Technical, energy performance, economic and environmental evaluation of vehicular and edge computing

Rosario Patanè

université PARIS-SACLAY

Technical, Energy Performance, Economic and Environmental Evaluation of Vehicular and Edge Computing

Évaluation technique, énergétique, économique et environnementale du calcul véhiculaire et du Edge Computing

Thèse de doctorat de l'Université Paris-Saclay

École doctorale n° 580, Sciences et Technologies de l'Information et de la Communication (STIC)
Spécialité de doctorat: Sciences des réseaux, de l'information et de la communication
Graduate School : Informatique et Sciences du Numérique. Faculté des Sciences d'Orsay.

Thèse préparée au **LISN (Université Paris-Saclay, CNRS)**, à **SAMOVAR (Télécom SudParis, IP Paris, IMT)** et à **Inria Saclay-Île-de-France (Université Paris-Saclay, Inria)**, sous la direction de **Lila BOUKHATEM**, Professeure, le co-encadrement de **Andrea ARALDO**, Maître de conférences, et de **Nadjib ACHIR**, Chargée de recherche.

Thèse soutenue à Paris-Saclay, le 4 Novembre 2025, par

Rosario PATANÈ

Composition du jury

Membres du jury avec voix délibérative

Riadh DHAOU Professeur, INP-ENSEEIHT, IRIT, Université de Toulouse, France	Président & Rapporteur
Marcelo DIAS DE AMORIM Directeur de Recherche, LIP6, Sorbonne Université, CNRS, France	Rapporteur & Examinateur
Kinda KHAWAM Maîtresse de Conférences, LISN, Université Paris-Saclay, CNRS, France	Examinatrice
Claudio Enrico PALAZZI Professeur, Département de Mathématiques « Tullio Levi-Civita », Università degli Studi di Padova, Italie	Examinateur

THESE DE DOCTORAT

NNT : 2025UPASG074

université PARIS-SACLAY

ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Titre: Évaluation technique, énergétique, économique et environnementale du calcul véhiculaire et du Edge Computing

Mots clés: Informatique en nuage, Théorie des jeux, Réseaux véhiculaires, Informatique en périphérie de réseau, Investissement en périphérie de réseau, 5G.

Résumé: Cette thèse étudie la faisabilité technique, économique et environnementale des architectures de calcul pour le support d'applications sensibles au délai, telles que la Réalité Augmentée (AR) et la Conduite Autonome (AD). Bien que le Cloud Computing (CC) constitue aujourd'hui le paradigme dominant, mais il ne peut pas garantir une latence suffisamment faible pour ce type d'applications fortement contraintes en délai. Cette limitation est surmontée par l'Edge Computing (EC), qui consiste à déployer des capacités de calcul à la périphérie du réseau d'accès. Toutefois, l'EC implique des coûts d'infrastructure élevés et soulève des préoccupations environnementales, liées notamment au cycle de vie court des équipements électroniques (environ quatre ans) et à l'augmentation de la consommation énergétique.

Parallèlement, le nombre de véhicules connectés ne cesse de croître. Ces véhicules embarquent déjà des ressources de calcul et de communication qui peuvent être exploitées de manière opportuniste, non seulement pour les tâches liées à la conduite, mais aussi pour le délestage de calcul en provenance de dispositifs externes, tels que les smartphones, ordinateurs portables ou appareils de santé portables des utilisateurs finaux. Ces ressources disponibles dans les véhicules peuvent être gérées selon le paradigme du Vehicular Cloud Computing (VCC).

Dans cette thèse, nous analysons tout d'abord la faisabilité économique du déploiement de l'Edge Computing à l'aide d'un modèle fondé sur la théorie des jeux, en montrant comment la coopération multi-acteurs peut atténuer les coûts de déploiement élevés. Ensuite, nous évaluons sous quelles conditions le VCC peut *remplacer* l'EC, c'est-à-dire dans quelle mesure le délestage de tâches vers les véhicules peut offrir des performances comparables à celles de l'EC. Les résultats sont obtenus à partir de simulations de réseaux basées sur des scénarios de mobilité urbaine réalistes. Nous montrons que le VCC peut atteindre une latence ultra-faible, de l'ordre de 10 ms, même lorsque la densité des véhicules est faible. Une analyse comparative des coûts révèle que remplacer l'EC par le VCC permet de réduire les dépenses d'infrastructure d'environ 10 % sur cinq ans.

Enfin, nous proposons une approche de gestion du VCC visant à optimiser la consommation énergétique, les émissions de carbone et à calculer une répartition équitable des revenus générés par les tâches des utilisateurs finaux. Cette approche est basée sur la programmation mathématique et la théorie des jeux coopératifs. À travers une simulation de Monte-Carlo, nous montrons que la consommation énergétique imputable au VCC représente moins de 0,1 % de la consommation totale du véhicule dans des scénarios réalistes, et que les propriétaires de véhicules reçoivent des incitations substantielles à participer à l'exécution des tâches.

En résumé, cette thèse démontre la faisabilité des architectures de réseaux mobiles de nouvelle génération, telles que l'Edge Computing et le Vehicular Cloud Computing, pour le support d'applications à très faible latence.

université PARIS-SACLAY
ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Title: Technical, Energy Performance, Economic and Environmental Evaluation of Vehicular and Edge Computing

Keywords: Cloud computing, Game Theory, Vehicular Networks, Edge Computing, Edge Investment strategies, 5G.

Abstract: This thesis investigates the technical, economical and environmental feasibility of computing architectures for supporting delay-sensitive applications such as Augmented Reality (AR) and Autonomous Driving (AD). While Cloud Computing (CC) is the prevalent computing paradigm today, it cannot offer sufficiently low latency. This limitation is overcome by Edge Computing (EC), which consists in deploying computational capability at the edge of the access network. However, EC entails high infrastructure costs and raises environmental concerns, due to the short lifecycle of electronic devices (around four years) and increased energy consumption. Meanwhile, the number of connected vehicles is steadily increasing. These vehicles already carry onboard computing and communication resources that can be opportunistically exploited, not only for driving-related tasks, but also for task offloading computation from external devices, e.g., smartphones, laptops, or wearable health devices of end-users. These resources available in the vehicles can be managed under the paradigm of Vehicular Cloud Computing (VCC). In this thesis, we first analyze the economic feasibility of Edge Computing deployment through a game-theoretic model, showing how multi-tenant cooperation can mitigate the high cost of deployment. Then, we evaluate under which conditions VCC can *replace* EC, i.e., whether offloading tasks to vehicles can provide similar performance to EC. Results are obtained via high fidelity network and mobility simulations, in an urban mobile network scenario. We find that VCC can achieve ultra-low latency, with delays of about 10 ms, even when vehicles are sparsely distributed. A comparative cost analysis shows that replacing EC with VCC can reduce infrastructure expenditure by approximately 10% over five years. Finally, we propose a VCC management scheme to optimize energy consumption, carbon emissions, and to compute a fair allocation of the revenues generated by service end-user tasks. The scheme is based on mathematical programming and coalitional game theory. Via Monte-Carlo simulation, we show that energy consumption due to VCC is below 0.1% of the overall vehicle consumption in realistic scenarios, and that vehicle owners receive substantial incentives for participating in task execution.

In summary, this thesis demonstrates the feasibility of future generation mobile network architectures, such as Edge Computing and Vehicular Cloud Computing, to support extremely low-latency applications.

Acknowledgements

I would like to express my sincere gratitude to Lila Boukhatem, Nadjib Achir, and Andrea Araldo for their invaluable guidance, constructive feedback, and continuous support throughout my doctoral studies. Their precision, attention to detail, and commitment to high-quality research have been instrumental in the development of this work.

I also extend my thanks to Tijani Chahed for his constant support and valuable scientific advice during these years.

I am grateful to my colleagues and collaborators: Alessa Mayer, Alessandro Spallina, Ali Al Khansa, Amal Sakr, Amirhesam Badeanlou, Anna Frisone, Ayoub Ben Ameur, Cristina Venitucci, Duo Wang, Jorge Mirande, Marco Di Prima, Massimo Gollo, Massinissa Ait Aba, Maya Kassis, Mohamed Abdullah, Mohamed Ourahou, Rania Sahraoui, Salvatore Cavallaro, Wei Huang, Wendlasida Abdoul Hakim Ouedraogo, Yarui Zhang, You Wu, and many others whose contributions and discussions have enriched this research.

Special recognition is given to Anna Frisone and Cristina Venitucci from the Laboratory of Information, Networking and Communication Sciences (LINCS) for their efforts in organizing scientific events and fostering a collaborative environment.

I also wish to thank the members of my thesis committee for evaluating my work and for their constructive feedback, which has significantly improved the quality of this manuscript.

I am particularly grateful to Laurent Roullet, invited committee member, for his presence during the defense and for the valuable insights he provided.

I also wish to express my gratitude to a few people whose names I deliberately choose not to mention here. They know who they are, and their support, kindness, and presence have meant more than I can put into words.

Finally, I acknowledge the financial support of Labex DigiCosme, which made this research possible.

Contents

List of Figures

List of Tables

Acronyms

3GPP Third Generation Partnership Project
5G Fifth Generation
AD Autonomous Driving
AP Access Point
AR Augmented Reality
AWS Amazon Web Services
BS Base Station
CAPEX Capital Expenditure
CC Cloud Computing
CN Core Network
EC Edge Computing
eMBB enhanced Mobile Broadband
ETSI European Telecommunications Standards Institute
FR1 Frequency Range 1
gNB next Generation Node Base Station
GPU Graphics Processing Unit
MIMO Multiple Input Multiple Output
MIPS Million Instructions Per Second
MI Million Instructions
MEC Multi-access Edge Computing
mMTC massive Machine-Type Communication
NO Network Operator
NR New Radio
NS3 Network Simulator version 3
OPEX Operational Expenditure
QoS Quality of Service
RB Resource Block
RSU Road-side Unit
SGW Serving Gateway
SP Service Provider

SUMO Simulator of Urban MObility

TTI Transmission Time Interval

URLLC Ultra-Reliable Low-Latency Communication

UE User Equipment

V2X Vehicle-to-Everything

VCC Vehicular Cloud Computing

VUE Vehicle User Equipment

1 - Introduction

1.1 . Motivation and Context

The number of connected vehicles is growing rapidly, and it is expected to continue its upward trend. Recent forecasts predict that the number of connected vehicles will exceed **367 million units globally by 2027**, with **Fifth Generation (5G) accounting for 23% of automotive cellular connections** [2]. These vehicles are not only mobile network clients, but are increasingly equipped with *advanced onboard computing resources*, such as Central Processing Units (CPUs), Graphics Processing Units (GPUs), and memory devices. These hardware components are primarily designed to support internal functions such as *navigation, braking, engine control, and chassis stabilization*, which are essential for the vehicle's safety and operational stability.

To fulfill these critical in-vehicle tasks, computing resources are dimensioned with deterministic performance guarantees in mind. However, empirical studies have indicated that such resources are *not fully utilized all the time*, particularly when the vehicle is idle or operating under non-challenging conditions [3]. This opens the possibility to *opportunistically reuse onboard computing resources for external purposes*, such as supporting nearby mobile devices by performing external task computations. This concept, commonly referred to as *task offloading*, has attracted increasing interest in the research community due to its potential to reduce end-device battery consumption [4] and improve application performance [5], across a wide range of services.

Task offloading refers to the delegation of computation-intensive or energy-consuming processes from resource-constrained devices (e.g., smartphones, wearables) to more powerful external nodes. This strategy brings several tangible benefits: improved *Quality of Service (QoS)*, extended *battery life*, and the enablement of *latency-sensitive applications* such as Augmented Reality (AR), real-time gaming, and video processing.

While offloading to the *Cloud* is well established, it is often unsuitable for ultra-low-latency applications. Round-trip delays between mobile devices and centralized data centers can easily exceed tens of milliseconds, which is unacceptable for real-time services [6]. As a response, the *Edge Computing (EC)* paradigm has been introduced, where computation is performed on servers deployed at the edge of the network, closer to the end-users, e.g., at Base Stations (BSs) or Road-side Units (RSUs). This reduces latency and network congestion, but the deployment of edge nodes is costly, involving significant Capital Expenditure (CAPEX), maintenance, and energy consumption [7–9]. In practice, operators have often been *reticent* to invest at scale due to unclear

business cases and limited monetization paths [7, 9], which turns an ostensibly technical solution into an economic and environmental challenge.

This is where *Vehicular Cloud Computing (VCC)* comes into play. VCC refers to the use of *idle computational resources* on vehicles to perform offloaded tasks originating from mobile users or nearby devices. Unlike EC, *VCC exploits already-deployed infrastructure*, thus significantly reducing or eliminating deployment costs. Vehicles equipped with processors such as NVIDIA's automotive-grade GPUs demonstrate that their hardware capabilities can match, or even surpass, those of edge servers in many scenarios [10].

In the literature, multiple use cases for VCC have been proposed. These include *inter-vehicle cooperation* to enhance road safety [11], *task offloading from pedestrians* to vehicles for improved QoS in applications like gaming and AR [12], and even *resource pooling* to support local cloudlets in smart cities [13]. Despite the promising potential, VCC has mostly been treated as a *backup layer* to the Edge, activated only when EC resources are saturated.

In our proposed studies, we challenge this hierarchy. In particular, through empirical simulation-based evaluations using *Simulator of Urban MObility (SUMO) for vehicular mobility* and *Network Simulator version 3 (NS3) for 5G communication modeling*, we show that *VCC can support a wide range of latency-sensitive applications* with *task completion times comparable to or better than EC*, particularly in dense urban areas [14]. These findings suggest that *VCC can be a viable substitute or complement to EC*, not only in terms of performance but also in terms of *economic sustainability*. Indeed, the *total cost of ownership* for EC includes not only hardware procurement but also installation, site leasing, cooling, power supply, and ongoing maintenance, whereas the marginal cost of VCC is minimal because the required resources are *already integrated in the vehicle and maintained by the manufacturer* [14]. In principle, network operators could access these resources via standardized APIs or cooperative agreements.

Naturally, VCC is not without challenges. Vehicles are mobile entities with *variable availability and connectivity*. Resource visibility is affected by the freshness of beacon messages, and *handover interruptions* may result in task failures. Moreover, *in-vehicle driving and safety tasks must always take precedence* over offloaded workloads, which imposes constraints on resource allocation [15]. Beyond technical volatility, adoption also hinges on *incentive compatibility*: stakeholders require credible economic returns, and society increasingly demands solutions that are *energy- and carbon-aware*.

Accordingly, this thesis adopts a *holistic perspective* on the *computing continuum* Cloud, Edge, and Vehicular and addresses three intertwined gaps. First, we examine **how to enable EC despite investment hesitancy** by proposing cooperative, game-theoretic co-investment schemes that align costs and benefits among heterogeneous actors [7–9]. Second, we determine

when VCC can substitute or complement EC without sacrificing latency or reliability, grounding the analysis in high-fidelity mobility and network simulations [14]. Third, we design **incentive mechanisms that internalize energy and environmental externalities**, showing how revenue sharing and carbon/energy-aware management can make participation attractive for vehicle owners and sustainable for operators.
In this light, the guiding research questions become: *(i) How can Edge Computing be economically enabled at scale? (ii) Under which conditions can vehicles replace or complement Edge Computing while meeting delay and reliability performance targets? (iii) Which incentive and sustainability mechanisms energy, carbon, and revenue-aware are necessary to make Vehicular Cloud Computing practically and socially viable?*

1.2 . Contributions of the Thesis

This thesis addresses whether the *VCC* can represent a sustainable and effective alternative to *EC* in 5G-enabled networks. The first step is to enable EC itself, by overcoming its main economic barrier through cooperative investment models. Once EC is made feasible, we then assess its actual contribution in terms of energy efficiency and QoS. This provides the necessary baseline to investigate under which conditions VCC can replace or complement EC. Accordingly, this thesis offers a structured investigation of these paradigms, moving from economic enablement of EC, to the evaluation of its limitations, and finally to positioning VCC as a viable, sustainable substitute.

The main contributions are summarized as follows:

a. **Assessment of EC deployment feasibility**
   We introduce a game-theoretic model showing how the main obstacle to large-scale, geographically distributed EC deployment can be overcome through cooperation between Network Operator (NO) and Service Provider (SP). By pooling their resources, these stakeholders can jointly invest in the edge infrastructure needed to handle a given traffic load, effectively supporting task offloading.

b. **Systematic comparison between EC and VCC for very low-latency applications.**
   We compare EC and VCC in supporting applications with low latency requirements. Using realistic vehicular mobility (SUMO) and 5G communication simulations (NS3), we demonstrate that VCC can effectively replace EC in urban and highway environments, once provided sufficient vehicular density and stable connectivity.

c. **Monetization strategies for energy-aware VCC using game theory.**
   Finally, we explore how VCC, as a **sustainable computing** paradigm,

can be monetized. By applying *coalitional game theory* and intelligent resource allocation mechanisms, we propose fair and incentive-compatible models for revenue sharing among stakeholders, ensuring economic viability while promoting sustainable use of vehicular resources.

Together, these contributions trace a progression from enabling EC, to assessing its limitations, and ultimately to demonstrating how VCC can emerge as a sustainable and high-performance alternative.

1.3 . Thesis Organization

The related work is presented for each different work to guide the reader across the progress of this thesis, which is more logically exposed at each chapter

- **Chapter 2 – Background:** This chapter provides an overview of the key concepts in 5G network architecture, CC, EC , and VCC. It also provides a definition of **task offloading** and describes its processing.
- **Chapter 3 – Coalitional Game-Theoretical Approach to Coinvestment with Application to Edge Computing:**
  We introduce an optimization framework for EC, focusing on profit maximization while deploying computation resources in a coinvestment framework ran by coalitional game theory.
- **Chapter 4 – VCC as an Alternative to Edge:**
  This chapter explores VCC as a substitute for EC. We analyze the feasibility of offloading tasks to vehicles. Extensive simulations are used to compare VCC performance with the EC.
- **Chapter 5 – Sustainable Monetization Potential of Vehicular Computing:**
  We propose architectural and economic models for integrating VCC into 5G networks. Using game-theory, we explore incentive mechanisms for stakeholders, ensuring sustainable energy-aware resource usage and cost-effective computation.
- **Chapter 6 – Conclusions and Future Work:**
  The final chapter summarizes the key findings, discusses the limitations of the current study, and outlines directions for future research, including multi-task applications and more advanced resource orchestration strategies.

1.4 . List of Publications

1.4.1 . Main publications

1 **Rosario Patanè**, Andrea Araldo, Tijani Chahed, Diego Kiedanski and Daniel Kofman, "Coalitional Game-Theoretical Approach to Coinvestment with Application to Edge Computing," 2023 IEEE 20th Consumer Communications & Networking Conference (CCNC), Las Vegas, NV, USA, **2023**, pp. 517-522, doi: 10.1109/CCNC51644.2023.10060093 [Link].

2 **Rosario Patanè**, Nadjib Achir, Andrea Araldo and Lila Boukhatem, "Can Vehicular Cloud Replace Edge Computing?," 2024 IEEE Wireless Communications and Networking Conference (WCNC), Dubai, UAE, **2024**, [Link].

3 **Rosario Patanè**, Nadjib Achir, Andrea Araldo and Lila Boukhatem, "Vehicular Cloud Computing: a Cost-Effective alternative to Edge Computing in 5G Networks", Computer Networks Journal (Elsevier), 2025, [Link].

4 **Rosario Patanè**, Nadjib Achir, Andrea Araldo and Lila Boukhatem, "A Management Framework for Vehicular Cloud toward Economicand Environmental Efficiency," **submitted to:** IEEE International Conference on Computer Communications (INFOCOM 2026).

1.4.2 . Other publications

5 Amal Sakr, Andrea Araldo, Tijani Chahed, **Rosario Patanè** and Daniel Kofman, "Co-investment Under Uncertainty: Coalitional Game Formulation and Application to Edge Computing," 2025 IEEE International Conference on Communications (ICC), Montreal, Canada **2025**, [Link]..

2 - Background

2.1 . Introduction to 5G Architecture and Services

The fifth generation of mobile networks (5G) represents a major architectural evolution compared to previous technologies like 4G and 3G. It has been designed to support a wide spectrum of heterogeneous applications, each with specific requirements in terms of latency, reliability, connection density, and energy efficiency. One of its main innovations is *network slicing*, which enables the creation of logically independent virtual networks *slices* on top of the same physical infrastructure.

Three major types of service use cases are supported:

- **enhanced Mobile Broadband (eMBB)**, which supports high-throughput services such as 4K/8K video streaming, augmented and virtual reality;
- **Ultra-Reliable Low-Latency Communication (URLLC)**, which targets applications that require extremely low end-to-end latency (≤ 1 ms) and high availability, such as autonomous vehicles and remote surgery;
- **massive Machine-Type Communication (mMTC)**, designed to support a very high density of devices (up to 1 million/km^2), typical in large-scale IoT deployments.

This service heterogeneity can be accommodated by **network slicing**, through which each logical network can be tailored to meet the requirements of a specific service category.

These capabilities enable a new class of applications that were either impossible or impractical in previous generations. For example, vehicular applications such as *object recognition* and *cooperative sensing* require real-time communication with strong guarantees for latency and reliability. This is now achievable through URLLC. Similarly, cloud gaming, AR, and AD require stringent response.

Hence, 5G networks are suitable for *task offloading*, a paradigm where tasks generated by mobile devices are transmitted to more powerful external nodes such as edge servers, cloud data centers, or even nearby vehicles to be processed. Offloading allows for better battery management and improved performance.

The feasibility of offloading in 5G is further enhanced by several radio innovations:

Table 2.1: Main 5G network services and their characteristics

Service Type	Use Case	Latency	Data Volume
eMBB	Enhanced Mobile Broadband services like 4K video, AR/VR, and gaming	$\sim$10–100 ms	High
URLLC	Mission-critical applications such as Autonomous Driving (AD) and remote surgery	$\leq$1 ms	Moderate
mMTC	Large-scale IoT like smart meters and sensors with sporadic transmission	Tolerant (seconds)	Very low

- **Multiple Input Multiple Output (MIMO):** By using large antenna arrays at both transmitter and receiver, 5G increases data throughput and robustness.
- **Beamforming:** 5G Base Stations (BS) can dynamically direct signal energy towards users, improving signal quality and reducing interference, particularly important in urban and vehicular environments.
- **Resource Block (RB) and Flexible Resource Allocation:** In 5G, the basic unit of radio scheduling is the Resource Block (RB), defined as 12 sub-carriers over a time slot. Unlike LTE, 5G introduces scalable sub-carrier spacing and adaptable transmission intervals, enabling the next Generation Node Base Station (gNB) to perform fine-grained, dynamic allocation of RBs. This flexibility, including the use of shorter mini-slots, allows resources to be tailored to application-level requirements such as latency sensitivity or bandwidth demand, thereby supporting a wide range of offloading scenarios.

Altogether, these technical advances make 5G not only faster, but also more deterministic and programmable, enabling real-time control, dynamic resource allocation, and context-aware service delivery. This forms the technical foundation for the computation and offloading paradigms discussed in the next sections.

2.2 . Task Offloading in 5G: Opportunities and Challenges

Task offloading is a paradigm in which computational tasks generated by a resource-constrained device are transferred to a more capable remote node for execution. This is particularly valuable in mobile and vehicular contexts, where devices such as smartphones, tablets, wearable sensors, or connected vehicles often face limitations in processing power, battery capacity, or thermal constraints.

The core idea behind offloading is to shift the computational burden from the device to the infrastructure, thereby improving performance, reducing energy consumption, and enabling new classes of services that would otherwise be infeasible. Thanks to the low-latency and high-throughput capabilities introduced by 5G, this strategy becomes far more effective than in previous network generations.

In a typical offloading scenario, the device collects or generates data (e.g., a camera image, a user interaction, or a sensor measurement), encapsulates it as a task, and transmits it to a more powerful node via the 5G radio access network. The node may reside in the cloud, at the edge, or in paradigms such as VCC on a nearby vehicle. Once processed, the result is returned to the originating device for further action or display.

The benefits of computation offloading in 5G include:

- **Reduced processing time**, as tasks are executed on more powerful hardware;
- **Lower energy consumption**, since the device avoids local prolonged CPU or GPU usage;
- **Support for demanding applications**, such as augmented reality, online gaming, and real-time object recognition;
- **Improved responsiveness**, due to 5G's reduced latency and fast uplink/downlink channels [16].

However, these advantages depend heavily on network conditions and the nature of the offloading destination. Tasks that are highly delay-sensitive such as AD decisions or haptic feedback may suffer from even minimal transmission delays, making the task offloading destination choice critical.

In addition, computation offloading introduces several challenges:

- **Task partitioning and scheduling:** deciding which parts of an application can be offloaded and when;
- **Context-awareness:** ensuring that decisions are made based on current network status, mobility, load, and task size;

- **Security and privacy:** as data may traverse third-party infrastructure, offloading exposes new attack surfaces and compliance concerns;

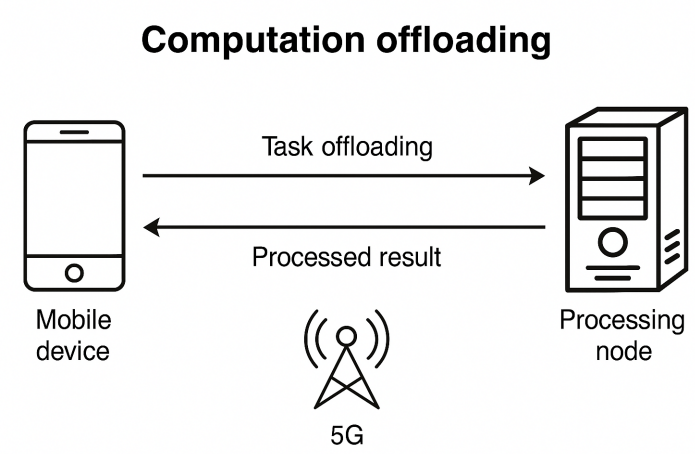


Figure 2.1: *5G-based computation offloading process. Tasks are generated by a mobile device and transmitted to an external processing node via 5G.*

The effectiveness of offloading, therefore, is not determined solely by the network, but by a careful integration of *connectivity*, *allocation protocols*, and *contextual choices*. In this light, the following sections explore the characteristics of the three main offloading destinations: the cloud, the edge, and the vehicular cloud.

2.3 . Cloud Computing in Task Offloading

CC provides virtually unlimited computational and storage resources by exploiting large-scale data centers deployed in geographically distributed locations. In the context of task offloading, the cloud represents the most powerful and scalable external node to which mobile devices can delegate processing tasks.

From a purely computational perspective, the cloud offers several advantages:

- It can handle **complex and resource-intensive workloads**, such as large-scale data analysis, machine learning training, or high-resolution rendering;
- It offers **high availability and redundancy**, making it suitable for services requiring robustness and scalability;
- It supports **elastic resource provisioning**, enabling dynamic adaptation to fluctuating workloads.

In early mobile systems, CC is today the default option for supporting offloading. However, as new latency-sensitive applications have emerged, its **intrinsic limitations** have become more evident. The primary drawback is **network latency**. Since cloud data centers are typically located far from end-users often in remote or centralized regions. This implies that the time required to transmit data to/from the cloud can severely impact application responsiveness.

For applications such as AD, AR, or remote control systems, even a few tens of milliseconds of latency can result in **QoS violations** or unsafe behavior. Moreover, the use of long-distance communication consumes more energy on mobile devices, since radio interfaces must remain active for longer transmission times. It may also lead to **variable performance**, as end-to-end traffic must traverse multiple network domains, where Internet congestion, variable queuing, and sub-optimal routing decisions can introduce fluctuating delays and throughput. In addition, error-control coding for long-distance transmissions typically requires higher redundancy to counteract noise and fading, thereby reducing the effective data rate and further impacting performance.

Another consideration is **backhaul congestion**. Offloading all tasks to the cloud may overload the transport network between BS and data centers, which is costly to scale and may become a bottleneck in high-density scenarios, such as smart cities or large public events.

In this context, CC remains a **strategic part of the *continuum computing***, particularly for background and long-running tasks, or services that tolerate high latency. However, for real-time or delay-sensitive applications, it must be complemented with closer, more responsive computing layers, namely *EC* and, as this thesis explores, *VCC*.

The following sections will examine these paradigms, emphasizing how they address the limitations of CC offloading while enabling more sustainable and context-aware resource usage.

2.4 . Edge Computing: Low-Latency Offloading at the Network Periphery

To overcome the limitations of typical cloud-based solutions, *EC* performs task offloading closer to the end-user typically at or near the BS, Access Point (AP), or small-scale edge data centers.

The main advantages of EC for task offloading include:

- **Reduced end-to-end latency**, thanks to the physical proximity of the computing node to the user;

- **Lower energy consumption** on user devices, as shorter transmission distances and faster processing reduce radio and CPU usage;
- **Improved QoS predictability**, since local resources are less affected by wide-area Internet congestion;
- **Contextual awareness**, allowing edge nodes to respond to local mobility, network conditions, or service popularity.

For applications such as real-time object detection, AR, or Vehicle-to-Everything (V2X) communications, EC allows mobile devices to meet strict deadline constraints while offloading compute-intensive tasks. It is also a key enabler for network slicing in URLLC scenarios, where microseconds of delay can affect safety or performance.

However, the benefits of EC come at a cost. Unlike centralized cloud data centers, edge nodes must be deployed densely and geographically distributed, which involves:

- High CAPEX for infrastructure installation and provisioning;
- Ongoing Operational Expenditure (OPEX) for energy, maintenance, and updates;
- Scalability concerns, since provisioning sufficient capacity at every site to match peak demand is economically inefficient.

Furthermore, the deployment of Edge Computing faces another obstacle: new cloud data centers are being established increasingly closer to end-users, while the advent of terabit-scale communication links will reduce network delays, making the latency guarantees of Edge comparable to those of Cloud Computing [17]. Nevertheless, it remains highly likely that Edge will be deployed in alternative forms, adapted to specific contexts and use cases. These challenges and transformations strengthen the case for exploring complementary offloading paradigms that do not rely exclusively on fixed infrastructure. One such paradigm is *VCC*, which leverages the existing onboard computing resources of vehicles as opportunistic, mobile edge nodes.

2.5 . Vehicular Cloud Computing

VCC exploits the underutilized computing resources of modern vehicles such as CPUs, GPUs, and memory to perform external tasks elaboration. These resources, originally provisioned to support critical in-vehicle operations (e.g., navigation, braking control, engine diagnostics), are typically dimensioned for peak load but remain partially idle or underused during regular operation. VCC proposes to exploit this unused capacity for computation

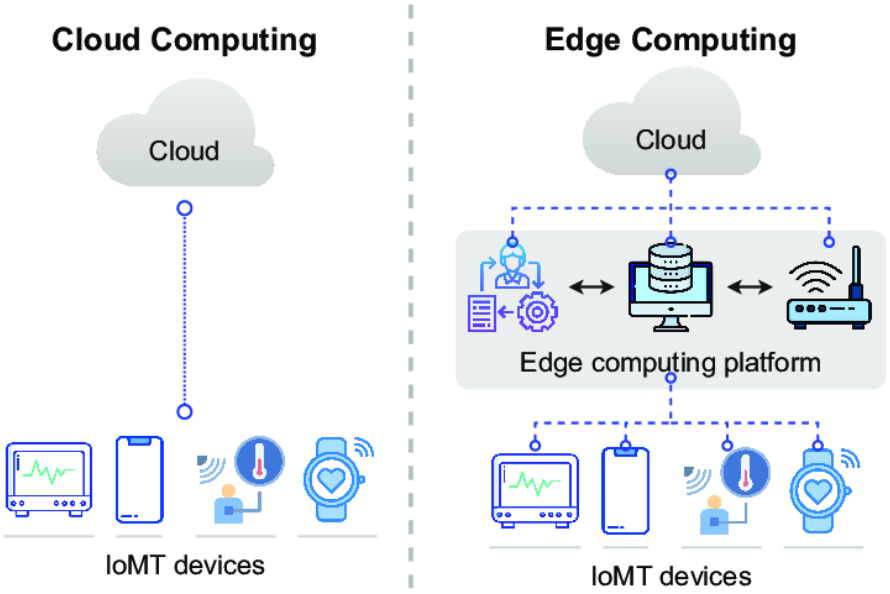


Figure 2.2: *Comparison between Cloud and EC paradigms. The EC is closer to the end-devices [1].*

offloading from external devices or other vehicles, creating a distributed, mobile, and cost-effective cloud infrastructure.

The rationale behind VCC is simple yet powerful: unlike edge or cloud nodes, vehicles are already equipped with substantial computational resources and are deployed at large scale. These resources can be made accessible and enable third-party services to run on-board opportunistically.

The main advantages of VCC are:

- **Infrastructure-free deployment:** no additional installation is required, as resources are already embedded in the vehicle hardware;
- **Proximity to users:** especially in dense urban settings, vehicles are often close to mobile users, enabling low-latency communication;
- **Cost-effectiveness:** since the hardware is maintained by vehicle owners or manufacturers, operational costs for network operators are minimized.

These features make VCC a promising complement or in some scenarios, an alternative to EC. It is particularly relevant in situations where fixed infrastructure is expensive or infeasible, such as in temporary deployments, sparsely covered regions, or pop-up networks.

However, the VCC model also introduces new challenges that must be addressed:

- **Resource availability is volatile**, since vehicles can join or leave the coverage area unpredictably;
- **Mobility introduces uncertainty** in communication delays and task completion times;
- **Internal vehicle tasks have priority**, so offloading tasks must not interfere with safety-critical operations;
- **Trust and security concerns** arise, especially when resources are shared among unknown users or entities.

Despite these challenges, VCC aligns well with the goals of sustainability and flexibility in next-generation networks. By turning idle vehicular resources into computation nodes, it reduces the need for costly infrastructure, balances computational load across the network, and supports sustainable computing strategies. Moreover, it opens opportunities for monetization and cooperation among network operators, vehicle manufacturers, and service providers, topics further explored in later chapters.

2.6 . Summary

This chapter has outlined the key technological foundations needed to analyze task offloading in 5G-enabled environments. Beyond simply describing the three main computation paradigms, it has highlighted their defining characteristics, advantages, and inherent limitations, showing how each addresses part of the offloading challenge while leaving open issues that motivate the analyses of the following chapters. In particular, the main paradigms can be contrasted as follows:

- **CC** provides virtually unlimited capacity and scalability, but its geographic centralization introduces high latency that makes it unsuitable for many real-time applications;
- **EC** reduces delay by placing resources closer to users, yet this comes at the expense of high deployment and maintenance costs and limited coverage, raising questions about its economic viability at scale;
- **VCC** leverages already-embedded vehicular resources to offer a flexible and low-cost option, though its inherent volatility and mobility-induced uncertainty make reliable provisioning more complex.

Each paradigm plays a distinct role within the *continuum computing*, which must balance performance, energy consumption, deployment cost, and service quality.

5G technology enhances all these paradigms through improved bandwidth, ultra-low latency, and advanced radio features such as massive MIMO and beamforming. These network capabilities create new opportunities for task offloading, making it a viable solution for latency-sensitive, energy-constrained, or compute-intensive applications.

However, choosing the right offloading destination node in the cloud, edge, or vehicle depends on the specific context: task characteristics, user location, infrastructure availability, and economic constraints.

The following chapters investigate this decision-making process in depth, through a series of original research contributions. We begin by modeling **cooperative investment in edge infrastructure**, before progressively introducing **vehicular computing as an alternative** to EC.

3 - Coalitional Game-Theoretical Approach to Coinvestment with Application to Edge Computing

3.1 . Introduction

In this chapter, we propose a coinvestment plan between several stakeholders of different types, namely a physical NO, operating network nodes, e.g. a network operator or a tower company, and a set of Service Providers (SPs) willing to use these resources to provide services as video streaming, AR, AD assistance, etc. One such scenario is that of deployment of EC resources. Indeed, although the latter technology is ready, the high CAPEX cost of such resources is the barrier to its deployment. For this reason, a solid economical framework to guide the investment and the returns of the stakeholders is key to solve this issue. We formalize the coinvestment framework using coalitional game theory. We provide a solution to calculate how to divide the profits and costs among the stakeholders, considering their characteristics: traffic load, revenues, utility function. We prove that it is always possible to form the grand coalition composed of all the stakeholders, by showing that our game is convex. We derive the payoff of the stakeholders using the Shapley value concept, and elaborate on some properties of our game. We show the performance of our solution through extensive simulations.

3.2 . Introduction to Coinvestment

Coinvestment represents a promising economic framework that enables multiple stakeholders to collaboratively share both expenses and revenues when deploying infrastructure projects that would otherwise be financially unfeasible for any single entity. This thesis develops a formal model of coinvestment using coalitional game theory, which can be applied to scenarios where:

- costly resources must be deployed across distributed nodes,
- access to these nodes is controlled by a single entity (the NO), and
- the deployed resources deliver value to multiple Service Providers.

EC serves as the primary application domain for this framework, as it exemplifies the economic challenges of modern distributed computing infrastructure. In EC, computational resources must be deployed at the network edge

by the NO (e.g., telecom operators like AT&T or tower companies), while Service Providers (ranging from video streaming platforms to automotive manufacturers offering AD capabilities) benefit from the reduced latency and improved service quality these resources enable. The high deployment costs of EC infrastructure which cannot be supported solely by NOs explain its limited adoption to date, despite its technical readiness. Meanwhile, Service Providers can realize substantial benefits by offering enhanced services to end-users through edge resources.

Our contributions in formalizing this coinvestment scenario include:

- proposing a coalitional game-theoretic model that accounts for the distinct roles of NOs and Service Providers,
- proving the existence and stability of the grand coalition through game convexity analysis,
- demonstrating that the Shapley value provides an equitable distribution of costs and benefits that lies within the core, and
- evaluating the model's performance under various realistic deployment scenarios.

This thesis demonstrates that through the designed coinvestment model, all stakeholders can achieve greater economic benefits through cooperation than they could independently, thus providing a pathway to accelerate EC adoption while ensuring fair distribution of both costs and benefits.

3.3. State of the Art in Multi-Tenant Coinvestment and Resource Sharing

The advantage of sharing resources between several tenants has been shown in [18], for the case of passive optical networks. However, they assume resources are already deployed and allocation to tenants is determined via auctions. We instead consider the case where no resources are deployed yet (as for EC) and thus devise a mechanism for all the players to coinvest to buy and deploy resources, jointly taking into account the resource allocation among players.

An example of coinvestment is given in [19]. Investors are producers/consumers of energy, and a battery exists that must be sized to meet everyone's needs. Each of them can decide whether to charge or discharge the battery to use stored energy later, when energy is more expensive (daily hours). The result of coinvestment is the price that each one has to pay to buy the battery. The problem is solved using coalitional game theory [20], as we shall do in this work. However, we cannot directly use these results

because [19] is a linear production problem, valid to model the cost of electricity, while in our case the benefit of using resources at the Edge can be non-linear, due to the diminishing return in resource deployment at the Edge [21]. Another peculiarity of our study is that we have a veto player, which corresponds to the NO.

In [22] a coinvestment between NOs is realized to improve energy efficiency in cellular access networks, i.e., in some parts of the day, except the peak hours, the resources of a single NP can be oversized, so, just a subset of NPs can serve all the load allowing the others to save energy. The resultant benefit of this sharing is later divided among the players. This paper is very close to the problem in this work, but does not match exactly because:

- the players are at the same level (all NPs), while in our case players have different roles (NO and SPs),
- in our problem, we do not care about exchanging load between players to save energy.

Let us now introduce some game theoretical background.

3.4 . Coinvestment model

Our coinvestment problem is modeled as a coalitional game with transferable payoff, i.e. players can share a common amount of utility/cost [23]. The game is defined by the tuple $(\mathcal{N}, v)$, where $\mathcal{N}$ is the set of players and the *coalitional value* $v(\mathcal{S})$ is a function that associates a value to any subset $\mathcal{S} \subseteq \mathcal{N}$, called *coalition*. The players are one physical NO and N SPs. The NO is the entity that owns the network nodes. It could be a network operator, who owns the location of an antenna or a central office. It could be a separated tower company [24]. The set of players is $\mathcal{N} = \{1, \ldots, N, NO\}$. The first question to be asked in coalitional games is whether the grand coalition $\mathcal{N}$, formed by all players, is stable, i.e. all the players have an incentive to be part of it, in terms of individual payoff. The payoff of any player $i \in \mathcal{N}$ is $x^i = r^i - p^i$, where r^i and p^i are the revenues and the payment, i.e. the capital cost, respectively.

We assume that the NO is willing to host physical resources on its nodes. To fix ideas, such nodes are edge nodes and resources are CPU in our case. They might also refer to GPU, etc.

After the deployment, the NO virtualizes and allocates the resources to SPs. SPs do not own physical resources, only the NO does, in BS for instance. Whenever players form a coalition, they invest together in deploying an amount of computational capacity C, measured in *millicores*. Denoting by h^i the resources allocated to player i, $\sum_{i \in S} h^i = C$. For d denoting the price expressed in *dollars* per resource unit, the sum of all players' payments

should be such that $\sum_{i \in S} p^i = d \cdot C$. As the NO is the only player that hosts the capacity, if it does not join the coalition, coinvestment cannot take place, and so, no computational capacity can be deployed and no EC can be realized. So, for any coalition $\mathcal{S}$, capacity is subject to:

$$C = \begin{cases} \frac{1}{d} \cdot \sum_{i \in S} p^i \text{ if NO} \in \mathcal{S} \\ 0 \text{ otherwise} \end{cases} \tag{3.1}$$

At every timeslot t, each player i has an expected load, l_t^i, i.e. the average number of requests coming from users of SP i at a time t, and which is exogenous to the problem. We assume that the average load profile is the same every day. Each player i has an instantaneous utility u_t^i, in monetary units, e.g. *dollars*, which represents the revenues coming from the end users of the services. In other words, the utility is what users pay to an SP to consume its services. We assume the load and the shape of the utility function are known and truthful inputs. Similarly to [25], the utility is a function of the expected load l_t^i (the more users consume the service, the more they pay) and of the allocated resources h^i (the more resources, the better the service, the more users are willing to pay):

$$u_t^i = u^i(l_t^i, h^i); \quad u^i(l_t^i, 0) = 0 \tag{3.2}$$

We thus assume (Eq. (3.2) on the right) that the utility is null if no resources are allocated.

Observe that the case when the NO uses some Edge resources for itself can be easily modeled in our framework by introducing a fictitious SP representing the NO using resources of EC. The NO does not offer any Edge service to the end users, so, its load is null ($l_t^{\text{NO}} = 0$) and thus it does not need Edge resources for itself ($h^{\text{NO}} = 0$), which implies that $u_t^{\text{NO}} = 0$ and $\sum_{i \in \mathcal{S} \setminus \{\text{NO}\}} h^i = C$. However, the NO gets a fraction of the value of the grand coalition if it exists.

The revenues of a coalition are the sum of the utilities of the SPs over the investment period: $\sum_{i \in \mathcal{S}} r^i = D \cdot \sum_{i \in \mathcal{S}} \sum_{t \in [T]} u_t^i$, where $D = 365 \cdot Y$, Y is the duration in years of the investment, $T = 96$ is the number of timeslots in one day, considering each timeslot has duration $15\ minutes$.

We now define the value $v(\mathcal{S})$ of any coalition $\mathcal{S} \subseteq \mathcal{N}$. When forming a coalition $\mathcal{S}$, the players involved choose allocation vector $\vec{h}$ and capacity C to maximize the coalition value $v(\mathcal{S})$:

$$v(\mathcal{S}) = \max_{\vec{h}, C} v^{\vec{h}, C, \mathcal{S}} \triangleq \max_{\vec{h}, C} D \sum_{i \in \mathcal{S}} \sum_{t=1}^{T} u^i(l_t^i, h^i) - d \cdot C \tag{3.3}$$

$$\text{s.t.} \sum_{i \in \mathcal{S} \setminus \{\text{NO}\}} h^i = C; \quad h^{\text{NO}} = 0. \tag{3.4}$$

$$C, h^i \geq 0, \ \forall t \in [T], \forall i \in \mathcal{S}. \tag{3.5}$$

Observe that NO is a **veto player**. Indeed, without NO, the investment does not take place [23, §13.2] (see (3.1)). Therefore, applying (3.4), (3.2) and (3.3), if NO $\notin \mathcal{S}$, we get $v(\mathcal{S}) = 0$. We will also see that the SPs form altogether a veto player too.

3.5 . Analysis

We now assess the cooperative structure of our game and its stability. We show the existence of the core, and hence the formation of the grand coalition.

3.5.1 . Core and convexity of the game

Let us define a payoff vector $(x^i)_{i\in\mathcal{N}}$. The core is a set of payoff vectors, such that a payoff vector is in the core if the payoffs of each player are such that no subgroup can gain by quitting the grand coalition and forming a different coalition [23]. A well-known result in coalitional game theory affirms that the core is non-empty if the game is convex [26] and that a particular payoff vector having some "fairness" properties, i.e. the Shapley value [20, 26] lies in the core.

Theorem 1. *(The game is convex). Our game $(\mathcal{N}, v)$, whose value function v is described by the optimization problem* (3.3)-(3.5) *is convex.*

Proof. We can rewrite $v(\mathcal{S})$, via (3.3), (3.4), as

$$v(\mathcal{S}) = \sum_{i\in\mathcal{S}} \max_{h^i} v^{h^i} \; ; \quad v^{h^i} \triangleq D \cdot \sum_{t=1}^{T} u^i(l_t^i, h^i) - d \cdot h^i \tag{3.6}$$

The maximum over h^i of Eq. (3.6) gives the contribution of a single player i to the coalitional value, considering its part of the revenues due to its utility function, and the cost of the resources h^i it uses to produce this utility. Observe that such a contribution is independent of the coalition $\mathcal{S}$ in which i participates. Eq. (3.6) shows that the contribution is separable, i.e. it is the summation of the value functions of the individual players.

We now prove that the game is supermodular, which implies its convexity, thanks to [27]. A game is supermodular if

$$\Delta_i(\mathcal{T}) \leq \Delta_i(\mathcal{S}), \forall \mathcal{T} \subseteq \mathcal{S} \subseteq \mathcal{N} \setminus \{i\}, \; \forall i \in \mathcal{N} \tag{3.7}$$

$$\text{where } \Delta_i(\mathcal{S}) = v(\mathcal{S} \cup \{i\}) - v(\mathcal{S}) \underset{(3.6)}{=} \max_{h^i} v^{h^i} \tag{3.8}$$

is the marginal contribution of player i to the coalition $\mathcal{S}$.

Let us fix any $i \in \mathcal{N}$. Given two coalitions, $\mathcal{S}$ and $\mathcal{T}$, such that $\mathcal{T} \subseteq \mathcal{S} \subseteq \mathcal{N} \setminus \{i\}$, we calculate the marginal contribution of a player i to both coalitions.

We consider four cases:

a. Case NO $\in \mathcal{T}$ and $i = \text{SP}^i$. For coalition $\mathcal{T}$ we have (see (3.6)): $\Delta_i(\mathcal{T}) = \max_{h^i} v^{h^i}$. For coalition, $\mathcal{S}$ we have: $\Delta_i(\mathcal{S}) = \max_{h^i} v^{h^i}$ so, the marginal contributions are $\Delta_i(\mathcal{S}) = \Delta_i(\mathcal{T})$.

b. Case $i =$ NO. In this case, the proof is trivial, in fact NO $\notin \mathcal{T} \cup \mathcal{S}$, so, for the fact that NO is a veto player.

$$v(\mathcal{T} \cup \{i\}) \geq 0, v(\mathcal{T}) = 0, \ \forall \mathcal{T} \setminus \{\text{NO}\} \tag{3.9}$$

$$\text{and } v(\mathcal{S} \cup \{i\}) \geq 0, v(\mathcal{S}) = 0, \ \forall \mathcal{S} \setminus \{\text{NO}\} \tag{3.10}$$

Therefore, Eq. (3.7) is verified if and only if

$$v(\mathcal{S} \cup \{i\}) - v(\mathcal{T} \cup \{i\}) \geq 0 \tag{3.11}$$

which is equivalent to $\sum_{j \in \mathcal{S} \setminus \mathcal{T}} \max_{h^j} v^{h^j} \geq 0$ where the last inequality is obviously true, since we have a sum of non-negative terms.

c. Case NO $\notin \mathcal{T} \cup \mathcal{S}$ and $i = \text{SP}^i$. This case is easy to prove because we get $v(\mathcal{T} \cup \{i\}) - v(\mathcal{T}) = v(\mathcal{S} \cup \{i\}) - v(\mathcal{S}) = 0$, which satisfies the definition of supermodularity.

d. Case NO $\notin \mathcal{T}$ and $i = \text{SP}^i$. In this case the marginal contribution of i to the coalition $\mathcal{T}$ is null and Eq. (3.7) becomes $\Delta_i(\mathcal{S}) \geq 0$. To verify this, we observe that, thanks to (3.6):

$$\Delta_i(\mathcal{S}) = v(\mathcal{S} \cup \{i\}) - v(\mathcal{S}) = \begin{cases} \max_{h^i} v^{h_i} & \text{if NO} \in \mathcal{S} \\ 0 & \text{otherwise} \end{cases} \geq 0$$

This completes the supermodularity proof and thus convexity. So, the grand coalition can be always formed. □

3.5.2 . Shapley value

Finding a mechanism to share the payoff among players is not trivial. One idea would be to divide the payoff equally among players. However, this would not be accepted, since some players contribute to the coalition more than others. First, the NO is a veto player, and its contribution is of primary importance. Second, the SPs do not contribute equally to the coalition: some SPs have more users than others. A second idea would be to share payoffs proportionally to the request load of each SP. However, this would still be unfair, as the benefits collected by SPs do not only depend on the quantity of requests, but also on their type (see §3.6.1).

The Shapley value, a somehow fair way to share the payoff, considers the marginal contribution of each player to all the possible coalitions. For convex games, which is our case thanks to h. 1, the Shapley value lies in the core [26]. It is computed as [23]: $x^i = \phi^i = \frac{1}{|\mathcal{N}|!} \sum_{\mathcal{S} \subseteq \mathcal{N} \setminus \{i\}} |\mathcal{S}|! \cdot (|\mathcal{N}| - |\mathcal{S}| - 1)! \cdot \Delta_i(\mathcal{S})$.

3.5.3 . Initial investment of players

Now that we derived the payoff x^i for each player, we need to calculate how much each player must pay at the beginning of the investment, i.e. p^i. This is obtained by solving the following equations:

$$r^i - p^i = x^i, \forall i \in \mathcal{N} \tag{3.12}$$

$$\text{s.t. } \sum_{i \in \mathcal{N}} r^i = D \cdot \sum_{i \in \mathcal{N}} \sum_{t=1}^{T} u^i(l_t^i, h^{*i}) \tag{3.13}$$

$$\text{where: } \vec{h}^*, C^* = \arg\max_{\vec{h}, C} v^{\vec{h}, C, \mathcal{N}} \text{ s.t. (3.4)-(3.5)} \tag{3.14}$$

3.5.4 . Relevant properties of our game

If a player i does not produce revenues and makes no payments, then it is a null player, i.e. $v(\mathcal{S} \cup \{i\}) = v(\mathcal{S})$ [28]. Note that there can be players who do not pay for the resources or are even paid ($p^i \leq 0$) and still positively contribute to the coalition by collecting revenues r^i. The NO is never a null player because it is a veto player who always contributes to any coalition.

Theorem 2. *(Payoff sharing) The Shapley outcome of the game $(\mathcal{N}, v)$, where v is described by the problem* (3.3)-(3.5)*, is divided equally between the NO and the set of all SPs.*

Proof. Consider the Shapley value of the game $(\mathcal{N}, v)$, i.e. the payoff vector $(\phi_i)_{i \in \mathcal{N}}$. We want to prove that the Shapley value of the NO is equal to the sum of the Shapley values of all SPs. To calculate the Shapley value, we need the value of the marginal contribution of any player i to the coalition, i.e. $\Delta_i(\mathcal{S}) = \max_{h^i} v^{h^i}$. The NO is a veto player, and so, the v function is null for coalitions without it.

$$\Delta_{\text{NO}}(\mathcal{S}) = v(\mathcal{S} \cup \{\text{NO}\}) - v(\mathcal{S}) = \sum_{i \in \mathcal{S}} \max_{h^i} v^{h^i} - 0 = \sum_{i \in \mathcal{S}} \Delta_i(\mathcal{S}), \forall \mathcal{S} \subseteq \mathcal{N} \setminus \{\text{NO}\} \tag{3.15}$$

Now, we can show that the Shapley value of the NO is equal to the sum of the SPs Shapley values. We know that the Shapley value is in the core, which is subject to the efficiency property, $\sum_{i \in \mathcal{N}} \phi_i = v(\mathcal{N})$. Hence, $v(\mathcal{N}) = \phi_{\text{NO}} + \sum_{i \in \mathcal{N} \setminus \{\text{NO}\}} \phi_i$. The Shapley value of the NO is

$$\phi_{\text{NO}} = \frac{1}{|\mathcal{N}|!} \sum_{\mathcal{S} \subseteq \mathcal{N} \setminus \{NO\}} |\mathcal{S}|! \cdot (|\mathcal{N}| - |\mathcal{S}| - 1)! \cdot \Delta_{\text{NO}}(\mathcal{S}) = \frac{1}{|\mathcal{N}|!} \sum_{\mathcal{S} \subseteq \mathcal{N} \setminus \{NO\}} |\mathcal{S}|! \cdot (|\mathcal{N}| - |\mathcal{S}| - 1)! \cdot \sum_{i \in \mathcal{S}} \Delta_i(\mathcal{S}) \tag{3.16}$$

as $\Delta_{\text{NO}}(\mathcal{S}) = \sum_{i \in \mathcal{S}} \Delta_i(\mathcal{S})\ \forall \mathcal{S} \subseteq \mathcal{N} \setminus \{\text{NO}\}$ (Eq. (3.15)).

This implies the coalitional value is divided equally between the NO and the set of SPs, $\phi_{\text{NO}} = \sum_{i \in \mathcal{N} \setminus \{\text{NO}\}} \phi_i = \frac{v(\mathcal{N})}{2}$. This completes the proof. □

The intuition behind this equal sharing of the Shapley value between the NO and the SPs is based on [29]:"*each game is decomposed into a weighted sum of unanimity games in which the Shapley value assigns an equal share of a unit to each veto player".* In our case, if the set of SPs is considered as one super-player, it is actually a veto player as well, because the value function is zero if no SP is in the coalition, since it would not be possible to collect revenues from users utilization.

3.5.5 . Edge Resource Deployment

EC represents an ideal application domain for our coinvestment framework, as it involves costly computational resources that must be deployed at the network edge by a NO, while providing benefits to multiple Service Providers with varying service requirements.

The deployment of EC resources follows the model described in Section 3, where the NO virtualizes physical resources (typically measured in millicores of CPU, though our framework can also accommodate GPU and other computation resource types) and allocates them to SPs based on their service needs and payment contributions.

To model realistic edge deployment scenarios, we capture the daily traffic patterns of SPs serving residential users using the model from [30]:

$$l_t^i = a_0 + \sum_{k=1}^{K} a_k \sin\left(2k\pi \frac{t - t_k}{T}\right) \tag{3.17}$$

where t is the timeslot, T is the number of timeslots in one day, and a_k and t_k are hyperparameters determining the amplitude and the offset of each sinusoidal component. This model captures the typical daily fluctuations in user demand.

The utility derived by each SP from Edge resources exhibits a diminishing return effect [21], reflecting the real-world observation that marginal benefits decrease as more resources are added. We model this utility using the following increasing and concave function:

$$u^i(l_t^i, h^i) = \beta^i \cdot l_t^i \cdot (1 - e^{-\xi \cdot h^i}) \tag{3.18}$$

where β^i represents the benefit factor of the player i the value gained from serving one unit of load at the Edge and ξ models the rate at which the utility approaches its upper bound $\beta^i \cdot l_t^i$.

3.5.6 . Revenue Sharing Mechanisms

Our model provides a principled approach to sharing both the costs and benefits of EC deployment. The Shapley value ensures that each stakeholder receives a fair portion of the overall value created, based on their marginal contributions to all possible coalitions.

An important property of our revenue sharing mechanism is the equal division of value between the NO and the collective of SPs each receiving exactly half of the coalition's value. This reflects the symmetry of power in the deployment process: the NO provides essential infrastructure, while the SPs collectively generate all revenue through their services.

The specific payment p^i from each player is determined by solving the equation system described in Sec. 3.5.3, which ensures that each player's net payoff matches their Shapley value. Interestingly, this sometimes results in negative payments for certain players, meaning they effectively receive compensation from other players rather than contributing to the initial investment. This occurs when a player's contribution to the coalition's value significantly exceeds the cost of resources they utilize.

3.6 . Application to Edge Computing

3.6.1 . Parameters

Load

We define the load as an exogenous variable (§3.4). To reproduce a realistic trend, we consider the daily traffic profile of a SP serving residential users, as modeled in [30], i.e. $l_t^i = a_0 + \sum_{k=1}^{K} a_k \sin\left(2k\pi \frac{t-t_k}{T}\right)$, where t is the timeslot and T is the number of timeslots in one day; a_k and t_k are hyperparameters determining the amplitude and the offset of each of the K sinusoidal components. We take their values from [30, Fig.2].

Utility function and price

As often observed in reality, we assume the utility (3.2) of any SP^i is characterized by a diminishing return effect [21]: the marginal utility increment becomes smaller by increasing the allocated resources h^i. For this reason, we model the utility with the following increasing and concave function, similar to (1) of [25]:

$$u^i(l_t^i, h^i) = \beta^i \cdot l_t^i \cdot (1 - e^{-\xi \cdot h^i}) \tag{3.19}$$

The term β^i is the *benefit factor* of player i which represents the benefit that a SP gets from serving one unit of load at the Edge. It is a multiplicative

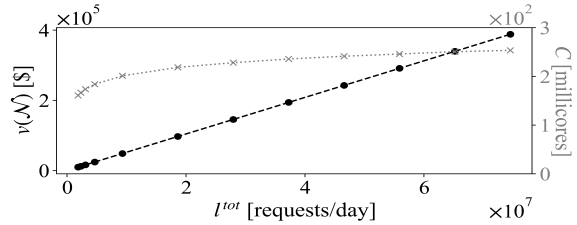


Figure 3.1: *Capacity and coalitional value as a function of the overall daily load.*

constant, null for the NO, $\beta^{\text{NO}} = 0$. The term ξ models the shape of the diminishing return, i.e. how fast it saturates to its upper bound $\beta^i \cdot l_t^i$. Note that this utility function follows property (3.2).

3.6.2 . Scenario with 2 SPs of the same type

In this case, there are two SPs of the same type: $\beta^{\text{SP}^1} = \beta^{\text{SP}^2} = \hat{p}$ where $\hat{p} \triangleq \frac{d}{D \cdot T}$ is the price, $d = 0.05$ *dollars/millicores*, amortized over each of the T time slots over the investment duration, D. SP^1 and SP^2 have the same temporal trends, but we take $l_t^1 = 4l_t^2 \; \forall t$.

In Fig. 3.1, we show the capacity of purchased CPU and the value of the grand coalition, as a function of the daily total load, $l^{\text{tot}} = \sum_{t=1}^{T} l_t^{\text{tot}}$. We observe that, the more the load, the more the capacity installed to serve it. However, recall that the utility functions follow a diminishing return regarding the resources, so, the trend of capacity C is sublinear. We observe a linear trend for coalitional value because the value function is linearly dependent on the load (see Eq. (3.19)).

We observe in Fig. 3.2 the capacity sharing between the SPs, SP^1 receives a larger capacity: it has to serve a larger part of the requests. Note that, even if the load of SP^1 is 4 times the load of SP^2, the difference between the resource allocated to them is not that big: a consequence of the diminishing return.

The contribution of SP^i to the coalitional revenues is defined as $\hat{r}^i = D \cdot \sum_{t=1}^{T} u^i(l_t^i, h^i)$. We denote the grand coalitional revenues as $r^{\mathcal{N}} = \sum_{i \in \mathcal{N}} r^i$, where r^i is the result obtained in (3.12)-(3.14) given by $r^i = D \cdot \sum_{t=1}^{T} u^i(l_t^i, h^{*i})$.

The term $\hat{r}^i$ is the number of revenues produced by SP i, due to the served load during the overall duration of the coinvestment.

Fig. 3.2 shows that most contribution comes from SP^1, since its load is four times higher than that of SP^2, and the utility of any SP (and thus its contribution to the grand coalitional revenues) is proportional to the served load; indeed, we observe that $\hat{r}^{\text{SP}^1} = 4\hat{r}^{\text{SP}^2}$. Note that h^{NO} and r^{NO} are not in the figure, as the NO does not use resources because its load is null; this implies that its utility is null, so it does not produce revenues for the grand coalition by serving a load.

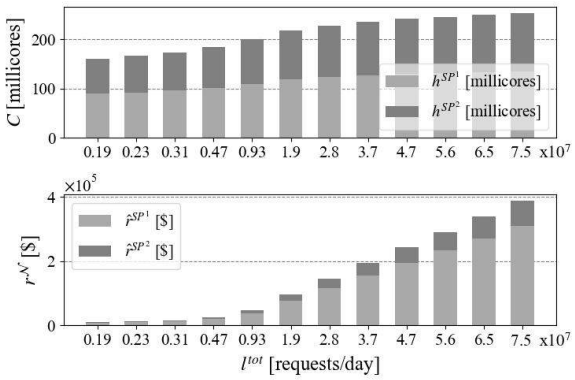


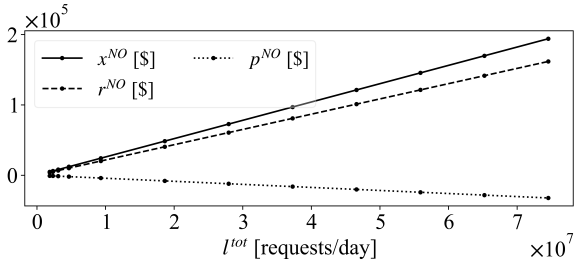
Figure 3.2: *Split of capacity and contributions to coalitional value.*

Fig. 3.3 shows the outcome of the game, i.e. the payment p^i, the revenue r^i and the payoff x^i of each player $i \in \mathcal{N}$ given by the Shapley value. We first observe that the payoff of each of them increases with the total load, which is obvious as the revenues of the grand coalition are the sum of the utilities of each player, which in turn increase with the number of served requests. It is interesting to notice that only SP^2 actually pays to deploy the resources at the Edge, while the NO and SP^1 have negative payments, so, they are not paying, but are being paid. The "privilege" of the NO and SP^1 can be explained by the fact that they are the most important for the coalition: NO is the veto player and SP^1 brings to the coalition most of the revenues collected from users (Fig. 3.2).

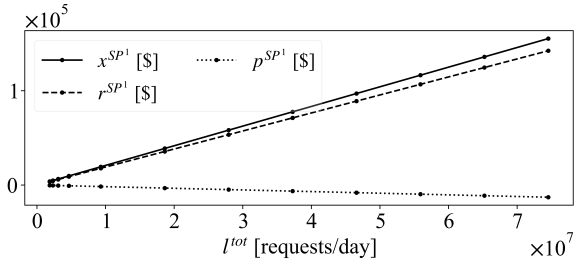
3.6.3 . Scenario with 2 SPs of different types

In this case, we have two SPs offering different types of services. For instance, SP^2 may offer an extremely low-latency service, e.g. augmented reality, while SP^1 may offer a less stringent service, e.g. online gaming. In this case, the utility coming from serving a request at the Edge is much higher for SP^2 than for SP^1, since the latter could serve some of the requests (for instance requests not related with interactions with the player) from the Cloud without degrading too much the perceived user experience. Therefore, we consider now that $\beta^{\text{SP}^2} \geq \beta^{\text{SP}^1}$. As in the previous scenario, $\beta^{\text{tot}} = \beta^{\text{SP}^1} + \beta^{\text{SP}^2} = 2\hat{p}$. We assume further that $\beta^{\text{SP}^1} = (1-\omega) \cdot \beta^{\text{tot}}$ and $\beta^{\text{SP}^2} = \omega \cdot \beta^{\text{tot}}$. We make ω vary in $[0.5, 1]$. The case $\omega = 0.5$ corresponds to the previous scenario, where the SPs were of the same type. By increasing ω, the two SPs become more heterogeneous, and in particular SP^2 has higher benefits per unit of load than SP^1. We keep the load as before.

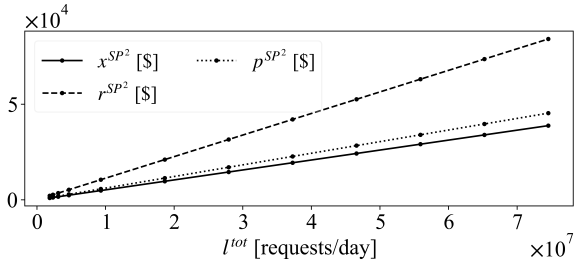
We now show how resource allocation and payoff change with ω. We observe in Fig. 3.4a that increasing ω, the percentage of CPU given to SP^1 decreases and is null for $\omega = 1$, at which point it is given entirely for SP^2. This is



(a) *Payoff, revenues and payment by NO.*



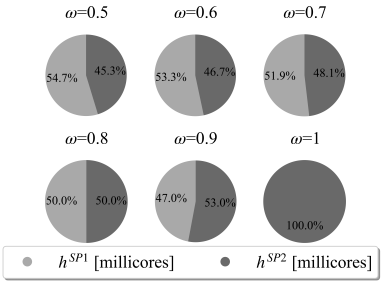
(b) *Payoff, revenues and payment by SP^1.*



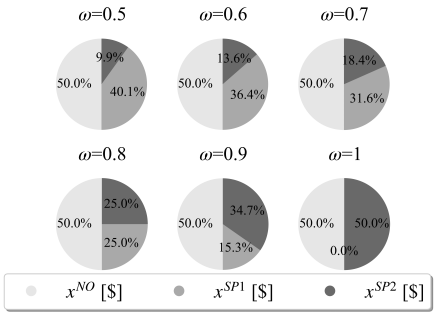
(c) *Payoff, revenues and payment by SP^2.*

Figure 3.3: *Shapley value: payoffs, revenues and payments.*

due to the fact that, despite it attracts most of the user load, SP[1] is less useful to assign resources to, as its benefit factor becomes smaller with ω. This tells us, as expected, that resource allocation at the Edge must be taken not only based on the load, but also on the nature of the services, and in particular on time-sensitivity, which is reflected in a different benefit per unit of load satisfied at the Edge. In Fig. 3.4 the payoff sharing reflects what we mentioned above. The marginal contribution brought by SP[2] increases since it produces most of the coalition revenues. In all the cases, the NO has 1/2 of the coalitional value, (Theo. 2).



(a) *Capacity subdivision among the players.*



(b) *Payoff sharing.*

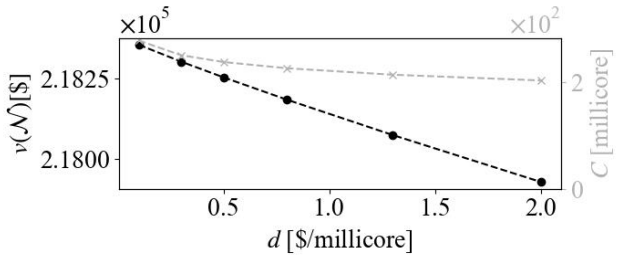
Figure 3.4: *Coalitional value and capacity function of ω.*

3.6.4 . Price sensitivity analysis

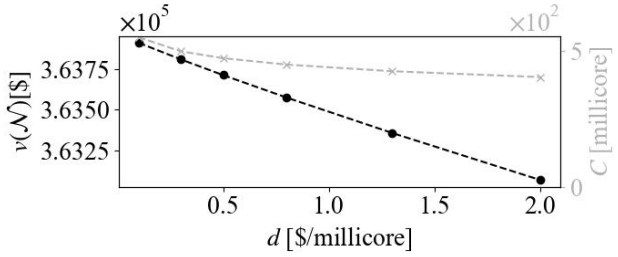
Fig. 3.5 assesses how the behavior of the coalition changes with an increasing number of SPs for different price d. Obviously, when the price for resources increases, the purchased capacity is reduced. It is interesting to notice that this reduction is sublinear, because of (3.19). On the other hand, coalitional value decreases linearly. These trends remain consistent when changing the number N of SPs. Fig. 3.5 also confirms that adding a player to the game brings a higher benefit in terms of the value of the coalition v. This is in line with the supermodularity of v which states that more value is obtained with a larger number of players in the coalition, which we used in (Theo. 1) to prove the convexity of our game and hence its stability.

3.7 . Conclusion

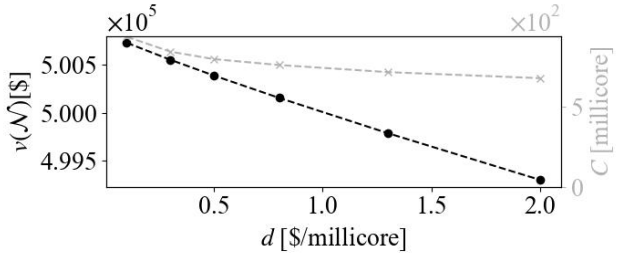
We proposed a coalitional game theory solution to enable coinvestment between heterogeneous players (NO and SPs) and applied it to the deployment of Edge Computing. By showing the convexity of the game, we proved that the core is non-empty and that the Shapley value, which provides a fair way to divide income among players, lies in the core. So, it is always possible



(a) *NO and $N = 2$ SPs.*



(b) *NO and $N = 4$ SPs.*



(c) *NO and $N = 7$ SPs.*

Figure 3.5: *Coalitional value and capacity as function of price d and number of SPs.*

to form the grand coalition, made of all players. We studied numerically the solution, in terms of purchased capacity and its sharing between SPs as well as their revenues, payments and payoffs under the Shapley value, and this for different scenarios, homogeneous and heterogeneous SPs, in terms of so-called benefit factor which may be related for instance to how time-critical is the service they offer. We eventually studied the sensitivity of capacity and coalitional value to the number of SPs and to resource unitary price.

4 - Vehicular Cloud Computing as an Alternative to Edge

In the previous chapter, we consolidated and assessed a framework to determine when EC can be deployed so that benefits are realised while costs are fairly shared among stakeholders. In this chapter, we take a further step: we evaluate edge performance against that of VCC. Our results show that VCC exhibits a strong capability for meeting strict low-latency requirements. The analysis also show the conditions under which EC could be substituted. EC is found to be roughly 10% more expensive than VCC.

4.1 . Introduction

At the beginning of the 21st century, major data center companies realized that their computing resources were not being fully utilized during regular working hours. Despite this under-utilization, these resources still consumed energy to remain operational [31]. To address this inefficiency, companies started renting out their unused computing resources through pay-on-demand models [32]. Notable examples include the introduction of Amazon Web Services (AWS) in 2006 [33] and Google App Engine in 2008 [34], both of which popularized commercial Cloud services based on the concept of CC.

Currently, the Cloud provides essential services such as storage and computing, making the concept of CC a practical and integral part of modern applications. Unfortunately, as the demand for stringent delay requirements evolves, CC is unable to meet these increasingly critical application needs. Indeed, while CC can theoretically handle large computational and storage resources [35], its latency is often excessive for modern delay-sensitive applications. For instance, according to Verizon's continuous network latency monitoring [6], typical round-trip delays are about 35 ms within North America and 70 ms for transatlantic communications. These values serve as a reliable baseline for assumptions about Cloud latency. As a result, a new paradigm emerged with the idea of geographically bringing Cloud services closer to end-users, giving rise to the concept of EC [36].

This new paradigm places computational resources closer to users, which may significantly decrease the delays [37]. Nevertheless, EC has some limitations compared to CC. Indeed, enabling EC requires deploying numerous Edge nodes across a wide geographic area. This endeavor turns out to be very costly, underscoring the need for research studies on the investment required for Edge resource deployment [12].

As a consequence, new "Edge" types have recently been proposed. Some of them have introduced the concept of VCC [38]. VCC entails creating a CC composed of the vehicle resources not used by the vehicles' On-Board Units (OBUs). VCC resources have already been deployed or are under deployment thanks to the proliferation of new connected and autonomous vehicles. For example, Tesla has equipped its cars with powerful NVIDIA GPUs [39] and the embedded computational resources of these vehicles would be highly valuable in reducing the investment in EC and infrastructure costs if a sufficient number of vehicles is available to satisfy task offloading requests [13].

In a preliminary work [14], not detailed in this thesis, we investigated whether VCC can effectively replace EC when adopting Wave-IEEE 802.11p communication technology between the vehicles and infrastructure. This study identified several parameters that enable VCC as an alternative to EC (such as traffic load, vehicle density, and VCC computational resources) and showed that VCC can satisfy low-latency applications across a wide range of these parameters.

To meet modern-day real-time vehicle communication needs such as AD, 5G represents a suitable alternative to IEEE 802.11p as it brings higher performance in terms of latency, capacity, and scalability. This is further supported by industry trends and regulatory decisions, such as the Federal Communications Commission (FCC) in the USA [40], reallocating spectrum to 5G-based technologies in recognition of its potential to enhance automotive safety and connectivity.

Therefore, even though Wave-IEEE 802.11p as a vehicle to infrastructure protocol has been largely adopted in literature, we rely in this work on 5G communications due to its high-performance potential in terms of coverage and achieved data rate and latency. This study is the first to investigate the conditions under which VCC can serve as a viable alternative to EC for task offloading in 5G networks. In this regard, this work presents:

- An extensive simulation campaign in which the mobility is modeled in SUMO and the 5G communication in NS3. The impact of the following factors is analyzed: vehicle density, computational power, task workload, and vehicle mobility in urban and non-urban scenarios. The criteria considered are latency requirements and task failure rate.
- A cost analysis model that shows cost savings offered by VCC over EC, making VCC attractive for network operators willing to provide offloading services.

The software used in this work is open source and provided on GitHub.[1]

[1] https://github.com/patan3saro/ComputationEcosystem_in_5G.git

4.2 . Related Work

Task offloading allows devices, such as smartphones or OBUs, to execute tasks in external nodes. This reduces the consumption of the device's battery or to execute complex tasks that would require a large amount of computation or special hardware that may not be available in the device itself [41]. Task offloading is generally performed on the Cloud. However, some task applications, e.g., online gaming and AD, may not tolerate the latency to reach the Cloud [42]. To execute tasks in the proximity of the device, EC has been extensively studied: by executing tasks in Edge nodes, such as BS or Wi-Fi APs, the latency can be greatly reduced. VCC could contribute to real-world task offloading as an alternative to EC. A vast amount of research has been devoted to task offloading strategies when tasks are offloaded from vehicles to Edge nodes to perform low latency AD functions [43, 44].

However, EC deployment at the level of BS and APs is still nonexistent due to its high cost of deployment [12]. This is why we examine in this work whether offloading on vehicles can replace offloading on Edge nodes.

Other works investigate solutions where vehicles share computational resources among themselves. In [45, 46], the authors consider two types of vehicles: Task Vehicles (TaVs) and Service Vehicles (SeVs). When the TaVs require additional resources to complete their tasks, the SeVs make their resources available to help them complete their workloads. Although those works have shown that VCC can behave as an Edge on the road, they suffer from inherent limitations. Paper [46] demonstrated that task allocation takes numerous seconds against the low-latency application requirements [47]. Paper [45] reported lower latency in the tens of milliseconds but introduced decision overhead on vehicles, which suffer from limited communication coverage (200 m). Finally, works like [48] and [49], authors proposed methods for resource allocation in VCC, by integrating Edge of Cloud infrastructure with vehicular ad hoc networks. Unfortunately, these works suffer from both limited communication range and relatively high latency delays, primarily because they rely on IEEE 802.11p-based communication technologies. In our work, 5G overcomes these issues of offloading decision overhead by providing broader coverage.

In [50], the authors focus on road safety-related offloading in areas with poor Edge node coverage, resulting in limited computational resources but at the expense of non-negligible computational overhead of the adopted node selection strategy. Moreover, unlike our work, which focuses on real-world urban scenarios (using the SUMO tool), the authors considered a simple mobility model assuming constant speed on a highway. The findings of [50] do not directly apply to our work, as this paper considers EC and vehicles as complementary and used together. In contrast, our work focuses on fully replacing EC, avoiding the expenses related to its deployment and maintenance.

The literature also presents architectures composed of EC, VCC, and CC, sharing their advantages. In [51], the authors consider a three-tier model comprising VCC, EC, and CC in which the three components operate in the same hierarchy. They propose a collaborative approach for computation offloading using vehicles with idle resources as Fog User Equipment (F-UE). However, vehicle mobility and communication channels are assumed to be semi-static, whereas we consider realistic time-varying channels and user mobility.

One of the first papers to introduce the use of a three-tier approach is [13]. The authors suggest an offloading scenario where task processing requests are initiated by user mobile terminals and managed by a Cloudlet connected to a VCC through Wi-Fi APs. End devices can connect to the Cloudlet via a BS using 3G/4G connections or directly through APs. The Cloudlet, which serves as the control center, determines whether a task should be offloaded to the Cloud, the Cloudlet itself, or a VCC. Inspired by this work, we adopt a similar architectural model, as it is described in Sec. 4.3.

Recent research, such as in [51], also relied on an integrated CC, EC, and VCC architecture. The authors aimed to minimize task offloading time for end-users by exploiting the collaborative effort of the three component tiers. They showed that VCC and EC may achieve much lower latency than CC to reduce the task offloading time. However, this work does not address the related high deployment costs of EC when resources must be geographically dense.

In conclusion, to the best of our knowledge, no previous study has clearly established the conditions under which VCC can fully replace EC in a 5G-enabled environment. This is despite extensive research into task offloading in 5G networks, such as the ones applying machine learning to EC task offloading or game theory for optimal strategies in large-scale EC scenarios [52]. Existing studies often view VCC as a complement to EC rather than as a standalone alternative.

Therefore, the objective of this work is not to add yet another strategy to this extensive collection. Instead, the aim is to conservatively analyze how VCC can take advantage of 5G wider coverage, better connectivity, and lower latency to effectively replace EC. We deliberately adopt simple strategies (more motivations on this choice can be found at the beginning of Sec. 4.4) to identify and investigate the parameters that govern the performance of an integrated three-tier Cloud architecture. The target is to analyze and understand the necessary conditions for complete EC replacement, especially in scenarios where the EC deployment may not be economically viable.

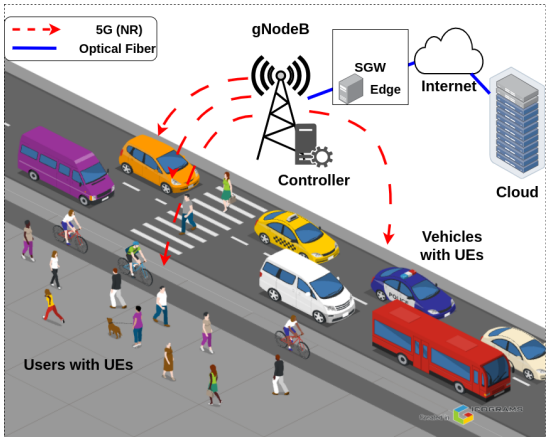


Figure 4.1: *Proposed communication architecture.*

4.3 . System Architecture and Networking

This section introduces the adopted architecture model and the 5G networking with the related parameters. The main objective is to define an architecture that can allow a fair comparison between EC and VCC. The fairness is guaranteed by adopting the same network settings for both computation paradigms.

4.3.1 . System architecture

The system architecture considered in this study is illustrated in Fig. 4.1. It is based on a 5G cellular access technology and includes six fundamental components.

Before introducing the architecture components, we first provide some definitions. The term **task** refers to the computation requested by an end-device user, which includes its execution on an external node and the feedback of the generated result (data or alternative code). A **task computation request** is the initial request sent by an end-user terminal, specifying the requirements for execution. **Task offloading** refers to the entire process of selecting a node, offloading the task, executing it, and returning the results. To maintain clarity, **task** can be used as a synonym for a request, while **task offloading** describes the complete execution process.

a. **End-users** in need of offloading their tasks are identified as 5G User Equipment (UE)s. UEs can be smartphones, smartwatches, laptops, tablets, etc. The UE of end-users is defined as Pedestrian-UE (PUE).

b. A **gNodeB** (5G BS) with 5G NR technology (see 5G-LENA in Sec. 4.6) are macro BS, allowing a wide coverage area serving several moving vehicles. The large coverage allows a longer dwell time for vehicles, which ensures the completion of offloading requests, reducing failures.

c. A **Controller**, implemented in the gNodeB, (i) keeps track of the vehicles available in the area (by collecting their beacons and recording their IP addresses) and (ii) decides the nodes where to offload incoming tasks. The detailed procedures executed in (i) and (ii) are given in Sec. 4.4.

d. An **Edge server** is located in the Serving Gateway (SGW). The SGW is a node at the Edge of the Core Network (CN) , which ensures communication between the UEs and the external network parts. This location of the Edge server follows the European Telecommunications Standards Institute (ETSI) standard for Multi-access Edge Computing (MEC) architecture [53]. The Edge node has specific computational resources (such as CPUs and GPUs) and, if available, can compute offloaded tasks.

e. **Vehicles** are equipped with on-board 5G UEs and computational resources capable of executing tasks but assumed less powerful than the Edge's resources. The UE of vehicles is defined as Vehicle User Equipment (VUE).

f. A **Cloud server** has a high amount of computational resources (they can be assumed to be infinite [54]), which outperforms those available at the Edge server.

The notion that Cloud resources are virtually unlimited is widely supported in the literature [54–56]. Thanks to the vast resource capacity of providers like Amazon or Google, users rarely experience task rejections. Offloading can be motivated by various factors, including energy savings and improvements in Quality of Experience (QoE) for resource-intensive applications. The QoE can be enhanced by reducing latency, which is achievable through EC and Virtual CC .

4.3.2 . Networking

This subsection describes the protocols and networking aspects of the architecture, with the main functions and parameters.

5G infrastructure

We consider a 5G system architecture enabling 5G communication links between devices. The system is composed of a 5G cell (gNodeB or BS) employing an 8×8 trisector antenna. We use the Third Generation Partnership Project (3GPP) Release 15 for simulation purposes, such as in [57]. The channel is modeled at the range of Frequency Range 1 (FR1), i.e., at the sub-6 GHz frequency bands with a 5G-compliant channel frequency bandwidth of 200 MHz. Furthermore, the adopted numerology ($\mu = 2$) allows shorter latency. Lower numerologies provide better coverage but suffer from higher latency

and reduced spectral efficiency. Higher numerologies enable lower latency and better spectral efficiency but have coverage limitations. The latter could reduce the Transmission Time Interval (TTI) too much, increasing packet loss.

Beam-forming is employed to increase the network capacity. The parameters above implement a macro BS that covers relatively distant vehicles in an urban scenario. The path-loss model is the urban macro line of sight (UMa LoS), suitable for urban scenarios. The UEs for end-users and vehicles are modeled similarly using a basic 1x1 array isotropic transmission antenna [58] with a transmission power (23 dBm for UEs) adapted to the case of study (task offloading on vehicles).

Other networking aspects

The PUEs, as the VUEs, use the NR 5G lower layers, integrating UDP/IP in the upper layers protocol stack. The UDP transport protocol is adopted as in [14] to make the communication lighter than TCP. Also, the **Edge server** stack is based on UDP/IP regarding upper layers, and the same above considerations for this choice are valid. The Cloud server is reachable over the Internet, implying higher network latency (35ms [6]) than those for reaching the Edge server. More details are reported in table 4.1.

4.3.3 . Beaconing for tracing vehicle availability

The Controller maintains a list of available vehicles to perform computation tasks. This list is generated based on the beacons sent by the vehicles, which announce their available computation resources. The Controller listens over appropriate multicast addresses (IPv6 or IPv4). Multicast addresses `beacon_address` and `request_to_offload_address` are used by vehicles and UEs to send their beacons and task offloading requests, respectively.

We assume two types of beacons: *periodic* and *aperiodic*. If a vehicle has computing resources available, it broadcasts a periodic beacon at a frequency of f. In our simulations, we set $f = 10$Hz (i.e., one beacon every 100ms). This frequency is consistent with the service requirements specified in 3GPP TS 22.186 [59], which recommends update intervals of 100 ms as a baseline for information exchange between vehicles and infrastructure in enhanced V2X scenarios. Upon receiving such a beacon, the Controller will add the corresponding vehicle to its list. If a vehicle is not available for computation of offloaded tasks (e.g., it is already processing a task), it does not send periodic beacons and hence does not appear in the Controller's list. In addition, when a controller sends a task to a specific vehicle, it removes that vehicle from its list. The vehicle will reappear in the list only after it transmits a new periodic or aperiodic beacon. Additionally, the Controller removes a vehicle from its list after a timeout (set to 500 ms in our simulations) if no new beacons are received. This timeout value is consistent with typical V2X beacon lifetimes

and with the upper bounds for message relevance and freshness in vehicular safety applications [60].

Note that there might be a mismatch between the list of available vehicles in the Controller and the vehicles available in reality. This happens in the following cases:

- A vehicle with available resources just entered the cell at time t, but the next periodic beacon is sent at time $t + \epsilon$, where $\epsilon \in [0, 1/f]$. In this case, in the interval $[t, t + \epsilon]$, the resources on such a vehicle cannot be exploited.
- A vehicle available until time t, just exited the cell at time t. In this case, the Controller will remove such a vehicle from its list after the timeout, i.e., at instant $t + \epsilon'$, where $\epsilon' \in [0, \text{timeout}]$. During interval $[t, t + \epsilon']$, the Controller may still send tasks to that vehicle, however, without obtaining responses. Such tasks are thus doomed to fail (see red bar in Fig. 4.7).

4.4 . Offloading Strategies

We consider in this work two basic offloading strategies, namely: *EC-First* and *VCCFirst*. The motivations behind such choices rather than more advanced strategies are provided at the end of this section.

Let us assume that an end-user device (PUE) requires an external node to compute a task related to a local application. The PUE sends an offloading request to a Controller, which is co-located with the BS. The Controller is responsible for forwarding the task to the EC node or a VCC node. If the Controller finds that no computational resources are available in EC or VCC, CC is selected as a backup option, as it is assumed to have infinite available resources [54]. The node that computes the task sends the request to the dedicated queue if it exists, waiting for execution before returning the result to the sender.

It is worth noting that any issues related to privacy, energy consumption, and incentives for car owners to accept offloaded tasks in their vehicles are outside the scope of this work and are addressed in the literature, such as [46, 61].

Below, we provide definitions for the two offloading strategies:

- *ECFirst*: This scenario involves EC and CC for task offloading, and CC is considered a backup solution for managing task overflow. To reduce latency, the Controller prioritizes transferring tasks to the EC and resorts to sending tasks to CC only when the EC is operating at full capacity, i.e., upon saturation of the tasks queue of the Edge node. The queue in EC is handled with a First-In-First-Out (FIFO) policy.

- *VCCFirst*: This strategy includes only VCC and CC, suppressing the need for EC. Whenever the Controller receives a task to offload, it checks for the available vehicles in its internal list (maintained as explained in Sec. 4.3.3). Among the vehicles on the list with sufficient available resources, one is randomly selected for the task processing. Otherwise, if no vehicle is available for computation, the task is offloaded to the CC. From the Controller's point of view, vehicles are all equivalent since they are assumed to have the same resources and are at a reasonable distance from the gNB, ensuring coverage and feasibility of offloading.

  If *VCCFirst* is adopted, vehicles send beacons to announce their presence and their available resource. To keep a conservative assessment, this work considers that vehicles can handle only one task at a time. Therefore, a vehicle already serving a task offloading request rejects additional requests. This conservative assumption also guarantees that task offloading does not exhaust vehicle resources, whose main use should be driving safety and navigation.

In both ECFirst and VCCFirst strategies, CC is used as a backup solution. This means that the Controller sends to the Cloud all "overflow tasks," i.e., tasks that arrive when the queue in the EC is full (in the ECFirst strategy) or if no available vehicle with sufficient resources is found (in the VCCFirst strategy). According to the assumption that the Cloud has infinite capacity (Sec. 4.3.1), the Cloud always has available resources and will never reject a task.

Note that the strategies presented here are intentionally basic. Numerous optimization strategies have been proposed in the literature to perform task allocation in the VCC. The selection of vehicles could be done based on speed, vicinity to the edge node, or predicted trajectories instead of randomly selecting them. One could be tempted to adopt all such advanced strategies for the purpose of this work. However, by doing so, findings will be tightly coupled to the particular strategy picked. This work, instead, aims to provide more *conservative* findings by deliberately adopting basic strategies. Indeed, if VCC can replace EC already "in the worst case," i.e., with such basic strategies, one can confidently conclude that such replacement will also be possible with more advanced strategies, as they would obviously improve the performance of VCC. The performances shown in this work are thus intended as pessimistic estimates of what one could achieve with their preferred state-of-the-art task allocation and vehicle selection algorithms.

It is worth noting that any strategy composed of VCC and EC is explicitly omitted. Indeed, using a Vehicular-EC strategy is not indicative since it behaves similarly to VCCFirst with EC instead of CC. However, to prove that the substitution of ECby VCC is feasible, the standalone paradigm (VCC or EC) with CC, used as a backup for managing the task overflow, is investigated.

4.5 . Task and offloading time models

In this section, we present the task and offloading time models related to the different architecture components, namely CC, EC, and VCC.

4.5.1 . Task Model

A task i is defined as a tuple (W^i, S^i, R^i), where W^i represents the workload (in Million Instructions (MI)), indicating the number of instructions required to execute task i. S^i denotes the input task size (in [KB]), and R^i represents the task processing result expressed in terms of the amount of data to be returned to the end user.

The offloading process unfolds as follows: once a given PUE decides to offload a task, it forwards the task with the corresponding tuple information to the BS. Then, the Controller within the BS selects the offloading destination (EC, VCC, or CC) based on the chosen strategy. In the selected destination node, the task is queued (only for EC) and executed, and the resulting output is fed back to the PUE.

4.5.2 . Offloading time models

The offloading time models include the fundamental components of all the offloading procedures. The overhead related to task preparation is not considered part of the offloading procedure, as it occurs at the PUE device following the offloading decision procedure [62]. The preparation time, if present, could be neglected, as the transmission time is typically much larger. Moreover, as the security considerations are out of the scope of this work, we neglected the time overhead inherent to the adopted security protocol. Finally, the protocol stack overhead is implicitly included in the model, as the NS3 simulations provide results including this overhead.

Cloud offloading time

The main assumption here is that the CC disposes of infinite computational resources. Hence, every offloading request is instantly elaborated at the CC when received without generating any queuing time. T^i_{CC} is the offloading time of task i to the CC. It includes the time for the task computation request to be sent, processed, and sent back through the result packet to the PUE. It is composed of the following time components (in [seconds]):

- The first component is the *Uplink time*, noted as $T^i_{\text{up, PUE-gNB}}$, which is the time required for the task to reach the gNodeB through 5G NR technology from the PUE.

- The *Uplink CN time*, noted $T^i_{\text{up, CN}}$. It is time for the offloading request to pass from the gNodeB to the SGW (i.e., to reach the Internet). This time is assumed constant (i.e., fixed to 2ms in the simulations [63]).
- Another component is the Internet latency, noted as $T^i_{\text{up, Internet}}$. It is the time delay that the offloading request takes to transfer across the Internet and reach the Cloud Server.
- After the overall *uplink time part,* there is the elaboration time of the request. The *Elaboration time* T^i_{elab} is as follows: $T^i_{\text{elab}} = W^i / C_{\text{CC}}$, where C_{CC} is the computational capacity of the Cloud Server, expressed in *Million Instructions Per Second (MIPS)*. W^i is the task workload measured in MI.
- Finally, the *Downlink time* is defined as the time required for the result to be sent back to the PUE who requested the task offloading. Similarly to the uplink time, it is composed by $T^i_{\text{down, Internet}}$ the latency from CC to the SGW, $T^i_{\text{down, CN}}$ the delay experienced from SGW to the gNB, and $T^i_{\text{down, gNB-PUE}}$ the transfer time from the gNodeB to the PUE who initiated for the offloading procedure.

Finally, the total task offloading time (a.k.a. task response time) of task i to the CC, $T^i_{\text{offloading, CC}}$, can be expressed as follows:

$$T^i_{\text{offloading, CC}} = T^i_{\text{up, PUE-gNB}} + T^i_{\text{up, CN}} + T^i_{\text{up, Internet}} + T^i_{\text{elab}} + T^i_{\text{down, Internet}} + T^i_{\text{down, CN}} + T^i_{\text{up, gNB-PUE}}. \quad (4.1)$$

Edge offloading time

As stated previously in Sec. 4.3.1, the Edge node is implemented at the SGW (see Fig. 4.1). The ECparadigm is composed of only one computational resource (one CPU or GPU), and its capacity, noted as C_{EC}, is measured in MIPS. If the computational resources are occupied, the ECemploys a FIFO queue to manage offloading requests, ensuring fair task processing.

The task offloading time to the ECnode, $T^i_{\text{offloading, EC}}$, is given as:

$$T^i_{\text{offloading, EC}} = T^i_{\text{up, PUE-gNB}} + T^i_{\text{up, CN}} + T^i_{\text{queue}} + T^i_{\text{elab}} + T^i_{\text{down, CN}} + T^i_{\text{down, gNB-PUE}} \quad (4.2)$$

where T^i_{queue} is the time that task i spends in the queue before the elaboration. $T^i_{\text{elab}} = \frac{W^i}{C_{\text{EC}}}$ is the elaboration time, and W^i is the task workload measured in MI.

Vehicular offloading time

Let C_{VCC} be the computational capacity of a single vehicle in the VCC. For simplicity, in this work, all the vehicles are assumed to have the same computational resources since the objective is to avoid case-specific dependency bias. Moreover, we assume that the quantity of resources that the vehicle makes available for external tasks is considered as a portion of idle resources (i.e., not used for internal tasks such as in-vehicle driving tasks). Since the resources are assumed to be logically separated, external tasks do not impact internal tasks. In the case the resources are not homogeneous, the contribution of the elaboration time in the results would be proportional to this availability. Note that considering only a fraction of the actual vehicle resources is a conservative approach. In practice, processing time can be greatly reduced by exploiting more on-board power, as modern GPUs like the NVIDIA Tesla V100 reach 100 TeraFLOPS [10]. We will later show through simulations (in Sec. 4.6) that the impact of increasing computational resources is no more beneficial to the offloading time after a certain point. This happens since the communication time is, in this case, much longer than computation. The effect of vehicle resources heterogeneity will be investigated in our future works.

We assume that no queuing time is considered for vehicles since all vehicles can serve only one task at a time. The offloading time of a given task i offloaded to a VUE is given by:

$$T^i_{\text{offloading, VCC}} = T^i_{\text{up, PUE-to-gNB}} + T^i_{\text{up, gNB-VUE}} + T^i_{\text{elab}} + T^i_{\text{down, VUE-gNB}} + T^i_{\text{down, gNB-PUE}} \quad (4.3)$$

where:

- $T^i_{\text{up, PUE-gNB}}$ is the time required by the offloaded request to pass from the PUE end-device to the gNodeB. The opposite time for the *Downlink phase* is $T^i_{\text{down, gNB-PUE}}$.
- $T^i_{\text{up, gNB-VUE}}$ is the time required to send the request to a vehicle from the gNodeB. The opposite time for the *Downlink phase* is $T^i_{\text{down, VUE-gNB}}$.
- $T^i_{\text{elab}} = \frac{W^i}{C_{\text{VCC}}}$ is the elaboration time of task i, where W^i is the task workload measured in MI.

4.6 . Performance evaluation

This section shows that VCC can be a powerful alternative to EC for low-latency applications. The extensive simulations identify the various conditions

crucial for this substitution, including user types, task workload, and VCC capacity in terms of the density of vehicles and onboard computation resources. The approach directs all task offloading requests through the gNodeB before reaching the vehicles using 5G links (Sec. 4.3).

4.6.1 . Simulation and Network Environment

NS3 is adopted to implement the communication environment of the system shown in Sec.4.3. The main simulation and network parameters are listed in Table 4.1. While being compliant with the 5G standard, several network parameters (such as bandwidth, transmission power, and operational frequency) are chosen empirically to ensure coherence with the requirements for latency and coverage. The simulation of the communication technology is provided by 5G-LENA [71], an NS3 module used for simulating the New Radio (NR) part of 5G (3GPP release 15 [57]). The module 5G-LENA[2] is supported by several research projects and a large and active community that ensures frequent updates and new releases.

The UEs of end-users and vehicles use the same physical channel with identical configuration parameters such as the bandwidth, carrier, and transmission power. The adopted urban path-loss model (Table 4.1) is useful to create large coverage. All the network model parameters are reported in Table 4.1. They follow the rationale presented in Sec. 4.3.2.

For the sake of simplicity, limited-size tasks are considered, and the default task size is 4KB, inspired by reference [14]. It is worth noting that a given task, even if it has a small size in KB, may be computation-intensive, i.e., it requires many instructions to be executed and, hence, a high workload. For instance, a simple "for loop" could have a size of less than 1 KB but could require a heavy workload in terms of Million Instruction (MI).

Vehicle mobility is simulated by the SUMO mobility simulator [72]. The mobility of the vehicles is based on the *Manhattan mobility model* [73], and all the obtained results are averaged over 9 simulation runs to ensure accurate values.

The considered scenario is the **total coverage scenario**. It is based on the Manhattan mobility model, implementing a single rectangular road composed of two lanes in the two possible directions of the road. The rectangle shape has length and width (respectively, on x and y axes) of 600m and 50m, respectively. In this mobility scenario, the BS coverage is total. It is positioned at the center of the scenario with coordinates $(x, y, z) = (300, 25, 30)$m, where z is the height of the BS (Table 4.1). This scenario allows vehicles to assume,

[2] 5G-Lena is supported by the *Centre Tecnològic de Telecomunicacions de Catalunya* (CTTC) a research institute in Spain focused on telecom and intelligent transportation tech.

Parameter	Value
Scenario	
Cloud nodes	1
Number of Edge nodes	1 (in the ECFirst scenario) or 0 (in the VCCFirst scenario)
Number of vehicles	40 by default (up to 60)
Number of end-users	8 [64] by default, (up to 100)
Simulation duration	120 seconds
Cloud computation resources	$C_{\text{CC}} = 2356230$ MIPS [65], ∞ processors
Edge computation resources	$C_{\text{EC}} = 749070$ MIPS [66], 1 processor
VCC computation resources	$C_{\text{VCC}} = 71120$ MIPS [67], 1 processor per vehicle
Task workload	$C_u = 500$ MI [68]
Task size	$D_u = 4000$ bytes [13]
Max queue length at EC	100 packets (FIFO policy)
End-user offloading request rate	A request every 200ms [69]
Vehicle average speed	downtown traffic 13.1km/h [70]
Used seeds for SUMO scenarios (total 9)	0,1,2,3,4,6,7,8,9
Communication	
gNodeB Height	30m [63]
UE Height	1.5m [63] (the average height of a device of a pedestrian)
CN latency	$T^i_{up,\text{CN}} = T^i_{down,\text{CN}} = 2$ milliseconds [63]
Internet latency	$T^i_{up,\text{Internet}} = T^i_{down,\text{Internet}} = 35$ milliseconds [6]
Operational frequency	6GHz (FR1 [57])
Numerology	$\mu = 2$ §4.6.1
Tx Power of gNodeB	20dBm §4.6.1
Tx Power of UE	23dBm §4.6.1
Bandwidth	200MHz §4.6.1
Antenna array gNB (three sectors)	8x8 [58]
Antenna array UE (isotropic)	1x1 [58]
Pathloss model	UMa LoS [58]

Table 4.1: Simulation parameters.

on average, the maximum speed in the long line part of the road, making it useful for studying the effect of speed in experiments.

Class name	Requirement	Example of applications
Extremely Low Latency (LL^{++})	$\leq$ 16ms	AR [47]
Very Low Latency (LL^{+})	$\leq$ 100ms	AR [47]
Low Latency (LL)	$\leq$ 500ms	Antivirus [75]

Table 4.2: Application classes and related latency requirements.

Regarding pedestrians, they remain stationary during task offloading, which seems reasonable given its short period (0.5s). Even a "fast" pedestrian would likely move less than 1 meter within this time frame. The selected default processors are the *AMD Ryzen Threadripper 3990X (64 cores)* for the CC [65, 74], *AMD Ryzen 9 3950X (16-core)* for the EC [66, 74], and *ARM Cortex A73 (4-core)* for each vehicle [14, 67]. These processor choices are made based on their market availability and computing power, as outlined in the existing literature on the computational capabilities of CC, EC, and VCC (embedded) [14, 68].

The task workload is measured in MI , and it is set at 500 MI. This assignment is typical *object recognition*. The practical implication of this choice is that a 500MI workload is representative of the variety of existing workloads. In fact, this value is classified as an average workload ("middle task"), as reported in [68].

The acquired offloading times are compared to three reference application classes and their corresponding latency thresholds (Table 4.2). Those classes can cover most of the expected latency requirements in 5G networks. For instance, a user can be required to use AR with object recognition and other related tasks. This is the case of extremely low and very low requirements. However, if an end-user needs some offloading of his antivirus application, the requirements for task completion can be less strict. We provide in the following an analysis of the impact of several parameters.

4.6.2 . Impact of the number of end-users

In figures 4.2, 4.4a, and 4.6, the three horizontal lines, they represent the latency requirements of the considered application classes (Table 4.2).

In Fig. 4.2, the offloading time (task completion time) is presented as a function of the number of users associated with the respective PUEs for both the adopted strategies, VCCFirst and ECFirst. The impact of the number of users is shown when considering two request rates per user, namely 5 and 100 requests/sec. As expected, by increasing the number of users, the total request rate increases proportionally. The default user offloading request rate (5 requests/sec, Table 4.1) is related to the considered task of object recognition [68]. The objective of this setting is to have continuous activity on the

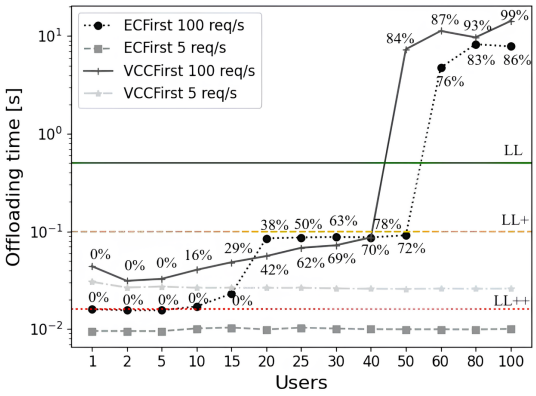


Figure 4.2: *Average offloading time (T^i_{EC}, T^i_{VCC}) of ECFirst and VCCFirst scenarios for different request rates. The reported percentage refers to the number of requests satisfied by CC.*

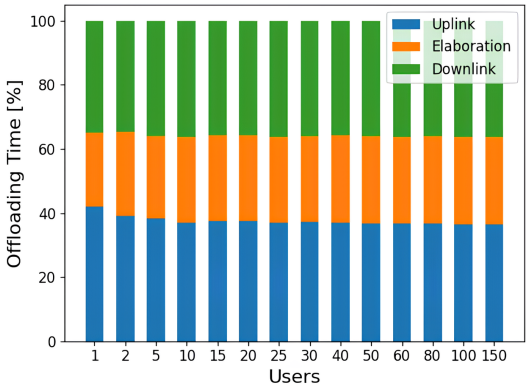


Figure 4.3: *Contribution of the different components of the offloading time in the VCCFirst scenario for the default request rate (percentage).*

network to measure the performance of traffic offloading intense cases. For the default task rate, both strategies can manage 100 users and more, with no need to be helped by CC. This indicates that the adopted resources and the network settings (Table 4.1) are sufficient for this scenario, since 5G brings large network capacity. The figure shows, just for the curves related to the default values of the request rate, that the offloading time in EC is always lower than VCC. The results indicate that VCC can replace EC from 1 to 100 users for LL$^+$ and LL application requirements. *Extremely low latency*, LL^{++}, applications cannot be supported by VCC, but they can be satisfied only by EC. The figure also shows that even numerous users do not influence the offloading time because the network capacity is large. Comparing these results with the previous work in [14], for the same type of analysis, the degradation of performance is very visible. This is an advantage of 5G compared to Wi-Fi (IEEE

802.11p) for vehicular applications. Under these conditions, the contribution of the CC is negligible. The underlying phenomenon is the increase in channel utilization. The rationale behind this is that the number of offloaded tasks per second increases with the number of users. Hence, for 1 user, there are, on average, 5 requests/sec, and for 100 users, this number is 500 requests/sec. The objective here was to stress the network through higher request rates to show its limits. The 5G network loses in performance because of a higher request rate (an average of 10000 requests/sec for 100 users). This shows clearly that even 5G links could reach their capacity limitation in both strategies. For more clarity, an indication of the task overflow quantity is presented in the figure. The task overflow is managed by CC and a percentage of the requests that are sent to the CC is provided on the curve. The figure shows that for higher loads, the CC takes over since both the EC and VCC have too many requests to manage when alone.

Fig.4.3 shows the contribution, in percentage, of the uplink, elaboration, and downlink time over the entire offloading time. The uplink and downlink parts offer similar contributions. The slightly minor contribution in terms of percentage over the offloading time is given by the elaboration time. This indicates that the number of adopted resources in the VCC is well-chosen after the parameters are fine-tuned.

This analysis concludes that the model exhibits resilience to high task loads and possesses the capability to manage tasks under high traffic conditions. This happens even for high numbers of users in only one gNB coverage area. Furthermore, both EC and VCC meet the established low latency requirements. While EC demonstrates superior performance for extremely low latency (approximately 10 ms), VCC displays potential for serving as a viable replacement for EC at around 30 ms. These results are supported by the ANOVA (Analysis of Variance) test presented in the following. ANOVA test is a statistical analysis. The analysis in this study is conducted to compare the impact of parameters between the two strategies, VCCFirst and ECFirst. The F-statistic in the table 4.3 indicates the ratio between the variance resulting from the independent variable (users, workload) and the variance within the groups of offloading time observed values. This result indicates when a factor parameter has a significant effect. The P-mind denotes that the possibility of achieving the results observed is true, following at the same time the zero hypothesis. Low values suggest statistical importance. The class-sum considers the total variation in the data. The degree of freedom (DF), instead, indicates the number of independent comparisons that can be created, i.e., if the x-axis has $n = 10$ values, the possible comparisons are 9 ($n - 1$). In conclusion, the residual represents the part of the unexplained variance, which is not part of the variability responsible for the model. The ANOVA test shows the findings mentioned above in the case of user number variation.

The correlation between the two strategy results is weak (F = 0.0045, p = 1.0) for the default request rate (5 requests/s) for strategy results. However, if the system is overloaded (100 requests/s), the sensitivity of the offloading time to the number of users becomes higher (F = 6.311177, p = 0.001174) due to the high channel utilization.

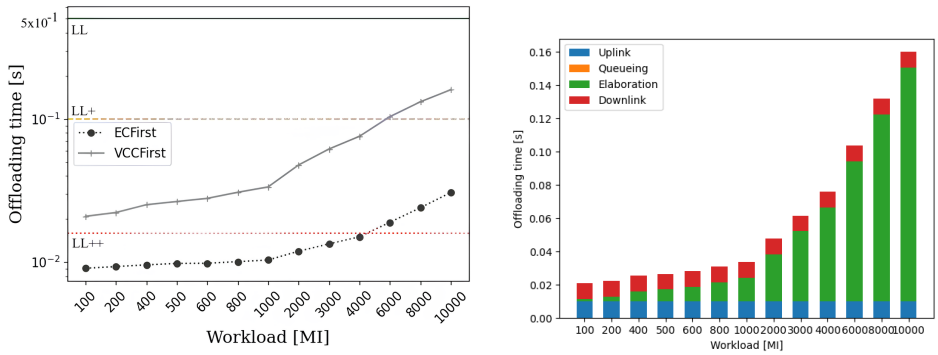
	Parameter	Comment	sum_sq	df	F	PR(>F)
C(Users)	Users(low rate)	No significant effect	0.000008	12.0	0.0045	1.0
Residual	-	-	0.001814	13.0	-	-
C(Users)	Users(high rate)	Significant effect	401.167688	12.0	6.311177	0.001174
Residual	-	-	68.861688	13.0	-	-
C(Workload)	Workload	No significant effect	0.016559	12.0	0.799279	0.647773
Residual	-	-	0.022444	13.0	-	-

Table 4.3: ANOVA results for users (request rate is firstly 5 requests/second then 100 requests/second) and workload and relative significance.

4.6.3 . Impact of task workload

This subsection presents the effect of the workload on the offloading time. An important impact on the offloading time, in particular, the computation time, is expected.

Fig. 4.4a shows the offloading time as a function of the task workload for each strategy.



(a) *ECFirst and VCCFirst offloading time.*

(b) *The VCC offloading time for the VCCFirst strategy.*

Figure 4.4: *Comparison of offloading times on workload variation.*

According to [68], the workload can be divided into three categories based on the task complexity, starting from 100 MI to 9784 MI. The task workload is fixed to 500 MI in the results of Fig. 4.2 and Fig. 4.6. The application related to this workload is the *object recognition* [68]. Figure 4.4a shows the offloading time for both strategies. The results strongly suggest that the EC can handle all the possible task workloads and stay within the deadline of 16 ms until 4000

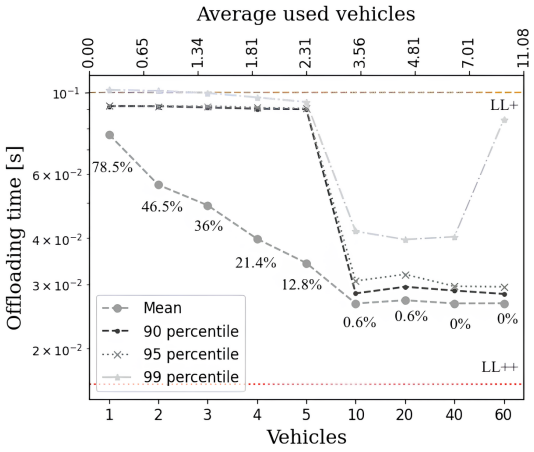


Figure 4.5: *Offloading time of VCCFirst scenario. Impact of the number of vehicles used vehicles, and vehicles in VCC. The reported percentage refers to the number of requests satisfied by CC. Several curves indicate the 90th, 95th, and 99th percentiles of all the offloading values over the simulation.*

MI. It can serve the remainder of the requirements after this value. This is not the case with *VCCFirst*.

The VCC meets LL$^+$ and LL requirements from 100 MI to 6000 MI. After 6000 MI, the VCC can fulfill the LL requirements, while the EC can still satisfy LL$^+$ and LL. This happens without relying on CC. Fig. 4.4b presents the different components of the offloading time. As expected, the elaboration time for the VCC increases proportionally to the workload. The conclusion from these results is that for all the presented task workloads, low latency is satisfied by both VCC and EC, which means that VCC can replace EC for this analysis. The ANOVA test presented in Table 4.3 shows that the tested workload range has no significant effect on the performance of both strategies.

4.6.4 . Impact on the density of vehicles

In this subsection, results are presented only for VCCFirst. Including EC in the analysis would be redundant, as the ECFirst strategy excludes vehicles altogether, making car density irrelevant to the results. In Fig. 4.5, the bottom x-axis represents all the simulated vehicles in the scenario, all evolving under the coverage of the BS. This means that on the bottom x-axis all the vehicles compose the VCC since they are all in BS's coverage area. The top x-axis shows the mean number of vehicles in the VCC that is used for task offloading, averaged over the simulation time and simulation runs (Table 4.1). This set corresponds to the vehicles from which the Controller has received fresh beacons, saying that the vehicular resources are available and exploited. The

number of vehicles used in the VCC is computed as the average, over all simulated offloaded tasks, of the vehicles that participated in each task offloading. As expected, this average increases with the number of vehicles. At the same time, the usage of CC, expressed by the percentage in the graph, diminishes. These results indicate that, with enough resources in the VCC, the response time of tasks improves, and VCC allows better performance regarding cases of great usage of CC. The CC use begins to be negligible from 10 vehicles in the scenario, and reduces significantly with the increasing VCC capacity.

In the same figure, the curves correspond to the 90th, 95th, and 99th percentiles. These curves are farther from the average offloading time values. In fact, in CC scenarios, when the variability of the offloading times is large, higher values correspond to these curves. As soon as the usage of VCC increases, these curves get closer to the average, showing reduced variability.

Another significant observation to note is that the 99th percentile is pretty high compared to the average, something one would expect given that CC is used very rarely, that there are outliers in offloading times. The percentiles presented here help highlight the extreme situations and offer a glimpse into the range of system performance and stability. This allows for the prevention of worst-case scenarios.

This analysis shows that independently of the vehicle number within the tested range, the VCCFirst strategy is sufficient to ensure the LL^+ latency. This is because even for low vehicle density, the Cloud can satisfy the overflow of requests under 100ms. However, when the number of vehicles is sufficient, the offloading time stabilizes at lower values, ensuring a better QoS. The latter can be explained by the proximity of vehicles to end-users, compared to CC, leading to shorter communication time.

4.6.5 . Impact of the computational resources deployed into vehicles

This subsection presents how the variation of resources on vehicles impacts the offloading time.

In Fig. 4.6, a CPU with capacity $C_{\text{VCC}} = 71120$ MIPS is considered, which is the *ARM Cortex A73*, and this is compared to Raspberry Pi 3, Raspberry Pi 4, Raspberry Pi 5 CPUs [76, 77]. The x-axis values are $C_{\text{VCC}} \cdot (1/128, 1/64, 1/32, 1/16, 1/8, 1/2, 1, 2, 3)$. The analysis of the increase in the resources reveals that at a certain point, more resources are very slightly beneficial to the performance because the quality ratio, price, and number of resources are no longer convenient. The figures show that by increasing the vehicle resources further (beyond the ARM Cortex A73 capacity), the benefits of offloading time would not be significant. Even though elaboration time tends to be zero, communication time (uplink and downlink) has a larger impact on the overall offloading time. The CC is never used, which means that the adopted CPU is large enough to satisfy the offloading.

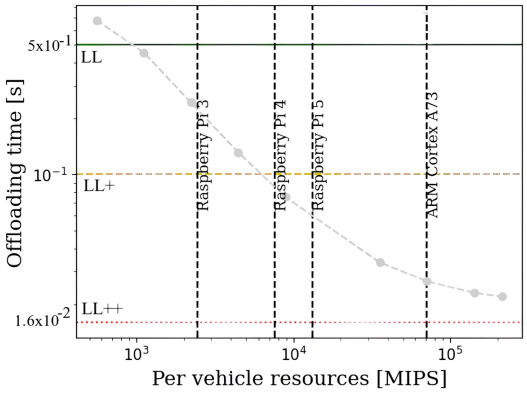


Figure 4.6: *Offloading time of VCCFirst scenario. Impact of computational capacity C_{VCC} of each vehicle.*

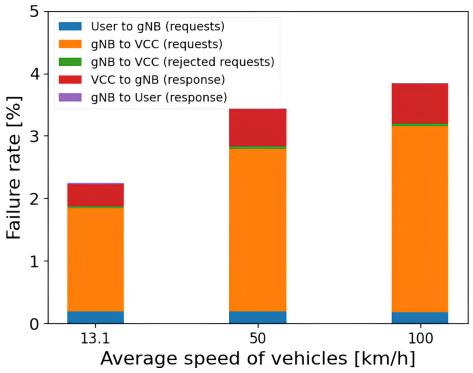


Figure 4.7: *Failure rate in percentage of the total offloading requests per diverse speeds.*

This analysis concludes that the adopted computational power makes VCC capable of satisfying LL^{+} and LL and very close to LL^{++} requirements. The latter strengthens the advantage of VCC as a replacement for EC.

4.6.6 . Impact of the speed

In this part, a scenario with partial coverage of the gNB is adopted. The x-axis length of the SUMO scenario is twice the length of the rectangle of the **total coverage scenario**. The extended rectangular simulation area (1200m x 50m) provides sufficient road length for vehicles to accelerate and reach higher speeds, allowing us to test performance under various mobility conditions.

Figure 4.7 shows the failure rate for the studied speeds using as default parameters the ones in Table 4.1.

The adopted values of speed are $\{13.1, 50, 100\}$km/h. The figure distinguishes the percentage of failure calculated regarding the total number of requests for offloading in the simulation.

The first observation is that the failure rate increases with speed. The first part of communication (user to gNB) is constant because the requests in the network do not change numbers. The second part of the failure is related to the failed requests related to the communication from the gNB to the VCC. This means that in this part of the communication, mobility (speed in this case) is crucial for success in the offload. After that, there is very little rejection failure in vehicles, which means that vehicles are occupied when they receive a request. This occurs because the Controller's list of available vehicles may become inconsistent and fail to reflect the actual situation, due to the delay between the last update and the effective use of resources. Observe that increasing the signaling of the vehicles' beacons to reduce rejection failure is not needed. Firstly, because this value is negligible, and secondly, because it corresponds to a realistic phenomenon. The last part of the failure (VCC to gNB) is due to the response after taking in charge an offloading request and returning the result to the gNB. The failure increases slightly, even if the speed increases considerably. This means that the failure related to an offloading request received by a vehicle when the vehicle goes out of coverage is very low. The last part of failure (gNB to the user) is negligible, and it is related to the normal function of the network, i.e., it depends on the error model of the 5G network. Another reason is that the distribution (the load of the network) is less in the response part of the task offloading. This is because of the offloading request lost in the previous steps of the communication, and the elaboration of tasks in one node is a part that makes the output slower than the input.

The underlying effect is due to several factors. Firstly, vehicle velocity impacts the channel quality in both uplink and downlink, resulting in the degradation of communication quality. This leads to higher failure rates, particularly at the gNB coverage border, which is emphasized by interference and noise phenomena. Furthermore, those effects create inconsistencies between the real number of vehicles under coverage available for offloading and the vehicles in the Controller list (see Sec. 3). The inconsistency occurs when vehicles close to the cell border signal their availability while immediately leaving the gNB coverage, affecting the number of failures in both uplink and downlink between gNB and vehicles. This phenomenon is magnified by the vehicles' velocity.

4.7 . Cost discussion

This section presents an evaluation of the economic interest of a NO relying on VCC as an alternative to EC. We intentionally keep the analysis at a high level, since a detailed technical and business cost assessment falls outside the scope of this work and would warrant a separate study. For simplicity, the evaluation is performed on a generic cell.

4.7.1 . CAPEX

The part of the cost related to the equipment is the CAPEX . The NO bears the CAPEX for the EC, but not for the VCC. Indeed, the NO must buy computational resources and install them at the Edge in the case of EC. In the case of VCC, instead, the NO opportunistically uses the resources already deployed in the vehicles.

Let $L_{\text{EC-CPU}}$ be the lifespan of a computational resource (CPU, GPU, ...) deployed at the Edge node, i.e., the time during which the node is supposed to work before a significant performance degradation and the need for replacement. Let Y denote the investment duration, expressed in *years*, along which the adoption of VCC versus EC is evaluated. The CAPEX incurred by the NO to deploy computational resources at the Edge is:

$$c_{\text{CAPEX-EC}} \geq c_{\text{EC-CPU}} \cdot \left\lceil \frac{Y}{L_{\text{EC-CPU}}} \right\rceil \tag{4.4}$$

The term $c_{\text{EC-CPU}}$ is the market price for a single computational resource as CPU. The factor $\lceil Y/L_{\text{EC-CPU}} \rceil$ indicates how many times the NO is expected to renew the computational capacity in EC over the investment duration, Y. The sign $\geq$ denotes the fact that the cost of deployment is higher than just considering the CPU, as other components, such as RAM, disk, etc., are also needed.

4.7.2 . OPEX

Let us now present the OPEX costs. The NO incurs costs $c_{\text{EC-req}}$ for each task offloaded using the EC. This accounts for the cost of the energy spent in EC for the entire offloading procedure, i.e., communication and computation required by the task. About the VCC, the NO incurs cost $c_{\text{VCC-req}}$ per offloading request. Indeed, a car owner (or the car manufacturer, depending on the business model that will develop around VCC) accepts to make the spare computational capacity available in the car only against payment. Such a payment is performed by the NO for each processed task. Moreover, the NO spends some energy transmitting the task to the car and receiving the response over the air, which also has a cost. The value $c_{\text{VCC-req}}$ is the cost borne by the NO for both payment and energy to send the task to the VCC and receive back the result from a VCC node.

In the case of EC, the OPEX also includes maintenance, which consists of hardware and software repairs to keep the EC functioning correctly. The NO does not instead incur any maintenance cost in the case of VCC, as it does not own any resources. Therefore, the OPEX for EC is:

$$c_{\text{OPEX-EC}} = c_{\text{EC-req}} \cdot R \cdot U \cdot Y \cdot \alpha + c_{\text{EC-main}} \cdot Y \tag{4.5}$$

where R denotes the average task offloading requests of a user per second. The term U is the average number of users present in a generic cell. α is the number of seconds users actively send offloading tasks in one year. It is assumed that users have 15 hours of activity per day. $c_{\text{EC-main}}$ is the annual cost of maintenance of an EC node.

Instead, the OPEX born by the NO related to the VCC is just how much the NO pays to send the offloading requests to car owners (or manufacturers:)

$$c_{\text{OPEX-VCC}} = c_{\text{VCC-req}} \cdot R \cdot U \cdot Y \cdot \alpha. \tag{4.6}$$

$$\begin{aligned} c_{\text{EC}} - c_{\text{VCC}} &= c_{\text{CAPEX-EC}} + c_{\text{OPEX-EC}} - (0 + c_{\text{OPEX-VCC}}) \\ &= c_{\text{CAPEX-EC}} + c_{\text{OPEX-EC}} - c_{\text{OPEX-VCC}} \\ &\overset{\text{Eqn. (4.4), (4.5), (4.6)}}{\geq} c_{\text{EC-CPU}} \cdot \left\lceil \frac{Y}{L_{\text{EC-CPU}}} \right\rceil + c_{\text{EC-main}} \cdot Y \\ &\quad + (c_{\text{EC-req}} - c_{\text{VCC-req}}) \cdot R \cdot U \cdot Y \cdot \alpha \end{aligned} \tag{4.7}$$

where c_{EC} and c_{VCC} are the total costs of EC and VCC, respectively, over the investment period, which include both CAPEX and OPEX (as explained before, the CAPEX for VCC is 0).

These savings are positive if

$$c_{\text{VCC-req}} \leq c_{\text{EC-req}} + \beta \tag{4.8}$$

where

$$\beta = \frac{c_{\text{EC-CPU}} \cdot \lceil Y / L_{\text{EC-CPU}} \rceil / Y + c_{\text{EC-main}}}{R \cdot U \cdot \alpha} \tag{4.9}$$

In other words, VCC remains economically more convenient than EC, even in cases where the NO pays the car owner (or manufacturer) a price per request that is larger than the cost per request that it would bear in EC. The term β, is called *VCC bonus*, indicates how much larger the payment can be for the VCC to remain competitive with EC. As expected, this bonus is larger when the cost of installing $c_{\text{EC-CPU}}$ and maintaining $c_{\text{EC-main}}$ resources in EC is larger. This bonus is also high when request rate R, users U offloading, and activity period α are low. This suggests that VCC can be a suitable alternative to EC

in an initial phase when the penetration rate of low-latency offloading applications is low. Before installing EC, the NO could satisfy the requirements of such applications via VCC. EC would then be installed only after a critical mass of users started to require low-latency offloading capabilities.

4.7.3 . Cost numerical results

Variable	Meaning	Value
c_{EC}	Total expenditure in EC	Eqn. (4.7)
c_{VCC}	Total expenditure in VCC	Eqn. (4.7)
$c_{\text{CAPEX-EC}}$	Expenditure of the EC computational equipment	Eqn. (4.4)
$c_{\text{CAPEX-VCC}}$	Expenditure of the VCC computational equipment	0 (4.7)
$c_{\text{OPEX-EC}}$	OPEX in EC	Eqn. (4.5)
$c_{\text{OPEX-VCC}}$	OPEX in VCC	Eqn. (4.6)
$c_{\text{EC-CPU}}$	Cost per one EC CPU	700[$] [78]
$L_{\text{EC-CPU}}$	EC CPU lifespan	3 years [79]
$c_{\text{EC-main}}$	Annual maintenance cost per EC node, assuming one technician maintains 50 nodes per year (cost is 1/50 of technician's salary)	$1368.46[\$] = (68423/50)$[$] [80]
$c_{\text{EC-req}}$	Offloading cost per request in EC	$2 \cdot 10^{-5}$$/request [81]
$c_{\text{VCC-req}}$	Offloading cost per request in VCC	Same as in EC
Y	Investment duration (operational period for EC and VCC)	[1, 3, 5] years
α	The number of seconds users actively send offloading tasks in one year	Active 15 hours per day
R	Request rate per user	5 requests/sec (Table 4.1)
U	Number of users in one day in one 5G cell who offload their tasks	100 (Table 4.1)
β	VCC bonus, indicates how much larger the payment can be for the VCC to remain competitive with EC	Eqn. (4.9)[$/request]

Table 4.4: Adopted values for the cost analysis.

In this subsection, an assessment of the cost is presented. The adopted parameters are reported in Table 4.4, and their values are representative for conducting the cost analysis. The chosen capacity cost for vehicular CPU and Edge CPU is based on market prices available at the time of the study. These values evolve over time as the market does since the prices are subject to variation over time. The proposed analysis reflects the perspective of the main stakeholder and investor, i.e., the NO. This stakeholder owns the internet and communication infrastructures. However, a dedicated and structured framework is required to better analyze the relevant parameters for other stakeholders. Such a model must focus entirely on economic analysis, modeling the benefits and decisions for all stakeholders. This aspect is beyond the scope of this work. Instead, in next chapter, we propose a game-theoretical framework that considers complementary parameters, such as network delay, task deadline, and energy consumption related to the entire offloading process. Such a model would provide a better suitable framework for understanding how investment decisions could be beneficial. The numerical results presented in this study reflect the perspective of the main investor in EC, the NO.

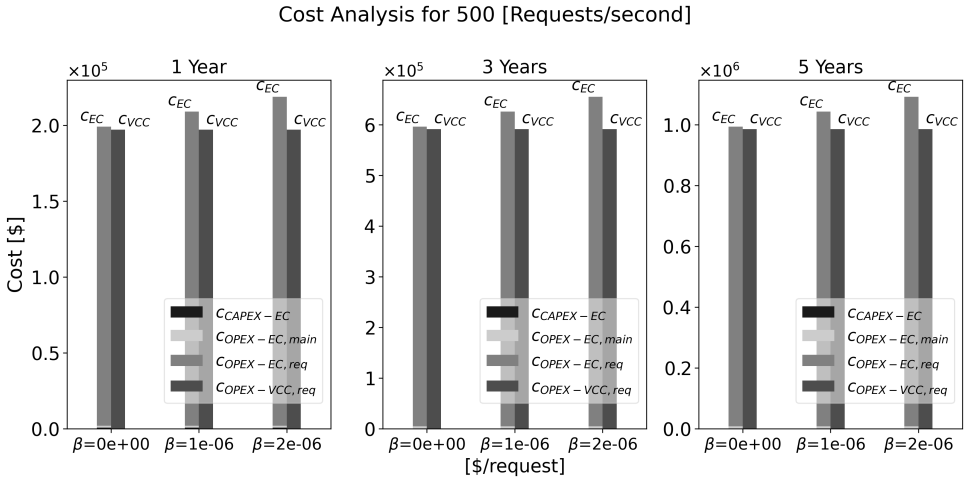


Figure 4.8: *Total cost of the EC compared with VCC for 1, 3 and 5 operational years.*

β	$C_{CAPEX-EC}$ (%)	$C_{EC-main}$ (%)	$C_{OPEX-EC,req}$ (%)	$C_{OPEX-VCC,req}$ (%)
0e+00	0.35	0.69	98.96	100.00
1e-06	0.33	0.65	99.01	100.00
2e-06	0.32	0.63	99.05	100.00

Table 4.5: Distribution of costs for EC and VCC in the first year, expressed in percentage.

Fig. 4.8 shows the total cost of EC and VCC. All the plots represent the total cost of the EC (left bars), with the total cost of the VCC (right bars). On the x-axis, three values of β are provided. According to the Eqn. (4.8), the savings obtained by the use of VCC and CC are positive if the β is positive or null. This indicates that VCC is more profitable. In these figures, the conservative case ($\beta = 0$) is represented. In fact, the cost per request of VCC is $c_{\text{VCC}} = c_{\text{EC-req}}$ as reported in table 4.4. Then, Eqn. (4.8) becomes: $\beta \geq 0$. We observe that choosing $\beta = 0$ does not imply the absence of Edge deployment or maintenance costs. Instead, by convention, $\beta = 0$ indicates that the VCC has opted renounce to its bonus.

The findings support the theory in Eqn. (4.8). In fact, for at least $\beta = 0$ every year, the savings are positive. Indeed, the gap between EC and VCC is precisely the sum of the maintenance cost plus the CAPEX . Hence, the findings confirm that the real barrier of the EC deployment is related to the initial costs, which is evident even in the worst conditions, i.e., $\beta = 0$ and $c_{\text{VCC}} = c_{\text{EC-req}}$. Furthermore, in table 4.5, the influence of the $c_{\text{CAPEX-EC}}$ and $c_{\text{EC-main}}$ result weak, while the OPEX related to requests for CC and VCC results in almost all the expenditure. Even though the findings show that deployment and maintenance are limited to the given parameters, if the NO has to invest in hundreds or thousands of nodes, the influence of the cost of deployment would be non-negligible. The percentage of EC would remain the same. However, the Fig. 4.9 and Table 4.6 show that using VCC is the best choice even in *unfavorable scenarios*, such as poorly planned EC investments or changing user needs leading to a market shift. The market substitution will be brought about by technological innovation, cheaper alternatives, or changes in the user's interests. The findings of this analysis confirm that just the use of VCC is a safer choice with respect to EC, in the case the offloading requests are the 1% of the ones in Fig 4.9 and Table 4.4. This study concludes that the VCC is a safer investment option and, at the same time, cheaper than the EC.

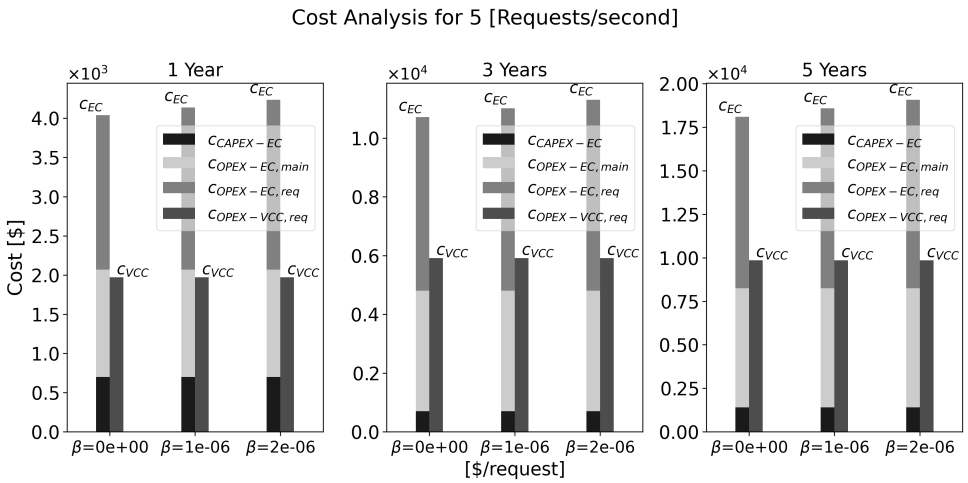


Figure 4.9: *Total cost of the EC compared with VCC for 1, 3, and 5 operational years for 1% of the total requests rate in the previous cost figure, i.e., 500 requests/second before and 5 requests/second here.*

β	$c_{CAPEX\text{-}EC}$ (%)	$c_{EC\text{-}main}$ (%)	$c_{OPEX\text{-}EC,req}$ (%)	$c_{OPEX\text{-}VCC,req}$ (%)
0e+00	17.33	33.88	48.79	100.00
1e-06	16.92	33.07	50.01	100.00
2e-06	16.52	32.30	51.18	100.00

Table 4.6: Distribution of costs for EC and VCC in the first year, expressed in percentage for the 1% of offloading requests.

4.8 . Conclusion

This chapter analyses and illustrates that VCC is a cost-effective replacement for EC, and can satisfy the requirement of low-latency applications. Indeed, failure rates remain very low (< 4%), even in high vehicle mobility scenarios. Moreover, even relatively few utilized vehicles, e.g., more than 4, can already match the needs of task offloading, except with very high request rates, where CC must be used as a backup. A cost analysis shows that VCC can bring considerable savings to NOs when deploying EC. However, EC remains irreplaceable in extreme cases when latency requirements fall below 16ms.

In the analysis, simple conservative task allocation strategies are employed. We chose to do so since we aimed to assess the benefits of VCC in a conservative setting. The next chapter addresses sustainability by considering energy consumption alongside task deadlines, and proposes an incentive-based strategy to foster the effective use of the VCC.

5 - A Management Framework for Vehicular Cloud toward Economic and Environmental Efficiency

We have shown in our previous chapters that Vehicular Cloud Computing (VCC) represents a promising solution in the context *ubiquitous computing*. Connected vehicles' underutilized resources enable scalable, responsive, and sustainable task offloading, especially in dense urban areas where their abundance allows leveraging mobile computing without extra infrastructure. However, as demand grows for low-latency and compute-intensive services, a key challenge arises: satisfying performance constraints while minimizing the environmental footprint of digital infrastructures. This dual goal has driven the emergence of *green computing* strategies aimed at reducing energy consumption and CO_2 emissions in distributed communication systems.

Therefore, despite the potential of VCC its practical viability remains unclear. Key questions include: *What are the energy implications for participating vehicles and infrastructure? How can profits be distributed to incentivize stable cooperation among all stakeholders?*

This chapter investigates the feasibility of VCC in 5G-enabled urban environments, considering performance, sustainability, and incentives. The main contributions are:

- **Integrated simulation of delay and energy-aware task allocation:** We develop a framework that integrates urban vehicular mobility traces, 5G communication, task characteristics, and energy models. Based on this, we design a utility-driven, lightweight allocation strategy that respects deadlines and energy constraints while promoting sustainable task offloading.
- **Coalitional revenue sharing under uncertainty:** We model a stochastic coalitional game to capture uncertainty in vehicular networks and prove that its robust core is non-empty. This guarantees the existence of stable, fair allocations and enables an incentive scheme that ensures participation under dynamic conditions.
- **Assessment of environmental and economic impact:** We quantify the CO_2 savings compared to traditional edge infrastructures and analyze long-term incentives for vehicle participation, including per-task compensation and hardware reuse.

The remainder of the chapter is organized as follows. Section 5.1 reviews the related work. Section 5.2 presents the system architecture. Section 5.3

defines the delay, energy, and utility models. Section 5.4 and Section 5.5 detail the task allocation and revenue-sharing mechanisms. Section 5.6 outlines the simulation setup. Section 5.7 and 5.8 report performance, energy, and emissions results. Finally, section 5.9 concludes the chapter.

5.1 . Related Work

This section reviews key references on task allocation, cooperation, and sustainability in vehicular systems, and highlight how our framework addresses existing limitations.

5.1.1 . Energy-aware Task Allocation in Vehicular Systems

Early studies such as [82] revealed energy-performance trade-offs in mobile offloading. Later, location and delay aware task allocation strategies were proposed [83], while Liu et al. [50] explored mobility-aware multi-hop task distribution. Hybrid models were also proposed including joint communication-computation [51] and mean-field game-theoretic decision making [52]. More recently, game-theoretic frameworks such as bargain-match have been introduced to tackle task offloading challenges such as handling the volatility of vehicular resources and ensuring incentive mechanisms in vehicular systems [84]. Benders decomposition was also applied to handle time-varying resources [85].

However, as highlighted in [86], many works assume stable infrastructure and overlook network volatility, heterogeneity, and energy limitations in real vehicular settings. Our model explicitly integrates such constraints into a cooperative optimization strategy.

5.1.2 . Cooperative and Intelligent Offloading

AI-based strategies like deep reinforcement learning [45] and online optimization [46] have demonstrated adaptability, while cooperative V2V offloading [50] supports redundancy. Wan et al. [87] proposed decentralized schemes for 5G VCC. Yet, signaling overhead and energy costs become critical, as shown by [88], limiting their scalability in practice.

Recent work by Tang et al. [89] introduced a link topology-adaptive offloading scheme based on graph reinforcement learning, improving latency performance in vehicular edge computing scenarios. While effective in topological adaptation, such approaches often overlook economic incentives and energy profiling, which are critical for real-world deployment.

Our solution mitigates these limitations by coupling cooperative task allocation with lightweight signaling mechanisms and CO_2-aware utility modeling. This ensures responsiveness, economic sustainability, and energy efficiency without incurring excessive communication overhead.

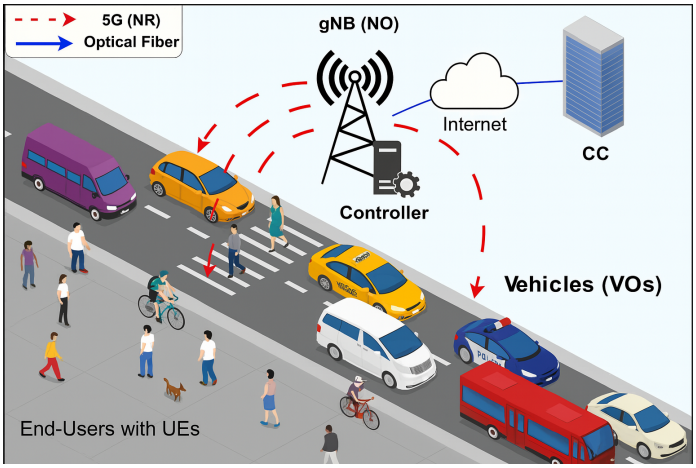


Figure 5.1: *Proposed 5G-based architecture with centralized control for coordinating vehicular and cloud resources.*

5.1.3 . Game-Theoretic Cooperation

Coalitional Game Theory has been adopted to model economic incentives among vehicles and stakeholders [90], using constructs like the Shapley Value. Yet, high computational cost and assumptions of static environments hinder their practical use [91].

A recent survey [92] covers classical and Evolutionary Game Theory (EGT) for Edge/Fog/Cloud offloading, noting the absence of integrated models balancing delay, energy, and profitability. Our model directly addresses this by embedding energy and delay aware incentives in the cooperative payoff structure.

5.1.4 . Positioning of Our Contribution

In this chapter, we propose a cooperative offloading framework that jointly accounts for latency, energy, and cost, and introducing carbon emission incentives into the utility design. Compared to prior works, we explicitly model stakeholder heterogeneity, incorporate real-time constraints, and validate performance using reproducible simulations grounded in urban vehicular mobility and energy models. This enables practical and sustainable deployment of VCC strategies in 5G vehicular environments.

5.2 . Architecture

Figure 5.1 illustrates the proposed 5G-based architecture for vehicular task offloading. Note that we rely on centralized control architecture. Indeed, as traditional Vehicular Ad-Hoc Networks (VANET) face scalability and flexibility issues, centralized solutions using Software-Defined Networking (SDN)

and Time-Sensitive Networking (TSN) are able to improve real-time communication, resource management, and service-oriented control in vehicular systems [93]. The system comprises:

- **End-users:** with mobile 5G User Equipments (UEs) devices issuing offload requests;
- **gNodeB:** 5G base station forwarding tasks to the Controller, owned by the **Network Operator (NO)**;
- **Controller:** co-located with gNodeB, manages beacons, task allocation (Sec. 5.4) and revenue sharing, owned by the **NO**;
- **Vehicles:** compute-capable 5G nodes broadcasting periodic status, owned by the **Vehicle Owners (VOs)**, equipped with 5G UEs;
- **Cloud (CC):** backup compute node or cluster with consistent availability, large computational power, and persistent network access.

5.2.1 . System Workflow and Interactions

As previously, we assume that end-users (i.e., the UEs) have already decided to offload to save their battery or have a better QoS, e.g., via decision making mechanisms in [62].

The *Controller* operates in sequential phases: it begins by collecting beacons from execution nodes, which include vehicles and, less frequently (order of minutes), the cloud. Vehicles use the ETSI CAM standard [60], sending beacons at a frequency of 10 Hz containing their current speed and position. Accordingly to this standard, a vehicle becomes unavailable if no beacon is received within 500 ms [60].

Vehicles also transmit aperiodic beacons upon task completion to update the controller about their resource availability over time. Each beacon contains the transmission timestamp, available computational resources, the number of accepted tasks, energy consumption by the processor and radio interface, their cost per kilowatt/hour, remaining energy for offloading, current geographic location, and speed. Upon receiving these beacons, the controller aggregates the information and ranks the candidate nodes based on their expected dwell time, energy availability, and computational capacity. The cloud node is considered to have infinite dwell time due to its static nature. Tasks are then assigned to the highest-ranked nodes that meet the corresponding task deadlines. Once executed, results are sent from the execution nodes back to the controller, which forwards them to the end-user.

5.3 . System Model

In this section, we formalize the *delay* and *energy consumption* models underlying the task allocation process. Then, the interactions among the stakeholders (NO, CC, VOs) are formalized through a cost model. The set of tasks to be offloaded at time-slot $z \in \mathcal{Z}$ is denoted as $\mathcal{T}_z$. At each time-slot z, if tasks $\mathcal{T}_z$ arrive from end-users, the NO considers set $\mathcal{E}_z$ of nodes as potential execution nodes. The cloud is always into $\mathcal{E}_z$. Then, the B vehicles with the highest ranking (see Sec. 5.2.1) are inserted in $\mathcal{E}_z$, where $B \geq \min(|\mathcal{T}_z|, M_{\text{veh}})$ and M_{veh} is the set of vehicles available for offloading. Parameter B must be chosen as a trade-off between having a large set of candidate execution nodes and restring them to only those in a condition favourable to offloading (in terms of mobility, capacity, energy).

5.3.1 . Offloading Time

A task is modeled as a tuple $\tau = (I_\tau, O_\tau, W_\tau, D_\tau)$, where I_τ and O_τ denote the input and output, including data and code, measured in [bits]. The term W_τ is the computational workload measured in Million Operations [MO], and D_τ is the execution deadline, in [seconds]. A task can be executed by an execution node n, which can be either the CC or a vehicle $v \in M_{\text{veh}}, z$, where M_{veh}, z is the set of offloading-capable vehicles at time slot s.

The offloading time for task τ executed by n is:

$$
\begin{aligned}
T_\tau^n = \; & \underbrace{\frac{I_\tau}{R_{\text{UL}}^{\text{5G}}} + \frac{O_\tau}{R_{\text{DL}}^{\text{5G}}}}_{\text{UE-NO transmission}} + \underbrace{2\,\frac{d_{\text{UE-NO}}}{c}}_{\text{Radio propagation}} \\
& + \underbrace{\frac{I_\tau}{R_{\text{UL}}^{n}} + \frac{O_\tau}{R_{\text{DL}}^{n}}}_{\text{NO-}n\text{ transmission}} + \underbrace{2\,\frac{d^n}{c^n} + 2\,\Delta^n}_{\text{Network delay}} \\
& + \underbrace{\frac{W_\tau}{C^n}}_{\text{Computation}} + \underbrace{T_{\text{queue}}^n}_{\text{Queuing}} + \underbrace{T_{\text{alloc}}^n}_{\text{Allocation decision}}.
\end{aligned}
\tag{5.1}
$$

The offloading time T_τ^n in (5.1) includes the end-users to NO (UE-NO) transmission time, given by the uplink and downlink 5G rates $R_{\text{UL}}^{\text{5G}}$ and $R_{\text{DL}}^{\text{5G}}$, followed by the radio propagation delay over distance $d_{\text{UE-NO}}$ with speed c (light in air). The data transfer between NO and execution node n (vehicle or cloud) is characterized by rates R_{UL}^n, R_{DL}^n, and a propagation delay over distance d^n at speed c^n. If n is a vehicle, $c^n = c$. If $n = \text{CC}$, $c^n = c^{\text{OF}}$ denotes the speed of light in optical fiber. The term $2\Delta^n$ accounts for switching latency, which is null for vehicles as no Internet routing occurs. The computation time is W_τ / C^n, while T_{queue}^n and T_{alloc}^n represent queuing and allocation algorithm

delays. Since offloading occurs at millisecond scale, the vehicle displacement during T_τ^n is negligible, so d^n can be assumed constant.

5.3.2 . Energy and Cost Model

For energy modeling, we logically exclude the UE uplink, as it is exogenous to the task allocation decision. This focuses the analysis on infrastructure-side energy consumption only.

We define energy consumption per task τ for each architecture stakeholder as follows:

Network Operator The NO's energy consumption includes transmission to execution node n (via 5G link if n is a vehicle, or through the Internet if CC), execution of the task allocation algorithm, and 5G downlink transmission gNB to the UE.

$$e_\tau^{\mathsf{NO}} = p_{\mathsf{tx}}^{\mathsf{NO}} \left(\frac{I_\tau}{R_{\mathsf{UL}}^n} + \frac{O_\tau}{R_{\mathsf{DL}}^{\mathsf{5G}}} \right) + p_{\mathsf{CPU}}^{\mathsf{NO}} \, T_{\mathsf{alloc}}^n \tag{5.2}$$

where $p_{\mathsf{tx}}^{\mathsf{NO}}$ is the NO's gNB transmit power and $p_{\mathsf{CPU}}^{\mathsf{NO}}$ is the *Controller* CPU power draw.

execution nodes Vehicles and CC consume energy for task computation and output transmission:

$$e_\tau^n = p_{\mathsf{CPU}}^n \, \frac{W_\tau}{C^n} + p_{\mathsf{tx}}^n \, \frac{O_\tau}{R_{\mathsf{UL}}^n} \tag{5.3}$$

The first term is the computational energy, with CPU power p_{CPU}^n. The second is the transmission energy, with p_{tx}^n for result upload. We note that beacon energy is not accounted for, as beacon information is embedded within CAM messages that vehicles already transmit for driving-related applications. The energy consumption is influenced by several random variables, including channel quality, transmission power, and task execution time. These variables are inherently correlated with vehicular mobility. For instance, the computational capacity of a vehicle depends on its current workload: if it is already performing internal computation (for instance related to driving), fewer computing resources remain available for newly offloaded tasks. Similarly, mobility affects network conditions and coverage may fluctuate due to changes in speed or position within the mobility scenario. In the case of CC, the randomness primarily stems from the network condition and the backhaul traffic. Network congestion, routing delays, and queuing effects in the core infrastructure contribute to the variability in energy consumption and latency. The offloading time and the corresponding energy consumption are random variables, as they depend on uncertain system conditions. We formalize these

uncertainties through a parameter vector defined, per each time-slot z, as $\boldsymbol{\theta} = (T^n_\tau, e^n_\tau, e^{\mathsf{NO}}_\tau)_{n\in\mathcal{K}_z,\tau\in\mathcal{T}_z}$, where each component represents a random variable. In contrast, the task parameters $\tau = (I_\tau, O_\tau, W_\tau, D_\tau)$ are assumed to be fixed upon arrival.

Cost Model For each task $\tau \in \mathcal{T}_z$ and slot $z \in Z$, the energy-related cost suffered by each stakeholder $i \in \mathcal{K}_z = \{\mathsf{NO}, \mathsf{CC}, \mathsf{VO}^1, \dots, \mathsf{VO}^m\}_z$ is a random variable:

$$c^i_\tau(\boldsymbol{\theta}) = \lambda^i e^i_\tau(\boldsymbol{\theta}) \quad [\$] \tag{5.4}$$

where λ^i denotes the energy-to-cash conversion rate [\$/J], and $e^i_\tau(\boldsymbol{\theta})$ is the stochastic energy consumption of i, defined in (5.2)-(5.3). In the following, $\boldsymbol{\theta}^\omega$ denotes the realization of $\boldsymbol{\theta}$, for outcome $\omega \in \Omega$.

5.4 . Task Allocation Problem

We define the system-wide utility f_z as the total value generated at time-slot $z \in Z$, through the assignment of tasks $\tau \in \mathcal{T}_z$ to the execution nodes $\mathcal{E}_z = \mathcal{K}_z \setminus \{\mathsf{NO}\}$. Although the NO does not execute tasks, it participates in every offloading operation via orchestration and transmission, incurring a per-task cost $c^{\mathsf{NO}}_{\tau,z}(\boldsymbol{\theta})$.

Each task τ yields payment $p(D_\tau, W_\tau)$ from the end-user. This payment is received by the NO through end-user *pay-per-use* subscriptions and will later be redistributed according to the game-theoretic mechanism described in Sec. 5.5. Execution nodes do not receive direct payments but are indirectly compensated through the cooperative cost-sharing scheme. Let $c^{\mathsf{off,i}}_{\tau,z}(\omega) = c^i_{\tau,z}(\omega) + c^{\mathsf{NO}}_{\tau,z}(\omega)$ denote the total cost incurred by execution node i and the NO for the offloading of task τ at time-slot z, under realization ω. The NO needs to decide allocation $\boldsymbol{y}_z = (y^i_{\tau,z})_{\tau\in\mathcal{T}_z, i\in\mathcal{E}_z}$, at time-slot z, where $y^i_{\tau,z}$ is 1 if the task is allocated to execution node i, and 0 otherwise. Note that the realizations of the random variables are not known at the moment of the decisions, which have thus to be taken based on nominal parameter values $\hat{\boldsymbol{\theta}}$. Some parameters are estimated directly by the NO, e.g., network conditions, such as $R^{\mathsf{5G}}_{\mathsf{UL}}, \dots$). Other parameters, such as time or energy (T^i_τ, e^i_τ) are estimated based on information reported by the execution nodes, e.g., position, power values p^i_{tx}, capacity available for offloading C^i, via Eqs. (5.1)-(5.3). Denote the indicator function as $\mathbf{1}_{i\tau} := \mathbf{1}_{\{T^i_\tau \le D_\tau\}}$. The indicator function equals 1 if execution node i completes task τ before its deadline D_τ, otherwise, it equals 0.

Decision $\boldsymbol{y}^*_z$ is the result of the following optimization (based on nominal values, which we denote with a hat):

$$\boldsymbol{y}^*_z = \underset{\boldsymbol{y}_z}{\mathsf{argmax}} \sum_{i\in\mathcal{E}_z}\sum_{\tau\in\mathcal{T}_z} \left(p(D_\tau, W_\tau)\mathbf{1}_{i\tau} - c^{\mathsf{off,i}}_{\tau,s}(\hat{\boldsymbol{\theta}})\right) y^i_{\tau,s} \tag{5.5}$$

Subject to:

$$\sum_{i \in \mathcal{E}_z} y^i_{\tau,z} \leq 1, \qquad \forall \tau \in \mathcal{T}_z \qquad \text{(one execution node per task)} \tag{5.6}$$

$$\sum_{\tau \in \mathcal{T}_z} y^i_{\tau,z} \leq q^i, \qquad \forall i \in \mathcal{E}_z \qquad \text{(execution node capacity)} \tag{5.7}$$

After offloading and task execution, the execution nodes report the actual parameter values via updated beacons. The NO can thus recompute consumed energy and actual execution times, which may differ from the nominal ones, which were used at the moment of decision. Revenue sharing is based on such updated values (Sec. 5.5). To prevent dishonest reporting, anti-fraud mechanisms could be introduced, but are beyond the scope of this work.

5.5 . Revenue Sharing via Cooperative Game Theory

At every time-slot $z \in Z$, tasks are allocated through optimization (5.5)-(5.7), and the resulting utility must be redistributed among the involved stakeholders. We model revenue-sharing via coalitional game theory. Let

$$\mathcal{A}_z = \left\{ i \in \mathcal{E}_z \mid \textstyle\sum_{\tau \in \mathcal{T}_z} y^{i,*}_{\tau,z} \geq 1 \right\}$$

be the subset of execution nodes that are selected by the optimal allocation $\boldsymbol{y}^*_z$ at time-slot z. The set of players is defined as $\mathcal{N}_z = \mathcal{A}_z \cup \{\text{NO}\}$. A classical coalitional game is defined by the pair $(\mathcal{N}_z, v_z)$, where $v_z : 2^{\mathcal{N}_z} \to \mathbb{R}$ assigns a value to each possible coalition $\mathcal{S} \subseteq \mathcal{N}_z$, representing the maximum utility that $\mathcal{S}$ can achieve independently of the remaining players. The grand coalition corresponds to the full set $\mathcal{N}_z$ (including the NO, the cloud and all the vehicles available, even those which were not assigned any task), and the domain $2^{\mathcal{N}_z}$ is the set of all its $2^{|\mathcal{N}_z|}$ possible subgroups. A payoff allocation $x \in \mathbb{R}^{\mathcal{N}_z}$ represents the revenue assigned to each player in the game. It is said to be **efficient** if it fully distributes the value of the grand coalition, i.e., $\sum_{i \in \mathcal{N}_z} x^i = v_z(\mathcal{N}_z)$, and it is **coalitionally rational** if no group of players receives less than its own value, i.e., $\sum_{i \in \mathcal{S}} x^i \geq v_z(\mathcal{S})$ for all $\mathcal{S} \subseteq \mathcal{N}_z$. The ***core*** of the game is the set of payoff allocation vectors that satisfy both *efficiency* and *coalitional rationality*. The core defines the set of stable distributions thanks to which no subset of players has an incentive to deviate from the grand coalition.

5.5.1 . Robust Coalitional Game

The NO plays a central role in our proposed setting, since the payments pass through it. As both the infrastructure owner and the communication coordinator, the NO is responsible for solving the task allocation problem (5.5). The value of a coalition $\mathcal{S} \subseteq \mathcal{N}_z$ is defined via a *value function*, which evaluates

the realized utility for each coalition after a specific outcome $\omega \in \Omega$, once the task allocation decision has been made based on nominal values (Sec. 5.4). The coalition value function $v_z^\omega(\cdot)$ is stochastic, due to the deviation from nominal values that impacts the costs and collected payments. We formalize a game with transferable payoff (propositions 262.1 and 264.2 [23]) since the value collected by all the coalition can be *transferred* among players. Given a weight $\beta^i \in [0,1]$ and multiplying it by payment $p(D_\tau, W_\tau)$ (defined in Sec. 5.4), we represent the proportion of payment given to player i, which could be the NO or an execution node. The sum is $\sum_{i\in\mathcal{N}_z} \beta^i = 1$. Payment is collected from the end-users only if the task completes before its deadline. Let $T_\tau^{i,\omega}$ denote the actual completion time of task τ by execution node i under realisation ω, and define the indicator $\mathbf{1}_{i\tau}^\omega := \mathbf{1}_{\{T_\tau^{i,\omega} \le D_\tau\}}$. This indicator equals 1 only when the task meets its deadline at sample ω, otherwise it is 0, in which case both the execution node and the network operator receive no payment for that task, regardless of its nominal allocation. For the NO, we do not introduce a completion time (since it does not execute tasks directly). Instead, we define its revenue share to be conditional on successful completion by at least one execution node, i.e.,

$$\mathbf{1}_{\mathrm{NO},\tau}^\omega := \max_{i\in\mathcal{N}_z\setminus\{\mathrm{NO}\}} \mathbf{1}_{i\tau}^\omega.$$

For all other aspects, the system parameters are treated as a realization denoted by $\boldsymbol{\theta}^\omega$. For any coalition $\mathcal{S} \subseteq \mathcal{N}_z$, we define the set of tasks actually offloaded within the coalition as:

$$\mathcal{T}_z^{\mathcal{S}} := \left\{\tau \in \mathcal{T}_z \;\middle|\; \exists\, i \in \mathcal{S}\setminus\{\mathrm{NO}\} \text{ such that } y_{\tau,z}^{i,*} = 1\right\}. \tag{5.8}$$

The value function is defined as follows. When the coalition is formed by the NO alone, i.e., $\mathcal{S} = \{\mathrm{NO}\}$, the NO incurs only the cost (5.4) to dispatch all tasks $\mathcal{T}_z^{\mathcal{N}_z}$, and receives a fraction β^{NO} of the total payment $p_\tau = p(D_\tau, W_\tau)$.

$$v_z^\omega(\mathcal{S}) = \sum_{\tau\in\mathcal{T}_z^{\mathcal{N}_z}} \left(\beta^{\mathrm{NO}}\, p_\tau\, \mathbf{1}_{\mathrm{NO},\tau}^\omega - c_\tau^{\mathrm{NO}}(\boldsymbol{\theta}^\omega)\right) \tag{5.9}$$

For calculating the value function at any coalition $\mathcal{S} \subseteq \mathcal{N}_z \setminus \{\mathrm{NO}\}$, each execution node receives its payment share and incurs a cost related to the energy spent in computation and transmissions (without taking into account the NO's energy cost). While (5.9) includes all tasks, as they are all handled by the NO, here the set of processed tasks considered into the value function depends on the coalition $\mathcal{S}$:

$$v_z^\omega(\mathcal{S}) = \sum_{i\in\mathcal{S}} \sum_{\tau\in\mathcal{T}_z^{\{i\}}} \left(\beta^i\, p_\tau\, \mathbf{1}_{i\tau}^\omega - c_\tau^i(\boldsymbol{\theta}^\omega)\right) \tag{5.10}$$

The formula above shows that for a task τ that is not served within the deadline ($\mathbf{1}_{i\tau}^\omega = 0$), the coalition still suffers the related energy cost (for processing it and transmitting the relative result), even if no payment is collected.

Finally, for every coalition $\mathcal{S} \ni \mathrm{NO}$, the value function includes the contributions of both the network operator and the participating execution nodes. The NO processes all tasks in $\mathcal{T}_z$, while each execution node $i \in \mathcal{S} \setminus \{\mathrm{NO}\}$ only contributes through the tasks allocated to it via solving the task allocation problem of Sec. 5.4. Thus, the utility of the coalition is the sum of the NO's contribution and the individual utilities of the execution nodes, including both their own cost and the supervision cost incurred by the NO.

$$v_z^\omega(\mathcal{S}) = v_z^\omega(\{\mathrm{NO}\}) + v_z^\omega(\mathcal{S} \setminus \{\mathrm{NO}\}) \tag{5.11}$$

We define a *robust cooperative game* as the tuple $(\mathcal{N}_z, \mathcal{V}_z)$, where $\mathcal{N}_z$ is the set of players and $\mathcal{V}_z = \{v_z^\omega \mid \omega \in \Omega\}$ is the set of all possible realizations of the coalition value function under uncertainty. To guarantee core stability, we prove that both *efficiency* and *coalitional rationality* hold for every $v_z^\omega \in \mathcal{V}_z$. Theorem 1 below, which relies on the Bondareva–Shapley theorem (Proposition 262.1 in [23]), proves that the core of game $(\mathcal{N}_z, v_z^\omega)$ is non-empty for every realization $\omega \in \Omega$.

Theorem 1 (Robust Core Non-Emptiness)**.** *Fix a slot $z \in Z$, a realisation $\omega \in \Omega$, and a weight vector $\boldsymbol{\beta} = (\beta^i)_{i \in \mathcal{N}_z}$ with $\beta^i \geq 0$ and $\sum_{i \in \mathcal{N}_z} \beta^i = 1$. Let $\mathcal{C} := 2^{\mathcal{N}_z}$ be the set of all possible coalitions, and let $G^\omega_{z,\boldsymbol{\beta}} = (\mathcal{N}_z, v_z^\omega)$ be the coalitional game with transferable utility, whose value function is defined by Eqs.* (5.9)–(5.11)*. Then the game $G^\omega_{z,\boldsymbol{\beta}}$ has a non-empty core.*

Proof. For brevity, set $d_{i\tau} := \beta^i p_\tau \mathbf{1}^\omega_{i\tau}$, $d^{\mathrm{NO}}_\tau := \beta^{\mathrm{NO}} p_\tau \mathbf{1}^\omega_{\mathrm{NO},\tau}$, $c_{i\tau} := c^i_\tau(\boldsymbol{\theta}^\omega)$, $c^{\mathrm{NO}}_\tau := c^{\mathrm{NO}}_\tau(\boldsymbol{\theta}^\omega)$. Let $\{\alpha_\mathcal{S}\}_{\mathcal{S} \in \mathcal{C}}$ be a balanced collection, i.e., $\alpha_\mathcal{S} \geq 0$ for all $\mathcal{S}$ and $\sum_{\mathcal{S} \ni i} \alpha_\mathcal{S} = 1$ for every $i \in \mathcal{N}_z$. The existence of at least one balanced collection is guaranteed (e.g., the collection defined as $\alpha_{\mathcal{N}_z} = \begin{cases} 1 & \text{if } \mathcal{S} = \mathcal{N}_z \\ 0 & \text{otherwise} \end{cases}$.

By Eqs. (5.9)–(5.11), the value of a coalition is additive between the contributions of the individual members and the contribution of the NO. Moreover, the sets $\mathcal{T}_z^{\{i\}}$ and $\mathcal{T}_z^{\mathcal{N}_z}$ are determined by the nominal allocation (fixed ex ante) and do not depend on the coalition $\mathcal{S}$.

Using these definitions and exchanging the order of summations, we obtain:

$$\sum_{\mathcal{S} \in \mathcal{C}} \alpha_\mathcal{S}\, v_z^\omega(\mathcal{S}) = \sum_{\mathcal{S} \in \mathcal{C}} \alpha_\mathcal{S} \left[\sum_{i \in \mathcal{S} \setminus \{\mathrm{NO}\}} \sum_{\tau \in \mathcal{T}_z^{\{i\}}} (d_{i\tau} - c_{i\tau}) + \mathbf{1}_{\{\mathrm{NO} \in \mathcal{S}\}} \sum_{\tau \in \mathcal{T}_z^{\mathcal{N}_z}} (d^{\mathrm{NO}}_\tau - c^{\mathrm{NO}}_\tau) \right]$$

$$= \sum_{\mathcal{S}\in\mathcal{C}} \sum_{i\in\mathcal{N}_z\setminus\{\mathrm{NO}\}} \sum_{\tau\in\mathcal{T}_z^{\{i\}}} (d_{i\tau} - c_{i\tau})\, \mathbf{1}_{\{i\in\mathcal{S}\}}\alpha_{\mathcal{S}} \;+\; \sum_{\mathcal{S}\in\mathcal{C}} \sum_{\tau\in\mathcal{T}_z^{\mathcal{N}_z}} (d_\tau^{\mathrm{NO}} - c_\tau^{\mathrm{NO}})\, \mathbf{1}_{\{\mathrm{NO}\in\mathcal{S}\}}\alpha_{\mathcal{S}}$$

$$= \sum_{i\in\mathcal{N}_z\setminus\{\mathrm{NO}\}} \sum_{\tau\in\mathcal{T}_z^{\{i\}}} (d_{i\tau} - c_{i\tau}) \underbrace{\sum_{\mathcal{S}\ni i} \alpha_{\mathcal{S}}}_{=1} \;+\; \sum_{\tau\in\mathcal{T}_z^{\mathcal{N}_z}} (d_\tau^{\mathrm{NO}} - c_\tau^{\mathrm{NO}}) \underbrace{\sum_{\mathcal{S}\ni \mathrm{NO}} \alpha_{\mathcal{S}}}_{=1}$$

$$= \sum_{i\in\mathcal{N}_z\setminus\{\mathrm{NO}\}} \sum_{\tau\in\mathcal{T}_z^{\{i\}}} (d_{i\tau} - c_{i\tau}) \;+\; \sum_{\tau\in\mathcal{T}_z^{\mathcal{N}_z}} (d_\tau^{\mathrm{NO}} - c_\tau^{\mathrm{NO}})$$

$$=^{(5.9)--(5.11)} v_z^\omega(\mathcal{N}_z \setminus \{\mathrm{NO}\}) \;+\; v_z^\omega(\{\mathrm{NO}\}) \;=^{(5.11)}\; v_z^\omega(\mathcal{N}_z).$$

Thus $\sum_{\mathcal{S}} \alpha_{\mathcal{S}}\, v_z^\omega(\mathcal{S}) = v_z^\omega(\mathcal{N}_z)$ for every balanced collection, which means that the game $G^\omega_{z,\boldsymbol{\beta}}$ is *balanced*. By the Bondareva–Shapley theorem (see [23, Prop. 262.1]), its core is therefore non-empty. □

For more details on the proof of the theorem, see Appendix A. The previous theorem ensures that there is at least a revenue allocation vector $\boldsymbol{\varphi} = (\varphi_i)_{i\in\mathcal{N}_z}$ that is in the core, i.t., that satisfies:

$$\sum_{i\in\mathcal{N}_z} \varphi_i = v_z^\omega(\mathcal{N}_z), \qquad \sum_{i\in\mathcal{S}} \varphi_i \geq v_z^\omega(\mathcal{S}) \quad \forall \mathcal{S} \subsetneq \mathcal{N}_z.$$

The previous result implies that in case of negative value function the situation is economically undesirable, everyone bears cost and no one earns payoff but the cooperative-game core is still mathematically non-empty. So even in the extreme "no one gets paid" case, the Bondareva–Shapley continues to hold. In other words, even in this extreme case, all players have the interest in collaborating in the grand coalition composed of all players instead of forming smaller coalitions and share among them the revenue from the tasks processed by the execution nodes within such smaller coalitions. One Possible imputation that is in the core is the following.

Example 1 (Proportional Core Imputation)**.** *Assume execution node cost* $c_1 = 1$*, NO cost* $c_{\mathrm{NO}} = 2$*, and total payment* $p = 13$*, yielding a surplus* $V = 10$*. Set* $\beta_1 = \frac{1}{3}$*,* $\beta_{\mathrm{NO}} = \frac{2}{3}$*, so the imputation becomes* $\varphi_1 = \frac{10}{3} \approx 3.33$*,* $\varphi_{\mathrm{NO}} = \frac{20}{3} \approx 6.67$*. Then:* $v(\{1\}) = \frac{13}{3} - 1 = \frac{10}{3} = \varphi_1$ *and* $v(\{\mathrm{NO}\}) = \frac{26}{3} - 2 = \frac{20}{3} = \varphi_{\mathrm{NO}}$*. Thus, the imputation lies in the core.*

5.6 . Simulation Framework

This section details the simulation framework used for modeling mobility, task generation, beaconing, and controller operations, with the aim of ensuring full repeatability. Default simulation parameters are listed in Table 5.1. The parameter notation “NV” indicates that values follow a normal distribution with a mean equal to the indicated value and a standard deviation equal

Parameter	Value
Architecture and Mobility	
Vehicle average speed	13.1 km/h (Rome downtown, SUMO) [70]
Cloud nodes	1
Number of vehicles	10 to 200
Number of end-users	100 [94]
General Settings	
Simulation duration	500 allocations (5 ms each)
Simulation Seeds	1 to 40
Task collection window	5 ms
Task rate	1 to 10 tasks/s (Poisson)
Computation	
Cloud compute capacity	NV 10^{15} FLOPS, ∞ processors [95]
Vehicle compute capacity	NV $1.3 \cdot 10^{11}$ FLOPS, 10% of NVIDIA V100 [10], 1 processor per vehicle
Tasks	
Workload	NV $W_\tau = 500$ MO [68]
Task size	NV $I_\tau = O_\tau = 8000$ bits [13]
Deadlines	NV $D_\tau \in (16, 100, 500)$ ms [47]
Payment per task	$p_\tau(D, W_\tau) \in (5.7, 3.8, 1.9)\ \mu\$$
Energy	
Energy cost	0.15 \$/kWh
CPU max power of CC and vehicles	200 W
Network	
Latency to Cloud	35 ms [6]
gNodeB Tx power	23 dBm [58]
UE Tx power	23 dBm [58]
Speed of light in air	$c \approx 3 \times 10^8$, m/s
Speed of light in optical fiber	$c^{\text{OF}} \approx (2/3)c$, m/s

Table 5.1: Simulation parameters.

to 10% of that mean. The per-task cost reported in the table is derived from U.S. mobile data subscription prices, assuming an average offloading rate of two tasks per second per user. Due to the lack of public real-world measurements, this remains a theoretical approximation. To ensure conservative estimates, we selected low-end subscription plans, thus avoiding artificially optimistic projections based on premium tariffs. Finally, we assume that vehicles dedicate only 10% of their compute capacity to offloading, as most resources are reserved for higher-priority functions such as navigation, safety, and sensor processing. The implementation and simulation scripts are available in a GitHub repository at reference [96].

5.6.1 . Mobility

The vehicles' mobility traces in simulations are generated by the Simulator Of Urban Mobility (SUMO). We reproduce real-world road and traffic data from the city of Rome. Vehicles move with an average speed of 13.1km/h [70]. The end-users' UEs are considered static during each offloading round, since their movement over such short intervals is negligible (1.8cm for a fast walker for 10ms offloading procedure).

5.6.2 . Networking and Beacon Management

5G uplink/downlink performance is simulated using the urban NLOS channel model from 3GPP TR 38.901, incorporating path loss, fading, and shadowing. Data rates are estimated via a Shannon-based SINR approximation and physical transmission parameters (e.g., power, bandwidth, beamforming) follow 3GPP specifications. Interference is computed dynamically across active vehicles, while cloud offloading uses fixed-bandwidth Internet links with core network latency.

5.7 . Simulation Results

In this section, we evaluate the CC and VCC architectures in terms of offloading delay, energy consumption, and utility. Results cover the impact of task rates on performance and the resulting payoff allocation.

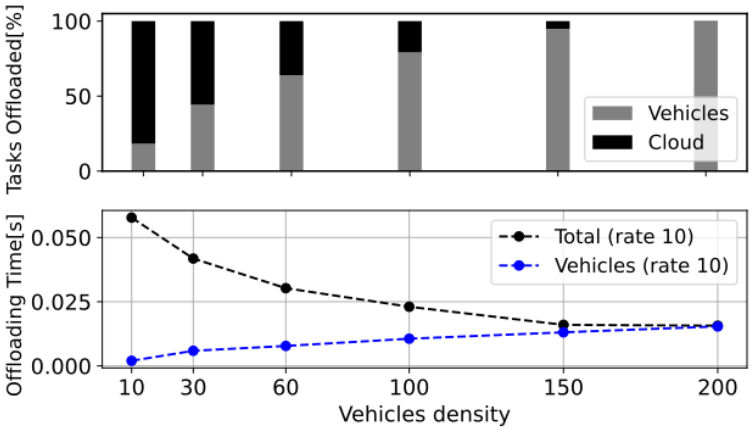


Figure 5.2: *Task offloading ratio to cloud and vehicles (stacked bars). Offloading time vs. vehicle density, shown for total (cloud + vehicles) and vehicle-only execution. Cloud latency is ∼70 ms across all densities.*

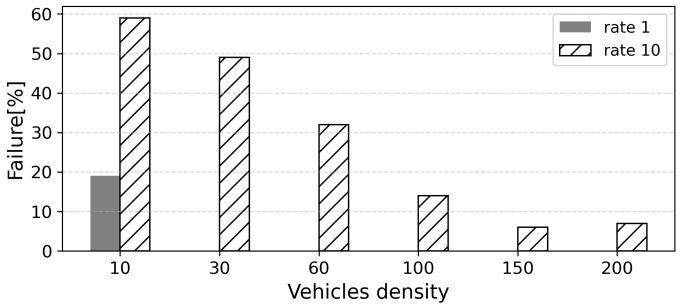


Figure 5.3: *Percentage of failed tasks as a function of the number of vehicles, evaluated for task rates of 1 and 10 tasks/s per 100 users.*

5.7.1 . Offloading Time and Failure rate

Figure 5.2 shows the offloading time as a function of vehicle density in the SUMO urban scenario, assuming a task rate of 10 tasks/s per user, i.e., each user offloads ten tasks per second. As vehicle density increases, the average total offloading time (cloud + vehicles) decreases, benefiting from reduced propagation and processing delays due to the proximity of vehicles. While cloud latency remains constant at 70 ms, the offloading time to individual vehicles increases slightly due to 5G bandwidth contention. However, the growing proportion of tasks served by nearby vehicles causes the overall average to converge toward vehicular performance. Assuming 10 tasks per user is a conservative scenario, allowing us to test system behavior under high load. The observed shift is confirmed by the decreasing share of tasks offloaded to the cloud, highlighting the scalability and increasing dominance of vehicles.

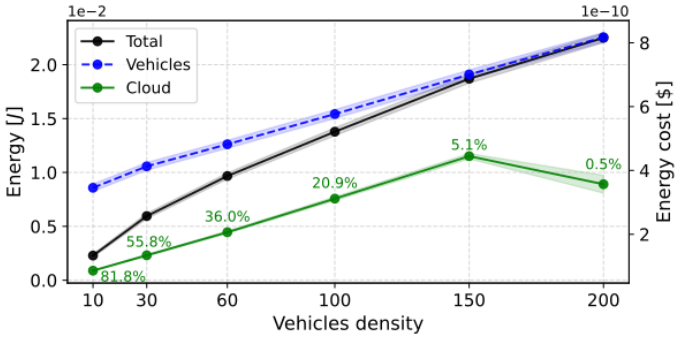


Figure 5.4: *Mean per-task* ***energy of the entire system*** *and associated cost versus vehicle density. Lines:* All executors *(total),* Cloud *(mean energy due to tasks allocated to cloud),* Vehicles *(mean energy due to tasks allocated to vehicles). Percentages indicate the proportion of tasks offloaded to the cloud or vehicular nodes. Request rate: 10 req/s per user; users: 100.*

Figure 5.3 shows the failure rate for the two task rates. A failure occurs when a UE sends a task, but no node can complete it within the firm deadline. For Task Rate 1, the failure rate quickly drops to zero due to low computational demand relative to the available VCC resources. Under Task Rate 10, failure is initially high with few vehicles, but decreases as their number grows, reaching about 5% at 150 vehicles an encouraging result for dense urban areas. However, at 200 vehicles, failure slightly increases due to signaling congestion: higher vehicle density intensifies beaconing, degrading network capacity. To address this, a signaling suppression mechanism should be introduced, where vehicles beyond a certain density stop broadcasting beacons, reducing overhead and preserving bandwidth.

5.7.2 . Energy Analysis

Figure 5.4 shows energy consumption as a function of vehicle density. The percentages on the curves match those in Fig. 5.2.

The energy consumption per task offloaded to vehicles exceeds that of tasks offloaded to the cloud, mainly due to the higher energy cost of 5G wireless up-links compared to wired connections. Wired communication is generally more efficient, with studies reporting up to 140% lower energy use than wireless links [97]. This conclusion holds for the architecture in Sec. 5.2. In real deployments, energy consumption may vary depending on the amount and the nature network equipment (e.g., switches, routers, firewalls) between the CC and the cellular core. These factors are hard to generalize. To stay architecture-agnostic, we assume a simplified setup where the CC connects directly to the 5G network. Observe that the per task energy consumption grows with the number of vehicles. This is because, the more the vehicles, the

more tasks are sent to them, the more the wireless channel is occupied, the less data-rate is available to transmit each task and the related results. This aspect is captured by, datarate $R_{\mathsf{DL}}^{\mathsf{5G}}$ available per task, which decreases as the reciprocal of the number of vehicles. This results, via (5.2), in an increase in the energy spent by the NO (and thus the entire system).

This does not mean that having more vehicles available is disadvantageous: on the contrary, it allows to serve more tasks within their deadlines (Fig. 5.3) and to get the consequent revenue (Sec. 5.7.3). Obviously, In order to do so, we need to send more tasks to vehicles and, as a consequence, spend more energy.

Note that the dip between 150 and 200 vehicles is an allocation effect: because a smaller share of tasks reaches the cloud, the BS↔cloud backhaul (parameter $R_{\mathsf{UL}}^{\mathsf{CC}}$) is less congested. The per-task data rate increases, transfer times shrink, and the communication-energy component in (5.2)–(5.3) drops. In that density range, this "free-backhaul" effect dominates the deterioration of the 5G downlink ($R_{\mathsf{DL}}^{\mathsf{5G}}$), so the average energy per cloud-offloaded task falls. Practically, selective offloading that centralizes only the residual load when backhaul is slack can be energy-advantageous.

This backhaul-relief mechanism also shows up in practice: a market analysis by iGR for Mavenir estimates that U.S. mobile operators could save over $546M in five years by offloading up to 32% of video traffic to the edge instead of the core network [98].

Impact on EV Battery Consumption: From the vehicle owner's perspective, a key question is whether offloading meaningfully affects overall energy consumption. Considering an electric vehicle such as Tesla Model S (17.5 kWh/100 km [99]), each offloading process consumes about 20 mJ (Fig. 5.4), amounting to only $3.17 \cdot 10^{-8}\%$ of total vehicle usage per trip. Most energy is used for propulsion (90%), with infotainment and other systems accounting for the remainder [100, 101].

5.7.3 . Utility Generation and Incentive Distribution

Figure 5.5 shows that as vehicle density increases, CC utility contribution drops while vehicles' one grows. Beyond 60 vehicles, vehicles outperform the CC.

The total utility reflects the combined net income of the NO and execution nodes after deducting energy costs, as illustrated in Fig. 5.4. This cloud result quite counter-intuitive since the result that we would expect is that the utility produced by vehicles is less since the energy consumed is higher than the one consumed by CC. However, the vehicles can serve tasks with a more stringent deadline ($<$ 16 ms) for which they get better payment, since their QoS is higher (see the prices in table 5.1). This means that the margin of profit

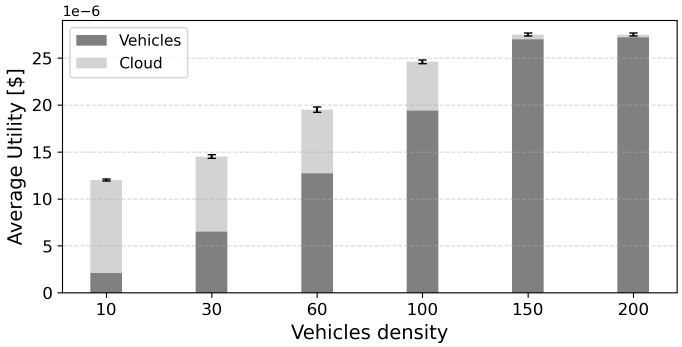


Figure 5.5: *Average utility ($) generated as a function of vehicle density.*

		Vehicle density					
Horizon	**Metric**	**10**	**30**	**60**	**100**	**150**	**200**
Monthly (×30 d)	Revenue [$]	1.22	1.24	1.20	1.13	0.99	0.77
	Charging [h]	1.16	1.18	1.14	1.08	0.94	0.73
	Range [km]	46	47	46	43	38	29
Yearly (×365 d)	Revenue [$]	14.84	15.07	14.59	13.69	12.09	9.41
	Charging [h]	14.14	14.37	13.90	13.09	11.47	8.92
	Range [km]	565	575	556	524	459	357
Lifetime (18.4 y)	Revenue [$]	273.04	277.26	268.49	251.96	222.49	173.22
	Charging [h]	260.11	264.38	255.85	240.92	211.07	164.17
	Range [km]	10404	10575	10234	9637	8443	6567

Table 5.2: Net revenues, equivalent EV charging hours on a 7 kW home wallbox, and corresponding driving range for a Tesla Model S (1h20 offloading/day, electricity at 0.15 $/kWh, vehicle consumption 17.5 kWh/100 km).

over the energy consumption is higher for stringent deadlines, hence vehicles in general have a higher net profit compared to cloud.

Incentives to car owners: A fundamental question in vehicular task offloading is: *What would motivate individuals to accept external tasks on their vehicles? What incentives justify their participation?* To answer these questions table 5.2 presents the revenues that a single Vehicle Owner in average could earn, when the price of the energy is 0.15 $/kWh [102]. Those computations are based on a maximum 1h20 daily offloading usage per vehicle [103]). The figures presented in the table show that the revenues decrease with the increasing number of vehicles according to the demand offer law: more vehicles imply more competition for the resources. In this evaluation, the lifetime of an electrical vehicle has been fixed to 18.4 years according to [104].

	Vehicle density					
Country	**10**	**30**	**60**	**100**	**150**	**200**
France [kg]	0.03	0.04	0.05	0.06	0.07	0.09
EU [kg]	0.13	0.17	0.20	0.24	0.30	0.35
USA [kg]	0.21	0.25	0.30	0.37	0.46	0.54
China [kg]	0.33	0.40	0.47	0.58	0.72	0.85

Table 5.3: Annual CO_2 emissions per vehicle (kg), assuming 1h20/day offloading, a task rate of 5 tasks/s, and country-specific grid intensity. The values are obtained by scaling the mean per-task energy of the entire system from Fig. 5.4.

5.8 . Carbon Emissions Analysis

5.8.1 . Estimated Carbon Emissions per Vehicle

Carbon emissions per vehicle are estimated by scaling the average energy used per task over a daily offloading duration of two hours and aggregating the result annually. Emissions are then calculated using national grid carbon intensity, which expresses CO_2 emissions per consumed kWh. This methodology provides a country-specific view of the environmental cost of vehicular offloading.

The carbon intensity of electricity production varies across countries: **France** emits 57 g CO_2/kWh, primarily due to its energy mix (5% fossil, 25% renewables, 70% nuclear) [105]; the **EU average** is 242 g CO_2/kWh (37% fossil, 39% renewables, 24% nuclear) [106]; the **United States** reaches 369 g CO_2/kWh (39% fossil, 39% renewables, 22% nuclear) [107]; and **China** peaks at 581 g CO_2/kWh (65% fossil, 31% renewables, 4% nuclear) [108].

As reflected in Table 5.3, the environmental cost of vehicular cloud computing varies widely from 70-190 g CO_2 per year in low-carbon countries to nearly 1.9 kg in high-emission countries. These figures underline the importance of grid carbon footprint intensity in evaluating the sustainability of offloading strategies.

5.8.2 . Vehicular vs. Edge Computing: Carbon Emissions

We contrast the greenhouse gas (GHG) footprint of a dedicated Edge server with that of a vehicle used for computation offloading. Following the ISO 14040 life-cycle assessment over a five-year functional unit [109], we account for production, operation, and end-of-life phases. An **Edge server** is assumed to follow a 1U (unit) form factor configuration, equipped with dual CPUs, 256 GB of RAM, and two SSDs, with an average power draw of 300 W, resulting in a total energy consumption of approximately 13.100 kWh over five years. The **manufacturing emissions** for such a device are estimated

Region	**Edge** [kg]	**VCC** [kg]	**Saving** [%]
France	1 372.00	0.25	99.98
EU-27	4 072.00	1.00	99.98
USA	5 817.00	1.50	99.97
China	8 753.00	2.35	99.97

Table 5.4: Five-year CO_2 emissions: Edge Server vs. single Vehicle in a 60-vehicle average density scenario. VCC values are obtained from Table 5.3 (density 60) over a 5-year horizon.

at around 900 $kgCO_2$, and up to 1.250 $kgCO_2$. End-of-life disposal provides a **recycling credit** ranging from −50 $kgCO_2e$ to −100 $kgCO_2e$. In contrast, a **VCC vehicle** reuses existing hardware and operates 2 hours per day over an estimated lifetime of 18.4 years [30, 104], with annual emissions of just 0.07 $kgCO_2$ in France and 0.73 $kgCO_2$ in China.

Besides emissions, VCC avoids material demand associated with Edge server production. Each saved server eliminates the need for 12 kg of steel (Fe), 1.5 kg of aluminum (Al), 1 kg of copper (Cu), and trace amounts of rare elements like silicon (Si), tantalum (Ta), neodymium (Nd), gold (Au), and silver (Ag). The extraction and processing of these materials entail severe environmental and geopolitical costs. VCC thus contributes not only to emissions reduction but also to raw-material conservation.

Table 5.4 compares the five-year CO_2 footprint of a dedicated edge server with that of a vehicle used for offloading tasks for 1h20/day. The savings column shows the relative emission reduction achieved by using VCC.

Even in carbon-intensive environments, using VCC instead of deploying dedicated edge servers cuts life-cycle emissions by over **99%**. This provides a strong incentive for vehicle owners, who benefit from both monetary rewards and environmental impact. Policymakers and sustainability-aware users can leverage this dual advantage to promote greener digital infrastructures.

In conclusion, each vehicle engaging in VCC can avoid up to 8.8 $tonsCO_2$ emissions over five years, reinforcing VCC's value as a sustainable alternative to infrastructure-based computing.

5.9 . Conclusion

The analysis conducted in this chapter confirms that Vehicular Cloud Computing is a technically viable and sustainable solution for low-latency task offloading in urban 5G networks. Vehicles absorb most of the traffic while maintaining delay within tens of milliseconds, satisfying applications with stringent latency constraints. Although individual vehicular executions consume more energy than cloud servers, the associated economic incentives offset this cost,

yielding (~ 1 hours of additional recharge time per vehicle monthly, and up to 1820 hours over an electric vehicle's lifetime. From an environmental standpoint, the added emissions from VCC are negligible: a vehicle operating in a dense scenario emits only 0.09-0.85 kg CO_2 annually depending on grid carbon intensity. Moreover, compared to traditional edge computing, VCC can save up to ~ 8.8 tons CO_2 over the lifetime of an edge server, reducing both emissions and the demand for rare earth elements. These results support the role of VCC as a sustainable and economically attractive complement to edge infrastructures.

6 - General Conclusions and Future Work

6.1 . General Conclusions

This thesis has been driven by a central question: can **Edge Computing** and **Vehicular Cloud Computing (VCC)** provide a technically viable, economically sustainable, and environmentally responsible alternative to centralized cloud infrastructures? The results obtained suggest that the answer is positive, although several barriers remain before large-scale deployment becomes a reality.

The first part of this work addressed the **economic barrier** to Edge deployment. Through a **coalitional game-theoretical co-investment** model, we (i) prove game convexity, hence grand-coalition stability (Sec. 3.5), and (ii) allocate costs and revenues via the Shapley value within the core, yielding an incentive-compatible split between NO and SPs. This provides a rational mechanism that *mitigates* the CAPEX/OPEX bottleneck, thereby dismantling one of the main obstacles to the deployment of EC. The need for such a mechanism is documented: mobile operators remain cautious due to weak business cases and limited use cases [7–9]. Moreover, global measurements show that "for most applications the cloud is already close enough for the majority of the world's population" [17], while hyperscalers are often better positioned to monetize edge-adjacent services at scale.

Before 2023 (year of the publication of my work on edge co-investment), the literature repeatedly flagged three major economic concerns: (*i*) unsustainable CAPEX/OPEX of distributed edge nodes [8], (*ii*) lack of compelling business cases for operators [7–9], and (*iii*) imbalance whereby NOs were expected to bear the costs while SPs reaped most benefits [110]. Our framework directly *mitigates* concerns (i) and (iii): by pooling investment among heterogeneous players, convexity ensures that cooperation is always beneficial, and Shapley-based allocations guarantee a fair split that lies in the core, so that both NO and SPs have incentives to participate. This removes the perception that the operator "pays while others gain." Concern (ii): the absence of strong use cases and the perception that latency gains are marginal is not solved by our model, as it depends on the evolution of applications and demand, but our results provide a credible economic framework that reduces the financial risk of deployment. In sum, this thesis mitigates the cost and incentive-alignment barriers identified in prior works, while acknowledging that the definition of compelling edge-native services remains an open challenge.

Enabling edge was itself a mitigation of the problem. This shifted the research question toward whether it is possible **to substitute edge infrastructures** with readily available, already deployed, and maybe sparse resources. The path explored naturally follows: the Vehicular Cloud Computing. The second contribution focused on the **technical feasibility** of Vehicular Cloud Computing. By combining realistic vehicular mobility (**SUMO**) with high-fidelity network simulations (**NS-3/5G-LENA**), we showed that vehicles can sustain **latency-sensitive applications** without dedicated edge servers, except in ultra-stringent cases below approximately sixteen milliseconds. This re-frames VCC not as a mere backup solution, but as a credible alternative or complement to Edge, depending on the application context and system requirements.

Once the conditions for replacing edge computing were assessed, such as vehicle density and related factors, new questions emerged from the research community. In particular, concerns were raised about the impact of external task offloading on vehicles' energy consumption and about the need to design incentives that would fairly benefit both the service providers managing the offloading and the vehicle owners contributing their resources. This brings to the third contribution, which concerns **sustainability and incentives**. Via a **life-cycle analysis** we demonstrated that replacing dedicated edge servers with vehicular resources reduces **carbon emissions by more than ninety-nine percent**, while also avoiding the extraction of scarce raw materials. Furthermore, cooperative game-theoretical mechanisms ensure that **vehicle owners** are compensated with tangible benefits, such as several full recharge cycles over a vehicle's lifetime, while operators and service providers also achieve positive utility. This **alignment of ecological and economic incentives** is fundamental for practical adoption.

Taken together, these results indicate that many of the usual objections to Edge and VCC deployment, such as excessive costs, insufficient latency performance, lack of environmental benefits, or absence of user incentives, can *conceivably* be overcome when the problem is addressed systematically. At the same time, it would be misleading to claim that all barriers have been removed. This thesis provides **fundamental first steps** to the removal of such barriers.

6.2 . Open Challenges

Several challenges remain before industrial deployment can be envisioned. **Safety, security, and privacy** remain at the forefront: car manufacturers are understandably reluctant to allow external tasks to interact with in-vehicle computation domains. Robust mechanisms of **task isolation** and **trusted execution** must therefore be developed or adapted to

vehicular environments. Although mature solutions exist in cloud computing (e.g., virtualization, containers, secure enclaves such as Intel SGX or ARM TrustZone), a direct import into vehicles is not feasible. Vehicular systems operate under stringent constraints. The Electronic Control Units (ECUs) have far less memory and computational power than servers, hardware and software stacks are highly heterogeneous across manufacturers, and failures may compromise functional safety as defined by standards such as ISO 26262 and UNECE WP.29. ISO 26262 specifies the requirements for functional safety in road vehicles, ensuring that electrical and electronic systems operate reliably even in the presence of faults, while UNECE WP.29 establishes the regulatory framework for automotive cybersecurity and software update management at the international level.

Moreover, connectivity is inherently intermittent, preventing reliance on a permanently available infrastructure, and the trust boundaries are more stringent, as vehicle owners and manufacturers remain reluctant to execute third-party code near critical control domains such as steering or braking. These factors highlight the need for domain-specific adaptations and novel mechanisms to ensure secure and reliable vehicular task offloading.

Experimental validation is essential: while simulations yield valuable insights, only prototypes, testbeds, and pilot deployments can secure industrial and regulatory trust. Notable examples include 5G-MOBIX, which conducted cross-border trials for connected automated mobility along corridors like Spain–Portugal and Greece–Turkey [111], and the 5GAA "Day One Deployment District," which showcased real-world C-V2X infrastructure in Atlanta [112]. Although these validate connectivity and safety use cases, they do not integrate economic incentive structures, cooperative offloading, or incentive-aligned orchestration. Future work should extend these testbeds to evaluate how VCC-specific mechanisms such as coalition formation and Shapley-based resource sharing perform in real vehicular conditions. **Market and policy frameworks** must be clarified. Some foundational work exists: the **GSMA Operator Platform** aims to federate edge computing capacity across operators by exposing network and compute capabilities via common APIs and architectures (cf. GSMA Operator Platform Telco Edge Requirements [113]); European projects like **5G-MOBIX** and **5GCroCo** have conducted cross-border trials for connected and automated mobility (CAM) use cases, addressing some regulatory and technical coordination issues [111, 114]. Despite this, there remains no formal framework that recognizes vehicles as transient infrastructure providers, no incentive schemes combining economic and ecological value for vehicle owners, and no market mechanisms ensuring fair value distribution among Network Operators, Service Providers, and end users. Until these elements are in place, VCC risks remaining confined to the lab rather than becoming a deployed technology.

Future research must therefore extend technical prototypes with market and regulatory models that legitimize and operationalize vehicular resource participation.

Finally, the **acceptance and behavior of vehicle owners** are crucial. The incentives analyzed in Chapter 5 show *positive but modest* monetary returns ($\sim$\$1 per month per vehicle; Table 5.2), *negligible* battery impact ($\sim 3.71 \cdot 10^{-8}$ % per offload; Fig. 5.4), and a *strong* ecological benefit (up to 8.8 t CO_2 avoided over five years), which *mitigate* but do not remove the adoption barrier. To persuade drivers at scale, further work should: (i) couple payments with non-monetary rewards (ecological credits, insurance discounts, electrical vehicles charging vouchers) *tied to verifiable participation*; (ii) provide safety/privacy assurances via attested execution and strict resource/energy caps with clear opt-in/opt-out controls; and (iii) run longitudinal pilots measuring willingness-to-participate, and the effectiveness of incentive mixes across driver segments.

6.3 . Future Work

Building on these results, several directions for future work can be envisaged. One promising line is the use of **digital twins** to validate safety and reliability before any road deployment. In the context of VCC, digital replicas of vehicular networks would allow testing how cooperative offloading behaves under faults or attacks, and whether incentive mechanisms remain stable when vehicles join or leave unpredictably. This would provide a risk-free environment to anticipate vulnerabilities that are otherwise impossible to probe on real roads.

A second step is the development of **prototype implementations**. Starting from toy cases with embedded controllers and 5G modules, prototypes could evaluate the feasibility of resource sharing across vehicles and quantify the actual energy and latency overhead of our models. Progressively extending to real vehicular testbeds and pilot fleets would reveal how economic incentives and orchestration mechanisms operate under genuine mobility patterns and heterogeneous vehicle hardware.

In parallel, the **stochastic availability** of vehicular resources calls for new **orchestration algorithms** that integrate Cloud, Edge, and Vehicle into a seamless continuum. Beyond what was proposed in this thesis, further work is needed on predictive orchestration that uses mobility traces and historical availability to anticipate resource gaps, and on robust scheduling strategies that can reallocate tasks in real time when vehicles leave the coalition unexpectedly.

Future wireless generations such as **6G**, with ultra-reliable low-latency communication, native **network slicing**, and **AI-driven orchestration**, open

additional research avenues. For VCC, 6G URLLC could enable safety-critical offloading such as cooperative perception or remote actuation. Network slicing could allow economic coalitions of vehicles to operate within isolated, QoS-guaranteed slices that align with their incentive structures. AI-driven orchestration offers the possibility to couple incentive mechanisms with adaptive scheduling, learning how to match tasks with the most reliable vehicular nodes. Exploring these directions would transform VCC from a theoretical alternative to a deployable architecture.

Finally, **incentive models** must evolve beyond pure **monetary rewards**. Other domains provide concrete inspiration: **carbon-credit trading** rewards **ecological contributions**, **usage-based insurance** offers **discounts** for safe driving, and **demand-response programs** remunerate flexible **energy use**. Transposed to VCC, these could translate into green credits for vehicles that offload tasks, **insurance discounts** linked to verifiable coalition participation, or rewards tied to **EV charging sessions**. Yet direct import is not possible: credits must be made tradable within mobility ecosystems, insurance must adapt to **dynamic vehicular coalitions**, and charging incentives must balance **battery health** with **driver needs**. Exploring these **adaptations** defines a precise and impactful line of future research.

The vision that emerges from this work is that of a **distributed infrastructure** that is **carbon-aware**, and **economically viable**. Vehicles, once passive clients of the network, can *likely* become **active computational agents**, micro data centers on wheels, supporting the resilience and sustainability of future smart cities. This thesis does not claim to have solved all the challenges of Vehicular Cloud Computing. Rather, it has laid down **essential conceptual and methodological foundations**, showing that the idea is not only technically feasible but also economically and environmentally desirable to pursue. The road ahead requires **collaboration across academia, industry, policymakers, and citizens**. Thanks to the foundations established here, that road is now visible and *realistically* worth traveling.

Résumé

Cette thèse analyse la faisabilité technique, économique et environnementale de nouvelles architectures de calcul distribuées destinées aux applications à très faible latence, telles que la Réalité Augmentée (AR, Augmented Reality) et la Conduite Autonome (AD). Le Cloud Computing (CC), bien qu'omniprésent, ne peut garantir les délais nécessaires à ces services, tandis que l'Edge Computing (EC) rapproche les ressources de calcul des utilisateurs mais implique des coûts d'infrastructure élevés, une consommation énergétique importante et une empreinte carbone accrue liée au cycle de vie court des équipements électroniques. Ces limites renforcent la nécessité d'explorer des alternatives plus flexibles et durables.

Parallèlement, le nombre de véhicules connectés augmente rapidement. Ceux-ci embarquent des ressources de calcul dimensionnées pour des tâches critiques mais largement sous-utilisées en pratique, ce qui ouvre la voie au Vehicular Cloud Computing (VCC). Ce paradigme exploite les capacités déjà déployées dans les véhicules pour exécuter des tâches déportées provenant d'utilisateurs ou d'appareils environnants. En tirant parti d'une infrastructure existante, le VCC permet de réduire les coûts de déploiement, de maintenir une proximité spatiale avec les utilisateurs et d'offrir une alternative évolutive et plus durable à l'Edge Computing traditionnel.

La première contribution de cette thèse propose un modèle de co-investissement basé sur la théorie des jeux coopératifs. Ce modèle permet à un opérateur réseau et à plusieurs fournisseurs de services de partager les coûts, les capacités déployées et les bénéfices générés par l'Edge Computing (EC). En formalisant les interactions entre acteurs hétérogènes, nous montrons que la convexité du jeu garantit la stabilité de la coalition complète, ce qui signifie que tous les participants ont intérêt à coopérer plutôt qu'à agir individuellement. La valeur de Shapley fournit une clé de répartition équitable des paiements, des revenus et de la capacité installée, tout en prenant en compte les contributions spécifiques de chaque acteur au sein de la coalition. Ce cadre méthodologique met en évidence comment un partage rationnel des investissements peut lever les principaux obstacles économiques qui freinent aujourd'hui l'implantation à grande échelle de l'EC et ouvrir la voie à un déploiement plus distribué, plus efficace et plus rentable.

La deuxième contribution examine les performances du VCC et identifie les conditions dans lesquelles il peut remplacer ou compléter l'EC. À travers des simulations à haute fidélité combinant SUMO pour la mobilité et NS-3 pour les communications 5G, nous montrons que le VCC peut atteindre des latences proches de 10 ms même en présence de densités modestes

de véhicules, et offrir des performances comparables à celles de l'EC pour une vaste gamme d'applications sensibles au délai. Une analyse des coûts sur plusieurs années révèle qu'un remplacement partiel de l'EC par le VCC peut réduire les dépenses d'infrastructure d'environ 10 % sur cinq ans, sans compromettre les exigences de qualité de service.

La troisième contribution développe un cadre de gestion complet intégrant consommation énergétique, émissions de carbone, allocation des tâches et répartition équitable des revenus générés par les utilisateurs. En mobilisant la programmation mathématique, un jeu coopératif robuste et des simulations de Monte-Carlo, nous montrons que l'énergie consommée par le VCC représente moins de 0.1 % de la consommation totale du véhicule. Les mécanismes proposés incitent efficacement les propriétaires à participer, tout en assurant une gestion durable, transparente et cohérente avec les objectifs environnementaux.

Dans l'ensemble, cette thèse montre que le VCC peut constituer une alternative crédible, durable et performante à l'Edge Computing, et fournit des outils théoriques et pratiques pour son intégration harmonieuse dans les réseaux mobiles de nouvelle génération.

A - Balancedness in Theorem 1 (Chapter 5)

To clarify why the weighted sum over a balanced collection coincides exactly with the value of the grand coalition in Theorem 1, it is useful to rewrite the coalition value in terms of elementary contributions. For each player $i \in \mathcal{N}_z \setminus \{\mathrm{NO}\}$ define

$$g_i := \sum_{\tau \in \mathcal{T}_z^{\{i\}}} (d_{i\tau} - c_{i\tau}),$$

and for the network operator (NO),

$$g_{\mathrm{NO}} := \sum_{\tau \in \mathcal{T}_z^{\mathcal{N}_z}} (d_\tau^{\mathrm{NO}} - c_\tau^{\mathrm{NO}}).$$

Equations (5.10)–(5.11) show that the value of a coalition is simply the sum of the contributions of its members:

$$v_z^\omega(\mathcal{S}) = \sum_{i \in \mathcal{S} \setminus \{\mathrm{NO}\}} g_i + \mathbf{1}_{\{\mathrm{NO} \in \mathcal{S}\}}\, g_{\mathrm{NO}}.$$

The task sets $\mathcal{T}_z^{\{i\}}$ and $\mathcal{T}_z^{\mathcal{N}_z}$ are fixed by the nominal allocation and are disjoint, so each g_i and g_{NO} represents a unique "atomic" contribution that cannot be double counted.

Now consider any balanced collection $\{\alpha_\mathcal{S}\}$. By definition, every player i (including the NO) appears in the coalitions with total weight one:

$$\sum_{\mathcal{S} \ni i} \alpha_\mathcal{S} = 1 \qquad \forall i \in \mathcal{N}_z.$$

When we compute the weighted sum of coalition values we therefore obtain

$$\sum_{\mathcal{S}} \alpha_\mathcal{S}\, v_z^\omega(\mathcal{S}) = \sum_{i \in \mathcal{N}_z \setminus \{\mathrm{NO}\}} g_i \Big(\sum_{\mathcal{S} \ni i} \alpha_\mathcal{S} \Big) + g_{\mathrm{NO}} \Big(\sum_{\mathcal{S} \ni \mathrm{NO}} \alpha_\mathcal{S} \Big).$$

Balancedness makes each parenthesis equal to one, so the whole expression reduces to

$$\sum_{\mathcal{S}} \alpha_\mathcal{S}\, v_z^\omega(\mathcal{S}) = \sum_{i \in \mathcal{N}_z \setminus \{\mathrm{NO}\}} g_i + g_{\mathrm{NO}} = v_z^\omega(\mathcal{N}_z).$$

In words, *balancedness ensures that each elementary contribution is counted exactly once in the weighted sum*. This is the game-theoretic meaning of the Bondareva–Shapley condition: a game is balanced if the total weight of coalitions covering each player is exactly one. Because our game is additive and the contributions are coalition-independent, the inequality of the theorem does not just hold but collapses into equality. As a consequence, the game $G_{z,\boldsymbol{\beta}}^\omega$ is balanced, and the Bondareva–Shapley theorem guarantees that its core is non-empty.

B - Detailed computation of the offloading energy share (Chapter 5)

This appendix presents the numerical assumptions and the explicit derivation of the percentage of total trip energy represented by a single computational offloading operation.

We consider a reference electric vehicle with an energy consumption of $17.5\,\text{kWh}/100\,\text{km}$. For consistency with this quantity, the reference trip length is set to $100\,\text{km}$, which directly corresponds to an energy expenditure of $17.5\,\text{kWh}$. The conversion between kilowatt-hours and joules uses the identity $1\,\text{kWh} = 3.6 \times 10^6\,\text{J}$. Applying this conversion gives

$$E_{\text{trip}}^{(J)} = 17.5 \times 3.6 \times 10^6 = 6.3 \times 10^7\,\text{J}.$$

The energy required by a single offloading operation is $20\,\text{mJ}$. Expressed in joules,

$$E_{\text{off}}^{(J)} = 20\,\text{mJ} = 20 \times 10^{-3}\,\text{J} = 2.0 \times 10^{-2}\,\text{J}.$$

To determine the fraction of trip energy represented by a single offloading operation, we take the ratio between these two quantities:

$$r = \frac{E_{\text{off}}^{(J)}}{E_{\text{trip}}^{(J)}} = \frac{2.0 \times 10^{-2}}{6.3 \times 10^7} \approx 3.17 \times 10^{-10}.$$

The corresponding percentage is obtained by multiplying by 100:

$$r_{\%} = 3.17 \times 10^{-10} \times 100 \approx 3.17 \times 10^{-8}\,\%.$$

This value shows that a single computational offloading operation contributes an extremely small fraction of the total energy consumed during a typical trip. Even when aggregated over many operations, the resulting consumption remains several orders of magnitude below any level that could be considered significant in the context of vehicle energy usage.

C - Derivation of revenues, charging hours and driving range (Chapter 5)

This appendix details the computation behind Table 5.2, which reports, for each vehicle density and time horizon, the net revenues per vehicle, the equivalent electric charging hours, and the corresponding driving range for a Tesla Model S.

The net revenues $R_{h,d}$ in Table 5.2 are obtained from the offloading model described in the main text, assuming 1h20 of offloaded computation per day and a given vehicle density d. For each time horizon h (monthly, yearly, and lifetime), the table entries provide the aggregated revenue per vehicle in US dollars. In order to interpret these values in terms of electric driving capability, the revenues are converted into energy, then into equivalent charging time on a representative home wallbox, and finally into a driving range using the energy consumption of a Tesla Model S.

The computation relies on three basic parameters. The unit electricity price is denoted by p and is set to $p = 0.15\,\$/\text{kWh}$. The charging power of the home wallbox is denoted by P_{ch} and is set to $P_{\text{ch}} = 7\,\text{kW}$, corresponding to a typical 7 kW AC home charger. The specific energy consumption of the vehicle is denoted by c_{veh} and is set to $c_{\text{veh}} = 17.5\,\text{kWh}/100\,\text{km}$, as reported for a Tesla Model S.

For each pair (h, d) of time horizon and vehicle density, the first step is to convert the net revenue $R_{h,d}$ (in dollars) into an equivalent amount of electrical energy $E_{h,d}$ (in kWh). This is done by dividing the monetary value by the unit price of electricity:

$$E_{h,d} = \frac{R_{h,d}}{p} \quad [\text{kWh}].$$

This quantity represents how many kilowatt-hours of energy could be purchased with the revenue generated by offloading over the chosen horizon.

The second step is to express this energy as an equivalent charging time on the home wallbox. Given the charging power P_{ch}, the charging time $t_{h,d}$ (in hours) is obtained by

$$t_{h,d} = \frac{E_{h,d}}{P_{\text{ch}}} = \frac{R_{h,d}}{p\,P_{\text{ch}}} \quad [\text{h}].$$

This is the number of hours that the vehicle could be connected to a 7 kW charger using only the energy paid by the offloading revenue.

The third step is to convert the energy $E_{h,d}$ into an effective driving range $D_{h,d}$ (in kilometres). The vehicle consumption c_{veh} is expressed in kWh per 100 km, so the corresponding distance covered by an energy budget $E_{h,d}$ is

$$D_{h,d} = E_{h,d} \times \frac{100}{c_{\text{veh}}} = \frac{R_{h,d}}{p} \times \frac{100}{c_{\text{veh}}} \quad [\text{km}].$$

This expression states that the range is proportional to the available energy in kWh and inversely proportional to the specific consumption of the vehicle.

Substituting the numerical values used in this work, namely $p = 0.15\,\$/\text{kWh}$, $P_{\text{ch}} = 7\,\text{kW}$, and $c_{\text{veh}} = 17.5\,\text{kWh}/100\,\text{km}$, gives the following simplified relations:

$$E_{h,d} = \frac{R_{h,d}}{0.15}, \qquad t_{h,d} = \frac{R_{h,d}}{0.15 \times 7}, \qquad D_{h,d} = \frac{R_{h,d}}{0.15} \times \frac{100}{17.5}.$$

For example, taking a monthly revenue of $R_{h,d} = 1.22\,\$$ for a given density, the equivalent energy is

$$E_{h,d} = \frac{1.22}{0.15} \approx 8.13\,\text{kWh},$$

the charging time on a 7 kW wallbox is

$$t_{h,d} = \frac{8.13}{7} \approx 1.16\,\text{h},$$

and the corresponding driving range for a Tesla Model S is

$$D_{h,d} = \frac{8.13}{0.175} \approx 46.5\,\text{km}.$$

The same formulas are applied to all revenue values in Table 5.2, for each time horizon and vehicle density, to obtain the charging hours and the driving ranges reported in the table.

D - Computation of Carbon Emissions (Chapter 5)

The CO_2 emissions associated with Vehicular Cloud Computing (VCC) are obtained by converting the mean per-task system energy consumption into annual electricity usage under region-specific carbon intensities. For a given vehicle density v, the per-task system energy $E_{\text{task}}(v)$ [J/task] is taken directly from the blue curve in Fig. 5.4, which accounts for on-board computation, wireless transmission, network forwarding, and fog/edge interactions.

Each vehicle offloads tasks at a fixed rate of $r = 5$ tasks per second over a daily offloading duration of $T_{\text{day}} = 1\text{h}20 = 4800$ seconds, resulting in $r\,T_{\text{day}}$ tasks per day. The total daily energy demand per vehicle is therefore

$$E_{\text{day}}(v) = E_{\text{task}}(v)\, r\, T_{\text{day}}.$$

Over one year (365 days), the energy becomes

$$E_{\text{year}}(v) = 365\, E_{\text{task}}(v)\, r\, T_{\text{day}}.$$

Joules are converted to kWh using 1 kWh $= 3.6 \times 10^6$ J, which yields

$$E^{\text{kWh}}_{\text{year}}(v) = \frac{365\, r\, T_{\text{day}}}{3.6 \times 10^6}\, E_{\text{task}}(v).$$

With the numerical parameters used here,

$$\kappa = \frac{365 \cdot 5 \cdot 4800}{3.6 \times 10^6} = 2.4333, \qquad E^{\text{kWh}}_{\text{year}}(v) = \kappa\, E_{\text{task}}(v).$$

Let I_c denote the carbon intensity of the electricity mix for region c: $I_{\text{FR}} = 0.057$, $I_{\text{EU}} = 0.242$, $I_{\text{US}} = 0.369$, $I_{\text{CN}} = 0.581$ kgCO_2/kWh. The annual CO_2 emissions produced by a single vehicle in region c at density v follow directly from

$$\text{CO2}_c(v) = E^{\text{kWh}}_{\text{year}}(v)\, I_c = \kappa\, E_{\text{task}}(v)\, I_c,$$

which is the expression used to compute all entries of Table 5.3. The five-year VCC values reported in Table 5.4 correspond simply to $5 \times \text{CO2}_c(v)$ evaluated at density $v = 60$, ensuring full consistency between emission estimates and the per-task energy profile extracted from Fig. 5.4.

E - Special Appendix: Online Materials and Updates (Chapter 5)

This document is not designed to be modified after publication. Readers who wish to access additional material, numerical extensions, clarifications, or a continuously maintained Frequently Asked Questions (FAQ) section may consult the dedicated online repository associated with this work.

The repository is publicly available at:

`https://github.com/patan3saro/PhDThesis`

For convenience, a QR code linking directly to the repository is provided below. When the document is printed, scanning the code grants immediate access to the most recent updates and supplementary resources.

QR code linking to the GitHub repository.

The repository may contain updated numerical notebooks, replication material, extended results, errata, and answers to questions commonly raised during seminars or peer-review.

Bibliography

[1] Rajeswari Chengoden, Nancy Victor, Thien Huynh-The, Gokul Yenduri, Rutvij H. Jhaveri, Mamoun Alazab, Sweta Bhattacharya, Pawan Hegde, Praveen Kumar Reddy Maddikunta, and Thippa Reddy Gadekallu. Metaverse for healthcare: A survey on potential applications, challenges and future directions. *IEEE Access*, 11:12765–12795, 2023. doi: 10.1109/ACCESS.2023.3241628.

[2] Connected vehicles to surpass 367 million globally by 2027 as 5g opens up data-intensive use cases. 5G.hr news article, 2023. URL https://www.5g.hr/en/news/connected-vehicles-to-surpass-367-million-globally-by-2027-as-5g-opens-up-data-intensive-use-cases/. Accessed Sep. 2025.

[3] Saurabh Jambotkar, Longxiang Guo, and Yunyi Jia. Adaptive optimization of autonomous vehicle computational resources for performance and energy improvement. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7594–7600, 2021. doi: 10.1109/IROS51168.2021.9635828.

[4] Niroshinie Fernando, Seng W. Loke, and Wenny Rahayu. Mobile cloud computing: A survey. *Future Generation Computer Systems*, 29(1):84–106, 2013. ISSN 0167-739X. doi: 10.1016/j.future.2012.05.023. URL https://www.sciencedirect.com/science/article/pii/S0167739X12001318. Including Special section: AIRCC-NetCoM 2009 and Special section: Clouds and Service-Oriented Architectures.

[5] Yuyi Mao, Changsheng You, Jun Zhang, Kaibin Huang, and Khaled B. Letaief. A survey on mobile edge computing: The communication perspective. *IEEE Communications Surveys & Tutorials*, 19(4):2322–2358, 2017. doi: 10.1109/COMST.2017.2745201.

[6] Verizon. Ip latency statistics. https://www.verizon.com/business/terms/latency/, 2025. Retrieved April 7, 2025.

[7] EE Times Europe. Why edge computing and 6g will continue to be 'aspirations'. https://www.eetimes.eu/why-edge-computing-and-6g-will-continue-to-be-aspirations/, 2024. Accessed: 2025-08-25.

[8] ITPro Today. Edge computing trends: Adoption, challenges, and future outlook. https://www.itprotoday.com/edge-computing/edge-

computing-trends-adoption-challenges-and-future-outlook, 2023. Accessed: 2025-08-25.

[9] Cisco and Analysys Mason. Edge computing: Operator strategies, use cases and implementation. https://www.cisco.com/c/en/us/solutions/service-provider/edge-computing.html, 2022. Accessed: 2025-08-25.

[10] NVIDIA. Nvidia tesla v100: The first tensor core gpu. https://www.nvidia.com/en-gb/data-center/tesla-v100/, 2025. Retrieved February 11, 2025.

[11] Maanak Gupta, James Benson, Farhan Patwa, and Ravi Sandhu. Secure v2v and v2i communication in intelligent transportation using cloudlets. *IEEE Transactions on Services Computing*, 15(4):1912–1925, 2022. doi: 10.1109/TSC.2020.3025993.

[12] Rosario Patanè, Andrea Araldo, Tijani Chahed, Diego Kiedanski, and Daniel Kofman. Coalitional game-theoretical approach to coinvestment with application to edge computing. In *2023 IEEE 20th Consumer Communications & Networking Conference (CCNC)*, pages 517–522, 2023. doi: 10.1109/CCNC51644.2023.10060093.

[13] Hongli Zhang, Qiang Zhang, and Xiaojiang Du. Toward vehicle-assisted cloud computing for smartphones. *IEEE Transactions on Vehicular Technology*, 64(12):5610–5618, 2015. doi: 10.1109/TVT.2015.2480004.

[14] Rosario Patanè, Nadjib Achir, Andrea Araldo, and Lila Boukhatem. Can vehicular cloud replace edge computing? In *WCNC 2024 - IEEE Wireless Communications and Networking Conference*, Dubai, United Arab Emirates, April 2024.

[15] Muhammad Atif Ur Rehman, Muhammad Salah ud din, Spyridon Mastorakis, and Byung-Seo Kim. Foggyedge: An information-centric computation offloading and management framework for edge-based vehicular fog computing. *IEEE Intelligent Transportation Systems Magazine*, 15(5):78–90, 2023. doi: 10.1109/MITS.2023.3268046.

[16] 3rd Generation Partnership Project (3GPP). Service requirements for the 5g system; stage 1. 3GPP TS 22.261 v19.5.0. Retrieved 2025, from https://www.3gpp.org/ftp/Specs/archive/22_series/22.261/, 2025.

[17] Nitinder Mohan, Lorenzo Corneo, Aleksandr Zavodovski, Suzan Bayhan, Walter Wong, and Jussi Kangasharju. Pruning edge research with latency shears. In *Proceedings of the 19th ACM Workshop on Hot Topics in Networks*,

HotNets '20, page 182–189, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450381451. doi: 10.1145/3422604.3425943. URL https://doi.org/10.1145/3422604.3425943.

[18] N. Afraz and M. Ruffini. A sharing platform for multi-tenant pons. *Journal of Lightwave Technology*, 2018.

[19] D. Kiedanski, A. Orda, and D. Kofman. Discrete and stochastic coalitional storage games. In *ACM e-Energy*. ACM, 2020.

[20] W. Saad, Z. Han, M. Debbah, A. Hjorungnes, and T. Basar. Coalitional game theory for communication networks. *IEEE Signal Processing Magazine*, 2009.

[21] K. J. Salchow, Jr. Clustered multiprocessing: Changing the rules of the performance game. White paper, F5, 2008.

[22] M. Sereno. Cooperative game theory framework for energy efficient policies in wireless networks. In *International Conference on Future Systems*, 2012.

[23] M. J. Osborne and A. Rubinstein. *A Course in Game Theory*. Massachusetts Institute of Technology, 1994.

[24] F. Guillemin and V. Rodriguez. Evaluating the impact of tower companies on the telecommunications market. In *IEEE International Conference on Innovation in Clouds, Internet and Networks (ICIN)*. IEEE, March 2021. doi: 10.1109/ICIN51074.2021.9385547. URL https://ieeexplore.ieee.org/document/9385547/.

[25] R. T. B. Ma, J. C. S. Lui, and V. Misra. Evolution of the internet economic ecosystem. *IEEE/ACM Transactions on Networking*, 2015.

[26] Lloyd S. Shapley. Cores of convex games. *International Journal of Game Theory*, 1(1):11–26, 1971. doi: 10.1007/BF01753431.

[27] T. Driessen. *Cooperative Games, Solution and Applications*. Kluwer Academic Publisher, 1988.

[28] R. van den Brink. Null or nullifying players: the difference between the shapley value and equal division solutions. *Journal of Economic Theory*, 2007.

[29] B. Sylvain, E. Remila, and P. Solal. Veto players, the kernel of the shapley value and its characterization. pre-print, 2014.

[30] A. P. Vela, A. Via, F. Morales, M. Ruiz, and L. Velasco. Traffic generation for telecom cloud-based simulation. In *International Conference on Transparent Optical Networks (ICTON)*, 2016.

[31] Boonyarith Saovapakhiran and Michael Devetsikiotis. Enhancing computing power by exploiting underutilized resources in the community cloud. In *2011 IEEE International Conference on Communications (ICC)*, pages 1–6, 2011. doi: 10.1109/icc.2011.5962544.

[32] Sohan Singh Yadav and Zeng Wen Hua. Cloud: A computing infrastructure on demand. In *2010 2nd International Conference on Computer Engineering and Technology*, volume 1, pages V1–423–V1–426, 2010. doi: 10.1109/ICCET.2010.5486068.

[33] AMAZON. The aws blog: The first five years. https://aws.amazon.com/blogs/aws/aws-blog-the-first-five-years/, 2009. Retrieved April 7, 2025.

[34] Google Cloud Platform. Your next home in the cloud. https://cloud.google.com/blog/products/gcp/google-cloud-platform-your-next-home-in-the-cloud/?hl=en, 2025. Retrieved April 7, 2025.

[35] Zongkai Liu, Penglin Dai, Huanlai Xing, Zhaofei Yu, and Wei Zhang. A distributed algorithm for task offloading in vehicular networks with hybrid fog/cloud computing. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 52(7):4388–4401, 2022. doi: 10.1109/TSMC.2021.3097005.

[36] Arif Ahmed and Ejaz Ahmed. A survey on mobile edge computing. In *2016 10th International Conference on Intelligent Systems and Control (ISCO)*, pages 1–8, 2016. doi: 10.1109/ISCO.2016.7727082.

[37] Li Lin, Xiaofei Liao, Hai Jin, and Peng Li. Computation offloading toward edge computing. *Proceedings of the IEEE*, 107(8):1584–1607, 2019. doi: 10.1109/JPROC.2019.2922285.

[38] Mario Gerla. Vehicular cloud computing. In *2012 The 11th Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net)*, pages 152–155, 2012. doi: 10.1109/MedHocNet.2012.6257116.

[39] NVIDIA Danny Shapiro. Tesla unveils top av training supercomputer powered by nvidia a100 gpus. https://blogs.nvidia.com/blog/2021/06/22/tesla-av-training-supercomputer-nvidia-a100-gpus/, 2023. Retrieved 7 April 2025.

[40] Ars Technica. Fcc takes spectrum from auto industry in plan to "supersize" wi-fi. https://arstechnica.com/tech-policy/2020/11/fcc-adds-45mhz-to-wi-fi-promising-supersize-networks-on-5ghz-band/, 2020. Retrieved 18 June 2025.

[41] Mohammad Aazam, Saif ul Islam, Salman Tariq Lone, and Assad Abbas. Cloud of things (cot): Cloud-fog-iot task offloading for sustainable internet of things. *IEEE Transactions on Sustainable Computing*, 7(1):87–98, 2022. doi: 10.1109/TSUSC.2020.3028615.

[42] Jaber Almutairi and Mohammad Aldossary. A novel approach for iot tasks offloading in edge-cloud environments. *Journal of Cloud Computing*, 10(1):28, 2021. doi: 10.1186/s13677-021-00243-9.

[43] Marco Malinverno, Josep Mangues-Bafalluy, Claudio Ettore Casetti, Carla Fabiana Chiasserini, Manuel Requena-Esteso, and Jorge Baranda. An edge-based framework for enhanced road safety of connected cars. *IEEE Access*, 8:58018–58031, 2020. doi: 10.1109/ACCESS.2020.2980902.

[44] Xiang Ju, Shengchao Su, Chaojie Xu, and Haoxuan Wang. Computation offloading and tasks scheduling for the internet of vehicles in edge computing: A deep reinforcement learning-based pointer network approach. *Computer Networks*, 223:109572, 2023. doi: 10.1016/j.comnet.2023.109572.

[45] Yuxuan Sun, Xueying Guo, Jinhui Song, Sheng Zhou, Zhiyuan Jiang, Xin Liu, and Zhisheng Niu. Adaptive learning-based task offloading for vehicular edge computing systems. *IEEE Transactions on Vehicular Technology*, 68(4):3061–3074, 2019. doi: 10.1109/TVT.2019.2895593.

[46] Fangming Liu, Jian Chen, Qixia Zhang, and Bo Li. Online mec offloading for v2v networks. *IEEE Transactions on Mobile Computing*, 22(10):6097–6109, 2023. doi: 10.1109/TMC.2022.3186893.

[47] Ayoub Ben Ameur, Andrea Araldo, and Francesco Bronzino. On the deployability of augmented reality using embedded edge devices. In *2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC)*, pages 1–6, 2021. doi: 10.1109/CCNC49032.2021.9369590.

[48] R. A. Isaac, P. Sundaravadivel, V. S. N. Marx, et al. Enhanced novelty approaches for resource allocation model for multi-cloud environment in vehicular ad-hoc networks. *Scientific Reports*, 15:9472, 2025. doi: 10.1038/s41598-025-93365-y.

[49] Hengwei Liu, Ni Tian, Deng-Ao Song, and Long Zhang. Digital twin-enabled multi-service task offloading in vehicular edge computing using soft actor-critic. *Electronics*, 14(4), 2025.

[50] Chen Chen, Yini Zeng, Huan Li, Yangyang Liu, and Shaohua Wan. A multihop task offloading decision model in mec-enabled internet of vehicles. *IEEE Internet of Things Journal*, 10(4):3215–3230, 2023. doi: 10.1109/JIOT.2022.3143529.

[51] Xinran Zhang, Mugen Peng, Shi Yan, and Yaohua Sun. Joint communication and computation resource allocation in fog-based vehicular networks. *IEEE Internet of Things Journal*, 9(15):13195–13208, 2022. doi: 10.1109/JIOT.2022.3140811.

[52] Kai Cui, Mustafa B. Yilmaz, Anam Tahir, Anja Klein, and Heinz Koeppl. Optimal offloading strategies for edge-computing via mean-field games and control. In *IEEE GLOBECOM*, pages 976–981, Rio de Janeiro, Brazil, 2022. doi: 10.1109/GLOBECOM48099.2022.10001412.

[53] Fabio Giust, Gianluca Verin, et al. Mec deployments in 4g and evolution towards 5g. White paper no. 24, ETSI, February 2018.

[54] Azzedine Boukerche and Robson E. De Grande. Vehicular cloud computing: Architectures, applications, and mobility. *Computer Networks*, 135:171–189, 2018. ISSN 1389-1286. doi: https://doi.org/10.1016/j.comnet.2018.01.004.

[55] William Tärneberg et al. Resource management challenges for the infinite cloud. In *10th International Workshop on Feedback Computing at CPSWeek*, 2015.

[56] Shahin Vakilinia, Mustafa Mehmet Ali, and Dongyu Qiu. Modeling of the resource allocation in cloud computing centers. *Computer Networks*, 91: 453–470, 2015.

[57] Amitabha Ghosh, Andreas Maeder, Matthew Baker, and Devaki Chandramouli. 5g evolution: A view on 5g cellular technology beyond 3gpp release 15. *IEEE Access*, 7:127639–127651, 2019. doi: 10.1109/ACCESS.2019.2939938.

[58] REM Maps. 5g-lena. https://5g-lena.cttc.es/features/rem/, 2025. Retrieved April 7, 2025.

[59] 3GPP. 5g; service requirements for enhanced v2x scenarios (ts 22.186 v16.2.0 release 16). Technical Report TS 122 186 V16.2.0, ETSI, November

2020. URL https://www.etsi.org/deliver/etsi_ts/122100_122199/122186/16.02.00_60/ts_122186v160200p.pdf.

[60] ETSI. Intelligent transport systems (its); vehicular communications; basic set of applications; part 2: Specification of cooperative awareness basic service. Technical Report EN 302 637-2 V1.4.1, ETSI, April 2019. URL https://www.etsi.org/deliver/etsi_en/302600_302699/30263702/01.04.01_60/en_30263702v010401p.pdf.

[61] Dawei Wei, Junying Zhang, Mohammad Shojafar, Saru Kumari, Ning Xi, and Jianfeng Ma. Privacy-aware multiagent deep reinforcement learning for task offloading in vanet. *IEEE Transactions on Intelligent Transportation Systems*, 24(11):13108–13122, 2023. doi: 10.1109/TITS.2022.3202196.

[62] Alessandro Zanni, Se-Young Yu, Paolo Bellavista, Rami Langar, and Stefano Secci. Automated selection of offloadable tasks for mobile computation offloading in edge computing. In *2017 13th International Conference on Network and Service Management (CNSM)*, pages 1–5, 2017. doi: 10.23919/CNSM.2017.8256026.

[63] GitLab Repository. 5g-lena. https://gitlab.com/cttc-lena/nr/-/tree/master/examples, 2024. Retrieved June 18, 2025.

[64] Spiceworks. What is wifi 6? meaning, speed, features, and benefits. https://www.spiceworks.com/tech/networking/articles/what-is-wifi-six/, 2022. Retrieved April 7, 2025.

[65] M. Chiappetta. Amd threadripper 3990x review: A 64-core multithreaded beast unleashed. https://hothardware.com/reviews/amd-ryzen-threadripper-3990x-cpu-review, February 2020. Retrieved April 7, 2025.

[66] M. Chiappetta. Amd ryzen 9 3950x review: A 16-core zen 2 powerhouse. https://hothardware.com/reviews/amd-ryzen-9-3950x-zen-2-review, November 2019. Retrieved April 7, 2025.

[67] Wikipedia. Instructions per second. https://en.wikipedia.org/wiki/Instructions_per_second, 2025. Retrieved April 7, 2025.

[68] Kaneez Fizza, Nitin Auluck, and Akramul Azim. Improving the schedulability of real-time tasks using fog computing. *IEEE Transactions on Services Computing*, 15(1):372–385, 2022. doi: 10.1109/TSC.2019.2944360.

[69] Syed Sahil Abbas Zaidi, Mohammad Samar Ansari, Asra Aslam, Nadia Kanwal, Mamoona Asghar, and Brian Lee. A survey of modern deep

learning based object detection models. *Digital Signal Processing*, 126: 103514, 2022. doi: 10.1016/j.dsp.2022.103514.

[70] G. de Préville. Paris: la vitesse moyenne d'un automobiliste en journée est de 13,1 km/h selon une étude. *Le Figaro*, October 2021. https://www.lefigaro.fr/societes/paris-la-vitesse-moyenne-d-un-automobiliste-en-journee-est-de-13-1-km-h-selon-une-etude-20211012. Retrieved June 18, 2025.

[71] CTTC. 5g-lena. https://5g-lena.cttc.es/, 2025. Retrieved April 7, 2025.

[72] SUMO Project. Sumo. https://eclipse.dev/sumo/, 2025. Retrieved April 7, 2025.

[73] Manhattan-SUMO Documentation. Manhattan. https://sumo.dlr.de/docs/Tutorials/Manhattan.html, 2025. Retrieved April 7, 2025.

[74] Wendlasida Ouedraogo, Andrea Araldo, Badii Jouaber, Hind Castel, and Remy Grunblatt. Can edge computing fulfill the requirements of automated vehicular services using 5g network? In *2024 IEEE 99th Vehicular Technology Conference (VTC2024-Spring)*, pages 1–5, Singapore, 2024. doi: 10.1109/VTC2024-Spring62846.2024.10683522.

[75] Yeongjin Kim, Hyang-Won Lee, and Song Chong. Mobile computation offloading for application throughput fairness and energy efficiency. *IEEE Transactions on Wireless Communications*, 18(1):3–19, 2019. doi: 10.1109/TWC.2018.2868679.

[76] Brian Benchoff. Pi 3 benchmarks: The marketing hype is true. https://hackaday.com/2016/03/01/pi-3-benchmarks-the-marketing-hype-is-true/, March 2016. Retrieved June 18, 2025.

[77] William Zimmer. Raspberry pi 5: Les premiers benchmarks dévoilent des performances en hausse. https://www.phonandroid.com/raspberry-pi-5-les-premiers-benchmarks-devoilent-des-performances-en-hausse.html, 2023. Retrieved April 7, 2025.

[78] Versus. Arm cortex-a73 review: specs and price. https://versus.com/en/arm-cortex-a73, 2024. Retrieved June 24, 2024.

[79] Unicornplatform. How long should a gpu actually last? plan for 3–5 years. https://unicornplatform.com/blog/how-long-should-a-gpu-actually-last-expect-3-5-years/, 2023. Retrieved April 7, 2025.

[80] Warehouse Technician. Warehouse technician salary in united states. https://www.indeed.com/career/warehouse-technician/salaries, 2025. Retrieved April 7, 2025.

[81] AWS. Aws lambda pricing. https://calculator.aws/#/addService/Lambda, 2025. Retrieved April 7, 2025.

[82] Neeraj Kumar, Sherali Zeadally, Naveen Chilamkurti, and Alexey Vinel. Performance analysis of bayesian coalition game-based energy-aware vm migration in vehicular mobile cloud. *IEEE Network*, 29(2):62–69, 2015. doi: 10.1109/MNET.2015.7064905.

[83] Yiming Xia et al. Location-aware and delay-minimizing task offloading in vehicular edge computing networks. In *Proc. IEEE GLOBECOM Workshops*, 2021. Extended abstract.

[84] Zemin Sun, Geng Sun, Yanheng Liu, Jian Wang, and Dongpu Cao. Bargain-match: A game theoretical approach for resource allocation and task offloading in vehicular edge computing networks. *IEEE Transactions on Mobile Computing*, 23(2):1655–1673, 2024. doi: 10.1109/TMC.2023.3239339.

[85] Haijun Zhang, Lizhe Feng, Xiangnan Liu, and Keping Long. Benders-based resource allocation for latency-constrained vehicular offloading. In *Proc. IEEE INFOCOM Workshops*, 2023. Short paper.

[86] Manzoor et al. Ahmed. A survey on vehicular task offloading: Classification, issues, and challenges. *Journal of King Saud University - Computer and Information Sciences*, 34(10):4135–4162, 2022. doi: 10.1016/j.jksuci.2022.05.016.

[87] Shaohua Wan, Xiang Li, Yuan Xue, Wenmin Lin, and Xiaolong Xu. Efficient computation offloading for internet of vehicles in edge computing-assisted 5g networks. *The Journal of Supercomputing*, 76:2518–2547, 2020.

[88] Mehdi Hosseinzadeh, Bruno Sinopoli, Ilya Kolmanovsky, and Sanjoy Baruah. Mpc-based emergency vehicle-centered multi-intersection traffic control. *IEEE Transactions on Control Systems Technology*, 31(1):166–178, 2023. doi: 10.1109/TCST.2022.3168610.

[89] Huijun Tang, Ming Du, Huaming Wu, and Pengfei Jiao. Link topology-adaptive offloading method on vehicular edge computing. In *IEEE INFOCOM 2024 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 01–02, 2024. doi: 10.1109/INFOCOMWKSHPS61880.2024.10620848.

[90] Zhenyu Zhou, Houjian Yu, Chen Xu, Yan Zhang, Shahid Mumtaz, and Jonathan Rodriguez. Dependable content distribution in d2d-based cooperative vehicular networks: A big data-integrated coalition game approach. *IEEE Transactions on Intelligent Transportation Systems*, 19(3):953–964, 2018. doi: 10.1109/TITS.2017.2771519.

[91] Jose Angel Leon Calvo and Rudolf Mathar. Connected vehicles coordination: A coalitional game-theory approach. In *2018 European Conference on Networks and Communications (EuCNC)*, pages 1–6, 2018. doi: 10.1109/EuCNC.2018.8442706.

[92] Hassan Pirhosseini, Edris Zarei, Mohammad Hossein Rezvani, Azadeh Pourkabirian, Achyut Shankar, Nebojsa Bacanin, Carsten Maple, and Wattana Viriyasitavat. Game-theoretic methods for edge/fog/cloud computation offloading: a systematic review. *Computing*, 107(166), 2025. doi: 10.1007/s00607-025-01515-x.

[93] Feng Luo, Yi Ren, Yanhua Yu, Yunpeng Li, Qin Liu, and Xiaobo Zhang. A centralized discovery-based method for integrating data distribution service and time-sensitive networking for in-vehicle networks. *Ad Hoc Networks*, 178:103950, 2025. ISSN 1570-8705. doi: https://doi.org/10.1016/j.adhoc.2025.103950. URL https://www.sciencedirect.com/science/article/pii/S1570870525001982.

[94] Rosario Patanè, Andrea Araldo, Nadjib Achir, and Lila Boukhatem. Vehicular cloud computing: A cost-effective alternative to edge computing in 5g networks. *Computer Networks*, page 111365, 2025. ISSN 1389-1286. doi: https://doi.org/10.1016/j.comnet.2025.111365. URL https://www.sciencedirect.com/science/article/pii/S1389128625003329.

[95] GEO. Géant de l'IA, Nvidia dévoile un ordinateur miniature 1000 fois plus puissant qu'un PC classique. https://www.geo.fr/sciences/geant-de-l-ia-nvidia-devoile-un-ordinateur-miniature-1000-fois-plus-puissant-qu-un-pc-classique-224030, January 2025. Retrieved June 2, 2025.

[96] Rosario Patanè. Greencomputing: Code repository. https://github.com/patan3saro/GreenComputing, 2025. Accessed August 2025.

[97] Juha Nuutinen, Marja Matinmikko-Blue, Moran Beniamini, and Greta Januskaitiene. A comparison of the energy consumption of broadband data transfer technologies. *Journal of Green Communications*, 5(2):45–58, 2021.

[98] iGR and Mavenir. Us mobile operators could save $546 million with edge computing, 2021. https://www.fierce-network.com/telecom/study-carriers-can-rake-cash-savings-by-using-mec-for-offloading-traffic.Retrieved July 2, 2025.

[99] Tesla, Inc. Vehicle power consumption, 2025. URL https://www.tesla.com/en_gb/support/power-consumption. Retrieved June 30, 2025.

[100] Aarian Marshall. Why your electric car's range drops in the cold—and what to do about it. https://www.wired.com/story/ev-battery-drain-tips, 2023. URL https://www.wired.com/story/ev-battery-drain-tips. Retrieved July 3, 2025.

[101] U.S. Department of Energy. Electric vehicle battery drains: Myth or fact? https://www.energy.gov/energysaver/electric-vehicle-battery-drains, 2021. URL https://www.energy.gov/energysaver/electric-vehicle-battery-drains. Retrieved July 3, 2025.

[102] U.S. Energy Information Administration. Average price of electricity to ultimate customers: Residential sector. Electric Power Monthly, 2025. URL https://www.eia.gov/electricity/monthly/epm_table_grapher.php?t=epmt_5_3. Retrieved June 30, 2025.

[103] New mobility patterns study: insights into passenger mobility and urban logistics. News article, Directorate-General for Mobility and Transport (European Commission), 2022. URL https://transport.ec.europa.eu/news-events/news/new-mobility-patterns-study-insights-passenger-mobility-and-urban-logistics-2022-12-20_en. Accessed Nov. 2025.

[104] Viet Nguyen-Tien, Chengyu Zhang, Eric Strobl, and Robert J.Ñ. Elliott. The closing longevity gap between battery electric vehicles and internal combustion vehicles in great britain. *Nature Energy*, 10:354–364, 2025. doi: 10.1038/s41560-024-01698-1. URL https://doi.org/10.1038/s41560-024-01698-1.

[105] RTE. Bilan Électrique 2024, 2025. URL https://www.rte-france.com. Average French carbon intensity 57 g CO_2/kWh.

[106] European Environment Agency. Greenhouse gas intensity of electricity generation in europe - 2024, 2025. URL https://www.eea.europa.eu. EU-27 average 242 g CO_2/kWh. Retrieved June 18, 2025.

[107] U.S. Environmental Protection Agency. egrid 2024 summary tables, 2025. URL https://www.epa.gov/egrid. U.S. average grid 369 g CO_2/kWh. Retrieved June 18, 2025.

[108] International Energy Agency. Electricity information 2024 - china, 2024. URL https://www.iea.org. Chinese grid average 581 g CO_2/kWh. Retrieved June 18, 2025.

[109] Romain Lorenzini. Digital & environment: How to evaluate server manufacturing footprint, beyond greenhouse gas emissions? https://boavizta.org/en/blog/empreinte-de-la-fabrication-d-un-serveur, 2021. Retrieved July 4, 2025.

[110] Matt Kapko. Operators face unproven edge computing business models. SDxCentral News, March 2020. Available at: https://www.sdxcentral.com/news/operators-face-unproven-edge-computing-business-models/ (accessed Sep. 2025).

[111] 5g-mobix: Connected and automated mobility trials on cross-border corridors. Project website, 2025. URL https://www.5g-mobix.com/. Conducts 5G-enabled connected automated mobility trials across cross-border and urban sites; accessed September 2025.

[112] ITS America. V2x day one deployment district (downtown atlanta). ITS America website, 2024. URL https://itsa.org/day-one-deployment-district/. Operational C-V2X deployment demonstrating real-world V2X applications in urban traffic; accessed September 2025.

[113] GSMA. Operator platform telco edge requirements. GSMA Permanent Reference Document, 2023. URL https://www.gsma.com/solutions-and-impact/technologies/networks/gsma_resources/operator-platform-telco-edge-requirements-october-22/. Defines architecture and APIs for federated multi-operator edge platforms; accessed Sep. 2025.

[114] 5gcroco: 5g cross-border control for connected and automated mobility. 5G PPP European project, 2018–2022. URL https://5g-ppp.eu/5gcroco/. Trialled 5G technologies for CAM across France, Germany, Luxembourg; accessed Sep. 2025.