Mobility-aware Software-Defined Service-Centric Networking for Service Provisioning in Urban Environments

Inaugural dissertation
of the Faculty of Science,
University of Bern

presented by
Diego Oliveira Rodrigues
from Brazil

Supervisor of the doctoral thesis:
Professor Dr. Torsten Braun
Institut für Informatik, Universitat Bern
Professor Dr. Leandro A. Villas
Institute of Computing, University of Campinas

Mobility-aware Software-Defined Service-Centric Networking for Service Provisioning in Urban Environments

Inaugural dissertation
of the Faculty of Science,
University of Bern

presented by
Diego Oliveira Rodrigues
from Brazil

Supervisor of the doctoral thesis:
Professor Dr. Torsten Braun
Institut für Informatik, Universitat Bern
Professor Dr. Leandro A. Villas
Institute of Computing, University of Campinas

Accepted by the Faculty of Science.

Bern, June 2023

The Dean:
Prof. Dr. Marco Herwegh

Universidade Estadual de Campinas
Instituto de Computação

Diego Oliveira Rodrigues

Mobility-aware Software-Defined Service-Centric Networking for Service Provisioning in Urban Environments

Redes Definidas por Software, Orientadas a Serviços e Cientes da Mobilidade para provisão de Serviços em Ambientes Urbanos

CAMPINAS
2023

Diego Oliveira Rodrigues

Mobility-aware Software-Defined Service-Centric Networking for Service Provisioning in Urban Environments

Redes Definidas por Software, Orientadas a Serviços e Cientes da Mobilidade para provisão de Serviços em Ambientes Urbanos

Tese apresentada ao Instituto de Computação da Universidade Estadual de Campinas como parte dos requisitos para a obtenção do título de Doutor em Ciência da Computação no âmbito do acordo de Cotutela firmado entre a Unicamp e a University of Bern.

Thesis presented to the Institute of Computing of the University of Campinas in partial fulfillment of the requirements for the degree of Doctor in Computer Science under the double-diploma agreement between Unicamp and University of Bern.

Supervisor/Orientador: Prof. Dr. Leandro Aparecido Villas
Co-supervisor/Coorientador: Prof. Dr. Torsten Braun

Este exemplar corresponde à versão da Tese entregue à banca antes da defesa.

CAMPINAS
2023

Acknowledgements

I would like to start by acknowledging the love, patience, and understanding of my family that was invaluable to me during my Ph.D. course. In addition, I want to express my gratitude to my girlfriend, who was a source of inspiration and for listening to me when I needed to talk.

I also would like to thank my supervisors Prof. Leandro Villas, who supported my studies since my master's degree in the University of Campinas, and Prof. Torsten Braun, who invited me to joining the CDS group in the University of Bern. Their valuable feedback and suggestions have helped me to refine my ideas throughout the execution of the Ph.D. project. Besides them I would also like to thank Prof. Guilherme Maia and Prof. Antonio Loureiro that also provided assistance and support during my thesis research.

I would like to extend my gratitude to my colleagues from Laboratório de Redes de Computadores (LRC) and Communication and Distributed Systems (CDS) groups, many of which became friends for life. I appreciate the multiple celebration and conversation moments we had during the Ph.D.

I acknowledge the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) and the São Paulo Research Foundation (FAPESP) grants number #2018/12447-3 and #2020/11259-9 for the concession of funding important for the realization of this thesis. The opinions, hypotheses and conclusions or recommendations expressed in this material are the responsibility of the author and do not necessarily reflect the views of FAPESP and CAPES.

Finally, I would like to express my gratitude to the Holy Spirit of God that welcomed me at all times and never failed me when I needed Him the most.

Diego.

Abstract

Disruptive applications for mobile devices, such as the Internet of Things, Connected and Autonomous Vehicles, Immersive Media, and others, have requirements that the current Cloud Computing paradigm cannot meet. These unmet requirements bring the necessity to deploy geographically distributed computing architectures, such as Fog and Mobile Edge Computing. However, bringing computing close to users has its costs. One example of cost is the complexity introduced by the management of the mobility of the devices at the edge. This mobility may lead to issues, such as interruption of the communication with service instances hosted at the edge or an increase in communication latency during mobility events, e.g., handover. These issues, caused by the lack of mobility-aware service management solutions, result in degradation in service provisioning.

The present thesis proposes a series of protocols and algorithms to handle user and service mobility at the edge of the network. User mobility is characterized when user change access points of wireless networks, while service mobility happens when services have to be provisioned from different hosts. It assembles them in a solution for mobility-aware service orchestration based on Information-Centric Networking (ICN) and runs on top of Software-Defined Networking (SDN). This solution addresses three issues related to handling user mobility at the edge: (i) proactive support for user mobility events, (ii) service instance addressing management, and (iii) distributed application state data management. For (i), we propose a proactive SDN-based handover scheme. For (ii), we propose an ICN addressing strategy to remove the necessity of updating addresses after service mobility events. For (iii), we propose a graph-based framework for state data placement in the network nodes that accounts for user mobility and latency requirements.

The protocols and algorithms proposed in this thesis were compared with different approaches from the literature through simulation. Our results show that the proposed solution can reduce service interruption and latency in the presence of user and service mobility events while maintaining reasonable overhead costs regarding control messages sent in the network by the SDN controller.

List of Figures

List of Tables

List of Acronyms

3GPP The 3rd Generation Partnership Project
AP Access Point
API Application Programming Interface
ASUSD Application State and User Session Data
AZ Action Zone
BS Base Station
CAPES Coordenação de Aperfeiçoamento de Pessoal de Nível Superior
C-RNTI Cell Radio Network Temporary Identification
C-V2X Cellular Vehicle-to-Everything
CDF Cumulative Distribution Function
CDS Communication and Distributed Systems
CS Content Store
CTI Cell Timing Information
D-PBU Deregistration Proxy Binding Update
DC Data Cover
DIB Data Information Base
DMRA DMR Acknowledgment
DMR Data Movement Request
DNS Domain Name System
EC Edge Computing
EPC Evolved Packet Core
FAPESP São Paulo Research Foundation
FIB Forwarding Information Base
FlowFIB Flow Forwarding Information Base
FlowMod Flow Modification
FMF Follow Me Fog
GUID Globally Unique Identifiers
HIP Host Identity Protocol
HET Handover Execution Time

ICN Information-Centric Networking
IoT Internet of Things
LRC Laboratório de Redes de Computadores
L-SCN Layered SCN
LTE Long Term Evolution
MBB Make-Before-Break
MDC Minimal Data Cover
ML Machine Learning
MSCN Mobility-aware Service-Centric Networking
MSTB Minimum Spanning Tree within a Budget
NFN Named-Function Networking
NDN Named Data Networking
NDNS Named Data Networking architecture based on Software-Defined networks
NFV Netwrok Function Virtualization
OON Object-Oriented Networking
PBU Proxy Binding Update
PBA Proxy Binding Update Acknowledgement
PBR Proxy Binding Response
PIT Pending Interest Table
QoE Quality of Experience
QoS Quality of Service
RA Router Acknowledgment
RACH Random Access Channel
RRC Radio Resource Control
RRCC Connection Reconfiguration Complete
RS Router Solicitation
RSSI Received Signal Strength Indicator
RTD Round-Trip Delay
SCN Service-Centric Networking
SDN Software-Defined Networking
SD-ICN Software-Defined Information-Centric Networking
SD-MSCN Mobility-aware Software-Defined Service-Centric Networking
SGW Serving Gateway
SDT Service Disruption Time
SI Service Instance
SLA Service Level Agreements

SRMH	SDN-enabled RACH-less MBB Handover
TA	Timing Alignment
TTI	Transmission Time Interval
UE	User Equipment
p-BS	previous Base Station
t-BS	target Base Station

Contents

Chapter 1

Introduction

Disruptive Future Internet applications for mobile devices, such as Internet of Things (IoT), Autonomous Driving, Immersive Media, and others, have requirements that cannot be met by the current Cloud Computing paradigm [30]. Among other requirements, latency levels to access services hosted in the cloud may be prohibitive for many of these applications that need to consume multiple services within a few milliseconds [133]. Such requirements are challenging when considering services accessed via cellular networks, where hundreds of users are connected to Access Points (APs) and sharing the same communication channels.

1.1 Motivation

One solution widely considered to meet Future Internet application requirements, such as, reducing latency to consume services over the Internet, is to deploy geographically distributed computing platforms, such as Fog and Multi-access Edge Computing. However, different challenges still exist to enable services to run close to users at the edge of the network. For instance, when users connect to different APs due to their mobility, network configurations must be updated to maintain their connectivity in a process known as network handover. The handover is one fundamental operation when managing the mobility of users at edge networks. Besides users, the concept of mobility also applies to services that compose applications. Service mobility is achieved by deploying service instances in different hosts at the edge. A service instance is a running piece of software able to deliver to users the features comprised in the service. Multiple service instances can coexist that provide the same set of features, which can happen for different reasons, for instance, to scale up the service provisioning for a significant number of users or to extend a given availability zone of the service.

Service mobility can be triggered by different factors, such as resource allocation, energy saving, or due to the mobility of its users. Mobility-induced service migration moves a service instance running at an edge node far from a consumer to a closer node. However, the migration of services leads to disruption in provisioning while the service is not fully moved. Therefore, complex service orchestration strategies are required at the edge to achieve seamless mobility. This Ph.D. thesis aims to study mobility-aware latency-constrained service management strategies and to create algorithms and protocols to enhance current state-of-the-art related to user and service mobility management at the edge.

The main challenges related to the management of user and service mobility are related to user handover, network and service connection continuity, and application context management. The main shortcomings (S.1-S.3) related to these challenges in the literature about Edge Computing (EC) architectures for service provisioning are:

S.1 Absence of Proactive User Mobility Support: Current solutions that reactively trigger handovers and service migrations lead to longer execution times for these procedures, which results in disruptions of communication or high delays of service provisioning.

S.2 Service Instances Addressing Issues: Applications at the edge might consume services from multiple hosts. The number of services, the dynamicity of selecting different provider nodes, and the mobility of service consumers make traditional addressing solutions sub-optimal. Specifically, topology-based approaches, such as IP, must always be updated when the consumer connects to a different access point or consumes a service from a different host. While addresses and network paths are not updated, service provisioning is degraded because the communication messages cannot reach their destinations. This disruption in provisioning is worse in edge environments because of the high frequency of these events due to the mobility of the users and services.

S.3 Lack of Distributed Application State Management: Many applications use state data to control users configurations and preferences, or to maintain and reuse historical data that may change output to requests even when the same input is provided. State data may be stored in different formats, such as, in the main memory of a server or in a data management service. Specifically, services that can query, organize, and transform state data (e.g., Database, Queue) according to requests are used to manage it. At the edge, these services are awaited to be running and migrated across multiple hosts, raising issues on providing low latency access to state data.

1.2 Problem Statement and Research Questions

Emerging applications have targeted EC infrastructure to meet requirements, such as offloading tasks and reducing latency to consume services. However, there are still issues related to service provisioning at the edge. For instance, inadequate user and service mobility management leads to a decrease of Quality of Service (QoS) levels, such as disruptions and access latency raise.

Problem Statement. *The lack of mobility-aware latency-constrained service management at the edge leads to degradation of QoS levels in service provisioning, particularly the increase of communication latency.*

This degradation of service provisioning is due to the usage of architectures designed for the Internet before the advance of wireless technologies. Then, the Internet was mostly composed of static nodes, leading to reduced topology variation, and applications were not sensitive to delays. We aim to investigate questions and hypotheses related to service management to reduce disruptions and latency for consuming services at the edge. Also, we aim to diminish the necessity of service migrations and context transfers triggered by user mobility since these events significantly impact the QoS. We specifically target the use of Software-Defined Networking (SDN) to tackle mobility-related issues at the edge of the network. The global view of network entities enabled by this networking paradigm can ease the adoption of creative solutions and allow fine and active control of communication flows at the edge. Therefore, we target answering research questions Q.1, Q.2, and Q.3, which relate to the literature Shortcomings S.1, S.2, and S.3 respectively.

Q.1 How to orchestrate communication and service provisioning considering user mobility aspects?

Current reactive solutions to handle user and service mobility, such as the traditional handover procedure implemented in 3G and LTE networks, lead to a decrease of QoS for services running at the edge. Proactively triggering mobility-related network procedures, such as handovers and updates of communication flows, can significantly reduce the impact of mobility in the QoS observed by the users. We use the *global view of the network enabled by Software-Defined Networking to facilitate the adoption of proactive coordination and triggering of mobility management procedures.*

Q.2 How to handle network addressing updates after mobility events?

When mobile users change APs, often, they need to connect to different service instances, which are more suitable to serve consumers at the current position. Accessing a service from a different AP or consuming a service instance from a different host requires addressing updates to maintain communication and service continuity when using a topology-based addressing schema. Few studies have addressed how communication flows and addresses can be updated to maintain service continuity. These studies focus on, for instance, combining multiple network interfaces [195]. Thus, one interface can consume the service from the previous instance while the communication setup is done to a new instance. One possible technology to ease addressing management of mobile entities in the network is Information-Centric Networking (ICN). ICN-based protocols have several advantages to handle user's and service mobility [69] since addressing is not based on network topology. We studied how *fewer configuration updates, enabled by name-based addressing, can decrease service downtime and latency when users switch between different access points and service instances.*

Q.3 How to manage application state in mobile latency-constrained scenarios?

Multiple applications make use of state data in order to run. For instance, in web applications, state data can be stored in different formats, such as in the main memory of the host – e.g., users roles, and preferences –, or in database systems – e.g., data generated by the interactions of the users with the system. One example of state data in the Connected Vehicles scenario is the current state and sensor data of a set of vehicles. This data may need to be fast accessed by different services to, for instance, maneuver these vehicles in order to avoid or reduce damage that will be caused in an imminent crash. Other applications, such as immersive games, may use state data to control different in-game mechanics. This state data is separated from business logic to facilitate application management, allow shared state, and enhance scalability. This approach results in services only being responsible for handling the state data. Applications have requirements to access this state data through these services. These requirements become challenging to be met for services provided to mobile users in distributed environments. Nevertheless, we used *distributed state service management to aid in maintaining continuous access latency levels to state data in distributed mobile scenarios.*

One example of Application State and User Session Data (ASUSD) in the Connected Vehicles scenario is the current state and sensor data of a set of vehicles. This data may need to be fast accessed by different services to, for instance, maneuver these vehicles in order to avoid or reduce damage that will be caused in an imminent crash. Other applications, such as immersive games, may use ASUSD to control different in-game mechanics, i.e., the behavior of the interaction between different players, characters, and their environment. In both scenarios, low latency is of paramount importance, either for safety reasons or to maintain high Quality of Experience (QoE) levels. Even data stream processing engines, working as micro-services used as the basis to develop different applications, are sometimes required to maintain a state [42, 283].

1.3 Goals and Contributions

General Objective: *Design and evaluate a service management solution to mitigate provision disruption caused by the mobility of users and services at edge networks.*

While pursuing this goal, we expect to achieve specific objectives in three aspects: (i) SDN-enabled user and service mobility support, (ii) SDN-enabled Service-centric Networking, (iii) Application State Management. We developed algorithms and protocols to enhance service management platforms targeting on-premise and edge-hosted applications. Therefore, we have three specific objectives, each relating to a strategy developed and evaluated. These three objectives compose a solution for service provisioning in EC infrastructure.

O.1 User Mobility Management supported by SDN to proactively coordinate and trigger mobility management procedures reducing their impact on service provisioning. We developed an SDN foundation to support mobility at edge networks. This SDN foundation eases the access to important information to create proactive triggers for network procedures. Also, it facilitates the implementation of dynamic configuration changes in the network, thus studying the research question Q.1. While pursuing this objective, a RACH-less MBB Handover solution was proposed [204], which is described in details in Chapter 4.

O.2 Mobility-aware Software-defined Service-centric Networking for services to reduce the need for configuration updates when redirecting communication flows due to migration events. Transparent mobility capabilities of Service-Centric Networking (SCN) enabled by SDN was studied in the context of the research question Q.2. While studying this objective, Mobility-aware Software-Defined Service-Centric Networking (SD-MSCN) [205], a addressing solution to mitigate mobility-related networking issues was introduced. This solution is studied in Chapter 5.

O.3 Distributed Application State and User Session Management based on ICN pointers to data distributed in edge infrastructure. A graph-based algorithm for stateful data placement in the network was proposed [206] in the scope of research question Q.3 that reduces the dependency on data movement triggered by the mobility of the users. This solution is discussed in Chapter 6.

The contributions achieved pursuing each of these objectives were described and published in internationally recognized venues. The complete list of publications is shown in Section 7.3.

1.4 Overview of the Solution

During the Ph.D. course, we assembled different prominent technologies proposed for the Future Internet, such as EC, SDN, and ICN to build a mobility-aware latency-constrained service management platform. We design and evaluate a SD-MSCN solution for service provisioning at edge networks. We specifically focus on the provisioning of stateful services consumed by users in urban environments – i.e., conditioned to urban mobility patterns. When users move within a city, QoS levels achieved using EC tend to decrease and might reach poor values due to the distance between users and the host of services instances. To mitigate this issue, researchers in EC have introduced mobility-triggered service migration. This service migration type aims to relocate service instances to run closer to consumers, i.e., reducing the number of network hops required to consume the service. However, this solution still leads to service provisioning degradation, in terms of service interruption or increased latency, while the migration occurs. The proposed SD-MSCN solution in this thesis creates management strategies to reduce disruption caused by user mobility while also avoiding mobility-triggered service migration.

As discussed in Section 1.3, we have three specific objectives related to the provision of stateful services for mobile users. Each objective refers to one component of the service management solution depicted in Figure 1.1. The first component is called User Mobility Management supported by SDN and related to Objective O.1. Our studies target a scenario where cellular networks are controlled using SDN. Therefore, our first goal is to study the characteristics of SDN-enabled handovers in order to propose mechanisms to improve this process. For this component, we used modern handover scheme proposals from the literature and combined them with SDN features to introduce an SDN-enabled RACH-less MBB handover scheme described in Chapter 4. The second component of our solution is an software-defined service-centric addressing scheme. This component relates to Objective O.2. One relevant strategy to address mobility at the edge of the network is the separation of addressing and network topology, which allows addressing network entities regardless of their position in the network. In this component, we introduce an addressing scheme that uses named-based routing inspired by ICN and SCN that allows this separation. Chapter 5 discusses the concept of Mobility-aware Service-Centric Networking (MSCN) and proposes an implementation on top of SDN. Finally, the third component of our solution is Distributed Application State and User Session Management in the context of Objective O.3. This third component aims to explore how different placements of the state data in the network can reduce the degradation of the service provisioning when service mobility events are triggered. Chapter 6 describes a graph-based algorithm to identify this data placement considering the mobility of the users.

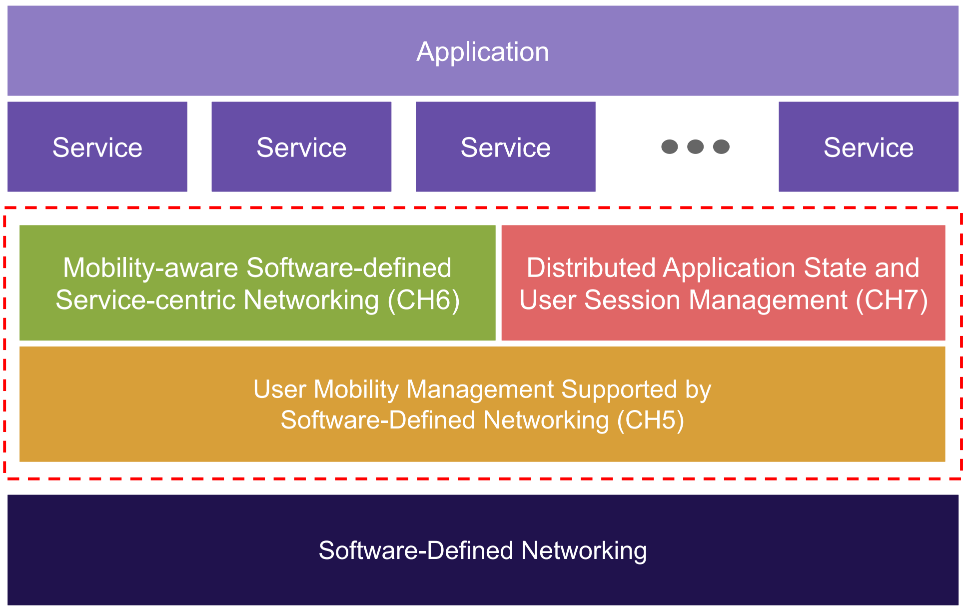

Figure 1.1: Overview of proposed SD-MSCN service management solution.

As depicted in Figure 1.1, the user mobility management component is a foundation for the other components. This happens because this component builds the topological user position knowledge for the SDN controller, which is then used by the other components. Since the SDN controller coordinates all handovers, it always knows the position of the users. This information can be used to improve other processes, such as the ones performed by the other components. This information is used in the second component to match User Equipment (UE) identifiers when routing and it is used to compute data placement in the third component. Furthermore, the distributed state management component uses the named-based addressing developed in SD-MSCN because it facilitates the consumption of the data from multiple sources and when this data has to be moved.

1.5 Thesis Structure

The remainder of this thesis is organized as follows. Chapter 2 introduces important concepts in the scope of this thesis, including EC, SDN, ICN, and mobility support in cellular networks. Then, Chapter 3 discusses works from the literature that have goals similar to the ones in this thesis. The complete solution proposed in this thesis has three components that are presented individually in Chapters 4, 5, and 6, as detailed in Section 1.4. Finally, final remarks and considerations of this thesis are given in Chapter 7.

Chapter 2

Literature Review

The present chapter presents an extensive review of the literature, published as a survey [207][1], of the three main subareas of this thesis: (i) Edge Computing (EC), (ii) mobility management at the edge, and (iii) services and applications supported by EC architectures. The chapter starts with an overview of the evolution of service provisioning over the Internet throughout the years in Section 2.1. EC architectures for provisioning of services for mobile devices are discussed in Section 2.2. Mobility management solutions for service provisioning at the edge are shown in Section 2.3. The services and applications that can take advantages of these architectures and solutions are described in Section 2.4. Furthermore, after examining enabling technologies, mobility solutions, and related applications, we identify some open challenges and research opportunities for service provisioning at the edge that are listed in Section 2.5. Finally, the lessons learned with this review of the literature and the final remarks of this chapter are given in Section 2.6.

2.1 History of Service Provisioning Technologies

Society has evolved into a state that continuous information exchange is required to improve citizens' lives in large urban centers. For instance, the advances achieved due to wireless communications and smartphone popularization make it hard to imagine a non-connected future. Indeed, the requirements for connectivity tend to increase with the emergence of new technologies such as IoT, Autonomous Vehicles, Immersive Media, and others. In this chapter we review the technologies developed to support these increasing requirements and also some envisioned applications that will take advantage of these technologies. Before, however, this section presents a brief historical overview of the computing and communication technologies for service provisioning. We organize this historical overview in four "stages" of evolution according to the placement of the computing resources: (1) Self-hosted Service Provisioning; (2) Cloud-based Service Provisioning; (3) Edge-based Service Provisioning; and (4) Ubiquitous Service Provisioning. Figure 2.1

[1]Partially reproduced in this chapter in accordance with the terms of the Creative Commons Attribution-NonCommercial 4.0 International Public License (CC BY-NC 4.0). https://creativecommons.org/licenses/by-nc/4.0/

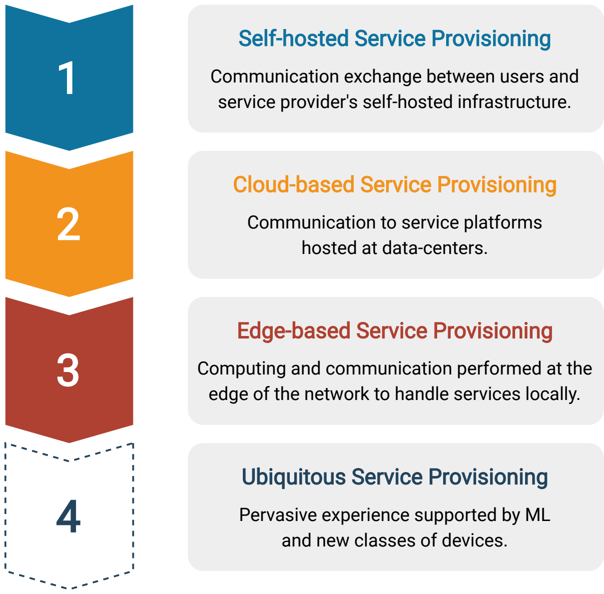

Figure 2.1: Evolution stages of service provisioning technologies.

illustrates these stages. The main cultural-technological shifts of each stage are discussed in Sections 2.1.1 to 2.1.4. Currently, technology and standards are being proposed and developed to fully achieve stage 3 of Edge-based service provisioning.

2.1.1 Self-hosted Service Provisioning

The Internet design, made in the 1970s, was created with a very different objective than today's Internet usage. By then, service providers would have to deploy and maintain their own infrastructure to provide their services, which characterizes stage 1 of self-hosted service provisioning. Users consumed services through emerging technologies of the time, such as the TCP/IP protocol. The IP protocol was designed to handle the addressing of hosts in a topology-based network and remains the main addressing protocol nowadays. The initial applications evolving by that time were e-mails, chat rooms, and later e-shopping and e-banking. However, such applications became very popular and started to face scalability issues. These concerns became the main reasons to push forward the service provisioning paradigm to the next stage.

2.1.2 Cloud-based Service Provisioning

The term Cloud Computing was initially used in the late 1990s [70], becoming broadly adopted by the late 2000s [259] when large companies start adopting it and pushing towards stage 2 of Cloud-based service provisioning. The idea was to offer companies the possibility to acquire computing resources that would scale on demand. Thus, virtually infinite computing resources were deployed to data centers strategically positioned around the globe. This paradigm is currently the main method for providing services on the Internet and allows companies to overcome some of the scalability issues previously faced. Nevertheless, scalability issues once again became a big concern with the massive adoption of applications such as Video Streaming, Gaming, and Social Media, i.e., resource-consuming applications accessed by millions of users. Different technologies were developed to handle the emerging 1990s-2000s applications and meet business requirements of resource and energy saving, and massive provision to millions of users. For instance, the platforms sold to companies to run their services should be virtualized to facilitate portability and application deployment. Consequently, service virtualization became an important research topic, with virtual machines, and more recently containers, being the main approach to provide Platform as a Service [60]. More complex solutions are still under development. Even network functions are being virtualized to handle these applications inside datacenters, using technologies such as NFV [110] and SDN [20].

In stage 2 of the evolution, we also experienced the popularization of different types of devices. Mobile devices became mainstream, such as tablets and smartphones, and later many other devices also started to be plugged into the Internet. Many envisioned applications are coming to reality due to the wide adoption of these devices. Such applications, however, are facing issues due to the 1970s topology-based design of the Internet. Therefore, we are again undergoing a paradigm shift in service provisioning.

2.1.3 Edge-based Service Provisioning

Multiple applications rely on low latency machine-type communication to run, thus, creating a need for faster communication and computation. Besides latency, the enormous amount of data generated by various devices connected to the Internet cannot constantly traverse significant distances, due to the risk of overloading the network infrastructure [215]. Furthermore, context-awareness became a requirement for the correct functioning of specific location-based applications. Therefore, the issues mentioned in Section 2.1.2, combined with the development of new applications supported by a wide range of devices, are driving service provisioning towards stage 3. In this stage, computing tasks run closer to end-users to maintain sustainable scalability. The main architectures of EC being studied to overcome those issues are Fog and Multi-access Edge Computing [208, 179]. These computing architectures propose the placement of computing resources closer to users, allowing services to execute locally and mitigate issues of latency, network overloading, and context awareness.

2.1.3.1 Fog and Multi-Access Edge Computing

Fog Computing is an extension of the Cloud Computing paradigm that expands the resource pool with resources from a plethora of devices, such as micro-datacenters, i.e., a smaller and self-contained category of a datacenter that comes in different sizes with cooling, security, and protection solutions out of the shelf. These computing facilities are deployed in spatially distributed Points of Presence (PoPs) to provide computational resources closer to users [208].

MEC [160] is a similar paradigm where computing tasks may run at resources closer to users. Both architectures aim to provide computation closer to the edge of the network, thus, causing Fog Computing and MEC to share many features. The two main differences between Fog Computing and MEC are [208]: (i) ownership: MEC is kept by telecommunications companies, while Fog Computing is typically maintained by private providers (e.g., AWS, Google); and (ii) deployment: MEC is only located at the edge of the network, while Fog Computing also uses resources placed strategically closer to end-users but not at the edge, for instance, data centers in neighbor cities/states or closer to the entrance of the core network (i.e., near-edge resources).

With the current development of 5G networks, partnerships between telecommunication companies and cloud infrastructure providers have been created to allow service provisioning at the edge. For example, Amazon Web Services (AWS) has established partnerships with multiple companies in different countries to create AWS Wavelength, a publicly available Multi-access Edge Computing (MEC) Infrastructure as a Service [252]. Still, EC-enabled applications have not yet been widely adopted. Emerging applications such as connected vehicles and AR are not freely available to the public in cellular networks. Despite the more controlled settings of EC-enabled applications in real-world deployments, research on EC technologies has been widely performed. The EC paradigm comprises the two main Edge Computing architectures, Fog Computing and MEC, and related architectures aimed to handle services closer to end-users.

Throughout the years, the definitions of Fog and MEC have evolved towards each other and many times, even in academic studies, these terms are used interchangeably. This evolution of the definitions is mostly due to two motivations. Firstly, a wider set of access technologies was envisioned for Multi-Access Edge Computing, which was previously called Mobile Edge Computing to highlight the usage of only mobile communication. And secondly, the expansion of the set of devices sometimes considered part of the Fog Computing infrastructure. These terms might be considered interchangeable in many aspects, being the most fundamental difference among them the focus of each architecture. Fog Computing studies target discussions about the infrastructure perspective and the placement of resources in the Cloud-Things continuum, whereas Edge Computing studies have their focus on the devices at the edge [224]. Devices discussed in Fog Computing may be placed not only at the edge, but also at the near-edge infrastructure, such as at the border to the core network or closer to the cloud. This distribution of resources often leads to the existence of a hierarchy of resources composing the Fog, with multiple layers closer or further to the end-user comprising more or less resources. Such hierarchy is not found when studying MEC, which is usually represented as a horizontal architecture.

In the EC stage of evolution, computing resources are deployed at the edge of the network, accessible through multiple access points and using different access technologies and channels. Wireless access points have limited coverage areas. Thus, when users move, they switch from one access point to another, causing a network handover, i.e., configuration updates in the network and user terminal to connect to the new access point. Current-state topology-based networking would struggle to handle multiple handovers while meeting the QoE requirements. Thus, research has been done to adapt networks for mobility-aware service provisioning. In this scenario, dynamically adaptable networks have an important role due to the ease of deploying new network algorithms and protocols to achieve better mobility management for connections.

2.1.3.2 Envisioned EC-enhanced Applications

Due to the possibilities envisioned by EC-enabled computing, developers started to foresee new classes of applications. The most discussed of these classes in the literature are: (i) Intelligent Transportation Systems (ITS), with applications for traffic management [256, 8], traffic lights control [151], and self-driving vehicles [228, 188]; (ii) Immersive Media [271], with applications in Augmented Reality (AR) or Virtual Reality (VR); (iii) Unmanned Aerial Vehicles (UAVs) [134, 284, 111] for monitoring or surveillance tasks; (iv) Smart Cities applications to handle public infrastructure [114, 137]; (v) other Internet of Things (IoT) related applications, such as healthcare [145], body area sensing with wearable and implantable devices [127], and also Industry 4.0 [98]. These applications will be widely adopted in the future, thus generating more requirements for EC-enabled scenarios. One of these requirements is mobility awareness since users with different mobility patterns will use many of these applications. Moreover, a composition of EC to run services and applications, and network management handled by SDN is envisioned as the Future Internet [210, 282, 181]. This vision is due to the possibility of using these technologies to deploy 5G networks. The challenges to fully achieve stage 3 of evolution are discussed in Section 2.5.

2.1.4 Ubiquitous Service Provisioning

Devices are evolving together with applications and adding a sense of pervasiveness to the Internet. Examples of these devices are smart glasses and head-mounted displays, some enabling movement-free access to immersive media virtually anywhere. These devices are key enablers on the information access revolution from desktop to smartphones and then toward freedom of form and location [53]. Other technologies such as wearable devices, body area and ultra-dense networks also contribute to such an increase in the sense of the pervasiveness of Internet services. To deploy future applications envisioned for these devices, such as Tactile Internet and Internet of Skills [21], the architecture for service provisioning will have to evolve into an envisioned stage 4 of service provisioning. In this stage, EC, extended with pervasive devices, relies on Machine Learning (ML) to enhance its context awareness, latency reduction, mobility support, and other capabilities. There is still a big technological gap for achieving fully immersive experiences and ubiquitous service provisioning, still the development of EC infrastructure and its architectures for seamless provisioning for mobile devices, as discussed in Section 2.2, are important steps towards reaching stage 4 of service provisioning.

2.2 EC Architectures for Seamless Service Provisioning for Mobile Devices

This section describes the envisioned Future Internet for massive service provisioning in EC-enabled settings while dealing with mobility. Figure 2.2 shows a three layer architecture composed by: *(i)* the edge; *(ii)* the core network; and *(iii)* the cloud. Multiple technologies interact with each other to support service provisioning in each layer. The

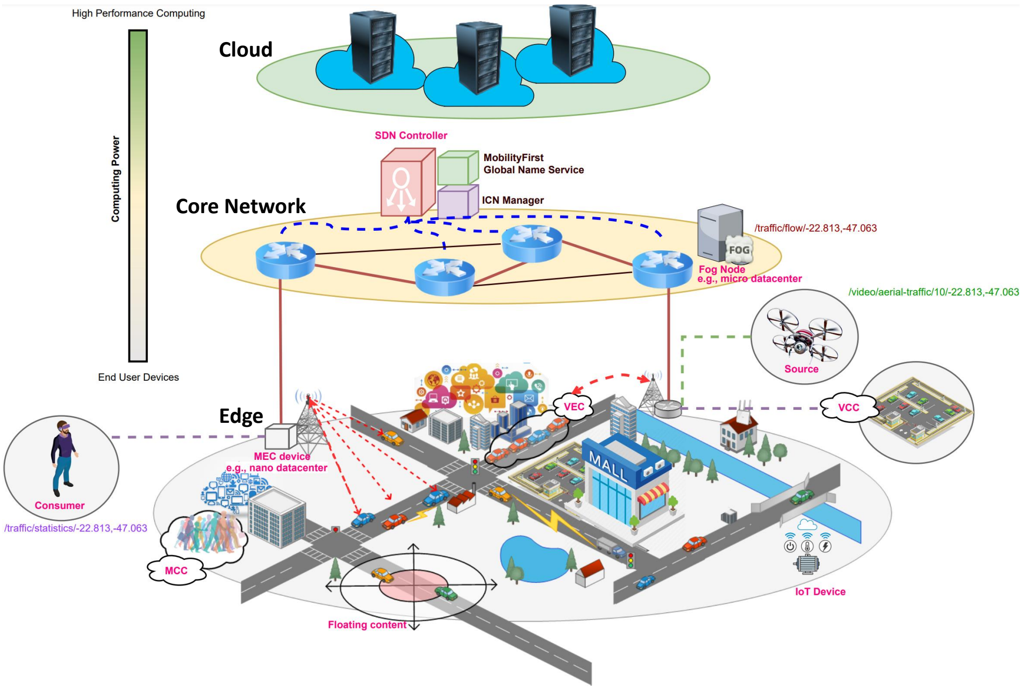

Figure 2.2: Landscape of Future Internet for massive service provisioning in EC-enabled settings.

bottom layer is the edge, where technologies such as Mobile Cloud Computing (MCC), Vehicular Cloud Computing (VCC), Vehicular Edge Computing (VEC), and Floating Content provide processing, storage and communication capabilities to run services. The core network layer in the middle manages networking and computing resources offered to users. This management is done by the SDN controller, which may use different abstractions and protocols for this task. Also, the core network layer enlarges the computing resource pool with near-edge Fog nodes, nodes deployed with more computing power than observed at the edge but not as far as the cloud. Finally, the highest computing power is provided at the cloud to handle eventual resource constraints of the lower layers.

The main difference between the layers in Figure 2.2 is the distance between users and resources. In Cloud Computing, a vast amount of resources is placed in data-centers usually positioned away from the users; while at the edge, the resources are distributed in smaller amounts closer to users. Furthermore, networking paradigms, such as SDN and ICN are expected to empower static networks and mobile networks such as Mobile Ad Hoc Networks (MANETs) and Vehicular Ad-hoc Networks (VANETs) and build the Future Internet. The computing architectures that compose EC are discussed in Section 2.2.1, while networking architectures are discussed in Section 2.2.2.

2.2.1 Computing Architectures

As smart edge devices become more popular, the IoT era emerges with the tendency of connecting these devices to the Internet to support different services and applications. Many of these devices are simple and resource constrained in terms of computing capabilities and power supply. Due to the observation that multiple services would demand more resources to execute, researchers developed offloading techniques to migrate their computation to the cloud [150]. Yet, relying only on the cloud has its drawbacks since datacenters are placed away from the edge devices and end-users.

2.2.1.1 Fog Computing

Cisco introduced the idea of Fog Computing in 2012 [36] with the objective of extending the cloud paradigm closer to people (i.e., end-users). This new computing architecture was aimed to be placed between the traditional cloud and the users in the form of datacenters positioned at the edge or near-edge of the network, before the gateway to the core network. Yet lately, the concept evolved to contemplate even idle computer resources in end-users Edge devices to be added to the pool [208]. This framework is built to provide features such as low latency, geo-distribution, location awareness, and mobility support [210]. These features aim to fill the gaps of the traditional Cloud Computing paradigm, thus creating a complementary Cloud-Fog Computing architecture, where the resources are selected according to the volume and speed of the processing tasks.

Fog Computing was considered an enabler technology for new classes of applications (e.g., Immersive Media, ITS) because they can take advantage of its features to tackle their constraints. However, to deploy such services, there are some challenges to be overcome. For instance, different protocols and APIs need to be established for services to access information from the network and sensors [208]. Also, mobility creates issues related to network management at the edge.

User mobility, handovers, and intermittent communication channels in general may disrupt service provisioning because of the difficulty to keep reliable, low-latency, and high-throughput links to send messages to Fog nodes placed at the near-edge of the network. One possible way of handling the mobility of the users is centralizing the handover control at the cloud [32]. This solution faces issues when a connection to the cloud data-centers fails. Thus, studies on handling mobility locally have emerged [179]. Yet, to execute more complex algorithms and enhance the quality of service in network handovers, programmable networks have been considered [210, 31]. These networks create a large set of possibilities due to the flexibility in using different routing protocols [178] and employing ML approaches to perform data-driven decisions [18]. Section 2.3 discusses the main possibilities found in the literature.

2.2.1.2 Multi-access Edge Computing

The MEC paradigm was proposed by ETSI in 2014 to use edge devices to enhance mobile devices capabilities through mobile cellular networks (e.g., 3G, 4G/LTE, 5G) [210, 161]. The concept of devices and connections was extended in MEC, which caused the terms Fog Computing and MEC computing to start to be used in an interchangeable fashion by the academic community [161]. Initially, MEC would consider only datacenters at the edge of the network deployed by telecommunication companies to offload processing tasks, and these devices should be accessed via the cellular network. Currently, even devices from end-users can be added to the MEC resource pool in some collaborative approaches. Different connectivity technologies can also be used to communicate to these resources.

To widely deploy MEC for massive usage, there are still some open issues to be addressed. Researchers have studied how to better place servers spatially to ease the coverage of wide areas [267]. For instance, UAV-mounted micro-servers can be used to provide in-locus additional resource [107]. Furthermore, to handle the vast amount of simultaneous users, techniques of MEC-enabled in-network caching have been proposed [69, 178]. The use of caching aims to take advantage of the high popularity of content and services by storing them in servers closer to users to reduce the necessity to load them from the cloud. User mobility causes several handover events in access networks, which is a complex task because of the many system configurations and policies to associate users and services [160]. For instance, according to some mobility management protocols [31, 189], the IP addresses of the users may have to change; also, a different host may be selected – according to a given policy – to execute services consumed by these users.

2.2.1.3 Mobile Cloud Computing

Another architecture aimed to enable the execution of computing-intensive tasks in resource-constrained mobile devices is Mobile Cloud Computing (MCC) [16]. This architecture advocates for offloading complete applications from Smart Mobile Devices to the cloud infrastructure, thus integrating mobile computing and Cloud Computing. This architecture differs from ones described in Sections 2.2.1.1 and 2.2.1.2 because it does not use other edge devices for offloading computing tasks.

2.2.1.4 Vehicular Edge Computing

An edge resource that has gained recent attention for task offloading is the vehicle [221, 180]. This attention is due to the large number of vehicles [19] and also the reasonable amount of computing resources on board both recently-released and upcoming vehicles [254]. These resources will be deployed in the form of On-Board Units (OBUs), which allow these vehicles to access network facilities. Due to the size of vehicles and its powerful batteries, when compared to other edge devices, these OBUs can be deployed with a significant amount of processing power. Building infrastructure based on Road Side Units (RSUs) is costly and may require additional effort with maintenance [221]. These costs could be reduced by using idle resources of vehicles. This idea supports the emergence of the Vehicular Edge Computing (VEC) paradigm – this paradigm is also referred to as Vehicular Fog Computing (VFC).

The VEC paradigm adds to the EC architecture the possibility of collaboratively use vehicular idle resources. These vehicles have the ability to capture data from nearby or remote environments and use it to run diverse applications. The data of the environment can be captured using Intra-vehicle communication with its own sensors (V2X), Inter-vehicle communication (V2V) to collect data from neighboring vehicles, or even Extra-vehicle communication (V2I and/or V2X), where data can be collected from RSUs, remote EC-enabled sensors or the cloud.

One way of building a VEC platform is by using VANETs. This approach is fully distributed, and nodes take decisions of sharing resources based on a limited view of the network status obtained from their neighborhoods. Studies considering using only

VANETs focus on deploying vehicular-centric protocols to address communication among vehicles [41] or integrate this network to cellular networks [119]. To achieve a better resource sharing solution, SDN is proposed to centralize the VANET control [3]. Studies on this field use RSUs-based VANETs [112], in which the SDN controller is placed in the RSUs. These controllers can also be hosted in some alternative infrastructure, such as UAVs [220, 219].

2.2.1.5 Vehicular Cloud Computing

MCC faced some issues when applied to a vehicular scenario because of the strategy of sending all data to be processed at the cloud. Vehicular applications depend on a great variety of sensors that collect large amounts of data and have to be processed in real, or near-real, time. Offloading the entire application to the cloud can create issues for the service provisioning because of the high volume of data that may be sent to the cloud. Observing the reasonable amount of computing resources envisioned on-board of future vehicles, the Vehicular Cloud Computing architecture (VCC) [23] was proposed as an extension for MCC. In this architecture, only some parts of the application are offloaded to the cloud, while others run in the vehicle itself. VCC is not fully able to cope with new application requirements in terms of delay and limited bandwidth as it relies on opportunistic communication with other vehicles, but part of this resources can be used in EC. To enable services to run in EC-enabled scenarios, approaches for network management, such as the ones presented in Section 2.2.2, have been proposed. Many of these techniques are expected to be used in conjunction to form a holistic platform and achieve the requirements established by the new generation of mobile networks [280].

2.2.2 Networking Architectures

Different communication architectures emerged in the literature to enable computing technologies to cooperate and form an environment for service provisioning [189, 55, 248, 282]. Many of these architectures envision a significant change on the basis of the Internet, such as virtualization of the network and a shift from a topology-based paradigm to new paradigms (e.g., information-centric). This section presents the main networking architectures proposed for the EC-enabled settings. To evolve the host-centric paradigm, researchers have proposed different paradigms, such as: (i) Geographical-based networking [135, 56, 90, 233, 55] which routes content according to geographical locations; (ii) Mobility-centric networking [248, 141], where mobile devices are addressed through unique identifiers generated to them; and (iii) Information-centric networking [135, 69, 282, 178, 61] that uses interest names of contents or services for routing. This section presents some of the characteristics of these paradigms, which are summarized in Table 2.1. More flexible network control via SDN is a trend to implement these paradigms on EC-enabled scenarios and is discussed in Section 2.2.2.1. Opportunistic and geographical-based networking is discussed in Sectino 2.2.2.2. Finally, different architectures for Future Internet are discussed in Section 2.2.2.3.

Table 2.1: Overview of main networking paradigms and protocols.

Paradigm	Addressing	Protocols
Host-centric	Numerical	IPv4
		IPv6
		MIPv6 [189]
		PMIPv6 [117]
		HMIPv6 [43]
Geo-centric	Geographic Coordinates	GRPI [105]
		BLR [95]
		GeOpps [135]
		GeoNetworking [240]
		FloatingContent [56]
		PFCS [90]
		GSOR [233]
		DGOR [55]
		CBF [67]
Future Internet	**Content/Service Identifiers**	HIP [175, 174]
		MobilityFirst [248, 141]
		CCN [104]
		NDN [281, 61]
		PUSUIT [75]
		SCN [38, 225]
		OON [147]
		NFN [243]

2.2.2.1 Software-Defined Networking

SDN has emerged as a networking paradigm to make networks more flexible and ease the adoption of new protocols and algorithms for network routing and the implementation of other network functions [200]. This paradigm shifts the routing complexity from the network routers to a centralized instance called controller, where a complete network overview gives several benefits [214], such as: (i) granular control of policies that can be oriented to sessions, users, devices, or services; (ii) easy and on-demand adaption to changes; and (iii) cost savings due to better resource management.

The idea of a global network view present in SDN was adapted from the telephone network, where it was shown to be a secure and cost-efficient strategy [200]. Major SDN deployments were only observed after the evolution of the programmable router switches and the emergence of the OpenFlow protocol [164]. This protocol is based on a three-layered separation of network entities: (i) the application layer with services and end-users; (ii) the infrastructure layer with hardware to support storage, connectivity, and computation; and (iii) a control layer responsible for the virtualization of the infrastructure and enable its control by the applications. OpenFlow is a south-bound API to control programmable switches; the SDN controller also provides a north-bound API for management to be used by the application layer. For instance, Frentic [89] is a north-bound API that abstracts the network management using the concept of slices. Pyretic [172] is another abstraction that uses a modular view of the network for management.

To handle all the expected traffic load exchanged at the edge of the network, different network traffic management tools were proposed to explore the centralized information maintained by SDN controllers [71, 193]. The programmability of the network enhances its flexibility because it allows not only adaptability and interoperability but it also opens space for innovation. This programmability also aids the process of intelligent management through the development of, for instance, efficient mobility management solutions [31], or the use of ML models to enhance networking [18].

The flexibility and interoperability provided by SDN can be observed in 5G networks research, mainly to integrate new technologies and services in the networks [130, 275]. SDN is also a key supporting technology to handle the scalability and complex management of IoT scenarios [210], where a vast number of heterogeneous devices need to be connected. The significant compatibility with other state-of-the-art technologies and applications makes SDN an important enabler for the next generation of networks.

Different protocols exist to allow SDN controllers to install communication flows in the network. The most well-known protocol is the OpenFlow [9], which is an Application Programming Interface (API) implemented by Openflow-enabled switches to receive messages from the controller that install flow rules in its forwarding tables. When using Openflow, every packet has a set of well-defined fields, e.g., IP source and destination addresses, which are used for routing. These fields are then checked and compared to rules installed in the switches in order to determine actions to be taken, such as outputting the packet to a given interface, forwarding it to yet another table inside the same switch, updating information in one field, drop the packet, and so on. Any packet that does not match the rules installed in the switch is forwarded to the controller in a Packet-In event. The controller will receive the packet, issue Flow Modification (FlowMod) messages to the switches to install new communication paths to match this packet in the network, and then generate a Packet-Out event to allow the packet to reach its destination. One example of switch flow tables is given in Table 2.2.

Table 2.2: Example flow table rules of a switch

Priority	**Match**	**Action**
	Flow Table 0	
MEDIUM	ether_type=IPv4, ip_proto=UDP	goto_table(1)
MEDIUM	ether_type=IPv4, ip_proto=TCP	goto_table(2)
LOW	ANY	OUTPUT=port_no(CONTROLLER)
	Flow Table 1	
HIGH	ipv4_dst=`0xABCABC`	OUTPUT=port_no(LOCAL)
MEDIUM	ipv4_dst=`0xDEFDEF`	OUTPUT=port_no(1)
LOW	ANY	OUTPUT=port_no(CONTROLLER)
	Flow Table 2	
HIGH	ipv4_dst=`0xABCABC`	OUTPUT=port_no(LOCAL)
MEDIUM	ipv4_dst=`0xDEFDEF`	OUTPUT=port_no(2)
LOW	ANY	OUTPUT=port_no(CONTROLLER)

Table 2.2 shows some example flow tables and rules installed on a switch. There are three flow tables: (i) Flow Table 0, an incoming classification table; (ii) Flow Table

1, a table responsible for forwarding UDP requests; and (iii) Flow Table 2, a table to forward TCP requests. After a packet arrives in a forwarding Openflow switch, it is sent to the first ingress table, Flow Table 0, in the example. This table classifies the packets according to the ether_type and ip_proto fields and sends the packet to be processed in another table. Flow Table 1 checks the IPV4_DST field to either forward the packet to the interface labeled as LOCAL or to the port labeled with the number 1. Similarly, Flow Table 2 checks the same field to either forward the packet to the LOCAL interface or the one labeled with the number 2. These interfaces might be physical or logical forwarding interfaces on the switch. Packets not matching any rule in the flow tables are sent to the controller. The switch may also include an egress processing composed of multiple tables similar to the ingress tables. All matching fields, possible values, actions, and other details of the protocol are described in the Openflow Switch Specification [182].

2.2.2.2 Geo-Centric Networking

An intuitive way of managing networks in the presence of mobility is through geographical coordinates. This class of protocols uses geographical coordinates of the destination to support routing decisions. For instance, Geographical Routing using Partial Information (GRPI) [105] is an approach where each node in the network uses partial network information about its neighborhood to route packets to the closest neighbor from the destination. The route is not fixed, and, thus, if a packet reaches a node that "knows" a better route (i.e., based on distances of neighbors to the destination) the packet is sent through that route. This approach composes a distributed routing algorithm since no single node is required to have an overview of the entire network, but only knows information about its neighbors. GRPI is meant to operate in Wireless Ad-Hoc Networks. However, the problem of node sparsity is not studied. Network sparsity is a critical issue for routing protocols in wireless mobile environments. Consequently, this issue drives most studies on geographical routing for mobile entities to focus on opportunistic routing, which explores the links created opportunistically by the mobility of nodes. There are many applications for wireless mobile networks that aim at disseminating content to specific geographical areas, such as accident notifications or traffic flow conditions. Therefore, Geo-Centric Networking (GCN) protocols aim to allow nodes to address geographical areas and distribute network messages within them.

Using opportunistic links to disseminate data, Geographical Opportunistic routing for vehicular networks (GeOpps) [135] is a protocol that focuses on delay-tolerant networks. These networks are used by applications that can run without a continuous network connection. In particular, this protocol enables content distribution in target areas without fixed infrastructure. This approach relies on the *store-and-forward* strategy, where mobile nodes receive the content and carry it to later on forward it to the next node. Distributed Geographical Opportunistic Routing (DGOR) [55] is a similar protocol that uses a different set of metrics to evaluate the link cost to select the forwarding path to send network packets.

It is worth noticing that these networks face scalability problems because most protocols rely on regularly sending messages to inform neighboring nodes of their existence and position in a process called beaconing. Therefore, scenarios with many nodes broadcasting beacon messages can create scalability issues, such as transmission interference. Beacon-less routing algorithm (BLR) [95] is a protocol designed to tackle such scalability issues in MANETs. In BLR, beacon messages are not used, since no information about the existence or position of neighbors is required. Instead, the protocol broadcasts data packets and uses a dynamic forwarding delay to ensure that only one node will forward the message. In this mechanism, every node computes a forwarding delay to send the data packet. The one that computes the shortest delay will send it first as a broadcast. Thus, the other nodes will receive it and cancel the forwarding of their own copies.

One important concept of Geo-centric Networking standardized by ETSI is Contention-Based Forwarding (CBF) [67, 78]. This is a distributed and scalable forwarding strategy based on interest regions aimed for mobile ad hoc networks. This strategy does not rely on acquiring information of the neighborhood of a node via beacons, instead the message carries information of a contention window where it should be spread. Once receiving this packet, every node uses this information and its own position to decide whether it should become a forwarding hop or not. To avoid flooding, the re-transmission of the message is also conditioned to probabilities and/or timeouts that are evaluated by the node. For instance, a node will only re-transmit a message if it does not receive a transmission of the same message from another node in a given timeout. Multiple extensions of CBF exist in the literature that target, for instance, supporting the strategy using infrastructure [28] or performing network congestion control [166].

GeoNetworking [240] is a store-and-forward protocol standardized by ETSI aimed at vehicular communication that uses the location of OBUs and RSUs to disseminate data. This protocol has two main features: geographical addressing and geographical forwarding. This addressing allows unicast, where geographical positions are used together with node identifiers to aid the routing. It also allows broadcast and multicast, which may be performed by geographical or topological routing. According to the type of addressing being used, different methods can be used. For instance, in topological broadcasts, the forwarding process uses a simple flooding approach. Unicast, on the other hand, uses an approach called *line forwarding*, which applies different heuristics to create a forwarding path from source to destination.

Floating Content [90] uses an epidemic model for broadcasting content in an anchor zone (AZ) by keeping the content stored in the vehicles interested in it. In particular, vehicles inside the AZ can access the content via opportunistic links with other vehicles inside the zone that is carrying the content. Floating Content can also take advantage of a centralized SDN-based approach [56]. SDN controllers, accessed via RSUs, can collect information from the moving vehicles and analyze this data to enhance Floating Content management.

It is worth noticing that while Geo-Centric Networking approaches address mobility issues in specific geographical zones, more complex and delay-critical applications require a higher level of quality of service to operate that sometimes cannot be obtained by opportunistic routing. Nevertheless, other networking paradigms can address these situations, such as Mobility-Centric and Information-Centric Networking, discussed hereafter.

2.2.2.3 Future Internet

Upon observing the mobility patterns of Internet users, the US National Science Foundation's Future Internet Architecture (NSF-FIA) project designed an architecture called MobilityFirst [248] in 2010. This architecture aims to produce a network protocol for scalable service provisioning in mobility scenarios. This protocol is based on Globally Unique Identifiers (GUID) for in-network elements. These identifiers are used to separate names from addresses and locations, thus easing mobility management. This protocol relies on a distributed Global Name Service (GNS) that maps GUIDs to addresses. The strategy is similar to the one used nowadays on the Internet, where domains are translated to addresses via the Domain Names System (DNS).

The idea of using a global view of the network is shared with SDN. These technologies could in fact be used in collaboration by adding the GNS module to run within the SDN Controller. For content distribution, MobilityFirst relies on a in-network caching scheme [279]. In this strategy, storage-aware routers are used to cache content along the path it makes from its source to the consumer; this scheme facilitates dealing with intermittent connections due to mobility. Services are held in a similar fashion by mapping the service URI to a GUID and then using the GNS to resolve the GUID to an address. While handling services, MobilityFirst suggests the usage of in-network caching to store dynamic data [139], which is unusual since dynamic data is supposed to change. Nevertheless, different caching strategies should be applied to handle services but not the same ones used for contents.

Another important protocol designed by the IETF is the Host Identity Protocol (HIP) [175, 174]. This protocol allows hosts to share IP-level states to facilitate service provisioning continuity despite changes in IP addresses. By establishing Host Identities, HIP decouples transport-layer logic from network-layer logic. This separation creates many possibilities for network-layer mobility management. IETF has specified a basic network-level host mobility protocol [96]. This protocol defines how to create message flows and also other procedures to achieve host mobility. It is important to notice that CCN/NDN, PURSUIT, HIP, and other protocols can operate together, creating possibilities to apply them in the best suitable use cases.

To serve users with named content that can be stored anywhere in the network, ICN is a paradigm that has gained attention in both academy [123, 268, 241, 57, 282, 178] and industry [177, 85, 103]. Due to its lack of information about the content's location in its naming/addressing, ICN has the potential to solve many issues associated with the Host-centric paradigm, such as mobility [69]. By replacing IP addresses by a naming-based scheme, ICN supports seamless mobility. A scalable content distribution in this scenario is achieved via the deployment of storage-aware routers throughout the network, augmenting the possibilities for caching and offloading the core network. Several architectures have been proposed, such as Content-Centric Networking (CCN) [104], Named Data Networking (NDN, an evolution of CCN) [281], and Publish-Subscribe Internet Routing Paradigm (PURSUIT, earlier called PSIRP) [75]. Although these three projects have similar objectives in terms of routing data based on its name, CCN and NDN advocate for hierarchical-based names to facilitate locating and sharing data. However, PURSUIT supports flat naming to allow a greater variety of naming approaches, in which names are organized in hierarchical scopes. This organization allows the constitution of information networks, similarly to IP topological sub-networks.

ICN is proposed to be a clean-slate paradigm, which means it demands infrastructure replacement. However, SDN is a promising future networking technology that can smooth this process of deployment of ICN [282]. The protocol could be implemented over the virtual network controlled by the SDN controller. Furthermore, the benefits of SDN to the current network paradigm also apply to ICN. ICN can take advantage of the global view of the network and of actively controlling communication flows.

MobilityFirst and ICN are expected to run in EC-enabled Future Internet and deal with mobility-related issues. When comparing these two approaches they show similar performance to support scalability and mobility requirements for IoT applications. MobilityFirst outperforms ICN in terms of control overhead [141]. However, ICN strategies focused on VANETs have gained more attention. For instance, RSU-assisted NDN (RA-NDN) [239] is a protocol that relies on RSUs to improve network connectivity in a VANET scenario. The protocol outperforms general ad-hoc communication in terms of data received, throughput, and reduction in total dissemination time and traffic load. Mobility in Vehicular NDN (MobiVNDN) [61] is a protocol to mitigate issues related to vehicular communication, such as broadcast storms, message redundancy, network partitions, reverse path partitioning, and content source mobility. This protocol has good performance when sharing wireless medium with multiple applications. Cooperative Caching with Mobility Prediction (COMP) [102] is a caching strategy that focus on reducing the impact of mobility in VNDN. COMP reduces access delay and increases cache hit ratio by adding cooperative caching to VNDN, which usually is non-cooperative. The cooperative caching uses RSU resources to run caching decision algorithms that allow the vehicles to cache globally popular data instead of making caching decisions only based on local data. This strategy clusters vehicles with similar mobility patterns to store content on their OBUs. Later, these vehicles can share the content among themselves since their links are more stable.

The expected new classes of services, such as IoT, Immersive Media, and Autonomous Driving have drawn attention to mobile service provisioning in EC-enabled settings, thus reflecting the emergence of service-oriented ICN. One strategy to allow services to take advantage of the in-networking caching of the ICN paradigm is allowing cached content to be transformed and serve the requests [38] (e.g., transcoding a cached video). This approach is named Service-Centric Networking (SCN). Some other strategies use naming schemes of ICN to facilitate service consumption. For instance, Layered SCN (L-SCN) divides the network into inter-domain and intra-domain. It allows nodes within a domain to possess more information about available services in that domain and, thus, reduce overhead to share information about these services [81]. Some other naming schemes have emerged, such as: Named-Function Networking (NFN) [243], which describes chaining of named λ-expressions to compose in-network services; and Object-Oriented Networking (OON) [147] that proposes a programmable network using the same abstraction of object-oriented programming languages, where a set of specific functions can be accessed via named operable objects.

2.3 Mobility Management Solutions for Seamless Service Provisioning

Robust mobility management solutions have to be applied to achieve the expected levels of QoS and QoE at the edge. These solutions are needed to prevent communication and service disruptions for some network mobility events occur, such as users changing access points or services being reallocated to different hosts. Poor mobility management may cause service disruption when these mobility-related events occur. Therefore, maintaining service continuity in this environment is a key aspect for achieving the full potential of edge-based service provisioning. This section explores technologies to support service continuity in the presence of user mobility.

When users are on the move they change from one access point to another. Thus, network configurations have to be updated to keep their connectivity. Such process is known as handover. There are different approaches to perform a handover, which will be discussed in Section 2.3.1. Also, the handover may result in other events to maintain the expected levels of QoS and QoE, such as service migration.

Figure 2.3 presents a classification of the main approaches used for the mobility management in EC environments. Network handover, and stateless and stateful service mobility definitions are present in the ETSI specification of end-to-end mobility aspects [65, 66]. The present section divides mobility management into two parts as depicted in Figure 2.3. The first is network handover, which is a network operation to guarantee service and communication continuity when users change access points. Besides user mobility, some events in the network might trigger service mobility, which reallocates services in edge nodes to (i) keep them near to consumers, to reduce latency and enhance bandwidth usage, or (ii) to better use of the resources (e.g., energy saving, load balancing). EC-enabled settings must support migration of two service types according to the presence of user-related state data (i.e., session): stateless and stateful services. Copies of stateless services can be deployed in different hosts, and the user can easily switch access points due to the absence of session data. Conversely, stateful services have session data to be migrated to keep service continuity without disruption, thus the mobility of stateful services is usually referred as service migration or service state transfer. Handover strategies are discussed in Section 2.3.1, while service mobility is solutions are shown in Section 2.3.2.

2.3.1 Network Handover

When users change access points, handover procedures to deal with the network transition process are used. Different approaches can be applied, in terms of using or not an anchor network or triggering or not the handover proactively, as depicted in the left branch of the diagram in Figure 2.3. The idea behind reactive and proactive handover approaches is straight forward. In reactive handover strategies, the process of migrating context occurs after the user connects to the new network. In proactive strategies, the migration process is anticipated by mobility prediction and can start before the user disconnects from the initial network. If the handover is executed without efficient mobility support, users will have to go over a set of repeated processes, such as service discovery and

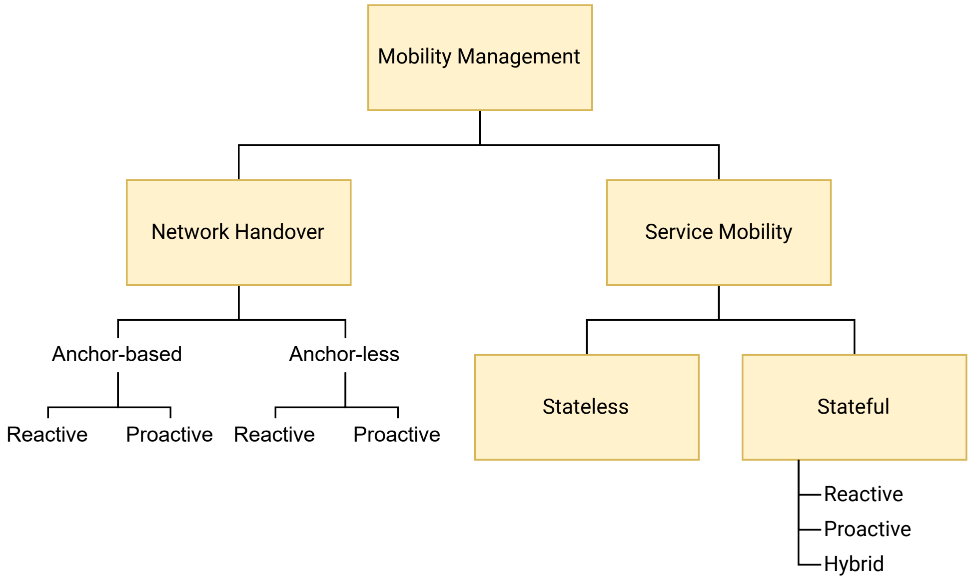

Figure 2.3: Overview of mobility management events and methodologies.

authentication, resulting in disruptions and reducing the QoE [31]. This section describes different approaches in the literature to deal with network handovers.

One approach is to perform both the network handover control logic and data forwarding procedure through mobility anchor networks. Despite the existence of multiple anchor networks, this approach is called Centralized Mobility Management (CMM) because it centralizes logic and forwarding. For instance, in Mobile IP [189], a Internet Engineering Task Force (IETF) standard, creates a transparent interface for the TCP layer in which the IP address of a mobile user is kept constant after mobility events. To provide this feature, Mobile IP approach assigns internally two addresses to the mobile user: the Home Address (HoA), which is seen by the TCP layer and kept constant, and Care of Address (CoA), which is internally handled by the network and updated when the user moves. Each of these addresses have a respective entity associated with it. The first entity is the Home Agent (HA), which tracks the mobile users that belong to its network and forward packages to these users by using their CoA. The second entity is the Foreign Agent (FA), which advertises CoA for mobile users that visit its network so these users can be achieved. All the traffic sent to a mobile user is initially forwarded to its HoA, at its home network, and just after fowarded to the foreing network, using the CoA.

Anchor-based approaches lead to some drawbacks. For instance, since all the traffic is sent to the anchor network and only then to the access network of the user, the delay in communication increases. Anchor-less handover strategies can update communication paths within the network, for instance using SDN [31], to carry packets directly to the current network of the mobile users, thus reducing the numnber of packets traveling in sunboptimal network paths.

Distributed Mobility Management (DMM) solutions have a similar objective as the SDN paradigm of separating the data forwarding from the control logic. According to the definitions of IETF [149, 132], DMM solutions should not allow packets to be forwarded through anchor networks, thus resulting in a sub-optimal route. The IETF combines existing protocols, such as MIP [189], PMIPv6 [117], and HMIPv6 [43]. The goal is to re-use the mobility management functions already deployed in these protocols, such as: (i) Anchoring: control of user original IP address; (ii) Localization: track of current access network where the user is connected; and (iii) Forwarding: receive and forward packets towards the user.

An SDN environment provides a series of advantages for mobility management. Since data forwarding and control logic are separated in the network, fewer configurations must be updated to perform the handover. The required update in the configurations can be achieved by updating SDN flow entries. For instance, user devices can request to the controller an address to use in the next access point [31]. Since the controller has a global view of the network, the switches in the network can be updated to forward data to users' new addresses without sending it through an anchor network. However, this protocol does not fully accomplish IETF requirements, since users have to change their IP addresses according to the networks they are currently connected, thus breaking the IP continuity.

The centralized view of the network of the SDN controller allows a better selection of the route to serve the users when considering their mobility. Furthermore, SDN already has the control logic and data forwarding separate from one another, one of the objectives to achieve optimal DMM according to IETF. Once users move to a new network, the attachment process is executed, and the SDN updates the flow rules in the forwarding switches. This is a reactive handover process; still, a proactive handover is also feasible. The mobile device gathers identifiers to connect to the new access point before leaving the previous one. Thus, all connections will be already set when the device arrives at the new network. Proactive handover increases the possibilities of enhancing QoS and QoE in the handover process since mobility prediction techniques can be used to estimate the users' positions in the future, allowing all the setup to run before the network shift happens. Mobility prediction to aid network management is one of the challenges to be addressed to achieve the full potential of EC. This challenge is discussed in more detail in Section 2.5.

In cellular networks, such as LTE and 5G, different types of handover, in different domains, can be triggered [236]. In the *frequency* domain, when the base stations involved in the handover operate with different frequencies and time multiplexing, user devices have to switch between different frequencies to perform measurements in both frequencies. Differently sized cells are deployed in the network to load balance users connected to a specific base station (e.g., macro, micro, or picocells). When a user device observes that a smaller cell has better QoS measurements, it offloads the bigger cell by migrating its connection to the smaller one. Handovers can also occur in the *radio access technology* (RAT) domain. In this case, a user device changes between different radio access technologies, such as 3G-LTE. In 5G networks, there are more access technology options, such as massive multiple-input multiple-output (MIMO) networks (i.e., networks implemented with arrays of multiple antennas), millimeter wave (mmWave) networks (i.e., network where carrier wavelength is between 1 and 10 millimeters), and also energy harvesting networks (i.e., network where user devices can obtain power, i.e., recharge) [148]. The base station initiates the RAT handover process, which instructs the user device to change access points. Again, not only connectivity variables are considered in the process. For instance, load balancing and other factors may drive the decision for a base station to perform the handover. Finally, handovers can happen in the *operator's* domain. A common example of a handover between different operators is roaming, when users leave the area covered by their original operator and have to switch to another operator that covers that area.

Different metrics are explored in the literature to evaluate the handover in LTE and 5G scenarios. For instance, the number of handover failures, handover success rates, and handover frequency [83]. Another metric is the number of ping-pong events, i.e., when a migration process from access points A to B is followed by another migration from B to A in a short period of time [237]. Some metrics measure directly the impact on the final QoS delivered from services to users [92], such as: (i) handover delay, the time between the user device receives the last packet at the original station and the reception of the first packet on the next base station; and (ii) handover interruption time, when user applications cannot send any packet. Many other metrics exist in other domains related to [148]: (i) spectrum efficiency, (ii) energy efficiency, and (iii) fairness. Spectrum and energy efficiency measure how well the resources in these domains are used by the connections, while fairness concerns the division of the communication resources fairly among users. The current state-of-the-art in handover management for cellular networks focuses on User Association in 5G networks. User Association creates policies to maintain acceptable levels of QoS and QoE in these networks. Different mature algorithms have been proposed in this scope [286, 269], and also ML models started to be considered using mobility-related and other data sources [148].

Different approaches in the literature propose ways to improve the handover procedure and reduce its execution time (HET). The most relevant proposals include (i) schemes where configuration setups required to communicate to a target antenna are made before the disconnection with the currently used antenna, namely Make-before-break (MBB); (ii) schemes that do not use a Random Access to Channel (RACH) to perform timing alignment between the device and the antenna, also known as RACH-less; and (iii) schemes that are coordinated by SDN controllers, SDN-enabled handovers.

The MBB scheme consists of a straightforward idea where execution time is saved by preparing configuration setups before disconnecting from the current base station. This way, they are ready when needed to establish communication with the next base station. This strategy is included in 3GPP standards to be used for the next generations of cellular communication infrastructure [1, 2]. In this case, the X2 interface for wired communication between the base stations is used to exchange information and prepare the configuration setups. RACH-less handovers consist of avoiding executing the RACH procedure during the handover, which is on average 10 12 ms when considering a total handover execution time of 40 50 ms [2]. RACH-less handovers were initially proposed for synchronized networks [27]. However, the exchange of internal clock references of current and target base stations on a handover can provide enough information for the user device to perform the timing alignment in a non-synchronized network without executing the RACH procedure [52]. Still, users need to reach both base stations during the process, which makes it possible to receive the last message from the current base station and send the next message to the target base station. When using SDN controllers, the handover execution aims to take advantage of the global information of the network available for the controller. This information is used, for instance, to trigger handovers proactively and also evaluate the best antenna candidate considering the data plan, not only the signal quality [31].

2.3.2 Service Mobility

Service mobility [65, 66] can happen in the network triggered by different events, such as resource management, energy saving, or accompanying user's mobility. For instance, mobile devices may move away from the infrastructure hosting a service while still consuming it. As depicted in Figure 2.4, there are two ways to keep service continuity in this scenario. First, requests to the service can be forwarded to the original server. Still, problems may arise for maintaining low levels of latency to services that require high reliability, low jitter or also have high data transfer volumes. In this situation, one option is reallocating the service instance, thus requiring a migration of the service so it can run on an infrastructure closer to the device. This migration may add overhead in terms of service downtime, network traffic, and computing. Nevertheless, overall QoS should be increased to the final user allowing to meet application requirements. Studies on service migration focus on virtualization technologies, such as hypervisor-based [278, 24] and container-based [234, 157], and how to perform the migration procedure.

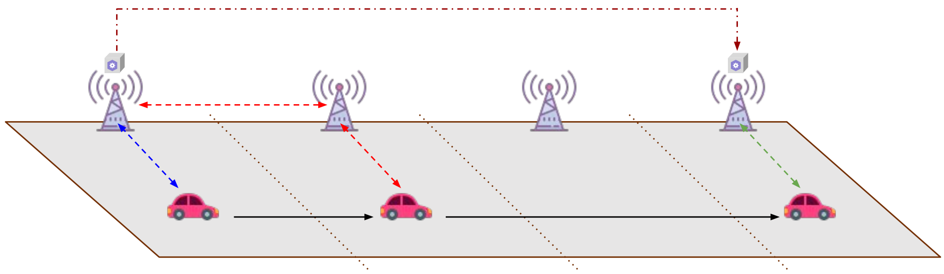

Figure 2.4: Service horizontal migration due to user mobility.

Service migration is divided into two broad categories according to the existence or absence of user session data. Services without sessions are called stateless services. In this case, the main data migrated is related to its running code. This code can be downloaded from the original host node, from other neighbor nodes, or replicated beforehand to enhance migration performance. On the other hand, stateful services hold sessions of users consuming them, thus all session data has to be migrated. This session data is stored in two forms: (i) main memory – i.e., stores data for immediate usage – and (ii) storage – also secondary memory, which stores persistent data. Main memory migration is the critical operation for migrating stateful services since this data is usually at constant usage. Storage migration is a bottleneck since it represents a large amount of traffic to traverse the network. In terms of storage migration, some distributed file systems have been proposed in the literature [171, 87, 187], which handle the responsibility of moving large chunks of data (this approach is the last topic discussed in Section 2.3). The migration process can be proactive or reactive for either main memory or storage. Thus, there is a great variety of combinations of proactive/reactive main memory migration with proactive/reactive storage migration and some hybrid approaches [278].

As mentioned before, some studies focus on migrating VMs. For instance, to enhance mobile user experience Follow-Me Cloud (FMC) [231] explores the migration of services among different datacenters. A Markov Chain decision algorithm is used to decide whether or not to migrate VMs. Besides user experience, other factors can influence the decision to migrate VMs, such as, the trade-off between energy consumption in the migration process and delay. These variables can be measured using models [24] to allow the decision to migrate a machine and also select the most energy-saving migration strategy. To reduce the overhead of VM migration, identification of segments of in-memory data with imminent access can be used [144]. Thus, only these data segments can be migrated, reducing the traversed load among the hosts. This identification is made in a pre-deployment step where the application code is parsed, and metadata about context information used more often in the main memory stack is extracted and applied to support the decision of what to migrate.

Container-based migration is a recent research topic in the literature [255], but has gained attention in the industry (with Docker Swarm and Kubernetes) and academy [235, 115, 157]. Similar to FMC, a Markov Decision algorithm can be applied to container-based migration. For instance, this approach can be used to evaluate a trade-off between delay constraints and power consumption in the migration process to decide whether to migrate a container or not [235]. Even mobility parameters are considered in this approach. CoESMS [115] is an EC migration framework that models power consumption and user QoE accessing a service in terms of utility functions in a cooperative game (i.e., game theory).

Docker containers are composed of a series of layers, each one of them added to the container as a result of one operation (e.g., move, copy or download files). This layered composition can be used to improve the migration process. For instance, multiple Docker containers share layers with the same content. These layers are mapped to unique identifiers. However, even containing the same content, they may be mapped differently. An algorithm that applies the same identifier to label the layers that have the same content were used to allow layer reuse. The goal is to reduce the amount of data downloaded from the cloud to deploy a given service [157].

To support services running and moving at the edge, most of the data persistence for EC-enabled applications is delegated to the cloud. Yet, reliable storage and data management at the edge are necessary to support some classes of applications that perform frequent update to this data. Some initial studies proposed to deploy EC-hosted file systems. ElfStore [171] is a methodology to store data blocks at selected locations to achieve data reliability. Stored data uses a block-level differential replication scheme to achieve a minimum reliability level. This replication scheme splits large chunks of data into blocks. These blocks can then be stored and replicated. Common segments between multiple blocks are then identified and some of the copies removed to reduce storage resource usage. The desired level of reliability can be maintained by replicating the different segments of the blocks and combining them with the common segments to obtain the complete block. Bloom filters are used to explore the hierarchical structure of

EC and enhance data block retrieval. This filter is a data structure used to determine whether an element belongs to a set or not. Another data storage service that supports reliability is Fog Store [87], which proposes a solution for the placement of replicas of data blocks for Fog Computing. The proposed mechanism takes into account network topology and device heterogeneity to decide about data storage placement. Fog FileSystem [187] is a solution to aid the process of migrating services in EC. Snapshotting and synchronization are applied to reduce the migration time of disk states between different nodes.

Besides data storage as a service, application state management services are important to support latency-critical applications. For instance, Do et.al. [59] proposed a latency-aware placement of state management functions for 5G scenarios. This placement solution stores the state data at the cloud, which might create barriers to access it under certain latency thresholds, even more if considering the high update rates of such data. A more general mobility-aware state support was proposed for SDN by [191]. The authors propose to rewrite communication flows to enable the user to consume the state from a static host. Yet, consuming from a static node may create bottlenecks when provisioning services. A similar solution for consuming state data from a constant node is proposed for SCN [82, 80]. In their study, the authors also fix the path to consume the data. This strategy limits the application of standard ICN multi-path capabilities, which negatively impacts the scalability of the solution. Filho et.al. [73] propose a transparent system to replicate state data in multiple hosts. The authors cover two main scenarios: (i) centralized state, in which the service should be stopped while the state is migrated; and (ii) distributed state, where a coherence mechanism is proposed that replicates all data update operations in all hosts.

2.4 EC-Enhanced Mobile Applications and Services

The discussed set of EC architectures would support a plethora of modern applications in different domains discussed in the preset section, such as the Internet of Things (IoT) (Section 2.4.1), Immersive Media (AR/VR) (Section 2.4.4.1), Intelligent Transportation Systems (ITS) (Section 2.4.2), Unmanned Aerial Vehicles (UAVs) Services (Section 2.4.3), Smart Cities (Section 2.4.5), and Edge AI (Section 2.4.6). Some of these applications already exist today, while others are still being studied. Broad deployment of EC technologies is required to handle the massive adoption of such applications. This section highlights some instances of applications and the strategies they use. We present different classes of applications for each of the domains mentioned above. Table 2.3 shows an overview of all applications explored in this section.

For each domain/class in Table 2.3, some studies were explored in order to provide a view of the requirements in terms of delay and data rate, shown in Table 2.4. In each domain, applications may have different requirements, such as in immersive media, where requirements for data rate in AR/VR and Gaming are more strict than in teleoperation applications. In contrast, in other scenarios, requirements are constant for most of the applications, like UAVs. Smart Cities is a peculiar case where different applications, even in the same classes, have varied delay and data rate requirements. This happens

Table 2.3: Taxonomy of main upcoming mobile applications and EC-enabling technologies used to deploy them.

			Computation				Communication		Service Virtualization	
Domain	Class	Instance	Fog Computing	MEC	VEC/VFC	SDN/NFV	ICN	Geo-Centric Networking	Container-based	VM-based
IoT	Industry 4.0	[98]	✓	✓					✓	
		[116]	✓	✓						
		[143]	✓	✓						
	Cognitive IoT	[17]					✓			
		[287]		✓	✓		✓			
		[48]	✓	✓		✓				
	Body Area Sensing	[142]				✓				
		[127]	✓	✓		✓	✓			
		[196]					✓			
	Healthcare	[145]	✓	✓						
		[4]	✓							
		[142]				✓				
ITS	Traffic Management Systems	[256]	✓	✓						
		[8]	✓			✓				
		[13]			✓		✓			
		[118]	✓	✓	✓					
		[29]		✓	✓	✓				
	CAVs	[228]			✓	✓				
		[47]	✓	✓		✓				
		[188]		✓		✓			✓	
		[287]		✓	✓		✓			
	Internet of Vehicles	[265]	✓	✓	✓	✓				
		[97]					✓			
		[153]					✓	✓		
		[249]				✓		✓		
UAV Services	Augmented Environment Information	[134]					✓			
		[284]		✓		✓	✓			
		[111]	✓	✓					✓	
	Navigation and Swarming	[197]				✓				
		[264]				✓				
		[288]				✓				
Immersive Media	Augmented and Virtual Reality	[121]		✓		✓				
		[76]	✓	✓						
	Teleoperation and Telepresence	[170]	✓	✓		✓				
		[185]	✓	✓		✓	✓			
		[217]		✓						
	Gaming	[261]	✓							
		[101]	✓							
Smart Cities	Public Services	[114]					✓			
		[250]	✓	✓		✓			✓	✓
		[137]			✓	✓			✓	
		[93]	✓			✓	✓			
	Location-Based Services	[232]		✓			✓			
		[168]	✓							✓
	Mobile Crowdsensing	[138]		✓		✓				
		[272]		✓			✓			
		[155]	✓	✓					✓	
Edge AI	Infrastructure Management	[211]	✓			✓			✓	
		[128]	✓			✓			✓	
		[154]	✓	✓		✓				
		[223]			✓					
		[270]			✓					
	Support for Smart Services	[253]	✓			✓			✓	
		[54]		✓	✓	✓			✓	✓
		[100]		✓						

because any application can use different types of data from nearby sensors and the cloud together. The values present in Table 2.4 are the average values for those applications with references where more information about these requirements can be found. Still, there are exceptional cases where the requirements can be more strict. For instance, some control applications in smart factories have 10 μs of delay tolerance [158]. In contrast, data collection for psychological applications in Body Area Networks (BANs) and healthcare require up to 10 Mb/s of data rate [238].

2.4.1 Internet of Things

IoT is an infrastructure of physical and virtual connected devices with sensing and actuating capabilities. This infrastructure aims to create a collaborative environment between for many devices, augmenting the possibilities of monitoring and acting over a cyber-physical domain [210]. Furthermore, these collaborative environments must simultane-

Table 2.4: Requirements for applications in different domain and classes.

Section	Domain/Class	Delay	Data rate
2.4.1	**IoT**		
2.4.1.1	Industry 4.0	1-10 ms [218]	<1 kb/s [218]
2.4.1.3, 2.4.1.4	Body Area Networks and e-Health	<250 ms [15]	0,1-50 kb/s [15]
2.4.2	**ITS**		
2.4.2.1	TMS	>1s [33]	<2 Mb/s [33]
2.4.2.2	CAVs	1-10 ms [136]	>1 Gb/s [136]
2.4.2.3	IoV	1-100 ms [33]	>25 Mb/s [33]
2.4.3	**UAV**	**5-50 ms** [274]	**<1 Mb/s** [276]
2.4.4	**Immersive Media**		
2.4.4.1,2.4.4.3	AR/VR and Gaming	5-30 ms [91]	<10 Gb/s [91]
2.4.4.2	Teleoperation	5-20 ms [33]	>25 Mb/s [33]
2.4.5	**Smart Cities**	**1 ms-1 s** [158]	**1 kb/s-1 Gb/s** [125]

ously support millions of mobile users. To handle these users, issues related to management and scalability arise. There is a need to decentralize information and communication technologies and bring them closer to users through the enabling EC architectures discussed earlier to support a massive adoption of IoT [210, 260, 62, 22]. Once these enabling architectures become well studied and deployed, a variety of applications may emerge. This section discusses the main classes of IoT applications: Section 2.4.1.1 shows how the industry applies IoT applications in a manufacturing process; Section 2.4.1.2 describes how ML can be applied to the IoT domain creating new possibilities of applications; Section 2.4.1.3 describes applications that build networks around a human body, with devices such as wearables; finally, Section 2.4.1.4 discusses applications to facilitate health care of users. We also discuss how applications use EC technologies to enhance aspects of QoS, QoE, and business models in each section.

2.4.1.1 Industry 4.0

Cyber-Physical Systems (CPS) supported by IoT is also an interesting technology applied to control Smart Factories in Industry 4.0 scenarios. CPSs are systems that connect virtual and real environments and allow the interaction between them while being controlled – or monitored – by humans. There is still resistance to adopt virtualized solutions to keep manufacturing infrastructure. This resistance is mainly due to the short deadlines that machines must respond to real-world observed events, usually real or near-real time. Yet, requirements such as distributed sensing, data analytics, and enhanced network bandwidth usage are pushing forward this evolution. In these factories, services run on a great variety of devices and they usually migrate vertically (i.e., from the edge towards the cloud), seeking for more resources. To make these devices portable for multiple platforms and also to enable these services to run with a varied amount of resources, a lightweight virtualization technique is preferred, such as container-based service virtualization [98]. These vertical migrations increase network and service management intricacy to keep the levels of QoS. Therefore, SDN solutions to handle cloud-edge interplay have been proposed [116, 143].

2.4.1.2 Cognitive IoT

Recent developments of IoT allied to advances in ML brought up the possibility of providing smart services. These services can collect and process information about the environment around them and make decisions to perform their tasks independently. This class of services is referred to as Cognitive IoT in the literature and can take advantage of EC technologies. Various network architectures exist to support these applications. Information-Centric Sensor Networks (ICSNs) are used to serve information based on user requirements, rather than providing an endpoint to read raw data [17]. The authors evaluate the usage of different machine learning models to identify the best communication paths in the network to deliver the data to the consumer. A distributed map-reduce framework [287] was proposed to gather enormous amounts of vehicular and infrastructure sensor data and feed it to an architecture to apply ML and other analytical models and provide an intelligent route planning service. This framework uses ICN to allow vehicles to consume sensor data to evaluate traffic conditions and based on a . Data analysis tasks execute on MEC and VEC infrastructures to produce this information. Cognitive-LPWAN [48] is a framework that uses SDN management of network traffic in Low-Power Wide Area Networks and some unlicensed spectrum technologies. This framework proposes the usage of a cognitive engine to create a smart orchestration of wireless communication technologies including 4G, 5G, LoRa, and SigFox. Using the cognitive engine and the combination of these wireless technologies the authors could achieve a sustainable trade-off between transmission delay and energy consumption compared to the technologies individually.

2.4.1.3 Body Area Sensing

Sensing devices have been spread in urban environments to facilitate the task of monitoring fast-changing city dynamics. Most commonly, these devices use Wireless Sensors Networks (WSNs) to connect and cover wide areas for different applications such as fire detection and building monitoring [120]. Studies point out that these networks have been brought closer to users with wearable (and implantable [213, 108]) devices, forming Wireless Body Area Networks (WBANs) [140]. Wearables in WBANs are constrained devices that still have to run multiple tasks and report data to other wider-area networks. Since these devices are attached to users, they are subject to the same mobility patterns as them. These characteristics create the need for solutions to handle communications within these networks and bridge their interaction with other networks. Some of the enabling EC technologies are expected to make this level of interaction of WBANs and other networks achievable. For instance, some applications for reporting users' vital signs use SDN-based solutions to handle network issues [142]. Some other studies have applied ICN-based solutions to improve efficiency in WBANs. By using ICN and exploring in-network caching thus reducing the amount of redundant sensors [196], or minimize traffic load when connecting to external networks [127].

2.4.1.4 Healthcare

Wearable and implantable devices gained wide attention due to their use in healthcare systems. In this scenario, mission-critical applications and mobility increase even more the complexity involved in deploying systems. EC-enabling technologies have a fundamental contribution in implementing such systems. Multiple studies discuss the application of these technologies individually. For instance, Fog Computing was used to deploy a task scheduling and offloading platform for healthcare with native support for patient mobility [5, 145]. SDN was applied to reduce in-network traffic load due to the vast amount of monitoring devices that need to access real-time information [142]. The combination of Fog Computing and ICN was studied together to reduce latency and allocate safer storage for privacy matters [86].

2.4.2 Intelligent Transportation Systems

The increasing number of vehicles has forced the deployment of complex transportation infrastructure in large urban centers, yet in many cases, this infrastructure is inefficient, which results in a waste of valuable time for the citizens. Due to this inefficiency, multiple studies have explored strategies to create a more intelligent transportation infrastructure [228, 12, 265, 203]. These efforts explore classes of applications such as Traffic Management Systems, Connected and Autonomous Vehicles (CAVs), and Internet of Vehicles (IoV). These specific classes of applications have high mobility requirements. This section focus on these classes and how they use EC technologies to run their services.

2.4.2.1 Traffic Management Systems

To enhance road network usage by vehicles, Traffic Management Systems (TMS) emerged as part of ITS. Studies in this class of applications vary from road accident detection based on vehicle to vehicle and vehicle to infrastructure communication [13], to issuing violation tickets in vehicular named data networks [118]. The distributed nature of EC brings advantages to collect and process localized data to produce real-time traffic information and reduce unnecessary movement of data, alleviating bandwidth of the core network and mitigating privacy issues. For instance, to obtain an overview of a road state, SDN-based crowdsensing [256] can be applied to collect and provide data to support context awareness. Also, SDN and VANETs are used to identify congestion-sensitive spots using GPS data collected from vehicles [29]. The SDN controller global view of the VANET is explored, centralizing the data and applying recurrent neural networks to forecast traffic behavior.

2.4.2.2 Connected and Autonomous Vehicles (CAVs)

The need to deploy fast-moving vehicles that could operate without human intervention on urban roads and solve traffic congestion issues raised multiple research projects. One challenge to achieve the full potential of CAVs is related to the computing-intensive services they have to run, such as trajectory and route planning, object detection and tracking, and even behavioral reasoning on proceeding in an intersection or overtaking.

Besides that, vehicle-to-vehicle communication can enable a better performance of the entire ITS by allowing more cooperative decisions to be taken. The amount of data exchange to support CAVs is expected to surpass 1 Gb/s [136] for every vehicle with use cases in which the maximum tolerable end-to-end latency is in the range from 1 to 10 ms [33], which challenges even emerging 5G networks. EC will be present inside the vehicles, in the form of On-board Units (OBUs), or attached to nearby infrastructure, in the form of Road-side Units (RSUs) that can be used to meet these requirements. SDN-based VANETs architectures have been studied to allow offloading of tasks to nearby vehicles [228] or infrastructure [47, 188] to address the problem of limited resources to run computing-intensive tasks for autonomous driving.

A market mechanism is used to motivate users to share the idle resources of their vehicles [228]. This mechanism creates an ad-hoc marketplace where the prices of the resources are settled according to the number of idle resources available in the seller vehicle. SDN/NFV network slicing is used to isolate certain driving functionalities in service slices [47] to attend to ultra-low latency requirements of autonomous driving services. AVNET [188] is an architecture to address issues related to the amount of data transmitted and the number of processing tasks in CAVs scenarios. This architecture proposes the usage of: (i) NFV to implement multiple network functions and allow more diverse CAVs services to be deployed; (ii) MEC to offload tasks of these services to infrastructure and also nearby vehicles; and (iii) SDN to maintain a global view of the network to achieve efficient resource management. Routing protocols to better manage task offloading are also studied to operate using the ICN principle [287].

2.4.2.3 Internet of Vehicles

Constituted by distributed transport communication networks, IoV [131] allows ITS applications to make decisions based on data collected from other vehicles and sensors, which aid the process of driving people and goods towards their destinations. The communication features provided by IoV are important for applications, such as TMS and CAVs, and also applications related to smart-parking and virtual traffic lights. EC, in the form of VEC/VFC, is essential to this class of applications since communication and processing facilities are deployed in the vehicles. These vehicles use OBUs to perform communication and run tasks to support services. These OBUs use dedicated short-range devices and enable the formation of VANETs. VANETs do not require any infrastructure to be formed, yet RSUs can be used to improve the network QoS and overall capacity. One of the duties of vehicular communication is to handle emergency communication, such as car accident notifications or traffic flow reports. In this context, eVNDN [97] applies ICN to broadcast emergency-related messages in vehicular networks, exploring the facility of communicating to fast-moving nearby nodes and also determining their interest in a given message. An emerging class of applications for vehicular networks is Vehicular Social Networking (VSN), in which vehicle riders share spatio-temporal data with other vehicles in similar conditions. SDN and Blockchain technologies can help to certify data exchange transactions in a distributed fashion while ensuring data source anonymity [265]. Different networking technologies have been combined to enhance VANETs and cope with applications that rely on vehicular communication, such as GOFP [153], which supports geographically tagged information retrieval in VNDN, or SCGRP [249], an SDN-enabled geographic routing protocol.

One of the main standards to realise IoV is the Cellular Vehicle-to-Everything (C-V2X) developed by 3GPP [50]. It envisions the evolution of LTE-V2X (4G) to NR-V2X (5G) to enable highly-reliable low-latency service provisioning for vehicular applications. This evolution will allow the support of advanced vehicular applications with stringent requirements. For instance, the most stringent application for LTE-V2X (i.e., pre-crashing sensing and warning) requires 20 ms latency and 95% reliability [64], while use cases for NR-V2X (e.g., emergency trajectory alignment) require latency levels to reduce to 3 ms with 99.999% reliability [68]. Finally, besides LTE and NR, more technologies can be combined and complement each other to meet application requirements in Heterogeneous Vehicular NETworks (HetVNETs) [289] framework (e.g., IEEE 802.11p).

2.4.3 UAV-Enabled Services

UAV-based platforms become an infrastructure alternative to network management and sensing for multiple applications. The advantage of such platforms is related to their aerial characteristics, which facilitate deployment almost anywhere. This possibility of easy deployment brought attention from the government and industry to adopt UAVs [198]. Communication among UAVs usually is supported by satellites, cellular networks, or Unmanned Aerial Vehicles Ad-hoc Networks (UAVANETs). This section highlights some UAV classes of applications and discusses how networking technologies are applied to support them.

2.4.3.1 Augmented Environment Information

Diverse applications for UAVs obtain information about a given variable of the environment to feed this information to other applications or systems. In these applications, UAVANETs have to handle a significant amount of data collected by themselves or terrestrial nodes that report to them. EC infrastructure can be used to aid the data collection by running tasks related to data aggregation and fusion and also by coordinating how data should be reported to its consumers. For instance, an early fire detection system that uses the sensing capabilities of UAVs to produce short videos that are sent for analysis at EC infrastructure is proposed [111]. A container orchestrator handles the processing tasks in EC infrastructure. This orchestrator can create, run, scale, and stop services. A different use case of UAVs to aid monitoring is to replace network infrastructure. Data collection can be executed in areas with no wired infrastructure by deploying network backbones with UAVs [284]. A load-balancing algorithm operated by an SDN controller manages the data traffic in this backbone. ICN in-network caching is used to mitigate the issue of content dissemination in UAVANETs [134]. A blockchain-based strategy to handle content poisoning that may contaminate cache and prevent the fetching of valid content is used to enhance the security of the UAVANET.

2.4.3.2 Navigation and Swarming

Due to the flying capabilities of UAVs, UAVANETs can quickly adapt their topology to respond to network events. EC infrastructure can aid the process of coordination of the drones by offloading tasks or providing a wider view of the system. One approach that can be used to handle dynamic changes to the network topology is by using SDN in UAVANETs. SDN controllers can be used to send control packets to UAVs, demanding that they move to different positions [197]. To allow this control, the controllers use a search procedure that looks at the rate demands and paths of each communication flow and changes the topology to maximize throughput. A more complex collaboration scenario for UAVs is the formation of drone swarms, which are open networks that can organize themselves. SD-UAVNet [288] is another architecture for UAV placement to optimize UAVANETs. SDN can control operational parameters of UAVs and mitigate the impact of the mobility of UAVANETs for streamed video transmissions by positioning relay UAV nodes. SDN is used with the MQTT[2] protocol to enable flexible swarms formation while allowing the control of topology and bandwidth [264]. A multi-path routing scheme is proposed, in which drones move to produce multiple communication paths. These multiple paths are used to increase the bandwidth to meet the desired QoS.

2.4.4 Immersive Interactive Media

AR, VR, and other mixed reality experiences have recently been applied to produce immersive media applications. Such applications extend reality by emulating it on a device and adding new layers of information. The increase in such applications is due mainly to the recent popularization of head-mounted devices and also smartphone capabilities to support immersive media (e.g., smartphone-enabled cardboard headsets). Applications for immersive media are resource-consuming, which reduces the user experience of such applications because of the necessity of the headset being wired to a powerful computer – or at least usage with limited mobility when in wireless scenarios. However, EC technologies can support applications consumed in (mobility-free) wireless headsets/devices in the near future by offloading resource-consuming tasks. In contrast, there are immersive-only classes of applications, such as 360º Videos [291]. This section focus on Augmented and VR and Teleoperation and Telepresence.

2.4.4.1 Augmented and Virtual Reality

A large amount of data is being gathered from IoT devices and other remote sources that can be accessed through the Internet. AR is an interactive experience with real-world mediated by human-machine interfaces. These interfaces allow a better visualization of this data collected from different sources. Such data needs to be organized and aligned to coordinate systems on top of the real-world coordinates to allow this visualization. Since AR/VR terminals have limited resources, these tasks can be offloaded to EC infrastructure. Such a setup has been applied, for instance, to build Industry 4.0. Navantia's Industrial AR [72, 76] is commercially used to facilitate the execution of certain tasks in

[2]http://mqtt.org/

shipyards. This system uses EC infrastructure to reduce the response delay to handle real or near-real-time communication wirelessly – wireless technology is required to allow the desired level of mobility inside the shipyard industry. VR-CPES [121] is an education system that also offloads tasks to EC. This system uses an SDN-enabled Time-Sensitive Networking (TSN) framework to address issues of QoE of users due to delay and packet loss in real-time network communication.

2.4.4.2 Teleoperation and Telepresence

Immersive media are also expanding the horizons of people with multiple interactive applications. For instance, different applications involve the remote operation of devices or even the telepresence of people; such applications allow to save time and reduce costs. Such applications require a reasonable amount of data to be sent and processed (e.g., videos and metadata about the environment around both ends of the communication), which can be handled by EC infrastructure. A use case of immersive media in the industry, for example, is remote live support [217]. In this application, a machine operator can receive help to fix an issue from an expert. The system uses an approach to offload AR tasks to EC infrastructure to deploy a real-time live support system. Thus, a remote expert can make annotations to a video stream recorded by the machine operator. The operator also visualizes these annotations to facilitate the process of fixing the issue. In this use case, an important concept is applied, transferable skills. Immersive media is one of the key enablers for the anticipated Internet of Skills and Tactile Internet [21]. The concept of Tactile Internet is to reproduce touch-based human communication to the network – the subject will be discussed later in Section 2.5.9. In the field of Tactile Internet, some applications are already being proposed, such as telesurgery. SDN-enabled EC infrastructure is used to reduce latency dramatically and allow surgeons to remote control a surgery robot [170]. To realize Tactile Internet, EC architectures (e.g., Fog Computing, MEC, SDN, and ICN) are being studied to work together while also using robust ML models to predict movement and actions of users, thus reducing latency even further [185].

2.4.4.3 Gaming

Among the top 10 highest downloaded games for mobile devices nowadays, Niantic's Pokemon GO[3], an AR game, can enhance the gaming experience provided to users using EC infrastructure. Games that immerse players in the real world have high QoE requirements and challenging mobility characteristics to be handled by communication facilities, which may require EC-enabling technologies. Indeed, in 2018 Deutsche Telekom[4] started placing decentralized micro-servers to leverage EC infrastructure deployment, and Pokemon GO was one of the first AR applications to use this platform [261]. Such infrastructure, and also 5G networks, will augment the possibilities for game development for this and other AR and VR games where mobility is a critical factor. Another type of EC approach to enhance mobile gaming experience is UAV-assisted EC [122, 101]. In such a scenario,

[3]https://www.pokemongo.com/
[4]https://www.telekom.com/

mobility awareness is an even more critical factor, since both consumers and producers will be mobile entities. This setup supports AR and VR games to be played inside vehicles [101]. Multiple UAVs are clustered to offload tasks related to computing, caching, communication, and AI-based decision-making.

2.4.5 Smart Cities

The application of information and communication technology to enhance the performance of services in large urban centers added a lot of attention to smart cities. The idea behind smart cities is to build infrastructure for monitoring of several city dynamics and act according to insights obtained with this data to serve citizens better. We discuss Public Services (Section 2.4.5.1) and Location-based Services (Section 2.4.5.2) that can be enhanced by EC infrastructure.

2.4.5.1 Public Services

Various services in urban centers can take advantage of information and communication technology, such as power, water, environment monitoring and waste management. Often, sensor networks are deployed to collect data over large regions. A4-Mesh [106] is a wireless mesh sensor network deployed to collect weather data in near-real-time. This network produces a large amount of data that needs to be sent to a central remote processing station. Wireless sensor networks are convenient since they can be easily deployed without a big effort on underlying infrastructure. In order to make better usage of the communication channels, multiple sensors in an urban center can use Narrowband communication technology for IoT applications (NB-IoT). Although this type of communication technology has a reduced bandwidth, it uses fewer frequencies of the wireless spectrum and has low power consumption. Such technology may be adequate in scenarios where a large number of sensors can be spread in the urban perimeter to increase data collection coverage. In environmental monitoring applications, for instance, each sensor does not transmit a large volume of data and usually has a limited battery, making it an interesting use case for exploring NB-IoT [222, 51].

Many applications for Smart Grid aimed to allow power generation and electricity transmission are delay-critical. ICN is used to enhance the communication needed to manage such an infrastructure [114]. ICN allows the reduction of delay to obtain data about the current state of the grid. Battery Status Sensing Software-Defined Multicast (BSS-SDM) [137] is a battery status sensing scheme based on SDN to reduce the latency of the communication between electric vehicles and the power grid. An SDN controller keeps the status of the batteries in vehicles. This information is used to schedule vehicle recharges. The vehicles are notified via messages transmitted by multicast. Another important issue pursued in smart cities is security. For instance, different technologies have been applied to analyze surveillance videos and identify events. One way to manage all the surveillance application and video analysis services – while also reducing the traffic load sent to the core network – is by orchestrating containerized services over the EC infrastructure [250]. SDN-enabled containers allow the SDN controller to orchestrate better and save EC resources. AODV-SPEED [11] is a communication protocol to enable smart street highlights. This protocol combines SDN and ICN to enhance network QoS for service provisioning.

2.4.5.2 Location-based Services

Location-based services consume strategic spatio-temporal information to deliver value to their users. In general, many services running on EC infrastructure can take advantage of location awareness. For instance, the position information of mobile users can be used to improve service resource scheduling and deployment in virtualized platforms [168]. Tracking moving objects is an important source of information when studying more reliable and predictive services. ICN and in-network service-provisioning functions were used to develop a moving object tracker application [232]. This application coordinates a distributed video service that produces a video stream of a single moving entity (e.g., vehicle) using a system of multiple cameras. The video consumer sends an interest containing the vehicle ID to the network, and, later on, each camera receives this vehicle ID and transmits the video only when the vehicle is in its capture area. The vehicle sends its position to the network to verify in which camera capture area it is at a given moment.

2.4.5.3 Mobile Crowdsensing

Due to the large urban perimeter in some cities, deploying infrastructure to sense entire urban areas may become challenging. One solution to tackle this issue is by applying mobile crowdsensing [272, 138, 155]. Mobile crowdsensing advocates for the sharing of spatiotemporal annotated data collected from mobile devices (e.g., smartphones and tablets) about different urban phenomena. To manage the formation of opportunistic networks in mobile crowdsensing, Software Defined Opportunistic Networks (SDON) [138] uses the centralized control of SDN. Statistical data stored at the SDN controller allow the creation of an incentive mechanism for users' participation in the sensing process. A different approach to manage opportunistic networks for mobile crowdsensing is by using ICN [272]. An urban pollution monitoring system orchestrates container-based microservices to integrate data from multiple heterogeneous data sources [155]. These services compose a layer where data streams from different sources (e.g., mobile phones, IoT sensors) are integrated.

2.4.6 Edge AI

Edge Computing enables a series of interesting use cases for applications to be explored, among them Edge AI aims at allowing the usage of Machine Learning models to enhance the applications already running or envisioned for the edge. Thus, in the present section, we discuss two main aspects of intelligent service provisioning at the edge: (i) how to manage the cloud-edge infrastructure in Section 2.4.6.1, and (ii) technologies to support smart services at the edge in Section 2.4.6.2. The usage of ML models raises challenges on data privacy that can be overcome by using Edge Computing architectures. Data privacy issue and related challenges are discussed in Section 2.5.7.

2.4.6.1 Infrastructure Management

One application of intelligence at the edge is the orchestration of the envisioned multiple services provisioned in Fog Computing and MEC infrastructure. Fog Scaler [211] is a service orchestrator that targets horizontally scaling services running at the edge. The authors use a Reinforcement Learning algorithm and model different cost functions to allow their solution to take decisions on the placement of containerized service instances. OctoFog [128] is another service orchestrator solution focusing on optimizing the migration of services at the edge. The authors minimize two cost functions that model latency and energy consumption of the migration procedure. This minimization is achieved by applying a Deep Reinforcement Learning algorithm that is divided into two layers, one hosted in the cloud and the other in the Fog. The cloud layer hosts the main control of the resources, while local decisions are taken at the Fog layer with reduced latency. Liu et.al. [154] divide IoT services into a collection of chained service functions. They proposed a VNF placement and service path routing framework that minimizes the end-to-end delay observed when consuming the services. Their solution uses a Deep Reinforcement Learning approach to achieve this minimization in real-time. The solution observes the IoT network state and the number of requests to the IoT services and outputs orchestration strategies composed by VNF placement and network routing paths.

Besides orchestrating services, allocating resources is an important aspect of provisioning services at the edge that can take advantage of ML models to be enhanced. Shi et.al. [223] propose a mechanism to match idle vehicular computing resources to tasks that have to be processed for delay-sensitive applications. The authors propose a reinforcement learning algorithm that evaluates wireless channel state and idle resource pool in vehicles and outputs efficient task allocation strategies. Since the resource of vehicles may not be voluntarily shared, the authors propose a pricing scheme to stimulate this sharing. In a similar setting, Ye et.al. [270] propose an alternative reinforcement learning solution for communication resource allocation in vehicular computing resources. One advantage of this method is the possibility of independent vehicles taking decentralized decisions to satisfy desired latency constraints.

2.4.6.2 Support for Smart Services

Kubernetes-Based Fog Computing IoT Platform for Online Machine Learning (KFIML) [253] is a platform for service orchestration at the edge developed on top of Kubernetes. This platform has the potential to facilitate the deployment and management of different service stacks at the edge, including mainstream data processing frameworks. The authors use their proposal to manage a LSTM-based real-time data stream processing applied in an IoT scenario. Dalgkitsis et.al. [54] propose a service orchestration platform for Vehicular-to-Everything scenarios that aims at predicting the next access point of vehicles in the cellular network and migrating services consumed by these vehicles proactively. The authors use a Convolutional Neural Network to predict the next access point of the vehicles, then a Genetic Algorithm is used to search for a service allocation strategy to place services closer to their users while considering user priorities and resource utilization.

When dealing with IoT applications, one common limitation is the reduced processing capacity and energy available in the end devices. Sometimes, although a ML model that performs well is trained, it cannot run on the device and therefore cannot be used. One solution in the literature for these scenarios is the partitioning of the model for inference acceleration. Partitioned models are composed of many layers that can run at different distances from the source node consuming the model predictions. Dynamic Adaptive DNN Surgery (DADS) [100] is a framework that allows the partitioning of DNN models to adapt its usage according to the status of the network. This model can dynamically adapt the model partitioning to maximize the throughput or minimize the delay of predictions. A similar strategy of partitioning models can be applied during the training phase in order to save the resources of devices. Adaptive REsource-aware Split-learning (ARES) [212] is a solution that accelerates the training phase of a global model by selecting split points for every device involved in the training considering network and computing resource variation over time. The approach also reduces the impact of slower devices in the time taken for training the model, which is important in highly heterogeneous scenarios.

2.5 Challenges and Opportunities for Mobility-Aware Service Provisioning

EC architectures for service provisioning in urban environments have been extensively studied [176, 236, 201, 160, 126, 278, 208]. Academia has put much work into proposing EC technologies, as shown throughout this chapter, and also some successful industry use cases can be observed[5], mass adoption of EC has not happened yet. However, some issues related to its practical deployment still need to be addressed to fully achieve the stage 3 depicted in Figure 2.2. Besides the costs involved in deploying such an infrastructure, it is unclear how to overcome many obstacles.

While running services at the edge facilitates the handling of some of these obstacles, it also brings new challenges to networking. In the present section, we discuss the new challenges that emerge together when computation is performed at the edge. Section 2.5.1 discusses how mobility predictors may be used to avoid communication disruption caused by the mobility of the users. Section 2.5.2 discusses issues related to the migration of service instances running at the edge. Section 2.5.3 outlines how caching strategies can be used to support service provisioning. Section 2.5.4 discusses the usage of distributed authorization to secure data access hosted in multiple nodes at the edge. Section 2.5.5 shows challenges when allocating tasks to run at Edge Computing infrastructure. Section 2.5.6 discusses the implementation of distributed file systems over edge networks. Section 2.5.7 highlights the importance of data privacy solutions when ML models are trained at the edge of the network. Section 2.5.8 describes scalability-related challenges of services. Finally, Section 2.5.9 discusses the challenges to be overcome when moving towards the next stage of service provisioning over the Internet.

[5] AWS Lambda@Edge (https://aws.amazon.com/en/lambda/Edge/) and Green Grass (https://aws.amazon.com/en/greengrass/), and Google Serverless with KNative (https://cloud.google.com/serverless/).

2.5.1 Mobility Prediction

Short-range coverage of EC access points will lead to multiple handovers due to users' mobility. These multiple handovers will turn mobility management into an essential aspect of service provisioning. These handovers may add significant overhead to use communication and computation infrastructure, depending on their implementation. One way to mitigate this issue is by exploring historical data about users' trajectories to enable proactive handover mechanisms. For instance, for a communication handover in an EC setting with SDN, the flow tables of the forwarding switches can be updated before the users even enter a specific access point. In terms of service migration, data prefetching can start loading service dependencies (e.g., software libraries) and also deploying services, thus making them ready for users and reducing migration downtime. Recent studies have shown a significant impact of ML models in mobility management [229], causing different methodologies for predicting users' mobility to appear in the literature using, for example: Markov Chain Models [194], Reinforcement Learning [277], and Deep Learning [109].

2.5.2 Service Migration

EC-enabled services are pushed forward because of their closeness to users. However, when users move, the host where services run may become far from them, which might be critical for some applications [66, 39]. For instance, optical X2 links for backhaul in mobile networks are expected to have latency of ≈0.3-0.5 ms when operating between 40%-70% of traffic load [142]. In this scenario, round trips with three or four hops adds a few milliseconds to respond a request, when considering also other delays in processing and in the wireless channel it might be challenging to meet the expectations of applications that demand (ultra-)low latency combined with high throughput, high reliability, or low tolerance to latency jitter. VMs and containers are technologies expected to handle the fast deployment of services to allow live migration to keep them running near users. Each methodology for virtualization has its advantages. VMs provide a more isolated and secure environment for services [159], while containers are more lightweight and have an overall better performance [44]. However, most of the available studies to compare these technologies do not profile them thoroughly. These studies lack awareness of the possibilities in terms of virtualization architectures. Different architectures may have an impact on the performance of the services. Also, according to the virtualization technology, various possibilities of migration strategies have been proposed and adopted, but comparative studies on the migration feature are still needed.

2.5.3 Service Caching

Mobile services may require different types of resources and have different requirement levels of QoE, which make this problem different from content caching (e.g., Content Distribution Networks (CDN)) [266]. For instance, VR applications may require a reasonable amount of processing power but a reduced amount of memory; compared to data collection tasks that may need more storage and not so much CPU and memory. One approach to enhance physical resource allocation for service caching is to perform *spatio-temporal* *popularity-driven service caching* [160]. Content and service popularity are modeled using the Zipf model [292, 165] to when evaluate caching solutions. Popularity-driven caching is used to predict which types of services are more used in a given location at a certain time. Thus, service shutdowns (i.e., stop a virtual instance of a service and clean unnecessary files from the host) may be prevented when predicting upcoming usage. This type of approach may also aid in the process of service migration combined with mobility prediction. Different service components can be stored for a more extended period in case a user is migrated to a specific host; popular services can be kept running in this scenario, reducing time with re-deployment. Besides controlling the life cycle of the service instances, in order to run stateful services in geographically distributed nodes, state data replication may be required [73]. Different from regular contents, state data is often more volatile, which may demand complex solutions to ensure availability and coherence.

2.5.4 Service Authorization and Session Support

Due to mobility, while changing access points, and performing handovers, applications have to perform multiple authorizations and exchange access keys to grant access and keep user sessions. This process may add significant overhead to consume services at the edge, which leads to a necessity of shared trust domains or alternative security protocols for accessing distributed services. Ideas to embed authorization in networking are present in Data-Oriented Network Architecture (DONA) [123], yet there is a lack for support of distributed and federated sessions. CCNx Key Exchange Protocol (CCNxKE) [173] proposes mobile sessions in CCNx. However, the mobility in this protocol is handled by exchanging keys (i.e., migration token) which still results in network overhead in this process. Session support has also been studied in SCN [80], yet it relies on consuming the service from the same host infrastructure and does not support the mobility of the sessions.

2.5.5 Service Load Balancing

EC offloads computing-intensive tasks from simple devices to idle resources in its resource pool. Resources in this pool might be in mobile network base stations or edge devices. In this second group, these devices have a limited communication radius that, given user mobility, may reduce the available time to use that resource. This problem becomes even more complicated when both consumers and resources are mobile entities. In this scenario, the communication links lifespan can become longer or shorter depending on their mobility patterns. Evaluation, clustering, and classification of these mobility patterns, which use this information for task offloading, are challenges for EC deployment. Research has been done in this direction for task offloading in the presence of mobility [242, 152]. However, most of these studies address the problem in single networks. EC will be achieved by the usage of a series of heterogeneous networks, which increase the complexity of the task offloading process. Models to understand the relevant metrics to decide when to migrate service instances have to be developed, for instance by considering ML models [245, 49].

2.5.6 Distributed File Systems

In Cloud Computing, different distributed storage mechanisms have emerged to address scalability, performance, and reliability issues and provide virtually infinite storage services in distributed resources. Ceph [258] is a commercially adopted solution to create such a storage service that relies on a pseudo-random function to distribute data to resources. Using this function, users can evaluate the location of the data, which saves resources to perform searches. Although Ceph is a well-known solution, it was designed to work in a scenario with plenty of resources, which is not a reality in EC cases. Limited bandwidth at the edge allied to massive access by users and connected devices makes the process of creating a fully-distributed EC-oriented file system difficult [157]. In the literature, some strategies for these file systems are already emerging, for instance snapshot and synchronization techniques to transfer disk state in the network [187]. Also, FogStore [87] is an EC-oriented file system that uses key-value storage and a relevance system to guarantee low latency in file access. Reliable mechanisms to provide file systems at the edge are still under study. One studied issue is how to perform redundant storage to ensure reliability while also not overusing resources. Duplicating contents to selected locations may be a solution to this issue. Another solution is to apply data deduplication [285], which identify intra- and inter-document duplicated data blocks and store these blocks only once. Fewer blocks (i.e., less data) to be stored can facilitate the process of deploying a distributed file system with reliability assured by copying blocks at selected locations.

2.5.7 Data Privacy and Federated Learning

Data leakage points are reduced when processing data at the edge since this processing happens closer to the data generation and fewer data transfers are needed. Still, similarly to what happens in the cloud, processing user data in public servers at the edge introduces issues on data privacy. Such problem becomes more apparent when using this data to train different ML models, which sometimes requires this user data to be shared across multiple servers. One possible solution that emerges to tackle this issue is Federated Learning (FL) [146], in which ML models can be trained locally in devices and only share its learning updates (i.e., model weights or gradients) with centralized servers, instead of sharing the raw data. These centralized servers can then collect multiple updates from several devices and aggregate them to produce a global model. By training the model locally and avoiding data sharing, FL reduces the number of points where user data may leak. However, personal data sometimes may still be extracted from the trained model with model-inversion attacks [94]. Besides not fully resolving the data leakage problem, FL still lacks maturity in other aspects. For instance, models trained with FL tend to have convergence problems resulting from the non-identical distribution of the data produced by each device [290]. On the communication perspective, when dealing with mobile devices some problems may arise. For example, when training an online model with FL with a large user pool, it is a common practice to select some users to be the ones feeding the global model with the learning updates. However, in mobile scenarios users may disconnect from the network or not be available for a certain period of time. Reducing the pool of selected devices for training may impact the training quality of the model. Furthermore, even when these users can return to the network, most FL approaches are synchronous [34], thus training has to be halted to wait for the updates of these users.

2.5.8 Scalability

Most of the issues for EC infrastructure arise from the existence of many connected devices creating tremendous amounts of data to be consumed. However, creating such a platform to support all this load is not an easy task. First, a plethora of heterogeneous devices will be integrated into the pool of resources. In this scenario, an interoperable architecture to control these devices is a requirement to allow the construction of EC. Second, multiple technologies to run EC are being proposed in the literature, as shown in Section 2.2. While most of them will not make it to the real world, many will need to be integrated, thus raising questions about the compatibility of devices, algorithms, and protocols. Even the integration between SDN, NFV, and ICN paradigms is not well defined in the literature, although these technologies are expected to run together in the future. Finally, the Cloud Computing paradigm has become mainstream due to the offer of IaaS solutions by big IT companies, such as Amazon, IBM, and Google. Market driving forces to produce a similar platform based on EC are studied [77]. However, a complete architecture to elastically serve this infrastructure for companies to deploy their service is unknown to the best of our knowledge.

2.5.9 Ubiquitous Service Provisioning

Applications and services consumed over the Internet are used as facilitators to perform activities that were previously only possible using physical means such as sending a letter, buying products, and attending a theatre play. As Internet technologies evolve, more activities are performed online. At stage 1 of service provisioning depicted in Figure 2.1, humans had to formalize instructions via messages through mostly textual interfaces. Later, these text interfaces evolved to graphical browser and phone-based interfaces and also virtual environments in video-games in stage 2 to facilitate the way humans interact with the applications. Recently, the advances in IoT and EC started to allow the direct control of devices to act over the real-world in stage 3. This last approach uses a virtual world as a middleware. Humans input instructions to a device, which modifies a virtual world, and the changes made to it are replicated by other devices (i.e., actuators) in the real world using the Cybertwin paradigm [273].

The bridge towards stage 4 of ubiquitous service provisioning will be built with the development of technologies that blend virtual and physical worlds. Important enablers for ubiquitous service provisioning are the Mixed Reality (XR) technologies [122, 45] such as AR, VR, and 360° videos. While multiple of these devices can already be commercially acquired, there is still a significant barrier to be surpassed in terms of user experience and freedom of movement. Most of these interfaces are developed in head-mounted, glasses, and other wearables devices with limited capabilities and sometimes mobility. EC and related communication infrastructure are expected to allow applications to meet the increasing QoE requirements from users by providing a reasonable resource pool accessible with very-low end-to-end latency, while also dealing with mobility-related issues. With immersive experiences applications such as the Metaverse [58] gain attention, which foresees the development of multiple 3D immersive virtual worlds. The popularization of immersive virtual worlds also brings interest on moving entities, objects and sensations in and out of the virtual reality with neural interfaces [209, 162, 84], immersive projections [230, 99], holographs [163, 227], and haptic communications [113, 216, 35]. New technologies easing the process of moving entities between different realities will then allow the emergence of novel and complex applications, such as the Tactile Internet [21] and the Internet of Skills [185, 21]. These applications are awaited to disrupt the way humans interact by allowing the transference of abilities, skills, and knowledge over the Internet.

2.6 Chapter Conclusions

Networks and the Internet are constantly evolving, and for every stage of evolution, different challenges have to be overcome. In the present stage, service provisioning is expanding from the centralized cloud to the edge of the network, which raises various questions as presented in this chapter. Also, in the future, new paradigm shifts will raise different questions about how to better provide Internet services. This chapter shows an overview of the main paradigm shifts experienced in the Internet and the application evolution that have pushed their changes. Also, we discussed the current state-of-the-art for the provisioning of in-network mobile services. We discussed many recently-proposed enabling technologies and also highlighted some applications that use these technologies. Furthermore, we presented incoming challenges for broad adoption of EC technologies in the near future and even upcoming opportunities to researchers looking for the next stages of evolution in service provisioning. We believe technologies such as SDN, NFV, and ICN have a significant role in deploying EC and future service provisioning paradigms.

Different technologies and architectures have been developed to fullfil the requirements of emerging and upcoming services and applications. The resource pool includes computing and communication infrastructure deployed in the cloud-edge continuum, and also user equipment such as the vehicles. All these technologies are expected to coexist and to be accessible to be used by different service providers in a similar fashion of cloud solutions nowadays. The interoperability, control and management of such a complex platform will rely on high levels of virtualization and orchestration of services, containers, VMs and communication infrastructure. These orchestration platforms will make use of multiple Machine Learning solutions to automate decision taking and automation of tasks.

The idea of EC emerged to bring computing power closer to users in a context where accessing cloud resources could take up to a few hundreds of milliseconds. Nowadays, latency to access Cloud Computing has reduced, still only EC will allow the support of emerging applications with requirements of ultra-reliable very-low latency with low jitters and high data throughput. For instance, a reliable and near-deterministic support of very-low latency for wireless communications can enable the provisioning of real-time services in which task deadline meeting is critical. EC must be able to operate isolated from the cloud in case of link failure and maintain services running. Furthermore, computing tasks running closer to data generation will reduce the amount of data sent to the core of the network. Reducing this data movement improves communication resource allocation and reduce the number of points for data leak, thus reducing concerns about data security and privacy. Finally, EC architectures will allow a better exploration of the spatio-temporal locality of data generation and service consumption. Innovative networking and computing solutions are emerging that exploit this locality in order to better provision services, such as the new networking paradigms for Future Internet (e.g., Geo-centric and Information-centric networking).

While helping to conceive, design, evaluate, and test EC infrastructure, academia has another important role on understanding the impacts caused by these technologies on society and envisioning what will be the next applications for service provisioning over the Internet. The popularization of immersive devices has created a trend for immersive experiences and put a lot of attention in applications such as the Metaverse and other possibilities of immersive-first applications such as the Internet of Skills. While difficult to predict whether these applications will penetrate society as the Internet and mobile devices, they have possibilities of highly impacting the traditional ways of human interactions. Possible use-cases for such applications are still to be unveiled and can lead to interesting research topics.

Chapter 3

Related Work

In the present chapter we will introduce the main related works in the scope of the present thesis. A wider overview of the related literature in the scoipe of this thesis was presented in Chapter 2. In the present chapter however, we focus on more closely related studies with similar goal. We start the discussion with studies that were used as foundation to developed the strategies proposed throughout the thesis. Section 3.1 discusses strategies from the literature to implement RACH-less handovers. These strategies are important because they are used in the context of Objective O.1, as later discussed in Chapter 4. After discussing these foundation technologies, we present studies that have similar objective to our thesis. The service management solution developed and evaluated in this thesis has three main components related to each of the specific objectives described in Chapter 1. We discuss in this chapter studies related to each individual component. Section 3.2 describes related works to user mobility management using Software-Defined Networking (SDN), regarding Objective O.1. Section 3.3 discusses studies that combine SDN and Service-Centric Networking (SCN) features related to Objective O.2. Section 3.4 shows works that discuss the management of stateful services considering the mobility at the edge of the network, concerning Objective O.3. Finally, Section 3.5 we summarizes the works presented in this chapter and highlight the main differences from them to our proposal.

3.1 RACH-less Handover Strategies

The bottleneck of user mobility in cellular networks happens when the User Equipment (UE) has to send its first message to a target Base Station (t-BS) to which it is not connected. When the UE is connected, the Base Station (BS) schedules communication windows structured in frames and subframes [63] to allow every UE to transmit their messages. A user not connected to a BS is unknown and, therefore, not expected. Thus, this user has to perform a random access to request an access grant to send messages over the communication channel. This procedure, known as Random Access Channel (RACH), is the most time-consuming operation when a user attaches to a t-BS. This RACH procedure consists of a four-message handshake between UE and BS to exchange information for timing synchronization and provide an uplink grant to transmit messages for the UE [129]. On average, this process takes around 10-12 ms during a regular handover procedure, which usually has an average duration of 45 ms [1]. During the process of detachment and re-attachment between BSs, the connection of the user is broken, and the UE cannot send messages nor consume services.

Different approaches emerged in order to create seamless mobility management schemes while users switch access points. For instance, The 3rd Generation Partnership Project (3GPP) studies the usage of modern handover mechanisms that do not require the RACH procedure, namely, RACH-less handovers [2]. A RACH-less approach consists of acquiring information for time alignment without performing the RACH, thus saving time with this procedure. Different strategies exist to perform RACH-less handovers, such as using multiple antennas connected to different BSs [2] or using synchronized networks that do not rely on Timing Alignment (TA) at all [27]. One strategy to implement a RACH-less handover with a single antenna on the UE and without the need for a synchronized network was proposed by Choi and Shin [52]. It consists of using the connection of the UE with a previous Base Station (p-BS) to exchange timing references between both BSs and the UE to make it possible to perform the TA before the UE needs to detach from p-BS. For this, the authors model Round-Trip Delay (RTD) of messages exchanged between BSs and UE. Figure 3.1 presents the delays in the RTD. Two timelines are presented for a BS and a UE clock. The blue and green rectangles represent transmission delays Δ. C_{ul} and C_{dl} are the time references to the start of up-link and down-link subframe transmissions, and δ is the TA used by UE to align messages with the start of the up-link subframe at the BS. Finally, M is the UE information from analyzing downlink messages from both BSs.

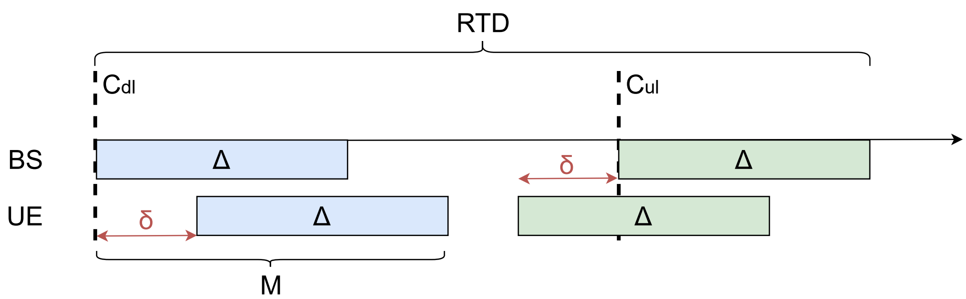

Figure 3.1: Timing diagram of RTD in cellular networks

The transmission delay difference between t-BS and p-BS can be evaluated by subtracting the downlink subframe duration from the RTD for each BS, which results in

$$\mathcal{D} = \mathrm{RTD}^t - (C_{ul}^t - C_{dl}^t) - \left[\mathrm{RTD}^p - (C_{ul}^p - C_{dl}^p)\right], \tag{3.1}$$

where $\mathcal{D}$ is the transmission delay difference and the superscripts t and p stand for t-BS and p-BS. Since the UE can measure $M = C_{dl} + \delta + \Delta$ from the down-link of both BSs, we can isolate the difference $M_D = M^t - M^p$ from Equation 3.1, obtaining

$$\mathcal{D} = 2 \times M_D + C_{ul}^t - C_{dl}^t + C_{ul}^p - C_{dl}^p. \tag{3.2}$$

With the transmission delay difference $\mathcal{D}$ between the BSs, UE can evaluate TA to align up-link messages. Since M_D can be measured, the UE only needs some Cell Timing Information (CTI) to perform the TA, which can be obtained as:

$$CTI = C^t_{ul} - C^t_{dl} + C^p_{ul} - C^p_{dl}. \tag{3.3}$$

This information is a composition of internal clock references from both BSs and can be sent to UE before disconnecting from p-BS, thus avoiding the RACH procedure. Choi and Shin [52] also consider offsets to compensate for the processing delays in both BSs to compose CTI .

Besides RACH-less handovers, for certain situations, the usage of Make-Before-Break (MBB) handovers is considered [2], which allows the UE to maintain connectivity with p-BS until the moment when all communication setup with t-BS is ready for use. This approach allows the UE to maintain service connectivity longer during the process of detachment from p-BS, allows a faster re-attachment to p-BS, and also allows safely aborting the handover when problems occur, as the UE never actually detached from its previous Access Point (AP). Besides these advantages, MBB is important because it can be combined with a RACH-less handover procedure, such as what happens in [52]. This combination can also be used to allow the sending of an up-link communication grant to the UE before detachment from p-BS, further reducing the time spent to re-establish communication after the handover.

3.2 SDN-enabled User Mobility Management

SDN can be used to develop handover schemes to enhance mobility management solutions. The possibility of decoupling the logic and forwarding functions of the network, and the general knowledge about conditions of network elements are interesting features of SDN that can be used to build such solutions. The authors in [30] propose two SDN-enabled handover mechanisms using these features to enable continuous provisioning of services running at the edge. They propose two protocols that allow the exchange of signaling messages needed for the handover using a reactive and a proactive approach, which we further discuss in section 3.2.1 and 3.2.2 respectively. These handover schemes have a similar goal to the study performed in the context of Objective O.1. When studying this objective, we proposed an SDN-enabled handover scheme that is described and compared to these proposals in Chapter 4.

3.2.1 SDN-enabled Reactive Handover

In a reactive handover, all SDN procedures related to re-establish the communication of the UE are performed after the handover attachment is finished. Thus, the UE disconnects from p-BS, connects to t-BS, and in the case of SDN-enabled handovers, communication path changes are then installed in the network. Figure 3.2 [30] shows a reactive handover strategy. The main roles in the handover belong to UE and t-BS, and p-BS has only a passive role in the procedure. The reactive handover starts executing all steps in a traditional handover, represented in the L2 attachment box in Figure 3.2. The SDN procedures needed for the communication re-establishment are only executed after this L2 attachment when the following signaling operations are executed:

1. the procedure starts with the UE sending a Router Solicitation (RS) to t-BS, after it is already attached to it;

2. when t-BS receives the RS message, a Proxy Binding Update (PBU) message is then issued to the controller;

3. once the controller receives the PBU message, it starts to install communication flows in the network in order to update the communication paths and re-establish communication; it also sends a Proxy Binding Update Acknowledgement (PBA) to inform t-BS that the communication path updates are taking place;

4. after the communication paths are updated, data received and stored at p-BS during the handover is forwarded to t-BS;

5. after receiving the PBA message, t-BS issues a Router Acknowledgment (RA) to the UE to inform that communication can be re-established;

6. finally, forwarded data from p-BS to t-BS can be sent to UE, from this point UE can communicate normally.

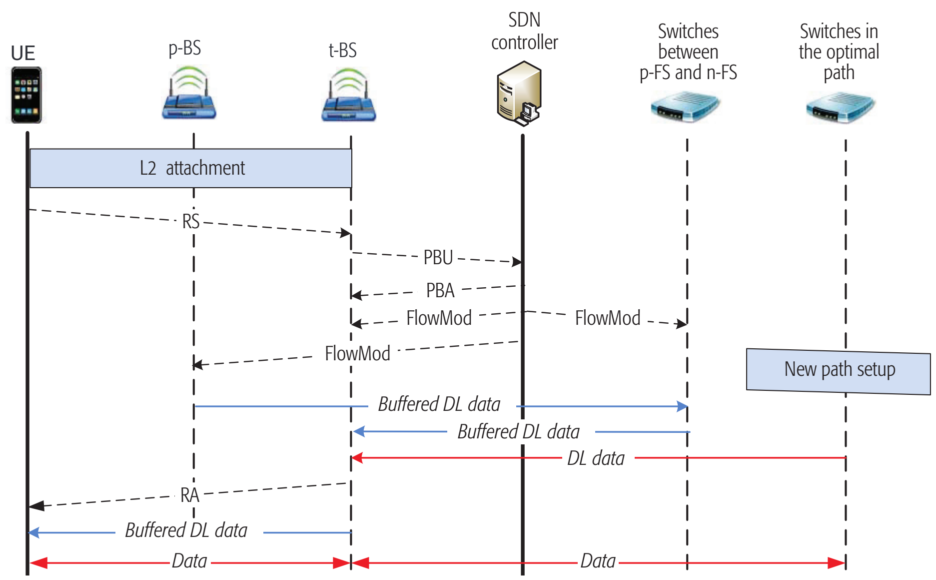

Figure 3.2: Reactive Handover in Software-Defined Networks [30] © 2018 IEEE.

Reactive handover schemes lead to longer Handover Execution Times (HETs), i.e., the time taken for a UE to complete connection transfer from one BS to another, since UE disconnects from p-BS without prior preparation. However, reactive approaches are important since most proactive strategies depend on the existence of an overlapping coverage zone of the BSs. When this zone does not exist, or communication with p-BS is not possible due to other factors, reactive schemes are used as a fallback.

3.2.2 SDN-enabled Proactive Handover

In the proactive handover proposed in [30], SDN-related procedures to re-establish communication start before the UE attaches to t-BS. In this scheme, UE sends a final message when detaching from p-BS that is used to start the handover. After the detachment from p-BS, the UE starts the attachment to t-BS, while the communication paths are updated in the network as shown in Figure 3.3. Then, the following steps are executed:

1. after UE disconnects from p-BS, a final L2 report message is sent to signal the start of the handover procedure;

2. when p-BS receives this L2 report, a Deregistration Proxy Binding Update (D-PBU) message is issued to the controller to allow the installation of the communication paths;

3. as the controller receives the D-PBU, the installation of communication flows starts with multiple Flow Modification (FlowMod) messages sent to the switches in the network;

4. when UE attaches to t-BS the communication paths were already set, and data sent during the handover procedure is already being forwarded from p-BS to t-BS, UE will then send a RS to t-BS;

5. after receiving the RS, t-BS will start the process of proxy updates by sending a PBU to the controller;

6. the controller will receive the PBU and respond with a PBA to t-BS;

7. once PBA arrives to t-BS, a RA is sent to the UE to inform that communication is re-established;

8. finally, forwarded data from p-BS to t-BS can be sent to UE, from this point UE can communicate normally.

The proactive approach saves time by carrying out SDN-related procedures during the detachment and re-attachment of the UE. While this approach reduces the time required to re-establish communication after the attachment, the RACH procedure is still used in this scheme, which leads to only a small reduction of the total HET when compared to the reactive approach. The RACH procedure is the bottleneck of traditional handover schemes and is still present in these proposals of the literature [30]. In order to further reduce HET, one strategy is to avoid the execution of the RACH, which is used in our approach discussed in Chapter 4. In our proposal, we increase the number of signaling messages exchanged in order to allow CTI information to be exchanged between the BSs, thus allowing TA to happen before the handover, and removing the necessity of the RACH procedure [52].

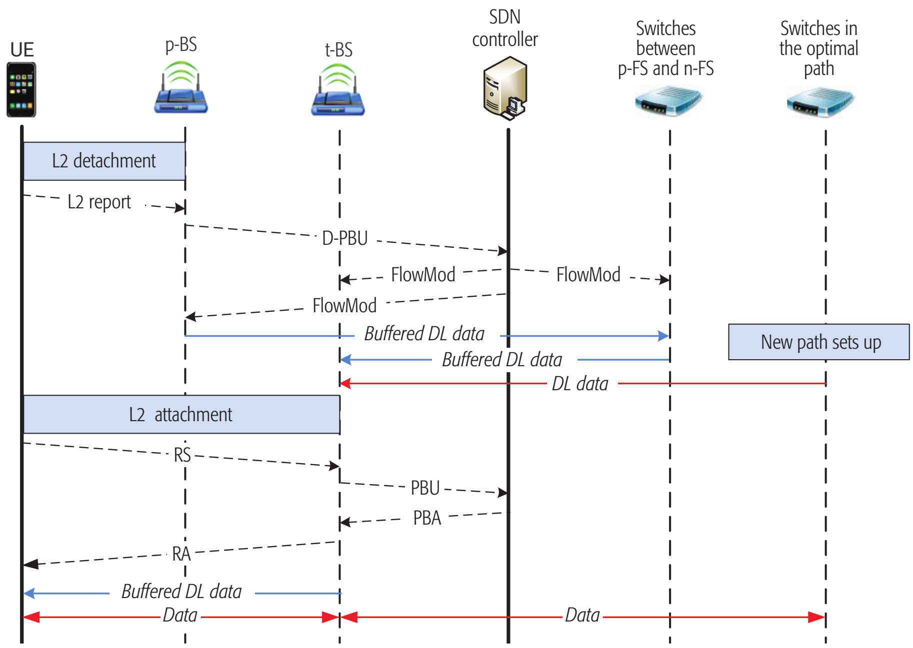

Figure 3.3: Proactive Handover in Software-Defined Networks [30] © 2018 IEEE.

3.3 Mobility-aware Networking

Some Mobility-aware Networking strategies were already discussed in Chapter 2, for instance, Host Identity Protocol (HIP) [175, 174], MobilityFirst [248] and name-based architectures, such as, SCN [37], Layered SCN (L-SCN) [81], Named-Function Networking (NFN) [243], and Object-Oriented Networking (OON) [147]. Besides those, there are still other important solutions in the scope of the present thesis. Ravindran et.al. [199] proposed the usage of name-based routing for high-level service and application provisioning at the edge of the network, such as enterprise applications and Internet of Things (IoT) services. The authors design a framework that combines SDN and Netwrok Function Virtualization (NFV) to realize SCN behavior. Other studies in the literature combine SDN and Information-Centric Networking (ICN) [282] and focus on two strategies. Firstly, on creating or extending traditional SDN protocols with routing fields and controller actions. For instance, Named Data Networking architecture based on Software-Defined networks (NDNS) is a routing protocol for Named Data Networking (NDN) that relies on SDN features when updating the traditional ICN tables. The authors propose additional tables not present in traditional ICN architectures to enable name-based routing over SDN, such as a Flow Forwarding Information Base (FlowFIB) that specifies actions to be taken according to name prefixes in requests, and a Data Information Base (DIB) that holds information of adjacent nodes connected to switches in the network. Or secondly on overwriting and re-signifying SDN fields to store ICN-related data allowing the emulation of ICN behavior without hardware and protocol changes. Software-Defined Information-Centric Networking (SD-ICN) [262] is a protocol that replaces traditional source and destination field in the Openflow protocol with hashes of names of network objects in order to route packages. In the present thesis, we used this second approach due to the possibility of emulating ICN with the already available features of Openflow.

Thus, our solutions were designed with the Openflow protocol in mind, yet our proposal is not bound to use Openflow since a different implementation could also work with other SDN solutions.

Xing et al. [262] propose a strategy to achieve ICN behavior on top of existing Openflow directives. The approach consists of using IP headers to store hashes of content names. Since the controller knows the position of the contents and can evaluate these hashes, it is possible to create the routes needed to consume these objects. Yet, this approach has two shortcomings: (i) the usage of both IP headers for hashing prevents the controller from getting information about the request origin, which impedes to evaluate content retrieval route; and (ii) the usage of simple hashing of the content naming, which leads to loss of prefix-based matching. In our proposal, we use a similar strategy of hashing service names to emulate ICN behavior over SDN. However, we keep information about the origin of the requests to allow the services to respond to the requests received. Also, we proposed a different hashing strategy that permits the usage of prefix-based matching. We compare our proposal to this solution in Chapter 5.

3.4 Distributed Application State and User Session Management

Whereas instances of stateless services can be replicated and deployed in multiple locations, stateful services are not interchangeable; thus, replication is more challenging [6]. These instances are not interchangeable because, depending on the current state of the application, they might provide different responses for the same request. For instance, regarding Long Term Evolution (LTE), the Serving Gateway (SGW) is a stateful function that runs in the 4G mobile core network, namely Evolved Packet Core (EPC). This gateway handles the state of the connections of the UEs when performing handovers. This state data includes IP address assignment, user mobility information, such as the ID of the cell the user is connected to, and information about the data consumption of that user. State data and user session data are the sets of information that can differentiate each service instance. This data will be referred to as Application State and User Session Data (ASUSD) throughout this thesis. When no synchronization is used for multiple instances of stateful services, these instances are not interchangeable because they have different states. While synchronization can be used to ensure the same data is accessible for multiple service instances, it comes with the cost of consuming network resources to exchange messages to allow the data to be kept synchronized.

At the edge of the network, if an instance needs to be relocated to a different host, ASUSD also has to be moved in a context transfer or service migration operation. These operations may lead to degradation in service provisioning or an increase in latency for service consumption. Therefore, different studies in the literature work on identifying strategies to manage this type of data for services running at the edge.

A more general mobility-aware state support mechanism was proposed for SDN [191]. The authors propose a flow rewriting mechanism that enables the user to consume the state from a static host. Yet, consuming from a static node may lead to an increase in latency. In the context of ICN, CCNx Key Exchange Protocol (CCNxKE) [173] proposes mobile sessions in CCNx. However, the mobility in this protocol is handled by exchanging keys (i.e., migration token), which still results in network configuration overhead in this process. Session support has also been studied in SCN [82], yet it relies on consuming the service from the same host infrastructure and does not support the mobility of the sessions. The authors propose adding a session token to the service name to establish a session with a given provider (i.e., host node). With this, the user can access the same session by consuming the service from the same host.

Filho et al. [73] propose a transparent mechanism to replicate state data in multiple hosts. The authors cover two main scenarios: (i) centralized state, in which the service should be stopped while the state is migrated; and (ii) distributed state, where a coherence mechanism is proposed that replicates all data update operations in all hosts. Both scenarios have issues. First, stopping services for long periods of time may be critical for some applications. Second, running data updates multiple times reduces scalability, consumes unnecessary resources, and may still lead to coherence issues when some instructions are lost.

Fondo-Ferreiro et al. [74] investigate how to migrate ASUSD from Cloud to Edge or between two different Edge nodes while maintaining service continuity. The users propose to (i) create a new instance of a service in a target host, (ii) replicate the state of the service, and then (iii) divert the traffic to the service using SDN. To migrate ASUSD, the authors propose to setup a new instance of the service at the destination, then replicate all requests received by the service to this new instance in order to produce the same final state. In our solution, we use a similar approach. However, instead of replicating all requests to a new service instance, we snapshot the original instance and replicate only the new requests received after this snapshot. This change can reduce the time for replicating the state by reducing the number of requests and processing time required to reproduce the actions related to each request. Also, these replicated requests are stored by the SDN controller; if fewer requests are required, fewer resources are used at the SDN controller.

Bao et al. [26], and Ouyang et al. [186] propose similar approaches in which users are closely followed by service instances that migrate between different BSs as UEs move. Bao et al. [26] study service migrations triggered whenever a user connects to a different BS. This approach leads to many migrations, which may eventually result in network overhead and service provisioning degradation. Ouyang et al. [186] propose a service placement methodology to minimize total perceived latency at the UE in the system. To avoid constant migration of service instances, the authors introduce migration costs and use them to balance the number of migrations with the latency. In this study, decisions about service migration are taken with a fixed frequency in well-defined time slots where all users send reports to a controller. At the end of every time window, the SDN controller estimates the near-optimal best UE-BS allocation for all users and implements it in the network. This approach may have issues in highly mobile scenarios because latency levels may vary excessively before the end of a given time window when service locations will be updated. Furthermore, multiple migrations of services simultaneously may overload the network. Our proposed approach operates asynchronously, responding to user mobility events, leading to a more specific control of latency for individual users while distributing the migration events over time.

3.5 Chapter Conclusions

The related studies discussed in this chapter were divided into three categories in sections 3.2, 3.3, and 3.4 relative to the specific objectives O.1, O.2, and O.3 respectively. While these objectives could be developed separately and obtain performance improvement in orchestrating stateful services at the edge, we expect the combination of the three strategies to achieve better results. Some of the proposed objectives pose more complex challenges than others. For instance, we expect to produce only minor contributions related to service naming. However, these minor contributions are important to the full composition of the target solution. The proposed naming solution aims to reduce the time consumed to update addressing configuration needed in IP networks – i.e., discovering and updating IP addresses.

The interdependence of the proposed topics can also be observed throughout the present chapter. For instance, when discussing state management in Section 3.4, many times addressing schemes (discussed in Section 3.3) have an important role in accessing this data. This highlights the lack of a complete solution for mobility-aware service orchestration at the edge of the literature. In this chapter, we show that specific contributions can be achieved in each area, and these contributions can be assembled and compose a robust solution.

Table 3.1 summarizes the studies discussed in this chapter. The column "Proactive" informs whether the proposal takes proactive measures to handle mobility events. The columns "User Mobility" and "Service Mobility" informs which types of mobility events are considered in the proposal. The columns "SDN" and "ICN" inform whether these technologies are used. The column "Addressing" informs if any strategy to improve addressing, i.e., dissociate addresses from network topology, is used in the solution. The column "Multi-host" informs whether the solution allows consuming services from multiple hosts. The column "State Migration" informs if the proposal allows state data reallocation. Finally, the column "Mobility Aspect" informs how mobility is considered when making decisions. The value "none" means that mobility is not considered, the value "independent" means that although mobility is considered, the actions taken do not depend on the position of users, the value "single" means that the proposal considers only current AP and next AP of the UE, and the value "multiple" means that the proposal considers besides current and next AP, also possible future APs.

Table 3.1: Solutions for mobility-related service orchestration at the edge of the network

Reference	**Proactive**	**User Mobility**	**Service Mobility**	**SDN**	**ICN**	**Addressing**	**Multi-host**	**State Migration**	**Mobility Aspect**
Bi et.al. [30]	✓	✓		✓					single
HIP [175, 174]		✓				✓	✓		independent
MobilityFirst [248]		✓			✓	✓	✓		independent
L-SCN [81],NFN [243],OON [147]					✓	✓			none
Xing, et.al. [262]				✓	✓	✓			none
CCNxKE [173]		✓			✓	✓	✓		independent
Filho et.al. [73]			✓				✓	✓	independent
Fondo et.al. [74]		✓	✓	✓				✓	independent
Bao et.al. [26]		✓	✓					✓	single
Ouyang et.al. [186]		✓	✓	✓				✓	multiple[a]
Proposal	✓	✓	✓	✓	✓	✓	✓	✓	multiple

[a]Considers information about multiple hosts but not the position, i.e., delay and energy consumption.

Chapter 4

User Mobility Management supported by SDN

Research Question 1: *How to orchestrate communication and service provisioning considering user mobility aspects?*

Future vehicular applications will depend on communication from vehicles to many other devices in their vicinity. One of the standards to allow this communication is the Cellular Vehicle-to-Everything (C-V2X) [257]. Still, current cellular handover schemes have to improve in order to cope with vehicular application demands. One way to enhance handover performance is by using modern handover strategies, such as avoiding the Random Access Channel (RACH) procedure during the handovers, namely RACH-less; or preparing all post-handover configuration to re-establish communication before the detachment of the User Equipment (UE) from the Base Station (BS), namely Make-Before-Break (MBB). The present chapter discusses a handover scheme that combines Software-Defined Networking (SDN) features with the RACH-less and MBB strategies [204][1] introduced in the context of the User Mobility Management (O.1) objective of this thesis. While studying user mobility in cellular networks, the main contributions discussed in this chapter were the design of a handover protocol that (i) combines SDN signaling with lower layer handover operations and (ii) allows the exchange of Cell Timing Information (CTI) between BSs in order to reduce the necessity of performing the RACH procedure. Our proposed scheme has a shorter Handover Execution Time (HET) when compared to other schemes from the literature [30] while keeping a reasonable message signaling cost.

4.1 Overview

Traditional handover schemes between different BSs today have an average HET of around 45 ms [1], which happens due to the RACH procedure that takes place during the handover. This procedure is a 4-way hand-shake between UE and BS that allows the UE to perform a Timing Alignment (TA) with a target BS. Since the user is not connected to this BS, it is not awaited and has to get a communication grant randomly accessing the

[1]Partially reproduced in this chapter – Copyright © 2011 IEEE.

channel and requesting it from the BS. This procedure is a bottleneck in the handover and is responsible for at least 10-12 ms in the entire procedure [1]. Besides that, current handover strategies force UE to fully detach from its BS before reconnecting to a new one. This approach further increases the time spent to re-establish communication after a handover.

In order to enhance the performance of cellular networks, The 3rd Generation Partnership Project (3GPP) has considered the design of RACH-less handovers, which involve the usage of multiple antennas in the UE or even consider synchronized networks. As discussed in Section 3.1, a proposal of RACH-less handover for non-synchronized networks that do not rely on the existence of the multiple antennas in the UE was proposed in the literature [52]. This proposal considers that CTI can be exchanged among both BSs involved in the handover, allowing the TA to happen without the need for the RACH procedure. Yet, this RACH-less procedure can only happen when UEs can reach both previous Base Station (p-BS) and target Base Station (t-BS).

Besides RACH-less handover, MBB handover is also expected to further decrease interruption times related to handover events by preparing all configurations needed for communication at t-BS before UE detaches from p-BS. One other factor to consider is the trend for SDN-controlled cellular networks. In an SDN scenario, centralized information at the controller can ease the management and adaptability of the network. However, in SDN, the controller must be aware of the attachment point of all users to perform routing, load balancing, among other controller tasks. This adds yet another responsibility when performing a handover in SDN networks, which is to update the knowledge base of UEs' Access Points (APs). The present chapter describes a handover scheme designed to cope with these limitations and characteristics of RACH-less, MBB, and SDN-enabled handovers. Our proposal, named SDN-enabled RACH-less MBB Handover (SRMH), is described in Section 4.2. Section 4.3 describes the general methodology used in the experiments in this chapter and also in chapters 5 and 6. We evaluate SRMH in comparison with two baseline schemes in Section 4.4. Finally, Section 4.5 gives some final remarks about the study exposed in this chapter.

4.2 SDN-enabled RACH-less MBB Handover

To allow the reduction of HET required to update the controller knowledge base and install network paths, signaling messages are triggered while the handover happens. Furthermore, the protocol is designed to exchange signaling messages carrying CTI to avoid the necessity of performing a RACH in regions where the UE can communicate with both its p-BS and t-BS. When the UE cannot reach both BSs, a handover is performed relying on the RACH procedure; both alternatives with or without the RACH are shown in Figure 4.1. The main flow of the diagram shows the RACH-less alternative. When RACH is used, the detachment and re-attachment to a new BS are performed together with sending the final L2 report.

In SRMH, instead of breaking the connection with p-BS after sending the L2 report that would normally trigger the handover, UE waits until it receives a Radio Resource

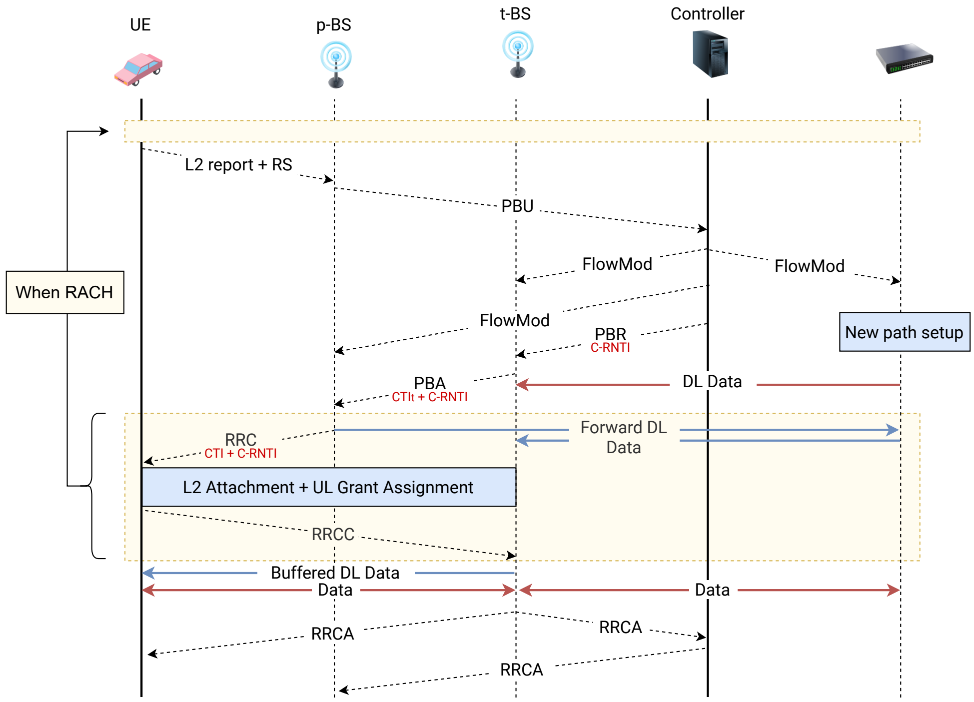

Figure 4.1: Workflow of the proposed handover mechanism [204] © 2021 IEEE.

Control (RRC) message that informs that network paths for the communication after the handover are ready and also brings required CTI for its TA [52]. Similar to the proactive handover discussed in Section 3.2.2, the whole process is started when the UE sends a final L2 report to its current BS. For the RACH-less procedure, the following steps are executed:

- After receiving the L2 report and Router Solicitation (RS), p-BS sends a Proxy Binding Update (PBU) to the controller.
- When the controller receives the PBU, it sends the Flow Modification (FlowMod) commands to set the communication paths and also a Proxy Binding Response (PBR) to t-BS. This message also contains Cell Radio Network Temporary Identification (C-RNTI), which is defined by the controller and needs to be known by both BSs.
- Once the PBR arrives at t-BS, this BS embeds its timing information CTI_t and forwards it to p-BS in a Proxy Binding Update Acknowledgement (PBA).
- After receiving the PBA, p-BS uses the CTI_t received and combines it with its own clock references to compute the final CTI, according to Equation 3.3. This CTI is sent to UE in the RRC message.
- The UE receives the RRC and performs the attachment to t-BS, it also negotiates the next uplink grant and uses it to transmit the Connection Reconfiguration Complete (RRCC) message indicating that the attachment is complete.

- When t-BS receives the RRCC, communication can be re-established, and buffered data can be forwarded to the UE. t-BS now has to emit RRC acknowledgments for the controller, p-BS, and the UE.
- Upon receiving the acknowledgments, the other entities involved in the handover finish the detachment process. This is important since these connections were kept alive to execute the MBB handover, and they can be finished just at this point.

When the UE cannot communicate to both BSs, the fallback handover, including the RACH procedure, takes place. Then, the detachment and re-attachment highlighted in Figure 4.1 are triggered at the start of the handover. Also, RS is sent to t-BS instead of p-BS. Besides that, the PBA message is not required because TA was already performed together with the RACH procedure. Therefore, the following steps are executed when a RACH handover is performed:

- UE uses RACH to connect directly with t-BS.
- After attachment, UE sends RS to t-BS.
- T-BS then issues a PBU to the controller to update information about UE and setup network paths.
- Upon receiving the PBU, the controller issues FlowMod commands and sets the new paths to and from UE.
- After issuing the FlowMod command, the controller sends the PBR to t-BS.
- When t-BS receives the PBR communication can already be re-established. t-BS still sends an RRCA to UE informing that communication paths were updated.

Compared with the baseline schemes [30], the differences in the messages sent impact the cost in terms of control messages sent over the network. This impact is caused because the messages have different destinations in SRMH and in the baseline scheme studied. HET and signaling cost are evaluated in Section 4.4.

4.3 Evaluation Methodology

In the present section we discuss the common methodology that was used in all experiments, not only from this chapter, but also in chapters 5 and 6. The performance evaluation of our solution was achieved using simulations with user and service mobility events at the edge of the network. All experiments were evaluated targeting vehicular applications, in which vehicles were connected to the network using Long Term Evolution (LTE) technology. The vehicular network scenario is appealing for our studies since it is a scenario that has high mobility standards, with speeds in the order of a few dozen kilometers per hour, and that has many latency-constrained applications, such as vehicular automated overtake (10 ms), pre-crashing sensing and warning (20 ms), and see-through (50 ms) [133].

In order to simulate vehicular mobility, we used a well-known urban mobility simulator, namely SUMO [124] (version 0.32.0). This simulator allows the creation of realistic mobility flows in different cities using openly available road network representations, such as OpenStreetMap [184]. There are also notable realistic mobility traces compatible with this simulator available in the literature, such as the vehicular mobility trace of the city of Cologne, Germany [244]. We used two mobility scenarios to perform the experiments later discussed in this thesis: a highway scenario and a scenario using the Cologne trace – specific details about these scenarios will be given in the respective sections when the results are discussed. In both cases, LTE BSs were positioned to allow the vehicles to consume the services, and a wired topology was generated to create a scenario where optical fiber connects these BSs. Table 4.1 shows specific parameters used for the simulation of wireless and backhaul links in the scenarios.

Table 4.1: Parameters used for the simulations [204] © 2021 IEEE.

Description	**Value**
Wireless channel	LTE model [251]
Transmission Time Interval (TTI)	125 μs or 1 ms
Wired link queueing delay	0.3 ms ($\pm$0.18 ms jitter) [142]
Switch processing time	75 ns + $n \times 5$ ns [192]

To simulate the network in our experiments, we used a traditional network simulator called Omnet++ [246] (version 5.6). This event-based simulator allows the modeling of various network components based on a modular abstraction. There are several frameworks and toolkits that facilitate the simulation of different technologies that run on top of Omnet++. In our studies, we used the three frameworks as shown in Figure 4.2: (i) INET [169] (version 3.6.8), a framework that makes available a series of models from different network components; (ii) SimuLTE [251] (version 1.1.0), a framework for LTE communication simulation; and (iii) Veins [226] (version 5.1) that works as a bridge to connect Omnet++ and SUMO simulators. The experiments described in chapters 4, 5, and 6 run on top of this simulation architecture, as depicted in Figure 4.2.

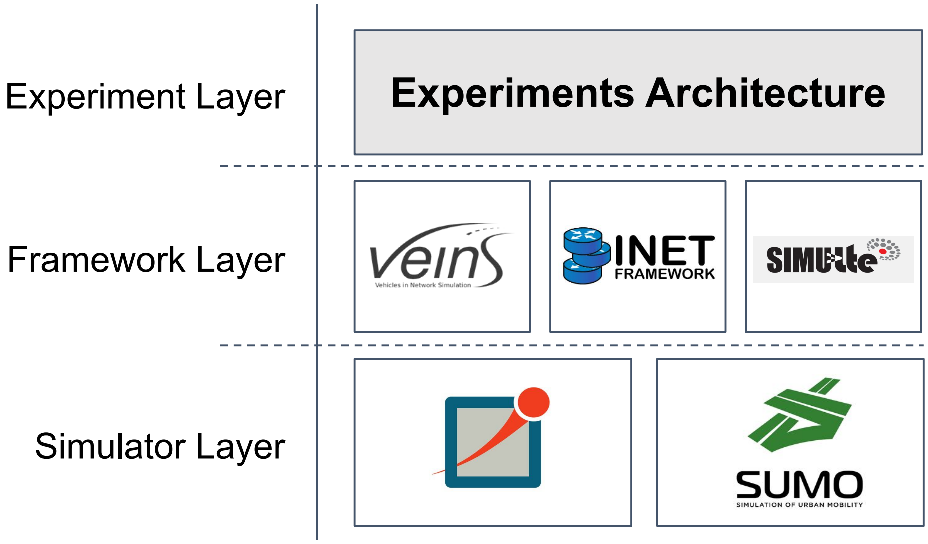

Figure 4.2: Overview of the simulation architecture used.

4.4 Performance Evaluation

In the present section, the performance of SRMH is compared to baseline proactive and reactive schemes from the literature [30] that were discussed in sections 3.2.1 and 3.2.2. We compare the solutions using two metrics, HET and control plane signaling cost; we also evaluate how scenarios with different coverage impact the performance of the handovers. The specific methodology for the results in this section is given in Section 4.4.1. The analysis of the HET is shown in Section 4.4.2. Simulation results of the system cost of the different approaches evaluated are shown in Section 4.4.3. Finally, Section 4.4.4 discusses about results obtained by varying the average coverage of the scenario.

4.4.1 Performance Evaluation Methods

HET is the time elapsed between the last message sent from UE to p-BS and the moment data communication was re-established with t-BS. To evaluate the system cost of a handover H, we consider the set M_H of control messages sent to perform H. Each message $m \in M_H$ has to be sent over multiple hops $e \in E$ over a path P_m in the network $G = (V, E)$. Every link e has a weight w_e defined according to the type of link, 1 for wired and 10 for wireless. Also, each message m has a size s_m displayed in Table 4.2. The cost of the handover H is defined as

$$\mathcal{C}[H] = \sum_{m \in M_H} \sum_{e \in P_m} w_e s_m. \tag{4.1}$$

The evaluation and comparison are made using the simulation stack and parameters described in Section 4.3. For the experiments discussed in this chapter, the scenario used is shown in Figure 4.3. The red line in Figure 4.3 indicates the path all the vehicles would take in the simulation. All circles represent BSs used by the UEs as AP. The light blue circle marks the BS that hosts the SDN controller, the red circle represents the BS that hosts the services consumed by the users, the black circles are fixed BSs in the topology, and the white circles represent BSs that were moved in, or removed from, the topology to achieve different values of coverage for the experiments discussed in Section 4.4.4. Furthermore, the summary of the main parameters used in the simulation is shown in Table 4.2.

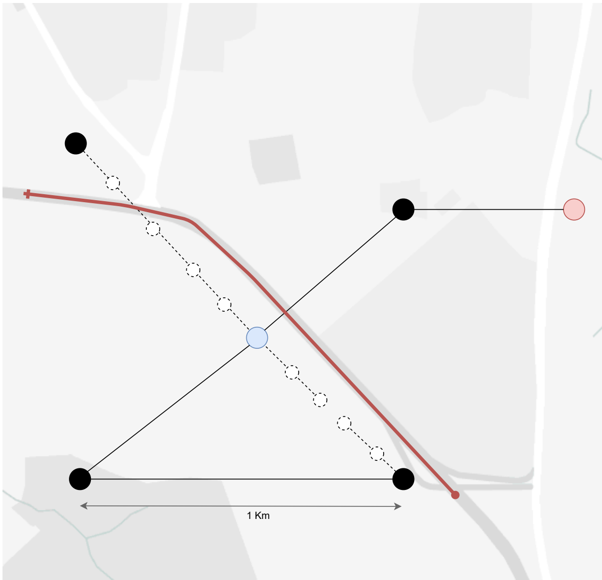

Figure 4.3: Network topology used in the experiments [204] © 2021 IEEE.

Table 4.2: Parameters used for the simulations [204] © 2021 IEEE.

Description	**Value**
Wired link cost weight	1
Wireless link cost weight	10
Flow Modification (FlowMod) message size	32 Bytes
L2 report size	52 Bytes
RS/RA size	52 Bytes
PBU/PBA/PBR size	72 Bytes
RRC/RRCA size	52 Bytes
Transmission Time Interval (TTI)	1 ms
Number of vehicles	100
Vehicle average speed	20 Km/h

4.4.2 Handover Execution Time Analysis

Figure 4.4 shows the complete distribution of HETs for the three approaches compared, the Reactive and Proactive approaches from the literature [30], and SRMH. Along the X-axis, the different handover schemes are displayed, while on the Y-axis, the observed HET in milliseconds. Both the reactive and proactive handovers displayed rely on using RACH to attach users to the BSs. Therefore, we observe these distributions above the 50 ms mark, which is around the average time consumed on a handover that uses RACH [1]. Our proposed handover scheme is composed of a proactive handover strategy with a reactive fallback strategy. This strategy is used when the UE cannot communicate to both BSs involved in the handover event, in which case a RACH procedure will be performed to allow UE to connect to t-BS. The existence of these two strategies created the bimodal distribution observed in Figure 4.4, the handovers that use RACH are responsible for the upper mode. In contrast, the RACH-less handovers are responsible for the bottom mode. In general, this distribution leads to a lower average HET in SRMH, as shown by the horizontal line in the middle of the distributions.

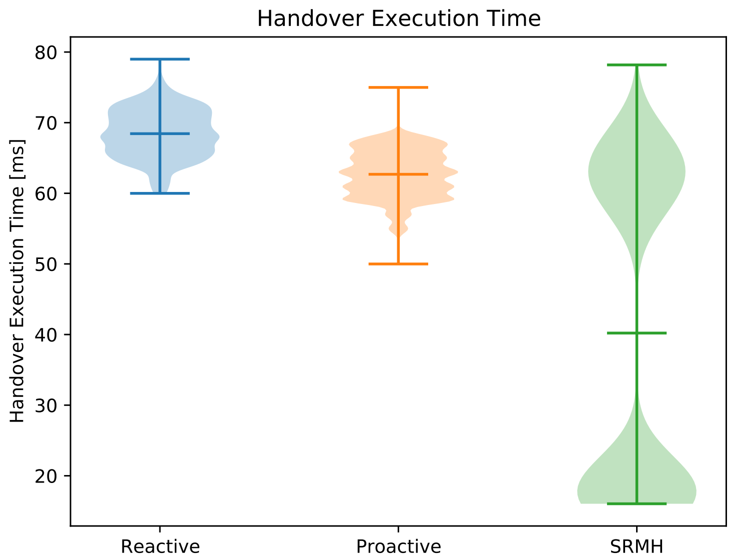

Figure 4.4: Handover Execution Time distribution.

4.4.3 System Cost Analysis

Figure 4.5 shows the distribution of signaling cost per handover according to Equation 4.1. Different handover schemes are shown along the X-axis, while the Y-axis shows the cost evaluated using Equation 4.1. The distributions of the three approaches have multiple modes. This happens because of the way messages are sent in the network. Since a simple topology is used for the experiments, many handovers occur between the same BSs, which causes the same value of cost to be observed many times. As the targets of these messages changes in each handover scheme, we observe multiple modes in Figure 4.5 indicated by the wider zones in the violine plot. Also, the amount of messages sent over the wireless link significantly impacts the distribution behavior because these links have a greater weight. It is important to notice that all vehicles are subject to the same BS transitions. Therefore, the mobility patterns do not have an impact on the cost observed. As shown in Figure 4.5, despite obtaining the highest values for cost among the three approaches, SRMH had an average cost that lies between the averages of the reactive and proactive baseline schemes, with the majority of the handovers in this interval.

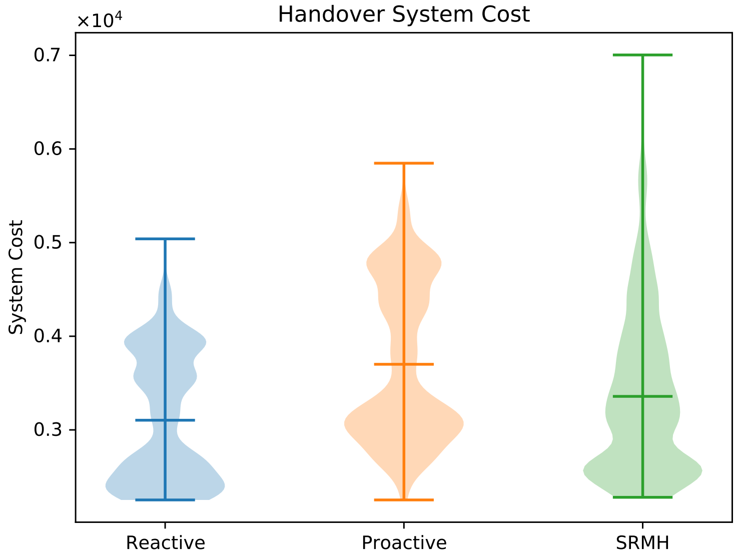

Figure 4.5: Handover system cost distribution.

4.4.4 Coverage Analysis

As discussed in Section 4.2, the proposed handover scheme operates in two modes, using the RACH procedure or not, depending on the reachability of p-BS and t-BS. This characteristic indicates that the coverage of the scenario may have an impact on the performance of SRMH. Therefore, the current section varies the coverage of the simulation scenario by placing BSs further apart from each other to evaluate this impact. The white circles in Figure 4.3 represent BSs that were added, removed, or moved to produce six different coverage configurations. The average distance between BSs varied in these configurations from 100 m to 600 m. These newly created scenarios had 13, 8, 5, 4, 3, and 3 BSs for each coverage from highest to lowest.

Figure 4.6 shows the variation of HET according to the changes in coverage. The average distance between BSs is shown along the X-axis, while the average HET is displayed on the Y-axis. Since the baseline schemes from the literature use RACH, there is no impact on the execution time depending on the coverage of the BSs. This can be observed in Figure 4.6, where the average HET stays almost constant for all coverage configurations for the two baseline schemes. On the other hand, SRMH is impacted by the changes in coverage. As shown in Figure 4.6, lower coverage impacts SRMH negatively since, more often, it will be necessary to use the RACH procedure. Nevertheless, for all configurations used, SRMH still showed a better performance than the baseline schemes.

Figure 4.7 shows the variation in signaling cost due to the changes in coverage of the scenario. On the X-axis, the average distance between BSs is shown, while on the Y-axis, the average cost per handover is displayed – again, this cost is evaluated according to Equation 4.1. For most of the configurations studied, SRMH had a behavior similar to the one observed in Section 4.4.3 where the cost seats between the two baseline schemes. We also observed a significant variation in the general behavior of the cost together with

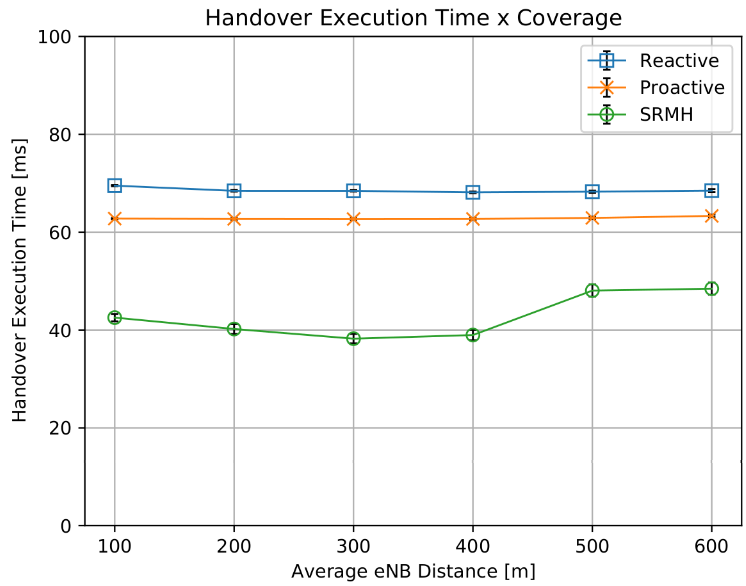

Figure 4.6: Comparison of average HET for different coverage scenarios with 99% confidence interval.

the coverage. This behavior is expected due to the different topologies of the network. A greater number of hops to be traversed in the network, induced by the existence of more network nodes, leads to greater cost values.

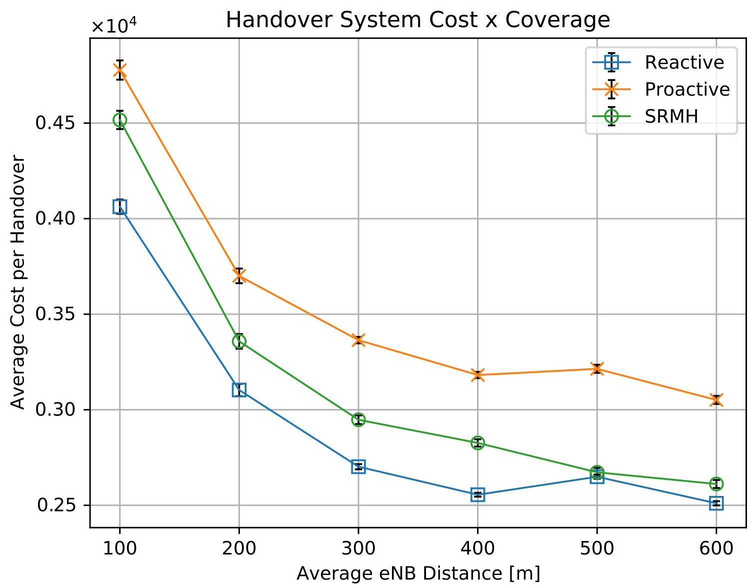

Figure 4.7: Comparison of average system cost for different coverage scenarios with 99% confidence interval.

4.5 Chapter Conclusions

The current chapter introduced an SDN-enabled handover scheme that assembles modern handover techniques, i.e., RACH-less and MBB. In order to enable a RACH-less procedure considering a non-synchronized network [52], the signaling protocol has to be re-designed to be able to collect and deliver the appropriate data, i.e., CTI and C-RNTI while also updating the controller knowledge base about position of the UEs in the network. We evaluated SRMH and compared it to baseline schemes from the literature [30] and observed that we could reduce average HET while keeping a reasonable cost with control messages. We also studied how BS coverage in the scenario could impact the usage of our proposed handover scheme. This was made since the RACH-less handover strategy for non-synchronized networks deepens on the reachability of the two BSs involved in the handover event. It was possible to notice that even with the negative impact of lower coverage, the proposed method still achieves better performance than the baseline schemes in terms of HET.

The present chapter shows the signaling protocol to implement the SDN-enabled RACH-less MBB handover. However, we do not discuss how the greater availability of information in the SDN controller could be used to improve the handover procedure further. For instance, information about position and mobility patterns of UEs could be used with Machine Learning (ML) algorithms to predict handover events and take proactive actions even before the event is triggered. Exploring this decision process could be an interesting next step for our study. Another possible extension of this work could be a more exploratory analysis of the scenario with different speeds or even more realistic mobility patterns. This is interesting since handover success rates may vary depending on the speed of the UEs.

After studying issues related to the management of the mobility of users using SDN, this thesis proceeds to explore the interaction and effects of user mobility and also service mobility when consuming services provisioned at the edge. In Chapter 5 a name-based protocol is proposed to mitigate the disruption caused by these mobility events.

Chapter 5

Mobility-aware Software-defined Service-centric Networking

Research Question 2: *How to handle network addressing updates after mobility events?*

Emerging applications, such as connected vehicles, have requirements that cannot be coped with using current networking protocols. Different architectures emerge in order to tackle these shortcomings. For instance, the usage of Software-Defined Networking (SDN) is widely awaited to enhance the management of different networks, as also other approaches such as Information-Centric Networking (ICN). In the present chapter, we discuss Mobility-aware Software-Defined Service-Centric Networking (SD-MSCN) [205][1], a network addressing scheme to pursue mobility-related issues when provisioning services at the edge of the network proposed in the context of the Mobility-aware Software-defined Service-centric Networking (O.2) objective of this thesis. SD-MSCN combines features of SDN and ICN to allow the proactive reaction to user and service mobility events in the network. We compare our proposal with other SDN-enabled IP and ICN solutions from the literature. The main contribution discussed in this chapter is the design of a name-based addressing scheme on top of SDN that has a good performance at reducing the end-to-end latency of requests sent after mobility events. When handling user mobility events, our proposal has a better performance compared to the IP-based reactive solutions but is similar when compared to proactive strategies. Additionally, when service mobility events are introduced, our proposal achieves notably better performance than the other strategies evaluated.

5.1 Overview

The mobility of users and services causes communication disruptions at the edge of the network. In the present chapter, our goal is to discuss a solution that changes the addressing scheme of the network in order to mitigate this mobility-related disruption. Therefore, we explore an ICN-inspired addressing scheme as an alternative to IP addressing. ICN advocates for the addressing of network entities based on interests to communicate to

[1]Partially reproduced in this chapter – Copyright © 2011 IEEE.

that entity rather than its topological location, i.e., network host. Regarding mobility, ICN-based architectures bring the advantage of separating addressing from the network topology by routing network objects using their names instead of the identification of their host. This way, after mobility events happen in the network, there is no need to update addresses in messages or stored by the users. The address of an entity is always consistent, which allows the routing of packages as soon as the routing tables are updated in the network.

While ICN-inspired architectures bring advantages when handling mobility events at the network, they also come with some challenges. The first challenge is the requirement for deployment of ICN-enabled hardware in order to implement the ICN on the network. One solution for this issue is by implementing ICN behavior on top of SDN [167, 263, 205]. Since SDN is already awaited to be used in the management of several types of networks, especially at the Edge [25] and with 5G [130], researchers consider this technology as a basis for deploying ICN solutions. Another issue of ICN architectures is the time taken to propagate routing information in the network, which is particularly a problem in wireless networks as the topology changes frequently [69]. This issue can also be tackled using SDN since the controller has the ability to inform routers directly about network changes in a timely fashion. Since these two features of SDN can mitigate issues of ICN, the combination of these two technologies is interesting in the scope of this thesis. Therefore, in our solution, besides describing the pure ICN-inspired solution, we discuss an SDN-based implementation in Section 5.2 and evaluate it in comparison with other IP and ICN architectures that also use SDN in Section 5.3.

5.2 Mobility-aware Service-Centric Networking

In this section, we discuss the idea of Mobility-aware Service-Centric Networking (MSCN) that consists of using ICN-inspired name-based addressing to route packages for mobile entities in the network. ICN-inspired protocols are interesting to address mobility since the dissociation between addressing of network objects and network topology generally eases routing [69]. When network entities are routed without considering their network position, it is possible to create a transparent mobility management for these mobile entities; these entities do not need to be informed when their counterpart changes its Access Point (AP). Despite easing mobility management, name-based routing has other advantages, such as removing the need for naming resolution and allowing optimized Service Instance (SI) selection [37].

Section 5.2.1 describes the concept of MSCN. Section 5.2.2 presents an SDN-based implementation, namely SD-MSCN, that is fully compatible with the Openflow protocol [182], the most important switch specification to enable SDN. Since SD-MSCN is based on a hashing strategy, we discuss the impacts of possible hash collisions in Section 5.2.3. Finally, the workflow of SD-MSCN is discussed in Section 5.2.4.

5.2.1 MSCN Concept

When using IP-based routing in a network, whenever a User Equipment (UE) is handed over to a different AP, network updates are required to maintain communication since the IP address in use is topologically bounded to the previous network. In general, cellular networks use the Mobile IP [189] protocol, in which the previous network is used as a proxy. Thus, packets first go to this network and just then are forwarded to the new AP of the UE. If name-based addressing is used, new packets can be routed using an optimal path as soon as the network updates are implemented in the network because the addresses are not bound to the topology. Reducing the number of packets sent through sub-optimal paths in the network can decrease communication latency after mobility events.

Similarly, when service mobility events occur while using IP-based routing, the address of the host of the service may have to be updated, which may further result in requests wrongly routed to the host where the SI was previously running. When this IP address update happens, UEs have to invalidate their Domain Name System (DNS) caches and get the new IP address to resume communication with the service. This process may lead to more complex mobility management since it may require the active participation of the UEs. When services are addressed by names, there is no need to invalidate their addresses, as the service names do not change after the mobility event.

MSCN allows the dissociation of the network topology from addressing by using name-based addresses inspired by ICN protocols. Another advantage of using name-based addressing is the possibility of the UE to consume the service from any instance in the network when multiple options are available. This allows a transparent load balancing of users without updating addresses but only routing tables in the network. Name-based routing still allows UEs to consume the service from multiple instances simultaneously or using multiple paths in the network. However, these possibilities are not studied in the scope of the present chapter. We only study MSCN in the context of handling mobility issues and improving the seamless provision of services at the edge of the network.

In MSCN, services are addressed using hierarchical names similar to NDN [10], where the name of a service $\mathbb{N}_k$ with k hierarchical components assumes the form

$$\mathbb{N}_k = \{c_1, c_2, ..., c_k\}, \tag{5.1}$$

where every component c_i has a different meaning for the routing. $\mathbb{N}_k$ can be represented as $\mathcal{N}_k \equiv /c_1/c_2/.../c_k$. For instance, one possibility of a service name is given in Example 1.

/x/com.provider/warn/-22.814/-47.064/25m

Ex. 1. A service name with parameters.

In Example 1, the first component, i.e., “x”, represents the MSCN strategy, which creates the possibility for the existence of multiple intra-MSCN strategies. These strategies allow the definition of different purposes for the remaining components of the name, thus creating more routing possibilities, for instance, when services have more specific requirements for routing. The following two components in Example 1 are the provider identifier of the service and the service name. Finally, the other components of the name store parameters for the service call that are controlled according to service provider business and networking rules. While in Example 1, many parameters are specified, MSCN only requires the first component to be fixed. All other components may have different meanings specified according to the strategy value.

One important difference of MSCN from existing Service-Centric Networking (SCN) protocols is the focus on the mobility of users and services. To also address the mobility of UEs, we introduce an approach inspired by MobilityFirst [248]. In MobilityFirst, mobile entities in the network are identified by Globally Unique Identifierss (GUIDs), which achieves the desired dissociation from network topology as these GUIDs do not have to be updated after mobility events. In our proposal, GUIDs are used only to address UEs and not the services, which allows us to not rely on a GUID resolution system. UEs are responsible for starting the communication with services. Therefore, there is no need for the service to resolve the GUIDs, as it is able to reply back to source address in the headers of the request received. This response can be routed by the network since the controller has the information about the attachment points of the UEs. Since MSCN keeps the information about the source and destination of the requests using service names and GUIDs for users, we also remove the need for traditional Forwarding Information Base (FIB) and Pending Interest Table (PIT) used in ICN protocols to forward back responses of requests. These tables are usually a concern when implementing ICN architectures since they demand the installation of specific hardware [69]. To avoid the necessity of a central unit to assign GUIDs to users, this process of GUID acquisition could happen locally in each UE, for instance, by generating random GUIDs with low collision probability. The advantage of these GUIDs to address UEs can be observed when the UE performs a handover while a request was sent to a service, but the response has not yet been received. Despite the handover, the source address in the request will remain accurate. Therefore, the response can be routed using the optimal path as soon as the network updates are implemented in the switches of the network. These network updates could be implemented proactively before the UE performs the handover in order to have the response already arriving to the new AP where the UE will connect. For instance, Section 5.2.2 describes an SDN-based implementation of MSCN, which takes advantage of the central controller to proactively update network communication paths to maintain service provision after handovers and service mobility events.

5.2.2 SDN-based Implementation

MSCN was designed to ease its implementation in SDN-enabled networks, specifically when using Openflow. Therefore, in this section, we describe SD-MSCN a fully Openflow-compatible implementation. When designing this implementation, a specific class of services was considered. We consider services that produce unique responses to requests, and, therefore, there is no need to cache these responses for reuse. This variation can be due to: (i) different service parameters; (ii) variation of application state with time; and (iii) authorization, or (iv) access control for users. As there is no caching of responses, the need for the traditional Content Store (CS) from ICN protocols is also removed. Therefore, we can fully emulate our expected behavior using Openflow-enabled switches. While it is a reasonable assumption that the existence of services where responses vary often enough to remove the need for caching, there might be services that could use caching to improve delivered Quality of Experience (QoE) to their users. It is still possible to implement a solution with caching using non-modified Openflow-enabled switches. For instance, caching nodes could be installed in the network to store the responses of the services. This would require the controller to have information about these caching nodes and create routes to consume cached responses from them. Yet, in this chapter, we do not consider this possibility and implement MSCN using already available Openflow [164] specification.

We designed SD-MSCN to use traditional Openflow matching fields to store information that represents service names. In MSCN, services and users are addressed using either names or GUIDs, which removes the need for IP source and destination addresses. Therefore, we give a different meaning for these IP fields and use them to store UE GUID and hashes of service names. Both the GUID and the hash can be created with the size of an IPv6 address, i.e., 128 bits, to be stored in these fields. Besides these fields, we also use the IP Proto field of Openflow to store a flag that informs whether the packet needs to be processed according to IP or MSCN rules. This is made to allow the coexistence of IP and MSCN routing on the same network, which allows exploring the most suitable protocol for a given communication. These changes, allied with a specific MSCN manager for the controller, allow the emulation of the desired behavior in the network that our proposal requires.

The usage of hashes replacing traditional fields in SDN-enabled networks was already proposed in the literature [247, 263]. However, these proposals still demand the extension of network switches with ICN specific features. Besides that, some proposals use flat hashes of network entity names, which may be a problem when routing services in which the hashes would vary dramatically when, for instance, some parameters at the end of the service name change. This hash variation would not allow the usage of prefix-based matching, which leads to an increase in the size of the routing tables in the switches and also an increase in the number of packets that require routing from the controller. Sending many packets to be routed at the controller is a problem since it generates control plane overhead and also controller scalability issues. To avoid these issues, we use a hierarchical hashing strategy, in which name components are hashed separately, and then these resulting hashes are concatenated. Thus, if a change in the parameters of the service is to happen, the hierarchical hash would only have a change in its final components, therefore still allowing prefix-based matching. The hashing space of every component is defined according to the strategy parameter stored in the first component of the name. Thus, the set of hashed components $\mathbb{H}_{\Lambda_K}$ using the strategy Λ_K of a name $\mathbb{N}_k$ is

$$\mathbb{H}_{\Lambda_K}[\mathbb{N}_k] = \{\lambda_1, h_{\lambda_2}(c_2), ..., h_{\lambda_k}(c_k)\}. \tag{5.2}$$

In Equation 5.2, c_1 assumes the value λ_1, which is a flag representing the strategy $\Lambda_K = \{\lambda_1, \lambda_2, ..., \lambda_K\}$ of address space sizes for a hash function h. Since $c_1 = \lambda_1$ is a controlled value, there is no need to hash it. The resulting hash $\mathcal{H}_{\Lambda_K}[I\!N_k]$ is

$$\mathcal{H}_{\Lambda_K}[I\!N_k] = 2^{S_{\Lambda_K}(1)}\lambda_1 + \sum_{i=2}^{K} 2^{S_{\Lambda_K}(i)} h_{\lambda_i}(c_i), \tag{5.3}$$

where

$$S_{\Lambda_K}(i) = \sum_{j=i+1}^{K} \lambda_j \tag{5.4}$$

is the number of bits taken by all the components after the i-th component – assuming the number of bits after the last component $S_{\Lambda_K}(K) = 0$. It is required that $K \geq k$ to obtain a valid hash. As observed in Equation 5.3, the final hashed result comprises hashes of all components. For instance, let $h_{\lambda_i}(x)$ be the first λ_i bits of the `SHA256` hash function of x, and $\Lambda_K = x_9 = \{\text{“x”}, 64, 8, 8, 8, 8, 8, 8, 8\}$. Then, the final hash for Example 1, using hexadecimal notation for the `SHA256` results, is

$$\begin{aligned} \mathcal{H}_{\Lambda_K}[I\!N_k] = \texttt{0x77} \times 2^{120} \ + \ & \texttt{0xb7dc56d9c459021b} \times 2^{56} \\ + \texttt{0xaa} \times 2^{48} \ + \ & \texttt{0xe8} \times 2^{40} \ + \ \texttt{0x0e} \times 2^{32} \\ + \texttt{0x08} \times 2^{24} \ + \ & \texttt{0x0} \ + \ \texttt{0x0} \ + \ \texttt{0x0}, \end{aligned} \tag{5.5}$$

which results in Example 2.

77b7:dc56:d9c4:5902:1baa:e80e:0800:0000

Ex. 2. Hash of service name in Example 1 formatted as IPv6 address.

Since the hashes of all components are concatenated, the controller can use prefix-based matching when routing requests in the network. For instance, the controller can install communication flows considering Example 3 as a prefix to all requests towards the service call `/x/com.provider/warn/*`. The aggregation of multiple requests achieved by using the prefix allows the reduction of the number of packets sent to the controller and also the reduction of the number of rules installed in the tables of the switches in the network. We can also identify from Equation 5.3 that collision probability when hashing may cause routing problems. Therefore, collision probabilities are discussed in Section 5.2.3.

77b7:dc56:d9c4:5902:1baa:::

Ex. 3. Prefix of service name considering hash in Example 2.

5.2.3 Address Space and Collisions

128-bits is a reasonably big address space with reduced collision probability, but since prefix-based matching is used for routing, even a collision in the initial bits of the address could already lead to routing disruption. This could lead, for instance, to requests to a given service provider being routed to another one. Although the remainder of the components does not collide, the request would be routed to a different provider unable to respond it. Thus, it is important to select SD-MSCN strategies to reduce the probability of prefix collision when hashing service names. The probability of collisions on the i-th component of strategy Λ_K when hashing m names is given by the product of the individual collision probabilities of every component up to i, as

$$P_{\Lambda_K}(i|m) = 1 - \bar{P}_{\Lambda_K}(i|m) = 1 - \prod_{j=1}^{i} 1 - Q(\lambda_j|m). \tag{5.6}$$

We always have $K \geq i$ in Equation 5.6, as i is one of the K components of the strategy Λ_K. $P_{\Lambda_K}(i|m)$ is the collision probability on the i initial components of the resulting hash, whereas $Q(\lambda_j|m)$ represents the individual collision probability of the j-th component alone with $j \leq i$. Both probabilities consider that m names are hashed. This result highlights that the collision probability is always higher for smaller i values, therefore, shifting the focus of reducing collision probability to the sizes λ_i of the first components. A similar reasoning of Equation 5.6 can be used to define $Q(\lambda_j|m)$ as

$$Q(\lambda_j|m) = 1 - \bar{Q}(\lambda_j|m) = 1 - \prod_{l=0}^{m-1} 1 - \frac{l}{2^{\lambda_j}}, \tag{5.7}$$

where 2^{λ_j} is the size of the address space for the j-th component – i.e., λ_j bits for addressing. Since $|\frac{-l}{2^{\lambda_j}}| \ll 1$, Q can be approximated using a first-order Taylor expansion [40] $e^x \approx 1 + x$ for $|x| \ll 1$ as

$$Q(\lambda_j|m) \approx 1 - \prod_{l=0}^{m-1} e^{-\frac{l}{2^{\lambda_j}}} = 1 - e^{\left[-2^{-\lambda_j} \sum_{l=0}^{m-1} l\right]}, \tag{5.8}$$

which leads to

$$Q(\lambda_j|m) \approx 1 - e^{-2^{-\lambda_j - 1} m(m-1)}. \tag{5.9}$$

Equation 5.9 is the collision probability for one component of size λ_j when hashing m names. Naturally, if we can control the values of a component, for instance, the strategy in component c_1, there is no collision as long as $m \leq 2^{\lambda_j}$, thus

$$Q(\lambda_j|m) \approx \begin{cases} 0, \text{ if controlled} \\ 1 - e^{-2^{-\lambda_j - 1} m(m-1)}, \text{ otherwise} \end{cases}. \tag{5.10}$$

Equation 5.10 allows the evaluation of collision probabilities when selecting the size of the hashed components of the names in a given strategy. These strategies differ from each other by the number of components and the size of each component. For instance, a generic strategy is represented by Λ_K, where K is the number of components. Using this notation, the set of all strategies with 9 components can be represented by Λ_9. To specify a strategy in this set, we need to define the sizes in bits of all components. These sizes could be defined, for instance, as $(\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5, \lambda_6, \lambda_7, \lambda_8, \lambda_9) = (8, 64, 8, 8, 8, 8, 8, 8, 8)$, a vector of integers that specify the size of each of the 9 components. For this example, we will represent the strategy with the flag "x" that must then be stored at the component λ_1. Thus, we use the first 8 bits to store the flag – λ_1 = "x". Using this flag, this strategy is defined as ("x", 64, 8, 8, 8, 8, 8, 8, 8) – a numeric value can be stored in λ_1, such as the ASCII code of "x", which is `0x78`. The following 64 bits are used to store the service provider hash, which could be used by a 64-bit long hash of a public key or an Internet domain. Finally, the other seven components of 8 bits are reserved for routing by the service provider, which can be used to create custom routing logic. Even the service providers have to design names that will reduce the probability of collisions in the prefixes, which can be done in different ways, such as: (i) removing parameters not related to routing from naming or (ii) using more components to hash service names and parameters if there are free components not used.

Using the second component with 64 bits leads the collision probabilities to increase significantly only after $m > 10^9$. In Figure 5.1 we compare different sizes of components λ_j to the collision probabilities to the hypothetical situation where we need to register 10 domains for each individual in the largest city in the world, which is currently Tokyo with around 37 millions inhabitants. The horizontal axis shows the number m of names hashed on a logarithmic scale. In contrast, the vertical axis shows the collision probability $Q(\lambda|m)$ given the λ selection and the number of names to hash m. The red vertical line is placed at the 370 million mark. The blue square shows the interception of the red vertical line with the probability curves in each graph.

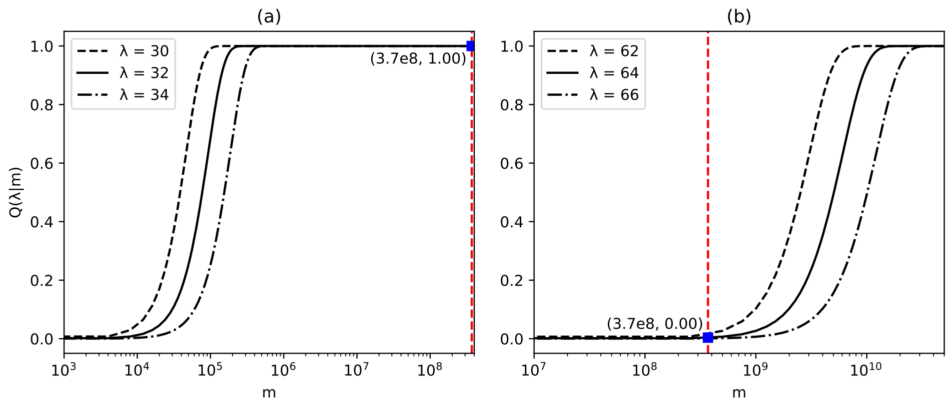

Figure 5.1: Comparison of collision probabilities for different λ values [205] © 2022 IEEE.

The spatio-temporal aspect of routing requests also plays an important role in the collision probabilities of SD-MSCN. Collisions only affect routing when they happen in the same network and at the same time. If a collision happens in two different networks, i.e., controlled by different controllers, these controllers will not even notice that the collision happened. Also, since the communication paths of the network are updated often, the same is valid if a collision happens between a hash in use and another that is no longer consumed. The spatio-temporal aspect then lowers the overall collision probability when using SD-MSCN.

As mentioned before, the strategy Λ_K can be changed to adapt SD-MSCN to different use cases, with only the requirement of reserving the initial 8 bits to store the value that represents the selected strategy. For instance, when providing a virtual reality service aiming to explore cached views of the space, i.e., cached service responses, another strategy $\Lambda_K = \upsilon_4 = \{\text{“v"}, 56, 8, 56\}$ can be used. In this strategy, c_1, c_2, and c_3 still represent the strategy, the provider, and the function, although fewer bits were used to represent the provider because $\lambda_2 = 56$. c_4 is used to allow exploration of parameters since its hashing space is $\lambda_4 = 56$ bits. This better exploration of the parameters can happen because they can be stored together in the final 56-bits long component (i.e., addressing space $\approx 7.2 \times 10^{16}$). Example 4 shows the use case in which all parameters are in a single component. Furthermore, even finer exploration of the addressing space is possible by using collision resolution measures implemented at selected SI.

/v/com.provider/render/x-27.592y-48.424h3mα35°ρ45°r10m

Ex. 4. A service name with all parameters in the same component.

5.2.4 SD-MSCN Protocol Workflow

The workflow of SD-MSCN is shown in Figure 5.2, where a UE is a first user that starts consuming a service, and later a joining UE will also consume the same service after an initial setup was already performed for the previous user. For every user that starts consuming the service, the SDN controller is responsible for installing communication flows in the network issuing Flow Modification (FlowMod) control messages [182] to the switches in the network. Figure 5.2-I shows the steps to allow the first request to arrive from UE to the service. UE starts by creating a local hash of the service name to then send the request message. This hash is created using Equation 5.4, similar to the example given in Equation 5.5. As this is the first request to the service, there is no path installed in the network. Thus a Packet-In control message [182] is created by the first switch that does not know what to do with the request. This message is forwarded to the controller that will inspect the message and install the communication flows required for the communication between UE and service. Also, after issuing FlowMods to update the network paths, the controller will generate a Packet-Out control message [182] to deliver the request to its destination. A similar process in the opposite direction is shown in Figure 5.2-II, which allows the response from that request to be delivered at the UE. Finally, in Figure 5.2-III, all communication paths are installed, and the communication can happen directly between UE and service.

Figure 5.2-IV shows the process of an UE starting to consume a service after the initial setup – shown in Figure 5.2-I to Figure 5.2-III – was done by another UE. In this case, if the joining UE attaches to one of the switches that already know where to send the request for that service, this request is directly forwarded to the service. However, when installing the initial response flow in Figure 5.2-II, we also installed a more generic flow rule that, besides forwarding the response back to the joining UE, creates a Packet-In message to notify the controller about other UEs consuming that service. This information allows the controller to build a database of services and users that consume them. When the controller receives this message, it also issues a FlowMod message to install a new specific response flow to prevent future notifications, similar to this, from being sent to

the controller by the same UE. After the controller was notified and the specific response flow was installed to the joining UE, the communication normally happens between UE and service as shown in Figure 5.2-V.

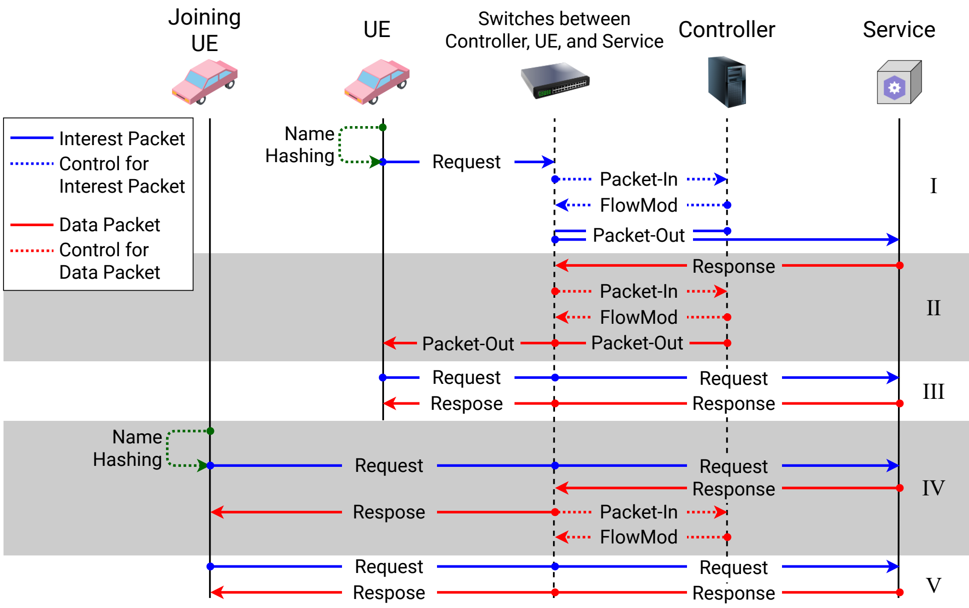

Figure 5.2: Operation of SD-MSCN [205] © 2022 IEEE.

The notification that the controller receives about joining users is important since it allows the controller to take proactive actions when handovers happen. For instance, since now the controller knows users and the services they consume, eventual paths that need to be installed after the handover to keep the users connected with the services can be done proactively. This approach helps to handle user mobility events, however, service mobility events also create issues in ICN architectures. This happens because, in general, the information about path changes takes time to be propagated in the network among the switches. SD-MSCN has an advantage in this scenario. Since it uses SDN, there is no need to wait for routing information to propagate in the network. It can be actively installed in the switches by the controller. Network updates due to service mobility events can also be handled proactively since the controller has a global view of the network and knows when services will be moved to different hosts. One advantage of using ICN instead of IP is the possibility of creating the service-users database only by inspecting the Openflow field of the message that stores the service hash. Implementing a similar mechanism in IP networks would demand a dedicated service to acquire this information and build this database, sometimes also requiring direct interaction from users and service providers.

5.3 Performance Evaluation

In the present section, we discuss the performance evaluation results comparing SD-MSCN with other ICN and IP approaches. We follow the general evaluation methodology detailed in Section 4.3 and focus the evaluation around two variables, (i) end-to-end latency to consume services, measured from the moment a request was created at an UE to the moment when a successful response was received from that request, and (ii) system cost, measured as the number of control messages sent in the network. The specific details of the methodology used for the experiments in this chapter not covered in Section 4.3 are given in Section 5.3.1. Section 5.3.2 shows discusses the results obtained in the presence of only user mobility, and Section 5.3.3 discusses the results when both users and services experienced mobility events.

5.3.1 Performance Evaluation Methods

In this chapter, four protocols were evaluated for comparison:

(a) An IP over SDN protocol, referred to as Standard IP supported by SDN (SD-IP), in which all addressing is done using IP addresses, and DNS is used for service name resolution.

(b) A Proactive IP over SDN protocol, referred to as Proactive SD-IP that differs from (a) in only two aspects: (i) data paths are proactively updated on handovers, and (ii) when a service instance changes hosts, DNS invalidation messages are sent to UEs so they can proactively request a new resolution of the service name. Since IP architectures do not keep a service-users database, similar to our proposal, the broadcast transmission was used to disseminate these invalidation messages.

(c) A protocol based on Software-Defined Information-Centric Networking (SD-ICN) [263] that was developed similar to SD-ICN [263] that uses flat hashing over service names and then replaces these hashes in traditional Openflow matching fields. In the original study, authors propose the usage of both source and destination fields to store these hashes. However, this approach would require specific features in the switches to emulate PIT behavior from ICN. Therefore, we used only the destination field to store the hash, and the source field was used to store information about the source of the request to allow the controller to create the response communication flows in the network.

(d) Our proposal referred as SD-MSCN as described in Section 5.2.

The topology used in the experiments is shown in Figure 5.3. The red line is the path used by the vehicles that moved with an average speed of 50 km/h. The circles represent Base Stations (BSs) that work as APs and also have the computing power to run services, while the dotted lines represent the wired optical fiber backhaul links between them. The blue circle is the host of the controller, and the red circle is the initial host of the services that were consumed by UEs. The topology was created in a way to create multiple handovers for the UEs in order to evaluate the behavior of the proposal. Also, the controller was placed a few hops away from the core of the topology in order to highlight the negative impact of protocols that send too many requests to the controller, as this is an undesired behavior.

The experiments were divided into two groups, using static and dynamic services. For the static services, the services were running in the red circle node throughout the

Figure 5.3: Top view of the topology used for simulations [205] © 2022 IEEE.

entire simulation, while for the dynamic services, they would start running there, but would change hosts randomly. This random service mobility pattern can be justified by the various reasons that services may need to be moved to the edge of the network. For instance, services may move for load balancing, energy saving, improved resource allocation, or according to user mobility patterns. During the experiments, three services were available with request rates of 1 per 1 s, 5 s, and 10 s. The adoption rates of these services were 100%, 10%, and 50% respectively, and each of them would take 1-2 ms to process the request before issuing a response.

5.3.2 Static Services

In the present section, we compare the four approaches aforementioned in the presence of user mobility. Thirty-three simulation runs were performed for each one of the approaches in order to record the values displayed in this section. The first variable studied is the average end-to-end latency observed by UEs, shown in Figure 5.4. On the horizontal axis, four types of requests are shown: (i) the first request executed by each UE, (ii) the first requests performed after handovers, (iii) the remainder of the requests not in the two first classes, and (iv) the combination of all three classes. On the vertical axis, the average latency is shown in milliseconds. Since IP-based solutions rely on DNS resolution to identify the host address to which they send the request, they achieve poor performance for the first request of each UE. This is due to the time consumed with DNS resolution during the process of sending this first request instead of the local hash operation that is performed by the ICN-based approaches. Furthermore, we observe that Proactive SD-IP and SD-MSCN perform better on the first requests after handovers. This happens since network paths are installed proactively during the handover, facilitating the re-establishment of communication after the handover. Despite the better performance in these two classes, the Overall class shows a similar behavior between both IP-based approaches and our proposal. This is a consequence of the small frequency in which events on the two first classes happen compared to the total number of requests. Just SD-ICN has a significantly worse performance overall, which is caused mainly by the flat hashing strategy adopted. Using this strategy, most of the requests have to be sent to the controller to get routed, which leads to an increase in the end-to-end latency as messages are sent through a sub-optimal path most of the time.

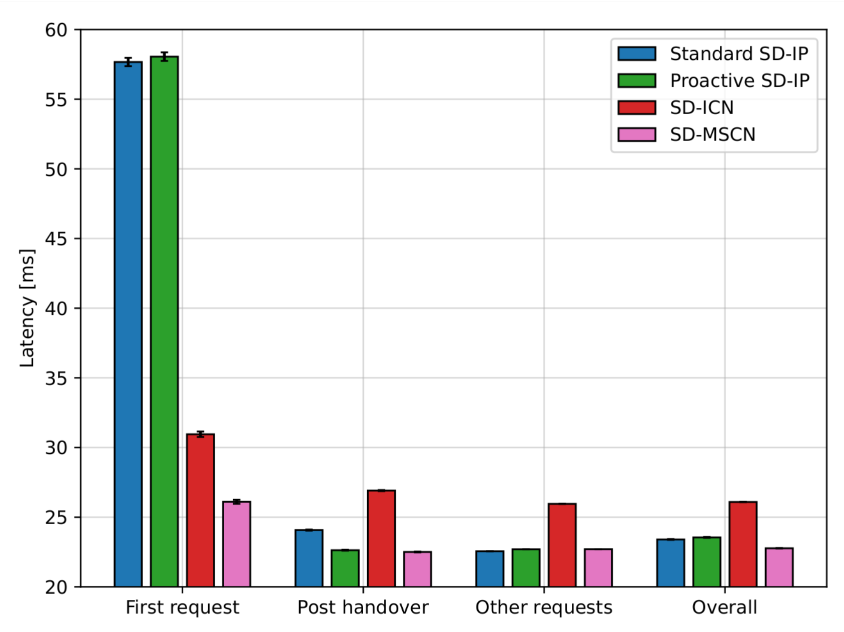

Figure 5.4: Simulation results for round-trip latencies with 99% confidence interval [205] © 2022 IEEE.

To better understand the differences between the latencies observed by using each of the approaches evaluated, Figure 5.5 shows the Cumulative Distribution Function (CDF) of latencies. Figure 5.5-a shows the distribution for all requests in the simulations, while Figure 5.5-b is a zoomed-in version of Figure 5.5-a in a specific region. The vertical axis shows the percentage of requests with latency under the value displayed on the horizontal axis in milliseconds. As shown in Figure 5.5-a Standard SD-IP, Proactive SD-IP, and SD-MSCN have a very similar behavior until the mark of 90% of the requests. Just SD-ICN has a worse performance, because most of the requests sent are routed using the controller. When requests have to use the controller to be routed, more time spent to deliver the request. This extra time spent leads to the right shift of the SD-ICN line in Figure 5.5-a, pushing this line further away from 20 ms mark when compared to the other approaches. After the 90% mark, the behaviors of the three other approaches start to be different, which can be better observed in Figure 5.5-b. The differences between IP-based and ICN-based approaches are represented by the area between their respective curves in Figure 5.5-b. This difference is created in situations where ICN-based approaches have advantages, such as the initial request and some requests after handovers. Also, it is possible to observe that 99% of the round trip latencies for successful requests are under 40 ms for ICN-based protocols, while it takes up to around 55 ms for the IP-based protocols to deliver the same percentage of the requests.

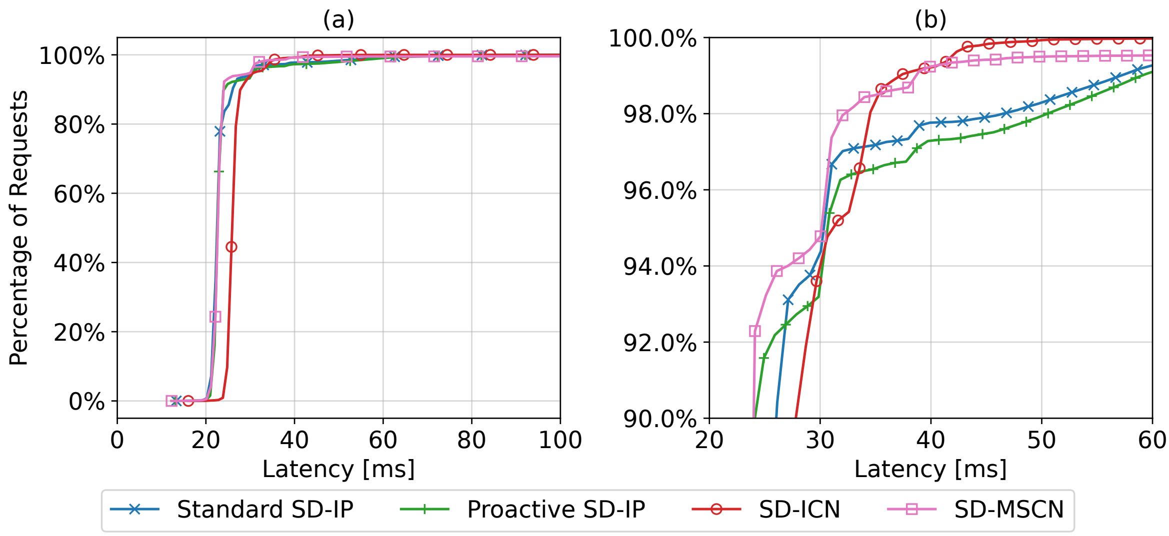

Figure 5.5: Cumulative distribution of round-trip latencies [205] © 2022 IEEE.

Figure 5.6 compares the costs of the different approaches evaluated. This cost is measured as the number of control messages sent during the simulations. The different approaches are listed on the horizontal axis, while the number of control messages sent is shown on the vertical axis. Four types of control messages are highlighted: (i) FlowMod, messages used to install communication flows in the network; (ii) Handover, messages related to the execution of handovers; (iii) DNS, messages for service name resolution; and (iv) Packet-In and Packet-Out, messages sent from switches to controller and from controller to switches. SD-ICN has the highest cost in FlowMod and Packet-In and Packet-Out messages. Again this behavior is due to the flat hashing that causes many messages to be sent to the controller for routing and consequent installation of communication flows. Our approach of using hierarchical hashing allows more re-usage of already installed communication flows, which causes the number of FlowMod and Packet-In and Packet-Out messages to be similar to the Proactive SD-IP approach. Figure 5.6 also shows that there is no cost with DNS for the ICN-based protocols, which is expected since these approaches do not need name resolution. Finally, all four approaches have similar behavior regarding the number of control messages used during handover.

5.3.3 Dynamic Services

In this section, we discuss the results obtained in the same conditions as in Section 5.3.2 but now including service mobility events. To evaluate these service mobility events, service instances were randomly assigned to migrate from one BS to another at given frequencies. Such service mobility events can be used to achieve different goals, such as load balancing, improving resource allocation, reducing communication latency, or saving power consumption with resources. We only evaluate what happens when service mobility events are triggered in the network without accounting for the cause of these events. The frequencies of mobility events were every 60 s, 30 s, 20 s, 10 s, and 5 s. These frequencies are compatible with other studies in the literature. For instance, Aissioui et.al. [14] used

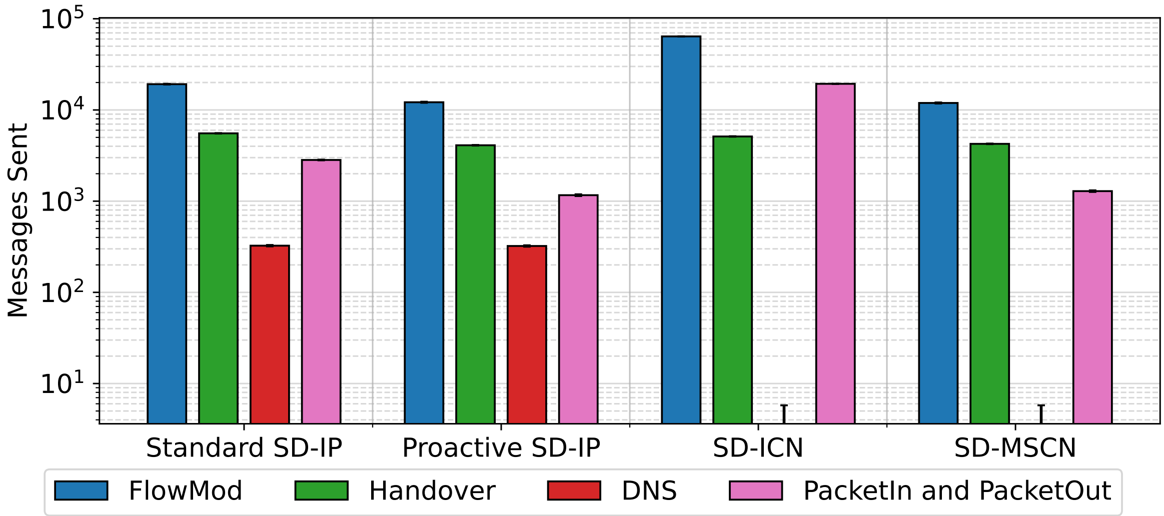

Figure 5.6: Simulation results for number of signaling messages sent with 99% confidence interval and using logarithmic scale on Y-axis [205] © 2022 IEEE.

the frequencies of 1 service mobility event every 25 s, 16 s, and 8 s when evaluating scenarios where only service mobility events induced by user mobility were accounted for. When more sources of service mobility events are included, we expect to achieve slightly higher frequencies, such as the ones we use for our experiments. In total, 165 simulations were executed for each approach, 33 for every data point observed in Figures 5.7 and 5.8.

Average values of end-to-end latency observed after service mobility events are shown in Figure 5.7. The horizontal axis shows the frequency with which service mobility events were triggered, while the vertical axis shows this average latency. Regarding latency, the highest values were observed when using Standard SD-IP, which is caused by its reactive nature. In this case, an UE only notices that the service moved after receiving an error message as a response to the original request. After this, UE needs to perform a name resolution to identify the new host address of the service, and just then, it can re-establish a connection with the service. The proactive SD-IP had good performance for lower frequencies since the DNS invalidation messages were used to proactively run name resolution and proactively identify future addresses of services. However, since these warnings were disseminated using broadcast, the latency increases significantly faster than other approaches, for higher frequencies of service mobility events. This fast increase in latency highlights the scalability issues of Proactive SD-IP, which is a result of the congestion in the network and the controller making it difficult for messages to be delivered. Both ICN-based approaches do not suffer from this scalability issue and keep good performance for higher service mobility events frequencies.

The control cost was also evaluated for different service mobility events frequency. Figure 5.8 shows on the vertical axis the number of control messages sent for every frequency displayed on the horizontal axis. It is important to mention that DNS invalidation messages were also computed with the other control messages as cost in this section – no service mobility was considered in Section 5.3.2; thus, there was no DNS invalidation messages. SD-ICN obtained the highest cost among the approaches evaluated. Again

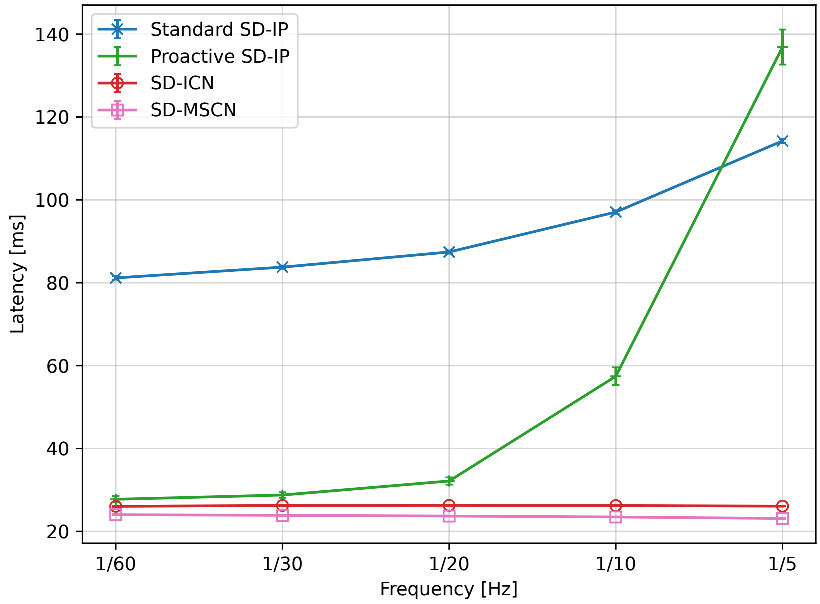

Figure 5.7: Simulation results for latency after service mobility events with 99% confidence interval [205] © 2022 IEEE.

this is the expected behavior due to the flat nature of the hashing used in this approach. SD-ICN sent more control messages when compared to Proactive SD-IP. Still, no congestion was caused in the network since these messages are sent more distributedly over time, differently from the invalidation messages in Proactive SD-IP. Multiple messages sent at the same moment are responsible for the congestion that lead to the increase in latency for higher frequencies. In general, SD-MSCN obtained the best performance, sending fewer control messages than the other approaches in all scenarios.

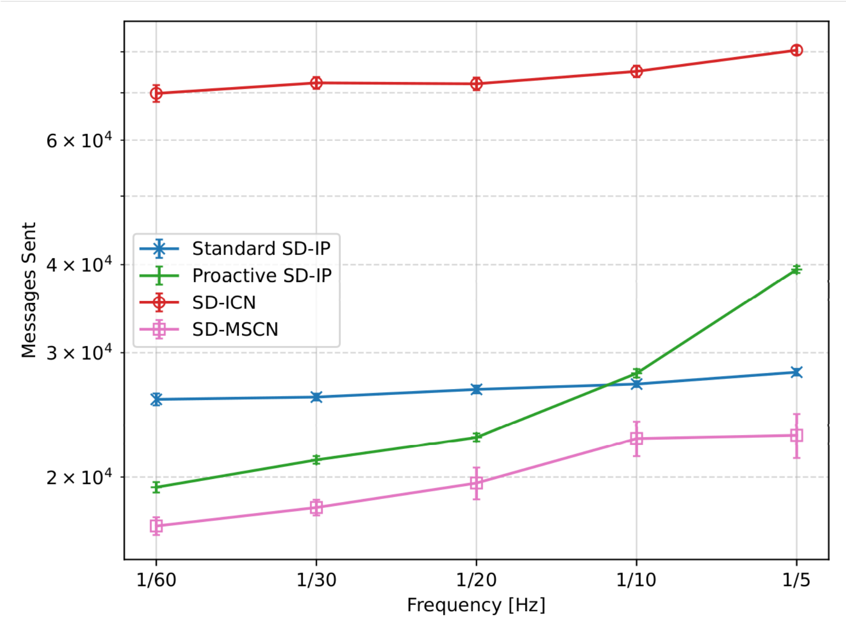

Figure 5.8: Simulation results for the number of signaling messages sent with service mobility with 99% confidence interval and using a logarithmic scale on Y-axis [205] © 2022 IEEE.

5.4 Chapter Conclusions

This chapter discusses the concept of MSCN and presents an implementation based on SDN called SD-MSCN. We explore the possibility of latency reduction by using SD-MSCN in the presence of user and service mobility events at the edge of the network. We also study the trade-off of this approach in terms of the number of signaling messages needed for the proper functioning of the proposed protocol. Simulation results show that our proposal outperforms Standard SD-IP, Proactive SD-IP, and SD-ICN approaches in scenarios such as the first service request and requests after service mobility events. Also, it keeps a similar behavior in other scenarios. Furthermore, the signaling cost of our proposal is only greater than the Standard SD-IP approach in the presence of frequent service mobility events, having a smaller cost than Proactive SD-IP and SD-ICN.

We used Long Term Evolution (LTE) models to simulate the wireless channels. Still, similar gains are expected when using more recent wireless communication technology, such as the 5G NR. Overall, communication behavior in long-lasting communication flows is determined by the messages sent when no mobility events happen since these represent the great majority of all messages. Although the proposed SD-MSCN can outperform the Standard SD-IP for post-handover requests, it is possible to obtain similar behavior by implementing proactive strategies still using IP-based protocols. Nevertheless, significant gains in our proposal were observed when service mobility events were introduced in the experiments. In these experiments, even the proactive strategy based on IP was outperformed by SD-MSCN for higher frequencies of service mobility events. This behavior happens because this proactive strategy relies on broadcasting DNS invalidation messages, which leads to scalability issues.

The differences of latency in a small number of messages can be important, especially in the vehicular use case where safety levels may decrease with more significant delays to receive safety-related information [133]. Besides the gains in latency, other gains enabled by our approach are outside of the scope of this study. For instance, mobility management is simpler when using our approach since: (i) user mobility events do not require UE addressing updates; and (ii) service mobility events do not require updates on the UE. Also, using hashes of service names as addresses removes the necessity of DNS services, simplifying network management even more. Finally, our approach enables the concurrent consumption of services from multiple hosts, as services are routed via names and not topological addresses. This possibility could further contribute to reducing latency, which we can lead to interesting future studies.

After studying the SCN-based addressing strategy to mitigate disruption caused by user and service mobility events, this thesis proceeds to discuss a application state positioning strategy. The time taken to consume application state stored in different hosts at the edge together with the user mobility may impact the final latency to consume a service. Thus, in Chapter 6, a graph-based algorithm for state data placement is discussed.

Chapter 6

Distributed Application State and User Session Management

Research Question 3: *How to manage application state in mobile latency-constrained scenarios?*

In order to satisfy Future Internet application demands, such as low latency levels, Edge Computing (EC) architectures are awaited to bring computing power to process tasks closer to final users. However, at the edge of the network, user and service mobility events raise issues for service provisioning. These events disrupt the communication between users and services when triggered. In this chapter, we discuss a graph-based algorithm [206][1] designed in the context of the Distributed Application State and User Session Management (O.3) objective of this thesis. This algorithm mitigates these issues by considering latency constraints and user and service mobility when positioning key datasets for services in a set of computing-enabled network nodes. This algorithm is used to assemble a mobility-aware latency-constrained solution to position user-specific service instances, i.e., service instances that use application state and user session data. Our proposal is compared with other baseline solutions. Simulation results show that our proposal delivers a similar or larger number of packets under the latency requirements of different vehicular applications. Furthermore, by analyzing historical mobility data, our solution reduces the number of service migration events by 21%, which leads to a decrease in service interruption time of 41% when compared to the baseline solutions.

6.1 Overview

One challenge of orchestrating services at the edge is managing the placement of state data. Literature related to service migration generally focuses on protocols and strategies to move this state in a reduced time window. This movement is triggered reactively, after user mobility events (e.g., handovers), or proactively, by using mobility prediction. In the present chapter, we discuss a distributed application state management strategy. We aim to use graph theory to solve the problem of replicating and distributing state data accounting for user mobility and constrained by latency.

Figure 6.1 exemplifies the distributed state management problem scenario. In this scenario, there are cubes representing logic services and cylinders representing state services. Isolating state in specific services is often applied in many application architectures at the Cloud to achieve shared state for services or to improve scalability by separating read and write storage. However, at the edge, the problem is more complex since access to data is conditioned to network topology. By storing the data distributedly, we aim to reduce state data migration by granting that the node where the logic service runs can always access state data with a given latency constraint.

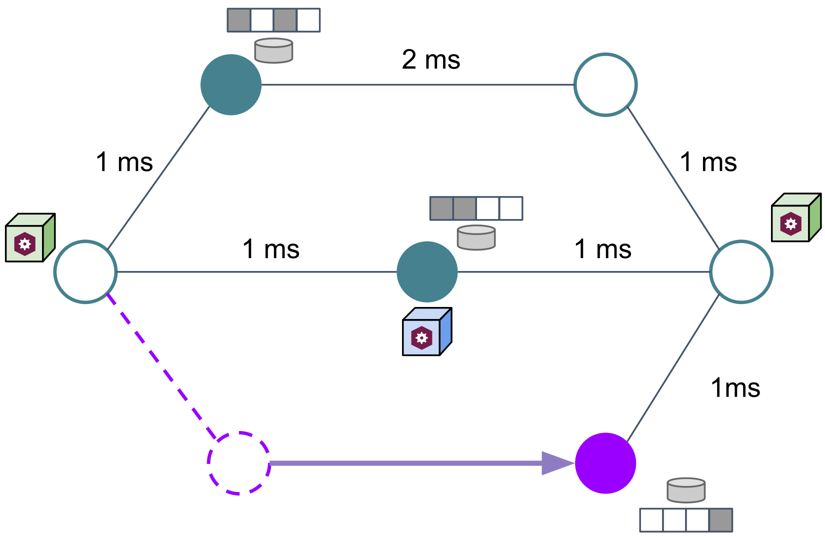

Figure 6.1: Distributed application state management considering latency and user mobility.

Distributed state management is the most challenging of the proposed objectives in this thesis. We aim to approach the data distribution problem using an extension of Data Graphs [88, 7], a subset of the graph labeling problem [79]. In our scenario, application state data is distributed in the vertices representing edge nodes, and the links are used to model latency. Furthermore, we will also introduce user mobility concepts when resolving the data distribution problem. One important remark is that stateless components of the services, represented by the cubes in Figure 6.1, are easily replicable, as they can be just copied. Therefore, during our experiments, we consider that these services are deployed in all edge nodes. This is a fair assumption, since there is no synchronization cost for these copies, and these copies can be kept in an idle state without processing costs. Therefore, we only have to account for the positioning of the Application State and User Session Data (ASUSD).

ASUSD usually is important in the execution of complex analyses, such as using Machine Learning algorithms in real-time data streams, which have gained attention, especially at the edge of the network with emerging classes of applications, such as connected vehicles and augmented reality. Traditionally, ASUSD is stored in web applications using different approaches, such as in the main memory of the host for fast or frequent access, e.g., user roles and access levels, or in databases, e.g., all data generated by users inter-acting with online systems. In the vehicular scenario, an example of application state may be the dataset composed of positions and speeds of a set of vehicles in a certain time window. This data may need to be fast accessed by an algorithm in order to perform crucial coordinated maneuvering with multiple vehicles, for instance, to avoid a car crash or reduce the damage of an inevitable one. Other domains also make use of ASUSD. For instance, immersive games may use this data to control different in-game mechanics, such as how characters interact with each other and their virtual environment.

6.2 Problem Definition and Formulation

In this chapter, we discuss an algorithm to identify the placement of stateful data in computing-enabled network nodes at the edge. When computing this data placement, we account for the latency requirements of applications and also for user and service mobility events. For this purpose, we assume an edge network composed of Base Stations (BSs) that works as Access Point (AP) for User Equipments (UEs) and also has computing and storage capacity to provide services to these UEs. UEs connect to this network using cellular technology. Also, this edge network is managed using Software-Defined Networking (SDN) to allow fast response to changes in the topology. We assume the services dealt within this chapter have strict Service Level Agreementss (SLAs) to work properly. Thus computing and storage resources, as well as energy consumption, are not limiting factors. The network providers must grant the requirements to make these applications run. This assumption is reasonable for a series of safety-related applications and also entertainment services with high Quality of Experience (QoE) demand. This scenario allows us to propose a data-placement algorithm that does not account for resource allocation. Still, there are plenty of studies in the literature that focus on managing network resources at the edge, which also include service migration events when needed [26, 186, 156].

We model our scenario as a data graph $\mathcal{G} = (V, E, W, D, \phi)$, an undirected graph with vertices $v \in V$ and edges $E \subseteq V \times V$. Each edge $e \in E$ has a weight $w_e \in W$. Also, D is a set of data items that is mapped to the vertices according to the mapping ϕ. ϕ is defined as $\phi : V \to \mathcal{P}(D)$, $v \mapsto D'$, where $\mathcal{P}(D)$ is the power set of D, thus $D' \subseteq D$. This means that ϕ can take any of the vertices in V as an argument and, for each of them, output any possible set generated by combining the data items in D. As any combination of the data items is allowed, it is also possible to have data duplication, i.e., the same data item mapped to more than one vertex.

We map the elements of the data graph to real-world entities in Figure 6.2. Each graph vertex represents a network node with computing and storage capacities. For simplification, we consider that these nodes also work as APs for UEs. Still, the proposed approach is capable of handling scenarios where some of the nodes are not APs, only micro or nano data centers at the Edge, and some are not eligible for running computing tasks because they lack installed computing resources at the BS. The vertices in Figure 6.2 are labeled with numbers, i.e., $V = \{1, 2, 3, 4, 5, 6, 7\}$. The links connecting vertices in the graph represent the wired connections between the BSs, and their weights are the expected latency on that link. For instance, $w_{4,5} = 0.3$ ms shows the expected latency between

Table 6.1: ϕ outputs for different v in Figure 6.2 [206] © 2023 IEEE.

v	1	2	3	others
ϕ	$\{b,c,d\}$	$\{a,b\}$	$\{e,f\}$	$\{\,\}$

BS 4 and 5. These expected latency values must be estimated according to the link technology and also historical data possessed by network providers. In our experiments, we used values for an optical network based mobile backhaul from the literature [142]. Finally, the mapping ϕ informs where data is stored in the network topology. It maps data chunks in the set D, represented by their labels, the elements of D, to nodes in the network topology where these data chunks are stored.

For the example in Figure 6.2, the dataset $D = \{a,b,c,d,e,f\}$ is the target of the data placement. Each data item in this set can be defined in multiple ways and have any size. For instance, each item may be a label for a data chunk. The set D is distributed in the infrastructure according to the mapping $\phi(v)$. Table 6.1 shows which data chunks are mapped, by ϕ, to be stored in which BS. For instance, the data mapped to be stored at the vertex $v = 1$ is $\phi(1) = \{b,c,d\}$, which means that the data chunks represented by b, c, and d will be stored at the BS labeled as 1.

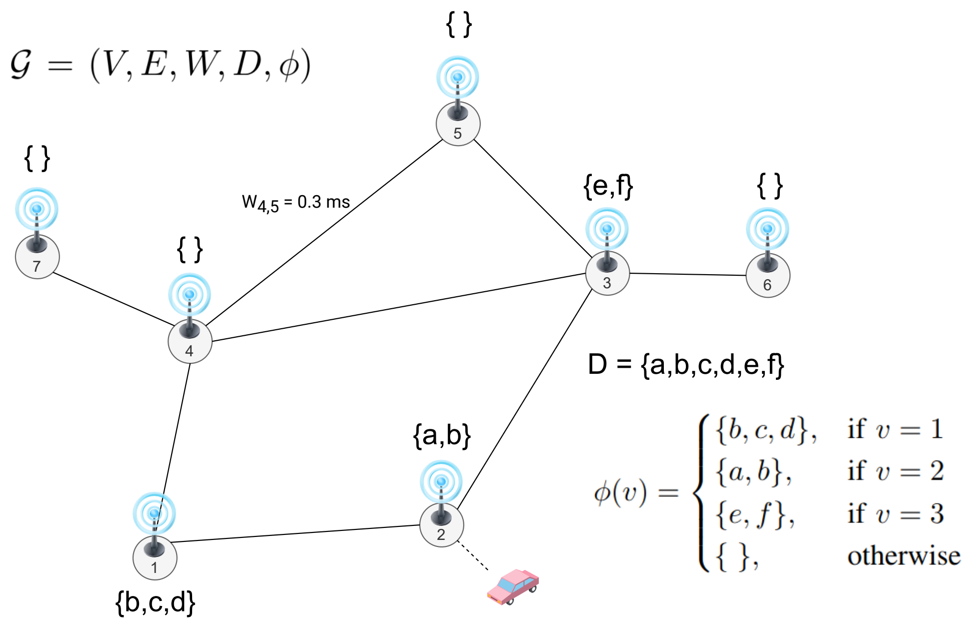

Figure 6.2: Example Data Graph and respective real-world counterparts [206] © 2023 IEEE.

Let $V_r \subseteq V$ be the set of reachable vertices of v in $\mathcal{G}$, v is data covered, denoted as $v \rightsquigarrow D$, when the complete set D is mapped to the vertices in V_r. This condition can be expressed as

$$\bigcup_{v_r \in V_r} \phi(v_r) = D. \tag{6.1}$$

Similarly, a set of vertices V_s is data covered, denoted by $V_s \rightsquigarrow D$, if every $v_s \in V_s$ meets the aforementioned criteria, in which case every v_s will have a separate set of reachable vertices V_r^s that must satisfy Equation 6.1. Note that a data graph only allows for one mapping to be selected. Thus, when dealing with Data Covers (DCs) of sets of vertices, a single mapping ϕ must provide the DC for V_s. All vertices in Figure 6.2 are data covered because all vertices can reach each other, and the complete set D is mapped in the graph. Finally, we define a target set V_t to be formed by all vertices v_t where $\phi(v_t) \neq \emptyset$, V_t is a notable set important when studying DCs. For instance, in Figure 6.2, $V_t = \{1, 2, 3\}$.

To study data placement at the edge of the network, we define three concepts to work with the data graphs: (i) Minimal Data Cover (MDC); (ii) δ-Mappings; and (iii) Budgets.

Minimal Data Cover. A MDC for a set V_s is a mapping that leads to the minimum possible data duplication in which V_s is still data covered, i.e., $V_s \rightsquigarrow D$. In fully connected graphs, the placement of a single copy of the dataset is enough to cover all vertices. However, this is not the case when the graph has disconnected components, which would require more copies of the dataset to be stored to cover individual components of the graph. The idea of reducing data duplication becomes more relevant after the introduction of the budgets to consume the data. When budgets are introduced, even fully connected graphs are partitioned in different unconnected components because of this budget. We denote a mapping ϕ that is a MDC for a set V_s as $\phi\big|_{V_s}$. It is possible to find multiple MDCs for the same set V_s in a data graph $\mathcal{G}$. A MDC is an optimal solution to the problem of minimizing data duplication while still covering all vertices in V_s. In Figure 6.2, $b \in \phi(1)$ and $b \in \phi(2)$, in which case data duplication could be avoided and therefore ϕ is not a MDC.

δ-Mappings. We define δ_v as a mapping where all set D is mapped to the vertex v, and $\emptyset$ is mapped to all other vertices. This notation also allows expressing ϕ as an addition of δ-mappings resulting in a mapping ϕ in which the set D is entirely mapped to all $v_t \in V_t$ and $\emptyset$ mapped to all other vertices, as

$$\phi = \sum_{v_t \in V_t} \delta_{v_t}. \tag{6.2}$$

One important property of the data graphs is that it is possible to produce a MDC composed only by δ-Mappings. This property is proven as the Winners Take All theorem.

Theorem 1. *Winners Take All: For any $V_s \subseteq V$ given, there is at least one mapping $\phi = \sum_{\forall v_t \in V_t} \delta_{v_t}$ that is a minimal data cover for V_s.*

Proof. We will prove by induction that the theorem Winners Take All holds for any set V_s.

For the base case, when $V_s = \{v_s\}$, to find an MDC for V_s, we need only that $v_s \rightsquigarrow D$. To find a DC for v_s, it only makes sense to map data in vertices in the same connected component of v_s in $\mathcal{G}$, as data mapped in other components would never be reachable by v_s. Further, choosing $\phi = \delta_{v_t}$ for any v_t where $v_s \rightsquigarrow v_t$ implies $v_s \rightsquigarrow D$ as all set D is reachable by v_s. $\phi = \delta_{v_t}$ is an MDC as no data duplication is needed. Thus it is possible to find an MDC for the base case where $V_s = \{v_s\}$.

For the induction step, let V_s be given and suppose that there is a mapping

$$\phi\big|_{V_s} = \sum_{v_t \in V_t} \delta_{v_t}, \tag{6.3}$$

which is an MDC for V_s. We can now expand V_s by adding another vertex v'_s not previously present in V_s. v'_s may either be present in (i) one of the connected components in which $\exists v : v \in V_s$; or be present in (ii) another isolated connected component in $\mathcal{G}$ in which $\forall v : v \notin V_s$.

For (i), as v_s and v'_s are in the same connected component, there is a path $v'_s \rightsquigarrow v_s$. Since v_s was data covered by one δ_{v_t}, there is $v_s \rightsquigarrow v_t$. Thus, there is also $v'_s \rightsquigarrow v_t$, and consequently $v'_s \rightsquigarrow D$. Therefore, the theorem holds when v'_s is added to V_s.

For (ii), as v'_s is unreachable from any v_s, v'_s also cannot reach any $v_t \in V_t$ and thus Equation 6.2 is not a DC to v'_s. Still, we can find

$$\phi'\big|_{V_s \cup \{v'_s\}} = \phi\big|_{V_s} + \delta_{v'_t}, \tag{6.4}$$

such that $v'_s \rightsquigarrow v'_t$ just by selecting v'_t in the same connected component as v'_s. $V_s \cup \{v'_s\}$ is data covered by ϕ', as $v \rightsquigarrow D$ for all $v \in V_s \cup \{v'_s\}$. Further, ϕ' is an MDC since if we remove any $d \in D$ from any of the mappings δ_{v_t} or from $\delta_{v'_t}$, either a $v_s \in V_s$, or v'_s will no longer be data covered. Therefore the theorem also holds when v'_s is added to V_s.

$\square$

Budget. A budget in the distributed state management problem translates in the real world to a threshold value of communication latency within which the data should be consumed. We say a vertex v is data covered within a budget b if $v \rightsquigarrow D$ and for every $v_t \in V_t$ there is at least one path $v \rightsquigarrow v_t$ within the budget, i.e., the sum of all weights in one of the paths $v \rightsquigarrow v_t$ is such that

$$\sum_{e \in v \rightsquigarrow v_t} w_e \leq b. \tag{6.5}$$

We denote by $v \overset{b}{\rightsquigarrow} D$ the case where the vertex v is data covered within a budget b, and $V_s \overset{b}{\rightsquigarrow} D$ when all vertices $v_s \in V_s$ are data covered within a budget b. Notice that budgets apply to individual paths from the source node v to nodes v_t. This modeling strategy accounts for the possibility of reading data in parallel from multiple sources.

It is also possible to consider an MDC within a budget b, denoted as $\phi\big|_{V_s}^{b}$. In the real world, this budget represents the desired latency allowed to access the complete data set. When evaluating this access latency, it is necessary only that the individual paths connecting the source node v_s (or set V_s) are within the budget and not the sum of all paths used to produce the DC. This is due to the possibility of accessing data in parallel.

When defining data access latency only in terms of communication latency, it is possible to compute MDCs in which data is fully mapped using δ-mappings even when considering budgets. This property of the data graphs is interesting because it leads to less complex solutions, i.e., simpler DCs that are easier to manage in the real world. Thus, we proceed to prove this feature of the DCs, which is later used in Section 6.3 to compose our data placement framework.

Theorem 2. *Winners Take All Within a Budget: For any $V_s \subseteq V$ and budget b given, there is at least one mapping $\phi = \sum_{\forall v_t \in V_t} \delta_{v_t}$ that is an MDC for V_s within the budget b.*

Proof. To prove the theorem, we propose an algorithm to identify the vertices $v_t \in V_t$ that lead to the mapping as presented in Equation 6.2 that is a MDC for a set V_s. Then we will proceed to prove that (i) the algorithm can find a DC within a budget, and further, (ii) this DC is minimal. The pseudo-code of this algorithm is shown and discussed in Section 6.3.2 as a part of our proposed data placement framework.

To find a set V_t to compose a mapping as stated in Equation 6.2, we first identify for every vertex $v_s \in V_s$, the respective set of reachable vertices V_r^s within the budget, composing a set $\mathcal{V}_r$. This can be done, for instance, by using a minimum spanning tree algorithm [190] for each vertex v_s and then cutting the branches of the resulting trees when the sum of weights in each path from v_s goes over b. Now, for every vertex v_s there is a set V_r^s such that $\forall v_r^s \in V_r^s \rightarrow v_s \overset{b}{\rightsquigarrow} v_r^s$. We now must select the vertices to map to D and produce the minimum data duplication.

To select the vertices to map D, we first choose the biggest set of reachable vertices V_r^s that share at least one vertex. We represent this selection by their indexes $s \in I_t$. I_t is such that (i) $\forall s_i, s_j \in I_t \rightarrow V_r^{s_i} \cap V_r^{s_j} \neq \emptyset$ and that (ii) it is not possible to find a set I_t with more elements. Let $C_t = \bigcap_{\forall s \in I_t} V_r^s$ be the set of nodes shared by the reachable sets in the I_t selection. We take one of these shared vertices as v_t and mark all sets V_r^s for $s \in I_t$ as data covered. We repeat this process with the remainder unmarked sets $V_r^s \in \mathcal{V}_r$ until all of them are marked as data covered, producing a set of vertices $v_t \in V_t$. This set is our goal, and the composition of the mapping of its vertices v_t as δ-mappings, i.e., as stated in Equation 6.2, results in an MDC for the set V_s.

For (i), it is trivial to prove that the result of the proposed algorithm is a DC because all vertices in V_t were selected from sets of vertices $v_t \in C_t$ in which $v_t \overset{b}{\rightsquigarrow} v_s$ for all $v_s \in V_s$. As paths in the graph are undirected, it is also true that for all $v_s \in V_s$, $\exists v_t \rightarrow v_s \overset{b}{\rightsquigarrow} v_t$; furthermore, $\forall v_t \rightarrow v_t \rightsquigarrow D$ with no additional cost, as the entire set D is mapped to v_t. Therefore, $V_s \overset{b}{\rightsquigarrow} D$.

For (ii), to prove that the result of the proposed algorithm is an MDC, we will imagine that there is a vertex $v_t' \notin V_t$ that can replace at least two vertices v_{t_i} and v_{t_j}, both present in V_t, to produce a DC with less data duplication. If v_t' can replace v_{t_i} and v_{t_j}, it means it provides a DC for both the vertices v_{s_i} and v_{s_j} in V_s that were previously covered only by v_{t_i} and v_{t_j}. This leads to $v_{s_i} \overset{b}{\rightsquigarrow} v_t'$ and $v_{s_j} \overset{b}{\rightsquigarrow} v_t'$, thus v_t' must belong to both reachable sets V_{r_i} and V_{r_j}. Additionally, if v_t' can replace both v_{t_i} and v_{t_j} it must also provide a DC to other eventual v_s vertices, different than v_{s_i} and v_{s_j}, that are covered by either of the vertices v_{t_i} and v_{t_j}. Therefore $v_t' \in C_{t_i}$ and $v_t' \in C_{t_j}$. Still, v_{t_i} and v_{t_j} were selected in two different steps of the algorithm, and thus $C_{t_i} \cap C_{t_j} = \emptyset$. As it is not possible that $v_t' \in \emptyset$, the existence of v_t' is contradictory. Therefore, there cannot be a vertex v_t' that can replace two or more vertices $v_t \in V_t$ to reduce data duplication. □

6.3 Distributed State Data Management Framework

We proposed a three-component solution for data placement at the edge of the network. In the first component, described in Section 6.3.1, we compute Action Zones (AZs), which are the inputs V_s for the DC identification algorithm. The second component uses the algorithm for identification of DCs, described in Section 6.3.2, to evaluate and decide whether DCs are still valid or need to be updated. Finally, the third component described in Section 6.3.3 details the process of data migration to take place when DCs become obsolete and have to be updated. Finally, Section 6.3.4 describes the process to synchronize multiple copies of the state data that are used in our solution.

6.3.1 Action Zone Identification

AZs are sets of BSs that either is currently in use by the UE, or that will be used in the near future. To identify these BSs, historical data maintained by network providers is used. We proposed the usage of Markov Chains [202] to create a mobility graph induced from historical UE-BS attachment data. In this graph, for each BS there is a probability of connecting to another BS represented as the weight of a link connecting these two BSs in the graph. This probability is measured according to the number of times this transition happened in a given time window of the historical data.

Using this approach, we can create a directed mobility graph $D_m = (V_m, E_m, W_m)$, in which $v_i \in V_m$ is the set of BSs to which UEs can attach, and we form the set E_m by creating edges e_{ij} whenever an UE detaches from v_i and attaches to v_j. $w_{ij} \in W_m$ are the weights of the edges e_{ij} given by the probability of a UE attached to v_i to attach to v_j in the next handover performed. A directed graph is used as a model to allow, for instance, $w_{ij} \neq w_{ji}$ in D_m. The AZ $A_n(v_i, c)$ of a given BS v_i is composed by all vertices $v_j \in V_m$ within maximum distance n from v_i and that the probability of future attachment $P(v_j|v_i)$ is greater than the cutoff probability c, where the probability of future attachment to v_j given that the UE is attached to v_i is:

$$P(v_j|v_i) = \prod_{e_{ij} \in v_i \rightsquigarrow v_j} w_{ij}. \tag{6.6}$$

Historical data is used to determine the probabilities taken into account when evaluating AZs for the BSs, which allows this procedure to be executed offline and updated with arbitrary frequency according to the requirements of the network providers. As mentioned before, we consider all BSs to be able to run computing tasks and work as APs; thus, V_m in D_m is the same as V in $\mathcal{G}$ – although $E_m \neq E$ since E represents the links in the network topology. This might not be the case in some scenarios. For instance, some data-center may be connected to the edge, providing computing power but not an access point, or some BS may not have installed computing capacity. Still, our approach is general enough to handle these scenarios by adding or removing nodes in the graph, which may lead to a V_m in D_m different than V in $\mathcal{G}$.

6.3.2 Data Covers Algorithm

Based on the proof of Theorem 2, we proposed Algorithm 1 to find MDCs within budgets. This algorithm uses Minimum Spanning Tree within a Budget (MSTB), a variation of the Minimum Spanning Tree algorithm [190] that, after computing the minimum spanning tree, cuts every path when the sum of the weights from the source gets greater than the budget. A simple possibility of implementation of MSTB is given in Algorithm 2. Still, implementations that run with better performance can be used as a replacement. For instance, some implementations of the Minimum Spanning Tree algorithm run in $O(E \log V)$ [46] and can be adapted to the use case of the budgets.

Algorithm 1 Data Covers Algorithm to find MDCs within budgets [206] © 2023 IEEE.

Input: Source vertices: V_s, Budget: b

Output: Set V_t generator of an MDC $\phi\big|_{V_s}^{b} = \sum_{\forall v_t \in V_t} \delta_{v_t}$.

1: $\mathcal{V}_r \leftarrow \emptyset$
2: **for** $v_s \in V_s$ **do**
3:     $V_r^s \leftarrow \mathrm{MSTB}(v_s, b)$
4:     include V_r^s in $\mathcal{V}_r$
5: **end for**

6: $V_t \leftarrow \emptyset$
7: **while** $\mathcal{V}_r \neq \emptyset$ **do**
8:     Let C_t be any of the sets in $\mathcal{V}_r$
9:     remove C_t from $\mathcal{V}_r$
10:     **for** $V_r^s \in \mathcal{V}_r$ **do**
11:         **if** $C_t \cap V_r^s \neq \emptyset$ **then**
12:             $C_t \leftarrow C_t \cap V_r^s$
13:             remove V_r^s from $\mathcal{V}_r$
14:         **end if**
15:     **end for**

16:     Let v_t be any vertex in C_t
17:     include v_t in V_t
18: **end while**

19: **return** V_t

Algorithm 1 receives as input a set of vertices $V_s = A_n(v, c)$ of a given BS v, and a budget b. The objective is to create data covers that cover all nodes in the AZ of a BS within the budget b. The algorithm then outputs a set V_t of vertices where data must be stored to form an MDC. Lines 1-5 will select all sets of reachable vertices V_r^s within the budget b centered in the vertices $v_s \in V_s$, which is done using MSTB. Then, starting in line 6, the algorithm builds the set V_t. For that a while loop (line 7) is used to iterate over all sets in $\mathcal{V}_r$; the sets $V_r^s \in \mathcal{V}_r$ are removed inside this loop, which will stop when $\mathcal{V}_r = \emptyset$. Inside the while loop, sets C_t are built in lines 8-15. These lines aim at selecting C_t to be the set of nodes shared by the greatest number of sets V_r^s. While C_t is built, all sets V_r^s that share nodes with C_t are removed from $\mathcal{V}_r$ (line 13). Whenever a C_t is finished, one vertex $v_t \in C_t$ is selected to figure in V_t in lines 16-17. The process repeats until $\mathcal{V}_r = \emptyset$. The time complexity of Algorithm 1 depends on the implementation of the MSTB algorithm and also on the relation between the number of edges and the number of nodes in the network. Assuming that MSTB runs in $O(E \log V)$ [46], Algorithm 1 runs in $O(VE \log V)$.

Algorithm 2 MSTB: Minimum spanning tree within a budget

Input: Vertex: v_i, Budget: b, Discovered vertices: Q (default: $Q = \emptyset$)
Output: Vertices in the MST: V

$V \leftarrow \{v_i\}$
$Q \leftarrow Q \cup \{v_i\}$

for $v_j \in \texttt{adj}(v_i)$ **do**
    if $v_j \notin Q$ and $w_{e_{ij}} \leq b$ **then**
        $V \leftarrow V \cup$ MSTB(v_j, $b - w_{e_{ij}}$, Q)
    end if
end for

$Q \leftarrow Q - \{v_i\}$
return V

At line 16 of Algorithm 1 a random vertex is selected from C_t to compose V_t. This random selection does not impact building MDCs. However, it is possible to improve the selection of these vertices to reduce data movement when reallocating these DCs. Thus, to increase the re-usage of vertices in different DCs and reduce data reallocation, we computed expanded AZs $A_N(v_i, c)$ for all vertices $v_i \in D_m$ by selecting N greater than n used to compute V_s. For instance, if we originally chose $V_s = A_1(v_i, c)$ the expanded AZ for v_i could be given by $A_2(v_i, c)$. Also, we compute the respective MDCs $\phi\big|^{b}_{A_N(v_i,c)}$ for the expanded AZs. When selecting vertices, at line 16, if the intersection

$$C_t \cap \phi\big|^{b}_{A_N(v_i,c)} \tag{6.7}$$

is not empty, then the vertex v_t would be taken from this intersection rather than taken from C_t.

6.3.3 Asynchronous Data Movement

A given DC may become obsolete depending on the mobility of UEs, which means that this DC does not fully cover all possible future attachment points of the UE within the latency budget. When this happens, the DC has to be updated by moving ASUSD from a given service instance. When a DC becomes obsolete, it still covers the current AP of the UE, which leads to a bigger time window to carry out the data movement and also removing the necessity of performing this movement at the same time of the handover procedure. Furthermore, the ASUSD can be copied while the service still runs, improving service continuity levels and also allowing the procedure to be aborted if needed. We proposed the data movement strategy shown in Figure 6.3.

Figure 6.3 pictures a situation where an UE consumes ASUSD hosted at previous Base Station (p-BS) – UE may not be necessarily attached to p-BS. Once the controller takes the decision that p-BS is no longer part of the DC for that UE, the data movement procedure starts to migrate the data to target Base Station (t-BS). Then, the following

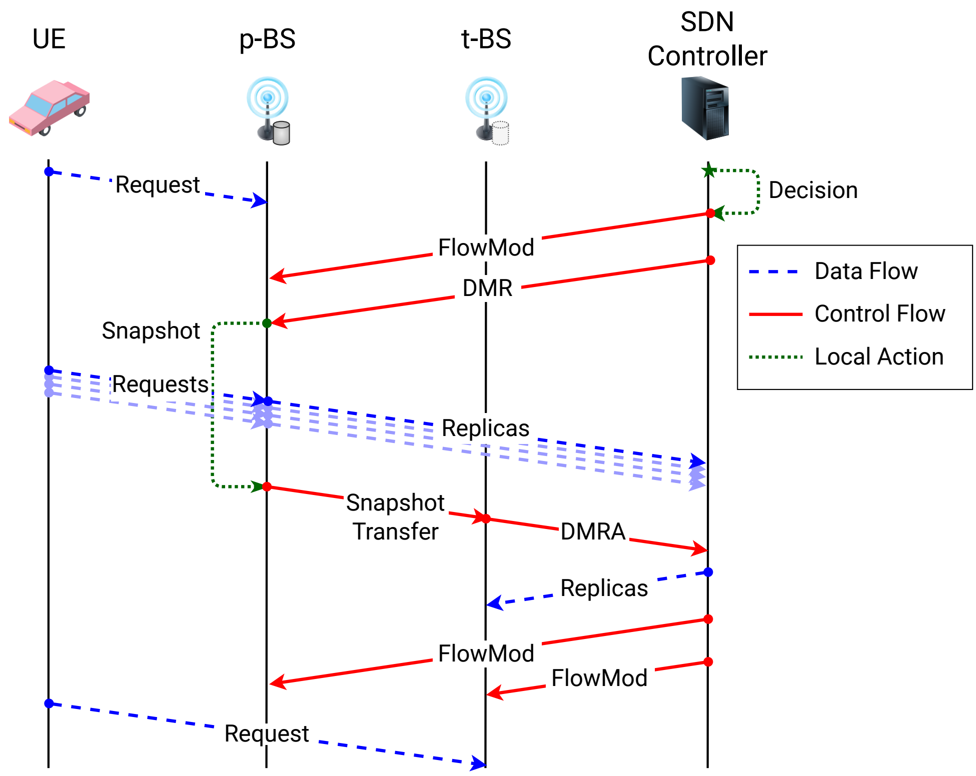

Figure 6.3: Workflow for moving ASUSD between different BSs [206] © 2023 IEEE.

steps will take place:

1. The SDN controller will issue Flow Modification (FlowMod) messages to install network communication flows to replicate all requests for stateful data at p-BS and forward these replicas to the controller.

2. The controller will send a Data Movement Request (DMR) to p-BS. This message has the objective of notifying p-BS to take a snapshot of the ASUSD and send it to t-BS.

3. During the time taken to produce the snapshot and send it to t-BS, incoming requests will still be served by p-BS, and replicas of these requests will be sent to the controller. This approach of storing data in the controller may not be ideal in some SDN architectures, for these cases a dedicated network service can be used as an alternative implementation. These replicas aim at reproducing the state changes that happened with the data stored at p-BS after the snapshot was taken when it arrived at t-BS.

4. After receiving the snapshot, t-BS sends a DMR Acknowledgment (DMRA) to the SDN controller. This informs the controller that the migration procedure is finished and that it can forward the replicas to t-BS. When multiple hosts need to receive the ASUSD, only one p-BS is sufficient to carry out the migration of the data, which should be the one currently in use by the UE, since this is the host receiving the new incoming requests to be replicated.

5. Once the DMRA has been received at the controller, the replica forwarding starts. Also, when needed, FlowMod messages are issued to update communication flow between UE and BSs that have the required ASUSD.

After this point, the service can consume the data from t-BS. During the migration process, the service did not need to stop serving the UE. This behavior allows the reduction of downtime of the service and also is important in cases when the migration needs to be aborted. Migrations may need to be aborted, for instance, when UE movements happen faster than the data movement. In this case, the service will keep running at p-BS without interruption, and the process can be restarted if needed after the SDN controller chooses a new t-BS candidate.

6.3.4 Data Synchronization

In order to operate our distributed application state strategy, copies of the ASUSD may have to be placed in different network nodes. A synchronization mechanism must be installed to ensure that all these copies are ready to be consumed by users. For that, we designed a simple synchronization mechanism that works in a similar fashion to the Asynchronous Data Movement described in Section 6.3.3, but without interaction with the SDN controller or another type of dedicated service in the network. Removing interaction with the controller is important to reduce the load placed on this network component.

Considering the example DC represented in Figure 6.4, after all handover procedures are executed, the UE will consume and write data to only one of the replicas, named Primary. The other replicas of the data exist to assist in the upcoming mobility events that happen. While they are not being consumed, they must be kept synchronized to be used when required. These copies are named Standby in Figure 6.4. The data synchronization process happens before the message reaches the application layer in the target data manager. Considering a user sent a "write" request, i.e., a request that will update the ASUSD, this request will be replicated by the Primary data manager, and each replica will be forwarded to a Standby data manager. This synchronization process is similar to different database replication architectures with one leader database and multiple follower databases.

In traditional leader/follower database architectures, the process of synchronization may lead to inconsistencies while the write messages are not fully propagated to all instances. However, the DCs scenario has some differences. First, the data synchronization happens among service instances that are connected directly on the wired network, and therefore do not need to use wireless hops to send the replicas of the write requests. This means that, in general, the synchronization process happens much faster than the consumption of the data by the UE. Second, the UE only consumes the data from a single instance at every attachment point. Since the UE will not write to one service instance and right after read from another, the synchronization has a significant time window to be carried out. The status data only needs to be consistent when the UE performs a handover. During the handover the UE is not able to consume the data since the attachment is not conclude. Thus, this synchronization have the entire handover time window

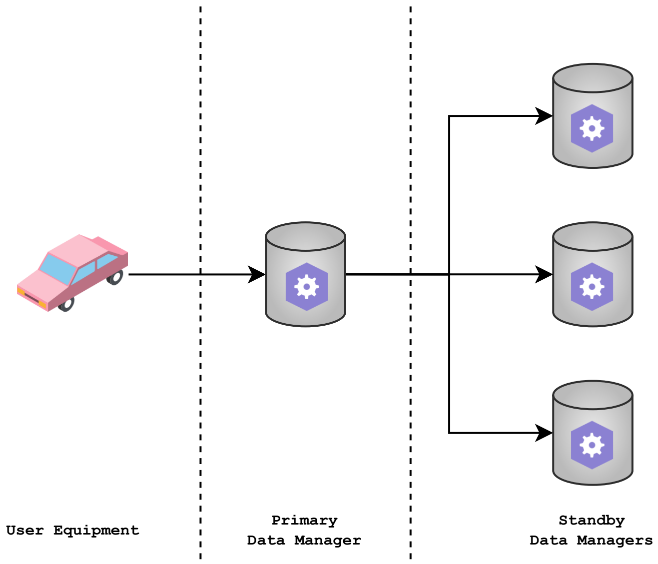

Figure 6.4: Example scenario for data synchronization within a DC.

plus the normal latency over the wireless communication hop to be finished. Finally, although small, the probability of achieving an inconsistent state still exists. One possible solution to ensure the synchronization will take place when it is needed is to consider a synchronization budget, similarly to the budget used when evaluating the DCs. This budget should be used to ensure that replicas of the ASUSD are not placed to far from each other and can be synchronized in a short time window. Ensuring data consistency at this level escapes the scope of the present thesis, however it is an interesting opportunity for future research.

6.4 Performance Evaluation

In this section, we compare three different service orchestration strategies with our proposal following the general evaluation methodology detailed in Section 4.3. Specific details of the methodology of the experiments of this chapter are given in Section 6.4.1. Section 6.4.2 discusses the latency and network path length results, whereas Section 6.4.3 describes the results related to Service Disruption Time (SDT).

6.4.1 Performance Evaluation Methods

The strategies evaluated are: (i) Static Services, where services were executed constantly in a data center connected to the edge network, i.e., single host; (ii) On-Demand Services, in which services were spawned at the nearest host to the UE and would not be moved; (iii) Follow Me Fog (FMF) based on a proposal from the literature [26] where services migrate to the nearest host to the UE after each handover; and our proposed DC strategy described in this chapter, using (iv) $b = 1$ ms, and (v) $b = 2$ ms. These budget values were selected considering the average round-trip wired link latency of the scenario, which is around 0.6 ms. These values lead to an average distance from data to UE access point

Table 6.2: Parameters used for the simulations [206] © 2023 IEEE.

Description	Value
Duration	1800 s
Repetitions	9 per scenario (45 total)
Mobility trace	Realistic [244]
Transmission Time Interval (TTI)	125 μs
Number of services	1-3 per vehicle
Service adoption rate	100%, 50%, and 10% respectively
Requests frequency	1 Hz, 0.2 Hz, and 0.1 Hz respectively
Service processing time	1-2 ms
Migration execution time	triang(100 ms, 1 s, 3 s)

of 1 hop in the 1 ms configuration and up to 3 hops for the 2 ms configuration.

Simulations were executed using a 750 BSs scenario shown in Figure 6.5. The blue nodes represent BSs with computing and storage power to run services, and that can work as AP for the UEs. The red nodes show the host of the SDN controller, and the green node shows where services were executed in the Static Services strategy. Finally, the links in Figure 6.5 represent the wired topology of the network. The position of the BSs was sampled from a real-world dataset [183] from the city of Cologne, Germany, with over 7000 LTE AP. Also, a realistic mobility trace from that city was used to simulate vehicular mobility [244]. More details about the simulation parameters are given in Table 6.2.

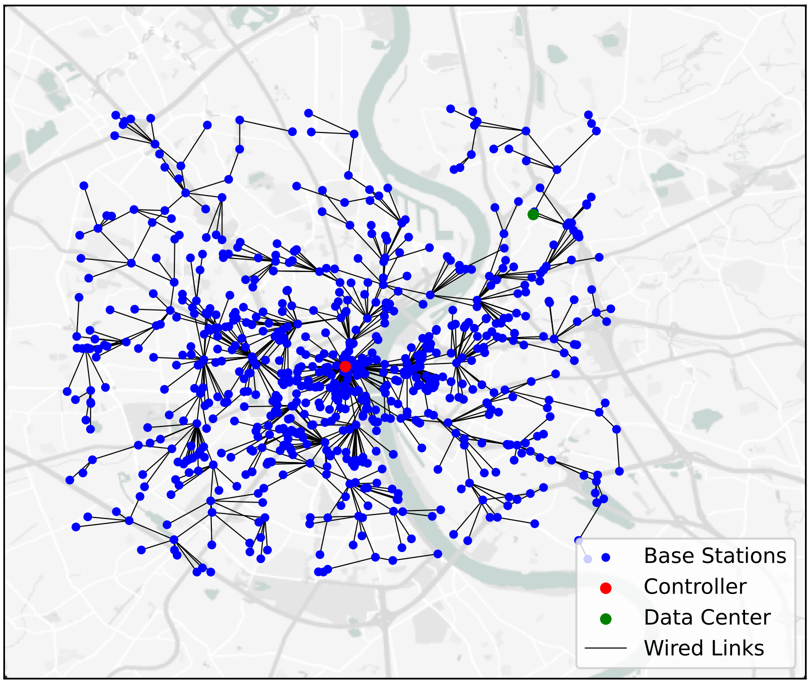

Figure 6.5: Network topology used in the experiments [206] © 2023 IEEE.

6.4.2 Latency and Network Path Length

Figure 6.6 shows the percentage of requests with latency under the requirements of three different classes of applications [133] listed along the X-axis: (i) vehicular automated overtake (10 ms), pre-crashing sensing and warning (20 ms), and see-through (50 ms). These latency values represent the required average latency for these applications to work properly. On the Y-axis this percentage of requests that satisfied the threshold is shown. The latency was measured as the end-to-end round-trip latency, i.e., since the request leaves the application layer at the UE until the success response is received at the same layer in the UE. This includes time consumed with re-attempts when services were unavailable at the target host. Our proposal with $b = 1$ ms and FMF were similar in the 10 ms class and performed better than the other approaches. In the 20 ms class, the two configurations of our proposal performed better the the other approaches. Finally, at the 50 ms class, the stability of the On-Demand approach together with the relative closer positioning to the UE leads this approach to be the best performance. From this data, it is possible to observe that service mobility is more important when meeting more restrictive latency thresholds. Furthermore, as shown in Figure 6.6, the budget parameter in our approach does not fully determine the latency observed in the requests. The final latency delivered is on the order of tens of milliseconds, while the budget was selected as 1 ms and 2 ms. This higher final latency is observed since many other factors have a significant impact on the latency values, such as the wireless link latency, network traffic, packets out of the optimal path, re-transmissions, and so on.

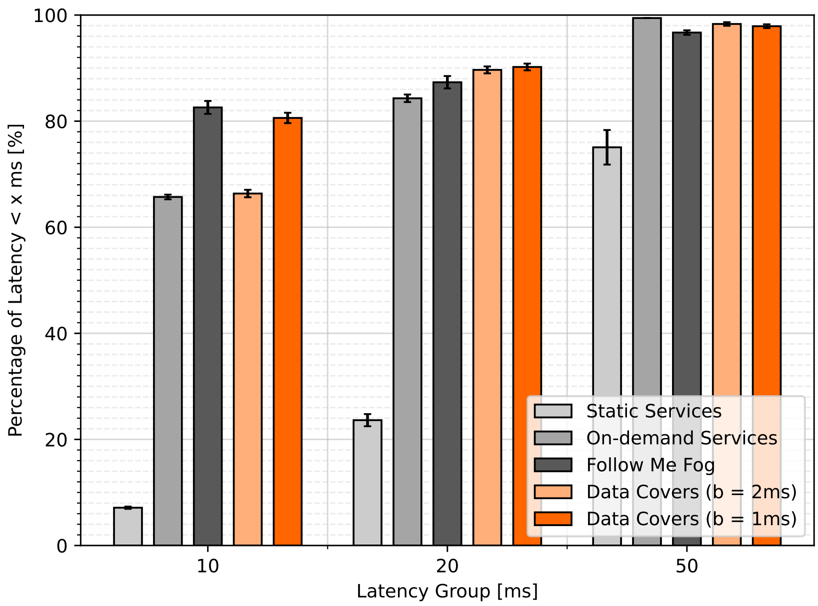

Figure 6.6: Distribution of round-trip latency perceived by UE per class with 99.9% confidence interval [206] © 2023 IEEE.

Figure 6.7 shows the overall distribution of latency observed where the final average latency delivered can be seen. On the Y-axis, the latency is shown in milliseconds, on the X-axis, the different strategies compared are listed. The red line in the middle of each box plot represents the median, with the exact value also shown in red. Latency outliers are not shown in the figure. In the simulation, requests were dropped only after 3 s without obtaining a response. FMF obtained a reduced latency compared to our approach. Still, on average, both could be used in strict latency requirement scenarios. This low latency level is obtained by FMF by performing many migrations, an approach that has a cost as shown in Section 6.4.3.

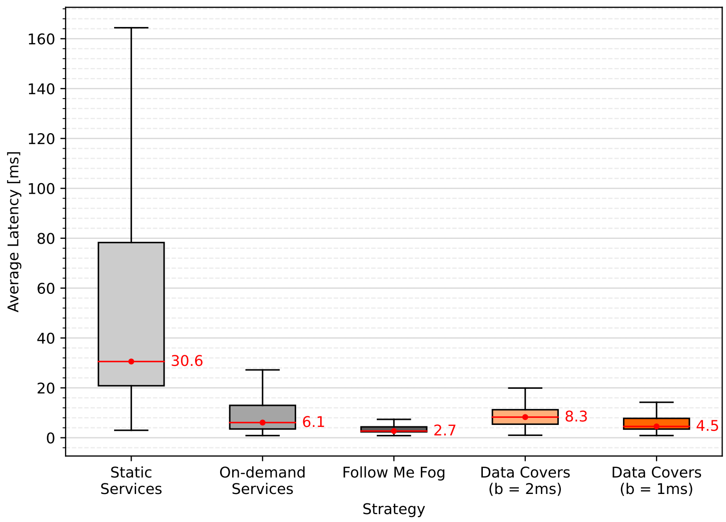

Figure 6.7: Overall distribution of latencies [206] © 2023 IEEE.

Figure 6.8 shows the distribution of the observed hop-distances from UE to services. On the X-axis, the different orchestration approaches are listed, while on the Y-axis, the single-way hop distance is shown. Each violin plot shows a black vertical rectangle indicating the range between the first and third quartiles of the distribution and also a white dot that indicates the mode of the distribution. Figure 6.8 allows the visualization of how much longer the distances from UE to services when using static approaches compared to approaches that re-locate services. FMF maintained service hop distance in the mark of 1 hop, only the wireless hop, for most of the time, with very low-frequency outliers. For our proposed approach, the modes of the distributions of hop distances were 4 and 2, i.e., 3 and 1 wired hops plus the wireless hop, for $b = 2$ and $b = 1$ milliseconds respectively. This is the expected behavior when considering the selection of these parameters, as described earlier in this section. Reduced distances from UEs to services are important to control latency stability and reduce overall network traffic, as less infrastructure is used to transmit the packets.

6.4.3 Service Disruption Time

Figure 6.9 shows the total SDT during the simulation, on the X-axis, the simulation time is shown, while on the Y-axis, the total SDT, both axes are shown in seconds. This variable was measured as the time that services were not available due to service mobility events. As expected, the two approaches with no service movement, i.e., Static Services and On-

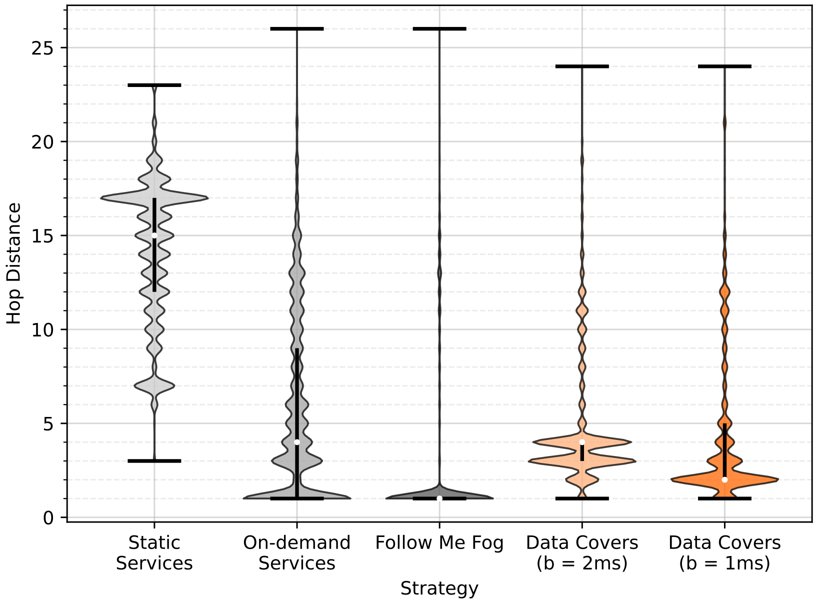

Figure 6.8: Overall distribution of hop distances between service and consumer for every request [206] © 2023 IEEE.

Demand Services, do not have any interruption time and, therefore, overlap at the zero mark. FMF obtained the worst performance on this variable because of the high number of service mobility events triggered by moving services whenever a handover happens. At the end of the simulation time, FMF had, on average, 41% more SDT when compared to our proposal. This high number of migrations can also be observed in Figure 6.10, which shows the total number of migrations throughout the simulation time. On the X-axis of Figure 6.10, the simulation time is shown in seconds, while on the Y-axis, the total number of migrations is displayed. On average, our proposal with a budget of 1 ms had 21% fewer migrations when compared to FMF. The higher number of migrations when using a budget of 1 ms when compared to a budget of 2 ms is due to the greater number of options available to host the ASUSD when combining the latency budget with mobility data. This higher number of host options leads to the formation of DCs that reduces the need for data movement triggered by user mobility.

Figure 6.11 shows the average total SDT per vehicle spawn in the simulation. On the X-axis, the simulation time is shown, and on the Y-axis, the average SDT in seconds per vehicle. The average SDT per vehicle settled between 2.5 s and 3 s for FMF, while settled between 1 s and 1.5 s for our proposal after t = 500 s in the simulation. As expected, our approach had a better performance compared to FMF, again justified by the reduced number of migrations performed in each approach.

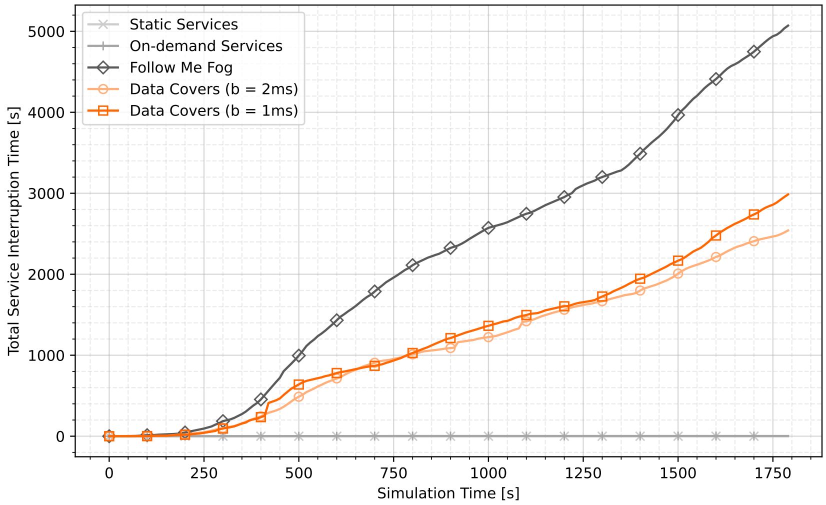

Figure 6.9: Cumulative Service Disruption Time (SDT) [206] © 2023 IEEE.

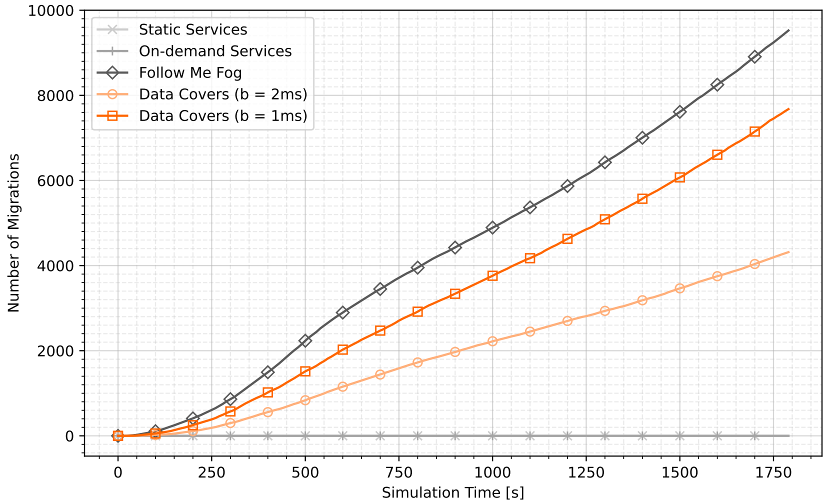

Figure 6.10: Total number of migrations performed over time [206] © 2023 IEEE.

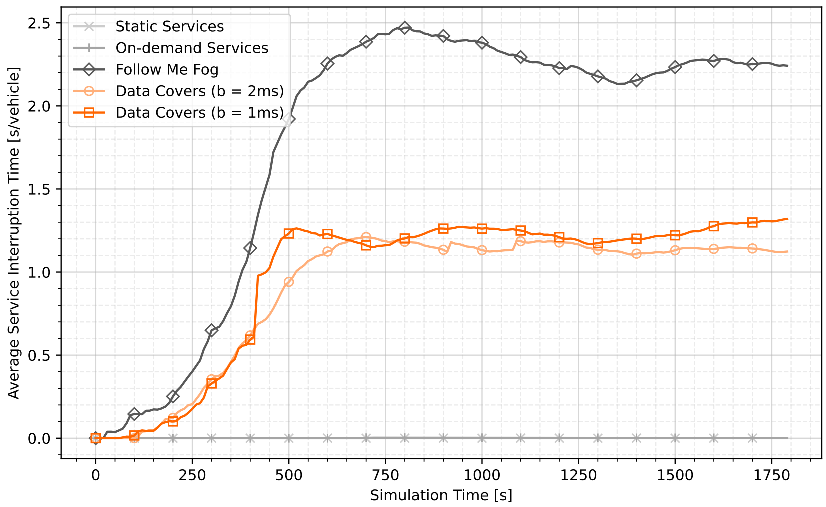

Figure 6.11: Average Service Disruption Time (SDT) per vehicle.

6.5 Chapter Conclusions

This chapter discusses data graphs and DCs that are used to compose a solution for application state and user session data at network nodes at the edge. We also discuss how this tool can be used together with user mobility data and a data movement protocol in the composition of this mobility-aware latency-constrained data placement solution. Our proposed approach is compared to other approaches, showing that we obtained similar to better performance when serving requests under end-to-end latency requirements of different classes of Future Internet applications while maintaining low service interruption time and reducing the number of data movement events required.

In the present chapter, we consider the usage of duplication of the data in different nodes to reduce the degradation in service consumption due to user mobility. Still, after handover events, users would consume data from a single host. However, partitioning and distributing ASUSD could also be used to reduce service latency further. This data distribution can be especially important when considering services where data reading and querying time is non-negligible and where time could be saved by reading data in parallel from different source hosts, which could lead to interesting future works.

Chapter 7

Conclusions and Future Works

This chapter discusses important remarks and lessons learned during the execution of the Ph.D. thesis. Section 7.1 discusses the thesis as a whole and lists takeaway insights obtained while investigating each individual research question. Section 7.2 lists possible next steps that could be explored regarding this thesis in the future.

7.1 Conclusions

Service provisioning has become one of the main aspects of communication over the Internet and is constantly evolving. In each step of evolution, new obstacles must be faced by industry and academia to meet the requirements of emerging services and applications. In the current step of evolution, service provisioning is moving from the centralized Cloud to the Edge. This shift raises issues related to the provisioning of the services, some of which are covered and studied in this thesis. Specifically, we study in this thesis how user and service mobility in the form of handovers in cellular networks and service host migrations can impact communication and how to reduce this impact.

We start the thesis at Chapter 1 by introducing the three main issues related to managing services in latency-constrained highly-mobile scenarios, such as when provisioning vehicular services. For each of these issues, we ask a research question and hypothesize how Future Internet paradigms, such as Edge Computing (EC), Software-Defined Networking (SDN), and Information-Centric Networking (ICN), could be used to mitigate these issues. These important paradigms in the scope of the present thesis, and also mobility management solutions and Future Internet applications are compiled in a literature review presented in Chapter 2. Besides this review, we also discuss, in Chapter 3, related solutions from the literature that also uses these paradigms and have a similar goal to this thesis. Some of these related solutions are used for comparison with our proposal, whereas some others were used as inspiration or basis for the algorithms and protocols proposed in this thesis.

Research Question Q.1 is investigated in Chapter 4, where we start by analyzing only the user mobility perspective. In this chapter, we study how SDN can be used when handling user mobility in the form of handover events in cellular networks. In order to propose our solution, we observe tendencies awaited for enhancing handover performance, specifically by avoiding the execution of the Random Access Channel (RACH) procedure and setting up the communication updates required after the handover before disconnection, i.e., Make-Before-Break (MBB). In our work, we use a recent proposal from the literature on how to perform RACH-less and MBB handovers in non-synchronized networks [52] as a basis to develop an SDN-enabled RACH-less MBB handover scheme. We design a protocol that allows the necessary information exchange between all parties involved in the handover, i.e., user, SDN controller, and both Base Stations (BSs), considering this literature proposal and also SDN requirements. With this scheme, we were able to significantly reduce Handover Execution Time (HET) compared to other SDN-enabled handovers from the literature [30].

Research Question Q.2, discussed in Chapter 5, focuses on what impacts different addressing approaches have on communication latency in the presence of user and service mobility events. In this chapter, we explore the usage of an ICN inspired protocol for service provisioning implemented on top of SDN features. In order to emulate ICN behavior on top of SDN we used a strategy based on hashes. Still, using simple flat hashes to represent service names would lead to the loss of an important feature of network routing, the prefix-based matching. This feature allows routing rules to match packets based on patterns, which reduces the size of routing tables, and in SDN, it also permits the reduction of the number of packets to be sent to the controller for routing. We proposed a hierarchical hashing strategy to keep the possibility of using prefix-based matching when routing. In the end, when using name-based routing instead of topology-based, such as in the IP, we observed our performance had a better performance for handling user and service mobility events. Proactive IP strategies using SDN could achieve similar performance compared to our proposal when handling user mobility. However, our proposal had a significantly better performance when handling service mobility events.

Finally, Research Question Q.3 is presented in Chapter 6. In this chapter, we focus orchestrating addressing stateful services, which are services that have specific application state data that differentiates one instance from another. When consuming this type of service, users cannot consume any instance of the service, but only the one that has the correct state for its usage. One problem when consuming this type of service happens when the user moves away, topologically, from the host of the state data. Higher latencies to access this state data may increase the total service latency and put at risk Quality of Service (QoS). We propose a graph-based algorithm to position state data in the network, considering user mobility patterns and latency constraints. Our main goal was to reduce the necessity of relocating this data, thus also reducing the interruption time due to service migration. Our graph-based algorithm uses a latency budget when selecting possible nodes to store state data. These nodes are then selected to host the data according to historical user mobility data. As expected, we observed a decrease in the number of service migrations and interruption time when comparing our approach to other service orchestration solutions.

7.2 Future Works

In the present thesis we studied different topics related to our research questions, each one of these topics may be further studied. Regarding Research Question Q.1, we focused our studies on designing a signaling protocol for handover, omitting the decision process of whether the handover should be performed and how to choose the best available BS. Studying this decision process would be an interesting future work to continue exploring Q.1. In our study, the decision was taking by simply looking into quality of signal variables, such as the Received Signal Strength Indicator (RSSI), at the User Equipments (UEs). However, Machine Learning (ML) algorithms, specifically models design for mobility prediction, could aid in this decision process. Another possible future work would be to perform a more exploratory analysis of different scenario configurations, e.g., random mobility, changing vehicle speed, or topology variation. Specifically for high-speed mobility, handover success rates tend to decrease, thus increasing HET, which might require the usage of other handover mechanisms to achieve expected HET.

When proposing an ICN addressing scheme on top of SDN in the scope of Research Question Q.2, we did not explore some of the features enabled by name-based routing. For instance, by addressing network objects using names, multi-path communication can be performed, and also consuming these objects from multiple hosts. This is an interesting feature of ICN-based protocols that can be used to further reduce communication latency by consuming data in parallel from different hosts, and also reliability by creating redundant requests to multiple nodes that can respond in case of failure of one of them.

In the context of Research Question Q.3, similarly to Q.2, the distributed positioning and consumption of stateful data can be used achieve further reduction of latency and reliability. In our studies we focused on achieving simpler placements for the data in which the state data was not divided. However, this distribution can be especially important when considering services where data reading and querying time is non-negligible and where time could be saved by reading data in parallel from different source hosts. In our studies we also focused on finding data placements that reduce the amount of data duplication. Still, duplicating data can also be used to increase reliability when accessing this data.

7.3 Publications

During the Ph.D. course the following studies were published in internationally recognized venues:

1. **Rodrigues, D. O.**, Souza, A. M. de, Braun, T., Maia, G., Loureiro, A. A. F., and Villas, L. A. (2023). Service Provisioning in Edge-Cloud Continuum: Emerging Applications for Mobile Devices, in *SBC Journal of Internet Services and Applications*, 2023.

2. **Rodrigues, D. O.**, Braun, T., Maia, G., and Villas, L. (2023). Mobility-aware Latency-constrained Data Placement in SDN-enabled Edge Networks, in *Proceedings of the IEEE/IFIP Network Operations and Management Symposium*, 2023.

3. **Rodrigues, D. O.**, Braun, T., Maia, G., and Villas, L. (2022). Mobility-aware Software-Defined Service-Centric Networking, in *Proceedings of the IEEE International Conference on Computer Communications and Networks*, 2022, 1–10.

4. **Rodrigues, D. O.**, Braun, T., Maia, G., and Villas, L. (2021). Towards SDN-enabled RACH-less Make-before-break Handover in C-V2X Scenarios, in *Proceedings of the IEEE International Conference on Wireless and Mobile Computing, Networking and Communications*, 2021, 337–344.

5. **Rodrigues, D. O.**, Maia, G., Braun, T., Loureiro, A. A. F., Peixoto, M. L. M., and Villas, L. A. (2021). Exploring Hybrid-Multimodal Routing to Improve User Experience in Urban Trips, in *MDPI Journal of Applied Sciences*, 2021, 11(10).

6. Meneguette, R. I., **Rodrigues, D. O.**, Costa, J. B. D., Rosário, D., Villas, L. A., da Costa, J. B. D., Rosario, D., and Villas, L. A. (2019). A Virtual Machine Migration Policy Based on Multiple Attribute Decision in Vehicular Cloud Scenario, in *Proceedings of the IEEE International Conference on Communications*, 2019, 1–6.

Bibliography

[1] 3GPP. TS 136 133 - V12.5.0 - LTE; Evolved Universal Terrestrial Radio Access (E-UTRA); Requirements for support of radio resource management (3GPP TS 36.133 version 12.5.0 Release 12), 2014.

[2] 3GPP. TR 36.881 V14.0.0 - Technical Specification Group Radio Access Network; Evolved Universal Terrestrial Radio Access (E-UTRA); Study on latency reduction techniques for LTE (Release 14). Technical Report Release 14, 3GPP, 2016.

[3] Muhammad Tahir Abbas, Afaq Muhammad, and Wang-Cheol Song. SD-IoV: SDN enabled routing for internet of vehicles in road-aware approach. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–16, may 2019.

[4] Randa M. Abdelmoneem, Abderrahim Benslimane, Eman Shaaban, Sherin Abdelhamid, and Salma Ghoneim. A Cloud-Fog Based Architecture for IoT Applications Dedicated to Healthcare. In *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, may 2019.

[5] Randa M. Abdelmoneem, Abderrahim Benslimane, Eman Shaaban, Sherin Abdelhamid, and Salma Ghoneim. A Cloud-Fog Based Architecture for IoT Applications Dedicated to Healthcare. In *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, may 2019.

[6] Leila Abdollahi Vayghan, Mohamed Aymen Saied, Maria Toeroe, and Ferhat Khendek. Microservice based architecture: Towards high-availability for stateful applications with kubernetes. In *2019 IEEE 19th International Conference on Software Quality, Reliability and Security (QRS)*, pages 176–185, 2019.

[7] Sergio Abriola, Pablo Barceló, Diego Figueira, and Santiago Figueira. Bisimulations on data graphs. *J. Artif. Int. Res.*, 61(1):171–213, jan 2018.

[8] Enas Ahmad, Maha Alaslani, Fahad R Dogar, and Basem Shihada. Location-Aware, Context-Driven QoS for IoT Applications. *IEEE Systems Journal*, pages 1–12, 2019.

[9] R. Ahmed and R. Boutaba. Design considerations for managing wide area software defined networks. *IEEE Communications Magazine*, 52(7):116–123, 2014.

[10] Syed Hassan Ahmed, Safdar Hussain Bouk, Dongkyun Kim, Danda B Rawat, and Houbing Song. Named Data Networking for Software Defined Vehicular Networks. *IEEE Communications Magazine*, 55(8):60–66, 2017.

[11] Syed Hassan Ahmed and Shalli Rani. A hybrid approach, Smart Street use case and future aspects for Internet of Things in smart cities. *Future Generation Computer Systems*, 79:941–951, 2018.

[12] Syed Hassan Ahmed, M.A. Yaqub, S.H. Bouk, and Dongkyun Kim. Towards content-centric traffic ticketing in VANETs: An application perspective. In *2015 Seventh International Conference on Ubiquitous and Future Networks*, pages 237–239. IEEE, jul 2015.

[13] Syed Hassan Ahmed, Muhammad Azfar Yaqub, Safdar Hussain Bouk, and Dongkyun Kim. SmartCop: Enabling Smart Traffic Violations Ticketing in Vehicular Named Data Networks. *Mobile Information Systems*, 2016:1–12, 2016.

[14] Abdelkader Aissioui, Adlen Ksentini, Abdelhak Mourad Gueroui, and Tarik Taleb. On Enabling 5G Automotive Systems Using Follow Me Edge-Cloud Concept. *IEEE Transactions on Vehicular Technology*, 67(6):5302–5316, 2018.

[15] M. S. Akbar, H. Yu, and S. Cang. Tmp: Tele-medicine protocol for slotted 802.15.4 with duty-cycle optimization in wireless body area sensor networks. *IEEE Sensors Journal*, 17(6):1925–1936, March 2017.

[16] Khadija Akherfi, Micheal Gerndt, and Hamid Harroud. Mobile cloud computing for computation offloading: Issues and challenges. *Applied Computing and Informatics*, 14(1):1–16, 2018.

[17] Fadi M Al-Turjman. Information-centric sensor networks for cognitive IoT: an overview. *Annals of Telecommunications*, 72(1):3–18, feb 2017.

[18] Imad Alawe, Adlen Ksentini, Yassine Hadjadj-Aoul, and Philippe Bertin. Improving Traffic Forecasting for 5G Core Network Scalability: A Machine Learning Approach. *IEEE Network*, 32(6):42–49, nov 2018.

[19] Mohammed Albrahim, Ahmed Al Zahrani, Anvita Arora, Rubal Dua, Bassam Fattouh, and Adam Sieminski. An overview of key evolutions in the light-duty vehicle sector and their impact on oil demand. *Energy Transitions*, 3(1):81–103, 2019.

[20] Rashid Amin, Martin Reisslein, and Nadir Shah. Hybrid SDN networks: A survey of existing approaches. *IEEE Communications Surveys and Tutorials*, 20(4):3259–3306, 2018.

[21] K. Antonakoglou, X. Xu, E. Steinbach, T. Mahmoodi, and M. Dohler. Toward haptic communications over the 5g tactile internet. *IEEE Communications Surveys Tutorials*, 20(4):3034–3059, Fourthquarter 2018.

[22] Sobia Arshad, Muhammad Awais Azam, Mubashir Husain Rehmani, Senior Member, and Jonathan Loo. Recent Advances in Information-Centric Networking-Based Internet of Things (ICN-IoT). *IEEE Internet of Things Journal*, 6(2):2128–2158, 2019.

[23] Ashwin Ashok, Peter Steenkiste, and Fan Bai. Vehicular Cloud Computing through Dynamic Computation Offloading. *Computer Communications*, 120:125–137, may 2018.

[24] Enzo Baccarelli, Michele Scarpiniti, and Alireza Momenzadeh. Fog-Supported Delay-Constrained Energy-Saving Live Migration of VMs Over MultiPath TCP/IP 5G Connections. *IEEE Access*, 6:42327–42354, 2018.

[25] Ahmet Cihat Baktir, Atay Ozgovde, and Cem Ersoy. How Can Edge Computing Benefit From Software-Defined Networking: A Survey, Use Cases, and Future Directions. *IEEE Communications Surveys & Tutorials*, 19(4):2359–2391, 2017.

[26] Wei Bao, Dong Yuan, Zhengjie Yang, Shen Wang, Wei Li, Bing Bing Zhou, and Albert Y. Zomaya. Follow Me Fog: Toward Seamless Handover Timing Schemes in a Fog Computing Environment. *IEEE Communications Magazine*, 55(11):72–78, 2017.

[27] Simone Barbera, Klaus I. Pedersen, Claudio Rosa, Per Henrik Michaelsen, Frank Frederiksen, Ejaz Shah, and Al Baumgartner. Synchronized RACH-less handover solution for LTE heterogeneous networks. In *Proceedings of the International Symposium on Wireless Communication Systems*, volume 2016-April, pages 755–759. IEEE, 2015.

[28] Thiwiza Bellache, Oyunchimeg Shagdar, and Samir Tohme. Dcc-enabled contention based forwarding scheme for vanets. In *2017 IEEE 13th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pages 1–8, 2017.

[29] Jitendra Bhatia, Ridham Dave, Heta Bhayani, Sudeep Tanwar, and Anand Nayyar. SDN-based real-time urban traffic analysis in VANET environment. *Computer Communications*, 149:162–175, 2019.

[30] Yuanguo Bi, Guangjie Han, Chuan Lin, Qingxu Deng, Lei Guo, and Fuliang Li. Mobility Support for Fog Computing: An SDN Approach. *IEEE Communications Magazine*, 56(5):53–59, may 2018.

[31] Yuanguo Bi, Guangjie Han, Chuan Lin, Mohsen Guizani, and Xingwei Wang. Mobility Management for Intro/Inter Domain Handover in Software-Defined Networks. *IEEE Journal on Selected Areas in Communications*, 37(8):1739–1754, 2019.

[32] Luiz F. Bittencourt, Javier Diaz-Montes, Rajkumar Buyya, Omer F. Rana, and Manish Parashar. Mobility-Aware Application Scheduling in Fog Computing. *IEEE Cloud Computing*, 4(2):26–35, mar 2017.

[33] Mate Boban, Apostolos Kousaridas, Konstantinos Manolakis, Josef Eichinger, and Wen Xu. Connected roads of the future: Use cases, requirements, and design considerations for vehicle-To-everything communications. *IEEE Vehicular Technology Magazine*, 13(3):110–123, 2018.

[34] K. A. Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloé M Kiddon, Jakub Konečný, Stefano Mazzocchi, Brendan McMahan, Timon Van Overveldt, David Petrou, Daniel Ramage, and Jason Roselander. Towards federated learning at scale: System design. In *SysML 2019*, 2019. To appear.

[35] Way Kiat Bong, Weiqin Chen, and Astrid Bergland. Tangible User Interface for Social Interactions for the Elderly: A Review of Literature. *Adv. Human-Computer Interaction*, 2018:7249378:1–7249378:15, 2018.

[36] Flavio Bonomi, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli. Fog computing and its role in the internet of things. In *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*, MCC '12, page 13–16, New York, NY, USA, 2012. Association for Computing Machinery.

[37] T Braun, V Hilt, M Hofmann, I Rimac, M Steiner, and M Varvello. Service-Centric Networking. In *2011 IEEE International Conference on Communications Workshops (ICC)*, pages 1–6, 2011.

[38] Torsten Braun, Andreas Mauthe, and Vasilios Siris. Service-centric Networking Extensions. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, SAC '13, pages 583–590, New York, NY, USA, 2013. ACM.

[39] Claudia Campolo, Antonio Iera, Antonella Molinaro, and Giuseppe Ruggeri. MEC Support for 5G-V2X Use Cases through Docker Containers. *IEEE Wireless Communications and Networking Conference, WCNC*, 2019-April:0–5, 2019.

[40] Claudio Canuto and Anita Tabacco. Taylor expansions and applications. In *Mathematical Analysis I*, volume 84 of *La Matematica per il 3+2*, chapter 7, pages 225–257. Springer International Publishing, 2 edition, 2015.

[41] Shengbin Cao and Victor C.S. Lee. A novel adaptive TDMA-based MAC protocol for VANETs. *IEEE Communications Letters*, 22(3):614–617, 2018.

[42] Paris Carboney, Stephan Ewenz, Gyula Fóra, Seif Haridi, Stefan Richter, Kostas Tzoumas, Paris Carbone, Stephan Ewen, Gyula Fóra, Seif Haridi, Stefan Richter, and Kostas Tzoumas. State management in apache flink®: Consistent stateful distributed stream processing. *Proc. VLDB Endow.*, 10:1718–1729, 2017.

[43] Claude Castelluccia. HMIPv6: A Hierarchical Mobile IPv6 Proposal. *SIGMOBILE Mob. Comput. Commun. Rev.*, 4(1):48–59, 2000.

[44] Minsu Chae, Hwamin Lee, and Kiyeol Lee. A performance comparison of linux containers and virtual machines using Docker and KVM. *Cluster Computing*, 22(s1):1765–1775, 2018.

[45] Jacob Chakareski. Vr/ar immersive communication: Caching, edge computing, and transmission trade-offs. In *Proceedings of the Workshop on Virtual Reality and*

Augmented Reality Network, VR/AR Network '17, page 36–41, New York, NY, USA, 2017. Association for Computing Machinery.

[46] Bernard Chazelle. A minimum spanning tree algorithm with inverse-ackermann type complexity. *Journal of the ACM*, 47(6):1028–1047, 2000.

[47] Djabir Abdeldjalil Chekired, Mohammed Amine Togou, Lyes Khoukhi, and Adlen Ksentini. 5G-Slicing-Enabled Scalable SDN Core Network: Toward an Ultra-Low Latency of Autonomous Driving Service. *IEEE Journal on Selected Areas in Communications*, 37(8):1769–1782, aug 2019.

[48] M Chen, Y Miao, X Jian, X Wang, and I Humar. Cognitive-LPWAN: Towards Intelligent Wireless Services in Hybrid Low Power Wide Area Networks. *IEEE Transactions on Green Communications and Networking*, 3(2):409–417, 2019.

[49] Min Chen, Wei Li, Giancarlo Fortino, Yixue Hao, Long Hu, and Iztok Humar. A dynamic service migration mechanism in edge cognitive computing. *ACM Trans. Internet Technol.*, 19(2), apr 2019.

[50] Shenzhi Chen, Jinling Hu, Yan Shi, Li Zhao, and Wen Li. A Vision of C-V2X: Technologies, Field Testing, and Challenges With Chinese Development. *IEEE Internet of Things Journal*, 7(5):3872–3881, 2020.

[51] Yuanhang Cheng, Xueshu Xu, Yingkui Du, Ping Guan, Shu Liu, and Lijuan Zhao. Design of air quality monitoring system based on nb-iot. In *2019 IEEE International Conference on Power, Intelligent Computing and Systems (ICPICS)*, pages 385–388, 2019.

[52] Ji-Hwan Choi and Dong-Joon Shin. Generalized RACH-Less Handover for Seamless Mobility in 5G and Beyond Mobile Networks. *IEEE Wireless Communications Letters*, 8(4):1264–1267, aug 2019.

[53] Anthony Cuthbertson. Galaxy Fold: Inside Samsung's struggle to deliver a foldable phone – and why the future of smartphones hinges on it, 2019.

[54] Anestis Dalgkitsis, Prodromos-Vasileios Mekikis, Angelos Antonopoulos, and Christos Verikoukis. Data driven service orchestration for vehicular networks. *IEEE Transactions on Intelligent Transportation Systems*, 22(7):4100–4109, 2021.

[55] Debasis Das. Distributed algorithm for geographic opportunistic routing in VANETs at road intersection. *Proceedings - 2017 IEEE 15th International Conference on Dependable, Autonomic and Secure Computing, 2017 IEEE 15th International Conference on Pervasive Intelligence and Computing, 2017 IEEE 3rd International Conference on Big Data Intelligence and Computing and 2017 IEEE Cyber Science and Technology Congress, DASC-PICom-DataCom-CyberSciTec 2017*, 2018-January:1202–1209, 2018.

[56] Antonio Di Maio, Ridha Soua, Maria Rita Palattella, Thomas Engel, and Gianluca A. Rizzo. A centralized approach for setting floating content parameters in VANETs. In *2017 14th IEEE Annual Consumer Communications and Networking Conference, CCNC 2017*, pages 712–715. IEEE, 2017.

[57] Ikram Ud Din, Suhaidi Hassan, Muhammad Khurram Khan, Mohsen Guizani, Osman Ghazali, and Adib Habbal. Caching in Information-Centric Networking: Strategies, Challenges, and Future Research Directions. *IEEE Communications Surveys and Tutorials*, 20(2):1443–1474, 2018.

[58] John David N. Dionisio, William G. Burns III, and Richard Gilbert. 3d virtual worlds and the metaverse: Current status and future possibilities. *ACM Comput. Surv.*, 45(3), jul 2013.

[59] Truong Xuan Do and Younghan Kim. Latency-aware placement for state management functions in service-based 5G mobile core network. In *2018 IEEE 7th International Conference on Communications and Electronics, ICCE 2018*, pages 102–106. Institute of Electrical and Electronics Engineers Inc., sep 2018.

[60] Rajdeep Dua, A. Reddy Raja, and Dharmesh Kakadia. Virtualization vs containerization to support PaaS. *Proceedings - 2014 IEEE International Conference on Cloud Engineering, IC2E 2014*, pages 610–614, 2014.

[61] Joao M. Duarte, Torsten Braun, and Leandro A. Villas. MobiVNDN: A distributed framework to support mobility in vehicular named-data networking. *Ad Hoc Networks*, 82:77–90, jan 2019.

[62] Corentin Dupont, Raffaele Giaffreda, and Luca Capra. Edge computing in IoT context: Horizontal and vertical Linux container migration. *GIoTS 2017 - Global Internet of Things Summit, Proceedings*, pages 2–5, 2017.

[63] Mustafa Ergen. *Mobile Broadband*. Springer US, Boston, MA, 2009.

[64] ETSI. TR 22.885 - V14.0.0 - Technical Specification Group Services and System Aspects; Study on LTE support for Vehicle to Everything (V2X) services (Release 14). Technical Report Release 14, European Telecommunications Standards Institute, 2015.

[65] ETSI. GR MEC 018 - V1.1.1 - Mobile edge computing: End to End Mobility Aspects. Technical report, European Telecommunications Standards Institute, 2017.

[66] ETSI. GR MEC 022 - V2.1.1 - Study on MEC Support for V2X Use Cases. Technical report, European Telecommunications Standards Institute, 2018.

[67] ETSI. TS 102 636-4-2 - V1.4.1 - Intelligent Transport Systems (ITS); Vehicular Communications; GeoNetworking; Part 4: Geographical addressing and forwarding for point-to-point and point-to-multipoint communications; Sub-part 2: Media-dependent functionalities F. Technical report, European Telecommunications Standards Institute, 2021.

[68] ETSI. TS 123 287 - V16.3.0 - 5G; Architecture enhancements for 5G System (5GS) to support Vehicle-to-Everything (V2X) services (3GPP TS 23.287 version 16.3.0 Release 16). Technical report, European Telecommunications Standards Institute, 2021.

[69] C Fang, H Yao, Z Wang, W Wu, X Jin, and F R Yu. A Survey of Mobile Information-Centric Networking: Research Issues and Challenges. *IEEE Communications Surveys Tutorials*, 20(3):2353–2371, 2018.

[70] George Favaloro. Internet Solutions Division Strategy for Cloud Computing, 1996.

[71] L Fawcett, M Mu, B Hareng, and N Race. REF: Enabling Rapid Experimentation of Contextual Network Traffic Management Using Software Defined Networking. *IEEE Communications Magazine*, 55(7):144–150, 2017.

[72] Tiago M. Fernández-Caramés, Paula Fraga-Lamas, Manuel Suárez-Albela, and Miguel Vilar-Montesinos. A Fog Computing and Cloudlet Based Augmented Reality System for the Industry 4.0 Shipyard. *Sensors*, 18(6):1–18, 2018.

[73] Roberto Rodrigues Filho and Barry Porter. Autonomous State-Management Support in Distributed Self-adaptive Systems. In *2020 IEEE International Conference on Autonomic Computing and Self-Organizing Systems Companion*, pages 176–181. Institute of Electrical and Electronics Engineers (IEEE), sep 2020.

[74] Pablo Fondo-Ferreiro, Felipe Gil-Castineira, Francisco Javier Gonzalez-Castano, and David Candal-Ventureira. A Software-Defined Networking Solution for Transparent Session and Service Continuity in Dynamic Multi-Access Edge Computing. *IEEE Transactions on Network and Service Management*, 18(2):1401–1414, 2021.

[75] Nikos Fotiou, Dirk Trossen, and George C. Polyzos. Illustrating a publish-subscribe Internet architecture. *Telecommunication Systems*, 51(4):233–245, 2012.

[76] Paula Fraga-Lamas, Tiago M. Fernández-Caramés, Óscar Blanco-Novoa, and Miguel A. Vilar-Montesinos. A Review on Industrial Augmented Reality Systems for the Industry 4.0 Shipyard. *IEEE Access*, 6:13358–13375, 2018.

[77] Benjamin Frank, Ingmar Poese, Yin Lin, Georgios Smaragdakis, Anja Feldmann, Bruce Maggs, Jannis Rake, Steve Uhlig, and Rick Weber. Pushing CDN-ISP collaboration to the limit. *ACM SIGCOMM Computer Communication Review*, 43(3):34, jul 2013.

[78] Holger Füßler, Jörg Widmer, Michael Käsemann, Martin Mauve, and Hannes Hartenstein. Contention-based forwarding for mobile ad hoc networks. *Ad Hoc Networks*, 1(4):351–369, 2003.

[79] Joseph A. Gallian. A dynamic survey of graph labeling. *Electronic Journal of Combinatorics*, 1(DynamicSurveys), 2018. Publisher Copyright: © 2018, Australian National University. All rights reserved.

[80] M Gasparyan, A Marandi, E Schiller, and T Braun. Fault-Tolerant Session Support for Service-Centric Networking. In *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, pages 312–320, apr 2019.

[81] Mikael Gasparyan, Torsten Braun, and Eryk Schiller. L-SCN: Layered SCN architecture with supernodes and Bloom filters. In *2017 14th IEEE Annual Consumer Communications Networking Conference (CCNC)*, pages 899–904, Las Vegas, NV, USA, jan 2017. IEEE.

[82] Mikael Gasparyan, Guillaume Corsini, Torsten Braun, Eryk Schiller, and Jonnahtan Saltarin. Session support for SCN. In *2017 IFIP Networking Conference (IFIP Networking) and Workshops*, pages 1–6. IEEE, jun 2017.

[83] X. Gelabert, G. Zhou, and P. Legg. Mobility performance and suitability of macro cell power-off in lte dense small cell hetnets. In *2013 IEEE 18th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, pages 99–103, Sep. 2013.

[84] Carles Grau, Romuald Ginhoux, Alejandro Riera, Thanh Lam Nguyen, Hubert Chauvat, Michel Berg, Julià L Amengual, Alvaro Pascual-Leone, and Giulio Ruffini. Conscious Brain-to-Brain Communication in Humans Using Non-Invasive Technologies. *PLOS ONE*, 9(8):1–6, 2014.

[85] GreenICN Project. GreenICN: Architecture and Applications of Green Information Centric Networking, 2013.

[86] Daphne Guibert, Jun Wu, Shan He, Meng Wang, and Jianhua Li. CC-fog: Toward content-centric fog networks for E-health. In *2017 IEEE 19th International Conference on e-Health Networking, Applications and Services (Healthcom)*, pages 1–5. IEEE, oct 2017.

[87] Harshit Gupta and Umakishore Ramachandran. Fogstore: A geo-distributed key-value store guaranteeing low latency for strongly consistent access. In *Proceedings of the 12th ACM International Conference on Distributed and Event-Based Systems*, DEBS '18, page 148–159, New York, NY, USA, 2018. Association for Computing Machinery.

[88] Sairam Gurajada and Martin Theobald. Distributed set reachability. In *Proceedings of the 2016 International Conference on Management of Data*, SIGMOD '16, page 1247–1261, New York, NY, USA, 2016. Association for Computing Machinery.

[89] Stephen Gutz, Alec Story, Cole Schlesinger, and Nate Foster. Splendid Isolation: A Slice Abstraction for Software-defined Networks. In *Proceedings of the First Workshop on Hot Topics in Software Defined Networks*, HotSDN '12, pages 79–84, New York, NY, USA, 2012. ACM.

[90] Ryo Hagihara, Yasuhiro Yamasaki, and Hiroyuki Ohsaki. On delivery control for floating contents sharing with epidemic broadcasting. *2017 14th IEEE Annual Consumer Communications and Networking Conference, CCNC 2017*, 1(2):353–356, 2017.

[91] Bo Han. Mobile Immersive Computing: Research Challenges and the Road Ahead. *IEEE Communications Magazine*, 57(10):112–118, 2019.

[92] D. Han, S. Shin, H. Cho, J. Chung, D. Ok, and I. Hwang. Measurement and stochastic modeling of handover delay and interruption time of smartphone real-time applications on lte networks. *IEEE Communications Magazine*, 53(3):173–181, March 2015.

[93] W Han, M Dong, K Ota, J Wu, J Li, and G Li. SD-OPTS: Software-Defined On-Path Time Synchronization for Information-Centric Smart Grid. In *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, pages 1–6, 2017.

[94] Zecheng He, Tianwei Zhang, and Ruby B. Lee. Model inversion attacks against collaborative inference. In *Proceedings of the 35th Annual Computer Security Applications Conference*, ACSAC '19, page 148–162, New York, NY, USA, 2019. Association for Computing Machinery.

[95] Marc Heissenbüttel, Torsten Braun, Thomas Bernoulli, and Markus Wälchli. BLR: Beacon-less routing algorithm for mobile ad hoc networks. *Computer Communications*, 27(11):1076–1086, 2004.

[96] Thomas Henderson, Christian Vogt, and Jari Arkko. Host mobility with the host identity protocol. Technical report, Internet Engineering Task Force, 2017.

[97] Y Ho, Chun-Han Lin, Shih-Jie Sun, and Ling-Jyh Chen. Emergency application for Vehicle-to-Vehicle Communication using Named Data Networking (eVNDN). In *2016 Eighth International Conference on Ubiquitous and Future Networks (ICUFN)*, pages 611–616, 2016.

[98] Florian Hofer, Martin A Sehr, Antonio Iannopollo, Ines Ugalde, Alberto L Sangiovanni-Vincentelli, and Barbara Russo. Industrial Control via Application Containers: Migrating from Bare-Metal to {IAAS}. *arXiv*, abs/1908.0, 2019.

[99] Marius Hoggenmueller and Martin Tomitsch. Enhancing Pedestrian Safety Through In-situ Projections: A Hyperreal Design Approach. In *Proceedings of the 8th ACM International Symposium on Pervasive Displays*, PerDis '19, pages 29:1—-29:2, New York, NY, USA, 2019. ACM.

[100] C Hu, W Bao, D Wang, and F Liu. Dynamic Adaptive DNN Surgery for Inference Acceleration on the Edge. In *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, pages 1423–1431, apr 2019.

[101] Long Hu, Yuanwen Tian, Jun Yang, Tarik Taleb, Lin Xiang, and Yixue Hao. Ready Player One: UAV-Clustering-Based Multi-Task Offloading for Vehicular VR/AR Gaming. *IEEE Network*, 33(3):42–48, 2019.

[102] W Huang, T Song, Y Yang, and Y Zhang. Cluster-Based Cooperative Caching With Mobility Prediction in Vehicular Named Data Networking. *IEEE Access*, 7:23442–23458, 2019.

[103] ICN 2020 Project. ICN2020: Advancing ICN towards real-world deployment through research, innovative applications, and global scale experimentation, 2016.

[104] Van Jacobson, Diana K. Smetters, James D. Thornton, Michael F. Plass, Nicholas H. Briggs, and Rebecca L. Braynard. Networking named content. In *Proceedings of the 5th international conference on Emerging networking experiments and technologies - CoNEXT '09*, page 1, New York, New York, USA, 2009. ACM Press.

[105] Rahul Jain, Anuj Puri, and Raja Sengupta. Geographical Routing Using Partial Information for Wireless Ad Hoc Networks. *IEEE Personal Communications*, pages 48–57, 2001.

[106] Almerima Jamakovic, Markus Anwander, Torsten Braun, Peter Kropf, Eryk Schiller, Jan Schwanbeck, and Thomas Staub. A4-mesh: Connecting remote sites. *SWITCH Journal*, pages 15–17, 2012.

[107] S Jeong, O Simeone, and J Kang. Mobile Edge Computing via a UAV-Mounted Cloudlet: Optimization of Bit Allocation and Path Planning. *IEEE Transactions on Vehicular Technology*, 67(3):2049–2063, 2018.

[108] H. Jiang, Y. Guo, Z. Wu, C. Zhang, W. Jia, and Z. Wang. Implantable wireless intracranial pressure monitoring based on air pressure sensing. *IEEE Transactions on Biomedical Circuits and Systems*, 12(5):1076–1087, Oct 2018.

[109] Renhe Jiang, Xuan Song, Zipei Fan, Tianqi Xia, Quanjun Chen, Satoshi Miyazawa, and Ryosuke Shibasaki. DeepUrbanMomentum: An Online Deep-Learning System for Short-Term Urban Mobility Prediction. In *Procedings of AAAI Conference on Artificial Intelligence*, pages 784–791, 2018.

[110] Yichao Jin and Yonggang Wen. When Cloud Media Meet Network Function Virtualization: Challenges and Applications. *IEEE Multimedia*, 24(3):72–82, 2017.

[111] N Kalatzis, M Avgeris, D Dechouniotis, K Papadakis-Vlachopapadopoulos, I Roussaki, and S Papavassiliou. Edge Computing in IoT Ecosystems for UAV-Enabled Early Fire Detection. In *2018 IEEE International Conference on Smart Computing (SMARTCOMP)*, pages 106–114, 2018.

[112] Eirini Kalogeiton and Torsten Braun. Infrastructure-Assisted Communication for NDN-VANETs. *19th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, WoWMoM 2018*, 2018.

[113] Keishirou Kataoka, Takuya Yamamoto, Mai Otsuki, Fumihisa Shibata, and Asako Kimura. A new interactive haptic device for getting physical contact feeling of virtual objects. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pages 1323–1324, 2019.

[114] K V Katsaros, W K Chai, N Wang, G Pavlou, H Bontius, and M Paolone. Information-centric networking for machine-to-machine data delivery: a case study in smart grid applications. *IEEE Network*, 28(3):58–64, 2014.

[115] K Kaur, T Dhand, N Kumar, and S Zeadally. Container-as-a-Service at the Edge: Trade-off between Energy Efficiency and Service Availability at Fog Nano Data Centers. *IEEE Wireless Communications*, 24(3):48–56, 2017.

[116] Kuljeet Kaur, Sahil Garg, Gagangeet Singh Aujla, Neeraj Kumar, Joel J.P.C. Rodrigues, and Mohsen Guizani. Edge Computing in the Industrial Internet of Things Environment: Software-Defined-Networks-Based Edge-Cloud Interplay. *IEEE Communications Magazine*, 56(2):44–51, feb 2018.

[117] Jari Kellokoski, Joonas Koskinen, Tuomas Rusanen, Pasi Kalliolahti, and Timo Hämäläinen. Proxy Mobile IPv6-Based Seamless Handover. In Sergey Balandin, Sergey Andreev, and Yevgeni Koucheryavy, editors, *Internet of Things, Smart Spaces, and Next Generation Networking*, pages 214–223, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[118] Kishwer Abdul Khaliq, Omer Chughtai, Abdullah Shahwani, Amir Qayyum, and Jürgen Pannek. Road Accidents Detection, Data Collection and Data Analysis Using V2X Communication and Edge/Cloud Computing. *Electronics*, 8(8), 2019.

[119] Ammara Khan, Mehran Abolhasan, Wei Ni, Justin Lipman, and Abbas Jamalipour. A Hybrid-Fuzzy Logic Guided Genetic Algorithm (H-FLGA) Approach for Resource Optimization in 5G VANETs. *IEEE Transactions on Vehicular Technology*, 68(7):1–1, 2019.

[120] I. Khan, F. Belqasmi, R. Glitho, N. Crespi, M. Morrow, and P. Polakos. Wireless sensor network virtualization: A survey. *IEEE Communications Surveys Tutorials*, 18(1):553–576, Firstquarter 2016.

[121] Hanjin Kim, Heonyeop Shin, Hyeong Su Kim, and Won Tae Kim. VR-CPES: A novel cyber-physical education systems for interactive VR services based on a mobile platform. *Mobile Information Systems*, 2018:10, 2018.

[122] Si Jung Kim, Yunhwan Jeong, Sujin Park, Kihyun Ryu, and Gyuhwan Oh. A Survey of Drone use for Entertainment and AVR (Augmented and Virtual Reality). In Timothy Jung and M Claudia tom Dieck, editors, *Augmented Reality and Virtual Reality: Empowering Human, Place and Business*, pages 339–352. Springer International Publishing, 2018.

[123] Teemu Koponen, Kye Hyun Kim, Scott Shenker, Mohit Chawla, Byung-Gon Chun, Andrey Ermolinskiy, Kye Hyun Kim, Scott Shenker, and Ion Stoica. A Data-oriented (and Beyond) Network Architecture. *SIGCOMM Comput. Commun. Rev.*, 37(4):181–192, aug 2007.

[124] Daniel Krajzewicz. *Traffic Simulation with SUMO – Simulation of Urban Mobility*, pages 269–293. Springer New York, New York, NY, 2010.

[125] Murat Kuzlu, Manisa Pipattanasomporn, and Saifur Rahman. Communication network requirements for major smart grid applications in HAN, NAN and WAN. *Computer Networks*, 67:74–88, 2014.

[126] Abdelquoddouss Laghrissi and Tarik Taleb. A Survey on the Placement of Virtual Resources and Virtual Network Functions. *IEEE Communications Surveys & Tutorials*, 21(2):1409–1434, 2019.

[127] K. N. Lal and A. Kumar. E-health application using network coding based caching for information-centric networking (icn). In *2017 6th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, pages 427–431, Sep. 2017.

[128] Dapeng Lan, Amir Taherkordi, Frank Eliassen, Zhuang Chen, and Lei Liu. Deep reinforcement learning for intelligent migration of fog services in smart cities. In Meikang Qiu, editor, *Algorithms and Architectures for Parallel Processing*, pages 230–244, Cham, 2020. Springer International Publishing.

[129] Andres Laya, Luis Alonso, and Jesus Alonso-Zarate. Is the Random Access Channel of LTE and LTE-A Suitable for M2M Communications? A Survey of Alternatives. *IEEE Communications Surveys Tutorials*, 16(1):4–16, 2014.

[130] Nam Tuan Le, Mohammad Arif Hossain, Amirul Islam, Do-Yun Kim, Young-June Choi, and Yeong Min Jang. Survey of Promising Technologies for 5G Networks. *Mobile Information Systems*, 2016:25, 2017.

[131] Eun Kyu Lee, Mario Gerla, Giovanni Pau, Uichin Lee, and Jae Han Lim. Internet of Vehicles: From intelligent grid to autonomous cars and vehicular fogs. *International Journal of Distributed Sensor Networks*, 12(9):0–14, 2016.

[132] J Lee, J Bonnin, P Seite, and H A Chan. Distributed IP mobility management from the perspective of the IETF: motivations, requirements, approaches, comparison, and challenges. *IEEE Wireless Communications*, 20(5):159–168, 2013.

[133] Kwonjong Lee, Joonki Kim, Yosub Park, Hanho Wang, and Daesik Hong. Latency of Cellular-Based V2X: Perspectives on TTI-Proportional Latency and TTI-Independent Latency. *IEEE Access*, 5:15800–15809, 2017.

[134] K Lei, Q Zhang, J Lou, B Bai, and K Xu. Securing ICN-Based UAV Ad Hoc Networks with Blockchain. *IEEE Communications Magazine*, 57(6):26–32, 2019.

[135] Ilias Leontiadis and Cecilia Mascolo. GeOpps: Geographical opportunistic routing for vehicular networks. In *2007 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, WOWMOM*, 2007.

[136] Changle Li, Quyuan Luo, Guoqiang Mao, Min Sheng, and Jiandong Li. Vehicle-Mounted Base Station for Connected and Autonomous Vehicles: Opportunities and Challenges. *IEEE Wireless Communications*, 26(4):30–36, 2019.

[137] Gaolei Li, Jun Wu, Jianhua Li, T Ye, R Morello, Student Member, Jun Wu, and Jianhua Li. Battery Status Sensing Software-Defined Multicast for V2G Regulation in Smart Grid. *IEEE Sensors Journal*, 17(23):7838–7848, 2017.

[138] H Li, K Ota, M Dong, and M Guo. Mobile Crowdsensing in Software Defined Opportunistic Networks. *IEEE Communications Magazine*, 55(6):140–145, 2017.

[139] Jun Li, Yan Shvartzshnaider, John-austen Francisco, Richard P Martin, Kiran Nagaraj, and Dipankar Raychaudhuri. Delivering Internet-of-Things Services in MobilityFirst Future Internet Architecture. In *2012 3rd IEEE International Conference on the Internet of Things*, pages 31–38, 2012.

[140] R. Li, D. T. H. Lai, and W. Lee. A survey on biofeedback and actuation in wireless body area networks (wbans). *IEEE Reviews in Biomedical Engineering*, 10:162–173, 2017.

[141] Sugang Li, Yanyong Zhang, Dipankar Raychaudhuri, and Ravishankar Ravindran. A comparative study of MobilityFirst and NDN based ICN-IoT architectures. *Proceedings of the 2014 10th International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness, QSHINE 2014*, 1:158–163, 2014.

[142] Ting Mei Li, Chen Chi Liao, Hsin Hung Cho, Wei Che Chien, Chin Feng Lai, and Han Chieh Chao. An e-healthcare sensor network load-balancing scheme using SDN-SFC. In *2017 IEEE 19th International Conference on e-Health Networking, Applications and Services, Healthcom 2017*, volume 2017-December, pages 1–4. Institute of Electrical and Electronics Engineers Inc., dec 2017.

[143] X. Li, D. Li, J. Wan, C. Liu, and M. Imran. Adaptive transmission optimization in sdn-based industrial internet of things with edge computing. *IEEE Internet of Things Journal*, 5(3):1351–1360, June 2018.

[144] Y Li and W Gao. Minimizing Context Migration in Mobile Code Offload. *IEEE Transactions on Mobile Computing*, 16(4):1005–1018, apr 2017.

[145] Yangzhe Liao, Xinhui Qiao, Liqing Shou, Xiaojun Zhai, Qingsong Ai, Quan Yu, and Qian Liu. Caching-Aided Task Offloading Scheme for Wireless Body Area Networks with MEC. In *2019 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*, pages 49–54. IEEE, jul 2019.

[146] Wei Yang Bryan Lim, Nguyen Cong Luong, Dinh Thai Hoang, Yutao Jiao, Ying-Chang Liang, Qiang Yang, Dusit Niyato, and Chunyan Miao. Federated learning in mobile edge networks: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 22(3):2031–2063, 2020.

[147] Boxi Liu, Tao Jiang, Zehua Wang, and Yang Cao. Object-Oriented Network: A Named-Data Architecture Toward the Future Internet. *IEEE Internet of Things Journal*, 4(4):957–967, aug 2017.

[148] Dantong Liu, Lifeng Wang, Yue Chen, Maged Elkashlan, Kai Kit Wong, Robert Schober, and Lajos Hanzo. User Association in 5G Networks: A Survey and an Outlook. *IEEE Communications Surveys and Tutorials*, 18(2):1018–1044, 2016.

[149] Dapeng Liu, Juan Carlos Zúñiga, Pierrick Seite, H. Anthony Chan, and Carlos Jesus Bernardos. Distributed mobility management: Current practices and gap analysis. Technical report, Internet Engineering Task Force, 2015.

[150] F Liu, P Shu, H Jin, L Ding, J Yu, D Niu, and B Li. Gearing resource-poor mobile devices with powerful clouds: architectures, challenges, and applications. *IEEE Wireless Communications*, 20(3):14–22, 2013.

[151] Jian Liu, Jiangtao Li, Lei Zhang, Feifei Dai, Yuanfei Zhang, Xinyu Meng, and Jian Shen. Secure intelligent traffic light control using fog computing. *Future Generation Computer Systems*, 78:817–824, jan 2018.

[152] Jianhui Liu and Qi Zhang. Code-partitioning offloading schemes in mobile edge computing for augmented reality. *IEEE Access*, 7:11222–11236, 2019.

[153] Xuejie Liu, M João Nicolau, António Costa, Joaquim Macedo, and Alexandre Santos. A Geographic Opportunistic Forwarding Strategy for Vehicular Named Data Networking. In Paulo Novais, David Camacho, Cesar Analide, Amal El Fallah Seghrouchni, and Costin Badica, editors, *Intelligent Distributed Computing IX*, pages 509–521, Cham, 2016. Springer International Publishing.

[154] Yicen Liu, Hao Lu, Xi Li, Yang Zhang, Leiping Xi, and Donghao Zhao. Dynamic service function chain orchestration for nfv/mec-enabled iot networks: A deep reinforcement learning approach. *IEEE Internet of Things Journal*, 8(9):7450–7465, 2021.

[155] A Longo, A De Matteis, and M Zappatore. Urban Pollution Monitoring Based on Mobile Crowd Sensing: An Osmotic Computing Approach. In *2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC)*, pages 380–387, 2018.

[156] Quyuan Luo, Shihong Hu, Changle Li, Guanghui Li, and Weisong Shi. Resource scheduling in edge computing: A survey. *IEEE Communications Surveys and Tutorials*, 23(4):2131–2165, 2021.

[157] Lele Ma, Shanhe Yi, Nancy Carter, and Qun Li. Efficient Live Migration of Edge Services Leveraging Container Layered Storage. *IEEE Transactions on Mobile Computing*, 18(9):2020–2033, 2018.

[158] Zheng Ma, Ming Xiao, Yue Xiao, Zhibo Pang, H. Vincent Poor, and Branka Vucetic. High-Reliability and Low-Latency Wireless Communication for Internet of Things: Challenges, Fundamentals, and Enabling Technologies. *IEEE Internet of Things Journal*, 6(5):7946–7970, 2019.

[159] Filipe Manco, Costin Lupu, Florian Schmidt, Jose Mendes, Simon Kuenzer, Sumit Sati, Kenichi Yasukata, Costin Raiciu, and Felipe Huici. My vm is lighter (and safer) than your container. In *Proceedings of the 26th Symposium on Operating Systems Principles*, SOSP '17, pages 218–233, New York, NY, USA, 2017. ACM.

[160] Yuyi Mao, Changsheng You, Jun Zhang, Kaibin Huang, and Khaled B. Letaief. A Survey on Mobile Edge Computing: The Communication Perspective. *IEEE Communications Surveys and Tutorials*, 19(4):2322–2358, 2017.

[161] 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.

[162] Nuno R B Martins, Amara Angelica, Krishnan Chakravarthy, Yuriy Svidinenko, Frank J Boehm, Ioan Opris, Mikhail A Lebedev, Melanie Swan, Steven A Garan, Jeffrey V Rosenfeld, Tad Hogg, and Robert A Freitas. Human Brain/Cloud Interface. *Frontiers in Neuroscience*, 13:112, 2019.

[163] Kyoji Matsushima and Noriaki Sonobe. Full-color digitized holography for large-scale holographic 3D imaging of physical and nonphysical objects. *Appl. Opt.*, 57(1):A150—-A156, jan 2018.

[164] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. OpenFlow: Enabling Innovation in Campus Networks. *ACM SIGCOMM Computer Communication Review*, 38(2):69, 2008.

[165] Mahshid Mehrabi, Dongho You, Vincent Latzko, Hani Salah, Martin Reisslein, and Frank H.P. Fitzek. Device-Enhanced MEC: Multi-Access Edge Computing (MEC) Aided by End Device Computation and Caching: A Survey. *IEEE Access*, 7:166079–166108, 2019.

[166] Bernd Meijerink and Geert Heijenk. Infrastructure support for contention-based forwarding. In *2019 IEEE Vehicular Networking Conference (VNC)*, pages 1–8, 2019.

[167] N. Blefari Melazzi, A. Detti, G. Mazza, G. Morabito, S. Salsano, and L. Veltri. An OpenFlow-based testbed for information centric networking. In *2012 Future Network Mobile Summit (FutureNetw)*, pages 1–9. IEEE, 2012.

[168] Rodolfo I Meneguette, Diego O Rodrigues, Joahannes B D Costa, Denis Rosário, and Leandro A Villas. A Virtual Machine Migration Policy Based on Multiple Attribute Decision in Vehicular Cloud Scenario. In *Proceeddings of The IEEE International Conference on Communications*, 2019.

[169] Levente Mészáros, Andras Varga, and Michael Kirsche. *INET Framework*, pages 55–106. Springer International Publishing, Cham, 2019.

[170] Yiming Miao, Yingying Jiang, Limei Peng, M. Shamim Hossain, and Ghulam Muhammad. Telesurgery Robot Based on 5G Tactile Internet. *Mobile Networks and Applications*, 23(6):1645–1654, 2018.

[171] Sumit Kumar Monga, Sheshadri K. R, and Yogesh Simmhan. Elfstore: A resilient data storage service for federated edge and fog resources. *CoRR*, abs/1905.08932, 2019.

[172] Christopher Monsanto, Joshua Reich, Nate Foster, Jennifer Rexford, and David Walker. Composing Software-defined Networks. In *Proceedings of the 10th USENIX Conference on Networked Systems Design and Implementation*, nsdi'13, pages 1–14, Berkeley, CA, USA, 2013. USENIX Association.

[173] Marc Mosko, Ersin Uzun, and Christopher A. Wood. Mobile sessions in content-centric networks. *2017 IFIP Networking Conference, IFIP Networking 2017 and Workshops*, 2018-January(2):1–9, 2017.

[174] Robert Moskowitz, Tobias Heer, Petri Jokela, and Thomas Henderson. Host identity protocol version 2 (hipv2). Technical report, Internet Engineering Task Force, 2015.

[175] Robert Moskowitz, Pekka Nikander, Petri Jokela, and Thomas Henderson. Host identity protocol. Technical report, Internet Engineering Task Force, 2008.

[176] Ranesh Kumar Naha, Saurabh Garg, Dimitrios Georgakopoulos, Prem Prakash Jayaraman, Longxiang Gao, Yong Xiang, and Rajiv Ranjan. Fog Computing: Survey of Trends, Architectures, Requirements, and Research Directions. *IEEE Access*, 6:47980–48009, 2018.

[177] NDN Project. Named Data Networking (NDN) - A Future Internet Architecture, 2010.

[178] Dinh Nguyen, Zhishu Shen, Jiong Jin, and Atsushi Tagami. ICN-Fog: An Information-Centric Fog-to-Fog Architecture for Data Communications. In *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, pages 1–6. IEEE, dec 2017.

[179] Tuan Nguyen Gia, Amir M. Rahmani, Tomi Westerlund, Pasi Liljeberg, and Hannu Tenhunen. Fog computing approach for mobility support in internet-of-things systems. *IEEE Access*, 6:36064–36082, 2018.

[180] Zhaolong Ning, Jun Huang, and Xiaojie Wang. Vehicular Fog Computing: Enabling Real-Time Traffic Management for Smart Cities. *IEEE Wireless Communications*, 26(1):87–93, feb 2019.

[181] Jéferson Campos Nobre, Allan M. de Souza, Denis Rosário, Cristiano Both, Leandro A. Villas, Eduardo Cerqueira, Torsten Braun, and Mario Gerla. Vehicular software-defined networking and fog computing: Integration and design principles. *Ad Hoc Networks*, 82:172–181, jan 2019.

[182] Open Networking Foundation. OpenFlow Switch Specification (Version 1.5.1), 2015.

[183] Unwired Labs OpenCellID. The world's largest open database of cell towers, 2022. accessed at 2022-04-28.

[184] OpenStreetMap contributors. Planet dump retrieved from https://planet.osm.org . `https://www.openstreetmap.org`, 2017.

[185] Sharief M.A. Oteafy and Hossam S. Hassanein. Leveraging tactile internet cognizance and operation via IoT and edge technologies. *Proceedings of the IEEE*, 107(2):364–375, 2019.

[186] Tao Ouyang, Zhi Zhou, and Xu Chen. Follow Me at the Edge: Mobility-Aware Dynamic Service Placement for Mobile Edge Computing. *IEEE Journal on Selected Areas in Communications*, 36(10):2333–2345, oct 2018.

[187] Andreas Pamboris, Panayiotis Andreou, Irene Polycarpou, and George Samaras. FogFS: A Fog File System For Hyper-Responsive Mobile Applications. In *2019 16th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, pages 1–6. IEEE, jan 2019.

[188] Haixia Peng, Qiang Ye, and Xuemin Shen. SDN-Based Resource Management for Autonomous Vehicular Networks: A Multi-Access Edge Computing Approach. *IEEE Wireless Communications*, pages 1–7, 2019.

[189] Charles E Perkins and Sun Microsystems. Mobile IP. *IEEE Communications Magazine*, 35(5):84–99, 1997.

[190] Seth Pettie and Vijaya Ramachandran. An Optimal Minimum Spanning Tree Algorithm. *J. ACM*, 49(1):16–34, jan 2002.

[191] Manuel Peuster, Hannes Kuttner, and Holger Karl. Let the state follow its flows: An SDN-based flow handover protocol to support state migration. In *2018 4th IEEE Conference on Network Softwarization and Workshops, NetSoft 2018*, pages 406–414. Institute of Electrical and Electronics Engineers Inc., sep 2018.

[192] Gergely Pongracz, Laszlo Molnar, and Zoltan Lajos Kis. Removing roadblocks from SDN: Openflow software switch performance on intel DPDK. *Proceedings - 2013 2nd European Workshop on Software Defined Networks, EWSDN 2013*, pages 62–67, 2013.

[193] M Priyadarsini and P Bera. A Secure Virtual Controller for Traffic Management in SDN. *IEEE Letters of the Computer Society*, page 1, 2019.

[194] Yuanyuan Qiao, Zhongwei Si, Yanting Zhang, Fehmi Ben Abdesslem, Xinyu Zhang, and Jie Yang. A hybrid Markov-based model for human mobility prediction. *Neurocomputing*, 278:99–109, 2018.

[195] Yuqing Qiu, Chung Horng Lung, Samuel Ajila, and Pradeep Srivastava. LXC Container Migration in Cloudlets under Multipath TCP. *Proceedings - International Computer Software and Applications Conference*, 2:31–36, 2017.

[196] W. Quan, J. Guan, Z. Jiang, and H. Zhang. I-wban: Enabling information-centric data retrieval in heterogeneous wban. In *2015 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 138–143, April 2015.

[197] S Rahman, G Kim, Y Cho, and A Khan. Positioning of UAVs for throughput maximization in software-defined disaster area UAV communication networks. *Journal of Communications and Networks*, 20(5):452–463, 2018.

[198] Bharat Rao, Ashwin Goutham Gopi, and Romana Maione. The societal impact of commercial drones. *Technology in Society*, 45:83 – 90, 2016.

[199] Ravishankar Ravindran, Xuan Liu, Asit Chakraborti, Xinwen Zhang, and Guoqiang Wang. Towards software defined ICN based edge-cloud services. In *2013 IEEE 2nd International Conference on Cloud Networking (CloudNet)*, pages 227–235. IEEE, 2013.

[200] Danda B Rawat, Senior Member, and Swetha R Reddy. Software Defined Networking Architecture , Security and Energy Efficiency : A Survey. *IEEE Communications Surveys & Tutorials*, 19(1):325–346, 2017.

[201] Zeineb Rejiba, Xavier Masip-Bruin, and Eva Marín-Tordera. A survey on mobility-induced service migration in the fog, edge, and related computing paradigms. *ACM Computing Surveys*, 52(5), 2019.

[202] Gareth O Roberts. Markov chain concepts related to sampling algorithms. *Markov chain Monte Carlo in practice*, 57:45–58, 1996.

[203] Diego O. Rodrigues, Azzedine Boukerche, Thiago H. Silva, Antonio A.F. Loureiro, and Leandro A. Villas. Combining taxi and social media data to explore urban mobility issues. *Computer Communications*, oct 2018.

[204] Diego O. Rodrigues, Torsten Braun, Guilherme Maia, and Leandro Villas. Towards SDN-enabled RACH-less Make-before-break Handover in C-V2X Scenarios. In *2021 17th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pages 337–344, 2021.

[205] Diego O Rodrigues, Torsten Braun, Guilherme Maia, and Leandro Villas. Mobility-aware Software-Defined Service-Centric Networking. In *2022 International Conference on Computer Communications and Networks (ICCCN)*, pages 1–10, 2022.

[206] Diego O. Rodrigues, Torsten Braun, Guilherme Maia, and Leandro Villas. Mobility-aware Latency-constrained Data Placement in SDN-enabled Edge Networks. In *Proceeddings of the IEEE/IFIP Network Operations and Management Symposium 2023*, 2023.

[207] Diego O. Rodrigues, Allan M. de Souza, Torsten Braun, Guilherme Maia, Antonio A. F. Loureiro, and Leandro A. Villas. Service provisioning in edge-cloud continuum: Emerging applications for mobile devices. *Journal of Internet Services and Applications*, 2023.

[208] Rodrigo Roman, Javier Lopez, and Masahiro Mambo. Mobile edge computing, Fog et al.: A survey and analysis of security threats and challenges. *Future Generation Computer Systems*, 78:680–698, jan 2018.

[209] K Sadeghi, A Banerjee, J Sohankar, and S K S Gupta. Optimization of Brain Mobile Interface Applications Using IoT. In *2016 IEEE 23rd International Conference on High Performance Computing (HiPC)*, pages 32–41, 2016.

[210] Ola Salman, Imad Elhajj, Ali Chehab, and Ayman Kayssi. IoT survey: An SDN and fog computing perspective. *Computer Networks*, 143:221–246, oct 2018.

[211] Hani Sami, Azzam Mourad, Hadi Otrok, and Jamal Bentahar. Fscaler: Automatic resource scaling of containers in fog clusters using reinforcement learning. In *2020 International Wireless Communications and Mobile Computing (IWCMC)*, pages 1824–1829, 2020.

[212] Eric Samikwa, Antonio Di Maio, and Torsten Braun. Ares: Adaptive resource-aware split learning for internet of things. *Computer Networks*, 218:109380, 2022.

[213] G. E. Santagati and T. Melodia. Experimental evaluation of impulsive ultrasonic intra-body communications for implantable biomedical devices. *IEEE Transactions on Mobile Computing*, 16(2):367–380, Feb 2017.

[214] Surbhi Saraswat, Vishal Agarwal, Hari Prabhat Gupta, Rahul Mishra, and Ashish Gupta. Journal of Network and Computer Applications Challenges and solutions in Software Defined Networking : A survey. *Journal of Network and Computer Applications*, 141(April):23–58, 2019.

[215] Mahadev Satyanarayanan. The emergence of edge computing. *Computer*, 50(1):30–39, 2017.

[216] Bertrand Schneider and Paulo Blikstein. Tangible User Interfaces and Contrasting Cases as a Preparation for Future Learning. *Journal of Science Education and Technology*, 27(4):369–384, aug 2018.

[217] Michael Schneider, Jason Rambach, and Didier Stricker. Augmented reality based on edge computing using the example of remote live support. In *Proceedings of the IEEE International Conference on Industrial Technology*, pages 1277–1282. IEEE, 2017.

[218] Philipp Schulz, Maximilian Matthe, Henrik Klessig, Meryem Simsek, Gerhard Fettweis, Junaid Ansari, Shehzad Ali Ashraf, Bjoern Almeroth, Jens Voigt, Ines Riedel, Andre Puschmann, Andreas Mitschele-Thiel, Michael Muller, Thomas Elste, and Marcus Windisch. Latency Critical IoT Applications in 5G: Perspective on the Design of Radio Interface and Network Architecture. *IEEE Communications Magazine*, 55(2):70–78, 2017.

[219] Hichem Sedjelmaci, Mohamed Ayoub Messous, Sidi Mohammed Senouci, and Imane Horiya Brahmi. Toward a lightweight and efficient UAV-aided VANET. *Transactions on Emerging Telecommunications Technologies*, 30(8):e3520, 2019.

[220] Hafez Seliem, Reza Shahidi, Mohamed Hossam Ahmed, and Mohamed S. Shehata. Drone-based highway-VANET and das service. *IEEE Access*, 6:20125–20137, 2018.

[221] Syed Sarmad Shah, Muhammad Ali, Asad Waqar Malik, Muazzam A. Khan, and Sri Devi Ravana. VFog: A Vehicle-Assisted Computing Framework for Delay-Sensitive Applications in Smart Cities. *IEEE Access*, 7:34900–34909, 2019.

[222] DongJian Shen, WeiKang Chen, Peng Liu, and ZhiJun Wang. Nb-iot-based environmental monitoring system for peach orchard. In *2022 3rd International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE)*, pages 1–4, 2022.

[223] Jinming Shi, Jun Du, Jian Wang, and Jian Yuan. Distributed v2v computation offloading based on dynamic pricing using deep reinforcement learning. In *2020 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1–6, 2020.

[224] Weisong Shi, Jie Cao, Quan Zhang, Youhuizi Li, and Lanyu Xu. Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, 3(5):637–646, 2016.

[225] P Simoens, D Griffin, E Maini, T K Phan, M Rio, L Vermoesen, F Vandeputte, F Schamel, and D Burstzynowski. Service-Centric Networking for Distributed Heterogeneous Clouds. *IEEE Communications Magazine*, 55(7):208–215, 2017.

[226] Christoph Sommer, David Eckhoff, Alexander Brummer, Dominik S. Buse, Florian Hagenauer, Stefan Joerer, and Michele Segata. *Veins: The Open Source Vehicular Network Simulation Framework*, pages 215–252. Springer International Publishing, Cham, 2019.

[227] Y Su, Z Cai, K Wu, L Shi, F Zhou, H Chen, and J Wu. Projection-Type Multiview Holographic Three-Dimensional Display Using a Single Spatial Light Modulator and a Directional Diffractive Device. *IEEE Photonics Journal*, 10(5):1–12, 2018.

[228] Zhou Su, Yilong Hui, and Tom H Luan. Distributed Task Allocation to Enable Collaborative Autonomous Driving With Network Softwarization. *IEEE Journal on Selected Areas in Communications*, 36(10):2175–2189, oct 2018.

[229] Yaohua Sun, Mugen Peng, Yangcheng Zhou, Yuzhe Huang, and Shiwen Mao. Application of Machine Learning in Wireless Networks: Key Techniques and Open Issues. *IEEE Communications Surveys & Tutorials*, 21(4):1–1, 2019.

[230] S Takeda, D Iwai, and K Sato. Inter-reflection Compensation of Immersive Projection Display by Spatio-Temporal Screen Reflectance Modulation. *IEEE Transactions on Visualization and Computer Graphics*, 22(4):1424–1431, apr 2016.

[231] Tarik Taleb, Adlen Ksentini, and Pantelis A Frangoudis. Follow-Me Cloud : When Cloud Services Follow Mobile Users. *IEEE Transactions on Cloud Computing*, 7(2):369–382, 2019.

[232] T. Tanaka, S. Eum, S. Ata, and M. Murata. Design and implementation of tracking system for moving objects in information-centric networking. In *2019 22nd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)*, pages 302–306, Feb 2019.

[233] Xing Tang, Junwei Zhou, Shengwu Xiong, Jing Wang, and Kunxiao Zhou. Geographic Segmented Opportunistic Routing in Cognitive Radio Ad Hoc Networks Using Network Coding. *IEEE Access*, 6:62766–62783, 2018.

[234] Z Tang, X Zhou, F Zhang, W Jia, and W Zhao. Migration Modeling and Learning Algorithms for Containers in Fog Computing. *IEEE Transactions on Services Computing*, page 1, 2018.

[235] Zhiqing Tang, Xiaojie Zhou, Fuming Zhang, Weijia Jia, and Wei Zhao. Migration Modeling and Learning Algorithms for Containers in Fog Computing. *IEEE Transactions on Services Computing*, 14(8), 2018.

[236] Muhammad Tayyab, Xavier Gelabert, and Riku Jantti. A Survey on Handover Management: From LTE to NR. *IEEE Access*, 7(1):118907–118930, 2019.

[237] M. K. Thakkar, L. Agrawal, A. K. Rangisetti, and B. R. Tamma. Reducing ping-pong handovers in lte by using a1-based measurements. In *2017 Twenty-third National Conference on Communications (NCC)*, pages 1–6, March 2017.

[238] Kasun M.S. Thotahewa, Jamil Y. Khan, and Mehmet R. Yuce. Power efficient ultra wide band based wireless body area networks with narrowband feedback path. *IEEE Transactions on Mobile Computing*, 13(8):1829–1842, 2014.

[239] S Tiennoy and C Saivichit. Using a Distributed Roadside Unit for the Data Dissemination Protocol in VANET With the Named Data Architecture. *IEEE Access*, 6:32612–32623, 2018.

[240] Andrea Tomatis, Hamid Menouar, and Karsten Roscher. *Forwarding in VANETs: GeoNetworking*, pages 221–251. Springer International Publishing, Cham, 2015.

[241] Mauro Tortonesi, Marco Govoni, Alessandro Morelli, Giulio Riberto, Cesare Stefanelli, and Niranjan Suri. Taming the IoT data deluge: An innovative information-centric service model for fog computing applications. *Future Generation Computer Systems*, 93:888–902, apr 2019.

[242] Tuyen X. Tran and Dario Pompili. Joint task offloading and resource allocation for multi-server mobile-edge computing networks. *IEEE Transactions on Vehicular Technology*, 68(1):856–868, 2019.

[243] Christian Tschudin and Manolis Sifalakis. Named functions and cached computations. In *2014 IEEE 11th Consumer Communications and Networking Conference (CCNC)*, pages 851–857. IEEE, 2014.

[244] Sandesh Uppoor, Oscar Trullols-Cruces, Marco Fiore, and Jose M. Barcelo-Ordinas. Generation and analysis of a large-scale urban vehicular mobility dataset. *IEEE Transactions on Mobile Computing*, 13(5):1061–1075, 2014.

[245] Rahul Urgaonkar, Shiqiang Wang, Ting He, Murtaza Zafer, Kevin Chan, and Kin K. Leung. Dynamic service migration and workload scheduling in edge-clouds. *Perform. Eval.*, 91(C):205–228, sep 2015.

[246] Andras Varga. *OMNeT++*, pages 35–59. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[247] Luca Veltri, Giacomo Morabito, Stefano Salsano, Nicola Blefari-Melazzi, and Andrea Detti. Supporting information-centric functionality in software defined networks. In *2012 IEEE International Conference on Communications (ICC)*, pages 6645–6650, 2012.

[248] Arun Venkataramani, James Kurose, Dipankar Raychaudhuri, Kiran Nagaraja, Morley Mao, and Suman Banerjee. MobilityFirst: a mobility-centric and trustworthy internet architecture. *ACM SIGCOMM Computer Communication Review*, 44(3):74–80, 2014.

[249] D K N Venkatramana, S B Srikantaiah, and J Moodabidri. SCGRP: SDN-enabled connectivity-aware geographical routing protocol of VANETs for urban environment. *IET Networks*, 6(5):102–111, 2017.

[250] R Vilalta, I Popescu, A Mayoral, X Cao, R Casellas, N Yoshikane, R Martínez, T Tsuritani, I Morita, and R Muñoz. End-to-end SDN/NFV orchestration of video analytics using edge and cloud computing over programmable optical networks. In *2017 Optical Fiber Communications Conference and Exhibition (OFC)*, pages 1–3, 2017.

[251] Antonio Virdis, Giovanni Stea, and Giovanni Nardini. SimuLTE - A modular system-level simulator for LTE/LTE-A networks based on OMNeT++. In *2014 4th International Conference On Simulation And Modeling Methodologies, Technologies And Applications (SIMULTECH)*, pages 59–70, 2014.

[252] Vodafone. White paper: Multi-access edge computing. Technical report, Vodafone GmbH, 2021.

[253] Ziyu Wan, Zheng Zhang, Rui Yin, and Guanding Yu. Kfiml: Kubernetes-based fog computing iot platform for online machine learning. *IEEE Internet of Things Journal*, 9(19):19463–19476, 2022.

[254] Jiadai Wang, Jiajia Liu, and Nei Kato. Networking and Communications in Autonomous Driving: A Survey. *IEEE Communications Surveys & Tutorials*, 21(2):1243–1274, 2019.

[255] Shangguang Wang, Senior Member, Jinliang Xu, Ning Zhang, Senior Member, and Yujiong Liu. A Survey on Service Migration in Mobile Edge Computing. *IEEE Access*, 6:23511–23528, 2018.

[256] Xiaojie Wang, Zhaolong Ning, Xiping Hu, Edith C Ngai, Lei Wang, Bin Hu, and Ricky Y K Kwok. ENABLING TECHNOLOGIES FOR SMART INTERNET OF THINGS A City-Wide Real-Time Traffic Management System : Enabling Crowdsensing in Social Internet of Vehicles. *IEEE Communications Magazine*, 56(September):19–25, 2018.

[257] Xuyu Wang, Shiwen Mao, and Michelle X. Gong. An overview of 3gpp cellular vehicle-to-everything standards. *GetMobile: Mobile Comp. and Comm.*, 21(3):19–25, nov 2017.

[258] Sage A Weil, Scott A Brandt, Ethan L Miller, Darrell D E Long, and Carlos Maltzahn. Ceph: A Scalable, High-Performance Distributed File System. In Brian N Bershad and Jeffrey C Mogul, editors, *7th Symposium on Operating Systems Design and Implementation (OSDI '06), November 6-8, Seattle, WA, USA*, pages 307–320. USENIX Association, 2006.

[259] Aaron Weiss. Computing in the clouds. *NetWorker*, 11(4):16–25, dec 2007.

[260] Zhenyu Wen, Michael Rovatsos, Renyu Yang, Jie Xu, Peter Garraghan, and Tao Lin. Fog Orchestration for Internet of Things Services. *IEEE Internet Computing*, 21(2):16–24, 2017.

[261] Wireless One. Pokémon GO first to DT "Edge network in the core", 2018.

[262] Changyou Xing, Ke Ding, Chao Hu, Ming Chen, and Bo Xu. SD-ICN: Toward wide area deployable software defined information centric networking. *KSII Transactions on Internet and Information Systems*, 10(5):2267–2285, 2016.

[263] Changyou Xing, Ke Ding, Chao Hu, Ming Chen, and Bo Xu. SD-ICN: Toward wide area deployable software defined information centric networking. *KSII Transactions on Internet and Information Systems*, 10(5):2267–2285, 2016.

[264] F Xiong, A Li, H Wang, and L Tang. An SDN-MQTT Based Communication System for Battlefield UAV Swarms. *IEEE Communications Magazine*, 57(8):41–47, aug 2019.

[265] Youcef Yahiatene, Abderrezak Rachedi, Mohamed Amine Riahla, Djamel Eddine Menacer, and Farid Nait-Abdesselam. A blockchain-based framework to secure vehicular social networks. *Transactions on Emerging Telecommunications Technologies*, 30(8):e3650, 2019.

[266] Lei Yang, Jiannong Cao, Guanqing Liang, and Xu Han. Cost Aware Service Placement and Load Dispatching in Mobile Cloud Systems. *IEEE Transactions on Computers*, 65(5):1440–1452, 2016.

[267] S Yang, F Li, M Shen, X Chen, X Fu, and Y Wang. Cloudlet Placement and Task Allocation in Mobile Edge Computing. *IEEE Internet of Things Journal*, 6(3):5853–5863, 2019.

[268] Haipeng Yao, Chao Qiu, Chao Fang, Xu Chen, and F. Richard Yu. A Novel Framework of Data-Driven Networking. *IEEE Access*, 4:9066–9072, 2016.

[269] A. Yazdinejad, R. M. Parizi, A. Dehghantanha, and K. R. Choo. Blockchain-enabled authentication handover with efficient privacy protection in sdn-based 5g networks. *IEEE Transactions on Network Science and Engineering*, pages 1–1, 2019.

[270] Hao Ye, Geoffrey Ye Li, and Biing-Hwang Fred Juang. Deep reinforcement learning based resource allocation for v2v communications. *IEEE Transactions on Vehicular Technology*, 68(4):3163–3173, 2019.

[271] Dongho You, Tung V. Doan, Roberto Torre, Mahshid Mehrabi, Alexander Kropp, Vu Nguyen, Hani Salah, Giang T. Nguyen, and Frank H. P. Fitzek. Fog Computing as an Enabler for Immersive Media: Service Scenarios and Research Opportunities. *IEEE Access*, 7:65797—-65810, 2019.

[272] T You, M Umair, and Y Hong. Mobile crowd sensing based on CICN. In *2017 International Conference on Information and Communication Technology Convergence (ICTC)*, pages 1272–1275, 2017.

[273] Quan Yu, Jing Ren, Yinjin Fu, Ying Li, and Wei Zhang. Cybertwin: An origin of next generation network architecture. *IEEE Wireless Communications*, 26(6):111–117, 2019.

[274] Zhenhui Yuan, Jie Jin, Lingling Sun, Kwan Wu Chin, and Gabriel Miro Muntean. Ultra-Reliable IoT Communications with UAVs: A Swarm Use Case. *IEEE Communications Magazine*, 56(12):90–96, 2018.

[275] Zainab Zaidi, Senior Member, Vasilis Friderikos, Zarrar Yousaf, Simon Fletcher, Mischa Dohler, and Hamid Aghvami. Will SDN Be Part of 5G ? *IEEE Communications Surveys & Tutorials*, 20(4):3220–3258, 2018.

[276] Yong Zeng, Rui Zhang, and Teng Joon Lim. Wireless communications with unmanned aerial vehicles: Opportunities and challenges. *IEEE Communications Magazine*, 54(5):36–42, 2016.

[277] Cheng Zhang and Zixuan Zheng. Task migration for mobile edge computing using deep reinforcement learning. *Future Generation Computer Systems*, 96:111–118, 2019.

[278] Fei Zhang, Guangming Liu, Xiaoming Fu, and Ramin Yahyapour. A Survey on Virtual Machine Migration: Challenges, Techniques, and Open Issues. *IEEE Communications Surveys and Tutorials*, 20(2):1206–1243, 2018.

[279] Feixiong Zhang, Kiran Nagaraja, Yanyong Zhang, and Dipankar Raychaudhuri. Content Delivery in the MobilityFirst Future Internet Architecture. *2012 35th IEEE Sarnoff Symposium*, pages 1–5, 2012.

[280] H Zhang, W Quan, H Chao, and C Qiao. Smart identifier network: A collaborative architecture for the future internet. *IEEE Network*, 30(3):46–51, 2016.

[281] Lixia Zhang, Alexander Afanasyev, Jeffrey Burke, Van Jacobson, kc Claffy, Patrick Crowley, Christos Papadopoulos, Lan Wang, and Beichuan Zhang. Named data networking. *ACM SIGCOMM Computer Communication Review*, 44(3):66–73, jul 2014.

[282] Qing-Yi Zhang, Xing-Wei Wang, Min Huang, Ke-Qin Li, and Sajal K. Das. Software Defined Networking Meets Information Centric Networking: A Survey. *IEEE Access*, 6:39547–39563, 2018.

[283] Shuhao Zhang, Yingjun Wu, Feng Zhang, and Bingsheng He. Towards concurrent stateful stream processing on multicore processors. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)*, pages 1537–1548, 2020.

[284] X Zhang, H Wang, and H Zhao. An SDN framework for UAV backbone network towards knowledge centric networking. In *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 456–461, apr 2018.

[285] Y. Zhang and N. Ansari. On protocol-independent data redundancy elimination. *IEEE Communications Surveys Tutorials*, 16(1):455–472, First 2014.

[286] Y. Zhang, R. Deng, E. Bertino, and D. Zheng. Robust and universal seamless handover authentication in 5g hetnets. *IEEE Transactions on Dependable and Secure Computing*, pages 1–1, 2019.

[287] Chengcheng Zhao, Mianxiong Dong, Kaoru Ota, Jianhua Li, and Jun Wu. Edge-MapReduce-Based Intelligent Information-Centric IoV: Cognitive Route Planning. *IEEE Access*, 7:50549–50560, 2019.

[288] Zhongliang Zhao, Pedro Cumino, Arnaldo Souza, Denis Rosário, Torsten Braun, Eduardo Cerqueira, Mario Gerla, Denis do Rosário, Torsten Braun, Eduardo Cerqueira, and Mario Gerla. Software-defined unmanned aerial vehicles networking for video dissemination services. *Ad Hoc Networks*, 83:68–77, 2019.

[289] Kan Zheng, Qiang Zheng, Periklis Chatzimisios, Wei Xiang, and Yiqing Zhou. Heterogeneous Vehicular Networking: A Survey on Architecture, Challenges, and Solutions. *IEEE Communications Surveys and Tutorials*, 17(4):2377–2396, 2015.

[290] Hangyu Zhu, Jinjin Xu, Shiqing Liu, and Yaochu Jin. Federated learning on non-iid data: A survey. *Neurocomputing*, 465:371–390, 2021.

[291] M. Zink, R. Sitaraman, and K. Nahrstedt. Scalable 360 video stream delivery: Challenges, solutions, and opportunities. *Proceedings of the IEEE*, 107(4):639–650, April 2019.

[292] G. K. Zipf. Human behavior and the principle of least effort. cambridge, (mass.): Addison-wesley, 1949, pp. 573. *Journal of Clinical Psychology*, 6(3):306–306, 1950.