UNIVERSITÉ DU LUXEMBOURG

PhD-FSTC-2016-50
The Faculty of Sciences, Technology and Communication

DISSERTATION

Defense held on "25/10/2016" in Luxembourg

to obtain the degree of

DOCTEUR DE L'UNIVERSITÉ DU LUXEMBOURG

EN Informatique

by

Claudio Fiandrino
Born on 18/12/1987 in Cuneo (Italy)

ENERGY-EFFICIENT COMMUNICATIONS IN CLOUD, MOBILE CLOUD AND FOG COMPUTING

Abstract

This thesis studies the problem of energy efficiency of communications in distributed computing paradigms, including cloud computing, mobile cloud computing and fog/edge computing. Distributed computing paradigms have significantly changed the way of doing business. With cloud computing, companies and end users can access the vast majority services online through a virtualized environment in a pay-as-you-go basis. Mobile cloud and fog/edge computing are the natural extension of the cloud computing paradigm for mobile and Internet of Things (IoT) devices. Based on offloading, the process of outsourcing computing tasks from mobile devices to the cloud, mobile cloud and fog/edge computing paradigms have become popular techniques to augment the capabilities of the mobile devices and to reduce their battery drain. Being equipped with a number of sensors, the proliferation of mobile and IoT devices has given rise to a new cloud-based paradigm for collecting data, which is called mobile crowdsensing as for proper operation it requires a large number of participants.

A plethora of communication technologies is applicable to distributing computing paradigms. For example, cloud data centers typically implement wired technologies while mobile cloud and fog/edge environments exploit wireless technologies such as 3G/4G, WiFi and Bluetooth. Communication technologies directly impact the performance and the energy drain of the system. This Ph.D. thesis analyzes from a global perspective the efficiency in using energy of communications systems in distributed computing paradigms. In particular, the following contributions are proposed:

- A new framework of performance metrics for communication systems of cloud computing data centers. The proposed framework allows a fine-grain analysis and comparison of communication systems, processes, and protocols, defining their influence on the performance of cloud applications.

- A novel model for the problem of computation offloading, which describes the workflow of mobile applications through a new Directed Acyclic Graph (DAG) technique. This methodology is suitable for IoT devices working in fog computing environments and was used to design an Android application, called TreeGlass, which performs recognition

of trees using Google Glass. TreeGlass is evaluated experimentally in different offloading scenarios by measuring battery drain and time of execution as key performance indicators.

- In mobile crowdsensing systems, novel performance metrics and a new framework for data acquisition, which exploits a new policy for user recruitment. Performance of the framework are validated through CrowdSenSim, which is a new simulator designed for mobile crowdsensing activities in large scale urban scenarios.

Acknowledgements

This Ph.D. dissertation is the result of more than three years activities and collaboration with a number of amazing people that I wish to thank, hoping to not forget any.

First of all, my deepest gratitude to my advisor Dzmitry Kliazovich for his support, guidance, constant feedback and expertise during all these years. I am also very grateful to his sponsorship in many events like the organization of IEEE CloudNet that significantly contributed to enrich my knowledge and experience. I would also like to thank the other members of my Ph.D. Committee, Pascal Bouvry and Albert Zomaya for their help and support and for sharing their expertise during the whole period of my studies. I am truly grateful for making me developing independence in performing research in topics like mobile crowdsensing: it was a long shot back in the days.

I would like to acknowledge Burak Kantarci, for hosting me in his Nextcon Lab at Clarkson University, the warm welcome, for his precious advise in research and not only. Thanks to Fazel, Maryam and Parvej for being great lab mates and to Audrey. Even if short, my stay in Potsdam will remain unforgettable.

I must thank Paolo Giaccone, former master thesis advisor and member of this Ph.D. dissertation committee, for his continuous guidance and support during these years.

It might be unconventional, but I would like to thank the master students I helped supervising during these years: Francesco, Claudio, Luca, Andrea, Nicholas and Giuseppe. In particular, I want to acknowledge Claudio, Andrea and Nicholas, who contributed to this work. I learned more from them than the other way round.

I was very happy to work with the other members of the team of Pascal Bouvry and for sharing office with amazing colleagues: Xihui, Patrick, Jun and Arash. Thank you all for the great time.

Special thanks go to Ilaria, for being the most inspiring person I have ever met and to all the other guys in the “Italians in Lux” team for the great moments we spent together. Many thanks also to Giovanna, Fabrizio and Daniel for being a constant Turin-presence during my time in Luxembourg.

Last but not least, I would like to express my sincere gratitude to my family for their patience, love and support in every moment.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Context

Cloud computing has dramatically changed the way industry and clients do business and consume services. Companies and end users can now access the vast majority of services on demand and online through a virtualized environment in a pay-as-you-go basis. According to Forbes, the revenues of worldwide public IT cloud services are expected to reach $ 128 billions by 2018 [1]. The main features making attractive cloud computing paradigm are scalability and elasticity. These properties eliminates the need for cloud users to plan ahead for resource provisioning. Cloud computing has made real the idea of computing utility. Computing is now identified as an essential service ad disposal of everybody like water, electricity, gas, and Internet connectivity.

The widespread diffusion of mobile devices such as laptop computers and smartphones first and of the Internet of Things (IoT) devices later, has created the need for extending the concept of utility computing in the mobile environment. Mobile cloud and fog computing are distributed computing paradigms filling this gap and enable the mobile users to access services and benefit from all the advantages cloud computing paradigm offers [2]. The process of offloading has given rise to additional benefits for the mobile users. Offloading is the process of outsourcing part of the computation from the resource constrained mobile devices to the cloud. Through offloading, mobile users augment the capabilities of the mobile and IoT devices, e.g., they can use applications that natively would not run locally because of resource limitations such as poor processing power, storage and memory. Moreover, avoiding local computing allows mobile and IoT devices to save energy.

Cloud providers deliver to the users mainly three types of service models, Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS). IaaS providers offer to the users a pool of computing, storage and network resources. PaaS providers enable users to access a platform to develop and deploy software while in SaaS users access software running on servers. The emerging of new trends such as Big Data and IoT has given rise to new types of services, including Sensing as a Service (S^{2}aaS) and Data as a Service (DaaS). S^{2}aaS makes sensor infrastructure accessible through the Internet and the data collected at users disposal. S^{2}aaS business models are projected to become a pillar solution for development of smart cities.

Mobile CrowdSensing (MCS) has become a popular paradigm for sensing data and operates exploiting sensors of smartphones and wearable devices. MCS is projected to greatly improve citizens everyday life and to provide new perspectives to urban societies. MCS is an essential solution for building smart cities of the future, which aim at using ICT solutions to improve management of everyday life of their citizens [3]. In such a context, active participation of citizens improves spatial coverage of already deployed sensing systems with no need of further investments. MCS takes advantage of human intelligence, which has a deeper context understanding than traditional sensor networks. For example, having human involved in detection of free parking spot detection using WiFi scans like in ParkSense [4] provides more accurate performance [5], [6]. Parking is only one of the possible city services where MCS will play a fundamental role thanks to its unique features. In addition, other potential applications are smart lighting, smart traffic management and environmental monitoring, including air and noise quality.

1.2 Motivation

The analysis of efficiency of communications from an energy perspective is a challenging problem and depends on the distributed computing paradigm and on the communication technology. For example, in cloud computing data centers, turning off network switches running at low levels of utilizations improves energy efficiency. In mobile cloud or fog computing, turning off communication devices in the infrastructure side would not save energy as the majority of energy costs is at the end-users side. Mobile and IoT devices are battery equipped therefore energy is scarce and, consequently, a precious resource. In this context, the choice of using a given communication technology, such as WiFi rather than 3G interface, is crucial both for the performance of the application and battery drain.

Each distributed computing domain, therefore, requires different approaches to optimize efficiency in using energy. Unlike focusing on a single domain, one of the strengths of this thesis is the analysis of the problem from a holistic and comprehensive perspective. The thesis provides contributions in all the distributed computing domains, including cloud computing as well as mobile cloud and fog computing.

1.3 List of contributions

The objective of this PhD thesis is to study energy efficiency in cloud, mobile cloud and fog computing paradigms. This section outlines the major contributions:

- A comprehensive literature study in the field of communications in data centers and mobile environments.
- A new framework of performance metrics for communication systems of cloud computing data centers. The proposed framework allows a fine-grain analysis and comparison of communication systems, processes, and protocols, defining their influence on the performance of cloud applications.
- A novel model for the problem of computation offloading in mobile cloud computing with wearable devices.
- A new Directed Acyclic Graph (DAG) model to describe the workflow of mobile cloud applications in fog computing.
- The development of an Android application, called TreeGlass, which performs recognition of trees using Google Glass. The performance of TreeGlass are evaluated experimentally measuring battery drain and time execution as metrics.
- New performance metrics for mobile crowdsensing systems to assess efficiency of recruitment policies, accuracy of task completion and distribution of generated samples.
- A new framework for data acquisition, which exploits a new policy for user recruitment and runs on a fog computing platform specifically designed for smart cities.
- The development of CrowdSenSim, which is a new simulator designed for mobile crowdsensing activities in large scale urban scenarios.

1.4 Thesis Structure

The manuscript is organized as follows:

- **Part I** presents essential notions required to read the dissertation. The notions include an overview of distributed computing paradigms like cloud computing, mobile cloud and fog computing (Chapter 2), an analysis of communication technologies (Chapter 3) and the study of initiatives to make green distributed computing paradigms and communication networks (Chapter 4).

- **Part II** presents background in cloud computing data centers, including networking, resource allocation and existing performance metrics currently adopted in industry (Chapter 5) and proposes the new framework of metrics for data center communication systems (Chapter 6).
- **Part III** presents background on mobile cloud and fog computing, detailing offloading and application partitioning techniques (Chapter 7) and illustrates the proposed contributions in the field, namely the model for computation offloading with wearable devices (Chapter 8), the DAG-based methodology and *TreeGlass* (Chapter 9).
- **Part IV** presents background on mobile crowdsensing (Chapter 10) and illustrates the contributions, the new performance metrics and the CrowdSenSim simulator (Chapters 11) and the new framework for data collection with the fog computing platform (Chapter 12).
- **Part V** concludes the work and outlines future research directions (Chapter 13).

Fig. 1.1 shows graphically the structure of the dissertation, outlining in blue the part containing the essential notions required to read the technical contributions (in the red boxes). In each technical part, the first chapter is devoted to present a more in depth background on the topic than the content presented in Part I. Finally, the green box presents the conclusions.

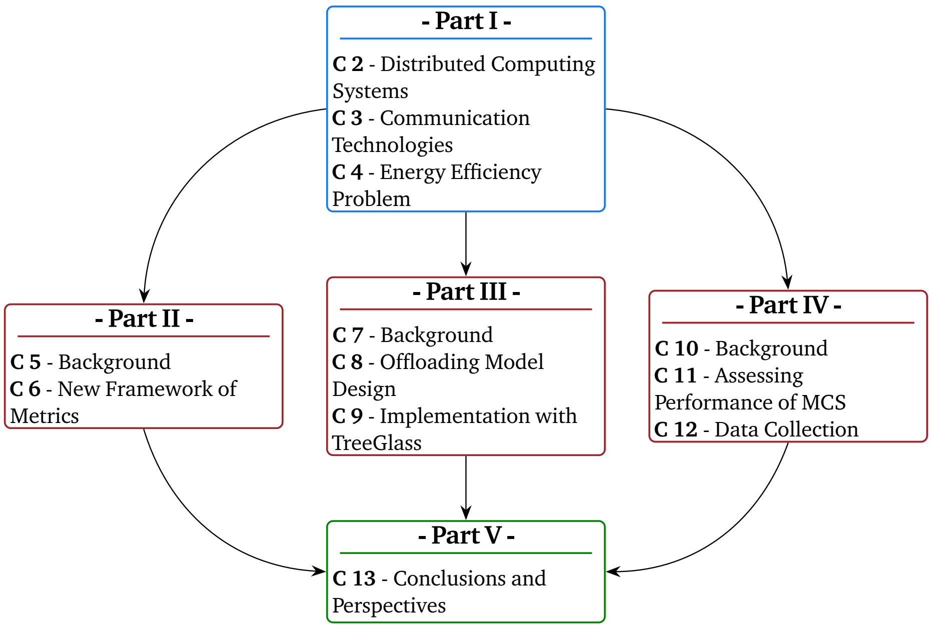

- Part I -
C 2 - Distributed Computing Systems
C 3 - Communication Technologies
C 4 - Energy Efficiency Problem
- Part II -
C 5 - Background
C 6 - New Framework of Metrics
- Part III -
C 7 - Background
C 8 - Offloading Model Design
C 9 - Implementation with TreeGlass
- Part IV -
C 10 - Background
C 11 - Assessing Performance of MCS
C 12 - Data Collection
- Part V -
C 13 - Conclusions and Perspectives

Figure 1.1: Dissertation Structure

Part I

Background, Preliminaries and Basic Notions

Chapter 2

Distributed Computing Systems

Distributed computing refers to the model where computation problems are divided into small pieces, usually called tasks, that are executed by multiple devices with the objective of improving efficiency and performance. This chapter analyzes the most widely adopted distributed systems nowadays, including cloud computing, mobile cloud computing and fog computing among the others.

2.1 Cloud Computing Paradigm

2.1.1 Introduction and Definition

The idea behind cloud computing, i.e., the provisioning of computing capabilities at user disposal is not new. John McCarthy in 1961 was already envisioning computing as general utility and Leonard Kleinrock in 1969 said: "As of now, computer networks are still in their infancy, but as they grow up and become sophisticated, we will probably see the spread of 'computer utilities' which, like present electric and telephone utilities, will service individual homes and offices across the country". Cloud computing has emerged from a combination of economical needs and technological advances and describes a new business model of provisioning services across the Internet.

The National Institute of Standards and Technology (NIST) provides the following definition for cloud computing [7]:

> Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

NIST, also identifies five essential characteristics of cloud computing paradigm, namely on-demand self-service, broad-network access, resource pooling, rapid elasticity, and measured service. These characteristics describe the capability for individuals and companies to benefit from cloud services without having to invest in acquiring specific resources. On the contrary, resources are rented according to the needs on a pay-as-you-go basis. The needs of the users vary along time, hence cloud providers can adapt the provisioning of resources according the requirements and not the peak load and save considerable amount of operational costs at low loads. The pool of shared computing resources are typically hosted in data centers and are accessible to the users in a multi-tenant model, i.e., physical and virtualized resources becomes assigned in elastic manner according to the users requests. Virtualization allows to abstract the physical details of the hardware to create virtual resources at disposal of applications [8].

It is worthwhile highlighting the difference between scalability and elasticity:

- *Scalability* is the ability of the system to accommodate incoming loads by increasing the amount of resources available. Two methods can be employed to scale the system. One methodology advances hardware capabilities (scale up), the second one adds additional nodes in the system such as computing servers or network devices (scale out).

- *Elasticity* is the ability of the system in adjusting the available resources according to the load. Therefore, incoming increasing loads are accommodated through scale out techniques and when the demands shrink, unnecessary resources are released. Elasticity makes possible the pay-as-you-go property of cloud services.

2.1.2 Service Models

The underlining business model behind cloud computing is the vision of “everything as a service” (XaaS), where the word “everything” is typically substituted with a well defined resource put at user disposal [9], [10]. Three are the main type of resources consumed as a service: infrastructure, platform and software. Infrastructure as a Service (IaaS) providers offer to the users a pool of computing (can be a physical of virtualized server), storage and network resources . The users rent the resources and pay according to the consumption similarly to a electricity bill. With IaaS, users can install, manage and run any platform or software on top of the infrastructure. Platform as a Service (PaaS) enable users to access a platform to develop and deploy software. With PaaS model, the developers accelerate the software design and development as well as testing, and deployment of applications in a more simpler and cost-effective manner. Moreover, eliminating the overhead of deployment and management allows the developers to focus only on the core parts of the new application. With Software as a Service (SaaS), users access software running on servers. The vast majority of SaaS applications can be run directly from a web browser. For the users, SaaS model eliminates the need for installing and running the application on a personal computer. For the providers, SaaS model eliminates high costs of maintaining the application and supporting the customers.

With the development of the market, more resources such as hardware, storage and network have become "offered as a service". Hardware as a Service (HaaS) is the business model for replacing outdated hardware equipment with new one. For the companies, it translates in avoiding high financial costs for major hardware upgrade, turning them into operational costs. In Storage as a Service (StaaS) models, the providers rent physical storage space to small companies or individuals. For companies, StaaS allows to save costs in terms of personnel and hardware acquisition and maintenance. Network as a service (NaaS) providers deliver to their users virtual network services. For example, small network operators can rent infrastructure from big operators and sell to the customers their access capabilities. This typically applies to cellular networks, where the cost of acquiring and running a Evolved Packet Core (EPC) is high (see Section 3.2.1 for more details).

2.2 Mobile Cloud/Edge/Fog Computing Paradigm

2.2.1 Introduction, Overview and Comparison

The Internet of Things (IoT) is a paradigm where everyday life things and objects become "smart", i.e., through wireless and wired connections they can communicate one with each other and with users to enable pervasive and ubiquitous computing [11], [12]. IoT devices are objects uniquely identifiable and are equipped with communication, computing, storage and sensing capabilities. Cooperation among different objects is essential to foster the creation of new application and services. According to Gartner [13], the number of connected things will reach nearly 21 billion by 2020 with an increase of nearly from 2016 estimations. The use of IoT devices is expected to generate $868 billion in 2016 and it is projected to surpass the Gross Domestic Product (GDP) of Germany by 2030 (see Fig. 2.1).

Fig. 2.2 shows graphically the sectors where companies started investing in IoT. The sectors are highly heterogeneous ranging from healthcare and fitness to gaming, robotics and automotive and agriculture. IoT devices, indeed appear to be suitable both for systems that require interconnection and coordination among the devices and with humans.

Mobile and IoT devices can perform the computation *locally* or it could be fully *outsourced* to the cloud using the mobile network connection. The

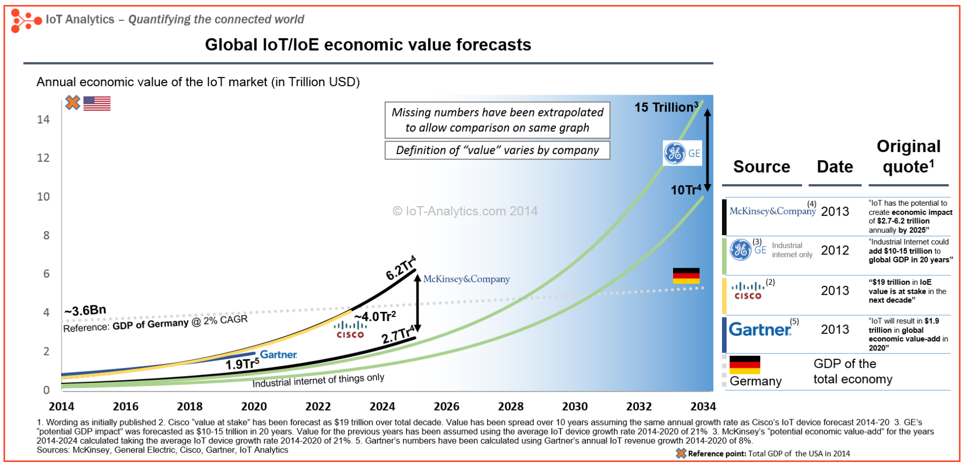


Figure 2.1: IoT market value forecasts. Source: IoT Analytics, Quantifying the connected world

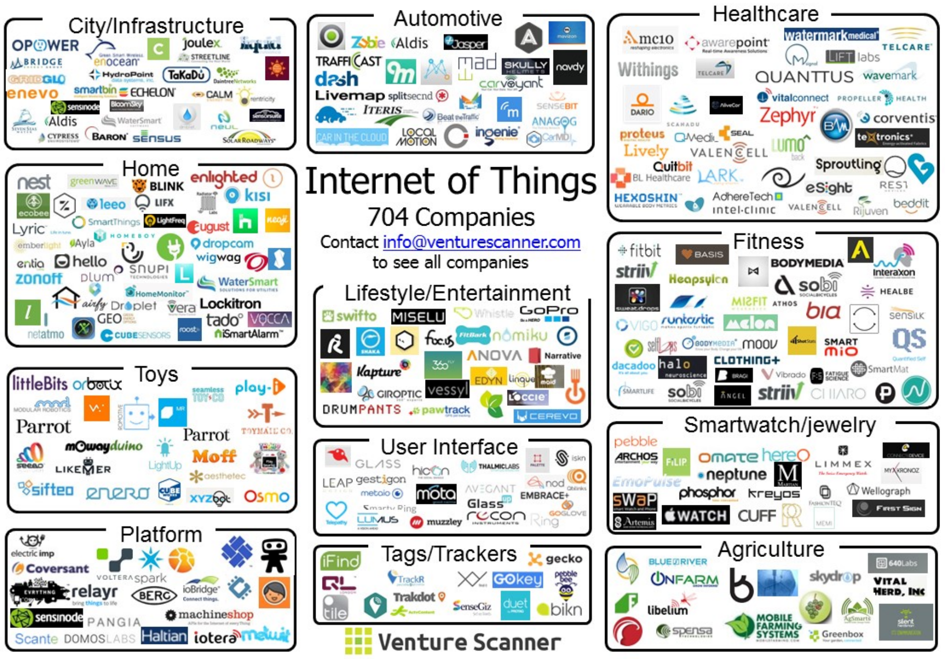


Figure 2.2: Visual map of IoT companies group per type of business. Source: Venture Scanner, Internet of Things Sector Update

second case is called *mobile cloud computing*. In between these two extremes, there are hybrid architectures in which computation is split between the devices and the cloud at various degrees. While traditional mobile cloud computing offers the highest computational power, they may lack the *context* that is typically of interest to the mobile devices in an environment. Due to the delay in accessing the cloud architecture, the *here* and *now* context may not be provided accurately. Moreover, the context may not be known to the cloud. For example, finding the nearest empty parking spot in an event would likely be provided more effectively by local devices. IoT devices are typically tiny devices and thus should operate using low power communication technologies. The vast majority of IoT devices communications are Machine to Machine (M2M) wireless communications [14]. Examples of such wireless technologies include WiFi and WiFi Direct, which implements advanced security standards with respect to existing WiFi ad-hoc networking, Bluetooth, ZigBee, IEEE 802.15.4, Z-wave, and LTE-Advanced. For proper design of energy-efficient context-aware applications, it is of paramount importance to consider the tradeoff between energy savings, computation offloading, and the cost of keeping active the communication interfaces for each distributed computing paradigm. Part III will elaborate in detail this concept.

Mobile Cloud Computing

Mobile Cloud Computing (MCC) is the natural extension of the cloud computing paradigm to mobile environment and is defined as the scenario where data storage and processing take place outside the mobile devices [2], [15]–[19]. The objective is two-fold: on one hand, offloading processing and storage to the cloud augments the capabilities of mobile devices, which are typically resource-constrained. On the other hand, avoiding local processing allows mobile devices to lower energy consumption due to reduced processing. However, offloading requires devices to keep active connectivity with the cloud. As a result, energy savings are a trade-off between the gain achieved from computation offloading and the cost increase due to additional communication. This concept will be explained in details in Chapter 8.

Fog/Edge Computing

Cisco proposed the *Fog Computing* (FC) concept to extend traditional cloud computing at the edge of the network [20], [21]. For this reason, fog computing is also called *edge computing*. Fog computing is tailored to serve applications that are geographically-distributed and require low latency and context awareness. As a result, FC is projected to play an essential role in the IoT framework [22]. Unlike MCC, FC infrastructure is highly heterogeneous and consists of both traditional cloud data centers, cloudlets, and other mobile devices at the edge of the network. Being more agile and closer to the end user, fog/edge computing infrastructure suits the IoT-based applications and context-awareness concept perfectly.

Transient, Follow-me, and Femto Clouds

In recent years, researches have proposed other distributed computing paradigms complementing MCC and FC:

Transient Clouds (TCs) are mobile clouds created on-the-fly by the devices present in an environment and would disappear as the nodes leave the network [23]. Such devices share their resources while they are a member of the TC.

Follow-me Clouds (FMCs) enable mobile cloud services to follow mobile users while traveling by migrating all or part of the service from the original data center to an optimal one [24] to guarantee a sufficient level of Quality of Service and Experience (QoSE). In the context of context-aware applications, MCC and FMC are similar from an infrastructure perspective, because both of them rely on computation offloading to remote datacenters.

Femto-Clouds (FeCs) are designed to cope with user mobility, but computation offloading takes place at the edge of the network, namely in the access network [25]–[27]. In FeCs, femto antennas – augmented with computing capabilities – become available to the mobile users for computation offloading to maximize local computation and reduce latency. Unlike the current definition, Habak et al. [28] have proposed Femto clouds with the aim of exploiting underutilized computing capacity of mobile devices in the vicinity. This proposal, however, matches perfectly the concept of FC.

Selecting the Optimum Paradigm

Table 2.1 compares the distributed computing paradigms in terms of i) computing capacity (COM CAP), ii) availability (AVA), iii) reliability (REL) and iv) heterogeneity (HET) of the infrastructure. Data centers offer virtually-unlimited computing capacity with high standards of availability and reliability, but result high access times for IoT devices. On the contrary, offloading computation to edge devices is faster, however, devices at the edge of the network such as cloutlets, laptops, smartphones, and tablets have limited computational capacity and are less reliable compared to data centers; such devices are battery-powered and have intermittent network connectivity. Furthermore, edge devices are heterogeneous, i.e., they offer highly variable computation capacity, storage and connectivity.

Reducing the distance between the computational sources and the local environment by utilizing cloudlets, nearby devices from fog, and transient cloud architectures is crucial to enable effective context-aware applications.

The complexity of the algorithms used to determine the context dictates where the computation should take place: Distant cloud resources can provide complex data analysis for long-term impacts and projections, while the local cloud (fog or transient) can provide simpler but faster results based on the behavior of the device. For example, in a health monitoring app, local data involves the activity of a user and his surrounding group, while the cloud can provide a rich data analysis by taking into consideration other correlated information, obtained from other sensors.

Table 2.1: Comparison of distributed computing paradigms. CD means Cloud Data Center while ED Edge Devices; H denotes High, M stands for Medium and L is Low.

Computing paradigm	Offloading Destination	COM CAP	AVA	REL	HET
MCC	CD	H	H	H	L
Fog/Edge	CD + ED	H	M	H	H
TCs	ED	M-L	L	M	M
FMCs	CD	H	H	H	L
FeCs	ED	M-L	M	M	M

2.2.2 Service Models

The proliferation of IoT devices equipped with different sensors and communication capabilities along with the emerging computing trends described in the previous section generate tremendous and unprecedented amount of data, also known as Big Data. In this context, IoT devices are the new frontier of sensor networks, just more powerful. Unlike sensor networks, three main features characterize IoT with respect to sensors networks [29]: periodic sensing, regular data collection and sense-compute-actuate loop [30]. As a result, novel services like Data as a Service (DaaS) and Sensing as a Service (S^2aaS) originated from the combination of IoT and Big Data.

DaaS provides to customers data files such as text, images, sounds, and videos. This model allows companies to better manage data, to abstract data from the origin applications, to filter out noise and to provide a better data understanding. Oracle supports DaaS business model through Oracle Data Cloud and Oracle ID Graph.

S^2aaS makes available to the public data collected from sensors. Consequently, companies have no longer the need to acquire an infrastructure to perform a sensing campaign. IoT and MCS are key enablers in the S^2aaS model, which in turn is envisioned to play are indispensable role in smart cities. Efficiency of S^2aaS models is defined in terms of the revenues obtained selling data and the costs. In S^2aaS, the organizer of a sensing campaign, such as a government agency, an academic institution or business corporation, sustains costs to recruit and compensate users for their involvement [31]. Also the users sustain costs while contributing data. These costs are the energy spent from the batteries for sensing and reporting data and, eventually, the data subscription plan if cellular connectivity is used for reporting.

IoT devices producing open data through S^{2}aaS-based systems such as mobile crowdsensing (see Part IV for definition and contributions to the field) can enhance the services in several areas, including tourism and cultural heritage with context-aware recommendation systems, healthcare, security and environmental monitoring [32]. To illustrate with few examples, applications in healthcare domain range from fitness and wellbeing to activity recognition, which can be employed to detect a fall and user reaction after the fall, as well as monitoring of less active users and encourage them for more activity. In the context of tourism, provisioning tourists with wearable bracelets creates the capability for the promoters like hotels and museums to make customized advertisements, while the users to benefit of particular discounts. The municipality, at the same time and with no further expenses, can control the flow of the tourists and obtain information about humidity, temperature, noise for environmental monitoring through the apposite sensors in the bracelets and road surface conditions through the accelerometer and gyroscope.

Chapter 3

Communications in Distributed Computing Systems

3.1 Wired Communication Systems

Wired communications exploit wire-based technologies for data transfer. In distributed computing systems, wired communications find applicability in data center networks and in the core of the cellular network (see Section 3.2.1 for more insights).

The most common technologies used for wire-based communications are electronic and optical. Currently, electronic-based communication systems can support up to 100 Gbits/s for a bi-directional link. The technology is standardized within the IEEE 802.3 working group, which is responsible for maintenance and extension of Ethernet.

Optical communication technologies offer higher throughputs than electronic and data rates are expected to reach the order of terabits per second [33]. Optical communications exploit fiber optics to transmit light from a source to a destination. Fiber optics are made by glass and exhibit low signal attenuation and dispersion, which increases the transmission distance [34].

3.2 Wireless Communication Systems

Wireless communications are certainly the most popular type of communications nowadays. Mobile phones and smartphones provide ubiquitous connectivity and are an essential tool for business and entertainment. Wireless networks are replacing wired networks in homes, companies and campuses and are an enabling technology for important research areas including sensor networks [35], [36], smart homes and buildings [37], smart grids [38] and in data center networks as well [39].

Wireless connectivity bring several advantages over wired networks:

- Support to mobility. For example, students in a campus can move from one lab to another one without losing connectivity;
- Increased coverage. Mobile connectivity can reach areas where fixed infrastructure is not currently deployed;
- Simple configuration. This property ensures flexibility and allows to expand the network easily, for example, welcoming or relocating employees of a company. Moreover, network can be expanded on demand. For example, restaurants, hotels or other public-based business can offer wireless connectivity to their client, which typically consume services for a limited amount of time.

3.2.1 Cellular (3G/4G-LTE/5G)

Mobile users access cloud applications from mobile devices using mainly cellular connectivity. According to Cisco [40] the global amount of traffic produced on a monthly basis by mobile devices will reach 30.6 Exabytes by 2020. Nowadays, the estimations account for 3.7 Exabytes per month at the end of 2015 and 3G and 4G connections represent the 47% and 43% of cellular traffic. The term 3G stands for 3rd Generation and is the most widely implemented and adopted technique nowadays for cellular services. With 3G devices achieved 4 times higher speed than the devices of the previous generation (2G). While 2G and 3G were mainly designed for carrying voice traffic, 4G and 5G technologies are and will be designed for transmission of data such as videos. Hence, it achieves higher data throughput. While in 2015 4G connections represented only 14% of mobile connections [40], 4G is increasingly replacing 3G on the market through LTE. LTE stands for Long Term Evolution and is the name given to the technology able to considerably improves 3G performance (sometimes it is called 3.75G [41]). ITU-R has nominated the advanced LTE solutions (LTE-A) as 4G technology. With LTE-A, users can achieve download and upload data rates up to 300 Mbits and 50 Mbits respectively [42]. Next generation technologies, 5G, is still under development and significant research efforts are necessary even to conceptualize the requirements [41]. One of the pillars of 5G will be the integration of Device to Device (D2D) communications with the objective of relieve the load that high traffic demands are injecting in the backbone of the cellular networks. D2D communications allows users in the vicinity to communicate directly one with each other in the unlicensed spectrum, thus avoiding the need to exploit the cellular infrastructure [43].

Fig. 3.1 shows the LTE architecture. The Packet Data Network Gateway (P-GW) allocates resources to UEs, provides connectivity between the UE and external networks and handles QoS and data encapsulation. Current

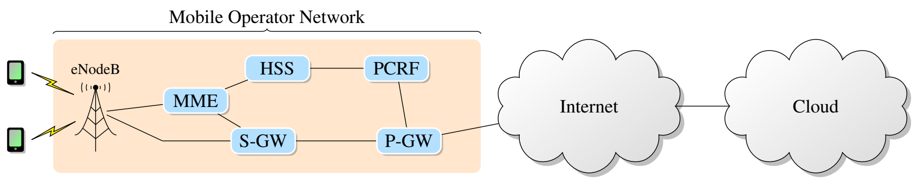


Figure 3.1: LTE architecture

P-GWs usually run Deep Packet Inspection (DPI) for filtering, screening and charging support per user basis. In addition, with the help of the Policy Control and Charging Rules Function (PCRF), the P-GW also provides flow-based charging control decisions. Indeed, the PCRF is in charge of providing to the P-GW the QoS parameters and the bit rates according to subscription profiles of the users. The Serving Gateway (S-GW) handles network traffic before it is delivered to eNodeBs and UEs. The S-GW is responsible of packet forwarding and collecting statistics for billing, i.e., the amount of traffic sent and received by each user. The S-GW also helps in user authentication and interacts with the Home Subscriber Server (HSS). The latter contains user information and subscription profiles containing QoS parameters, such as traffic classes and allowed bit rates. The Mobility Management Entity (MME) controls signaling between UEs and the core network including, for example, procedures for S-GW selection for initial UE attachment and during horizontal handovers. Finally, the eNodeB provides radio connectivity to the UEs and among the other duties, it is responsible for admission control, radio resource management, cyphering/decyphering and IP packet header compression for tunneling data transmission to P-GW. All the described modules but the eNodeBs are called the Evolved Packet Core (EPC) network.

With regard to delays, P-GWs are responsible of high delay. P-GW are expensive hardware and even large operators can not afford to buy and maintain a large number distributed all over a country. For example, the operators in UK have installed a limited number of P-GW only in London. Small operators, typically find more convenient to rent P-GW functionalities from larger operators in form of NaaS (see Section 2.1.2).

3.2.2 WiFi and WiFi Direct

WiFi stands for "*Wi*reless *Fi*delity" and is a technology for Wireless Local Area Networks (WLANs). WLANs limit their extension to a given geographical area and can work both in ad hoc mode or with an infrastructure. In infrastructure mode, the devices communicate with a single Access Point (AP), typically a wireless router. Viceversa, in ad-hoc, which is also known as peer-to-peer mode the devices do not require an AP to communicate. Devices on the network all communicate through a single access point, which is generally the wireless router. In WiFi Direc, each device can become AP, therefore they can dynamically change role from client to AP [44]. WLANs were developed as the mobile extension of wired LANs. Consequently, they support high bit rates and are suitable for both data and real-time traffic. According to the WiFi Alliance, the generic term WiFi pertains to any type of WLAN product based on any of the IEEE 802.11 standard. IEEE has released the first version of 802.11 standard, also known as 802.11 legacy, in 1997 and later many developments have been carried on with the objective of extend the capabilities or to adapt the standard in particular cases.

With 802.11 legacy, devices could transmit over the unlicensed Industrial Scientific Medical (ISM) frequency band at 2.4 GHz. The maximum operating sending rate was 2 Mbits using either frequency-hopping spread spectrum (FHSS) or direct-sequence spread spectrum (DSSS) as access techniques. The most widely adopted standards have been 802.11a/b/g. 802.11a operates at 5 GHz frequencies and uses the orthogonal frequency division multiplexing (OFDM) radio technology, which allows to reach higher throughputs (up to 54 Mbits) since it distributes data over multiple frequency channels. 802.11b extends 802.11 legacy supporting data rates up to 11 Mbits according to the signal strength using DSSS only, while 802.11g supports the same data rates of 802.11a, but is designed to be compatible with 802.11b using 2.4 GHz frequencies. Nowadays, the most common implemented standard in smartphones is 802.11n. Unlike precedent standards, 802.11n features Multiple Input Multiple Output (MIMO) antennas, which allows to achieve higher throughputs. 802.11n maintains compatibility with b/g standards using 2.4 GHz frequencies, but works best at the 5 GHz frequency band.

Because of its design purposes, 802.11 offers high throughput to a limited number of connected devices both in ad hoc or infrastructure modes. As a result, it is not suitable for IoT devices like sensors, which operates well with lower throughputs, but require connectivity with a potentially high number of other devices. For this reason, the IEEE 802 LAN/MAN Standards Committee formed the IEEE 802.11ah Task Group (TGah) to extend 802.11 standard for IoT domain, [45]. The WiFi Alliance has recently announced that the 802.11ah standard, which is expected to be approved in 2016, will be known as WiFi HaLow.

3.2.3 Bluetooth, ZigBee and NFC

The vast majority of wearable devices is equipped with short range connectivity technologies, including Bluetooth, ZigBee and Near Field Communications (NFC). If compared to WiFi, the commonality these technology share is lower transmission range and lower bit rates. Bluetooth is a standard (IEEE 802.15.1, no longer maintained) for wireless technology created on purpose for short-range communication. It exists two types of classes of devices:

- Class 1 devices support communications up to 100 meters (typically

20-30 meters);

- Class 2 devices support communications up to 30 meters (typically 5-10 meters).

Bluetooth finds applications in domains that require to synchronize devices locally. Any Bluetooth device can be connected with several other devices communicating in point-to-point or or point-to-multipoint fashion. Thus it finds applicability in connecting keyboards, headsets, printers, trackpads and speakers with computers or cars.

The original versions of Bluetooth were quite demanding from an energy point of view. Version 4.0, developed in 2011 has mitigated such problem and it is called Bluetooth Low Energy (BLE). A device running BLE has the potential to last up to five years.

ZigBee was originally designed before BLE to be a low power demanding alternative solution to Bluetooth. ZigBee has been standardized by IEEE in the 802.15.4 standard and supports communications in the range 10-100 meters. Similarly to Bluetooth and BLE, Zigbee operates within the 2.4 GHz ISM frequency band. ZigBee finds applicability in automation, security systems, smart lighting among the others.

To conclude, Near Field Communications (NFC) are designed for contactless, very short-range type of communications (in the order of 10 cm). NFC is compatible and can operate with Bluetooth and finds applicability in micro payments with credit cards at supermarket or of subway and train tickets.

Table 3.1 compares properties of the most important wireless technologies according to the following properties: i) maximum data transmission rate, energy consumption, transmission range and type of spectrum utilized.

Table 3.1: Comparison of most popular wireless technologies

Technology	Max Data Rate	Energy	Range	Spectrum
Cellular	Up to 300 Mb/s	High	More than 10 km (1 km)	Licensed
WiFi (802.11n)	Up to 100 Mb/s	Medium	Up to 250 m (120 m)	Unlicensed
Bluetooth	Up to 3 Mb/s	Low	Up to 100 m (20-30 m)	Unlicensed

Chapter 4

Energy Efficiency in Distributed Computing Systems and Communication Networks

Energy consumption is a crucial problem and researchers in distributed computing systems and communication networks have started to investigate proper remedies. In distributed computing systems, the vast majority of energy is consumed by data center facilities. Only in the US, data centers consumed about 70 billion kWh of electricity in 2014, which corresponds to the consumption of nearly 6.4 million average homes in the country [46]. Communication networks, including wired and wireless networks, are an important contributor to energy consumption. In a recent analysis, Andrae et al. [47] state that by 2030 communication networks will account for 51% of the global electricity consumption if proper countermeasures are not put in action. The study takes into account the forecasts of traffic growth (see Section 3.2.1 for insights on mobile traffic growth) and energy trends for fixed, residential networks, mobile networks and data centers as well as actions already taken in using renewable energy.

Reducing energy consumption is worldwide recognized as an essential target to achieve and attracted interest of both governments, industry and consumers for a number of reasons. First, the growing environmental concerns advocate proper measures to reduce greenhouse gas emissions (GHG) and carbon footprint to alleviate the threats to the environment. The second reason is economic, as sustainable development is much more effective from an economical point of view. High energy consumption impact considerably on the energy bill the industry has to sustain. Moreover, companies actively engaged in developing *green* initiatives reinforce their brand and are more attractive from a market perspective.

The following sections analyze initiatives to make *green* distributed computing systems (Section 4.1) and communication networks (Section 4.2).

4.1 Energy Efficiency in Distributed Computing Systems

In cloud computing systems, data centers are the major contributors to energy consumption. Data center performance and efficiency can be expressed in terms of the amount of supplied electrical energy that is actually turned into computing power. Data centers require a tremendous amount of energy to operate. In Europe, data centers are forecasted to consume up to 93 TWh by 2020 [48]. According to Zheng et al. [49], 76% of this consumption is attributed to the IT equipment while the remaining 24% is lost in cooling (12%) in power distribution (7%) and other facility operations (5%). More details on energy efficiency in cloud data centers is provided in Chapter 5.

In mobile cloud and fog computing systems, energy efficiency solutions have been developed to reduce the battery drain of the devices. The objective can be achieved by reducing the contribution given by the computation, the communications or the combination of the two. Practically, this translates in putting in force techniques such as offloading and application partitioning to relive the load of the mobile devices. Chapter 7 provides more details on this regard, while Section 4.2.2 illustrate techniques concerning only the networking part.

4.2 Energy Efficiency in Communication Networks

4.2.1 Green Wired Networks

Wired networks are typically used in the core of the Internet. Energy consumption of transport and core networks accounts for nearly 30% on the overall energy budged of a typical Internet Service Provider (ISP)/telecom operator network configuration, while 70% is attributed to access devices [50]. Despite this huge unbalance, making green core networks provides substantial savings. During 2012, the german ISP Deutsche Telekom Group reported a total annual energy consumption of 4.11 TWh of which approximately 3 TWh accounted in form of electricity and the remaining in form of fossil fuels, district heating, and fuels for the vehicle fleet. This consumption corresponds of 0.5% of the needs of the overall country [51].

According to Bolla et al. [50], three are the possible approaches to achieve energy savings in wired networks: i) re-engineering, ii) dynamic adaptation, and iii) smart sleeping. With re-engineering approaches, new technologies that are more energy efficient replace old ones. For example, optical technologies are replacing more and more links of the core network as they are capable of supporting higher throughputs and are more energy efficient than electronic ones. Exploiting dynamic adaptation makes the technology adjusting the provisioning of network resources to current level of demands. A popular technique is Dynamic Frequence Voltage Scaling (DVSF), which adjust the power consumption according to the offered load. Finally, smart sleeping techniques makes the devices turning into low state energy or idle mode by freezing some functionalities.

4.2.2 Green Wireless Networks

In wireless networks such as cellular and WLAN, the high density of base stations (BS) and access points (AP) allows on one hand to provide high access speed to the users, but on the other hand it is the main cause of energy consumption. Provisioning wireless network resources is a challenging problem as it is difficult to have precise knowledge on the demands. The demands depend mainly user mobility, which typically follows patterns that can be predicted only with a certain level of accuracy. User mobility patterns are usually divided into two main categories: time-based and location- or activity-based patterns. Time-based mobility patterns are characterized by the timing of user movements, for example during going to or back from work place. Activity- or location-based patterns are characterized by the the destination users reach, for example work place, gym or restaurants. Although it is not strictly necessary, the two types of patterns are usually dependent one each other, i.e., during morning users preferentially go to work. According to Budzisz et al. [52], two types of algorithms are applied to obtain energy savings: offline and online. Offline algorithms generates predefined time schedule when BS and AP are turned on or off. These algorithms are based on historical measures of the users demands and may underestimate the necessary resources when special events occur. Online algorithms instead adjust dynamically the resource provisioning according to real-time measures of users demands, therefore in case of unexpected events they are able to ensure connectivity to the users. One of the remedies to green wireless networks consists in making the hardware more energy-proportional. Nowadays, hardware of wireless networking is not energy proportional as the consumption at low levels of utilization may account 60% to 80% of the peak power consumption. More energy-proportional hardware would make easy to adapt consumption to current user demands.

In wireless sensor networks, the techniques to achieve energy efficiency are more numerous [53]. In sensors, batteries resources are extremely precious therefore devices are designed to go in idle/sleep mode as soon the context makes it possible. Sensors can reduce their transmission power to save energy. This results limiting data transmission only to other sensors that are close. In addition to limit transmission power, routing techniques impact significantly on the sensors decisions of data transmission. Communications in wireless sensor networks are typically multihop, hence the selection of proper paths from source to destination limits energy consumption of all the sensors in the network. Another method to achieve energy savings consists in performing data aggregation, which implies to limit the frequency of transmission of sensing readings. Network coding techniques [54], [55] represent an excellent tool to perform data aggregation.

Part II

Assessing Energy Efficiency of Communications in Data Centers

Chapter 5

Background

5.1 The Role of Communications in Data Centers

Cloud computing has become fundamental for IT operations worldwide and has replaced traditional business models. Enterprises can now access the vast majority of software and services online through a virtualized environment, avoiding the need for expensive investments into IT infrastructure, while covering only costs of infrastructure required to provide connectivity. Instead, they can focus on their core business directly and consume IT services over the Internet on a pay-as-you-go basis. For operation, cloud computing relies on the network of geographically distributed data centers. Cloud data centers are composed of several systems: IT equipment, which is further divided into computing and communication equipment, the cooling system and power distribution system. Among all the component, the IT equipment is essential to guarantee service to end users. Being located outside of the data center, they can access and exploit the data center computing capabilities through the Internet and the data center network architecture. A number of data center network architectures has been proposed in the literature and have been analyzed by extensive surveys [56]–[60]. Selecting the proper architecture is important as it has a direct impact of the overall efficiency, scalability and power consumption of the data center. For the data center operators, measuring these key performance indicators is essential to analyze costs and revenues. Resource allocation is an important factor in the costs-revenues analysis: proper allocation policies permit to exploit efficiently the data center resources while guaranteeing performance of cloud applications that are the primary source of revenues. With the only exception of High Performance Computing (HPC) applications, where the dominant component is the computing part, cloud computing applications are communication-intensive [61]. For example, application like video streaming, cloud storage and backup require high bandwidth to transfer large amounts of data coming from or destined to the end users.

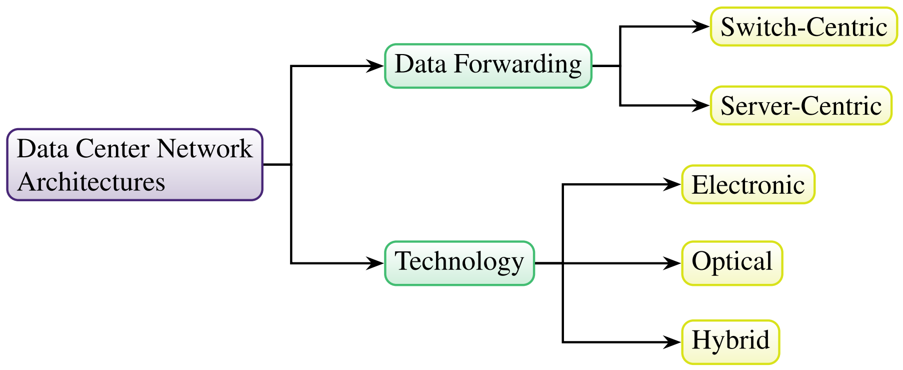


Figure 5.1: Classification of Data Center Network Architectures

The following sections overview and classify data center architectures (Section 5.1.1) and communication-aware resource allocation policies for data centers (Section 5.1.2).

5.1.1 Data Center Architectures

Introduction and Classification

Before analyzing in detail data center architectures and illustrating their classification, it is important to distinguish between the notions of data center *infrastructure* and *architecture* that are often confused. On one hand, the data center infrastructure is defined as the "*home to the computational power, storage, management and dissemination of data and information necessary to a particular body of knowledge or pertaining to a particular business*". On the other hand, the data center network architecture "*is the set of network nodes and links that characterize the interconnectivity among the computing servers and to the external world*".

Architectures can be classified according to which component is responsible for data forwarding or the technology used for communications (see Fig. 5.1). The first classification groups the architectures in *switch-* or *server-*centric. In *switch-centric* architectures, switches are the key components responsible for traffic forwarding while in *server-centric* architectures servers are demanded to perform interconnecting tasks in addition to computation. The main difference between switch- and server-centric architectures is in the hardware used. Server-centric architectures use Commercial Off-The-Shelf (COTS) switches, that simple and low cost devices for two reasons: they are cheap and do not consume large amounts of energy. Viceversa, switch-centric architectures employ very complex, energy-hungry and costly switches. In addition, switch-centric architectures suffer of the problem of bandwidth oversubscription: in layered topologies, servers are prevented to exploit at 100% the available bandwidth because of the difference between ingress and egress bandwidth provisioned in each switch. The ingress and egress bandwidth are respectively the incoming traffic arriving and the outgoing traffic processed by the switch. More details on bandwidth oversubscription problem are given in Section 6.2.2. To summarize the properties:

- Switch-centric:
    - Switches are the key components in data forwarding;
    - Switches are complex;
    - Servers are in charge of forwarding functionalities.
- Server-centric :
    - Servers are the key components in data forwarding;
    - Switches are very simple (COTS);
    - Servers are not in charge of forwarding functionalities.

According to the technology used for data transfer, data center architectures can be grouped in electronic, optical or hybrid. Electronic architectures are the most widely adopted and are summarized in the next subsection. The adoption of optical technology has been investigated mainly to mitigate the communication capacity problem of the electronic data centers. Having new servers equipped with 10 GE links makes the capacity in the aggregation network a severe problem leading to high bandwidth oversubscription ratios [62]. Optical data center networks rely on a mix of active and passive optical devices, being Micro-Electro-Mechanical Systems Switches (MEMS) the most widely adopted [63], [64]. MEMS is a popular technologies enabling matrix optical switching that allow fully non-blocking all-optical connection between a number of input and output fibers. Only few architectures like Proteus are fully optical [64], the majority are hybrid, i.e., like Helios [65], C-through [66] and Flyways [67] incorporate both electronic and optical components. Another optic-based approach is the adoption of free space optical communications [68]. In indoor systems, atmospheric impairments do not represent an issue and the speed of light is approximately 1.5 times faster than that of in fiber optics, which reduces latency. Moreover, unlike copper cables and optical fibers, in free space optical communications links can be re-deployed after deployment.

Overview of Data Center Architectures

A number of surveys has studied and analyzed data center architectures [56]–[60]. In the following, the properties and characteristics of the most popular architectures are briefly summarized. The analysis is limited to switch- and server-centric electronic architectures.

Three-tier Architectures: In a three-tier architecture, the network topology is based on a Clos construction and consists of three layers: at the lowest level, the computing servers are interconnected to the network through access switches. The upper layers, aggregation and core layer, provide connectivity towards the data center gateway. The family of three-tier architectures belongs to the switch-centric category. Several three-tier-like architectures have been proposed, including Al-Fares et al. [69], PortLand [70], VL2 [71].

Jupiter: [72] is the architecture Google currently implements in its data centers. The building blocks of this architecture are heterogeneous. The smallest unit consists of a set of switches that are called TOR and are used to build the blocks of each layer. Inside a block, the switches can be placed on two levels. The Aggregation blocks are divided into sub-groups that are called Middle Blocks, each of them is composed by several TOR switches. The high modularity makes Jupiter one of the most scalable switch-centric architecture that can host up to 400 k servers and 1.3 Pbps of bisection bandwidth.

BCube: [73] BCube is a server-centric architecture specifically designed for container data centers and consists of servers equipped with multiple network ports connected with low cost COTS switches. In BCube two servers are never interconnected directly. BCube is build recursively and two parameters, the number of ports per server k and the number of ports per switch n. Then, the total number of servers and switches is n^{k+1} and $(k+1) \cdot n^k$ respectively. BCube is designed to be easily scalable, cost effective, fault tolerant and to provide high cross bisection bandwidth and load balancing. However, because of the high cabling cost, this architecture was never actually implemented in real datacenters [72].

DCell: [74] is designed to provide rich connectivity among servers and replaces expensive core and aggregation switches of three-tier architectures with COTS switches. Similarly to BCube, DCell is build recursively and the total number of servers and switches can be determined from two parameters n and k. However, in DCell n stands for the number of servers per basic cell, DCell$_0$ while k is the number of DCell levels and the server degree of connectivity is defined as $k+1$. Then, the total number of servers and switches is $t_{j+1} = t_j \cdot (t_j + 1), \quad t_0 = n$ and $g_j = t_{j-1} + 1$ respectively. Unlike BCube, in DCell servers are interconnected directly. DCell exhibits all properties of BCube.

5.1.2 The Impact of Resource Allocation in Data Centers

Resource allocation is essential for proper data center operation and directly impacts overall system performance. The cloud users negotiate certain Service Level Agreements (SLA) with the cloud provider [75], such as execution deadline, service availability etc. Then, is responsibility of the cloud provider to allocate a sufficient amount of computing capacity/memory/storage to not violate the SLA. Traditionally, resource allocation in data center was communication-unaware, but it has been proved to perform poorly with respect to communication-aware schemes. In the following, the popular communication-aware resource allocation schemes are summarized.

DENS [76] is a communication-aware scheduler and stands for Data center Energy-efficient Network-aware Scheduling. DENS takes into account communication aspects through analysis of feedback from the network. Specifically, DENS monitors the status of the queues of outgoing links of Top of Rack (TOR) network switches with the aim of favoring selection of empty queues and penalize fully loaded ones. This helps preventing congestion. Servers selection has the objective of penalize unloaded servers and favor allocation of jobs in servers with high load. It should be noted that overutilized servers are penalized to prevent running at peak load for long time periods, which decreases hardware reliability and affects the job execution deadlines.

Takouna et al. [77] propose Peer VMs Aggregation (PVA), which is a methodology to aggregate in the same server the VMs that exchange traffic one with each other through migration. The objective is to reduce the total traffic in the network by determining the bandwidth demands and the communication patterns of VMs. Based on VM migration, PVA suffers of overheads such as time and bandwidth require to reconfigure the system.

e-STAB [78] stands for energy-efficient Scheduler for cloud computing applications with Traffic loAd Balancing. The name is explicative as its objective is optimization of energy consumption in data centers and it is pursued balancing the communication flows produced by the allocated jobs while consolidating the jobs on a minimum amount of the computing servers. e-STAB was one of the first schedulers assigning equally importance to computing and communication requirements.

HEROS [79] stands for Heterogeneous Energy-efficient Resource allocation Optimizing Scheduler and takes inspiration from DENS [76] and e-STAB [78] methodologies. Unlike DENS, which appears to be most suitable for three-tier like architectures, HEROS can be adopted efficiently from data centers with any network architecture. e-STAB has been proved favoring excessively load balancing with respect to consolidation while HEROS maintains a good tradeoff between load balancing and consolidation by preventing selection of servers that run at too high utilization levels.

Zotkiewicz et al. [80] present an on-line, energy- and communication-aware scheduler for SaaS applications. To each application corresponds a workflow that is composed of several tasks. Tasks are modeled with Directed Acyclic Graphs (DAGs), see Section 7.2.2 for more insights. The scheduler assigns tasks in a two phases-procedure and the scheduling decisions are based on the servers and network utilization levels. In the first phase, the scheduler calculates virtual deadlines for each task of the workflow. In the second phase, tasks are assigned to servers able to satisfy the virtual deadlines similarly to HEROS [79].

Belabed et al. [81] present an optimization framework for VM placement supporting virtual bridging and multipath forwarding. Virtual bridging stands for the capability of a container in taking the function of a bridge at hypervisor level. IEEE has released a standard for virtual bridging, which is called Edge Virtual Bridging (EVB) and was recently enhanced in two different proposals, namely 802.1qbg and 802.1qbh. Multipath forwarding is the capability of ensuring that packets follow the specified paths while being routed. Multipath forwading is part of multipath routing problem, which addresses the usage of path diversity to improve resource utilization, reliability and quality of experience (QoE) in networks. Multipath routing is used in data centers to achieve traffic engineering through load balancing [82].

MAPLE [83] is a communication-aware VM placement. MAPLE is designed to perform optimal allocation of computing and communication resources and provision predictable network performance to guarantee QoS and SLA. It takes into account network requirements by estimating the *effective bandwidth* the communications between servers require in order to prevent Quality of Service violations and guarantee SLA the cloud users or tenants have negotiated. Effective bandwidth stands for the "minimum amount of bandwidth required by a traffic source to maintain specified QoS targets" [84]. Bandwidth estimations are determined in a decentralized fashion with the objective of improving the run-time performance.

5.2 Existing Performance Metrics for Data Centers

Existing metrics that assess efficiency, performance and quality of cloud computing systems can be attributed to the following categories:

- Power and energy efficiency;
- Environment and air management;
- Cooling efficiency;
- Other metrics.

5.2.1 Power and energy efficiency

The most important metrics in this category are the Power Usage Effectiveness (PUE) and Data Center infrastructure Efficiency (DCiE) [85]. PUE is defined as the ratio between the power consumed by the facility and the power consumed by the IT equipment, while DCiE is the reciprocal of PUE. Energy Utilization Effectiveness (EUE) [86] is analogous to PUE, but is based on energy rather than power. Another metric in analogy with PUE and EUE is Carbon Usage Effectiveness (CUE) [87]. The CUE metric analyzes the total

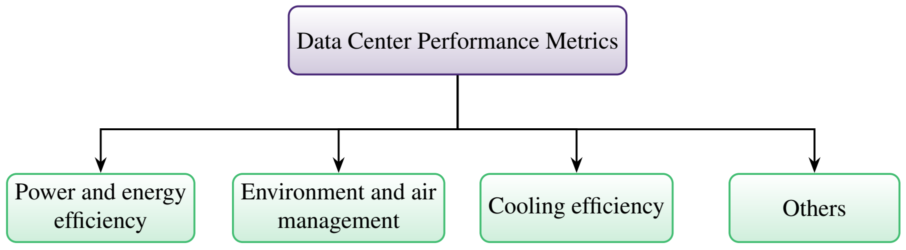

Data Center Performance Metrics
Power and energy efficiency
Environment and air management
Cooling efficiency
Others

Figure 5.2: Classification of Metrics for Data Centers

GHG emissions of a data center facility in relation to the power spent by the IT equipment. The Energy Reuse Effectiveness (ERE) and Energy Reuse Factor (ERF) [88] measure how much energy can be reused outside the data center, and the Uninterruptible Power Supply (UPS) Load Factor [89] accounts for the average UPS load over the total UPS capacity. Two more generic metrics are Data Center Energy Productivity (DCEP) [90] and Power to Performance Effectiveness (PPE) [91]. They evaluate respectively the amount of energy spent to produce useful work and effectiveness of the IT equipment in terms of power consumption relative to the delivered performance. A new metric called Performance per Watt (PpW) [92] defines energy efficiency of a server and can be used for server selection during resource allocation [79].

5.2.2 Environment and air management

One of the most important metrics for environment and air management is the Return Temperature Index (RTI) [93]. RTI evaluates the energy performance of air management in isolating cold and hot airstreams. The Recirculation Index (RI) [94] accounts for the percentage of recirculated air absorbed by a rack while the Capture Index (CI) [95] evaluates the fraction of airflow, entering or exiting a rack, which follows the desired path.

Amokrane et al. [96] propose new metrics to determine how data centers are environmental friendly. The first is a network-based carbon intensity metric, which computes the amount of CO_2 emissions per byte and is measured in tonCO_2/Mbps. Akamai and Verizon, for example, already make use of this metric as they report respectively on annual basis the CO_2 emissions per GiB and TiB of data delivered. The second metric determines the amount of CO_2 emissions per core and is measured in tonCO_2/Core. The objective of these metrics is to strength the use of green SLAs.

5.2.3 Cooling efficiency

The Rack Cooling Index (RCI) [93] is a metric evaluating rack cooling efficiency according to the thermal guidelines provided by manufacturers. Data Center Cooling System Efficiency (DCCSE) [97] assesses the amount of power needed to operate cooling equipment. It is defined as the ratio between the average cooling system power consumption and load of the data center. Airflow Efficiency (AE) [97] assesses the fans and the efficiency of air circulation. The Air Economizer Utilization Factor (AEUF) [97] measures the number of hours in a year during which the air economizer exploits low external temperatures for chilling water. The technique is also known as "free-cooling."

5.2.4 Other metrics

The metrics assessing performance of traditional communication networks typically focus on network latency, bandwidth and error rates as main indicators. The detailed surveys are available in [98]–[100].

Certain aspects of data center networks are explored in [101], [102]. The focus is devoted to the evaluation of latency and bandwidth between pairs of running VMs [101] and the analysis of data center network costs and capacity [102].

To enforce green SLAs, the CP should compute the carbon footprint of each VDC request. To do so, we use two metrics: (1) carbon emission per unit of bandwidth (tonCO2/Mbps) and (2) carbon emission per core (tonCO2/Core). These metrics are chosen because the bandwidth and the CPU are the major fac- tors that determine the power consumption in data centers and they are already considered in industry. For instance, Akamai reports annually its carbon emission in CO 2 per gigabyte of data delivered (tonCO2/Gbps), Verizon reports its carbon emissions per terabyte of transported data across its network

5.3 Open Research Problems

Assessing performance and energy efficiency of data centers has become fundamental. It allows to understand and optimize the operation of existing data centers and it is crucial for the design and construction of next generation cloud computing data centers. Currently, industries and data center operators use a number of metrics to assess efficiency and energy consumption of cloud computing systems. As outlined in Section 5.2, these metrics focus mainly on the efficiency of IT equipment, cooling and power distribution systems. However, none of the existing metrics is precise enough to distinguish and analyze the performance of data center communication systems from IT equipment. Distinguishing communication systems from the IT equipment and assessing their performance is fundamental as most cloud applications follow a Software-as-a-Service (SaaS) model [103], and communication processes, not computing, tend to become the bottleneck limiting overall system performance.

5.4 Contributions

This thesis proposes a new framework of metrics for assessing performance and energy efficiency of communication systems for cloud computing data centers. Unlike existing metrics, the proposed framework allows a fine-grain analysis and comparison of communication systems, processes, and protocols, defining their influence on the performance of cloud applications. The presented framework is being considered for standardization and is positioned to become an essential tool for scientific research in cloud computing and data center industries.

Chapter 6

New Framework of Performance Metrics for Communication Systems in Data Centers

6.1 Introduction and Motivation

Cloud computing has emerged and has become fundamental for IT operations worldwide, replacing traditional business models. Enterprises can now access the vast majority of software and services online through a virtualized environment, avoiding the need for expensive investments into IT infrastructure, while covering only costs of infrastructure required to provide connectivity. Instead, they can focus on their core business directly and consume IT services over the Internet on a pay-as-you-go basis. For operation, cloud computing relies on the network of geographically distributed data centers. Therefore, assessing data center performance is important for understanding the operation of existing data centers and crucial for the design and construction of next generation systems for cloud computing.

Current research and industry standards propose a number of metrics for assessing efficiency of energy distribution [104]–[107] and cooling [93], [108], [109]. The most popular metric used nowadays is Power Usage Effectiveness (PUE) [110]. It measures the portion of the supplied electricity actually delivered to the IT equipment. Unfortunately, most of the available metrics are too generic. They indeed are unable to differentiate between individual IT subsystems. For example, using existing metrics, it is not possible to distinguish the efficiency of the data center communication system from the efficiency of the computing servers, as both remain considered under the common umbrella of IT equipment [111].

Ideally, power consumption of network devices should be proportional to the workload. However, in reality their power consumption is composed of two parts, fixed (for switch chassis and line cards) and variable (for active transmitters). Only the latter scales with the transmission rate, or the presence of forwarded traffic, while the former part remains constant, even when the switch is idle. This phenomenon, known as energy proportionality, describes the relation between energy consumption and offered load in the system or a component. With current network switches, the difference between peak and idle consumption is less than 8%, while turning off an unused port saves only 1-2 Watts [112]. As computing servers' power consumption becomes more proportional to the workload and effective at low utilization levels, network power consumption remains a concern. In certain scenarios, network power consumption can account for nearly 50% of the overall data center power consumption [113]. To assess the degree of energy proportionality for network devices, we propose a metric which can be applied to investigate both the energy proportionality of individual network devices as well as of the whole system.

Distinguishing communication systems from the IT equipment and assessing their performance is very important. Most cloud applications follow a Software-as-a-Service (SaaS) model [103], and communication processes, not computing, tend to become the bottleneck limiting overall system performance [61]. Specifically, latency, available bandwidth or both can become limiting factors. Voice conferencing, for example, imposes severe constraints on the communication latency, but does not require high bandwidth availability. On the opposite side, video streaming, cloud storage and cloud backup applications require high bandwidth to transfer huge amounts of data, but remain tolerant to network delays. Finally, cloud gaming produces high traffic load and requires tight delay constraints to keep players synchronized.

Cloud communications can be categorized according to the direction of information flow: cloud-to-user and intra-cloud. The former is related to serving cloud users located in the access network. This was not needed in the PC era when all data and software were available on user devices. The latter corresponds to the traffic which remains inside a data center. Cisco estimates that network traffic is the fastest-growing data center component, rising to 4.3 ZiB in 2016 with a combined annual growth rate of 44% [114]. Networking solutions, architectures and protocols therefore must be carefully considered to achieve good performance.

6.2 Design

This section defines a framework of metrics that characterize performance and energy efficiency of communication systems in cloud computing data centers.

Cloud applications, with the only exception for High Performance Computing (HPC), are communication-intensive [61]. Therefore, the communication parameters, such as bandwidth capacity, latency and error rate, can affect system performance dramatically. Unfortunately, existing performance and power-related metrics that are widely used in the data center industry (see Section 5.2) fail to distinguish communication systems from the category of IT equipment. The proposed metrics address this gap by allowing finer granularity. At the same time, they remain general and intuitive to be universal and applicable to the vast majority of existing data centers and their communication systems.

The proposed metrics can be broadly attributed to the following three categories:

- *power*-related metrics;
- *performance*-related metrics;
- *network traffic*-related metrics.

Power-related metrics assess energy efficiency of communication systems by analyzing how much of the electric power is actually turned into the work of information delivery. *Performance*-related metrics analyze communication rate, capacity, and latency for information delivery. Finally, *network traffic*-related metrics provide insights into the nature of the transmitted information and help to measure control traffic overheads. Table 6.1 summarizes the power, performance and traffic related metrics presented in the following sections.

6.2.1 Power-Related Metrics

Communication Network Energy Efficiency

The communication network turns the supplied electricity into the job of information delivery. The efficiency of this process can be measured by the metric Communication Network Energy Efficiency (CNEE):

$$\text{CNEE} = \frac{\text{Power Consumed by Network Equipment}}{\text{Effective Network Throughput Capacity}}. \tag{6.1}$$

The data center network equipment includes all the hardware components that take part in information delivery between servers, including network switches, routers, communication links, and Network Interface Cards (NICs) of the servers[1]. The effective network throughput capacity is a maximum end-to-end throughput offered by the network to the computing servers. In the context of this paper, the terms “computing servers” and “servers” are

[1]The servers, excluding their NICs, are considered to be devoted to computing and not as a communication equipment.

Table 6.1: Communication Metrics for Cloud Computing Data Centers.

TYPE	METRIC	NAME	DESCRIPTION
Power	CNEE	Communication Network Energy Efficiency	Energy to deliver a single bit of information
	NPUE	Network Power Usage Effectiveness	Ratio between total IT power and network equipment power
	EPC	Energy Proportionality Coefficient	Degree of energy proportionality of a device or a system
Performance	UDCL	Uplink/Downlink Communication Latency	Communication latency between DC gateway and computing servers
	UDHD	Uplink/Downlink Hop Distance	Hop distance between data center gateway and computing servers
	ISCL	Inter-Server Communication Latency	Communication latency between computing servers
	ISHD	Inter-Server Hop Distance	Hop distance between computing servers
	DAL	Database Access Latency	Average latency of accessing database from computing servers
	BOR	Bandwidth Oversubscription Ratio	Actual bandwidth servers can exploit under full load
	UDER	Uplink/Downlink Error Rate	Error rate of the paths between data center gateway and servers
	ISER	Inter-Server Error Rate	Error rate of the network paths between computing servers
	ALUR	Average Link Utilization Ratio	Average link level occupancy
	ASDC	Average Server Degree Connectivity	Average number of network links per server
Network Traffic	ITR	Internal Traffic Ratio	Traffic exchanged within the data center
	ETR	External Traffic Ratio	Traffic destined outside the data center
	MMTR	Management and Monitoring Traffic Ratio	Traffic generated by management and monitoring operations
	MMTE	Management and Monitoring Traffic Energy	Energy consumption of management and monitoring traffic

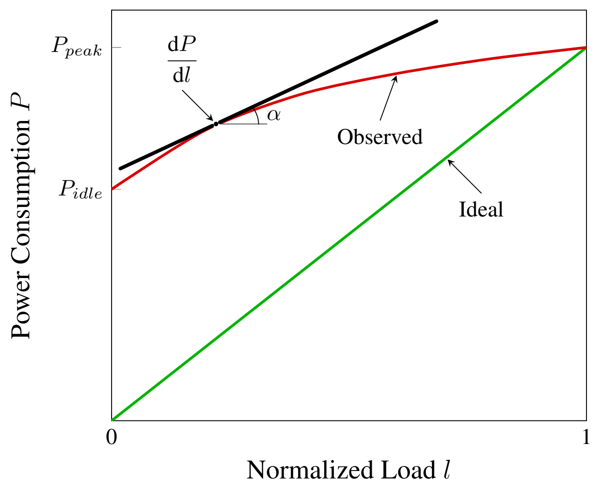


Figure 6.1: Energy proportionality

used interchangeably. The CNEE is measured in Watts/bit/second, which is equivalent to joules/bit, or the amount of energy required to deliver a single bit of information by the network.

Network Power Usage Effectiveness

Another measure of the communication system effectiveness is in the power consumed by the network equipment as a fraction of the total power consumed by the IT equipment. This metric is called Network Power Usage Effectiveness (NPUE) and is defined as follows:

$$\text{NPUE} = \frac{\text{Total Power Consumed by IT Equipment}}{\text{Power Consumed by Network Equipment}}. \tag{6.2}$$

NPUE specifies which fraction of the power consumed by the IT equipment is used to operate data center communication system. In a similar way PUE [85] measures the portion of the amount of energy used by a data center facility that is delivered to power IT equipment. The NPUE metric assumes values greater or equal to 1. For example, for NPUE equal to 6 for every 6 Watts consumed by IT equipment, 1 Watt is devoted to operate network equipment. The NPUE value equal to 1 corresponds to the system where all the IT-related power is consumed by the network equipment, which is a not desirable target: if all the IT power is consumed by the network equipment, there is nothing left for computing servers. However, NPUE values approaching 1 are not necessarily symptoms of network inefficiency. It can signal that the computing servers were upgraded and became more energy efficient.

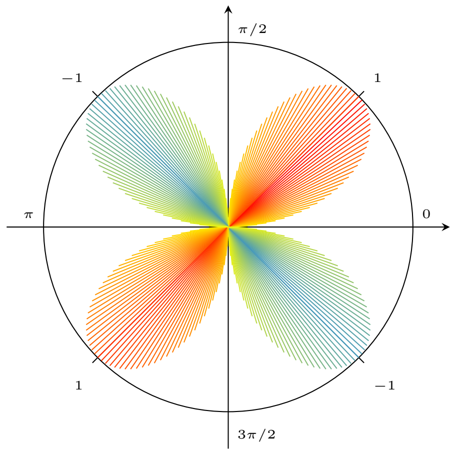


Figure 6.2: Energy Proportionality Coeffient (EPC)

Energy Proportionality Coefficient

Ideally, energy consumption of network devices should be proportional to their workload. However, in reality neither computing servers nor network switches are energy proportional. Many servers consume up to 66% of their peak power consumption when idle [115]. For network switches this ratio is even higher and can reach 85% [116].

Energy Proportionality Coefficient (EPC) is measured as energy consumption of a system or a device as a function of the offered load. In the ideal case, represented by a straight line in Figure 6.1, every increase in load l should correspond to the equivalent increase in power consumption P. In reality, the observed power consumption is often non-linear. Its energy proportionality varies depending on the incline with the respect to the ideal case. To analyze this deviation for continuous power consumption profiles, a tangent line can be built at every point to the observed curve. The angle of this tangent line α can be obtained by computing the first derivative of the observed function:

$$\tan\alpha = \frac{\mathrm{d}P}{\mathrm{d}l}. \tag{6.3}$$

Having $\tan\alpha$ we can define the measure of energy proportionality as follows:

$$\mathrm{EPC} = \int_0^1 \sin 2\alpha(l)\,\mathrm{d}l = \int_0^1 \frac{2\tan\alpha(l)}{1+\tan^2\alpha(l)}\,\mathrm{d}l. \tag{6.4}$$

Figure 6.2 plots the values of EPC metric for different values of α in polar coordinates. For $\alpha = \pi/4$, which corresponds to a fully proportional case where each increase in the system load leads to an equal increase in energy consumption, EPC is equal to 1. On the contrary, for $\alpha = -\pi/4$, which means for every increase in the system load the energy consumption is equally decreased, the EPC is equal to -1. In between, EPC turns to zero for $\alpha = 0$, which describes the case when system energy consumption is constant and does not depend on the load, and $\alpha = \pi/2$, which is the asymptote of the power consumption function.

Energy proportionality has been first discussed for computing servers [117] and then for network equipment [113], [118]. In [118] the authors analyze how different routing strategies, energy-aware and energy-unaware, affect energy proportionality of several data center network topologies.

Several metrics evaluating energy proportionality have already been proposed in the literature. The Energy Proportionality Index (EPI) [116] captures the difference between the measured power and the ideal power, the power that the device should consume if it is fully energy proportional. EPI is expressed through the idle and peak power only. EPI equal to zero shows that the energy consumption is agnostic to the workload, while EPI equal to 100 % indicates that the device is fully energy proportional.

The Idle-to-peak Power Ratio (IPR) and the Linear Deviation Ratio (LDR) [119] measure the ratio between the idle and the peak power consumptions and deviation of the observed power consumption from the fully proportional case respectively. IPR values tending to zero indicate energy proportional designs. LDR, instead, measures maximum deviation of the power consumption from a straight line connecting idle and peak power consumption values. Positive LDR values indicate that the measured power is above the line, while negative values are for the measured power below the line. When power consumption is perfectly linear, the LDR is equal to 0.

Unlike other existing metrics, EPC is able to express energy proportionality of a device or a system in every point of the observed power consumption, for any load level, allowing more accurate estimation. In contrast, EPI and IPR depend only on idle and peak power consumptions and LDR depends only on the absolute value of the highest deviation from a fully proportional case. Similar to EPC, EPI can be computed considering angles of ideal and measured power consumption functions. However, the functions where energy remains constant with the increase in the workload are not taken into account. EPC, instead, can differentiate between constant and non-constant functions.

6.2.2 Performance-Related Metrics

Cloud computing systems provide on-demand access to the pool of shared computing resources over the Internet. Therefore, communication processes, not computing, often define the efficiency of the cloud. In this section, we present a set of metrics which capture performance and describe energy efficiency of data center communication systems. These metrics combine traditional performance characteristics, such as bandwidth and delay, and data center specific parameters, such as degree of servers' connectivity.

Network Latency

Cloud applications are found to be extremely sensitive to communication delays [61], [120]. Therefore, an ability to monitor and control network latency is especially important to guarantee Quality of Service (QoS) and Service Level Agreements (SLAs). Network delays are composed of signal transmission time, channel propagation delay, packet processing delays at every node and queuing delays. As a result, communication latency is proportional to the number of hops between information senders and receivers.

The most important latency-related metric is Uplink/Downlink Communication Latency (UDCL), or Uplink/Downlink Hop Distance (UDHD) if expressed in the number of hops. UDCL measures the time (in seconds) needed for an incoming to the data center request to reach a computing server (downlink) or the time it takes for a computing result to leave the data center network (uplink) and be on the way to the end user. UDCL is added on top of the task execution time for every processed user request. Network topologies hosting computing servers closer to the data center gateway have smaller UDCL and can provide faster response times.

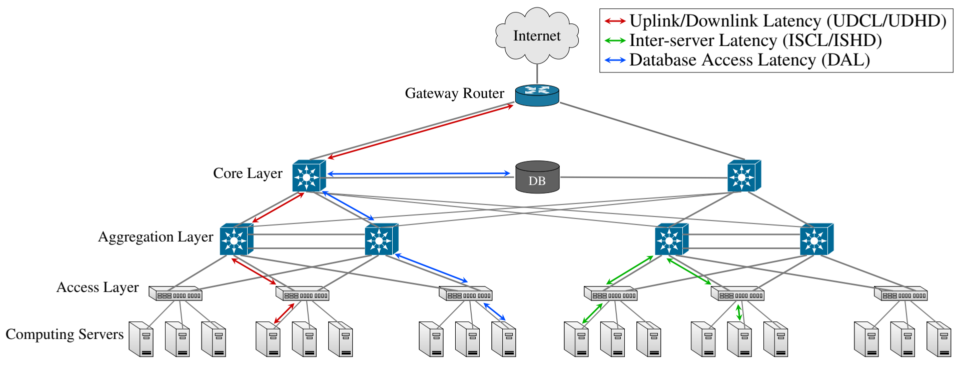
Another important metric is Inter-Server Communication Latency (ISCL), or Inter-Server Hop Distance (ISHD) if expressed in the number of hops. These metrics measure the time (in seconds), or the number of hops, it takes for one task to communicate with another task executed on a different server:

$$\text{ISHD} = \frac{1}{N(N-1)} \sum_{i=1}^{N} \sum_{\substack{j=1 \\ j \neq i}}^{N} h_{ij}, \tag{6.5}$$

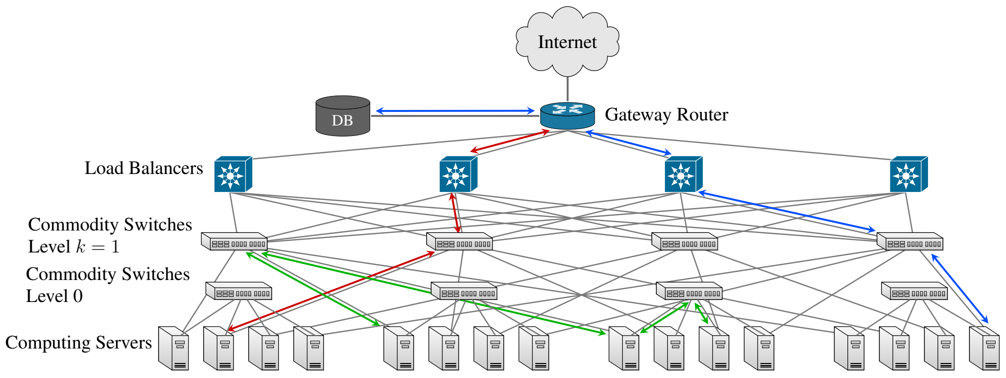
where N is the total number of servers, and h_{ij} is the number of hops between the servers i and j.

ISCL and ISHD are particularly relevant for cloud applications whose execution can be parallelized. Their tasks will need to exchange data and will perform faster in network architectures with fewer hops between servers and smaller inter-server delays. However, inter-server delays will make no difference for standalone applications whose execution is confined to single server.

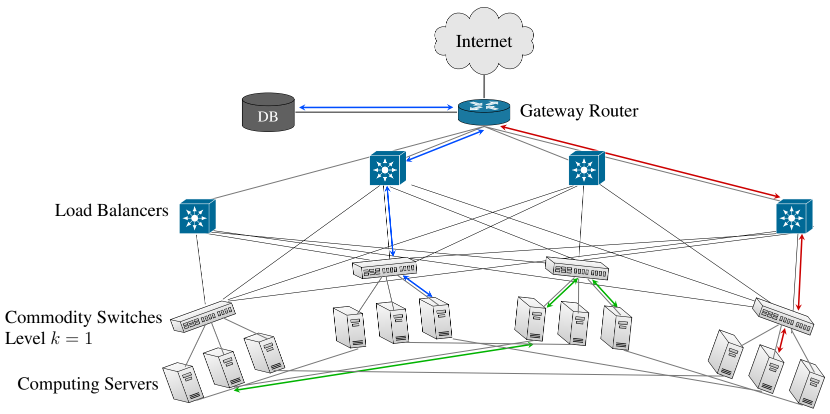
In addition to measuring average values, it is important to analyze deviation in the distribution of inter-server delays. Small deviation values will characterize data center networks with small distances between computing servers (e.g., switch-centric architectures, such as Al-Fares et al. proposal [69], PortLand [70] and VL2 [71]), and allow placement of interdependent tasks at any server, not depending on its location. However, for data centers with highly variable inter-server delays, such as server-centric architectures like BCube [73] and DCell [74], it becomes highly beneficial to consolidate heavily communicating tasks to reduce network delays and improve performance.


Internet
Uplink/Downlink Latency (UDCL/UDHD)
Inter-server Latency (ISCL/ISHD)
Database Access Latency (DAL)
Gateway Router
Core Layer
DB
Aggregation Layer
Access Layer
Computing Servers

(a) Three-tier architecture



(b) BCube architecture



(c) DCell architecture

Figure 6.3: Communication latency in different data center architectures

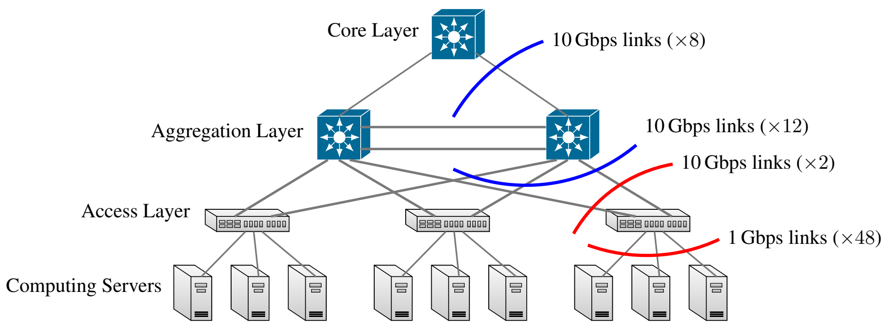


Figure 6.4: Bandwidth oversubscription in three-tier topology

The third delay-related metric is the Database Access Latency (DAL). DAL is defined as an average Round-Trip Time (RTT) measured between computing servers and the data center database. DAL is measured in seconds. An overwhelming majority of cloud applications store and obtain data from database [61]. Thus, reducing the time required for sending a query and receiving data can significantly speed up performance. As an alternative to bringing databases physically closer, a number of data replication techniques can be employed [121]. Data replication reduces DAL for the cached data, but can also introduce traffic overhead for propagating replica updates in the system. Figure 6.3 illustrates the aforementioned delays in the three-tier, BCube and DCell data center architectures.

Bandwidth Oversubscription Ratio

Bandwidth oversubscription can be defined as the ratio between the aggregate ingress and aggregate egress bandwidth of a network switch[122]. For example, in a typical three-tier topology (see Fig. 6.4), Top-of-Rack (ToR) switches are equipped with two 10 Gb/s links to the aggregation network and can support up to 48 servers in the access network, each connected with a 1 Gb/s link. This entails a Bandwidth Oversubscription Ratio (BOR) of $(48\,\text{Gb/s})/(20\,\text{Gb/s}) = 2.4:1$, which corresponds to a per-server bandwidth of $(1\,\text{Gb/s})/2.4 = 416\,\text{Mb/s}$ under full load. Further bandwidth aggregation of 1.5:1 occurs at the aggregation level, where each switch has eight 10 Gb/s links to the core network and twelve 10 Gb/s links to the access network. As a result, the per-server available bandwidth can be as low as $(416\,\text{Mb/s})/1.5 = 277\,\text{Mb/s}$ in a fully loaded topology [122]. Server-centric architectures do not introduce points of bandwidth oversubscription; as a result, BOR is equal to 1.

Computing BOR is important to estimate the minimum non-blocking bandwidth available to every server. When the computing servers produce more traffic that the available bandwidth, ToR and aggregation switches can become congested and start to drop packets from the overflowed buffers, significantly degrading performance of cloud applications.

Network Losses

The packets travelling in a data center network may be lost and fail to reach their destination due to link errors. These losses may cause significant communication delays, as retransmissions are performed only at the transport layer using TCP protocol. For this, it is important to analyze end-to-end error rates at bit and packet levels to assure network performance and the desired level of QoS.

In data centers, interconnection links are not identical. For example, a typical fat-tree three-tier architecture (see Fig. 6.3a) contains optical 10 Gb/s links with per-link Bit Error Rate (BER) in the range of 10^{-12} to 10^{-18} in the core and access layers. While in the access layer a less expensive twisted pair gigabit Ethernet technology is used with BERs as high as 10^{-10}. Knowing the topology and characteristics of the network links, it becomes possible to calculate average end-to-end error rates depending on the communication paths involved, e.g., servers-to-gateway or servers-to-servers.

In this paper we define two metrics for error rate estimation. The first is the Uplink/Downlink Error Rate (UDER). UDER measures average BER on the paths between data center gateway and computing servers and is defined as follows:

$$\mathrm{UDER} = \frac{1}{N} \cdot \sum_{n=1}^{N} \sum_{l=1}^{L} \mathrm{BER}_{nl}, \tag{6.6}$$

where N is the number of computing servers, L is the number of hierarchical layers in network topology and BER_{nl} is the BER of the layer l link interconnecting server n with the data center gateway.

The Inter-Server Error Rate (ISER), instead, evaluates the average error rate of inter-server communications:

$$\mathrm{ISER} = \frac{1}{N(N-1)} \cdot \sum_{i=1}^{N} \sum_{\substack{j=1 \\ j \neq i}}^{N} \mathrm{BER}_{ij}, \tag{6.7}$$

where N is the number of computing servers and BER_{ij} is the BER of the path interconnecting server i and server j. The latter is calculated as a sum of BERs of all links between servers i and j.

Measuring error rates is important. It allows diversifying resource allocation strategies that take into account sensitivity of cloud applications to transmission errors and helps detecting hardware faults.

Average Link Utilization Ratio

Average Link Utilization Ratio (ALUR) shows average traffic load on data center communication links and can be defined as follows:

$$\mathrm{ALUR} = \frac{1}{N_i} \cdot \sum_{n=1}^{N_i} u_n, \tag{6.8}$$

where u_n is the utilization ratio of link n and N_i is the number of links of type i. ALUR is an aggregate network metric and is designed to improve analysis of traffic distribution and load levels in different parts of the data center network. It helps to define proper traffic management policies, can be used to detect network hotspots and becomes an essential tool for preventing performance degradation of cloud applications due to network congestion.

For a three-tier fat tree topology ALUR can be measured separately for the access, aggregation and core segments of the network. High congestion in any of these segments will signal the need to increase the capacity of network links and switches or even reconsider bandwidth oversubscription ratios between these segments. For BCube and DCell topologies, ALUR can be measured over server-to-server and server-to-switch segments of the network.

Average Server Degree Connectivity

Depending on the design strategy, data center topologies are either switch-centric or server-centric. In switch-centric architectures, such as fat-tree, each server is usually connected to a single ToR switch with only one link. In server-centric architectures, instead, the computing servers are connected to several commodity switches (BCube) and/or a number of other servers (DCell) to increase network capacity and provide resilience to node and switch failures.

A higher degree of connectivity increases network capacity, makes the whole topology fault tolerant and helps to balance the load. However, having a high number of connections increases network power consumption as more links and NICs have to be deployed and utilized. To analyze how well the computing servers are connected, Average Server Degree Connectivity (ASDC) can be computed:

$$\mathrm{ASDC} = \frac{1}{N} \cdot \sum_{n=1}^{N} c_n, \tag{6.9}$$

where N is a total number of data center servers and a number of c_n network links connects server n to other devices, switches and/or servers.

6.2.3 Network Traffic-Related Metrics

Knowing the properties of network traffic is a key to understanding efficiency of data center communication systems. By the direction of signaling, network traffic can be classified into internal and external.

Internal traffic remains within the data center network and accounts for almost 75% of all communications in modern data centers [114]. It is mostly composed of storage and database interactions of cloud applications as well as communications between individual tasks executed in parallel. The performance of data center internal communications is subject to database access delays (metric DAL) as well as bandwidth availability (metric BOR) and latency between servers (metrics ISCL/ISHD). Neither bandwidth nor latency on the uplink and downlink paths of the data center network affect the performance of internal communications significantly.

External traffic is that destined to the end users. It includes the traffic produced by cloud applications as well as inter data center traffic [114]. The external traffic is highly sensitive to the available bandwidth (metric BOR) and communication latency in the uplink and downlink (metrics UDCL/UDHD). At the same time, the bandwidth and communication latency between servers (metrics ISCL/ISHD) do not affect the performance of external communications significantly.

The proportion between internal and external data center traffic can be estimated as follows.

- Internal Traffic Ratio (ITR) is the ratio of the traffic that remains inside the data center to the total data center traffic:

- Internal Traffic Ratio (ITR) is the ratio of the traffic that remains inside the data center to the total data center traffic:

$$\mathrm{ITR} = \frac{\text{Internal Traffic}}{\text{Total Data Center Traffic}}. \tag{6.10}$$

- External Traffic Ratio (ETR) is the fraction of traffic that leaves the data center network:

$$\mathrm{ETR} = 1 - \mathrm{ITR} = \frac{\text{External Traffic}}{\text{Total Data Center Traffic}}. \tag{6.11}$$

In addition to categorizing network traffic according to its destination, it is important to distinguish user- or application-related messaging from the rest of the traffic which includes network management and monitoring. The latter is required to operate communication networks. Management operations include transmissions for address resolution (e.g., ARP) and routing (e.g., OSPF). Control messaging and problem detection (e.g., ICMP) can also be attributed to management operations, while SNMP traffic is

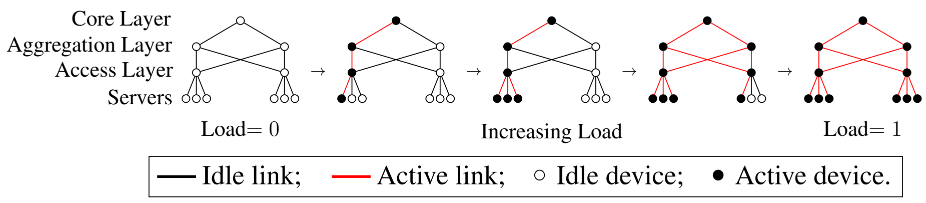


Figure 6.5: Powering up equipment as data center load increases

related to monitoring operations. The Management and Monitoring Traffic Ratio (MMTR) helps to unveil traffic overhead for network management and can be computed as follows:

$$\text{MMTR} = \frac{\text{Management and Monitoring Traffic}}{\text{Total Data Center Traffic}}. \tag{6.12}$$

To obtain the energy spent on network management and not for transporting application-related traffic, we can use the CNEE metric (showing communication network energy efficiency) and compute Management and Monitoring Traffic Energy (MMTE) as follows:

$$\text{MMTE} = \text{CNEE} \cdot \text{Management and Monitoring Traffic}. \tag{6.13}$$

MMTE is measured in Joules and shows the amount of energy consumed by the communication equipment to keep the network operational. In an ideal case MMTE should assume values close to zero, when most of the consumed energy is attributed to application-related traffic delivered at the full effective network capacity.

Understanding data center traffic is very important. Network traffic analysis at the micro- and macroscopic levels can help in estimating the impact on network processes [123], design traffic engineering solutions [124], capture interdependencies between executed workloads [125], and optimize communication between several geographically distributed data centers [126].

6.3 Evaluation

This section presents evaluation and numerical comparison of the proposed metrics in categories of power, performance, and network traffic of data center communication systems.

6.3.1 Evaluation Scenario

Several data center architectures have been proposed in the literature [56], [58]. For evaluation purposes, we consider the following four architectures: fat-tree three-tier [69]–[71], BCube [73], DCell [74] and Optically cross-braced Hypercube [127] (OH). For a fair comparison, all the architectures are configured to host 4096 computing servers.

In the fat-tree three-tier topology, these servers are arranged into 128 racks and served by 8 core and 16 aggregation switches. The core and aggregation switches as well as the aggregation and access switches are interconnected using 10 Gb/s, 0.24 μs optical links. The links connecting computing servers and access network ToR switches are 1 Gb/s, 0.01 μs twisted pair links.

In the BCube and DCell topologies, the 4096 computing servers are arranged in groups of $n = 8$. This entails a BCube architecture of level $k = 4$ with 3 layers of commodity switches per group of servers and a level $k = 2$ DCell. 1 Gb/s links are used to interconnect computing servers with the commodity switches. In the lowest layer these links are 2 meters long, while in the upper layers they are 10 and 50 meters long, respectively. The gateway router is connected to the data center network through a number of load balancers using 50 m long, 40 Gb/s optical fibers.

In OH, 12 hypercube dimensions are needed to support 4096 servers. This requires $12 \cdot 2^{12}/4 = 12228$ 2-by-2 optical switches for interconnection.

In all architectures, optical fibers are assumed to support single-mode light propagation for a 1550 nm operating wavelength.

6.3.2 Evaluation of Power-Related Metrics

In this section we evaluate power-related metrics, including CNEE, NPUE and EPC, for different data center architectures.

Evaluation of Network Energy and Power Usage Effectiveness

To obtain CNEE and NPUE it is necessary to calculate the power consumption of the computing servers and network equipment as the load of the data center increases. This increase can be non-linear as waking up new servers in already operational racks does not require waking up additional network switches. However, starting up a new rack would require powering on the top-of-rack switch and possibly aggregation and core switches. Figure 6.5 illustrates this concept using a three-tier topology.

To estimate the power consumption of a single server we selected the most widely used hardware models from different vendors, Dell PowerEdge R720 [128], Huawei Tecal RH2288H V2 [129], and IBM System x3500 M4 [130], and computed their average peak and idle power consumptions. Assuming the servers implement Dynamic Voltage and Frequency Scaling (DVFS), their power consumption $P(l)$ can be estimated as follows [131]:

$$P(l) = P_{\text{idle}} + \frac{P_{\text{peak}} - P_{\text{idle}}}{2} \cdot (1 + l - \mathrm{e}^{-(\frac{l}{\tau})}), \qquad (6.14)$$

where l is the load of the server and τ is the utilization level at which servers attain asymptotic power consumption, which is typically in the range $[0.5, 0.8]$.

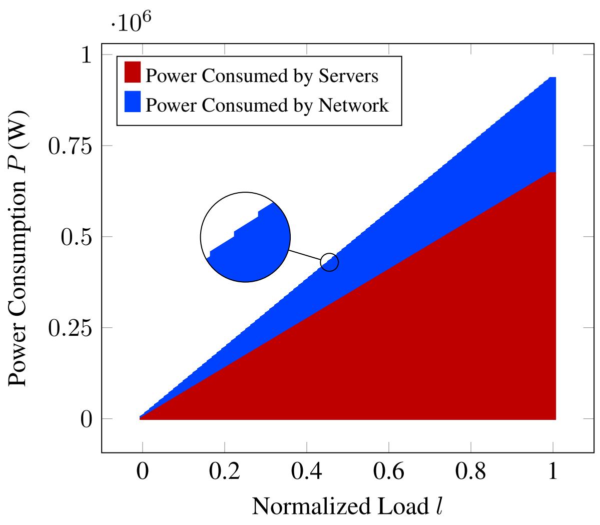


Figure 6.6: IT power consumption in fat tree three-tier data center

For network equipment we considered HP FlexFabric 11908 [132] to be used in the aggregation and core layers of the fat tree three-tier architecture, and HP 5900 AF [133] for the ToR and the commodity switches in BCube and DCell architectures. Finally, PRISMA II optical switches are considered for OH architecture [134].

Figure 6.6 shows normalized power consumption of the data center IT equipment for a fat-tree three-tier architecture. The power consumed by the servers excludes network interface card power consumption, which is included in network power consumption. The leaps highlighted in the zoomed part correspond to a server wake up in a previously idle rack. It causes a wake up of the access, aggregation and core layer switches and leads to non-proportionality in network power consumption.

The CNEE computed for all four data center architectures considered is reported in the first row of Table 6.2. The CNEE is the highest for the fat-tree three-tier topology, which is mainly caused by high bandwidth oversubscriptions performed at several layers. As a result, the energy is spent to support higher bitrates, but they cannot be fully utilized by the servers. In contrast, the throughput in BCube and DCell architectures can achieve 100 % of the network capacity. CNEE, besides being sensitive to bandwidth oversubscription, also depends on the overall network power consumption. This is the reason why CNEE is higher for BCube than for DCell. BCube hosts a large number of commodity switches $(k+1) \cdot n^k$ (2048), while DCell has only one commodity switch per group of n servers (512). OH architecture hosts 12228 2-by-2 optical switches whose power consumption is significantly lower than commodity switches used for BCube and DCell. As a result, the CNEE value computed for OH topology is more similar to the DCell value rather than that

Table 6.2: Evaluation of Power-related Metrics

METRICS	ARCHITECTURES			
	Three-tier	BCube	DCell	OH
CNEE	0.203 μJ/bit	0.109 μJ/bit	0.027 μJ/bit	0.033 μJ/bit
NPUE	3.58	2.50	6.86	5.99

of BCube.

Having evaluated energy spent to deliver a single bit of information, it is possible to assess the overall power effectiveness of data center networks with NPUE. With the lowest NPUE, BCube appears to be the most power-hungry topology. As already mentioned, it is due to the fact that BCube hosts a high number of switches. In addition to the number of network devices, their power efficiency plays an important role in NPUE. For example, DCell has a higher number of switches when compared with the three-tier topology. However, these are commodity switches whose power consumption is several magnitudes lower that the consumption of core and aggregation level switches. Despite low power consumption of individual optical switches, OH architecture has lower NPUE than DCell. In OH, transceivers and the high number of active ports per servers are the two main components contributing to network power consumption.

Evaluation of Energy Proportionality

Figure 6.7 shows normalized power consumption along with the computed EPC values for several network switches with different profiles. The dashed line represents an ideal case with EPC equal to 1.

Switch 1 shows a curvilinear behavior. For intermediate loads in the range $(0.2, 0.8)$, the power consumption increases at a smaller rate than the workload, while for the low (< 0.2) and high (> 0.8) load levels it increases more rapidly than the incoming workload. As a result, the obtained EPC is equal to 0.69. With EPC equal to 0.2, switch 2 shows a realistic energy consumption profile with a large idle part and a stepwise power consumption attributed to communication ports. This is very close to the case represented by Switch 3. Being completely insensitive to the workload, EPC value of Switch 3 is equal to 0. Switch 4 has negative EPC equal to -0.89. It signals that the device starts consuming less energy as the workload increases.

6.3.3 Evaluation of Performance-Related Metrics

This subsection presents evaluation results of the proposed metrics for network latency (UDCL, UDHD, ISCL, ISHD, DAL), network losses (UDER, ISER) and connectivity (ASDC) with the exception of BOR and ALUR. Server-centric

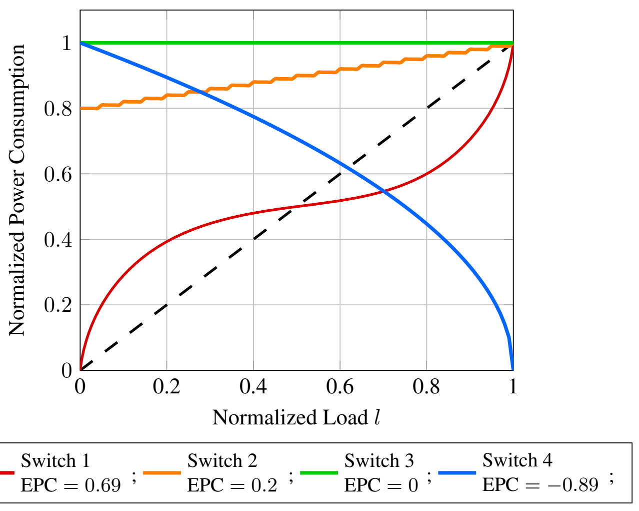


Figure 6.7: Power consumption profiles of different network switches

architectures typically do not introduce points of bandwidth multiplexing and oversubscription, which makes their BOR metric to be equal to 1. Computing ALUR metric requires having per-link traffic statistics, which can be obtained either from detailed traces or, more realistically, directly measured in real data centers during runtime.

Network Latency, Network Losses and Server Degree Connectivity

To evaluate UDCL, ISCL, DAL, UDER and ISER we considered transmission of two test packets of 40 Bytes and 1500 Bytes, corresponding to a TCP acknowledgement and a maximum Ethernet transmission unit respectively. A one-way transmission delay is measured for UDCL and ISCL, and a round-trip delay for DAL. For signal losses, a BER of 10^{-12} is considered for copper cables and 10^{-14} for optical fibers. As no other traffic is present in the data center network, Ethernet inter-frame gap and thus queuing delays can be neglected.

The network delay of a single packet is composed of the transmission delay D_t and link propagation delay D_p. D_t is expressed as a ratio between packet size S and link rate R, while D_p is defined as the link length L over the signal propagation speed P:

$$D_t = \frac{S}{R}, \qquad D_p = \frac{L}{P}. \tag{6.15}$$

P defines the physical characteristic of the medium. In copper it is two thirds of the light speed c, while in optical fiber the speed of light is scaled with the refractive index, taken to be equal to 1.468 for glass fiber [135].

Table 6.3: Evaluation of Network Latency, Network Losses and Server Degree Connectivity.

METRICS		ARCHITECTURES			
		Three-tier	BCube	DCell	OH
40 B	UDCL	1.45 µs	1.38 µs	1.19 µs	1.16 µs
40 B	ISCL	1.98 µs	3.93 µs	4.73 µs	1.2 µs
1500 B	UDCL	15.7 µs	14.47 µs	15.50 µs	14.42 µs
1500 B	ISCL	28.34 µs	73.72 µs	93.92 µs	24.47 µs
	DAL	18.11 µs	17.15 µs	17.15 µs	15.71 µs
	UDHD	4	3	3	3
	ISHD	5.78	7.00	8.94	3.25
	UDER	$1.03 \cdot 10^{-12}$	$1.02 \cdot 10^{-12}$	$1.02 \cdot 10^{-12}$	$1.02 \cdot 10^{-12}$
	ISER	$1.77 \cdot 10^{-12}$	$4.21 \cdot 10^{-12}$	$5.34 \cdot 10^{-12}$	$2.00 \cdot 10^{-14}$
	ASDC	1	4	2.79	12

Table 6.3 presents the results for network latency, losses and connectivity related metrics. The results show that the OH architecture can provide better support to internal communications with ISCL, ISHD and ISER all being lower in comparison to the other architectures. The result is expected as OH is the architecture with the highest ASDC value, which guarantees having short paths even between distant servers. With respect to BCube and DCell, the three-tier topology supports internal communications better. This might be surprising as the three-tier connectivity degree measured with ASDC is the lowest among all architectures. However, both BCube and DCell, while being much better interconnected, need to traverse a large number of hops to communicate between distant servers.

The error rate between servers, measured by ISER, is the highest for BCube and DCell due to their heavy reliance on copper links. The error rate between servers and the gateway, measured with UDER, on the contrary, is lower in BCube and DCell as packets sent by the servers traverse fewer number of hops to reach the gateway.

6.3.4 Evaluation of Network Traffic-Related Metrics

To evaluate network traffic related metrics MMTR and MMTE we used packet traces collected in real data centers, UNIV1 and UNIV2 [136]. Along with the user application data these traces also include ARP, ICMP and OSPF flows. Both data centers follow a two-tier architecture design. The traces contain one and a half hours of traffic for UNIV1 and two and a half hours for UNIV2

Table 6.4: Evaluation of Management and Monitoring Traffic Energy

MMTE	Architectures			
	Three-tier	BCube	DCell	OH
UNIV1	169.19 J	90.62 J	22.23 J	27.31 J
UNIV2	30.98 J	16.59 J	4.09 J	5.00 J

data centers.

To analyze the fraction of network management and monitoring traffic we computed MMTR, which is 0.79% for UNIV1 and 0.025% for UNIV2 data centers. The results show that although UNVI1 has a smaller number of network devices, its network is managed less efficiently.

Table 6.4 shows energy consumed by the data center network to process and deliver management and monitoring traffic. The MMTE metric is computed taking into account data center topologies presented in Section 6.3.1. As expected, the energy consumption of UNIV2 management and monitoring traffic is lower than in UNIV1 for all the architectures. DCell always outperforms other architectures as it spends the lowest energy to transfer a single bit of information (see CNEE values reported in Table 6.2), while the fat-tree three-tier architecture is the most energy consuming topology.

The choice of the employed resource allocation strategy would certainly impact most of the presented metrics. Workload (or VM) migration would increase the radio of monitoring and management traffic in MMTR and MMTE metrics, increase a portion of the internal traffic in ITR and ETR metrics, and even change average link utilization ratio (ALUR). This again confirms that a set of presented metrics could become an essential tool in developing and refining resource allocation in cloud computing data centers and can lead to novel network-aware scheduling solutions [76], [78].

6.4 Summary and Perspectives

Energy efficiency and infrastructure monitoring are two of the main parameters for successful data center operation. The proposed framework of metrics is positioned to become an essential tool for monitoring, comparing and assessing performance of data center communication systems.

The power-related metrics (see Section 6.2.1), such as NPUE, assess with a fine granularity energy efficiency of the network and allow data center operators to optimize their investments in networking equipment and interconnects. The performance-related metrics (see Section 6.2.2), such as ALUR, enable detailed monitoring and assessment of network throughput, delay and error rate performance. They are especially relevant for the largest class of SaaS cloud applications which often communicate intensively with the end users and also internally. The analysis of these metrics helps to ensure and guarantee QoS and SLA to the customers. Finally, network traffic-related metrics (see Section 6.2.3) permit the development of proper traffic management and infrastructure-aware resource allocation policies. The proposed framework of metrics for networks of cloud computing data centers is essential for optimization of operation and to plan capacity extensions of existing facilities as well as the design of future data centers.

In addition, the proposed metrics are easy-to-integrate metrics into existing data center monitoring systems, such as VMware vCenter Log Insight [137] or Cisco Prime Data Center Network Manager [138]. Most data center monitoring systems already provide information that is required for computing these metrics, including runtime power consumption, link utilization levels or error rates. For example, simple analysis of destination addresses can help to differentiate between the internal and outgoing data center traffic. Data center monitoring software maintains statistics for each server, for example the status of the links. Consequently, a simple query on the average number of active links for each server will allow the computation of the ASDC metric. The availability of up-to-date link- and traffic-related statistical information enables the design of network-aware resource allocation and scheduling solutions (see Section 5.1.2).

Table 6.5 provides a top-level comparison of the evaluated data center architectures. For the purpose of simplicity, the values are reported as high (H), medium (M) and low (L), while the precise measurement values and evaluation details are reported in Section 6.3. High bandwidth oversubscription of the three-tier architecture prevents computing servers from exploiting full available network capacity and, as a consequence, leads to the highest energy-per-bit consumption. DCell appears as the most "green" architecture with the lowest energy-per-bit ratio and high power usage effectiveness. BCube is less effective in terms of the power usage effectives because it hosts the highest number of switches. The analysis of communication latency shows that hierarchical architectures, such as three-tier fat tree, favor internal server-to-server communications, while distributed data center architectures, including BCube and DCell have shorter paths for the traffic directed out of the data center. On the other hand, server-centric architectures, such as OH, can reduce the number of hops between distant servers significantly. As a consequence, they provide better support to internal communications than hierarchical architectures.

Table 6.5: Performance comparison of evaluated architectures. Values are categorized as (H) High, (M) Medium and (L) Low.

Architectures	Metrics										
	CNEE	NPUE	UDCL	UDHD	ISCL	ISHD	DAL	UDER	ISER	ASDC	MMTE
Three-tier	H	M	M	M	L	M	H	L	H	L	H
BCube	M	L	M	L	H	H	M	L	H	M	M
DCell	L	H	M	L	H	H	M	L	H	M	L
OH	L	M	M	L	L	L	L	L	L	H	L

Part III

Energy Efficiency in Mobile Cloud/Fog Computing

Chapter 7

Background on Mobile Cloud/Fog Computing

The chapter presents in more details Mobile Cloud Computing and Fog Computing paradigms, with a particular focus on the concepts of offloading and application partitioning. Then, it overviews potential research challenges and illustrates the proposed contributions in the field.

7.1 Analysis of Mobile Cloud and Fog Computing

Mobile devices have become essential for our everyday activities such as business, health-care, social activities and entertainment [18], [139]. According to Gartner, the worldwide smartphone sales reached 1.4 billion units [140] in 2015, while the sales of wearable devices recorded 232 million units in 2015 and are projected to reach 322 million units in 2017 [141]. Smart watches, glasses, rings, gloves and helmets are the most popular wearable devices currently available on the market, which will worth $ 30.2 billion by 2018 [142]. Although being constrained by the battery, modern mobile devices have computing, communication and storage resources. However, these resources are limited. The role of mobile distributed computing paradigms is to overcome limitations such as computational capability, battery power, connectivity, opportunity to gather more sensing data, access to different content and to make use of idling processing power [2]. These are the motivations leading to the development of all mobile distributed computing paradigms described in Section 2.2. In this chapter, MCC and FC are analyzed in detail.

With MCC and FC, developers can exploit the virtually unlimited resources of the cloud and or the fog in terms of storage, power consumption and computation capabilities for their applications. The key ingredient for the success of MCC and FC is the capability of outsourcing part of the computing tasks from weak mobile devices to the powerful cloud or fog. The technique used for outsourcing is application partitioning. Specifically, the applications partitioned into tasks whose execution can be partially or completely offloaded on the cloud environment in order to narrowing the mobile devices effort. The main goal is to make users mobile devices to performs input/output operations only. MCC and FC also bring other new opportunities such the usage of the mobile devices and the cloud to collecting data, perform distributed sensing and crowdsourcing.

With respect to MCC, FC is still in its infancy. Fogs were first proposed by Cisco to specifically support IoT unique characteristics like geo-distribution, location awareness and low latency in addition to mobility, which is also a requirement in MCC [20], [21]. Recently, Sarkar et al. [22] have studied the suitability of FC to IoT providing a comparison study with traditional cloud paradigm. The study focuses on the capability of fogs and clouds in serving latency-sensitive applications while in another work the authors model fog computing architecture to be energy-efficient and guarantee low latency. Resource management in FC is a concern [143]–[145]. Aazam et al. [143] group IoT devices into three categories: static, small and large mobile devices. The fog provides resources according to the requirements of each category. Static devices do not need support for mobility, hence they require a lower amount of resources than mobile devices. Pricing is dynamic and it is based on the amount of resources each category consumes as well as the type of customers and service asked to the fog. According to Deng et al. [144], fog computing-based resource allocation policies should consider the tradeoff power consumption vs. communication delay of the mobile devices. Ningning et al. [145] aim at providing load balancing in fog environments considering that nodes can join and leave the fog. Security is a fundamental aspect in any distributed computing systems, including FC. However, only few works have analyzed this aspect [146], [147]. Stojmenovic et al. [146] discuss security and privacy issues and present a case study with analysis of countermeasures to the man-in-the-middle attack. Wang et al. [147] study and compare security problems in fogs and clouds digital forensics, i.e., the application of science to the identify, collect, and analyze data while preserving its integrity during examination. Practical application of fog computing paradigm is mainly in the healthcare domain [148], [149].

7.2 Offloading and Application partitioning

The need to augment the performance of the devices and the application, including a better usage of the battery drives adoption of mobile distributed computing paradigm and the key pillar feature enabling mobile/edge fog computing is offloading. In the literature, the concepts of offloading and application partitioning are often confused and used interchangeably. However, it exists a subtle difference. The term “offloading” is general and used to describe the process of outsourcing part of the computation in mobile cloud

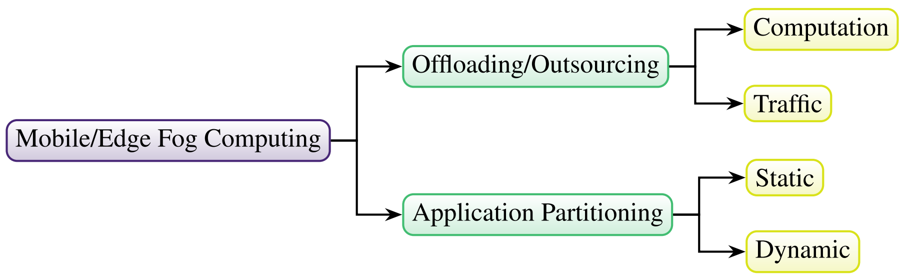


Figure 7.1: Taxonomy of Key Features in Mobile/Edge/Fog Computing

computing where only two entities are usually involved, namely the mobile device and the cloud. Application partitioning describes *the methodologies* to perform offloading. Moreover, as in fog computing the entities involved are more than two, the mobile or IoT device initiating the process, the mobile or IoT devices in the vicinity and the cloud, the term application partitioning is more suitable. Fig. 7.1 illustrates the salient aspects of both offloading and application partitioning.

This section first provides a general definition and a deep analysis of offloading (Section 7.2.1) and finally overviews the techniques for application partitioning and existing modeling tools (Section 7.2.2).

7.2.1 Offloading

Offloading is the process of *outsourcing* functionalities from one device to another. This definition is general enough because in the literature two type of functionalities that can be offloaded: traffic and computation. Although in the following sections both aspects will be covered, this thesis will analyze in the next chapters contributions in the area of computation offloading only. A general taxonomy on offloading is presented in [150].

Traffic Offloading

Traffic offloading is the process of redirecting traffic delivered with cellular network with other types of communication technologies.

Mobile cloud applications is one of the fastest growing markets. Currently, more than 7 billion people use mobile devices connected to Internet. Mobile gadgets and smartphones are already essential in our daily activities as they help doing business, communicating and entertaining [18], [139]. In 2015, smart devices accounted for only 36% of the total number of Internet connections originated by mobile devices, but they generated more than 89% of the total mobile data traffic [40]. Currently, mobile data traffic is growing at an unprecedented rate and is projected to generate up to 30 EB per month by 2018 [40], which overloads current cellular networks.

A number of promising solutions has been recently proposed to address growing traffic demands. With the objective of "bringing network closer to the user", a mix of macro, micro, pico, femto and relay base stations has been proposed in LTE-A [151]. The reduction of the cell size helps to increase network capacity and coverage, but comes with additional costs of installing and maintaining the new base station [152]. Traffic offloading to other networks can help to avoid having additional cellular network equipment. The offloading is usually performed to WiFi [153]–[155] or to the opportunistic networks [156]–[158]. WiFi operates on unlicensed frequencies and, unlike LTE, which uplink and downlink data rates remain constant to every user, WiFi channels are shared between the served users. WiFi infrastructure is not expensive and already widespread in many areas. Opportunistic networks do not require any infrastructure, but they operate relying on intermittent contacts of in proximity users.

From the user perspective, with traffic offloading applications can directly contact the server in the cloud without the need to pass by the mobile core network. WiFi and opportunistic communications are more energy efficient than LTE or 3G [159], [160] and commonly available "free of charge". As a result, users do not consume data from the monthly traffic plan they subscribed and for which they have to pay the operators. On the other hand, traffic offloading latency can be an issue affecting performance [161] as both WiFi and opportunistic communications become unreliable with high levels of user mobility. Network performance can change quickly and degrade user experience making it worse than in the case without offloading. In addition, most of the operating systems for smartphones, including Android and iOS, already make preference to WiFi over cellular connectivity for data transmission. However, keeping both interfaces constantly active excessively drains battery power. For these reasons, traffic offloading decisions should be taken carefully and take into account performance and mobility in the offloaded networks.

Computation Offloading

Khan et al. [15] identify four main objectives for computation offloading: i) to enhance and augment performance of the applications, ii) to reduce battery consumption of the devices, iii) to allows execution of applications in constrained environments, and iv), pursuing multiple objectives at the same time. Objectives i), ii) and iv) are the most widely investigated in the literature [2], [15], [16]. Despite several offloading techniques have been proposed so far, it does not exist a technique outperforming the others in terms of performance [17].

Developers typically employ offloading to improve performances of weak mobile devices by considering carefully computing components to offload and/or communication technology used during data transfer. Tout et al. [162]

exploit offloading to augment performance of mobile devices working in multi-persona fashion. It has been proved that a worker carrying multiple devices, for example to separate accounts of different workplaces, drains its productivity. Thus, multi-persona systems have been developed with the objective of consolidating all different profiles in a single terminal providing to the user the capability to clearly distinguish between each profiles and the contexts in which it is used. Mahmoodi et al. [163] specifically analyze the impact of having multiple communication technologies at disposal for efficient offloading. In [164] Wolski et al. study an offloading decision mechanism driven by the availability of bandwidth in the network. Offloading decisions take into account the risk costs as well as execution time while performing task execution locally or remotely. A hybrid solution, which bases offloading decisions not only on bandwidth availability, but also on computing effort is discussed in [165]. Performance of the offloading strategy are evaluated for data transfer with WiFi, cellular technology and a combination of the two. Deng et al. [166] propose a genetic algorithm to optimize offloading decisions. The algorithm is proposed for mobile service workflows with inter-dependency among the components and takes into account node mobility.

Nodari et al. [167] use a machine learning approach to preserve battery life in tracking user position. The algorithm predicts future movements of the users and it runs in the cloud, while the mobile device communicates with the server only for updating the position. In this way, the sensors used to collect position data are involved with less frequently. This mechanism improves considerable battery lifetime of the mobile device. In [168] the authors develop a speech recognition library as a case of study for comparing several dynamic offloading strategies with both WiFi and 3G communication technologies. Miettinen et al. in [169] take into account the trade off between local computation and communication costs for saving energy offloading computing tasks. The authors perform real energy measurements on different smartphones running different applications, such as compression algorithms, video encoding. Similarly, also Segata et al. [170] study the trade-off between energy consumption for data transmission to the cloud and the energy cost of local computation. The authors focus on the communication part with an analysis of the cellular technologies 2G and 3G, and of WiFi. The results are obtained with real measurement. The analysis performed by Altamimi et al. [171] is similar to [170], but focus on energy models for WLAN, 3G and 4G technologies and validated experimentally the theoretical models proposed to assess the energy cost of offloading. While assessing energy-efficiency in performing offloading, it is also necessary to quantify costs that are not directly related to offloading itself. In [172] propose a job-scheduling offloading mechanisms and a methodology to measure the efficiency of offloading considering both direct and indirect costs.

The following paragraphs overview the most popular techniques for computation offloading.

MAUI: [173], developed by researches at Microsoft, is a system where developers have to modify the application code to identify the methods to be offloaded. The acronym MAUI stands for Mobile Assistance Using Infrastructure. The objective is to minimize energy consumed for computing purposes. A profiler collects information on energy and data transfer requirements of the application. Offloading decisions are taken at client side on the basis of the information collected. This framework needs to be set up during development phase and it requires to recompile the application in case of any change of the offloading scenarios. MAUI has some limitations. First, it operates only for applications developed in .NET. Second, offloading decisions do not consider the energy and time spent for communications with the cloud.

CloneCloud: [174], developed by researches at Intel Labs Berkeley, is system that augments performance of smartphones through dynamic offloading execution of applications to the cloud. Offloading is seamless, i.e., does not require any conversion mechanisms to port applications from the device to the cloud. The entire application stack is fully cloned, thus when the system decides that migration is convenient, it migrates treads at selected points of the application to the cloud. Therefore, proper decisions are essential to guarantee the performance of the application. Wen et al. [175] uses a similar philosophy and create perfect copy of the application in the cloud in order to perform complete execution on the cloud and to limit the mobile device effort to only display the result.

eXCloud: [176] exploits State-On-Demand (SOD) on top of the VM for a lightweight task migration from the mobile devices to the cloud at VM instance level. Offloading occurs when the smartphone resources are not sufficient to execute some tasks or the computing load exceeds certain levels. The migration to the cloud is transparent to the applications and only copies the required data are copied to the cloud with the objective of saving network bandwidth resources. A limitation of this method is that offloading decisions do not consider energy consumption or communication costs at all.

ThinkAir: [177] similarly to MAUI [173] supports offloading at method-level and similarly to CloneCloud [174] in the cloud a copy of the smartphone execution is created. Unlike other methods, ThinkAir supports on-demand resource allocation and parallelization to reduce delays and optimize system performance. Offloading decisions are based on energy consumption of CPU, screen, GPS, WiFi, 3G, and audio interfaces while considering that execution of tasks requiring GPS and audio can not be outsourced. With ThinkAir, developers are asked to mark any method that may require offloading.

Cuckoo: [178] is system that allows developers to easily write and run applications in Android platform empowered with offloading. Applications programmed with the Cuckoo model can offload execution to any Java Virtual

Machine on the cloud. For this reason, developers need to design both client and server side of the application. However, offloading decisions are not taken during runtime and are not based on energy consumption.

7.2.2 Application partitioning

Application partitioning describes the methodologies to divide an application into tasks that are processed and executed by different devices. Although according to [179] applications can be partitioned only in *static* and *dynamic* fashion, other studies further differentiate dynamic techniques into *casual* and *periodic* [17].

Techniques

Static application partitioning: is normally used to divide computational heavy tasks from light ones and let them run in the cloud to save resources. In static partitioning techniques the task division is pre-determined by the developer. This solutions is not very effective. Because of the heterogeneity of a number of factors such as type of devices, network conditions, computing load, battery status, it is almost impossible to find a priori an effective partitioning scheme able to work under any circumstance.

Dynamic application partitioning: divides the tasks of the application during runtime after having analyzed contextual information in order to take decisions. For example, information on network status may involve the computation of available bandwidth and current network latency, information on the availability of the resources may require knowledge about current battery status of the device and CPU load. Decisions to partition the application aim to maximize the benefits, such as reducing execution time or energy consumption. According to [17], dynamic partitioning can be *casual* or *periodic*. The first method refers to techniques that activates offloading mechanisms only when certain conditions are met, for example, the load of the mobile device reaches given thresholds. Periodic partitioning instead checks the conditions periodically.

Modeling Application Partitioning

The problem of application partitioning involves the modeling and analysis of *how* and *where* tasks/methods/fragments are executed. In the literature, Directed Acyclic Graphs (DAGs) are the best tool to represent tasks and the dependencies among the tasks. Given a graph G with a set of nodes $V(G)$ and edges $E(G)$, $V(G)$ denotes the tasks and $E(G)$ the dependencies between the tasks. G has a cycle if a series of out edges starting from the nodes v lead to itself. A DAG has no cycles. DAGs are typically employed for scheduling tasks in data centers [80] or in grids [180]. DAG-based scheduling is an NP-HARD problem, hence it does not exist a solution or algorithm to solve the problem in polynomial time [181]. In cloud applications, communications play a key role. As a result, The modeling of *TreeGlass* tasks dependencies is inspired by previous work Communication-Aware DAG (CA-DAG) [61]. CA-DAG allows fine modeling of all the components of a communication-aware system, namely computing and communication parts. In more details, CA-DAG models computing tasks with squared blocks and communication tasks with rounded blocks.

Ahmad et al. [182] propose to partitioning data-intensive applications in order to minimize movement of data. The model is suitable for applications such as large-scale and extensive simulations where the communication effort is higher than the computation. In such applications, moving data from one node to another introduces latency and reduces throughput. Data-intensive applications are modeled trough DAGs, where each node represents a task and each edge the precedence of tasks and the direction of the data stream. Data-intensive stream applications are also the focus of [183].

Yang et al. [184] propose a *method call tree*. Each node of the tree corresponds to a method and each edge (i, j) indicates that the method i calls j. The tree is constructed in such a way that methods are executed in post-order traversal manner, i.e., to display the current node, first the left and then the right subtrees are traversed by recursively calling the post-order function. Moreover, nested migration is not allowed, i.e., if one module is migrated from the mobile device to the cloud, then all the modules in its sub-tree are migrated as well.

In [185], the authors propose to model applications with the so called *execution dependency trees* (EDTs). EDTs profile with a fine level of detail the cost of executing applications by identifying three different type of relations between tasks or modules. The first type of relation is denoted as *series*, i.e., a given module can not start its execution before the execution of the previous ended. It also exist *probabilistic* relations, which describe conditionals. Then, *parallel* relation describes modules that can be executed simultaneously.

7.3 Open Research Problems

Wearable devices have already become essential for our day life. Smart watches, glasses, rings, gloves and helmets are the most popular wearable devices currently available on the market [186], which is projected to rise up to $30.2 billion by 2018 [142]. According to Juniper Research, companies will spend up to $68.7 million in wearable advertising by 2018 [187]. In genereal, wearable devices are defined as electronic technology to be incorporated in clothing or worn on the body, able to perform different tasks, such the tracking, monitoring of physiological functions and provide biofeedback [188]. Unlike generic IoT devices, wearables have unique features such as mobility and are utilized by humans, hence the management is more interactive and complex. The hardware of mobile devices improved considerably in recent years, but not the batteries, which take decades to double performance [189]. Consequently, energy is a very precious resource for wearables in particular. In addition, 75% of users consider battery lifetime as the main feature they look at while buying mobile devices [190]. Wearable devices are usually paired with a smartphone for several operations such as option setup, synchronization and data visualization. Lumo Lift[1], for example, uses biomechanic motion sensors to track posture and physical activities. Collected data is then delivered to the smartphone via Bluetooth. Consequently, while performing computing offloading from wearable devices, it is of paramount importance to benefit from the resources smartphones provide in addition to the cloud. Then, wearable devices and smartphones form a fog. Efficient computing and communication offloading in fog computing and the interaction between fog and cloud remain unexplored fields so far.

Several models have been proposed to study and model the application execution in cloud and mobile cloud computing. However, none of them appears to be suitable in fog domain. In fog computing, it is essential to capture *the location* where each task is executed and *which* technology is used for data transfer. The plethora or possible locations for task execution and communication technologies calls for a novel DAG-based application partitioning model not only able to differentiate between computing and communication tasks and between series/probabilistic/parallel tasks execution.

7.4 Contributions

Given the challenges presented in Section 7.3, the next chapters illustrate the following contributions:

- A general and theoretic model for computation offloading in mobile cloud/fog computing platforms for wearable devices. Performance of the model are validated with simulations performed with NS-3 with a face recognition application (Chapter 8).

- An Android-based application called *TreeGlass*, which is developed with the objective of assessing the performance of application partitioning in fog computing. *TreeGlass*'s tasks are partitioned between the wearable device (Google Glass), the smartphone and the cloud according to a new proposed partitioning scheme in different scenarios. Under each configuration, the execution time and energy drain are evaluated experimentally (Chapter 9).

[1] `http://www.lumobodytech.com/`

Chapter 8

Computing- and Communication-Aware Offloading: Model Design

Taking into account both computing and communications is essential to devise effective offloading mechanisms. This chapter presents and validates a model for offloading and application partitioning with wearable devices in mobile/edge fog computing.

8.1 Model for Computing and Communication Offloading with Wearable Devices

Fig. 8.1 illustrates a typical mobile cloud computing scenario. Wearable devices are often equipped with wireless WiFi and/or Bluetooth connectivity, but usually have no 3G/LTE interfaces, because of their significantly higher energy consumption [191], [192]. Nevertheless, as 3G/LTE connectivity is expected to be present in future devices, the case when wearable devices exploit cellular connectivity using the user smartphone as relay is included as well. Furthermore, as smartphones can be used for offloading tasks in addition to clouds, it is important to take into account also their energy consumption.

For the aforementioned reasons, the proposed model accounts for all three main architectural components: wearable devices, smartphones, and cloud data centers. Computation offloading can be performed in four different ways. In the first case, labeled in Table 8.1 as "w-l", wearable devices can execute complete tasks locally, which corresponds to no offloading. They use considerable amount of computing resources, drain energy, but can safe on communication with no delays introduced because of offloading. The second possibility is to offload processing from the wearable device to the

Table 8.1: Offloading scenarios. Values are categorized as (H) High, (M) Medium, (L) Low and (-) No cost.

Offloading Device	Scenario	Label	Computing		Communication	
			Energy	Time	Energy	Time
Wearable Device (w)	Local processing (l) at the wearable device	w-l	H	H	-	-
	Offloading to smartphone (s)	w-s	-	-	L	L
	Offloading to cloud (c) via smartphone (s)	w-s-c	-	-	L	H
	Offloading to cloud (c) via access point (ap)	w-ap-c	-	-	L	M
Smartphone (s)	Local processing (l) at the smartphone	s-l	M	M	L	H
	Offloading to cloud (c)	s-c	-	-	H	M

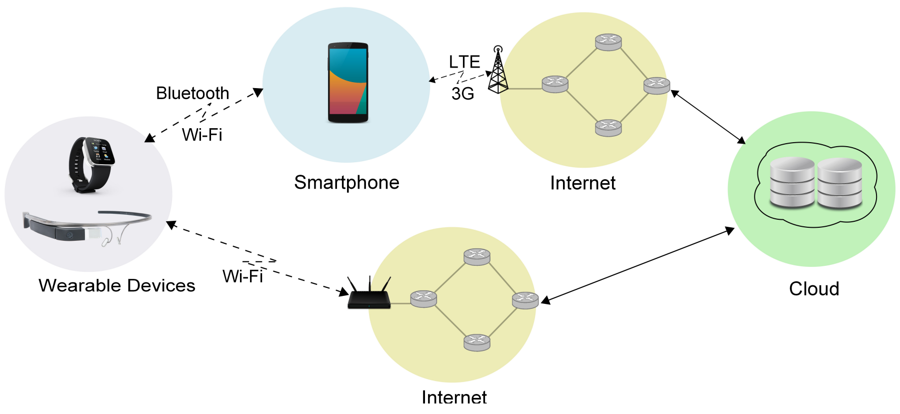


Figure 8.1: Mobile cloud computing scenario for wearable devices

smartphone (w-s). The offloaded task needs to be transmitted from wearable device to the smartphone. The smartphone performs processing and returns the result back, which introduces communication delays, but helps to avoid local processing at the wearable device. The third case, labeled as "w-s-c", allows the smartphone to relay and send the offloaded tasks to the cloud. Processing in the cloud is especially beneficial for computationally intensive tasks, but introduces higher communication delays to account for wide-area network delivery to distant servers. Finally, in the case labeled as "w-ap-c", wearable devices are connected to the cloud through WiFi. It involves only one radio link in the access and the processing is performed in the cloud.

From the smartphones' point of view, two scenarios can be identified. First, when they perform processing locally (s-l), medium amount of energy is required with minimum cost of communications involved, due to the proximity to wearable device and low consumption of the WiFi/Bluetooth interfaces. On the other hand, the time necessary for computing and transmitting back to the wearable device results is high, as smartphones have limited computing power, which results in increased time of processing. Alternatively, the smartphone can offload tasks to the cloud (s-c). In such scenario, no computing is required at the smartphone, but communications require to keep alive both LTE and WiFi/Bluetooth interfaces, which inquires high energy consumption.

8.1.1 Communications in Mobile Cloud Computing

Mobile devices can exploit different technologies for communications. Wearable devices are usually equipped with WiFi, Bluetooth and Near Field Communication (NFC) interfaces or a combination of them. For the scope of this analysis, WiFi technology is considered, in particular the standard IEEE 802.11g as it provides high data rates (up to 54 Mbit/s) and the longest

Table 8.2: WiFi Setup Parameters

SYMBOL	VALUE	DESCRIPTION
ρ_{id}	3.68 W	Power in idle mode
ρ_{tx}	0.37 W	Power during transmission
ρ_{rx}	0.31 W	Power during reception
λ_r	1000 fps	Packet reception rate
λ_g	1000 fps	Packet generation rate
γ_{xr}	$0.09 \cdot 10^{-3}$ J	Processing energy during packet reception
γ_{xg}	$0.11 \cdot 10^{-3}$ J	Processing energy during packet generation

operating range for wearable devices. The most common technology for the communication between the smartphone and the cloud is cellular 3G/LTE. It supports data rates of up to 300 Mbit/s in the downlink and 75 Mbit/s in the uplink.

Equation (8.1) describes the WiFi transmission time T_w of N packets. T_p represents individual packet transmission time. T_{ack} is the time required for acknowledgment. $DIFS$ and $SIFS$ are inter-frames spaces specified by the IEEE 802.11 standard. B is a backoff time, which helps to avoid network contention if multiple nodes need to access the shared channel simultaneously.

$$T_w = DIFS + B + N \cdot (T_p + T_{ack} + SIFS). \tag{8.1}$$

WiFi power consumption P_w is described according to the model proposed by Garcia et al. in [193]:

$$P_w = \rho_{id} + \rho_{tx} \cdot \tau_{tx} + \rho_{rx} \cdot \tau_{rx} + \gamma_{xg} \cdot \lambda_{g+}\gamma_{xr} \cdot \lambda_r, \tag{8.2}$$

where τ_{tx} and τ_{rx} are channel airtime fraction for transmission and reception respectively. Table 8.2 describes remaining parameters and their corresponding values used to validate the model.

The time T_l spent for transmission of D bytes sent at rate r on LTE link is as follows:

$$T_l = T_{pr} + (D \cdot 8)/r. \tag{8.3}$$

T_{pr} is the promotion time, which is the time necessary to allocate resources to the device. This involves switching from a low-power state to a high-power state for transmission. Equation (8.3) is applicable for both uplink and downlink traffic. According to the model presented in [159], the power consumption P_l over the LTE link can be described as follows:

$$P_l = \alpha_u \cdot t_u + \alpha_d \cdot t_d + \beta. \tag{8.4}$$

Table 8.3 lists typical values for power consumption on LTE links.

8.1.2 Computation Offloading from Wearable Devices

Four different task execution models are available for wearable devices. They can execute tasks locally (w-l), offload to the smartphone (w-s), offload to the cloud (c) via smartphone (w-s-c) or offload to the cloud via WiFi access point (w-ap-c). The offloaded tasks are considered to be always accepted and processed in the cloud.

Local Processing in Wearable Devices (w-l): The simplest way is to start task execution locally. Task completion time is deterministic and depends on the computing power of the local hardware, which is quite limited for the majority of wearable devices. As a result, only tasks requiring low- and medium-size computing can be executed locally. While high-performance computing tasks would either take too long to complete or will drain the available battery power too fast.

Offloading from Wearable Devices to Smartphone (w-s): The nearest place to offload execution is a smartphone. Task execution can take benefits of larger computing, battery power, and storage resources of the smartphone, but require sending tasks for execution and receiving back the completion result. Yet, this communication occurs over short range and is energy efficient. The time needed to obtain results $T_{\text{w-s}}$ consists of the time necessary for sending data to the smartphone and the time to receive the results back using WiFi connection T_w and the processing time T_{sp} taken by the smartphone:

$$T_{\text{w-s}} = T_w + T_{sp}. \tag{8.5}$$

From the wearable device point-of-view, energy $E_{\text{w-s}}$ is only spent for communication purposes and can be described as follows:

$$E_{\text{w-s}} = T_w \cdot P_w. \tag{8.6}$$

We recall that definition of P_w can be found in (8.2).

Offloading from Wearable Devices to Cloud via Smartphone (w-s-c): When the smartphone acts as a relay in offloading tasks to the cloud, the time needed to obtain the results $T_{\text{w-s-c}}$ is defined as follows:

$$T_{\text{w-s-c}} = T_w + T_l + \delta_i + T_{cp}, \tag{8.7}$$

where T_w is the WiFi communication time between wearable device and the smartphone, T_l corresponds to the time spent over the LTE link as per (8.3) and δ_i corresponds to the wide-area network delay [194], while T_{cp} corresponds to the time taken by the cloud to perform computing. Energy consumption remains identical to the case (w-s) and it is described in (8.6).

Offloading from Wearable Devices to Cloud via Access Point (w-ap-c): Similarly to the second and third cases (w-s, w-s-c), when wearable devices

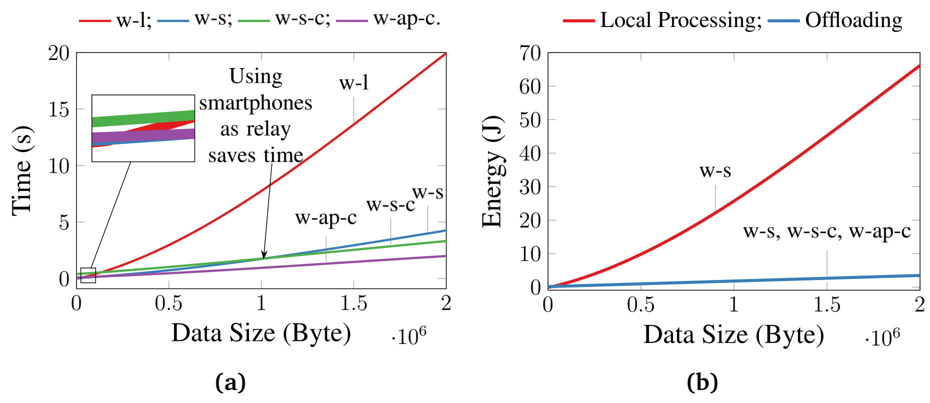


Figure 8.2: Wearable device (a) processing and communication time and (b) energy consumption

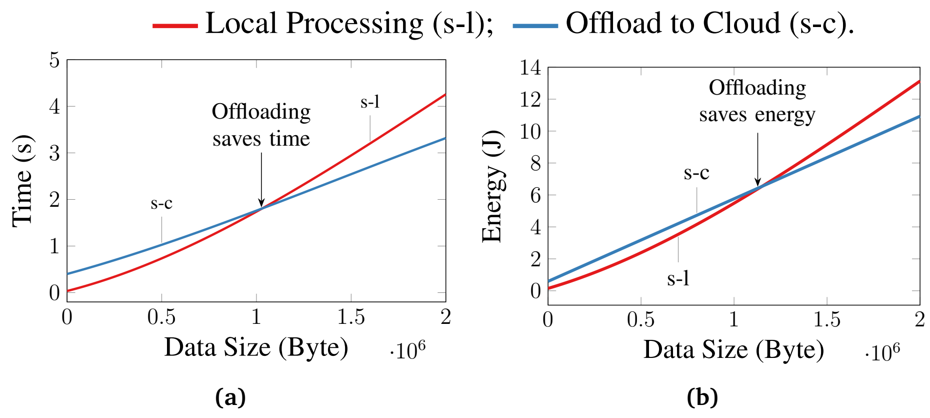


Figure 8.3: Smartphone (a) processing and communication time and (b) energy consumption

offload tasks to the remote cloud without using a smartphone as a relay, wearables spend energy only to communicate with the AP (8.6). To be generic, the proposed model differentiates between distances of wearable devices and the AP and distance between wearable devices and the smartphone. This allows capturing different channel conditions and accounts to variable latencies. In this scenario, the time wearable devices need to receive back the results is described as follows:

$$T_{\text{w-ap-c}} = T_w + \delta_i + T_{cp}, \tag{8.8}$$

where $T_w + \delta_i$ corresponds to the communications with the cloud through the AP and the Internet and T_{cp} is the processing time in the cloud.

Fig. 8.2 shows the time and energy comparison between the different cases having considered a face recognition task with images of variable

Table 8.3: LTE Setup Parameters

SYMBOL	VALUE	DESCRIPTION
α_u	$438.39 \cdot 10^{-9}$ W/bps	Power for bps in uplink
α_d	$51.97 \cdot 10^{-9}$ W/bps	Power for bps in downlink
α_p	$1210.7 \cdot 10^{-3}$ W/bps	Power for promotion
β	$1288.04 \cdot 10^{-9}$ W	Idle Power

size. Face recognition is both computing- and communication-intensive task [195]. We used Matlab model to derive relationship between the size of the picture and the number of instructions. The power spent for processing is set to be equal to 3318 mW per instruction on the wearable device and 2845 mW on the smartphone [191], [196]. Tables 8.2 and 8.3 list the communication parameters used for analytical validation of the model. As expected, Fig. 8.2 shows that offloading appears to be highly beneficial in terms of responsiveness and energy consumption as the amount of data to be transferred and processed increases. Specifically, wearable devices have advantage to offload tasks for local execution in the smartphone over offloading to the cloud when the data size is smaller than 1 MB. For objects of a larger size, the cloud always provides faster responses. In addition, Fig. 8.2(a) highlights that WiFi technology should be preferred over LTE when tasks are executed in the cloud.

8.1.3 Computation Offloading from Smartphones

From the smartphones' point of view, upon receiving a task from wearable devices they can either (a) execute the task locally (s-l) or (b) offload the task to the cloud (s-c).

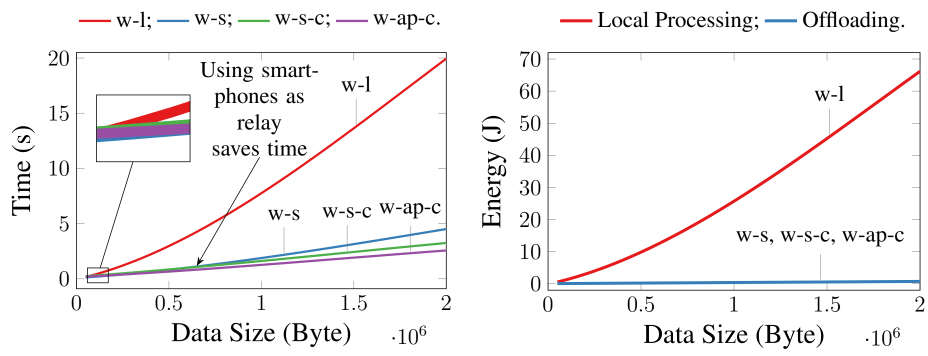
Local Processing in Smartphones (s-l): In such a scenario, the smartphone receives data from the wearable device performs the task locally and sends the results back. Communications happen through the WiFi link only. As a result, the time $T_{\text{s-l}}$ spent by the smartphone in assisting the wearable device for offloading is defined as:

$$T_{\text{s-l}} = T_w + T_{sp}, \tag{8.9}$$

where T_w corresponds to the communication time through WiFi and T_{sp} the time spent for local processing. Smartphones spend energy $E_{\text{s-l}}$ for both computing and communication:

$$E_{\text{s-l}} = T_w + P_w + T_{sp} \cdot P_{sp}, \tag{8.10}$$

where P_w corresponds to the WiFi power consumption as per (8.2) and P_{sp} is the power spent for local processing.



(a) **(b)**

Figure 8.4: Wearable device (a) processing and communication time and (b) energy consumption

Offloading from Smartphones to Cloud (s-c): Smartphones can receive data from wearable devices and to perform offloading to the cloud. In such case, no energy costs are associated with processing, but the smartphone is required to keep active both WiFi and LTE interfaces for communication for the time T_w and T_l. As a result, $T_{\text{s-c}}$ the time spent by the smartphone in assisting the wearable device for offloading is defined as:

$$T_{\text{s-c}} = T_w + T_l. \tag{8.11}$$

The energy spent by the smartphone is defined as:

$$E_{\text{s-c}} = T_w \cdot P_w + T_l \cdot P_l + \alpha_p \cdot T_{pr}, \tag{8.12}$$

where where P_w and P_l correspond to the WiFi and LTE power consumption defined in (8.2) and (8.4) respectively, while α_p corresponds to the power spent by smartphones during promotion time T_{pr} as per Table 8.3.

Fig. 8.3 shows the time and energy for smartphones. In contrast with the results obtained for wearables, Fig. 8.3(b) shows that it becomes more energy efficient if smartphones perform offloading if the size of data transfer is larger than 1.2 MB. This value is in full agreement with the expected during analysis value.

8.2 Validation

This section provides performance evaluation performed with NS-3 network simulator extended with LTE functionality from LENA project framework. Similarly to Section 8.1, face recognition is the application used for our analysis.

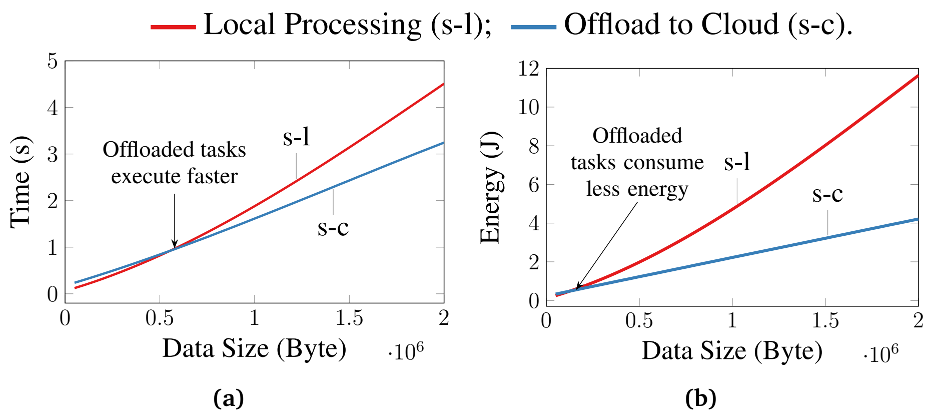


Figure 8.5: Smartphone (a) processing and communication time and (b) energy consumption

8.2.1 System Scenario

Google Glasses have been used as wearable device. They are equipped with a dual core ARM Cortex-A9 CPU with maximum frequency at 1 GHz [197]. This architecture is able to offer a maximum computational power of 5000 DMIPS. The power consumed to perform an OpenCV detection algorithm is 3318 mW, while data transmission consumes 653 mW using WiFi at 734 Kbps [196], [198]. LG Nexus 5 is the smartphone used for simulations. It is equipped with a quad core Qualcomm Snapdragon with a maximum computational power of 30645 DMIPS [199]. The simulated cloud is assumed to have a computational power of an Intel Core i7 3770K able to elaborate a maximum of 106926 DMIPS at 3.9 GHz [200].

For communications, the simulated WiFi data rates range from 734 Kbps to a maximum of 24 Mbps [196]. For LTE, in the uplink the supported data rates range from 0.924 Mbps up to 20 Mbps while in downlink the considered range is 2.24-40.2 Mbps. Distances between the devices (wearable and smartphones) and antennas (WiFi AP and LTE antennas) range between 0.5 and 15 m for WiFi and from 50 to 500 m for LTE. The size of simulated graphical objects range from 50 KB to a maximum size of 2 MB.

For analysis of energy consumption, the setting used for computing is exactly the same of Section 8.1. Wearable devices spend 653 mW and smartphone 1749 mW for communications over the WiFi links [196] and smartphones consume 2200 mW [192] over the LTE links.

8.2.2 Results

Fig. 8.4 shows the results of the simulation from the point of view of the wearable device. Similar to the model, the decision whether to offload a task

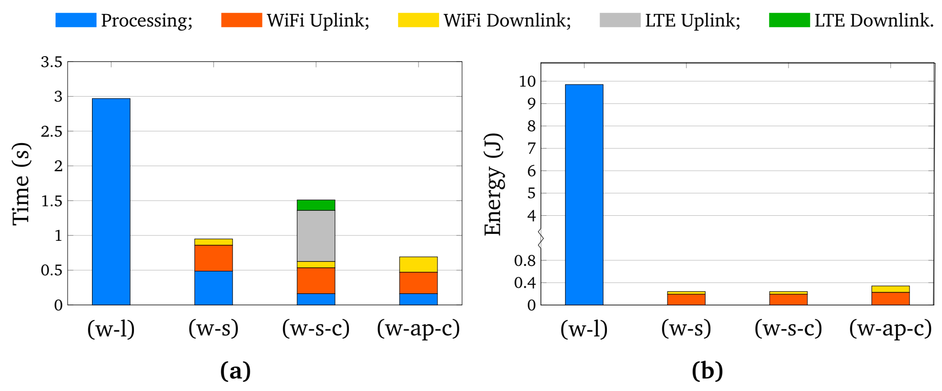


Figure 8.6: Distribution of (a) processing and communication time and (b) energy consumption for wearable devices

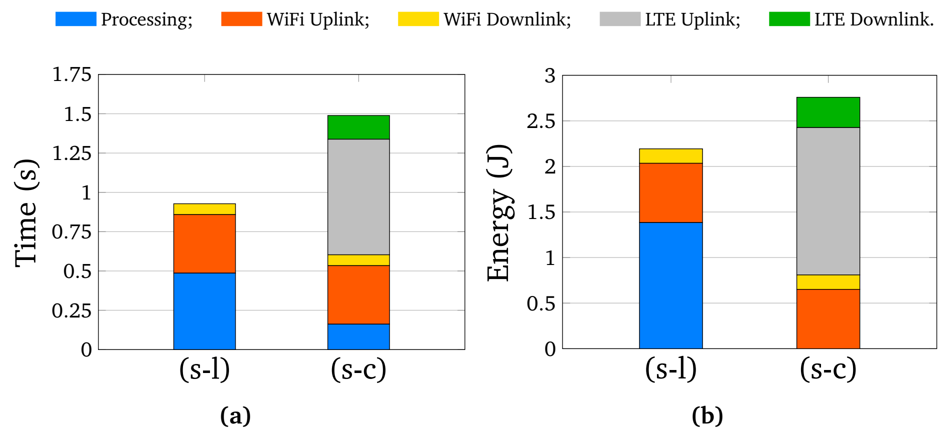


Figure 8.7: Distribution of (a) processing and communication time and (b) energy consumption for smartphones

or to execute locally depends on the amount of the data to be processed. For small data sizes, below 150 KB, the difference in execution time is marginal, while for bigger amounts of data to be processed the results are consistent with the proposed model. Offloading results being always more energy efficient as compared to local processing.

Fig. 8.5 shows the simulation results for smartphones. With respect to previous analysis, the behaviour of the charts is consistent with the model only for time responsiveness. More precisely, performing offloading starts being beneficial for amounts of data to be processed larger than 600 KB. For energy consumption, instead, offloading is practically always beneficial as the threshold is around 150 KB. The results presented in Fig. 8.5 differ from the theoretical model. This is especially evident for energy consumption (see comparison of Fig. 8.5(b) and Fig. 8.3(b)). On one hand, it is because NS3 implements more realistic models for communication, while on the other hand the results in Fig. 8.5 are averages having considered different channel conditions, which can not be captured by the theoretical model.

Fig. 8.6 shows task execution time and energy consumption measured at the wearable device during offloading of a 500 KB file. If performed locally, task execution results being a heavy operation. Indeed, offloading is beneficial for the lifetime of the device as all cases outperform local execution in terms of both time and energy. From time perspective, the best way is to offload tasks to the cloud using WiFi connection, because cloud performs computing faster than a mobile device. Direct connection with the cloud avoids communication over the time-consuming LTE network at the expense in energy consumption. Because of the need to cross the Evolved Packet Core (EPC) network [54], communications over LTE are time consuming and it becomes clear comparing the cases (w-s) and (w-s-c) in Fig. 8.6(a). For this amount of data (500 KB), the local processing at the smartphone leads to a faster response to the wearable device with the same energy cost. This can be considered as an intermediate case, as the data size is not small enough to be efficiently processed on the wearable device and not so large to be sent to the cloud.

Fig. 8.7 shows task execution time and energy consumption measured at the smartphone during offloading of a picture of 500 KB. Local processing at the smartphone provides faster response if compared to offloading the task to the cloud because of the communication over the LTE network. Indeed, even if the cloud can perform the task faster, the time spent in communication over LTE will affect negatively the performance with respect of both time and energy.

Chapter 9

Implementation: TreeGlass, an Android application for Google Glass

Following the guidelines the model developed in Chapter 8, this Chapter presents results obtained developing an application for Google Glasses. The application is designed to be easily partitioned into tasks. For evaluation purposes, the tasks can be run with a certain degree of freedom in different entities of the fog environment, such as the Google Glasses, the smartphone and the cloud. A Power monitor is used to profile and monitor energy consumption of the mobile devices.

9.1 The TreeGlass Application

9.1.1 Introduction and Objectives

TreeGlass application is designed to perform recognition of trees. The application runs in a fog computing platform with tasks distributed among the wearable device, the smartphone and the cloud. In a nutshell, *TreeGlass* operation is similar to the workflow of face recognition applications [195], [201], [202], i.e., unique features are first extracted from a picture (detection phase) and are compared against a predefined database (recognition phase). *TreeGlass* uses Google Glass to take pictures of leaves for detection and recognition. Fig. 9.1 illustrates the typical workflow of the application, where Fig. 9.1a shows the capture of the image and Fig. 9.1b the display of the result from the point of view of the user wearing the Google Glass.

To illustrate the workflows of *TreeGlass* with more details, first the leaf is detected from the picture and key features such as the color and the contour of the leaf are extracted. The contour of the leaves is a closed curve shape and can be analyzed with a similarity metric [203]. These features are

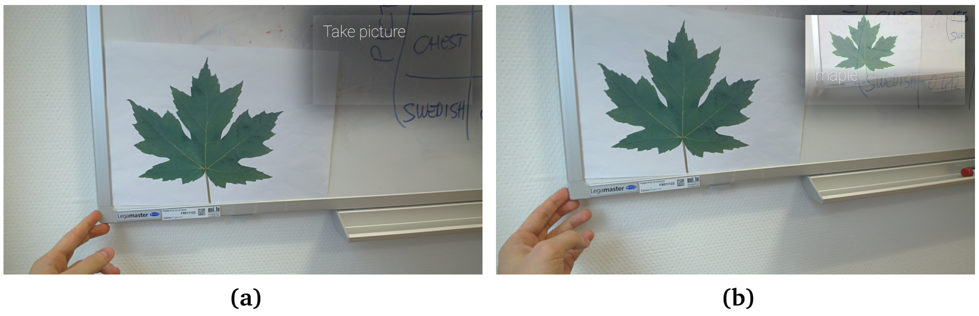


Figure 9.1: Execution TreeGlass from Google Glass (a) Initial and (b) Final Steps

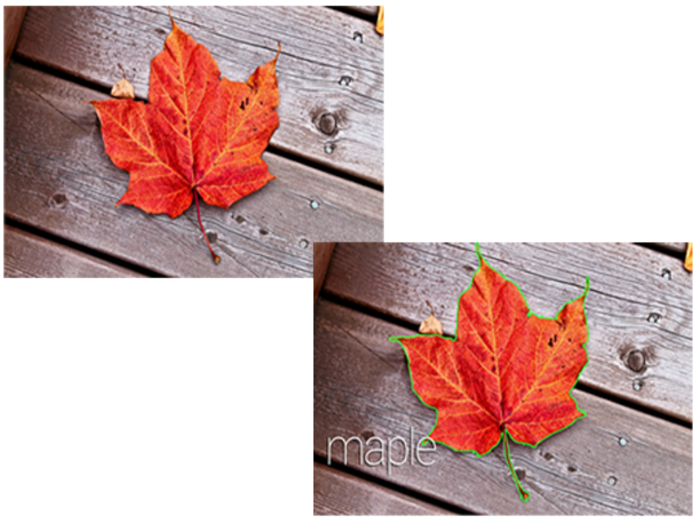


Figure 9.2: TreeGlass example execution

sent to the cloud and compared with the database in order to find a match. Whether the match is found or not, an answer is sent back and displayed to the user. Fig. 9.2 shows an example of positive match: the contour of the leaf is highlighted and the name of the tree displayed.

The ultimate goal of the application is to test in realistic scenarios the performance of application partitioning techniques (explained in details in Section 9.1.3) when applied to wearable devices in fog computing.

9.1.2 The architecture

TreeGlass has been developed with Android and can potentially run over any Android-based wearable device equipped with camera, smartphones and the cloud. In the experiments, Google Glasses were the reference wearable

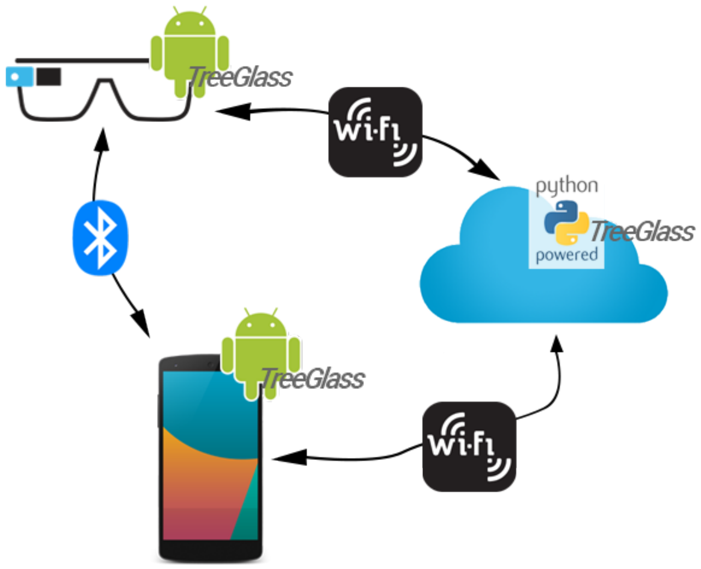


Figure 9.3: TreeGlass architecture

device. It should be noted that the application is designed in such a way that its entities can run simultaneously on wearable and smartphone devices. Fig. 9.3 illustrates *TreeGlass* architecture with all the three entities of the fog computing environment. The figure also highlights the technologies that allows the entities to communicate one with each other and the programming languages used.

The minimum operative system to support the application is Android KitKat 4.4.4 (API Level 19), which is the native version running on the Google Glass and that was used during all the experiments. The mobile application is written in Java and it exploits the standard APIs provided by the Android SDK.

The task that is always executed in the cloud is the recognition phase, i.e., the cloud is responsible to find a match between the features extracted from the picture and the features stored in the database. The cloud however, is also designed to run some of the tasks of the Android application devoted to image processing. This design choice has the objective of guaranteeing high flexibility in partitioning the application and to obtain an exhaustive comparison between all possible offloading scenarios. The part of *TreeGlass* that runs in the cloud is written in Python. Indeed, an Android version of the application on the cloud would run only under a simulator which is a not effective solution as it introduces overheads.

For the sake of simplicity, the database of the application consists of three types of leaves: oak, maple and chestnut. Each entry of the database contains a master picture of the leaf and the list of features such as color and contour.

The core part of the application is based on the OpenCV[1] library. OpenCV provides high level functions to manipulate images and extract features of objects within them. Moreover this library provides methods to compare features of different images and then return a usable metrics for the match finding. On top of the OpenCV library, the custom Java and Python libraries aggregate methods of the latter with the objective of to dividing the application in tasks. The tasks will be later run in different configurations to verify the performance for any possible offloading scenario. The Java and Python form a unique library called *TreeRecLib*. *TreeRecLib* Java version is exploited by the instances of the application that run on Google Glass and on the smartphone, while the Python version is used by the server in the cloud. Both versions of the library contain the same function calls and perform the same operations.

9.1.3 Application Partitioning in TreeGlass

TreeGlass is real-time application modeled with a set of tasks that have specific precedence constraints. The division in tasks is essential to quantify coherently the performance of the application in different offloading scenarios. A *task* is defined as a portion of the application execution which involves the call to one or more methods that having the same objective within the task.

Similarly to CA-DAG (see Section 7.2 for the details), *TreeGlass* DAGs are characterized by squared blocks representing computing tasks. Unlike CA-DAG, communication tasks are not depicted with rounded blocks. The motivation behind this design choice is that in fog computing, the description of a communication tasks requires the knowledge of the type of technology involved and the destination rather than a generic description. The main goal of CA-DAG, indeed, was to distinguish between computing and communication tasks, not to understand *where* computing tasks are run and *how* the information is transferred between one task and the other. As a result, *TreeGlass* DAGs layout is column-based, where each column represent an entity of the fog where the computation takes place and communications are represented with different types of arrows according to the technology used.

Depending on the offloading scenario (see Section 9.1.4 for the details), the application is partitioned into 5 computing tasks and up to 4 communication tasks. Computing tasks are the following:

- *Image Capture* (IC): Google Glass device captures the pictures. In all the offloading scenarios, this task is always executed locally on the Google Glass and does not contribute for the analysis of energy consumption for the following reasons. First, the task can not be logically and physically offloaded to other entities of the fog platform.

[1]`http://opencv.org/`

Table 9.1: Task Description

Task-ID	Color	Description
Ic	🟧	Pictures capture from the Google Glass.
Ip	🟪	Processing of input image is processed.
Fe	🟫	Extraction of the features of the image.
Fm	◼	Database query for match finding.
Bs	🟩	Preparation and display of the result.

Second, users spend arbitrary time to capture images and, moreover, they often perform this operation multiple times before obtaining an acceptable picture and then proceed to the next steps.

- *Image Processing* (Ip): The image processing methods of the OpenCV library prepare the picture for detection and recognition.

- *Features Extraction* (Fe): The elaborate image is analyzed for extraction of the key features. In this task allows to perform detection of the leaf.

- *Find Match* (Fm): Thanks to the features extracted, this task queries the database to find a match with the existing entries and sends back the positive or negative outcome of the process.

- *Build and Show* (Bs): Once the wearable device receives the feedback, it presents the outcome to the user. The result is "built" according to the user interface guidelines and then displayed to the user. This task is always performed by the Google Glass as it is the device in charge of the input/output operations.

To improve readability and understanding of offloading scenarios and the results, a color uniquely defines a computing task. Table 9.1 details task description and its associated color.

In *TreeGlass*, communication tasks transfer information from one computing task to another, which can run over a different entity of the fog platform or within the same device. The technologies used are Bluetooth and WiFi. Within the workflow of the application both of the technologies can be used, although the communication between two task can be performed by only one technology at a time. The Bluetooth communications are represented with a double arrow and they are always performed between Google Glass and smartphone. The WiFi communications are represented with a dot-dashed arrow and this technology is always used for data exchange involving the cloud independently of the device on the other side of the communication. Table 9.2 summarizes the different types of arrow and presents the description of a regular arrow used as representation of a time linear execution between

Table 9.2: Arrows Description

Arrow type	Description
$\longrightarrow$	A regular arrow indicates the simple time linear execution of tasks within the same entity.
$\dashrightarrow$	A dash dotted arrow indicates a data transfer between two entities using WiFi technology.
$\Longrightarrow$	A double arrow indicates a data transfer between two entities using Bluetooth technology.

computation tasks within the same entity of the fog platform. Similarly to the computing tasks, all the communication operations such as *data sent via Bluetooth* or *data received via WiFi* are uniquely characterized by a color. However, the mapping of operations and colors is presented in Section 9.2 as it is not utilized for the DAGs description.

9.1.4 Offloading Scenarios

TreeGlass takes advantages of all devices and entities of the fog platform where the computing tasks are executed. The wearable device and the cloud are always part involved in computing. If the smartphone is utilized as well, then formally *TreeGlass* is partitioned in a fog computing platform otherwise *TreeGlass* is partitioned in a mobile cloud computing platform. Overall, four scenarios are identified and they cover all the possible cases of *TreeGlass* partitioning over wearable devices, smartphones and the cloud. The scenarios comply with the theoretical model proposed in Section 8.1.2.

Scenario Gg-C Local Fig. 9.4 depicts the first scenario, which involves only two entities: the wearable device and the cloud. This scenario o is called Gg-C Local because almost the whole computation effort is carried locally on the Google Glass. The communication between the two entities is WiFi. The wearable device performs locally the first three tasks of the application (Ic, Ip, Fe) and it sends the resulting data to the cloud. The data sent to the cloud is a JSON string containing the features extracted from the picture captured by the user. Once the cloud receives this information, it compares the data with the entries in the database and it sends back to the Google Glass another JSON string containing the resulting match or mismatch (task Fm). The final task the wearable device has to complete is to build the received result and display it to the user (task Bs).

Scenario Gg-C Remote The second scenario is very similar to the previous one. Fig. 9.5 shows that the entities involved in the computation are the

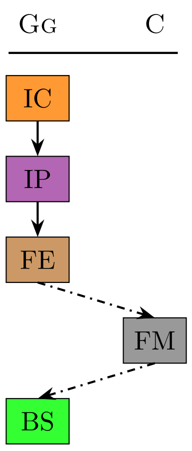


Figure 9.4: Scenario GG-C LOCAL

same and they use the same communication technology as before. This scenario differs from the previous one because the wearable device is only in charge of the input/output operations while all the computation is completely offloaded to the cloud. This scenario is therefore called GG-C REMOTE and from the computation perspective is at the opposite side of GG-C LOCAL. The Google Glass only performs the tasks IC and BS providing the input data to the cloud and displaying the final result to the user. Thus, after the wearable device captures the picture, *the whole picture* is sent via WiFi to the cloud where tasks IP, FE and FM are executed. The result is then sent back to the Google Glass using the same technology as the initial communication. The two communications this time exchange the whole picture instead of just a string as in the GG-C LOCAL scenario.

Scenario GG-SM-C FOG Fig. 9.6 shows the third scenario. Unlike GG-C LOCAL and GG-C REMOTE, which follow a traditional mobile cloud computing paradigm, this scenario is an example of fog computing approach. In fact, part of the computation is offloaded close to the user, namely on the smartphone which is the third entity of the fog platform with the wearable device and the cloud.

Similarly to GG-C REMOTE, in this scenario the Google Glass is in charge of the input/output operations by capturing the input image in task IC and displaying the final result in task BS. From the point of view of the wearable device, the computation is completely offloaded like in the GG-C REMOTE scenario. The wearable device sends via Bluetooth the image captured in task IC to the smartphone. The motivations behind this design choice are the following: i) Bluetooth is more energy efficient than WiFi and is specifically

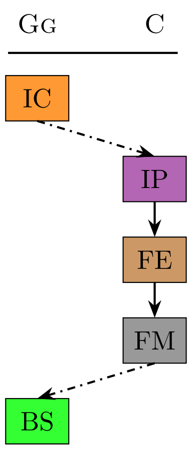


Figure 9.5: Scenario GG-C REMOTE

designed for communications in small operative range; ii) the majority of mobile OS already provide pairing functionalities between smartphones and IoT devices using Bluetooth, iii) the Google Glass can connect to a new WiFi network only if it is not protected by any password. The companion application installed on the smartphone helps the wearable device to connect to a new network protected by a password. Unfortunately, it is not possible to connect the Google Glass to a network that needs credential and this is the case where the smartphone is needed to connect the wearable device to the network using the latter as relay.

Once the information reaches the smartphone, it can proceed and reach the cloud via cellular technology or through WiFi according to the theoretical model proposed in Section 8.1.2. First, for practical reasons, WiFi technology is simpler to adopt and to profile and assess its energy consumption. The smartphone performs some preliminary computation on the received image and executes tasks IP and FE before involving the cloud. The smartphone and the cloud communicate trough WiFi and, as in scenario GG-C LOCAL, the exchanged data is a string containing the featured extracted by the smartphone. The cloud always executes task FM and it sends back the result to the smartphone as string with the same communication technology. The final result (task BS) has to be displayed by the Google Glass because of the direct interaction with the user. Thus the smartphone sends via Bluetooth the resulting image which the wearable device has to display with task BS at the end. Finally the smartphone is only in charge of forwarding the final result to the wearable device.

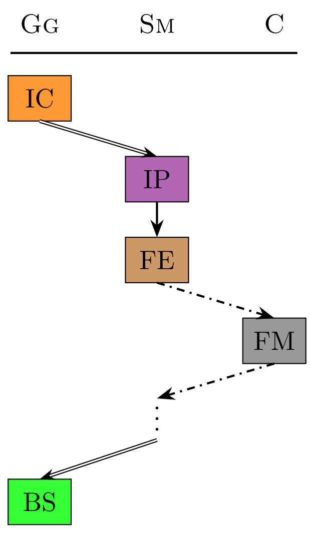


Figure 9.6: Scenario GG-SM-C FOG

Scenario GG-SM-C RELAY Fig. 9.7 depicts the fourth and last scenario, where the entities and the communications technologies are the same of the GG-SM-C FOG scenario. Consequently, the GG-SM-C RELAY is also an example of fog computing approach. In this scenario, the smartphone is only used for communication purposes and the computation is completely offloaded to the cloud. As a result, in GG-SM-C FOG scenario the computing capabilities of the fog are utilized while GG-SM-C RELAY uses its communication potential. The smartphone, indeed, behaves as relay for the Google Glass by redirecting the received data from the wearable device to the cloud and vice versa when the cloud makes available the result. The Google Glass execute input/output related tasks such as tasks IC and BS. Communications are performed again using Bluetooth technology with the smartphone and the exchanged data is the raw input image (output of task IC), and the result given from the cloud (input of task BS). The smartphone, after having received the data from the Google Glass, immediately sends the image to the cloud via WiFi with no additional operations than the preparation of data needed to manipulate the stream coming from the Bluetooth communication stack and redirect it to the WiFi communication stream. Thus, the cloud is in charge of performing the tasks IP, FE and FM. After successful execution of FM, the cloud sends back to the Google Glass the result, using the smartphone as relay in the opposite communication order of the beginning. Fig. 9.7 shows two blank spaces left in correspondence of the smartphone column. They represent the operations that the smartphone does in order to forward the data between wearable device and the cloud. It is not a unique arrow because between the

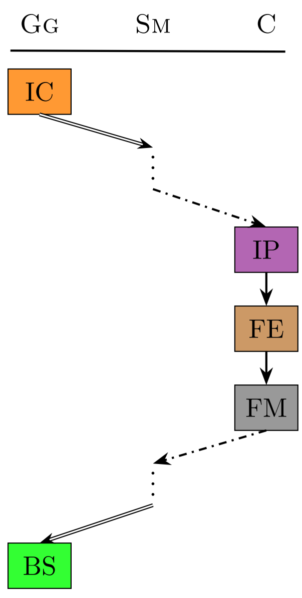


Figure 9.7: Scenario GG-SM-C RELAY

two communication there are small operations performed by the smartphone so it is not immediate forwarding.

9.2 Evaluation

9.2.1 Equipment and Settings

TreeGlass has been practically implemented on real devices. In the following, the details of the all equipment are presented.

Smartphone

The smartphone device used for the experiments is the Samsung Galaxy Note 4. The choice of smartphone is not driven by a particular need; the only requirement is that the device should be an Android-based device. The Google Glass could be paired with an iOS device as well but, using an Android phone provided more flexibility during the implementation and while performing the experiments. Having different OS would have introduced overhead by translating the application code to the language needed to execute it.

During the experiments, the smartphone runs Android Lollipop 5.1.1 (API Level 23) which is the latest version available for the device at the time. It is equipped with a quad-core 2.7 GHz Krait 450 and 3 GB of RAM; furthermore it has a GPU Adreno 420 but, for evaluation purposes, this element is not involved in the process. The chipset is a Qualcomm Snapdragon 805. The display is a 5.7 in super AMOLED capacitive touchscreen with a resolution of 1440×2560 pixels. Moreover it is equipped with a 16 MP camera, 32 GB of flash storage and other less relevant sensor for the purposes of this research. On the connectivity side, the Galaxy Note 4 provides WiFi connectivity with standard 802.11 a/b/g/n/ac and a Bluetooth v4.1. Finally this smartphone device is powered by a 3220 mA and 4.4 V battery.

Wearable device: Google Glass

The Google Glass is the wearable device used in the experiment. It is equipped with many sensors such as gyroscope, accelerometer, compass and, unlike the vast majority of other wearable devices, it is equipped with a camera. The camera is crucial for the purposes of the application and the related experiments because it provides the input of the application process. The Google Glass operative system is Android based and it runs a special release of Android KitKat 4.4.4 (API Level 19). This wearable device is equipped with a dual-core OMAP 4430 System on a chip processor and 2 GB of RAM. The display is characterized by a Prism projector of 640×360 pixels (equivalent of a 25 in/64 cm screen from 8 ft/2.4 m away. Moreover it is equipped with a 5 MP camera, 16 GB of flash storage and other less relevant sensor for my purposes. On the connectivity side, Google Glass provides WiFi connectivity with standard 802.11 b/g at 2.4 GHz and a Bluetooth antenna for device to device communication. Finally this wearable device is powered by a 570 mA and 3.7 V battery. Google provides a companion application called *MyGlass*[2]: it accessible to everyone on the official Google Play and Apple AppStore and let the user set up, manage, and add new features to the Glass device. By connecting the wearable device to the smartphone via Bluetooth, this application allows to add WiFi networks, manage the photo gallery, perform some realtime screencast of the running operative system and install/uninstall applications. The pairing between the two devices allows the wearable device to access the internet by using the cellular network of the smartphone and the tethering function of the latter.

Cloud

The cloud is simulated with a personal laptop connected to the same LAN. Since the computational power and the hardware of the laptop offer better performances than the two mobile devices, it can be used to simulate the cloud part of the environment. The laptop is a MacBook Pro (Retina, 13-inch, Mid 2014). The laptop runs OS X El Capitan version 10.11 at the time of

[2]http://play.google.com/store/apps/details?id=com.google.glass.companion

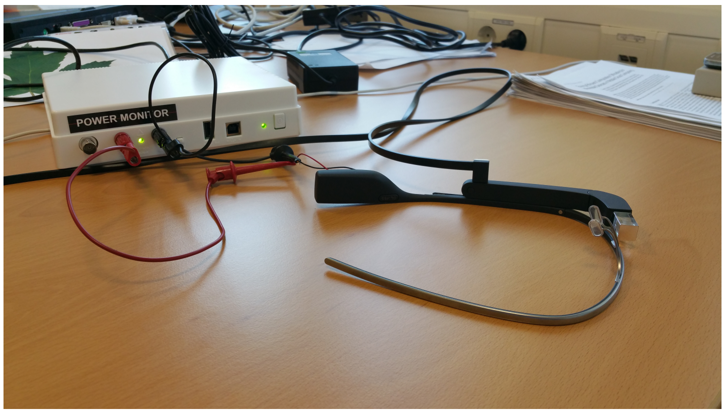


Figure 9.8: Power measurement set up for Google Glass

the tests; it is equipped with a dual-core 3.0 GHz Intel Core i7 and 16 GB of RAM; it has a 256 GB Apple SSD as storage and an AirPort Extreme card for Wi-FI 802.11 a/b/g/n/ac connectivity.

Power Monitor

The Power Monitor hardware by Monsoon[3] is the fundamental tool in the power measurements phase. Several papers in the scientific literature use the approach of collecting power consumption data via software by means of system calls [204], [205]. Even though this method may be valid, the power monitor provides extremely accurate results if used directly to acquire measurements. In order to retrieve data, the power monitor substitutes the battery of the mobile device. At the other side, the monitor is connected to a laptop running a Windows operative system. Thanks to a specific software provided by Monsoon, data is downloaded in real time to the laptop with a sampling rate of 5000 samples per second. The software displays a real time chart of the sampled data at the same time and it allows me to save the session in a `csv` file at the end of the session.

9.2.2 Results

The power monitor helped measuring two different performance metrics, i.e., execution time and power consumption from the point of view of the Google Glass and the smartphone, see Fig. 9.8 and Fig. 9.9 respectively.

Taking into account both computing and communication operations performed by [TreeGlass], execution time was measured first. The results achieved are averages over multiple runs. Having knowledge of the execution time at the granularity of a task makes easier to determine the energy

[3]http://www.msoon.com/LabEquipment/PowerMonitor/

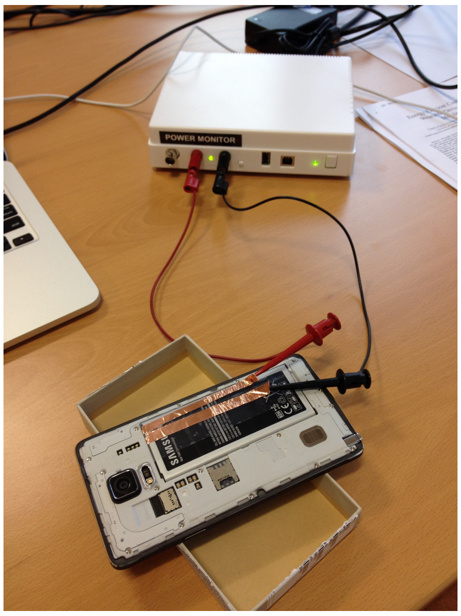


Figure 9.9: Power measurement set up for Google Glass

consumption of each task and, therefore, to compute the overall energy consumption of the application.

Results of Execution Time

The execution time of any application depends on many factors including the environmental conditions, current level of the battery and eventual energy-saving mechanisms set on the devices. It should be noted the aforementioned issues do not affect computing tasks as much as communication tasks. For example, environmental aspects such as fading and shadowing, influence channel conditions and directly affect performance of communications.

Execution time of Computing Tasks: To limit the influence of external factors in the measurements, *TreeGlass* is the only application running on Google Glass and the smartphone. For this evaluation, the scenarios GG-C LOCAL and GG-SM-C FOG are considered as in these cases the cloud is involved in computing aspects but the database search. Table 9.3 presents the results for the execution time spent for computing by all the tasks but tasks IC and FM. Indeed, these two tasks are not relevant to assess the performance of the application. Task IC involves taking the picture and the duration of the tasks is highly dependent on the user. Task FM corresponds to the query in the database to find a match and given that is always performed in the cloud, its duration can be considered negligible with respect to the other tasks. It should be noted that task BS displaying the final result is always performed

Table 9.3: Execution times of computing tasks on mobile devices

TASK	GOOGLE GLASS	SMARTPHONE
TASK IP	0.711 s	0.594 s
TASK FE	0.125 s	0.114 s
TASK BS	2.853 s	–

Table 9.4: Execution times of computing tasks on mobile devices

TECHNOLOGY & DATA	GOOGLE GLASS		SMARTPHONE	
	SEND	RECEIVE	SEND	RECEIVE
WiFi - Text	0.209 s	–	0.017 s	–
WiFi - Image data	12.079 s	2.906 s	1.809 s	0.416 s
Bluetooth - Image Data	15.402 s	4.035 s	0.370 s	8.017 s

locally at the Google Glass and the table clearly shows that it is the task that last longer. Indeed, behind the scenes, the BS task is responsible of image merging in addition to display the result, which is a very time and energy consuming operation. As expected, Table 9.3 also shows that the tasks IP and FE are executed faster if performed by the smartphone.

Execution time of Communication Tasks: For the measurements, all the devices in the fog platform were connected under the same LAN (eduroam), which is however not under control. Therefore, having other users connected and generating traffic somehow degrades the performance of the test because of interferences. Similarly to computing tasks, Table 9.4 shows the results performed for IP, FE and BS tasks in GG-C LOCAL and GG-SM-C FOG scenarios. The smartphone always outperforms the Google Glass of at least one order of magnitude as it is equipped with most recent hardware and support updated versions of the communication technologies. The Google Glass transfer different type of data. In GG-C LOCAL scenario, as most of the computing part is performed locally, only a JSON string is sent to the cloud. In GG-SM-C FOG, the Google Glass are only in charge of capturing the image, which is then sent to the smartphone and or relayed to the cloud with Bluetooth and or WiFi. Considering Bluetooth as communication technology, the differences in terms of time execution are evident. The smartphone performs operations faster than the wearable device, with the sole exception when the smartphone receives data via Bluetooth. This happens for the GG-SM-C FOG scenario and the it is due to the fact that the bottleneck is the time spent from the wearable device to send data via Bluetooth to the smartphone. The latter is then forced to use more time in order to wait the wearable device to send all the data.

Table 9.5: Color code definition in charts

COLOR	DESCRIPTION
▨	Execution of computing tasks IP and FE.
🟩	Execution of computing task BS: preparation and display of the result.
🟥	Send data via WiFi from the current entity to another one.
■	Receive data via WiFi in the current entity from another one.
🟪	Send data via Bluetooth from the current entity to another one.
🟦	Receive data via Bluetooth in the current entity from another one.
🟨	Waiting time while the device is in idle mode in respect to the current application.

Results of Enegy Consumption

The results presented in this section are obtained with the help of the Monsoon power monitor and with precise knowledge of the task duration estimated in Section 9.2.2. More precisely, the power monitor presents only the power consumption profile for the *entire* application duration. Therefore, the energy spent by each task is determined splitting the whole profile into pieces of predetermined duration.

Table 9.5 shows the color code use to present the results. It differentiate between computing and communication tasks. The charts are mainly characterized by colors that refer to the execution of the communication operations because time execution of computing tasks is faster in comparison to the duration of communication tasks. It should be noted that for computing tasks the color code is similar to the one presented in Table 9.1, and for task BS it is the same. Being very short, tasks IP and FE are merged together and represented with a dotted-pattern where the background is a combination of the correspondent backgrounds shown in Table 9.1. This operation allows to improve the readability, otherwise they would have not appeared in the charts. Communication tasks are colored with different shadings of the same color according to the operation and technology used, WiFi - Bluetooth and Send - Receive. Finally, yellow denotes the waiting time, i.e., the time when an entity waits the result from another one in order to proceed with the execution of the task.

Analysis of GG-C LOCAL: Fig. 9.10 shows the power consumed by Google Glass in the scenario GG-C LOCAL and Table 9.6. The initial part of the power profile is characterized by picks of energy consumption that almost reach values of 2500 mW due to the start up of the application and the local computation performed on the wearable device. As explained, the

Table 9.6: Energy values of GG-C LOCAL scenario

OPERATION	ENERGY CONSUMPTION
TASK IP & FE	1.041 J
WIFI SEND	0.505 J
WAITING	1.304 J
WIFI RECEIVE	3.772 J
TASK BS	2.637 J

execution of the tasks IP and FE is fast, but the sum of the two operation is easy to be detected in the chart. After the FE phase, the extracted features are sent to the cloud in form of a JSON string via WiFi. This requires less energy than the local computation, but it takes more time to complete the operation (see the red section). Then, the Google Glass wait for the cloud performing the database search and sending back the results. It is interesting to notice that during the waiting time, the device consumes a similar amount of energy that while transmitting data with WiFi. The reason is that the device remains in listening mode, keeping running in background all the functionalities of *TreeGlass*. The longer task is the reception of the result from the cloud (pink section). The power trend is similar to the sending and waiting periods but it presents higher peeks towards the end of the phase to process the received stream of data. Consequently the energy consumption is higher in respect to the other sections. It is interesting to notice that during the communication phases and the waiting time, the power consumption presents a trend almost stable around 1300 mW. The final green sector represents the final computation task executed by the Google Glass where the result is built and displayed to the user. Task BS requires lot of power because it has to update the user interface of the application in order to show the result to the user. Thus, the screen consumes energy to be refreshed and the peeks of power consumption have almost the same height of the ones in the initial phase, tasks IP and FE.

Analysis of GG-C REMOTE: Fig. 9.11 shows the performance of the GG-C REMOTE scenario and the correspondent values of energy consumption are presented in Table 9.7. Similarly to the previous case, these results are presented from the point of view of the Google Glass because from the computing point of view, this scenario is the opposite of the previous one. In GG-C LOCAL all the computing tasks are executed locally, in GG-C REMOTE the computing tasks are offloaded. Thus, the large red sector that characterize the initial WiFi send represents the sending of the input image to the cloud in order to be further processed. Differently from the previous scenario, the data sent to the cloud is not only text, but the entire image is compressed and sent as a stream. The amount of data is then significantly

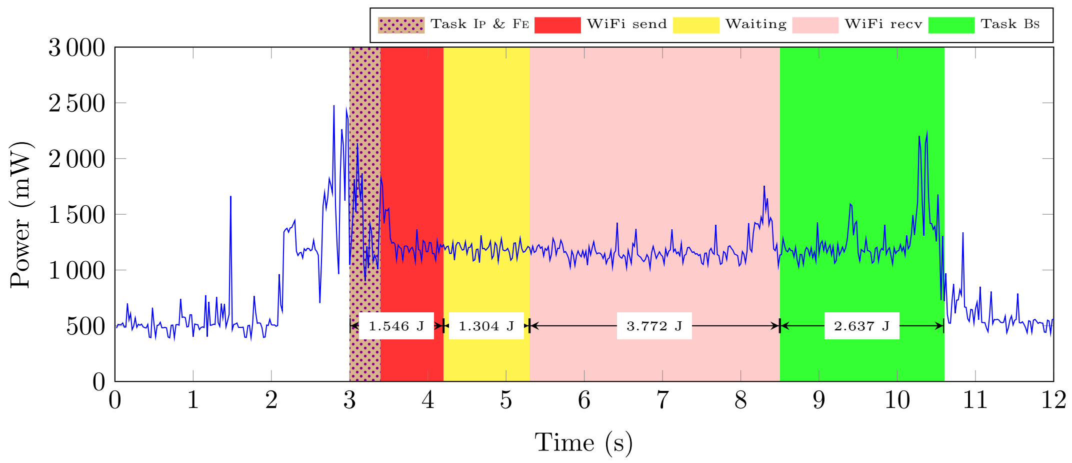


Figure 9.10: Scenario GG-C LOCAL Power consumption

Table 9.7: Energy values of GG-C REMOTE scenario

OPERATION	ENERGY CONSUMPTION
WIFI SEND	12.505 J
WAITING	0.429 J
WIFI RECEIVE	1.075 J
TASK BS	1.116 J

higher than in the GG-C LOCAL scenario and it requires more time and power to the Google Glass to perform the task. The power profile reaches peeks higher than 2500 mW in some points and the total amount of energy spent for this operation is around 12.505 J. Once the image reaches the cloud, it is processed extremely fast (tasks IP, FE and FM see Fig. 9.5) and the Google Glass wait spends little amount of time waiting. As all the results are already prepared to be showed, the WiFi - recv operation is short although consumes a considerable amount of power. Similarly, also the duration of the BS task is faster than in the GG-C LOCAL scenario and reaches higher peak of power consumption.

Summary: GG-C LOCAL vs GG-C REMOTE:

The comparison between the GG-C LOCAL and GG-C REMOTE scenarios is very important to understand pros and cons of mobile cloud computing. Both cases have the same entities involved and they differ one from the other by the amount of computing tasks offloaded and the type of data involved in the communication. In GG-C REMOTE, the long time needed by the wearable device to send the data to the cloud limits the effect of offloading the computation performed locally in the GG-C LOCAL scenario. Even if the computation is way faster if performed by the cloud, the initial

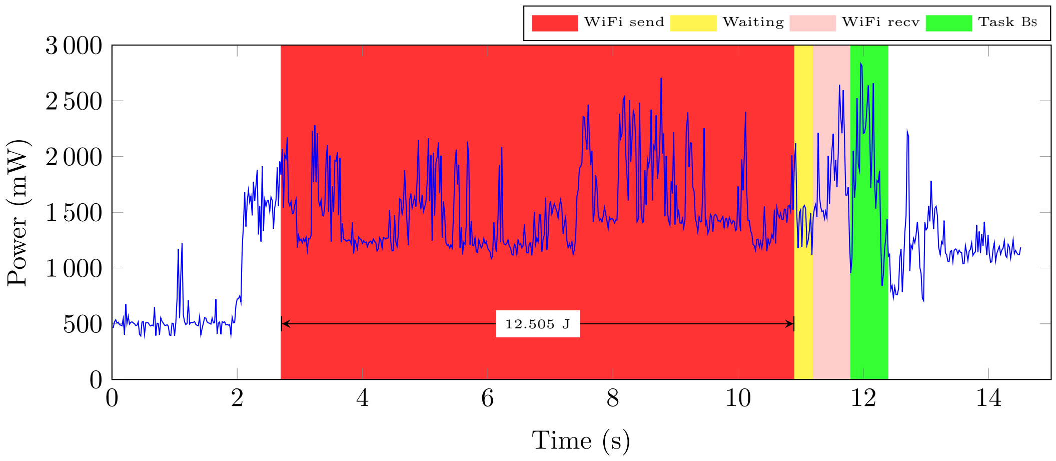


Figure 9.11: Scenario GG-C REMOTE Power consumption

communication operation requires a large amount of energy that is higher than the overall consumption of GG-C LOCAL. It is important to notice that to maintain the same functionalities, the two scenarios requires that Google Glass sends two different type of data: a string for GG-C LOCAL and the raw image for GG-C REMOTE. On one hand, in the GG-C REMOTE scenario, compressing and sending the image are the expensive operation in terms of energy and time consumption. On the other hand, in the GG-C LOCAL scenario the wearable device is able to stabilize its power consumption before receiving the result from the cloud, in the second scenario the long time spent sending the image does not allow the device to cool down and save energy.

Analysis of GG-SM-C FOG: The scenario GG-SM-C FOG is a fog computing approach. Indeed, the GG-SM-C FOG case involves all three entities in performing the computing tasks. More in details, the smartphone is in charge of prepare the data and send it to the cloud for further processing and database lookup. Fig. 9.12 displays the power consumption of the Google Glass, while Fig. 9.13 displays the power consumption of the smartphone. Similarly to the previous scenarios, the values of the energy consumption per task are summarized in Table 9.8 for the wearable device and in Table 9.9 for the smartphone. The Google Glass and the smartphone communicate through Bluetooth and the smartphone uses WiFi for communications with the cloud. Similarly to GG-C REMOTE scenario, the most expensive operation in terms of energy consumption executed by the wearable device is the initial data transmission (see the blue section). This task takes a quite long time to be completed: indeed, in addition to the the data transfer itself, it involves the compression of the image. Altogether, this corresponds to a power draw which is on average higher more than 1500 mW. Once this first operation is

Table 9.8: Energy values of GG-SM-C FOG scenario for Google Glass

OPERATION	ENERGY CONSUMPTION
BLUETOOTH SEND	19.926 J
WAITING	7.011 J
BLUETOOTH RECEIVE	5.250 J
TASK BS	4.107 J

completed, the wearable device waits. The smartphone processes the image, sends the results to the cloud for the database lookup and receives back a reply, which is then forwarded to the Google Glass. In this time window, the power profile varies a lot because the device is listening on the Bluetooth connection for the reply and tries to cool down and stabilize. The cyan sector identifies the reception of the result by the Google Glass. It is a quite costly operation because of the decompression of the image, which leads to an overall energy consumption of 5.250 J. The green sector represents again the final step of the result display performed in task BS.

Fig. 9.13 is the power profile from the smartphone point of view during GG-SM-C FOG. Since the wearable device takes a long time to complete the image data transfer, the smartphone remains in receiving mode for a long time as well (see the cyan sector). Unlike the Google Glass, the smartphone consumes less energy during this period because of the more performing Bluetooth antenna and the power consumption profile is almost stable around 500 mW. Once the reception of the image is completed, the smartphone executes tasks IP and FE. The execution is visible as the peeks that are high as much as 3500 mW highlighted in the brown-dotted-pattern of Fig. 9.13. After the execution of task FE, the smartphone sends the features extracted from the image in form of a JSON string to the cloud (red period), waits (yellow period) the reception of the results (pink period), which are forwarded back to the Google Glass (dark blue period). It is worth noting the efficiency of the device in stabilizing the power consumption between the data uploading and downloading performed with WiFi. Another interesting fact is that Bluetooth send and receive operations lead to similar energy consumption profiles.

Analysis of GG-SM-C RELAY: From the point of view of the Google Glass, the scenario GG-SM-C RELAY is almost identical to GG-SM-C FOG. Fig. 9.14 illustrates the power consumption profile and Table 9.10 shows the corresponding values, which are totally in line with the ones already presented in Table 9.8. From the smartphone point of view, however, this is not true. Indeed, the GG-SM-C RELAY uses the communication and not computing resources of the fog and the smartphone acts as a relay, forwarding the raw image to the cloud without performing any other operation. While the initial reception of the image with Bluetooth costs a similar amount of time and

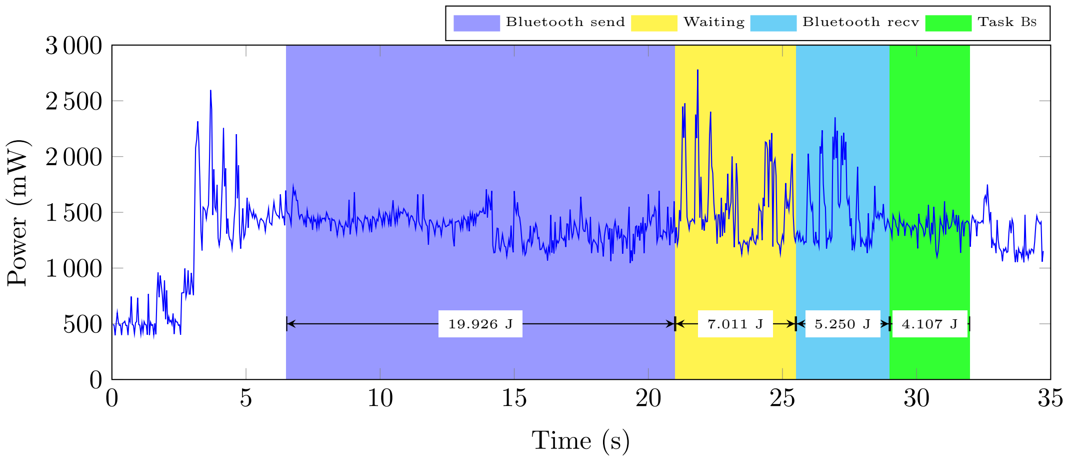


Figure 9.12: Scenario GG-SM-C FOG Google Glass power consumption

Table 9.9: Energy values of GG-SM-C FOG scenario for the smartphone

OPERATION	ENERGY CONSUMPTION
BLUETOOTH RECEIVE	4.360 J
TASK IP & FE	1.973 J
WIFI SEND	0.247 J
WAITING	0.558 J
WIFI RECEIVE	0.439 J
BLUETOOTH SEND	0.618 J

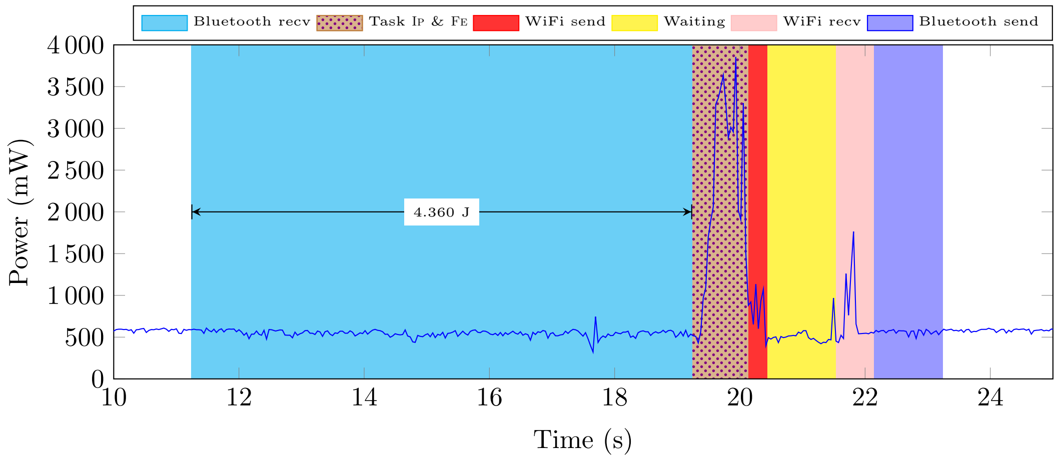


Figure 9.13: Scenario GG-SM-C FOG smartphone power consumption

Table 9.10: Energy values of GG-SM-C RELAY scenario for Google Glass

OPERATION	ENERGY CONSUMPTION
BLUETOOTH SEND	20.658 J
WAITING	4.867 J
BLUETOOTH RECEIVE	5.281 J
TASK BS	5.362 J

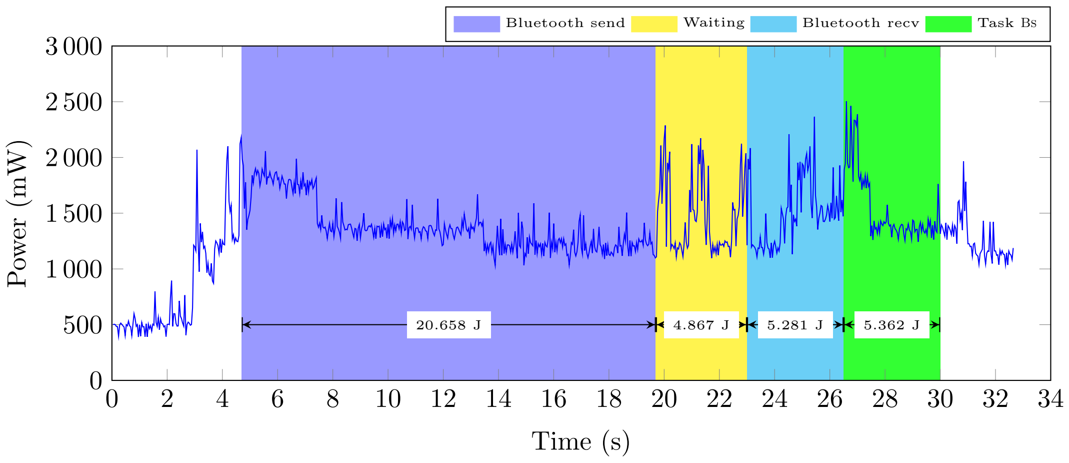


Figure 9.14: Scenario GG-SM-C RELAY Google Glass power consumption

energy than the GG-SM-C FOG scenario, then the uploading of the image and the download of the results from the cloud with WiFi have an higher cost as expected. Both operations are similar in time and power consumption as Table 9.11 confirms. Between the two operations there is a small time window when the smartphone waits for the execution of the tasks IP, FE and FM from the cloud. This time is highlighted in yellow and is short and almost imperceptible due to the fast execution of the cloud. The final part of the power profile, shown in dark blue, indicates the Bluetooth communication with the wearable device where the result is sent.

Table 9.11: Energy values of GG-SM-C RELAY scenario for the smartphone

OPERATION	ENERGY CONSUMPTION
BLUETOOTH RECEIVE	4.400 J
WIFI SEND	1.975 J
WAITING	0.255 J
WIFI RECEIVE	2.066 J
BLUETOOTH SEND	0.534 J

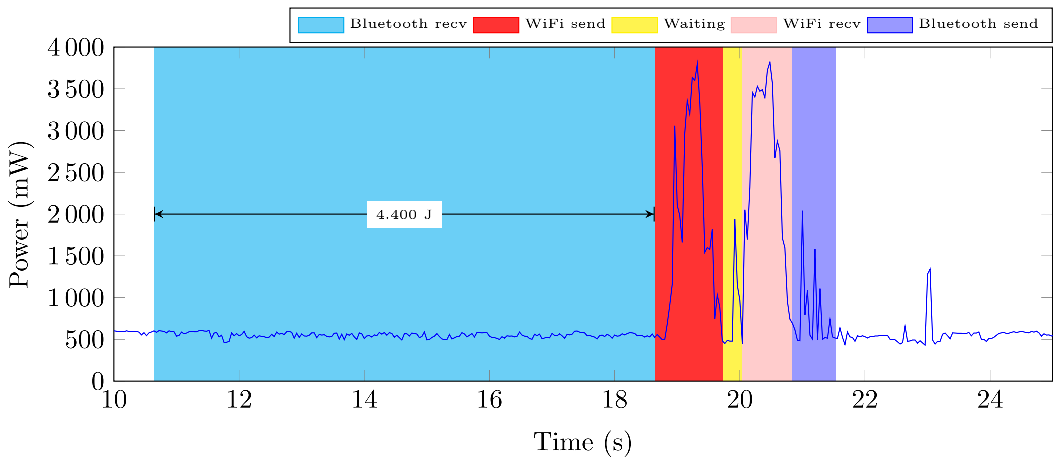


Figure 9.15: Scenario GG-SM-C RELAY smartphone power consumption

Summary: GG-SM-C FOG vs GG-SM-C RELAY:

Looking at the two fog computing-based scenarios where all the three entities are involved is not easy to find relevant differences. The execution time and power consumption values are similar in both cases. In fact the overall energy consumption of the Google Glass is 36.294 J in GG-SM-C FOG scenario and 36.168 J in GG-SM-C RELAY scenario while the smartphone consumes 8.195 J in GG-SM-C FOG scenario and 9.230 J in GG-SM-C RELAY scenario. From the point of view of the smartphone, performing locally the computation is almost negligible with respect to the communication tasks. A significant difference though is visible between the two scenarios GG-SM-C RELAY is almost identical to GG-SM-C FOG for what concern the data transfer: the transmission of a string is fast and it costs a low amount of energy, while the transfer of the entire image requires more resources.

Comparison of all the scenarios:

After having presented all the scenarios separately, this section provides an overall comparison of energy consumed for computing tasks (Fig. 9.16) and communication tasks (Fig. 9.17) in all the scenarios. Moreover, Fig. 9.18 gives an understanding of the overall consumption per scenario, including the waiting time.

As pointed out in the previous sections, the communication tasks are costly in terms of time and power consumption consequently. On the other hand, the computation tasks require less time but the power consumption is higher. Thus, the final outcome is very similar between the two types of task. Fig. 9.18 shows clearly this concept. For the scenarios GG-SM-C FOG and GG-SM-C RELAY, although the computation is distributed among several entities (Google Glass, smartphone and cloud) and the energy con-

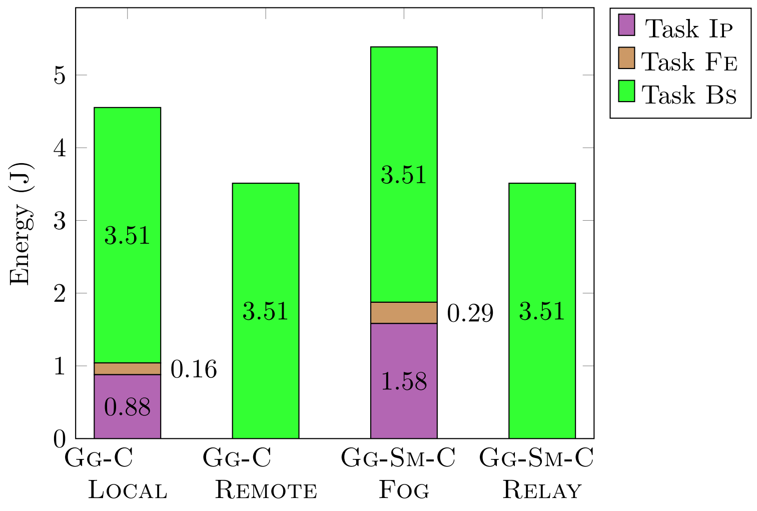


Figure 9.16: Energy spent for computation by mobile devices

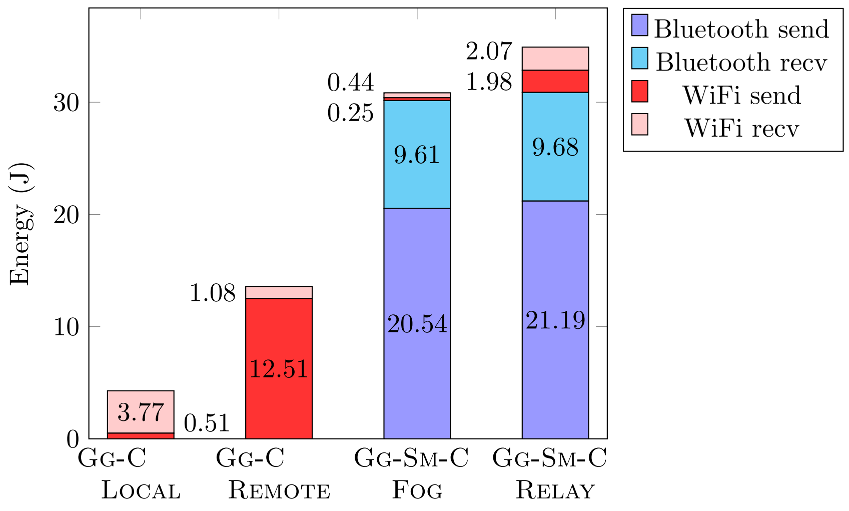


Figure 9.17: Energy spent for communication by mobile devices

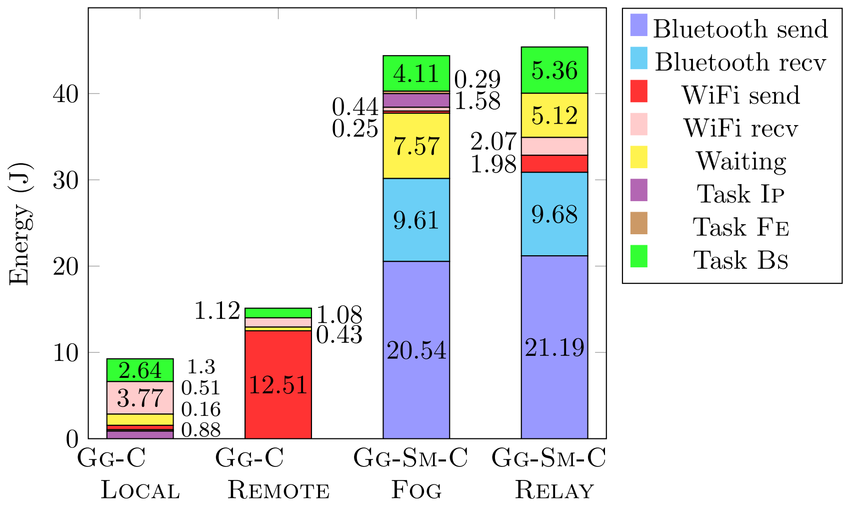


Figure 9.18: Energy spent for communication by mobile devices

sumption is almost identical from the point of view of the Google Glass, the communication dominates and makes the two cases different. Indeed, direct communications between the Google Glass and the cloud lead to faster execution time and lower power consumption. This trend becomes evident comparing the couples of scenarios (GG-C LOCAL and GG-C REMOTE) and (GG-SM-C FOG and GG-SM-C RELAY): the first couple presents an energy consumption profile which is lower by more than two times with respect to the second couple. As mentioned before, one of the causes for this behaviour is the type of data exchanged text or image.

Bluetooth communications appear to be the more costly than communications with WiFi (see Fig. 9.17). The reason is the long execution time required by the Google Glass to send and receive all the data. Looking at the smartphone energy consumption during the same operations, the same operations require significantly less amount of energy energy.

For the computation side, the collected data reveals that the smartphone requires more energy for processing with respect to the Google Glass. The gap between the two devices is not very big and is attributed to the different hardware the two entities are equipped with. Moreover, another reason is the different battery capacity which provide a different amount of energy.

Finally, waiting times are very demanding in terms of power, especially for the Google Glass. The wearable device experiences waiting times of different durations and in these periods the power consumption trend looks like the activity on the device is still running actively. Despite the fact any sort of external operation was removed in this phase of the execution, operations performed by the operative system appear in this unpredictable behavior.

Part IV

Energy Efficiency in Cloud-Based Mobile CrowdSensing Systems

Chapter 10

Background on Mobile CrowdSensing

This chapter introduces Mobile crowd sensing (MCS), illustrates open research challenges in the field and outlines the proposed contributions.

10.1 Mobile CrowdSensing in a Nutshell

MCS has become in the recent years an appealing paradigm for sensing and collecting data. In MCS, users contribute data gathered from sensors embedded in mobile devices, which include smartphones, tablets and in general IoT devices, which is then delivered to a collector in the cloud [5], [206] (see Fig. 10.1).

The term *mobile crowd sensing* was first introduced by Ganti et al. [5] and indicates a more general paradigm than mobile phone sensing[207], [208]. Guo et al. in [206] give a definition that clearly highlights this difference: *"MCS is a new sensing paradigm that empowers ordinary citizens to contribute data sensed or generated from their mobile devices, aggregates and fuses the data in the cloud for crowd intelligence extraction and people-centric service delivery".*

To operate efficiently, MCS systems require participation and contribution of a large number of users. Although entire communities can potentially benefit from such a contribution, singular person may be reluctant to participate, being selfish or having privacy concerns. To easy this burden, in the last years the research community has put lot of effort in developing proper incentive mechanisms [209], [210] and in investigating privacy issues [211], [212].

The ubiquitous diffusion of mobile devices and the rich set of built-in sensors they are equipped with are two main key enablers leading to the success of MCS paradigm. Accelerometer, GPS, camera and microphone are only a representative set of sensors equipped in mobile devices. Although

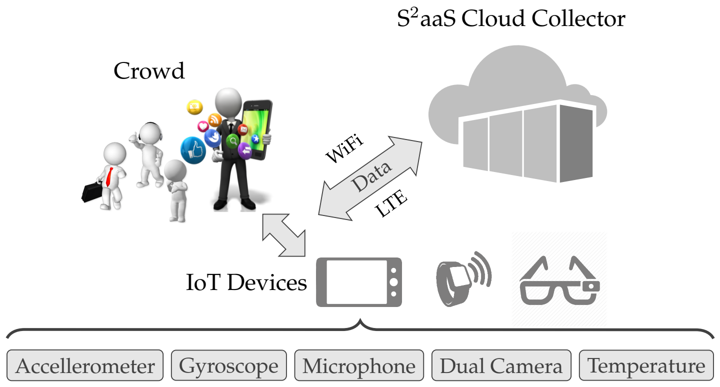


Figure 10.1: Cloud-based MCS system components

MCS is an emerging paradigm, a number of MCS applications relying on mobile device sensors have already been developed in different scenarios, including health-care, environmental monitoring, public safety and intelligent transportation systems such as traffic monitoring and management [207], [208], [213]. All these applications suit urban scenarios very well. MCS is also a key solution for building smart cities, which aim at using ICT solutions to improve management of everyday life of their citizens [3], [214]. Google, for example, uses crowd-sourced information about smartphone locations to offer real-time view of congested traffic roads and has recently released a new application, called Science Journal, which allows to gather and visualize data from smartphones' sensors [215]. Pokemon Go, developed by Niantic in collaboration with Google, is an application which may act as MCS framework to collecting data like user location and movements. The application is extremely popular. In the USA, 10 M users downloaded the application in two weeks. The application has the potential to become the first large-scale MCS framework daily used by millions of people and can bring side-effect advantages. For example, in Italy, users while playing have filmed and reported a robbery that lead to the arrest of the thief [1].

In MCS systems, data collection can be either *opportunistic* or *participatory* [207], [208] (see Fig. 10.2). The user involvement in *opportunistic* sensing systems is minimal or none, which means that the decisions to perform sensing and report data are application- or device-driven. On the contrary, in *participatory* sensing systems rely on active user engagement in the sensing process. For example, a user spontaneously contribute to the system without having received a specific task. Participatory sensing is tailored to crowd sensing architectures with a "central intelligence" responsible to assign tasks to the users, e.g., to ask one user to record a video in a given

[1]`http://www.intelligonews.it/articoli/29-luglio-2016/47061/ragazzini-a-caccia-di-pokemon-a-roma-filmano-ladro-arrestato`

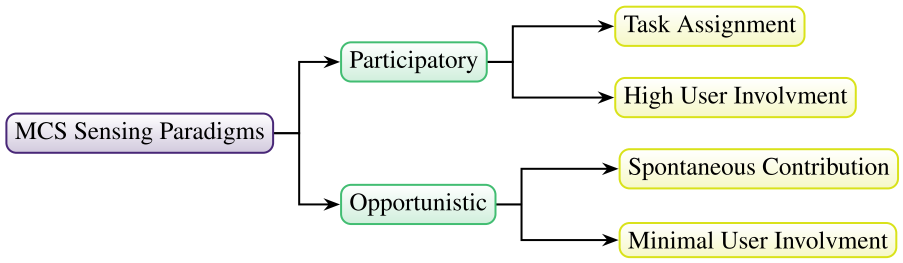


Figure 10.2: Taxonomy of MCS Sensing Paradigms

area at a given time. Unlike opportunistic sensing systems, the participatory paradigm imposes a higher cost on the user in terms of cooperation effort. Having devices or applications responsible for sensing lowers the burden for user participation and makes opportunistic sensing ideal for distributed solutions. The following chapters present solutions for participatory MCS systems.

To this date, research in MCS is still in its infancy, with many of the core paradigms undefined or unclear. To illustrate, for example, there is no general consensus on the term "opportunistic sensing". According to Ganti et al. [5] "opportunistic sensing" is defined as "*On the other hand, opportunistic sensing is where the sensing is more autonomous and user involvement is minimal (e.g. continuous location sampling)*". However, Khan et al. [208] state that opportunistic sensing requires no user involvement at all, since the decisions to perform sensing is a prerogative of the device itself. Finally, Han et al. [216] enlarge previous vision of opportunistic sensing in the context of single user involvement and they describe opportunistic sensing as a paradigm enabling cooperation among smartphones. Typically, both terms "opportunistic" and "participatory" sensing remain consider under the common umbrella of MCS [5], [208], [217]. In other cases, both "mobile crowd sensing" and "participatory sensing" are used interchangeably [218]. Other times, both "mobile crowd sensing", "participatory sensing" and "opportunistic sensing" are synonyms [219].

10.2 Research Challenges and Open Questions

Although being recent, MCS is already a largely explored research field. Nevertheless, to be widely adopted in real world, several challenges and open questions have to be addressed.

In the recent years, the research community has put lot of effort in developing incentive mechanisms to foster user participation in MCS [210], [220] and in investigating privacy issues [211], [212]. After the advent of Pokemon Go, an interesting research challenge in this area would be

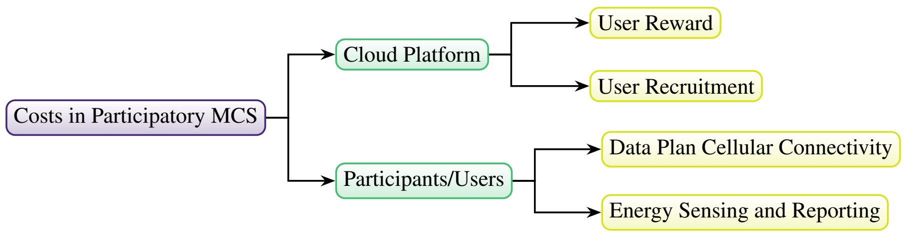

Costs in Participatory MCS
Cloud Platform
Participants/Users
User Reward
User Recruitment
Data Plan Cellular Connectivity
Energy Sensing and Reporting

Figure 10.3: Taxonomy of Costs in Participatory MCS

exploring the privacy concerns of both actors, players and governments. In China, for example, the authorities are reluctant to make public available the application bearing in mind that players while walking may reveal military bases. While privacy concerns is certainly a limiting factor, the cost of sensing often defines the level of user participation. On one hand, mobile devices are battery constrained and it is important to use all available energy wisely, i.e., refrain from unnecessary sensing and data reporting operations. While on the other hand, reporting collected samples using wireless communication technologies, such as 3G/4G, WiFi or Bluetooth, affects battery lifetime [159], [160] and has associated data plan costs.

User recruitment is one of the key challenges in participatory MCS systems. In urban environments, the high number of potential contributors calls for the design of efficient recruitment policies. Proper policies allow selection of users able to fulfill sensing tasks with high accuracy and to minimize the system costs. From the standpoint of the central platform, which organizes and dispatches tasks, the efficiency of a data acquisition framework is defined in terms of the revenues, obtained through Sensing as a Service (S^2aaS) business models [221], and the costs sustained. Costs in participatory MCS have a double nature (see Fig. 10.3). The central platform sustains monetary costs to recruit and reward users for their contribution. Users as well sustain costs while contributing data, i.e., the energy spent from the batteries for sensing and reporting data and, eventually, the data subscription plan if the cellular connectivity is utilized for reporting. As in cloud-based S^2aaS the mobile devices are the most hungry components in the ecosystem [222], cost-effective solutions in data acquisition not only allows to minimize the energy expenditure, but are also are a powerful incentive to stimulate user participation [223].

Other important challenges to be tackled in MCS are data accuracy. On one hand, several research work assess quality of generated data by evaluating trustworthiness of the users, for example through reputation-based mechanism [213], [224]–[226]. On the other hand, several techniques can be employed to estimate data quality and accuracy for example statistical-based techniques like *leave-one-out* and *bootstrap* [227], [228]. Another method to improve quality of data consists in periodic calibration user-devices with the help of fixed infrastructure such as weather stations [229].

10.3 Contributions

Given the challenges presented in Section 10.2, the following chapters are devoted to illustrate the solutions proposed:

- Novel performance metrics to assess efficiency of recruitment policies, accuracy of task completion and distribution of sample generation (Section 11.1).
- A new simulator, CrowdSenSim, able to assess in large scale urban environments costs for the users of crowdsensing activities and amount of generated data (Section 11.2).
- A novel user recruitment policy for participatory MCS systems, based on three selection criteria that define user eligibility. The user recruitment policy is the key component of a new data collection framework (Chapter 12). Performance of the framework is assessed with CrowdSenSim and the efficiency of the user recruitment policy through the new metrics.

Chapter 11

Assessing Performance of Mobile CrowdSensing Systems

Measuring and assessing performance is essential for any system. It allows to minimize costs while maximizing the efficiency of the process, to identify best practice and solutions to expand. This chapter presents a methodology to assess performance of MCS systems providing an overview of existing and novel proposed metrics. Simulations are a common tool to assess performance in a controlled environment. Because of the complex nature of MCS systems, simulations and not real testbed are the candidate solution for performance evaluation. In this context, the chapter reviews simulators suitable for MCS systems and presents CrowdSenSim, the first simulator able to assess performance of MCS systems in realistic large scale urban environments.

11.1 Performance Metrics for Mobile CrowdSensing Systems

Metrics are essential to measure properties, quantify and assess performance of systems. This section first reviews performance metrics in the literature and discuss possible directions to design new metrics (Section 11.1.1). Finally, the section illustrates new metrics designed to assess the efficiency of user recruitment, the accuracy of task completion and distribution of sample generation (Section 11.1.2).

11.1.1 Background on Existing Performance Metrics and Open Challenges

Existing Metrics

In the recent years, a number of metrics have proposed metrics to address mainly two challenges in MCS, namely quality of data and coverage.

In wireless sensor networks such as MCS systems, mobile users and devices are in charge of collecting data of the surroundings while moving. Assessing quality of information because the contact time between users and phenomena can be short, mobility follows random and uncontrollable trajectories and the accuracy of sensors can vary. As a result, the information collected depends on multiple factors such as location, sensing conditions and modalities and ambient noise levels. Several research works propose metrics to evaluate the quality of information [230], [231] and service [232]. In [230] the authors propose to assess quality of information according to three main features: sampling precision, data accuracy and granularity. This methodology is general and applicable to urban scenarios. Tham et al. [231] propose IQ, Information Quality of Contributors, which relates the quality of information to the presence or absence of a given participants in a location of a sensing task. Di Stefano et al. [232] have proposed a non-markovian stochastic Petri net formulation to evaluate the performance of MCS systems assessing Quality of Service (QoS) metrics.

Assessing coverage is very important [233], [234]. Indeed, it allows to quantify zones with high/low density of data generation and take proper countermeasures to respectively decrease or increase the generation process. In particular the most challenging problem is to deal with low amounts of data, which do not allow to proper characterize and map a phenomena. A common method to cope with the low amount of data problem, is data correlation, i.e., applying machine learning techniques to predict the value of sensor readings according to the spatiotemporal patterns of generated data. In [233], the authors propose the *coverage quality*, which is a metric defining data coverage in terms of the number of sensor readings obtained per sensing cycle in a given area. In urban environments, the spatiotemporal location of the users is dynamic, i.e., it changes over time and geographical position and makes the quality of sensed data dependent on time and location. In the context environmental monitoring with image type of data, in [234] the authors propose *urban resolution r*, a new metric to measure the quality of sensing images and help recovery of blank zones through spatiotemporal data correlation.

Open Challenges

To this date, it exists a number of possible directions to explore for the design metrics to form a complete and coherent framework of performance evaluation in MCS.

So far, no metrics were proposed to evaluate efficiency of user recruitment policies. As explained in Section 10.2, proper selection of users is fundamental in participatory MCS systems to maximize utility in data collection, task fulfillment and minimize the expenditure. Performance metrics in this area should assess efficiency of policies in terms of costs, number of users contacted and number of users recruited per sensing task and according to spatiotemporal windows.

Opportunistic MCS systems are largely unexplored at the time of this writing. Novel metrics have to be designed in this area. For example, the evaluation of the amount of data generated per user is essential to design proper rewarding mechanisms and determine the accuracy in capturing phenomena.

The following section addresses some of the challenges illustrated.

11.1.2 New Performance Metrics

User Recruitment Effectiveness (URE)

The effectiveness of any recruitment policy can be defined in terms of the number of contacted users and the ones that are actually recruited. To quantify such effectiveness, a novel metric called User Recruitment Effectiveness (URE) is proposed:

$$\text{URE} = \mathbb{E}\left[\frac{N^r}{N^c}\right], \tag{11.1}$$

where $\mathbb{E}[N^c]$ and $\mathbb{E}[N^r]$ correspond to the average number of contacted and recruited users respectively. The URE metric can assume real values in the range $[0, 1]$. Values of URE close to 1 indicate efficient policies. More precisely, URE = 1 corresponds to have all the contacted users actually recruited.

Global Task Accuracy (GTA)

Measuring task accuracy is important. Obviously, task accuracy depends on time distribution of the contribution provided by N users. Let us consider the case for all the N users deliver sensed data during the first timeslot $t = 1$ and remain idle for the remaining $T - 1$ timeslots. Consequently, the task is accomplished with poor accuracy. Conversely, if N users contribute data *uniformly* along the entire period T, the task is accomplished with high accuracy. The Global Task Accuracy (GTA) metric quantifies accuracy of accomplished task as follows:

$$\text{GTA} = \frac{1}{T} \cdot \left[\sum_{t=1}^{T} x_t \cdot \frac{n_t}{N} - \sum_{t=1}^{T} q \cdot \left(\sum_{j=1}^{t} y_t\right)\right], \tag{11.2}$$

where n_t is the number of users in timeslot t contributing data and q is a penalization term, which reduces task accuracy when in a given timeslot the contribution is null. The term q is set to be inverse proportional to the number of timeslots T:

$$q = 1/T. \tag{11.3}$$

The rationale behind this choice is that having no contribution in one timeslot affects more severely short tasks than longer ones. The penalization should be more severe if during consecutive timeslots none of the users contributed data. For this reason, in (11.2) q linearly increases with the number of timeslots with no contribution. The terms x_t and y_t are boolean variables:

$$x_t = \begin{cases} 1 & \text{if } n_t > 0; \\ 0 & \text{otherwise.} \end{cases}, \quad y_t = \begin{cases} 1 & \text{if } n_t = 0; \\ 0 & \text{otherwise.} \end{cases} \tag{11.4}$$

It is worth mentioning that both URE and GTA metrics should not be computed run time, but after task completion, i.e., the duration T_i of the task i expired.

Sample Distribution (SD)

Having the knowledge on the amount of data the users can contribute is important for the applications and to determine the accuracy in mapping a phenomena. However, for drawing more precise conclusions, it is fundamental to determine also where and when the user devices generate samples. To this end, the Sample Distribution (SD) metric measures the amount of generated samples per meter and is defined as follows:

$$\text{SD} = \frac{N_s^a|_t}{\overline{\Delta}}, \tag{11.5}$$

where $\overline{\Delta}$ is the average distance between samples and $N_s^a|_t$ is the number of samples generated. The parameter $\overline{\Delta}$ is defined as follows:

$$\overline{\Delta} = \frac{\sum_{\substack{i,j \\ i \geq j}}^{n} d(i,j)}{\frac{n(n-1)}{2}}. \tag{11.6}$$

The term $d(i, j)$ is the distance (in meters) between the location where the samples i and j were generated.

The SD metric belongs to the family of metrics assessing coverage of data generation. Unlike the metrics presented in Section 11.1.1, SD is totally independent of the type of data generated.

11.2 CrowdSenSim: Mobile Crowd Sensing Simulator

For proper operation MCS systems require the contribution from a large number of participants. Sensing campaigns are often in the order of hours or days. Therefore assessing performance of MCS systems through the development of real testbeds is costly and often not feasible. Nevertheless, it exists in the literature few examples. This section reviews existing testbeds and simulators suitable for MCS, presents designs principles of CrowdSenSim and finally illustrates the architecture.

11.2.1 Background on Real Testbed and Simulators

Testbed

With the objective of studying coverage and scaling properties of MCS systems, in [235] the authors have recruited 85 people to collect data for a two months period in Seoul, Korea. Several data types were collected and among those, nearly 22 000 audio clips and 6 200 photos were included in the resulting dataset.

In [236] the authors have developed an application called TrackMaison (Track my activity in social networks) to collect data concerning usage of social networks with ultimate goal of designing a behaviometric identification framework. Behaviometric identification refers to continuous identification and authentication of IoT devices and is envisioned to co-exist and strengthen biometric authentication, which is unfeasible to provide continuously. A population of 10 individuals were providing data for 15 days from the following set of social networking applications: Facebook, Twitter, LinkedIn, WhatsApp and Skype.

SilentSense[237] is an application providing non-intrusive user identification mechanism to identify whether the person using the device is the owner, a guest or an attacker. To assess the effectiveness of the application a number of 100 volunteers were recruited.

Simulators

Tanas et al. propose to exploit Network Simulator 3 (NS-3) for crowdsensing simulations [238]. The objective is to assess the performance of a crowdsensing network taking into account the mobility properties of the nodes together with the wireless interface in ad-hoc network mode. Furthermore, the authors present a case study about how participants could report incidents in the public rail transport. NS-3 provides highly accurate estimations of network properties. However, having detailed information on communication properties comes with the cost of losing scalability. First, it is not possible to simulate tens of thousands of users contributing data. Second, the granularity of the duration of NS-3 simulations is typically in the order of minutes. Indeed, the objective is to capture specific behaviors such as the changes of the TCP congestion window. However, the duration of real sensing campaigns is typically in the order of hours or days.

In [239], Farkas and Lendák present a simulation environment developed to investigate performance of crowdsensing applications in an urban parking scenario. Although the application domain is only parking-based, the authors claim that the proposes solution can be applied to other crowdsensing scenarios. However, the scenario considers only drivers as type of users and users travel from one parking spot to another. The authors consider humans as sensors that trigger parking events. However, to be widely applicable, a crowdsensing simulator has to take into account data generated from mobile and IoT devices' sensors carried by human individuals.

Mehdi et al. propose CupCarbon [240], which is a discrete-event wireless sensor network (WSN) simulator for IoT and smart cities. One of the major strengths is the possibility to model and simulate WSN on realistic urban environments through OpenStreetMap. To set up the simulation, the users have to deploy on the map the various sensors and the nodes such as mobile users, gas and media sensors and base stations. The approach is not intended for crowdsensing scenarios with thousands of users.

11.2.2 Design Principles and Key Performance Indicators

The section outlines the design principles and the main key performance indicators were used for the design of CrowdSenSim.

Design Principles

To design a novel MCS simulator, the main aspect to consider are the *scalability*, the implementation in a *realistic urban environment*, the *user mobility* and *communications*.

Scalability: For proper operation, MCS systems require a large number of participants. Hence MCS simulators should be designed to host in the order of tens of thousands participants moving in wide geographical space. Each user can potentially own several IoT and mobile devices, each of them is a potential data contributor. Time dimension is also important. The duration of a sensing campaign ranges from hours to days and a simulator should address this challenge efficiently. For instance, let us consider 10 000 users producing data with an average of only 30 minutes per day. Each user delivers 12 bits long samples of the accelerometer working at 50 Hz frequency. The total amount of generated data is 1.35 GB. Considering prolonged duration of user contribution and additional sensors would considerably augment the figure.

Realistic urban environment: Similarly to CupCarbon, MCS simulators should rely on realistic urban environments for several reasons. First, exploiting realistic layouts of urban environments makes the simulator flexible and easy to be adopted in any city. Second, it allows to perform analysis providing meaningful insights to the municipality to understand the feasibility and the potentiality of the proposed MCS solution. Simulations over a grid or a square area as abstraction levels lower the complexity, but do not allow to take into account important features such as movements in real streets and physical obstacles such as buildings.

User mobility: Human mobility is defined as sequences of spatiotemporal user movements. Understanding human mobility in an urban environments is crucial to design mobility patterns that meet social behaviors and scale to the requirements of modern smart cities [241].

Communication technologies: IoT and mobile devices are equipped with several communication technologies, including 3G/LTE, WiFi and Bluetooth. Each communication technology drains battery of the devices differently and can have associated costs (e.g., users have a limited monthly plan).

Key Performance Indicators

This subsection details important types of KPI that CrowdSenSim is designed to assess, including *data generation* and *cost evaluation*.

Data Generation: Sensors work with different sampling frequency and sample size. After data collection, mobile devices deliver samples to a central collector using different communication technologies. In S^2aaS models, revenues are proportional to the amount of generated data. Therefore, it is important to assess KPI concerning data generation such as the amount of data collected in a given time window, or per area.

Costs: Nowadays energy consumption is one of the most important and challenging issues. In MCS, energy is consumed to perform sensing and reporting. The energy spent per sensing is typically proportional to the sampling frequency of the sensor. The energy spent for communications depends on the technology used. Rapid battery drain due to MCS-related applications can lower user participation.

11.2.3 Architecture

Fig. 11.1 describes the architecture of CrowdSenSim, which is build in modular fashion and it is composed of the following modules:

- City Layout module.
- User Mobility module;

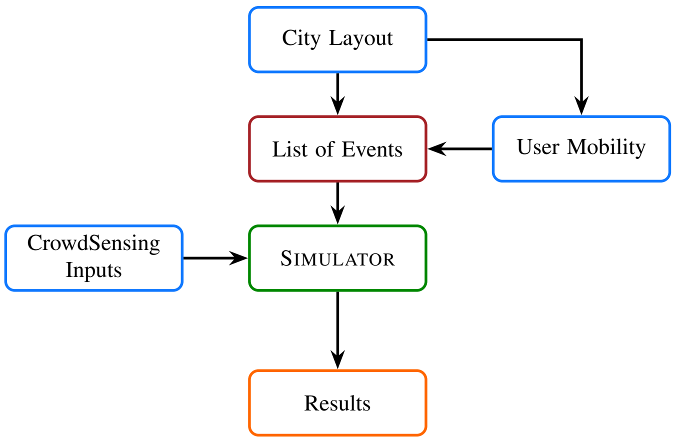


Figure 11.1: Architecture of CrowdSenSim

- Crowdsensing input parameters module.

The first two modules are in charge of defining the list of events, which is used to perform the simulations along with the inputs provided by the Crowdsensing input parameters module. It is worth mentioning that all the modules are independent by design. The objective is to guarantee to future developers the possibility to implement, for example, other mobility models without compromising the overall functionality of the simulator. The remainder of this section describes the functionalities of each module in detail.

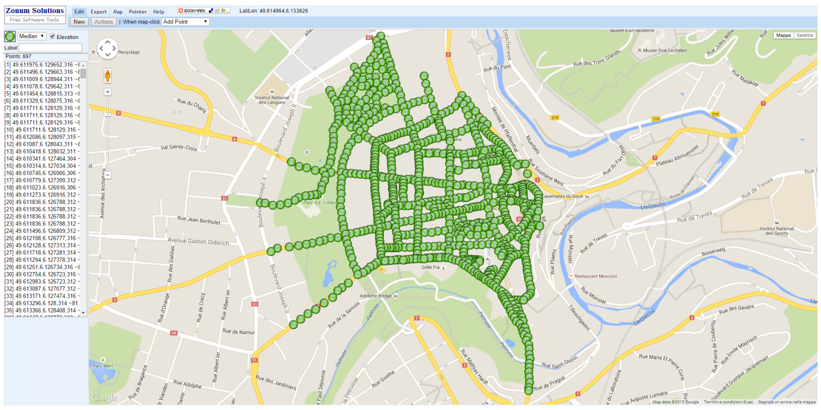
City Layout

In CrowdSenSim, the users move in a real city setting, namely the City of Luxembourg. It covers an area of 1.11 km^2 and is the home of many national and international institutional buildings. The information about the streets of the city is obtained from a crowdsourced application which provides free access to street-level maps[1] in form of a set of coordinates C containing <*latitude, longitude, altitude*>, see Fig. 11.2. The set of coordinates composing the street layout is one input of the simulator and is used as basis for the user mobility model and the engine in charge of generating the list of events.

User Mobility

The participants move along the streets of the city and their original location is randomly assigned from the set of coordinates C. The initial time of user deployment T_{in} is also randomly assigned: by default it is uniformly

[1]DigiPoint: `http://www.zonums.com/gmaps/digipoint.php`



```
Points: 897
[1] 49.611975,6.129652,316 ~8
[2] 49.611496,6.129663,316 ~8
[3] 49.611009,6.128944,311 ~8
[4] 49.611078,6.129642,311 ~8
[5] 49.611454,6.128815,313 ~8
[6] 49.611329,6.128075,316 ~8
[7] 49.611711,6.128129,316 ~8
[8] 49.611711,6.128129,316 ~8
[9] 49.611711,6.128129,316 ~8
[10] 49.611711,6.128129,316 ~
[11] 49.612086,6.128097,315 ~
[12] 49.61087,6.128043,311 ~8
[13] 49.610418,6.128032,311 ~
[14] 49.610341,6.127464,304 ~
[15] 49.610314,6.127034,304 ~
[16] 49.610745,6.126906,306 ~
[17] 49.610779,6.127399,312 ~
[18] 49.611023,6.126916,306 ~
[19] 49.611273,6.126916,312 ~
[20] 49.611836,6.126788,312 ~
[21] 49.611836,6.126788,312 ~
[22] 49.611836,6.126788,312 ~
[23] 49.611836,6.126788,312 ~
[24] 49.611496,6.126809,312 ~
[25] 49.612198,6.126777,316 ~
[26] 49.612128,6.127313,314 ~
[27] 49.611718,6.127281,314 ~
[28] 49.611294,6.127378,314 ~
[29] 49.61251,6.126734,316 ~8
[30] 49.612754,6.126723,316 ~
[31] 49.612983,6.126723,312 ~
[32] 49.613087,6.127077,312 ~
[33] 49.613171,6.127474,316 ~
[34] 49.613296,6.128,314 ~81
[35] 49.613366,6.128408,314 ~
```

Figure 11.2: Street-level information of Luxembourg

distributed between 8:00 AM and 1:30 PM. Each participant decides the amount of time T_{tr} s/he travels and, by default, it is a value uniformly distributed between 10 and 30 minutes. The mobility model implemented is a random mobility model where user movements are constrained by the physical layout of the streets. More precisely, each participant *jumps* from a given coordinate c_1 to another coordinate c_2 in C. It should be noted that c_2 is selected as follows: by default, it is in the same street of c_1, otherwise it belongs to another street in case c_1 is in proximity of intersections. Given that the participant is in c_1 at time t_1 and s/he moves with an average speed uniformly distributed between 1 and 1.5 m/s, it is possible to compute t_2 after having determined the spatial distance between c_1 and c_2. In CrowdSenSim, the arrival of a user in a given location at a given time is an *event*.

Crowdsensing input parameters

Each MCS system has unique features to take into account. For example, the notion of tasks is essential in participatory MCS systems, but not in opportunistic ones. Considering participatory systems, this module is responsible to define parameters such as number of tasks and task coverage radius.

Simulations and Results

Receiving in input the list of events ordered by time and the specific parameters of crowdsensing system, the simulation engine runs to provide the results. For that, a file is created which contains information on user, location, number of sensing events, energy consumption, number of users assigned to a task. The file is used in input of the engine responsible to generate results. Through scripts, simulations are run a number of times to generate statistical results with 95% confidence interval.

Chapter 12

Energy Efficient Data Collection: Model Design and Implementation

The chapter proposes a novel framework for data acquisition in participatory MCS systems. The framework runs over a fog computing platform deployed in a distributed fashion in a urban environment. The fog facilitates the most important operations in data acquisition, such as user recruitment and task completion.

12.1 The Role of Fog Computing in CrowdSensing

Fog computing architectures are heterogeneous and consists of both devices at the edge of the network and traditional cloud data centers. In the front-end, the mobile devices are IoT devices, smartphones and cloudlets with various degree of computing, storage and networking capabilities. Cloudlets are typically local processing units such as notebook or desktops used for temporary storage and processing [242]. In addition to provision computing resources, they can also provide data aggregation functions to reduce the amount of information delivered to the cloud. Cloud data centers, in the back-end of the architecture, provide centralization of functionalities and backup.

To properly support IoT-based services and applications, including MCS, an efficient deployment of fog architectures in urban environments is essential [243]. Edge devices such as cloudlets, that are responsible for provisioning location-aware and low latency computing, have to be geographically distributed across the city. Bus stops are a natural choice for installing cloudlets for a number of reasons. First, bus stops are widespread across urban environments. Second, they are already existing and deployed infras-

Table 12.1: Classification of stop relevance. Each interval corresponds to the number of trips passing by a stop from 6:00 AM to 10:00 PM.

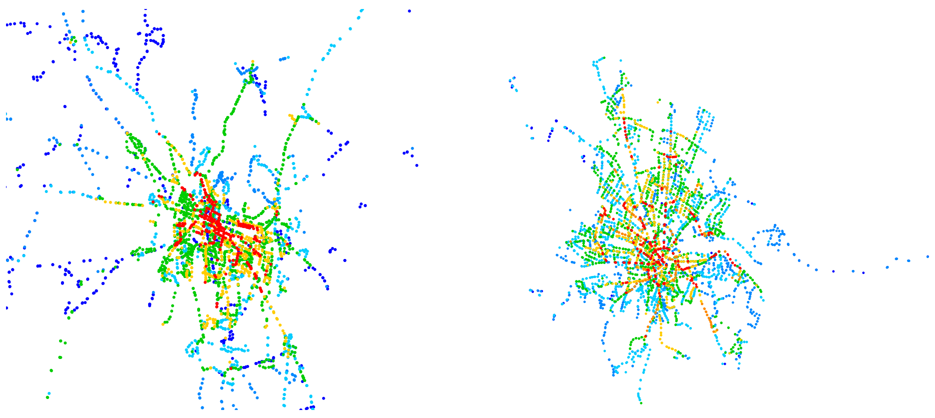
COLOR	RELEVANCE INTERVAL		
	TORINO	BUDAPEST	OTTAWA
🔵	0 - 5	0 - 25	0 - 25
🔵	5 - 10	25 - 100	25 - 50
🔵	10 - 50	100 - 250	50 - 100
🟢	50 - 100	250 - 500	100 - 250
🟡	100 - 250	500 - 750	250 - 500
🟠	250 - 500	750 - 1000	500 - 1000
🔴	>500	>1000	>1000

tructure. As a consequence, such solution would save capital expenditure costs. For example, bus stops are already connected to the city power grid therefore there is no need for investments. Third, bus-stops are expected to become *intelligent* in the near future and provide additional services such as hosting femto cells to increase cellular capacity and connectivity [244]. In cities, not all the bus stops are identical. Because of the location, some of the them are hubs, i.e., they are stops used by a huge number of passengers every day. Therefore, such stops are more *relevant* to the system than the ones used by few passengers. Obviously, stops need to be equipped with computing capacity proportionally to their relevance. To study the relevance property of bus stops, we analyze data from Google Transit Feed. Google Transit [245] is a tool that integrates information on public transportation system like stop location, routes, bus schedule and fare on Google Maps to let trip planning easier and accessible for everyone.

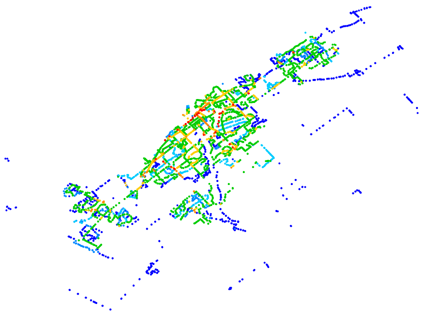
Similarly to [246], the *relevance* of stop i is defined in terms of the number of trips crossing i during a given time period. For the evaluation, the time period is set to 6:00 AM - 10:00 PM of a weekday, namely 2016-06-17. Fig. 12.1 shows relevance stop distribution in different cities, namely Torino, Budapest and Ottawa according to the classification proposed in Table 12.1. In all the cities, the number of stops with high relevance is low. Relevant stops are typically concentrated in the city center or in proximity of popular public parking facilities or train stations, which are expected to serve a high number of users along the day.

12.2 The Data Collection Framework

A data acquisition framework defines the set of steps necessary to produce and deliver the information from the participants to the organizer/collector. Fig. 12.2 shows the architecture considered and illustrates the functions each



(a) Torino **(b)** Budapest



(c) Ottawa

Figure 12.1: Analysis of stop relevance distribution in different cities

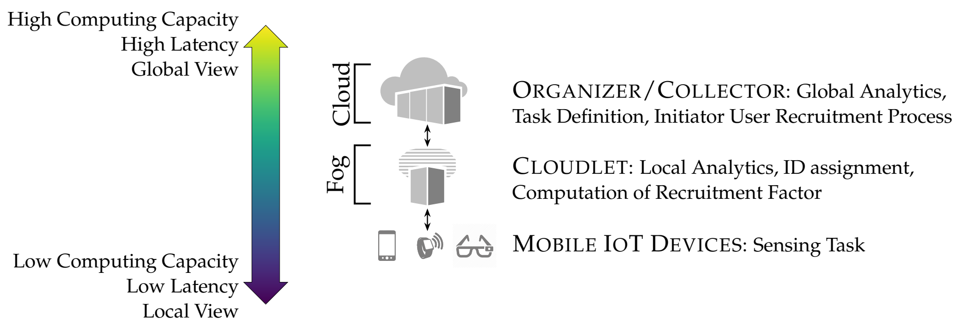

High Computing Capacity
High Latency
Global View
Low Computing Capacity
Low Latency
Local View
Cloud
Fog
ORGANIZER/COLLECTOR: Global Analytics, Task Definition, Initiator User Recruitment Process
CLOUDLET: Local Analytics, ID assignment, Computation of Recruitment Factor
MOBILE IOT DEVICES: Sensing Task

Figure 12.2: Architecture of the proposed fog-based data acquisition framework

actor carries out.

The *organizer* of the crowdsensing campaign C is interested in acquiring data from given *points of interest* in the city, also called the *sensing terrain*. The organizer, located in the cloud, is in charge of analyzing data after it has been collected and make it available to S^{2}aaS applications. The organizer also defines the set of sensing tasks $\mathcal{W} = \{w_1, w_2, \dots, w_W\}$ of C. Each task w_i is described in terms of its location L_i and time duration T_i, i.e., $w_i(L_i, T_i)$. The location L consists of latitude and longitude parameters. The time duration T is given in timeslots. As a result, the duration $\mathcal{T}$ of the campaign C is as follows:

$$\mathcal{T} = \sum_{i \in \mathcal{W}} T_i. \tag{12.1}$$

In this work we exploit the computational capacity the fog provides for efficient user recruitment. However, user recruitment is not the sole application the fog platform deployed as in Section 12.1 can support. Local analytics [247], privacy preservation and evaluation of trust of data contributed [248] are a few examples of functionalities that the proposed fog platform can support. The cloudlets in the fog receive the tasks to dispatch from the organizer in the cloud. The cloudlets in proximity of the location of the sensing tasks are in charge of recruit the user using the policy proposed in Section 12.2.1. The participants to the campaign communicate periodically to cloudlets information about their sociability and remaining battery charge. Consequently, the cloudlets can determine which users are eligible to become contributors and contact them for recruitment.

After being contacted, the users can decide whether to accept the task. In positive case, users acquire the status of *recruited*, they are *assigned* to the task and can contribute data. Acceptance depends on user sociability and remaining battery of charge of user devices. Users with high values of sociability factor use social media often and are more likely to accept the task [249].

Table 12.2: Symbols list and description

Symbol	Description
C	Crowdsensing campaign
w	Task w
$\mathcal{W}$	Set of tasks | $w \in \mathcal{W}$
u	User u
$\mathcal{U}$	Set of users | $u \in \mathcal{U}$
t	Timeslot t
T_i	Duration task i
$\mathcal{T}$	Duration of the sensing campaign
L	Location of users and tasks
R_i	Recruitment factor of user i
D_i	Distance factor of user i
S_i	Sociability factor of user i
E_i	Energy factor of user i
$R_{\min}$	Minimum recruitment factor for eligibility
$D_{u,w}$	Distance (m) between user u and task w
$D_{\max}$	Maximum task coverage radius
E_s	Energy consumed for sensing
E_r	Energy consumed for data delivery (reporting)
P_{tx}	Power consumed for data delivery
A_i	Task acceptance factor of user i
N	Minimum number of users to mark a task as *accomplished*
P_i	Popularity factor of location i
$\mathbb{E}[N^c]$	Average number of contacted users
$\mathbb{E}[N^r]$	Average number of recruited users
C	Set of coordinates <*latitude, longitude, altitude*> of city layout

12.2.1 User Recruitment

User recruitment is a fundamental step in participatory data acquisition frameworks. Recruitment policies delineate the set of criteria for user eligibility in contributing to crowdsensing campaign. Contrary to traditional recruitment solutions, in this paper we define a policy able to select participants on the basis of three parameters: i) the distance between users and sensing task location, ii) user sociability, and iii) remaining battery of charge of users devices. The policy is named DSE (Distance, Sociability, Energy). Table 12.2 lists description of symbols used to define the user recruitment policy.

Let $\mathcal{U} = \{u_1, u_2, \ldots, u_U\}$ be the set of users potentially employed to accomplish the tasks. Each user u_i is described in terms of their current location, sociability factor and energy, i.e. $u_i(L_i, S_i, E_i)$. It is worthwhile mentioning that both user location and sociability factor are time dependent and S_i and E_i can assume real values in $[0, 1]$.

During each timeslot, the recruitment policy selects users with highest *recruitment factor* R from the set $\mathcal{U}$. Only users with values of $R > R_{\min}$ are taken into consideration and *contacted*. $R_{\min}$ defines the minimum recruitment factor and is set by the organizer to be identical for all the tasks in the campaign C. For each user i, the *recruitment factor* is defined as follows:

$$R_i = \alpha \cdot D_i + \beta \cdot S_i + \gamma \cdot E_i, \tag{12.2}$$

where the parameters α, β, γ are weighted coefficients defining the impact of the corresponding component, distance, sociability and energy on R. Taking into account that $\alpha + \beta + \gamma$ must equal unity, high values of α will prioritize selection of users close to the sensing task. High values of β will favor selection of highly sociable users while high values of γ will make the remaining battery charge of devices the most important component for recruitment.

The component D_i is the distance factor, which measures the distance of user u_i from the sensing task w_j with respect to a maximum coverage radius for the task $D_{\max}$.

$$D_i = D_{u_i, w_j} / D_{\max}. \tag{12.3}$$

Users located farther than $D_{\max}$ from the location of a sensing task are not considered eligible to contribute data for that task. Indeed, the closer the users are to the sensing task location, the higher the accuracy in capturing the phenomenon is. The Haversine formula can be employed to compute D_{u_i, w_j} [250].

User sociability S can be defined in terms of the amount of data users consume or the time they spend using mobile social network applications, or their combination [236]. Sociability is an essential parameter to consider for user recruitment. Users with high sociability are more active and use their devices online intensively, which makes them excellent candidates during the selection process. Moreover, they tend to visit more places and get connected to more users, which further increases their mobile social activity [251]. To assess sociability, it is necessary to determine the data usage or the total time that a user spends on a particular social network application in a single session. Once acquired, the instantaneous values are averaged by the number of sessions in a time window, e.g., an hour or a day. The actual user sociability is then determined through the Exponential Weighted Moving Average filter (EWMA) over the values obtained in each time window. This allows tuning and eventually limits the contribution of older values. It is worth mentioning that the sociability metric determined with this method is a relative metric based on a normalized value of user's sociability by the maximum sociability value in the network.

The parameter E indicates the energy, i.e., the remaining battery charge of users devices. The most widely adopted mobile OS like iOS and Android, provide APIs to obtain information on current level of battery charge of mobile devices. For crowdsensing operation, the devices consume energy to perform sensing (E_s) and reporting (E_r) operations:

$$E = E_s + E_r. \tag{12.4}$$

The energy E_s drain due to sensing is the sum over all sensors $\mathcal{K}$ involved to fulfill a task during T:

$$\sum_{k\in\mathcal{K}} \sum_{t=1}^{T} f_k \cdot \rho_k, \tag{12.5}$$

where f_k is the sampling frequency of sensor k and ρ_k a constant, different per sensor, which describes the energy cost per sample [247]. Typically, the parameter ρ can be obtained from the data sheet of the sensor.

The users exploit WiFi connectivity for data reporting and communication with the cloudlets. Most of the mobile operating systems, including Android and iOS, tend to prefer WiFi over cellular connectivity for data transmission, as it is more energy efficient [252] and users do not consume the data plan they pay to the cellular operators [253]. As a result, when both WiFi and LTE interfaces are active, transmissions take place via WiFi. The energy E_r spent during the transmission time τ_{tx} is defined as:

$$E_r = \int_0^{\tau_{tx}} P_{tx}\,\mathrm{d}t, \tag{12.6}$$

where P_{tx} is the power consumed for transmissions of WiFi packets generated at rate λ_g [193]:

$$P_{tx} = \rho_{id} + \rho_{tx} \cdot \tau_{tx} + \gamma_{xg} \cdot \lambda_g. \tag{12.7}$$

12.2.2 Task Completion

To recruit users, the campaign organizer sustains a cost. For each request sent to the users, the cost c associated to the task w is equal to 1 unit of cost. The costs have a different nature. For example, costs could be financial or expressed in terms of the bandwidth used to broadcast recruitment messages. Costs can also be social: contacting persistently a users who has refused a task in previous timeslots may diminishes the chances that s/he will accept the task. The objective of the organizer is to minimize the total cost sustained while maximizing the number of accomplished tasks. The tradeoff between the recruitment cost and the number of accomplished tasks defines the efficiency of the recruitment policy.

Users with high recruitment factor are contacted and can decide whether to accept or refuse the task. Upon acceptance, the user acquires the *recruited*

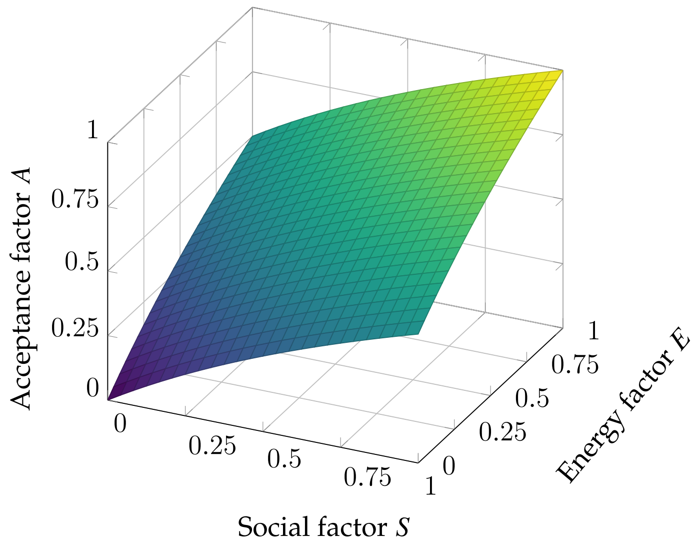


Figure 12.3: Example of acceptance factor A for $\sigma = 0.5$

status. Acceptance is based on user sociability and remaining battery charge. Users with high values of sociability and energy factors, S and E respectively, are more likely to accept the task. The acceptance factor A is computed by the user devices and it is modelled as a logarithmically increasing function:

$$A(S,E) = \sigma \cdot \log(1+S) + (1-\sigma) \cdot \log(1+E), \tag{12.8}$$

σ is a balancing coefficient that shows a relative importance between the sociability and energy factors. Fig. 12.3 shows the relation between A, S and E, which allows to perform a fine-grain comparison of the task acceptance probability of users with low versus high sociability and energy ratings. For users with high values of sociability and remaining battery charge, the acceptance factor A assumes values close to 1. Viceversa, for users with low values of sociability and remaining battery charge, a small difference between two factors S_1 and S_2, E_1 and E_2 corresponds to a considerable difference in the respective acceptance factors A_1 and A_2.

Upon acceptance, the user acquires the *recruited* status and contributes as long as s/he remains within a distance closer than D_{max}. In such a case, s/he is not contacted to contribute to the same task any longer. Viceversa, users refusing a task can be contacted again if the eligibility criteria are still met. After rejection during timeslot t, a user is contacted again during t_{next} as follows:

$$t_{\text{next}} = t + i \cdot \tau, \tag{12.9}$$

where τ is a fixed number of timeslots the systems backs off and i is the number of times the user has previously refused the same task. Consequently, the higher the number of rejection, the longer the system will wait before contacting again the user for the same task.

System-level accuracy increases if the organizer does not recruit persistently the same group of users to accomplish a task [254]. For this reason, each task w acquires the status *accomplished* if, during t, a given number N of individual users are involved and contribute by reporting data. During t_i, whenever it is not possible to recruit a sufficient number of users, the task i is marked as *failed*.

Like in social networks, some locations in cities are *hubs*, i.e., they attract a large number of individuals, whereas others do not [255]. To capture this phenomenon, each location l is assigned a popularity factor P, and P can take real values in the range $[0, 1]$. Practically, tasks associated to locations with high popularity factor should require a high number of users to successfully complete the task. In addition to the location popularity, also the time dimension plays a crucial role in defining N. Longer tasks require a higher number of users than short ones to guarantee good levels of accuracy. As a result, the number of users N_i necessary to accomplish the task i out of $\mathcal{U}$ is calculated as follows:

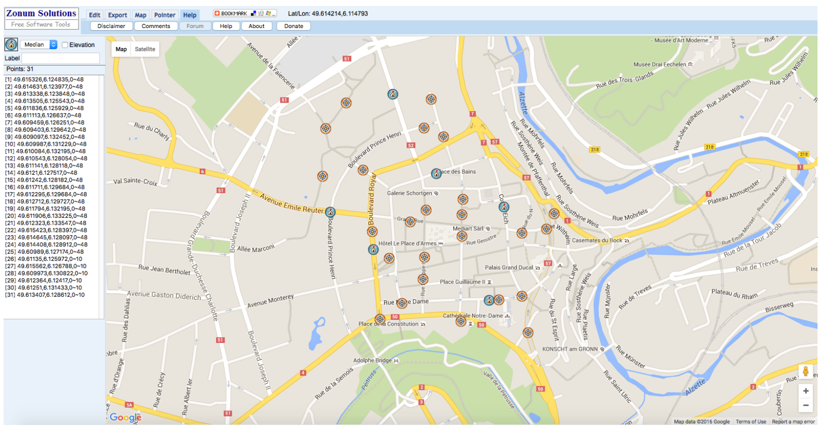
$$N_i = P_i \cdot (t_i/T) \cdot |\mathcal{U}| \,. \tag{12.10}$$

12.3 Performance Evaluation

To evaluate and assess efficiency of the data acquisition framework, CrowdSenSim was used. The participants move along the streets of the city and their original location is randomly assigned from the set of coordinates C. The number of participants ranges from 2 000 to 10 000, which corresponds to nearly one tenth of the population of Luxembourg (107 340 inhabitants as of late 2014). For simplicity, the start time of the walk is uniformly distributed between 8:00 AM and 1:30 PM. Each participant has only one mobile device, whose initial remaining charge of the battery is uniformly distributed between 0.5 and 0.9. The devices are equipped with accelerometer, temperature and pressure sensors, and transmit information using WiFi. As sensing equipment, the devices exploit real sensors implemented in current smartphones and tablets. Specifically, we select the FXOS8700CQ 3axis linear accelerometer from Freescale Semiconductor [256] and the BMP280 from Bosch [257], which is a digital pressure and temperature sensor. Table 12.3 describes the parameters used for sensing equipment. Equation (12.7) describes WiFi power consumption the devices spend for communication. Table 8.2 presents the detailed information on communication and the parameters. Users walk for a period of time that is uniformly distributed between 10 and 30 minutes with an average speed uniformly distributed between 1 and 1.5 m/s. The participants push data to the collector while walking. Once the period of

Table 12.3: Sensing Equipment Parameters

SENSOR	PARAMETER	VALUE	UNIT
Accelerometer	Sample rate	50	Hz
	Sample size	12	Bits
	Current	35	µA
Temperature	Sample rate	182	Hz
	Sample size	16	Bits
	Current	182	µA
Pressure	Sample rate	157	Hz
	Sample size	16	Bits
	Current	423.9	µA



```
Points: 31
[1] 49.615326,6.124835,0~48
[2] 49.614631,6.123977,0~48
[3] 49.613338,6.123848,0~48
[4] 49.613505,6.125543,0~48
[5] 49.611836,6.125929,0~48
[6] 49.611113,6.126637,0~48
[7] 49.609459,6.126251,0~48
[8] 49.609403,6.129642,0~48
[9] 49.609097,6.132452,0~48
[10] 49.609987,6.131229,0~48
[11] 49.610084,6.132195,0~48
[12] 49.610543,6.128054,0~48
[13] 49.611141,6.128118,0~48
[14] 49.6121,6.127517,0~48
[15] 49.61242,6.128182,0~48
[16] 49.611711,6.129684,0~48
[17] 49.612295,6.129684,0~48
[18] 49.612712,6.129727,0~48
[19] 49.611794,6.132195,0~48
[20] 49.611906,6.133225,0~48
[21] 49.612323,6.133547,0~48
[22] 49.615423,6.128397,0~48
[23] 49.614645,6.128097,0~48
[24] 49.614408,6.128912,0~48
[25] 49.60989,6.127174,0~48
[26] 49.61135,6.125972,0~10
[27] 49.615562,6.126788,0~10
[28] 49.609973,6.130822,0~10
[29] 49.612364,6.12417,0~10
[30] 49.61251,6.131433,0~10
[31] 49.613407,6.128612,0~10
```

Figure 12.4: Location of sensing tasks ● and cloudlets ●

walking ends, they stop moving and contributing. As a consequence, users can contribute for only a small portion of the day, which allows us to study the system performance under a relatively worst case scenario. Each user has an associated sociability profile uniformly distributed between 0 and 1.

A set of 6 cloudlets dispatching 25 tasks is deployed in different locations of the city, see Fig. 12.4 for the details. The starting time of each task is distributed uniformly in the time period 8:00 AM - 2:00 PM. Table 12.4 lists the details on the simulation settings. For simplicity, each task lasts 40 timeslots and each timeslot corresponds to 1 minute. In the first set of experiments, the popularity factor P of each location, the minimum recruitment factor R_{min} and the maximum task coverage radius D_{max} are fixed and set equal to 0.2, 0.55 and 55 m respectively.

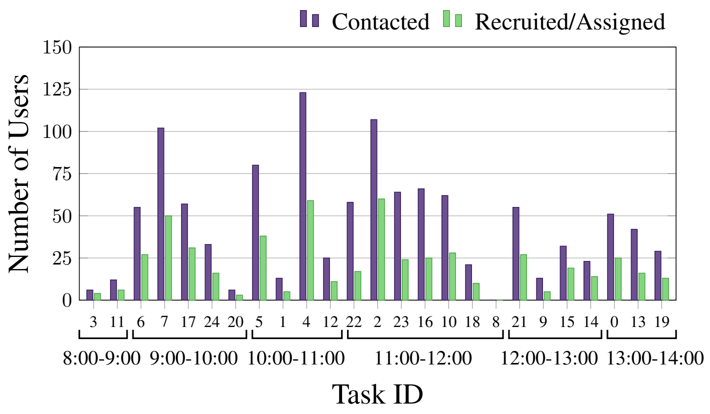
Table 12.4: Simulation settings

PARAMETER	VALUE
Number of users	[1 000 - 10 000]
Overall evaluation period	8:00 AM - 2:00 PM
Time of travel per user	Uniformly distributed in $[10, 30]$ min
Average user velocity	Uniformly distributed in $[1, 1.5]$ m/s
Initial remaining charge of the battery	Uniformly distributed in $[0.5, 0.9]$
Timeslot duration	1 minute
Task duration	40 timeslots
Number of tasks	25
Number of cloudlets	6
Popularity factor P	0.2

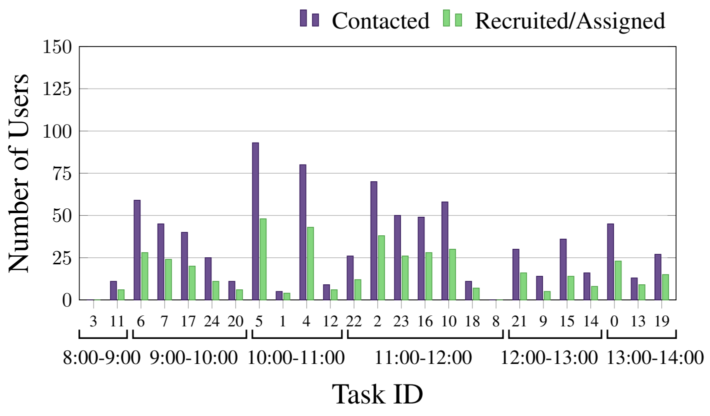
Performance of the DSE Policy

Having fixed the parameters of the recruitment policy $\alpha = \beta = \gamma = 0.33$ and $\sigma = 0.5$, Fig. 12.5 shows the number of contacted and recruited or assigned users per task. Tasks are grouped according to the initial time of deployment The number of contacted users corresponds to the cost the system sustains for recruitment. For this experiment we compare the performance of the DSE policy with a recruitment policy where the distance is the only criterion for user eligibility. The second policy is called Distance-Based policy (DB) and requires to set $\alpha = 1$, $\beta = \gamma = 0$. The DSE policy outperforms the DB policy in terms of the number of users recruited. The average number of users recruited per task is 21.23 and 17.08 for DSE and DB respectively, which corresponds to an increase of 19.55 %. Moreover, when the DB policy fails to contact users like in task # 3 or contacts very few users like in task # 1, the DSE policy makes a significant difference. Consequently, considering user sociability and remaining battery of charge of the devices in addition to task spatial coverage for recruitment is an effective solution. Although being more efficient in recruiting users, the DSE policy is more costly than the DB policy. The average number of contacted users is 45.4 while for the DB policy is 32.92, which corresponds to an increase of costs of around 27 %.

To better understand the number of completed tasks, Fig. 12.6 details the number of unique users assigned to each task. The gray line plots N, the minimum number of users necessary to denote a task as accomplished. N is computed by (12.10) and is equal for all the tasks as the location popularity and the task duration have been fixed. Consequently, partial relaxation of any of the constraints on task completion would increase the number of accomplished tasks. As it is possible to see, DSE accomplishes 10 tasks out of 25 and one more is close to completion. On the other hand, DB accomplishes



(a) DSE Policy



(b) DB Policy

Figure 12.5: Number of contacted versus recruited users. Distance-based (DB) policy is used as baseline for comparison.

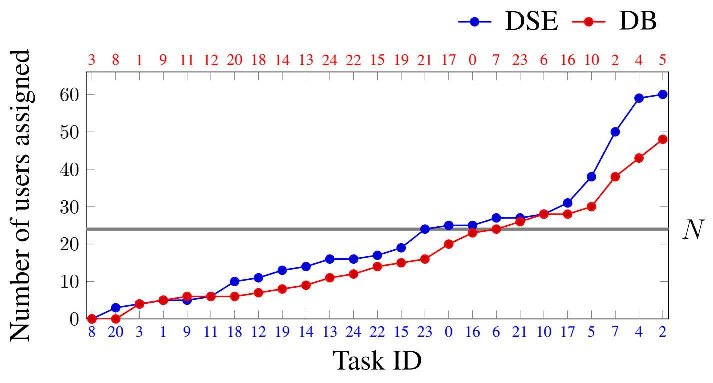


Figure 12.6: Number of assigned users per task

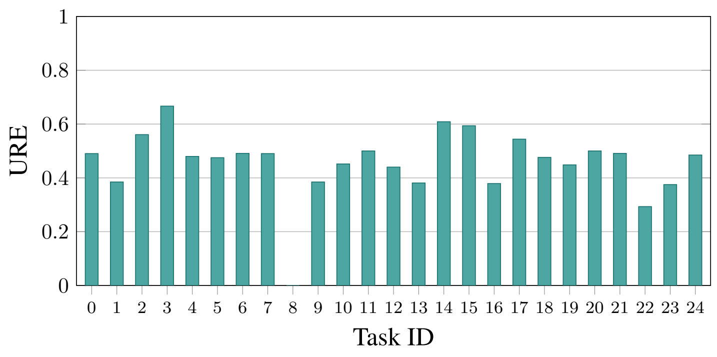


Figure 12.7: URE metric per task

7 tasks and two are close to completion. Only the campaign organizer can compare the tradeoff between cost increase and return, and pursue proper measures, e.g., to reduce the cost of user recruitment.

The next set of experiments aims at evaluating the efficiency of recruitment and accuracy of task completion with URE and GTA metrics. Fig. 12.7 shows the recruitment efficiency of the campaign, which is 0.46 and it is computed as an average of the URE values of each task. As a consequence, under the current settings, the systems contacts many users that refuse to contribute to the campaign. It is worth mentioning that for accomplished tasks such as task # 2 (see Fig. 12.6), the values of URE metric are always equal to or higher than 0.5. Therefore, for accomplished tasks, the recruitment process is more efficient as at leas half of the contacted users are actually recruited. The system achieves the highest efficiency in recruitment for task # 3 although the number of recruited users is only 4 and the task is not successfully accomplished.

Fig. 12.8 plots values of GTA metric per task. In general, the tasks are not carried out with very high accuracy. The main reason is that only few users out of N contribute to the task. The second most important reason is that not all the N users remain for the entire duration of the task under the maximum task coverage radius $D_{\max}$. For example, task # 3 is carried out with very low accuracy because few users contribute for a very short duration. As a result the penalization component in (11.2) diminishes considerably the achieved accuracy.

Analysis of the parameters of the model

Having evaluated the performance of the DSE policy, in this section we study the impact of all main parameters such as the minimum recruitment factor $R_{\min}$ and the maximum task coverage radius $D_{\max}$. We also evaluate the influence of the number of users presents in the system as well as the control parameters α, β, γ and σ. For the evaluation, we exploit as common

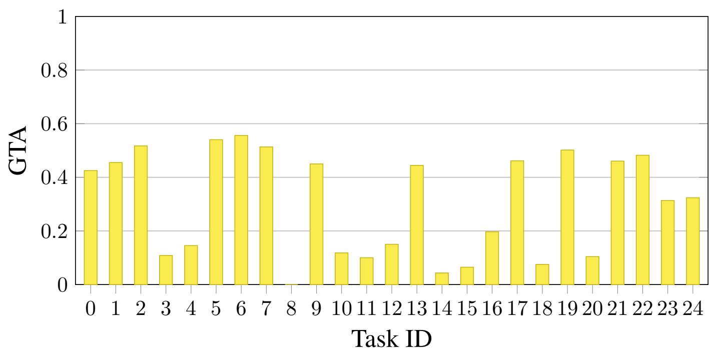


Figure 12.8: GTA metric per task

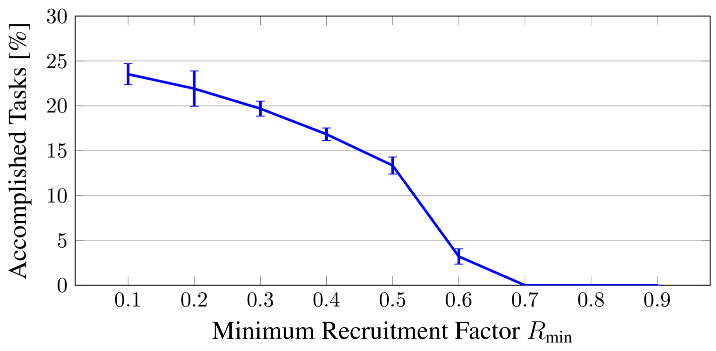


Figure 12.9: Average number of successfully accomplished tasks with increasing values of minimum recruitment factor R_{min}

performance metric the average, over 100 runs, of the number of successfully accomplished tasks. The results are expressed in percentage and the bars indicates the 95% confidence interval.

Fig. 12.9 evaluates the impact of the minimum recruitment factor R_{min}. Proper tuning of this parameter is important: high values of R_{min} make the user selection strict and only few users will be eligible for selection. On the other hand, low values of R_{min} relax the conditions for eligibility allowing the system to contact more users. The plot confirms the model: for values of $R_{\text{min}} < 0.5$, the percentage of accomplished tasks is higher than 10 %. Indeed, contacting more users it increases the chances of having N users assigned to a task, which is the minimum number of individual users necessary to denote the task as successfully accomplished.

Having fixed $R_{\text{min}} = 0.3$ from this point on, Fig. 12.10 evaluates the impact of the maximum task coverage radius D_{max}. The higher the values D_{max} assume, the larger is the area users can be contacted. The plot highlights this property and it is interesting to note a linear increase of the percentage of accomplished tasks for values of D_{max} in the range [30 – 70] m, while

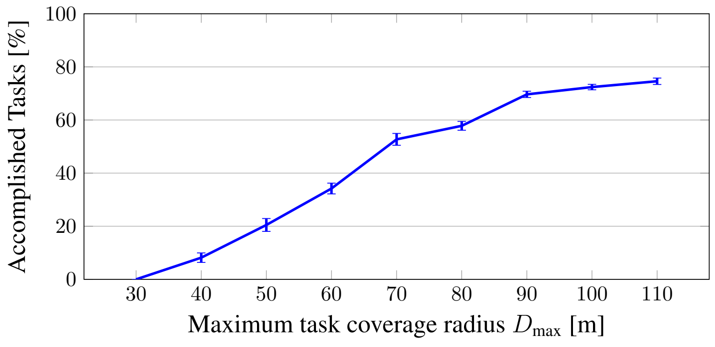


Figure 12.10: Average number of successfully accomplished tasks with increasing values of radius $D_{\max}$

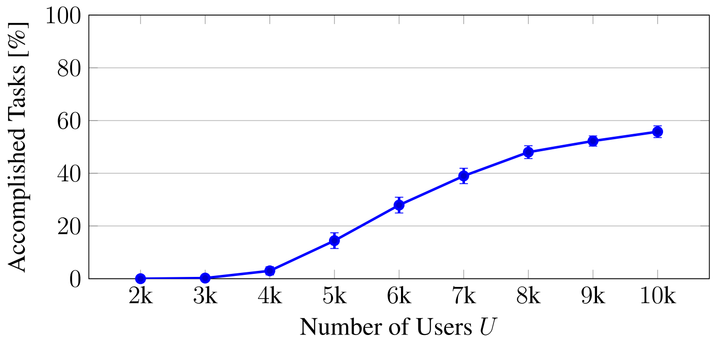


Figure 12.11: Average number of successfully accomplished tasks with increasing number of users in the system

for $D_{\max} > 70$ the increase becomes smoother. The behavior suggests that increasing the maximum task coverage radius significantly helps to contact higher number of users and to cope with user movement. However, recruiting users far from the location of the sensing task may result in poor accuracy for particular applications. For example, if the sensing task requires users to take a picture, being closer to the location of the task is crucial. On the other hand, for the vast majority of S^{2}aaS applications requiring monitoring of phenomena such as noise or air pollution, temperature and pressure, having users far for the location of the sensing task can be acceptable.

The previous experiments were conducted having fixed the population. The following analysis aims to assess the impact of the total number of users in the system $\mathcal{U}$. A higher number of users in the system makes larger the selection pool and, intuitively, should favor task accomplishment. However, the hypothesis has to be verified for two reasons: i) the proportionality between N and $\mathcal{U}$, see (12.10), and ii) the limited period the users move with respect to the total evaluation period (see Table 12.4) may lead users

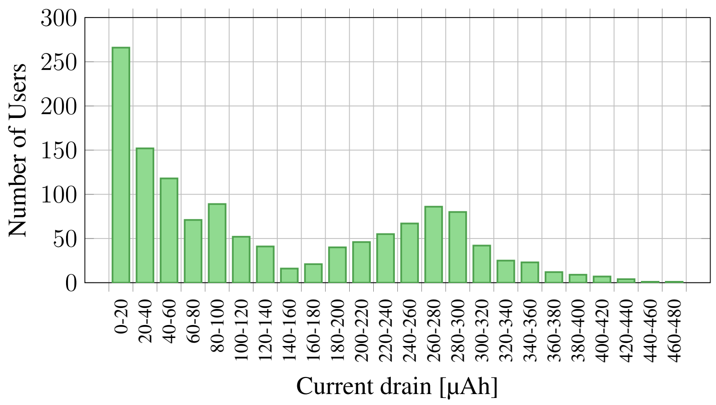


Figure 12.12: Sensing Cost

be active when the task already expired or did not started yet. Having set $D_{\max} = 70$ m, Fig. 12.11 shows that the number of tasks accomplished increases with the number of users $\mathcal{U}$. The number of accomplished tasks increases substantially (around 40%), when the population in the system changes from 3 k to 7 k. Because of the aforementioned reasons, the increase becomes more gentle when the number of users in the systems is greater than 7 k.

Analysis of the energy consumption of the users

This section evaluates the battery drain of user devices assessing the distribution of user energy consumption for sensing and data reporting. As expected, both distributions follow the same profile because data after being collected from the sensor is immediately delivered. Fig. 12.12 shows the distribution of users battery consumption due to sensing operations. The results are measured in form of current drain. The vast majority of the users spends little amount of energy for sensing. The motivation is twofold. First, many users contribute to only one task and because of the mobility, they contribute for few timeslots. Second, modern sensors are designed to be energy efficient. When compared with the battery capacity available in today smartphones, which is in the order of 2500 mAh, it is clear that the energy consumed for sensing is negligible with respect to the energy spent for communications (see Fig. 12.13).

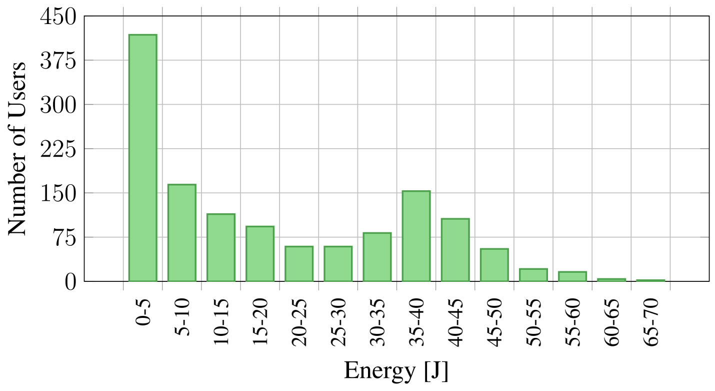


Figure 12.13: Communication Cost

Part V

Conclusion

Chapter 13

Conclusions and Perspectives

This chapter recalls the context of the thesis, provides a summary of the main strengths of this dissertation, and finally summarizes in each area the main contributions outlining perspectives and future research directions.

13.1 Summary

This thesis has studied the problem of energy efficiency of communications in distributed computing paradigms like cloud computing, mobile cloud and fog/edge computing. The analysis is a challenging problem as the overall performance depend on the computing paradigm and on the communication technology. Each domain, therefore, requires different approaches to optimize efficiency in using energy. In cloud data centers, saving energy can be obtained through turning off unutilized hardware or strengthen the efficiency of resource allocation. In mobile environments, energy efficiency can be achieved through offloading computing tasks for remote execution using the most adequate communication technologies. Finally, in mobile crowdsensing, efficient user recruitment leads to energy savings.

The strengths of this Ph.D. are in the proposition of contributions in all distributed computing paradigms. These includes not only traditional and well established research areas like cloud computing and mobile cloud computing, but also prospective areas such as fog computing and mobile crowdsensing. In addition, this Ph.D. thesis presents a wide range of methodologies used in the proposed contributions. Besides analytical models, in this thesis experimental solutions and a new simulator were designed. In the context of mobile cloud and fog computing an experimental Android application was designed for Google Glass and performance evaluation was carried profiling energy consumed through real measurements with a power monitor. As for proper operation, MCS systems require the contribution from a large number of participants, the development of a real testbed is not feasible. Therefore, simulations are the candidate tool to assess the costs and understand the performance of MCS systems. These motivations lead to the development of CrowdSenSim.

13.2 Cloud Computing: Contributions and Perspectives

In the context of cloud computing, this thesis has proposed a new framework of performance metrics for communication systems of data centers. The proposed framework allows a fine-grain analysis and comparison of communication systems, processes, and protocols, defining their influence on the performance of cloud applications. Thus, the framework of metrics is positioned to become an essential tool for monitoring, comparing and assessing performance of data center communication systems. In addition, the information provided by the metric is essential for optimizing daily operation of data center and to plan capacity extensions of existing facilities as well as to design future data centers.

As future work, the proposed metrics could be integrated into existing data center monitoring systems, such as VMware vCenter Log Insight or Cisco Prime Data Center Network Manager. The information the metrics provides, such as link- and traffic-related statistics as well as insights on energy are an excellent base to devise and design novel resource allocation and scheduling solutions. Moreover, the framework is projected to be considered for standardization.

13.3 Mobile Cloud/Fog Computing: Contributions and Perspectives

In this field, this dissertation has contributed a novel model for the problem of computation offloading, which describes the workflow of mobile applications through a new Directed Acyclic Graph (DAG) technique. This methodology is suitable for IoT devices working in fog computing environments and was used to design an Android application, called TreeGlass. As the name explains, the application performs recognition of trees using Google Glass. TreeGlass is evaluated experimentally in different offloading scenarios by measuring battery drain and time of execution as key performance indicators. The results highlights that communications during computation offloading may result being expensive on the basis of the type of data is necessary to transfer. In addition, from an energy perspective, mobile cloud computing should be preferred to fog computing in case of heavy data transfers.

The DAG methodology presented is general and therefore applicable to the design of new applications suitable for partitioning. Mobile cloud and fog computing can play an essential role in remote control and tactile Internet.

Accelerometer and gyroscope enable human hand gestures to become control commands for other devices such as drones or robots. Despite other control devices, smarthphones are very popular and have a much simpler interface, the touch screen. Being battery equipped, smartphones have constrained computing, battery and storage resources, which requires offloading the execution of heavy tasks to the cloud to enrich quality of experience and prolong battery lifetime.

13.4 Mobile CrowdSensing: Contributions and Perspectives

In the context of MCS, this Ph.D. has contributed novel metrics to assess efficiency of recruitment policies, accuracy of task completion and distribution of generated samples. Moreover, it was proposed a new framework for data acquisition in participatory MCS systems, which operates on a fog computing platform deployed widespread in urban environments. In participatory MCS systems, user recruitment is important and the proposed data acquisition framework exploits a novel policy. Users selection is based on three criteria, distance from the task location, user sociability and energy of the device. The performance of the framework was validated with CrowdSenSim, which is the first simulator designed for research use in large-scale urban MCS systems.

The preliminary work illustrated in this manuscript has a number of potential future directions. First, the proposed data acquisition framework is suitable for participatory systems. However, the participatory paradigm imposes a higher cost on the user in terms of cooperation effort. Having devices or applications responsible for sensing allows lowering the burden for user participation and makes opportunistic sensing suitable for distributed solutions. Thus, the proposal or the adaptation of the current framework to opportunistic systems is an extremely promising solution. Second, CrowdSenSim possible future developments are numerous and are projected to involve many research areas. From a technical point of view, the notion of places should be introduced to mark differently open areas than closed buildings such as restaurants, pubs, museums, hospitals, etc... Novel mobility models should be included to consider the case where users move towards a given destination after occurrence of an event. To conclude, CrowdSenSim can be used in the context of smart lighting. Lighting is an essential community service, but current implementations are not energy efficient and impacts on the energy budged of the municipalities for at least 40%. New smart lighting techniques makes use of Internet of Things (IoT) based lamppost, which save energy by turning off or dimming the light according to the presence of citizens. Therefore, CrowdSenSim has the potential to assess the costs and the benefits in adopting the new smart lighting solutions in real urban environments.

Appendix A

List of Publications

A.1 Journals

1. C. Fiandrino, D. Kliazovich, P. Bouvry, and A. Zomaya, "Performance and energy efficiency metrics for communication systems of cloud computing data centers", IEEE TRANSACTIONS ON CLOUD COMPUTING, 2015, ISSN: 2168-7161. DOI: 10.1109/TCC.2015.2424892

2. A. Capponi, C. Fiandrino, D. Kliazovich, P. Bouvry, and S. Giordano, "A Cost-Effective Distributed Framework for Data Collection in Cloud-based Mobile Crowd Sensing Architectures", *Submitted for second round of revision to* IEEE TRANSACTIONS ON SUSTAINABLE COMPUTING, 2016

3. C. Fiandrino, F. Anjomshoa, B. Kantarci, D. Kliazovich, P. Bouvry, and J. Matthews, "Sociability-Driven Framework for Data Acquisition in Mobile Crowd- sensing over Fog Computing Platforms for Smart Cities", *Submitted to* IEEE TRANSACTIONS ON SUSTAINABLE COMPUTING - *Special Issue on Orchestrating Sustainable Smart Cities: Methods and Techniques for Intelligence from clouds to edges*, 2016

4. M. Pouryazdan, C. Fiandrino, B. Kantarci, D. Kliazovich, T. Soyata, and P. Bouvry, "Game-Theoretic Trustworthiness Assurance for Crowd-sensed Big Data", *Submitted to* IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, 2016.

A.2 Conferences

1. C. Fiandrino, D. Kliazovich, P. Bouvry, and A. Zomaya, "NC-CELL: Network coding-based content distribution in cellular networks for cloud applications", in IEEE Global Communications Conference (GLOBE-

COM), Dec. 2014, pp. 1205–1210.
DOI: 10.1109/GLOCOM.2014.7036973

2. C. Fiandrino, D. Kliazovich, P. Bouvry, and A. Y. Zomaya, “Network-assisted offloading for mobile cloud applications”, in IEEE International Conference on Communications (ICC), Jun. 2015, pp. 5833–5838. DOI: 10.1109/ICC.2 015.7249252

3. C. Fiandrino, D. Kliazovich, P. Bouvry, and A. Y. Zomaya, “Performance met- rics for data center communication systems”, in 2015 IEEE 8th International Conference on Cloud Computing (CLOUD), Jun. 2015, pp. 98–105. DOI: 10.1109/CLOUD.2015.23.

4. C. Ragona, C. Fiandrino, D. Kliazovich, F. Granelli, and P. Bouvry, “Energy- efficient computation offloading for wearable devices and smartphones in mobile cloud computing”, in IEEE Global Communications Conference (GLOBECOM), Dec. 2015, pp. 1–6. DOI: 10.1109/GLOCOM.2015.7417039

5. C. Fiandrino, D. Kliazovich, P. Bouvry, and A. Zomaya, “Network coding-based content distribution in cellular access networks”, in 2016 IEEE International Conference on Communications (ICC), May 2016, pp. 1–6. DOI: 10.1109/ICC.2016.7510784.

6. P. Ruiu, A. Bianco, C. Fiandrino, P. Giaccone, and D. Kliazovich, “Power comparison of cloud data center architectures”, in IEEE International Conference on Communications (ICC), May 2016, pp. 1–6. DOI: 10.1109/ICC.201 6.7510998

7. A. Sciarrone, C. Fiandrino, I. Bisio, F. Lavagetto, D. Kliazovich, and P. Bouvry, “Smart probabilistic fingerprinting for indoor localization over fog computing platforms”, in IEEE 5th International Conference on Cloud Networking (CloudNet), Oct. 2016, ***Best Paper Award***

8. C. Fiandrino, B. Kantarci, F. Anjomshoa, D. Kliazovich, P. Bouvry, and J. Matthews, “Sociability-Driven user recruitment in mobile crowdsensing Internet of Things Platforms”, *Accepted for publication* in IEEE Global Communications Conference (GLOBECOM), Dec. 2016

9. A. Capponi, C. Fiandrino, C. Franck, U. Sorger, D. Kliazovich, and P. Bouvry, “Assessing performance of Internet of Things-based mobile crowdsensing systems for sensing as a service applications in smart cities”, *Accepted for publication in* 8th IEEE International Conference on Cloud Computing Technology and Science (CloudCom), Dec. 2016.

A.3 Workshops

1. M. Pouryazdan, C. Fiandrino, B. Kantarci, D. Kliazovich, T. Soyata, and P. Bouvry, “Game-Theoretic recruitment of sensing service providers for trustworthy Cloud-Centric Internet-of-Things (IoT) applications”, IEEE Global Communications Conference Workshops: Fifth International Workshop on Cloud Computing Systems, Networks, and Applications (GC16 Workshops CCSNA), Dec. 2016.

Bibliography

[1] Forbes, *Roundup of cloud computing forecasts and market estimates*, 2016. [Online]. Available: http://www.forbes.com/sites/louiscolumbus/2016/03/13/roundup-of-cloud-computing-forecasts-and-market-estimates-2016/#3ab56c0074b0 (cit. on p. 15).

[2] N. Fernando, S. W. Loke, and W. Rahayu, “Mobile cloud computing: A survey”, *Future Generation Computer Systems*, vol. 29, no. 1, pp. 84 –106, 2013, ISSN: 0167-739X. DOI: http://dx.doi.org/10.1016/j.future.2012.05.023 (cit. on pp. 15, 25, 72, 75).

[3] H. Schaffers, N. Komninos, M. Pallot, B. Trousse, M. Nilsson, and A. Oliveira, “Smart cities and the future internet: Towards cooperation frameworks for open innovation”, in *The Future Internet*, ser. Lecture Notes in Computer Science, vol. 6656, Springer Berlin Heidelberg, 2011, pp. 431–446, ISBN: 978-3-642-20897-3 (cit. on pp. 16, 119).

[4] S. Nawaz, C. Efstratiou, and C. Mascolo, “Parksense: A smartphone based sensing system for on-street parking”, in *19th Annual International Conference on Mobile Computing &#38; Networking*, ser. MobiCom ’13, Miami, Florida, USA: ACM, 2013, pp. 75–86, ISBN: 978-1-4503-1999-7. DOI: 10.1145/2500423.2500438 (cit. on p. 16).

[5] R. Ganti, F. Ye, and H. Lei, “Mobile crowdsensing: Current state and future challenges”, *IEEE Communications Magazine*, vol. 49, no. 11, pp. 32–39, Nov. 2011, ISSN: 0163-6804. DOI: 10.1109/MCOM.2011.6069707 (cit. on pp. 16, 118, 120).

[6] S. Mathur, T. Jin, N. Kasturirangan, J. Chandrasekaran, W. Xue, M. Gruteser, and W. Trappe, “Parknet: Drive-by sensing of road-side parking statistics”, in *Proceedings of the 8th international conference on Mobile systems, applications, and services*, ACM, 2010, pp. 123–136 (cit. on p. 16).

[7] P. Mell and T. Grance, “The nist definition of cloud computing”, 2011 (cit. on p. 21).

[8] M. Guzek, P. Bouvry, and E. G. Talbi, “A survey of evolutionary computation for resource management of processing in cloud computing [review article]”, *IEEE Computational Intelligence Magazine*, vol. 10, no. 2, pp. 53–67, May 2015, ISSN: 1556-603X. DOI: 10.1109/MCI.2015.2405351 (cit. on p. 22).

[9] G. Coulouris, J. Dollimore, T. Kindberg, and G. Blair, *Distributed systems: Concepts and design*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2011 (cit. on p. 22).

[10] Y. Duan, G. Fu, N. Zhou, X. Sun, N. C. Narendra, and B. Hu, “Everything as a service (xaas) on the cloud: Origins, current and future trends”, in *IEEE 8th International Conference on Cloud Computing*, Jun. 2015, pp. 621–628. DOI: 10.1109/CLOUD.2015.88 (cit. on p. 22).

[11] O. Vermesan and P. Friess, *Internet of Things - From research and innovation to market deployment*. River Publishers Aalborg, 2014 (cit. on p. 23).

[12] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, “Internet of Things: A survey on enabling technologies, protocols, and applications”, *IEEE Communications Surveys Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015, ISSN: 1553-877X. DOI: 10.1109/COMST.2015.2444095 (cit. on p. 23).

[13] *Worldwide sales of iot devices*, Gartner, 2015. [Online]. Available: http://www.gartner.com/newsroom/id/3165317 (cit. on p. 23).

[14] S. Andreev, O. Galinina, A. Pyattaev, M. Gerasimenko, T. Tirronen, J. Torsner, J. Sachs, M. Dohler, and Y. Koucheryavy, “Understanding the IoT connectivity landscape: a contemporary M2M radio technology roadmap”, *IEEE COMMUNICATIONS MAGAZINE*, vol. 53, no. 9, pp. 32–40, Sep. 2015, ISSN: 0163-6804. DOI: 10.1109/MCOM.2015.7263370 (cit. on p. 25).

[15] A. u. R. Khan, M. Othman, S. A. Madani, and S. U. Khan, “A survey of mobile cloud computing application models”, *IEEE Communications Surveys Tutorials*, vol. 16, no. 1, pp. 393–413, 2014, ISSN: 1553-877X. DOI: 10.1109/SURV.2013.062613.00160 (cit. on pp. 25, 75).

[16] Z. Sanaei, S. Abolfazli, A. Gani, and R. Buyya, “Heterogeneity in mobile cloud computing: Taxonomy and open challenges”, *IEEE Communications Surveys Tutorials*, vol. 16, no. 1, pp. 369–392, 2014, ISSN: 1553-877X. DOI: 10.1109/SURV.2013.050113.00090 (cit. on pp. 25, 75).

[17] M. Shiraz, A. Gani, R. H. Khokhar, and R. Buyya, “A review on distributed application processing frameworks in smart mobile devices for mobile cloud computing”, *IEEE Communications Surveys Tutorials*, vol. 15, no. 3, pp. 1294–1313, 2013, ISSN: 1553-877X (cit. on pp. 25, 75, 78).

[18] H. T. Dinh, C. Lee, D. Niyato, and P. Wang, “A survey of mobile cloud computing: Architecture, applications, and approaches”, *Wireless Communications and Mobile Computing*, vol. 13, no. 18, pp. 1587–1611, 2013, ISSN: 1530-8677. DOI: 10.1002/wcm.1203 (cit. on pp. 25, 72, 74).

[19] S. Abolfazli, Z. Sanaei, E. Ahmed, A. Gani, and R. Buyya, “Cloud-based augmentation for mobile devices: Motivation, taxonomies, and open challenges”, *IEEE Communications Surveys Tutorials*, vol. 16, no. 1, pp. 337–368, 2014, ISSN: 1553-877X (cit. on p. 25).

[20] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, “Fog computing and its role in the Internet of Things”, in *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*, ser. MCC ’12, Helsinki, Finland: ACM, 2012, pp. 13–16, ISBN: 978-1-4503-1519-7 (cit. on pp. 25, 73).

[21] F. Bonomi, R. Milito, P. Natarajan, and J. Zhu, “Fog computing: A platform for Internet of Things and analytics”, in *Big Data and Internet of Things: A Roadmap for Smart Environments*, Springer, 2014, pp. 169–186 (cit. on pp. 25, 73).

[22] S. Sarkar, S. Chatterjee, and S. Misra, “Assessment of the suitability of fog computing in the context of Internet of Things”, *IEEE Transactions on Cloud Computing*, 2015, ISSN: 2168-7161. DOI: 10.1109/TCC.2015.2485206 (cit. on pp. 25, 73).

[23] I. Bisio, F. Lavagetto, A. Sciarrone, T. Penner, and M. Guirguis, “Context-Awareness over Transient Cloud in D2D networks: Energy Performance Analysis and Evaluation”, *TRANSACTIONS ON EMERGING TELECOMMUNICATIONS TECHNOLOGIES*, 2015 (cit. on p. 26).

[24] T. Taleb and A. Ksentini, “Follow me cloud: interworking federated clouds and distributed mobile networks”, *IEEE NETWORK*, vol. 27, no. 5, pp. 12–19, Sep. 2013 (cit. on p. 26).

[25] S. Tanzil, O. Gharehshiran, and V. Krishnamurthy, “A distributed coalition game approach to femto-cloud formation”, *IEEE Transactions on Cloud Computing*, 2016, ISSN: 2168-7161. DOI: 10.1109/TCC.2016.2594175 (cit. on p. 26).

[26] S. M. S. Tanzil, O. N. Gharehshiran, and V. Krishnamurthy, “Femto-Cloud Formation: A Coalitional Game-Theoretic Approach”, in *IEEE GLOBAL COMMUNICATIONS CONFERENCE (GLOBECOM)*, Dec. 2015, pp. 1–6 (cit. on p. 26).

[27] S. Yu, R. Langar, W. Li, and X. Chen, “Coalition-based energy efficient offloading strategy for immersive collaborative applications in femto-cloud”, in *IEEE International Conference on Communications (ICC)*, May 2016, pp. 1–6. DOI: 10.1109/ICC.2016.7511581 (cit. on p. 26).

[28] K. Habak, M. Ammar, K. A. Harras, and E. Zegura, “Femto clouds: Leveraging mobile devices to provide cloud service at the edge”, in *IEEE 8th International Conference on Cloud Computing*, Jun. 2015, pp. 9–16. DOI: 10.1109/CLOUD.2015.12 (cit. on p. 26).

[29] A. B. Zaslavsky, C. Perera, and D. Georgakopoulos, “Sensing as a service and big data”, *CoRR*, vol. abs/1301.0159, 2013. [Online]. Available: http://arxiv.org/abs/1301.0159 (cit. on p. 27).

[30] P. Patel, A. Pathak, T. Teixeira, and V. Issarny, “Towards application development for the internet of things”, in *8th Middleware Doctoral Symposium*, ser. MDS ’11, Lisbon, Portugal: ACM, 2011, pp. 1–6, ISBN: 978-1-4503-1072-7. DOI: 10.1145/2093190.2093195 (cit. on p. 27).

[31] C. Fiandrino, B. Kantarci, F. Anjomshoa, D. Kliazovich, P. Bouvry, and J. Matthews, “Sociability-Driven user recruitment in mobile crowdsensing Internet of Things Platforms”, in *IEEE Global Communications Conference (GLOBECOM)*, Dec. 2016 (cit. on p. 28).

[32] Y. Sun, H. Song, A. J. Jara, and R. Bie, “Internet of things and big data analytics for smart and connected communities”, *IEEE Access*, vol. 4, pp. 766–773, 2016, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2016.2529723 (cit. on p. 28).

[33] P. J. Zabinski, B. K. Gilbert, and E. S. Daniel, “Coming challenges with terabit-per-second data communication”, *IEEE Circuits and Systems Magazine*, vol. 13, no. 3, pp. 10–20, 2013, ISSN: 1531-636X. DOI: 10.1109/MCAS.2013.2271441 (cit. on p. 29).

[34] J. M. Senior and M. Y. Jamro, *Optical fiber communications: Principles and practice*. Pearson Education, 2009 (cit. on p. 29).

[35] P. Rawat, K. D. Singh, H. Chaouchi, and J. M. Bonnin, "Wireless sensor networks: A survey on recent developments and potential synergies", *The Journal of Supercomputing*, vol. 68, no. 1, pp. 1–48, 2014, ISSN: 1573-0484. DOI: 10.1007/s11227-013-1021-9 (cit. on p. 29).

[36] N. A. Pantazis, S. A. Nikolidakis, and D. D. Vergados, "Energy-efficient routing protocols in wireless sensor networks: A survey", *IEEE Communications Surveys Tutorials*, vol. 15, no. 2, pp. 551–591, 2013, ISSN: 1553-877X. DOI: 10.1109/SURV.2012.062612.00084 (cit. on p. 29).

[37] C. Gomez and J. Paradells, "Wireless home automation networks: A survey of architectures and technologies", *IEEE Communications Magazine*, vol. 48, no. 6, pp. 92–101, Jun. 2010, ISSN: 0163-6804. DOI: 10.1109/MCOM.2010.5473869 (cit. on p. 29).

[38] M. Erol-Kantarci and H. T. Mouftah, "Wireless sensor networks for cost-efficient residential energy management in the smart grid", *IEEE Transactions on Smart Grid*, vol. 2, no. 2, pp. 314–325, Jun. 2011, ISSN: 1949-3053. DOI: 10.1109/TSG.2011.2114678 (cit. on p. 29).

[39] A. Hamza, J. Deogun, and D. Alexander, "Wireless communication in data centers: A survey", *IEEE Communications Surveys Tutorials*, 2016, ISSN: 1553-877X. DOI: 10.1109/COMST.2016.2521678 (cit. on p. 29).

[40] *Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2015–2020*, Cisco White paper, 2016 (cit. on pp. 30, 74, 75).

[41] A. Gupta and R. K. Jha, "A survey of 5g network: Architecture and emerging technologies", *IEEE Access*, vol. 3, pp. 1206–1232, 2015, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2015.2461602 (cit. on p. 30).

[42] S. Sesia, I. Toufik, and M. Baker, *Lte-the umts long term evolution*. Wiley Online Library, 2015 (cit. on p. 30).

[43] 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, 2014, ISSN: 1553-877X. DOI: 10.1109/COMST.2014.2319555 (cit. on p. 30).

[44] D. Camps-Mur, A. Garcia-Saavedra, and P. Serrano, "Device-to-device communications with Wi-Fi Direct: Overview and experimentation", *IEEE Wireless Communications*, vol. 20, no. 3, pp. 96–104, Jun. 2013, ISSN: 1536-1284. DOI: 10.1109/MWC.2013.6549288 (cit. on p. 31).

[45] E. Khorov, A. Lyakhov, A. Krotov, and A. Guschin, "A survey on {ieee} 802.11ah: An enabling networking technology for smart cities", *Computer Communications*, vol. 58, pp. 53 –69, 2015, Special Issue on Networking and Communications for Smart Cities, ISSN: 0140-3664. DOI: http://dx.doi.org/10.1016/j.comcom.2014.08.008 (cit. on p. 32).

[46] Y. Sverdlik, *Here's how much energy all us data centers consume*, 2016. [Online]. Available: http://www.datacenterknowledge.com/archives/2016/06/27/heres-how-much-energy-all-us-data-centers-consume/ (cit. on p. 35).

[47] A. S. Andrae and T. Edler, "On global electricity usage of communication technology: Trends to 2030", *Challenges*, vol. 6, no. 1, pp. 117–157, 2015 (cit. on p. 35).

[48] G. Sauls, *Measurement of data centre power consumption*, Falcon Electronics Pty LTD, White Paper, 2008 (cit. on p. 36).

[49] K. Zheng, X. Wang, L. Li, and X. Wang, "Joint power optimization of data center network and servers with correlation analysis", in *IEEE Conference on Computer Communications (INFOCOM)*, Apr. 2014, pp. 2598–2606. DOI: `10.1109/INFOCOM.2014.6848207` (cit. on p. 36).

[50] R. Bolla, R. Bruschi, F. Davoli, and F. Cucchietti, "Energy efficiency in the future internet: A survey of existing approaches and trends in energy-aware fixed network infrastructures", *IEEE Communications Surveys Tutorials*, vol. 13, no. 2, pp. 223–244, 2011, ISSN: 1553-877X. DOI: `10.1109/SURV.2011.071410.00073` (cit. on p. 36).

[51] C. Lange, D. Kosiankowski, D. von Hugo, and A. Gladisch, "Analysis of the energy consumption in telecom operator networks", *Photonic Network Communications*, vol. 30, no. 1, pp. 17–28, 2015, ISSN: 1572-8188. DOI: `10.1007/s11107-015-0492-4` (cit. on p. 36).

[52] Ł Budzisz, F. Ganji, G. Rizzo, M. A. Marsan, M. Meo, Y. Zhang, G. Koutitas, L. Tassiulas, S. Lambert, B. Lannoo, M. Pickavet, A. Conte, I. Haratcherev, and A. Wolisz, "Dynamic resource provisioning for energy efficiency in wireless access networks: A survey and an outlook", *IEEE Communications Surveys Tutorials*, vol. 16, no. 4, pp. 2259–2285, 2014, ISSN: 1553-877X. DOI: `10.1109/COMST.2014.2329505` (cit. on p. 37).

[53] T. Rault, A. Bouabdallah, and Y. Challal, "Energy efficiency in wireless sensor networks: A top-down survey", *Computer Networks*, vol. 67, pp. 104 –122, 2014, ISSN: 1389-1286. DOI: `http://dx.doi.org/10.1016/j.comnet.2014.03.027` (cit. on p. 37).

[54] C. Fiandrino, D. Kliazovich, P. Bouvry, and A. Y. Zomaya, "Nc-cell: Network coding-based content distribution in cellular networks for cloud applications", in *IEEE Global Communications Conference (GLOBECOM)*, Dec. 2014, pp. 1205–1210. DOI: `10.1109/GLOCOM.2014.7036973` (cit. on pp. 38, 92).

[55] C. Fiandrino, D. Kliazovich, P. Bouvry, and A. Zomaya, "Network coding-based content distribution in cellular access networks", in *2016 IEEE International Conference on Communications (ICC)*, May 2016, pp. 1–6. DOI: `10.1109/ICC.2016.7510784` (cit. on p. 38).

[56] M. Bari, R. Boutaba, R. Esteves, L. Granville, M. Podlesny, M. Rabbani, Q. Zhang, and M. Zhani, "Data center network virtualization: A survey", *IEEE Communications Surveys Tutorials*, vol. 15, no. 2, pp. 909–928, 2013, ISSN: 1553-877X. DOI: `10.1109/SURV.2012.090512.00043` (cit. on pp. 40, 42, 62).

[57] K. Bilal, S. U. R. Malik, O. Khalid, A. Hameed, E. Alvarez, V. Wijaysekara, R. Irfan, S. Shrestha, D. Dwivedy, M. Ali, U. S. Khan, A. Abbas, N. Jalil, and S. U. Khan, "A taxonomy and survey on green data center networks", *Future Generation Computer Systems*, vol. 36, pp. 189 –208, 2014, ISSN: 0167-739X. DOI: `http://dx.doi.org/10.1016/j.future.2013.07.006` (cit. on pp. 40, 42).

[58] A. Hammadi and L. Mhamdi, "A survey on architectures and energy efficiency in data center networks", *Computer Communications*, vol. 40, pp. 1 –21, 2014, ISSN: 0140-3664. DOI: `http://dx.doi.org/10.1016/j.comcom.2013.11.005` (cit. on pp. 40, 42, 62).

[59] B. Wang, Z. Qi, R. Ma, H. Guan, and A. V. Vasilakos, "A survey on data center networking for cloud computing", *Computer Networks*, vol. 91, pp. 528 –547, 2015, ISSN: 1389-1286. DOI: `http://dx.doi.org/10.1016/j.comnet.2015.08.040` (cit. on pp. 40, 42).

[60] J. Shuja, K. Bilal, S. A. Madani, M. Othman, R. Ranjan, P. Balaji, and S. U. Khan, "Survey of techniques and architectures for designing energy-efficient data centers", *IEEE Systems Journal*, vol. 10, no. 2, pp. 507–519, Jun. 2016, ISSN: 1932-8184. DOI: `10.1109/JSYST.2014.2315823` (cit. on pp. 40, 42).

[61] D. Kliazovich, J. Pecero, A. Tchernykh, P. Bouvry, S. U. Khan, and A. Zomaya, "Ca-dag: Modeling communication-aware applications for scheduling in cloud computing", *Journal of Grid Computing*, pp. 1–17, 2015, ISSN: 1570-7873. DOI: `10.1007/s10723-015-9337-8` (cit. on pp. 40, 50, 51, 56, 58, 79).

[62] K. Chen, A. Singla, A. Singh, K. Ramachandran, L. Xu, Y. Zhang, X. Wen, and Y. Chen, "Osa: An optical switching architecture for data center networks with unprecedented flexibility", *IEEE/ACM Transactions on Networking*, vol. 22, no. 2, pp. 498–511, Apr. 2014, ISSN: 1063-6692. DOI: `10.1109/TNET.2013.2253120` (cit. on p. 42).

[63] Y. Zhao, S. Wang, S. Luo, V. Anand, H. Yu, X. Wang, S. Xu, and X. Zhang, "Dynamic topology management in optical data center networks", *Journal of Lightwave Technology*, vol. 33, no. 19, pp. 4050–4062, Oct. 2015, ISSN: 0733-8724. DOI: `10.1109/JLT.2015.2464079` (cit. on p. 42).

[64] A. Singla, A. Singh, K. Ramachandran, L. Xu, and Y. Zhang, "Proteus: A topology malleable data center network", in *9th ACM SIGCOMM Workshop on Hot Topics in Networks*, ser. Hotnets-IX, Monterey, California: ACM, 2010, pp. 1–6, ISBN: 978-1-4503-0409-2. DOI: `10.1145/1868447.1868455` (cit. on p. 42).

[65] N. Farrington, G. Porter, S. Radhakrishnan, H. H. Bazzaz, V. Subramanya, Y. Fainman, G. Papen, and A. Vahdat, "Helios: A hybrid electrical/optical switch architecture for modular data centers", in *ACM SIGCOMM 2010 Conference*, ser. SIGCOMM, New Delhi, India: ACM, 2010, pp. 339–350, ISBN: 978-1-4503-0201-2. DOI: `10.1145/1851182.1851223` (cit. on p. 42).

[66] G. Wang, D. G. Andersen, M. Kaminsky, K. Papagiannaki, T. E. Ng, M. Kozuch, and M. Ryan, "C-through: Part-time optics in data centers", in *ACM SIGCOMM 2010 Conference*, ser. SIGCOMM, New Delhi, India: ACM, 2010, pp. 327–338, ISBN: 978-1-4503-0201-2. DOI: `10.1145/1851182.1851222` (cit. on p. 42).

[67] V. B. Srikanth Kandula Jitu Padhye, “Flyways to de-congest data center networks”, Tech. Rep., Aug. 2009. [Online]. Available: `https://www.microsoft.com/en-us/research/publication/flyways-to-de-congest-data-center-networks/` (cit. on p. 42).

[68] A. S. Hamza, J. S. Deogun, and D. R. Alexander, “Free space optical data center architecture design with fully connected racks”, in *IEEE Global Communications Conference*, Dec. 2014, pp. 2192–2197. DOI: `10.1109/GLOCOM.2014.7037133` (cit. on p. 42).

[69] M. Al-Fares, A. Loukissas, and A. Vahdat, “A scalable, commodity data center network architecture”, in *Proceedings of the ACM SIGCOMM 2008 Conference on Data Communication*, ser. SIGCOMM ’08, Seattle, WA, USA: ACM, 2008, pp. 63–74, ISBN: 978-1-60558-175-0. DOI: `10.1145/1402958.1402967` (cit. on pp. 43, 56, 62).

[70] R. Niranjan Mysore, A. Pamboris, N. Farrington, N. Huang, P. Miri, S. Radhakrishnan, V. Subramanya, and A. Vahdat, “PortLand: A scalable fault-tolerant layer 2 data center network fabric”, in *ACM SIGCOMM Computer Communication Review*, ACM, vol. 39, 2009, pp. 39–50 (cit. on pp. 43, 56, 62).

[71] A. Greenberg, J. R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D. A. Maltz, P. Patel, and S. Sengupta, “VL2: A scalable and flexible data center network”, in *ACM SIGCOMM Computer Communication Review*, ACM, vol. 39, 2009, pp. 51–62 (cit. on pp. 43, 56, 62).

[72] A. Singh, J. Ong, A. Agarwal, G. Anderson, A. Armistead, R. Bannon, S. Boving, G. Desai, B. Felderman, P. Germano, A. Kanagala, J. Provost, J. Simmons, E. Tanda, J. Wanderer, U. Hölzle, S. Stuart, and A. Vahdat, “Jupiter rising: A decade of clos topologies and centralized control in google’s datacenter network”, in *ACM Conference on Special Interest Group on Data Communication*, ser. SIGCOMM ’15, London, United Kingdom: ACM, 2015, pp. 183–197, ISBN: 978-1-4503-3542-3. DOI: `10.1145/2785956.2787508` (cit. on p. 43).

[73] C. Guo, G. Lu, D. Li, H. Wu, X. Zhang, Y. Shi, C. Tian, Y. Zhang, and S. Lu, “BCube: A high performance, server-centric network architecture for modular data centers”, *ACM SIGCOMM Computer Communication Review*, vol. 39, no. 4, pp. 63–74, 2009 (cit. on pp. 43, 56, 62).

[74] C. Guo, H. Wu, K. Tan, L. Shi, Y. Zhang, and S. Lu, “DCell: A scalable and fault-tolerant network structure for data centers”, in *ACM SIGCOMM Computer Communication Review*, ACM, vol. 38, 2008, pp. 75–86 (cit. on pp. 43, 56, 62).

[75] I. A.A. Z. Abdallah, D. Kliazovich, and P. Bouvry, “On service level agreement assurance in cloud computing data centers”, in *IEEE 9th International Conference on Cloud Computing*, Jun. 2016, pp. 921–926 (cit. on p. 43).

[76] D. Kliazovich, P. Bouvry, and S. Khan, “DENS: Data center energy-efficient network-aware scheduling”, *Cluster Computing, Special Issue on Green Networks*, vol. 16, pp. 69–75, 2013 (cit. on pp. 44, 68).

[77] I. Takouna, R. Rojas-Cessa, K. Sachs, and C. Meinel, “Communication-aware and energy-efficient scheduling for parallel applications in virtualized data centers”, in *IEEE/ACM 6th International Conference on Utility and Cloud Computing (UCC)*, Dec. 2013, pp. 251–255. DOI: 10.1109/UCC.2013.50 (cit. on p. 44).

[78] D. Kliazovich, S. Arzo, F. Granelli, P. Bouvry, and S. Khan, “E-STAB: Energy-efficient scheduling for cloud computing applications with traffic load balancing”, in *IEEE International Conference on Green Computing and Communications (GreenCom)*, Aug. 2013, pp. 7–13. DOI: 10.1109/GreenCom-iThings-CPSCom.2013.28 (cit. on pp. 44, 68).

[79] M. Guzek, D. Kliazovich, and P. Bouvry, “Heros: Energy-efficient load balancing for heterogeneous data centers”, in *2015 IEEE 8th International Conference on Cloud Computing*, Jun. 2015, pp. 742–749. DOI: 10.1109/CLOUD.2015.103 (cit. on pp. 44, 46).

[80] M. Zotkiewicz, M. Guzek, D. Kliazovich, and P. Bouvry, “Minimum dependencies energy-efficient scheduling in data centers”, *IEEE Transactions on Parallel and Distributed Systems*, 2016, ISSN: 1045-9219. DOI: 10.1109/TPDS.2016.2542817 (cit. on pp. 44, 79).

[81] D. Belabed, S. Secci, G. Pujolle, and D. Medhi, “Striking a balance between traffic engineering and energy efficiency in virtual machine placement”, *IEEE Transactions on Network and Service Management*, vol. 12, no. 2, pp. 202–216, Jun. 2015, ISSN: 1932-4537. DOI: 10.1109/TNSM.2015.2413755 (cit. on p. 45).

[82] S. K. Singh, T. Das, and A. Jukan, “A survey on internet multipath routing and provisioning”, *IEEE Communications Surveys Tutorials*, vol. 17, no. 4, pp. 2157–2175, 2015, ISSN: 1553-877X. DOI: 10.1109/COMST.2015.2460222 (cit. on p. 45).

[83] R. Wang, J. A. Wickboldt, R. P. Esteves, L. Shi, B. Jennings, and L. Z. Granville, “Using empirical estimates of effective bandwidth in network-aware placement of virtual machines in datacenters”, *IEEE Transactions on Network and Service Management*, vol. 13, no. 2, pp. 267–280, Jun. 2016, ISSN: 1932-4537. DOI: 10.1109/TNSM.2016.2530309 (cit. on p. 45).

[84] F Kelly, “Notes on effective bandwidths”, *Stochastic Networks: Theory and Applications*, vol. 4, 1996 (cit. on p. 45).

[85] The Green Grid, *Data center power efficiency metrics: PUE and DCiE*, White Paper, 2008 (cit. on pp. 45, 53).

[86] J. Yuventi and R. Mehdizadeh, *A critical analysis of power usage effectiveness and its use as data center energy sustainability metrics*, http://cife.stanford.edu/sites/default/files/WP131_0.pdf, 2013 (cit. on p. 45).

[87] The Green Grid, *Carbon usage effectiveness (cue): A green grid data center sustainability metric*, White Paper, 2010 (cit. on p. 45).

[88] The Green Grid, *A metric for measuring the benefit of reuse energy from a data center*, White Paper, 2010 (cit. on p. 46).

[89] *UPS load factor*, http://hightech.lbl.gov/benchmarking-guides/data-p1.html, 2009 (cit. on p. 46).

[90] The Green Grid, *A framework for data center energy productivity*, White Paper, 2008 (cit. on p. 46).

[91] *Data center efficiency - beyond PUE and DCiE*, http://blogs.gartner.com/david_cappuccio/2009/02/15/data-center-efficiency-beyond-pue-and-dcie/, 2009 (cit. on p. 46).

[92] X. Fan, W.-D. Weber, and L. A. Barroso, "Power provisioning for a warehouse-sized computer", *SIGARCH Comput. Archit. News*, vol. 35, no. 2, pp. 13–23, Jun. 2007, ISSN: 0163-5964. DOI: 10.1145/1273440.1250665 (cit. on p. 46).

[93] M. K. Herrlin, "Airflow and cooling performance of data centers: Two performance metrics.", *ASHRAE Transactions*, vol. 114, no. 2, pp. 182 –187, 2008, ISSN: 00012505 (cit. on pp. 46, 49).

[94] J. W. Vangilder and S. K. Shrivastava, "Real-time prediction of rack-cooling performance", *ASHRAE Transactions*, vol. 112, no. 2, pp. 151–162, 2006 (cit. on p. 46).

[95] J. W. Vangilder and S. Shrivastava, "Capture index: An airflow-based rack cooling performance metric", *ASHRAE Transactions*, vol. 113, no. 1, pp. 126–136, 2007 (cit. on p. 46).

[96] A. Amokrane, R. Langar, M. F. Zhani, R. Boutaba, and G. Pujolle, "Greenslater: On satisfying green slas in distributed clouds", *IEEE Transactions on Network and Service Management*, vol. 12, no. 3, pp. 363–376, Sep. 2015, ISSN: 1932-4537. DOI: 10.1109/TNSM.2015.2440423 (cit. on p. 46).

[97] P. Mathew, "Self-benchmarking guide for data centers: Metrics, benchmarks, actions", *Lawrence Berkeley National Laboratory*, 2010 (cit. on pp. 46, 47).

[98] F. Michaut and F. Lepage, "Application-oriented network metrology: Metrics and active measurement tools", *IEEE Communications Surveys & Tutorials*, vol. 7, no. 2, 2005 (cit. on p. 47).

[99] A. Hanemann, A. Liakopoulos, M. Molina, and D. M. Swany, "A study on network performance metrics and their composition", *Campus-Wide Information Systems*, vol. 23, no. 4, pp. 268–282, 2006 (cit. on p. 47).

[100] R. Baumann, S. Heimlicher, M. Strasser, and A. Weibel, "A survey on routing metrics", *TIK Report*, vol. 262, 2007, ETH-Zentrum, Zurich, Switzerland (cit. on p. 47).

[101] H. Khandelwal, R. R. Kompella, and R. Ramasubramanian, *Cloud monitoring framework*, White Paper, 2010 (cit. on p. 47).

[102] L. Popa, S. Ratnasamy, G. Iannaccone, A. Krishnamurthy, and I. Stoica, "A cost comparison of datacenter network architectures", in *Proceedings of the 6th International Conference*, ser. Co-NEXT, Philadelphia, Pennsylvania: ACM, 2010, 16:1–16:12, ISBN: 978-1-4503-0448-1. DOI: 10.1145/1921168.1921189 (cit. on p. 47).

[103] *Cloud computing market*, http://www.zdnet.com/blog/btl/cloud-computing-market-241-billion-in-2020/47702, 2011 (cit. on pp. 47, 50).

[104] M. Uddin, A. A. Rahman, and A. Shah, "Criteria to select energy efficiency metrics to measure performance of data centre", *International Journal of Energy Technology and Policy*, vol. 8, no. 3, pp. 224–237, 2012 (cit. on p. 49).

[105] P. A. Mathew, S. E. Greenberg, S. Ganguly, D. A. Sartor, and W. F. Tschudi, "How does your data center measure up? Energy efficiency metrics and benchmarks for data center infrastructure systems", *HPAC Journal*, 2009 (cit. on p. 49).

[106] L. Wang and S. U. Khan, "Review of performance metrics for green data centers: A taxonomy study", *The Journal of Supercomputing*, vol. 63, no. 3, pp. 639–656, 2013 (cit. on p. 49).

[107] T. G. Grid, *Harmonizing global metrics for data center energy efficiency*, White Paper, 2014 (cit. on p. 49).

[108] R. Tozer and M. Salim, "Data center air management metrics-practical approach", in *12th IEEE Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems (ITherm)*, 2010, pp. 1–8. DOI: 10.1109/ITHERM.2010.5501366 (cit. on p. 49).

[109] S Flucker and R Tozer, "Data centre cooling air performance metrics", in *CIBSE Technical Symposium, Leicester, CIBSE*, 2011, pp. 1–16 (cit. on p. 49).

[110] E. Volk, A. Tenschert, M. Gienger, A. Oleksiak, L. Siso, and J. Salom, "Improving energy efficiency in data centers and federated cloud environments: Comparison of CoolEmAll and Eco2Clouds approaches and metrics", in *2013 Third International Conference on Cloud and Green Computing (CGC)*, 2013, pp. 443–450. DOI: 10.1109/CGC.2013.76 (cit. on p. 49).

[111] D. Kliazovich, P. Bouvry, F. Granelli, and N. Fonseca, "Energy consumption optimization in cloud data centers", in *Cloud Services, Networking, and Management*, edited by N. Fonseca and R. Boutaba, Eds., Wiley-IEEE Press, Apr. 2015 (cit. on p. 49).

[112] B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma, S. Banerjee, and N. McKeown, "Elastictree: Saving energy in data center networks.", in *NSDI*, vol. 3, 2010, pp. 19–21 (cit. on p. 50).

[113] D. Abts, M. R. Marty, P. M. Wells, P. Klausler, and H. Liu, "Energy proportional datacenter networks", in *ACM SIGARCH Computer Architecture News*, ACM, vol. 38, 2010, pp. 338–347 (cit. on pp. 50, 55).

[114] Cisco, *Cisco Global Cloud Index: Forecast and Methodology, 2014-2019*, White paper, 2016 (cit. on pp. 50, 61).

[115] G. Chen, W. He, J. Liu, S. Nath, L. Rigas, L. Xiao, and F. Zhao, "Energy-Aware Server Provisioning and Load Dispatching for Connection-Intensive Internet Services.", in *NSDI*, vol. 8, 2008, pp. 337–350 (cit. on p. 54).

[116] P. Mahadevan, P. Sharma, S. Banerjee, and P. Ranganathan, "A power benchmarking framework for network devices", in *NETWORKING 2009*, Springer, 2009, pp. 795–808 (cit. on pp. 54, 55).

[117] L. Barroso and U. Holzle, "The Case for Energy-Proportional Computing", *Computer*, vol. 40, no. 12, pp. 33–37, 2007, ISSN: 0018-9162. DOI: 10.1109/MC.2007.443 (cit. on p. 55).

[118] Y. Shang, D. Li, and M. Xu, "A Comparison Study of Energy Proportionality of Data Center Network Architectures", in *32nd International Conference on Distributed Computing Systems Workshops (ICDCSW)*, 2012, pp. 1–7. DOI: 10.1109/ICDCSW.2012.17 (cit. on p. 55).

[119] G. Varsamopoulos and S. K. S. Gupta, “Energy Proportionality and the Future: Metrics and Directions”, in *39th International Conference on Parallel Processing Workshops (ICPPW)*, 2010, pp. 461–467. DOI: 10.1109/ICPPW.2010.68 (cit. on p. 55).

[120] P. Fan, J. Wang, Z. Zheng, and M. Lyu, “Toward Optimal Deployment of Communication-Intensive Cloud Applications”, in *IEEE International Conference on Cloud Computing (CLOUD)*, 2011, pp. 460–467. DOI: 10.1109/CLOUD.2011.54 (cit. on p. 56).

[121] D. Boru, D. Kliazovich, F. Granelli, P. Bouvry, and A. Y. Zomaya, “Energy-efficient data replication in cloud computing datacenters”, *Springer Cluster Computing*, vol. 18, no. 1, pp. 385–402, 2015 (cit. on p. 58).

[122] *Cisco Data Center Infrastructure 2.5 Design Guide*, https://www.cisco.com/application/pdf/en/us/guest/netsol/ns107/c649/ccmigration_09186a008073377d.pdf, 2007 (cit. on p. 58).

[123] T. Benson, A. Akella, and D. A. Maltz, “Network traffic characteristics of data centers in the wild”, in *Proceedings of the 10th ACM SIGCOMM Conference on Internet measurement*, ACM, 2010, pp. 267–280 (cit. on p. 62).

[124] T. Benson, A. Anand, A. Akella, and M. Zhang, “Understanding data center traffic characteristics”, *ACM SIGCOMM Computer Communication Review*, vol. 40, no. 1, pp. 92–99, 2010 (cit. on p. 62).

[125] S. Kandula, S. Sengupta, A. Greenberg, P. Patel, and R. Chaiken, “The nature of data center traffic: Measurements & analysis”, in *Proceedings of the 9th ACM SIGCOMM Conference on Internet measurement*, ACM, 2009, pp. 202–208 (cit. on p. 62).

[126] Y. Chen, S. Jain, V. K. Adhikari, Z.-L. Zhang, and K. Xu, “A first look at inter-data center traffic characteristics via Yahoo! datasets”, in *INFOCOM, 2011 Proceedings IEEE*, IEEE, 2011, pp. 1620–1628 (cit. on p. 62).

[127] H. Cui, D. Rasooly, M. R. N. Ribeiro, and L. Kazovsky, “Optically cross-braced hypercube: A reconfigurable physical layer for interconnects and server-centric datacenters”, in *Optical Fiber Communication Conference and Exposition (OFC/NFOEC), 2012 and the National Fiber Optic Engineers Conference*, Mar. 2012, pp. 1–3 (cit. on p. 62).

[128] *Dell PowerEdge R720 Specification Sheet*, http://www.dell.com/downloads/global/products/pedge/dell-poweredge-r720-spec-sheet.pdf, 2012 (cit. on p. 63).

[129] *Huawei Tecal RH2288H V2*, http://enterprise.huawei.com/en/products/itapp/server/rh-series-rack-servers/hw-261421.htm, 2013 (cit. on p. 63).

[130] *IBM System x3500 M4*, http://www-03.ibm.com/systems/x/hardware/tower/x3500m4/specs.html, 2013 (cit. on p. 63).

[131] *Server power and performance characteristics*, http://www.spec.org/power_ssj2008/, 2014 (cit. on p. 63).

[132] *HP FlexFabric 11900 switch series quickspecs*, http://h18000.www1.hp.com/products/quickspecs/14585_div/14585_div.pdf, 2013 (cit. on p. 64).

[133] *HP 5900 switch series quickspecs*, http://h18004.www1.hp.com/products/quickspecs/14252_div/14252_div.pdf, 2013 (cit. on p. 64).

[134] *Scientific Atlanta - Cisco company - PRISMA II Ancillary Modules*, http://www.cisco.com/c/dam/en/us/products/collateral/video/prisma-ii/product_data_sheet0900aecd806c4b0e.pdf, 2008 (cit. on p. 64).

[135] *Calculating optical fiber latency*, http://www.m2optics.com/blog/?Tag=refractive%20index, 2012 (cit. on p. 67).

[136] *Data Set for IMC 2010 Data Center Measurement*, http://pages.cs.wisc.edu/~tbenson/IMC10_Data.html, 2010 (cit. on p. 67).

[137] *VMware vCenter Log Insight*, https://my.vmware.com/web/vmware/evalcenter?p=log-insight, 2014 (cit. on p. 69).

[138] *Cisco Prime Data Center Network Manager*, http://www.cisco.com/c/en/us/products/cloud-systems-management/prime-data-center-network-manager/index.html, 2014 (cit. on p. 69).

[139] H. Falaki, D. Lymberopoulos, R. Mahajan, S. Kandula, and D. Estrin, "A first look at traffic on smartphones", in *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement*, ser. IMC '10, Melbourne, Australia: ACM, 2010, pp. 281–287, ISBN: 978-1-4503-0483-2. DOI: 10.1145/1879141.1879176 (cit. on pp. 72, 74).

[140] *Worldwide sales of smartphones*, Gartner, 2016. [Online]. Available: http://www.gartner.com/newsroom/id/3215217 (cit. on p. 72).

[141] *Worldwide sales of wearables*, Gartner, 2016. [Online]. Available: http://www.gartner.com/newsroom/id/3198018 (cit. on p. 72).

[142] BBC Research, *Wearable computing: Technologies, applications and global markets*, Feb. 2014 (cit. on pp. 72, 79).

[143] M. Aazam and E.-N. Huh, "Fog computing micro datacenter based dynamic resource estimation and pricing model for IoT", in *IEEE 29th International Conference on Advanced Information Networking and Applications (AINA)*, Mar. 2015, pp. 687–694. DOI: 10.1109/AINA.2015.254 (cit. on p. 73).

[144] R. Deng, R. Lu, C. Lai, T. H. Luan, and H. Liang, "Optimal workload allocation in fog-cloud computing towards balanced delay and power consumption", *IEEE Internet of Things Journal*, 2016, ISSN: 2327-4662. DOI: 10.1109/JIOT.2016.2565516 (cit. on p. 73).

[145] S. Ningning, G. Chao, A. Xingshuo, and Z. Qiang, "Fog computing dynamic load balancing mechanism based on graph repartitioning", *China Communications*, vol. 13, no. 3, pp. 156–164, Mar. 2016, ISSN: 1673-5447 (cit. on p. 73).

[146] I. Stojmenovic and S. Wen, "The fog computing paradigm: Scenarios and security issues", in *Federated Conference on Computer Science and Information Systems (FedCSIS)*, Sep. 2014, pp. 1–8. DOI: 10.15439/2014F503 (cit. on p. 73).

[147] Y. Wang, T. Uehara, and R. Sasaki, "Fog computing: Issues and challenges in security and forensics", in *IEEE 39th Annual Computer Software and Applications Conference (COMPSAC)*, vol. 3, Jul. 2015, pp. 53–59. DOI: 10.1109/COMPSAC.2015.173 (cit. on p. 73).

[148] Y. Shi, G. Ding, H. Wang, H. E. Roman, and S. Lu, "The fog computing service for healthcare", in *2nd International Symposium on Future Information and Communication Technologies for Ubiquitous HealthCare (Ubi-HealthTech)*, May 2015, pp. 1–5. DOI: `10.1109/Ubi-HealthTech.2015.7203325` (cit. on p. 73).

[149] M. Hassanalieragh, A. Page, T. Soyata, G. Sharma, M. Aktas, G. Mateos, B. Kantarci, and S. Andreescu, "Health monitoring and management using internet-of-things (iot) sensing with cloud-based processing: Opportunities and challenges", in *IEEE International Conference on Services Computing (SCC)*, Jun. 2015, pp. 285–292. DOI: `10.1109/SCC.2015.47` (cit. on p. 73).

[150] N. I. M. Enzai and M. Tang, "A taxonomy of computation offloading in mobile cloud computing", in *2nd IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud)*, Apr. 2014, pp. 19–28. DOI: `10.1109/MobileCloud.2014.16` (cit. on p. 74).

[151] A Khandekar, N. Bhushan, J. Tingfang, and V. Vanghi, "LTE-Advanced: Heterogeneous networks", in *2010 European Wireless Conference (EW)*, Apr. 2010, pp. 978–982. DOI: `10.1109/EW.2010.5483516` (cit. on p. 75).

[152] M. Siegrist, T. C. Earle, H. Gutscher, and C. Keller, "Perception of mobile phone and base station risks", *Risk Analysis*, vol. 25, no. 5, pp. 1253–1264, 2005 (cit. on p. 75).

[153] K. Lee, J. Lee, Y. Yi, I. Rhee, and S. Chong, "Mobile data offloading: How much can WiFi deliver?", *IEEE/ACM Transactions on Networking*, vol. 21, no. 2, pp. 536–550, Apr. 2013, ISSN: 1063-6692. DOI: `10.1109/TNET.2012.2218122` (cit. on p. 75).

[154] A. Balasubramanian, R. Mahajan, and A. Venkataramani, "Augmenting mobile 3G using WiFi", in *8th International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys'10, San Francisco, California, USA: ACM, 2010, pp. 209–222, ISBN: 978-1-60558-985-5. DOI: `10.1145/1814433.1814456` (cit. on p. 75).

[155] S. Dimatteo, P. Hui, B. Han, and V. O. K. Li, "Cellular traffic offloading through WiFi networks", in *IEEE International Conference on Mobile Ad hoc and Sensor Systems (MASS)*, 2011, pp. 192–201, ISBN: 978-1-4577-1345-3 (cit. on p. 75).

[156] V. Sciancalepore, D. Giustiniano, A. Banchs, and A. Picu, "Offloading cellular traffic through opportunistic communications: Analysis and optimization", *CoRR*, vol. abs/1405.3548, 2014 (cit. on p. 75).

[157] Y. Zhu, C. Zhang, and Y. Wang, "Mobile data delivery through opportunistic communications among cellular users: A case study for the D4D challenge", *Third International Conference on the Analysis of Mobile Phone Datasets (NetMob)*, May 2013 (cit. on p. 75).

[158] B. Han, P. Hui, V. Kumar, M. Marathe, J. Shao, and A Srinivasan, "Mobile data offloading through opportunistic communications and social participation", *IEEE Transactions on Mobile Computing*, vol. 11, no. 5, pp. 821–834, May 2012, ISSN: 1536-1233. DOI: `10.1109/TMC.2011.101` (cit. on p. 75).

[159] J. Huang, F. Qian, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck, "A close examination of performance and power characteristics of 4G LTE networks", in *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '12, Low Wood Bay, Lake District, UK: ACM, 2012, pp. 225–238 (cit. on pp. 75, 85, 121).

[160] R. Friedman, A Kogan, and Y. Krivolapov, "On power and throughput tradeoffs of WiFi and Bluetooth in smartphones", *IEEE Transactions on Mobile Computing*, vol. 12, no. 7, pp. 1363–1376, Jul. 2013, ISSN: 1536-1233. DOI: 10.1109/TMC.2012.117 (cit. on pp. 75, 121).

[161] X. Zhuo, W. Gao, G. Cao, and S. Hua, "An incentive framework for cellular traffic offloading", *IEEE Transactions on Mobile Computing*, vol. 13, no. 3, pp. 541–555, Mar. 2014, ISSN: 1536-1233. DOI: 10.1109/TMC.2013.15 (cit. on p. 75).

[162] H. Tout, C. Talhi, N. Kara, and A. Mourad, "Selective mobile cloud offloading to augment multi-persona performance and viability", *IEEE Transactions on Cloud Computing*, vol. PP, no. 99, pp. 1–1, 2016, ISSN: 2168-7161. DOI: 10.1109/TCC.2016.2535223 (cit. on p. 76).

[163] S. Mahmoodi, K. Subbalakshmi, and V. Sagar, "Cloud offloading for multi-radio enabled mobile devices", in *IEEE International Conference on Communications (ICC)*, Jun. 2015, pp. 5473–5478. DOI: 10.1109/ICC.2015.7249194 (cit. on p. 76).

[164] R. Wolski, S. Gurun, C. Krintz, and D. Nurmi, "Using bandwidth data to make computation offloading decisions", in *IEEE International Symposium on Parallel and Distributed Processing (IPDPS)*, Apr. 2008, pp. 1–8. DOI: 10.1109/IPDPS.2008.4536215 (cit. on p. 76).

[165] H. Wu, W. Knottenbelt, and K. Wolter, "Analysis of the energy-response time tradeoff for mobile cloud offloading using combined metrics", in *27th International Teletraffic Congress (ITC 27)*, Sep. 2015, pp. 134–142. DOI: 10.1109/ITC.2015.23 (cit. on p. 76).

[166] S. Deng, L. Huang, J. Taheri, and A. Y. Zomaya, "Computation offloading for service workflow in mobile cloud computing", *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 12, pp. 3317–3329, Dec. 2015, ISSN: 1045-9219. DOI: 10.1109/TPDS.2014.2381640 (cit. on p. 76).

[167] A. Nodari, J. Nurminen, and M. Siekkinen, "Energy-efficient position tracking via trajectory modeling", in *10th International Workshop on Mobility in the Evolving Internet Architecture*, ser. MobiArch '15, Paris, France: ACM, 2015, pp. 33–38, ISBN: 978-1-4503-3695-6. DOI: 10.1145/2795381.2795385 (cit. on p. 76).

[168] L. D. Pedrosa, N. Kothari, R. Govindan, J. Vaughan, and T. Millstein, "The case for complexity prediction in automatic partitioning of cloud-enabled mobile applications", *Small*, vol. 20, p. 25, 2012 (cit. on p. 76).

[169] A. P. Miettinen and J. K. Nurminen, "Energy efficiency of mobile clients in cloud computing", in *2nd USENIX Conference on Hot Topics in Cloud Computing*, ser. HotCloud'10, Boston, MA, 2010, pp. 4–4 (cit. on p. 76).

[170] M. Segata, B. Bloessl, C. Sommer, and F. Dressler, “Towards energy efficient smart phone applications: Energy models for offloading tasks into the cloud”, in *IEEE International Conference on Communications (ICC)*, Jun. 2014, pp. 2394–2399. DOI: 10.1109/ICC.2014.6883681 (cit. on p. 76).

[171] M. Altamimi, A. Abdrabou, K. Naik, and A. Nayak, “Energy cost models of smartphones for task offloading to the cloud”, *IEEE Transactions on Emerging Topics in Computing*, vol. 3, no. 3, pp. 384–398, Sep. 2015, ISSN: 2168-6750 (cit. on p. 76).

[172] K. Fekete, K. Csorba, B. Forstner, T. Vajk, M. Fehér, and I. Albert, “Analyzing computation offloading energy-efficiency measurements”, in *IEEE International Conference on Communications Workshops (ICC)*, Jun. 2013, pp. 301–305. DOI: 10.1109/ICCW.2013.6649248 (cit. on p. 76).

[173] E. Cuervo, A. Balasubramanian, D.-k. Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl, “Maui: Making smartphones last longer with code offload”, in *8th International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys ’10, San Francisco, California, USA: ACM, 2010, pp. 49–62, ISBN: 978-1-60558-985-5. DOI: 10.1145/1814433.1814441 (cit. on p. 77).

[174] B.-G. Chun, S. Ihm, P. Maniatis, M. Naik, and A. Patti, “Clonecloud: Elastic execution between mobile device and cloud”, in *Proceedings of the Sixth Conference on Computer Systems*, ser. EuroSys ’11, Salzburg, Austria: ACM, 2011, pp. 301–314, ISBN: 978-1-4503-0634-8. DOI: 10.1145/1966445.1966473 (cit. on p. 77).

[175] Y. Wen, W. Zhang, and H. Luo, “Energy-optimal mobile application execution: Taming resource-poor mobile devices with cloud clones”, in *IEEE INFOCOM*, Mar. 2012, pp. 2716–2720. DOI: 10.1109/INFCOM.2012.6195685 (cit. on p. 77).

[176] R. K. K. Ma, K. T. Lam, and C. L. Wang, “Excloud: Transparent runtime support for scaling mobile applications in cloud”, in *International Conference on Cloud and Service Computing (CSC)*, Dec. 2011, pp. 103–110. DOI: 10.1109/CSC.2011.6138505 (cit. on p. 77).

[177] S. Kosta, A. Aucinas, P. Hui, R. Mortier, and X. Zhang, “Thinkair: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading”, in *IEEE INFOCOM*, Mar. 2012, pp. 945–953. DOI: 10.1109/INFCOM.2012.6195845 (cit. on p. 77).

[178] R. Kemp, N. Palmer, T. Kielmann, and H. Bal, “Cuckoo: A computation offloading framework for smartphones”, in *Mobile Computing, Applications, and Services: Second International ICST Conference, MobiCASE*, M. Gris and G. Yang, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 59–79. DOI: 10.1007/978-3-642-29336-8_4 (cit. on p. 78).

[179] B.-G. Chun and P. Maniatis, “Dynamically partitioning applications between weak devices and clouds”, in *1st ACM Workshop on Mobile Cloud Computing &#38; Services: Social Networks and Beyond*, ser. MCS ’10, San Francisco, California: ACM, 2010, pp. 1–5, ISBN: 978-1-4503-0155-8. DOI: 10.1145/1810931.1810938 (cit. on p. 78).

[180] H. B. Prajapati and V. A. Shah, “Advance reservation based DAG application scheduling simulator for grid environment”, *CoRR*, vol. abs/1211.1447, 2012. [Online]. Available: `http://arxiv.org/abs/1211.1447` (cit. on p. 79).

[181] O. Sinnen, *Task scheduling for parallel systems*. Wiley-Interscience, 2007, ISBN: 0471735760 (cit. on p. 79).

[182] S. G. Ahmad, C. S. Liew, M. M. Rafique, E. U. Munir, and S. U. Khan, “Data-intensive workflow optimization based on application task graph partitioning in heterogeneous computing systems”, in *IEEE Fourth International Conference on Big Data and Cloud Computing (BdCloud)*, Dec. 2014, pp. 129–136. DOI: `10.1109/BDCloud.2014.63` (cit. on p. 79).

[183] L. Yang, J. Cao, S. Tang, T. Li, and A. T. S. Chan, “A framework for partitioning and execution of data stream applications in mobile cloud computing”, in *IEEE 5th International Conference on Cloud Computing (CLOUD)*, Jun. 2012, pp. 794–802. DOI: `10.1109/CLOUD.2012.97` (cit. on p. 79).

[184] L. Yang, J. Cao, S. Tang, D. Han, and N. Suri, “Run time application repartitioning in dynamic mobile cloud environments”, *IEEE Transactions on Cloud Computing*, 2014, ISSN: 2168-7161. DOI: `10.1109/TCC.2014.2358239` (cit. on p. 79).

[185] J. Barrameda and N. Samaan, “A novel statistical cost model and an algorithm for efficient application offloading to clouds”, *IEEE Transactions on Cloud Computing*, 2016, ISSN: 2168-7161. DOI: `10.1109/TCC.2015.2513404` (cit. on p. 79).

[186] Weareable. (2014). 50 wearable gamechangers for 2015, [Online]. Available: `http://www.wareable.com/wearable-watchlist/50-best-wearable-tech` (cit. on p. 79).

[187] Juniper Research, *Digital advertising: Online, mobile & wearables 2015-2019*, 2015 (cit. on p. 79).

[188] K. Tehrani and M. Andrew. (Mar. 2014). Wearable technology and wearable devices: Everything you need to know., [Online]. Available: `http://www.wearabledevices.com/what-is-a-wearable-device` (cit. on p. 80).

[189] F. Schlachter, “No Moore’s law for batteries”, *Proceedings of the National Academy of Sciences*, vol. 110, no. 14, p. 5273, 2013. DOI: `10.1073/pnas.1302988110` (cit. on p. 80).

[190] K. Kumar, J. Liu, Y.-H. Lu, and B. Bhargava, “A survey of computation offloading for mobile systems”, *Mobile Network Applications*, vol. 18, no. 1, pp. 129–140, Feb. 2013, ISSN: 1383-469X (cit. on p. 80).

[191] A. Carroll and G. Heiser, “The systems hacker’s guide to the galaxy: Energy usage in a modern smartphone”, in *Asia-Pacific Workshop on Systems (APSys)*, ACM, Jul. 2013, p. 7, ISBN: 10.1145/2500727.2500734 (cit. on pp. 82, 88).

[192] M. Y. Malik, “Power consumption analysis of a modern smartphone”, *CoRR*, vol. abs/1212.1896, 2012. [Online]. Available: `http://arxiv.org/abs/1212.1896` (cit. on pp. 82, 90).

[193] A. Garcia-Saavedra, P. Serrano, A. Banchs, and G. Bianchi, “Energy consumption anatomy of 802.11 devices and its implication on modeling and design”, in *Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies*, ser. CoNEXT ’12, Nice, France: ACM, 2012, pp. 169–180 (cit. on pp. 85, 138).

[194] Verizon Enterprise Solutions. (2015). IP latency statistics, [Online]. Available: `http://www.verizonenterprise.com/about/network/latency/` (cit. on p. 86).

[195] T. Soyata, R. Muraleedharan, C. Funai, M. Kwon, and W. Heinzelman, “Cloud-vision: Real-time face recognition using a mobile-cloudlet-cloud acceleration architecture”, in *IEEE Symposium on Computers and Communications (ISCC)*, Jul. 2012, pp. 59–66 (cit. on pp. 88, 93).

[196] R. LiKamWa, Z. Wang, A. Carroll, F. X. Lin, and L. Zhong, “Draining our glass: An energy and heat characterization of Google Glass”, *CoRR*, vol. abs/1404.1320, 2014. [Online]. Available: `http://arxiv.org/abs/1404.1320` (cit. on pp. 88, 90).

[197] ARM Ltd. (2014). Cortex-a9 processor, [Online]. Available: `http://www.arm.com/cortex-a9.php` (cit. on p. 90).

[198] Google. (2016). Google Glass, [Online]. Available: `https://www.google.com/glass/start/` (cit. on p. 90).

[199] LG Corporation. (2014). LG NEXUS 5 - Technical Specifications, [Online]. Available: `http://www.lg.com/uk/mobile-phones/lg-D821/technical-specifications` (cit. on p. 90).

[200] Intel Corporation. (2012). Intel® Core™ i7-3770K Processor, [Online]. Available: `http://ark.intel.com/products/65523` (cit. on p. 90).

[201] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld, “Face recognition: A literature survey”, *ACM Comput. Surv.*, vol. 35, no. 4, pp. 399–458, Dec. 2003, ISSN: 0360-0300. DOI: `10.1145/954339.954342` (cit. on p. 93).

[202] N. Powers, A. Alling, K. Osolinsky, T. Soyata, M. Zhu, H. Wang, H. Ba, W. Heinzelman, J. Shi, and M. Kwon, “The cloudlet accelerator: Bringing mobile-cloud face recognition into real-time”, in *2015 IEEE Globecom Workshops*, Dec. 2015, pp. 1–7 (cit. on p. 93).

[203] G. Demisse, D. Aouada, and B. Ottersten, “Similarity metric for curved shapes in euclidean space”, in *IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2016 (cit. on p. 93).

[204] J. Valdivia Bedregal, R. Arisaca M, and E. Castro Gutierrez, “Optimizing energy consumption per application in mobile devices”, in *International Conference on Information Society*, Jun. 2013, pp. 106–110 (cit. on p. 104).

[205] G. P. Perrucci, F. H. P. Fitzek, and J. Widmer, “Survey on energy consumption entities on the smartphone platform”, in *IEEE 73rd Vehicular Technology Conference (VTC Spring)*, May 2011, pp. 1–6. DOI: `10.1109/VETECS.2011.5956528` (cit. on p. 104).

[206] B. Guo, Z. Yu, X. Zhou, and D. Zhang, “From participatory sensing to mobile crowd sensing”, in *IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*, Mar. 2014, pp. 593–598. DOI: 10.1109/PerComW.2014.6815273 (cit. on p. 118).

[207] N. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. Campbell, “A survey of mobile phone sensing”, *IEEE Communications Magazine*, vol. 48, no. 9, pp. 140–150, Sep. 2010, ISSN: 0163-6804. DOI: 10.1109/MCOM.2010.5560598 (cit. on pp. 118, 119).

[208] W. Khan, Y. Xiang, M. Aalsalem, and Q. Arshad, “Mobile phone sensing systems: A survey”, *IEEE Communications Surveys Tutorials*, vol. 15, no. 1, pp. 402–427, 2013, ISSN: 1553-877X. DOI: 10.1109/SURV.2012.031412.00077 (cit. on pp. 118–120).

[209] X. Zhang, Z. Yang, W. Sun, Y. Liu, S. Tang, K. Xing, and X. Mao, “Incentives for mobile crowd sensing: A survey”, *IEEE Communications Surveys Tutorials*, vol. 18, no. 1, pp. 54–67, 2016, ISSN: 1553-877X. DOI: 10.1109/COMST.2015.2415528 (cit. on p. 118).

[210] L. Jaimes, I. Vergara-Laurens, and A. Raij, “A survey of incentive techniques for mobile crowd sensing”, *IEEE Internet of Things Journal*, vol. 2, no. 5, pp. 370–380, Oct. 2015, ISSN: 2327-4662. DOI: 10.1109/JIOT.2015.2409151 (cit. on pp. 118, 120).

[211] L. Pournajaf, L. Xiong, D. A. Garcia-Ulloa, and V. Sunderam, “A survey on privacy in mobile crowd sensing task management”, *Technical Report TR-2014-002, Department of Mathematics and Computer Science, Emory University*, 2014 (cit. on pp. 118, 120).

[212] D. Christin, A. Reinhardt, S. S. Kanhere, and M. Hollick, “A survey on privacy in mobile participatory sensing applications”, *Journal of Systems and Software*, vol. 84, no. 11, pp. 1928–1946, 2011 (cit. on pp. 118, 120).

[213] B. Kantarci and H. T. Mouftah, “Trustworthy sensing for public safety in cloud-centric Internet of Things”, *IEEE Internet of Things Journal*, vol. 1, no. 4, pp. 360–368, Aug. 2014, ISSN: 2327-4662. DOI: 10.1109/JIOT.2014.2337886 (cit. on pp. 119, 121).

[214] G. Cardone, A. Cirri, A. Corradi, and L. Foschini, “The participact mobile crowd sensing living lab: The testbed for smart cities”, *IEEE Communications Magazine*, vol. 52, no. 10, pp. 78–85, Oct. 2014, ISSN: 0163-6804. DOI: 10.1109/MCOM.2014.6917406 (cit. on p. 119).

[215] Google Inc., *Google science journal*, 2016. [Online]. Available: https://makingscience.withgoogle.com/science-journal/ (cit. on p. 119).

[216] Y. Han, Y. Zhu, and J. Yu, “A distributed utility-maximizing algorithm for data collection in mobile crowd sensing”, in *IEEE Global Communications Conference (GLOBECOM)*, Austin, USA, Dec. 2014, pp. 277–282 (cit. on p. 120).

[217] H. Ma, D. Zhao, and P. Yuan, “Opportunities in mobile crowd sensing”, *IEEE Communications Magazine*, vol. 52, no. 8, pp. 29–35, Aug. 2014, ISSN: 0163-6804. DOI: 10.1109/MCOM.2014.6871666 (cit. on p. 120).

[218] K. Farkas, G. Feher, A. Benczur, and C. Sidlo, "Crowdsending based public transport information service in smart cities", *IEEE Communications Magazine*, vol. 53, no. 8, pp. 158–165, Aug. 2015, ISSN: 0163-6804. DOI: 10.1109/MCOM.2015.7180523 (cit. on p. 120).

[219] P. Jayaraman, C. Perera, D. Georgakopoulos, and A. Zaslavsky, "Efficient opportunistic sensing using mobile collaborative platform mosden", in *9th International Conference Conference on Collaborative Computing: Networking, Applications and Worksharing (Collaboratecom)*, Oct. 2013, pp. 77–86 (cit. on p. 120).

[220] X. Zhang, Z. Yang, W. Sun, Y. Liu, S. Tang, K. Xing, and X. Mao, "Incentives for mobile crowd sensing: A survey", *IEEE Communications Surveys Tutorials*, vol. 18, no. 1, pp. 54–67, 2016, ISSN: 1553-877X. DOI: 10.1109/COMST.2015.2415528 (cit. on p. 120).

[221] K. Noyen, D. Volland, D. Wörner, and E. Fleisch, "When money learns to fly: Towards sensing as a service applications using bitcoin", *CoRR*, vol. abs/1409.5841, 2014. [Online]. Available: http://arxiv.org/abs/1409.5841 (cit. on p. 121).

[222] CEET, *The power of wireless cloud*, White Paper, 2013. [Online]. Available: http://www.ceet.unimelb.edu.au/publications/downloads/ceet-white-paper-wireless-cloud.pdf (cit. on p. 121).

[223] N. D. Lane, Y. Chon, L. Zhou, Y. Zhang, F. Li, D. Kim, G. Ding, F. Zhao, and H. Cha, "Piggyback crowdsensing (pcs): Energy efficient crowdsourcing of mobile sensor data by exploiting smartphone app opportunities", in *11th ACM Conference on Embedded Networked Sensor Systems*, Roma, Italy: ACM, 2013, 7:1–7:14, ISBN: 978-1-4503-2027-6 (cit. on p. 121).

[224] M. Pouryazdan, B. Kantarci, T. Soyata, and H. Song, "Anchor-assisted and vote-based trustworthiness assurance in smart city crowdsensing", *IEEE Access*, vol. 4, pp. 529–541, 2016, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2016.2519820 (cit. on p. 121).

[225] B. Kantarci, P. M. Glasser, and L. Foschini, "Crowdsensing with social network-aided collaborative trust scores", in *IEEE Global Communications Conference (GLOBECOM)*, Dec. 2015, pp. 1–6. DOI: 10.1109/GLOCOM.2015.7417008 (cit. on p. 121).

[226] C. Prandi, S. Ferretti, S. Mirri, and P. Salomoni, "Trustworthiness in crowdsensed and sourced georeferenced data", in *IEEE International Conference on Pervasive Computing and Communication Workshops (PerComW)*, Mar. 2015, pp. 402–407. DOI: 10.1109/PERCOMW.2015.7134071 (cit. on p. 121).

[227] R. Kohavi *et al.*, "A study of cross-validation and bootstrap for accuracy estimation and model selection", in *Ijcai*, vol. 14, 1995, pp. 1137–1145 (cit. on p. 121).

[228] L. Wang, D. Zhang, Y. Wang, C. Chen, X. Han, and A. M'hamed, "Sparse mobile crowdsensing: Challenges and opportunities", *IEEE Communications Magazine*, vol. 54, no. 7, pp. 161–167, Jul. 2016, ISSN: 0163-6804. DOI: 10.1109/MCOM.2016.7509395 (cit. on p. 121).

[229] S.-C. Son, B.-T. Lee, S. K. Ko, and K. Kang, “Hybrid sensor calibration scheme for mobile crowdsensing–based city-scale environmental measurements”, *ETRI Journal*, vol. 38, no. 3, pp. 551–558, Jun. 2016. DOI: http://dx.doi.org/10.4218/etrij.16.0115.0640 (cit. on p. 121).

[230] P. Barnaghi, M. Bermudez-Edo, and R. Tönjes, “Challenges for quality of data in smart cities”, *J. Data and Information Quality*, vol. 6, no. 2-3, pp. 1–4, Jun. 2015, ISSN: 1936-1955. DOI: 10.1145/2747881 (cit. on p. 124).

[231] C. K. Tham and T. Luo, “Quality of contributed service and market equilibrium for participatory sensing”, *IEEE Transactions on Mobile Computing*, vol. 14, no. 4, pp. 829–842, Apr. 2015, ISSN: 1536-1233. DOI: 10.1109/TMC.2014.2330302 (cit. on p. 124).

[232] D. Salvatore, L. Francesco, and S. Marco, “QoS assessment of mobile crowdsensing services”, *Journal of Grid Computing*, pp. 1–22, 2015, ISSN: 1570-7873 (cit. on p. 124).

[233] H. Xiong, D. Zhang, G. Chen, L. Wang, and V. Gauthier, “Crowdtasker: Maximizing coverage quality in piggyback crowdsensing under budget constraint”, in *IEEE International Conference on Pervasive Computing and Communications (PerCom)*, IEEE, Mar. 2015, pp. 55–62 (cit. on p. 124).

[234] X. Kang, L. Liu, and H. Ma, “Data correlation based crowdsensing enhancement for environment monitoring”, in *IEEE International Conference on Communications (ICC)*, May 2016, pp. 1–6. DOI: 10.1109/ICC.2016.7511255 (cit. on p. 124).

[235] Y. Chon, N. D. Lane, Y. Kim, F. Zhao, and H. Cha, “Understanding the coverage and scalability of place-centric crowdsensing”, in *ACM International Joint Conference on Pervasive and Ubiquitous Computing*, ser. UbiComp ’13, Zurich, Switzerland: ACM, 2013, pp. 3–12, ISBN: 978-1-4503-1770-2. DOI: 10.1145/2493432.2493498 (cit. on p. 127).

[236] F. Anjomshoa, M. Catalfamo, D. Hecker, N. Helgeland, A. Rasch, B. Kantarci, M. Erol-Kantarci, and S. Schuckers, “Mobile behaviometric framework for sociability assessment and identification of smartphone users”, in *2016 IEEE Symposium on Computers and Communication (ISCC)*, Jun. 2016, pp. 1084–1089. DOI: 10.1109/ISCC.2016.7543880 (cit. on pp. 127, 137).

[237] C. Bo, L. Zhang, X.-Y. Li, Q. Huang, and Y. Wang, “Silentsense: Silent user identification via touch and movement behavioral biometrics”, in *International Conference on Mobile Computing and Networking*, ser. MobiCom ’13, Miami, Florida, USA: ACM, 2013, pp. 187–190, ISBN: 978-1-4503-1999-7. DOI: 10.1145/2500423.2504572 (cit. on p. 127).

[238] C. Tanas and J. Herrera-Joancomartí, “Crowdsensing simulation using NS-3”, *Citizen in Sensor Networks: Second International Workshop, CitiSens 2013*, J. Nin and D. Villatoro, Eds., pp. 47–58, 2014, Springer International Publishing (cit. on p. 127).

[239] K. Farkas and I. Lendák, “Simulation environment for investigating crowdsensing based urban parking”, in *International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*, Jun. 2015, pp. 320–327. DOI: 10.1109/MTITS.2015.7223274 (cit. on p. 128).

[240] K. Mehdi, M. Lounis, A. Bounceur, and T. Kechadi, "Cupcarbon: A multi-agent and discrete event wireless sensor network design and simulation tool", in *7th International ICST Conference on Simulation Tools and Techniques*, ser. SIMUTools '14, Lisbon, Portugal, 2014, pp. 126–131, ISBN: 978-1-63190-007-5 (cit. on p. 128).

[241] K. K. Jahromi, M. Zignani, S. Gaito, and G. P. Rossi, "Simulating human mobility patterns in urban areas", *Simulation Modelling Practice and Theory*, vol. 62, pp. 137 –156, 2016, ISSN: 1569-190X. DOI: http://dx.doi.org/10.1016/j.simpat.2015.12.002 (cit. on p. 129).

[242] M. Chen, Y. Hao, Y. Li, C. F. Lai, and D. Wu, "On the computation offloading at ad hoc cloudlet: Architecture and service modes", *IEEE Communications Magazine*, vol. 53, no. 6, pp. 18–24, Jun. 2015, ISSN: 0163-6804. DOI: 10.1109/MCOM.2015.7120041 (cit. on p. 132).

[243] B. Tang, Z. Chen, G. Hefferman, T. Wei, H. He, and Q. Yang, "A hierarchical distributed fog computing architecture for big data analysis in smart cities", in *Proceedings of the ASE BigData & SocialInformatics*, Kaohsiung, Taiwan: ACM, 2015, 28:1–28:6, ISBN: 978-1-4503-3735-9 (cit. on p. 132).

[244] J. Hoydis, M. Kobayashi, and M. Debbah, "Green small-cell networks", *IEEE Vehicular Technology Magazine*, vol. 6, no. 1, pp. 37–43, Mar. 2011, ISSN: 1556-6072. DOI: 10.1109/MVT.2010.939904 (cit. on p. 133).

[245] Google Inc., *Google transit*, 2016. [Online]. Available: http://www.google.com/landing/transit/cities/index.html (cit. on p. 133).

[246] K. K. Jahromi, M. Zignani, S. Gaito, and G. P. Rossi, "Simulating human mobility patterns in urban areas", *Simulation Modelling Practice and Theory*, vol. 62, pp. 137 –156, 2016, ISSN: 1569-190X (cit. on p. 133).

[247] P. P. Jayaraman, J. B. Gomes, H. L. Nguyen, Z. S. Abdallah, S. Krishnaswamy, and A. Zaslavsky, "Scalable energy-efficient distributed data analytics for crowdsensing applications in mobile environments", *IEEE Transactions on Computational Social Systems*, vol. 2, no. 3, pp. 109–123, Sep. 2015, ISSN: 2329-924X (cit. on pp. 135, 138).

[248] N. Haderer, V. Primault, P. Raveneau, C. Ribeiro, R. Rouvoy, and S. Ben Mokhtar, "Towards a practical deployment of privacy-preserving crowd-sensing tasks", in *Proceedings of the Middleware Posters & Demos Session*, Bordeaux, France: ACM, 2014, pp. 43–44, ISBN: 978-1-4503-3220-0 (cit. on p. 135).

[249] J. J. Jussila, H. Kärkkäinen, and J. Multasuo, "Social media roles in crowd-sourcing innovation tasks in b2b-relationships", in *ISPIM Conference Proceedings*, The International Society for Professional Innovation Management (ISPIM), 2013, pp. 1–12 (cit. on p. 135).

[250] C. Robusto, "The cosine-haversine formula", *The American Mathematical Monthly*, vol. 64, no. 1, pp. 38–40, 1957 (cit. on p. 137).

[251] S. Scellato, A. Noulas, and C. Mascolo, "Exploiting place features in link prediction on location-based social networks", in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '11, San Diego, California, USA, 2011, pp. 1046–1054, ISBN: 978-1-4503-0813-7. DOI: 10.1145/2020408.2020575 (cit. on p. 137).

[252] J. Huang, F. Qian, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck, "A close examination of performance and power characteristics of 4G LTE networks", in *Proceedings of the 10th international conference on Mobile systems, applications, and services*, ACM, 2012, pp. 225–238 (cit. on p. 138).

[253] C. Fiandrino, D. Kliazovich, P. Bouvry, and A. Y. Zomaya, "Network-assisted offloading for mobile cloud applications", in *IEEE International Conference on Communications (ICC)*, Jun. 2015, pp. 5833–5838. DOI: `10.1109/ICC.2015.7249252` (cit. on p. 138).

[254] D. Mendez, M. Labrador, and K. Ramachandran, "Data interpolation for participatory sensing systems", *Pervasive and Mobile Computing*, vol. 9, no. 1, pp. 132 –148, 2013, ISSN: 1574-1192. DOI: `http://dx.doi.org/10.1016/j.pmcj.2012.11.001` (cit. on p. 140).

[255] D. Hristova, M. J. Williams, M. Musolesi, P. Panzarasa, and C. Mascolo, "Measuring urban social diversity using interconnected geo-social networks", in *Proceedings of the 25th International World Wide Web Conference (WWW)*, Montréal, Canada, Apr. 2016, ISBN: 978-1-4503-0813-7 (cit. on p. 140).

[256] *FXOS8700CQ: Digital Sensor - 3D Accelerometer + 3D Magnetometer*, `http://cache.nxp.com/files/sensors/doc/data_sheet/FXOS8700CQ.pdf?pspll=1`, Accessed March 20, 2016, 2015 (cit. on p. 140).

[257] *BMP280, Barometric Pressure Sensors*, `https://www.bosch-sensortec.com/bst/products/all_products/bmp280`, Accessed March 20, 2016, 2015 (cit. on p. 140).