Infrastructure-less D2D Communications through Opportunistic Networks

by

Noelia Pérez Palma

in partial fulfillment of the requirements for the degree of Doctor in

Telematic Engineering

Universidad Carlos III de Madrid

Advisor: Marco Ajmone Marsan
Co-Advisor: Vincenzo Mancuso

November 2021

Infrastructure-less D2D Communications
through Opportunistic Networks

Prepared by:
Noelia Pérez Palma, IMDEA Networks Institute, Universidad Carlos III de Madrid
contact: noelia.perez@imdea.org

Under the advice of:
Marco Ajmone Marsan, Politecnico di Torino, IMDEA Networks Institute
Vincenzo Mancuso, IMDEA Networks Institute
Telematic Engineering Department, Universidad Carlos III de Madrid

This work has been supported by:

institute IMDEA networks

‘It ain’t what you don’t know that gets you into trouble. It’s what you know for sure that just ain’t so.”

Mark Twain

Acknowledgements

First of all I want to thank my supervisors Vincenzo Mancuso and Marco Ajmone for giving me the opportunity to work with them, dedicating their time and patience for the benefit of my learning. My growth during these years has not only been professional but also personal thanks mainly to their effort. Special mention to Gianluca Rizzo, without his collaboration all most of my achievements wouldn't have been possible.

I want to especially thank my family from whom I have always had not only the support but also the drive to pursue my goals. To my mother, from whom I learned perseverance, hard work and resolution. To my father, who instilled in me a devotion to the study of science and a sense of justice and honesty. To Ginés, my life partner, who has humbly taught me to see life from other perspectives and has lovingly accompanied me through the hardest stages of these years.

Of course, I want to thank all my more than colleagues, friends at IMDEA, during the PhD for teaching me so much and especially for the time we have shared. IMDEA has been a wonderful place to go every morning knowing that they were going to be there, we have become a little family, supporting each other, that will last forever despite impediments. Moha my great technical and moral support, with his helpful smile to cheer on Mondays. Edgar always ready to solve my more than stupid questions about math. Dario, my eternal coffee mate trying to convince me that Italy is the best country in the world. Marta, always listening to my stories and encouraging me in my purposes. A heartfelt thanks to Hany, my little confidant, who has been giving me tips for research survival and gossip since the beginning of my PhD. The Italian gang at IMDEA, with whom I shared most of my breaks before the pandemic hit. Thank you also for sharing so many good moments and laughs (and hate) to Roberto, Ander, Pablo, Guillermo, Norbert, Alejandro, Joan, Dolo, Nina, Constantine, Julien, Alvaro, Segun, Pelayo and Yago. I think I will never find another group with such sense of humor. I also want to thank Professor Sergey Gorinsky, for placing his trust in me at event collaborations and giving me the opportunity to learn from his experience. Alejandro Delgado and David, the best IMDEA concierges, with whom I have been fortunate to work for several years and who have given me so much serenity, I will miss the evenings of confessions and complaints in front of the door so much. And, of course, the former students who were with me at the beginning of times when everything was unknown and terrifying. You have become great researchers and I hope to follow your example.

Special thanks to CCS Labs with whom I stayed during my internship in Paderborn, Germany. Thanks to Falko for giving me a warm welcome in his group and dedicating time and resources to my research, always involved, willing to help and, best of all, always with an endearing attitude. Of course, thanks to all the guys in the group: Pannu, Agon, Sigrid, Jorge, Max, Julian, Fabian, Dominik, Johannes, Sohaib, Florian, Christoph and Tobi. So attentive, they made my stay not only professionally fruitful but they were incredibly hospitable and kind to me.

I do not want to end without thanking Professor Antonio Fernández Anta. From his hand I got to know the world of research when I did my TFG at IMDEA back in 2014. He conveyed to me how exciting research can become when you invest love and effort into it. That motivation and the good times working with him left a spark in me that inevitably made me come back later to pursue my PhD and brought me to this point in life.

Published and Submitted Content

This thesis is based on the following published and submitted papers:

[**1**] **Noelia Pérez Palma**, Vincenzo Mancuso, Marco Ajmone Marsan. Infrastructureless pervasive information sharing with cots devices and software. Published in *2018 IEEE 19th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM)*, 12-15 June 2018, Chania, Greece. https://doi.org/10.1109/WoWMoM.2018.8449733

- This work is partially included and its content is reported in Chapter 2.
- The author's role in this work is focused on the design, implementation and experimentation with regarding of the concepts proposed in the paper, as well as co-leader role in the writing of the full paper.

[**2**] **Noelia Pérez Palma**, Vincenzo Mancuso, Falko Dressler. Precise: Predictive Content Dissemination Schemes Exploiting Realistic Mobility Patterns. Published in *2021 Elsevier Computer Networks (COMNET).*

- This work is fully included and its content is reported in Chapter 3.
- The author's role in this work is focused on the design, implementation and experimentation of the schemes presented in the paper, as well as leader role in the writing of the full paper.

[**3**] Gianluca Rizzo, **Noelia Pérez Palma**, Marco Ajmone Marsan, Vincenzo Mancuso. A Walk Down Memory Lane: On Storage Capacity in Opportunistic Content Sharing Systems. Published in *2020 IEEE 21th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM)*, 31-3 September 2020, Virtual Conference. https://doi.org/10.1109/WoWMoM49955.2020.00022

- This work is fully included and its content is reported in Chapter 4.
- The author's role in this work is focused on the design, implementation and experimentation of the validation system presented in the paper, as well as as well as being involved in the development of the analytical model and co-leader role in the writing of the paper.

[4] Gianluca Rizzo, **Noelia Pérez Palma**, Marco Ajmone Marsan, Vincenzo Mancuso. Storage Capacity of Opportunistic Information Dissemination Systems. Published in *2021 IEEE Transactions on Mobile Computing.* https://doi.org/10.1109/TMC.2021.3057259

- This work is fully included and its content is reported in Chapter 4.
- The author's role in this work is focused on the design, implementation and experimentation of the validation system presented in the paper, as well as being involved in the development of the analytical model and co-leader role in the writing of the paper.

[5] **Noelia Pérez Palma**. On the persistence of wireless advertising without infrastructure support. Published as the author's master's thesis in *2017.* http://hdl.handle.net/20.500.12761/444

- This work is partially included and its content is reported in Chapter 2.
- The author's role in this work is focused on the design, implementation and experimentation with regarding of the concepts proposed in this master's thesis, as well as leader role in the writing of the full master's thesis.

Abstract

In recent years, we have experienced several social media blackouts, which have shown how much our daily experiences depend on high-quality communication services. Blackouts have occurred because of technical problems, natural disasters, hacker attacks or even due to deliberate censorship actions undertaken by governments. In all cases, the spontaneous reaction of people consisted in finding alternative channels and media so as to reach out to their contacts and partake their experiences. Thus, it has clearly emerged that infrastructured networks—and cellular networks in particular—are well engineered and have been extremely successful so far, although other paradigms should be explored to connect people. The most promising of today's alternative paradigms is Device-to-Device (D2D) because it allows for building networks almost freely, and because 5G standards are (for the first time) seriously addressing the possibility of using D2D communications.

In this dissertation I look at opportunistic D2D networking, possibly operating in an infrastructure-less environment, and I investigate several schemes through modeling and simulation, deriving metrics that characterize their performance. In particular, I consider variations of the Floating Content (FC) paradigm, that was previously proposed in the technical literature.

Using FC, it is possible to probabilistically store information over a given restricted local area of interest, by opportunistically spreading it to mobile users while in the area. In more detail, a piece of information which is injected in the area by delivering it to one or more of the mobile users, is opportunistically exchanged among mobile users whenever they come in proximity of one another, progressively reaching most (ideally all) users in the area and thus making the information dwell in the area of interest, like in a sort of distributed storage.

While previous works on FC almost exclusively concentrated on the communication component, in this dissertation I look at the storage and computing components of FC, as well as its capability of transferring information from one area of interest to another.

I first present background work, including a brief review of my Master Thesis activity, devoted to the design, implementation and validation of a smartphone opportunistic information sharing application. The goal of the app was to collect experimental data

that permitted a detailed analysis of the occurring events, and a careful assessment of the performance of opportunistic information sharing services. Through experiments, I showed that many key assumptions commonly adopted in analytical and simulation works do not hold with current technologies. I also showed that the high density of devices and the enforcement of long transmission ranges for links at the edge might counter-intuitively impair performance.

The insight obtained during my Master Thesis work was extremely useful to devise smart operating procedures for the opportunistic D2D communications considered in this dissertation. In the core of this dissertation, initially I propose and study a set of schemes to explore and combine different information dissemination paradigms along with real users mobility and predictions focused on the smart diffusion of content over disjoint areas of interest. To analyze the viability of such schemes, I have implemented a Python simulator to evaluate the average availability and lifetime of a piece of information, as well as storage usage and network utilization metrics. Comparing the performance of these predictive schemes with state-of-the-art approaches, results demonstrate the need for smart usage of communication opportunities and storage. The proposed algorithms allow for an important reduction in network activity by decreasing the number of data exchanges by up to 92%, requiring the use of up to 50% less of on-device storage, while guaranteeing the dissemination of information with performance similar to legacy epidemic dissemination protocols.

In a second step, I have worked on the analysis of the storage capacity of probabilistic distributed storage systems, developing a simple yet powerful information theoretical analysis based on a mean field model of opportunistic information exchange. I have also extended the previous simulator to compare the numerical results generated by the analytical model to the predictions of realistic simulations under different setups, showing in this way the accuracy of the analytical approach, and characterizing the properties of the system storage capacity.

I conclude from analysis and simulated results that when the density of contents seeded in a floating system is larger than the maximum amount which can be sustained by the system in steady state, the mean content availability decreases, and the stored information saturates due to the effects of resource contention. With the presence of static nodes, in a system with infinite host memory and at the mean field limit, there is no upper bound to the amount of injected contents which a floating system can sustain. However, as with no static nodes, by increasing the injected information, the amount of stored information eventually reaches a saturation value which corresponds to the injected information at which the mean amount of time spent exchanging content during a contact is equal to the mean duration of a contact.

As a final step of my dissertation, I have also explored by simulation the computing and learning capabilities of an infrastructure-less opportunistic communication, storage and computing system, considering an environment that hosts a distributed Machine Learning (ML) paradigm that uses observations collected in the area over which the FC system operates to infer properties of the area. Results show that the ML system can operate in two regimes, depending on the load of the FC scheme. At low FC load, the ML system in each node operates on observations collected by all users and opportunistically shared among nodes. At high FC load, especially when the data to be opportunistically exchanged becomes too large to be transmitted during the average contact time between nodes, the ML system can only exploit the observations endogenous to each user, which are much less numerous. As a result, I conclude that such setups are adequate to support general instances of distributed ML algorithms with continuous learning, only under the condition of low to medium loads of the FC system. While the load of the FC system induces a sort of phase transition on the ML system performance, the effect of computing load is more progressive. When the computing capacity is not sufficient to train all observations, some will be skipped, and performance progressively declines.

In summary, with respect to traditional studies of the FC opportunistic information diffusion paradigm, which only look at the communication component over one area of interest, I have considered three types of extensions by looking at the performance of FC:

- over several disjoint areas of interest;
- in terms of information storage capacity;
- in terms of computing capacity that supports distributed learning.

The three topics are treated respectively in Chapters 3 to 5.

Table of Contents

List of Tables

List of Figures

List of Acronyms

3GPP 3rd Generation Partnership Project

ANN Artificial Neural Network

AP Access Point

API Application Programming Interface

AR Autoregressive

BS Base Station

COTS Commercial-Off-The-Shelf

CTMC Continuous Time Markov Chain

D2D Device-to-Device

DFS Distributed Floating System

DHCP Dynamic Host Configuration Protocol

DSRC Dedicated Short-Range Communications

EWM Emergency Warning Message

FC Floating Content

FE Floating Element

FG Floating Gossip

FL Federated Learning

GL Gossip Learning

GO Group Owner

ICN Information-centric networking

ISM Industrial, Scientific and Medical

LFS Localized Floating System

LTE Long Term Evolution

MBB Mobile Broadband

MC Mobile Crowdsensing

ML Machine Learning

MS Mobile Sensing

NDN Named Data Networking

NR New Radio

ODE Ordinary Differential Equation

OppNet Opportunistic Network

P2P Peer-to-Peer

PAN Personal Area Network

PCTMC Population Continuous Time Markov Chain

PIS Proximity-Interest-Social

PoI Point of Interest

PPP Poisson Point Process

PRECISE Predictive Content Dissemination Scheme

ProSe Proximity Service

RD Random Direction

RDMM Random Direction Mobility Model

RSU Road-Side Unit

RWP Random Waypoint

RZ Replication Zone

SARP	social Acquaintance based Routing Protocol
SCH	Social Circle Heuristic
SIG	Special Interest Group
SIR	Susceptible-Infected-Recovered
SSID	Service Set Identifier
SUMO	Simulation of Urban Mobility
TDS	Trust based Dissemination Scheme
TTL	Time To Live
UAV	Unmanned Aerial Vehicle
UE	User Equipment
UHF	Ultra High Frequency
UPnP	Universal Plug and Play
URLLC	Ultra-Reliable Low-Latency Communication
V2V	Vehicle-to-Vehicle
V2X	V2X Vehicle-to-Everything
VANET	Vehicular Ad hoc Network

VR	Virtual Reality
VSN	Vehicular Social Network
WPS	Wi-Fi Protected Setup
ZOI	Zone of Interest

1 Introduction

Our society is experiencing a massive growth in the number of active devices connected over the Internet generating vast amounts of data. For many applications, there is need to offload the communication from cellular networks to direct Device-to-Device (D2D) communications [5] through Opportunistic Networks (OppNets) [6]. This is useful, e.g., when users are experiencing poor network connection or they are unable to connect, or when the available network infrastructure cannot be trusted. For the above mentioned cases, and for the ones in which it results to be more cost-effective than infrastructure-based access, the use of D2D is favorable [7].

For instance, consider that the number of users concerned about their privacy keeps steadily growing, so that many services will rather not trust intermediary parties to distribute—and be in possess of—certain information. They would rather trust “friend” devices (e.g., devices owned by people belonging to the same community or a social network) than network infrastructures and service operators. Examples of this case range from context-aware social networking to covert communications during protests.

Networking of mobile users through opportunistic D2D communications has recently received considerable attention from the research community [1], [5], [7], although technology to support such infrastructure-less scenario initially was barely available. In recent years, with the standardization of the 5G New Radio (NR) Sidelink [8], in addition to the LTE Sidelink [9], to Wi-Fi Direct [10], and to Bluetooth 5.2 [11], a number of options now exist for wireless D2D and opportunistic mobile networks have become a viable possibility. This is likely to encourage the development of service creation approaches that are not bound to the availability of a cellular infrastructure. Their application domains, traditionally including scenarios in which infrastructure is not available, such as disaster areas or battlefields [12], have now spread to pandemic-driven warnings and spontaneous protests. Indeed, several countries have recently invested in the development of proximity-based applications based on D2D communications to help contact tracing, thus stimulating technology development in this area. In addition, smartphone apps like

FireChat [1] and Bridgefy [2], exploit D2D communications to enable information exchanges either when Internet access is unavailable, or when localized distribution is desired, or where infrastructure-based communications are not trusted. For instance, FireChat was the communication medium of choice in several civil protests.

1.1. FC in Disjoint Geographical Areas

During the last decade, a multitude of content dissemination techniques have been developed [13]–[16]. Survey papers such as [17]–[20] provide deep insights into the different perspectives adopted. Nevertheless, heterogeneous and limited resources and capabilities at nodes still impose many limitations for real-world scenarios. Most importantly, the dynamically evolving network topology still determines one of the main challenges.

The presence of memory-constrained devices and network congestion are some of the causes of these limitations. The scheme proposed in this work aims to introduce a significant reduction in the amount of data kept in mobile devices' memory, along with a drastic alleviation of network traffic. We achieve this goal by cutting data exchanges down to only meaningful ones. Furthermore, we aim at exploiting contact opportunities leveraging nodes movement pattern based on typical daily routines, which leads to accurate predictions of the network users' future behavior. We present a set of configurations that make use of the previously gathered information to manage network and device resources efficiently and, therefore, to deliver messages in a more effective manner with respect to legacy content dissemination schemes.

The social component is also a great asset to boost forwarding strategies [21]–[23]. Similar to Pannu et al. [24], both in the algorithmic design as well as in our evaluation, we support certain interest areas (we call these hotspots or Replication Zones (RZs)) towards which pedestrians and vehicles are more likely to head.

Focusing on disjoint geographical areas (hotspots) leads us to use realistic mobility patterns and to be accurate in the management of data dissemination. In our example in Chapter 3, population samples relate to two often visited hotspots, but our approach can be used with any number of hotspots. Furthermore, we claim that the presence of a social component in the forwarding scheme justifies the use of D2D.

Our service quality metrics are average delivery delay and content "availability". The former is the average time needed for a mobile node to receive a piece of dissemination content after moving into a hotspot. The content availability is the probability that a piece of content be stored on mobile nodes in the hotspot and so be available for D2D dissemination to newly arrived users. As an overall objective, we want to achieve delivery delay and availability levels as if we were using epidemic diffusion schemes, except we

[1]https://apps.apple.com/us/app/firechat/id719829352
[2]https://bridgefy.me/

want to reduce the overhead in terms of number of connections and use of storage on mobile devices.

Note that, differently from epidemic routing schemes, any user in the hotspot is the destination of any content to be disseminated. In this sense, heterogeneous and limited resources and capabilities of the involved nodes impose additional limitations for real-world applications. Note also that existing opportunistic schemes cannot capture the social-aware nature of the applications considered in this work. They can be used if need be, but, as shown in Chapter 3, they end up wasting precious resources to disseminate information beyond the needs of the applications, with no tangible performance gain. For this purpose, we have designed an application-dependent scheme that will serve information to users with same interests in an independent fashion and which take into account the specificity and predictability of mobility patterns by learning from past events. For example, nodes involved in a university environment will subscribe to the same specific channel and, consequently, share only related event advertisements. This detached approach, compared to state-of-the-art works where everyone's devices are involved in the content distribution process, is crucial to avoid misusing resources from nodes that are not willing to cooperate to the routing process as well as spamming users with different interests.

In order to assess the performance of our Predictive Content Dissemination Scheme (PRECISE), we have implemented a simulation model, analyzed the collected data describing occurring events, and assessed and compared the performance of PRECISE to existing dissemination strategies. Our results provide essential insights on how to manage available resources in an efficient manner according to the studied scenarios and mobility requirements.

The main contributions of this chapter can be summarized as follows:

- We design powerful yet lightweight scheme, named PRECISE, whose algorithms improve data forwarding and storage efficiency in opportunistic communication scenarios.

- We implement the proposed forwarding and storing scheme in a custom-made simulator, which uses state-of-the-art approaches for mobility modeling based on maps and real user mobility.

- We assess the performance of PRECISE in terms of content availability, storage load, network resource utilization, and content lifetime, delivery delay and losses.

- We also compare our results with three other benchmarks, such as, content dissemination restricted to RZs, Epidemic, and Proximity-Interest-Social (PIS) routing protocol. We show that our algorithm clearly outperforms other solutions by reducing between 65-92% the needed number of connections while producing comparable content availability values.

1.2. FC Storage Capacity

Key questions about the usefulness of infrastructure-less communications are about how fast and reliable they are, and how much information they can store. While several studies have shown that localized infrastructure-less content dissemination schemes such as Floating Content (FC) [13], [25], [26], Locus [27] or Hovering information [28], [29] can be effective in dissemination contexts, little has been done so far to quantify their storage capacity, which is the focus of this work.

In Chapter 4, we keep relying on FC, which aims at disseminating information over a defined geographic area (called RZ), based solely on direct D2D connectivity [30]. By so doing, FC stores information spatially in a probabilistic fashion, despite the mobility of User Equipments (UEs) and the unreliability of information exchanges, and with no need for centralized servers. Thus, FC can deliver the stored content proactively to users which are expected to traverse a specific region (Zone of Interest (ZOI)), before they reach it. Hence, the main performance metric in such systems is the *success ratio*, i.e., the average fraction of nodes that enter the ZOI with content.

Clearly, guaranteeing (probabilistically) content persistence and a given target performance in such a volatile setting, without the support of a centralized static infrastructure, comes at a cost. The main additional cost as compared to traditional centralized infrastructure-based solutions (e.g., with respect to commonly studied distributed information storage schemes in which mobile UEs cache popular content items [31]–[34]), is in a drastic increase both in content redundancy across the user population, and in the volume of communications required to reach the target population of users.

A strong point of FC is that, by enabling direct D2D content transfer and sharing without routing through a Access Points (APs) or Base Stations (BSs), it offers a parsimonious approach to distributed edge storage because it can achieve higher energy efficiency while decreasing the utilization of BS resources.

A major open issue for the practical viability of FC as a distributed edge storage system is the characterization of its scalability, i.e., of the amount of information that FC is able to store for a given set of system parameters. This is the problem we address in Chapter 4.

We propose a simple analytical model of the storage capacity of probabilistic distributed edge storage systems such as FC, based on a mean field model of the opportunistic information exchange, which allows for a first order characterization of the scaling laws of the storage capacity of these systems. Specifically, the contributions are as follows:

- We develop an analytical model of FC performance, based on a mean field model of the dynamics of the population of users storing the information items, and of the population of users which are in the process of exchanging (sending or receiving) such items;
- We derive analytical expressions of the FC storage capacity, as a function of node mobility and of the geometry of the replication zone;
- We formulate an optimization problem for the derivation of the maximum amount of information which can be stored with FC, showing that it can be solved efficiently. To the best of our knowledge, this is the first work to characterize analytically the storage capacity of probabilistic distributed storage schemes such as FC;
- We evaluate numerically our results, validating our assumptions against simulation, under different mobility models, showing the accuracy of our mean field approach, and characterizing the properties of FC storage capacity as a function of the main system parameters.
- Leveraging the insights provided by our model on the FC behavior as an opportunistic information storage system, we derive an extremely simple and intuitive closed form approximate expression for FC storage capacity.

1.3. FC Computational Capacity

Finally, services that require the acquisition of information about a specific context (termed observations) and the subsequent elaboration in each UE of a model on which it is possible to elaborate a strategy that allows the optimization of an utility function while performing a given task, are natural candidates for opportunistic D2D information exchanges, and computation and storage on board individual UEs.

In the last chapter of this thesis, we study the effectiveness of D2D-based services for decision making. The main goal is to determine what is the maximum amount of information that a group of users can learn without the support of an infrastructure, only exploiting the computation capacity of their UEs combined with D2D communications.

The problem tackled is within the domains of crowdsensing and Machine Learning (ML), but, to the best of our knowledge, no previous work has tried to characterize the intrinsic limits to ML in a fully distributed and opportunistic environment like ours.

Several works have previously looked at components of the system we consider. For example, a number of works on FC have characterized the opportunistic D2D aspect, as mentioned in the sections above. Other works on Gossip Learning (GL) have tackled the ML aspects. These previous works will be briefly reported in Chapter 2.

The main contributions of this chapter are the following:

- We introduce a novel distributed computing and communication scheme rooted in FC and GL, named Floating Gossip (FG).

- We characterize the performance of the FG approach taking into consideration the following metrics: model availability, observation availability and learning capacity of the system.

- We provide definitions and performance evaluation for FG to identify the general limits and trade-offs of infrastructure-less communication and computing systems used to implement ML in a cooperative way.

- We extend the features of the simulator used during this thesis to analyze, through detailed simulation experiments, the computing aspects typical of ML operations in this novel distributed FG environment.

1.4. Thesis Structure

The thesis is organized as follows. In Chapter 2, I introduce the thesis background and discuss some of the previous literature in the field. In Chapter 3, I present PRECISE, a system to explore and combine the D2D paradigm along with real mobility and predictions focused on the dissemination of content among disjoint geographical areas. In Chapter 4, I elaborate a model to characterize the storage capacity of FC systems, and compare the results to the predictions of realistic simulations under different setups. Chapter 5 focuses on the computational capacity of the previously presented system, introducing a novel proposal called FG, which consists in a distributed computing and communication scheme rooted in FC and GL. Chapter 6 concludes the thesis. In the Appendix I include the proofs of lemmas from Chapter 4 and finally, as closing, I report the development and technical documentation of the simulator implemented to support this dissertation.

2 Background and Related Work

Information sharing is becoming a relevant issue for Mobile Broadband (MBB) operators, due to the increasing popularity of social networks, to the increasing volumes of shared information, and to the steady increase in the number and capabilities of mobile devices connected to the Internet. Offloading information sharing services from the cellular infrastructure to Device-to-Device (D2D) opportunistic communications can offer a welcome reduction of the traffic on MBB networks.

2.1. Background on Floating Content in Infrastructure-less Opportunistic Communications

The work reported as background of this dissertation concerns an experimental study of the performance of information sharing services that do not require infrastructure support from MBB operators, while aiming to pervasively spread messages among groups of devices that subscribe to one or more classes of information (e.g., commercial advertisements, local news, traffic warnings, etc.). Since infrastructure-less pervasive mechanisms strongly depend and rely on the mobility of devices, this work shows if and how state-of-the-art technologies like Wi-Fi Direct, which is available in Commercial-Off-The-Shelf (COTS) devices, can be used for—and impose limits to—information sharing services accessed by real users.

Formerly to this thesis, experiments with a smartphone information sharing application that can be used on COTS devices, with no need to root the device's software were discussed in [1]. In order to avoid unrealistic assumptions on the behavior of D2D communications, the work included and built upon the implementation of an Android application that supports infrastructure-less distributed content sharing among wireless devices using Wi-Fi Direct. The collected experimental data permitted a detailed analysis of the occurring events, and a careful assessment of the performance of pervasive information sharing services. The experiments revealed that many assumptions commonly used in the literature do not hold in real settings with state-of-the-art technologies.

This work's contribution supported the idea that the pervasive use of devices along with the human mobility will lead to a wide range of opportunities to create opportunistic networks, and thus benefit information dissemination and acquisition. However, we showed that using WLAN-like coverage for D2D, although devices reach much shorter distances than advertised, can have a detrimental impact on information dissemination speeds, depending on the user density. In general, technological limitations are still huge. Specifically, the contribution of the work was fourfold:

- Assess the feasibility of infrastructure-less pervasive data sharing networks to provide connection establishment between Wi-Fi Direct devices.
- Discuss the limitations imposed by Wi-Fi Direct and by COTS devices and operating systems.
- Illustrate the difference between real technology constraints and ideal assumptions often used in modeling works.
- Understand the key mechanisms that affect the performance of the class of services under evaluation.

We concluded that delay tolerant services can be supported, albeit we also showed that the high density of devices and the enforcement of long transmission ranges for D2D links might counter-intuitively impair performance.

2.1.1. Wi-Fi Direct in COTS Devices

Wi-Fi Direct is a technology defined by the Wi-Fi Alliance to allow direct D2D communications. Wi-Fi Direct supports typical Wi-Fi speeds (up to 250 Mbit/s) and has the same nominal range (up to 200 m). Featuring better time and throughput characteristics than other D2D protocols, such as Bluetooth, Wi-Fi Direct has become the reference standard to support D2D communications, allowing group formation of one-to-one and one-to-many devices [35]. Indeed, it is available in the majority of today's COTS communications devices.

In the next subsections we describe the operation of Wi-Fi Direct and its limitations when using a legacy Android OS.

2.1.1.1. Wi-Fi Direct Operation

First of all, when Wi-Fi Direct devices are willing to form a group, they alternately listen and send probe requests with additional Peer-to-Peer (P2P) information elements on the "social channels" 1, 6 and 11 in the 2.4 GHz Industrial, Scientific and Medical (ISM) band. At this point, available P2P devices receiving a probe request reply with probe response frames including P2P information elements describing the group characteristics.

An optional feature during this process is the so called service discovery. Once a peer device has been discovered, it can be asked to describe the services it provides. Thus, the device asking for a service may choose to connect to the discovered peer based on whether it provides the required service. This is carried out by a higher layer protocol, e.g. Universal Plug and Play (UPnP) or Bonjour [36].

After the device discovery phase, P2P devices can start forming groups. There are three different ways in which devices can form a group, described in what follows.

2.1.1.2. Standard

This is the main procedure to form a group. In a P2P group any P2P device can take the role of P2P Group Owner (GO), which acts as an access point, or P2P Client which associates to the GO. So, this procedure implements a negotiation that determines which of the peers becomes the GO. The procedure also sets the operating channel, the Wi-Fi Protected Setup (WPS) configuration method, and whether the group is a *persistent* group (see below).

2.1.1.3. Persistent

If a standard negotiation sets a group as persistent, the session information will be stored by the participating devices and reused in future connection. Therefore, if a device wants to establish a group with an already known peer, they will simply skip the negotiation phase.

2.1.1.4. Autonomous

In the autonomous group formation type, a device decides to create a P2P group by its own in which it will take the role of GO.

After any type of group is formed, the GO will start announcing the group by means of beaconing with the negotiated (or autonomously chosen) Service Set Identifier (SSID). In this way, other P2P devices can discover the group and request to join.

Finally, GO and group members exchange credentials using the WPS protocol to support a secure connection. Usually, WPS requires user interaction to enter a PIN code or push a button on the device to allow the connection to be instantiated.

During this procedure the GO assumes the role of authenticator.

Additionally, the GO conducts a Dynamic Host Configuration Protocol (DHCP) exchange to assign IPv4 addresses. Finally, the connection between the devices is established and secured, and the data exchange can take place. The entire process is depicted in the upper part of Fig. 2.1.

It is worth noting that re-instantiating a persistent group is profitable, because devices can automatically and quickly re-connect when required, avoiding the manual interaction

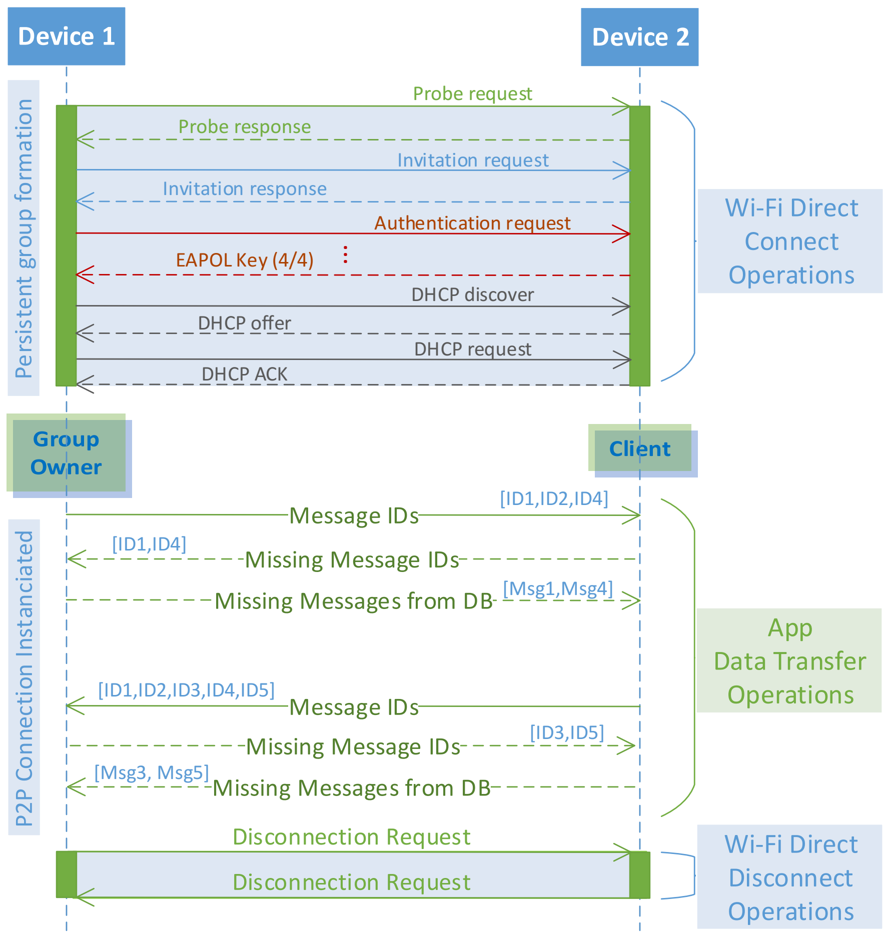


Figure 2.1: Service workflow.

of the user for authentication.

2.1.2. Wi-Fi Direct in Android

The Android OS complying with the Wi-Fi Alliance's Wi-Fi Direct certification program has developed a Wi-Fi Direct Application Programming Interface (API) for mobile apps, also know as Wi-Fi P2P [37], and which is available in Android 4.0 and more recent OS versions.

The API implemented by Android developers is a tool that enables programmers to manage the Wi-Fi Direct functionalities provided by the smartphones manufacturers. Therefore, mobile apps that share data among users, such as multi-player games or content sharing apps, can be built on top of communication services using the Wi-Fi P2P API or directly use the API.

The Wi-Fi P2P API provides developers with methods to discover, connect to, and disconnect from peers. It also includes interfaces called *event listeners* that use callbacks to report on events (e.g., connection requests received, connection drops, etc.) and outcome of operations (e.g., the result of a connection attempt or a data transfer).

The current Android Wi-Fi P2P API presents a number of limitations when applied to different scenarios, mainly because the Wi-Fi Direct protocol was designed to fulfill specific requirements and meet strong security constraints. For instance, at least since the roll-out of Android 5.0, it is not possible—not even by rooting a device—to avoid user authentication with user interaction, hence it is not possible to establish new connections automatically. Similarly, unless the device is rooted, it is neither possible to have devices acting as a GO and client at the same time in different groups, nor participating to multiple groups at all [38].

2.1.3. An App for Pervasive Information Sharing

We aimed at creating a pervasive information sharing framework in which devices carry and exchange messages without the help of a network infrastructure. The relevance of messages is limited in space and time, i.e., information has to be spread within a limited region and before a deadline. These are parameters to be selected according to the nature of the information carried by the devices. Here we limit our study to the service *template*, rather than exploring the performance of specific services that could be built on top of such template. The typical operational scenario is depicted in Fig. 2.2, in which devices running the information sharing app generate messages to be delivered to any other device running the same app within a delimited region called “anchor zone”.

In a sense, our work uses similar techniques as proposed in [38] and [39], although, differently from those works, we do not rely on already set topologies and architectures, and do not limit the scope of our work to static deployments. We built instead a pure dynamic opportunistic network [40], [41].

To achieve our goal, we used Wi-Fi Direct on COTS Android devices, without using root user’s privileges to run any software. This choice incurs in the limitations described in the previous section. However, this choice also makes our work realistic for the deployment of real services, with the real limitations and security constraints of existing and commercial operating systems. Thus, we had to deal with these limitations and find *legal* workarounds to implement an infrastructure-less pervasive information sharing system that requires no user interaction and can be installed by all Android users with no need of rooting their devices.

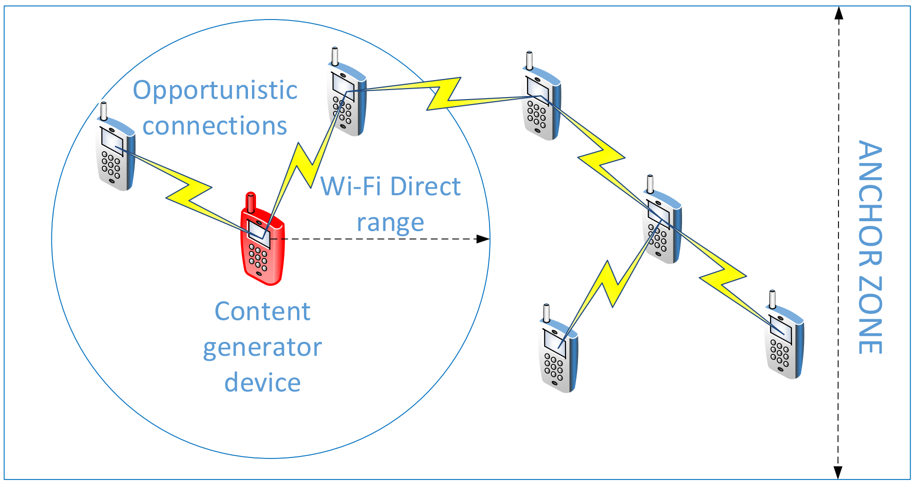


Figure 2.2: Infrastructure-less pervasive information sharing.

2.1.3.1. Design of App and Information Sharing System

We have used the *geofence* mechanism to delimit the anchor zone [42]. Geofencing is based on GPS and defines the perimeter of a geographical region, and it is available on Android. In addition, we set up a Time To Live (TTL) for every message generated, which is carried by the message itself.

Given the limitations of Wi-Fi Direct in Android, and the fact that mobility makes groups unstable, we decided to enforce one-to-one links only, with a GO and a client in each group. Groups are formed using the standard procedure the first time a pair of devices try to establish a connection. Afterwards, they use the persistent procedure.

Our app operates in an *epidemic* manner [41] trying to connect to as many other devices as possible and *infect* them by forwarding all the messages it stores locally (either self-generated or previously received). Information sharing is done in loops, following the states shown in the state machine of Fig. 2.3. We describe next the main characteristic of every state and how the transitions take place.

When a device enters the marked geofence it starts scanning the social channels looking for potential peer devices. This is the `Scanning` state in the state machine of Fig. 2.3. While the scanning is running, any other device can send an invitation request to the first device, so the latter moves to the `Connected` state directly. A second option is to keep scanning during the time period set and connect to one of the peers discovered when the scanning time expires. At this point, the device will send an invitation request to the chosen peer device, which will imply going through the `Connecting` state and then to `Connected` state when the peer accepts the invitation. There is also the possibility that no peers are discovered by the device so the scanning phase will begin again. When in

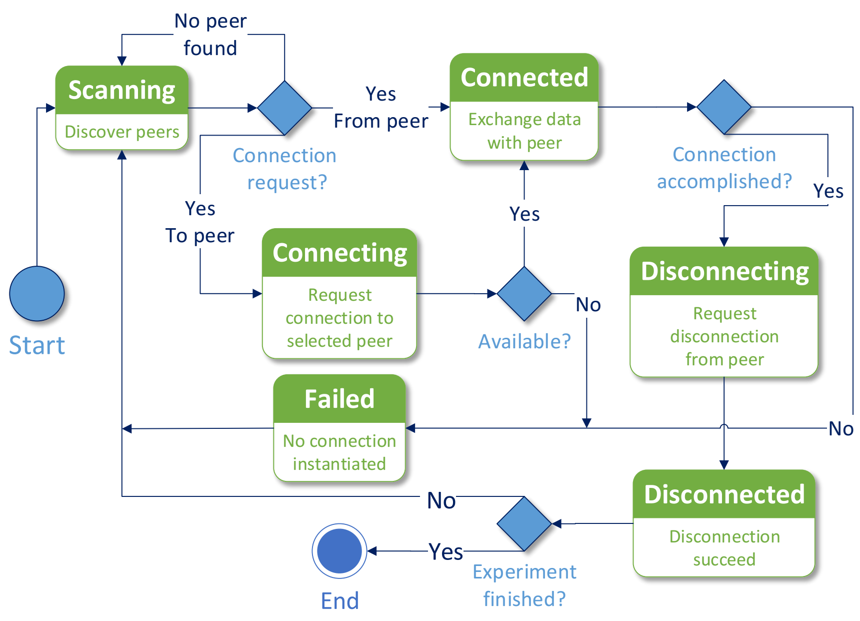


Figure 2.3: State Machine of our app.

`Connected` state, paired devices exchange data with each other, and, right after, both of them enter the `Disconnecting` state. Next state, `Disconnected`, will take place in the exact moment that the app is notified with a disconnection callback triggered from the previous `Disconnecting` state.

Due to the nature of human mobility, frequent link disruptions can happen during the connection attempt. In the first place, two devices can have established a connection and, before the data transfer has finished, the connection can be dropped. On the other hand we can find other types of failures before establishing a connection, that is, during the connecting phase. If the local system is busy, it will fail in the connection attempt. If the peer device is busy for a long period of time, it will never accept the invitation request, so that the device attempting to connect will fail after a timeout. In both cases there will be a transition to the `Failed` state.

Both `Disconnected` and `Failed` states will eventually lead back to the `Scanning` state to repeat the process steadily. In this way, there will be no possibility of getting stuck in any state. Thanks to this aggressive operation mode the app will get to a situation where it has to share content with a peer as fast as possible or, in case of failure, try with another recipient, resulting in the creation of an opportunistic and dynamic pervasive information sharing system.

2.1.4. Bluetooth and Sidelink

As previously mentioned, since very demanding applications started pervading cellular networks, D2D communication paradigm emerged as a novel technique for offloading traffic from the core network, increase spectral efficiency and reduce the energy and the cost per bit [43].

Networks have always been hierarchical in nature. Devices have connected to and communicated with one or more base stations ever since the birth of cellular communications. For that reason, operators did not initially perceive D2D communication as a viable alternative to cellular networks. However, the rise of context-aware and location discovery services twisted this archaic perspective [44]. In the past decade new types of cellular services that go beyond traditional mobile broadband have had a strong impact on the scoping and development of the 5G NR standard. These new cellular services were motivated by the business and economic needs of making the 3GPP ecosystem capable of supporting industrial requirements ranging from direct automotive communication between vehicles to industrial automation with Ultra-Reliable Low-Latency Communication (URLLC) for mission-and business-critical applications. But these same technologies can also be used for consumers to enhance their communication experience.

In fact, not only the discussed protocol Wi-Fi Direct can be used to enable D2D communication, but other short-range wireless technologies like Bluetooth and Long Term Evolution (LTE) Sidelink were also devised. They differ mostly in the data rates, distance between direct devices, device discovery mechanisms and scanning modes. As a comparison, Bluetooth 5 supports a maximum data rate of 50 Mbit/s and a range close to 240 m, WiFi Direct allows up to 250 Mbit/s rate and 200 m range while LTE Sidelink provides rates up to 13.5 Mbit/s and a range of 500 m [45]. With this new communication paradigm, cellular devices are able to communicate without relaying their data via the network, allowing cars, robots and even consumer gadgets to create their own ad hoc networks without using the radio access network as an intermediary.

More in detail, Bluetooth, managed by the Bluetooth Special Interest Group (SIG), is a short-range wireless technology standard used for exchanging data between fixed and mobile devices over short distances using Ultra High Frequency (UHF) radio waves in the ISM bands, from 2.402 GHz to 2.48 GHz, and building Personal Area Networks (PANs). It is mainly used as an alternative to wire connections, to exchange files between nearby portable devices and connect smart phones and music players with wireless headphones. On the bright side, most smart phones nowadays provide a Bluetooth interface. However, it is primarily designed for low-power consumption and in the most widely used mode, transmission power is limited to 2.5 mW, giving it a very short range of up to 10 m which makes it unsuitable for services that require high data rates, such as Floating Content (FC) services.

The latter, Sidelink (defined by the 3rd Generation Partnership Project (3GPP) [46]), is an LTE feature first introduced in 3GPP Release 12 and later enriched in Releases 13 and 14. It aims at enabling D2D communications within legacy cellular-based LTE radio access networks. D2D is applicable to public safety and commercial communication use-cases, and recently (Rel.14) to V2X (V2X) scenarios. In legacy uplink/downlink, two User Equipments (UEs) communicate through the Uu interface and data are always traversing the LTE eNB. Differently, Sidelink enables the direct communication between proximal UEs using the newly defined PC5 interface, and data does not need to go through the eNB, so the device gains more control of how to use network resources. It allows devices to discover and communicate with one another at extremely high data rates and low latency, making them ideal for peer-to-peer gaming and streaming services as well as enhanced Virtual Reality (VR) and other wearable device communications. Services provided in this way are often called Proximity Services (ProSes) and the UEs supporting this feature ProSe-enabled UEs.

2.1.5. Lessons learnt

The results we obtained compare somehow negatively with those in [47], which reported that with an app using Bluetooth on 12 smartphones, in a research institute, almost 90% of the devices are reached within a few minutes, although messages often do not reach all devices within a full working day. Our results are better than—yet practically close to—those obtained in [48], with an app using Bluetooth in a university campus with ∼50 to ∼70 mobile devices, where it took more than 4 hours to distribute a message to 70% of the devices. While message transfer times are comparable with the two technologies, in our experiments the wider coverage of Wi-Fi did not help, because of the huge fraction of failed connection attempts due to the large number of devices discovered within a scanning phase. A key parameter for the performance of the information sharing app seems to be the average number of devices within transmission range, which should be ideally close to 1, as indicated by previous studies about the intrinsic performance of opportunistic communications [14], [49].

The analysis of WhatsApp carried out in [50] shows that multimedia flows have a typical size of a few hundreds of kilobytes, and that the throughput experienced by users is of the order of 1 Mb/s in both uplink and downlink, thus resulting in transfer times of the order of a few seconds.

Therefore, the times we have observed for D2D communications once a contact occurs are comparable with the ones of infrastructure-based communications. However, we remark that infrastructure-less services have several advantages: they incur no monetary costs, occupy zero commercial network resources, do not suffer for the presence of network and server bottlenecks, and are available under undesirable circumstances like natural disasters, hacker attacks and censorship attempts. On the other hand, the use of a networked messaging app would relieve the terminal from the discovery of peers and the establishment of connections, which, as we have seen, is prone to multiple collisions due to the fact that a device can only join one group under current Android limitations. This effect exacerbates when the terminal density grows.

Indeed, as previously remarked, in our work we have found that Wi-Fi Direct on Android suffers some serious limitations. From the viewpoint of an infrastructure-less service designer, the most critical limitation consists in the fact that a device cannot be part of multiple groups. This makes connection management a hard task when users move and groups have to be continuously teared down and re-established as soon as the GO leaves. This is one of the main reasons why we have used one-to-one groups in our work, thus limiting the management complexity and connection re-establishment overheads.

2.2. Related Work

2.2.1. Information dissemination heuristics

For more than a decade now, forwarding strategies for D2D communications in opportunistic networks have represented one of the most challenging questions to cope with in terms of content dissemination performance, due to high node mobility, dynamic evolution of networks and devices heterogeneity.

In the third chapter of this dissertation, we mainly focus on providing enhanced heuristics and combinations of those for data sharing among devices contributing to resource usage efficiency and dissemination effectiveness. Devices involved in such scenarios are mostly carried by users who are considered to influence, to some extent, the behavior of their smartphones, tablets, etc. According to this perspective, previous works on content dissemination in opportunistic networks can be categorized into four main groups, as discussed in what follows.

2.2.1.1. Context-oblivious heuristics

Early logical and elementary techniques for content distribution can be classified into *context-oblivious heuristics*. For example, works like Grossglauser and Tse [14] and Spyropoulos et al. [15] proposed different schemes that constrain the number of content copies in the system to improve bandwidth, storage capacity, and energy consumption. Beyond the previous concept and given the fact that dropping messages too early may reduce the speed of information diffusion, Hernández-Orallo et al. [51] introduce a dynamic expiration time setting to limit the effects of early content loss. With these techniques the authors try to overcome the shortcomings of basic flooding-based schemes but still pose limitations when mobility patterns are restricted. Chancay-García et al. [52] study the impact of contact duration for message broadcasting. They leverage the division of large messages into smaller parts to improve dissemination and demonstrate that a fixed size partition is the best approach. Our scheme does not include context-oblivious mechanisms since we believe introducing context and social-aware heuristics better adapt to the mobility dynamics of most urban scenarios, as explained in what follows and demonstrated in Chapter 3.

2.2.1.2. Context-aware heuristics

Obviously, there was still a need for more sophisticated methods to solve dissemination challenges in frequently disconnected networks, not only aiming at reducing flooding and overhead but also effectively distributing data content, i.e., providing valuable content to potential nodes at acceptable time delay. For that purpose, an advanced sort of *context-aware heuristics* to achieve smarter decision making processes has been explored. For instance, Dhurandher et al. [53] present a history-based routing protocol that exploits nodes mobility information to predict the best next hop for content exchange. Lindgren et al. [54] and Barrett et al. [55] combine history of previous encounters with probabilistic techniques. In both papers, nodes decide to which peer they will forward the content assessing various parameters to compute the probability that the chosen node will deliver the content to its destination. Furthermore, Burns et al. [56] incorporate information not only about past encounters but also about previous visited regions.

More recently, research studies have coupled several of the cited features to develop more accurate techniques for specific D2D communications scenarios. For example, Liu et al. [57] introduce a distributed online algorithm that focuses on the optimal node pause strategy in order to select the best transmission peer. Yamamoto et al. [58] propose a method that adaptively adjusts the transmission timing and effective radius of the area in which information is shared. This decision is based on terminal density and terminal encounter rate in order to estimate further communication opportunities. In Rizzo et al. [3], the authors present an information theoretical model of the storage capacity of probabilistic distributed storage systems where nodes are only allowed to exchange content based on their current position and storage capacity.

Some other works refer to this D2D paradigm with the *Floaty Content* term. For example, Pérez Palma et al. [1] and Rizzo et al. [59] go further and develop Android applications that support infrastructureless distributed content sharing among wireless devices using state-of-the-art technologies, such as Bluetooth and Wi-Fi Direct. The authors also discuss results gathered from real experiments and conclude that high device densities determine the performance.

What is missing in all these studies is the social factor, which we instead leverage to increase the efficiency of dissemination schemes. In Chapter 3, we assume that nodes move according to similar patterns every weekday following social behaviors like going to their work place, returning home or to some other frequently visited places. This allows the application to predict with high accuracy whether passing information to a user is going to be useful or not, which reduces unnecessary information exchanges typical of epidemic schemes.

2.2.1.3. Social-aware heuristics

Several studies emerged using *social-aware heuristics*. Researchers started developing dissemination strategies initially based on the idea that human mobility presents certain behavioral patterns that can benefit forwarding decision making.

A good example has been introduced by Boldrini et al. [60]. They present ContentPlace, a system that defines social-oriented policies and analyzes the behavior of users in pursuance of optimizing content availability by locating data content in appropriate spots. Boldrini et al. [21] also exploit a combination of social information to pick the most suitable next hop based on the similarity of each peer node context to the destination context. Ying et al. [61] introduce a Markov chain model of users' social ties. They formulate the problems of unfair traffic distribution and unfair delivery success ratio based on the evaluation of users' social relationship. Rahim et al. [62] present a social Acquaintance based Routing Protocol (SARP) for Vehicular Social Networks (VSNs). SARP considers the global and local community acquaintance of nodes to make a forwarding decision. Moreover, Ullah et al. [63] developed a reputation mechanism that calculates a trust-score for each node based on its social-utility behavior and contribution to the network. Built on that idea, the authors propose a Trust based Dissemination Scheme (TDS) for Emergency Warning Messages (EWMs) to detect malicious alarms. Hui et al. [22] analyze the contact patterns between nodes and infer the social communities which these nodes belong to. This system aims to exchange data to nodes belonging to the destination community based on previous context information and assuming sociable nodes will have more chances to forward the content to its destination. Vegni et al. [64] assess a previously introduced probabilistic-based broadcasting scheme for vehicular communications leveraging the computation of nodes' social degree. They demonstrate its effectiveness in packet transmission reduction while guaranteeing network dissemination in realistic scenarios with real traffic traces. They also compare it with state-of-the-art schemes showing a significant improvement in terms of delivery ratio.

A very relevant work in this field is also introduced by Xia et al. [65], where authors propose Proximity-Interest-Social (PIS) a routing protocol based on three different social factors, and disclosing next slots social information, in order to decide the best next hop for content sharing. They present their results applying the proposed approach to SIGCOMM09 [66] and INFOCOM06 [67] data sets. The results show that PIS outperforms other well-known protocols such as Epidemic [16], PROPHET [54] and SimBet [68]. Same authors developed a similar approach in [69] that integrates vehicles' social factors into their geographical information. They introduce a new concept called geo-social distance and combine it, among other processes, with the message copy control protocol used in PIS.

Our work in Chapter 3 is partially shared with this category. However, unlike previous mentioned works, we use only information from past traces in order to predict future positions of the nodes and make decisions according to it. Furthermore, we also take into account for how long nodes remain in their positions based on typical social standards like 7-8 hours work day, 7-9 hours sleep, etc. Filling the gap of previous approaches, we consider a set of nodes to be the final destination of the data content instead of targeting for an individual. We assume that the pieces of information shared using our paradigm will be relevant for the whole portion of the population subscribed to a given communication channel. We are able to significantly reduce network load by leveraging nodes mobility predictions and light computation for decision making, contrary to existing social-based data dissemination approaches, which still fail to achieve efficient data broadcasting due to high volumes of overhead and redundant connections.

Part of our work is also devoted to real scenarios, for instance Rome city center. We have worked with real taxi cabs traces obtained from [70] and applied Predictive Content Dissemination Scheme (PRECISE) to carry out content dissemination. Additionally, we have implemented the previous mentioned PIS approach to compare with our solution over a more realistic scenario, closer to what can be found in urban areas.

2.2.1.4. Cognitive heuristics

Cognitive heuristics conform to a whole new set of forwarding algorithms to which researchers are paying great attention now. The central concept behind cognitive science applied to forwarding protocols is to build algorithms based on human information processing schemes.

In this direction, Mordacchini et al. [71] introduce Social Circle Heuristic (SCH). Their proposal is to evaluate not only the importance of the content according to the individual but also take into account the judgement of its community. In a similar way, Khelifi et al. [72] focus on vehicular networks from the Information-centric networking (ICN) perspective and discuss the role of Named Data Networking (NDN) providing a detailed and systematic review of NDN-driven Vehicular Ad hoc Network (VANET).

Our perspective focuses on how valuable a piece of content is for the population at a given time. For example, if a node is traveling to a hotspot where the content is relevant, after a too long travel period, it will be considered as outdated.

2.2.2. Storage capacity

In a subsequent line of research, the analysis of the previously mentioned FC paradigm is a contribution to a more general research effort aimed at optimizing the utilization of resources (bandwidth and user storage) in gossip-based information diffusion paradigms. Since the epidemic spreading of information, as mentioned above, may easily saturate network resources (e.g., as a result of broadcast storms), a large array of techniques and approaches has been proposed (see, e.g., [18], [73] and references therein) in order to obtain an efficient use of resources in content diffusion processes.

These techniques mainly depend on the specific goal of the content diffusion process. In closed systems, in which nodes cannot leave a given area, the aim of gossip-based schemes is to achieve completeness, i.e., to deliver content to all users in the given area, in the most resource-efficient way [74].

The present thesis however, like the majority of realistic applications, focuses on scenarios in which nodes may join and leave [75] the area. These applications adopt different approaches to identify the set of users to which a given content should be delivered. An example is given by schemes for popularity-based P2P opportunistic content replication, e.g., for cooperative in-network content caching [76]–[80], where content must be delivered to a subset of nodes: those that requested such content.

In applications in which the delivery delay plays a key role, (e.g., delivery of data related to unexpected events, or to potential safety hazards), the target population coincides with (a large fraction of) all nodes present in the given area at the time in which the hazardous or unexpected event takes place [18], [73].

In all such systems, control over content availability and persistence is achieved by involving in the scheme also nodes which are not among the set of requesters. Thus, a key performance trade-off is between the amount of involved nodes (and thus resource utilization) on the one side, and content availability and likelihood of content persistence on the other. Several techniques for limiting content replication and/or the amount of involved nodes have been proposed, based on, e.g., a maximum lifetime or hop count of the content, among others [81].

In this respect, the specificity of FC schemes lies in associating content with a given spatial context, and in controlling the amount of nodes involved in the scheme by means of location-based criteria, such as the definition of the Replication Zone (RZ) borders.

Given the infrastructure-less and probabilistic nature of such distributed storage paradigms, a crucial issue is determining the conditions under which content persists for a significant amount of time in the given area. [31] proposes a model for the interval of the time during which the content floats. [13] introduces the criticality condition, i.e., a sufficient condition for the content to float indefinitely with very high probability, under various mobility models. [82] demonstrates the feasibility of FC (in terms of ability to sustain content persistence within the RZ for a given period of time) even in setups with sparse node distributions. [83] proposes a model for content persistence for outdoor pedestrian mobility over large open spaces, such as city squares. [84], [85] characterize the mean time to information loss in several scenarios, based on synthetic mobility and on measurement-based vehicular mobility traces. [86], [87] propose a modeling approach based on mean field theory and a stochastic Susceptible-Infected-Recovered (SIR) epidemic model to evaluate content lifetime, and to determine sufficient conditions for content persistence. Other works (e.g., [47], [88]) focus on how to engineer the replication and storage strategies in realistic settings in order to efficiently guarantee a given success probability within a given time range. Despite focusing on resource efficiency, none of these works investigates those issues arising when several different contents are exchanged among a same set of nodes, and thus the impact of their approach on the amount of content which can be supported in a same area by the FC scheme.

A few works propose techniques for coping with resource contention among different contents floating in a same set of nodes. [89] proposes a content-centric dissemination scheme. Its solution is based on a policy which sets the order of content exchanges on a contact between two nodes, and the probability for a node to drop a content in a way which tries to maximize the total delivery rate over a set of contents of different popularity. The paper [82] proposes a scheme in which, in scenarios with several different floating contents and overlapping RZs, users prioritize the contents to exchange based on measures such as RZ size, or total amount of users with content in each RZ. In [90], authors assume that there are multiple different content items present in the network and that replicating all of them using FC may lead to overloading the wireless network and/or the storage capacity of nodes. In order to avoid these issues, they propose a strategy which controls the number of copies of a particular content item within a RZ.

None of these works however characterizes the limit performance of FC in terms of maximum number of contents which can be supported with a given minimum performance, as a function of system parameters, like this thesis does.

An important application of gossip-based information diffusion protocols is distributed caching [91]–[93]. In VANETs, self-organized storage or cloudlets adopt Vehicle-to-Vehicle (V2V) communications in order to retain information in a given area for a range of time, particularly in those conditions in which infrastructure is not available [24], [94], [95]. However, these works focus on such issues as the relationship between cache hit ratio and the amount of content redundancy, on communication overhead, on reliability, leaving open the key issue of the relationship between the performance of such caching systems and the limit capacity of the distributed storage system on which they rely. [84], [96] characterize the mean time during which information persists in a FC-based storage system. These works are based on simulations in various settings (such as highways [85], or city centers), and on an informal definition of FC-based storage. Most importantly, they still neglect the characterization of the capacity of such a distributed storage system. Similar to the present thesis, [97], [98] consider scenarios where a few nodes are static, and act as "repositories" in order to improve content retrieval and persistence.

Storage capacity quantifies the amount of information that can be actually stored, as a function of the number and size of contents to be made available, and hence it is a fundamental descriptor of the usefulness of such systems. To the best of my knowledge, the only work in the literature which tackles a problem similar to the one tackled in this thesis is [99]. Its authors consider a FC-based file storage scheme for vehicular nodes over a two-lane highway, in which the file is split into blocks, and erasure coding is used to enable recovery of the whole file.

For such a specific setup, and for the time period in which all blocks keep floating in the RZ, the authors propose a simple upper bound for storage capacity, which corresponds to the right member of the inequality in Equation (4.17) in Chapter 4.

However, they completely omit the issue of characterizing the tightness of the bound as a function of system parameters. In addition, their approach is based on an ad-hoc approach which critically relies on the very specific geometrical configuration of the road and its induced patterns of contact and mobility, in a way which does not generalize to other settings, such as, e.g., the urban vehicular scenarios considered in this work.

2.2.3. Computational capacity

The last chapter of this work is inspired by Gossip Learning (GL) [100], a collaborative Machine Learning (ML) approach that is motivated mainly by the need to guarantee privacy and scalability. GL is a modification of the Federated Learning (FL) [101] concept. Differently from FL, GL assumes that the raw data collected by (and hence available on) each device are not shared with other devices. Rather, they remain on the device itself. Learning is achieved both by collecting new data and by sharing and fusing models. Devices exchange models opportunistically, and progressively aggregate them. We will describe in more detail the GL scheme in Chapter 5.

Chapter 5 is also related to the domain of Mobile Sensing (MS). Most MS applications can be classified into either personal or community sensing. Personal sensing applications focus on the individual. On the contrary, community sensing, also termed *opportunistic crowdsensing*, takes advantage of a population of individuals to measure large-scale phenomena that cannot be measured by a single individual. In most cases, the population of individuals participating in crowdsensing applications share a common goal [102].

With the increasing availability of mobile devices capable of sensing and computing (e.g., smartphones, tablets, wearable devices, etc.), Mobile Crowdsensing (MC) has emerged as a compelling paradigm to collect large-scale data with the collective effort of a large number of mobile users, who leverage the built-in sensors in their handheld devices to accomplish sensing tasks. Examples of mobile crowdsensing applications are Gigwalk [103], Waze [104], and Million Agents [105], to name a few. A typical mobile crowdsensing system consists of three parts: participants (a.k.a. users), service requestors, and an application platform. Users are motivated by monetary or non-monetary incentives provided by service requestors to contribute sensor data from certain

Point of Interests (PoIs). The application platform processes customized tasks requested from the requestors, and coordinates users to jointly complete sensing tasks. Most of the existing research in mobile crowdsensing assumes that users leverage cellular network resources to upload sensor data, including measurement readings, audios, photographs, or video clips, to the application platform as soon as the data are generated. However, the above assumption becomes unrealistic when either infrastructure-based wireless network support is unavailable (e.g., in post-disaster recovery or in vast rural areas), or the infrastructure-based wireless network is overloaded or too expensive for the application goal. Han et al. [106] for example describe a scenario where, in the Yellowstone National Park, the ranger office launches a crowdsensing task, requiring users to locate and rescue a wounded elk by providing multimedia content (e.g., photos or video clips) of suspected animals. However, according to Yellowstone's Wireless Plan, the cellular coverage areas are intentionally limited to preserve the park wilderness character. As a result, until recently, in most areas of the park, the cellular network is completely unavailable. Another example is provided by situations of natural disaster, such as flood, hurricane, and earthquake, when the cellular infrastructure is completely or partially damaged. In these cases, it is obviously advantageous to migrate cellular data traffic onto alternative communication channels, D2D in particular.

The realization of mobile crowdsensing tasks over opportunistic D2D networks requires users to collaborate. Specifically, in addition to generating sensing data from involved PoIs, users must collaboratively relay data to the application platform. During the data transfer it is useful to collaboratively aggregate the collected raw sensor data so as to generate a summary of the results. The benefits of such a strategy are in the reduction of the volume of data and in a simpler processing at the platform [107]). For instance, collaborative fusion [108] based on opportunistic information exchange has been proposed in Intelligent Transport Systems, where sharing of tracks or perception data allows implementing a “virtual sensor", enabling the elaboration of a representation of the environment which is much more complete than that which can be implemented by only locally sensed data. In those applications, the sheer size of the data justifies the use of D2D communications in order not to overload the cellular network. Furthermore, in these applications, the sharing of the representation obtained by fused data (instead of the raw sensed data itself) is potentially more efficient, as only information relevant for the specific service/application is shared.

3 Floating Content in Disjoint Geographical Areas

Device-to-Device (D2D) communications have expanded the way of managing available network resources to efficiently distribute data between users. D2D exploits communication alternatives, in Opportunistic Networks, based on short range wireless radio technologies such as Bluetooth and WiFi-Direct. Besides, nowadays in most urban areas, realistic human mobility is characterized by often repeated patterns that can be used to accurately predict the next visited regions—we call these disjoint regions hotspots (or Replication Zones (RZs)). In this chapter, we present Predictive Content Dissemination Scheme (PRECISE), to explore and combine the D2D paradigm along with real mobility and predictions focused on the dissemination of content among hotspots. To analyze the viability of such scheme, we show simulation results and evaluate the average content availability, lifetime and delivery delay, storage usage and network utilization metrics. We compare the performance of PRECISE with state-of-the-art approaches, such as Epidemic, restricted Epidemic, and Proximity-Interest-Social (PIS) routing protocols. Our results underline the need for smart usage of communication opportunities and storage. We demonstrate that PRECISE allows for a neat reduction in network activity by decreasing the number of data exchanges by up to 92%, requiring the use of up to 50% less of on-device storage, while guaranteeing the dissemination of contents as with legacy epidemic dissemination protocols.

3.1. Social-aware Opportunistic Dissemination

3.1.1. Infrastructureless dissemination system

We study the dissemination of information between mobile users, opportunistically leveraging D2D and without the support of a network infrastructure and controllers running outside the mobile devices. We refer to the information to be disseminated as *pieces of content* (or *contents*, for short), and we assume that each piece of content is exchanged via a single message, which contains a complete set of instructions and data that can be processed by a user application. Hence, each piece of content can be disseminated independently of the others. Pieces of content are exchanged upon contacts between mobile devices, i.e., devices can establish a connection and attempt to exchange pieces of content when they are in transmission range T_r from one another. Mobile nodes continuously send beacons and scan the wireless spectrum looking for beacons sent by other pears, like, e.g., in a Bluetooth system. For simplicity, we assume that the transmission range is fixed and constant for all devices, because we assume that D2D connections are established only upon the detection of a strong link. We also assume that, once a connection is established, mobile nodes exchange pieces of content simultaneously in the two directions, at a constant speed, which is realistic if again we consider that only strong links are used. As a consequence, the impact of channel errors and transport protocol dynamics is neglected, also because we assume short range transmissions of small pieces of content (and one might think of using smart transport protocols like QUIC [109] to make this assumption realistic even in the presence of large pieces of content). However, a content transfer can fail when mobile devices exchanging data get out of their transmission range before the transfer is complete. In other words, differently from many epidemic-like dissemination schemes, we assume that content transfer is not instantaneous.

3.1.2. Mobility assumptions

We further assume that specific disk-shaped areas of the region are marked as *hotspots*, also referred to as RZs in the rest of the article. RZs are visited regularly by mobile users. More in general, in a planar region, we identify a large set of points where users can *dwell*, which we refer to as dwelling points. Some dwelling points, but not all of them, are within the RZs. Those points are selected based on the nature of the place (e.g., they correspond to an apartment, a university building, an office, etc.). Dwelling points within an RZ are chosen by mobile users uniformly at random, because we consider that an RZ is a homogeneous area including equally important dwelling points. However, each RZ has a given probability of being visited as a whole, and different RZs have different probabilities of being visited. Dwelling points outside the RZs are chosen uniformly at random, with a total probability equal to 1 minus the cumulative probability to select dwelling points within RZs. The users move from one dwelling point to another by following a path on the planar region, which is not necessarily a straight line due to topological constraints (e.g., in a taxi scenario, devices can only follow the roads reported on the city map). The specific mobility pattern reflects the social behavior of mobile users by specifying the probability to visit a random dwelling point within an RZ or to move towards a random destination outside the RZ. To realistically model user patterns, we consider that users alternate movements and pauses, and we consider two cases: (*i*) pedestrian mobility with synthetic traces alternating moves over shortest path trajectories at a constant speed (chosen uniformly at random for each move) and pauses with uniformly distributed duration; and (*ii*) trace-based vehicular mobility, in which the speed and trajectories reflects realistic traffic conditions, and the duration of pauses represent realistic inactivity periods of taxi drivers.

Since users express *stochastic preferences* when they decide to move to the next dwelling point, and preferences depend on the profile of the user, we assume that it is possible for a mobile device to predict with high accuracy where it will be dwelling next, given that it is at a certain location. Likewise, when two devices are in transmission range, they can predict whether they will reach or not a new RZ within a certain time. The specific prediction mechanism is out of the scope of this work, but we remark that devices can build statistics or use machine learning (e.g., Q-learning) to estimate if and when they will reach a hotspot.

With the above, it is clear that content exchange can be limited to cases in which mobile users are within an RZ or predicted to move to RZs soon. The dissemination of pieces of content outside RZs, in general, is far less critical than in the case of legacy opportunistic routing schemes. As such, we claim that, although traditional opportunistic routing schemes would serve the purposes of the described application scenario, more specific schemes are needed to make content dissemination efficient when the pieces of content are relevant for the RZs only. Indeed, existing schemes cannot be efficient in terms of use of resources, i.e., the way they use connection opportunities and buffer space on mobile devices, which are often limited resources. To this purpose, in the next section we propose a new content forwarding and storage scheme that leverages the social behavior of mobile users, and specifically their ability to predict their location in the near future.

3.1.3. Reference scenario and notation

The reference scenario considered in this work is a planar region with topological constraints for the mobility of users, e.g., a 2D city map with paths, buildings and obstacles. The radius of an RZ disk is denoted by r_{RZ}, and we assume that there are K RZs, each denoted as RZ_k, $k = 1, 2, \cdots, K$. An RZ represents the area in which the disseminated information is relevant. This means that we suppose the existence of an application running on the devices of mobile users, which will consume the received pieces of content independently, and only when the mobile device is inside an RZ. There are $N_k(t)$ mobile devices in RZ_k at time t, and the goal of the dissemination scheme in this context does not consist in reaching a specific destination device nor a fully indiscriminate epidemic broadcast of the pieces of content. Therefore, what is important is not the time to delivery to a specific destination a piece of content since its generation instant, but the number of devices that pass through the RZ and get the piece of content, and the time that elapses since when a user enters an RZ to when it gets the piece of content (which can be 0 in case the content was obtained before entering the RZ). We call *availability* of an RZ, for a given piece of content and at a given time instant, the ratio between the number of mobile devices within the RZ possessing the content and the total number of devices in the RZ. In Section 3.3 we formally define how to measure the availability per content and RZ and as a function of the time since when the content was generated, and the corresponding averages.

The notation used in the rest of the chapter is presented in Table 3.1 with a short description of each quantity later used in algorithms and mathematical expressions. Full details are given in the following two sections, which describe Precise and the relevant key performance indicators, respectively.

3.2. Precise: Data Communication and Storage Paradigm

Due to frequent link disruptions in opportunistic networks, the fundamental forwarding approach is to adopt pervasive forwarding solutions.

Epidemic spreading techniques, for instance, provide the most elementary and effective manner of content dissemination where nodes simply exchange content at any given opportunity [16], [21]. In general, at any encounter nodes will try connect and send content to their peers, sometimes restricted to a zone of interest. However, epidemics-based dissemination schemes introduce a high overhead, which not only causes network congestion but also high energy consumption at each node. Often, the number of content replicas and connections exceeds by far what would be essential for an efficient distribution of the data content in a realistic environment. For that reason, our work introduces a simple yet smart scheme to avoid some of the unnecessary connections and content replication, focusing at the same time on finding more beneficial exchange opportunities with no significant increase in computational cost.

A basic example of an epidemic strategy, which we will use as baseline, consists of allowing nodes to exchange content with their peers when in communication range and only within an RZ. Nodes within an RZ can always keep the content they are carrying. Once they leave the RZ, they will automatically drop all contents in order to free-up resources. In the following, we introduce a set of more advanced predictive scheme, Precise, which helps (a) to also carry content to other RZs, (b) to specifically pass content to nodes "going" towards an RZ, and (c) to drop content after some expiration time.

3.2.1. Forwarding scheme of Precise

The way nodes forward their content in PRECISE varies depending on which zone they are located in. In our forwarding algorithm (cf. Algorithm 1), nodes within an RZ can always exchange content and nodes in the outer area are only allowed to exchange contents if any of them is likely to reach an RZ before a time expiration threshold T_e, which represents the maximum time during which nodes keep running their data exchange

Table 3.1: Notation

Symbol	Meaning
α	Discount coefficient of the Autoregressive (AR) filter (adapted to the distance between AR updates so as to obtain an exponential decay of past values with T).
A_{ki}	Availability of content i in RZ$_k$, i.e., the fraction of nodes that possess content i with respect to the total number of nodes inside the k-th RZ.
$A(t)$	Mean availability computed over all existing contents and RZs, at time t.
$\overline{A_i}(\tau)$	Time-average availability for content i generated τ seconds after its injection in the network over all RZs
$\overline{A_G}(\tau)$	Statistical mean of the time-average availability for the group of contents G, computed for content lifetime equal to τ.
C_i^j	Binary variable indicating whether a certain content i is available at node j.
C_{ki}	Number of nodes possessing content i within the k-th RZ.
C_{te}	Time elapsed since a node has left an RZ without re-entering in another RZ.
$D(t)$	Set of contents injected in the network until time t.
$F_k(t_n)$	AR filtered value of $M_k(t_n)$.
G	Target group of contents.
g_i	generation time of content i.
K	Number of RZs.
$L_j(t)$	Buffer load of a node j.
$L(\tau_1, \tau_2)$	Time-average of the buffer load of all nodes, for interval $[\tau_1, \tau_2]$.
$M_A(t_n)$ $(M_B(t_n))$	Number of contents that node A (or B) attempts to retrive upon a connection is established at t_n.
$M_k(t_n)$	Number of contents to exchange upon a connection is established at time epoch t_n in the k-th RZ ($k = 0$ outside of RZs).
N_k	Set of nodes within RZ_k.
P_k	Probability to decide to exchange content upon a meeting, with Precise.
r_{RZ}	RZ radius.
$R(t)$	Mean fraction of injected contents held by a node at time t.
RZ_k	The k-th replication zone.
t, τ	Continuous values of time.
t_n	Current time epoch (discrete value). n is the time slot index
T	Exponential decay time of the AR filter.
T_e	Maximum time that a node can spend outside an RZ before emptying its buffer when using Precise; similarly, nodes outside RZs accept to receive contents if they predict to reach an RZ within T_e.
T_r	Transmission range of mobile devices.
$x(t_n)$	Variable containing the time epoch of the last connection, as observed at time t_n.

application while they are outside RZs. Since we identify similar node mobility patterns during weekdays, in order to predict whether the nodes are likely to visit an RZ at a given time, we analyze the information obtained from past traces at coinciding times of the day with a certain probability of failure. We do not use future information but we assume that the devices can predict their future position based on statistics collected in the past. Moreover, in case of nodes that are moving, they know were they are going (because they have selected a precise destination) and can reasonably predict when they will arrive. This process is represented in Algorithm 1 with the call to the function *isVisitingRZ(node/peer)*. More precisely, what the function isVisitingRZ(node/peer) does is, given that a node's speed is constant, get its value and check whether the node, moving at its current speed, will enter an RZ before T_e expires. In case one of the node's next hop lands inside one of the RZs, the output of the function isVisitingRZ(node/peer) will be a True value flag. When nodes are not expected to visit the RZs we assume the data are not relevant for them.

With Precise, when two nodes establish a connection, they will only transfer those pieces of content that are missing at the peer node. The order in which the contents are transferred is random: prior to the exchange, both nodes content lists are shuffled to prevent certain pieces from being repeatedly exchanged in the first place. This way, we guarantee that all contents have the same probability of being selected.

In case the data of one node do not completely fill their assigned capacity, the remaining quota will be relocated to the peer node. The established connection stays active until both peers have transmitted the total amount of contents. Therefore, connections are only interrupted in two cases: when nodes move away from each other beyond the transmission range or when both nodes fill up their storage capacity during the connection.

Besides, if nodes belonging to an established connection have nothing to exchange or their storage capacity is already full, the connection will be dropped right after a small fixed interval. Such interval represents the time needed to check each node status, which cannot be informed to the peer in advance with any current technology. In general, when a connection is interrupted, incomplete file transfers are dropped.

3.2.2. Decision making process in Precise

Decision making is a local process, running at each mobile node, and is based on the computation of the number of contents that nodes have to exchange when they meet, i.e., the number of missing contents at a generic node pair (A, B), indicated as M_k,[1] where $k = 0, 1, \cdots, K$, depending on which area the nodes are located in the k-th RZ ($k = 0$

[1]In the notation, we omit the dependency on the node when not necessary, but remark that each node has its own version of M_k and of the derived quantities.

means that the nodes are outside any RZ):

$$M_k(t_n) = M_A(t_n) + M_B(t_n), \tag{3.1}$$

where t_n represents the current time epoch, and $M_A(t_n)$ (respectively, $M_B(t_n)$) is the number of contents that node B (respectively, A) possesses and are missing in node A (respectively, B).

Considering that not all the previous pieces of information might be interesting at a given time, a low pass filter that covers a predefined previous amount of events is needed, specially to filter out some noise from instantaneous measurements, as done in any real system. We then use $F_k(t_n)$ as the mean observed number of contents to exchange, which is obtained by applying an AR filter to $M_k(t_n)$ when a connection occurs, i.e.:

$$\begin{cases} \alpha = e^{-\frac{t_n - x(t_{n-1})}{T}}, \\ x(t_n) = t_n, \\ F_k(t_n) = \alpha F_k(x_{n-1}) + (1-\alpha) M_k(t_n), \end{cases} \tag{3.2}$$

where t_n is the current time epoch, $x(t_n)$ stores the time epoch of the last connection started until t_n, and α is an adaptive value that accounts for the time elapsed in between two consecutive connections, so that the AR filter operates with an exponential decay time T. This corresponds to a negative exponential decrease of the importance of old samples. We used the time constant T of the order of hours because we follow realistic human patterns, which have to be measured in hours. If no connection occurs at t_n, the values of $F_k(t_n)$ and $x(t_n)$ are set as their respective values at t_{n-1}.

We need to wisely choose the value of T according to the amount of previous encounters that nodes are going to consider in order to derive the average number of contents that were exchanged in the past. Then, whatever happened before the decay time does not practically affect the value of the current average.

By computing $F_k(t_n)$ and comparing it with previously computed values, we tune an exchange probability P_k of actually starting a content exchange, i.e., the meeting devices might decide to skip a content exchange, to save resources. To this purpose, we use Algorithm 2. The proposed algorithm uses a negative control feedback: the more contents to exchange, the less nodes need to connect and exchange (because contents are already present in the scenario). The opposite is true when a node sees less contents than in the past: this is taken as a sign that less contents are around, so that the nodes must help the system more, by connecting and exchanging more frequently (with higher probability). The constant step chosen to adapt the exchange probability at every connection is set to 0.01, and we bound P_k to the interval $[0, 0.1]$ according to the sensitivity analysis presented in Section 3.4.3. Those values score a good tradeoff between avoiding large oscillations and adapting fast while reducing the number of data exchanges without paying in terms of dissemination performance.

It is important to note that nodes operate with different probability values depending on which RZ they are. This is enforced because nodes present different mobility patterns and therefore heterogeneous information exchange behaviors that will be more accurately analyzed separately.

3.2.3. Local storage management

To make smarter storage management decisions, we define a scheme to either preserve or drop the content from nodes' local buffers. In PRECISE, nodes within an RZ can always keep the content. If a node is in the outer area and visiting an RZ after a long period of time, we assume the content stored on its local buffer will be outdated and, thus, irrelevant for the RZ. Therefore, nodes should eventually discard such content to also reduce the resource consumption. We perform this by allowing nodes leaving an RZ to keep their stored content for a certain time, i.e., until the *time elapsed* since leaving the RZ reaches a maximum allowed value, which to be consistent with the forwarding scheme described in Section 3.2.1, is set to T_e.

Our storage scheme (described in Algorithm 3), along with the forwarding scheme, seeks to favor content availability by allowing nodes traveling between RZs to carry and exchange data in advance to other nodes heading down the RZs. Besides, nodes returning to the same RZ after a period of time shorter than T_e will also keep their data alive.

3.3. Key Performance Indicators

There are many parameters in PRECISE that can be fine-tuned to optimize the system. In the following, we briefly discuss the key performance indicators of the system.

We use $C_i^j(t)$ to denote a binary variable indicating whether a certain content i is available at node j at time t. Therefore, the number of nodes possessing content i within RZ_k is expressed as

$$C_{ki}(t) = \sum_{j \in N_k(t)} C_i^j(t); \tag{3.3}$$

note that $C_{ki}(t)$ is the number of replicas of content i available within RZ_k.

We therefore measure the availability per RZ and per content, $A_{ki}(t)$, as the fraction between the number of nodes that possess content i and the total number of nodes inside the RZ, i.e., nodes with contents outside the RZ are not considered. With the above, the availability at time t for content i in RZ k is expressed as

$$A_{ki}(t) = \frac{C_{ki}(t)}{|N_k(t)|} \;, \tag{3.4}$$

where $|\cdot|$ denotes the number of elements in a set. To evaluate the overall scheme, we will use the mean content availability at time t, which is the statistical average of the A_{ki} values computed over all RZs and contents, i.e.:

$$A(t) = \frac{1}{K|D(t)|} \sum_{k=1}^{K} \sum_{i \in D(t)} A_{ki}(t) \ , \tag{3.5}$$

where $D(t)$ denotes the set of contents injected in the network until time t.

Another relevant metric is the time-average availability for content i generated at time g_i, computed τ seconds after its injection in the network over all RZs. This quantity can be expressed as

$$\overline{A_i}(\tau) = \frac{1}{\tau} \int_{g_i}^{g_i+\tau} \frac{1}{K} \sum_{k=1}^{K} \frac{C_{ki}(t)}{|N_k(t)|} dt \ , \tag{3.6}$$

and the overall statistical mean of the time-average availability (the *total content availability*, for short) for a target group of contents (denoted by G) is the statistical average of per-content availability values $\overline{A_i}(\tau)$, $\forall i \in G$:

$$\overline{A_G}(\tau) = \frac{1}{|G|} \sum_{i \in G} \overline{A_i}(\tau) \ . \tag{3.7}$$

It is important to note that, at content generation time ($\tau = 0$), the total content availability $\overline{A_G}$ is low given that each piece of content belongs to a unique node before the spreading process starts. Thus, the availability curve over time undergoes a transient period prior to stabilizing according to the system capacity.

It is also critical to understand the load $L_j(t)$ of each node's local buffer, so to compare the efficiency of the different configurations applied to the system. The load of a node j is defined as

$$L_j(t) = \sum_{i \in D(t)} C_i^j(t) \ , \tag{3.8}$$

and the following quantity expresses the mean fraction of injected contents held by a node at time t:

$$R(t) = \frac{1}{\sum_{k=1}^{K} |N_k(t)|} \sum_{j \in \cup_{k=1}^{K} N_k(t)} \frac{L_j(t)}{D(t)} \ . \tag{3.9}$$

Taking into account the nodes' load, we can observe the saturation value of the system as a whole, using the total average load L over the time interval $[\tau_1, \tau_2]$, which is defined as

$$L(\tau_1, \tau_2) = \frac{1}{\tau_2 - \tau_1} \int_{\tau_1}^{\tau_2} \sum_{k=1}^{K} \sum_{j \in N_k(t)} \sum_{i \in D(t)} C_i^j(t) dt. \tag{3.10}$$

Finally, other important performance indicators are: (i) the number of connections used by the nodes, which depends on mobility, frequency of meeting events and availability of nodes to connect when they are in transmission range, (*ii*) the fraction of contents that disappear from the network after being injected, (*iii*) the time during which a content survives in the network, and (*iv*) the delivery time needed by a node to receive a content after entering an RZ. The number of connections measures the communication load due to the dissemination process, while the loss count and the content lifetime express the ability of the dissemination process to keep information available over time without the help of any infrastructure. The delivery time tells how efficient the dissemination is with respect to RZ visits.

3.4. Performance Evaluation

We have built an opportunistic content dissemination simulator using Python, to reproduce the D2D-enabled application scenarios described in Section 3.1, with the algorithms of Section 3.2. New features are developed in our custom simulator that are not present in state-of-the-art simulators. For instance, the ability to exchange contents combining different circumstances, such as, certain periods of time and selected places in the scenario, basing the previous decisions on predictive information. This way, we can flexibly explore diverse scenarios according to our specified input parameters. It also facilitates the configuration of more complex scenarios, the post-processing of several metrics and opens doors to further modular extensions [3]. We have used the Floating Content (FC) simulator version for this chapter, as detailed in Chapter 6.

3.4.1. Geographical and mobility scenarios

To evaluate PRECISE, we consider two realistic geographical environments, which are suitable to simulate pedestrian and vehicular (taxi) mobility, respectively. The structure of our application scenarios is composed of the following mutually associated components:

- The scenario consists of a squared 2D region, extracted from either a map of Paderborn, Germany (for pedestrian mobility cases) or from the one of Rome, Italy (for taxi mobility patterns).

- Within this region, we define two (circular) RZs that represent hotspots where disseminated data is especially valuable for nodes traversing them. RZs are placed at opposite locations of the maps and all data contents are aimed to travel between these two zones.

- The scenario also contains mobile users (*nodes*), that move according to two different methods: for pedestrians, we use synthetic traces based on the city map of Paderborn, while for taxis, we use real traces obtained from [70]. In case nodes follow synthetic traces, they are uniformly distributed in space with a certain user

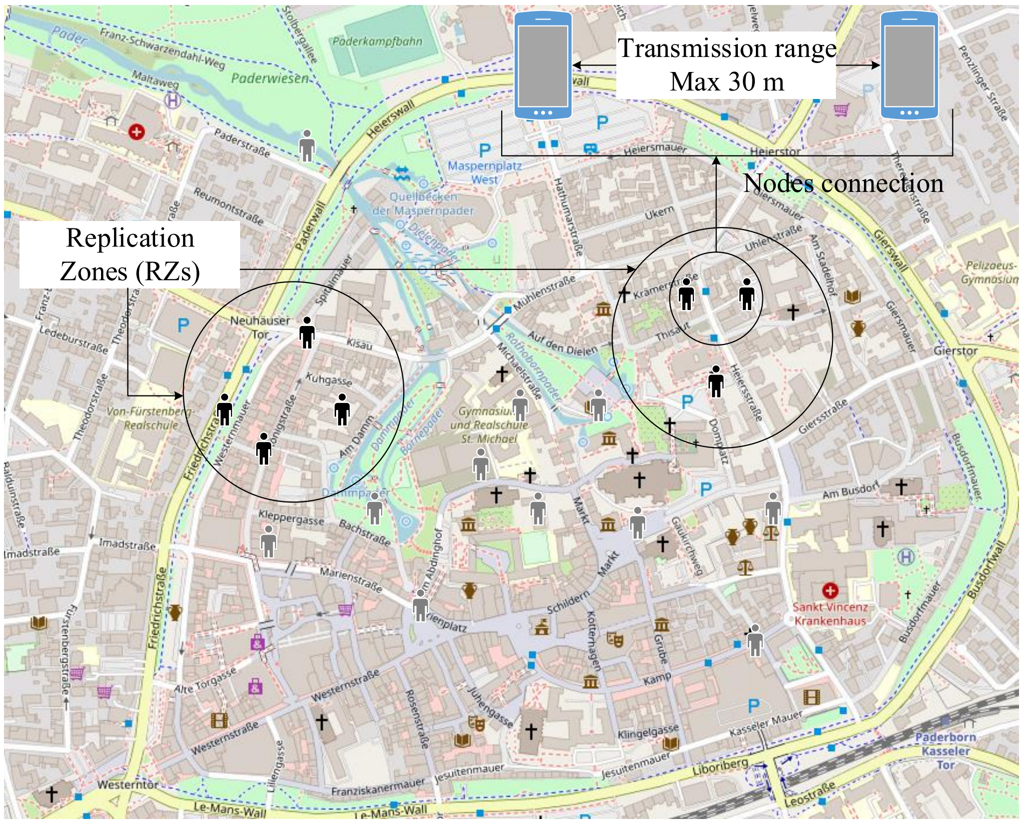


Figure 3.1: Scenario of Paderborn including two RZs and 50 users moving within the scenario

density. In all cases, nodes have an accurate estimate of their position and can forecast the time at which they will reach RZs.

Figs. 3.1 and 3.2 show the map sections used in this chapter, from the city of Paderborn and Rome respectively. In the case of Rome, the figure only shows a detail of the full map. The traces used cover a larger area which cannot be modeled as a uniform square region. However, the taxis spend 67% of their time in a square area of dimensions 4 km$x$6 km within which we selected the RZs. Figures depict, approximately, the size and location of both defined RZs together with the type of nodes of each scenario, being pedestrians in the first case and vehicles in the second. We also indicate in the figures the node transmission range of each setting (see Table 3.2 for numerical values). We initialize the system based on a set of parameters explained in Section 3.4.2, which can be tuned according to the desired scenario. Note that the structure of a scenario can take more complex configurations, composed of one or multiple RZs and also supports multiple types of content per RZ.

We have used the ONE Simulator [110] to generate pedestrian traces based on the

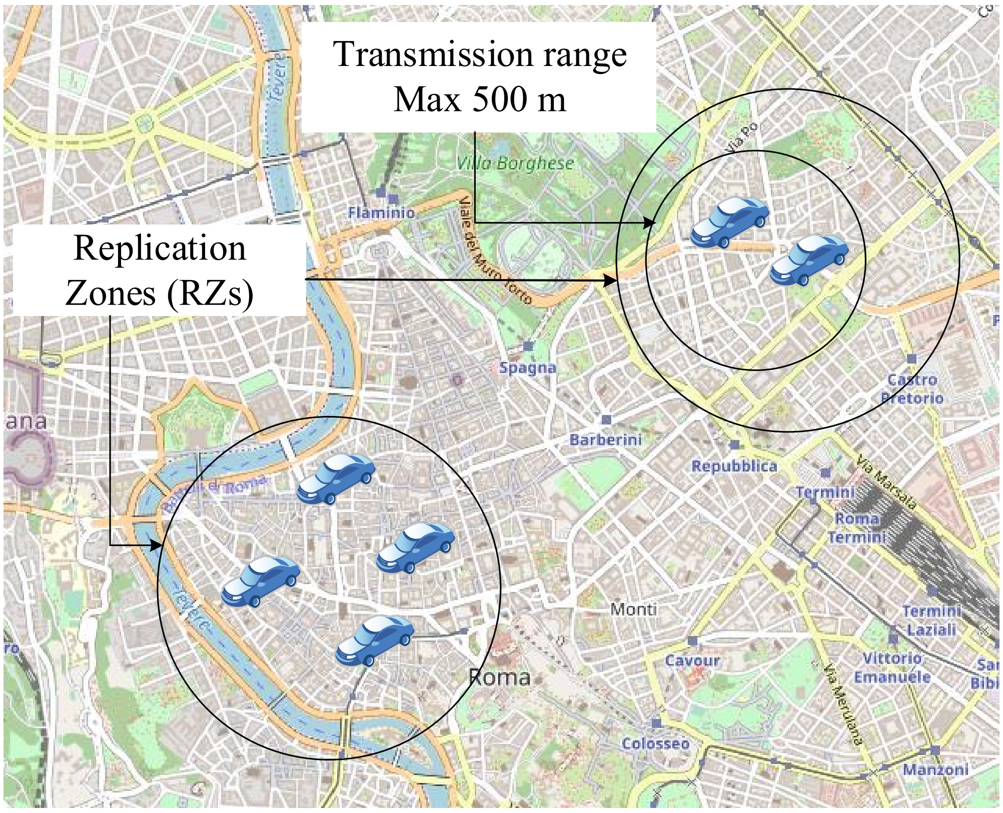


Figure 3.2: Detail of the full scenario of Rome including two RZs and up to 199 mobile devices moving on taxi cabs within the scenario.

map of the center of Paderborn and a set of manually selected dwelling points which corresponds to gathering places in the city. Moreover, we consider three pedestrian mobility models, which show the different dynamicity levels typical of a *Businessman*, a *Clerk*, or a *Student*. *Business*'s mobility assumes rather long pause time and inhomogeneous RZ visiting probabilities. This leads to a less routine movement of the nodes, which resembles business people's mobility. *Clerk*'s mobility increases pause periods and RZ visiting probabilities. This mobility represents the case of commuters. *Student*'s mobility represents an intermediate case, with nodes moving between positions at shorter periods and visiting a RZ with a probability higher than in the Businessman's mobility case. This reflects a university environment.

In the case of taxi cab's mobility in Rome, we count on 5 sets of 2 days traces each, obtained from cabs mobility measurements [70]. These traces were collected during consecutive complete days where cabs reported their position with 15 s granularity. By previously studying the sets of traces, we could observe the most visited regions of the city along different days, such as the airport and the main streets of the city center. Thus,

Table 3.2: Input parameters

Parameter	Paderborn	Rome
Region size	2.25 km^2	24 km^2
Radius of replication (r_{RZ})	200 m	1000 m
Transmission range (T_r)	30 m	500 m
Channel rate	10 Mbit/s	10 Mbit/s
Memory limit	infinite	infinite
Number of users in the region	50	199
User speed	0.5–1.4 m/s	From traces
Slot length	1 s	1 s
Content size	5 Mbit	5 Mbit
Injected contents	2	2
Periodic injection time	1000 s	1000 s
Elapsed time (T_e)	400, 600 and 800 s	400, 600 and 800 s
Pause time	2–5 h (Businessman) 3–8 h (Clerk) 15–30 min (Student)	4.6 min on average
RZ visiting prob	0.2 and 0.1 (Businessman) 0.7 and 0.2 (Clerk) 0.4 and 0.3 (Student)	0.036 (RZ_1) 0.441 (RZ_2)

defining them as our RZs. The key characteristic of the traces from Rome, compared to Paderborn scenario, is the dynamicity of the nodes. In this case nodes are vehicles, not pedestrians, which move at a higher speed. This will entail larger number of visited places and nodes encounters but will impose shorter contact intervals.

In all cases, we assign nodes to two groups, each with different RZ visiting probabilities, so that each group has a preferred RZ to visit. This way, nodes will likely visit both RZs, but will stay for longer periods within one of them. By so doing, we test the ability of PRECISE to efficiently move contents between two disjoint hotspots located within a larger area.

3.4.2. Parameter setting

In the simulations, time is subdivided into *slots* of 1 second. In each slot, nodes can connect and exchange data: connection establishment is always completed within a slot, while data exchange can involve one or more slots. Simulations were run for a duration of 172800 time slots, i.e., 48 h, which we identified to be sufficient to observe the system out of any transient. Besides, we have carried out 20 simulation runs per specific configuration in Paderborn and 15 for Rome scenarios, which was enough to gather sufficient statistics.

As for the simulation settings, the main parameters used are listed in Table 3.2. We differentiate among Paderborn and Rome scenarios due to the different characteristics and dimensions of each case.

3.4.2.1. Set up

In the simulations, the radius of the RZs has been set to 200 m for Paderborn and 1000 m in Rome. A randomly chosen node within an RZ injects a new piece of content every 1000 slots, and the initial number of injected contents in the first slot is two, i.e., one content per RZ. The max amount of stored contents per node is given by the storage capacity described by the *memory limit* parameter, which here we consider unbounded, for simplicity. We assume that when two nodes are in contact, the channel rate is constant over time and equal to 10 Mbit/s.

For pedestrian mobility, 50 nodes are uniformly distributed across allowed dwelling points, and when they move they follow the shortest path allowed by roads and squares in the map. The two selected RZs have quite different characteristics: the RZ on the left of the map only includes dwelling points spaced more than 30 m, which is the transmission range, while the RZ on the right includes a cluster of attractor points within transmission range. This will enforce differences in the metrics observed in the two RZs, because dwelling points within transmission range behave like a single dwelling point with the sum of the respective mobile users.

For Rome, 199 nodes follow the trajectories provided by the taxi cabs deployed across the city from which not all nodes might be present in the scenario at initial time, neither during the whole simulation. Note that, as a consequence, nodes will remain within the area for limited periods of time.

3.4.2.2. Precise

Experiments with PRECISE are structured to study three fundamental features. The first feature is **node mobility**, defining speed and pause times. For Paderborn scenario, we consider *Businessman*'s, *Clerk*'s, and *Student*'s mobility scenarios, according to mobility dynamicity. In all scenarios, nodes are split in two equally sized groups: each group has a preferred RZ, which is visited more often (see Table 3.2). *Business*'s mobility assumes rather long pause time of 2–5 h, and RZs are visited with relatively low probabilities (0.2 and 0.1). *Clerk*'s mobility increases pause periods to 3–8 h and RZ visiting probabilities are quite high (0.7 and 0.2), leading to low probability to select a dwelling point outside the RZs. *Student*'s mobility sees nodes moving often, visiting for only 15–30 min the two RZs with mildly high probabilities of 0.4 and 0.3, respectively.

For Rome scenario, the pause time reported in Table 3.2 corresponds to the average pause time computed over the first 2 days of traces. A pause is described as every interval of time longer than 30 s where taxi cabs report the same position. We have also directly obtained from traces the RZ visiting probabilities for Rome by computing the average time spent by taxis at every RZ and outside.

The second feature concerns the **content exchange scheme**, which is mainly affected by T_e, i.e., the interval during which nodes can keep data stored and also exchange content when they are outside the RZs. We studied a wide range of configurations for this parameter, between 0–800 s.

The third feature concerns the **data exchange area**, i.e., the restrictions on where data exchange can take place. We label with 'in' the scenarios in which exchanges are allowed only inside RZs and with 'out' those in which exchanges are allowed everywhere.

3.4.2.3. Benchmarks

Besides, we consider two extreme benchmark cases. In the first, labeled as 'Restricted Only In RZ', nodes cannot carry any piece of content outside the RZ. This is the typical approach used in other works dealing with content dissemination in RZs. In a second benchmark, we use a scheme that allows to keep and exchange contents limitless outside RZs, labeled as 'Epidemic $T_e = \infty$' in plots. This represents a typical uncontrolled epidemic diffusion scenario.

In addition, we have implemented PIS, a content dissemination protocol that bases its content dissemination decisions on three different social dimensions called similarities. First, PIS takes into account the physical proximity between nodes. Each node builds a so called Ego Matrix to keep track of, not only its own previously met contacts, but also the contacts of the peer devices that it encounters. Each node's Ego matrix is updated at every time slot with the information of the neighboring nodes. Then, the physical proximity similarity is computed using the information stored in the matrix for the next n time slots, and we use $n = 1$. Second, PIS considers users interests to decide whether they will possibly meet the destination node and therefore, be selected as a next hop candidate. We have randomly generated lists of interests for PIS with the dwelling points selected in the application scenarios, which is the same as we do for PRECISE. Third, PIS spots friendship between nodes according to the number of unicast messages that they have exchanged in the past. Social relationships for PIS are also randomly generated since we assume that the only piece of information that a system will be able to retrieve is the mobility of nodes, thus avoiding extra overhead.

We run PIS routing protocol in our scenarios with the previous settings and, bearing in mind that we consider all nodes as potential destinations, we also inject a high number of content copies. When computing the exchange decision parameter *simPIS*, as explained in [65], a constant value of γ needs to be selected in order to assist or constrain the dissemination of contents. This value has an impact on the system when the number of content copies is low. Given that, when γ is high it allows for a rapid content dissemination, thus PIS takes the risk of sharing all pieces of content earlier than the destination node is reached according to the following equation:

$$simPIS + \gamma > 0. \tag{3.11}$$

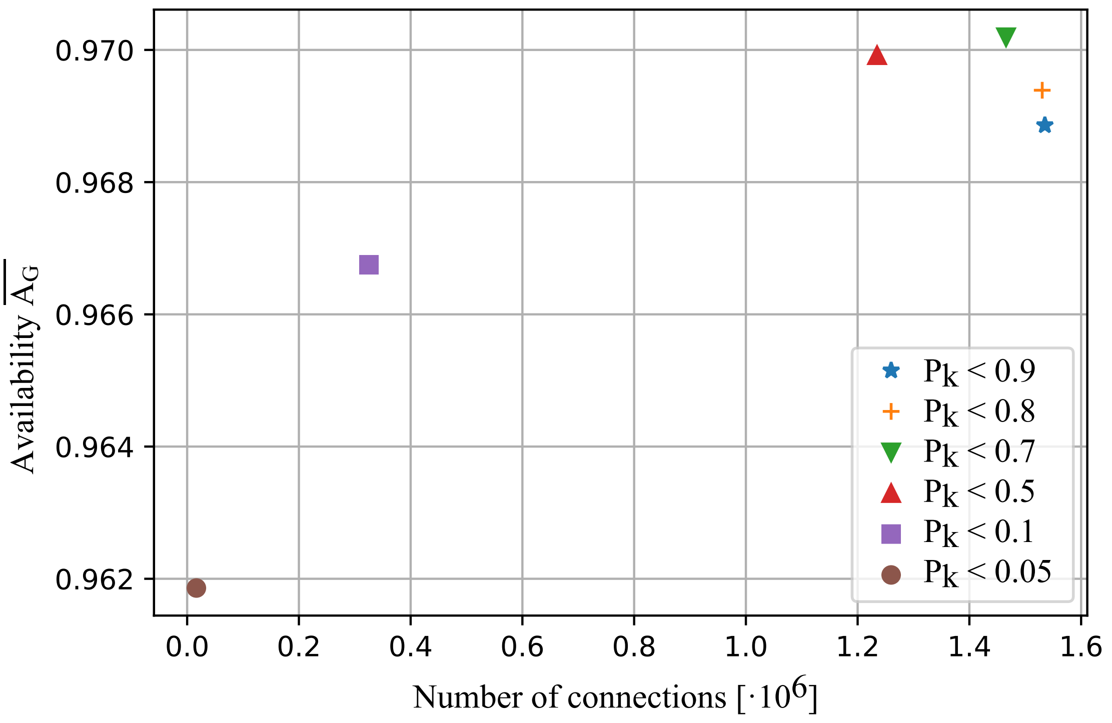


Figure 3.3: Average content availability and standard deviation versus number of connections for different values of the content exchange probability.

To prove that, we have run experiments for the two extreme cases of γ: 0.2 and 0.8.

We compare our results with these three benchmark cases, whose results are reported within each figure.

3.4.3. Sensitivity analysis

We have carried out a parameter study of the probabilities involved in the decision making process explained in Section 3.2.2. Our goal is to reach a fair trade-off between the content availability and the overhead generated due to the number of connections established. This trade-off can be regulated by adapting the probability with which a content can be exchanged given the requirements of our predictive scheme.

We performed a set of one-day simulations for the group of student mobility nodes on Paderborn scenario, with $T_e = 800s$ and allowing nodes to exchange contents also outside RZs. We test different probability values as shown in Fig. 3.3 and observe that using a very restrictive condition, i.e., forcing $P_k < 0.05$, gives a desirable reduction in the number of connections in exchange of some decrease in the content availability value. However, we aim at comparing PRECISE with other approaches such as Epidemic or PIS routing protocol, which are proven to be very competitive in terms of content availability [65]. We notice a sharp flop of performance in the availability from $P_k < 0.1$ to $P_k < 0.05$. For that reason, we observe that configurations closer to $P_k < 0.1$ are more beneficial because

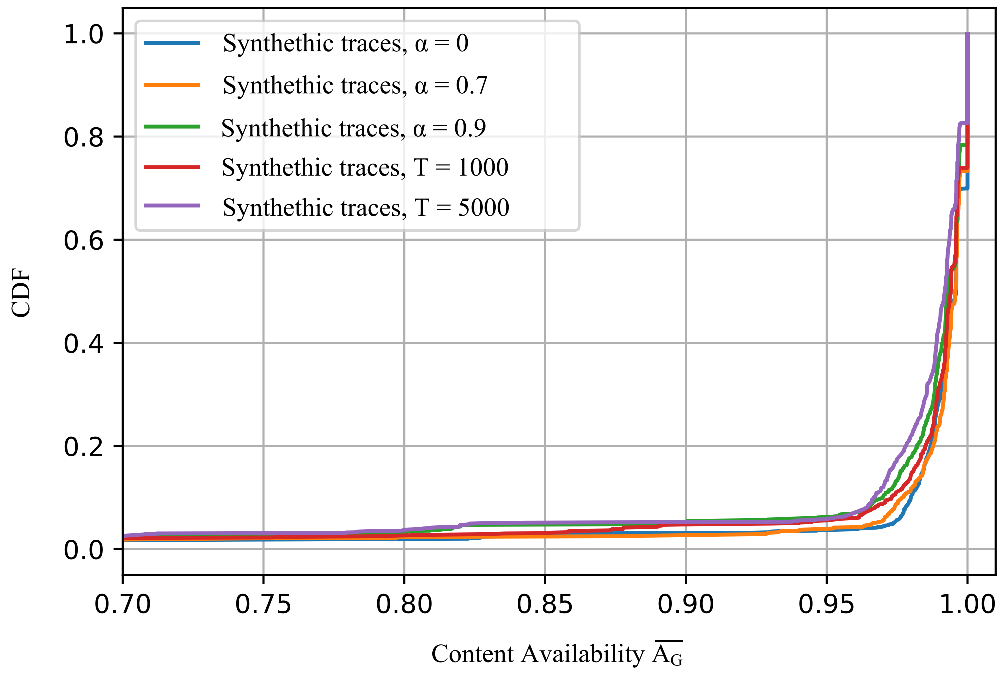


Figure 3.4: eCDF of content availability over one day experiment in Paderborn.

they increase content availability while still getting benefit from the overhead compared to the mentioned approaches. In the following results, we fixed the constraint to $P_k < 0.1$

In Fig. 3.3, we can see the significant impact of the parameter chosen for the probability. This is one of the reasons why, instead of using a fixed value for the probability condition, we have decided to use Algorithm 2, in which P_k is adjusted in steps, based on a quantity F_k which is the output of an AR filter. We based the probability on the average number of contents that two nodes have to exchange at every encounter as presented in Eq. (3.2).

Content availability results for adaptive values of α are depicted in Figs. 3.4 and 3.5 for synthetic traces and real traces from the literature [70], respectively. Note that, in both cases we run again a single one-day simulation which is a good representative example for the parameter study. From the figures, we can observe that the variations in content availability are almost negligible. In the case of synthetic traces in Paderborn scenario, the best results are grasped when α has a fixed value of 0.7. In the case of real traces in Rome, the content availability increase is sensed when adaptive α is computed from a large value of $T = 5000$, which means larger memory, corresponding to about 7 h[2].

Surprisingly, Figs. 3.4 and 3.5 do not show a significant improvement when no memory is taken into account, i.e., $\alpha = 0$ which means that all connections are allowed at any

[2]With an auto-regressive filter like the one used in this work, the importance of a sample fades exponentially with decay time T, therefore it becomes practically negligible after $5T$.

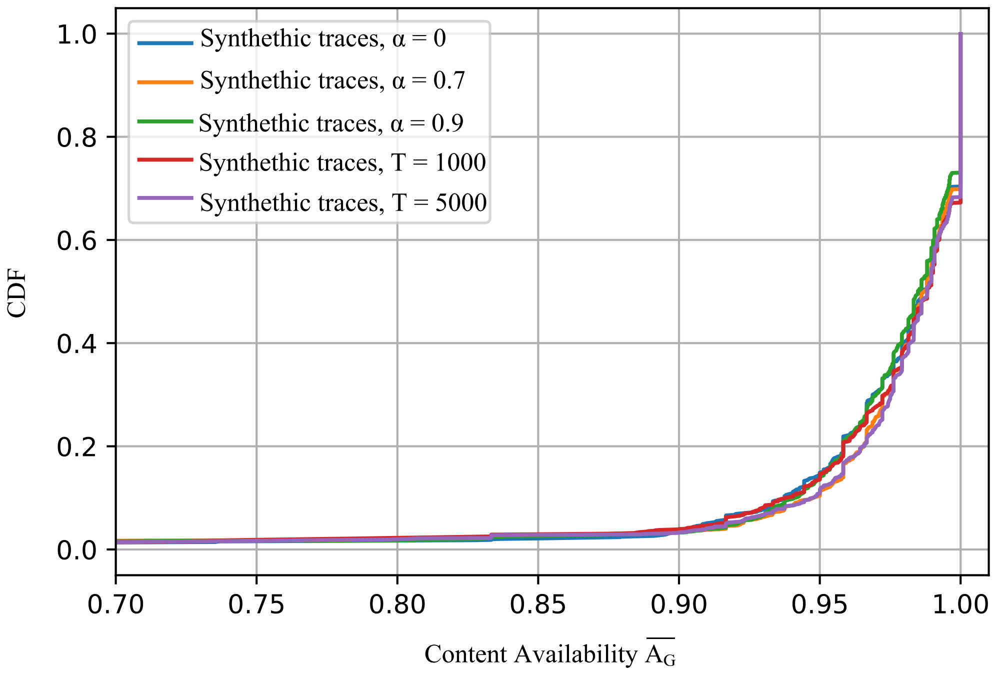


Figure 3.5: eCDF of content availability over one day experiment in Rome.

time. Therefore, increasing the network load, as shown in Section 3.4.6, is not essential to get content availability values close to 1.

This way, we have verified that the value of T, or even fixing alpha, does not bring in notable differences, so we picked a round value of $T = 1000$, which is enough to keep in the memory of the filter whatever happens while a node is outside the RZs for about T_e.

3.4.4. Content availability

One of the main performance parameters of the system is the mean content availability $\overline{A_G}$ inside RZs. In each of the tested configurations, we have obtained $\overline{A_G}$ by averaging per-content availabilities across all contents (i.e., we use $G = D(t)$). Fig. 3.6 shows this metric for each configuration. The figure shows also the standard deviation observed in the simulations. The computed confidence intervals are small and hence we do not report them in the figure. Their value indicates that, with probability 95%, the actual average is within $\pm 0.19\%$ of the estimated average reported in the plot. As it can be seen, if data exchange outside the RZs is permitted, the content availability increases in comparison to the traditional paradigm, where content exchange is only allowed for nodes inside RZs. When it is possible to carry a piece of content outside the RZ, performance improves drastically: in fact, results for the case 'Restricted Only In RZ,' in which nodes clear their storage as they leave the RZ, are the worst. Moreover, configuring our proposed scheme so to keep contents stored on nodes for a limited amount of time performs comparably well

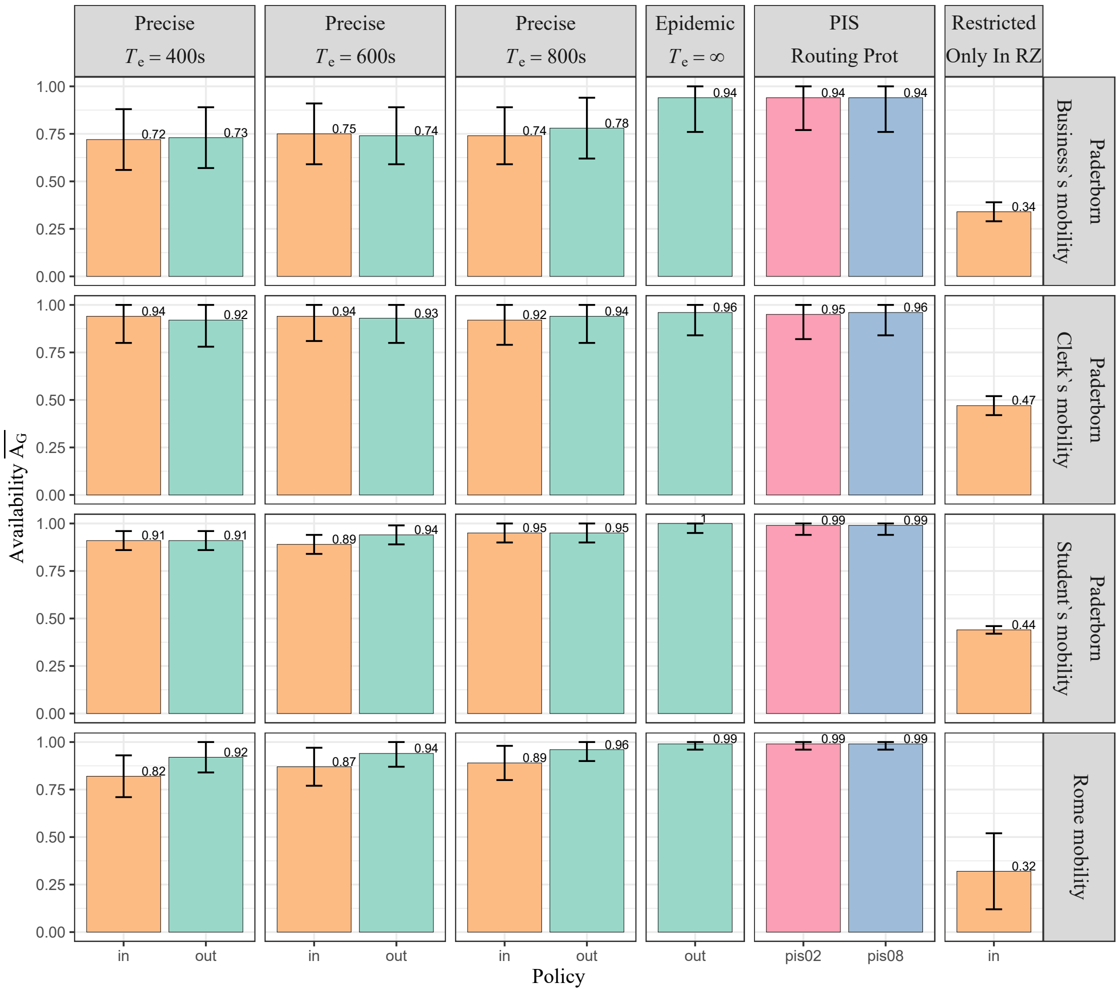


Figure 3.6: Time-average content availability (cf. Eq. (3.6)–Eq. (3.7)) for all forwarding and storage configurations, computed over the entire duration of the experiments and for all contents ($G = D(12\text{h})$ in Eq. (3.7)). Plotted is the mean together with the standard deviation.

with respect to the uncontrolled dissemination case ('Epidemic $T_e = \infty$' in the figures). These results provide valuable information on how to manage resources of the network.

Results show a similar trend over all mobility cases. However, if we compare results obtained from different mobility patterns in Paderborn, Clerk's and Student's mobility cases look similar while Businessman's mobility depicts lower availability numbers. In the Student's case, nodes move with the lowest pause time at hotspots and, therefore, the higher number of nodes encounters explains the rapid pervasion of contents. We can clearly appreciate this fact in Fig. 3.7, where we plot the mean content availability across all contents, $A(t)$, as it evolves over time[3]. Student's mobility subplots illustrate

[3]Note that, in Fig. 3.7, time is relative to content generation epoch. This is to be able to homogeneously compute the statistical mean for a set of contents generated at different instants.

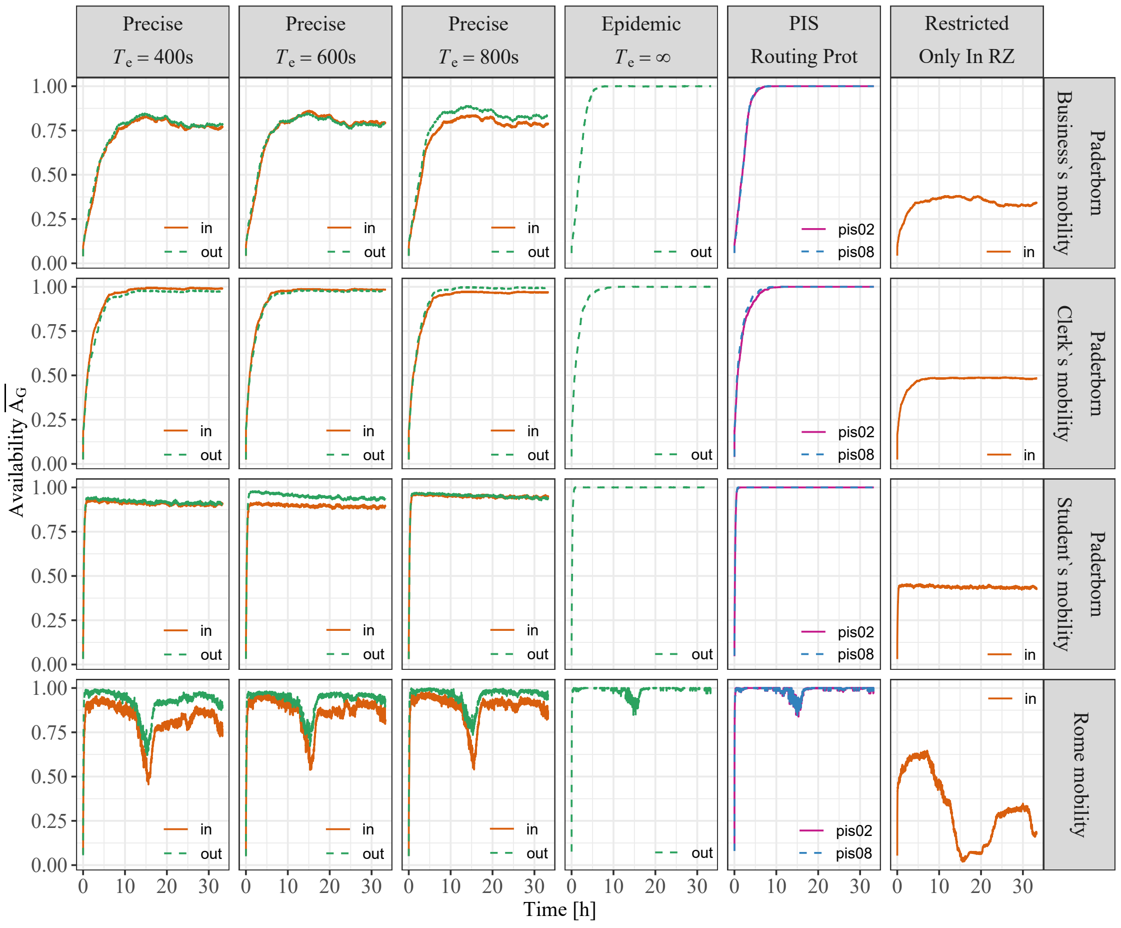
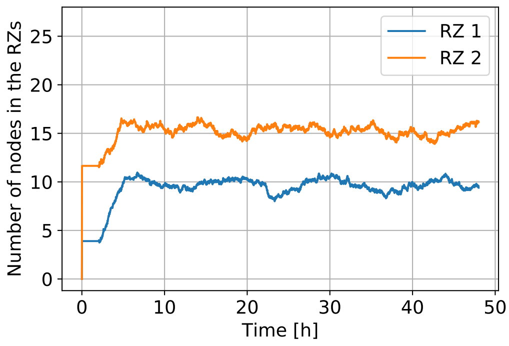


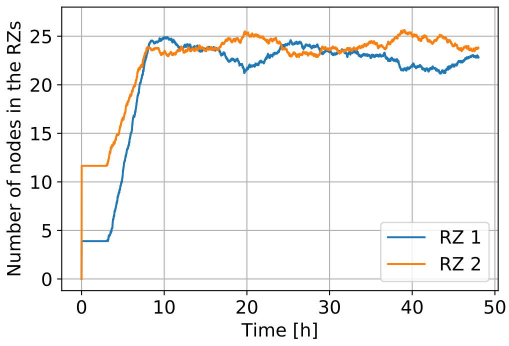
Figure 3.7: Evolution over time of the mean content availability (cf. Eq. (3.6)–Eq. (3.7)) observed under different forwarding and storage configurations.

a quick content spreading among nodes and very small variation across different contents behavior, that is why standard deviation in Fig. 3.6 is smaller than under Businessman's and Clerk's mobility cases. Here, when contents are first generated, it takes around one hour only to hit the highest availability values compared to Clerk's mobility patterns that reach its highest point later on but keeps it in steady state. Businessman's mobility subplots depict a rising curve that, in most of the cases, does not reach either availability values as high as nodes under the Clerk's or Student's mobility over the entire run.

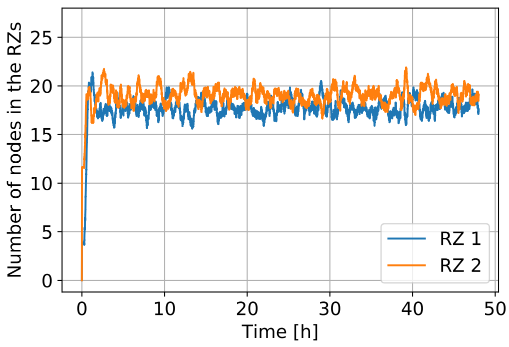
Finally, the simulations executed over the traces obtained from taxi cabs in Rome depict lower values of content availability due to higher nodes speed and the frequent disappearance of nodes from the considered area. On average, nodes are present in the scenario for 11 h, and once they leave the scenario the content stored in their local buffer is erased. In view of these results, we can perceive the influence of the nodes mobility pattern over the content availability. Fig. 3.8 shows the evolution over time of the



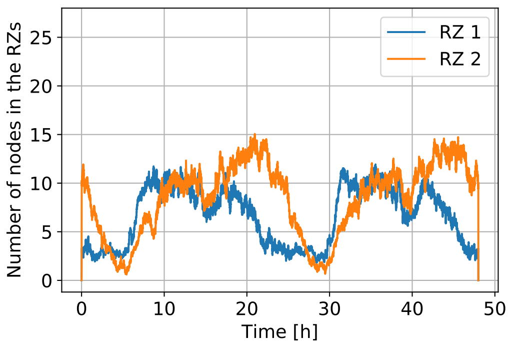
(a) Businessman's scenario



(b) Clerk's scenario



(c) Student's scenario



(d) Taxi's scenario

Figure 3.8: Evolution over time of the number of users present at each RZ for each scenario.

number of users present at each RZ for each scenario. Comparing the results with the number of nodes in the scenario over time, and specifically for each RZ in Fig. 3.8d, we can appreciate that the rising trend in content availability corresponds to higher visits to the RZs, however we are clearly able to overcome the decreasing trend by applying our proposed scheme. Nodes leaving the RZs in a short period of time but remaining in the scenario are still a critical asset to spread the collected contents until *time elapsed* value is up.

From the results, it is clear that availability increases dramatically when nodes are allowed to keep the contents while outside RZs. Likewise, it is clear that there is an extra benefit due to allowing nodes to connect and exchange data outside RZs, at least under the mobility models explored. More importantly, we demonstrate that there is no need to allow nodes to store and exchange contents for long periods of time. A fair performance can be reached by constraining the time elapsed parameter T_e according to the scenario settings while maintaining comparable values of content availability to those reached by epidemic dissemination techniques.

3.4.5. Storage usage

Another significant system metric is the amount of storage used. As shown in Fig. 3.9 (we plot again the mean together with the standard deviation and we do not report confidence intervals in the figure since value indicates that, with probability 95%, the actual average is within ±2.71% of the estimated average reported in the plot.), our predictive scheme performs better than the benchmark that restricts storage and data exchanges to RZs, and is comparable to the uncontrolled dissemination scheme in terms of available replicas. Note that, with the high values of content availability achieved by our predictive scheme, the number of replicas should be comparable to the number of nodes inside each RZ (see Fig. 3.8). With the parameters chosen for the three mobility scenarios that we have simulated in Paderborn, 30%, 70% and 90% of nodes will lay within RZs, for Businessman's, Student's and Clerk's mobility cases, respectively. Therefore, with the availability values reported for our scheme in Fig. 3.6, we should expect to observe about 13 to 15 replicas for the Businessman's case, 19 replicas for Student's case, and 23 for the Clerk's case. For Rome, we expect to have between 9 and 11 replicas inside the RZs according to the average number of nodes visiting the selected RZs. However, Fig. 3.9 shows lower values because it accounts for the delay incurred for spreading newly injected contents. More in detail, Fig. 3.10 shows how the number of contents stored in each node tends to increase as time passes. Abrupt changes visible in the figure are due to the periodic injection of fresh contents.

In the figure, we can observe how fast the storage capacity of nodes fills up, according to each mobility pattern. It draws our attention that the Businessman's case presents a slower increase in memory usage compared to Clerks and Students nodes. This is due to the fact that, in Businessman's case, nodes do not visit RZs as often as in Clerk's case, therefore, they miss opportunities to retrieve the data; in the Businessman's case, nodes neither have as many new peers encounters as in the Students mobility case, due to higher pause times. However, if we analyze again Fig. 3.6, we see that Businessman's mobility settings provide fair availability compared to Clerk's and Student's mobility cases, while occupying 50% of their storage and experiencing less content exchanges, as explained later in Section 3.4.6. Fig. 3.11 further shows the normalized number of per-node stored content, $R(t)$, i.e., the fraction computed with respect to the number of injected contents. The computed confidence intervals are small and hence we do not report them in the figure. Their value indicates that, with probability 95%, the actual average is within ±0.22% of the estimated average for the whole runs. Content injection epochs are clearly visible in this figure. The figure shows that the system tends to quickly spread fresh injected contents, though this process is strongly correlated with the speed and dynamicity of nodes. Indeed, it is clear that in Student's mobility scenario contents are distributed more rapidly. Despite the fact that the results for Businessman's and Clerk's mobility traces present a slower increase, Clerk's mobility figures illustrate a more

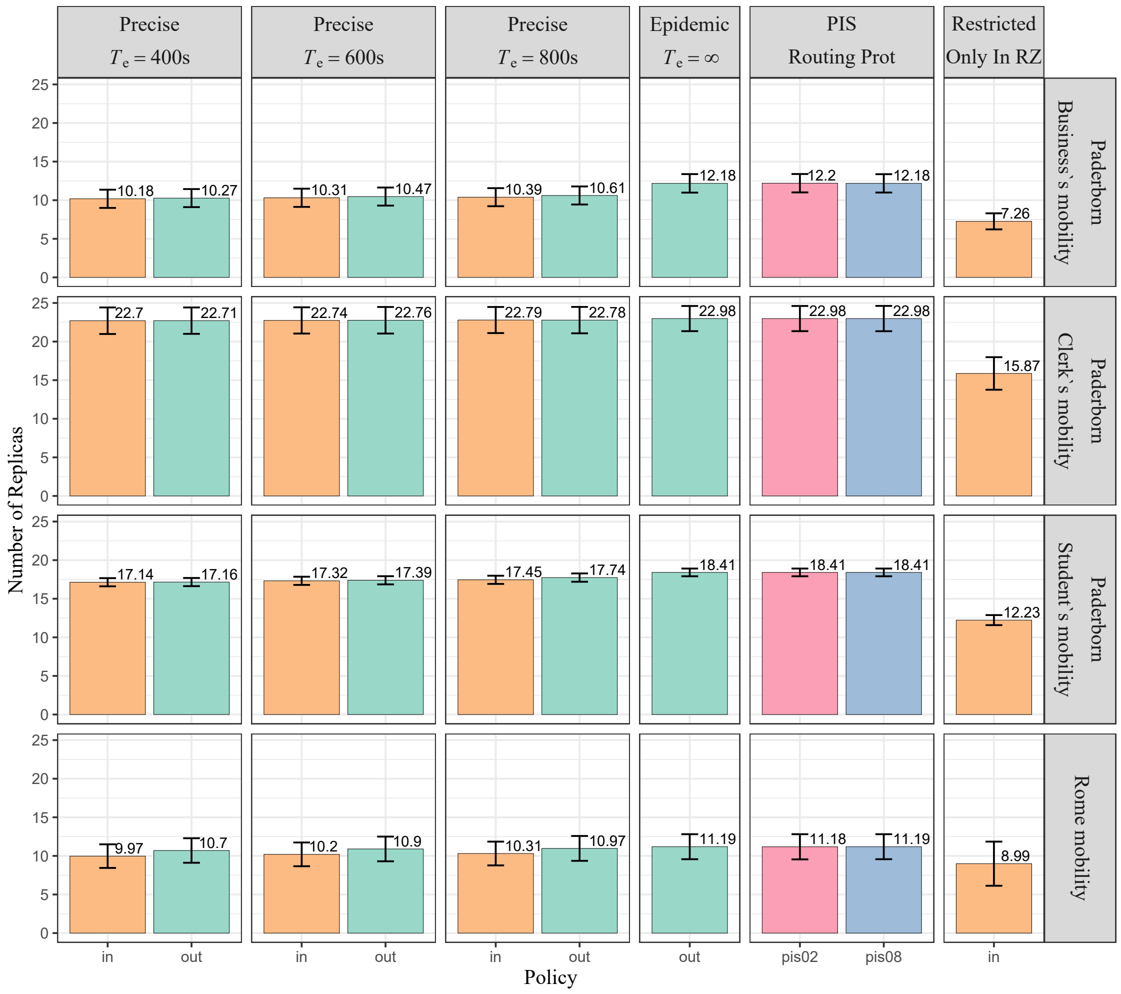


Figure 3.9: Average number of replicas of a content, computed across both RZs and time-averaged over the entire simulation. Plotted is the mean together with the standard deviation (taken with respect to the time-averaged number of replicas of each content).

agile dissemination of contents among nodes due to higher probability of visiting RZs.

Again in Fig. 3.10 and Fig. 3.11, we can notice the evident impact of real mobility traces in the average number of contents stored at the nodes. In Rome, even though the RZs are highly visited areas, the speed of the cabs and the low pause time in both zones introduce a considerable decrease in the storage usage of the nodes, due to faster content acquisition and imminent loss, as we could earlier notice when analyzing Businessman's mobility case.

Still, when connections are allowed outside the RZs, we can observe a slightly higher usage of the system storage, although less than in the case in which dissemination operations are unbounded (‘$T_e = \infty$’). However, while gaining some content availability from adding connections outside RZs, the overall storage capacity is not significantly affected.

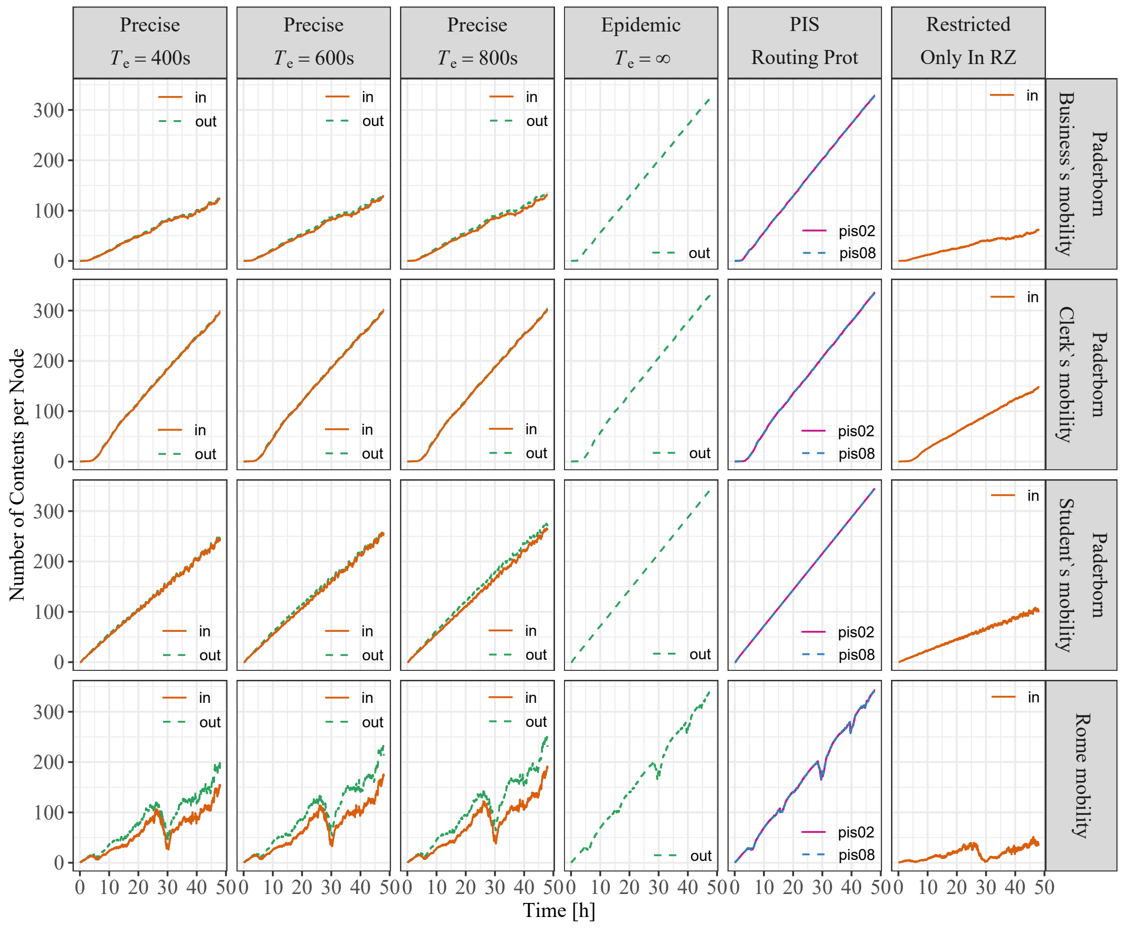


Figure 3.10: Average number of contents stored at a node (this quantity corresponds to $R(t) \cdot D(t)$). The final amount of injected contents in one simulation run is 345.

3.4.6. Number of connections

Since any forwarding policy will introduce constraints in terms of data sharing attempts, the number of connections will inevitably affect the network performance and, therefore, also the content availability. Fig. 3.12 reports the average number of connections observed during simulations. Connections are divided into three groups depending on where they take place: inside RZ_1, inside RZ_2, or outside RZs. Under the Businessman's mobility, the number of connections is lower than in the other cases for Paderborn city, which is due the lower probabilities of visiting the RZs. The same happens in Rome's subplots, despite the fact that the RZs defined in this scenario are the most frequently visited areas, they are still not as visited as in Paderborn scenario plus nodes do not remain there for long periods.

In all considered cases, even if negligible in the plots, the number of connections

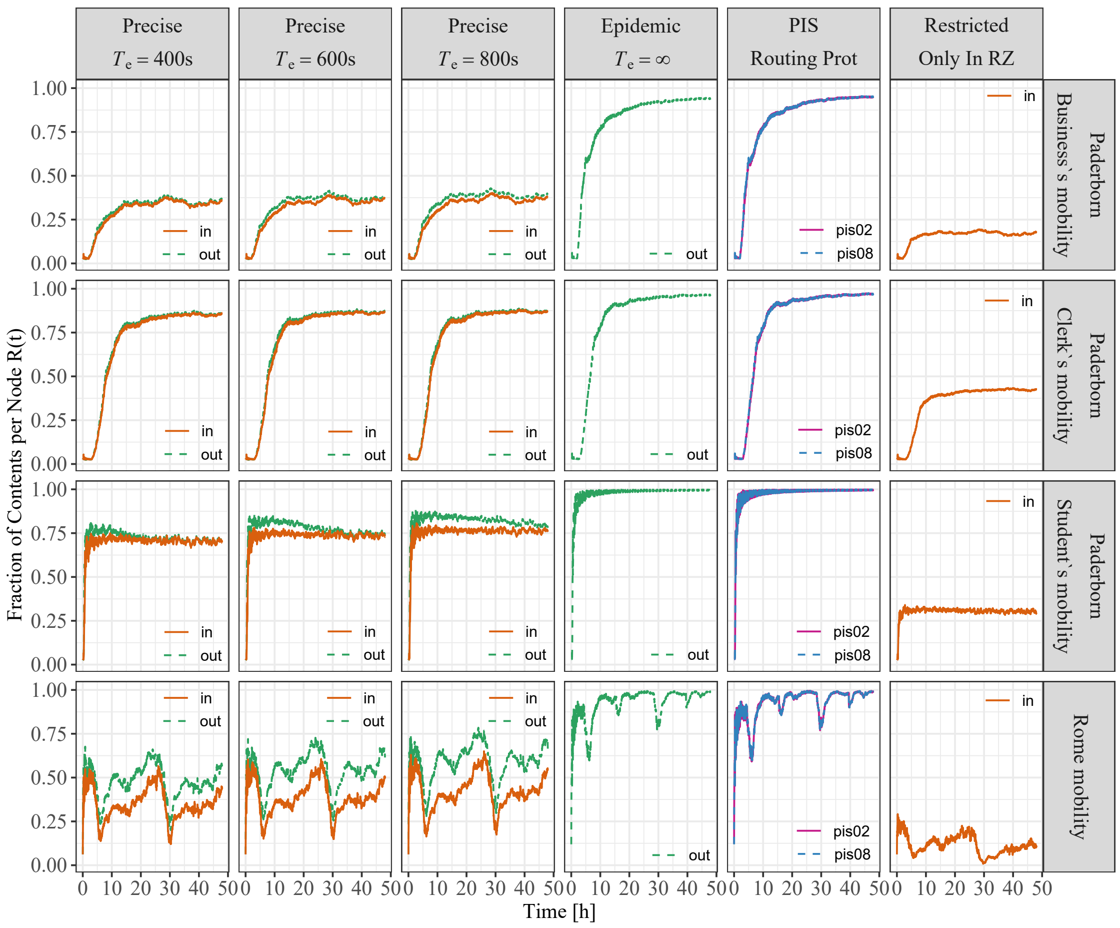


Figure 3.11: $R(t)$, mean fraction of contents possessed by a node (cf. Eq. (3.9)).

Table 3.3: Comparison of the total number of connections for Epidemic and PIS schemes with and without the restriction of exchanging contents only inside RZs

Configuration	Number of connections with restriction	Number of connections without restriction
Epidemic Business	1.1684×10^6	1.6154×10^6
PIS08 Business	1.1693×10^6	1.6167×10^6
PIS02 Business	1.1568×10^6	1.6031×10^6
Epidemic Clerk	3.1815×10^6	3.272×10^6
PIS08 Clerk	3.1819×10^6	3.2722×10^6
PIS02 Clerk	3.0559×10^6	3.1399×10^6
Epidemic Students	2.2924×10^6	2.5739×10^6
PIS08 Students	2.2932×10^6	2.5746×10^6
PIS02 Students	2.2331×10^6	2.5028×10^6
Epidemic Rome	0.9048×10^6	3.2551×10^6
PIS08 Rome	0.9058×10^6	3.2560×10^6
PIS02 Rome	0.8877×10^6	3.2289×10^6

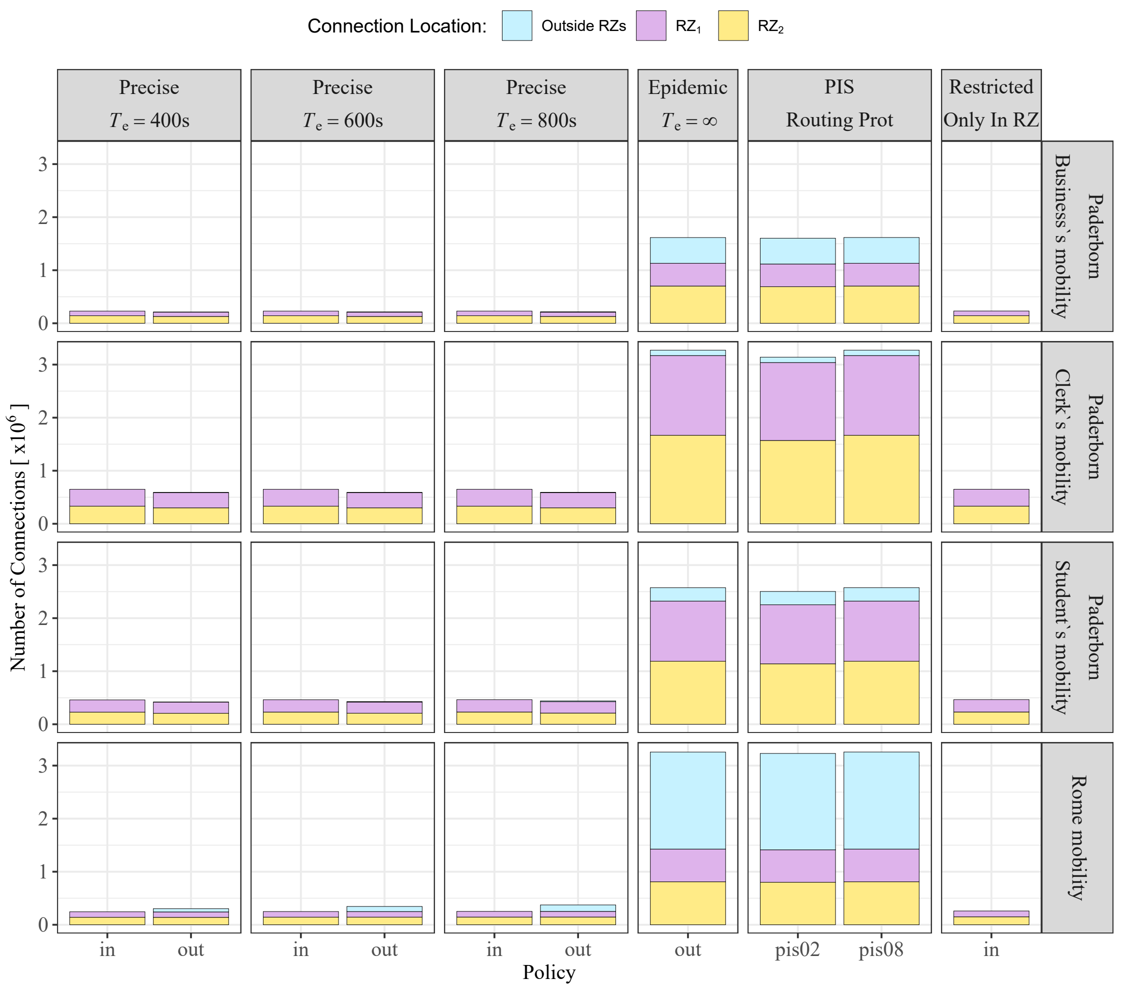


Figure 3.12: Total number of connections per simulation run in millions (average over 5 runs). Shown is a stacked plot of connections in each RZs and outside.

outside RZs increases with T_e, and it is more pronounced in Businessman's and Rome's mobility scenarios, as we can clearly see when T_e is unbounded, in which case all nodes will always attempt to establish a connection when they meet, independently from their mobility characteristics. Therefore, since in the Businessman's and Rome's scenario nodes tend to travel to the defined RZs with lower probability, having more opportunities for complete new encounters, they also end up using the most amount of connections outside RZs, even when $T_e \to \infty$. Note that the number of connections outside RZs is quite limited for finite values of T_e, which means that our scheme does not require much use of network capacity when nodes are outside RZs. Therefore, our predictive scheme performs comparably to unbounded epidemic diffusion schemes for what concerns dissemination of contents, although they require much less network resources. In some cases, the number of connections in the two RZs is not symmetric. Indeed, the RZ in which we observe less connections is the one in which dwelling points are spaced apart, out of transmission range. When the average number of nodes in each RZ is low, they end up being, with high probability, far apart to establish connections, which results in less connections than in the other RZ. Instead, under mobility scenarios where the number of nodes in the RZs is higher, various nodes will be going towards the same dwelling point and thus they will be able to connect.

Besides, to assess the importance of bounding content exchanges within or nearby RZs, we have tested the behavior of Epidemic and PIS in case they are forbidden to operate outside RZs. Table 3.3 compares the number of connections with Epidemic and PIS (with $\gamma = 0.2$ and $\gamma = 0.8$) with and without such restriction. In the table, we can appreciate that, with the restriction, the number of connections experienced decreases substantially only in case of highly dynamic mobility—e.g., in the taxi cab scenario—, although in all cases it remains much higher than what observed for PRECISE (cf. Fig. 3.12).

3.4.7. Data lifetime, losses and delivery delay

We now compare our predictive scheme to the benchmarks in terms of their ability to keep contents alive without the support of an infrastructure. Fig. 3.13 shows the average lifetime of contents that were generated between $t \approx 14 - 15\,\text{h}$, with a residual simulation duration of $\approx 33\,\text{h}$. The computed confidence intervals are small and hence we do not report them in the figure. Their value indicates that, with probability 95%, the actual average is within $\pm 0.25\%$ of the estimated average reported in the plot.

We observe that most of the contents are kept alive a bit longer than 1.5 h, for all settings. However, the baseline 'Restricted Only In RZ' scheme shows a smaller average lifetime, up to only $\approx 45\,\text{min}$ in Rome scenario, and more variability across contents and simulations. This shows that, even with only 50 nodes in the case of Paderborn, our predictive forwarding scheme, along with realistic mobility patterns, is as resilient as pure unconstrained dissemination schemes, although it requires less use of communication resources.

Losses are very infrequent, as shown in Fig. 3.14. Indeed, the number of contents disappearing is less than 1 on average, under any of the tested configurations. The computed confidence intervals are small and hence we do not report them in the figure. Their value indicates that, with probability 95%, the actual average is within $\pm 0.03\%$ of the estimated average reported in the plot.

It is interesting that most of the lost contents disappear at the very beginning of their lifetime, when only one node possesses each specific piece of content injected. Nevertheless, contents that manage to survive to that transient period remain alive until the end of the simulation, thanks to a fast dissemination. Especially, in the cases of Clerk's mobility with more nodes gathering for longer around the same areas and Student's mobility with shorter pause times and, consequently, higher number of encounters. Additionally, with Businessman's and Rome's mobility, we notice a significant reduction of content loss when we allow exchanges outside RZs.

In most cases, and especially in Rome scenario, our predictive scheme suffers much less losses than the 'Restricted Only In RZ' benchmark, although not as good as for the uncontrolled case with $T_e \to \infty$. We can conclude that the introduction of the T_e parameter in our predictive scheme plays an interesting role: by varying its value, it is possible to trade-off reliability (losses) for costs (network resources) with only a limited impact on the availability of contents in RZs.

Finally, we have also measured the content delivery delay in Paderborn and Rome scenarios. Note that, the classic delivery delay metric cannot be applied for our scheme since the behavior of content exchanges is different for each case. I.e., in our predictive scheme, nodes drop the content when they are out of the RZs for too long ($> T_e$) and that does not happen in epidemics-like configurations. In epidemic (or PIS), nodes never drop the content so only the first time they get the content is taken into account. However, in our scheme, nodes can get the content soon for the first time and drop it after a while if they are out of RZs. Then, they can retrieve the content again if the conditions allow for it. Besides, the contents are needed only inside the RZs, so what matters is that they are received by the time a node enter an RZ or shortly after. Given these circumstances, we measure the delivery time as the time elapsed since a node enters an RZ. With this metric, in Fig. 3.15 we see that cases where content exchanges are allowed outside the RZs, the content delivery delay decreases since nodes that enter the RZs already possessing the content report a delay of 0 s. Despite the fact that epidemic and PIS configurations show the lowest content delivery delay values, PRECISE presents comparable results with the smallest difference in the order of milliseconds for the most dynamic case (Rome). It also draws our attention the higher values obtained for Clerk's mobility. As stated in Section 3.4.2.1, the RZs defined for Paderborn scenario contain some attractor points towards the nodes move with higher probability and, in this case, remain for longer. This will imply that, even though nodes eventually get the content because they remain during long periods inside the RZs, they will need more time to find someone to exchange content with. In one of the RZs, the distance between some attractor points is larger than the transmission range. This means that if nodes fall into separate points, they will not be able to retrieve the content until they move again after a long pause.

Algorithm 1 Predictive forwarding scheme

```
Input: ${m_0, m_1, ...}$: shuffled list of node's neighbors
    $T_e$: maximum amount of time for data exchanging outside RZs
    $C_{te}$: node's expiration time counter
 1: if node in RZ then
 2:   for peer in ${m_0, m_1, ...}$ do
 3:     if peer not busy and peer in RZ then
 4:       dataExchange(node, peer) {Local node exchanges data with peer}
 5:       break
 6:     end if
 7:   end for
 8: else
 9:   if node not busy and node $C_{te} < T_e$ then
10:     for peer in ${m_0, m_1, ...}$ do
11:       if peer not busy and peer $C_{te} < T_e$ then
12:         node_prediction ← isVisitingRZ(node)
13:         peer_prediction ← isVisitingRZ(peer)
14:         if node_prediction or peer_prediction then
15:           dataExchange(node, peer) {Local node exchanges data with peer}
16:           break
17:         end if
18:       end if
19:     end for
20:   end if
21: end if
```

Algorithm 2 Computation of Exchange probability per RZ

```
Input: $t_n$: current time slot
    $F_k(t_n)$: current mean number of contents to exchange
    $F_k(t_{n-1})$: previous mean number of contents to exchange
    $P_k$: probability to exchange content
1: if $F_k(t_{n-1}) < F_k(t_n)$ and $P_k > 0$ then
2:   $P_k \leftarrow P_k - 0.01$ {Decrease probability}
3: end if
4: if $F_k(t_{n-1}) > F_k(t_n)$ and $P_k < 0.1$ then
5:   $P_k \leftarrow P_k + 0.01$ {Increase probability}
6: end if
```

Algorithm 3 Local storage scheme

```
Input: $r_{RZ}$: radius of RZ
    $T_e$: max amount of time for data exchange outside RZs
    $C_{te}$: node's expiration time counter
1: $pos \leftarrow getPosition()$ {Set node's position}
2: if $pos$ not within $r_{RZ}$ then
3:     if $C_{te} == T_e$ then
4:         $dropData()$ {Drop all data}
5:     else
6:         $C_{te} \leftarrow C_{te} + 1$ {Keep data and increase $C_{te}$}
7:     end if
8: end if
```

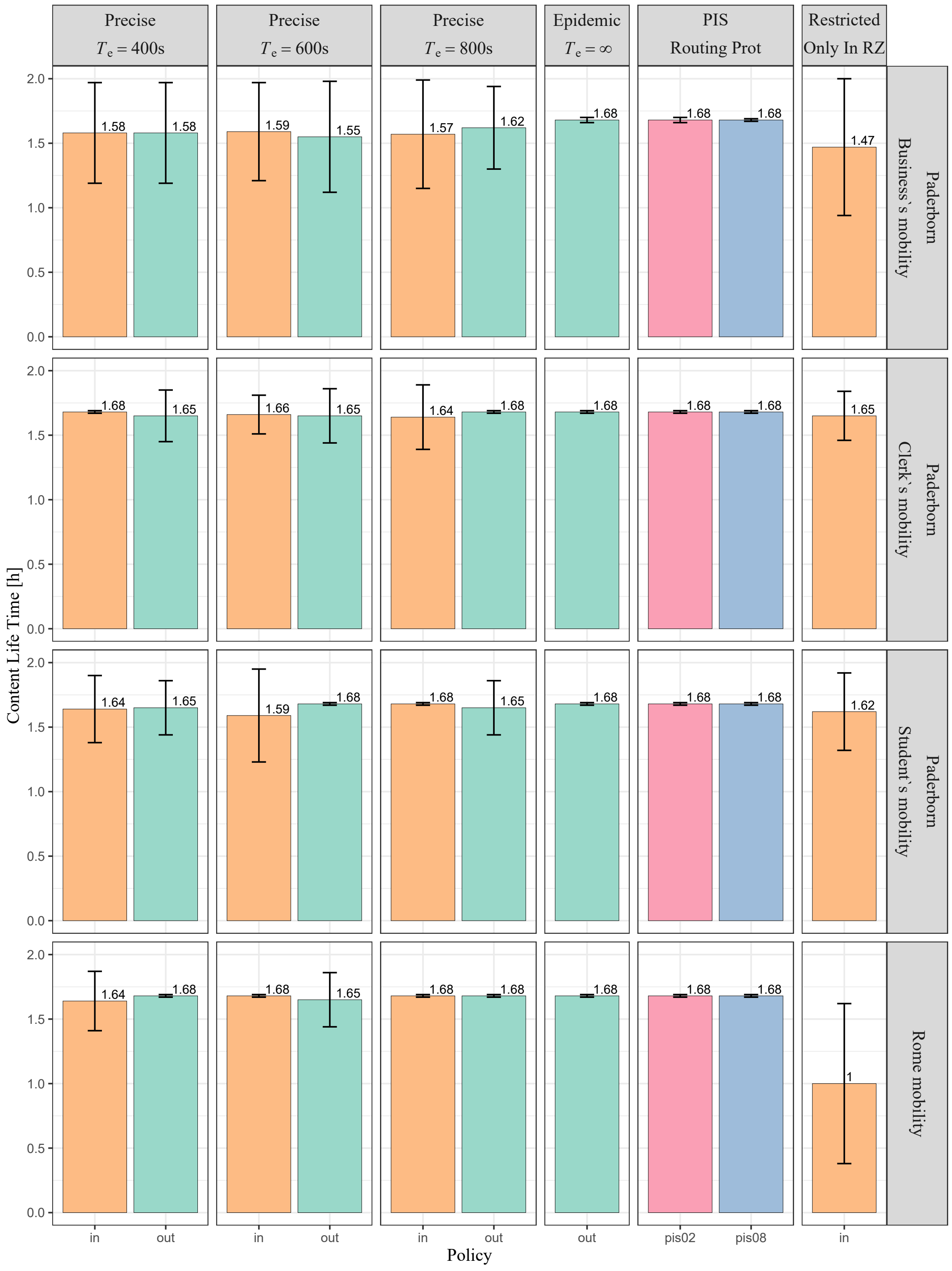


Figure 3.13: Average lifetime of contents generated between $t \approx 14-15$ h. Plotted is the mean together with the standard deviation.

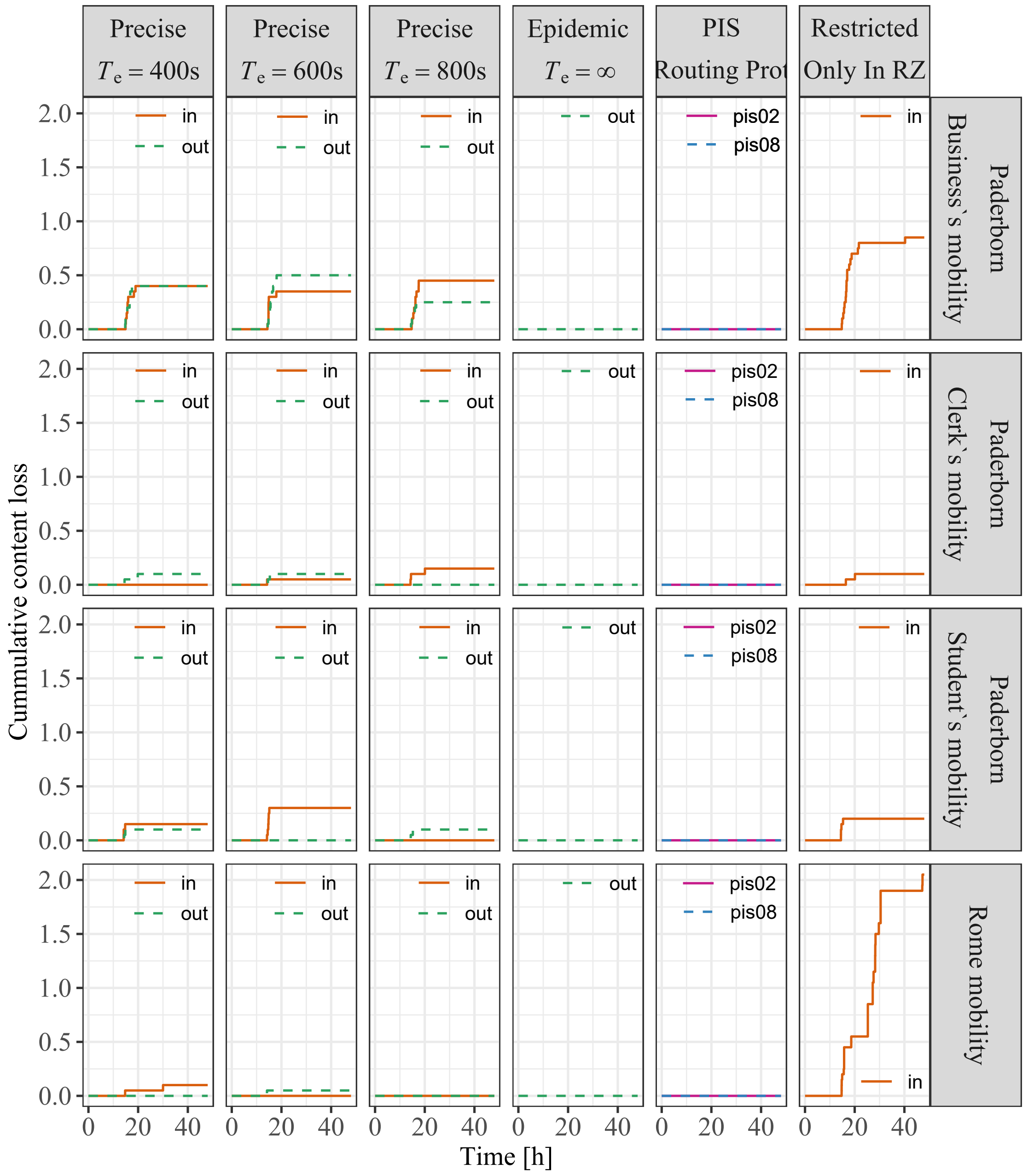


Figure 3.14: Average content pieces lost over time. Average losses are much less than a single piece of content in 6.5 simulated hours (in most of the runs, there are no losses at all).

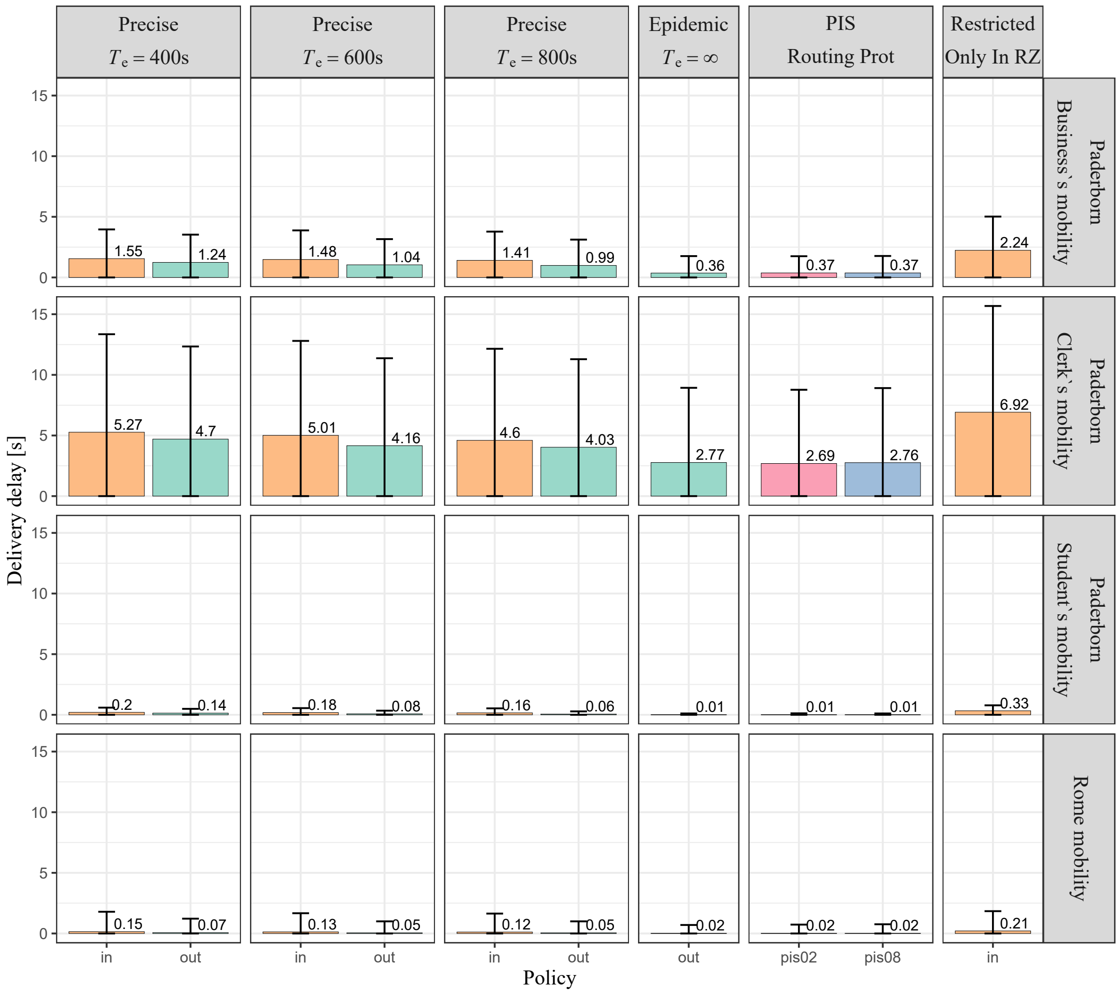


Figure 3.15: Average delivery delay of contents. Plotted is the mean together with the standard deviation.

4 Floating Content Storage Capacity

Floating Content (FC) is a paradigm for localized infrastructure-less content dissemination, that aims at sharing information among nodes within a restricted geographical area by relying only on opportunistic content exchanges. FC provides the basis for the probabilistic spatial storage of shared information in a completely decentralized fashion, usually without support from dedicated infrastructure. One of the key open issues in FC is the characterization of its performance limits as functions of the system parameters, accounting for its reliance on volatile wireless exchanges and on limited user resources. This chapter takes a first step towards tackling this issue, by elaborating a model for the storage capacity of FC, i.e., for the maximum amount of information that can be stored through the FC paradigm. The storage capacity of FC, and of similar probabilistic content dissemination systems, is evaluated with a powerful information theoretical approach, based on a mean field model of opportunistic information exchange. In addition, an extremely simple explicit approximate expression for storage capacity is derived. The numerical results generated by our analytical models are compared to the predictions of realistic simulations under different setups, proving the accuracy of our analytical approaches, and characterizing the properties of the FC storage capacity.

4.1. System Description

In this section the basic FC operations are briefly discussed, and the key performance indicators of a FC system are defined. In addition, I state the main modeling assumptions used in the analysis of the FC system storage capacity, and introduce notation.

4.1.1. Floating Content basic operation

I consider a plane over which two populations of nodes are present. The two populations comprise moving (or dynamic – the two terms are used as synonyms in this work) and static nodes, respectively. Each node is equipped with a wireless transceiver and a data storage. Every node knows its position in space.

Two nodes are *in contact* when they are able to directly exchange information via wireless communications.

At a time t_0 that defines the start of the FC system operation, a node (the *seeder*) generates a piece of content (e.g., a text message, or a picture)[1].

Within a region of the plane called Replication Zone (RZ) that contains the location of the seeder at t_0, whenever a node with content comes in contact with a node without it, content is exchanged. When a node moves out of the RZ, content is discarded.

When two nodes come in contact, a setup time τ_0 is required before content transfer, because nodes need to detect the presence of neighbors, verify the availability of content, and set up the transfer. A content transfer is successfully completed when the contact time and the channel capacity between the two nodes are such that content can be transferred in full, and the receiving node has free storage space for the content.

Every node can exchange content (i.e., send or receive) with one node at a time. Moreover, nodes do not interrupt a content exchange with one node in order to start another exchange with a different node, unless the former exchange is completed, or the two nodes are not in contact any more. However, other modes of communications, e.g., broadcast, can be easily accounted for in the proposed approach.

From this description, it emerges clearly that in FC, when proper conditions (in terms of user density and mobility, and of size of the RZ) are met, content *floats* (i.e., it persists probabilistically) in the RZ, even after the seeder has left it. In practice, content never floats forever, unless, e.g., one or more static nodes are present in the considered area, or it is possible to actively influence node mobility (such as with Unmanned Aerial Vehicles (UAVs)) to avoid content disappearance from the RZ.

At each contact, each content which is owned by only one of the two nodes, and for which there is enough memory to store it at the other node, has the same probability to be chosen for the transfer, regardless of which of the two nodes it resides on. Content exchanges between two nodes are unidirectional.

Limited modifications in the system operations can be easily accounted for in this approach, which can easily be expanded to include, e.g., bidirectional content exchanges, the effects of content broadcasting, or any specific priority scheme for contents to be exchanged, for example based on content relevance or popularity.

The system corresponding to one content item floating in its RZ according to the scheme described above is called a Floating Element (FE). In this work, I focus on systems composed by several floating elements.

With respect to the relative position of the RZ of each FE, two types of systems are considered. In *distributed floating systems*, the centers of the RZs of each FE are distributed on the plane so that RZs can partially overlap. This is typical of setups where

[1]It is also possible to consider the case of multiple seeders that simultaneously receive a piece of content, e.g., through the cellular infrastructure.

Table 4.1: Main notation used in the study

Notation	Parameter	Unit
M	Data storage capacity of each node	*bits*
D	Total node density	m^{-2}
ψ	Fraction of moving nodes	m^{-2}
$\aleph$	Ratio of static over moving nodes	
γ	Intensity (mean number of floating elements per unit area)	m^-2
C_0	Mean capacity of the link between two nodes in contact	$bits/s$
L	Content size	*bits*
R	Replication Zone (RZ) radius	m
$i\in \{s, d\}$	Node label (s for static, d for moving)	
g_i	Mean contact rate per unit area	
	involving static (resp. moving) nodes	$s^{-1}m^{-2}$
α	Linear flow	$s^{-1}m^{-1}$
τ_0	Transfer setup time	s
$\tau_s(\tau_d)$	Contact time static-moving (resp. moving-moving) nodes	s
$f_i(\tau_i)$	PDF of τ_i	

each application or end user defines its own Zone of Interest (ZOI) location. In *localized floating systems* instead, all FEs share a same RZ and ZOI. This second scenario allows ruling out the impact of randomness in RZ overlapping on the storage capacity of the system, and hence it allows investigating the maximum amount of information which can be stored in a given location (the RZ) via the FC paradigm.

4.1.2. Floating Content performance indicators

The goal of the FC paradigm is to ensure, through opportunistic replications, that the content item is delivered to a given target population of users. Hence, one of the main performance metrics for FC is content *availability* at time $t \geq t_0$, i.e., the mean fraction of nodes with content, at time t in the RZ. High values of availability in the RZ typically correlate with low likelihood of quick content disappearance, and with high probability of transferring content to nodes entering the RZ.

The specific definition of the probability of successful delivery (i.e., the *success probability* P_{succ}) varies according to the performance objectives induced by the application, and by the way in which the population of target users is identified.

For instance, when the application requests for content dissemination are related to a specific limited geographical region (ZOI) within the RZ, then the success probability can be measured as the fraction of moving nodes which possess the content item by the time they *enter* the ZOI. As an example, in an application whose performance target is to deliver an advertisement (e.g., info on a sale, or a movie trailer) or a safety warning (e.g., info on the presence of potentially infected people, or on the safety measures which have to be respected inside a building) to a given minimum percentage of the people who enter the building (a shopping center, a movie theater, a hospital), the borders of the ZOI should include all entrances to the building.

Other applications induce different definitions for the performance of the delivery task. In a cinema, there are content items in which users are likely to be interested *after* watching the movie (such as taxi services, public transport schedules, or gadgets related to a movie). In this case, the content item must be delivered to users by the time they *leave* the ZOI, and the success probability is the probability that they get out of the ZOI with the given content.

In general therefore, the expression of P_{succ} is not only a function of content availability, of the geometry of the ZOI, and of user mobility patterns, but also of the specific application supported by the FC scheme.

4.1.3. Basic assumptions and notation

I assume that moving nodes move according to a stationary mobility model such that, at any time instant, nodes are uniformly distributed in space[2], with mean density ψD. I assume that static nodes are distributed over the plane according to a Poisson Point Process (PPP) with intensity $(1-\psi)D$.

With τ_i, $i \in \{s,d\}$, I denote the node contact time, i.e., the duration of the time interval during which a static and a dynamic node (resp. two dynamic nodes) are in contact, and with $f_i(\tau_i)$ $i \in \{s,d\}$, I indicate the contact time PDF. With C_0 I denote the mean channel capacity between two nodes that are in contact.

Let g_i, $i \in \{s,d\}$, denote the mean rate of contacts per unit area between static and dynamic nodes, and among dynamic nodes, respectively.

I call *linear flow* the rate at which moving nodes traverse a segment of unit length in a same direction (e.g., from left to right of the segment), denoted by α. In order to derive an expression for α, let x denote a position in space and θ an angle, and let $\alpha(x,\theta)$ be the *angular node flow*, i.e., the rate of nodes moving in direction $(\theta, \theta + d\theta)$ across a small line segment of length ds centered at x and orthogonal to θ, divided by $ds \cdot d\theta$. The linear flow α is given by

$$\alpha = \int_0^{\pi} \alpha(x,\theta) d\theta. \tag{4.1}$$

In order to simplify notation, in what follows the mobility model is assumed to be isotropic, and hence, α does not depend on the position in space of the unit segment, nor on the specific direction of the node flow. However, note that this approach can be easily extended to more general, non-isotropic mobility models.

With $S_{i,d}$, $i \in \{s,d\}$ I denote the probability that a content transmission completes successfully during a contact with a static node, or between two moving nodes.

In what follows, without loss of generality, I assume that the RZ and the ZOI of a FE are circular and concentric, and that the radius of the ZOI is always strictly less than the

[2]The impact of nonuniform user distributions will be discussed later in the chapter, through simulations.

radius of the RZ, R. For simplicity I assume that all RZs have a same RZ radius and all ZOIs have a same ZOI radius.

Moreover, all contents have the same size L bits, and M is the size in bits of the memory that can be used for FC in each node. Note however that the proposed approach can be extended to contents of different size and to scenarios where nodes have different memory size, mainly at the cost of increasing the notation complexity.

In distributed floating systems, the centers of the RZ of each FE are taken to be distributed according to a PPP with intensity γ. In localized floating systems, the intensity γ is the ratio of the total number of FEs, over the RZ area.

4.2. A Mean Field Model of Floating Content

In this section a mean field model of FC is introduced, present in the main theorem of this work, and the validity of this mean field model over finite and infinite time horizons is discussed.

4.2.1. Content diffusion over finite time intervals

In FC, the set of nodes exchanging content within a RZ can be modeled as a system of interacting objects, in which interactions bring to changes in the state of the objects via content replication. When the number of these objects becomes large, the analytical performance study of such system becomes difficult, due to the exponential growth of the state space size.

Indeed, in order to model the temporal evolution of content diffusion and availability in the implemented system, we focus on the temporal evolution of a set of parameters, related to different node populations. If K is the total number of distinct contents in the system, for a RZ radius R at any time instant t we can associate to the system the state vectors $(N_s^1(t,R),...,N_s^k(t,R),...N_s^K(t,R))$, and $(N_d^1(t,R),...,N_d^k(t,R),...N_d^K(t,R))$ such that $N_s^k(t,R)$, $(N_d^k(t,R))$ is the number of static (resp. dynamic) nodes in the $k-th$ RZ with content at time t. Such a system can be assumed to evolve according to Markovian dynamics, and it can therefore be modelled as a Continuous Time Markov Chain (CTMC)[3].

In order to derive meaningful insight into the performance of such systems, in what follows I adopt a technique based on the *mean field interaction model* or *fluid limit* [111], hence on an approximate model of the interactions between nodes. The first approximation step of such approach is based on assuming that the following *homogeneous conditions* hold:

[3]The Markovian characteristics of the system dynamics descend from the Poisson distribution of nodes, the random choice of the exchanged content, and the independence assumptions that will be introduced later in the chapter.

Definition 1 (Homogeneous conditions). *A floating system satisfies the homogeneous conditions if:*

- *at $t = 0$ the mean number of nodes per unit surface possessing a given content is the same for all contents;*
- *at any time instant, nodes possessing a given content are uniformly distributed within the RZ for that content;*
- *the probability of a node to have a given content is independent from the probability of any other node to have the same content.*

The homogeneous conditions assumption (equivalent to, e.g., the "well stirred" assumption in chemistry [112], and to the assumption of stochastic equivalence of nodes within a same class in [113]) allows deriving simpler expressions for the evolution of the main performance parameters of the system, at the cost of neglecting spatial inhomogeneities. Note however that the presented approach can be extended to account for spatial variations as well as for possible nonuniform seeding strategies (e.g., through the notion of node class, as in [113]), though at the cost of an increase in complexity of the analytic expressions.

In order to model the temporal evolution of content diffusion and availability in the system, I focus on the temporal evolution of four classes of node populations. The first and second class are composed by those static (resp. moving) nodes possessing a given content at a given time instant. The other two classes are composed by those static (resp. moving) nodes which are *busy*, i.e., exchanging content, at a given time. Note that the busy state is not associated with a given content, but with the fact that the node is involved in an exchange of content at a given time instant, and that each node can be part of both classes of populations at the same time.

Let $\bar{N}(R)$ be the mean number of nodes in a RZ. As nodes have been assumed to be uniformly distributed at any point in time, and all RZs to be of equal size, $\bar{N}(R)$ does not depend on time, and it is the same for every RZ.

In order to apply the mean field approximation, instead of the number of (static or dynamic) nodes with content for each RZ at time t, I consider the ratio between these quantities and $\bar{N}(R)$, which is therefore the parameter used for normalizing the state occupancy in every RZ. Specifically, the following parameters are considered:

- For every radius R and content k, the fraction of static (resp. dynamic) nodes which possess k (henceforth denoted as the *availability* of content k) at time t, denoted with $a_s^k(t, R)$ (resp. $a_d^k(t, R)$).
- The fraction of static (resp. dynamic) nodes which are *busy*, i.e. exchanging content, at a given time t for a RZ radius R, denoted with $b_s(t, R)$ (resp. $b_d(t, R)$).

For ease of notation, in what follows I drop the indication of the dependency of variables on time t and RZ radius R, unless required by the context.

Parameters b_s and b_d are key in order to model the decrease of the rate at which contents get replicated successfully when the mean time required to transfer the exchangeable contents is comparable or larger than the mean contact time. Indeed, in those conditions (which correspond to the conditions in which the system is approaching the maximum amount of information it can sustain) a significant amount of content exchanges do not take place or are delayed because nodes are still engaged in a content transfer at the time in which they come in contact with other nodes. Moreover, note that the busy state is not associated with a specific content, but with the fact that the node is involved in any exchange of content at a given time instant.

As a consequence of the homogeneous conditions and of the random scheduling of content transfers, for both static and dynamic nodes at any time instant the stochastic process of the fraction of nodes possessing a given content within the RZ for that content (i.e., of the *availability* of the given content) has the same distribution for all contents, in both distributed and localized floating systems.

In what follows, for a given choice of RZ radius R, I indicate with a_i, $i \in \{s, d\}$, the mean availability at time t, averaged over all contents, for static and dynamic nodes respectively. The following result derives the PDF of the number of contents possessed by a node.

Lemma 1. *With the given assumptions, in a floating system (distributed or localized) the number of contents possessed by a static (resp. moving) node at time t, denoted as m_i, is distributed as a binomial $Bin(n, p)$, with parameters $n = \lfloor \gamma \pi R^2 \rfloor$ and $p = a_i$, and truncated in $\left\lfloor \frac{M}{L} \right\rfloor$.*

For the proof, please refer to the Appendix in Chapter 6. The fact that the distribution is binomial should not be a surprise, given the independence assumption included in the homogeneous conditions, and the Markovian assumptions necessary for the development of a mean field model. The truncation is necessary to account for the limit in the memory in end user devices, hence for the maximum number of contents that can be simultaneously stored at a node.

With $\bar{m}_i$ I denote the mean of the number of contents possessed by nodes.

On the occurrence of a contact event, a relevant parameter is the amount of *exchangeable* contents, i.e., of contents which are possessed only by one of the two nodes, and for which there is enough free memory at the receiving node. This parameter is key in determining the likelihood of a content to be exchanged on a contact event.

Let $i, j \in \{s, d\}$, and let x_{ij} denote the random variable modeling the number of exchangeable contents from a node i (static or moving) to a node j (static or moving) at time t and for a RZ radius R.

Lemma 2. *In a floating system (distributed or localized), the PDF of x_{ij} is given by the expectation with respect to m_i and m_j, of a binomial PDF $Bin(n,p)$, with parameters $n = \lfloor m_i \rfloor$ and $p = 1 - a_j$, and truncated in $\left\lfloor \frac{M}{L} - m_j \right\rfloor$.*

For the proof, please refer to Chapter 6 in the Appendix.
Let $E[x_{ij}|M = \infty]$ be the mean number of exchangeable contents, for the case in which host memory is infinite. A key parameter for modeling content diffusion within a RZ is the probability of successful transfer of a *single* content during a contact of duration τ. The following result gives an analytical expression for it, for the considered system.

Lemma 3. *The probability of successful transfer of a single content from a node i (static or moving) to a node j (static or moving) for a contact is well approximated by*

$$S_{ij} = \frac{E[x_{ij}]}{E[x_{ij}|M=\infty]} \int_{\tau_0}^{+\infty} \min\left(1, \left\lfloor \frac{(\tau-\tau_0)}{\frac{L}{C_0}} \right\rfloor \frac{1}{E[x_{ij}+x_{ji}]}\right) f_i(\tau) d\tau \tag{4.2}$$

The lemma tells that the success of a transfer when there is infinite host memory is just the probability that the duration of a contact initiated in the RZ be larger than what needed to exchange the data (this is expressed by the term in the integral). Whereas, in case the memory is bounded to $M < \infty$, it is enough to re-scale the success probability according to the average number of exchangeable contents. In order to correctly model the dynamics of information exchange over time, another key parameter is the mean time spent exchanging contents during a contact, which varies according to whether only one or both of the nodes in contact are moving.

Lemma 4. *The mean time spent exchanging contents during a contact between a static node and a dynamic node (denoted with T_s), and between two dynamic nodes (denoted with T_d), are well approximated by*

$$T_i = \int_0^{+\infty} \min\left(\tau, \frac{E[x_{ij} + x_{ji}]L}{C_0} + \tau_0\right) f_i(\tau) d\tau \tag{4.3}$$

For the proof of Lemmas 3 and 4, please refer to the Appendix in Lemma 6. The lemma expresses the fact that the average amount of time spent exchanging data is obtained by averaging the contact time, whose distribution conditional to a meeting in the RZ is f_i, under the constraint that data transfer duration is upper-bounded by the volume of data to exchange, as expressed in Eq. (4.3).

We observe that, in the spirit of the mean field approach, the derivation of these expressions is also based on a deterministic approximation, which neglects the stochastic nature of channel throughput and of the amount of contents to transfer. In Section 4.4 the impact of this as well as of other approximations on the accuracy of the model is verified.

The following result models the asymptotic dynamics of our system over finite time intervals, for large RZ areas and hence for a large amount of nodes involved in the process of diffusion of each content.

Theorem 1. *In a floating system (distributed or localized), for any initial condition* $(a_s(0,R), a_d(0,R), b_s(0,R), b_d(0,R))$ *with* $b_s(0,R) = b_d(0,R) = 0$*, for large* R*, the quantities* (a_s, a_d, b_s, b_d) *converge almost surely over any finite horizon to the solution* $(\bar{a}_s, \bar{a}_d, \bar{b}_s, \bar{b}_d)$ *(the mean field limit) of the following Ordinary Differential Equations (ODEs):*

$$\begin{cases} \dfrac{d\bar{a}_s}{dt} = \dfrac{\bar{b}_s}{\bar{T}_s}\bar{a}_d(1-\bar{a}_s)\bar{S}_{ds} \\ \dfrac{d\bar{a}_d}{dt} = (1-\bar{a}_d)\left[\aleph\dfrac{\bar{b}_s}{\bar{T}_s}\bar{a}_s\bar{S}_{sd} + \frac{\bar{b}_d-\aleph\bar{b}_s}{\bar{T}_d}\bar{a}_d\bar{S}_{dd}\right] - \dfrac{2\alpha}{\psi DR}\bar{a}_d \\ \dfrac{d\bar{b}_s}{dt} = \dfrac{g_s}{(1-\psi)D}(1-\bar{b}_s)(1-\bar{b}_d) - \dfrac{\bar{b}_s}{\bar{T}_s} \\ \dfrac{d\bar{b}_d}{dt} = \dfrac{g_d}{\psi D}2(1-\bar{b}_d)^2 + \aleph\dfrac{d\bar{b}_s}{dt} - \frac{\bar{b}_d-\aleph\bar{b}_s}{\bar{T}_d} - \dfrac{4\alpha}{\psi DR}\bar{b}_d \end{cases} \tag{4.4}$$

with $\bar{b}_d \geq \aleph\bar{b}_s$*, and with initial condition* $(\bar{a}_s(0), \bar{a}_d(0), \bar{b}_s(0), \bar{b}_d(0)) = (a_s(0,R), a_d(0,R), b_s(0,R), b_d(0,R))$*. With* $\bar{S}_i$*,* $\bar{T}_i$*,* $i \in \{s,d\}$ *I indicate the expressions in Lemma 3, respectively, with* $\bar{a}_s$ *and* $\bar{a}_d$ *instead of* a_s *and* a_d*.*

For the proof, please refer to the Appendix in Theorem 6. Note that $\forall t$, the constraint $\bar{b}_d \geq \aleph\bar{b}_s$ must be satisfied. I.e., the number of busy nodes which move cannot be inferior to the number of busy nodes which are static, as for every busy static node there is (at least) a busy moving node with which the static node is exchanging contents.

This result states that the probability of observing a difference between any point of the trajectory of the given system and the solution of the ODEs goes to zero as R (and hence the mean total number of nodes in each RZ) grows. That is, in the limit, the error made by considering a deterministic system characterized by $\bar{a}_i$, and $\bar{b}_i$, $i \in \{s,d\}$, instead of the actual system goes to zero. Moreover, at any time t, for any R, variables $\bar{a}_i$, and $\bar{b}_i$ are the expected values of a_i, and b_i, respectively. Note that I consider $b_i(0,R) = 0$ as I assume that at $t = 0$ no node has initiated any content transfer yet.

Finally, note that the focus is on settings for which no strongly connected component exists in the network graph of static nodes. This makes sense in those setups in which static nodes are deployed in a deterministic manner, e.g., to implement content seeding, to guarantee a rapid content diffusion and thus a minimum level of content availability. Nonetheless the present approach, and the ODEs in Theorem 1 in particular, can be easily extended to account for direct exchange of content among static nodes. We observe however that the only effect of these direct exchanges is a more rapid diffusion of content among static nodes, without affecting the system performance once the diffusion transient is exhausted.

4.2.2. Quasi-stationary regime

In this work, I are mainly interested in the performance of the system after *enough time* has passed from the initial seeding of the content, i.e., at times in which the dynamics of initial content diffusion are exhausted.

However, in a finite system there is always a non-zero chance of having a content item disappear from its RZ due to random fluctuations in the population of nodes which possess that content. The possibility of such an event is present in any finite floating system, which therefore for $t \to \infty$ inevitably tends towards the empty state. In the studied system in particular, this event may happen when there are no static nodes, or when a given content is not possessed by any static node at $t = 0$. As the CTMC of the system has an absorbing state (the empty state), its only equilibrium state is the empty system.

Therefore, in what follows I focus on the performance of the system in its *quasi-stationary* regime, i.e., at a time from content seeding which is *large enough* for the initial dynamics of seeding to be exhausted, and at the same time *small enough* for content not to have been absorbed yet. Indeed, this is the performance regime which is most relevant, as in any practical setting the amount of time during which a content should be stored and made available is not infinite (e.g., due to day/night patterns in vehicles and pedestrians mobility) but it is often long enough for the initial transient to have only a marginal impact on the overall performance.

The computation of an estimate of the time to content extinction based on the original CTMC is unfeasible due to the explosion in the number of the states which should be considered.

In this section, under some mild assumptions, a condition for the content to float for an indefinitely long amount of time is derived (i.e., for the quasi-stationary regime to exist), as well as expressions for the main performance parameters at times from content seeding in which the initial transient effects no longer influence the system behavior. In the numerical section I will assess the accuracy of the model, and will investigate the conditions under which the decay of the system towards the empty state has a significant impact on system performance.

Given the existence of the absorbing state for the original system (which has been denoted as $\mathcal{S}$), in order to apply the mean field approach and compute the performance parameters for the time before absorption, I consider a slightly different system (denoted with $\mathcal{S}'$). This new system is obtained from the original one by assuming that for each content, when the system is empty, content is re-seeded. Specifically, I assume that, when a content is absorbed in $\mathcal{S}$, in $\mathcal{S}'$, at rate ϵ (constant and independent on any parameter of the system) a management function selects one node (or a few nodes) at random in the RZ, and injects the content in that node(s). Note that, in the time period between content seeding and content absorption, $\mathcal{S}$ and $\mathcal{S}'$ are indistinguishable. This is confirmed by the following result, relative to the mean field limit over finite time intervals:

Theorem 2. *For any $\epsilon \geq 0$ the mean field limit of systems $\mathcal{S}$ and $\mathcal{S}'$ are the same.*

Proof:This result is due to the fact that the rate ϵ, being constant with RZ radius R, vanishes with increasing RZ radius at a rate R^{-2}, and is hence negligible in the mean field approximation, in which all neglected terms are $\mathcal{O}(R^{-2})$. ■

Theorem 2 implies that the content seeding process has a vanishing impact on system performance in the mean field regime, as modeled in the previous section. As a result, the mean field approximation results of Theorem 1 hold also for $\mathcal{S}'$. This result allows exploiting Theorem 1 in order to derive a mean field approximation for the stationary state, which is only defined for $\mathcal{S}'$. Then, thanks to the fact that the two systems are indistinguishable at times between content generation and absorption, in a regime in which content absorption is relatively infrequent (i.e., in a regime where the quasi-stationary state is defined) the mean field approximation for the stationary state of $\mathcal{S}'$ is also an approximation of the quasi-stationary regime of $\mathcal{S}$. In Section 4.4 the accuracy of this approach is assessed.

The following result establishes a relation between the stationary state of $\mathcal{S}'$ and the steady-state solutions of the problem in Eq. (4.4).

Theorem 3. *For any $\epsilon > 0$, for large R the steady-state solutions of Eq.* (4.4) *are an approximation of the state distribution of $\mathcal{S}'$ for $t \to \infty$.*

Proof: The CTMC associated to $\mathcal{S}'$ has no absorbing states, and its state diagram presents no cycles. Hence it belongs to the class of reversible stochastic processes [114].

Therefore, from Theorem 1.2 in [115] it follows that the stationary behavior of the CTMC associated to $\mathcal{S}'$ is completely determined by the solutions of the ODEs in Eq. (4.4). ■

The values of the variables $\bar{a}_i$, and $\bar{b}_i$, $i \in \{s, d\}$ at the steady state are therefore the solutions of the following system of equations derived from the ODEs of Theorem 1:

$$\frac{\bar{b}_s}{\bar{T}_s}\bar{a}_d(1-\bar{a}_s)\bar{S}_{ds}=0 \tag{4.5}$$

$$(1-\bar{a}_d)\left[\aleph\frac{\bar{b}_s}{\bar{T}_s}\bar{a}_s\bar{S}_{sd}+\frac{\bar{b}_d-\aleph\bar{b}_s}{\bar{T}_d}\bar{a}_d\bar{S}_{dd}\right]-\frac{2\alpha}{\psi DR}\bar{a}_d=0 \tag{4.6}$$

$$\frac{g_s}{(1-\psi)D}(1-\bar{b}_s)(1-\bar{b}_d)-\frac{\bar{b}_s}{\bar{T}_s}=0 \tag{4.7}$$

$$\frac{g_d}{\psi D}2(1-\bar{b}_d)^2-\frac{\bar{b}_d-\aleph\bar{b}_s}{\bar{T}_d}-\frac{4\alpha}{\psi DR}\bar{b}_d=0 \tag{4.8}$$

with the constraints

$$\bar{b}_d-\aleph\bar{b}_s\geq 0 \tag{4.9}$$

$$0\leq\bar{a}_i\leq 1, \qquad 0\leq\bar{b}_i\leq 1, \quad i\in s,d \tag{4.10}$$

$$\tag{4.11}$$

By solving it directly, it can be verified that among the solutions of the system of equations, those which satisfy constraints (4.9) and (4.10) are only two. Specifically, if the system starts from the empty state (i.e., one in which a_s and a_d are both zero), the system persists in the empty state. Consider instead the case in which the system starts from a non-empty state (i.e., a state in which at least one among $\bar{a}_s(0)$ and $\bar{a}_d(0)$ are non-zero). Let

$$y=\aleph\frac{\bar{b}_s}{\bar{T}_s}\bar{S}_{sd}$$

$$z=\frac{\bar{b}_d-\aleph\bar{b}_s}{\bar{T}_d}\bar{S}_{dd}$$

$$x=y+\frac{2\alpha}{\psi DR}-z$$

then the system converges to the unique feasible solution given by

$$\bar{a}_d=\frac{-x+\sqrt{x^2+4yz}}{2z} \tag{4.12}$$

only if the *criticality condition* $x\geq 0$ is satisfied, with $\bar{b}_d$, $\bar{b}_s$ given by Eqs. (4.7) and (4.8). Otherwise, the system converges to $\bar{a}_d=0$.

Thus, as can be seen from the study of the gradient of the system of equations, if the criticality condition is satisfied, any trajectory starting from a non-empty state converges to the unique non-empty feasible steady-state solution.

The significance of the mean availability of content over moving nodes in steady state derives from the fact that, in the mean field limit, it coincides with the mean availability of those nodes that enter the ZOI, and is therefore key in determining the success probability.

4.3. The Storage Capacity of Floating Content

As already noted, the FC paradigm can be seen as a way to implement, through opportunistic replications, a distributed information storage service, enabling *probabilistic* content persistence and retrieval in a limited area, typically with no direct support from infrastructure (except possibly for the initial seeding of content). In this section, I characterize the *storage capacity* of a FC storage system, i.e., the maximum expected amount of information which can be stored probabilistically. First, a powerful model of FC storage based on information theory is derived, which allows the computation of the storage capacity of a FC system. Then, driven by the results of the information theoretic model, I obtain an extremely simple explicit expression for the FC storage capacity in saturation conditions.

4.3.1. An information theoretic storage model of FC

In order to derive a model for the FC storage capacity in a way which is analogous to classical information storage systems, I start by considering a single FE belonging to a floating system (be it localized or distributed). For this system, let us define the read and write operations as follows.

The **write operation** consists in the initial seeding of the content within its RZ, with the goal of enabling content to persist probabilistically even after the transient of content diffusion has passed. From Section 4.2.2 we know that a seeding strategy which attributes each content to at least one node within the RZ enables the system to converge to a nonempty steady state, though in finite systems more conservative seeding strategies are often necessary in order to decrease the likelihood of content disappearance from the system during the initial transient of content diffusion.

In order to define the **read operation**, we recall that the main goal of FC is to deliver the content to a given population of users, in a probabilistic fashion, by proactively populating the local memory of the target hosts in a distributed, collaborative manner, based on opportunistic content replications between the target hosts and all the other users in the RZ. In this context, every content delivery is a read operation, as it makes available to the node the stored content. For instance, the ZOI might correspond to the location of a movie theater, and the content item to a movie trailer which we assume will start being requested when users enter the theater. Therefore, to each node entering the ZOI corresponds a content request, and hence a read operation, which is considered as failed if the node does not possess the given content. In what follows, we consider the case in which the ZOI radius is much smaller than that of the RZ, and the RZ radius large enough for content availability to be well approximated as uniform across the whole RZ. In this case P_{succ} is well approximated by the mean content availability within the RZ.

In information theory, a storage system is modeled as a communication channel [116], [117]. In this view, information is transmitted over the channel through a write operation, and it is received through a read operation. According to the operational definition given by Shannon [117], the channel capacity is the largest amount of bits *per channel use* at which information can be sent on the considered channel with arbitrarily low error probability. For the specific case in which FC is used as a storage technology, a channel use is the operation of setting a content to float in the floating area, i.e., a write operation for a content of size $L < M$ (remember that M is the node memory size). Let us denote as *stored information* of a FE the mean amount of information which can be recovered (i.e., read). Then, similarly to communication channels, if we consider the maximum of this quantity over all channel uses (and therefore content sizes), we have the following definition of storage capacity of a floating element:

Definition 2 (**Storage capacity**). *The storage capacity of a floating element with radius R is the maximum of the stored information, over all content sizes $L \leq M$.*

In what follows, we consider the floating element and its floating system to be in a stationary state. With $\bar{a}_d(R, \gamma, L)$ we denote the mean field limit of the availability of moving nodes for the stored content, for a RZ radius R. With $P_{succ}(R, \gamma, L)$ we denote the success probability at the mean field limit, which we assume is a monotonically increasing function of $\bar{a}_d$.

Theorem 4. *In a floating system in quasi-stationary regime, the mean storage capacity of a floating element is*

$$C_{FE}(R, \gamma) = \max_{L \leq M} L P_{succ}(R, \gamma, L) \tag{4.13}$$

Proof: At the mean field limit, every node in the RZ has the same probability $\bar{a}_d(R, \gamma, L)$ to possess the content, independently from its position.

In the communication channel model of storage systems, a FE can be modeled as a *packet erasure channel* [118], with packet size equal to the size of the content, L, and packet erasure probability $1 - P_{succ}$.

In such a model, every channel use is a write operation, consisting in setting a content of size L bits to float in the floating region, and erasures in the channel derive from the fact that the read operation is not deterministic. The amount of bits which can be recovered, on average, on a single channel use is hence $L P_{succ}(R, \gamma L)$.

The maximum of this quantity over all channel uses, and hence over all content sizes $L \leq M$, is the storage capacity of the system. ■

In a floating system composed by more than one floating element, the overall amount of information stored is also a function of how the RZs overlap, hence of both FE intensity γ and RZ radius R.

The following result gives the capacity per unit area of a floating system for a given FE intensity, and a given RZ area.

Corollary 1 (**Area capacity of a floating system**). *In a floating system (localized or distributed) in the stationary regime, the mean storage capacity per unit area is given by*

$$C_{FS}(\gamma, R) = \gamma \max_{L \leq M} L P_{succ}(R, \gamma, L) \tag{4.14}$$

Proof: The system can be modeled as a set of γ parallel, independent packet erasure channels per unit area, one per distinct content. The independence between the channels holds at the mean field limit and it derives from the "propagation of chaos" result, by which at the mean field limit for each user the probability to have a content is independent from the probability of having another content. For each content, the expression of the capacity of the associated packet erasure channel is given by Theorem 4. ■

We observe that the mean amount of information possessed by a node is given by $\max(M, \gamma\pi r^2 L\bar{a}_d(R, \gamma, L))$. Hence, when the success probability is defined as the probability to transit in a given location of the RZ with a copy of the content, *the mean amount of content possessed by a node is equal to the product of the area capacity and of the area within the transmission range of the node*, and upper bounded by node storage space M. It is therefore equal to the minimum between the amount of information which can be read within its transmission range, and the available storage at the node.

In a floating system, the RZ radius modulates the average amount of users, hence of system resources, dedicated to a given FE, while the FE intensity tells how many floating contents on average are sustained by the system per unit area. In what follows we are interested in how to modulate these parameters in order to maximize the amount of information per unit area stored in a floating system. We have therefore the following optimization problem:

Problem 1 (Maximum area capacity of a floating system).

$$\begin{aligned}
&\underset{R,\gamma,L}{\mathit{maximize}}\ \gamma L P_{succ}(R,\gamma,L) \\
&\ \mathit{subject\ to}: \\
&\mathit{Equations}\ (4.2),\ (4.3) \\
&\mathit{Equations}\ (4.5),\ (4.6),\ (4.7),\ (4.8) \\
&\mathit{Constraints}\ (4.9),\ (4.10) \\
&R \geq 0 && (4.15) \\
&1 \leq L \leq M && (4.16) \\
&0 \leq \gamma \leq \frac{DM}{L} && (4.17)
\end{aligned}$$

Constraints (4.5) to (4.10) allow deriving the expression of the steady-state variables $\bar{a}_i$, and $\bar{b}_i$. Constraint (4.17) derives from the fact that an upper bound to the intensity γ, and hence to the average number of different contents floating in a given area is given by the average number of nodes present in that area, multiplied by the number of contents which each node can store. As a consequence, an upper bound to the amount of stored information per unit area is DM, and it corresponds to the case in which, for each content, a single copy exists in the system, so that the system has no redundancy.

As for the content size L, we have the following result:

Proposition 1. *If* (γ^*, R^*, L^*) *is a solution of Problem 1, then* $L^* = 1$.

This result derives from the fact that, holding constant all else, the lower the content size, the lower the amount of contact time wasted in content transfers which do not complete due to the end of contact time. Moreover, the lower the content size, the higher the amount of information which each node can store in its finite memory, as the memory size is not always an exact multiple of content size. As a consequence, Problem 1 becomes a maximization problem over R and γ only, with content size equal to its minimum value $L = 1$. [4]

Finally, it can be easily shown that, for any choice of the other system parameters, there exists always a RZ radius beyond which availability decreases monotonically with increasing R. Summing up, despite Problem 1 is non-convex and nonlinear, it is function of two variables over finite intervals, hence can be solved efficiently by brute force approaches.

[4]Note that in real systems, in which per-content communication overhead is non negligible, the optimal content size is not the minimal one ($L = 1$). For those systems however, the capacity computed in the ideal case represents an upper bound to what achievable when overheads are accounted for.

4.3.2. A simplified model for capacity under saturation

As we will see in numerical results, floating systems can be in one of two regimes with respect to the amount of injected information. In what can be called the *linear regime*, stored information increases linearly with injected content, and availability is very close to one. In the *saturation regime* instead, the available bandwidth (or equivalently, the contact time available for content exchanges) becomes the system bottleneck, since it is fully used to transfer content. That is, on a contact between two nodes, the likelihood that the two nodes terminate the exchange before the end of the contact is very low, and possibly the contact duration is not sufficient to transfer all contents. In a system with infinite host memory, bandwidth (paired with contact duration) thus becomes the storage capacity limiting factor. Our mean field model of FC capacity shows that the transition between the two regimes is quite sharp, so that FC performance can be well approximated by considering each of them separately.

In order to derive a first-order approximated expression of the capacity in saturation conditions, we assume contact duration between static and moving nodes (resp. between moving nodes) to be constant and equal to the mean amount of time taken by a node to cross a distance equal to $2r$ (where r is the node transmission radius), given by $2r/v_i$ where v_i is the mean node speed for the contact between static and moving nodes, and the mean relative node speed for contacts between moving nodes. We further assume that (since the system is in saturation) increasing the amount of injected contents does not change the performance of the system.

Given these assumptions, capacity in saturation conditions can be approximated as the maximum amount of information that can be transferred upon nodes contact, given by

$$C_{FS}^{sat} = C_0 2r \left(\frac{(1-\psi)}{v_s} + \frac{\psi}{v_d} \right) \tag{4.18}$$

The results obtained with this approximation will be discussed in the next section, together with those obtained with the information theoretic model.

Numerical results will show that the simplified model provides quite close an approximation of the floating system capacity in saturation conditions predicted by the detailed model, and estimated by simulations.

It is important to stress that the derivation of this extremely simple and useful result was possible only thanks to the insight into the floating system behavior provided by the accurate mean field model developed in this work.

4.4. Numerical Assessment

In this section, we evaluate numerically the accuracy of our models by means of simulations, and we characterize the storage capacity of a floating system as a function of the main system parameters and of node mobility. We have used the FC simulator version for this work, with some extensions related to the storage capacity metrics, as detailed in Chapter 6.

We assume nodes move according to the Random Direction Mobility Model (RDMM), with reflection at the boundary of the simulation area. When two nodes are in contact, we assume the channel data rate is constant over time and equal to 10 Mbit. Unless otherwise specified, nodes have a transmission radius of 30 m, and they have unlimited memory. For localized floating systems, the simulation area is a square of side 500 m, the RZ has a default radius of 100 m and it is located at the center of the area.

At the beginning of each simulation run, nodes are distributed uniformly at random. In order to minimize the probability of content loss during the content diffusion transient, all nodes within the RZ for a given content possess it at start time. When node memory is finite, the set of contents possessed by each node at start time is a random subset (different for each node, and of total size equal to the node memory) of all those contents in whose RZ the node is located. Simulated time has been divided into equally sized slots, and their duration has been chosen so as to minimize the effects of quantization in time on accuracy of simulations, and in particular on the errors in detecting when two nodes are in range or when a node is within a given RZ. Content size has been set in such a way to have the start and end of time slots coincide with the start and end of content transfers. Specifically, it has been set to 5 Mbit in setups with node speed of 0.2 m/s, and to 100 kbit for those with a node speed of 10 m/s. Simulations have been run for a duration of 10 000 time slots, which proved to be sufficient to observe the system out of any transient, and data affected by transient effects have been discarded.

4.4.1. Baseline

In a first set of simulations, we considered scenarios with only dynamic nodes, and we measured the amount of information stored per unit area in both distributed and localized floating systems, as a function of the *injected information per unit area*, i.e., of the product of the content size times the mean number of contents per unit area which persist in the system for the whole duration of the simulation (which coincides with the intensity γ, when the resulting density of floating contents can be sustained by the systems according to conditions (4.5) to (4.10)).

As Figs. 4.1 and 4.2 suggest, two regimes can be observed.

For low amounts of injected information and with infinite host memory, resource contention among different floating contents is weak, as the mean contact time is larger

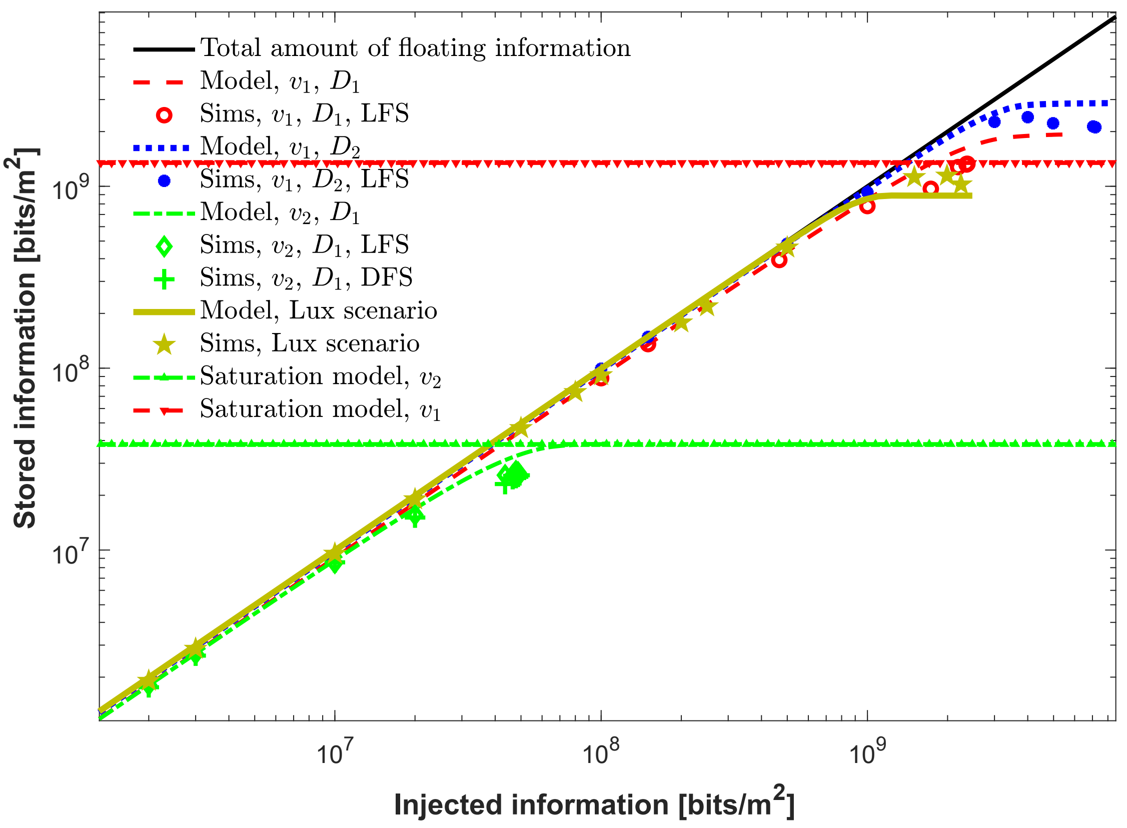


Figure 4.1: Information stored per unit area in a distributed (Distributed Floating System (DFS)) and localized (Localized Floating System (LFS)) floating system versus injected information per unit area. $v_1 = 0.2\ m/s$, $v_2 = 10\ m/s$, $D_1 = 800\ users/km^2$, $D_2 = 4000\ users/km^2$. Simulation are with a 95% confidence interval of at most 0.35%.

than the mean amount of time required to exchange contents. Therefore, each FE performs almost as if in isolation. Indeed, mean availability remains constant with increasing injected information, while stored information grows proportionally to it.

For larger values of injected information, the effects of contention (mainly, on contact time) kick in, the mean content availability decreases, and the stored information saturates. As expected, decreasing the node speed (and therefore increasing the mean contact time) and increasing the node density (and therefore the rate of contacts between nodes, and the opportunities for content replication) have the effect of increasing the saturation value of the stored information, which coincides with the capacity of the system.

As Figs. 4.1 and 4.2 show, the estimates of mean availability and of the amount of stored information derived with our mean field based approach are accurate across different values of node density, of injected information, and of mean node speed. When the amount of injected information gets close to the maximum amount which can be sustained by the system, our mean field model yields slightly optimistic results, due to the difference between finite systems and their mean field limit. Indeed, as the plots show,

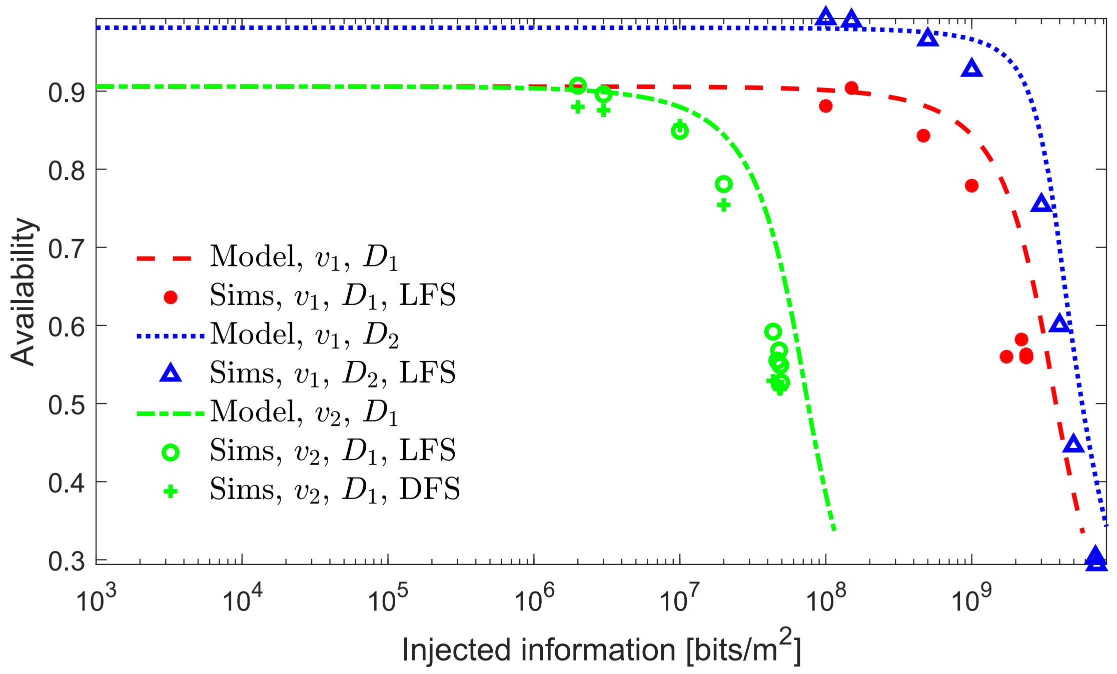


Figure 4.2: Mean availability in a distributed (DFS) and localized (LFS) floating system versus injected information per unit area. $v_1 = 0.2\ m/s$, $v_2 = 10\ m/s$, $D_1 = 800\ users/km^2$, $D_2 = 4000\ users/km^2$. Simulation are with a 95% confidence interval of at most 0.35%.

increasing the mean number of nodes in the RZ improves accuracy.

Specifically, a reason for such discrepancy is that while contact events are uniformly distributed within the RZ, in finite systems, nodes with content are slightly more numerous around the center of the RZ [13]. As a consequence, there are overall less opportunities for content replications within a RZ (e.g., at the border of the RZ). This is also the reason for the slightly more pessimistic simulation results of distributed floating systems with respect to localized ones. Indeed, in systems with distributed RZs, and particularly for low densities of FEs, much of the overlapping involves mainly the border regions of each RZ. As shown in the figures, such difference tends to decrease as the FE intensity grows.

Another reason for the observed discrepancy is the effect of random fluctuations in a finite population of nodes. Indeed, when the system gets close to those conditions in which the steady state solutions of the ODEs in Theorem 1 do not satisfy the constraints in Equation (4.9) and Equation (4.10) (that is, the conditions in which contents cannot persist in their RZs), the effects of even small random perturbations in a finite population of nodes with content gets amplified, because of resource contention and the consequent loss of efficiency of the content replication process. This brings to an average decrease in content availability, and to the difficulty in achieving those values of maximum injected

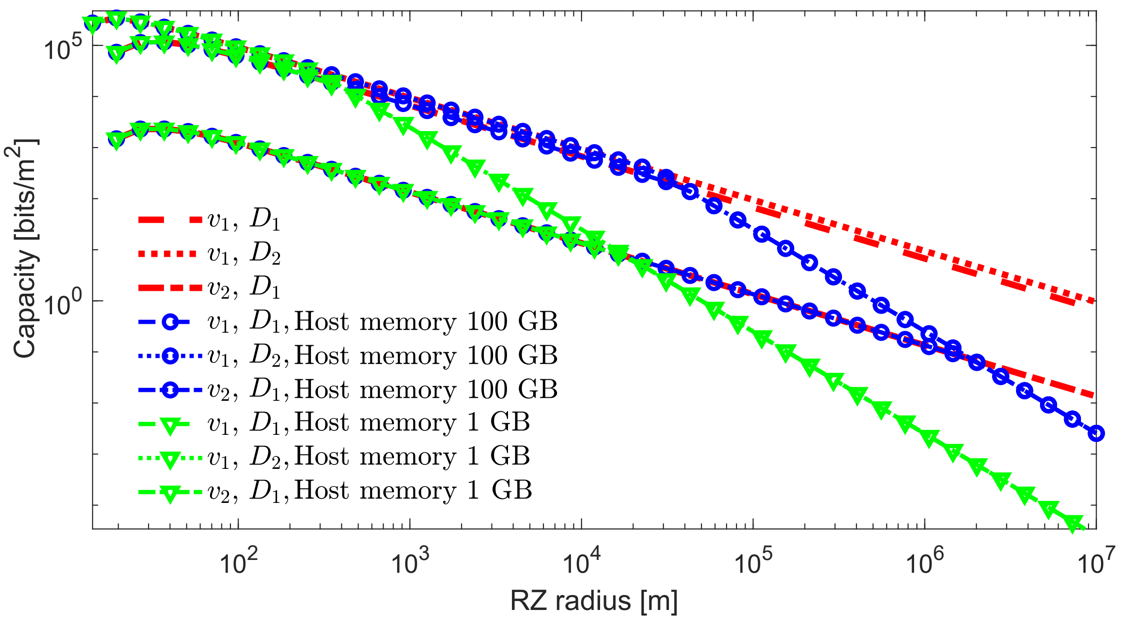


Figure 4.3: Maximum storage capacity of a floating system over FE intensity, as a function of RZ radius R, for different configurations of host memory, and with content size $L = 1$. $v_1 = 0.2\ m/s$, $v_2 = 10\ m/s$, $D_1 = 800\ users/km^2$, $D_2 = 4000\ users/km^2$.

information forecasted by the model.

In Fig. 4.1 we also report the predictions of storage capacity in saturation conditions obtained with the simplified approximate expression (4.18). We can see that predictions are extremely accurate, obviously only in the range of validity of the approximation, i.e., under saturation. However, before saturation, stored information coincides with injected information, so that it is possible to approximate the curves obtained from the information theoretic model with two lines, one where stored and injected information coincide, and one with constant stored information, at the level of the saturation capacity.

One of the key parameters affecting the mean area capacity of a floating system is RZ radius, which is related to the total amount of node memory and content exchanges dedicated to storing probabilistically a single content.

Fig. 4.3 shows the maximum area capacity of a floating system (Problem 1) as a function of RZ radius.

As the figure shows, the solution of Problem 1 is achieved for values of RZ radius only slightly larger than the minimum values below which contents do not persist in the RZ.

For lower values of RZ radius, content availability, hence maximum storage capacity, decrease rapidly, as smaller RZs imply less opportunities for contents to replicate. For values of RZ radius larger than the optimum instead, the benefits of a larger RZ (in terms of a larger amount of time spent in the RZ by nodes, bringing more opportunities for content replication, hence higher availability) are offset by the fact that a much larger amount of nodes is involved in content replication, so that the marginal utility of adding

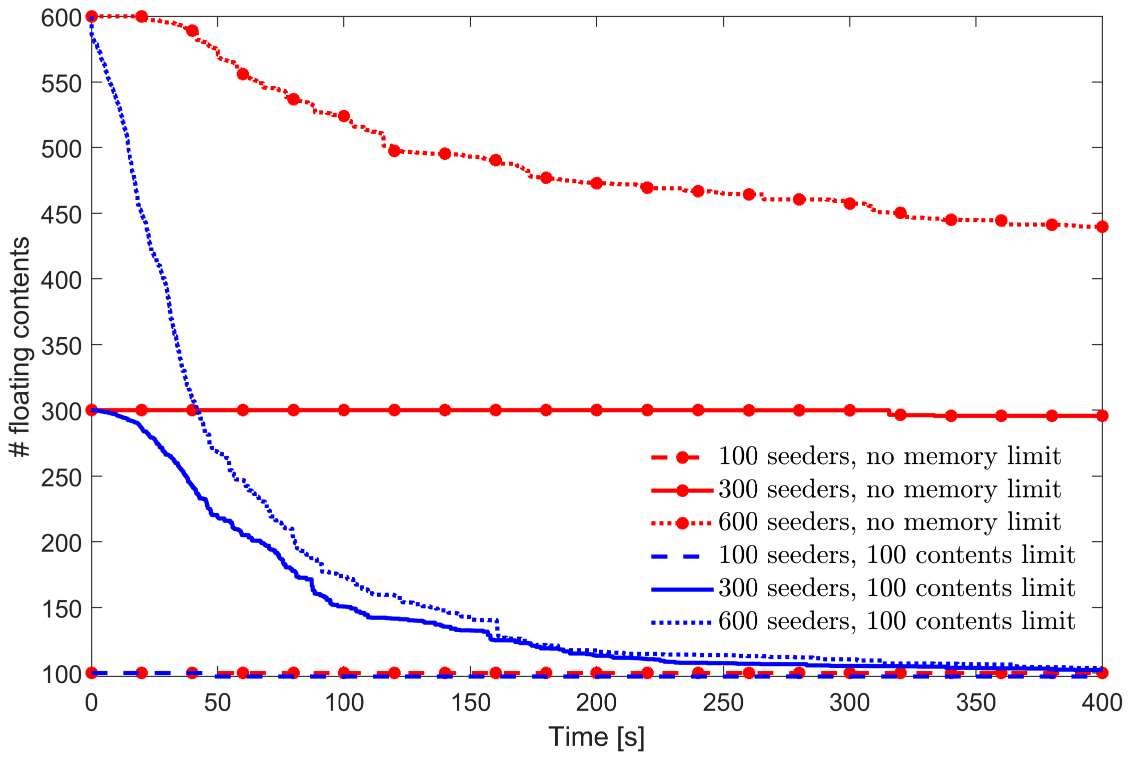


Figure 4.4: Average over 20 simulation runs of the number of floating contents over time, in a localized floating system, for different initial number of seeders. Node speed is equal to 10 m/s, and node density is 800 $users/km^2$.

more users to each RZ is negative.

By further increasing the RZ radius, the maximum stored information reaches a regime where host memory limits start to affect system performance, further decreasing the marginal utility of adding more users to each RZ.

As we have seen, for a given content size, node density and RZ radius, when the density of contents seeded in a floating system is larger than the maximum amount which can be sustained by the system in steady state (i.e., if the system operates in conditions in which there is no feasible solution to Problem 1), according to our model the system converges to the empty state. In practical terms, as Fig. 4.4 shows, this implies that a process of content disappearance from the system is initiated, and it continues until the contents which remain in the system can be sustained in the mean field regime. Similarly, when host memory is finite, if the system is seeded with a higher number of contents than those which can be stored in host memory, the total number of floating contents decreases rapidly during the initial transient of content diffusion, until it coincides with the maximum number of contents which can be stored in host memory.

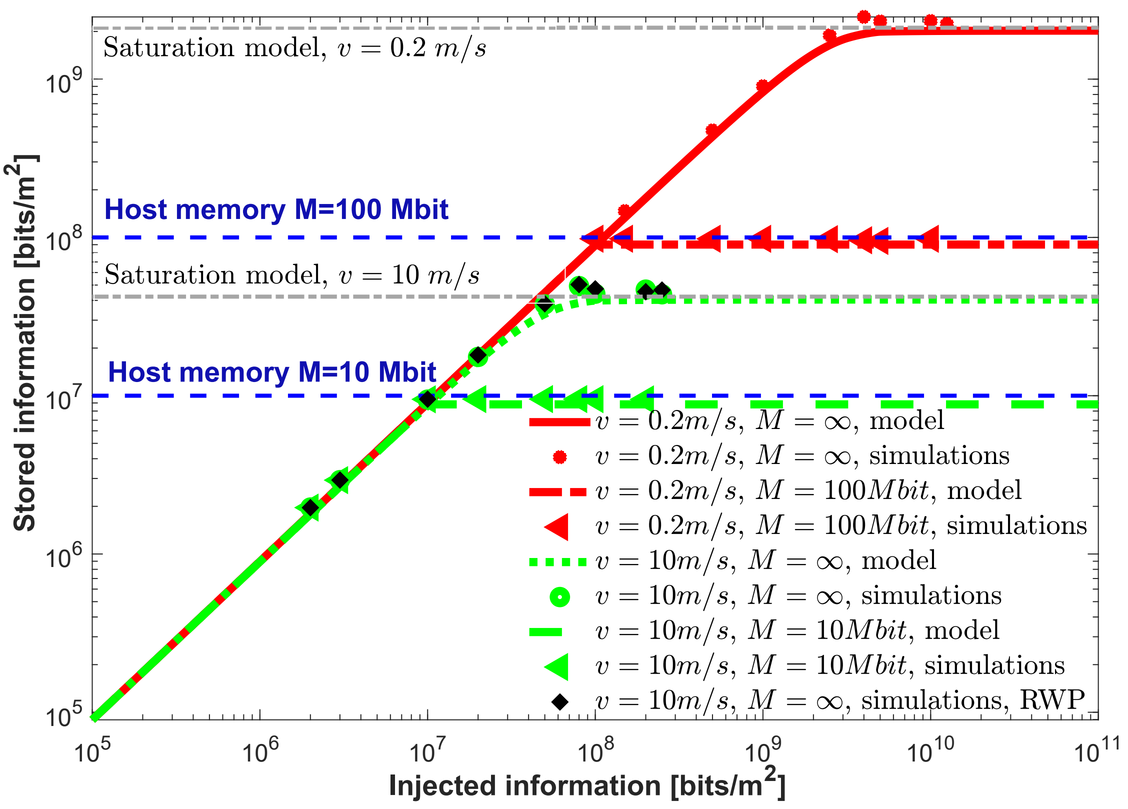


Figure 4.5: Information stored per unit area in a distributed (DFS) and localized (LFS) floating system versus injected information, with 15% of static nodes, for the Random Direction as well as the Random Waypoint (RWP) mobility model. Simulations are with a 95% confidence interval of at most 0.35%. Node density is 800 $users/km^2$.

4.4.2. Impact of static nodes

In the FC scheme, a key role is played by node mobility. On the one side, mobility induces volatility of content, since nodes that exit a RZ are allowed to discard the content associated to that RZ (and we assume they do so in the present chapter). One consequence of such volatility is an increase of the amount of resources (in terms of user memory, and of rate of content replications) required to effectively store a given amount of information. On the other side, mobility and the induced pattern of contacts allow content to replicate and thus persist in the RZ, and to be delivered to the target users. The question thus arises of what is the impact of that portion of the nodes which are not mobile on the performance of a FC probabilistic storage system.

To address this issue, we performed a set of experiments in which a varying fraction of nodes is static. Fig. 4.5 shows the amount of stored information in a floating system, when a portion of the nodes is static. Also in this new set of experiments, our analytical approach proves very accurate across a variety of settings and for different choices of system parameters.

By comparison with Fig. 4.1, it emerges clearly that the presence of static nodes allows at least a fraction of contents to persist deterministically in the RZ. As a consequence, in a system with infinite host memory and at the mean field limit, there is no upper bound to the amount of injected contents which can be kept floating. However, similarly to the case with no static nodes, by increasing the injected information (e.g., by increasing the amount of contents injected while keeping constant content size), the amount of stored information reaches a saturation value.

The point of saturation corresponds to a value of injected information at which the mean amount of time spent exchanging content during a contact is equal to the mean duration of a contact. Indeed, as the plots show, to a higher node speed corresponds a lower value of stored information at saturation (i.e., of capacity). As these results suggest, when host memory is limited, the capacity of the floating system is the minimum between the saturation value without memory limits, and the host memory.

Also in this case that includes static nodes, we report the predictions of storage capacity in saturation conditions obtained with the simplified approximate expression (4.18), which are again extremely accurate.

The impact of node speed and therefore of the mean contact time is also clearly visible from Fig. 4.6, which shows how capacity varies as a function of the percentage of static nodes in the system. As the plots show, when the fraction of static nodes is small, their effect on system capacity is negligible. When the fraction of static nodes is significant, however, the mean duration of a contact increases, and therefore, as observed already, so does the value of saturation of the amount of stored information.

Another key factor affecting the capacity of a floating system is node density, which determines the amount of resources involved in the process of information storage. As Figs. 4.7 and 4.8, by increasing node density, capacity increases (as expected) until it reaches a saturation value which is function of the fraction of static nodes in the system. The initial increase is due to the fact that by increasing node density, the rate of contact between nodes increases (quadratically, for the considered mobility model). However, for scenarios with a majority of moving nodes, an upper bound to the amount of exchanges which can take place at the same time is given by the fact that, if a node is busy exchanging contents with another one, it is not available for other exchanges until it is done. At high node densities, this decreases the likelihood for a node that just came in contact with another node to successfully engage in a content exchange.

In these settings, the reason for which the scenario with mainly static nodes performs better does not reside only on the fact that mean contact duration is larger. When dynamic nodes are only a small percentage of the total nodes in the scenario, as the rate of contacts between static nodes is zero, the overall contact rate is drastically reduced, and the saturation effects described above kick in for much larger values of node density. Finally, note that our model does not account for the bandwidth saturation which might

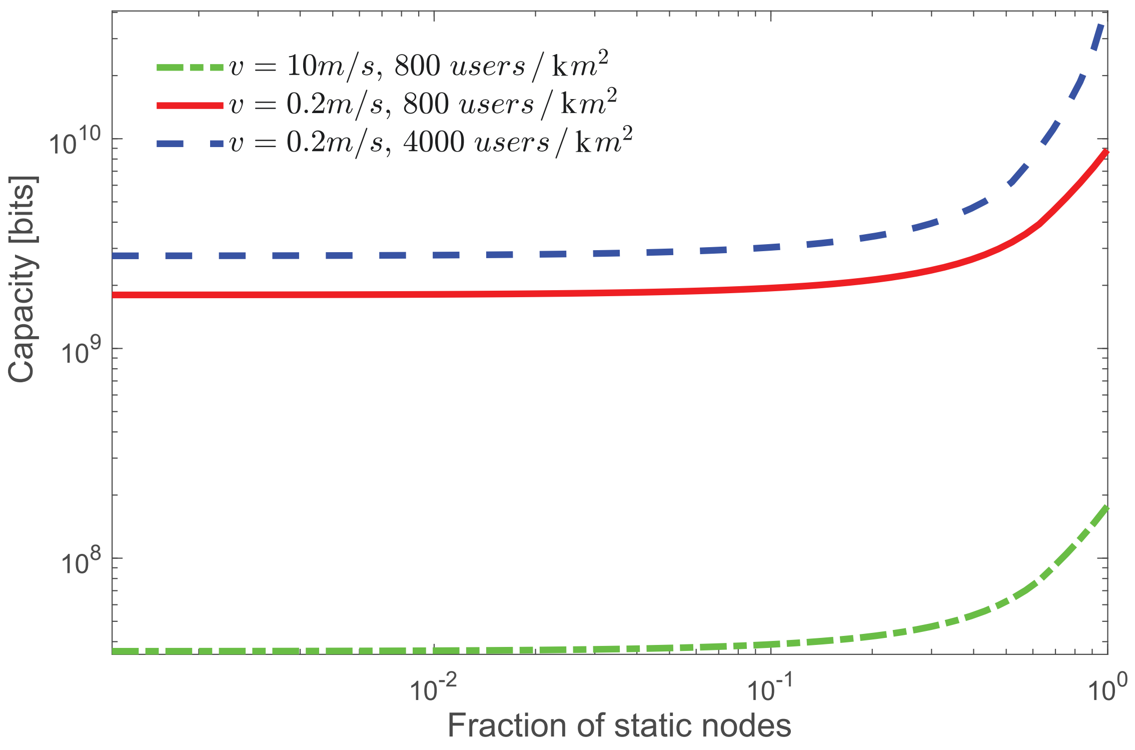


Figure 4.6: Information stored per unit area in a distributed (DFS) and localized (LFS) floating system versus injected information per unit area, with 15% of static nodes. Simulation are with a 95% confidence interval of at most 0.35%.

take place at high densities. Indeed, for large enough node densities (or equivalently, large enough transmission rates) nodes form clusters, thus boosting the likelihood of content persistence and successful replication, further improving capacity beyond the bounds given by our model. However, as already stated, as in those conditions store-carry-and-forward is not any more the main mode of communication, more efficient paradigms of communications and content diffusion than FC are available.

One of the main drawbacks of FC as a probabilistic information storage system is its reliance on the user's willingness to make resources available for the FC scheme. A possible alternative implementation of such a system may consist in delegating to a set of static nodes (*totems*) the task of content dissemination to moving nodes. This approach would avoid dealing with incentives for cooperation, at the cost of deploying dedicated infrastructure for spatial information storage and diffusion. For instance, Road-Side Units (RSUs) could play the role of totems in a vehicular communication scenario.

In Fig. 4.9 we plot the density of totems required to achieve the same capacity as in the case in which content is exchanged among moving nodes without any infrastructure.

Reported curves start at the density of moving nodes below which the content does not float. As the figure shows, a very large amount of totems would be required to achieve the same capacity achievable with opportunistic content replications among dynamic users. In

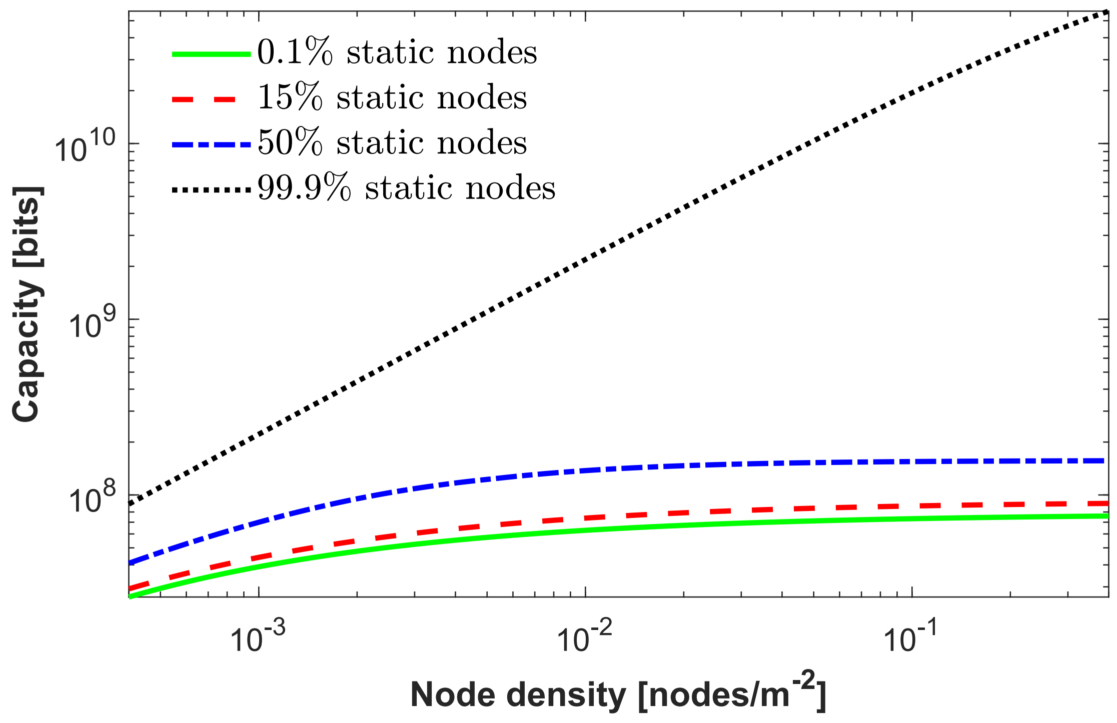


Figure 4.7: Capacity of a distributed (DFS) and localized (LFS) floating system versus node density, for different percentages of static nodes. $v = 10m/s$, tx radius $30m$, no memory limit, $R = 100m$.

practice, the number of totems should be comparable with the number of dynamic users. E.g., one totem should be deployed every two or three users, in the cases considered in Fig. 4.9. The cost of such infrastructure would clearly be unsustainable for any operator in any dense scenario like the previously mentioned vehicular communication scenario.

4.4.3. Impact of non-uniform mobility models

The results we discussed so far were derived using the RDMM, which enjoys the property of generating a uniform distribution of users, as requested by our modeling assumptions. However, real mobility patterns are more complex, and they do not share the above property, thus violating one of the assumptions used for modeling.

In order to validate the robustness of the predictions generated by our model, we now discuss results obtained with two other mobility models. The first one is the well-known RWP mobility model, that is known to generate clustering of users in the center of the simulated area. The second one is a more realistic mobility model, generated using the Simulation of Urban Mobility (SUMO) tool [119], considering measurement-based mobility traces of the the city of Luxembourg [120]. We feed SUMO with traffic data to produce realistic traces of vehicle movements over the road grid of the city. Specifically, we have considered a square area of side 4 km near the centre of Luxembourg City (see

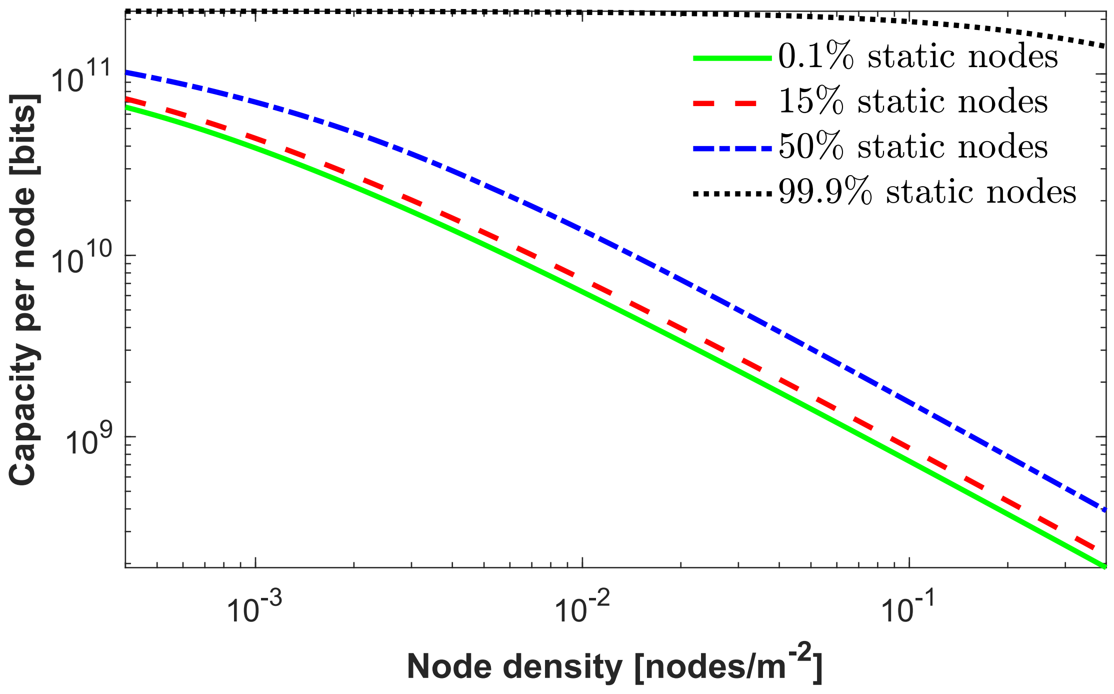


Figure 4.8: Capacity per node of a distributed (DFS) and localized (LFS) floating system versus node density, for different percentages of static nodes. $v = 10m/s$, tx radius $30m$, no memory limit, $R = 100m$.

Fig. 4.10), and the vehicular traces relative to the time interval going from midnight to 3 AM, so as to observe a low node density condition. We have considered a RZ radius of 800 m, a transmission radius of 240 m (such as the one achievable by Dedicated Short-Range Communications (DSRC) [121]), and a data rate of 10 Mbit/s. An average of 22.02 cars were present in the RZ during the given time interval, exiting the RZ at a mean rate of 0.105 cars per second, with a mean contact time of 37.5 s and generating an average of 0.959 contacts per second within the RZ.

Results in the case of the RWP mobility model are reported in Fig. 4.5, assuming the presence of 15% of static nodes. Our modeling approach can be seen to be quite accurate in spite of the nonuniform user distribution generated by mobility.

Results in the case of the SUMO mobility model over the streets of a portion of Luxembourg city are reported in Fig. 4.1, for moving nodes only. Also in this case, results can be seen to be quite accurate in comparison to model predictions generated with the corresponding parameters.

We can conclude from these experiments that our model is quite robust to variations of the node mobility pattern, and in particular, that it performs well on realistic mobility patterns which generate non-uniform user distributions.

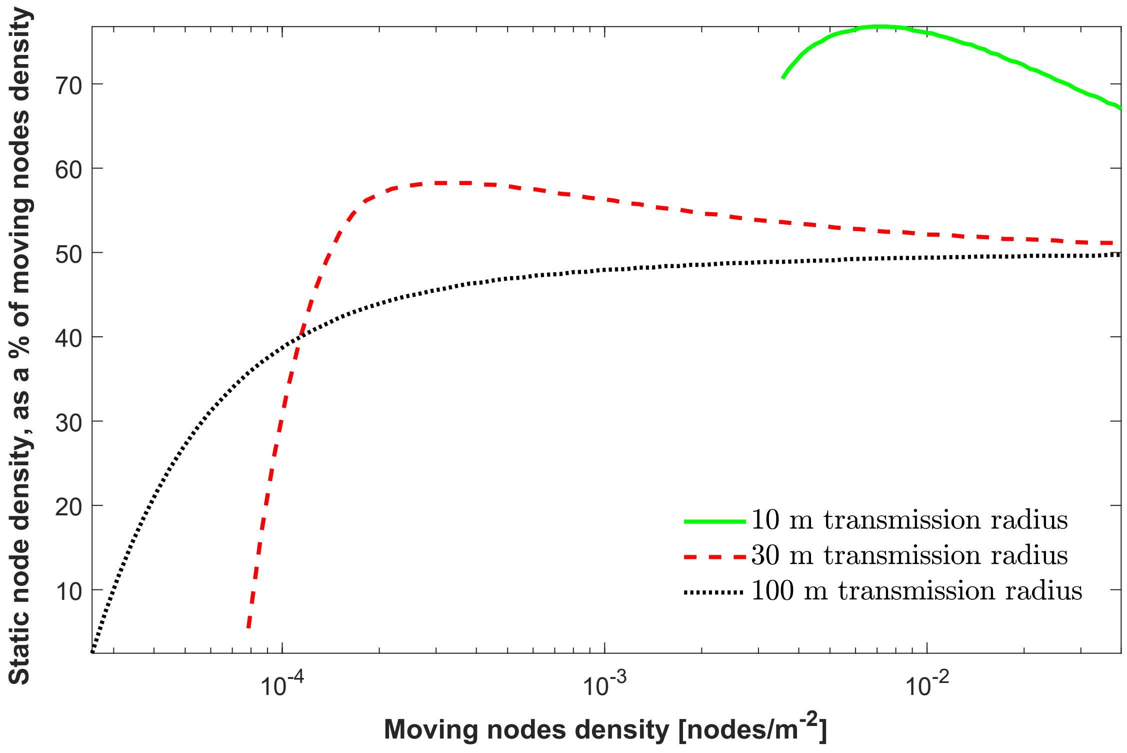


Figure 4.9: Density of static nodes required, in a floating system without content exchange among moving nodes, to achieve the same capacity as in the case in which content is exchanged among moving nodes and no static nodes are present. The density of static nodes is expressed as a percentage of the density of moving nodes. $v = 1\ m/s$, no memory limit, $R = 100\ m$.

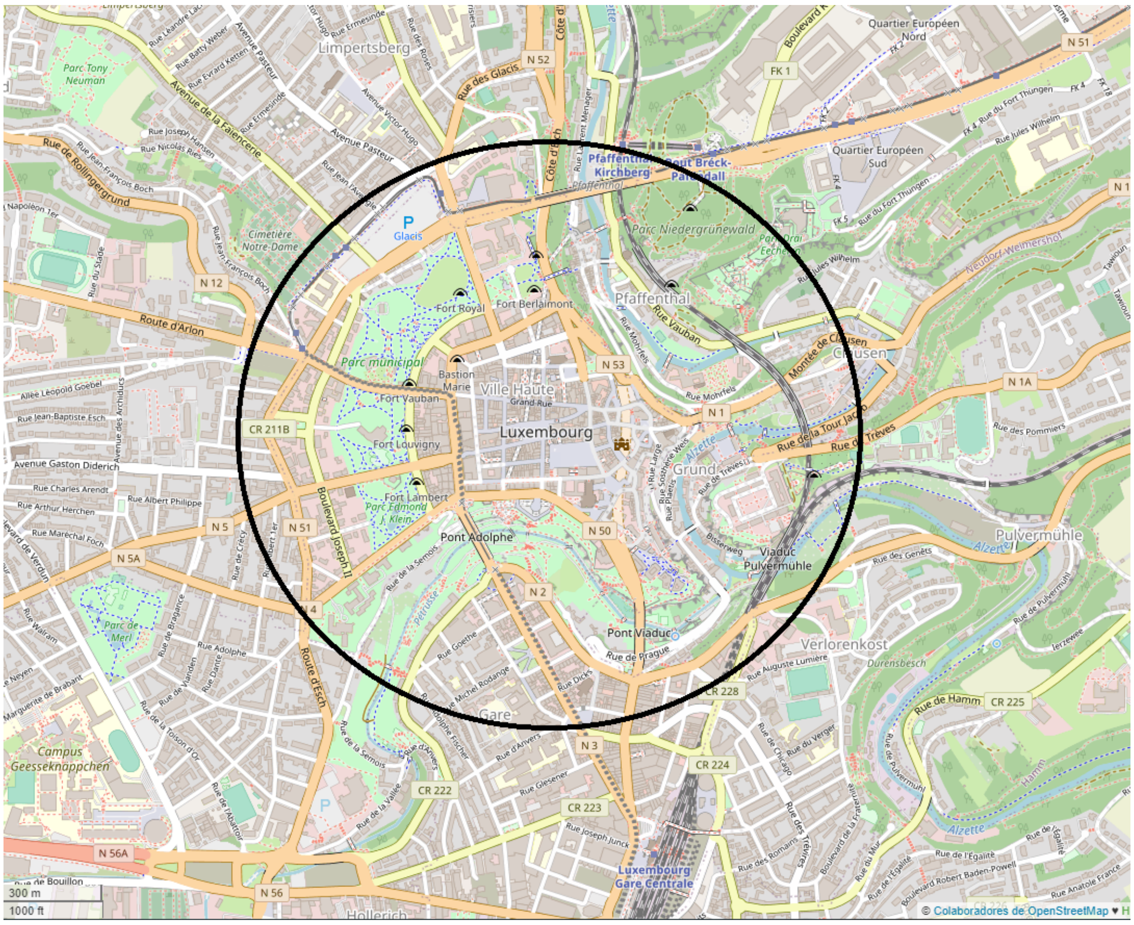


Figure 4.10: Detail of the area of Luxembourg City considered in the simulations. The circle denotes the RZ, with a radius of 800 m.

5 Floating Content Computational Capacity

This chapter focuses on the computational capacity of an infrastructure-less mobile system whose communication behavior follows the description provided in the previous chapters. Specifically, this chapter provides definitions and performance evaluation figures for Floating Gossip (FG). FG is a novel proposal of this thesis, and consists in a distributed computing and communication scheme rooted not only in Floating Content (FC) but also in Gossip Learning (GL) [100]. Moreover, in FG, the computation power is purely provided by mobile users.

The investigation complements the work presented so far in the thesis by analyzing computing aspects that are typical of Machine Learning (ML) operations (namely, learning and producing models with observed data) in a distributed environment in which information (i.e., *observations* captured from the environment and *models* trained with past observations) is moved through FC. The work leverages a generic ML distributed process and does not enter into the details of more specific applications. Indeed, the work aims to identify the general limits and trade-offs of infrastructure-less communication and computing systems used to implement ML in a cooperative way.

The goal of this chapter consists in characterizing the performance of the FG approach when one or more models circulate in a Replication Zone (RZ). Each model identifies an observable process, and there can be multiple instances of a same model, depending on the set of observations used to train the model. The training set is not necessarily the same at each node in possession of the model. Moreover, as common in distributed ML approaches, we assume that nodes can exchange model instances and merge them. The FG characterization presented in what follows takes into consideration the three key aspects, i.e.:

- Model availability, i.e., the mean fraction of nodes in the RZ with a model.
- Observation availability, i.e., the mean fraction of nodes in the RZ that possess a model instance that has been trained with a given observation.
- Learning capacity, i.e., the maximum amount of observations which can be

incorporated in models floating in the RZ. Note that the learning capacity depends on computing and storage capacity at nodes as well as efficiency of FC, and in particular on the average number of nodes whose models incorporate a given observation.

The proposed FG scheme is characterized through detailed simulation experiments carried out with a customized Python simulator which branches out of the FC simulator used in the rest of the thesis.

5.1. The Floating Gossip Scheme

Consider a dynamic environment where many events happen (e.g., a crowded theme park or a post-disaster area). Over this environment, a number of actors (humans or machines, that we term *nodes*) operate for a variable amount of time. During their stay in the environment, these nodes need to collect information that helps them make decisions (e.g., which attraction to visit or where to provide additional workforce) and/or perform some task.

The individually collected information consists in local viewpoints (termed *observations*) of what is happening in the environment at a given time. By using observations, each node builds its own partial, incomplete representation of one or more dimensions of the environment (each termed *model*), and may therefore take potentially sub-optimal decisions. Indeed, possessing a global (or at least, a less partial) model can be very valuable to improve decisions' quality.

For such an environment, a centralized scheme for observation collection and fusion in a model might raise privacy concerns, or not be viable when communication between nodes and the central infrastructure is not available (this typically happens in a post-disaster scenario). For these reasons, we consider an environment in which nodes exchange their models using Device-to-Device (D2D) when they come in proximity of one another. Then, by fusing their own model instance with the one received from another node, they generate a better model of the state of the environment. The fused model is a better representation of the environment because it incorporates a larger number of observations, some older, some more recent. In general, the utility of such a model is related to the number of incorporated observations, as well as to the time at which they have been collected.

Furthermore, we consider that models are relevant only in specific geographical areas, and that observations become obsolete and hence not relevant after a fixed amount of time. Therefore stale observations are discarded and nodes that move outside the region of interest drop the instances of the models they own.

Our approach is thus similar to the one adopted in GL, but it is based on models and observations whose availability persists thanks to an opportunistic FC scheme. We term the resulting learning scheme FG. While we use this more general term, our approach can be seen as a ML process, where each observation is incorporated into the model (through a *training* algorithm), and models are exchanged and fused (through a *merging* algorithm), as in GL. In the rest of this chapter we will refer to this case.

5.1.1. Formal definitions

We consider a population of nodes that cooperate in performing one or more tasks for the construction of a multidimensional representation of the environment in which they operate. Each dimension of the representation is associated with a model.

For the sake of simplicity we describe the system operations in the case of one model, concerned with one dimension of the environment (e.g., the road interruptions in the disaster area), but later we will also look at the case of several models evolving in parallel.

Nodes move within an area, a portion of which defines the model RZ. When entering the RZ, nodes have a *model structure*, not yet populated with fresh data (for example, a rescue team entering a disaster area has a map that does not yet contain recent information on the critical issues in the area). While in the RZ, nodes collect data (i.e., *observations*) that are used to integrate the model structure with a training operation. By so doing, each node generates and updates its own *model instance*.

Each node is equipped with a wireless transceiver, and a positioning system that gives its position in space. We say two nodes are *in contact* when they are able to directly exchange information via wireless communications.

When two nodes come in contact of one another within the RZ, they exchange their model instances (unless those already incorporate identical sets of observations) by means of opportunistic communications, following the FC paradigm. The two models are then locally merged by each node, to produce a better informed version of the model instances at both nodes.

In order to more precisely describe the process of model building and updating, we introduce some definitions. Note that they are inspired by GL terminology, but they can be generalized to other tasks based on collaborative processing.

Definition 3 (Model)**.** *A model is a description of the environment according to one dimension, which can be used to support decisions related to such dimension. For example, a model of the road system in a disaster area allows finding paths to reach a rescue location avoiding road interruptions.*

Each node maintains a *local model instance*, generally different from that of other nodes.

Definition 4 (Model instance)**.** *A model instance is a version of a model residing on a node, which has been trained with the node's own observations (which form the model local training set), and has possibly been merged with other nodes' model instances to* *incorporate their observations (the union of the local training sets of all merged models forms the model total training set).*

Definition 5 (Training)**.** *The model training operation is a transformation which takes as input a model instance, and a new set of observations (possibly just one), and it gives as output a new model instance, with a training set given by the union of the training set of the old model instance and the new set of observations.*

Definition 6 (Observation age)**.** *The age of an observation is the time since its generation.*

Definition 7 (Merging)**.** *The merging operation is a transformation which takes as input two or more instances of a model, and gives as output a new model instance, whose training set is the union of the training sets of all of the input instances.*

The process by which the new model is derived is not specified in our work. Typically, in the case of Artificial Neural Networks (ANNs), the coefficients of the model obtained through merging are derived with a weighted average of the coefficients of the merged model instances, though more complex approaches are possible. Such a definition of the merging operation seems to suggest that it produces a model instance with the same performance (e.g., in terms of accuracy) as a model instance obtained by training over the union of the two training sets. However, this is well known to be an approximation.

Every time a node produces a new observation, it uses the new observation to train its local instance of the model. As a result, in general, a fraction of the observations within a model instance training set are endogenous, i.e., incorporated in the instance through training by the node on which they reside, while the others are exogenous, that is, incorporated through merging.

Finally, users exiting the RZ discard their local model instance, as well as all collected observations.

5.1.2. The FG System

In addition to the FC communication paradigm that has been studied in the previous chapters, in this chapter the system is enriched with a new feature, the computing component. With this new approach, the nodes will be able not only to exchange contents, named models in this context, but to also perform computing tasks (e.g., train models and merge model instances). The principal idea in order to assess the system computing capabilities, consists in measuring the *learning capacity*, i.e., how many observations are incorporated in a learned model instance. In this way, we will demonstrate whether this kind of communications-computing systems are feasible and how much information can be learnt for different setups.

Table 5.1: Main notation used in the chapter.

Notation	Parameter
$A^{(\mathrm{m})}(t)$, $\overline{A^{(\mathrm{m})}}$	Mean model availability at time t and its time average
$A^{(\mathrm{o})}(\theta)$, $\overline{A^{(\mathrm{o})}}$	Mean observation availability θ time units after observation generation and its time average
g	Mean contact rate per node (s^{-1})
L	Model size (*bits*)
M	Number of models floating in the RZ
M_i	Number of nodes possessing an instance of model i
M_i^j	Binary variable indicating whether an instance of a certain model i is available at node j
N	Mean number of nodes in the RZ
$N(t)$	Number of nodes in the RZ at time t
$N_i(t)$	Number of nodes with an instance of model i at time t
O_i	Number of models floating in the RZ with observation i
O_i^j	Binary variable indicating whether a certain observation i has been trained or added at model j
$\overline{T}$	Learning capacity (i.e., time average of the mean number of observations embedded in all model instances owned by a node)
t_c	Duration of contacts between nodes
t_I	Mean time between contacts observed by a node
T_j	Mean number of observations of each model's training set
T_L	Model transfer time
T_M	Merging time
T_T	Training time
T_{RZ}	Mean time spent by a node in the RZ
W	Number of models which can be stored at a node
λ	Observation generation rate (per model)
τ_i	Generation time of observation i
τ_l	Observation maximum age
Ω	Number of observations

There are M models in total, and each node can hold a single instance of up to W models, assuming that a realistic implementation of FG will impose such practical limitation. When the number of models M is larger than W, we assume each node chooses a subset of models of interest, which is equivalent to subscribe to up to W observation channels. For the remaining $M - W$ models, the node will not accept any instance, nor it will record any observation associated to them.

As it is done in legacy GL systems, FG relies on the diffusion of local model instances built by individual nodes, rather than by exchanging observations directly. However, differently from GL, in FG learning occurs in a fully distributed way and using opportunistic data exchange without the help of a network infrastructure, which means that we rely on a FC scheme. In turn, this means that, to transfer models, we resort to D2D protocols like Bluetooth and Wi-Fi Direct. We partially neglect the overhead of such protocols and retain two main aspects: *i*) connection establishment takes time t_0, which we consider as a constant, and *ii*) data transfer is not instantaneous but depends on the quantity of data to exchange bidirectionally, though we consider the transfer rate as fixed. We also realistically consider that data exchanges occur pairwise, and it is not possible to create and manage groups with more than two connected nodes, which would be highly unstable due to mobility. We say that two nodes establishing a connection or exchanging models are *busy*, and so they cannot accept any request to connect with other nodes until they finish and immediately disconnect.

These parameters are important to determine the success of a model transfer upon a contact between two nodes. Other impart factors are the frequency of contacts observed by each node, g, and the duration of contacts, which is denoted by t_c and represents the time during which two mobile nodes are closer than a transmission range interval. The mean time necessary for the transfer of a model instance between two nodes is instead denoted as T_L, which is a function of the capacity of the channel between two nodes in contact, as well as of the size of the model instances, which we take as fixed to L bits. The order in which model instances are transferred upon a contact is important but since scheduling optimization is out of the scope of our investigation, we consider a system in which they are picked uniformly at random among the available ones.

Once exchanged, instances belonging to the same model are then fused through local computation, again without the help of an infrastructure. Received model instances are not used and propagated until they have been fused with the local instance. This requires storage and computing resources at the nodes not only for incorporating observations into local model instances (training), but also to fuse different instances of the same model, upon they are received from neighbors (merging). Storage and compute resources are used as in a system with a FIFO queue where pending tasks are stored while waiting for their execution. Training and merging times are constant and are indicated as T_T and T_M, respectively.

Moreover, since observations are sequentially generated and model instances are asynchronously acquired, observations and model instances are queued individually and then trained or merged one by one. However, an observation can be recorded simultaneously by at most W nodes. Model instances resulting from training and merging are uniquely identified by the set of observations used to build the merged instances. Therefore, two instances of the same model with a same (aggregate) training set are indistinguishable, independently on whether the observations were locally collected or embedded thanks to merging with other model instances. However, observations relevant for the performance of the application to which each model is associated are all those generated within a maximum age, which we denote as τ_l. Observations older than τ_l do not need to be incorporated into a model, and they are not counted as part of the training set. This makes sense in those scenarios in which the process which underlies the generation of the data of the observations is not stationary over time, in which case training a ML model with old data might make it unfit for producing accurate inferences.

The notation used throughout the chapter is summarized in Table 5.1

5.2. Key performance indicators

To illustrate the system behaviour, this study is based on the three main metrics described in what follows.

Model availability: model availability is described as the average number of nodes inside the RZ that possess a given model, divided by the total number of nodes in the RZ. We compute the average across all models in the scenario.

We use $M_i^j(t)$ to denote a binary variable indicating whether an instance of a certain model i is available at node j at time t. Therefore, the number of nodes possessing an instance of model i within an RZ at time t, within a set $N(t)$ of nodes in the RZ, is expressed as

$$M_i(t) = \sum_{j \in N(t)} M_i^j(t). \tag{5.1}$$

Note that $M_i(t)$ is the number of instances of model i available within the RZ and that we are considering a single RZ.

We therefore measure the availability of model i in the defined RZ at time t, denoted by $A_i^{(\mathrm{m})}(t)$, as the fraction between the number of nodes that possess model i and the total number of nodes inside the RZ. With the above, the availability at time t for model i is expressed as

$$A_i^{(\mathrm{m})}(t) = \frac{M_i(t)}{|N(t)|} \ , \tag{5.2}$$

where $|\cdot|$ denotes the number of elements in a set. To evaluate the overall scheme, we will use the mean model availability at time t, which is the average of the $A_i^{(\mathrm{m})}$ values computed over all the models of the RZ, i.e.:

$$A^{(\mathrm{m})}(t) = \frac{1}{M} \sum_{i=1}^{M} A_i^{(\mathrm{m})}(t) \ , \tag{5.3}$$

where M denotes the number of models.

Model availability is one of the key metrics of the system because it reveals whether a given scenario behaves as expected, i.e., considering a given model availability we can understand if models are floating around and how easily new nodes reaching the RZ will be able to retrieve an instance of them.

Observation availability: given a certain observation, its availability is obtained as the number of nodes in the RZ for which one of its models has been trained with that observation, divided by the total number of nodes within the RZ.

Specifically, we define $O_i^j(t)$ as the binary variable indicating whether a certain observation i, generated at most τ_l time units before the current time t, has been trained or added at node j by time t. Therefore, the number of models that have the observation i in its training set within the RZ is expressed as

$$O_i(t - t_i) = \sum_{j \in N(t)} O_i^j(t). \tag{5.4}$$

In the same way as with models, we measure the availability per observation, $A_i^{(o)}(t)$, as the fraction between the number of models (possessed by the nodes) that comprise observation i in their training set and the total number of models inside the RZ. With the above, the availability at time t for observation i is expressed as

$$A_i^{(o)}(t) = \frac{O_i(t)}{|N_i(t)|} \ . \tag{5.5}$$

where $N_i(t)$ is the number of nodes in the RZ that own the model to which the observation i belongs, at time t. We use then the mean observation availability θ time units after the generation of the observation, which is the average of the $A_i^{(o)}$ values computed over all the observations, i.e.:

$$A^{(o)}(\theta) = \frac{1}{\Omega} \sum_{i=1}^{\Omega} A_i^{(o)}(\tau_i + \theta) \ , \quad \theta \in [0, \tau_l]; \tag{5.6}$$

where Ω is the number of observations and τ_i is the time at which observation i was generated.

Observation availability gives us a clear understanding about the effectiveness of the merging and training processes of the system. Based on the amount of different observations comprised in each model training set, i.e., the availability of different observations around the RZ, new nodes entering the RZ will be able to retrieve more complete model instances with higher probability.

Now we address the following question: What is the maximum amount of information which can be learned, i.e., incorporated in a model, using FG, i.e., a decentralized, FC-based GL framework? Here, we are interested in the overall amount of information stored in model instances *in a node* in our FG system and whose age is not larger than τ_l, in the steady state for the process of model diffusion. This quantity is important because it measures the *learning capacity* of a node in the FG context. Note that this approach considers what a single node can learn thanks to FG because what matters in the system is that individual nodes learn one or more models and use them to make their own individual decisions.

Therefore, the most relevant metric to understand the learning capacity is the **mean number of observations** $T_j(t)$ of each model's training set, i.e., the amount of observations present at every model possessed by a node. The number of observations of a node j is defined as

$$T_j(t) = \sum_{i=1}^{M} O_i^j(t) \ . \tag{5.7}$$

Taking into account the average number of observations in a node, we can observe the learning capacity value of the system as a whole, i.e.:

$$\overline{T} = \frac{1}{\tau_2 - \tau_1} \int_{\tau_1}^{\tau_2} \frac{1}{N(t)} \sum_{j \in N(t)} T_j(t) dt, \tag{5.8}$$

where the result is averaged over a sufficiently large interval $[\tau_1, \tau_2]$, after the system reaches the steady state (i.e., after a transient period to be accounted for when injecting a new model).

There are a number of parameters that can affect the learning capacity of the system, and which we will study in Section 5.3. Since FG has both connectivity and computing components, parameters affecting either of the two components will be relevant. For what concerns connectivity, the parameters that affect the capacity of FC also affect FG. Thus, the density of nodes, the radius of transmission and the other parameters studied in the previous chapters will also impact on the performance of FG. Besides, the number of models indirectly affects the size of data exchanges, so it also contributes to generate load on the communication front. This is one of the most relevant parameter because the more models we inject in the system, the larger the number of instances that will be replicated when exchanges between nodes occur. Moreover, the size of the models is also key to determine the load of the communication component of FG. Both the number of models and the model size also contribute to generating the computing load of FG. Specifically, the higher the number of models and their size, the higher the computing load and the probability that computing tasks get delayed because merging and training task share a common buffer and a common server at the host node. In addition, observation generation rate, as well as merging and training times are crucial parameters that will directly affect the computing power of our system. By modifying these values we will be able to discern the most suitable processing resources that such system needs to employ to achieve the desirable learning capacity. Note also that communication and computing component feed each other in the FG system. Indeed, when communication is limited (e.g., in case of low densities of nodes or low transmission rates), the computing load due to merging model instances goes low, and so goes for the learning capacity of the system, which becomes communication-limited. Similarly, when the computing power is too low for the tasks to run (e.g., causing large delays in making new model instances available), the communication channel becomes underutilized and the learning capacity becomes computation-limited. Therefore, in a real system, any of its two components, i.e., communication or computing, can become a bottleneck for FG.

5.3. Performance Evaluation

In this section, we evaluate numerically the performance of the FG system by means of simulations. To this aim, I have used the FG version of the FC simulator used in the rest of the thesis, which allows to evaluate the fundamental factors that affect the learning capacity of FG. The description of the simulator used for FG is contained in Chapter 6.

5.3.1. System simulation and setup

The simulator implements FC for the exchange of model instances, which is done like for contents in a normal FC systems. For this work, we have applied a restrictive data transfer policy in which nodes can only store and exchange models while they are inside the RZ. When nodes leave the RZ, all models and pending models, that are queued to be trained or merged, will be erased from node databases.

In the simulations, we consider a rectangular region which embeds a single RZ. When a new observation is generated, only one node in the RZ, chosen uniformly at random every time, receives it. We define this node as the "seeder" and such node will also obtain a model instance related to such observation in case it does not posses it before observation generation time. Within the RZ, nodes that come into contact, and are not already enrolled in another connection, will be able to connect and exchange the missing model instances in the peer node. After model exchange or observation generation, each node holds a FIFO queue of tasks (pending model instances) to be sequentially computed according to predefined merging and training times. Tasks are never dropped unless the node leaves the RZ, but can suffer large delays before being completed.

At the beginning of each simulation run, nodes are distributed uniformly at random, then they move according to the Random Direction Mobility Model (RDMM) model, with wrap-around at the boundary of the simulation area. When two nodes are within transmission range, the channel data rate is constant over time. Simulated time has been divided into equally sized slots, whose duration has been chosen so as to minimize errors in detecting when two nodes are in range or when a node is within a given RZ, according to the input parameters used in the simulations settings.

The simulator introduces the following approximations on the behavior of FG: for simplicity, the results reported in what follows have been computed by considering that the length of model instances do not depend on the number of observations used to train the model. Models have a predefined size and trained observations are only considered when exchanging models, i.e., a node will exchange a given model instance to a peer node only if the model instance has observations in its training set that are missing in the peer node model instance. Training time and merging time are also considered as constant values, which means that the complexity of training and merging operations is barely or not affected by the description of an observation and the structure of the model. This occurs, for instance, for distributed ML algorithms that need to exchange and update a fixed number of parameters. Most of the times a training process will be preceded by a merging process since the most common case is that a node already possesses an instance of the same model which will have to be merged with the new received instance. It is important to note that, there is only one case in which this merging process is definitely not needed, when a seeder generates an observation and it does not possess the model instance belonging to that observation. In this specific case, the seeder will directly train a default model instance with the generated observation.

Unless otherwise specified, the simulation area is a square of side 200 m, the RZ has a default radius of 100 m and it is located at the center of the area. Observations are generated at a rate $\lambda = 0.1\,\text{obs/s}$ per model, and inter-arrival time of observations is exponential. Each observation has a lifetime of 300 s. Therefore, 30 fresher observations are generated, on average, per model, during the lifetime of an observation. The channel rate is equal to $10\,\text{Mbit}/s$ when nodes are within 5 m, and it is $0\,\text{Mbit}/s$ otherwise. There are 200 nodes in total, which makes $N \simeq 157$ nodes in the RZ in total, on average. The speed of nodes is 0.5 m/s. On average, the sojourn of a node in the RZ lasts $\sqrt{2}$ times the radius of the RZ divided by the speed, which means about $T_{\text{RZ}} \simeq 282$ s in the adopted settings.

For model size we have chosen different values depending on three configurations, using 1 to 10 models, which span very different communication and computing loads. We also report results of experiments with small and large training and merging time, so as to be able to focus on communication limits (when the processing time is short) or on computing limits (when processing time is long). All the input parameters of the simulator are shown in Table 5.2.

We count on 20 simulation runs per configuration and each run has been executed for a duration of 10 000 time slots, which proved to be sufficient to observe the system out of any transient, and data affected by transient effects have been discarded.

5.3.2. Numerical results

Fig. 5.1 shows the time average of the model availability $A^{(\text{m})}(t)$, computed over the last 30% of the simulations, for increasing values of model size, with short training and merging times. In this case, for the three different configurations (with 1, 3 and 10 injected models) the model availability starts at high values, between 0.75 and 0.8, depending on the number of models. This happens because, for small model sizes, transfer times are much shorter than contact times, on average. With a model size of 10 000 bits, and a data rate of 10 Mb/s, the transfer time is between 1 ms and 2 ms, depending on whether transmission is unidirectional or bidirectional. This has to be compared with an average contact time t_c of about 6 s, and an average inter-contact time (g^{-1}) of about 40 s, as

Table 5.2: Input parameters

Parameter	Value
Number of models (M)	1, 3, 10
Model size (L)	0.01 Mb - 100 Mb
Observation generation rate (g)	0.1 obs/s
Total number of users	200
User Speed	0.5 m/s
Slot length	0.5 s
Number of RZs	1
Transmission range	5 m
Radius of replication	100 m
Simulated area	200 m × 200 m
Channel rate	10 Mb/s
Memory limit	infinite
Merging time (T_M)	2.5 s, 25 s
Training time (T_T)	5 s, 50 s
Simulation length	10 000 slots

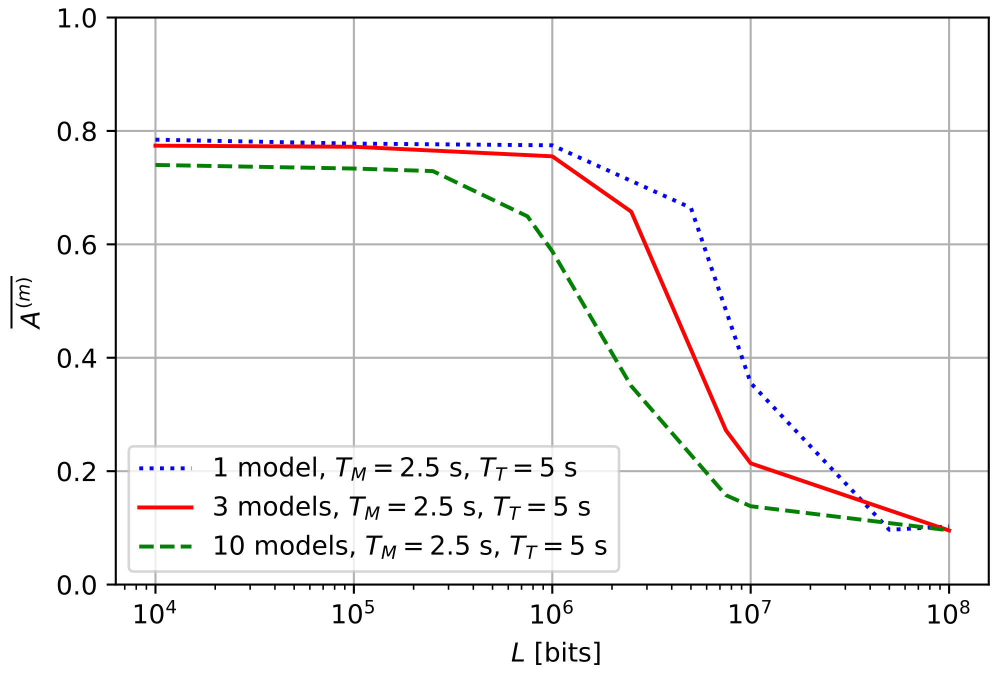


Figure 5.1: Model availability with low training and merging times.

observed experimentally in the simulations.[1] Under these circumstances, observations are mostly acquired via model exchanges and merging rather than via direct training. Indeed, note that, with the observation generation rate of 0.1 obs/s per model, an average sojourn

[1]Indeed, a standard analysis of the RDMM mobility model would tell that there should be, on average 1.96 contacts per second in the RZ, each contact involving 2 nodes. Thus, it will take $N/(1.96 \cdot 2) \simeq 40$ s for all $N \simeq 157$ nodes in the RZ to be involved in a contact.

time of 282 s in the RZ, and 157 nodes in the RZ, on average, each node will receive 0.18 obs/model while in the RZ. Being the generation of observations a Poisson process, we also know that the probability that a node receives no observation is about 0.84 and the probability to receive exactly 1 observation is about 0.15.

With an extreme simplification, we can estimate the average model availability in the case of one model by looking at the probability of acquiring the model within the first 3 contacts (i.e., the probability to acquire the model only after more than 3 contacts becomes negligible). Assuming that the contacts occur at deterministic spacing, i.e., every $g^{-1} = 40$ s in our case, and accounting for time averaging, the model availability obeys the following equation:

$$\overline{A^{(m)}} = \sum_{k=1}^{3} \overline{A^{(m)}} \left(1 - \overline{A^{(m)}}\right)^{k-1} \frac{T_{\mathrm{RZ}} - k/g - (T_M + (T_M + T_T)/2)}{T_{\mathrm{RZ}}}, \tag{5.9}$$

where the merging time T_M is 2.5 s and the training time T_T is 5 s. Discounting half of $T_M + T_T$ from the available time is done to account for the fact that a merging task can be queued if the node is busy with computation when a model exchange occurs. After factorizing out the term $\overline{A^{(m)}}$, the above expression reduces to a second order equation with only one solution in the range $(0, 1)$. In particular, for the case of the experiments reported in Fig. 5.1, the solution of Eq. (5.9) is $\overline{A^{(m)}} \simeq 0.79$. This result matches well enough the experimental value reported in the figure for one model. With more models, the model transfer and merging times grow linearly, which implies a delay in merging models, hence a reduction of time over which the models are available. This effect is indeed observed in the figure.

At the other extreme, with very large models, contact times are not sufficiently long for model transfer. Models can only be acquired with the training of endogenous observations. As mentioned above, each node will receive 0.18 observations per RZ traversal, and each observation, in practice, will be trained into a different model since the probability to receive more than one observation per node is negligible. Considering that the observation will be received, on average, for half of the sojourn through the RZ, and training time is negligible with respect to the sojourn time, we should expect an average model availability of about 0.09. This value is very close to what we see in Fig. 5.1. Moreover, with such a short training time, the result applies independently of the number of models.

Finally, the transition between the two regimes can be expected to happen before model sizes are such that transfer times are comparable to contact times. In the case of Fig. 5.1, during a contact time is possible to transfer between 30 and 60 Mb, and indeed we can see in the figure that transitions happen between 6 and 60 Mb, in the case of one model. With more models, the transition moves to the left because of the increased volume of data to be transferred, which roughly scales with the number of models.

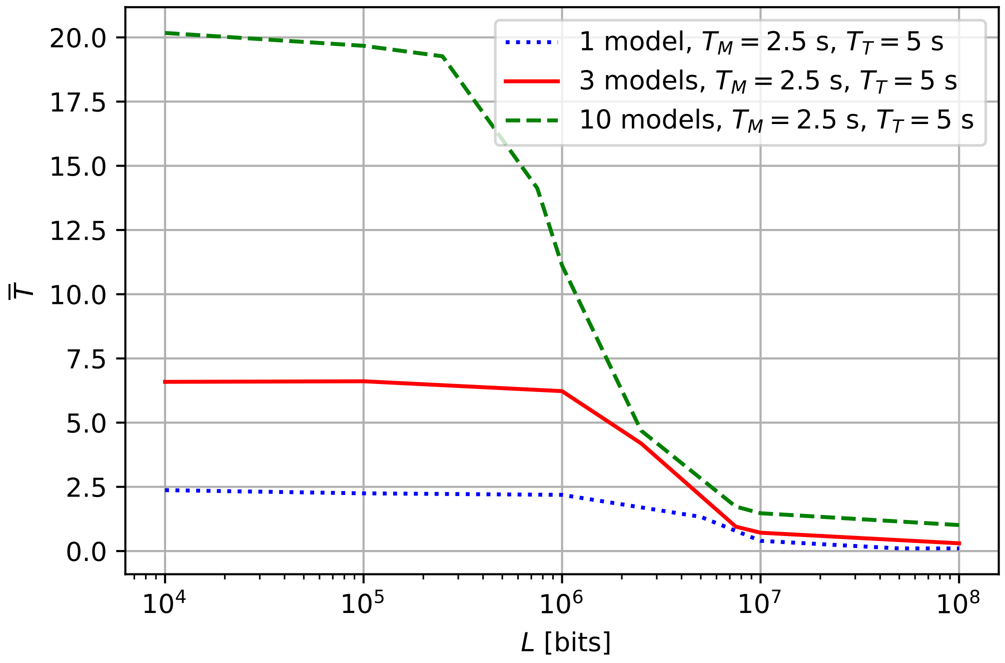


Figure 5.2: Learning capacity with low training and merging times.

The learning capacity that corresponds to the average model availability discussed so far is depicted in Fig. 5.2. In the figure, the time-average of the number of observations per node is taken over the entire simulation. The trend shown in the figure is similar to the one of average model availability, and in particular the transition between high and low capacity regions occur like for availability metrics. It is therefore full clear that relying on communication though an FC system is doable under low/medium loads, and that FC introduces a sharp phase transition on ML and FG performance. The impact of computing load, which in the figures discussed so far is visible in the differences between curves computed with a different number of models, is much less disruptive. Indeed, notwithstanding the fact that here the values reported are cumulative values over the number of models M, it is interesting to observe that the performance metrics do not scale with M but slightly change their behavior.

The values of $\overline{T}$ on the right part of Fig. 5.2 correspond to the equivalent values of model availability (see Fig. 5.1) multiplied by the number of models. This is due to the fact that models are only acquired by endogenous observation and training, and, as commented before, each model contains only one observation with high probability.

Instead, for low model sizes (left part of the figure), we know the average fraction of nodes with model, and we need to estimate the average number of observations per model at a generic node, so as to estimate the values reported in the figure. Neglecting the

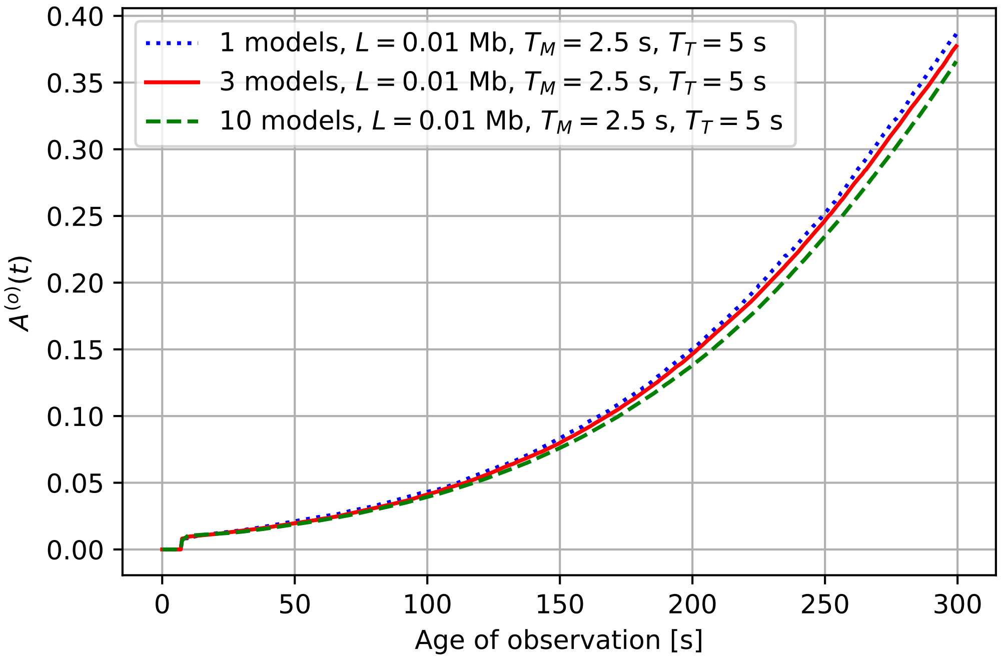


Figure 5.3: Observation availability over time with short training and merging times and $L = 0.01$ Mb.

queuing delay of processing tasks, which is possible only for small values of T_T and T_M and for a limited number of floating models, the number of nodes at which each observation is available can be approximated as an exponentially growing function of the observation lifetime. More specifically, from the moment an observation is integrated in a model for the first time, which happens, on average when the node training the observation is halfway through the RZ, the number of nodes with that observation tends to double every inter-contact time $g^{-1} \simeq 40$ s. However, we must account for the fact that the observation diffusion time starts with a delay equal to the merging plus training times $T_M + T_T$; besides, nodes exit the RZ with a rate of $T_{\mathrm{RZ}}^{-1} \simeq 1/282\ \mathrm{s}^{-1}$, and hence drop all their observations. Therefore, in g^{-1} s, the fraction of nodes that leave the RZ is $1/(g\,T_{\mathrm{RZ}}) \simeq 0.14$. This fraction applies to all nodes, and in particular to nodes with the considered observation. Thus, we can approximate the number of nodes with the observation, θ seconds after its generation as 0 if $\theta < T_M + T_T$, and $\left(2 \cdot \left(1 - \frac{1}{g\,T_{\mathrm{RZ}}}\right)\right)^{g\,(\theta-(T_M+T_T))}$ otherwise:

$$n(\theta) \simeq \left(2 \cdot \left(1 - \frac{1}{g\,T_{\mathrm{RZ}}}\right)\right)^{g\,(\theta-(T_M+T_T))} u\left(t - (T_M + T_T)\right), \quad \theta \leq \tau_l,$$

where $u(\cdot)$ is the unit step function and the expression is valid θ time units after observation generation and up to τ_l. The described behavior is reflected in Fig. 5.3, which reports observation availability curves as they evolve in the lifespan of an observation when the model size is small. The curves roughly correspond to $n(\theta)/(N\,\overline{A^{(m)}})$, which is an approximation for the observation availability, given the average model availability. The figure shows that, as expected, the number of models does not make an important difference in the curves, at least up to $M = 10$. Indeed, consider that the average processing time required at each node can be, in the worst case the one required to merge $M\,g\,T_{\text{RZ}}$ times per RZ traversal, plus $M\,\lambda\,T_{\text{RZ}}/N$ observation training occurrences. With the values used in the described experiments, this sums up to about $18.5\,M$ seconds, which is sufficiently lower than the total time in the RZ, so that we can, as a first approximation, neglect the queueing delay of processing tasks.

However, for what concerns the average number of observations present in each node, we have to consider that each observation i is received first by a non-seeder node when it is θ_i time units old, and then it is kept until the node leaves the RZ. Some observations will expire during the node's journey through the RZ, but they will be replaced by other observations received directly or by means of model instance exchanges. Thus, putting together what seen in the RZ for an observation and its replacement after expiration, the average node sees two parts of the average observation diffusion curve for each of the $\lambda\,\tau_l$ observations per model that can be present at the same time (i.e., only observations which have not expired, hence they are at most τ_l time units old). In total, for each observation diffusion curve, the node sees an interval of time $T_{\text{RZ}} < \tau_l$. This means that, in the computation of the average number of observations contained in a model, the time-average over the lifetime of a generic observation is limited to T_{RZ} rather than τ_l. The missing interval is a randomly located time window of average size $\tau_l - T_{\text{RZ}}$, expressing the random *gap window* in between one observation and its replacement, and over which the observation diffusion curve contributes as zero.[2] Under such approximations, taking per-node and per-model averages, and further averaging over the position of a gap window of size equal to its mean, we can write the following approximated expression:

$$\overline{T} \simeq \overline{A^{(m)}}\,M\,\frac{\lambda\,\tau_l}{N}\cdot\frac{1}{T_{\text{RZ}}}\int_0^{T_{\text{RZ}}}\frac{1}{T_{\text{RZ}}}\left(\int_0^x n(\theta)d\theta+\int_{x+\tau_l-T_{\text{RZ}}}^{\tau_l} n(\theta)d\theta\right)dx. \tag{5.10}$$

Note that the above expression sums, for each of the $\overline{A^{(m)}}\,M$ models owned by a node, the time-average of the number of observations per model received during its sojourn in the RZ crossing time. A numerical evaluation of Eq. (5.10) with $M = \{1, 3, 10\}$ and the corresponding model availability values of Fig. 5.1 yields the following values: 2.3, 6.7, and 21.7, which are decent estimates of the leftmost values of the curves depicted in Fig. 5.2.

[2]There could be multiple gaps, expressing the fact that an observation is replaced and then the replacement expires as well and is replaced in turn. However, the probability that this event occurs is negligible because the lifetime of an observation is typically longer than the average sojourn time of a node in the RZ.

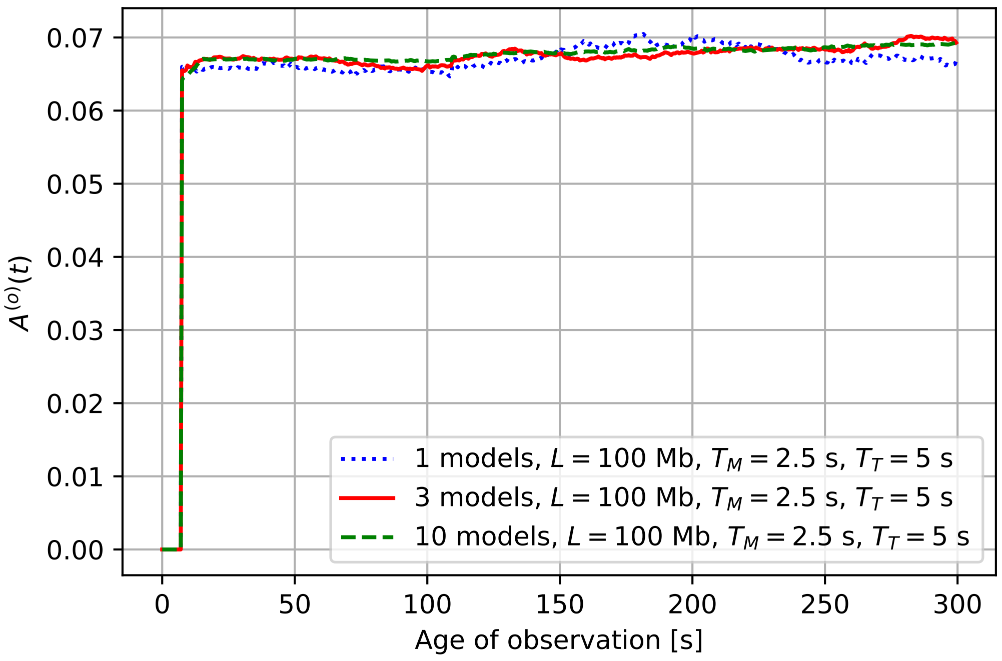


Figure 5.4: Observation availability over time with short training and merging times and $L = 100$ Mb.

As mentioned before, the diffusion of observation becomes impossible for large values of L. In that case, the curve of observation availability vs observation lifetime can be approximated with a step function, which is similar to the experimental results shown in Fig. 5.4. This behavior depends on the fact that each observation is available, after training, only at the seeder, and the number of nodes with model in the RZ shows only small fluctuations. For the case of Fig. 5.4, the average model availability is about 0.1 (see Fig. 5.1), which means about 15.7 nodes have the model, and the observation availability, once the observation has been trained, is $1/15.7 = 0.064$, which is close to what shown in Fig. 5.4.

To further evaluate the impact of the computing load, the following figures report the case in which training and merging times become large and non-negligible with respect to the time spent by a node in the RZ. To do so, the figures report results obtained in the same operational conditions as in the figures commented before, but for ten times longer training an merging times. In particular, with $T_M + T_T = 75$ s, the rate of 0.18 observations to train per second brings a load of about 5% per model, which means that just training, e.g., with 10 models, would keep the node busy for half of its journey in the RZ. Merging tasks, at a rate of 1 merging per model every $g^{-1} = 40$ s and $T_M = 25$ s would keep a node busy all the time with as few as 4 models as soon as their availability

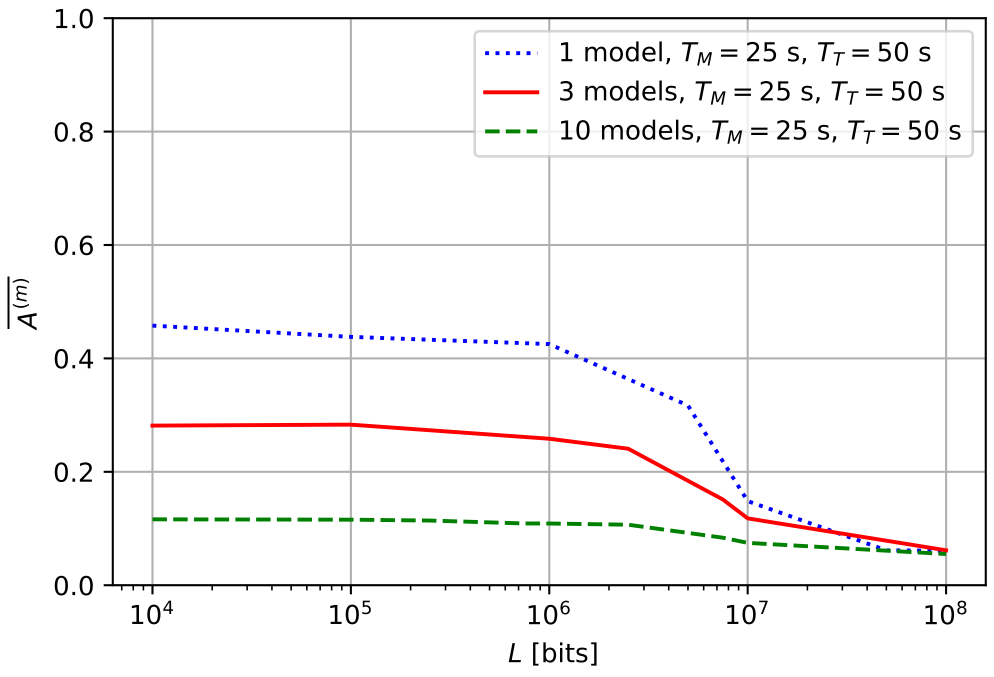


Figure 5.5: Model availability with high training and merging times.

reaches 0.4.

Comparing Fig. 5.5 to Fig. 5.1, we can observe how the model availability decreases when nodes spend more time with computing tasks, even if the model sizes are small, so they are not able to exchange their model instances until they have been merged and trained.

The approximation of Eq. (5.9) can be of use also in this case for small L. It actually reveals that we should expect $\overline{A^{(m)}} = 0.46$, which is very much in line with the result in the figure. However, here the impact of computation and communication bottlenecks, as the number of models increases, is much heavier than in the case of Fig. 5.1.

Model availability values with large L can be also estimated as before: the probability to receive an observation is 0.15, each observation trains a different model with high probability, and then the node carries the observation during its average residual sojourn time in the RZ, which in this case is $T_{\text{RZ}}/2 - (T_M + T_T)$. This implies that the model availability reduces to $0.15 \cdot (282/2 - 25)/282) \simeq 0.062$. This result is in line with what obtained via simulation with any number of models. Zooming into Fig. 5.5 at $L = 10^8$ b would reveal that the simulation results yield availability 0.062 for $M = 1$, 0.061 for $M = 3$ and 0.055 for $M = 10$, which also indicates that the computing load plays a role, with a slightly detrimental impact on performance, although the effect is minor.

With long processing times and small L, Fig. 5.6 shows that the observation

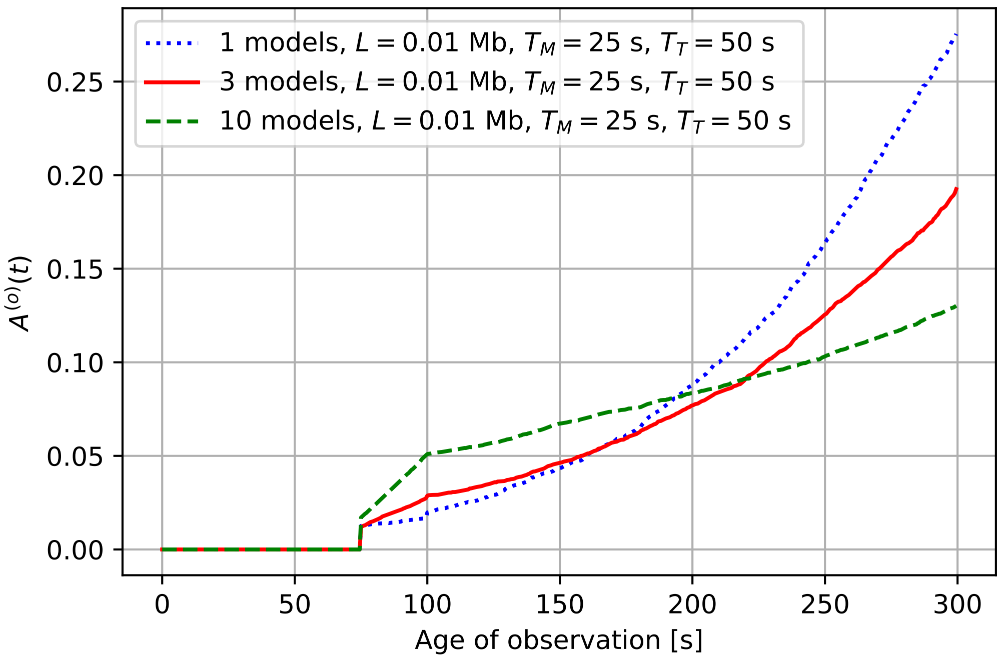


Figure 5.6: Observation availability over time with high training and merging times and $L = 0.01$ Mb.

availability evolves more slowly than with short processing time. The figure also clearly shows how the number of models becomes very important. Both phenomena are due to the relevance of computing tasks in this scenario. Here the case with only one model has a load that can be handled by the system, hence its observation availability curve is very similar to the case with low processing times, except for the initial delay due to longer training and merging times. Instead, the curves with 3 and 10 models clearly show that computation has become a bottleneck, and the growth rate of the curves becomes progressively smaller as M grows. Note that the higher the number of models, the smaller the model availability (see Fig. 5.5 for small L) and the smaller the number of nodes with model that appears at the denominator of the expression used to compute the observation availability curves. This explains why the curves for 3 and 10 models appear to start fast in Fig. 5.6, although their growing rate is clearly smaller than at $M = 1$.

With large L, the behavior shown in Fig. 5.7 is qualitatively similar to the one of Fig. 5.4, except for the fact that the average number of nodes with model is now smaller, which turns into higher values of instantaneous observation availability. The number of observations alive is however the same in all cases (it is about $\lambda \tau_l = 30$ per model), and the differences between the curves with 1, 3 and 10 models in Fig. 5.7 are due to the fact that the average model availability is slightly different in the three cases, as observed in

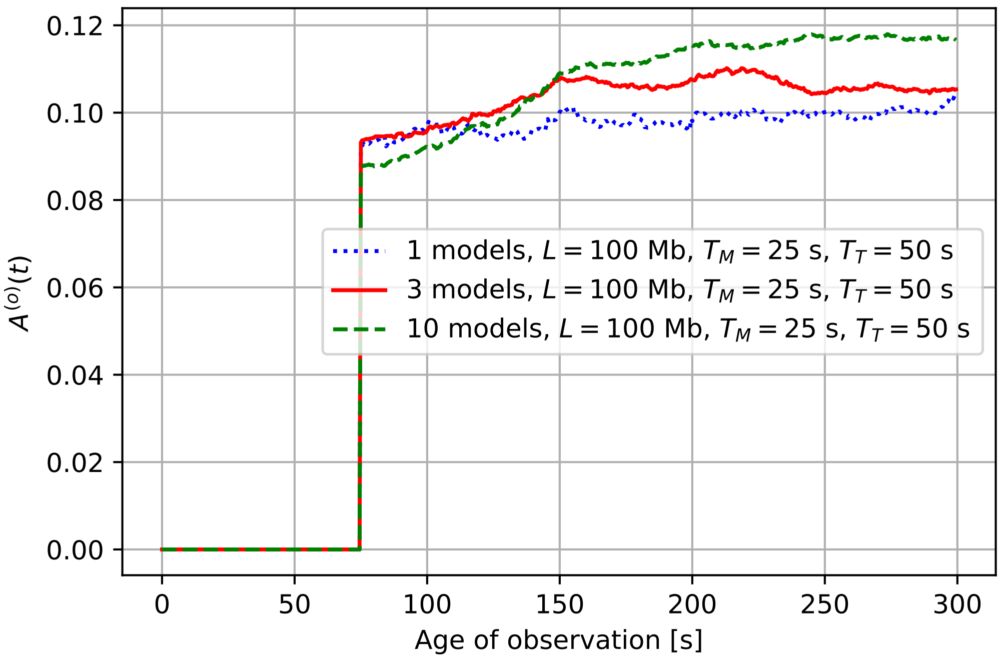


Figure 5.7: Observation availability over time with high training and merging times and $L = 100$ Mb.

the comment to Fig. 5.5, hence so is the number of nodes in the RZ with model, whose inverse value is the observation availability for this case in which only one node carries each observation. In particular, with the model availability at $L = 10^8$ b, we should see, on average 9.73 nodes with model for $M = 1$, hence the observation availability should go to $1/9.73 = 0.103$, while with $M = 3$ and $M = 10$ we should see observation availability values of about 0.104 and 0.116. Those values are close to what shown in Fig. 5.7.

Finally, Fig. 5.8 shows the learning capacity for high computing times T_T and T_M. Here, we observe that even if the trend of the curves is similar, the amount of observations at every node, on average, is much lower than in Fig. 5.2 for all sizes of L but specially for small model sizes. In this case, it is clear that the merging and training times have a significant impact in the computing power of the system and for that reason nodes are not able to retrieve as many observations as in the analyzed case at lower computing load. For low L, here the capacity does not scale with the number of models, while it roughly does at high values of L. The reason is that with large model sizes the number of observations per model tend to be similar, independently on the number of models, as observed in Fig. 5.4 and Fig. 5.7. Note also that the rightmost values of the curves for $\overline{T}$ roughly correspond to the rightmost values of the curves for $\overline{A^{(m)}}$ multiplied by the number of models, as for the case of low processing times. Instead, with small model sizes,

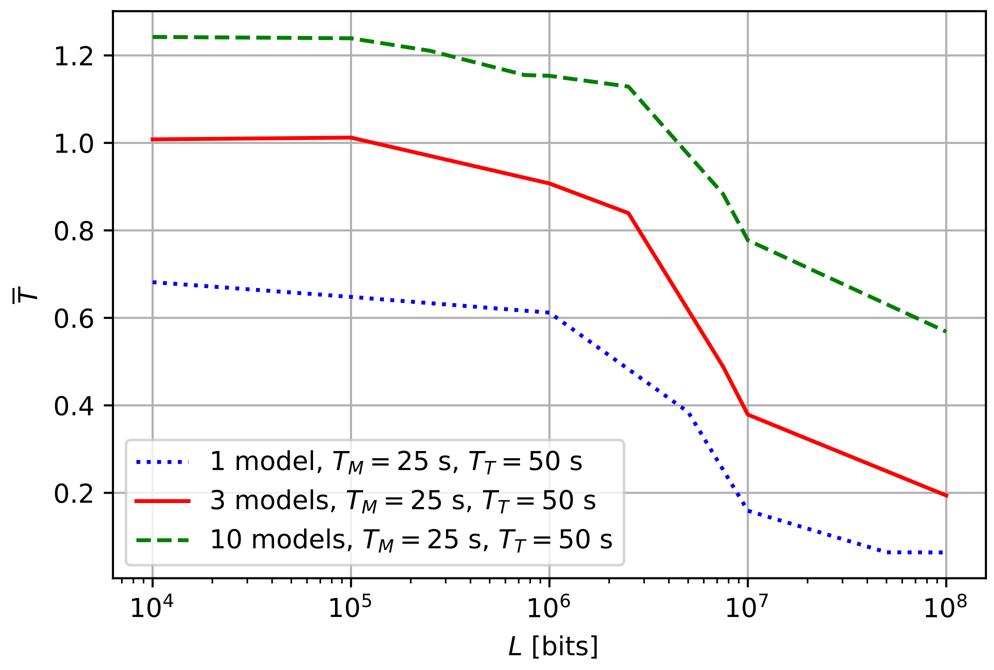


Figure 5.8: Learning capacity with high training and merging times.

the computing load—with more than one model and the model availability of Fig. 5.5—becomes high and clogs up the system. For instance, with 10 models, a node will receive about 1.3 new instances every 40 s, which takes at least 80% of its time. This limits the learning capacity of nodes, although not as much as the computing bottleneck does when L grows. Moreover, the impact of computing load is much more gradual then the impact of the FC bottleneck, which is still well visible in the figures with high processing time for training and merging.

6 Conclusions

This dissertation has considered three extensions of the Floating Content (FC) opportunistic information diffusion paradigm that was previously proposed and investigated in the technical literature. Using FC, it is possible to probabilistically store information over a given restricted local area of interest, by opportunistically spreading it to mobile users while in the area. A piece of information which is injected in the area by delivering it to one or more of the mobile users, is opportunistically exchanged among mobile users whenever they come in proximity of one another, progressively reaching most (ideally all) users in the area and thus making the information dwell in the area of interest. While traditional studies of the FC paradigm only looked at the communication component over one area of interest, I have considered:

- FC as a paradigm to move information from one area to another, and diffuse it over several disjoint areas of interest;
- FC as a paradigm for probabilistic information storage over an area of interest, exploring the associated storage capacity;
- FC as a paradigm that enables distributed computing, having in mind its application to collaborative learning algorithms such as Gossip Learning (GL), and exploring the associated learning capacity.

In the first part of this dissertation I have presented background work, including a brief review of my Master Thesis activity, devoted to the design, implementation and validation of a smartphone opportunistic information sharing application. The insight obtained during my Master Thesis work was extremely useful to devise smart operating procedures for schemes based on FC.

In the core of this dissertation, initially I proposed and studied a set of schemes to explore and combine different information dissemination paradigms along with real users mobility and predictions focused on the smart diffusion of content over disjoint areas of interest. I presented Predictive Content Dissemination Scheme (PRECISE), a series of data forwarding, storing and decision making schemes to benefit the infrastructureless content dissemination process in opportunistic networks, particularly focusing on Device-to-Device (D2D) data exchange between hotspots. PRECISE leverages node mobility patterns to make effective forwarding decisions and efficiently use network resources while maintaining fair content availability values. PRECISE specifies how a node adapts its probability to decide to exchange content when it meets another node, so as to speed up content diffusion when encounters are rare, or save resources when contacts are frequent. It also uses the knowledge of pedestrians and vehicles' position and mobility pattern to carry out meaningful connections between nodes and improves the main performance indicators of our system. Since node density and device limitations impair dissemination and system scalability, PRECISE encourages only the exchange of valuable data between potential peers according to their predicted movement, encounters and content expiration time. In such a way, PRECISE dramatically reduces the number of connections by 65-92%, consequently avoiding network congestion and drastically limiting the amount of local memory usage, as shown by means of a detailed simulation tool that I developed for the analysis of FC schemes.

In a second step, I have characterized the storage capacity of probabilistic distributed storage systems, developing a simple yet powerful information theoretical analysis based on a mean field model of opportunistic information exchange. I have also extended the previously developed simulation package to compare the numerical results generated by the analytical model to the predictions of realistic simulations under different setups, showing in this way the accuracy of the analytical approach, and characterizing the properties of the system storage capacity. The analytical and simulation results show that when the density of contents seeded in a floating system is larger than the maximum amount which can be sustained by the system in steady state, the mean content availability decreases, and the stored information saturates due to the effects of resource contention. With the presence of static nodes, in a system with infinite host memory and at the mean field limit, there is no upper bound to the amount of injected contents which a floating system can sustain. However, as with no static nodes, by increasing the injected information, the amount of stored information eventually reaches a saturation value which corresponds to the injected information at which the mean amount of time spent exchanging content during a contact is equal to the mean duration of a contact.

As a final step of my dissertation, I have also explored by simulation the computing and learning capabilities of an infrastructure-less opportunistic communication, storage and computing system. I have considered an environment that hosts a distributed Machine Learning (ML) paradigm that uses observations collected in the area over which the FC system operates to provide communication to users that are interested in inferring properties of the area. This scheme resembles a GL scheme, but for relying on FC, so we call it Floating Gossip (FG). Results show that the FG system can operate in two regimes, depending on the load of the FC scheme. At low FC load, the ML system in each node operates on observations collected by all users and opportunistically shared among nodes. At high FC load, especially when the data to be opportunistically exchanged becomes too large to be transmitted during the average contact time between nodes, the ML system can only exploit the observations endogenous to each user, which are much less numerous. As a result, I conclude that such setups are adequate to support general instances of distributed ML algorithms with continuous learning, only under the condition of low to medium loads of the FC system. While the communication load of the FC system induces a sort of phase transition on the ML system performance, the effect of computing load is more progressive. When the computing capacity is not sufficient to train all observations, some will be skipped, and performance progressively declines. This implies that the learning capacity of the proposed paradigm is tightly bound to the FC component of the system and it is able to support continuous learning under the condition of low to medium loads of the FC system.

Many extensions of the work presented in this dissertation are possible, from the development of models that can capture the temporal dynamics of the system, to experimental validations of the behaviors observed from simulation and analysis. The latter seem especially relevant in light of the increasing interest in cooperative distributed learning algorithms.

Appendix

Proof of Lemma 1

Proof: In distributed floating systems, the number of contents which a node can possess and replicate at a given point in time is equal to the minimum between the number of Replication Zones (RZs) in which the node is at that time, whose mean is $\gamma\pi R^2$, and the number of contents which can be stored in its memory. In localized floating systems, the number of RZs in which a node is located is $\gamma\pi R^2$. Since the probability of possessing a given content is well approximated by $a_s(t,R)$ (resp. $a_d(t,R)$), and since by the homogeneous assumption the probability for a node to possess a content is independent from other nodes, and the same for all nodes, the number of contents possessed by a node follows a binomial distribution, truncated at $\left\lfloor \frac{M}{L} \right\rfloor$. ■

Proof of Lemma 2

Proof: Let us denote with m_i, $i \in \{s,d\}$ and m_j, $j \in \{s,d\}$ denote the amount of contents possessed by each of the two nodes in contact at time t for RZ radius R, and let x_{ij} be the amount of exchangeable contents at node i (i.e. the amount of contents possessed by node i but not to node j, and for which there is enough storage space at node j). The probability that node i has x_{ij} exchangeable contents is equal to the probability that x_{ij} out of the m_i contents are not possessed by the other node, and that the remaining contents are possessed by the other node. Moreover, x_{ij} is upper bounded by the available storage space at node j, equal to $\frac{M}{L} - m_j$.

Note that m_i and m_j are random variables, whose distribution is given by Lemma 1. Therefore, the distribution of x_{ij} conditioned to m_i, m_j is distribute as a binomial $Bin(n,p)$, with parameters $n = \lfloor m_i \rfloor$ and $p = 1 - a_j(t,R)$, and truncated in $\left\lfloor \frac{M}{L} - m_j \right\rfloor$.

The PDF of x_{ij} is therefore the expectation with respect to m_i and m_j of $P(x_{ij}|m_i, m_j)$. ■

Proof of Lemma 3

Proof: Let us consider first the case of a contact between a static and a moving node. The amount of contacts exchanged between two nodes depend on the amount of exchangeable contents (i.e. of contents possessed by only one of the two nodes) that each of the two nodes has, as well as of the ratio between the contact time available for the exchange, and by the amount of free storage available at each node for storing the received contents. The probability that a content will be considered for being exchanged is the probability that the node is among those for which there is enough storage space at the other node, which is well approximated by the ratio between the mean amount of exchangeable contents for which the other node has storage space, and the mean amount of exchangeable contents when the other node has infinite storage space, i.e. by the ratio $\frac{E[x_{ij}]}{E[x_{ij},M=\infty]}$.

For a contact of duration τ, the amount of time available for transferring contents is given by $\tau - \tau_0$. The mean time taken for transferring a content is given by $\frac{L}{C_0}$. Therefore, on average the maximum amount of contents which can be exchanged during a contact of duration τ is $\lfloor\frac{(\tau-\tau_0)}{\frac{L}{C_0}}\rfloor$. The mean amount of contents which have to be exchanged during a contact taking place at time t is given by $E[x_{ij}+x_{ij}]$. When this quantity is larger than zero, if $E[x_{ij}+x_{ij}] \geq \lfloor\frac{(\tau-\tau_0)}{\frac{L}{C_0}}\rfloor$, then there is enough time for transferring all contents which can be exchanged. Otherwise, on average, the likelihood for a single content to be exchanged is equal to the ratio between $E[x_{ij}+x_{ij}] \geq \lfloor\frac{(\tau-\tau_0)}{\frac{L}{C_0}}\rfloor$ and $E[x_{ij}+x_{ij}]$. Averaging over contact duration, we get Eq. (4.2). The derivation of T_i follows along the same line. ■

Derivation of Theorem 1

In order to apply the mean field approximation, a key step is the derivation of the expression of the mean dynamics (also called *drift*), which describes the average local variation of the Continuous Time Markov Chain (CTMC) with respect to time [111], [122].

Lemma 5. *When the system satisfies the homogeneous conditions, the* drift *of the CTMC is given by (notice that, for ease of notation, we drop the indication of the dependency on* *time and RZ radius R from all the variables):*

$$\begin{cases} \dfrac{da_s}{dt} = \dfrac{b_s}{T_s} a_d(1-a_s)S_{ds} \\ \dfrac{da_d}{dt} = (1-a_d)\left[\aleph \dfrac{b_s}{T_s} a_s S_{sd} + \frac{b_d - \aleph b_s}{T_d} a_d S_{dd}\right] - \dfrac{2\alpha}{\psi D R} a_d \\ \dfrac{db_s}{dt} = \dfrac{g_s}{(1-\psi)D}(1-b_s)(1-b_d) - \dfrac{b_s}{T_s} \\ \dfrac{db_d}{dt} = \dfrac{g_d}{\psi D} 2(1-b_d)^2 + \aleph \dfrac{db_s}{dt} - \frac{b_d - \aleph b_s}{T_d} - \dfrac{4\alpha}{\psi D R} b_d \end{cases} \tag{6.1}$$

with $0 \le a_i, b_i \le 1$*,* $\aleph = \frac{1-\psi}{\psi}$*, and* $b_d \ge \aleph b_s$*.*

Proof:(Lemma 5) Let N_s denote the mean number of static nodes in a RZ of radius R possessing a given content at time t, averaged across all j. We have therefore $N_s = a_s \bar{N}(R)(1-\psi) = a_s(1-\psi)\pi R^2 D$, and similarly, for dynamic nodes, $N_d = a_d \bar{N}(R)\psi = a_d \psi \pi R^2 D$. Let us compute the rate at which N_s varies over time. As nodes are static, this quantity can only increase over time. The increase is due solely to static nodes which exit from the busy state due to completion of content transfers. The mean rate at which nodes exit the busy state is given by the ratio between the mean number of static busy nodes at time t in the RZ, given by $\bar{N}(R)(1-\psi)b_s$, and the mean time taken by an exchange between a static and a dynamic node, T_s. Moreover, let us consider one of these terminating exchanges. The probability that the j-th content was transferred to the static node during such exchange is equal to the probability that the dynamic node had the content and that the static node did not have it, given by $a_d(1-a_s)$, multiplied by the probability that the j-th content was transferred during the contact time, given by S_{ds}.

Summing up, we have

$$\tfrac{dN_s}{dt} = \tfrac{\bar{N}(R)(1-\psi)b_s}{T_s} a_d(1-a_s)S_{ds} \tag{6.2}$$

Normalizing this expression by $\bar{N}(R)(1-\psi)$ we get the first differential equation in Eq. (6.1).

The rate of change of N_d is given by the sum of three components. The first is given by those nodes which complete a content transfer in a contact with a static node. It is derived in a similar way as in the previous point, and it is given by:

$$\frac{\bar{N}(R)(1-\psi)b_s}{T_s} a_s(1-a_d)S_{sd}$$

This exploits exploiting the fact that the mean number of busy static nodes coincides with the mean number of dynamic nodes busy in contact with static nodes. The second component is the increase in N_d due to contacts among dynamic nodes. The number of busy dynamic nodes involved in such contacts is given by the difference between the total number of busy dynamic nodes, and the total number of busy static nodes, $\bar{N}(R)\psi b_d - \bar{N}(R)(1-\psi)b_s$.

When a couple of such nodes ends up being busy, only one of the two has acquired the given content. Hence the rate of these events is given by $\frac{\bar{N}(R)\psi b_d - \bar{N}(R)(1-\psi)b_s}{2T_d}$. The probability that the given content has been exchanged during a contact is given by the probability that only one of the two dynamic nodes in contact has the content, $2a_d(1-a_d)$ multiplied by the probability that the content is successfully exchanged between the two nodes, given by S_{dd}. Hence, the second contribution takes the form

$$\bar{N}(R)\frac{\psi b_d - (1-\psi)b_s}{T_d}2a_d(1-a_d)S_{dd}$$

The third contribution is given by those dynamic nodes which move out of the RZ for the given content. The rate of this type of events is $4\alpha\pi R b_s$. Note that this is computed by doubling the rate at which busy dynamic nodes exit the RZ. Indeed, if a busy node exits the RZ, two busy nodes are not busy anymore, even if the other node remains in the RZ. Putting all together, we have

$$\frac{dN_d}{dt} = \frac{\bar{N}(R)(1-\psi)b_s}{T_s}a_s(1-a_d)S_{sd} +$$

$$+\bar{N}(R)\frac{\psi b_d - \bar{N}(R)(1-\psi)b_s}{T_d}2a_d(1-a_d)S_{dd} + 4\alpha\pi R b_s \tag{6.3}$$

Normalizing by the mean total number of dynamic nodes in a RZ, we obtain the second differential equation in Eq. (6.1).

Let us now consider the rate at which the mean number of busy static nodes in a RZ at time t changes over time. Their increase is due to contacts between a static and dynamic node (an event which happens with a rate of $g_s\pi R^2$ contacts per second), both of which must be non-busy (with a probability $(1-b_s)(1-b_d)$). Their decrease is due to static nodes which complete the process of contents exchange with another node. Such an event takes place with a mean rate $\frac{\pi R^2 D(1-\psi)b_s}{T_s}$. Putting all together, and normalizing by $\pi R^2 D(1-\psi)$, we get the third differential equation in Eq. (6.1).

Finally, we consider the rate at which the mean number of busy dynamic nodes in a RZ at time t changes over time. The first contribution is given by those dynamic nodes which come in contact with static nodes. The contribution due to this population of nodes is equal to the rate at which the mean number of busy static nodes in a RZ at time t changes over time, because for each of these contacts involves a static and a dynamic node. The second contribution is given by content exchanges among dynamic nodes. With a similar reasoning as above, its expression is given by $g_d \pi R^2 2(1-b_d)^2$. The decrease in the number of busy nodes is due to two effects. The first is the end of content exchanges between two nodes in contact, due to either completion of all the pending transfers, or to the fact that the two nodes are not in contact anymore, given by $\pi R^2 D \frac{\psi b_d - (1-\psi) b_s}{T_d}$. Finally, the number of busy dynamic nodes in the RZ decreases when these nodes exit the RZ. The rate of this type of events is $4\alpha\pi R b_d$, i.e twice the rate at which busy nodes exit the RZ. This is due to the fact that a busy node exiting the RZ stop exchanging contents (and therefore being busy), hence also the other node involved in the exchange is not busy anymore, even if it remains in the RZ. Normalizing by the mean number of dynamic nodes in a RZ, $\pi R^2 D\psi$, we get the fourth differential equation in Eq. (6.1). ■

We can prove now the main result.

Proof:*(Theorem 1)* First, we show that, for any initial conditions $\mathbf{I}(0,R) = (a_i(0,R), b_i(0,R))$, there exists an array $\mathbf{I}_0$ such that $\lim_{R\to\infty} \mathbf{I}(0,R) = \mathbf{I}_0$ (*convergence of initial conditions* condition [111]). Let us choose $\mathbf{I}_0 = \mathbf{I}(0,R)$. Then for each content j, if $N_i^k(0,R)$ ($N_k^T(0,R)$) is the number of nodes with the k-th content in the RZ of content k (respectively, the total number of nodes in the RZ of content k) at time $t=0$ in the RZ, choosing $N_i^k(0,R) = \left\lfloor a_i(0,R) N_k^T(0,R) \right\rfloor$, and setting to zero the number of busy nodes at $t=0$ allows satisfying the convergence condition.

Given the assumption of stationarity of the mobility patterns, and of uniform node distribution, the mean total number of nodes in a RZ for a content k is equal for each content (given that all RZ have the same shape and size) and we denote it with $N(R)$. In order to apply the mean field approximation approach, we start by assuming $N_k^T(t,R) = N(R)$, for any content k an any time $t \geq 0$. As a consequence of the homogeneous condition, $N(R)$ grows proportionally to R. With these properties, the considered system can be modeled as a Population Continuous Time Markov Chain (PCTMC) [111]. Specifically, to each value of R we can associate a PCTMC model with a total number of nodes $N(R)$. As for the *size* of the model (i.e., as for the parameter used for normalizing the state occupancy), we choose the parameter $N(R)$ itself. Let us consider now a sequence of increasing values of R, to which we can associate a sequence of PCTMC models, each with the features described so far. By the nature of the system, one can easily verify that for any state transition, the state change vector (i.e., the difference between the state occupancy before and after the state transition) is independent of R and hence of the size of the model.

From Lemma 5 it is easy to see that the drift of the generic PCTMC is continuous. Let $\mathbf{I}(t,R) = (a_i(t,R), b_i(t,R))$ and $\mathbf{I}(t) = (\bar{a}(t), \bar{b}(t)) = \lim_{R\to\infty} \mathbf{I}(t,R)$.

As our sequence of PCTMC models satisfies these properties, by Theorem 1 in [111] we have that for any finite time horizon $T \leq \infty$, $\mathbb{P}\left\{\lim_{R\to\infty}\left(\sup_{0\leq t\leq T} \|\mathbf{I}(t,R) - \mathbf{I}(t)\|\right) = 0\right\} = 1$. That is, the sequence of population models associated to R converges *almost surely* to the dynamics of the Ordinary

Differential Equations (ODEs) in Theorem 1. Finally, in the case in which the number of nodes in each RZ is not constant, one can follow the same approach and derive an additional differential equation for the mean number of nodes in each RZ. Indeed, as we assumed the mobility is stationary, such differential equation would be a balance equation, giving a mean number of nodes in each RZ which does not vary over time, and which is not affected by the evolution over time of the other two variables of the system. Given such decoupling, the mean field approach can be applied separately to the mean number of nodes in each RZ, and to the two variables we have considered so far, obtaining again the ODEs in Eq. (6.1). $\blacksquare$

Floating Content and Floating Gossip Simulator Documentation

To carry out the work for Chapters 3 to 5, I have entirely built a floating content simulator to reproduce Device-to-Device (D2D) communication scenarios. All modules of the simulator have been fully implemented in Python programming language complying with the object oriented paradigm, in order to have multiple instances of the simulated classes. In such way, we can set diverse scenarios according to the specified input parameters. It is important to point out that there are currently two main versions of the simulator. The first version, whose class diagram is illustrated in Fig. 6.1, has been used in Chapter 3 and later adapted to the new metrics involved in the research performed in Chapter 4, we name it Floating Content (FC) version. The second simulator version, named Floating Gossip (FG) version, is the one used in Chapter 5 to reproduce FG scenarios. This one was meant as a branch of the first version that finally ended up as a parallel variant. The class diagram for this version is depicted in Fig. 6.2

Simulator structure

The structure of a simulated scenario is composed of a set of mutually associated classes. These classes slightly differ from one version to the other, thus, in the following we explain in detail what is the purpose of each one and in which version can be found, in case they are not common to both paradigms:

- Class *Scenario* consists of a squared Anchor Zone representing the total simulated area.

- Within the previous area, we define a number of circular zones that can take different sizes, namely Zone of Interests (ZOIs) or Replication Zones (RZs). These circular zones are described by the class *Zoi* and represent hot spots where disseminated data are especially valuable for nodes traversing them.

- A scenario will also contain a collection of nodes illustrating the users density, uniformly distributed in space, with the aim of spreading pieces of content to other nodes within a transmission range. Nodes can move according to different methods, that we will be detailed later: Random Direction (RD) mobility model, Random Waypoint (RWP) mobility model, Wrap-around mobility model, following synthetic traces based on maps or real traces. Nodes are instances of the class *User*.

- The pieces of contents to be disseminated can be instances of two different classes, *Message* and *Model*, depending on which version of the simulator we are considering. As explained during this thesis, in Chapter 3 and Chapter 4 users disseminate pieces of content that would represent messages, while in Chapter 5 we introduce a new paradigm where users exchange models that will be later trained with previously observed events, named observations, contained in users' databases. In the following, we may refer to either messages or models as content in order to simplify the descriptions.

- Finally, for the FG simulator branch, as presented in Chapter 5, we introduce the concept of Observations. These are pieces of information about local viewpoints observed by the nodes, that will be stored in nodes' databases and used to train their local models. They are implemented with the class *Observation*.

Note that the class diagrams shown in Figs. 6.1 and 6.2 comprise only the most relevant attributes and methods of each class in order to ease the understanding of the simulator structure. The complete documented code can be found in GitHub[1][2].

Simulator operation

Definition of classes and attributes

Class *User*: This class represents the nodes of the scenario. The attributes that compose the User class are represented by the following data structures:

- id: Integer that identifies the node.

- scenario: instance from the Scenario class. Represents the Scenario in which the nodes has been created.

- messages_list: in the FC version, List containing the messages retrieved by the node.

- models_list: in the FG version, List containing the models retrieved and/or trained by the node.

[1]https://github.com/noeliamp/FC-Simulator
[2]https://github.com/noeliamp/VERSION-FL-GL

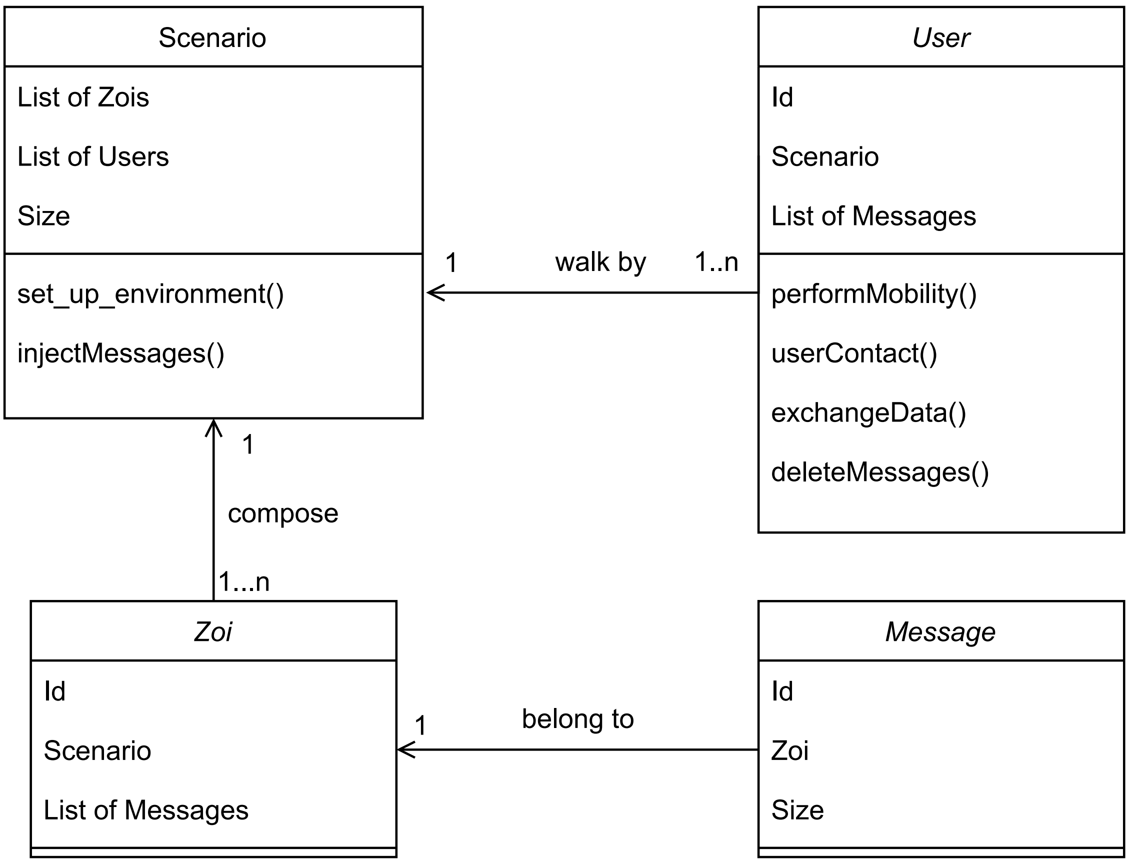


Figure 6.1: Class diagram of the Floating Content version of the simulator.

- total_memory: Integer that describes the memory limit in bits of the node's database.
- busy: Boolean value that represents whether the node is involved in connection.

The method performed by the User class are described below:

- performMobility(): Method called at every time slot to perform the node mobility according to the chosen mobility model.
- userContact(): Method called at every time slot to look for node peers and connect to one of them.
- exchangeData(): Method called at every time slot once a neighbour peer is chosen (after userContact()), to carry out the exchange of content between both peer nodes.
- computeTask(): in the FG version, method that performs the training of a model instance. This method is called at every time slot, to start training a new model or to continue with a previous training.

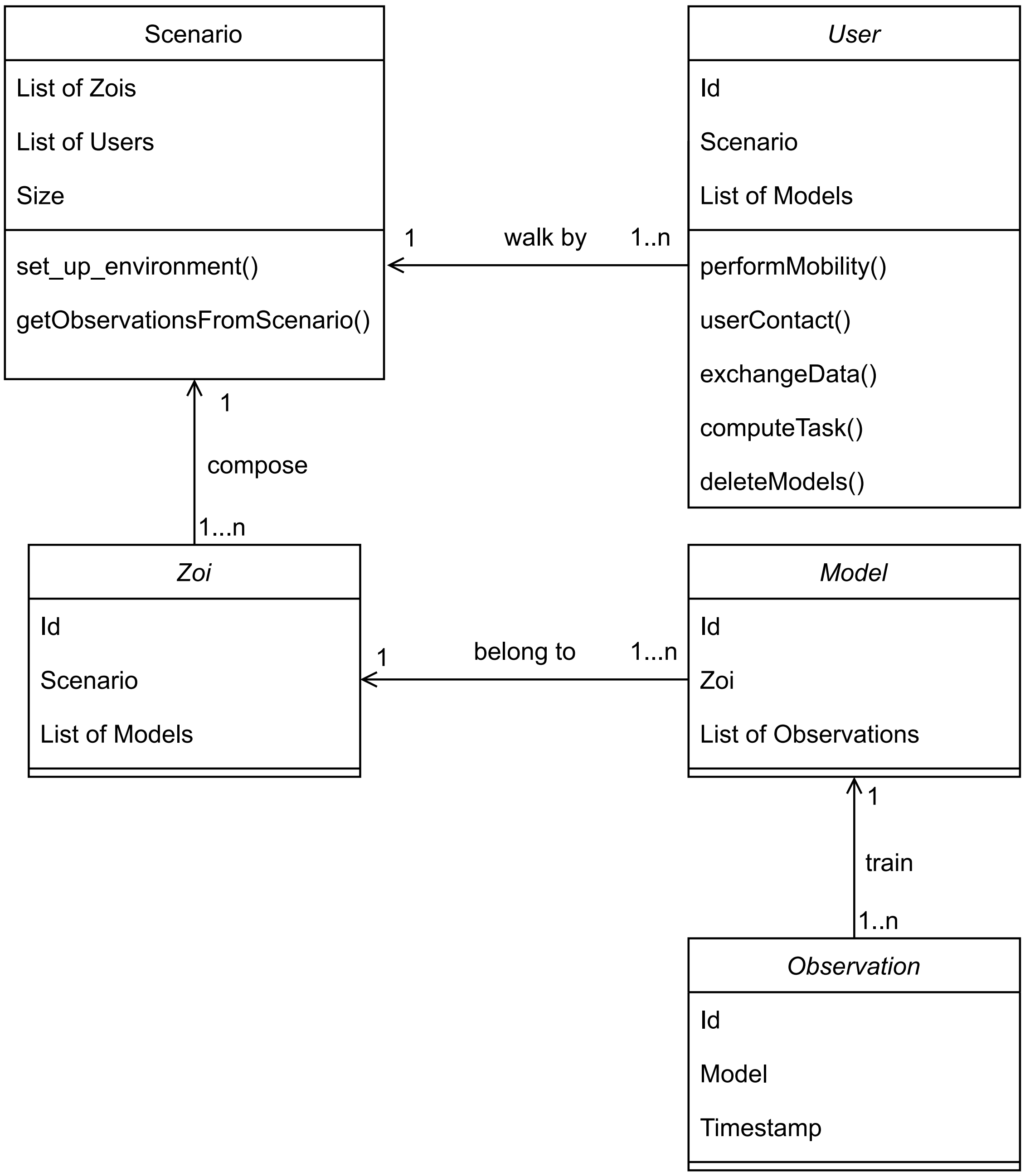


Figure 6.2: Class diagram of the Floating Gossip version of the simulator.

- deleteMessages(): Method called every time a node leaves the RZ. This method erases the node's database.

- deleteModels(): In FG version, method called every time a node leaves the RZ. This method erases the node's database. Note that, in this case the node's database may contain models and observations. Both classes' instances will be dropped from the nodes.

Class *Scenario*: This class describes the total simulated area. The attributes that compose the Scenario class are represented by the following data structures:

- list_of_zois: List structure containing the existing instances of the class **Zoi**.
- list_of_users: List containing the instances of the class *User* that correspond to the nodes of the scenario.
- max_area: Integer value that defines the side size in meters of the total squared simulated area.
- max_generation_time: in FG version, integer value in slots. It defines the maximum age of an observation, after which, the observation must be deleted from the whole Scenario, specifically from the observations list if zois, nodes and models.

The method performed by the Scenario class are described below:

- set_up_environment(): initial configuration of the Scenario, called only once.
- getObservationsFromScenario(): in FG version, method that allows nodes to generate new observations inside the RZs. This method is called according to the observation generation rate defined in the input parameters.

Class *Zoi*: This class describes an RZ. The attributes that compose the Zoi class are represented by the following data structures:

- id: Integer that identifies the Zoi instance.
- scenario: instance of the Scenario class to which the Zoi instance belongs.
- models_list: List containing the instances of the Model class that belong to a given Zoi.

The Zoi class does not perform any method.

Class *Message*: In the FC version, this class describes a message or piece of content. The attributes that compose the Message class are represented by the following data structures:

- id: Integer that identifies the Message instance.
- zoi: instance of the Zoi class to which the Model instance belongs.
- size: Integer value in bits corresponding to the message size.

The Message class does not perform any method.

Class *Model*: In the FG version, this class describes a model. The attributes that compose the Model class are represented by the following data structures:

- id: Integer that identifies the Model instance.
- zoi: instance of the Zoi class to which the Model instance belongs.
- observations_list: List containing the instances of the Observation class that belong to a given model.
- size: Integer value in bits corresponding to the model size.

The Model class does not perform any method.

Class *Observation*: In the FG version, this class describes the observation of an event. The attributes that compose the Observation class are represented by the following data structures:

- id: Integer value that identifies the Observation instance.
- model: instance of the Model class to which the Observation instance belongs.
- generation_time: Integer value that defines the time value at which the observation instance was generated.

The Observation class does not perform any method.

Description of the operation

To start running simulations, we initialize the simulator based on a set of parameters from a Json input file. The simulator operation time is slotted and the slot length can be tuned according to the input parameters. The status of the simulation is updated at every slot end, unlike event-based simulators. We first build the simulated area and the zones of interest. Then, in case we are running RD or RWP mobility models, nodes are uniformly distributed throughout the total square area. Note that, by setting different input parameters, the structure of a scenario can take not only the basic form previously explained but more complex configurations, composed of one or multiple RZs, as well as multiple pieces of content per RZ.

Once all the elements are placed, we create a predefined number of contents to be injected in the scenario. Here, three different options of the simulator can be selected:

- In the first option, only nodes within the zone of interest are allowed to start the simulation run with some stored contents. The amount of stored contents per node will depend on their memory capacity, also set by the configuration file. To determine which contents should be given to each node at the initial time we compute, based on the memory limit, the amount of fitting contents. The list of all contents generated in the scenario is shuffled every time we assign contents to a different node at time $t = 0$, and only the first n contents of that shuffled list that fit in the storage limits are copied in the corresponding node. In this way, we guarantee that all contents have the same probability of being selected.

- On the other hand, we have the option to periodically inject new pieces of content. A number of randomly chosen nodes located at the RZs will generate one content each. Both the number of nodes and the generation frequency will be specified in the input file.

- For the FG version of the simulator, where nodes work with models instead of messages, a default model will be assigned to each node the moment a new observation is generated. The generation of observations happens at a fixed rate with a call to the function *getObservationsFromScenario()*, also predefined in the input file.

- In the FG case, we can also define the seeding number of a given observation, i.e. the number of nodes that will obtain an observation in the moment of its generation. This parameter can be fixed or randomly generated among the total number of nodes present in a given RZ and varies for each new observation.

We decided to implement the process above in such a way that we could study the system performance, the storage and computational capacity of the scenario regarding the FC paradigm of this work.

From this point, the set up is ready to start simulating any given configuration. In the following sections we describe specifically how different chunks of the simulator code perform.

Nodes Mobility

We have implemented four different methods to perform nodes mobility in our system. In the simulator, one of these mobility methods will be called at the user class depending on the scenario configuration, in Figs. 6.1 and 6.2 the method is called *performMobility()* for the sake of simplicity, in the following descriptions we include the full name of each method according to its functionality. Next we describe the main characteristics of these methods.

Random Direction and Random Waypoint Mobility Models

All nodes in the scenario move freely throughout the whole simulated area according to RD or RWP mobility models and with a predetermined speed. While RWP has been implemented according to its exact definition, we have modified a set of specific features for the RD mobility model in pursuance of simplicity: nodes randomly choose a direction towards which they will keep moving with a constant speed until they hit a border of the outer square area, at that point we have implemented two different mechanisms. On the one hand, nodes can bounce mirroring the angle in which they collided with the edge. On the other, nodes can follow the wrap-around method. With this feature, when nodes leave the scenario they will simply appear at the exact opposite side from which they left as if the scenario were cylindrical in any direction. This motion persists up to the simulation end. In the simulator code, these methods can be found named as *randomWaypoint()* and *randomDirection()*.

Synthetic traces

To get closer to realistic scenarios we included an improved method based on city maps. In our work, we have specifically chosen the cities of Paderborn in Germany, although its application is possible to any other location. For this purpose, we have used the ONE Simulator [110] to generate some of the traces based on daily human mobility within the city through the shortest path algorithm. We first selected the map section in which the nodes will dwell during the simulations. Then, we set two main hot spots towards which the nodes will travel with higher probability. This idea arose from the need of keeping data alive between different parts of a city, due to the fact that users with similar interests can be found at different spots. For instance, traveling from the main campus of the University of Paderborn to second buildings belonging to the institution is a common practice among students and professors. To perform this type of mobility, in the simulator we have defined a function called *readTraces()* that will obtain at every slot the corresponding position coordinates for each node. For doing so, first we have defined a method, *parsePaderbornTraces()*, that runs at the beginning of the simulation and parses the traces file to convert them to a data structure, a python dictionary in our case, to avoid reading from a file at every new node movement. In case the traces files are in the right format, this same method can be used to parse any other city map. The proper file formats are defined in the following section.

Real traces

It is also possible to reach a higher realistic level in the behaviour of the simulator by introducing real mobility trajectories obtained from external sources. To be usable, these traces have to provide a node ID, the geographical coordinates of the node at a given time and the timestamp of the event. Additionally, due to the arising of works on realistic nodes mobility involving SUMO simulator, we have also included a module to be able to read such traces generated in SUMO format. The last addition to the simulator are modules to parse real scenarios, for instance Rome city center. We have worked with real nodes traces obtained from [70]. As explained in the previous section, to execute this feature, the simulator includes a method called *readTraces()*, in this case, the method reads the traces previously parsed by the function *parseRomeTraces().* Many other data sets obtained from the literature can be accessed from external sources such as CRAWDAD [3].

Connection establishment

Upon updating the position of nodes as the simulator moves to the next time slot, each node makes a call to the method *userContact()*, which will scan the peer neighbours around the node at a maximum distance specified in the input parameters file as the transmission radius *radius_tx*. Once the node finds a suitable neighbour to exchange content with, i.e., a node that is in range and not involved in any another connection (not busy), the establishment of a connection between both nodes will start. Each node traces its connections status with the following variables:

- *connection_duration*: integer value for the duration of a connection in slots.
- *prev_peer*: instance of the class User which represents the peer neighbour connected to a given node in the previous time slot. This is used to keep track of the nodes that are connected while the exchange of content has not finished.
- *ongoing_conn*: Boolean value. Similar to the previous, this flag tells whether a node is already involved in a connection that started in a previous slot and which has not finished yet because there is still content being transferred.

Data transfer policies

The way nodes handle the content varies depending on three different policies. The following three policies are implemented into the method *userContact()* that will run the corresponding one depending on the input parameters:

- A restrictive policy where nodes can only store and exchange content while they are inside the replication zones. Once nodes cross the borders of the RZs they will drop all their database and will not exchange anything related to it. This is the approach used in most of the works related to FC paradigm and has been included as a base line method. The main idea is to locally disseminate the content only where it is valuable.
- A second policy, starting from the previous point, also permits nodes to store content during periods of time, specified in the input file, while they are traveling outside the RZs.

[3]www.crawdad.org

- A third policy, less restrictive, includes the previous policies and also allows nodes to exchange content during a predefined period of time while they are outside the RZs only if one of the nodes is heading to a RZ before time elapses. This and the previous approaches favor the dissemination of content between more than one RZ, as well as an increase in the content availability due to possibility of new nodes entering the RZs already possessing the content.

- We have also implemented the PIS algorithm [65], where nodes make the decision of whether to exchange content and the selection of the peer node based on a set of similarity parameters, such as common interests and encountered nodes. From the method *userContact()*, in case Proximity-Interest-Social (PIS) is established as the dissemination protocol in the input parameters, there is a call to an external method called *PIS()*, here the simulator computes the main features of the PIS algorithm as explained in Section 3.4.2.3. At every time slot, the simulator main class will also run the methods called *socialFactorsUpdating()* and *similaritiesCalculation()* for the social factors and similarity values updating required by PIS protocol.

- Finally, as a base line, the simulator also contains the basic Epidemic dissemination algorithm.

Once two nodes establish a connection according to the *userContact()* code, they will only transfer the pieces of content that are missing at the peer node, with a call to the method *dataExchange()*. The order in which the contents are transferred is random: prior to the exchange, both nodes' lists -with only the peer missing contents- are shuffled to prevent certain pieces to be repeatedly exchanged in the first place. To accomplish this task, both nodes will share the channel rate defined by the scenario to perform a bidirectional communication. Each node will exploit half of the channel capacity to transmit its contents to the peer node in parallel. In case the data of one node do not completely fill their assigned capacity, the remaining quota will be relocated to the peer node. The established connection stay active for a number of slots until both peers have transmitted the total amount of contents. Therefore, connections are only interrupted in two cases: when nodes walk away from each other beyond the transmission range or when both nodes fill up their memory capacity during the connection.

When a connection interruption happens, incomplete file transfers are dropped with the method *deleteMessages()* or *deleteModels()*. Besides, if nodes belonging to an established connection have nothing to exchange or their memory is already full, the connection will be dropped right after one time slot. This slot represents the time needed to check each node status, which cannot be informed to the peer in advance with any current technology. When a connection ends the nodes go back again to an idle status, storing the information of the connection duration for future processing as detailed in the previous section.

Decision making for content exchange

Besides the policies previously explained, the simulator counts with an additional decision making strategy that can be turned on and tuned to control the greediness of the content dissemination process. This strategy can be found inside the method *userContact()* of the *User* class.

In each node's list called *prev_contents*, we store the mean number of contents that nodes own and are willing to exchange upon establishing a connection, which is obtained by applying an Autoregressive (AR) filter to the total number of contents missing in each node, *exchange_length*, when a connection occurs. The variables involved in this computation are c that represents the current time slot, *self.t_previous* indicating the time slot of the last connection establishment, *scenario.alpha* is an adaptive value that accounts for the time elapsed in between two consecutive connections, so that the AR filter operates with an exponential decay time defined as t in the code.

By computing this metric at every slot and comparing it with the value obtained in the previous slot, *previous_contents* we are able to tune the probability to exchange content, as detailed in Algorithm 2.

At each opportunity to establish a connection, the connection is established with probability p updated at each time slot as described in Chapter 3.

Model merging and training

In the FG version of the simulator, we have extend the users functionality to perform model training when they retrieve an observation from the scenario. The training is executed by the method *computeTask()* and requires two main input parameters: the merging length, this is the time duration required to merge model instances with the same ID previously to the training, and training length which is the time duration required to perform the training itself. Thus, merging and training operations are implemented as a countdown that lasts a fixed number of slots defined as an input parameter, both parameters are provided in slot units.

Nodes may have different instances of the same model before they start training, for that reason they need to first merge all instances of a model with the same ID to later train only one instance that involves the observations of all previous instances plus the new new trained one (the last one after the training session). When nodes only have one instance of a given model, they will directly enter in the training phase without merging. Each observation received by a node causes a given model training session, i.e., only one observation will be incorporated into the model contributions at each training process.

Tasks (or models) to be merged and/or trained are queued (using FIFO queues without space limits) in a specific data structure at each node called *pending_model_list*. It is important to point out that the pending models will only pass to the official models list (*model_list*) once they have been merged and/or trained. From both lists models are shuffled before selected to merge, train or exchange.

Metrics

Once the scenario is set, nodes carry out the operation explained in the sections above: at each slot the position of nodes is updated as well as their content exchange status, considering the allocated bandwidth. If the connection between nodes has not finished before the end of a slot, it will continue in the next one.

At every time slot, we compute a series of metrics to evaluate the performance of our system. These metrics do not belong to any specific class, but they are measured at the *main.py* script that act as a controller for the correct functioning of the whole simulator.

First of all, we measure the availability per content, i.e., the fraction between the number of nodes with a certain content and the total number of nodes in the RZ with the variable *a_per_content* and *a_per_model*. One of the options, in FC version, is to start providing all nodes in the RZ with a collection of contents up to their memory limit. As an alternative, contents are generated by single nodes and then start spreading. The availability values are also computed for models and observations in the FG version of the simulator.

Post-processing

Regarding the data analysis and post processing, we parse all dump files obtained from the simulator using Python programming language into Jupyter notebook framework.

We can also compute the volume of contents that has survived at the end of a simulation by dumping the final *model_list* of each node.

In general, any system behaviour can be measured by dumping the necessary data structures with the corresponding information. It is up to the metrics involved in the research work to decide how much information to provide after the simulation run. This can be easily done by adding methods to our class *Dump*, in which we have already implemented several metrics dump explained during this thesis in Chapters 3 to 5

Open source code

All code generated for the development of both simulator versions has been released in a GitHub repository in the following addresses: https://github.com/noeliamp/FC-Simulator and https://github.com/noeliamp/VERSION-FL-GL.

Bibliography

[1] N. Pérez Palma, V. Mancuso, and M. Ajmone Marsan, «Infrastructureless Pervasive Information Sharing with COTS Devices and Software», in *19th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2018)*, Chania, Greece: IEEE, Jun. 2018, pp. 1–9.

[2] N. Pérez Palma, F. Dressler, V. Mancuso, et al., «Precise: Predictive Content Dissemination Schemes Exploiting Realistic Mobility Patterns», *Computer Networks*, 2021.

[3] G. A. Rizzo, N. Pérez Palma, M. Ajmone Marsan, and V. Mancuso, «A Walk Down Memory Lane: On Storage Capacity in Opportunistic Content Sharing Systems», in *21th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2020)*, Virtual Conference: IEEE, Sep. 2020.

[4] G. Rizzo, N. Pérez Palma, M. Ajmone Marsan, and V. Mancuso, «Storage Capacity of Opportunistic Information Dissemination Systems».

[5] M. Conti and M. Kumar, «Opportunities in Opportunistic Computing», *IEEE Computer*, vol. 43, no. 1, pp. 42–50, Jan. 2010.

[6] J. Dede, A. Förster, E. Hernández-Orallo, J. Herrera-Tapia, K. K. Kuladinithi, V. Kuppusamy, P. Manzoni, A. b. Muslim, A. Udugama, and Z. Vatandas, «Simulating opportunistic networks: Survey and future directions», *IEEE Communications Surveys & Tutorials, 2*, pp. 1547–1573, Dec. 2017.

[7] G. Gorbil, «No way out: Emergency evacuation with no internet access», in *13th IEEE International Conference on Pervasive Computing and Communication (PerCom 2015), Workshop on Pervasive Networks for Emergency Management (PerNEM 2015)*, St. Louis, MO: IEEE, Mar. 2015.

[8] S.-Y. Lien, D.-J. Deng, C.-C. Lin, H.-L. Tsai, T. Chen, C. Guo, and S.-M. Cheng, «3GPP NR sidelink transmissions toward 5G V2X», *IEEE Access*, vol. 8, pp. 35 368–35 382, 2020.

[9] J. Schlienz and A. Roessler, «Device to Device Communication in LTE Whitepaper», *ROHDE & SCHWARZ: Munich, Germany*, 2015.

[10] I. C. S. L. M. S. Committee et al., «Wireless LAN medium access control (MAC) and physical layer (PHY) specifications», *ANSI/IEEE Std. 802.11-1999*, 1999.

[11] M. Woolley, «Bluetooth Core Specification Version 5.2 Feature Overview», *Bluetooth SIG: Kirkland, WA, USA*, 2020.

[12] Z. Zhang and R. Krishnan, «An overview of opportunistic routing in mobile ad hoc networks», in *IEEE Military Communications Conference (MILCOM 2013)*, San Diego, CA, Nov. 2013.

[13] E. Hyytia, J. Virtamo, P. Lassila, J. Kangasharju, and J. Ott, «When does content float? Characterizing availability of anchored information in opportunistic content sharing», *IEEE INFOCOM*, Apr. 2011.

[14] M. Grossglauser and D. N. C. Tse, «Mobility increases the capacity of ad hoc wireless networks», *IEEE/ACM Transactions on Networking (TON)*, vol. 10, no. 4, pp. 477–486, Aug. 2002.

[15] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, «Spray and wait: an efficient routing scheme for intermittently connected mobile networks», in *ACM SIGCOMM Workshop on Delay-Tolerant Networking (WDTN 2005)*, Philadelphia, PA: ACM, Aug. 2005.

[16] A. Vahdat and D. Becker, «Epidemic routing for partially connected ad hoc networks», Duke University, Technical Report CS-200006, Jun. 2000.

[17] A. Bujari, «A Survey of Opportunistic Data Gathering and Dissemination Techniques», in *IEEE International Conference on Computer Communication Networks (ICCCN 2012)*, Munich, Germany: IEEE, Aug. 2012.

[18] R. Ghebleh, «A comparative classification of information dissemination approaches in vehicular ad hoc networks from distinctive viewpoints: A survey», *Elsevier Computer Networks (COMNET)*, vol. 131, pp. 15–37, Feb. 2018.

[19] J. Wang, X. Kong, F. Xia, and L. Sun, «Urban Human Mobility», *ACM SIGKDD Explorations Newsletter*, vol. 21, no. 1, pp. 1–19, May 2019.

[20] J. Zhang, W. Wang, F. Xia, Y.-R. Lin, and H. Tong, «Data-Driven Computational Social Science: A Survey», *Big Data Research*, vol. 21, p. 100 145, Sep. 2020.

[21] C. Boldrini, M. Conti, J. Jacopini, and A. Passarella, «HiBOp: a History Based Routing Protocol for Opportunistic Networks», in *8th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2007)*, Helsinki, Finland: IEEE, Jun. 2007.

[22] P. Hui, J. Crowcroft, and E. Yoneki, «Bubble rap: social-based forwarding in delay tolerant networks», in *9th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2008)*, Hong Kong, China: ACM, May 2008, pp. 241–250.

[23] Y. Zhao and W. Song, «Survey on Social-Aware Data Dissemination Over Mobile Wireless Networks», *IEEE Access*, vol. 5, pp. 6049–6059, Jun. 2017.

[24] G. S. Pannu, F. Hagenauer, T. Higuchi, O. Altintas, and F. Dressler, «Keeping Data Alive: Communication Across Vehicular Micro Clouds», in *20th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2019)*, Washington, D.C.: IEEE, Jun. 2019.

[25] G. Manzo, M. A. Marsan, and G. Rizzo, «Performance Modeling of Vehicular Floating Content in Urban Settings», in *ITC 29*, vol. 1, Sep. 2017, pp. 99–107.

[26] L. Chancay-García, E. Hernández-Orallo, P. Manzoni, C. T. Calafate, and J. Cano, «Evaluating and Enhancing Information Dissemination in Urban Areas of Interest Using Opportunistic Networks», *IEEE Access*, vol. 6, pp. 32 514–32 531, 2018.

[27] N. Thompson, R. Crepaldi, and R. Kravets, «Locus: A location-based data overlay for disruption-tolerant networks», in *Proceedings of the 5th ACM workshop on Challenged networks*, ACM, Chicago, Illinois, USA, 2010, pp. 47–54.

[28] A. A. V. Castro, G. D. M. Serugendo, and D. Konstantas, «Hovering information–self-organizing information that finds its own storage», in *Autonomic Communication*, Springer, 2009, pp. 111–145.

[29] T. Nikolovski and R. W. Pazzi, «A Lightweight and Efficient Approach (LEA) for Hovering Information Protocols», in *DIVANet*, Miami, Florida, USA: ACM, 2017, pp. 31–38.

[30] A. Asadi, Q. Wang, and V. Mancuso, «A Survey on Device-to-Device Communication in Cellular Networks», *IEEE Communications Surveys & Tutorials*, vol. 16, no. 4, pp. 1801–1819, Apr. 2014.

[31] X. Zhang and Q. Zhu, «Distributed mobile devices caching over edge computing wireless networks», in *2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, May 2017, pp. 127–132.

[32] T. Xu, J. Jiao, X. Chen, and Y. Chen, «Social-Aware D2D Caching Content Deployment Strategy over Edge Computing Wireless Networks», in *2018 27th International Conference on Computer Communication and Networks (ICCCN)*, Jul. 2018, pp. 1–6.

[33] X. Zhang and Q. Zhu, «Collaborative Hierarchical Caching over 5G Edge Computing Mobile Wireless Networks», in *2018 IEEE International Conference on Communications (ICC)*, May 2018, pp. 1–6.

[34] S. Zhang, J. Wu, Z. Qian, and S. Lu, «MobiCache: Cellular traffic offloading leveraging cooperative caching in mobile social networks», *Computer Networks*, vol. 83, pp. 184–198, 2015.

[35] D. Feng, L. Lu, Y. Yuan-Wu, G. Y. Li, S. Li, and G. Feng, «Device-to-Device communications in cellular networks», *IEEE Communications Magazine*, vol. 52, no. 4, pp. 49–55, Apr. 2014.

[36] W. K. Edwards, «Discovery systems in ubiquitous computing», *IEEE Pervasive Computing*, vol. 5, no. 2, pp. 70–77, 2006.

[37] Creative Commons Attribution 2.5. (2017). Android Wi-Fi Peer-to-Peer (available online at https://developer.android.com/guide/topics/connectivity/wifip2p.html), (visited on 07/11/2017).

[38] C. Casetti, C. F. Chiasserini, L. C. Pelle, C. Del Valle, Y. Duan, and P. Giaccone, «Content-centric routing in Wi-Fi Direct multi-group networks», in *proceedings of IEEE WoWMoM 2015*, 2015, pp. 1–9.

[39] C. Funai, C. Tapparello, and W. Heinzelman, «Enabling multi-hop ad-hoc networks through Wi-Fi Direct multi-group networking», in *proceedings of IEEE ICNC 2017*, 2017, pp. 491–497.

[40] A. Asadi and V. Mancuso, «A survey on opportunistic scheduling in wireless communications», *IEEE Communications Surveys Tutorials*, vol. 15, no. 4, pp. 1671–1688, Apr. 2013.

[41] M. Garetto, W. Gong, and D. Towsley, «Modeling malware spreading dynamics», in *proceedings of IEEE INFOCOM 2003*, vol. 3, Mar. 2003, 1869–1879 vol.3.

[42] Creative Commons Attribution 3.0. (2017). Geofence (available online at https://developers.google.com/android/reference/com/google/android/ gms/location/Geofence), (visited on 08/14/2017).

[43] M. N. Tehrani, M. Uysal, and H. Yanikomeroglu, «Device-to-device communication in 5G cellular networks: challenges, solutions, and future directions», *IEEE Communications Magazine*, vol. 52, no. 5, pp. 86–92, 2014.

[44] U. N. Kar and D. K. Sanyal, «An overview of device-to-device communication in cellular networks», *ICT express*, vol. 4, no. 4, pp. 203–208, 2018.

[45] M. Haus, M. Waqas, A. Y. Ding, Y. Li, S. Tarkoma, and J. Ott, «Security and privacy in device-to-device (D2D) communication: A review», *IEEE Communications Surveys & Tutorials*, vol. 19, no. 2, pp. 1054–1079, 2017.

[46] (3GPP), *3GPP, 3rd generation partnership project*, Sep. 2017. [Online]. Available: http://www.3gpp.org/.

[47] S. Ali, G. Rizzo, V. Mancuso, V. Cozzolino, and M. Ajmone Marsan, «Experimenting with floating content in an office setting», *IEEE Communications Magazine*, vol. 52, no. 6, pp. 49–54, Jun. 2014.

[48] S. Ali, G. Rizzo, V. Mancuso, and M. Ajmone Marsan, «Persistence and availability of floating content in a campus environment», in *proceedings of IEEE INFOCOM 2015*, 2015, pp. 2326–2334.

[49] P. Gupta and P. R. Kumar, «The Capacity of Wireless Networks», *IEEE Transactions on Information Theory*, vol. 46, no. 2, pp. 388–404, Sep. 2006. [Online]. Available: http://dx.doi.org/10.1109/18.825799.

[50] P. Fiadino, M. Schiavone, and P. Casas, «Vivisecting WhatsApp Through Large-scale Measurements in Mobile Networks», in *proceedings of ACM SIGCOMM'14*, Chicago, Illinois, USA, 2014, pp. 133–134. [Online]. Available: http://doi.acm.org/10.1145/2619239.2631461.

[51] E. Hernández-Orallo, C. Borrego, P. Manzoni, J. M. Marquez-Barja, J. C. Cano, and C. T. Calafate, «Optimising data diffusion while reducing local resources consumption in Opportunistic Mobile Crowdsensing», *Pervasive and Mobile Computing*, vol. 67, p. 101 201, Sep. 2020.

[52] L. Chancay-García, E. Hernández-Orallo, P. Manzoni, A. M. Vegni, V. Loscrí, J.-C. Cano, and C. T. Calafate, «Optimising message broadcasting in opportunistic networks», *Elsevier Computer Communications*, pp. 162–178, May 2020.

[53] S. K. Dhurandher, D. K. Sharma, I. Woungang, and S. Bhati, «HBPR: History Based Prediction for Routing in Infrastructure-less Opportunistic Networks», in *27th IEEE International Conference on Advanced Information Networking and Applications (AINA 2013)*, Barcelona, Spain: IEEE, Mar. 2013.

[54] A. Lindgren, A. Doria, and O. Schelén, «Probabilistic routing in intermittently connected networks», in *4th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2003)*, Annapolis, MD: ACM, Jun. 2003.

[55] C. L. Barrett, S. J. Eidenbenz, and L. Kroc, «Parametric Probabilistic Sensor Network Routing», in *9th ACM International Conference on Mobile Computing and Networking (MobiCom 2003)*, San Diego, CA, Sep. 2003.

[56] B. Burns, O. Brock, and B. N. Levine, «MV Routing and Capacity Building in Disruption Tolerant Networks», in *24th IEEE Conference on Computer Communications (INFOCOM 2005)*, Miami, FL, Mar. 2005.

[57] Y. Liu, A. E. Bashar, F. Li, and Y. Wang, «Multi-copy data dissemination with probabilistic delay constraint in mobile opportunistic device-to-device networks», in *17th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2016)*, Coimbra, Portugal: IEEE, Jun. 2016.

[58] R. Yamamoto, A. Kashima, T. Yamazaki, and Y. Tanaka, «Adaptive Contents Dissemination Method for Floating Contents», in *90th IEEE Vehicular Technology Conference (VTC 2019-Fall)*, Honolulu, HI: IEEE, Sep. 2019.

[59] G. A. Rizzo, V. Mancuso, S. Ali, and M. Ajmone Marsan, «Stop and forward: Opportunistic local information sharing under walking mobility», *Elsevier Ad Hoc Networks*, vol. 78, pp. 54–72, Sep. 2018.

[60] C. Boldrini, M. Conti, and A. Passarella, «ContentPlace: social-aware data dissemination in opportunistic networks», in *11th ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM 2008)*, Vancouver, Canada: ACM, Oct. 2008.

[61] B. Ying, K. Xu, and A. Nayak, «Fair and Social-Aware Message Forwarding Method in Opportunistic Social Networks», *IEEE Communications Letters*, vol. 23, no. 4, pp. 720–723, Apr. 2019.

[62] A. Rahim, T. Qiu, Z. Ning, J. Wang, N. Ullah, A. Tolba, and F. Xia, «Social acquaintance based routing in Vehicular Social Networks», *Future Generation Computer Systems*, vol. 93, pp. 751–760, Apr. 2019.

[63] N. Ullah, X. Kong, Z. Ning, A. Tolba, M. Alrashoud, and F. Xia, «Emergency warning messages dissemination in vehicular social networks: A trust based scheme», *Vehicular Communications*, vol. 22, p. 100 199, Apr. 2020.

[64] A. M. Vegni, C. Souza, V. Loscrí, E. Hernández-Orallo, and P. Manzoni, «Data transmissions using hub nodes in vehicular social networks», *IEEE Transactions on Mobile Computing (TMC), 7*, pp. 1570–1585, Jul. 2020.

[65] F. Xia, L. Liu, B. Jedari, and S. K. Das, «PIS: A multi-dimensional routing protocol for socially-aware networking», *IEEE Transactions on Mobile Computing (TMC)*, Jan. 2016.

[66] A.-K. Pietiläinen, E. Oliver, J. LeBrun, G. Varghese, and C. Diot, «Mobiclique: middleware for mobile social networking», in *9th ACM SIGCOMM Conference on Internet Measurement (IMC 2009), Proceedings of the 2nd ACM workshop on Online social networks (WOSN '09)*, J. Crowcroft and B. Krishnamurthy, Eds., Barcelona, Spain: ACM, Nov. 2009, pp. 49–54.

[67] J. Scott, R. Gass, J. Crowcroft, P. Hui, C. Diot, and A. Chaintreau, *CRAWDAD dataset cambridge/haggle (v. 2009-05-29)*, en, May 2009. [Online]. Available: https://crawdad.org/cambridge/haggle/20090529,.

[68] E. M. Daly and M. Haahr, «Social network analysis for routing in disconnected delay-tolerant MANETs», in *8th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2007)*, Montréal, Canada: ACM, Sep. 2007.

[69] J. Li, Z. Ning, B. Jedari, F. Xia, I. Lee, and A. Tolba, «Geo-Social Distance-Based Data Dissemination for Socially Aware Networking», *IEEE Access*, vol. 4, pp. 1444–1453, Oct. 2016.

[70] L. Bracciale, M. Bonola, P. Loreti, G. Bianchi, R. Amici, and A. Rabuffi, *CRAWDAD dataset roma/taxi (v. 2014-07-17)*, traceset: taxicabs, Jul. 2014. [Online]. Available: https://crawdad.org/roma/taxi/20140717/taxicabs.

[71] M. Mordacchini, A. Passarella, and M. Conti, «Social Cognitive Heuristics for adaptive data dissemination in Opportunistic Networks», in *16th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2015)*, Boston, MA: IEEE, Jun. 2015.

[72] H. Khelifi, S. Luo, B. Nour, H. Moungla, Y. Faheem, R. Hussain, and A. Ksentini, «Named Data Networking in Vehicular Ad Hoc Networks: State-of-the-Art and Challenges», *IEEE Communications Surveys & Tutorials*, vol. 22, no. 1, pp. 320–351, 2020.

[73] E. Leme, N. Ivaki, N. Laranjeiro, and R. Moraes, «Analyzing Gossip Protocols for Reliable MANET Applications», in *2017 IEEE International Conference on Edge Computing (EDGE)*, 2017, pp. 98–105.

[74] A. Datta, S. Quarteroni, and K. Aberer, «Autonomous gossiping: A self-organizing epidemic algorithm for selective information dissemination in wireless mobile ad-hoc networks», in *International Conference on Semantics for the Networked World*, Springer, 2004, pp. 126–143.

[75] C. Barberis and G. Malnati, «Epidemic information diffusion in realistic vehicular network mobility scenarios», Nov. 2009, pp. 1–8.

[76] S. Zhao, L. Chang, T. Zhao, and W. Yan, «LCS-MANET: A mobile storage architecture with location centric storage algorithm in manets», in *2013 International Conference on Computing, Networking and Communications (ICNC)*, 2013, pp. 973–977.

[77] H. Hsu and K. Chen, «Optimal caching time for epidemic content dissemination in mobile social networks», in *2016 IEEE International Conference on Communications (ICC)*, 2016, pp. 1–6.

[78] S. Glass, I. Mahgoub, and M. Rathod, «Leveraging MANET-Based Cooperative Cache Discovery Techniques in VANETs: A Survey and Analysis», *IEEE Communications Surveys Tutorials*, vol. 19, no. 4, pp. 2640–2661, 2017.

[79] H. Khelifi, S. Luo, B. Nour, H. Moungla, Y. Faheem, R. Hussain, and A. Ksentini, «Named Data Networking in Vehicular Ad Hoc Networks: State-of-the-Art and Challenges», *IEEE Communications Surveys Tutorials*, vol. 22, no. 1, pp. 320–351, 2020.

[80] K. Machado, A. Boukerche, E. Cerqueira, and A. A. F. Loureiro, «A Socially-Aware In-Network Caching Framework for the Next Generation of Wireless Networks», *IEEE Communications Magazine*, vol. 55, no. 12, pp. 38–43, 2017.

[81] E. Hernández-Orallo, C. Borrego, P. Manzoni, J. M. Marquez-Barja, J. C. Cano, and C. T. Calafate, «Optimising data diffusion while reducing local resources consumption in Opportunistic Mobile Crowdsensing», *Pervasive and Mobile Computing*, vol. 67, 2020.

[82] J. Ott, E. Hyytiä, P. Lassila, T. Vaegs, and J. Kangasharju, «Floating content: Information sharing in urban areas», in *proceedings of IEEE PerCom 2011*, IEEE, 2011, pp. 136–146.

[83] M. Desta, E. Hyytia, J. Ott, and J. Kangasharju, «Characterizing Content Sharing Properties for Mobile Users in Open City Squares», in *IEEE WONS*, Mar. 2013.

[84] B. Liu, B. Khorashadi, D. Ghosal, C.-N. Chuah, and M. H. Zhang, «Assessing the VANET's local information storage capability under different traffic mobility», in *INFOCOM, 2010 Proceedings IEEE*, IEEE, 2010, pp. 1–5.

[85] B. Xie, Y. W. Chen, M. Xu, and Y. G. Wang, «Mathematical modeling of locally information storage capability of VANET for highway traffic», in *Applied Mechanics and Materials*, Trans Tech Publ, vol. 373, 2013, pp. 1914–1919.

[86] L. Pajevic and G. Karlsson, «Modeling opportunistic communication with churn», *Computer Communications*, vol. 96, pp. 123–135, 2016.

[87] L. Pajevic, V. Fodor, and G. Karlsson, «Ensuring Persistent Content in Opportunistic Networks via Stochastic Stability Analysis», *ACM Transactions on Modeling and Performance Evaluation of Computing Systems (TOMPECS)*, vol. 3, no. 4, p. 16, 2018.

[88] G. Manzo, J. S. Otalora, M. A. Marsan, and G. Rizzo, «A Deep Learning Strategy for Vehicular Floating Content Management», *ACM SIGMETRICS Performance Evaluation Review*, vol. 46, no. 3, pp. 159–162, Jan. 2019.

[89] F. Neves dos Santos, B. Ertl, C. Barakat, T. Spyropoulos, and T. Turletti, «Cedo: Content-centric dissemination algorithm for delay-tolerant networks», in *ACM MSWIM*, 2013, pp. 377–386.

[90] R. Hagihara, Y. Yamasaki, and H. Ohsaki, «On delivery control for floating contents sharing with epidemic broadcasting», in *IEEE CCNC*, Jan. 2017, pp. 353–356.

[91] S. Gagliardi et al., «A Distributed Caching System in DTNs», Master's thesis, Helsinki University of Technology, 2009.

[92] W. Gao, G. Cao, A. Iyengar, and M. Srivatsa, «Supporting cooperative caching in disruption tolerant networks», in *2011 31st International Conference on Distributed Computing Systems*, IEEE, 2011, pp. 151–161.

[93] J. Pääkkönen, C. Hollanti, and O. Tirkkonen, «Device-to-device data storage for mobile cellular systems», in *2013 IEEE Globecom Workshops (GC Wkshps)*, IEEE, 2013, pp. 671–676.

[94] D. Haritha and R. Lalitha, «Cluster Based Neighbor Coverage Relaying (CBNCR)-A Novel Broadcasting Mechanism for Dissemination of Data in VANETs», *Computer Engineering and Intelligent Systems*, vol. 5, no. 9, pp. 36–43, 2014.

[95] Y. Li and W. Wang, «Can mobile cloudlets support mobile applications?», in *IEEE INFOCOM*, 2014, pp. 1060–1068.

[96] B. Liu, B. Khorashadi, D. Ghosal, C.-N. Chuah, and H. M. Zhang, «Analysis of the information storage capability of VANET for highway and city traffic», *Transportation Research Part C: Emerging Technologies*, vol. 23, pp. 68–84, 2012.

[97] C.-N. Lai, «A Multi-Custodians Distributed Storage Mechanism for DTN Storage-based Congestion Problem», *Malaysian Journal of Computer Science*, vol. 29, no. 1, pp. 28–44, 2016.

[98] B. Baron, P. Spathis, M. D. de Amorim, and M. Ammar, «Cloud Storage for Mobile Users Using Pre-Positioned Storage Facilities», in *ACM SmartObjects*, New York City, 2016, pp. 11–16.

[99] B. Hu, L. Fang, X. Cheng, and L. Yang, «Vehicle-to-vehicle distributed storage in vehicular networks», in *2018 IEEE International Conference on Communications (ICC)*, IEEE, 2018, pp. 1–6.

[100] R. Ormándi, I. Hegedűs, and M. Jelasity, «Gossip learning with linear models on fully distributed data», *Concurrency and Computation: Practice and Experience*, vol. 25, no. 4, pp. 556–571, 2013.

[101] Q. Yang, Y. Liu, Y. Cheng, Y. Kang, T. Chen, and H. Yu, «Federated learning», *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 13, no. 3, pp. 1–207, 2019.

[102] P. P. Jayaraman, C. Perera, D. Georgakopoulos, and A. Zaslavsky, «Efficient opportunistic sensing using mobile collaborative platform MOSDEN», in *9th IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing*, 2013, pp. 77–86.

[103] (). Gigwalk. Accessed: 2021-10-15.

[104] (). Waze. Accessed: 2021-10-15.

[105] (). Millionagents. Accessed: 2021-10-15.

[106] Y. Han, *Ultra-Large-Scale Crowdsensing in Device-to-Device Networks.* University of Louisiana at Lafayette, 2017.

[107] L. Wang, Z. Yu, D. Yang, T. Ku, B. Guo, and H. Ma, «Collaborative Mobile Crowdsensing in Opportunistic D2D Networks: A Graph-Based Approach», *ACM Trans. Sen. Netw.*, vol. 15, no. 3, May 2019. [Online]. Available: https://doi.org/10.1145/3317689.

[108] M. Vasic and A. Martinoli, «A collaborative sensor fusion algorithm for multi-object tracking using a Gaussian mixture probability hypothesis density filter», in *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, IEEE, 2015, pp. 491–498.

[109] A. Langley, A. Riddoch, A. Wilk, A. Vicente, C. Krasic, D. Zhang, F. Yang, F. Kouranov, I. Swett, J. Iyengar, J. Bailey, J. Dorfman, J. Roskind, J. Kulik, P. Westin, R. Tenneti, R. Shade, R. Hamilton, V. Vasiliev, W.-T. Chang, and Z. Shi, «The QUIC Transport Protocol: Design and Internet-Scale Deployment», in *ACM SIGCOMM 2017*, Los Angeles, CA: ACM, Aug. 2017.

[110] A. Keranen, J. Ott, and T. Kämäräinen, «The ONE simulator for DTN protocol evaluation», in *2nd ACM/ICST International Conference on Simulation Tools and Techniques for Communications, Networks and Systems (SIMUTools 2009)*, Rome, Italy: ICST, Mar. 2009.

[111] A. Kolesnichenko, V. Senni, A. Pourranjabar, and A. Remke, «Applying Mean-Field Approximation to Continuous Time Markov Chains», in *ROCKS*, 2012.

[112] J. H. Argyris, G. Faust, M. Haase, and R. Friedrich, *An Exploration of Dynamical Systems and Chaos - 2^{nd} Ed.* Springer, 2015.

[113] A. Chaintreau, J.-Y. Le Boudec, and N. Ristanovic, «The age of gossip: spatial mean field regime», in *ACM SIGMETRICS Performance Evaluation Review*, ACM, vol. 37, 2009, pp. 109–120.

[114] J. Virtamo, «38.3143 Queuing Theory - Markov processes», Tech. Rep.

[115] J.-Y. L. Boudec, «The stationary behaviour of fluid limits of reversible processes is concentrated on stationary points», *arXiv preprint arXiv:1009.5021*, 2010.

[116] T. M. Cover and J. A. Thomas, *Elements of information theory.* John Wiley & Sons, 2012.

[117] C. E. Shannon, W. Weaver, and A. W. Burks, «The Mathematical Theory of Communication», 1951.

[118] A. E. Mohr, E. A. Riskin, and R. E. Ladner, «Unequal loss protection: Graceful degradation of image quality over packet erasure channels through forward error correction», *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 6, pp. 819–828, 2000.

[119] D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker, «Recent Development and Applications of SUMO - Simulation of Urban MObility», *International Journal On Advances in Systems and Measurements*, vol. 5, no. 3-4, pp. 128–138, Dec. 2012.

[120] L. Codeca, R. Frank, and T. Engel, «Luxembourg sumo traffic (LUST) scenario: 24 hours of mobility for vehicular networking research», in *IEEE Vehicular Networking Conference (VNC)*, 2015, pp. 1–8.

[121] F. Bai, D. D. Stancil, and H. Krishnan, «Toward understanding characteristics of dedicated short range communications (DSRC) from a perspective of vehicular network engineers», in *ACM Mobicom*, 2010, pp. 329–340.

[122] L. Bortolussi, J. Hillston, D. Latella, and M. Massink, «Continuous approximation of collective system behaviour: A tutorial», *Performance Evaluation*, vol. 70, no. 5, pp. 317–349, 2013.