓				✓		DRL
Kouser e Manikandan (KOUSER; MANIKANDAN, 2023)	Centralizada	✓				✓		Otimização bioinspirada
Sun et. al. (SUN *et al.*, 2023)	Centralizada	✓				✓		Teoria dos Jogos
HARMONIC	Centralizada			✓	✓	✓	✓	Teoria dos Jogos

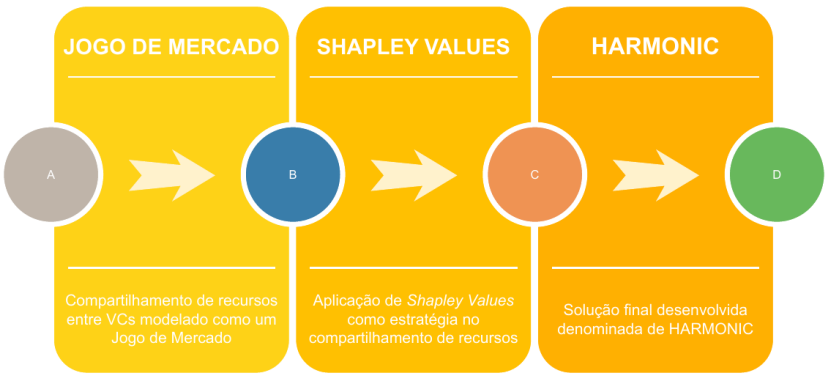
CAPÍTULO

4

ALOCAÇÃO DE RECURSOS EM REDES VEICULARES BASEADA EM TEORIA DOS JOGOS

Neste capítulo, são apresentadas a modelagem e a implementação final desenvolvidas para o problema de alocação de recursos em nuvens veiculares, juntamente com as etapas seguidas ao longo de seu desenvolvimento. Para proporcionar uma compreensão completa da evolução deste projeto, o capítulo é dividido em três seções correspondentes aos objetivos específicos. Nessas seções, são detalhadas a modelagem do problema como um Jogo de Mercado (Seção 4.1), a aplicação de estratégias baseadas em *Shapley Values* para otimizar a alocação de recursos, e, por fim, é apresentada a solução final denominada HARMONIC (Seção 4.3). A Figura 1 ilustra a sequência das fases de evolução do projeto, enquanto o Quadro 1 detalha a notação empregada na descrição dos algoritmos neste capítulo.

Figura 1 – Evolução do projeto - Visão geral



Fonte: Elaborada pelo autor.

Quadro 1 – Notação utilizada na implementação dos algoritmos

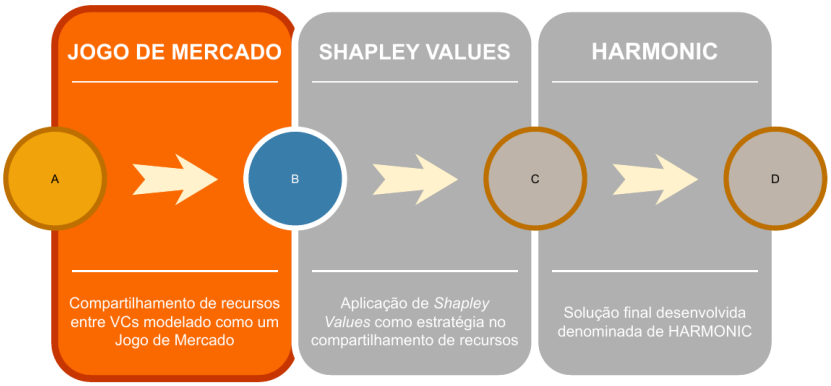
Notação	Definição
A	Dotações iniciais
A_c	Dotação inicial do VCC
C	Coalizão formada pelas VCs e o VCC
S	Tarefas atendidas pela coalizão
T	Tarefas para alocação
T_a	Tarefas alocadas pela VC
U	Valor agregado gerado pela coalizão
U_i	Valor agregado gerado pela VC
V	Nuvens veiculares
VCC	Controlador de nuvem veicular
X	Histórico de tarefas alocadas por cada VC

Fonte: Elaborada pelo autor.

4.1 Compartilhamento de recursos entre nuvens veiculares modelado como um Jogo de Mercado

Esta seção descreve a primeira fase do projeto (Figura 2), a utilização do conceito de Jogos de Mercado para a modelagem do problema de alocação de recursos entre nuvens veiculares (VCs). Um Jogo de Mercado, conforme apresentado no Capítulo 2, é um jogo de coalizão onde os produtores se organizam em um Mercado a fim de maximizar o ganho com suas mercadorias. A seguir, apresenta-se o modelo desenvolvido, que organiza as VCs como um Jogo de Mercado, com o objetivo de maximizar a alocação de tarefas na rede veicular e, consequentemente, a otimização do uso dos recursos disponíveis.

Figura 2 – Fase 1 - Modelagem como um Jogo de Mercado



Fonte: Elaborada pelo autor.

O modelo proposto é um Jogo de Mercado derivado do Mercado $(N, L, (a_i, u_i)_{i \in N})$ tal que:

- $N = \{1,2,...,n\}$ é o conjunto formado pelas VCs e o VCC;
- $L = \{1,2,...,l\}$ é o conjunto de pesos das tarefas T, onde $l = \max\{peso(tarefa_i)_{i \in T})\}$. O peso de uma tarefa é determinado pela quantidade de recursos computacionais exigidos para sua alocação na rede veicular. Esses recursos incluem, mas não se limitam a, capacidade de processamento, armazenamento ou memória demandados;
- $\forall i \in N$, $a_i \in \mathbb{R}_+^L$ é o vetor de *slots* disponíveis, sendo que $a_{i,j}$ representa a quantidade de tarefas de peso j alocadas se i é uma VC ou a serem alocadas se i representa o VCC. A Equação 4.1 apresenta os valores iniciais para os *slots*, onde $|T_j|$ representa a quantidade de tarefas com peso j a serem alocadas pelo VCC.
- $\forall i \in N$, $u_i : \mathbb{R}_+^L \to \mathbb{R}$ é a função de produção do participante i (VCs ou VCC) definida conforme a Equação 4.2, onde $|T_{i,j}|$ representa a quantidade de tarefas com peso j alocadas na VC i.

$$a_{i,j} = \left\{ \begin{array}{ll} 0, & \text{se VC} \\ |T_j|, & \text{se VCC} \end{array} \right\} \forall j \in [1,l] \tag{4.1}$$

$$u_i = \left\{ \begin{array}{ll} \sum_{j=1}^{l} j|T_{i,j}|, & \text{se VC} \\ 0, & \text{se VCC} \end{array} \right\} \forall j \in [1,l], i \in N \tag{4.2}$$

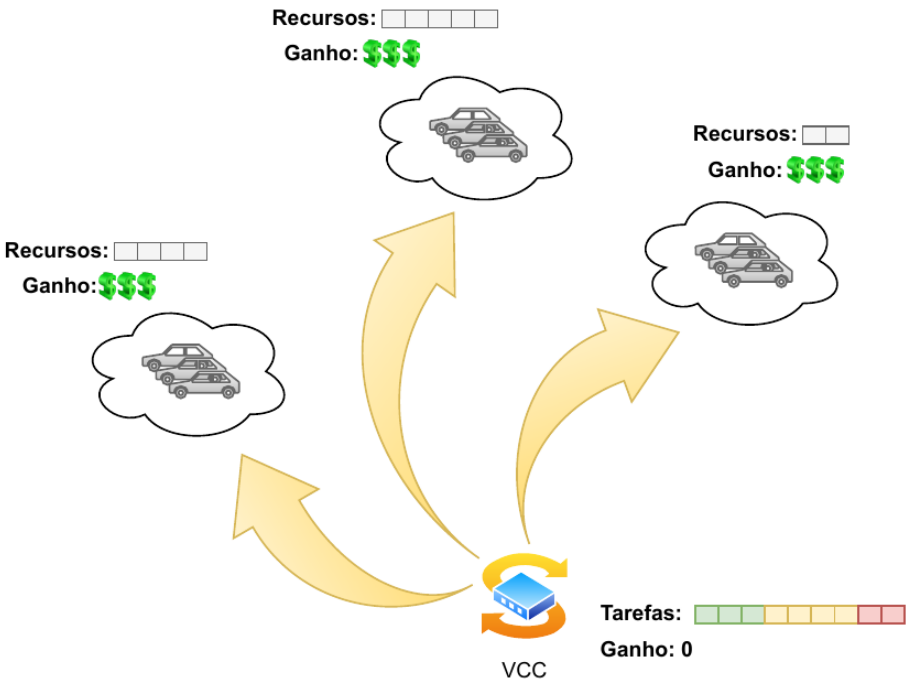
Portanto, o valor agregado produzido por uma VC é igual à soma total do número de tarefas alocadas multiplicadas pelos seus respectivos pesos j. Assim, o VCC ao alocar tarefas nas VCs participantes da coalizão gera um valor agregado total correspondente a $\sum_{i \in N} u_i(x_i)$ conforme mostrado na seção 2.2. Assim, para converter o Mercado anterior em um Jogo de Mercado é necessário que se atenda à Equação 2.2 sujeita às restrições de recursos computacionais disponíveis Ω_i de cada VC_i do mercado. Desta forma, o modelo é formalizado de acordo com a Equação 4.3.

$$\begin{array}{ll} \textit{Maximizar} & \sum_{i \in S} u_i(x_i) \\ \text{sujeito a} & u_i(x_i) \leq \Omega_i \end{array} \tag{4.3}$$

A Figura 3 ilustra o modelo que foi desenvolvido, onde as VCs e o VCC constituem o mercado e assumem o papel de produtores. O modelo é organizado de modo que, inicialmente, o VCC está ciente da lista de tarefas que necessitam de recursos para serem executadas, assim como das VCs que possuem recursos disponíveis. Inicialmente, nenhuma das tarefas está alocada nas nuvens (Equação 4.1).

Para otimizar a alocação de tarefas nas nuvens veiculares, foram modeladas as funções de produção (valor agregado) dos produtores da seguinte maneira (Equação 4.2): para o VCC, essa função é definida como 0; e para as VCs, é diretamente proporcional tanto ao número quanto

Figura 3 – Ilustração do Jogo de Mercado Modelado



Fonte: Elaborada pelo autor.

ao peso das tarefas alocadas. Isso significa que o modelo assegura que não há vantagem em manter tarefas inativas no VCC. Assim, as tarefas começam a fluir na direção indicada pelas setas amarelas na Figura 3, direcionando-as para as VCs a fim de maximizar a utilização dos recursos na rede, conforme demonstrado na Equação 4.3.

O Algoritmo 1 detalha a implementação da solução proposta. Inicialmente, forma-se o mercado (linha 2) entre as VCs e o VCC, inicializam-se as dotações iniciais dos participantes do mercado (linha 3) conforme Equação 4.1 e a dotação inicial do VCC é destacada (linha 4), pois representa as demandas por recursos computacionais necessárias às tarefas a serem alocadas nas VCs. Em seguida cria-se uma matriz (linha 5) com a finalidade de manter todo o histórico de pacotes de recursos alocados em cada um dos participantes da coalizão, o valor agregado total gerado (linha 6), a lista de tarefas atendidas é inicializada (linha 7) e por fim ordenam-se as VCs em ordem decrescente de recursos disponíveis (linha 8).

Para cada uma das VCs do mercado (linha 9), se ainda existir tarefa na lista do VCC a ser alocada (linha 10), a solução continua a procura por recursos nas VCs. É a função MERCADO que realiza a busca por recursos na nuvem (linha 11) e está descrita no Algoritmo 2. Por fim, são atualizados o valor agregado total da coalizão (linha 12), a lista de tarefas ainda não atendidas (linha 13) e a lista de tarefas atendidas até o momento (linha 14). Ao encerrar a busca por recursos para atender o conjunto de tarefas informado, o algoritmo retorna o valor agregado gerado pela coalizão, a lista de tarefas atendidas, não atendidas e o histórico detalhado de recursos utilizados em cada uma das VCs (linha 17).

Algoritmo 1 – Implementação do Jogo de Mercado

1: **função** JOGO(T, V, VCC) ▷ T - conjunto de tarefas, V - nuvens, VCC - controlador
2: $C = V \cup VCC$ ▷ C - formação do mercado
3: $A = DotacoesIniciais($C,T$)$
4: $Ac = A[N]$ ▷ N - número de participantes do mercado C
5: $X \leftarrow \vec{0}$
6: $U = 0$
7: $S = \emptyset$
8: $V = OrdenarDec(V, recurso\_disponivel)$
9: **para** $i = 1$ **até** $N - 1$ **faça** ▷ N-1 - VCs. A posição N é o VCC.
10: **se** $\sum(Ac) \neq 0$ **então**
11: $X[i], Ui, Ta = MERCADO(Ac, V[i].recurso\_disponivel, T)$ ▷ Algoritmo 2
12: $U = U + Ui$
13: $T = T - Ta$
14: $S = S \cup Ta$
15: **fim se**
16: **fim para**
17: **retorna** U, S, T, X
18: **fim função**

O Algoritmo 2 detalha a função MERCADO. Como entrada, são fornecidas as demandas por recursos, o total de recursos disponíveis na VC e as tarefas a serem alocadas. Inicialmente, a solução procura pela maior necessidade de recursos entre as tarefas a serem alocadas (linha 2), inicializa o valor agregado a ser gerado pela alocação de tarefas na VC (linha 3), a lista de tarefas atendidas (linha 4) e o histórico de alocação de tarefas por número de recursos na VC (linha 5).

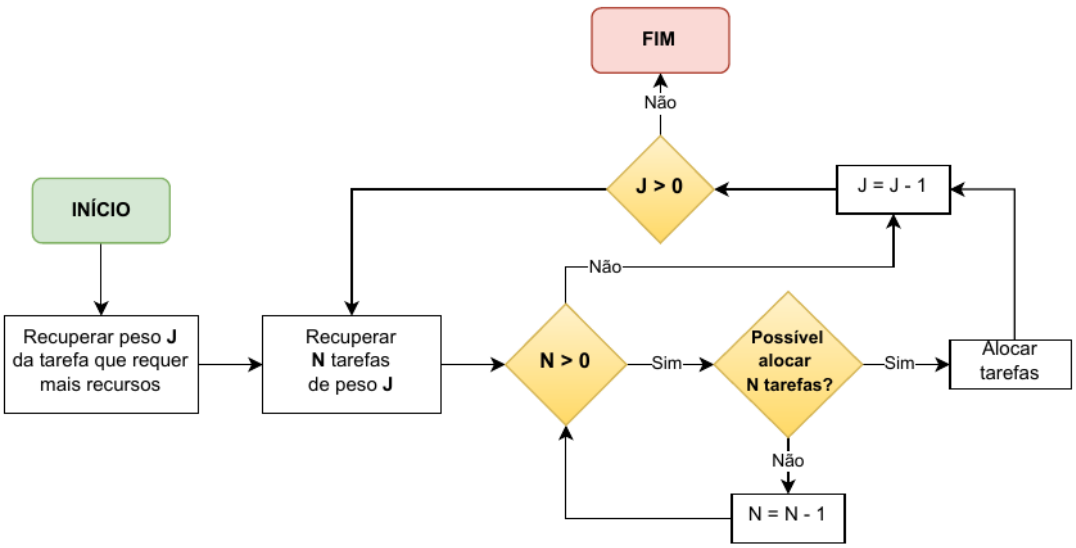
A seguir, percorrendo a lista de tarefas em ordem decrescente de necessidade por recursos (linha 6), o algoritmo encontra quantas são as tarefas de peso j e que podem ser alocadas na VC (linha 7-14). Caso seja possível alocar ao menos uma tarefa (linha 15), o valor agregado gerado pela VC é atualizado (linha 16), assim como a demanda restante por recursos (linha 17) e o histórico de alocações de tarefas alocados por quantidade de recursos (linha 18). Por fim, são selecionadas as n primeiras tarefas que necessitam de j recursos e a lista de tarefas atendidas é atualizada (linha 19). O algoritmo retorna o histórico de alocações, o valor agregado gerado pela alocação de tarefas na VC atual e a lista de tarefas alocadas (linha 22). A Figura 4 apresenta a solução implementada em forma de fluxograma.

A Figura 5 exemplifica a execução dos algoritmos. São apresentadas diferentes tarefas com pesos que variam de 1 a 6 unidades de recurso necessárias para alocação da tarefa. Existem 3 VCs que fazem parte do mercado juntamente com o VCC que não foi diretamente ilustrado. A utilização das VCs inicia-se pela nuvem VC2 que tem mais recursos disponíveis, em seguida a VC1 é considerada e finalmente VC3, que é a nuvem com menos recursos disponíveis. Essa ordem é destacada pelas letras A, B e C. Os polígonos amarelos mostram a evolução dos recursos disponíveis nas VCs a cada iteração do Algoritmo 2 e o polígonos vermelhos destacam os recursos restantes em cada nuvem ao fim da execução do algoritmo.

Algoritmo 2 – Execução do Mercado

função MERCADO(Ac, Ω, T) ▷ Ac - recursos necessários, Ω - recursos da VC, T - tarefas para alocação
  $L = ArgMax(\text{Ac})$ ▷ L - maior quantidade de recursos necessários entre as tarefas
  $U_i = 0$
  $Ta = \emptyset$
  $X = \vec{0}$
  para $j = L$ **até** 0 **faça**
    $n = Ac[j]$
    $U_{ij} = n * j$
    se $U_{ij} > \Omega$ **então**
      enquanto $n \neq 0$ **e** $U_i + U_{ij} > \Omega$ **faça**
        $n = n - 1$
        $U_{ij} = n * j$
      fim enquanto
    fim se
    se $n > 0$ **então**
      $U_i = U_i + U_{ij}$
      $Ac[j] = Ac[j] - n$
      $X_j = n$
      $Ta = Ta \cup NTarefasPesoK(\text{n,j})$
    fim se
  fim para
  retorna X, U_i, Ta
fim função

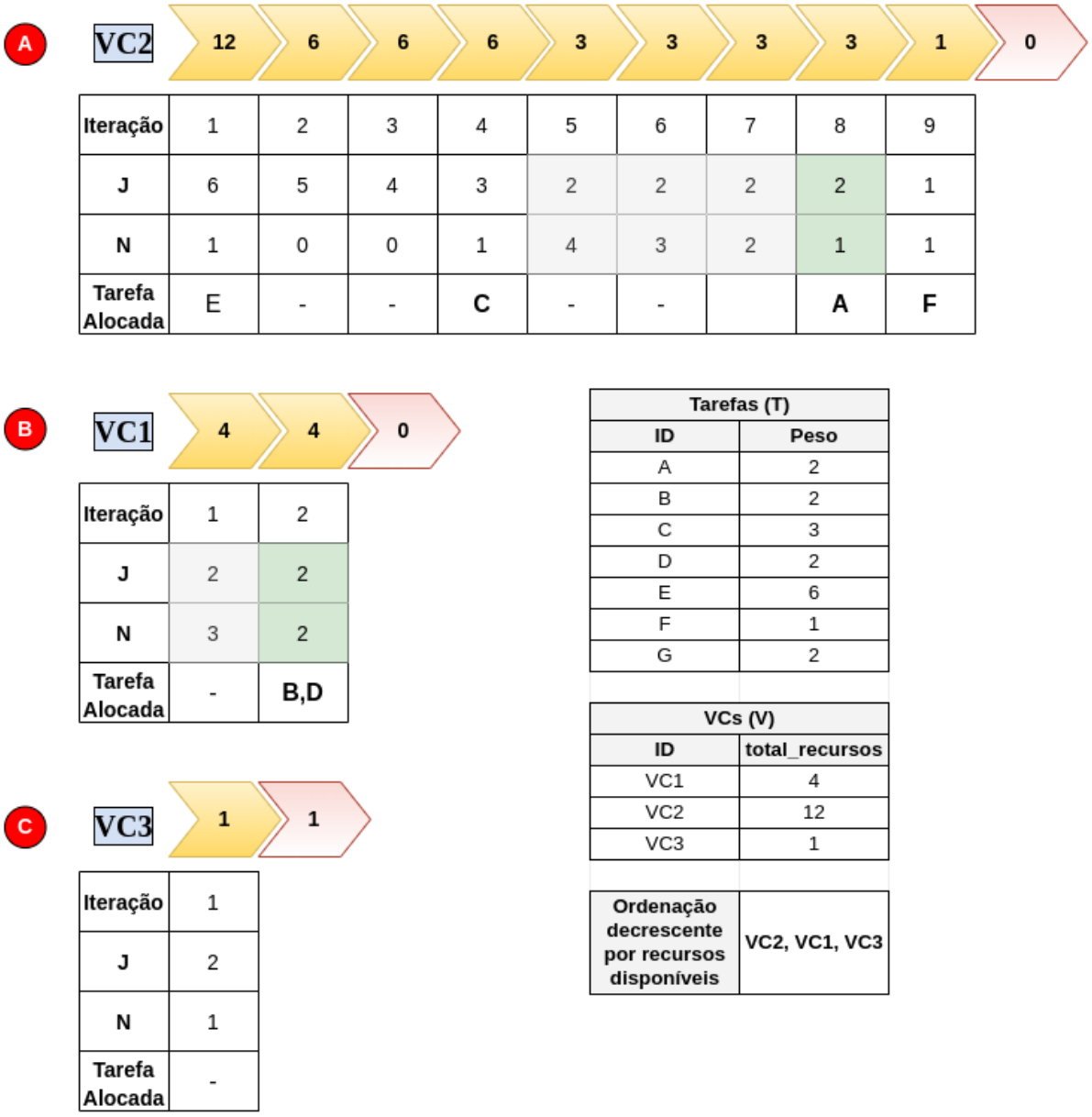
Figura 4 – Fluxograma da solução implementada na fase 1



Fonte: Elaborada pelo autor.

Na letra A, destacam-se as iterações de 5 a 8. No iteração 5, a VC2 tem 3 recursos disponíveis e existem 4 tarefas (linha N) de peso 2 (linha J). A solução testa a possibilidade de alocar em lote todas essas 4 tarefas, o que não é possível. Na iteração 6 verifica-se a possibilidade

Figura 5 – Exemplo da solução implementada na Fase 1



Iteração	1	2	3	4	5	6	7	8	9
J	6	5	4	3	2	2	2	2	1
N	1	0	0	1	4	3	2	1	1
Tarefa Alocada	E	-	-	C	-	-		A	F

Iteração	1	2
J	2	2
N	3	2
Tarefa Alocada	-	B,D

Iteração	1
J	2
N	1
Tarefa Alocada	-

Tarefas (T)	
ID	Peso
A	2
B	2
C	3
D	2
E	6
F	1
G	2

VCs (V)	
ID	total_recursos
VC1	4
VC2	12
VC3	1

Ordenação decrescente por recursos disponíveis	VC2, VC1, VC3

Fonte: Elaborada pelo autor.

de alocar 3 das 4 tarefas de peso 2, o que também não é viável uma vez que seriam necessárias 8 unidades de recursos para alocar todas essas tarefas, porém a VC2 tem somente 3 disponíveis no momento.

É então na iteração 8, ao tentar alocar 1 tarefa de peso 2, que é o algoritmo é bem sucedido uma vez que são necessários apenas 2 recursos e existem 3 disponíveis na VC. Assim, a tarefa é alocada e resta 1 recurso na VC2. Por fim, O algoritmo continua e, na iteração 9, a tarefa rotulada como F é alocada uma vez que requer apenas 1 recurso para sua alocação.

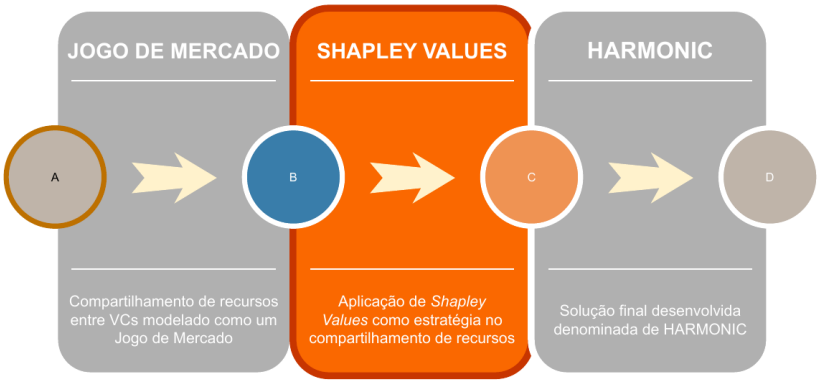
A próxima VC considerada é a VC1 (letra B) onde a execução da solução atua da mesma

maneira, porém somente com as tarefas ainda não alocadas. Em seguida, para finalizar, a VC3 (letra C) é verificada sem sucesso na alocação de tarefas e finaliza com 1 recurso não utilizado.

4.2 Aplicação de *Shapley Values* como estratégia no compartilhamento de recursos

Nesta segunda fase (Figura 6) observou-se que, considerar as tarefas em ordem decrescente de peso (necessidade de recursos) nem sempre leva à melhor alocação de tarefas e utilização de recursos.

Figura 6 – Fase 2 - O uso de *Shapley Values*



Fonte: Elaborada pelo autor.

Para verificar essa observação desenvolveu-se um novo modelo que, com o uso de *Shapley Values* (Capítulo 2), calcula a contribuição marginal de cada tarefa caso fossem alocadas e utiliza esses valores para definir a ordem em que as tarefas serão durante o processo de alocação. A Tabela 2 exemplifica um caso onde considerar a ordem das tarefas em ordem decrescente de peso não leva à melhor alocação.

Tabela 2 – Exemplo onde a alocação de tarefas em ordem decrescente de pesos não é a ideal. São 2 nuvens VC1 e VC2 com 4 e 3 recursos compartilhados e as tarefas A, B, C com pesos 3, 2 e 2 respectivamente.

Linha	Ordem das Tarefas	Tarefas Alocadas	Alocação Total
1	A, B, C	A, B	5
2	A, C, B	A, C	5
3	B, A, C	B, C, A	7
4	B, C, A	B, C, A	7
5	C, A, B	C, B, A	7
6	C, B, A	C, B, A	7

Fonte: Elaborada pelo autor.

Nessa tabela é apresentado um cenário onde existem 2 VCs (VC1, VC2) e 3 tarefas para alocação (A, B, C). As nuvens tem 4 e 3 recursos compartilhados e as tarefas tem pesos 3, 2

e 2 respectivamente. Utilizando o Algoritmo 1 desenvolvido na fase anterior, as VCs seriam utilizadas na seguinte ordem $[VC1, VC2]$ e as tarefas na ordem $[A, B, C]$ que resultaria no total de 5 recursos utilizados conforme destacado em amarelo na tabela. No entanto, a linha 3, destacada em verde, mostra uma ordenação diferente de tarefas $[B, C, A]$ que resultaria em um valor igual a 7 e portanto superior. As linhas 4, 5 e 6 também mostram ordens de alocação mais eficientes do que a linha 1.

O Algoritmo 3 apresenta a solução proposta e utiliza *Shapley Values* para determinar a ordem em que as tarefas serão alocadas. As tarefas que requerem uma quantidade de recursos maiores que o máximo que uma única VC pode suportar são descartadas (linha 2 e 3), pois as tarefas são consideradas indivisíveis e não podem ser parcialmente executadas por uma VC e parcialmente por outra. Em seguida, calculam-se os *Shapley Values* para cada uma das tarefas restantes (linha 4), ordena-se o conjunto de tarefas em ordem decrescente desses valores (linha 5) e executa-se o escalador de tarefas (linha 6) que é apresentado no Algoritmo 5.

Algoritmo 3 – Alocação com o uso de *Shapley Values* para definir a ordem das tarefas

```
1: função ALOCACAOSHAP(T, V)                          ▷ T - conjunto de tarefas, V - nuvens
2:     R_max = Max(V, recurso_disponivel)
3:     T = Filtrar(T, R_max)
4:     ShapT = CalcShapAprox(T, FuncaoColizao(T, V))   ▷ Algoritmo 4
5:     T = OrdenarDec(T, ShapT)
6:     EscalonadorTarefa(T, V)                         ▷ Algoritmo 5
7: fim função
```

Para calcular os *Shapley Values* correspondentes a cada uma das tarefas (linha 4) utilizou-se, por ser um problema $\mathcal{NP}$-difícil, um método de aproximação baseado em amostras conforme discutido na Seção 2.4 do Capítulo 2. Para esse cálculo é necessário modelar a função de coalizão, ou seja, a função que quantifica o valor agregado gerado pela alocação de uma tarefa em um determinado momento. Essa função é definida pelo Algoritmo 4.

Algoritmo 4 – Função de coalizão

```
1: função FUNCAOCOALIZAO(T, V)                        ▷ T - conjunto de tarefas, V - nuvens
2:     U = 0
3:     para t em T faça
4:         V_p = Filtrar(V, t.peso)
5:         V_e = EscolherVC(V_p)
6:         V_e.recurso_disponivel = V_e.recurso_disponivel − t.peso
7:         U = U + t.peso
8:     fim para
9:     retorna U
10: fim função
```

O Algoritmo 4 inicializa o valor agregado total da coalizão (linha 2). Para cada tarefa (linha 3), recupera-se quais são as VCs que ainda possuem recursos disponíveis (linha 4) e

escolhe-se aleatoriamente uma delas (linha 5). Em seguida, atualiza-se a quantidade de recursos disponíveis (linha 6) e incrementa-se o valor agregado total (linha 7). O algoritmo retorna qual seria o valor agregado total gerado pela lista de tarefas se fossem alocadas na ordem informada (linha 9).

A Tabela 3 ilustra um exemplo do cálculo de valor agregado para cada uma das tarefas consideradas em diferentes ordens. Ao final da tabela é calculada a média das contribuições marginais (valor agregado) de cada tarefa. Esses valores médios são exatamente o que seriam retornados ao se calcularem os *Shapley Values* correspondentes às tarefas. São esses os valores considerados pela solução proposta nesta seção para determinar a ordem de apresentação das tarefas ao escalonador; assim, no exemplo apresentado as tarefas serão ordenadas em ordem decrescente da seguinte maneira: $[B,A,C]$.

Tabela 3 – Exemplo de cálculo de valor agregado (U) de tarefas em diferentes ordens de alocação. São 2 nuvens VC1 e VC2 com 4 e 3 recursos compartilhados e as tarefas A, B, C com pesos 3, 2 e 2 respectivamente.

Linha	Ordem das Tarefas	U(A)	U(B)	U(C)	Alocação Total
1	A, B, C	3	2	0	5
2	A, C, B	3	2	0	5
3	B, A, C	2	3	2	7
4	B, C, A	2	3	2	7
5	C, A, B	2	3	2	7
6	C, B, A	2	3	2	7
	Valor Médio:	2,33	2,67	1,33	

Fonte: Elaborada pelo autor.

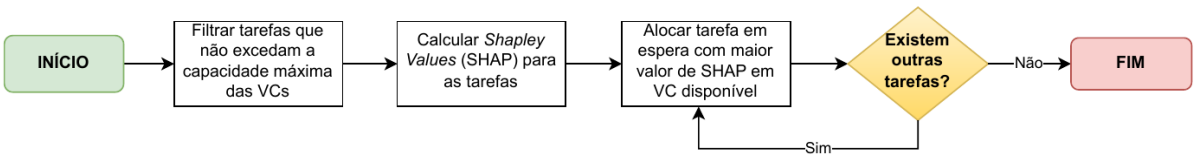
Algoritmo 5 – Escalonador de tarefas

```
1:  função ESCALONADORTAREFA(T, V)          ▷ T - conjunto de tarefas, V - nuvens
2:      U = 0
3:      Ta = ∅
4:      para t em T faça
5:          V_p = Filtrar(V, t.peso)
6:          V_e = EscolherVC(V_p)
7:          U = U + t.peso
8:          Ta = Ta ∪ {t}
9:          V_e.recurso_disponivel = V_e.recurso_disponivel − t.peso
10:     fim para
11:     retorna U
12: fim função
```

Por fim, determinada a ordem em que as tarefas devem ser alocadas, o escalonador definido pelo Algoritmo 5 inicializa o valor agregado total da coalizão (linha 2) e a lista de tarefas atendidas (linha 3). Para cada tarefa (linha 4), na ordem definida pelo Algoritmo 3, a solução recupera quais são as VCs que ainda possuem recursos disponíveis (linha 5), escolhe

aleatoriamente uma delas (linha 6) para realizar a alocação, incrementa o valor agregado total (linha 7), adiciona a tarefa à lista de tarefas alocadas (linha 8) e atualiza os recursos disponíveis da VC (linha 8). O algoritmo retorna o valor agregado total gerado pela alocação de tarefas (linha 11). A Figura 7 apresenta a solução implementada em forma de fluxograma.

Figura 7 – Fluxograma da solução implementada na fase 2

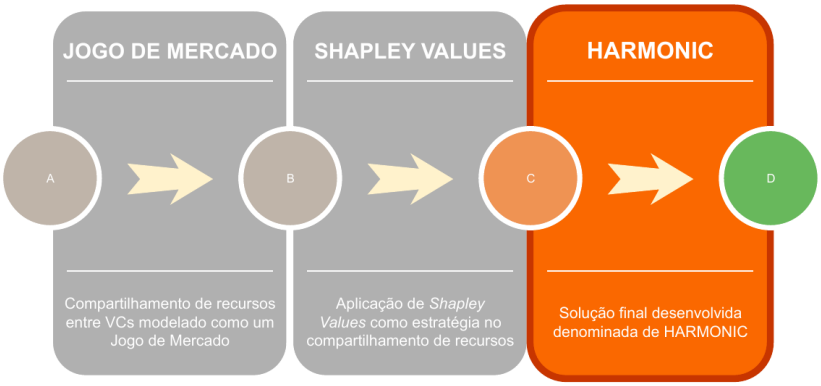


Fonte: Elaborada pelo autor.

4.3 HARMONIC - Solução Final

Nesta última fase (Figura 8), é apresentado o HARMONIC, a solução final desenvolvida e implementada neste trabalho. Na seção anterior introduziu-se o uso de *Shapley Values* para determinar a ordem em que as tarefas devem ser consideradas a fim de maximizar a quantidade de tarefas atendidas e, consequentemente, a utilização de recursos na rede. No entanto, não foi considerada a distribuição de carga entre as VCs existentes.

Figura 8 – Fase final



Fonte: Elaborada pelo autor.

Assim, o HARMONIC tem como objetivo utilizar os *Shapley Values* não apenas para determinar a ordem das tarefas, mas também para estabelecer a sequência de utilização das VCs a fim de otimizar a distribuição de tarefas entre um maior número de VCs, equilibrando a carga e, ao mesmo tempo, maximizando a alocação de recursos na rede veicular.

O Algoritmo 6 apresenta a solução final proposta e implementada. Inicializa-se o valor agregado total do mercado (linha 2) e descarta-se as tarefas que requeiram uma quantidade de

recursos maior que o máximo que uma única VC pode suportar (linha 3 e 4). As tarefas são consideradas indivisíveis e não podem ser parcialmente executadas por uma VC e parcialmente por outra.

Em seguida, a fim de determinar a ordem de alocação das tarefas que maximize a utilização de recursos na rede veicular, calculam-se os *Shapley Values* para cada uma das tarefas restantes (linha 5) e ordena-se o conjunto de tarefas em ordem decrescente desses valores (linha 6). Definida a ordem das tarefas, calcula-se o valor agregado que seria gerado caso as tarefas fossem alocadas (linha 8). Esse valor será utilizado como referência a seguir ao se buscar por novas soluções que melhor distribuam a carga de tarefas entre as VCs.

Algoritmo 6 – Algoritmo HARMONIC

```
1:  função HARMONIC(T,V)                          ▷ T - conjunto de tarefas, V - nuvens
2:      U = 0
3:      R_max = Max(V, recurso_disponivel)
4:      T = Filtrar(T, R_max)
5:      ShapT = CalcShapAprox(T, FuncaoColizao(T,V))          ▷ Algoritmo 4
6:      T = OrdenarDec(T, ShapT)
7:
8:      U_ref = FuncaoColizao(T,V)                            ▷ Valor de referência
9:
10:     ShapV = CalcShapAprox(V, FuncaoProducao(T,V,U_ref))   ▷ Algoritmo 7
11:     V = OrdenarDec(V, ShapV)
12:
13:     para t em T faça
14:         Ui = MERCADOSHAP(V,t)                             ▷ Algoritmo 8
15:         U = U + Ui
16:     fim para
17:     retorna U
18: fim função
```

Em seguida, o algoritmo procura por soluções que distribuam a carga de tarefas entre as VC, mas que resultem em uma utilização de recursos no mínimo igual ao valor de referência definido anteriormente. Para isso são calculados os *Shapley Values* para cada uma das VCs (linha 10) utilizando a função de produção definida no Algoritmo 7. E então ordenam-se as VCs em ordem decrescente de seus respectivos *Shapley Values* (linha 11). Respeitando a ordem das tarefas para alocação e a ordem em que as VCs devem ser consideradas, o algoritmo realiza a alocação por meio do Algoritmo 8 (linha 14) e atualiza o valor agregado gerado (linha 15). A solução retorna o valor agregado total obtido (linha 17).

Para calcular os *Shapley Values* correspondentes a cada uma das VCs (linha 10) utilizou-se, por ser um problema $\mathcal{NP}$-difícil, um método de aproximação baseado em amostras conforme discutido na Seção 2.4 do Capítulo 2. Para esse cálculo é necessário modelar a função de produção do mercado, ou seja, a função que quantifica o valor agregado gerado pelo mercado

formado pelas VCs em determinada ordem para a alocação de uma lista de tarefas. Essa função é definida pelo Algoritmo 7.

Algoritmo 7 – Função de produção do HARMONIC

```
1:  função FUNCAOPRODUCAO(T, V, U_ref)        ▷ T - conjunto de tarefas, V - nuvens, U_ref -
2:      U = 0
3:      VCU = ∅
4:      N_vc = 0                               ▷ N_vc - Número de VCs utilizadas
5:      para t em T faça
6:          para vc em V faça
7:              se t.peso ≤ vc.recurso_disponivel então
8:                  U = U + t.peso
9:                  vc.recurso_disponivel = vc.recurso_disponivel − t.peso
10:                 se vc ∉ VCU então
11:                     VCU = VCU ∪ {vc}
12:                 fim se
13:                 break
14:             fim se
15:         fim para
16:     fim para
17:     se U ≥ U_ref então
18:         N_vc = contar(VCU)
19:     senão
20:         N_vc = 0
21:     fim se
22:     retorna N_vc
23: fim função
```

O objetivo desse algoritmo é buscar por ordenações de VCs que gerem valor agregado igual ou superior ao valor de referência ao alocar o conjunto de tarefas. No entanto, diferentemente do algoritmo da seção anterior que retorna o valor agregado gerado, retorna-se aqui o número de VCs que foram utilizadas nessa alocação. Assim, no Algoritmo 6, ao calcular os *Shapley Values* (linha 10) para as VCs e ordená-las por esses valores em ordem decrescente (linha 11), tem-se como objetivo determinar a ordem das VCs que, ao alocar a lista de tarefas definida, resulte em uma alta utilização de recursos e uma maior distribuição da carga entre as diferentes VCs.

O Algoritmo 7 é definido como se segue. Inicializam-se o valor agregado gerado (linha 2), a lista de VCs utilizadas (linha 3) e a variável que conterá o número de VCs utilizadas (linha 4). Para cada tarefa (linha 5), e respeitando a ordem definida para as VCs (linha 6), verifica-se se a VC tem recursos suficientes para alocá-la (linha 7). Caso negativo, verifica-se a próxima VC. Sendo possível, atualizam-se o valor agregado (linha 8) e o total de recursos disponíveis da VC (linha 9) e, caso seja a primeira vez que a VC está sendo utilizada (linha 10), adiciona-a à lista de VCs utilizadas (linha 11). Uma vez alocada a tarefa, não há mais necessidade de percorrer as demais VCs da lista e a próxima tarefa pode ser considerada (linha 13).

Ao final verifica-se se o valor agregado é igual ou superior ao valor de referência (linha 17). Caso positivo, conta-se o número de VCs utilizadas (linha 18). Caso contrário o valor de VCs utilizadas é configurado para 0 (linha 20) com o objetivo de indicar que, no momento do cálculo de *Shapley Values*, essa ordem de VCs deve ser ignorada. O algoritmo retorna o valor configurado nesse último teste (linha 23).

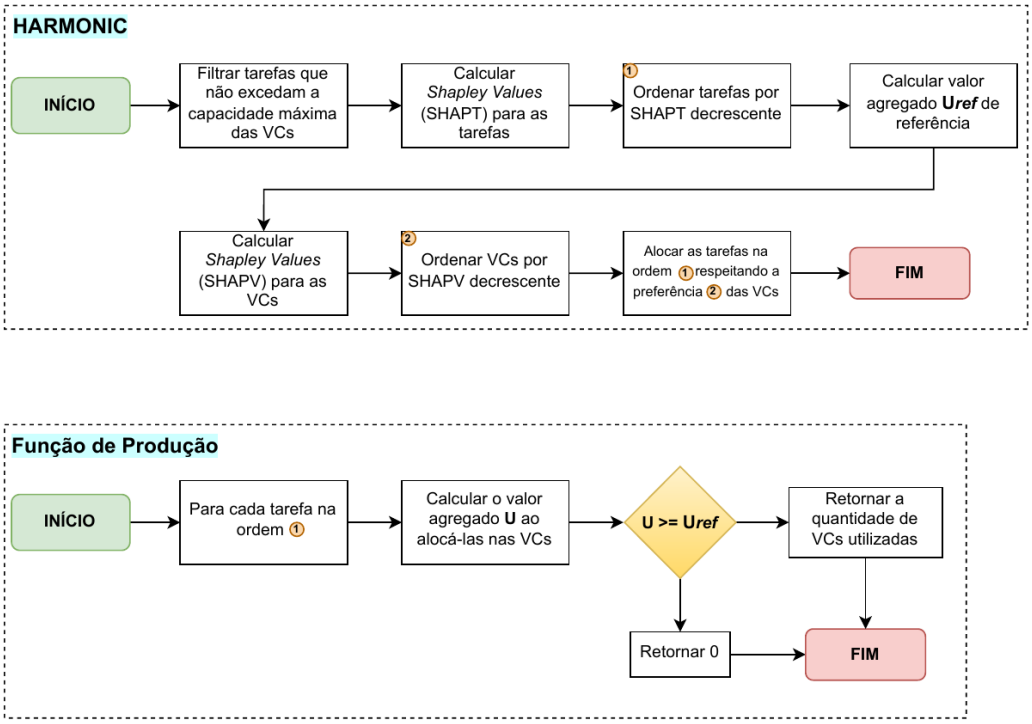
Após determinar a ordem de atendimento das tarefas, assim como a sequência em que as VCs devem ser utilizadas, o Algoritmo 8 realiza a alocação. Inicializam-se o valor agregado gerado (linha 2) e a lista de tarefas atendidas (linha 3).

Em seguida, tenta-se alocar a tarefa em alguma das VCs respeitando a ordem em que foram colocadas (linha 4). Caso a VC tenha recursos suficientes (linha 5), a tarefa é adicionada à lista de tarefas atendidas (linha 6), o valor agregado é atualizado (linha 7), assim como a quantidade de recursos disponíveis na VC (linha 8). Uma vez alocada a tarefa, o algoritmo encerra a busca (linha 9) e retorna o valor agregado gerado e a lista de tarefas alocadas (linha 12). A Figura 9 apresenta a solução implementada, juntamente com sua função de produção, em forma de fluxograma.

Algoritmo 8 – Execução do Mercado do HARMONIC com o uso de *Shapley Values*

```
1:  função MERCADOSHAP(V,t)                    ▷ V - nuvens, t - tarefa para alocação
2:      U = 0
3:      Ta = ∅
4:      para vc em V faça
5:          se t.peso ≤ vc.recurso_disponivel então
6:              Ta = Ta ∪ {t}
7:              U = U + t.peso
8:              vc.recurso_disponivel = vc.recurso_disponivel − t.peso
9:              break
10:         fim se
11:     fim para
12:     retorna U, Ta
13: fim função
```

Figura 9 – Fluxogramas do HARMONIC e da função de produção implementada



Fonte: Elaborada pelo autor.

CAPÍTULO

5

AVALIAÇÃO METODOLÓGICA

Neste capítulo, é apresentada a metodologia de avaliação dos modelos, juntamente com a análise dos resultados obtidos. O cenário é examinado sob diferentes configurações de parâmetros de simulação, e o desempenho do HARMONIC é avaliado em comparação com outros 4 algoritmos previamente descritos na literatura.

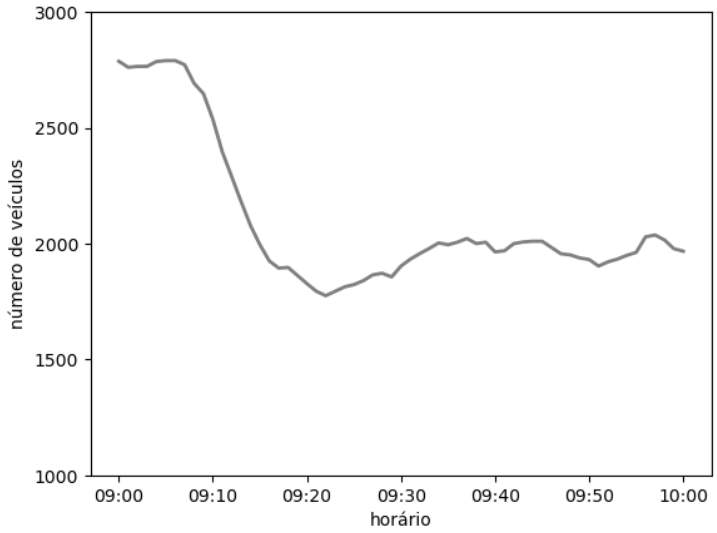
5.1 Cenário

Os experimentos utilizaram o *Simulation of Urban Mobility* (SUMO) (LOPEZ *et al.*, 2018), na versão 1.12.0, em conjunto com o cenário TAPAS Cologne (UPPOOR; FIORE, 2011). Esse *trace* de mobilidade descreve o tráfego na cidade de Colônia, na Alemanha, ao longo de um dia inteiro. Trata-se de um dos conjuntos de dados de simulação de tráfego gratuitos mais extensos disponíveis. Os algoritmos foram implementados em Python, fazendo uso da biblioteca *Traffic Control Interface* (TraCI) para conectar-se ao SUMO.

Avaliou-se o HARMONIC entre as 9h e as 10h, imediatamente após o horário de pico, devido à menor densidade de veículos no mapa, o que cria ambientes mais desafiadores com menos recursos. No entanto, mesmo durante esse horário, o tráfego no mapa requer alto processamento computacional para realizar as simulações. Por essa razão, o tráfego foi reduzido em um fator de $0,3$ e o SUMO foi utilizado no modo mesoscópico. A Figura 10 apresenta a quantidade de veículos no período.

O modo mesoscópico refere-se a um nível intermediário de resolução de mobilidade entre os modos microscópico e macroscópico de mobilidade urbana. No modo microscópico, os movimentos dos veículos nas ruas são simulados, enquanto no modo macroscópico, o fluxo de veículos é simulado. No modo mesoscópico, os veículos são tratados como entidades discretas, e suas interações com outros veículos e a infraestrutura viária são modeladas com base em regras simples, que se baseiam no fluxo de tráfego e no comportamento do condutor. Nesse

Figura 10 – Número de veículos entre 9h e 10h da manhã



caso, as simulações podem ser executadas em uma escala maior do que no modo microscópico, capturando ao mesmo tempo as características essenciais do tráfego urbano.

Além disso, o modelo de direção inteligente do SUMO foi empregado, sendo um modelo de "seguimento de veículos"que descreve o comportamento dos veículos ao seguir outros veículos na estrada. Esse modelo leva em consideração velocidade, distância e aceleração dos veículos para determinar a aceleração e desaceleração apropriadas a fim de seguir o veículo à frente com segurança e eficiência.

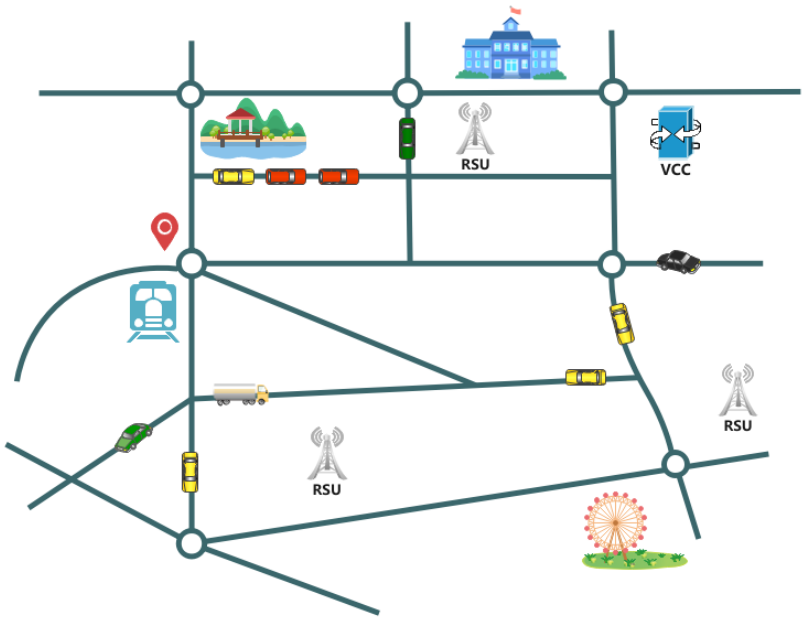
Em uma rede veicular, o agrupamento é um processo de reunir veículos em nuvens veiculares para compartilhar seus recursos e atender a mais solicitações na rede. O intervalo de agrupamento determina de quanto em quanto tempo essas nuvens serão reorganizadas. Dessa forma, para realizar o processo de agrupamento, é necessário usar um algoritmo de agrupamento, que pode se basear em diferentes técnicas, como distância ou densidade.

O raio de comunicação é essencial nesse processo, pois define a distância máxima entre veículos para que sejam considerados parte da mesma nuvem veicular. Além disso, é necessário definir a distribuição de tarefas na rede veicular, ou seja, com que frequência as solicitações de recursos chegam ao sistema. A taxa de serviço também é um fator relevante, pois determina o número de solicitações de tarefas que chegam à rede veicular em um período específico. Cada tarefa possui um peso que representa a quantidade total de recursos necessários para sua alocação, e é considerado ao definir quais tarefas serão alocadas em quais nuvens veiculares. O tempo de simulação é o tempo total definido pelo cenário TAPAS Cologne.

O cenário da rede veicular considerado durante as simulações inclui os veículos, as unidades de acostamento (RSU) e o controlador de nuvem veicular (VCC) (Figura 11). As RSUs estão conectadas entre si (Figura 12a) e se comunicam com o VCC que está instalado em uma delas. Os veículos se deslocam pelo mapa rodoviário seguindo o *trace* de mobilidade e podem se

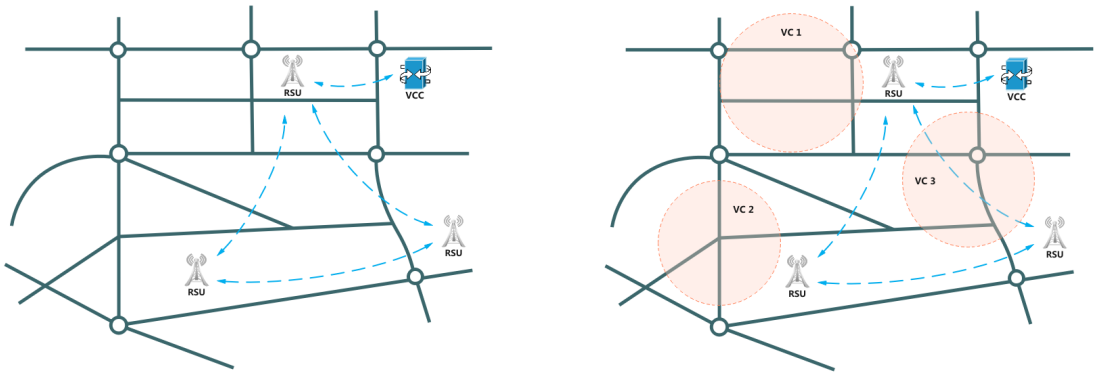
comunicar com as RSUs (Figura 13a). As RSUs atuam como intermediárias entre os veículos e o VCC.

Figura 11 – Cenário



Fonte: Elaborada pelo autor.

Figura 12 – Interconexões entre RSUs e VCC



(a) Antes da formação das VCs

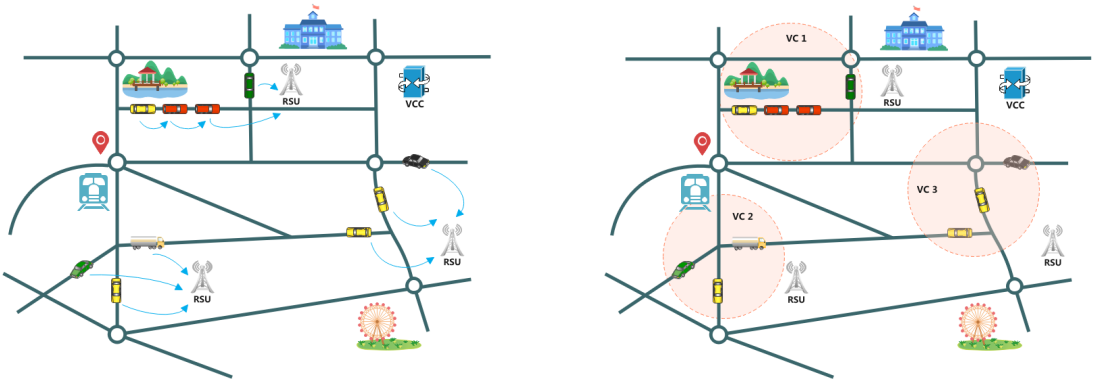
(b) Depois da formação das VCs

Fonte: Elaborada pelo autor.

Cada veículo está equipado com uma unidade de bordo (OBU, *On Board Unit*) e ao entrarem na área de cobertura da rede, informam à RSU mais próxima (Figura 13a) a quantidade de recursos disponíveis para compartilhamento e suas solicitações de serviços. Essas informações, juntamente com as coordenadas de localização dos veículos, são repassadas ao VCC que é o responsável por agrupar os veículos em nuvens veiculares (Figuras 12b e 13b). De posse das

informações de VCs, e da oferta e da demanda por recursos da rede, o VCC é responsável por executar o algoritmo de alocação de recursos e garantir que os recursos sejam alocados adequadamente.

Figura 13 – Formação das nuvens veiculares



(a) Veículos informam recursos às RSUs (b) Depois da formação das VCs

Fonte: Elaborada pelo autor.

Neste estudo, assume-se que o VCC aloca recursos para as tarefas com todas as informações necessárias para a alocação, estabelecendo um prazo (tempo de execução do sistema) igual a 1. Devido à alta mobilidade no cenário, o VCC executa o algoritmo DBSCAN (ESTER *et al.*, 1996) a cada 60 segundos para criar as VCs, nas quais cada veículo pode compartilhar até 5 unidades de recursos. Diferentes raios de comunicação (em metros) são considerados, variando de $\{10, 25, 50, 75, 100, 150, 200\}$. A taxa de chegada de tarefas segue uma distribuição de Poisson, com diferentes taxas médias de solicitação em $\{20, 50, 100, 150, 200, 250\}$, e o peso das tarefas segue distribuições uniformes variando de $[1, \mu]$, com $\mu = \{25, 50, 75, 100, 150, 200, 250\}$.

O HARMONIC foi comparado a 4 outros algoritmos em um total de 1470 configurações diferentes de parâmetros, com cada configuração sendo executada 33 vezes. Essa análise abrangente considerou todas as combinações dos diferentes parâmetros listados no Quadro 2, permitindo capturar uma gama de variações na configuração experimental.

Quadro 2 – Parâmetros de simulação

Parâmetro	**Valor**
Raio de comunicação	10, 25, 50, 75, 100, 150, 200 metros
Taxa de serviços (λ)	20, 50, 100, 150, 200, 250
Peso das tarefas	25, 50, 75, 100, 150, 200, 250
Recursos por veículo	1, 2, 3, 4, 5

Fonte: Elaborada pelo autor.

Para avaliar as diferenças de desempenho entre os algoritmos, foram utilizados o teste de Friedman (FRIEDMAN, 1937) em conjunto com o teste de Nemenyi (NEMENYI, 1963; DUNN, 1964; DEMŠAR, 2006). O teste de Friedman permitiu verificar se houve variações significativas no desempenho dos algoritmos nas diversas configurações. Ao identificar diferenças significativas no teste de Friedman, o teste de Nemenyi foi conduzido para realizar comparações em pares entre os algoritmos. Isso permitiu determinar quais algoritmos apresentaram disparidades estatisticamente significativas em seu desempenho. Para ambos os testes, de Friedman e de Nemenyi, foi escolhido um nível de significância de $p < 0,05$, indicando que diferenças com valores de p abaixo desse limite são consideradas estatisticamente significativas.

5.2 Métricas

Para avaliar a solução, foram consideradas as notações do Quadro 3, onde se denota UVC como o conjunto de VCs efetivamente utilizadas para alocar ao menos 1 tarefa e AT como a porcentagem de recursos utilizados para atender às tarefas alocadas. Essas métricas são definidas pela Equação 5.1 e pela Equação 5.2 respectivamente. O valor de k é definido da seguinte maneira: $k = 1$ indica que a VC_i foi utilizada e $k = 0$ indica que a VC_i não foi utilizada.

$$UVC = \bigcup_{i=0}^{|VC|} VC_i \times k \tag{5.1}$$

$$AT(\%) = \frac{\sum_{i=0}^{|T^A|} peso(T_i^A)}{\sum_{j=0}^{|T|} peso(T_j)} \cdot 100 \tag{5.2}$$

Quadro 3 – Notação utilizada na avaliação da solução

Notação	Definição
VC_i	VC referenciada por i
VC	Conjunto de todas as VCs
UVC	Conjunto de VCs com pelo menos 1 tarefa alocada
T	Conjunto de todas as tarefas
T^A	Conjunto de tarefas alocadas
$peso(T)$	Quantidade de recursos necessários para executar a tarefa T
$recursos(VC_i)$	Quantidade de recursos disponíveis na VC i

Fonte: Elaborada pelo autor.

Além disso, para demonstrar a eficácia do uso de *Shapley Values* na alocação de recursos, foi contabilizado o número de ciclos (Nc) necessários para alocar AT e o número de falhas de alocação (Nf) que ocorreram durante esse processo. Um ciclo é considerado toda vez que

os algoritmos tentam alocar uma tarefa i (T_i) em uma nuvem veicular j (VC_j). Uma falha de alocação é quando essa tentativa não é bem-sucedida. Em seguida, foi avaliada a taxa de alocação pelo número de ciclos (TAc), conforme definido na Equação 5.3.

$$TAc = \frac{AT}{Nc} \tag{5.3}$$

Quanto maior o valor de TAc, mais eficiente é a alocação de recursos, pois indica uma maior porcentagem de recursos alocados por ciclo. A redução no número de ciclos necessários para alocação pode levar a economia de energia e menor latência na rede. Também foi considerada a taxa de alocação pelo número de falhas de alocação (TAf), definida pela Equação 5.4.

$$TAf = \frac{AT}{Nf} \tag{5.4}$$

Um valor mais alto de TAf indica uma alocação de recursos mais eficiente, com menos falhas de alocação. Isso reflete na eficácia do algoritmo de alocação em atribuir recursos com sucesso às tarefas, resultando em melhor desempenho do sistema e reduzindo o desperdício de recursos.

Por fim, NTL é definido como a razão entre os recursos totais necessários para atender a todas os solicitações de serviços e entre a quantidade de recursos disponíveis nas VCs. Essa medida é expressa em porcentagens e é definida pela Equação 5.5.

$$NTL = \frac{\sum_{i=0}^{|T|} peso(T_i)}{\sum_{j=0}^{|VC|} recursos(VC_j)} \cdot 100 \tag{5.5}$$

O HARMONIC foi comparado a 4 outros mecanismos, considerando a taxa de alocação de tarefas, o número de rodadas, o número de falhas de alocação e a distribuição de carga entre as VCs. Os mecanismos são os seguintes: (i) MORFEU (COSTA *et al.*, 2020), que utiliza uma abordagem de otimização combinatória para alocar tarefas; (ii) GREEDY e (iii) GREEDY-N, sendo este último baseado em Nabi *et al.* (NABI *et al.*, 2017). Ambos utilizam uma abordagem gananciosa para a alocação de tarefas. No entanto, o GREEDY-N considera mais de uma VC no processo de alocação de recursos; e (iv) DP, que emprega uma abordagem de programação dinâmica.

Os algoritmos MORFEU e GREEDY-N consideram todas as VCs existentes ao definir o melhor esquema de alocação de tarefas, enquanto o GREEDY e o DP consideram apenas a VC com maior quantidade de recursos disponíveis. Todos esses algoritmos inicialmente utilizam a VC com mais recursos para iniciar a alocação, enquanto que o HARMONIC utiliza o cálculo dos *Shapley Values* para determinar qual a ordem em que as VCs serão utilizadas.

Para todas as análises realizadas, quando um parâmetro de simulação foi avaliado, todos os valores diferentes dos outros parâmetros foram considerados. Por exemplo, quando os

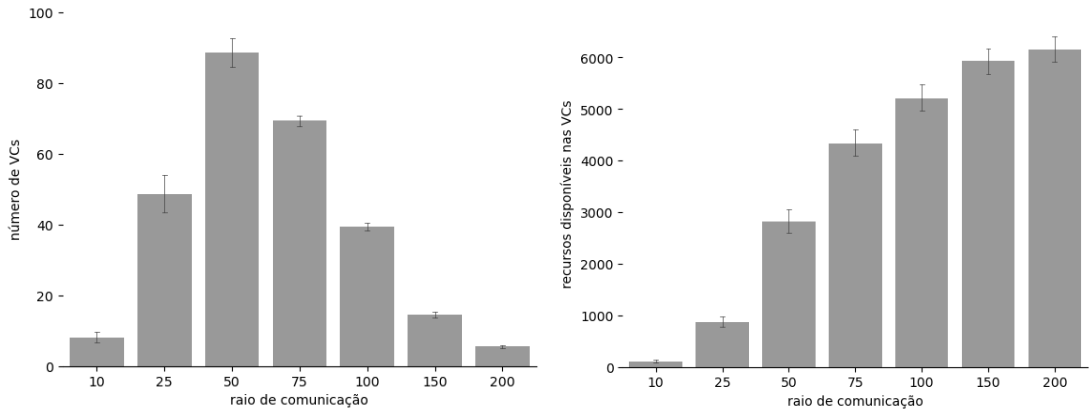
resultados mostram diferentes valores de raio de comunicação, para cada raio de comunicação, foi calculada a média de todos os valores de taxa de serviço, pesos da tarefas e quantidade de recursos por veículo. Isso permitiu avaliar o impacto do raio de comunicação em todos os parâmetros e valores, em vez de se concentrar em parâmetros específicos que poderiam não representar o cenário completo.

5.3 Resultados

Nesta seção, o desempenho da estratégia de alocação de recursos do HARMONIC é avaliado e comparado a 4 outros algoritmos descritos na literatura.

A Figura 14 mostra o número de VCs criadas e os recursos disponíveis nas VCs em cenários com diferentes raios de comunicação. O algoritmo de agrupamento define as VCs e os veículos que participam delas. Portanto, todos os algoritmos de alocação de tarefas compartilham as VCs resultantes e os recursos disponibilizados por elas.

Figura 14 – Número de VCs criadas e recursos disponíveis



(a) Número de VCs criadas

(b) Recursos disponíveis nas VCs

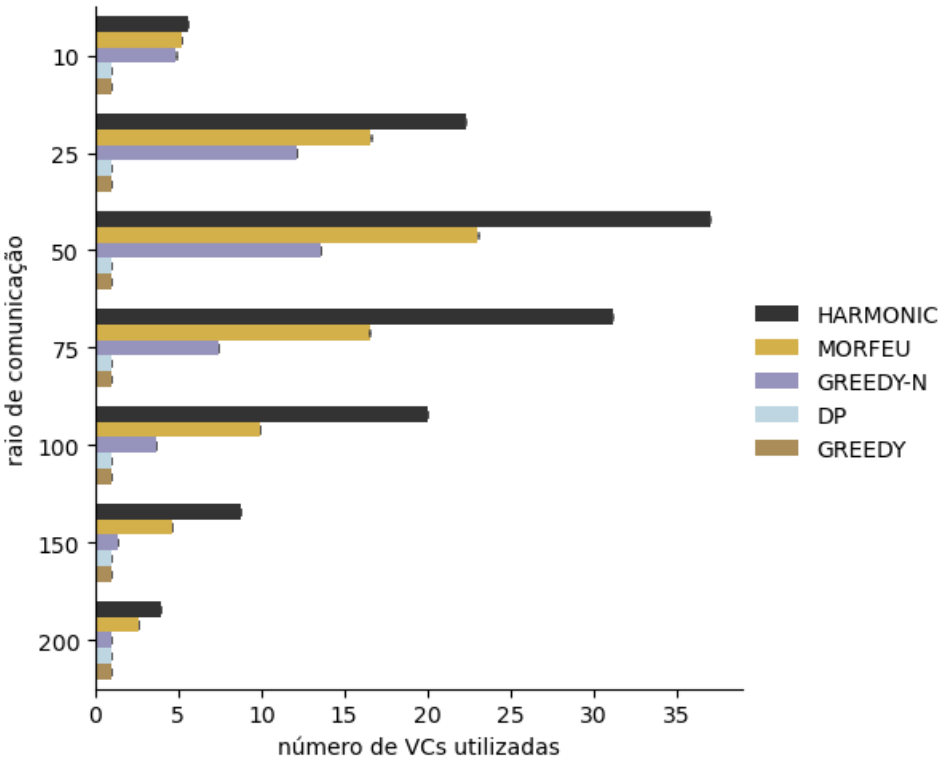
Na Figura 14a, é importante notar que, com o aumento do raio de comunicação, o número de VCs criadas na rede também aumenta, atingindo o pico quando o raio de comunicação é igual a 50 e depois começa a diminuir. O número de VCs formadas aumenta à medida que o raio de comunicação aumenta porque mais carros estão próximos uns dos outros dentro da faixa dada. Quando os raios de comunicação variam de 75 a 200, o número de VCs formadas diminui, pois a cobertura é grande o suficiente para abranger mais carros dentro da mesma VC.

A Figura 14b mostra o aumento da quantidade de recursos disponíveis nas VCs para diferentes valores de raio de comunicação. Conforme explicado na Figura 14a, quanto maior o raio de comunicação, mais carros são agrupados. Consequentemente, mais recursos são agrupados nas VCs, e ficam disponíveis para compartilhamento.

A Figura 15 mostra o número de VCs envolvidas na alocação de recursos ($|UVC|$) em configurações com diferentes raios de comunicação. Em todos as configurações, o HARMONIC

é o mecanismo que envolve o número maior de VCs na alocação de tarefas. Isso ocorre devido ao uso de *Shapley Values*, que distribuem as tarefas de forma justa avaliando seus pesos entre os membros da coalizão. Nesse caso, os axiomas, revisados na Seção 2.4 do Capítulo 2, definem o conceito de equidade.

Figura 15 – Número de VCs utilizadas em diferentes raios de comunicação

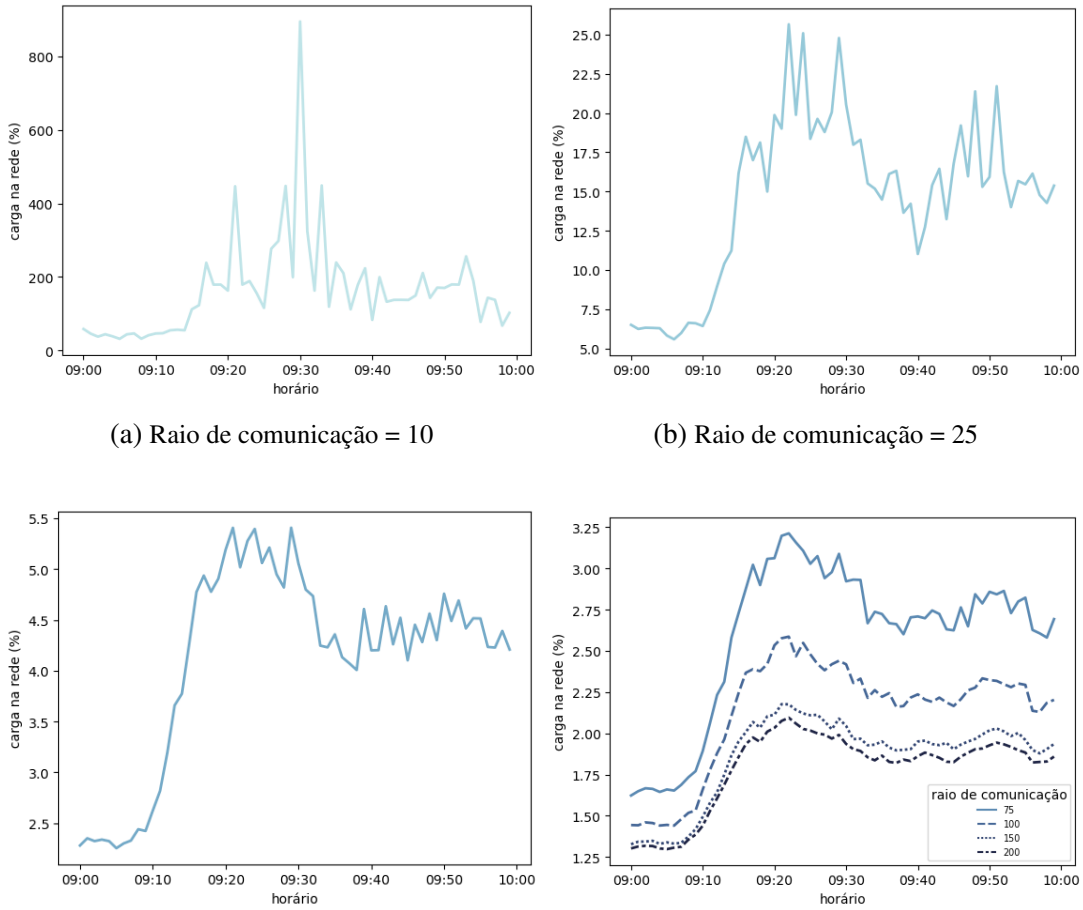


A Figura 16 mostra a carga de tarefas na rede (NTL), conforme definido na Equação 5.5, em cenários com diferentes raios de comunicação. Como esperado, com o aumento do raio de comunicação, o NTL diminui uma vez que há mais recursos disponíveis na rede para atender aos requisitos das solicitações de serviços. A demanda por recursos é mais de 700 vezes maior ($NTL = 800\%$) que a oferta de recursos disponíveis na rede (Figura 16a às 09:30). O valor de NTL continua a diminuir à medida que a cobertura aumenta ao longo do tempo (Figuras 16b, 16c e 16d), mas não se aproxima de $NTL = 1$, o que indicaria que a demanda é igual ao fornecimento de recursos. Essas figuras sugerem que os cenários têm restrições severas de recursos e são insuficientes para atender a todas as solicitações de serviços.

Após a realização dos testes de Friedman e pós-teste de Nemenyi com um nível de significância de $p < 0,05$, analisou-se os algoritmos para avaliar a taxa de alocação (AT), a taxa de alocação por ciclo (TAc) e a taxa de alocação por falha (TAf). Entre os 1470 cenários avaliados, 443 cenários apresentaram diferenças significativas de desempenho de um algoritmo sobre os demais.

A Tabela 4 apresenta o número de cenários significativos para cada métrica, com 361 cenários ($81,48\%$) indicando a superioridade do algoritmo HARMONIC. Ao considerar apenas

Figura 16 – Carga de tarefas na rede em diferentes raios de comunicação



(a) Raio de comunicação = 10

(b) Raio de comunicação = 25

(c) Raio de comunicação = 50

(d) Raio de comunicação = [75,100,150,200]

taxa de alocação (Figura 17), não há predominância significativa de um algoritmo sobre os outros. No entanto, ao examinar as taxas de alocação por ciclos necessários (Figura 18) e pelas falhas ocorridas (Figura 19) durante o processo de alocação, as diferenças entre o algoritmo HARMONIC e os demais se tornam evidentes. Explorou-se os dados a partir de 3 perspectivas para cada métrica: o aumento da taxa de serviço, do raio de comunicação e dos pesos das tarefas.

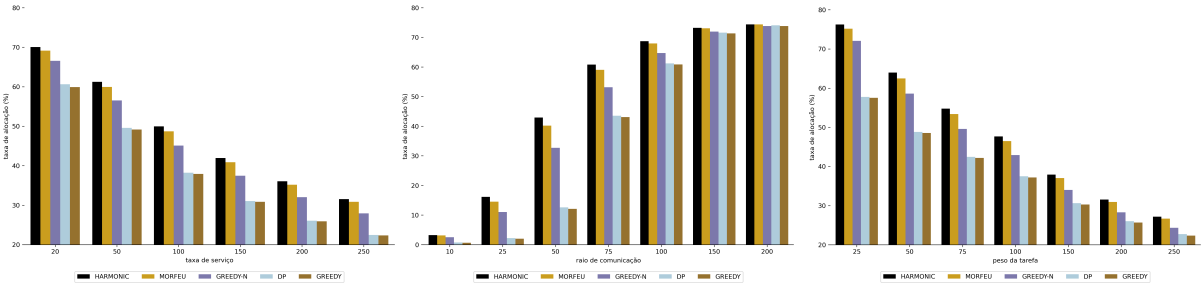
Tabela 4 – Número de configurações com predominância significativa de um algoritmo sobre os demais nas métricas avaliadas

Metric	HARMONIC	MORFEU	GREEDY-N	DP	GREEDY
AT	0	0	0	0	0
TAc	261	0	0	48	6
TAf	100	24	4	0	0

Na Figura 17, a taxa de alocação (AT) foi avaliada para os 443 cenários listados na Tabela 4. Na Figura 17a, observa-se a evolução da métrica à medida que a taxa de serviço aumenta. Conforme a taxa de serviço aumenta, a porcentagem de alocação diminui. Essa redução pode ser

atribuída ao aumento do volume de tarefas no sistema e à disponibilidade limitada de recursos veiculares. Essa combinação cria cenários mais desafiadores e restritivos no processo de alocação de recursos.

Figura 17 – Taxas de alocação (%)



(a) Taxa de serviço crescente. (b) Raio de comunicação crescente. (c) Pesos crescentes das tarefas.

Em seguida, na Figura 17b, foi avaliado o impacto do aumento do raio de comunicação na taxa de alocação de tarefas. Os resultados mostram que a taxa de alocação aumenta com a expansão do raio de comunicação. Esse comportamento ocorre porque, como explicado na Figura 14b, o aumento do raio permite uma maior cobertura veicular e, consequentemente, uma maior disponibilidade de recursos veiculares, resultando em uma taxa de sucesso mais alta na realização de tarefas.

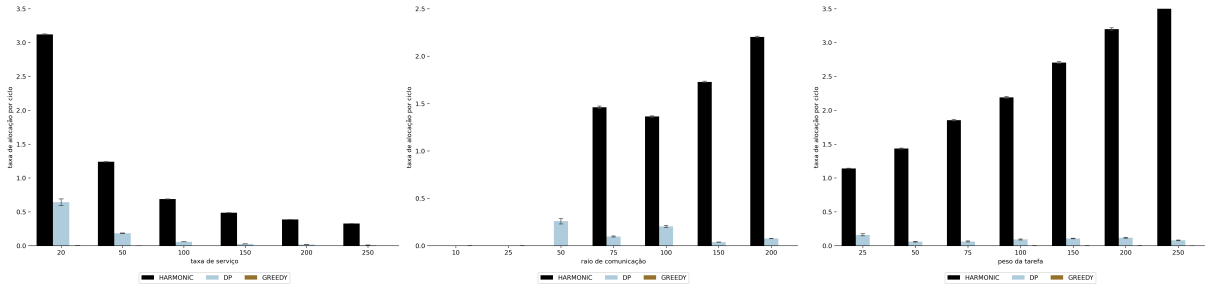
Por fim, na Figura 17c, foi analisada a taxa de alocação em relação ao aumento dos pesos das tarefas. Observa-se que, à medida que o peso aumenta, a porcentagem de alocação diminui. Essa diminuição ocorre porque as tarefas são consideradas unidades atômicas que são alocadas ou não, e alocar recursos suficientes para a execução dessas tarefas torna-se mais desafiador. Não foram encontradas predominâncias significativas entre os algoritmos em nenhuma das configurações (Tabela 4, linha 1).

Na Figura 18, apresenta-se a taxa de alocação por ciclo (TAc) para as 315 configurações da Tabela 4 (linha 2). Em 261 delas ($82,86\%$), o HARMONIC apresenta resultados superiores aos demais algoritmos comparados. Não foram encontradas configurações com predominância dos algoritmos MORFEU e GREEDY-N, que, portanto, não estão representados nos gráficos.

Na Figura 18a, é apresentada a taxa de alocação por ciclo (TAr) em relação à taxa de serviço. Conforme listado na Tabela 5, quanto à taxa de serviço, a taxa de alocação por ciclo do HARMONIC é $234,04\%$ e $224185,71\%$ maior que as obtidas pelo DP e GREEDY respectivamente. O valor elevado em relação ao GREEDY ocorre devido ao baixo número de ocorrências (Tabela 4, linha 2) com valores que somam apenas $0,0014\%$.

A predominância do HARMONIC é atribuída a sua abordagem de determinar a ordem em que as VCs devem ser utilizadas antes de iniciar a alocação de tarefas, ao contrário de outros algoritmos que priorizam a VC com mais recursos. Essa abordagem permite maior flexibilidade

Figura 18 – Taxas de alocação por ciclo (%)



(a) Taxa de serviço crescente. (b) Raio de comunicação crescente. (c) Pesos crescentes das tarefas.

Tabela 5 – Predominância do HARMONIC quanto a taxa de alocação por ciclo (%)

HARMONIC %	DP	GREEDY
Taxa de serviço	234,04%	224185,71%
Raio de comunicação	892,65%	421775,00%
Pesos das tarefas	2263,24%	308938,46%

em cenários com diferentes taxas de serviços. Além disso, o uso de *Shapley Values* permite avaliar como as tarefas devem ser alocadas entre as VCs que compartilham recursos na rede.

Na Figura 18b, é apresentada a taxa de alocação por ciclo (TAr) em relação ao raio de comunicação. Conforme listado na Tabela 5, quanto ao raio de comunicação, a taxa de alocação por ciclo do HARMONIC é $892,65\%$ e $421775,00\%$ maior que as obtidas pelo DP e GREEDY respectivamente. É a partir do valor igual a 75 que o HARMONIC começa a se destacar.

A diferenciação entre os algoritmos ocorre devido ao grande número de VCs formadas e à disponibilidade de recursos nas redes, conforme representado na Figura 14. Vale ressaltar que, mesmo com um maior número de VCs formadas quando o raio de comunicação é igual a 50 (Figura 14a), ainda não existem recursos suficientes disponíveis (Figura 14b). Com um maior número de VCs formadas e recursos disponíveis na rede, os algoritmos podem se diferenciar por meio de suas estratégias de alocação.

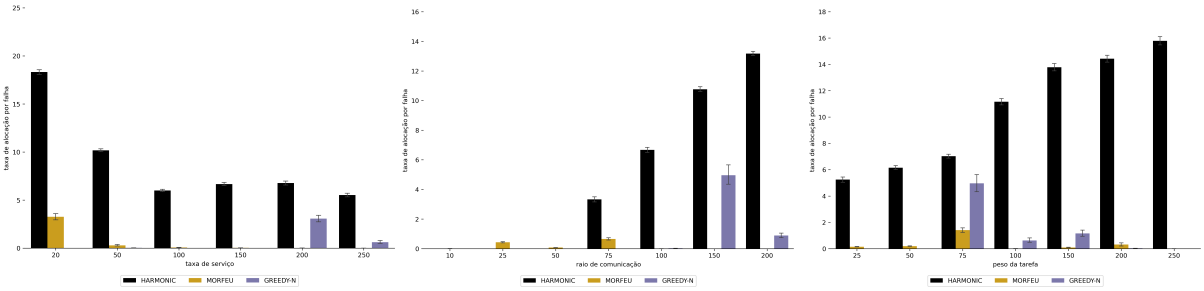
Na Figura 18c, é apresentada a taxa de alocação por ciclo (TAr) em relação aos pesos das tarefas. O HARMONIC supera as outras soluções em todos os valores de pesos de tarefas considerados durante as simulações. Observa-se que, à medida que os pesos das tarefas aumentam, a taxa de alocação por ciclo também aumenta. Esse aumento ocorre porque cada tarefa, quando alocada com sucesso, influencia diretamente a métrica de forma proporcional. Portanto, quanto mais tarefas um algoritmo pode alocar, melhor é o seu desempenho.

A capacidade do HARMONIC de alocar eficientemente tarefas, independentemente do peso, destaca sua eficácia na alocação de recursos da rede veicular. Os resultados destacam a robustez e eficácia do algoritmo HARMONIC em diversas configurações, demonstrando seu desempenho superior na alocação distribuída de tarefas entre as VCs, exigindo para isso um

número reduzido de ciclos. Quando comparados aos demais algoritmos, conforme listado na Tabela 5, quanto aos pesos das tarefas, a taxa de alocação por ciclo do HARMONIC é $2263,24\%$ e $308938,46\%$ maior que as obtidas pelo DP e GREEDY respectivamente.

Na Figura 19, apresenta-se a taxa de alocação por falha (TAf) para as 128 configurações da Tabela 4 (linha 3). Em 100 delas ($78,12\%$), o HARMONIC apresenta resultados superiores aos demais algoritmos comparados. Não foram encontradas configurações com predominância dos algoritmos DP e GREEDY, que, portanto, não estão representados nos gráficos.

Figura 19 – Taxas de alocação por falha (%)



(a) Taxa de serviço crescente. (b) Raio de comunicação crescente. (c) Pesos crescentes das tarefas.

Na Figura 19a, é apresentada a taxa de alocação por falha (TAf) em relação à taxa de serviço. Conforme listado na Tabela 6, quanto à taxa de serviço, a taxa de alocação por falha do HARMONIC é $1365,97\%$ e $1339,89\%$ maior que as obtidas pelo MORFEU e GREEDY-N respectivamente. Nota-se que quando a taxa de serviço é igual a 20, o HARMONIC alcança sua maior taxa de alocação por falha. Com uma quantidade menor ser solicitações na rede, o algoritmo tem menos restrições de recursos e pode expandir o espaço conceitual da solução.

Tabela 6 – Predominância do HARMONIC quanto a taxa de alocação por falha (%)

HARMONIC %	**MORFEU**	**GREEDY-N**
Taxa de serviço	1365,97%	1339,89%
Raio de comunicação	2773,84%	477,21%
Pesos das tarefas	3274,14%	987,00%

Na Figura 19b, é apresentada a taxa de alocação por falha (TAf) em relação ao raio de comunicação. Conforme listado na Tabela 6, quanto ao raio de comunicação, a taxa de alocação por ciclo do HARMONIC é $2773,84\%$ e $477,21\%$ maior que as obtidas pelo MORFEU e GREEDY-N respectivamente. Como explicado na Figura 19a, essa superioridade do HARMONIC ocorre quando há uma maior relação entre os recursos solicitados e os recursos fornecidos pelos veículos, tornando o modelo menos restritivo e mais propenso a encontrar soluções precisas. No caso da Figura 19b, o aumento do raio de comunicação torna o ambiente mais favorável para a formação de VCs, devido à maior cobertura de veículos na rede, aumentando assim a quantidade de recursos compartilhados.

Por fim, na Figura 19c, é apresentada a taxa de alocação por falha (TAr) em relação aos pesos das tarefas. Conforme listado na Tabela 6, quanto aos pesos das tarefas, a taxa de alocação por ciclo do HARMONIC é $3274,14\%$ e $987,00\%$ maior que as obtidas pelo MORFEU e GREEDY-N respectivamente.

A análise demonstra que o algoritmo HARMONIC aloca eficientemente os recursos veiculares a fim de atender às solicitações de serviços na rede em um número menor de ciclos e de falhas durante o processo de alocação. Ao identificar as ordens de tarefas e VCs a serem utilizadas, minimiza a necessidade de iterações adicionais e melhora a eficiência geral.

Além disso, a capacidade do algoritmo de reduzir o número de falhas durante a alocação, contribui para a estabilidade e confiabilidade do processo, eliminando a necessidade de reatribuição de tarefas e minimizando interrupções. Portanto, o HARMONIC se destaca como uma opção mais eficiente e confiável para a alocação de recursos, alocando a mesma quantidade de recursos que outros métodos, mas com menos rodadas e falhas durante o processo.

CAPÍTULO

6

CONCLUSÃO

Com o crescimento do número de veículos conectados e o aumento da complexidade das aplicações desenvolvidas para ambientes de redes veiculares, enfrenta-se um desafio substancial na alocação eficiente de recursos computacionais para atender às mais diversas solicitações de serviços.

A fim de abordar esse desafio, neste trabalho de Mestrado apresentou-se e desenvolveu-se uma solução heurística denominada HARMONIC. Essa solução, modelada com o uso de Teoria dos Jogos, utiliza-se da formação de uma coalizão de nuvens veiculares (VC) em um contexto de Jogo de Mercado e incorpora o conceito de *Shapley Values* com o propósito de otimizar a alocação de recursos na rede veicular e aumentar a distribuição as tarefas entre as VCs disponíveis.

Conforme apresentado no Capítulo 5, a solução foi comparada a 4 estratégias de alocação de recursos amplamente utilizadas em ambientes de VC, sob diferentes condições, como o número de recursos disponíveis em cada veículo, os pesos da tarefa, a carga de tarefas na rede e o raio de comunicação. As estratégias associadas a essas abordagens incluem Teoria dos Jogos, Otimização Combinatória, Algoritmos Gananciosos e Programação Dinâmica.

Os resultados obtidos indicaram que o HARMONIC, uma estratégia baseada em Teoria dos Jogos, é a que apresenta melhor desempenho diante do aumento dos pesos das tarefas, demonstrando maior capacidade na distribuição da carga de trabalho entre as VCs em menos ciclos e com menor número de falhas durante o processo de alocação. Os resultados também mostram que:

- Com o aumento da taxa de serviço e, consequentemente, o aumento no número de tarefas para as quais mais recursos precisam ser alocados, o HARMONIC se destaca como a estratégia mais eficaz, capaz de gerar configurações otimizadas para a distribuição de tarefas entre os recursos disponibilizados pelas VCs. Em outras palavras, dada a

quantidade de VC e seus recursos disponíveis, o HARMONIC demonstra consistentemente a capacidade de utilizar recursos de um maior número de VCs, distribuindo assim a carga de tarefas entre elas. Por outro lado, as demais estratégias tendem a concentrar recursos em um subconjunto de VCs, aumentando assim o risco de atrasos na execução de tarefas, saturação da capacidade individual da VC gerando gargalos e desperdício de recursos;

- À medida que o número de recursos solicitados pelas tarefas aumenta, e diante da restrição de recursos disponíveis, o HARMONIC demonstra sua capacidade superior em atender às demandas de um grande número de tarefas com menos iterações e ocorrência de falhas durante o processo de alocação. Isso resulta em uma alocação mais equilibrada e reduz a necessidade de recursos adicionais.

- Em cenários onde as restrições sobre a quantidade de recursos disponíveis são removidas, resultando em uma abundância, o HARMONIC demonstra ser o mais eficaz ao utilizar os recursos das VCs quando comparado às outras soluções da literatura, que frequentemente concentram a carga de trabalho em um subconjunto dos recursos das VCs.

Além disso, conforme descrito na Seção 4.3 do Capítulo 4, a alocação de tarefas do HARMONIC é mais simples, intuitiva e direta, uma vez que segue a ordem decrescente do valor de Shapley atribuído a cada VC. Em outras palavras, a solução aloca inicialmente as tarefas para as VCs com *Shapley values* mais elevados.

Por fim, os planos para trabalhos futuros incluem:

- Explorar cenários nos quais a formação de estruturas de coalizão menores seja mais vantajosa do que a criação de uma única coalizão com todas as VCs;

- Considerar outras características da rede, tais como atraso e *throughput*, no cálculo dos *Shapley values*, especialmente em cenários mais complexos;

- Avaliar abordagens de arquitetura híbrida, buscando aproveitar os benefícios das arquiteturas centralizadas e descentralizadas;

- E desenvolver mecanismos de incentivo que promovam a criação e manutenção de coalizões, com o objetivo de aprimorar a eficiência na alocação de recursos em redes veiculares.

6.1 Principais Contribuições

As principais contribuições deste trabalho podem ser descritas da seguinte maneira:

- A proposição de um esquema eficiente, cooperativo e dinâmico de alocação de recursos em várias nuvens veiculares, utilizando um modelo de Jogo de Mercado;

- A aplicação de um método aproximado para cálculo de *Shapley Values* para abordar o problema de alocação de recursos em redes veiculares;
- E uma extensa simulação da solução desenvolvida, variando parâmetros como o raio de comunicação, a taxa de serviços, os peso das tarefas e a quantidade de recursos compartilhados pelos veículos.

6.2 Publicações

O trabalho desenvolvido durante o Mestrado resultou em 3 artigos que foram publicados em uma revista internacional renomada na área, em uma conferência internacional e em uma conferência nacional:

- RIBEIRO JR., A.; da Costa, J. B. D.; FILHO, G. P. R.; VILLAS, L. A.; GUIDONI, D. L.; SAMPAIO, S.; MENEGUETTE, R. I. HARMONIC: Shapley values in market games for resource allocation in vehicular clouds. **Ad Hoc Networks**, v. 149, p. 103224, out. 2023. ISSN 1570-8705. (RIBEIRO JR. *et al.*, 2023).
- RIBEIRO JR., A.; FILHO, G. P. R.; GUIDONI, D. L.; de Grande, R. E.; SAMPAIO, S.; MENEGUETTE, R. I. A Shapley Value-based Strategy for Resource Allocation in Vehicular Clouds. In: **GLOBECOM 2022 - 2022 IEEE Global Communications Conference**. [S.l.: s.n.], 2022. p. 5801–5806. (RIBEIRO JR. *et al.*, 2022)
- RIBEIRO JR., A.; COSTA, J. B. D. da; FILHO, G. P. R.; VILLAS, L. A.; GUIDONI, D. L.; MENEGUETTE, R. I. Alocação de Tarefas em Nuvens Veiculares Utilizando Jogos de Mercado. In: **Anais do Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC)**. [S.l.]: SBC, 2022. p. 210–223. ISSN 2177-9384. (RIBEIRO JR. *et al.*, 2022)

REFERÊNCIAS

AL-SULTAN, S.; AL-DOORI, M. M.; AL-BAYATTI, A. H.; ZEDAN, H. A comprehensive survey on vehicular ad hoc network. **Journal of Network and Computer Applications**, v. 37, p. 380–392, 2014. ISSN 1084-8045. Disponível em: <https://www.sciencedirect.com/science/article/pii/S108480451300074X>. Citado na página 25.

ARTHURS, P.; GILLAM, L.; KRAUSE, P.; WANG, N.; HALDER, K.; MOUZAKITIS, A. A taxonomy and survey of edge cloud computing for intelligent transportation systems and connected vehicles. **IEEE Transactions on Intelligent Transportation Systems**, v. 23, n. 7, p. 6206–6221, 2022. Citado na página 26.

CISCO. **Driving Profits from Connected Vehicles**. [S.l.], 2020. Disponível em: <https://www.cisco.com/c/en/us/solutions/service-provider/mobile-internet/driving-profits-from-connected-vehicles.html>. Citado na página 25.

COSTA, J. B. D. da; PEIXOTO, M. L. M.; MENEGUETTE, R. I.; ROSÁRIO, D. L.; VILLAS, L. A. MORFEU: Mecanismo baseado em Otimização Combinatória para Alocação de Tarefas em Nuvens Veiculares. In: **Anais do Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC)**. [S.l.]: SBC, 2020. p. 505–518. ISSN 2177-9384. Citado nas páginas 26, 36, 38 e 60.

da Costa, J. B. D.; MENEGUETTE, R. I.; ROSÁRIO, D.; VILLAS, L. A. Combinatorial optimization-based task allocation mechanism for vehicular clouds. In: IEEE. **Proceedings of the IEEE 91st Vehicular Technology Conference (VTC Spring)**. [S.l.], 2020. p. 1–5. Citado nas páginas 34 e 38.

DANQUAH, W. M.; ALTILAR, D. T. Vehicular cloud resource management, issues and challenges: A survey. **IEEE Access**, v. 8, p. 180587–180607, 2020. Citado nas páginas 25 e 26.

DEMŠAR, J. Statistical comparisons of classifiers over multiple data sets. **Journal of Machine Learning Research**, v. 7, n. 1, p. 1–30, 2006. Disponível em: <http://jmlr.org/papers/v7/demsar06a.html>. Citado na página 59.

DUNN, O. J. Multiple comparisons using rank sums. **Technometrics**, Taylor & Francis, v. 6, n. 3, p. 241–252, 1964. Citado na página 59.

ESTER, M.; KRIEGEL, H.-P.; SANDER, J.; XU, X. A density-based algorithm for discovering clusters in large spatial databases with noise. In: **Proceedings of the Second International Conference on Knowledge Discovery and Data Mining**. [S.l.]: AAAI Press, 1996. p. 226–231. Citado na página 58.

FAN, W.; SU, Y.; LIU, J.; LI, S.; HUANG, W.; WU, F.; LIU, Y. Joint Task Offloading and Resource Allocation for Vehicular Edge Computing Based on V2I and V2V Modes. **IEEE Transactions on Intelligent Transportation Systems**, v. 24, n. 4, p. 4277–4292, abr. 2023. ISSN 1524-9050, 1558-0016. Citado nas páginas 36 e 38.

FRIEDMAN, M. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. **Journal of the american statistical association**, Taylor & Francis, v. 32, n. 200, p. 675–701, 1937. Citado na página 59.

GAI, K.; QIU, M.; ZHAO, H.; TAO, L.; ZONG, Z. Dynamic energy-aware cloudlet-based mobile cloud computing model for green computing. **Journal of Network and Computer Applications**, Elsevier, v. 59, p. 46–54, 2016. Citado na página 26.

KOUSER, R. R.; MANIKANDAN, T. A novel clustering and optimal resource scheduling in vehicular cloud networks using MKMA and the CM-CSO algorithm. **International Journal of Communication Systems**, v. 36, n. 5, p. e5424, 2023. ISSN 1099-1131. Citado nas páginas 36 e 38.

LEE, S.-S.; LEE, S. Resource allocation for vehicular fog computing using reinforcement learning combined with heuristic information. **IEEE Internet of Things Journal**, v. 7, n. 10, p. 10450–10464, 2020. Citado nas páginas 34 e 38.

LIEIRA, D. D.; QUESSADA, M. S.; CRISTIANI, A. L.; MENEGUETTE, R. I. Resource Allocation Technique for Edge Computing Using Grey Wolf Optimization Algorithm. In: **2020 IEEE Latin-American Conference on Communications (LATINCOM)**. [S.l.: s.n.], 2020. p. 1–6. ISSN 2330-989X. Citado na página 26.

LIU, J.; LI, J.; ZHANG, L.; DAI, F.; ZHANG, Y.; MENG, X.; SHEN, J. Secure intelligent traffic light control using fog computing. **Future Generation Computer Systems**, v. 78, p. 817 – 824, 2018. ISSN 0167-739X. Citado na página 26.

LIU, L.; FENG, J.; MU, X.; PEI, Q.; LAN, D.; XIAO, M. Asynchronous Deep Reinforcement Learning for Collaborative Task Computing and On-Demand Resource Allocation in Vehicular Edge Computing. **IEEE Transactions on Intelligent Transportation Systems**, p. 1–14, 2023. ISSN 1524-9050, 1558-0016. Citado nas páginas 36 e 38.

LOPEZ, P. A.; BEHRISCH, M.; BIEKER-WALZ, L.; ERDMANN, J.; FLÖTTERÖD, Y.-P.; HILBRICH, R.; LÜCKEN, L.; RUMMEL, J.; WAGNER, P.; WIESSNER, E. Microscopic traffic simulation using sumo. In: **The 21st IEEE International Conference on Intelligent Transportation Systems**. IEEE, 2018. Disponível em: <https://elib.dlr.de/124092/>. Citado na página 55.

LUO, Q.; LI, C.; LUAN, T.; SHI, W. Minimizing the delay and cost of computation offloading for vehicular edge computing. **IEEE Transactions on Services Computing**, IEEE, v. 1374, p. 1–12, 2021. Citado nas páginas 33 e 38.

MARQUES, H. A. P.; MENEGUETTE, R. I. Um Mecanismo de Alocação de Recursos em Nuvens Veiculares baseado em Teoria dos Jogos. In: **Anais Estendidos do Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC)**. [S.l.]: SBC, 2021. p. 241–248. ISSN 2177-9384. Citado nas páginas 35 e 38.

MASCHI, L. F.; PINTO, A. S.; MENEGUETTE, R. I.; BALDASSIN, A. Data summarization in the node by parameters (dsnp): local data fusion in an iot environment. **Sensors**, MDPI, v. 18, n. 3, p. 799, 2018. Citado na página 25.

MASCHLER, M.; SOLAN, E.; ZĀMÎR, Š. **Game Theory**. [S.l.: s.n.], 2020. ISBN 978-1-108-49345-1 978-1-108-82514-6. Citado nas páginas 27, 29, 30 e 31.

MENEGUETTE, R.; BOUKERCHE, A.; GRANDE, R. D. SMART: an efficient resource search and management scheme for vehicular cloud-connected system. In: **2Proceedings of the IEEE Global Communications Conference: Mobile and Wireless Networks**. Washington, USA: [s.n.], 2016. Citado na página 26.

MENEGUETTE, R.; GRANDE, R. D.; UEYAMA, J.; FILHO, G. P. R.; MADEIRA, E. Vehicular edge computing: Architecture, resource management, security, and challenges. **ACM Comput. Surv.**, Association for Computing Machinery, New York, NY, USA, v. 55, n. 1, 2021. ISSN 0360-0300. Disponível em: <https://doi.org/10.1145/3485129>. Citado na página 26.

MENEGUETTE, R. I.; BOUKERCHE, A. A cooperative and adaptive resource scheduling for vehicular cloud. In: **2017 IEEE Symposium on Computers and Communications (ISCC)**. [S.l.: s.n.], 2017. p. 398–403. Citado na página 25.

MENEGUETTE, R. I.; GRANDE, R. E. D.; LOUREIRO, A. A. F. **Intelligent Transport System in Smart Cities: Aspects and Challenges of Vehicular Networks and Cloud**. Cham: Springer International Publishing, 2018. (Urban Computing). ISBN 978-3-319-93331-3 978-3-319-93332-0. Disponível em: <http://link.springer.com/10.1007/978-3-319-93332-0>. Citado nas páginas 25 e 26.

MENEGUETTE, R. I.; MADEIRA, E. R. M.; BITTENCOURT, L. F. Multi-network packet scheduling based on vehicular ad hoc network applications. In: **2012 8th international conference on network and service management (cnsm) and 2012 workshop on systems virtualiztion management (svm)**. [S.l.: s.n.], 2012. p. 214–218. Citado na página 26.

MITCHELL, R.; COOPER, J.; FRANK, E.; HOLMES, G. Sampling permutations for Shapley value estimation. **The Journal of Machine Learning Research**, v. 23, n. 1, p. 43:2082–43:2127, jan. 2022. ISSN 1532-4435. Citado na página 32.

NABI, M.; BENKOCZI, R.; ABDELHAMID, S.; HASSANEIN, H. S. Resource assignment in vehicular clouds. In: IEEE. **2017 IEEE International Conference on Communications (ICC)**. [S.l.], 2017. p. 1–6. Citado na página 60.

NEMENYI, P. B. **Distribution-free multiple comparisons.** [S.l.]: Princeton University, 1963. Citado na página 59.

PAUL, A.; CHILAMKURTI, N.; DANIEL, A.; RHO, S. **Intelligent vehicular networks and communications: fundamentals, architectures and solutions**. [S.l.]: Elsevier, 2016. Citado na página 26.

PENG, H.; SHEN, X. Deep Reinforcement Learning Based Resource Management for Multi-Access Edge Computing in Vehicular Networks. **IEEE Transactions on Network Science and Engineering**, v. 7, n. 4, p. 2416–2428, out. 2020. ISSN 2327-4697. Citado na página 26.

PEREIRA, R. S.; LIEIRA, D. D.; SILVA, M. A. C. da; PIMENTA, A. H. M.; COSTA, J. B. D. da; ROSÁRIO, D.; VILLAS, L.; MENEGUETTE, R. I. RELIABLE: Resource allocation mechanism for 5g network using mobile edge computing. **Sensors**, MDPI AG, v. 20, n. 19, p. 5449, set. 2020. Disponível em: <https://doi.org/10.3390/s20195449>. Citado nas páginas 33 e 38.

PEREIRA, R. S.; LIEIRA, D. D.; SILVA, M. A. da; PIMENTA, A. H.; COSTA, J. B. da; ROSÁRIO, D.; MENEGUETTE, R. I. A novel fog-based resource allocation policy for vehicular clouds in the highway environment. In: **2019 IEEE Latin-American Conference on Communications (LATINCOM)**. [S.l.: s.n.], 2019. p. 1–6. Citado na página 26.

QUALCOMM. **Connecting vehicles to everything with C-V2X**. [S.l.], 2020. Disponível em: <https://www.qualcomm.com/research/5g/cellular-v2x>. Citado na página 25.

RIBEIRO JR., A.; COSTA, J. B. D. da; FILHO, G. P. R.; VILLAS, L. A.; GUIDONI, D. L.; MENEGUETTE, R. I. Alocação de Tarefas em Nuvens Veiculares Utilizando Jogos de Mercado. In: **Anais do Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC)**. [S.l.]: SBC, 2022. p. 210–223. ISSN 2177-9384. Citado na página 71.

RIBEIRO JR., A.; da Costa, J. B. D.; FILHO, G. P. R.; VILLAS, L. A.; GUIDONI, D. L.; SAMPAIO, S.; MENEGUETTE, R. I. HARMONIC: Shapley values in market games for resource allocation in vehicular clouds. **Ad Hoc Networks**, v. 149, p. 103224, out. 2023. ISSN 1570-8705. Citado na página 71.

RIBEIRO JR., A.; FILHO, G. P. R.; GUIDONI, D. L.; de Grande, R. E.; SAMPAIO, S.; MENEGUETTE, R. I. A Shapley Value-based Strategy for Resource Allocation in Vehicular Clouds. In: **GLOBECOM 2022 - 2022 IEEE Global Communications Conference**. [S.l.: s.n.], 2022. p. 5801–5806. Citado na página 71.

SAFDARIAN, A.; DIVSHALI, P. H.; BARANAUSKAS, M.; KESKI-KOUKKARI, A.; KULMALA, A. Coalitional game theory based value sharing in energy communities. **IEEE Access**, IEEE, v. 9, p. 78266–78275, 2021. Citado na página 29.

SHAPLEY, L. S. **A value for n-person games, Contributions to the Theory of Games, 2, 307–317**. [S.l.]: Princeton University Press, Princeton, NJ, USA, 1953. Citado na página 31.

SUN, Z.; LIU, Y.; WANG, J.; LI, G.; ANIL, C.; LI, K.; GUO, X.; SUN, G.; TIAN, D.; CAO, D. Applications of Game Theory in Vehicular Networks: A Survey. v. 23, n. 4, p. 2660–2710, 2021. ISSN 1553-877X. Citado na página 35.

SUN, Z.; SUN, G.; LIU, Y.; WANG, J.; CAO, D. BARGAIN-MATCH: A Game Theoretical Approach for Resource Allocation and Task Offloading in Vehicular Edge Computing Networks. **IEEE Transactions on Mobile Computing**, p. 1–18, 2023. ISSN 1536-1233, 1558-0660, 2161-9875. Citado nas páginas 37 e 38.

TANG, C.; ZHU, C.; WEI, X.; WU, H.; LI, Q.; RODRIGUES, J. J. Intelligent resource allocation for utility optimization in rsu-empowered vehicular network. **IEEE Access**, IEEE, v. 8, p. 94453–94462, 2020. Citado nas páginas 35 e 38.

UPPOOR, S.; FIORE, M. Large-scale urban vehicular mobility for networking research. In: **2011 IEEE Vehicular Networking Conference (VNC)**. [S.l.: s.n.], 2011. p. 62–69. Citado na página 55.

WEI, W.; YANG, R.; GU, H.; ZHAO, W.; CHEN, C.; WAN, S. Multi-objective optimization for resource allocation in vehicular cloud computing networks. **IEEE Transactions on Intelligent Transportation Systems**, IEEE, 2021. Citado nas páginas 26, 34 e 38.

WU, Q.; SHEN, J.; YONG, B.; WU, J.; LI, F.; WANG, J.; ZHOU, Q. Smart fog based workflow for traffic control networks. **Future Generation Computer Systems**, v. 97, p. 825 – 835, 2019. ISSN 0167-739X. Citado na página 26.

WU, X.; ZHAO, S.; ZHANG, R.; YANG, L. Mobility prediction-based joint task assignment and resource allocation in vehicular fog computing. In: IEEE. **IEEE Wireless Communications and Networking Conference (WCNC)**. [S.l.], 2020. p. 1–6. Citado nas páginas 33 e 38.

YANG, W.; ZHANG, R.; CHEN, C.; CHENG, X. Secrecy-Based Resource Allocation for Vehicular Communication Networks with Outdated CSI. In: **2017 IEEE 86th Vehicular Technology Conference (VTC-Fall)**. [S.l.: s.n.], 2017. p. 1–5. Citado na página 26.

YU, R.; HUANG, X.; KANG, J.; DING, J.; MAHARJAN, S.; GJESSING, S.; ZHANG, Y. Cooperative resource management in cloud-enabled vehicular networks. **IEEE Transactions on industrial electronics**, IEEE, v. 62, n. 12, p. 7938–7951, 2015. Citado nas páginas 35 e 38.

ZHANG, T.; GRANDE, R. E. D.; BOUKERCHE, A. Vehicular cloud: Stochastic analysis of computing resources in a road segment. In: **Proceedings of the 12th ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, & Ubiquitous Networks**. New York, NY, USA: Association for Computing Machinery, 2015. (PE-WASUN '15), p. 9–16. ISBN 9781450337595. Disponível em: <https://doi.org/10.1145/2810379.2810383>. Citado na página 26.

ZHANG, X.; SHEN, Z.; YANG, D. A permutation-based model for analysis of resource allocation overheads in vehicular ad hoc networks. **IEEE Access**, v. 9, p. 12282–12290, 2021. Citado na página 26.


ICMC
USP
SÃO CARLOS