Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be "used for any purpose other than private study, scholarship, or research." If a, user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of "fair use" that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select "Pages from: first page # to: last page #" on the print dialog screen

ABSTRACT

MOBILE CLOUD COMPUTING AND NETWORK FUNCTION VIRTUALIZATION FOR 5G SYSTEMS

by
Ali Al-Shuwaili

The recent growth of the number of smart mobile devices and the emergence of complex multimedia mobile applications have brought new challenges to the design of wireless mobile networks. The envisioned Fifth-Generation (5G) systems are equipped with different technical solutions that can accommodate the increasing demands for high date rate, latency-limited, energy-efficient and reliable mobile communication networks.

Mobile Cloud Computing (MCC) is a key technology in 5G systems that enables the offloading of computationally heavy applications, such as for augmented or virtual reality, object recognition, or gaming from mobile devices to cloudlet or cloud servers, which are connected to wireless access points, either directly or through finite-capacity backhaul links. Given the battery-limited nature of mobile devices, mobile cloud computing is deemed to be an important enabler for the provision of such advanced applications. However, computational tasks offloading, and due to the variability of the communication network through which the cloud or cloudlet is accessed, may incur unpredictable energy expenditure or intolerable delay for the communications between mobile devices and the cloud or cloudlet servers. Therefore, the design of a mobile cloud computing system is investigated by jointly optimizing the allocation of radio, computational resources and backhaul resources in both uplink and downlink directions. Moreover, the users selected for cloud offloading need to have an energy consumption that is smaller than the amount required for local computing, which is achieved by means of user scheduling.

Motivated by the application-centric drift of 5G systems and the advances in smart devices manufacturing technologies, new brand of mobile applications are developed that are immersive, ubiquitous and highly-collaborative in nature. For example, Augmented Reality (AR) mobile applications have inherent collaborative properties in terms of data collection in the uplink, computing at the cloud, and data delivery in the downlink. Therefore, the optimization of the shared computing and communication resources in MCC not only benefit from the joint allocation of both resources, but also can be more efficiently enhanced by sharing the offloaded data and computations among multiple users. As a result, a resource allocation approach whereby transmitted, received and processed data are shared partially among the users leads to more efficient utilization of the communication and computational resources.

As a suggested architecture in 5G systems, MCC decouples the computing functionality from the platform location through the use of software virtualization to allow flexible provisioning of the provided services. Another virtualization-based technology in 5G systems is Network Function Virtualization (NFV) which prescribes the instantiation of network functions on general-purpose network devices, such as servers and switches. While yielding a more flexible and cost-effective network architecture, NFV is potentially limited by the fact that commercial off-the-shelf hardware is less reliable than the dedicated network elements used in conventional cellular deployments. The typical solution for this problem is to duplicate network functions across geographically distributed hardware in order to ensure diversity. For that reason, the development of fault-tolerant virtualization strategies for MCC and NFV is necessary to ensure reliability of the provided services.

MOBILE CLOUD COMPUTING AND NETWORK FUNCTION VIRTUALIZATION FOR 5G SYSTEMS

by
Ali Al-Shuwaili

A Dissertation
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Electrical Engineering

Helen and John C. Hartmann
Department of Electrical and Computer Engineering

August 2017

APPROVAL PAGE

MOBILE CLOUD COMPUTING AND NETWORK FUNCTION VIRTUALIZATION FOR 5G SYSTEMS

Ali Al-Shuwaili

Dr. Osvaldo Simeone, Dissertation Advisor — Date
Professor of Electrical and Computer Engineering, NJIT

Dr. Joerg Kliewer, Committee Member — Date
Associate Professor of Electrical and Computer Engineering, NJIT

Dr. Abdallah Khreishah, Committee Member — Date
Assistant Professor of Electrical and Computer Engineering, NJIT

Dr. Bipin Rajendran, Committee Member — Date
Associate Professor of Electrical and Computer Engineering, NJIT

Dr. Gesualdo Scutari, Committee Member — Date
Associate Professor of Industrial Engineering, Purdue University

BIOGRAPHICAL SKETCH

Author: Ali Al-Shuwaili

Degree: Doctor of Philosophy

Date: August 2017

Undergraduate and Graduate Education:

- Doctor of Philosophy in Electrical Engineering,
  New Jersey Institute of Technology, Newark, NJ, 2017
- Master of Science in Communication Engineering,
  University of Technology, Baghdad, 2008
- Bachelor of Science in Electronic and Communication Engineering,
  University of Technology, Baghdad, 2005

Major: Electrical Engineering

Presentations and Publications:

A. Al-Shuwaili, A. Bagheri and O. Simeone, "Joint uplink/downlink and offloading optimization for mobile cloud computing with limited backhaul," in *Proc. IEEE Annual Conference on Information Science and Systems (CISS)*, Princeton, NJ, pp. 424-429, Mar. 2016.

A. Al-Shuwaili, O. Simeone, J. Kliewer and P. Popovski, "Coded network function virtualization: Fault tolerance via in-network coding," *IEEE Wireless Communications Letters*, vol. 5, no. 6, pp. 644-647, Dec. 2016.

A. Al-Shuwaili, O. Simeone, A. Bagheri and G. Scutari, "Joint uplink/downlink optimization for backhaul-limited mobile cloud computing with user scheduling," *IEEE Transactions on Signal and Information Processing over Networks*, to appear, 2017. [Online]: Available: http://ieeexplore.ieee.org/document/7850968/

A. Al-Shuwaili and O. Simeone, "Energy-efficient resource allocation for mobile edge computing-based augmented reality applications," *IEEE Wireless Communications Letters*, to appear, 2017. [Online]: Available: http://ieeexplore.ieee.org/document/7906521/

To my family

ACKNOWLEDGMENT

First and foremost, I would like to express my sincere gratitude to Professor Osvaldo Simeone for giving me the opportunity to join his team as well as his valuable guidance and encouragement.

I will be always thankful to Professors Gesualdo Scutari, Joerg Kliewer, Bipin Rajendran and Abdallah Khreishah for serving as committee members. Special thanks go to Professor Scutari for his valuable technical advice and also for inspiring me with his work. I'm also deeply grateful to Professor Joerg Kliewer for the motivation and support he provided throughout our collaboration.

I also would like to thank all my colleagues, members and staff at the Elisha Yegal Bar-Ness Center for Wireless Information Processing (CWiP) at the New Jersey Institute of Technology for the motivating environment.

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

Chapter **Page**

TABLE OF CONTENTS
(Continued)

Chapter **Page**

LIST OF TABLES

Table **Page**

LIST OF FIGURES

Figure **Page**

LIST OF FIGURES
(Continued)

Figure **Page**

CHAPTER 1

INTRODUCTION

1.1 Overview of Computing Solutions for 5G Systems

The recent advances in mobile hardware capability have increased the demand for the deployment of fully interactive and highly complex multimedia applications on mobile devices. Unlike the massive expansions in storage and processing power, the mobile devices still have inherited resource limitations in terms of battery capacity which prohibit the deployment of such applications [3–7]. Future Fifth-Generation (5G) systems will be based on different information and communication technologies to meet the high processing power and stringent latency requirements for these applications [8–11].

Mobile Cloud Computing (MCC) is an emerging infrastructures in 5G systems that augment the hardware capability of the mobile devices by migrating the execution of the computational intensive tasks from the mobile devices to the cloud infrastructures. However, computational tasks offloading, and due to the variability of the communication network through which the cloud is accessed, may incur unpredictable energy expenditure or intolerable delay for the communications between mobile devices and the cloud [4, 6, 12].

Driven by the virtualization of computational and storage resources of the cloud infrastructures, mobile cloud computing in 5G systems enables on-demand provisioning of these resources for the offloaded tasks through the utilization of a shared pool of general-purpose and programmable hardware. Also envisioned in 5G architecture, Network Function Virtualization (NFV) is another virtualization-based technology that prescribes the instantiation of network functions on general-purpose network devices, such as servers and switches [10, 13–15]. However, such implemen-

tations of MCC and NFV raise a reliability concern since the dynamic deployment of the general-purpose hardware is usually less reliable than the dedicated network appliances [16–18]. More details about MCC and NFV are given next, while the motivation behind the work of the dissertation is discussed in the next section.

1.1.1 Mobile Cloud Computing (MCC)

Mobile cloud computing enables the offloading of computationally heavy applications, such as for gaming, object recognition or video processing, from mobile users (MUs) to cloudlet or cloud servers, which are connected to wireless access points, either directly or through finite-capacity backhaul links [3–7]. Given the battery-limited nature of mobile devices, mobile cloud computing makes it possible to provide services, such as augmented reality, that may otherwise not be available to mobile users. Offloading may take place to remote servers in the "cloud", which is accessed by the wireless access points via backhaul links, or to "cloudlet" servers that are directly connected to an access point [7]. In the latter case, the approach is also known as mobile edge computing, which is currently seen as a key enabler of the so called tactile internet [19] and is subject to standardization efforts [9]. However, computational tasks offloading, and due to the variability of the communication network through which the cloud or cloudlet is accessed, may incur unpredictable energy expenditure or intolerable delay for the communications between mobile devices and the cloud or cloudlet servers [4, 6, 12]. Offloading to peer mobile devices is also being considered [20]. Examples of architectures that are based on mobile cloud/cloudlet computation offloading include MAUI [21], ThinkAir [22], and MobiCoRE [23], while examples of commercial application based on mobile cloud computing include Apple iCloud, Shazam, and Google Goggles [24, 25].

1.1.2 Network Function Virtualization (NFV)

Network Function Virtualization (NFV) is a novel architectural paradigm for cellular wireless networks that has been put forth within the European Telecommunications Standards Institute (ETSI) with the goal of simplifying network management, update and operation [15]. NFV decouples the Network Functions (NFs), such as baseband processing at the base stations and firewalling or routing at the core network, from the physical network equipment on which they run. This is done by leveraging virtualization technology in order to map NFs into Virtual Network Functions (VNFs) that are instantiated on Commercial Off-The-Shelf (COTS) hardware resources, such as servers, storage devices and switches [13, 14]. NFV enables an adaptive "slicing" of the available network physical resources so as to accommodate different network services, e.g., mobile broadband, machine-type or ultra-reliable communications [10]. While yielding a more flexible and cost-effective network architecture, NFV is potentially limited by the fact that commercial off-the-shelf hardware is less reliable than the dedicated network elements used in conventional cellular deployments.

1.2 Motivation of the Dissertation

The main motivation of the dissertation is to study mobile cloud computing systems with the focus on the design of energy-efficient and reliable solutions for computational tasks offloading. The motivations are mentioned in details in this section, while the related work and contributions are discussed in the next two sections, respectively.

The design of a mobile cloud computing system for multiple MUs transmitting over a shared wireless medium across multiple cells requires: (*i*) the management of interference for the *uplink*, through which MUs offload the data needed for computation in the cloud; (*ii*) the management of interference for the *downlink*, through which the outcome of the cloud computations are fed back to the MUs; (*iii*) the allocation of *backhaul resources* for communication between wireless edge

and cloud; and (*iv*) the allocation of *computing resources* at the cloud. This joint allocation is mostly motivated by the increasing importance of backhaul capacity limitations, which are well understood to be often the bottleneck in modern dense network deployments (see, e.g., [8, 26]) as well as the fact that the downlink transmission for the outcome of cloud computing is of considerable amount in most cases, e.g., for rich multimedia applications in heterogenous networks [2, 27]. To this end, joint optimization over radio, computational resources and backhaul resources and in both uplink and downlink directions is the main theme of this dissertation.

If the number of users simultaneously choose to offload their computational tasks is high, the resulting interference on the wireless channel may require an energy consumption at the mobile for wireless transmission that exceeds the energy that would be needed for local computing at some MUs. Moreover, the backhaul and computing delays may make the latency requirements for the applications impossible to satisfy, and thus the offloading infeasible. Therefore, user selection, or scheduling mechanisms need to be considered with the aim of maximizing the number of users that perform offloading while guaranteeing that the selected MUs can satisfy their latency constraints and, at the same time, consume less energy than with local computing.

A recent line of work has demonstrated that it is possible to design an energy-efficient mobile cloud computing systems under latency constraints by performing a joint optimization of the allocation of communication and computational resources [28–31]. These investigations apply to generic applications run independently by different users. However, as discussed previously, the important class of AR applications have the unique property that the applications of different users share part of the computational tasks and of the input and output data [2, 32]. Therefore, it is suggested to leverage this property to reduce communication and computation overhead via the joint optimization of communication and computational resources.

By leveraging virtualization technology, MCC and NFV enable the instantiation of computational task or network function on general-purpose hardware [10, 11, 33]. One of the key challenges for such deployment is the fact that general-purpose hardware is significantly less reliable than the dedicated network devices used in conventional network deployments [14, Section VI]. Hardware outages may in fact be caused by random failures, intentional attacks, software malfunction or disasters. This problem is motivating an emerging line of work on developing fault-tolerant virtualization strategies for NFV [16–18]. The typical solution, as summarized in [17], is to adapt to NFV well established policies introduced in the context of virtualization for data centers. These strategies are based on *overprovisioning* and *diversity*: NFs are split into multiple constituents VNFs, which are then mapped on Virtual Machines (VMs) instantiated on multiple distributed servers in order to minimize the probability of a disruptive failure as well as the mean time to recovery from a failure. This motivate us to propose a novel principle for the design of fault-tolerant NFV that moves from *diversity-based* solutions to *coded* solutions.

The previous work related to the dissertation is summarized in Section 1.3, while the contributions of this work is specified in Section 1.4.

1.3 State of the Art

In relation to the focus of the dissertation, previous studies in the literature can be grouped to the following three categories and the contribution of each work is summarized next.

1.3.1 Joint Users Scheduling and Resource Allocation for Mobile Cloud Computing

The design of a mobile cloud computing system, when optimized solely at the application layer, entails decision of whether a mobile can decrease its energy expenditure by offloading the execution of the entire application [3, 4, 34–36] or of some selected subtasks (see, e.g., [37,38]). Given that offloading requires transmission and reception on the wireless interface, a more systematic approach involves the joint optimization of offloading decisions and communication parameters, such as uplink power allocation [38,39], link and subcarrier selection [40,41], and uplink data rate [30,42].

The joint optimization of the set of subtasks to be offloaded and of the uplink transmission powers was studied in [38,39] under static channel condition. A related dynamic computation offloading approach was presented in [34,42] that determines which application subtasks are to be executed remotely given the available wireless network connectivity. Assuming a simplified execution model, in which input data can be arbitrary partitioned for separate processing, references [30,43] study the fraction of the mobile data to be offloaded to the cloud, with the rest being executed locally at the mobile device. A framework that integrates wireless energy transfer and cloud mobile computing was put forth in [44].

The works summarized thus far focus on the operation of a *single* MU. In contrast to the single-MU problem formulation, in a scenario with multiple MUs transmitting over a shared wireless medium across multiple cells, the design of a mobile cloud computing system requires: (i) the management of interference for the

uplink, through which MUs offload the data needed for computation in the cloud; (*ii*) the management of interference for the *downlink*, through which the outcome of the cloud computations are fed back to the MUs; (*iii*) the allocation of *backhaul* resources for communication between wireless edge and cloud; and (*iv*) the allocation of *computing* resources at the cloud. Furthermore, the optimization should include *user selection*, or *scheduling* mechanisms whereby the offloading users are guaranteed an energy consumption that is smaller than the amount required for local computing at the device.

The limited literature on resource allocation and offloading decisions for the multiuser case includes papers [31, 45–52]. The problem of decentralized user scheduling when modeling the aspect (*i*) of uplink interference is considered in [45,46] for single-antenna elements within a game-theoretic framework. The joint allocation of radio and computational resources is considered in [47] by accounting for the elements (*i*) and (*iv*), in the presence of multiple clouds, with the aim of maximizing network operator revenue via resource pool sharing. A problem formulation including elements (*i*) and (*iv*) was studied in [48] with MIMO transceivers and for a fixed set of scheduled users. Scheduling in a single-cell was considered in [49] with the goal of minimizing the weighted sum mobile energy consumption; it was shown that the optimal scheduling and cloud resource allocation policy (element (*iv*)) have a threshold-based structure. Another scheduling strategy for multiuser offloading systems in a small-cell set-up is presented in [31], where the resources are allocated under the objective of minimizing the average latency experienced by the worst-case user by accounting for element (*iv*) with the inclusion of uplink and downlink tasks schedulers. A scheme that jointly optimizes the computation offloading decisions and the radio resource allocation in heterogeneous networks by accounting for element (*i*) so as to minimize the mobile energy expenditure under latency constraints was proposed in [50]. An energy-efficient resource allocation for interference-free

multi-users scenario was discussed in [51] with the aim of optimizing uplink and downlink transmissions duration while considering element (*iv*). Finally, the problem of scheduling tasks between cloud and edge processors was studied in [52] without modeling the physical layer.

1.3.2 Energy-efficient Resource Allocation and Cloudlet Offloading for Augmented Reality Mobile Applications

Augmented Reality (AR) applications have emerged as a way for enriching human perception by combining physical reality with artificial data [53–55], which have various practical purposes, e.g., in navigation [56, 57], touring [58, 59], entertainment [60–62] and personal assistance [63, 64]. The envisioned 5G architecture holds the promise of allowing such immersive mobile applications to be efficiently deployed on the resource-constrained mobile devices over wireless networks. Just recently, the concept of deploying AR applications over mobile wireless networks have gained an increasing attention [53, 54, 65].

AR applications are computational-intensive and delay-sensitive, and their execution on mobile devices is generally prohibitive when satisfying users' expectations in terms of battery lifetime [2, 12, 66]. To address this problem, it has been proposed to leverage mobile cloud computing [27, 67–69] and mobile edge, or cloudlet computing [2, 27, 32, 70]. Accordingly, users can offload the execution of the most time- and energy-consuming computations of AR applications to cloud servers via wireless access points and backhaul links or to cloudlet servers directly. The use of local cloudlet servers is instrumental in providing the required real-time experience, as it forgoes the use of backhaul network to access a distant cloud server [2, 12, 70].

Nevertheless, the stringent delay requirements pose significant challenges to the offloading of AR application via mobile edge computing [2, 70–72]. A recent line of work has demonstrated that it is possible to significantly reduce mobile energy consumption under latency constraints by performing a joint optimization

of the allocation of communication and computational resources [28–31]. These investigations apply to generic applications run independently by different users. However, AR applications have the unique property that the applications of different users share part of the computational tasks and of the input and output data [2, 32]. Most of the work related to AR applications, e.g., [2, 71, 73, 74], focus only on implementation aspects at the application layer without benefiting from the shared data and computations in a cross-layers optimization design.

1.3.3 Coded Network Function Virtualization: Fault Tolerance via In-Network Coding

The adaptive NFV-based deployment of the virtualized resources enhances user experience by allowing dynamic service provisioning of network functions as well as increases reliability by implementing fault tolerance mechanisms [13,18,33]. Practical platforms that implement NFV based on cloud computing include CloudNFV [75], THE REAL-TIME CLOUD [33] and CLOUDBAND [76].

Generally, NFV decouples the Network Functions (NFs), such as baseband processing at the base stations and firewalling or routing at the core network, from the physical network equipment on which they run. This is done by leveraging virtualization technology in order to map NFs into Virtual Network Functions (VNFs) that are instantiated on Commercial Off-The-Shelf (COTS) hardware resources, such as servers, storage devices and switches [13,14,33]. NFV enables an adaptive "slicing" of the available network physical resources so as to accommodate different network services, e.g., mobile broadband, machine-type or ultra-reliable communications [10].

Most research activity on NFV focuses on the design of mapping rules between VNFs and hardware resources via the solution of mixed integer problems (see, e.g., [77]). One of the key challenges for the adoption and deployment of NFV is the fact that COTS hardware is significantly less reliable than the dedicated network devices used in conventional network deployments [14, Section VI]. Hardware

outages may in fact be caused by random failures, intentional attacks, software malfunction or disasters. This problem is motivating an emerging line of work on developing fault-tolerant virtualization strategies for NFV [16–18]. The typical solution, as summarized in [17], is to adapt to NFV well established policies introduced in the context of virtualization for data centers. These strategies are based on *overprovisioning* and *diversity*: NFs are split into multiple constituents VNFs, which are then mapped on VMs instantiated on multiple distributed servers in order to minimize the probability of a disruptive failure as well as the mean time to recovery from a failure.

1.4 Dissertation Outline and Contributions

The main contributions of the dissertation encompass the design and the optimization of an energy-efficient and reliable resources scheduling in mobile cloud computing networks. The joint scheduling of resources is considered first for generic mobile application and without regards to the implementation aspects in Section 1.4.1. Then the joint allocation scheme is optimized for the offloading of specific AR-type of mobile applications in Section 1.4.2. Finally, the reliability of providing mobile cloud computing via virtualization is addressed in Section 1.4.3.

1.4.1 Joint Users Scheduling and Resource Allocation for Mobile Cloud Computing

Chapter 2 considers the problem of minimizing the mobile energy consumption for offloading under latency constraints over uplink and downlink precoding, uplink and downlink backhaul resource allocation, as well as cloud computing resource allocation for general multi-antenna transceivers. Unlike previous work that consider joint allocation of radio and computational resources, see Section 1.3.1, the problem formulation in Section 2.2 explicitly models the optimization of downlink communication for downloading the outcome of the optimization at the MUs as well as of the backhaul

resource allocated to the active MUs in both uplink and downlink. The resulting problem requires the development of a novel adaptation of the Successive Convex Approximation (SCA) scheme [78, 79] that accounts for downlink and backhaul transmissions as shown in Section 2.3.

In Section 2.4, the optimization of users scheduling is tackled jointly with the operation of uplink/downlink, backhaul and computational resources under the key constraint that each offloading MU should not consume more energy than that required for local computation. The resulting mixed integer problem is tackled by a means of SCA coupled with the smooth l_p-norm approximation approach [80].

A hybrid cloud-edge computing set-up is studied in Section 2.5 in which, beside a cloud server, "cloudlet" or "edge" servers are available locally at the wireless access points. The cloudlet servers are able to execute offloaded applications without incurring backhaul latency but with a generally smaller CPU frequency [7]. For the first time, the investigations here have studied the optimal task allocation between cloudlet and the cloud via SCA with regards to the uplink and downlink radio and backhaul resources as well as the computing resources at the cloud.

The impact of cooperative downlink transmission via network MIMO [81] on the achievable energy-latency trade-off by accounting for the backhaul overhead needed to deliver user data to multiple access points for transmission to the MUs is studied in Section 2.6. This has also not previously studied in the context of mobile cloud computing with backhaul limitations. Comprehensive numerical results are finally presented in Section 2.7.

The work in this chapter is based on:

- A. Al-Shuwaili, A. Bagheri and O. Simeone, "Joint uplink/downlink and offloading optimization for mobile cloud computing with limited backhaul," in *Proc. IEEE Annual Conference on Information Science and Systems (CISS)*, Princeton, NJ, pp. 424-429, Mar. 2016.
- A. Al-Shuwaili, O. Simeone, A. Bagheri and G. Scutari, "Joint uplink/downlink optimization for backhaul-limited mobile cloud computing with user scheduling,"

IEEE Transactions on Signal and Information Processing over Networks, to appear, 2017. [Online]: Available: http://ieeexplore.ieee.org/document/7850968/

1.4.2 Energy-efficient Resource Allocation and Cloudlet Offloading for Augmented reality Mobile Applications

Chapter 3 studies the problem of resource allocation for augmented reality (AR) applications that are implemented via mobile cloud/edge computing and leverages the unique collaborative transmission and processing features of AR applications. The resource allocation problem is formulated in Section 3.3 for minimizing the total mobile energy expenditure for offloading under latency constraints over communication and computation parameters by explicitly accounting for the *collaborative* nature of AR applications and tackled by means of a proposed Successive Convex Approximation (SCA) [78, 79] solution in Section 3.4. The proposed offloading solution optimizes the fraction of data transmitted by each user and thus potentially limiting the transmission of redundant information in the uplink as well as utilizes multicasting in the downlink. The gains that can be achieved by means of resource allocation as a function of the fraction of the data and computations that can be shared among users are quantified numerically in Section 3.5.

The work in this chapter is based on:

- A. Al-Shuwaili and O. Simeone, "Energy-efficient resource allocation for mobile edge computing-based augmented reality applications," *IEEE Wireless Communications Letters*, to appear, 2017. [Online]: Available: http://ieeexplore.ieee.org/document/7906521/,

1.4.3 Coded Network Function Virtualization: Fault Tolerance via In-Network Coding

A novel principle for the design of fault-tolerant NFV that moves from *diversity-based* solutions to *coded* solutions is proposed in Chapter 4. The proposed approach addresses the NF of uplink data decoding in a Cloud Radio Access Network (C-RAN) architecture, in which the baseband processing operations of the base station are carried out remotely at the "cloud" [82]. The proposed coded NFV solution that

leverages the algebraic structure of the transmitted coded data frames is introduced in Section 4.3. The performance of the coded NFV is compared to the diversity-based fault tolerance solution in Section 4.4.

The work in this chapter is based on:

- A. Al-Shuwaili, O. Simeone, J. Kliewer and P. Popovski, "Coded network function virtualization: Fault tolerance via in-network coding," *IEEE Wireless Communications Letters*, vol. 5, no. 6, pp. 644-647, Dec. 2016.,

CHAPTER 2

JOINT UPLINK/DOWNLINK OPTIMIZATION FOR BACKHAUL-LIMITED MOBILE CLOUD COMPUTING WITH USER SCHEDULING

2.1 Introduction

Mobile devices provide a limited processing capability to the local and battery-powered computing environment due to the highly constrained battery capacity. This shortcoming have prevented the deployment of computationally-demanding mobile applications such as for gaming, object recognition or video processing on mobile devices. Mobile cloud computing enables the offloading of the execution of such heavy applications from mobile devices to cloudlet or cloud servers, which are connected to wireless access points, either directly or through finite-capacity backhaul links. The optimization of the operation of a mobile cloud computing system amounts to the problem of minimizing the energy required for offloading across all users under latency constraints at the application layer. Since multiple mobile users across multiple cells are communicating over a shared wireless medium and also sharing the computational resources at the cloud, this problem requires the management of interference for both the uplink, through which users offload the data needed for computation in the cloud, and for the downlink, through which the outcome of the cloud computation are fed back to the users, as well as the allocation of backhaul resources for communication between wireless edge and cloud and of computing resources at the cloud. In this chapter, the basic system model and the adopted problem formulation is described in Section 2.2. The basic model assumes fixed user scheduling, offloading to a centralized cloud processor, and non-cooperative transmission at the wireless access points. Generalizations that address user scheduling, local computing capabilities at the access points and cooperative transmission will be treated later in Section 2.4, Section 2.5 and Section 2.6, respectively.

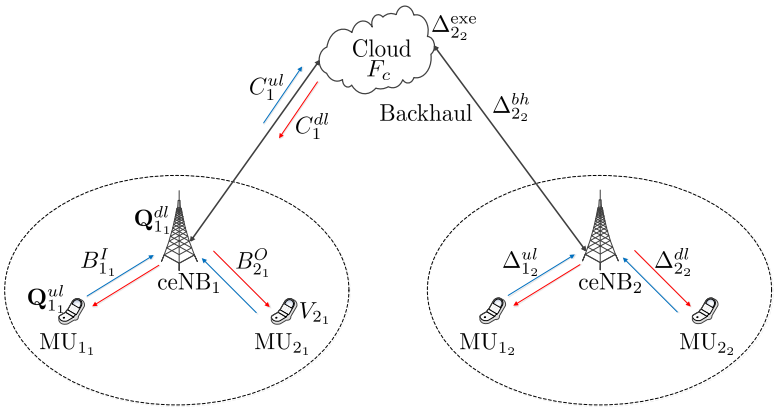


Figure 2.1 Basic system model: Mobile users (MUs) offload the execution of their applications to a centralized cloud processor through a wireless access network and finite capacity backhaul links.

2.2 General System Model and Problem Formulation

2.2.1 System Model

Consider a network composed of N_c cells of possibly different sizes such as micro- or femto-cells. Each cell $n = 1, \ldots, N_c$ includes a base station, referred to as cloud-enhanced e-Node B (ceNB) borrowing from LTE nomenclature, which is connected to a common cloud server that provides computational resources. As shown in Figure 2.1, each cell contains K active mobile users (MUs) that have been scheduled for the offloading of their local applications to the cloud processor. The K MUs in the same cell transmit in orthogonal spectral resources, in the time or frequency domain. We denote by i_n the MU in cell n that is scheduled on the i-th spectral resource, and by $\mathcal{I} \triangleq \{i_n : i = 1, \ldots, K, n = 1, \ldots, N_c\}$ the set of all the active MUs in the system. Each MU i_n and ceNB n is equipped with $N_{T_{i_n}}$ transmit and N_{R_n} receive antenna, respectively. Note that MUs in different cells that are scheduled on the same spectral resources interfere with each other.

Each MU i_n wishes to run an application within a given maximum latency T_{i_n}. The application to be executed is characterized by the number V_{i_n} of CPU cycles necessary to complete it, by the number $B_{i_n}^I$ of input bits, and by the number $B_{i_n}^O$

of output bits encoding the result of the computation. Table 2.1 summarizes the notations and parameters used in the system model.

Table 2.1 System Parameters

Parameter	Description
F_c, F_n^{ceNB}	cloud and cloudlet computational capacity
C_n^{ul}, C_n^{dl}	uplink and downlink backhaul capacity of cell n
H_{i_n}, G_{i_n}	uplink and downlink channel matrix for user i_n
$P_{i_n}^{ul}, P_n^{dl}$	uplink and downlink power budget constraint
W^{ul}, W^{dl}	uplink and downlink bandwidth
N_0	noise power spectral density
$N_{T_{i_n}}, N_{R_n}$	number of transmit and receive antenna
$B_{i_n}^I, B_{i_n}^O$	number of input and output bits for user i_n
V_{i_n}	number of CPU cycles for user i_n
T_{i_n}	latency constraint for user i_n
N_c, K	number of cells and number of users in each cell
d_{i_n}	reception energy constant for user i_n
κ	mobile device switched capacitance
α, δ	step size constant and termination accuracy

The energy and latency resulting from offloading of the applications of all active MUs are derived next. The offloading latency consist of the time $\Delta_{i_n}^{ul}$ needed for the MU to transmit the input bits to its ceNB in the uplink; the time $\Delta_{i_n}^{\text{exe}}$ necessary for the cloud to execute the instructions; the round-trip time $\Delta_{i_n}^{bh}$ for exchanging information between ceNB and the cloud through the backhaul link; and the time $\Delta_{i_n}^{dl}$ to send the result back to the MU in the downlink (see Figure 2.2). We can hence write the total offloading latency for MU i_n as

$$\Delta_{i_n} = \Delta_{i_n}^{ul} + \Delta_{i_n}^{\text{exe}} + \Delta_{i_n}^{bh} + \Delta_{i_n}^{dl}. \tag{2.1}$$

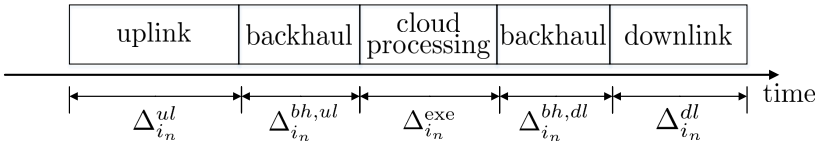


Figure 2.2 Timeline of offloading from an MU i_n. Note that the total two-way backhaul latency is given as $\Delta_{i_n}^{bh} = \Delta_{i_n}^{bh,ul} + \Delta_{i_n}^{bh,dl}$.

The energy E_{i_n} of each MU i_n instead depends only on the power used for transmission in the uplink and reception energy in the downlink. These latency and energy terms are computed as a function of the radio and computational resources as detailed next.

1) *Uplink transmission:* The optimization variables at the physical layer for the uplink are the users' transmit covariance matrices $\mathbf{Q}^{ul} \triangleq \left(\mathbf{Q}_{i_n}^{ul}\right)_{i_n \in \mathcal{I}}$, where $\mathbf{Q}_{i_n}^{ul} = \mathbb{E}[\mathbf{x}_{i_n}^{ul}\mathbf{x}_{i_n}^{ul^H}]$ with $\mathbf{x}_{i_n}^{ul} \sim \mathcal{CN}(\mathbf{0}, \mathbf{Q}_{i_n}^{ul})$ being the signal transmitted by the user i_n. These matrices are subject to power budget constraints

$$\mathcal{Q}_{i_n}^{ul} \triangleq \left\{ \mathbf{Q}_{i_n}^{ul} \in \mathbb{C}^{N_{T_{i_n}} \times N_{T_{i_n}}} : \mathbf{Q}_{i_n}^{ul} \succeq 0, \mathrm{tr}(\mathbf{Q}_{i_n}^{ul}) \leq P_{i_n}^{ul} \right\}, \tag{2.2}$$

where $P_{i_n}^{ul}$ is the maximum allowed transmit energy per symbol of MU i_n. For any given profile $\mathbf{Q}^{ul}$, the achievable transmission rate, in bits per symbol, which corresponds to the mutual information evaluated on the MIMO Gaussian channel (see, e.g., [83]) of MU i_n when multi-user interference is treated as additive Gaussian noise, is

$$r_{i_n}^{ul}(\mathbf{Q}) = \log_2 \det\left(\mathbf{I} + \mathbf{H}_{i_n}^H \mathbf{R}_n^{ul}(\mathbf{Q}_{-i_n}^{ul})^{-1} \mathbf{H}_{i_n} \mathbf{Q}_{i_n}^{ul}\right), \tag{2.3}$$

where

$$\mathbf{R}_n^{ul}(\mathbf{Q}_{-i_n}^{ul}) \triangleq N_0 \mathbf{I} + \sum_{j_m \in \mathcal{I}, m \neq n} \mathbf{H}_{j_m n} \mathbf{Q}_{j_m}^{ul} \mathbf{H}_{j_m n}^H \tag{2.4}$$

is the covariance matrix of the sum of the noise and of the inter-cell interference affecting reception at the n-th ceNB in the i-th spectral resources; $\mathbf{Q}_{-i_n}^{ul} \triangleq$

$\left(\left(\mathbf{Q}_{j_m}^{ul}\right)_{j=1}^{K}\right)_{n\neq m=1}^{N_c}$; N_0 is the noise power spectral density; and $\mathbf{H}_{i_n}$ is the uplink channel matrix for MU i_n to the ceNB in the cell n, whereas $\mathbf{H}_{j_m n}$ is the cross channel matrix between the interfering MU j_m in the cell m and the ceNB in cell n. The channel matrices account for path loss, slow and fast fading. The time, in seconds, necessary for user i in cell n to transmit the input bits $B_{i_n}^I$ to its ceNB in the uplink is then

$$\Delta_{i_n}^{ul}\left(\mathbf{Q}^{ul}\right) = \frac{B_{i_n}^I}{W^{ul} r_{i_n}^{ul}\left(\mathbf{Q}^{ul}\right)}, \tag{2.5}$$

where W^{ul} is the uplink channel bandwidth allocated to each one of the orthogonal spectral resources. The corresponding energy consumption due to uplink transmission is then

$$E_{i_n}^{ul}\left(\mathbf{Q}^{ul}\right) = B_{i_n}^I \frac{\mathrm{tr}\left(\mathbf{Q}_{i_n}^{ul}\right)}{r_{i_n}^{ul}\left(\mathbf{Q}^{ul}\right)}, \tag{2.6}$$

since around $B_{i_n}^I / r_{i_n}^{ul}\left(\mathbf{Q}^{ul}\right)$ channel uses are needed in order to transmit reliably in the uplink.

2) *Downlink transmission:* The optimization variables for the downlink are the ceNBs' transmit covariance matrices $\left(\mathbf{Q}_{i_n}^{dl}\right)_{i=1}^{K}$, which are subject to per-ceNB power constraints P_n^{dl}:

$$\mathcal{Q}_n^{dl} \triangleq \left\{ \left(\mathbf{Q}_{i_n}^{dl}\right)_{i=1}^{K} \in \mathbb{C}^{N_{T_{i_n}} \times N_{T_{i_n}}} : \mathbf{Q}_{i_n}^{dl} \succeq 0, \sum_{i=1}^{K} \mathrm{tr}(\mathbf{Q}_{i_n}^{dl}) \leq P_n^{dl} \right\}. \tag{2.7}$$

Similar to the uplink, one can write the achievable rate in bits per symbol for each MU in the downlink as

$$r_{i_n}^{dl}(\mathbf{Q}^{dl}) = \log_2 \det\left(\mathbf{I} + \mathbf{G}_{i_n}^H \mathbf{R}_n^{dl}(\mathbf{Q}_{-i_n}^{dl})^{-1} \mathbf{G}_{i_n} \mathbf{Q}_{i_n}^{dl}\right), \tag{2.8}$$

with

$$\mathbf{R}_n^{dl}(\mathbf{Q}_{-i_n}^{dl}) \triangleq N_0 \mathbf{I} + \sum_{j_m \in \mathcal{I}, m \neq n} \mathbf{G}_{j_m n} \mathbf{Q}_{j_m}^{dl} \mathbf{G}_{j_m n}^H; \tag{2.9}$$

the corresponding required transmission time reads

$$\Delta_{i_n}^{dl}\left(\mathbf{Q}^{dl}\right) = \frac{B_{i_n}^{O}}{W^{dl} r_{i_n}^{dl}\left(\mathbf{Q}^{dl}\right)}, \tag{2.10}$$

where $\mathbf{G}_{i_n}$ is the downlink channel matrix between the ceNB in cell n and the MU i_n; $\mathbf{G}_{j_m n}$ is the cross channel matrix between the interfering MU j_m in the cell m and the ceNB in cell n; W^{dl} is the downlink channel bandwidth; and $\mathbf{Q}_{-i_n}^{dl} \triangleq \left(\left(\mathbf{Q}_{j_m}^{dl}\right)_{j=1}^{K}\right)_{n\neq m=1}^{N_c}$. The downlink energy consumption is given by

$$E_{i_n}^{dl}\left(\mathbf{Q}^{dl}\right) = B_{i_n}^{O} \frac{d_{i_n}}{r_{i_n}^{dl}\left(\mathbf{Q}^{dl}\right)}, \tag{2.11}$$

where d_{i_n} is parameter that indicates the receiver energy expenditure for each symbol interval. Note that (2.7)-(2.8) implicitly assume that the downlink spectral resources are allocated to the MUs in the same way as for the uplink, so that MUs i_n for $n = 1, \ldots, N_c$ are mutually interfering in both uplink and downlink. This assumption can be easily alleviated at the cost of introducing additional notation.

3) *Cloud processing:* Let F_c be the capacity in terms of number of CPU cycles per second of the cloud; and let $f_{i_n} \geq 0$ be the fraction of the processing power F_c assigned to user i_n, so that $\sum_{i_n \in \mathcal{I}} f_{i_n} \leq 1$. The time needed to run V_{i_n} CPU cycles for user i_n remotely is then

$$\Delta_{i_n}^{\text{exe}}(f_{i_n}) = \frac{V_{i_n}}{f_{i_n} F_c}. \tag{2.12}$$

Define $\mathbf{f} \triangleq (f_{i_n})_{i_n \in \mathcal{I}}$.

4) *Backhaul transmission:* Let us denote by C_n^{ul} the capacity in bits per second of the backhaul connecting the ceNB in cell n with the cloud, and by C_n^{dl} the capacity in bits per second of the backhaul connecting the cloud with the ceNB in cell n. Let $c_{i_n}^{ul}, c_{i_n}^{dl} \geq 0$ be the fraction of the backhaul capacities C_n^{ul} and C_n^{dl}, respectively, allocated to MU i in cell n. We then have the constraint $\sum_{i=1}^{K} c_{i_n}^{ul} \leq 1$ and $\sum_{i=1}^{K} c_{i_n}^{dl} \leq 1$ for all n. Moreover, the time delay due to the backhaul transfer between ceNB n and

the cloud in both directions is given by

$$\Delta_{i_n}^{bh}(c_{i_n}^{ul}, c_{i_n}^{dl}) = \frac{B_{i_n}^{I}}{c_{i_n}^{ul} C_n^{ul}} + \frac{B_{i_n}^{O}}{c_{i_n}^{dl} C_n^{dl}}, \tag{2.13}$$

where the first term represents the latency in the uplink direction, denoted as $\Delta_{i_n}^{bh,ul}$ (cf. Figure 2.2), and the second term represents the latency in the downlink direction, denoted as $\Delta_{i_n}^{bh,dl}$ in the same figure. The uplink/downlink backhaul allocation vectors are defined as $\mathbf{c}^{ul} \triangleq (c_{i_n}^{ul})_{i_n \in \mathcal{I}}$ and $\mathbf{c}^{dl} \triangleq (c_{i_n}^{dl})_{i_n \in \mathcal{I}}$, respectively.

2.2.2 Problem Formulation

The total energy consumption due to the offloading of user i_n data, i.e., transmitting $B_{i_n}^{I}$ bits and receiving $B_{i_n}^{O}$ bits, is then given by

$$\begin{aligned} E_{i_n}\left(\mathbf{Q}^{ul}, \mathbf{Q}^{dl}\right) &= E_{i_n}^{ul}\left(\mathbf{Q}^{ul}\right) + E_{i_n}^{dl}\left(\mathbf{Q}^{dl}\right) \\ &= B_{i_n}^{I} \frac{\mathrm{tr}\left(\mathbf{Q}_{i_n}^{ul}\right)}{r_{i_n}^{ul}\left(\mathbf{Q}^{ul}\right)} + B_{i_n}^{O} \frac{d_{i_n}}{r_{i_n}^{dl}\left(\mathbf{Q}^{dl}\right)}. \end{aligned} \tag{2.14}$$

The optimal offloading problem can be stated as the minimization of the sum of the energy spent by all MUs to run their applications remotely, subject to individual latency and power constraints. Stated in mathematical terms, the problem reads

$$\begin{aligned} \min_{\mathbf{Q}^{ul}, \mathbf{Q}^{dl}, \mathbf{f}, \mathbf{c}^{ul}, \mathbf{c}^{dl}} E\left(\mathbf{Q}^{ul}, \mathbf{Q}^{dl}\right) \quad & \triangleq \sum_{i_n \in \mathcal{I}} E_{i_n}\left(\mathbf{Q}^{ul}, \mathbf{Q}^{dl}\right) \\ & = \sum_{i_n \in \mathcal{I}} B_{i_n}^{I} \frac{\mathrm{tr}\left(\mathbf{Q}_{i_n}^{ul}\right)}{r_{i_n}^{ul}\left(\mathbf{Q}^{ul}\right)} + B_{i_n}^{O} \frac{d_{i_n}}{r_{i_n}^{dl}\left(\mathbf{Q}^{dl}\right)} \\ \text{s.t.} \quad \mathbf{C.1} \quad & \frac{B_{i_n}^{I}}{W^{ul} r_{i_n}^{ul}\left(\mathbf{Q}^{ul}\right)} + \frac{B_{i_n}^{I}}{c_{i_n}^{ul} C_n^{ul}} + \frac{V_{i_n}}{f_{i_n} F_c} + \frac{B_{i_n}^{O}}{c_{i_n}^{dl} C_n^{dl}} + \frac{B_{i_n}^{O}}{W^{dl} r_{i_n}^{dl}\left(\mathbf{Q}^{dl}\right)} \leq T_{i_n}, \forall i_n \in \mathcal{I}, \\ \mathbf{C.2} \quad & f_{i_n} \geq 0, \forall i_n \in \mathcal{I}, \sum_{i_n \in \mathcal{I}} f_{i_n} \leq 1, \\ \mathbf{C.3} \quad & c_{i_n}^{ul}, c_{i_n}^{dl} \geq 0, \forall i_n \in \mathcal{I}, \sum_{i=1}^{K} c_{i_n}^{ul} \leq 1, \sum_{i=1}^{K} c_{i_n}^{dl} \leq 1, \forall n = 1, \ldots, N_c, \\ \mathbf{C.4} \quad & \mathbf{Q}_{i_n}^{ul} \in \mathcal{Q}_{i_n}^{ul}, \forall i_n \in \mathcal{I}, \left(\mathbf{Q}_{i_n}^{dl}\right)_{i=1}^{K} \in \mathcal{Q}_n^{dl}, \forall n = 1, \ldots, N_c. \end{aligned} \tag{P.1}$$

Constraint C.1 enforces that the latency for any MU i_n to be less than or equal to the maximum tolerable delay of T_{i_n} seconds; C.2 imposes the mentioned limit on the cloud computational resources; C.3 enforces the limited backhaul capacities in uplink and downlink; and C.4 guarantee that power budget constraint on the radio interface of both uplink and downlink is satisfied. Note that problem (P.1) depends only on the ratios $B^I_{i_n}/W^{ul}$, $B^I_{i_n}/C^{ul}_n$, V_{i_n}/F_c, $B^O_{i_n}/W^{dl}$ and $B^O_{i_n}/C^{dl}_n$. We denote by $\mathbf{Z} \triangleq \left(\mathbf{Q}^{ul}, \mathbf{Q}^{dl}, \mathbf{f}, \mathbf{c}^{ul}, \mathbf{c}^{dl}\right)$ the set of all optimization variables, and by $\mathcal{Z}$ the feasible set of (P.1). Note that (P.1) is non-convex, due to the non-convexity of the objective function and of the constraint C.1.

Remark 1. As discussed, the expression in C.1 for the latency of each user assumes that the communication and processing steps, including uplink transmission, edge-to-cloud backhaul transmission, cloud processing, cloud-to-edge backhaul transmission and downlink transmission take place one after the other for each user, as illustrated in Figure 2.2. Given that the uplink, backhaul, execution and downlink latencies may be different across the users, the communication steps of different users may not be aligned. While this fact may be exploited by a sophisticated receiver that tracks the variation of the interference power within a communication block, the resulting achievable rates are extremely difficult to characterize and optimize. In contrast, the expression of the uplink and downlink rates (2.3) and (2.8), in which interference is assumed to be caused by all users, are achievable by means of standard decoders (see, e.g., [84]) and can be efficiently computed and optimized, as it will be shown in Section 2.3.

Feasibility. Problem (P.1) has a non-empty feasible set if there exist matrices $\mathbf{Q}^{ul}$ and $\mathbf{Q}^{dl}$ such that the following inequalities hold

a) $T_{i_n} > \frac{B^I_{i_n}}{W^{ul} r^{ul}_{i_n}\left(\mathbf{Q}^{ul}\right)} + \frac{B^O_{i_n}}{W^{dl} r^{dl}_{i_n}\left(\mathbf{Q}^{dl}\right)}$, $\forall i_n \in \mathcal{I}$;

b) $\sum\limits_{i_n \in \mathcal{I}} \dfrac{V_{i_n}/F_c}{T_{i_n} - \frac{B^I_{i_n}}{W^{ul} r^{ul}_{i_n}\left(\mathbf{Q}^{ul}\right)} + \frac{B^O_{i_n}}{W^{dl} r^{dl}_{i_n}\left(\mathbf{Q}^{dl}\right)}} \leq \alpha_1$;

c) $\sum_{i=1}^{K} \dfrac{B_{i_n}^I / C_n^{ul}}{T_{i_n} - \frac{B_{i_n}^I}{W^{ul} r_{i_n}^{ul}(\mathbf{Q}^{ul})} + \frac{B_{i_n}^O}{W^{dl} r_{i_n}^{dl}(\mathbf{Q}^{dl})}} \leq \alpha_2, \forall n = 1, \ldots, N_c;$

d) $\sum_{i=1}^{K} \dfrac{B_{i_n}^O / C_n^{dl}}{T_{i_n} - \frac{B_{i_n}^I}{W^{ul} r_{i_n}^{ul}(\mathbf{Q}^{ul})} + \frac{B_{i_n}^O}{W^{dl} r_{i_n}^{dl}(\mathbf{Q}^{dl})}} \leq \alpha_3, \forall n = 1, \ldots, N_c;$

for some $\alpha_1, \alpha_2, \alpha_3 \geq 0$ with $\alpha_1 + \alpha_2 + \alpha_3 = 1$. In fact, if above conditions are met, one can always choose $f_{i_n} = \alpha_1^{-1}(V_{i_n}/F_c)\Big(T_{i_n} - \frac{B_{i_n}^I}{W^{ul} r_{i_n}^{ul}(\mathbf{Q}^{ul})} - \frac{B_{i_n}^O}{W^{dl} r_{i_n}^{dl}(\mathbf{Q}^{dl})}\Big)^{-1}$, and similarly for $c_{i_n}^{ul}$ and $c_{i_n}^{dl}$, so that conditions b), c) and d) are satisfied with equality.

Remark 2. When the goal is identifying the achievable trade-off curve between energy consumption and latency, assuming for simplicity that all MUs have the same latency constraint T, e.g., $T_{i_n} = T$, the following problem may also be considered

$$\begin{aligned} \min_{\mathbf{Q}^{ul}, \mathbf{Q}^{dl}, \mathbf{f}, \mathbf{c}^{ul}, \mathbf{c}^{dl}, T} \quad & E\left(\mathbf{Q}^{ul}, \mathbf{Q}^{dl}\right) + \lambda T \\ \text{s.t.} \quad & \mathbf{C.1} \quad \tfrac{B_{i_n}^I}{W^{ul} r_{i_n}^{ul}(\mathbf{Q}^{ul})} + \tfrac{B_{i_n}^I}{c_{i_n}^{ul} C_n^{ul}} + \tfrac{V_{i_n}}{f_{i_n} F_c} + \tfrac{B_{i_n}^O}{c_{i_n}^{dl} C_n^{dl}} + \tfrac{B_{i_n}^O}{W^{dl} r_{i_n}^{dl}(\mathbf{Q}^{dl})} \leq T, \forall i_n \in \mathcal{I}, \\ & \mathbf{C.2} - \mathbf{C.4} \text{ of (P.1)}, \end{aligned} \tag{P.2}$$

where $\lambda > 0$ is a parameter identifying the desired relative weight between energy and latency minimization.

Remark 3. The problem formulation (P.1) can be easily extended to account for a more general backhaul topology in which the ceNBs are connected to the cloud via a multi-hop network with predefined routes between ceNBs and cloud. We do not further elaborate on this model here, because of the space limitation.

Remark 4. Problem (P.1) was tacked in [48] for the special case where the joint optimization was over radio and computational resources and only in the uplink direction. Problem (P.1), instead, considers a general setup in which joint optimization carried over backhaul capacities in both uplink and downlink, as well

as the optimization over the downlink radio transmission from the ceNB to the MU. This generalization entails different formulations for the objective function and for the latency constraint from that in [48] and thus calls for novel SCA-based solution methods.

2.3 Successive Convex Approximation Optimization

Problem (P.1) is non-convex due to the non-convexity of the objective function and of the constraint C.1. To address this issue, one can leverage the SCA method proposed in [78, 79] for general non-convex problems. The SCA algorithm in [78, 79] is proved to converge to a stationary solution of the (NP-hard) non-convex problem by solving a sequence of convex sub-problems, each one of which can be solved in polynomial time, e.g., by interior-point methods [85]. To do so, one needs to identify convex approximants for the objective function and for the non-convex constraints C.1 that satisfy the conditions specified in [78, 79] as discussed next.

2.3.1 Convex Surrogate for the Objective Function

Define as $\mathcal{K} \supseteq \mathcal{Z}$ any compact convex set containing the feasible set $\mathcal{Z}$ such that all functions in (P.1) are well defined on it. Note that such a set always exists. This set exists by the same arguments used in [48, Section IV-B]. Let $\mathbf{Z}(v) \triangleq \left(\mathbf{Q}^{ul}(v), \mathbf{Q}^{dl}(v), \mathbf{f}(v), \mathbf{c}^{ul}(v), \mathbf{c}^{dl}(v)\right)$, with v being the current iterate index of the SCA algorithm. According to [78, 79], in order to be used within the SCA scheme, a convex approximant $\tilde{E}(\mathbf{Z}; \mathbf{Z}(v))$ of the objective function $E\left(\mathbf{Q}^{ul}, \mathbf{Q}^{dl}\right)$ around the current feasible iterate $\mathbf{Z}(v) \in \mathcal{Z}$ must satisfy the following properties ([78, Section II]):

A1: $\tilde{E}(\bullet; \mathbf{Z}(v))$ is uniformly strongly convex on $\mathcal{K}$;

A2: $\nabla_{\mathbf{Q}^{ul*}} \tilde{E}(\mathbf{Z}(v); \mathbf{Z}(v)) = \nabla_{\mathbf{Q}^{ul*}} E\left(\mathbf{Q}^{ul}(v), \mathbf{Q}^{dl}(v)\right)$ and

$\nabla_{\mathbf{Q}^{dl*}} \tilde{E}(\mathbf{Z}(v); \mathbf{Z}(v)) = \nabla_{\mathbf{Q}^{dl*}} E\left(\mathbf{Q}^{ul}(v), \mathbf{Q}^{dl}(v)\right)$, for all $\mathbf{Z}(v) \in \mathcal{Z}$;

A3: $\nabla_{\mathbf{Z}^*}\tilde{E}\left(\bullet;\bullet\right)$ is Lipschitz continuous on $\mathcal{K}\times\mathcal{Z}$;

where $\nabla_{\mathbf{x}^*}f(\mathbf{x};\mathbf{y})$ denotes the conjugate gradient of the function $f(\mathbf{x};\mathbf{y})$ with respect to its first argument $\mathbf{x}$. It is noted that, besides the convexity and smoothness conditions A1 and A3, A2 enforces that the first-order behavior of the approximant be the same as for the original function. In what follows, an approximant for the objective function (2.14) satisfying A1-A3 above is derived. Let us write

$$\begin{aligned}\tilde{E}\left(\mathbf{Z};\mathbf{Z}\left(v\right)\right) &\triangleq \sum_{i_n\in\mathcal{I}}\tilde{E}_{i_n}\left(\mathbf{Z}_{i_n};\mathbf{Z}\left(v\right)\right)+\bar{\mathrm{E}}\left(\mathbf{Z};\mathbf{Z}\left(v\right)\right)\\ &=\sum_{i_n\in\mathcal{I}}\left(\tilde{E}_{i_n}^{ul}\left(\mathbf{Q}^{ul};\mathbf{Q}^{ul}(v)\right)+\tilde{E}_{i_n}^{dl}\left(\mathbf{Q}^{dl};\mathbf{Q}^{dl}(v)\right)\right)+\bar{\mathrm{E}}\left(\mathbf{Z};\mathbf{Z}\left(v\right)\right),\end{aligned} \tag{2.15}$$

where $\mathbf{Z}_{i_n}\triangleq\left(\mathbf{Q}_{i_n}^{ul},\mathbf{Q}_{i_n}^{dl},f_{i_n},c_{i_n}^{ul},c_{i_n}^{dl}\right)$ being the vector of the optimization variables with the function $\bar{\mathrm{E}}\left(\mathbf{Z};\mathbf{Z}\left(v\right)\right)\triangleq\frac{\gamma_{q^{ul}}}{2}\sum_{i_n\in\mathcal{I}}\left\|\mathbf{Q}^{ul}-\mathbf{Q}^{ul}\left(v\right)\right\|^2+\frac{\gamma_{q^{dl}}}{2}\sum_{i_n\in\mathcal{I}}\left\|\mathbf{Q}^{dl}-\mathbf{Q}^{dl}\left(v\right)\right\|^2+\frac{\gamma_f}{2}\left\|\mathbf{f}-\mathbf{f}\left(v\right)\right\|^2+\frac{\gamma_{c^{ul}}}{2}\left\|\mathbf{c}^{ul}-\mathbf{c}^{ul}\left(v\right)\right\|^2+\frac{\gamma_{c^{dl}}}{2}\left\|\mathbf{c}^{dl}-\mathbf{c}^{dl}\left(v\right)\right\|^2$ is added to make the approximant objective function (2.15) strongly convex on $\mathcal{K}$, with $\gamma_{q^{ul}},\gamma_{q^{dl}},\gamma_f,\gamma_{c^{ul}}$ and $\gamma_{c^{dl}}$ being arbitrary positive constants (see [78, 79]).The functions $\tilde{E}_{i_n}^{ul}\left(\mathbf{Q}^{ul};\mathbf{Q}^{ul}(v)\right)$ and $\tilde{E}_{i_n}^{dl}\left(\mathbf{Q}^{dl};\mathbf{Q}^{dl}(v)\right)$ are the convex approximants for the uplink and downlink energy terms, respectively, as derived next.

1) Convex approximant for $E_{i_n}^{ul}\left(\mathbf{Q}^{ul}\right)$: It is not difficult to check that an approximant that utilize the "partial" convexity of the uplink energy function (2.6) can be obtained as (cf. [48, Section IV-B])

$$\begin{aligned}\tilde{E}_{i_n}^{ul}\left(\mathbf{Q}^{ul};\mathbf{Q}^{ul}(v)\right) &= \mathrm{tr}\left(\mathbf{Q}_{i_n}^{ul}\left(v\right)\right)\frac{B_{i_n}^I}{r_{i_n}^{ul}\left(\mathbf{Q}_{i_n}^{ul},\mathbf{Q}_{-i_n}^{ul}\left(v\right)\right)}\\ &\quad+\mathrm{tr}\left(\mathbf{Q}_{i_n}^{ul}\right)\frac{B_{i_n}^I}{r_{i_n}^{ul}\left(\mathbf{Q}_{i_n}^{ul}\left(v\right),\mathbf{Q}_{-i_n}^{ul}\left(v\right)\right)}\\ &\quad+\sum_{j_m\in\mathcal{I},m\neq n}\left\langle\nabla_{\mathbf{Q}_{i_n}^{ul}{}^*}E_{j_m}\left(\mathbf{Q}^{ul}\left(v\right)\right),\mathbf{Q}_{i_n}^{ul}-\mathbf{Q}_{i_n}^{ul}\left(v\right)\right\rangle,\end{aligned} \tag{2.16}$$

where $\langle \mathbf{A}, \mathbf{B} \rangle \triangleq \mathrm{Re}\left\{\mathrm{tr}\left(\mathbf{A}^H \mathbf{B}\right)\right\}$, and the conjugate gradient $\nabla_{\mathbf{Q}_{i_n}^{ul}*} E_{j_m}\left(\mathbf{Q}^{ul}(v)\right)$ is given by [48, eq. (18)]

$$\begin{aligned}\nabla_{\mathbf{Q}_{i_n}^{ul}*} E_{j_m}\left(\mathbf{Q}^{ul}(v)\right) = & \frac{\mathrm{tr}\left(\mathbf{Q}_{j_m}^{ul}(v)\right) \Delta_{j_m}^{ul}\left(\mathbf{Q}^{ul}(v)\right)}{\log(2)\, r_{j_m}^{ul}\left(\mathbf{Q}^{ul}(v)\right)} \\ & \cdot [\mathbf{H}_{i_n m}^H (\mathbf{R}_m^{ul}\big(\mathbf{Q}_{-j_m}^{ul}(v)\big)^{-1} - (\mathbf{R}_m^{ul}\left(\mathbf{Q}_{-j_m}^{ul}(v)\right) \\ & + \mathbf{H}_{j_m} \mathbf{Q}_{j_m}^{ul}(v)\, \mathbf{H}_{j_m}^H)^{-1}) \mathbf{H}_{i_n m}].\end{aligned} \tag{2.17}$$

2) Convex approximant for $E_{i_n}^{dl}\left(\mathbf{Q}^{dl}\right)$: To obtain the desired convex approximant of downlink energy in (2.11), one needs to construct the convex surrogate for the downlink rate function, i.e., $\tilde{r}_{i_n}^{dl}\left(\mathbf{Q}^{dl}; \mathbf{Q}^{dl}(v)\right)$. To obtain such an approximant, we exploit first the concave-convex structure of the rate functions $r_{i_n}^{dl}\left(\mathbf{Q}^{dl}\right)$

$$r_{i_n}^{dl}\left(\mathbf{Q}^{dl}\right) = \underbrace{\log_2 \det\left(\mathbf{R}_n^{dl}\left(\mathbf{Q}_{-i_n}^{dl}\right) + \mathbf{H}_{i_n}^H \mathbf{H}_{i_n} \mathbf{Q}_{i_n}^{dl}\right)}_{r_{i_n}^{dl\,+}\left(\mathbf{Q}^{dl}\right)} - \underbrace{\log_2 \det\left(\mathbf{R}_n^{dl}\left(\mathbf{Q}_{-i_n}^{dl}\right)\right)}_{r_{i_n}^{dl\,-}\left(\mathbf{Q}_{-i_n}^{dl}\right)}, \tag{2.18}$$

where $r_{i_n}^{dl\,+}\left(\mathbf{Q}^{dl}\right)$ and $r_{i_n}^{dl\,-}\left(\mathbf{Q}_{-i_n}^{dl}\right)$ are concave functions. The convex approximant is then obtained as

$$\begin{aligned}\tilde{r}_{i_n}^{dl}\left(\mathbf{Q}^{dl}; \mathbf{Q}^{dl}(v)\right) = & r_{i_n}^{dl+}\left(\mathbf{Q}^{dl}\right) - r_{i_n}^{dl-}\left(\mathbf{Q}_{-i_n}^{dl}(v)\right) \\ & - \sum_{j_m \in \mathcal{I}} \left\langle \nabla_{\mathbf{Q}_{j_m}^{ul}*} r_{i_n}^{dl\,-}\left(\mathbf{Q}_{-i_n}^{dl}(v)\right), \mathbf{Q}_{j_m}^{dl} - \mathbf{Q}_{j_m}^{dl}(v) \right\rangle,\end{aligned} \tag{2.19}$$

with

$$\nabla_{\mathbf{Q}_{j_m}^{dl}*} r_{i_n}^{dl-}\left(\mathbf{Q}_{-i_n}^{dl}(v)\right) = \mathbf{H}_{j_m n}^H \mathbf{R}_n^{dl}\big(\mathbf{Q}_{-i_n}^{dl}(v)\big)^{-1} \mathbf{H}_{j_m n}. \tag{2.20}$$

Then, we can simply obtain the convex approximant for $E_{i_n}^{dl}\left(\mathbf{Q}^{dl}\right)$ as

$$\tilde{E}_{i_n}^{dl}\left(\mathbf{Q}^{dl}; \mathbf{Q}^{dl}(v)\right) = B_{i_n}^O \frac{d_{i_n}}{\tilde{r}_{i_n}^{dl}\left(\mathbf{Q}^{dl}; \mathbf{Q}^{dl}(v)\right)}. \tag{2.21}$$

Remark 5. The key advantage of SCA [78, 79] as compared to more conventional approaches such as Difference-of-Convex (DC) programming [86] is that the convex

surrogate $\tilde{E}\left(\mathbf{Z};\mathbf{Z}\left(v\right)\right)$ need not be a global upper bound on the function $E(\mathbf{Q}^{ul},\mathbf{Q}^{dl})$ – a condition that appears to be difficult to ensure for the objective function of (P.1).

2.3.2 Inner Convexification of the Constraints

Next, let us consider the latency constraint C.1. Note that this constraint differs from the corresponding latency constraint in [48] by virtue of the contributions due to downlink and backhaul transmissions. Let us define the non-convex part of the left-hand side of C.1 as

$$g_{i_n}\left(\mathbf{Q}^{ul},\mathbf{Q}^{dl}\right)\triangleq\frac{B_{i_n}^{I}}{W^{ul}r_{i_n}^{ul}\left(\mathbf{Q}^{ul}\right)}+\frac{B_{i_n}^{O}}{W^{dl}r_{i_n}^{dl}\left(\mathbf{Q}^{dl}\right)}. \tag{2.22}$$

To apply SCA, one needs to obtain an approximant $\tilde{g}_{i_n}\big(\mathbf{Q}^{ul},\mathbf{Q}^{dl};\mathbf{Z}(v)\big)$ at the current iterate $\mathbf{Z}\left(v\right))\in\mathcal{Z}$ that satisfies the following properties ([78, Section II]):

B1: $\tilde{g}_{i_n}\big(\bullet,\mathbf{Z}(v)\big)$ is uniformly convex on $\mathcal{K}$;

B2: $\nabla_{\mathbf{Z}^*}\tilde{g}_{i_n}\big(\mathbf{Q}^{ul}\left(v\right),\mathbf{Q}^{dl}\left(v\right);\mathbf{Z}(v)\big)=\nabla_{\mathbf{Z}^*}g_{i_n}\big(\mathbf{Q}^{ul}\left(v\right),\mathbf{Q}^{dl}\left(v\right)\big)$, for all $\mathbf{Z}(v)\in\mathcal{Z}$;

B3: $\nabla_{\mathbf{Z}^*}\tilde{g}_{i_n}(\bullet;\bullet)$ is continuous on $\mathcal{K}\times\mathcal{Z}$;

B4: $\tilde{g}_{i_n}\big(\mathbf{Q}^{ul},\mathbf{Q}^{dl};\mathbf{Z}(v)\big)\geq g_{i_n}\big(\mathbf{Q}^{ul},\mathbf{Q}^{dl}\big)$, for all $\big(\mathbf{Q}^{ul},\mathbf{Q}^{dl}\big)\in\mathcal{K}$ and $\mathbf{Z}(v)\in\mathcal{Z}$;

B5: $\tilde{g}_{i_n}\big(\mathbf{Q}^{ul}\left(v\right),\mathbf{Q}^{dl}\left(v\right);\mathbf{Z}(v)\big)=g_{i_n}\big(\mathbf{Q}^{ul}\left(v\right),\mathbf{Q}^{dl}\left(v\right)\big)$, for all $\mathbf{Z}(v)\in\mathcal{Z}$;

B6: $\tilde{g}_{i_n}(\bullet;\bullet)$ is Lipschitz continuous on $\mathcal{K}\times\mathcal{Z}$.

Besides the first-order behavior and smoothness conditions B2, B3 and B6, the key assumptions B1, B4 and B5 enforce that the approximant $\tilde{g}_{i_n}\big(\mathbf{Q}^{ul},\mathbf{Q}^{dl};\mathbf{Z}(v)\big)$ be a locally tight (condition B5) convex (condition B1) upper bound (condition B4) on the original constraint $g_{i_n}\left(\mathbf{Q}^{ul},\mathbf{Q}^{dl}\right)$. The desired surrogate approximation $\tilde{g}_{i_n}\left(\mathbf{Q}^{ul},\mathbf{Q}^{dl};\mathbf{Z}(v)\right)$ is then obtained from (2.19) as

$$\tilde{g}_{i_n}\left(\mathbf{Q}^{ul},\mathbf{Q}^{dl};\mathbf{Z}(v)\right)\triangleq\frac{B_{i_n}^{I}}{W^{ul}\tilde{r}_{i_n}^{ul}\left(\mathbf{Q}^{ul};\mathbf{Q}^{ul}\left(v\right)\right)}+\frac{B_{i_n}^{O}}{W^{dl}\tilde{r}_{i_n}^{dl}\left(\mathbf{Q}^{dl};\mathbf{Q}^{dl}\left(v\right)\right)}, \tag{2.23}$$

The proposed approximant (2.23) is an upper bound (condition B4) and is a convex function of $\mathbf{Q}^{ul}$ and $\mathbf{Q}^{dl}$ (condition B1). This is because both terms in (2.23)

are the reciprocal of a concave and positive function, and the sum of the two convex functions is convex. Furthermore, it is easy to show that the original constraint (2.22) and its convex approximant (2.23) have the same first-order behavior (condition B2) by evaluating the gradient of both functions at the current iterate. The remaining properties B3-B6 can also be checked in a similar manner to [48, Section IV-B]. For instance, since the approximant functions (2.16) and (2.19) are twice continuously differentiable over the compact convex set $\mathcal{K} \supseteq \mathcal{Z}$, the Lipschitz continuity of their conjugate gradients follows readily.

2.3.3 SCA Algorithm

The SCA algorithm operates by iteratively solving the following problem around the current iterate $\mathbf{Z}(v) \in \mathcal{Z}$,

$$\begin{aligned}
&\hat{\mathbf{Z}}(\mathbf{Z}(v)) \triangleq \underset{\mathbf{Q}^{ul},\mathbf{Q}^{dl},\mathbf{f},\mathbf{c}^{ul},\mathbf{c}^{dl}}{\operatorname{argmin}} \tilde{E}(\mathbf{Z};\mathbf{Z}(v)) \\
&\qquad\qquad \text{s.t.} \\
&\mathbf{C.1} \quad \tilde{g}_{i_n}\left(\mathbf{Q}^{ul},\mathbf{Q}^{dl};\mathbf{Z}(v)\right) + \frac{B^I_{i_n}}{c^{ul}_{i_n}C^{ul}_n} + \frac{B^O_{i_n}}{c^{dl}_{i_n}C^{dl}_n} + \frac{V_{i_n}}{f_{i_n}F_c} \le T_{i_n}, \forall i_n \in \mathcal{I}, \\
&\mathbf{C.2} - \mathbf{C.4} \text{ of (P.1).}
\end{aligned} \tag{P.3}$$

The unique solution of the strongly convex optimization problem (P.3) is denoted $\hat{\mathbf{Z}}\big(\mathbf{Z}(v)\big) \triangleq \big(\hat{\mathbf{Q}}^{ul},\hat{\mathbf{Q}}^{dl},\hat{\mathbf{f}},\hat{\mathbf{c}}^{ul},\hat{\mathbf{c}}^{dl}\big)$. Note that $\tilde{E}(\mathbf{Z};\mathbf{Z}(v))$ is a function of $\mathbf{Z}$, given the current iterate $\mathbf{Z}(v)$.

The SCA scheme is summarized in Algorithm 1. In step 1, the termination criterion is $\left|E\left(\mathbf{Q}^{ul}(v+1),\mathbf{Q}^{dl}(v+1)\right) - E\left(\mathbf{Q}^{ul}(v),\mathbf{Q}^{dl}(v)\right)\right| \le \delta$, where $\delta > 0$ is the desired accuracy. The step size rule we used is $\gamma(v) = \gamma(v-1)(1-\alpha\gamma(v-1))$ with $\gamma(0) \in (0,1]$ and $\alpha \in (0, 1/\gamma(0))$ (other step size rules can also be adopted, see [78, 79]). Algorithm 1 converges to a stationary point of the problem (P.1) in the sense of [48, Theorem 2].

Remark 6. The SCA scheme can also be easily adapted to tackle the weighted sum problem (P.2) discussed in Remark 2. This alternative formulation has the key advantage that the identification of an initial feasible point $\mathbf{Z}(0) \triangleq \left(\mathbf{Q}^{ul}(0), \mathbf{Q}^{dl}(0), \mathbf{f}(0), \mathbf{c}^{ul}(0), \mathbf{c}^{dl}(0), T(0)\right)$ for the SCA is a trivial task. This is because one can always select a value of $T(0)$ that satisfies the constraint C.1 in (P.2) for given values of the other variables.

Remark 7. Each instance of the optimization problem (P.3) tackled by SCA can be solved with complexity $O(\max\{n^3, n^2 m\})$ using interior-points methods [87, Ch. 1], where n is the size of the optimization variables, namely $\left(2N_{T_{in}}^2 + 3\right) KN_c$, and m is the number of constraints, namely $m = 7KN_c + 3N_c + 1$. Note that the complexity scales polynomially with the number K of users and with all system parameters. While here we have focused on a centralized implementation, the complexity could be further reduced by developing distributed solutions as described in [78, Section IV]. Finally, we would also like to mention that, in practice, rather than solving problem (P.3) using SCA at each time slot for the given realization of the channels, it would be possible to solve the problem for a number of representative channels so as to build a sufficiently dense look-up table. More interestingly, as recently explored in [88] for power allocation in an interference channel, one could use such representative channels to train a neural network, or another learning machine, to "interpolate" the solution to other channel realizations. These aspects are left for future investigations.

2.4 Users Scheduling

In the previous section, it was assumed that a given number of active users, namely K per cell, was scheduled for transmission. The premise of this section, is that, if too many MUs simultaneously choose to offload their computational tasks, the resulting interference on the wireless channel may require an energy consumption at the mobile for wireless transmission that exceeds the energy that would be needed for

Algorithm 1: SCA Solution for (P.3)

Input: Parameters from Table 2.1; $\mathbf{Z}(0) \in \mathcal{Z}$; $v = 0$; $\{\gamma(v)\}_v \in (0, 1]$; $\gamma_{q^{ul}}, \gamma_{q^{dl}}, \gamma_f, \gamma_{c^{ul}}, \gamma_{c^{dl}} > 0$.

1: If $\mathbf{Z}(v)$ satisfies the termination criterion, stop.

2: Compute $\hat{\mathbf{Z}}(\mathbf{Z}(v))$ from (P.3).

3: Set $\mathbf{Z}(v+1) = \mathbf{Z}(v) + \gamma(v)\left(\hat{\mathbf{Z}}(\mathbf{Z}(v)) - \mathbf{Z}(v)\right)$.

4: $v \leftarrow v + 1$, and return to step 1.

Output: $\mathbf{Z} = \left(\mathbf{Q}^{ul}, \mathbf{Q}^{dl}, \mathbf{f}, \mathbf{c}^{ul}, \mathbf{c}^{dl}\right)$.

local computing at some MUs. Moreover, the backhaul and computing delays may make the latency constraint in problem (P.1) impossible to satisfy, and thus problem (P.1) infeasible. For theses reasons, in this section, we consider user selection with the aim of maximizing the number of MUs that perform offloading while guaranteeing that the selected MUs can satisfy their latency constraints and, at the same time, consume less energy than with local computing. In the rest of this section, the local computation energy model is first elaborated on and then the user scheduling problem is formulated and tackled by integrating SCA with smooth l_p-norm approximation methods.

2.4.1 Local Computation Energy

When the application is executed at the mobile device, the energy consumption $E_{i_n}^M$ is determined by the number of CPU cycles required by the application, V_{i_n}, and by the clock frequency of the device chip, which is denoted here as F_{i_n}. In particular, for CMOS circuits, the energy per operation is proportional to the square of the supply voltage to the chip, and when the supply voltage is low, the clock frequency of the chip is a linear function of the voltage supply [89]. As a result, the mobile energy for

computing can be expressed as

$$E_{i_n}^M = \kappa V_{i_n} F_{i_n}^2, \tag{2.24}$$

where κ is the effective switched capacitance, which depends on the MU processor architecture, and the clock frequency is selected so as to meet the latency constraint, yielding

$$F_{i_n} = \frac{V_{i_n}}{T_{i_n}}. \tag{2.25}$$

By plugging this into (2.24), the total consumption energy for mobile execution is obtained as

$$E_{i_n}^M = \kappa \frac{V_{i_n}^3}{T_{i_n}^2}. \tag{2.26}$$

For each MU i_n, offloading is advantageous when the energy for local mobile computing is higher than the energy required for offloading, i.e.,

$$E_{i_n}^M \geq E_{i_n}^{ul}\left(\mathbf{Q}^{ul}\right), \tag{2.27}$$

where $E_{i_n}^{ul}\left(\mathbf{Q}^{ul}\right)$ is given in (2.6). Note that here for brevity only the uplink energy contribution in (2.14) is considered.

2.4.2 User Scheduling

To proceed, let us introduce the auxiliary slack variables (x_{i_n}, y_{i_n}) for each MU i_n measuring the violation of the latency constraint C.1 in (P.1) and the energy constraint (2.27), respectively. Our system design becomes maximizing the number of MUs that can perform offloading, while satisfying the latency constraints and guaranteeing energy savings with respect to local computation when offloading is performed. This amounts to maximizing the number of MUs i_n with no violation of the mentioned constraints, i.e., with $x_{i_n} = 0$ and $y_{i_n} = 0$. This is done here by

minimizing the ℓ_0-norm

$$\|\mathbf{x}\|_0 + \|\mathbf{y}\|_0 = \sum_{i_n \in \mathcal{X}} \mathbf{I}(x_{i_n} > 0) + \sum_{i_n \in \mathcal{X}} \mathbf{I}(y_{i_n} > 0), \tag{2.28}$$

where $\mathbf{I}$ is an indicator function that returns 1 for $x_{i_n}, y_{i_n} > 0$ and 0 otherwise, also define $\mathbf{x} \triangleq [x_{i_n}]_{i_n \in \mathcal{X}}$ and $\mathbf{y} \triangleq [y_{i_n}]_{i_n \in \mathcal{X}}$. The sum of the l_0-norms of the slack vectors $\mathbf{x}$ and $\mathbf{y}$ in (2.28) counts the number of constraints C.1 and C.2 that are violated by the users. Therefore, minimizing this sum enforces the selection of users that satisfy the largest number of constraints. Accordingly, the problem, defined for generality over any arbitrary subset of users $\mathcal{X} \subseteq \mathcal{I}$, reads

$$\begin{aligned}
\min_{\mathbf{Q}^{ul}, \mathbf{Q}^{dl}, \mathbf{f}, \mathbf{c}^{ul}, \mathbf{c}^{dl}, \mathbf{x}, \mathbf{y}} \quad & \|\mathbf{x}\|_0 + \|\mathbf{y}\|_0 \\
\text{s.t.} \quad \mathbf{C.1} \quad & \tfrac{B^I_{i_n}}{W^{ul} r^{ul}_{i_n}\left(\mathbf{Q}^{ul}\right)} + \tfrac{B^I_{i_n}}{c^{ul}_{i_n} C^{ul}_n} + \tfrac{V_{i_n}}{f_{i_n} F_c} + \tfrac{B^O_{i_n}}{c^{dl}_{i_n} C^{dl}_n} \\
& + \tfrac{B^O_{i_n}}{W^{dl} r^{dl}_{i_n}\left(\mathbf{Q}^{dl}\right)} - T_{i_n} \leq x_{i_n}, \forall i_n \in \mathcal{X}, \\
\mathbf{C.2} \quad & E^{ul}_{i_n}\left(\mathbf{Q}^{ul}\right) - E^M_{i_n} \leq y_{i_n}, \forall i_n \in \mathcal{X}, \\
\mathbf{C.3} \quad & x_{i_n} \geq 0, y_{i_n} \geq 0, \forall i_n \in \mathcal{X}, \\
\mathbf{C.2} - \mathbf{C.4} \quad & \text{of (P.1)}, \forall i_n \in \mathcal{X}.
\end{aligned} \tag{P.4}$$

Define $\mathbf{Z} \triangleq \left(\mathbf{Q}^{ul}, \mathbf{Q}^{dl}, \mathbf{f}, \mathbf{c}^{ul}, \mathbf{c}^{dl}, \mathbf{x}, \mathbf{y}\right)$.

The scheduling algorithm is described in Algorithm 2. The algorithm maximizes the number of scheduled MUs by progressively removing the MUs i_n that have the largest entries in the vector $\mathbf{w} \triangleq \mathbf{x}/\sum_{i_n \in \mathcal{X}} x_{i_n} + \mathbf{y}/\sum_{i_n \in \mathcal{X}} y_{i_n}$, which measures the relative amount, with respect to all users in $\mathcal{X}$, by which a user violates the two constraints. Note that the normalizations by $\sum_{i_n \in \mathcal{X}} x_{i_n}$ and $\sum_{i_n \in \mathcal{X}} y_{i_n}$ ensure that the two constraints are considered on an equal footing. The outlined iterative procedure is repeated until a subset of users $\mathcal{X}^*$ is found for which problem (P.4) returns vector $\mathbf{w}$ that is close to zero, signifying feasibility of offloading under constraints C.1-C.4 of (P.1) as well as (2.27). It is observed that, unlike the admission control scheme in [80, Algorithm 2], the proposed algorithm requires two set of auxiliary variables in

order to account for the constraints C.1 and C.2.

Algorithm 2: User Scheduling

Input: Parameters used by Algorithm 3.

1: Solve problem (P.4) using Algorithm 3 for $\mathcal{X} = \mathcal{I}$ to obtain $\mathbf{w} \triangleq \frac{\mathbf{x}}{\sum_{i_n \in \mathcal{I}} x_{i_n}} + \frac{\mathbf{y}}{\sum_{i_n \in \mathcal{I}} y_{i_n}}$ and sort the MUs in ascending order of the value of $\mathbf{w}$ as $w_{\pi_1} \leq w_{\pi_2} \leq \ldots \leq w_{\pi_{KN_c}}$, where π is a permutation of $\mathcal{I}$.

2: Set: $K_{\text{low}} = 0$, $K_{\text{up}} = KN_c$.

3: **Repeat**

4: Set $s \leftarrow \lfloor \frac{K_{\text{low}} + K_{\text{up}}}{2} \rfloor$.

5: Perform the feasibility test by solving (P.4) using Algorithm 3 for $\mathcal{X} = \mathcal{X}^{[s]} \triangleq \{\pi_1, \ldots, \pi_s\}$: if it is feasible, set $K_{\text{low}} = s$; otherwise, set $K_{\text{up}} = s$.

6: **Until** $K_{\text{up}} - K_{\text{low}} = 1$.

7: Set $s^* = K_{\text{low}}$ and $\mathcal{X}^* = \{\pi_1, \ldots, \pi_{s^*}\}$.

Output: Number of scheduled MUs s^*.

Algorithm 2 returns the solution s^*, from which the set of MUs scheduled for offloading is obtained as $\mathcal{X}^* \triangleq \{\pi_1, \ldots, \pi_{s^*}\}$. In more details, upon obtaining the solution of (P.4), the set of MUs is ordered according to the respective values of the entries of vector $\mathbf{w}$. Then, the subset $\mathcal{X}^*$ of scheduled users is computed by bisection. In particular, bisection searches for the minimum number of users in the interval $[0, KN_c]$, where KN_c is the total number of users, that should be removed, so that the rest of the users can be scheduled for offloading while satisfying the desired constraints. Specifically, as described in Algorithm 2, set $\mathcal{X}^*$ is defined as $\mathcal{X}^* \triangleq \{\pi_1, \ldots, \pi_{s^*}\}$, where the value of $s^* \in [0, KN_c]$ is found by successively searching within the interval $[K_{\text{low}}, K_{\text{up}}]$, which is initialized as $[0, KN_c]$. At each step, first, the search interval is halved using the midpoint s. Then, a feasibility test is performed to check whether the constraints C.1-C.4 of (P.1) can be met if the subset of MUs

$\mathcal{X}^{[s]} \triangleq \{\pi_1, \ldots, \pi_s\}$ is scheduled and the limits $[K_{\text{low}}, K_{\text{up}}]$ is updated accordingly. The feasibility test is carried out by solving an instance of problem (P.4) over the subset of MUs $\mathcal{X}^{[s]}$. The feasibility status is determined by the value of the resulting auxiliary variables, i.e., the problem is considered to be feasible if the slack variables are smaller than a positive value η close to zero.

Let us now discuss how to solve problem (P.4). Problem (P.4) is non-convex due to the non-convexity of the objective and of the constraints C.1 and C.2. Based on the limit $\|\mathbf{x}\|_0 = \lim_{\mathrm{p}\to 0} \|\mathbf{x}\|_{\mathrm{p}}^{\mathrm{p}} = \lim_{\mathrm{p}\to 0} \sum_{i_n \in \mathcal{X}} |x_{i_n}|^{\mathrm{p}}$, the objective function of (P.4) can be approximated by a higher-order norm to make the problem mathematically tractable. In particular, in a manner similar to [80], the following smooth objective is adopted

$$\mathrm{f}(\mathbf{x}, \mathbf{y}) \triangleq \sum_{i_n \in \mathcal{X}} (x_{i_n}^2 + \epsilon^2)^{\mathrm{p}/2} + \sum_{i_n \in \mathcal{X}} (y_{i_n}^2 + \epsilon^2)^{\mathrm{p}/2}, \tag{2.29}$$

where $\epsilon > 0$ is a small fixed regularization parameter. Substituting (2.29) as the objective in (P.4), one can now apply the SCA approach to obtain a local optimal solution of the resulting problem.

To this end, a convex upper bound satisfying conditions A1-A3 described in Section 2.3.1 for the smoothed ℓ_p-norm objective function (2.29) can be obtained from the result in [80, Proposition 1] and is given by

$$\tilde{\mathrm{f}}\left(\mathbf{Z}; \mathbf{Z}\left(v\right)\right) \triangleq \sum_{i_n \in \mathcal{X}} \omega'_{i_n} x_{i_n}^2 + \sum_{i_n \in \mathcal{X}} \omega''_{i_n} y_{i_n}^2 + \bar{\mathrm{f}}\left(\mathbf{Z}; \mathbf{Z}\left(v\right)\right), \tag{2.30}$$

where $\mathbf{Z}\left(v\right) \triangleq \left(\mathbf{Q}^{ul}\left(v\right), \mathbf{Q}^{dl}\left(v\right), \mathbf{f}\left(v\right), \mathbf{c}^{ul}\left(v\right), \mathbf{c}^{dl}\left(v\right), \mathbf{x}\left(v\right), \mathbf{y}\left(v\right)\right)$; $\omega'_{i_n} = \frac{\mathrm{p}}{2}\left[\left(x_{i_n}(v)\right)^2 + \epsilon^2\right]^{\frac{\mathrm{p}}{2}-1}$ and $\omega''_{i_n} = \frac{\mathrm{p}}{2}\left[\left(y_{i_n}(v)\right)^2 + \epsilon^2\right]^{\frac{\mathrm{p}}{2}-1}$; the function $\bar{\mathrm{f}}\left(\mathbf{Z}; \mathbf{Z}\left(v\right)\right) \triangleq \frac{\gamma_{q^{ul}}}{2} \sum_{i_n \in \mathcal{I}} \left\|\mathbf{Q}^{ul} - \mathbf{Q}^{ul}\left(v\right)\right\|^2 + \frac{\gamma_{q^{dl}}}{2} \sum_{i_n \in \mathcal{I}} \left\|\mathbf{Q}^{dl} - \mathbf{Q}^{dl}\left(v\right)\right\|^2 + \frac{\gamma_f}{2}\|\mathbf{f} - \mathbf{f}\left(v\right)\|^2 + \frac{\gamma_{c^{ul}}}{2}\left\|\mathbf{c}^{ul} - \mathbf{c}^{ul}\left(v\right)\right\|^2 + \frac{\gamma_{c^{dl}}}{2}\left\|\mathbf{c}^{dl} - \mathbf{c}^{dl}\left(v\right)\right\|^2 + \frac{\gamma_x}{2}\|\mathbf{x} - \mathbf{x}\left(v\right)\|^2 + \frac{\gamma_y}{2}\|\mathbf{y} - \mathbf{y}\left(v\right)\|^2$ is added to realize the strong convexity of (2.30) with $\gamma_x, \gamma_y > 0$.

The convexification of constraint C.1 is done as in (2.23). Lastly, to obtain an inner convexification for the energy constraint C.2 that satisfies the conditions B1-B6 in Section 2.3.2, one can utilize the concave-convex structure of the rate function $r_{i_n}^{ul}\left(\mathbf{Q}^{ul}\right)$ as in (2.18), to rewrite constraint C.2 as

$$\mathrm{tr}\left(\mathbf{Q}_{i_n}^{ul}\right) - \frac{E_{i_n}^M}{B_{i_n}^I} r_{i_n}^{ul+}\left(\mathbf{Q}^{ul}\right) - \frac{E_{i_n}^M}{B_{i_n}^I} r_{i_n}^{ul-}\left(\mathbf{Q}_{-i_n}^{ul}\right) \leq y_{i_n}, \tag{2.31}$$

where $r_{i_n}^{ul+}\left(\mathbf{Q}^{ul}\right)$ and $r_{i_n}^{ul-}\left(\mathbf{Q}_{-i_n}^{ul}\right)$ are given in (2.18). Using the linearization (2.19), the desired upper bound on C.2 is then obtained as

$$\mathrm{tr}\left(\mathbf{Q}_{i_n}^{ul}\right) - \frac{E_{i_n}^M}{B_{i_n}^I} \tilde{r}_{i_n}^{ul}\left(\mathbf{Q}^{ul}; \mathbf{Q}^{ul}\left(v\right)\right) \leq y_{i_n}. \tag{2.32}$$

Given a feasible point $\mathbf{Z}\left(v\right)$, let us define the following strongly convex problem

$$\begin{aligned}
&\hat{\mathbf{Z}}\left(\mathbf{Z}\left(v\right)\right) \triangleq \underset{\mathbf{Q}^{ul},\mathbf{Q}^{dl},\mathbf{f},\mathbf{c}^{ul},\mathbf{c}^{dl},\mathbf{x},\mathbf{y}}{\mathrm{argmin}} \ \tilde{\mathrm{f}}\left(\mathbf{Z}; \mathbf{Z}\left(v\right)\right) \\
&\qquad\qquad \text{s.t.} \\
&\mathbf{C.1} \quad \tilde{g}_{i_n}\left(\mathbf{Q}^{ul}, \mathbf{Q}^{dl}; \mathbf{Z}\left(v\right)\right) + \frac{B_{i_n}^I}{c_{i_n}^{ul} C_n^{ul}} + \frac{B_{i_n}^O}{c_{i_n}^{dl} C_n^{dl}} \\
&\qquad + \frac{V_{i_n}}{f_{i_n} F_c} - T_{i_n} \leq x_{i_n}, \forall i_n \in \mathcal{X}, \\
&\mathbf{C.2} \quad \mathrm{tr}\left(\mathbf{Q}_{i_n}^{ul}\right) - \frac{E_{i_n}^M}{B_{i_n}^I} \tilde{r}_{i_n}^{ul}\left(\mathbf{Q}^{ul}; \mathbf{Q}^{ul}\left(v\right)\right) \leq y_{i_n}, \forall i_n \in \mathcal{X}, \\
&\mathbf{C.3} \quad x_{i_n} \geq 0, y_{i_n} \geq 0, \forall i_n \in \mathcal{X}, \\
&\mathbf{C.2} - \mathbf{C.4} \quad \text{of (P.1)}, \forall i_n \in \mathcal{X},
\end{aligned} \tag{P.5}$$

where $\hat{\mathbf{Z}}\left(\mathbf{Z}\right) \triangleq \left(\hat{\mathbf{Q}}^{ul}, \hat{\mathbf{Q}}^{dl}, \hat{\mathbf{f}}, \hat{\mathbf{c}}^{ul}, \hat{\mathbf{c}}^{dl}, \hat{\mathbf{x}}, \hat{\mathbf{y}}\right)$ denote the unique solution of (P.5). The SCA scheme for solving (P.5) is described in Algorithm 3. As a technical note, we observe that here, since the approximant (2.30) of the objective function (2.29) is an upper bound on (2.29), convergence of Algorithm 3 is guaranteed also by setting $\gamma(v) = 1$ [78, Section III-A].

Algorithm 3: SCA Solution for (P.5)

Input: Parameters from Table 2.1; $\mathrm{p} = 0.5$; $v = 0$; $\mathbf{Z}(0) \in \mathcal{Z}$; $\{\gamma(v)\}_v \in (0, 1]$; $\gamma_{q^{ul}}, \gamma_{q^{dl}}, \gamma_f, \gamma_{c^{ul}}, \gamma_{c^{dl}}, \gamma_x, \gamma_y, \epsilon > 0$; ${\omega'}_{i_n}(0) = {\omega''}_{i_n}(0) = 1$.

1: If $\left|\tilde{\mathrm{f}}(\mathbf{Z}; \mathbf{Z}(v+1)) - \tilde{\mathrm{f}}(\mathbf{Z}; \mathbf{Z}(v))\right| \leq \delta$, stop.

2: Compute $\hat{\mathbf{Z}}(\mathbf{Z}(v))$ from (P.5).

3: Set $\mathbf{Z}(v+1) = \mathbf{Z}(v) + \gamma(v)\left(\hat{\mathbf{Z}}(\mathbf{Z}(v)) - \mathbf{Z}(v)\right)$.

4: Update

$$
\begin{aligned}
{\omega'}_{i_n}(v+1) &= \frac{\mathrm{p}}{2}\left[(x_{i_n}(v+1))^2 + \epsilon^2\right]^{\frac{\mathrm{p}}{2}-1}, \\
{\omega''}_{i_n}(v+1) &= \frac{\mathrm{p}}{2}\left[(y_{i_n}(v+1))^2 + \epsilon^2\right]^{\frac{\mathrm{p}}{2}-1}.
\end{aligned}
$$

5: $v \leftarrow v + 1$, and return to step 1.

Output: $\left(\mathbf{Q}^{ul}, \mathbf{Q}^{dl}, \mathbf{f}, \mathbf{c}^{ul}, \mathbf{c}^{dl}, \mathbf{x}, \mathbf{y}\right)$.

2.5 Hybrid Edge and Cloud Computing

In the previous sections, the considered scenario assumes that MUs can offload applications to a cloud server. In this section, besides the cloud server, the analysis is extended to a more general set-up in which the ceNBs are directly connected to local computing servers, also known as cloudlets [7], which may run some of the MUs' applications. Specifically, each ceNB can either execute the computation task on the behalf of the MU or offload it to the cloud. Let F_n^{ceNB} be the computation capability in CPU cycles per second of ceNB n, and let $f_{i_n}^{\mathrm{ceNB}} \geq 0$ be the fraction of the ceNB's computing power assigned to user i_n, so that $\sum_i f_{i_n}^{\mathrm{ceNB}]} \leq 1$. If implemented at the ceNB, the execution time of the task of MU i_n is then given as

$$
\Delta_{i_n}^{\mathrm{exe|ceNB}} = \frac{V_{i_n}}{f_{i_n}^{\mathrm{ceNB}} F_n^{\mathrm{ceNB}}}. \tag{2.33}
$$

In the same way, if the cloud processes the task of user i_n, the execution time $\Delta_{i_n}^{\mathrm{exe|cloud}}$ is given by the right-hand side of (2.12). The overall latency Δ_{i_n} experienced

by each MU i_n can be expressed as

$$\Delta_{i_n} = \Delta_{i_n}^{ul} + (1 - u_{i_n})\,\Delta_{i_n}^{\text{exe|ceNB}} + u_{i_n}\Delta_{i_n}^{\text{exe|cloud}} + u_{i_n}\Delta_{i_n}^{bh} + \Delta_{i_n}^{dl}, \tag{2.34}$$

where $\Delta_{i_n}^{ul}, \Delta_{i_n}^{bh}$ and $\Delta_{i_n}^{dl}$ have the same definition as in (2.5), (2.13) and (2.10), respectively; u_{i_n} is a binary variable that indicates whether if the task of MU i_n is processed on the ceNB ($u_{i_n} = 0$) or on the cloud ($u_{i_n} = 1$).

To proceed, one can relax the binary variable u_{i_n} to be defined in the interval $[0, 1]$. This relaxation not only provides a lower bound on the minimum energy expenditure that can be obtained with a hard choice between cloudlet and cloud offloading, perhaps more importantly, it also captures a system in which the input data to the application of the MU i_n can be split into two parts, of sizes $1 - u_{i_n}$ and u_{i_n}, that can be processed separately at the ceNB and cloud, respectively. In the following, we will adopt this latter justification of the model.

As in Section 2.2.2, the problem targets the minimization of the total energy consumed by the MUs to execute their tasks remotely under latency and power constraints. The problem is given by

$$\begin{aligned}
\min_{\mathbf{Q}^{ul},\mathbf{Q}^{dl},\mathbf{u},\mathbf{f}^{\text{ceNB}},\mathbf{f},\mathbf{c}^{ul},\mathbf{c}^{dl}} \quad & E^{ul}\left(\mathbf{Q}^{ul}\right) = \sum_{i_n\in\mathcal{I}} E_{i_n}^{ul}\left(\mathbf{Q}_{i_n}^{ul}, \mathbf{Q}_{-i_n}^{ul}\right) \\
& = \sum_{i_n\in\mathcal{I}} B_{i_n}^{I} \frac{\text{tr}\left(\mathbf{Q}_{i_n}^{ul}\right)}{r_{i_n}^{ul}\left(\mathbf{Q}^{ul}\right)} \\
\text{s.t.} \quad & \\
\mathbf{C.1} \quad & \frac{B_{i_n}^{I}}{W^{ul} r_{i_n}^{ul}\left(\mathbf{Q}^{ul}\right)} + \frac{(1-u_{i_n})V_{i_n}}{f_{i_n}^{\text{ceNB}} F_n^{\text{ceNB}}} + \frac{u_{i_n} B_{i_n}^{I}}{c_{i_n}^{ul} C_n^{ul}} + \frac{u_{i_n} V_{i_n}}{f_{i_n} F_c} + \frac{u_{i_n} B_{i_n}^{O}}{c_{i_n}^{dl} C_n^{dl}} + \frac{B_{i_n}^{O}}{W^{dl} r_{i_n}^{dl}\left(\mathbf{Q}^{dl}\right)} \leq T_{i_n}, \forall i_n \in \mathcal{I}, \\
\mathbf{C.2} \quad & 0 \leq u_{i_n} \leq 1, \forall i_n \in \mathcal{I}, \\
\mathbf{C.3} \quad & f_{i_n}^{\text{ceNB}}, f_{i_n} \geq 0, \forall i_n \in \mathcal{I}, \sum_{i_n\in\mathcal{I}} f_{i_n} \leq 1, \sum_{i=1}^{K} f_{i_n}^{\text{ceNB}} \leq 1, \forall n = 1, \ldots, N_c, \\
\mathbf{C.4} \quad & c_{i_n}^{ul}, c_{i_n}^{dl} \geq 0, \forall i_n \in \mathcal{I}, \sum_{i=1}^{K} c_{i_n}^{ul} \leq 1, \sum_{i=1}^{K} c_{i_n}^{dl} \leq 1, \forall n = 1, \ldots, N_c, \\
\mathbf{C.5} \quad & \mathbf{Q}_{i_n}^{ul} \in \mathcal{Q}_{i_n}^{ul}, \forall i_n \in \mathcal{I}, \left(\mathbf{Q}_{i_n}^{dl}\right)_{i=1}^{K} \in \mathcal{Q}_n^{dl}, \forall n = 1, \ldots, N_c.
\end{aligned} \tag{P.6}$$

Note that, unlike (P.1), here we have the additional optimization variables $\mathbf{u} \triangleq (u_{i_n})_{i_n \in \mathcal{I}}$ and $\mathbf{f}^{\text{ceNB}} \triangleq \left(f_{i_n}^{\text{ceNB}}\right)_{i_n \in \mathcal{I}}$.

It can be observed that (P.6) is non-convex due to the non-convexity of the objective function and constraint C.1. The problem is tackled by means of the SCA method convexifying the objective function as done in Section 2.3.1. Furthermore, one needs to calculate a convex upper bound for the C.1 constraint in (P.6) that satisfies conditions B1-B6 in Section 2.3.2. Let us write the left-hand side of C.1 by

$$\begin{aligned} g_{i_n}\left(\mathbf{Q}^{ul}, \mathbf{Q}^{dl}, \mathbf{u}, \mathbf{f}^{\text{ceNB}}, \mathbf{f}, \mathbf{c}^{ul}, \mathbf{c}^{dl}\right) \triangleq & \frac{B_{i_n}^I}{W^{ul} r_{i_n}^{ul}\left(\mathbf{Q}^{ul}\right)} + \frac{B_{i_n}^O}{W^{dl} r_{i_n}^{dl}\left(\mathbf{Q}^{dl}\right)} \\ & + \frac{\left(1-u_{i_n}\right) V_{i_n}}{f_{i_n}^{\text{ceNB}} F_n^{\text{ceNB}}} + \frac{u_{i_n} V_{i_n}}{f_{i_n} F_c} + \frac{u_{i_n} B_{i_n}^I}{c_{i_n}^{ul} C_n^{ul}} + \frac{u_{i_n} B_{i_n}^O}{c_{i_n}^{dl} C_n^{dl}}. \end{aligned} \tag{2.35}$$

To build the desired bound on g_{i_n}, it is observed that the first two terms can be handled as in Section 2.3.2, while for the last four terms, they are all seen to be ratios of linear functions, it is easy to show that the relationship

$$\frac{x}{y} = \frac{1}{2}\left(x + \frac{1}{y}\right)^2 - \frac{1}{2}\left(x^2 + \frac{1}{y^2}\right), \tag{2.36}$$

holds, where the right-hand side is the difference of two convex functions if $x \geq 0$ and $y > 0$. Therefore, a locally tight convex upper bound on the left-hand side of (2.36) can be obtained by linearizing the concave part as

$$\frac{x}{y} \leq \frac{1}{2}\left(x + \frac{1}{y}\right)^2 - \frac{1}{2}\left(\left(x^v\right)^2 + \frac{1}{\left(y^v\right)^2}\right) - x^v\left(x - x^v\right) + \frac{1}{\left(y^v\right)^3}\left(y - y^v\right), \tag{2.37}$$

where superscript v identifies the point (x^v, y^v) at which the upper bound is tight. Using (2.37) to the last four terms in (2.35), the desired approximants is then defined by substituting for x and y in (2.37) the numerator and denominator, respectively, of each of the last four terms in (2.35). This yields the SCA procedure in Algorithm 1 with the difference that we defined $\mathbf{Z} \triangleq \left(\mathbf{Q}^{ul}, \mathbf{Q}^{dl}, \mathbf{u}, \mathbf{f}^{\text{ceNB}}, \mathbf{f}, \mathbf{c}^{ul}, \mathbf{c}^{dl}\right)$ and $\mathbf{Z}(v) \triangleq \left(\mathbf{Q}^{ul}(v), \mathbf{Q}^{dl}(v), \mathbf{u}(v), \mathbf{f}^{\text{ceNB}}(v), \mathbf{f}(v), \mathbf{c}^{ul}(v), \mathbf{c}^{dl}(v)\right)$.

$$\hat{\mathbf{Z}}\left(\mathbf{Z}\left(v\right)\right) \triangleq \underset{\mathbf{Q}^{ul},\mathbf{Q}^{dl},\mathbf{u},\mathbf{f}^{\text{ceNB}},\mathbf{f},\mathbf{c}^{ul},\mathbf{c}^{dl}}{\operatorname{argmin}} \tilde{E}^{ul}\left(\mathbf{Z};\mathbf{Z}\left(v\right)\right)$$

s.t.

$$\mathbf{C.1} \quad \tilde{g}_{i_n}\left(\mathbf{Z};\mathbf{Z}\left(v\right)\right) \leq T_{i_n}, \forall i_n \in \mathcal{I},$$

$$\mathbf{C.2} - \mathbf{C.5} \quad \text{of (P.6)}, \tag{P.7}$$

where $\hat{\mathbf{Z}}\left(\mathbf{Z}\left(v\right)\right) \triangleq (\hat{\mathbf{Q}}^{ul}, \hat{\mathbf{Q}}^{dl}, \hat{\mathbf{u}}, \hat{\mathbf{f}}^{\text{ceNB}}, \hat{\mathbf{f}}, \hat{\mathbf{c}}^{ul}, \hat{\mathbf{c}}^{dl})$ denotes the unique solution of the strongly convex optimization problem (P.7); the objective function $\tilde{E}^{ul}\left(\mathbf{Z};\mathbf{Z}\left(v\right)\right) \triangleq \sum_{i_n\in\mathcal{I}} \left(\tilde{E}^{ul}_{i_n}\left(\mathbf{Q}^{ul};\mathbf{Q}^{ul}(v)\right) + \bar{E}_{i_n}\left(\mathbf{Z}_{i_n};\mathbf{Z}_{i_n}\left(v\right)\right)\right)$ where $\tilde{E}^{ul}_{i_n}\left(\mathbf{Q}^{ul};\mathbf{Q}^{ul}(v)\right)$ is given as in (2.16) and we define $\bar{E}\left(\mathbf{Z}_{i_n};\mathbf{Z}_{i_n}\left(v\right)\right) \triangleq \frac{\gamma_{q^{ul}}}{2}\left\|\mathbf{Q}^{ul}_{i_n} - \mathbf{Q}^{ul}_{i_n}\left(v\right)\right\|^2 + \frac{\gamma_{q^{dl}}}{2}\left\|\mathbf{Q}^{dl}_{i_n} - \mathbf{Q}^{dl}_{i_n}\left(v\right)\right\|^2 + \frac{\gamma_u}{2}\left\|u_{i_n} - u_{i_n}\left(v\right)\right\|^2 + \frac{\gamma_{f^{\text{ceNB}}}}{2}\| f^{\text{ceNB}}_{i_n} - f^{\text{ceNB}}_{i_n}\left(v\right)\|^2 + \frac{\gamma_f}{2}\left\|f_{i_n} - f_{i_n}\left(v\right)\right\|^2 + \frac{\gamma_{c^{ul}}}{2}\left\|c^{ul}_{i_n} - c^{ul}_{i_n}\left(v\right)\right\|^2 + \frac{\gamma_{c^{dl}}}{2}\left\|c^{dl}_{i_n} - c^{dl}_{i_n}\left(v\right)\right\|^2$ with $\gamma_u, \gamma_{f^{\text{ceNB}}} > 0$; and $\tilde{g}_{i_n}\left(\mathbf{Z};\mathbf{Z}\left(v\right)\right)$ is the locally convex upper bound defined above.

2.6 Enhanced Downlink via Network MIMO

In the considered system model so far, the assumption was that each ceNB n serves the users $\{i_n : i = 1, \ldots, K\}$ in its cell, hence interfering with other ceNBs. In this section, instead, a more enhanced downlink transmission based on network MIMO is considered. Specifically, assume that each MU can receive the result of the cloud execution in the downlink not only from ceNB in the same cell, but also from other ceNBs that cooperatively transmit to the MU. Cooperation among ceNBs is enabled by the transmission on the backhaul links of the outcome of the cloud execution for a given MU to multiple ceNBs. The extreme case of full ceNB cooperation is considered in this section, so that all the ceNBs transmit cooperatively to each MU. The extension to a more general model with clustered cooperation is straightforward and will not be pursued further.

To express the achievable downlink rate with network MIMO, it is convenient to define user-centric transmit covariance matrices $\mathbf{Q}_j^{dl} \in \mathbb{C}^{N_T \times N_T}$ for every MU $j \in \mathcal{I}$, where we have defined $N_T \triangleq \sum_{n=1}^{N_c} N_{T_n}$, such that the n-th $N_{T_n} \times N_{T_n}$ block on the main diagonal, denoted by $[\mathbf{Q}_j^{dl}]_n$, represents the contribution of ceNB n to the transmission to MU j. Note that the out-of-diagonal blocks in $\mathbf{Q}_j^{dl}$ describe the correlation among signals sent by different ceNBs, which designates cooperative transmission at the ceNBs. The set of downlink covariance matrices, $\mathbf{Q}^{dl} = \left\{\mathbf{Q}_j^{dl}\right\}_{j\in\mathcal{I}}$ is

$$\mathcal{Q}^{dl} \triangleq \left\{\mathbf{Q}_j^{dl} \in \mathbb{C}^{N_T \times N_T},\ j \in \mathcal{I} : \sum_{j\in\mathcal{I}} \mathrm{tr}\big([\mathbf{Q}_j^{dl}]_n\big) \leq P_n^{dl}, \forall n\right\}, \tag{2.38}$$

so that the per-ceNB power constraints are satisfied. Also, it is convenient to define the channel matrix from all the ceNBs to MU j as $\tilde{\mathbf{G}}_j \in \mathbb{C}^{N_{R_j} \times N_T}$, where $\tilde{\mathbf{G}}_j \triangleq [\mathbf{G}_{j1}, \mathbf{G}_{j2}, \ldots, \mathbf{G}_{jN_c}]$ and we have set $\mathbf{G}_{j,n} = \mathbf{G}_j$. With these definitions, the achievable downlink rate is given by

$$r_{i_n}^{dl}(\mathbf{Q}^{dl}) = \log_2 \det\Big(\mathbf{I} + \tilde{\mathbf{G}}_{i_n}^H \tilde{\mathbf{R}}_{i_n}^{dl}(\mathbf{Q}_{-i_n}^{dl})^{-1} \tilde{\mathbf{G}}_{i_n} \mathbf{Q}_{i_n}^{dl}\Big), \tag{2.39}$$

with

$$\tilde{\mathbf{R}}_{i_n}^{dl}(\mathbf{Q}_{-i_n}^{dl}) \triangleq N_0\mathbf{I} + \sum_{i_m \in \mathcal{I}\setminus\{i_n\}} \tilde{\mathbf{G}}_{i_n} \mathbf{Q}_{i_m}^{dl} \tilde{\mathbf{G}}_{i_n}^H, \tag{2.40}$$

and $\mathbf{Q}_{-i_n}^{dl} \triangleq (\mathbf{Q}_{j_m}^{dl})_{j_m \neq i_n}$.

In order to enable network MIMO, one needs to ensure that downlink transmissions from all ceNBs take place at the same time. To this end, there is a necessity to impose that uplink transmission, computing and backhaul transmissions for all MUs are constrained to be completed by a given time T_1. At time T_1, then, the downlink transmission is initiated and takes a given time T_2. Accordingly, following the weighted sum approach discussed in Remark 2, the optimization problem is

formulated as

$$
\begin{aligned}
\min_{\mathbf{Q}^{ul},\mathbf{Q}^{dl},\mathbf{f},\mathbf{c}^{ul},\mathbf{c}^{dl},T_1,T_2} \quad & \sum_{i_n\in\mathcal{I}} B^I_{i_n}\frac{\mathrm{tr}\left(\mathbf{Q}^{ul}_{i_n}\right)}{r^{ul}_{i_n}\left(\mathbf{Q}^{ul}\right)} + \lambda(T_1+T_2) \\
\text{s.t.} \quad \mathbf{C.1} \quad & \frac{B^I_{i_n}}{W^{ul}r^{ul}_{i_n}\left(\mathbf{Q}^{ul}\right)} + \frac{B^I_{i_n}}{c^{ul}_{i_n}C^{ul}_{n}} + \frac{V_{i_n}}{f_{i_n}F_c} + \frac{B^O_{i_n}}{C^{dl}_{m}c^{dl}_{i_n,m}} \leq T_1, \forall i_n \in \mathcal{I}, \\
\mathbf{C.2} \quad & \frac{B^O_{i_n}}{W^{dl}r^{dl}_{i_n}\left(\mathbf{Q}^{dl}\right)} \leq T_2, \forall i_n \in \mathcal{I}, \\
\mathbf{C.3} \quad & f_{i_n} \geq 0, \forall i_n \in \mathcal{I}, \sum_{i_n\in\mathcal{I}} f_{i_n} \leq 1 \\
\mathbf{C.4} \quad & c^{ul}_{i_n}, c^{dl}_{i_n,m} \geq 0, \forall i_n \in \mathcal{I}, \sum_{i=1}^{K} c^{ul}_{i_n} \leq 1, \forall n = 1,\ldots,N_c, \sum_{j\in\mathcal{I}} c^{dl}_{j,m} \leq 1, \\
\mathbf{C.5} \quad & \mathbf{Q}^{ul}_{i_n} \in \mathcal{Q}^{ul}_{i_n}, \forall i_n \in \mathcal{I}, \mathbf{Q}^{dl} \in \mathcal{Q}^{dl},
\end{aligned}
\tag{P.8}
$$

where $\mathbf{c}^{dl} \triangleq \left(c^{dl}_{i_n,m}\right)_{i_n\in\mathcal{I},m}$, with $c^{dl}_{i_n,m}$ being the fraction of the backhaul capacity to ceNB m allocated to transmit the output bits intended for MU i_n; and $\lambda > 0$ is a parameter defining the relative weight of energy and latency.

Problem (P.8) is non-convex due to the non-convexity of the objective function and constraints C.1 and C.2. One can tackle this problem using SCA in a way similar to that used for (P.1), i.e., to obtain a convex approximants for the objective functions using (2.16) and for the latency constraints using (2.23). The problem is then solved using the SCA procedure described in Algorithm 1.

2.7 Numerical Results

In this section, numerical results are presented to validate the model and algorithms discussed in the previous sections. Throughout, we consider a network composed of three cells with five users in each cell, i.e., $N_c = 3$ and $K = 5$. All transceivers are equipped with $N_{T_{i_n}} = N_{R_n} = 2$ antennas. The channel matrices are generated with independent and identically distributed complex Gaussian entries having zero mean and variance equal to the path loss, which is assumed to be identical for uplink and downlink. The path loss is given by 170 dBm between an MU and the ceNB in the same small cell and 180 dBm between an MU and the ceNB in the other small cell. The values 170 dBm and 180 dBm can be justified by considering the

Walfish-Ikegami model [1] with distances of 500 meters between MU and ceNB in the same cell and 700 meters for MU and ceNB in different cells. The other parameters of the Walfish-Ikegami model are selected so as to simulate a small-cell environment in a typical urban setting as listed in Table 2.2. Targeting a small-cell scenarios, we will explore values of the backhaul capacities as low as few Mbits/s [8, 26]. Finally, note that setting $N_0 = -170$ dBm/Hz [1], yields an average signal-to-noise ratio (SNR) on the direct link of 40 dBm per receive antenna and 30 dBm on the interference link for a signal transmitted at 0.01 Joule per symbol from a single antenna in the uplink and downlink. Furthermore, unless stated otherwise, each number of input bits $B^I_{i_n}$ and output bits $B^O_{i_n}$ is selected uniformly at random in the interval $[0.1 - 1]$ Mbits and we set the number of CPU cycles as $V_{i_n} = 2640 \times B^I_{i_n}$ CPU cycles. These choices reflect computational-intensive applications, as demonstrated by the measurements in [35, 90]. The cloud capacity is $F_c = 10^{11}$ CPU cycles/s, which is, e.g., obtained by a four-core server with Intel Xeon processor with 3.3 GHz that is commercially used by Amazon elastic compute cloud (EC2) [36, 91]. Other system parameters are set to $W^{ul} = W^{dl} = 10$ MHz, $C^{ul}_n = C^{dl}_n = 100$ Mbits/s, $d_{i_n} = 10^{-5}$ J/symbol [92], and $T_{i_n} = 0.1$ seconds. Throughout, averages are intended with respect to the channel realizations.

Let us start by illustrating the typical convergence properties of the SCA algorithm by plotting in Figure 2.3 the average minimal mobile energy consumption $E\left(\mathbf{Q}^{ul}, \mathbf{Q}^{dl}\right)$ versus the iteration index v. Besides the joint optimization across uplink, downlink, backhaul and computing resources, studied in Section 2.2.2, let us consider the following more conventional solutions: *(i) Equal backhaul and cloud allocation:* the computing and backhaul resources are equally allocated to all MUs, that is, $f_{i_n} = 1/(N_c K)$ and $c^{ul}_{i_n} = c^{dl}_{i_n} = 1/K$ for all $i_n \in \mathcal{I}$, while the covariance transmit and receive matrices at the physical layer are optimized using SCA [48]; *(ii) Equal cloud allocation:* computing resources at the cloud are equally allocated

Table 2.2 Parameters in the Walfish-Ikegami Path Loss Model [1].

parameter	value	description
f	1800	frequency (MHz)
θ	45°	road orientation angle (degree)
h_{Tx}	3	height of transmitter (meters)
h_{Rx}	1	height of receiver (meters)
h_{Roof}	5.5	mean value of buildings height (meters)
s_{Roof}	8	mean value of buildings separation (meters)
str_wid	5	mean value of street width (meters)
k_a	54	path loss penalty when ceNB below rooftop
k_d	5	correlation adjustment constant

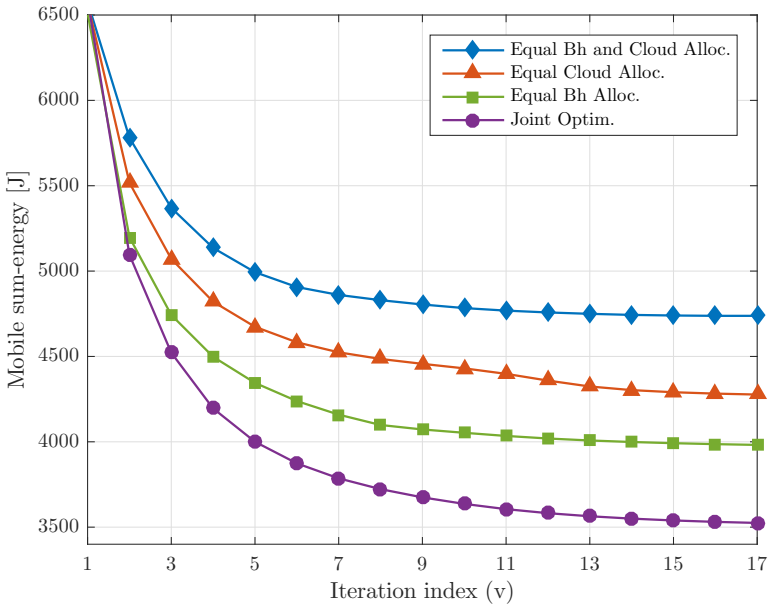


Figure 2.3 Minimum average mobile sum-energy consumption versus iteration index ($T_{i_n} = 0.12$ seconds, $N_c = 3, K = 5$, $W^{ul} = W^{dl} = 100$ MHz, $B^I_{i_n}$ and $B^O_{i_n} \sim \mathcal{U}\{0.1, 1\}$ Mbits, $V_{i_n} = 2640 \times B^I_{i_n}$ CPU cycles, $C^{ul}_n = C^{dl}_n = 100$ Mbits/s, and $F_c = 10^{11}$ CPU cycles/s).

among the MUs, while the rest of the parameters are jointly optimized using SCA; *(iii) Equal backhaul allocation:* the backhaul resources are equally allocated among the MUs, while the rest of the parameters are jointly optimized using SCA. It is observed that SCA has fast convergence, which requires around 17 iterations to obtain results close to convergence (at iteration 17 the termination criterion $\left|E\left(\mathbf{Q}^{ul}(v+1), \mathbf{Q}^{dl}(v+1)\right) - E\left(\mathbf{Q}^{ul}(v), \mathbf{Q}^{dl}(v)\right)\right| \leq \delta$ is satisfied with $\delta = 10^{-3}$ for $\alpha = 10^{-5}$). Furthermore, it can be seen that the proposed joint optimization method shows a considerable gain compared to the equal allocation of computational and backhaul resources.

The gains that can be accrued by means of joint optimization are further investigated in Figure 2.4 (obtained in the same setting of Figure 2.3), which depicts the minimum average mobile energy as a function of the latency constraints T_{i_n}, which are assumed to be the same for all MUs. The energy saving due to joint optimization can be seen to be particularly pronounced in the regime in which the latency constraint is more stringent. For instance, at $T_{i_n} = 0.12$ seconds, which is the smallest latency for which all schemes are feasible, joint optimization saves 66% in terms of sum-energy as compared to equal allocation of backhaul and cloud resources, 48% as compared to equal cloud allocation, and 32% as compared to equal backhaul allocation.

To account for the case in which the network may operate under asymmetrical bandwidth allocation for uplink and downlink, it is helpful to compare the mobile sum-energy consumption of the schemes considered above in the same set-up of Figures 2.3 and 2.4, with the caveat that to set $W^{dl} = 10$ MHz and to vary the uplink/downlink bandwidth ratio W^{ul}/W^{dl}, in Figure 2.5. It is observed that joint optimization is especially advantageous in term of mobile energy consumption when the uplink bandwidth is more constrained than the downlink bandwidth. This is because, in this regime, it is particularly useful to allocate more computing and

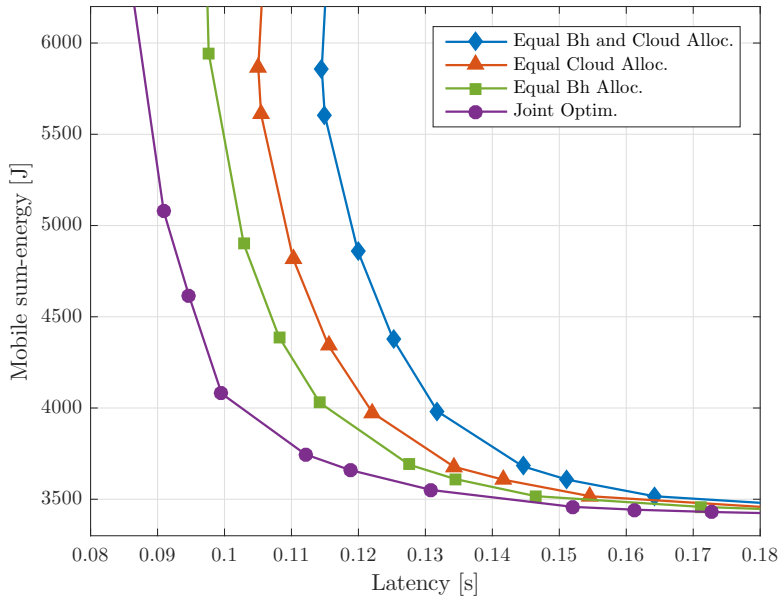


Figure 2.4 Minimum average mobile sum-energy consumption versus the latency constraint T_{i_n}, assumed to be the same for all MUs ($N_c = 3, K = 5$, $W^{ul} = W^{dl} = 100$ MHz, $B^I_{i_n}$ and $B^O_{i_n} \sim \mathcal{U}\{0.1, 1\}$ Mbits, $V_{i_n} = 2640 \times B^I_{i_n}$ CPU cycles, $C^{ul}_n = C^{dl}_n = 100$ Mbits/s, and $F_c = 10^{11}$ CPU cycles/s).

backhaul resources to users with worse channel condition in order to meet the latency requirements with minimal energy expenditure. For example, when W^{ul} is five times smaller than the downlink bandwidth, the joint optimization scheme is 50% more energy efficient than the fixed allocation of backhaul and cloud resources.

While the previous results assume that all MUs perform computation offloading, we next turn to the study of user selection that is presented in Section 2.4. To account for mobile energy consumption, the effective switched capacitance of the mobile is set to $\kappa = 10^{-26}$ so that the mobile energy consumption is consistent with the measurements made in [35] for a Nokia N900 mobile device operating at frequency 500 MHz. To this end, it is plotted in Figure 2.6, the average number of selected MUs as a function of the latency constraint T_{i_n}, assumed to be the same for all MUs, for different values of backhaul. The figure shows the results obtained by means of the efficient algorithm proposed in Section 2.4 with $\eta = 10^{-5}$, as well as by an *exhaustive*

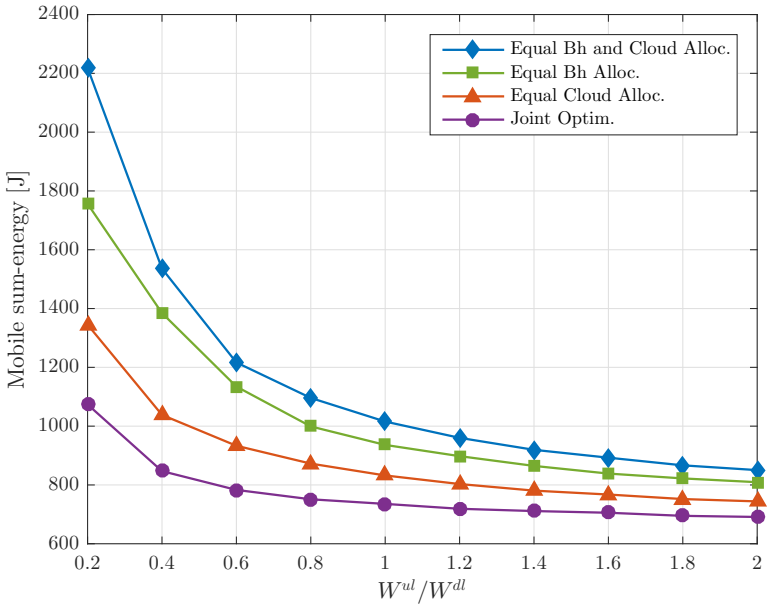


Figure 2.5 Minimum average mobile sum-energy consumption versus bandwidth allocation ratio W^{ul}/W^{dl} ($W^{dl} = 10$ MHz, $N_c = 2, K = 2, T_{i_n} = 0.09$ s, $B^I_{i_n}$ and $B^O_{i_n} \sim \mathcal{U}\{0.1, 1\}$ Mbits, $V_{i_n} = 2640 \times B^I_{i_n}$ CPU cycles, $C^{ul}_n = C^{dl}_n = 100$ Mbits/s, and $F_c = 10^{11}$ CPU cycles/s).

search strategy in which, for every subset of users, a joint optimization problem is solved as per Section 2.2.2 using SCA (Algorithm 1) and then the subset of users which yields the minimal energy consumption is selected. The figure demonstrates how a less restrictive latency constraint and/or a larger backhaul increases the number of users that should be allowed to offload. For instance, for $T_{i_n} \leq 0.07$ seconds, there is no MU on average offloads its applications since offloading is more energy demanding. Instead, for $T_{i_n} > 0.2$ seconds, as long as the backhaul capacity is larger than 500 Mbits/s, all MUs tend to offload. It is also observed that the proposed scheme selects a number of users close to that chosen by exhaustive search.

The corresponding overall energy consumption for the example of Figure 2.6 with backhaul capacity $C^{ul}_n = C^{dl}_n = 100$ Mbits/s is shown in Figure 2.7. Note that the overall energy consumption is the sum of the mobile computing energy for the MUs that perform local computing, namely (2.26), and of the energy used for

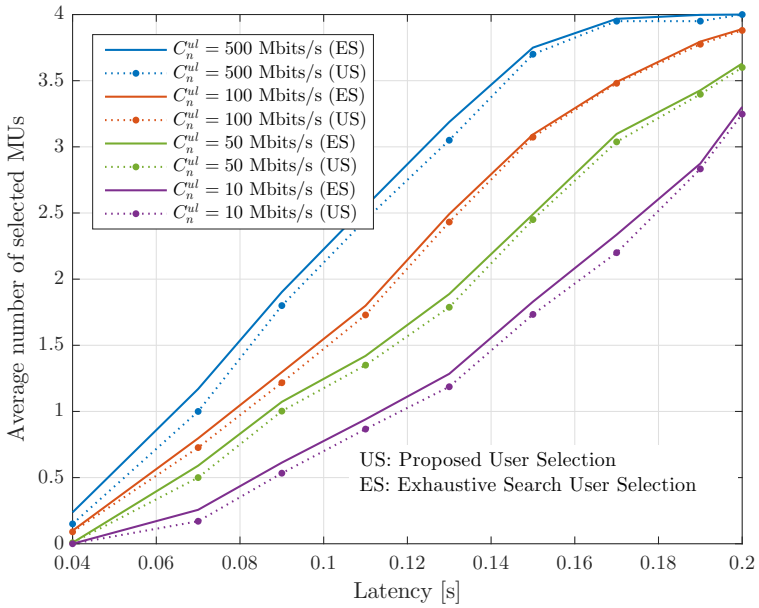


Figure 2.6 Average number of selected MUs versus latency constraint T_{i_n} for both the proposed efficient scheme in Section 2.4 and exhaustive search ($N_c = 2, K = 2$, $B_{i_n}^I = B_{i_n}^O = 1$ Mbits, $V_{i_n} = 10^9$ CPU cycles, $F_c = 10^{11}$ CPU cycles/s, and $C_n^{dl} = C_n^{ul}$).

transmission, namely (2.6), for the MUs that offload. For reference, it is also shown the energy required when all MUs perform their tasks locally. The proposed user selection scheme is seen to achieve a near-optimal energy performance compared to the exhaustive search method. As an example, when the latency constraint $T_{i_n} = 0.17$ seconds, around 83% energy saving can be obtained with the proposed user offloading selection scheme as compared to *local computing*.

In the previous results, central cloud processing was assumed. It is time now to investigate the optimal allocation of computing tasks between the edge and the cloud in the set-up studied in Section 2.5. To this end, in Figure 2.8, we plotted the fractions $(\hat{u}_{i_n})_{i_n \in \mathcal{I}}$ of the application of each MU i_n to be performed at the cloud as a function of the backhaul capacity when the computing capability of the cloudlet is given by $F_n^{\text{ceNB}} = 0.1F_c$. To highlight the impact of asymmetries in the offloading requirements, It is assumed here that in the first cell the two users have $B_{1_1}^I = B_{1_1}^O = 1$ Mbits, $V_{1_1} = 10^9$ CPU cycles, and $B_{2_1}^I = B_{2_1}^O = 0.7$ Mbits, $V_{2_1} = 0.7V_{1_1}$ CPU

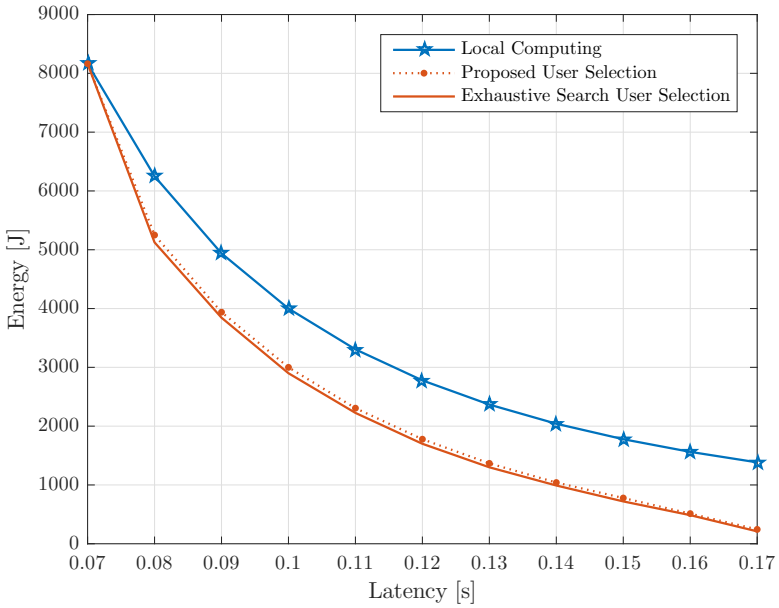


Figure 2.7 Minimum average mobile sum-energy consumption versus the latency constraint T_{i_n} ($N_c = 2, K = 2$, $B^I_{i_n} = B^O_{i_n} = 1$ Mbits, $V_{i_n} = 10^9$ CPU cycles, $C^{ul}_n = C^{dl}_n = 100$ Mbits/s, and $F_c = 10^{11}$ CPU cycles/s).

cycles, while the users in the second cell have smaller offloading requirements for data transfer and computing, normally $B^I_{1_2} = B^O_{1_2} = 0.5$ Mbits, $V_{1_2} = 0.5V_{1_1}$ CPU cycles, and $B^I_{2_2} = B^O_{2_2} = 0.1$ Mbits, $V_{2_2} = 0.1V_{1_1}$ CPU cycles. It is observed that, when the backhaul capacity is limited, tasks tends to be performed at the edge, especially for users with more stringent data transfer and computing requirements, for example, at $C^{ul}_n = C^{dl}_n = 3$ Mbits/s, the two users in the first cell execute 50% and 80% percent of their offloaded tasks at the cloud, while the users in the second cell have a complete tasks execution at the cloud due to the moderate number of input/output bits. As the backhaul capacity is increased, more tasks are executed at the cloud, benefiting from the improved connection between ceNBs and the cloud.

To obtain further insights, Figure 2.9. shows the mobile sum-energy consumption versus the backhaul capacity for cloud mobile computing, whereby all users offload to the cloud, and edge mobile computing, whereby all users offload to the local cloudlet, for $N_c = 2, K = 2$, and $F^{\text{ceNB}}_n = 10^{10}$ CPU cycles/s. It is also

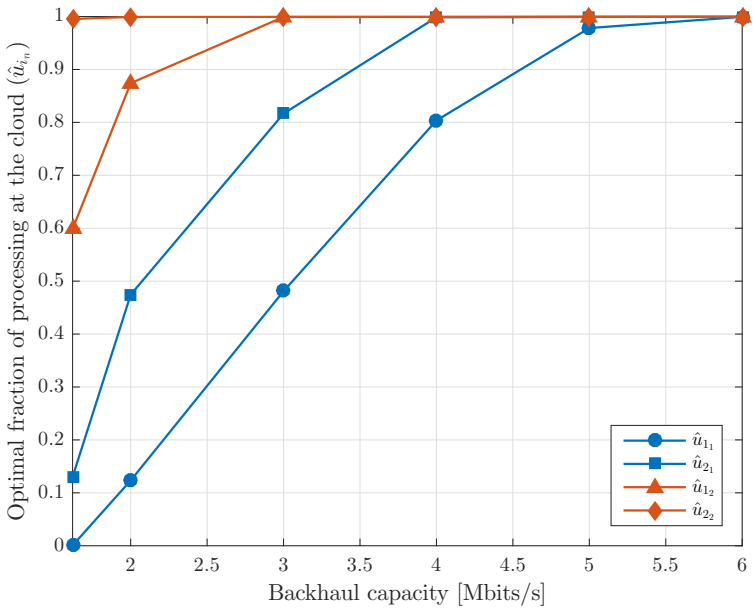


Figure 2.8 Offloading cloud vs. cloudlet indicator versus the backhaul capacity ($N_c = 2, K = 2, T_{i_n} = 0.9$ seconds, $B^I_{1_1} = B^O_{1_1} = 1$ Mbits, $B^I_{2_1} = B^O_{2_1} = 0.7$ Mbits, $B^I_{1_2} = B^O_{1_2} = 0.5$ Mbits, $B^I_{2_2} = B^O_{2_2} = 0.1$ Mbits; $V_{1_1} = 10^9$ CPU cycles, $V_{2_1} = 0.7V_{1_1}$, $V_{1_2} = 0.5V_{1_1}$ and $V_{2_2} = 0.1V_{1_1}$, $C^{ul}_n = C^{dl}_n = 100$ Mbits/s, $F^{\text{ceNB}}_n = 10^{10}$ CPU cycles/s, and $F_c = 10^{11}$ CPU cycles/s).

included, the performance of the solution introduced in Section 2.5, which allows for hybrid edge-cloud offloading. For cloud offloading, different values for computational capacity F_c of the cloud have been considered. It is seen that when the backhaul capacity constraint is stringent, edge offloading is more energy efficient as compared to cloud offloading, e.g., by about 44% for cloud computational capacity $F_c = 10^{10}$ CPU cycles/s and backhaul capacity $C^{ul}_n = C^{dl}_n = 2.4$ Mbits/s. Furthermore, larger values of cloud capacity F_c can compensate for limited backhaul resources, making cloud computing preferable to edge computing. It is also observed that the hybrid edge-cloud offloading scheme can significantly outperform edge computing when backhaul capacity is limited.

Finally, we turn to the performance of the downlink network MIMO scheme presented in Section 2.6. Figure 2.10 plots the average mobile sum-energy with and without cooperative transmission in the downlink. The performance of cooperative

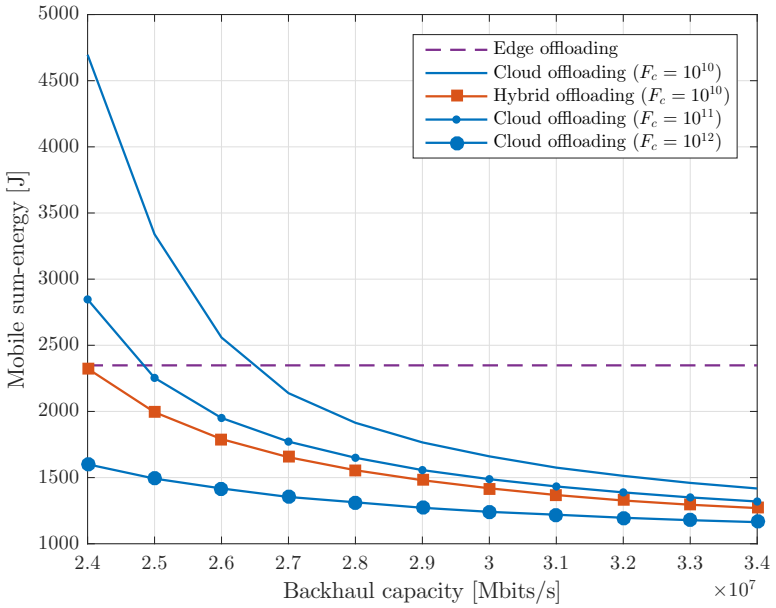


Figure 2.9 Minimum average mobile sum-energy consumption for cloud and edge offloading, along with the proposed hybrid edge-cloud offloading scheme, versus backhaul capacity $C_n^{ul} = C_n^{dl}$ ($N_c = 2, K = 2, T_{i_n} = 0.1$ s, $W^{ul} = W^{dl} = 10$ MHz, $B_{i_n}^I$ and $B_{i_n}^O \sim \mathcal{U}\{0.1, 1\}$ Mbits, $V_{i_n} = 2640 \times B_{i_n}^I$ CPU cycles, and $F_n^{\text{ceNB}} = 10^{10}$ CPU cycles/s).

scheme is obtained from the solution of (P.8), while the performance of non-cooperative scheme is obtained from the solution of (P.1) under the indicated values of backhaul capacity. The users offloading requirements are identical to that used in Figure 2.6. The key observation here is that the performance of the downlink cooperative transmission is strongly limited by the backhaul capacity. For instance, one can see that, with backhaul $C_n^{ul} = C_n^{dl} = 10$ Mbits/s, the cooperative scheme is about 85% more energy-consuming as compared to the non-cooperative scheme at latency 0.5 seconds. The energy performance gap between the two schemes is diminished as the backhaul increases as observed with $C_n^{ul} = C_n^{dl} = 100$ Mbits/s. With this value of backhaul capacity, the cooperative scheme is less energy efficient by 45% around $T_{i_n} = 0.3$ seconds. However, with backhaul $C_n^{ul} = C_n^{dl} = 10$ Gbits/s, as for a standard fiber optic channel, the cooperative scheme starts to have noticeable

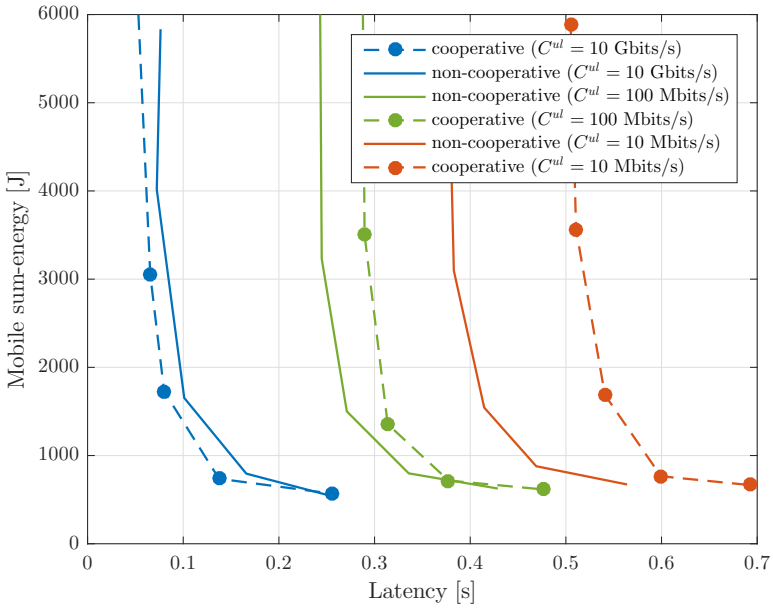


Figure 2.10 Minimum average mobile sum-energy consumption versus the latency constraint T_{i_n}, assumed to be the same for all MUs ($N_c = 2, K = 2$, $B^I_{i_n} = B^O_{i_n} = 1$ Mbits, $V_{i_n} = 10^9$ CPU cycles, $F_c = 10^{11}$ CPU cycles/s, and $C^{dl}_n = C^{ul}_n$).

energy gains. At latency 0.08 seconds, for example, the cooperative scheme attains 57% energy saving as compared to the non-cooperative scheme.

2.8 Concluding Remarks

In this chapter, we investigated the design of cloud mobile computing systems over MIMO cellular networks as a joint optimization problem over radio, computational resources and backhaul resources in both uplink and downlink directions. An iterative algorithm based on successive convex approximations was presented for solving the resulting non-convex problems under latency and power constraints. Numerical results show that the proposed joint optimization yields significant energy saving compared to the conventional solutions based on separate allocations of computing and/or backhaul resources. This saving is more pronounced in low latency regimes, where, in our results, it leads to energy saving as high as 66%. The mentioned baseline problem was further generalized in several directions. First, user selection

with the aim of ensuring an energy savings for all users that perform offloading is tackled. Then, a hybrid architecture that leverages both edge and cloud computing was studied by addressing the optimal allocation between cloud and edge. It was seen that the optimal allocation is significantly affected by the backhaul capacity. Finally, joint downlink transmission based on network MIMO was considered, demonstrating the critical importance of the backhaul for the viability of this technique.

CHAPTER 3

ENERGY-EFFICIENT RESOURCE ALLOCATION FOR MOBILE EDGE COMPUTING-BASED AUGMENTED REALITY APPLICATIONS

3.1 Introduction

The mobile applications considered for offloading in the previous chapter are assumed to be user-centric with each MU offloads its application independently. In this chapter, Augmented Reality (AR) type of mobile applications is considered for joint offloading by multiple MUs in an application-centric fashion as discussed next.

Augmented Reality (AR) mobile applications are gaining increasing attention due to the their ability to combine computer-generated data with the physical reality. AR applications are computational-intensive and delay-sensitive, and their execution on mobile devices is generally prohibitive when satisfying users' expectations in terms of battery lifetime [2,12,66]. To address this problem, it has been proposed to leverage mobile edge computing [2, 27, 32, 70]. Accordingly, users can offload the execution of the most time- and energy-consuming computations of AR applications to cloudlet servers via wireless access points. The use of local cloudlet servers is instrumental in providing the required real-time experience, as it forgoes the use of backhaul network to access a distant cloud server [2, 12, 70].

Nevertheless, the stringent delay requirements pose significant challenges to the offloading of AR application via mobile edge computing [2, 70]. A recent line of work has demonstrated that it is possible to significantly reduce mobile energy consumption under latency constraints by performing a joint optimization of the allocation of communication and computational resources [28–31]. These investigations apply to generic applications run independently by different users. However, AR applications have the unique property that the applications of different users share part of the computational tasks and of the input and output data [2, 32]. Therefore, this

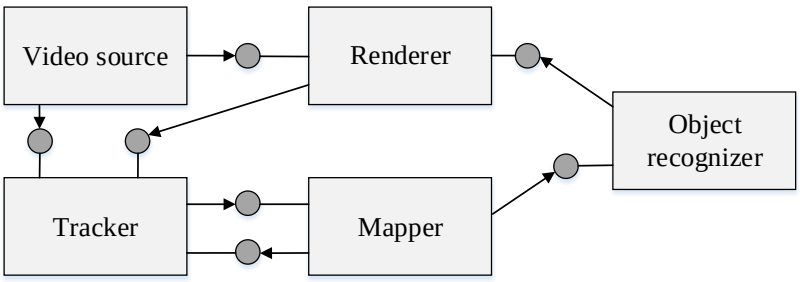


Figure 3.1 Example of a component-based model of an AR application [2]. The application includes the Video source and Renderer components, which need to be executed locally on the mobile device, and the three main components of Mapper, Tracker and Object recognizer, which may be offloaded.

chapter proposes to leverage this property to reduce communication and computation overhead via the joint optimization of communication and computational resources.

To illustrate the problem at hand, consider the class of AR applications that superimpose artificial images into the real world through the screen of a mobile device as described in [2, 32]. The block diagram of such applications shown in Figure 3.1 identifies the following components [2, 32]: (*i*) *Video source*, which obtain raw video frames from the mobile camera; (*ii*) *Tracker*, which tracks the position of the user with respect to the environment; (*iii*) *Mapper*, which builds a model of the environment; (*iv*) *Object recognizer*, which identifies known objects in the environment; and (*v*) *Renderer*, which prepares the processed frames for display. The Video source and Renderer components must be executed locally at the mobile devices, while the most computationally intensive Tracker, Mapper and Object recognizer components can be offloaded. Moreover, the input and output data, as well as the computational tasks of the offloaded components, can be partially shared among users. In fact, Mapper and Object recognizer can collect inputs from all the users located in the same area, potentially limiting the transmission of redundant information in the uplink. In a similar manner, the outcome of the Mapper and Object recognizer components can be multicast to all co-located users in the downlink.

In this chapter, unlike prior papers [28–31], the problem of minimizing the total mobile energy expenditure for offloading under latency constraints over communication and computation parameters by explicitly accounting for the discussed *collaborative* nature of AR applications is tackled in Section 3.3. Section 3.4 introduces the proposed Successive Convex Approximation (SCA) [78, 79] solution. Numerical results are finally provided in Section 3.5.

3.2 System Model

Consider the mobile edge computing system illustrated in Figure 3.2, in which K users in a set $\mathcal{K} = \{1, \ldots, K\}$ run a computation-intensive AR application on their single-antenna mobile devices with the aid of a cloudlet server. The server is attached to a single-antenna Base Station (BS), which serves all the users in the cell using Time Division Duplex (TDD) over a frequency-flat fading channel. Following the discussion in Section 3.1, it is assumed that the offloaded application has shared inputs, outputs and computational tasks, which pertain to the Tracker, Mapper and Object recognizer components. To elaborate, let us first review the more conventional set-up, studied in, e.g., [28–31], in which users perform the offloading of *separate* and independent applications. In this case, offloading for each user k would require: (i) *Uplink:* transmitting a number B_k^I input bits from each user k to the cloudlet in the uplink; (ii) *Cloudlet processing:* processing the input by executing V_k CPU cycles at the cloudlet; (iii) *Downlink:* transmitting B_k^O bits from the cloudlet to each user k in the downlink. In contrast, as discussed in Sec. 3.1, the collaborative nature of the Tracker, Mapper and Object recognizer components (recall Figure 3.1) can be leveraged to reduce mobile energy consumption and offloading latency, as detailed next. One can note that the non-collaborative components can potentially also be carried out locally if this reduces energy consumption (see, e.g., [28–31]). The study of the optimization of this aspect is considered in future work. In contrast, as

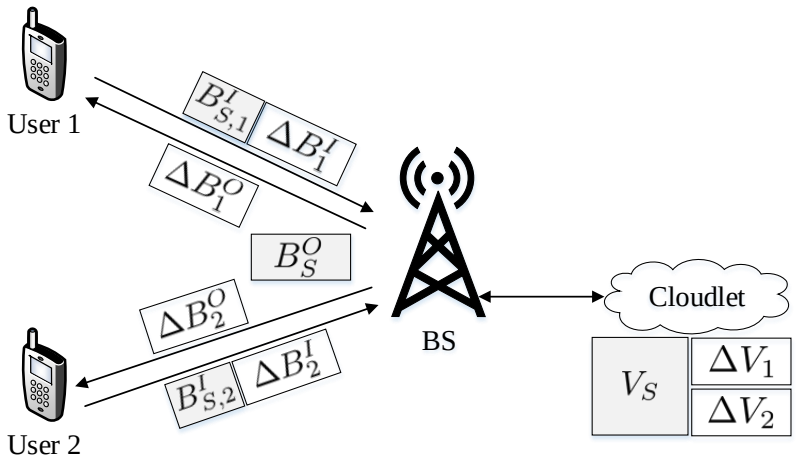


Figure 3.2 Offloading of an AR application to a cloudlet attached to the BS. Shared data and computations are shaded.

illustrated in Figure 3.2, for AR applications, the following collaborative features can be leveraged to reduce mobile energy consumption and offloading latency.

1) *Shared uplink transmission:* A given subset of $B^I_S \leq \min_k\{B^I_k\}$ input bits is shared among the users in the sense that it can be sent by *any* of the users to the cloudlet. For example, the input bits to the offloaded Object recognizer component in Figure 3.1 can be sent by any of the users in the same area. As a result, each user k transmits a fraction of $B^I_{S,k}$ bits of the B^I_S shared bits, which can be optimized under the constraint $\sum_k B^I_{S,k} = B^I_S$, as well as $\Delta B^I_k = B^I_k - B^I_{S,k}$ bits that need to be uploaded exclusively by user k.

2) *Shared cloudlet processing:* Part of the computational effort of the cloudlet is spent producing output bits of interest to all users. An example is the computational task of updating the model of the environment carried out by the mobiles. Therefore, we assume that $V_S \leq \min_k\{V_k\}$ CPU cycles are shared, whereas $\Delta V_k = V_k - V_S$ CPU cycles are to be executed for each user k.

3) *Multicast downlink transmission:* Some of the output bits need to be delivered to all users. For example, a co-located group of users may need the output bits from

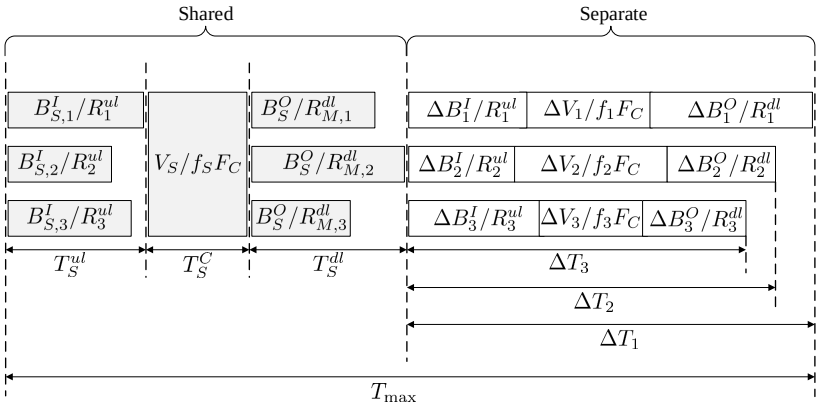


Figure 3.3 Data frame structure for a system with $K = 3$ users. The preamble containing training sequences is not shown.

the Mapper component for a map update. To model this, we assume that a subset of $B^O_S \leq \min_k\{B^O_k\}$ output bits can be transmitted in multicast mode to all users in the cell, while $\Delta B^O_k = B^O_k - B^O_S$ bits need to be transmitted to each user in a unicast manner.

The frame structure is detailed in Figure 3.3. Not shown are the training sequences sent by the users prior to the start of the data transmission frame, which enable the BS to estimate the uplink Channel State Information (CSI), and hence also the downlink CSI due to reciprocity. The CSI is assumed to remain constant for the frame duration. As seen in Figure 3.3, in the data frame, the shared communication and computation tasks are carried out first, followed by the conventional separate offloading tasks, as detailed next.

1) *Uplink transmission:* The achievable rate, in bits/s, for transmitting the input bits of user k in the uplink is given by

$$R^{ul}_k(P^{ul}_k) = \frac{W^{ul}}{K}\log_2\left(1 + \frac{\gamma_k P^{ul}_k}{N_0 W^{ul}/K}\right), \tag{3.1}$$

where P^{ul}_k is the transmit power of the mobile device of user k; the uplink bandwidth W^{ul} is equally divided among the K users, e.g., using OFDMA; γ_k is the uplink

and downlink channel power gains of user k; and N_0 is the noise power spectral density at the receiver. Referring to Figure 3.3 for an illustration, the time, in seconds, necessary to complete the shared uplink transmissions is defined as $T_S^{ul} = \max_k B_{S,k}^I / R_k^{ul}(P_k^{ul})$, whereas the time needed for user k to transmit the separate ΔB_k^I bits is $\Delta B_k^I / R_k^{ul}(P_k^{ul})$. The corresponding mobile energy consumption due to uplink transmission is

$$E_k^{ul}(P_k^{ul}, B_{S,k}^I) = \left(\frac{P_k^{ul}}{R_k^{ul}(P_k^{ul})} + l_k^{ul} \right) \left(B_{S,k}^I + \Delta B_k^I \right), \tag{3.2}$$

where l_k^{ul} is a parameter that indicates the amount of energy spent by the mobile device to extract each bit of offloaded data from the video source. In (3.2) and in subsequent equations, it is made explicit the dependence on variables to be optimized.

2) *Cloudlet processing:* Let F_C be the capacity of the cloudlet server in terms of number of CPU cycles per second. Also, let $f_k \geq 0$ and $f_S \geq 0$ be the fractions, to be optimized, of the processing power F_C assigned to run the ΔV_k CPU cycles exclusively for user k and the V_S shared CPU cycles, respectively, so that $\sum_{k \in \mathcal{K}} f_k \leq 1$ and $f_S \leq 1$. As shown in Figure 3.3, the execution time for the shared CPU cycles is $T_S^C = V_S / (f_S F_C)$ and the time needed to execute ΔV_k CPU cycles of interest to user k remotely is $\Delta V_k / (f_k F_C)$.

3) *Downlink transmission:* The common output bits B_S^O are multicast to all users. Let P_M^{dl} be the transmit power for multicasting, which is subject to the optimization. The resulting achievable downlink rate for user k is given by

$$R_{M,k}^{dl}(P_M^{dl}) = W^{dl} \log_2 \left(1 + \frac{\gamma_k P_M^{dl}}{N_0 W^{dl}} \right), \tag{3.3}$$

with $R_M^{dl}(P_M^{dl}) = \min_k R_{M,k}^{dl}(P_M^{dl})$, with W^{dl} being the downlink bandwidth. The downlink transmission time to multicast B_S^O bits can hence be computed as $T_S^{dl} = B_S^O / R_M^{dl}(P_M^{dl})$ (see Figure 3.3). The ΔB_k^O output bits intended exclusively for each user k are sent in a unicast manner in downlink using an equal bandwidth allocation,

with rate

$$R_k^{dl}(P_k^{dl}) = \frac{W^{dl}}{K}\log_2\left(1+\frac{\gamma_k P_k^{dl}}{N_0 W^{dl}/K}\right), \tag{3.4}$$

where P_k^{dl} is the BS transmit power allocated to serve user k. The overall downlink mobile energy consumption for user k is

$$E_k^{dl}(P_k^{dl}, P_M^{dl}) = \left(\frac{\Delta B_k^O}{R_k^{dl}(P_k^{dl})} + \frac{B_S^O}{R_{M,k}^{dl}(P_M^{dl})}\right) l_k^{dl}, \tag{3.5}$$

where l_k^{dl} is a parameter that captures the mobile receiving energy expenditure per second in the downlink. Extensions that consider the optimization of the bandwidth allocation across users for uplink and downlink are left for future work.

3.3 Energy-Efficient Resource Allocation

In this section, the resource allocation problem that considers the minimization of the mobile sum-energy required for offloading across all users under latency and power constraints is tackled. Stated in mathematical terms, one can consider the following problem:

$$\begin{aligned}
\min_{\mathbf{z}} \quad & \sum_{k\in\mathcal{K}} E_k^{ul}(P_k^{ul}, B_{S,k}^I) + E_k^{dl}(P_k^{dl}, P_M^{dl}) \\
\text{s.t.} \quad & \mathbf{C.1} \quad \tfrac{\Delta B_k^I}{R_k^{ul}(P_k^{ul})} + \tfrac{\Delta V_k}{f_k F_C} + \tfrac{V_S}{f_S F_C} + \tfrac{\Delta B_k^O}{R_k^{dl}(P_k^{dl})} \leq T_{\max} - T_{\mathrm{S}}^{ul} - T_{\mathrm{S}}^{dl}, \forall k \in \mathcal{K}, \\
& \mathbf{C.2} \quad \tfrac{B_{S,k}^I}{R_k^{ul}(P_k^{ul})} \leq T_{\mathrm{S}}^{ul}, \forall k \in \mathcal{K}, \\
& \mathbf{C.3} \quad \tfrac{B_S^O}{R_{M,k}^{dl}(P_M^{dl})} \leq T_{\mathrm{S}}^{dl}, \forall k \in \mathcal{K}, \\
& \mathbf{C.4} \quad \sum_{k\in\mathcal{K}} f_k \leq 1; 0 \leq f_S \leq 1; f_k \geq 0, \forall k \in \mathcal{K}, \\
& \mathbf{C.5} \quad \sum_{k\in\mathcal{K}} B_{S,k}^I = B_S^I, \\
& \mathbf{C.6} \quad \sum_{k\in\mathcal{K}} P_k^{dl} \leq P_{\max}^{dl}; P_M^{dl} \leq P_{\max}^{dl}; P_k^{ul} \leq P_{\max}^{ul}, \forall k \in \mathcal{K}.
\end{aligned} \tag{P.9}$$

The optimization variables are collected in vector $\mathbf{z} \triangleq \left(\mathbf{P}^{ul}, \mathbf{B}_S^I, \mathbf{f}, \mathbf{P}^{dl}, P_M^{dl}, T_{\mathrm{S}}^{ul}, T_{\mathrm{S}}^{dl}\right)$, where $\mathbf{P}^{ul} \triangleq (P_k^{ul})_{k\in\mathcal{K}}$, $\mathbf{B}_S^I \triangleq (B_{S,k}^I)_{k\in\mathcal{K}}$, $\mathbf{f} \triangleq ((f_k)_{k\in\mathcal{K}}, f_S)$, $\mathbf{P}^{dl} \triangleq (P_k^{dl})_{k\in\mathcal{K}}$, and

we defined $\mathcal{Z}$ as the feasible set of problem (P.9). As illustrated in Figure 3.3, constraints C.1-C.3 enforce that the execution time of the offloaded application to be less than or equal to the maximum latency of $T_{\max}$ seconds. Constraints C.4-C.5 impose the conservation of computational resources and shared input bits, and C.6 enforces transmit power constraints at BS and users.

Problem (P.9) is not convex because of the non-convexity of the energy function $E_k^{ul}(P_k^{ul}, B_{S,k}^I)$ and of the latency constraints C.2, which can be rewritten as $g_k(P_k^{ul}, B_{S,k}^I) \leq T_S^{ul}, \forall k \in \mathcal{K}$. This issue is addressed by developing an SCA solution following [78, 79] in the next section. Theorem 2 in [78] shows that the SCA algorithm converges to a stationary point, and hence in practice to a local optimum, of the (NP-hard) problem (P.9). Furthermore, such convergence requires a number of iterations proportional to $1/\epsilon$, where ϵ measures the desired accuracy in terms of the stationarity metric $\|F(\mathbf{z})\|_2^2$ defined in [93, Eq. (6)].

3.4 The Proposed SCA Solution

In order to apply the SCA method, one needs to derive convex approximants for the functions $E_k^{ul}(P_k^{ul}, B_{S,k}^I)$ and $g_k(P_k^{ul}, B_{S,k}^I)$ that satisfy the conditions specified previously in Section 2.3 and also in [78, Section II]. Using such approximants, the SCA scheme detailed in Algorithm 4 is obtained. In the algorithm, at each iteration v, the unique solution $\hat{\mathbf{z}}(\mathbf{z}(v)) \triangleq \left(\hat{\mathbf{P}}^{ul}, \hat{\mathbf{B}}_S^I, \hat{\mathbf{f}}, \hat{\mathbf{P}}^{dl}, \hat{P}_M^{dl}, \hat{T}_S^{ul}, \hat{T}_S^{dl}\right)$ of the following strongly convex problem

$$\begin{aligned}
\hat{\mathbf{z}}(\mathbf{z}(v)) \triangleq \underset{\mathbf{z}}{\operatorname{argmin}} & \sum_{k\in\mathcal{K}} \tilde{E}(\mathbf{z}_k; \mathbf{z}_k(v)) \\
& \text{s.t.} \\
\mathbf{C.2}\ \ & \tilde{g}_k\left(P_k^{ul}, B_{S,k}^I; P_k^{ul}(v), B_{S,k}^I(v)\right) \leq T_{\mathrm{S}}^{ul}, \forall k \in \mathcal{K}, \\
\mathbf{C.1}, \mathbf{C.3} - \mathbf{C.6}\ \ & \text{of (P.9)},
\end{aligned} \tag{P.10}$$

is obtained, where we have defined $\mathbf{z}_k \triangleq \left(P_k^{ul}, B_{S,k}^I, f_k, f_S, P_k^{dl}, P_M^{dl}, T_S^{ul}, T_S^{dl}\right)$ as well as $\tilde{E}_k\left(\mathbf{z}_k; \mathbf{z}_k\left(v\right)\right) \triangleq \tilde{E}_k^{ul}\left(\mathbf{z}_k; \mathbf{z}_k\left(v\right)\right) + E_k^{dl}(P_k^{dl}, P_M^{dl})$. The approximants functions $\tilde{E}_k^{ul}\left(\bullet;\bullet\right)$ and $\tilde{g}_k\left(\bullet;\bullet\right)$ are discussed next.

The approximant $\tilde{E}_k^{ul}\left(\mathbf{z}_k; \mathbf{z}_k\left(v\right)\right)$ around the current feasible iterate $\mathbf{z}_k(v)$ can be obtained following [78, Section III, Example #8] as

$$\begin{aligned}\tilde{E}_k^{ul}\left(\mathbf{z}_k; \mathbf{z}_k\left(v\right)\right) = &\frac{P_k^{ul}(v)\left(B_{S,k}^I(v) + \Delta B_k^I\right)}{R_k^{ul}\left(P_k^{ul}\right)} + \frac{P_k^{ul}(v)\left(B_{S,k}^I + \Delta B_k^I\right)}{R_k^{ul}\left(P_k^{ul}(v)\right)} \\ &+ \frac{P_k^{ul}\left(B_{S,k}^I(v) + \Delta B_k^I\right)}{R_k^{ul}\left(P_k^{ul}(v)\right)} + \bar{E}_k^{ul}\left(\mathbf{z}_k; \mathbf{z}_k(v)\right) + l_k^{ul}\left(B_{S,k}^I + \Delta B_k^I\right),\end{aligned} \tag{3.6}$$

where $\bar{E}_k^{ul}\left(\mathbf{z}_k; \mathbf{z}_k(v)\right) \triangleq \left(\mathbf{z}_k - \mathbf{z}_k(v)\right)^T \Psi \left(\mathbf{z}_k - \mathbf{z}_k(v)\right)$, with Ψ being a diagonal matrix with non-negative elements $\tau_{P^{ul}}, \tau_{B_S^I}, \tau_f, \tau_{f_S}, \tau_{P^{dl}}, \tau_{P_M^{dl}}, \tau_{T_S^{ul}}$ and $\tau_{T_S^{dl}}$. For the second approximant, in light of the relation $g(x_1, x_2) = x_1 x_2 = 1/2(x_1 + x_2)^2 - 1/2(x_1^2 + x_2^2)$, a convex upper bound is obtained as requested by SCA by linearizing the concave part of $g_k(P_k^{ul}, B_{S,k}^I)$ [78, Section III, Example #4], which results in

$$\begin{aligned}\tilde{g}_k\left(P_k^{ul}, B_{S,k}^I; P_k^{ul}(v), B_{S,k}^I(v)\right) = \frac{1}{2}\Bigg(&\left(B_{S,k}^I + \frac{1}{R_k^{ul}(P_k^{ul})}\right)^2 - \left(B_{S,k}^I(v)\right)^2 - \\ &\left(\frac{1}{R_k^{ul}(P_k^{ul}(v))^2}\right)\Bigg) - \Bigg(\left(B_{S,k}^I(v)\left(B_{S,k}^I - B_{S,k}^I(v)\right)\right. \\ &- \frac{R_k^{ul}\left(P_k^{ul}(v)\right)}{\left(1 + \frac{\gamma_k P_k^{ul}(v)}{N_0 W^{ul}/K}\right) R_k^{ul}\left(P_k^{ul}(v)\right)^4} \\ &\left.\left(\frac{1}{R_k^{ul}\left(P_k^{ul}\right)} - \frac{1}{R_k^{ul}\left(P_k^{ul}(v)\right)}\right)\Bigg).\end{aligned} \tag{3.7}$$

The convexity of (3.7) is established by noting that the second term in the right-hand side is the reciprocal of the rate function (concave and positive) and the fourth power (convex and non-decreasing) of a convex function is convex [87].

Algorithm 4: SCA Solution for (P.10)

Input: $\mathbf{z}(0) \in \mathcal{Z}$; $\alpha = 10^{-5}$; $\epsilon = 10^{-5}$; $v = 0$;

$\tau_{P^{ul}}, \tau_{B_S^I}, \tau_f, \tau_{f_S}, \tau_{P^{dl}}, \tau_{P_M^{dl}}, \tau_{T_S^{ul}}, \tau_{T_S^{dl}} > 0$.

1: Compute $\hat{\mathbf{z}}(\mathbf{z}(v))$ from (P.10).

2: Set $\mathbf{z}(v+1) = \mathbf{z}(v) + \delta(v)(\hat{\mathbf{z}}(\mathbf{z}(v)) - \mathbf{z}(v))$, with $\delta(v) = \delta(v-1)(1 - \alpha\delta(v-1))$.

3: If $\|F(\mathbf{z}(v))\|_2^2 \leq \epsilon$, stop.

4: Otherwise, set $v \leftarrow v + 1$, and return to step 1.

Output: $\mathbf{z} \triangleq \left(\mathbf{P}^{ul}, \mathbf{B}_S^I, \mathbf{f}, \mathbf{P}^{dl}, P_M^{dl}, T_S^{ul}, T_S^{dl}\right)$

3.5 Numerical Results

In this section, numerical examples are provided with the aim of illustrating the advantages that can be accrued by leveraging the collaborative nature of AR applications for mobile edge computing. We consider a scenario where eight users are randomly deployed in a small cell. The radio channels are Rayleigh fading and the path loss coefficient is obtained based on the small-cell model in [94] for a carrier frequency of 2 GHz. The users' distances to the BS are randomly uniformly selected between 100 and 1000 meters and the results are averaged over multiple independent drops of users' location and of the fading channels. The noise power spectral density is set to $N_0 = -147$ dBm/Hz. The uplink and downlink bandwidth is $W^{ul} = W^{dl} = 10$ MHz. The uplink and downlink power budgets are constrained to the values $P_{\max}^{ul} = 50$ and $P_{\max}^{dl} = 60$ dBm, respectively. The cloudlet server processing capacity is $F_C = 10^{10}$ CPU cycles/s [2]. We also set $l_k^{ul} = 1.78 \times 10^{-6}$ J/bit [95] and $l_k^{dl} = 0.625$ J/s [92].

The size of the input data generally depends on the number and size of the features of the video sources obtained by the mobiles that are to be processed at the cloudlet. Here the selected value is $B_k^I = 1$ Mbits, which may correspond to the

transmission of 1024×768 images [27]. A fraction of the input bits $B_S^I = \eta B_k^I$ bits can be transmitted cooperatively by all users for some parameter $0 \leq \eta \leq 1$. The required CPU cycles of the offloaded components is set to $V_k = 2640 \times B_k^I$ CPU cycles, representing a computational intensive task [35]. The shared CPU cycles are assumed to be $V_S = \eta V_k$ for the same sharing factor η. The output bits are assumed to equal the amount of input bits $B_k^O = B_k^I = 1$ Mbits with shared fraction $B_S^O = \eta B_k^O$. Practical latency constraints for AR applications are of the order of 0.01 s [27, 32]. Throughout the experiments, we found that the accuracy $\epsilon = 10^{-5}$ was obtained within no more than 25 iterations.

For reference, the performance of the proposed scheme, in which uplink and downlink transmissions and cloudlet computations are shared as described, is compared with the following offloading solutions: *(i) Shared Cloudlet Processing and Downlink Transmission*: CPU cycles and output data are shared as described in Section 3.2, while the input bits B_k^I are transmitted by each user individually, i.e., set $B_{S,k}^I = 0$ and $\Delta B_k^I = B_k^I$ for all $k \in \mathcal{K}$; and *(ii) Shared Uplink Transmission*: Only the input bits are shared as discussed, while no sharing of computation and downlink transmission takes place, i.e., $\Delta V_k = V_k$ and $\Delta B_k^O = B_k^O$ for all $k \in \mathcal{K}$. We also include for reference the result obtained by solving problem (P.9) using the global optimization BARON software running on NEOS server with a global optimality tolerance of 10^{-6} [96]. As shown in Figure 3.4, for $T_{\max} = 0.05$ s and $\eta = 0.3$, the *Shared Cloudlet Processing and Downlink Transmission* scheme achieves energy saving about 37% compared to separate offloading (which sets $\Delta B_k^I = B_k^I$, $\Delta V_k = V_k$ and $\Delta B_k^O = B_k^O$ for all $k \in \mathcal{K}$). This gain can be attributed to the increased time available for uplink transmission due to the shorter execution and downlink transmission periods, which reduces the associated offloading energy. Under the same conditions, the energy saving of around 50% with respect to separate offloading brought by *Shared Uplink Transmission* is due to the ability of the system to adjust

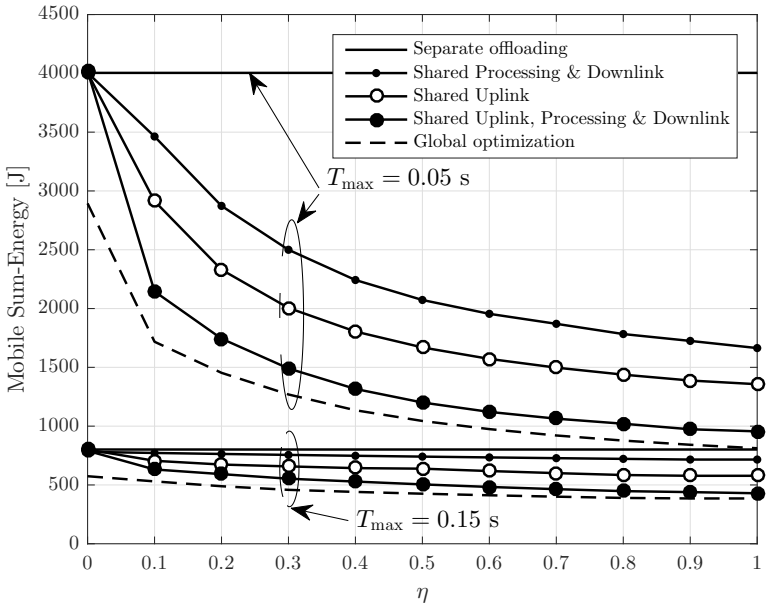


Figure 3.4 Average mobile sum-energy consumption versus the fraction η of shared data in uplink and downlink and of shared CPU cycles executed at the cloudlet.

the fractions of shared data transmitted by each user in the uplink based on the current channel conditions. These two gains combine to yield the energy saving of the proposed shared data offloading scheme with respect to the conventional separate offloading of around 63%. Both separate and shared offloading schemes have similar energy performance for the relaxed delay requirement of $T_{\max} = 0.15$ s, which can be met with minimal mobile energy expenditure even without sharing communication and computation resources. The figure also shows that SCA yields a solution that is close to the global optimum, for this example.

3.6 Concluding Remarks

Mobile edge computing enables the provision of computationally demanding AR applications on mobile devices. AR mobile applications have inherent collaborative properties in terms of data collection in the uplink, computing at the edge, and data delivery in the downlink. In this chapter, a resource allocation approach is proposed whereby transmitted, received and processed data are shared partially among the

users to obtain an efficient utilization of the communication and computation resources. The approach, implemented via Successive Convex Approximation (SCA), is seen to yield considerable gains in mobile energy consumption as compared to the conventional independent offloading across users.

CHAPTER 4

CODED NETWORK FUNCTION VIRTUALIZATION: FAULT TOLERANCE VIA IN-NETWORK CODING

4.1 Introduction

As discussed in Chapter 1, network functions in NFV and computing functionality in MCC are both implemented via software virtualization techniques. Network Function Virtualization (NFV) is a novel architectural paradigm for cellular wireless networks that has been put forth within the European Telecommunications Standards Institute (ETSI) with the goal of simplifying network management, update and operation [15]. NFV decouples the Network Functions (NFs), such as baseband processing at the base stations and firewalling or routing at the core network, from the physical network equipment on which they run. This is done by leveraging virtualization technology in order to map NFs into Virtual Network Functions (VNFs) that are instantiated on Commercial Off-The-Shelf (COTS) hardware resources, such as servers, storage devices and switches [13, 14]. NFV enables an adaptive "slicing" of the available network physical resources so as to accommodate different network services, e.g., mobile broadband, machine-type or ultra-reliable communications [10].

A simplified view of the NFV architecture is illustrated in Figure 4.1 [10, 13, 14, 16]. The top layer in this architecture describes the logical functionality of the given network service as a so-called forwarding graph, which characterizes the functional relationship among the VNFs that implement the network service. The bottom layer contains the general-purpose hardware appliances that provide storage, computation and networking capabilities. Finally, the intermediate virtualization layer is responsible for mapping VNFs to physical resources. In the example of Figure 4.1, VNF1 and VNF2 are instantiated on Virtual Machines (VMs) running on Server 1, while VNF3 and VNF4 are instantiated on VMs running on Server 2 and Server 3,

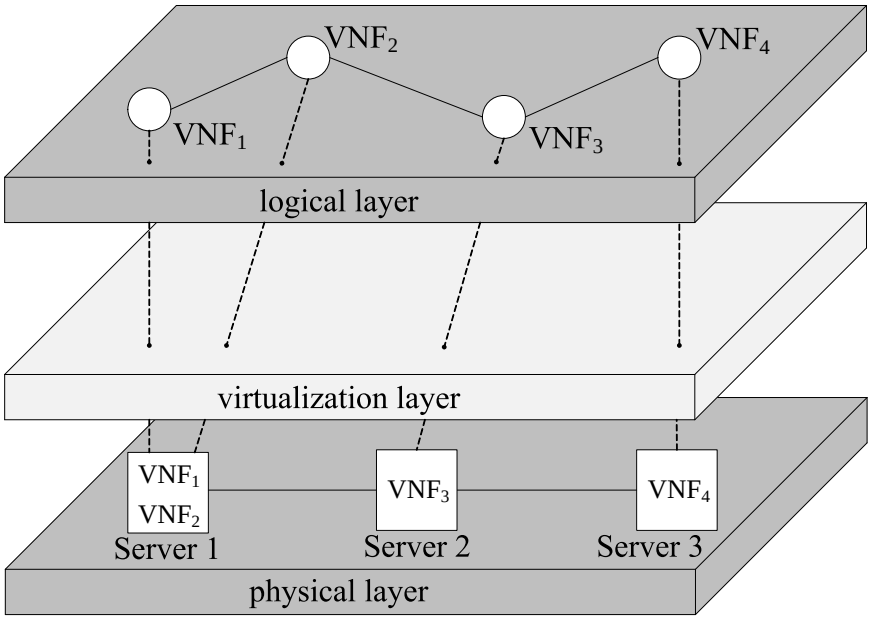


Figure 4.1 Simplified architectural view of NFV.

respectively. The combination of the last two layers constitutes the Network Function Virtualization Infrastructure (NFVI), and the three planes are under the control of a Network Functions Virtualization Management and Orchestration (NFV-MANO) block (see [13–15] for details). Most research activity on NFV focuses on the design of mapping rules between VNFs and hardware resources via the solution of mixed integer problems (see, e.g., [77]).

One of the key challenges for the adoption and deployment of NFV is the fact that COTS hardware is significantly less reliable than the dedicated network devices used in conventional network deployments [14, Section VI]. Hardware outages may in fact be caused by random failures, intentional attacks, software malfunction or disasters. This problem is motivating an emerging line of work, also within ETSI, on developing fault-tolerant virtualization strategies for NFV [16–18]. The typical solution, as summarized in [17], is to adapt to NFV well established policies introduced in the context of virtualization for data centers. These strategies are based on *overprovisioning* and *diversity*: NFs are split into multiple constituents VNFs, which

are then mapped on VMs instantiated on multiple distributed servers in order to minimize the probability of a disruptive failure as well as the mean time to recovery from a failure.

In this chapter, a novel principle for the design of fault-tolerant NFV that moves from *diversity-based* solutions to *coded* solutions is proposed. The proposed approach addresses the NF of uplink data decoding in a Cloud Radio Access Network (C-RAN) architecture, in which the baseband processing operations of the base station are carried out remotely at the "cloud" [82]. The focus on uplink channel decoding is dictated by the fact that the latter is known to be among the most demanding baseband functions in terms of computational complexity (see, e.g., [97, 98]).

The proposed coded NFV solution leverages the algebraic structure of the transmitted coded data frames in order to enhance the robustness of channel decoding. To elaborate, assume that there are a number of servers on which VMs carrying out channel decoding can be instantiated, as illustrated in Figure 4.2. A conventional diversity-based technique would duplicate the decoding task at multiple servers. In this chapter, instead, a coded approach is proposed, whereby received data frames are encoded prior to being processed by the VMs that implement decoding at the distributed servers. This chapter elaborates on a simple embodiment of this idea, which is illustrated in Figure 4.3 and introduced in Section 4.3 after a description of the system model in Section 4.2. Numerical examples are provided in Section 4.4. Extensions and more general applications of the principle of coded NFV are presented in Section 4.5.

4.2 System Model

Consider a C-RAN system implemented by means of NFV, and focus on the implementation of the NF of uplink channel decoding. In this system, as illustrated in Figure 4.2, a Remote Radio Head (RRH) is connected to the cloud by means of a

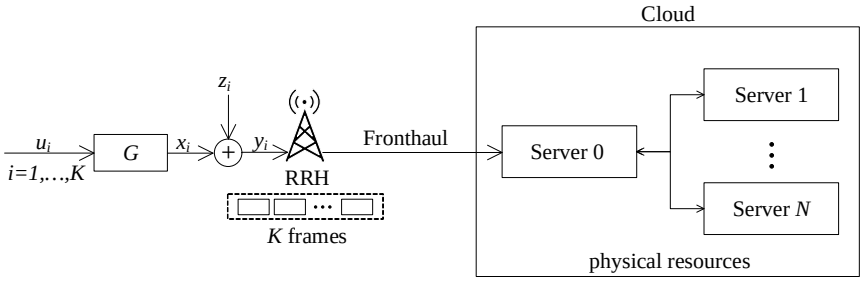


Figure 4.2 NFV set-up for the virtualization of the NF of uplink channel decoding. Server 0 acts as controller and is assumed to be reliable, while servers $1, \ldots, N$ can carry out the computationally heavy NF of channel decoding but may be unavailable, due to failures or overload, with probability q.

fronthaul link. The RRH forwards the received baseband packets to the cloud on the fronthaul link in order to enable channel decoding. The overprovisioning of hardware resources is assumed, such that N servers are available in the cloud, each of which can run a VM performing the decoding of a single received frame in the allotted time. More specifically, due to latency constraints, the decoding of K received data frames should be carried out on the servers by allocating at most one frame to decode to each of $N \geq K$ servers. As in conventional implementations (see, e.g., [17]), we further assume that the VMs implementing channel decoding on Servers $1, \ldots, N$ are managed by a controller VM, which is characterized by lower computational requirements and is instantiated at a server, marked as Server 0 in Figure 4.2, that is connected with bidirectional links to Servers $1, \ldots, N$.

Servers $1, \ldots, N$ are distributed strategically across multiple locations throughout the service provider's network, and are hence assumed to have independent availabilities [17]. In particular, it is by assumption that each one of Servers $1, \ldots, N$ fails independently with probability q. It is emphasized that a failure here means that a server is not available to perform the given task within an acceptable deadline due to software or hardware issues (see Section 4.1).

The transmitted frames are encoded with the same (n, k) linear code with a given rate k/n, such as convolutional, turbo or LDPC codes. Furthermore, in order

to present the key ideas, we consider first a Binary Symmetric Channel (BSC) model between the user under study and the RRH. As a result, for each transmitted frame $x_i \in \{0,1\}^n$, with $i \in \{1,\dots,K\}$, the transmitted signal can be written as $x_i = u_i G$, where $u_i \in \{0,1\}^k$ represents the data encoded in the i-th frame and G is the $k \times n$ generator matrix of the code. Furthermore, the signal received for the i-th frame is given as $y_i = x_i \oplus z_i$ where z_i is a vector of independent Bernoulli variables with probability p of being equal to 1. Generalizations of the system model are discussed in Section 4.5.

Throughout, we take as the performance metric of interest the probability of error, that is, the complement of the probability that decoding of all the K frames is carried out successfully by the cloud. Note that, according to the introduced model, a failure may occur due to either errors on the communication channel between user and RRH or due to a failure of the servers.

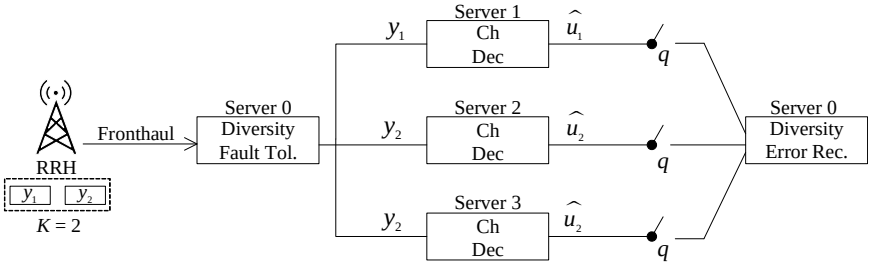
4.3 Fault Tolerance via Coded NFV

In this section, the conventional diversity-based fault-tolerant approach as applied to the problem at hand of uplink channel decoding in a C-RAN via NFV is first reviewed. Then, the proposed coded NFV approach is presented. For both schemes, the focus is on the case $N = 3$ and $K = 2$ and present a simple analysis of the probability of error. The problem statement in the general case is treated in Section 4.5.

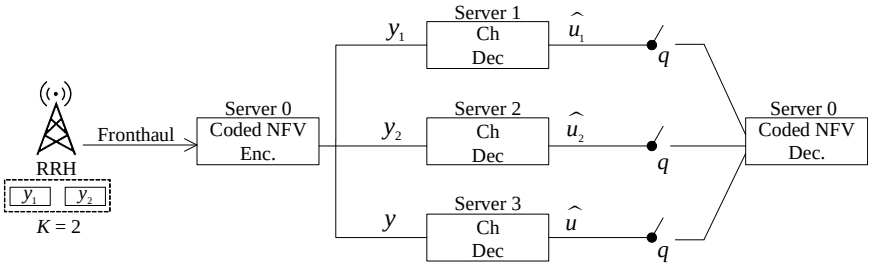
4.3.1 Fault Tolerance via Diversity

A conventional solution based on diversity is illustrated in Figure 4.3(a) for $N = 3$ servers and $K = 2$ frames. In this scheme, the controller VM instantiated at Server 0 duplicates one of the received frames, namely y_2 in the figure, at the input of both Server 2 and Server 3. Server 1, Server 2 and Server 3 each run a VM that performs channel decoding as well as error detection (via a Cyclic Redundancy Check test) on

the input frame. The outcome of the decoders is fed back to the VM in Server 0. It is noted that, for general values of N and K with $N > K$, the scheme would just duplicate one or more frames at the input of multiple servers.



(a) Diversity-based scheme



(b) Coded NFV scheme

Figure 4.3 Illustration of the idea of coded NFV for channel decoding in the case of an overprovisioning factor of $N/K = 3/2$ ($N = 3$ servers for $K = 2$ received frames): (a) Mapping of NF and servers for a conventional diversity-based scheme in which one of the received frames is duplicated at the input of two servers; (b) Mapping of NF and servers for the proposed coded NFV scheme in which the XOR of the two received frames $y = y_1 \oplus y_2$ is the input to Server 3, whose output is $\hat{u} = \hat{u}_1 \oplus \hat{u}_2$.

The conventional diversity-based system succeeds in decoding both packets as long as: (i) Server 1 decodes correctly data u_1 and is available; and (ii) Server 2 and/or Server 3 decode correctly u_2 and are available. As a consequence, the error probability can be written as

$$\mathrm{P}_{\mathrm{err}} = 1 - \sum_{\substack{\mathcal{S}\subseteq\{1,2,3\}:\\ |\mathcal{S}|\geq 2, \{1\}\in\mathcal{S}}} \Pr(\mathcal{S})(1-q)^{|\mathcal{S}|}, \tag{4.1}$$

where $|\mathcal{S}|$ is the cardinality of set $\mathcal{S}$, which is a subset of the Servers 1, 2 and 3; $\Pr(\mathcal{S})$ is the probability that only the decoders in $\mathcal{S}$ successfully decode the input frame, while the rest of the servers decode incorrectly.

4.3.2 Fault Tolerance via Coded NFV

In the proposed coded NFV scheme, as illustrated in Figure 4.3(b), Server 0 pre-processes the received frames by computing the linear combination $y = y_1 \oplus y_2$ of the received frames y_1 and y_2. Note that this operation is of much lower complexity as compared to channel decoding. Server 0 then assigns frame y_1 for decoding at Server 1, frame y_2 to Server 2 and frame y to Server 3. A key observation is that Server 3 can decode over the same linear code as Server 1 and 2 since we have

$$y = y_1 \oplus y_2 = (u_1 \oplus u_2)G \oplus (z_1 \oplus z_2). \tag{4.2}$$

Hence, Server 3 can decode $u = u_1 \oplus u_2$ over a BSC with parameter $2p(1-p)$, which is the probability that the effective noise $z_1 \oplus z_2$ equals 1. As a result, as long as *any* two servers decode successfully and are available, Server 0, which receives the outputs of all other servers as in the diversity-based scheme, can decode both data messages u_1 and u_2.

Based on the description above, the proposed coded NFV scheme can be interpreted as a form of *concatenated code* in which the outer linear code encodes each frame, while the inner NFV code is applied on the noisy received signals in order to obtain robustness with respect to infrastructure failures. The probability of error for this scheme is given by

$$\mathrm{P_{err}} = 1 - \sum_{\substack{\mathcal{S}\subseteq\{1,2,3\}:\\ |\mathcal{S}|\geq 2}} \Pr(\mathcal{S})(1-q)^{|\mathcal{S}|}, \tag{4.3}$$

where $\Pr(\mathcal{S})$ is defined as in (4.1), with the key difference that Server 3 decodes based on (4.2).

We conclude this section by emphasizing that, beside the advantages in terms of the error probability which will be further discussed in Section 4.4, the proposed coded scheme increases the Minimum Failure Removal (MFR) [18]. The MFR is the minimum number of servers whose removal leads to failure. In particular, with the conventional diversity-based scheme, even the non availability of a single server, namely Server 1, causes a failure, while the proposed scheme has a MFR of two.

4.4 Numerical Results

This section presents numerical experiments to compare the performance of the conventional diversity-based scheme and the proposed coded NFV for the presented example with $N = 3$ and $K = 2$. To this end, we consider a 1/2 feedforward convolutional code, in which the constraint length is 7, the code generator polynomial matrix is [171 133], with $k = 70$ and $n = 140$ and Viterbi decoders are implemented at Servers 1, 2 and 3. The probabilities $\Pr(\mathcal{S})$ in (4.1) and (4.3) are evaluated via Monte Carlo simulations.

The error probability as a function of the servers' failure probability q is plotted in Figure 4.4 for the indicated values of the BSC parameter p for both schemes. It is seen that, in the regime in which hardware failures have similar or smaller probability as compared to channel errors, coded NFV can provide significant gains. For instance, to achieve $\mathrm{P_{err}} \approx 2 \times 10^{-3}$ with $p = 0.05$, the conventional diversity-based method requires hardware with a server failure probability of $q = 10^{-4}$, while the coded NFV requires $q \approx 10^{-3}$, which is an order of magnitude larger.

4.5 Extensions

The robust coded NFV scheme was presented in Section 4.3.2 for $N = 3$, $K = 2$ and for a BSC channel between user and RRH. In this section, extensions are briefly discussed.

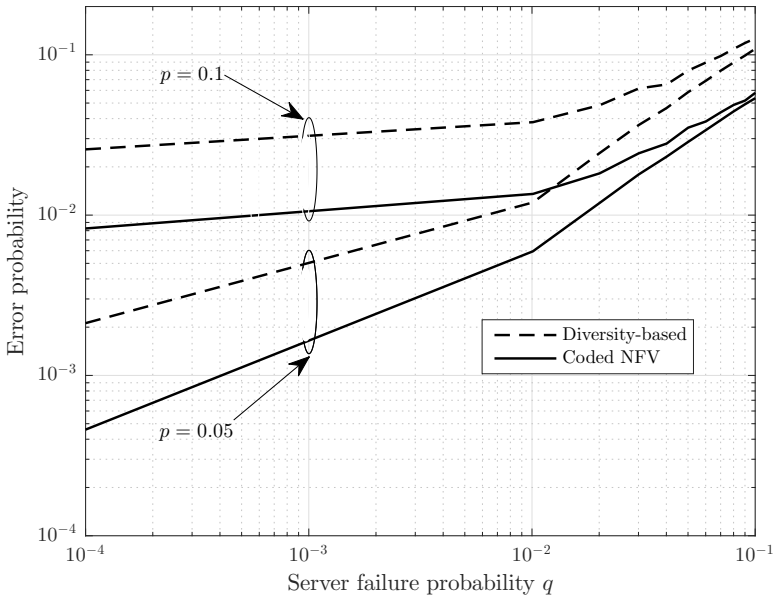


Figure 4.4 Error probability versus the server failure probability q for diversity-based and coded NFV schemes ($n = 140, k = 70, N = 3, K = 2$).

1) *Frames encoded with different rates*: Different rates can be accommodated by using rate-compatible codes obtained from the same master linear code for each frame.

2) *Additive Gaussian noise or fading channels*: For such channels between user and RRH, lattice codes can be used to encode the frames, instead of linear binary codes. The input to Server 3 (see Figure 4.3(b)) is computed as the sum of the received signals on the real field, or complex field for complex Gaussian or fading channels. Server 3 then decodes the XOR of the two messages, namely $u = u_1 \oplus u_2$, by decoding over the lattice code using the technique of computation over a multiple access channel (see [99] and references therein for an introduction).

3) *Generalization to any values of the parameters N and K*: For any values of N and K, each one of the n bits input to Server j, with $j = 1, \ldots, N$, is obtained as a binary-field linear combination of the corresponding bits of the received frames y_i, with $i = 1, \ldots, K$. The resulting NFV code can be then described by a $K \times N$ generator matrix G_{NFV} such that the input bits to the servers can be computed as

yG_{NFV}, with $y \in \{0,1\}^K$ collecting one bit from every frame. Note that $G_{\text{NFV}} = \left(\begin{smallmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{smallmatrix}\right)$ in the discussed $N = 3$, $K = 2$ example.

Regarding the design of the generator matrix G_{NFV}, one can note that an NFV code benefits from a sparse structure of G_{NFV}, as well as from a large minimum distance — two conflicting requirements. For the former requirement, as seen in Section 4.3.2, summing more received signals at the input of a server increases the noise level. More formally, the NFV code operates over an erasure channel in which the probability of error associated to each server k equals $q + (1-q)f(d_k)$, where d_k is the number of ones in the kth column of matrix G_{NFV} and $f(d)$ represents the probability of incorrect decoding when d received signals are summed (which is an increasing function of d). This novel property of NFV codes sets an interesting research challenge for code design.

4.6 Concluding Remarks

Software-based virtual network functions enabled by NFV are less reliable than those provided via the traditional hardware-based platforms. To alleviate this shortcoming, this chapter proposed to enhance traditional diversity-based solutions by means of channel coding. The proposed solutions addresses the important network function of uplink channel decoding at the base station and leverages the algebraic structure of the received encoded data frames. Numerical results demonstrate the potential gains obtained with the proposed scheme as compared to the conventional diversity-based fault-tolerant scheme in terms of error probability.

CHAPTER 5

CONCLUSIONS

While most of the previous work on mobile cloud computing have focused on joint allocation of only uplink radio resources and cloud computational resources, the main contribution of the dissertation has been the analysis and design of a joint allocation scheme where radio and backhaul resources are optimized in both uplink and downlink as well as the optimization of computing resources at the cloud. The dissertation also introduced users scheduling scheme that guaranteed an energy-saving gains for offloading users.

In particular, driven by the rapid shift towards pervasive mobile applications, e.g., AR applications, the optimization design can also benefit from the shared collaborative feature of such applications so as to obtain the optimal allocation of the shared communication and computational resources. It is envisioned that mobile cloud computing, along with NFV are two enabling technologies for the deployment of such applications. Consequently, the dissertation considered the virtualization-based implementation of these technologies and addressed the reliability concern by proposing a coded NFV approach.

Overall, the dissertation shows that the proposed joint optimization schemes have brought considerable gains for mobile devices in terms of energy consumption and for both independent and collaborative computational tasks offloading to robust cloud infrastructures.

Open research challenges for future work encompass the inclusion of higher-layer aspects such as queuing in the definition of the latency for the design of mobile cloud computing systems, the training of a neural network, or another learning machine, to solve the joint resources allocation problem for a number of representative channels

so as to build a sufficiently dense look-up table, the design of NFV codes and the application of the principle of coded NFV to other network functions, such as routing, the understanding of the trade-off between sparsity and large minimum distance in the design of the generator matrix for coded NFV and finally the study of caching strategies for higher-priority components of AR applications.

BIBLIOGRAPHY

[1] A. R. Mishra, *Advanced Cellular Network Planning and Optimisation: 2G/2.5G/3G and Evolution to 4G*, 1st ed. Chichester, U.K.: Wiley Publishing, 2007.

[2] T. Verbelen, P. Simoens, F. D. Turck, and B. Dhoedt, "Leveraging cloudlets for immersive collaborative applications," *IEEE Pervasive Comput.*, vol. 12, no. 4, pp. 30–38, Dec. 2013.

[3] K. Kumar, J. Liu, Y.-H. Lu, and B. Bhargava, "A survey of computation offloading for mobile systems," *Mobile Netw. App.*, vol. 18, no. 1, pp. 129–140, Feb. 2013.

[4] H. T. Dinh, C. Lee, D. Niyato, and P. Wang, "A survey of mobile cloud computing: architecture, applications, and approaches," *Wireless commun. mobile comput.*, vol. 13, no. 18, pp. 1587–1611, Dec. 2013.

[5] Z. Sanaei, S. Abolfazli, A. Gani, and R. Buyya, "Heterogeneity in mobile cloud computing: taxonomy and open challenges," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 1, pp. 369–392, Feb. 2014.

[6] L. Lei, Z. Zhong, K. Zheng, J. Chen, and H. Meng, "Challenges on wireless heterogeneous networks for mobile cloud computing," *IEEE Wireless Commun.*, vol. 20, no. 3, pp. 34–44, Jun. 2013.

[7] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for VM-based cloudlets in mobile computing," *IEEE Pervasive Comput.*, vol. 8, no. 4, pp. 14–23, Oct. 2009.

[8] J. G. Andrews, S. Buzzi, W. Choi, S. V. Hanly, A. Lozano, A. C. K. Soong, and J. C. Zhang, "What will 5G be?" *IEEE J. Sel. Areas Commun.*, vol. 32, no. 6, pp. 1065–1082, Jun. 2014.

[9] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing–A key technology towards 5G," ETSI White Paper, no. 11, tech. rep., Sep. 2015.

[10] P. Rost, A. Banchs, I. Berberana, M. Breitbach, M. Doll, H. Droste, C. Mannweiler, M. A. Puente, K. Samdanis, and B. Sayadi, "Mobile network architecture evolution toward 5G," *IEEE Commun. Magazine*, vol. 54, no. 5, pp. 84–91, May 2016.

[11] F. Boccardi, R. W. Heath, A. Lozano, T. L. Marzetta, and P. Popovski, "Five disruptive technology directions for 5G," *IEEE Commun. Magazine*, vol. 52, no. 2, pp. 74–80, Feb. 2014.

[12] F. Liu, P. Shu, H. Jin, L. Ding, J. Yu, D. Niu, and B. Li, "Gearing resource-poor mobile devices with powerful clouds: architectures, challenges, and applications," *IEEE Wireless Commun.*, vol. 20, no. 3, pp. 14–22, Jun. 2013.

[13] B. Han, V. Gopalakrishnan, L. Ji, and S. Lee, "Network function virtualization: Challenges and opportunities for innovations," *IEEE Commun. Magazine*, vol. 53, no. 2, pp. 90–97, Feb. 2015.

[14] R. Mijumbi, J. Serrat, J. L. Gorricho, N. Bouten, F. D. Turck, and R. Boutaba, "Network function virtualization: State-of-the-art and research challenges," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 236–262, Firstquarter 2016.

[15] "ETSI GS NFV 003 V1.2.1: Network functions virtualisation (NFV); terminology for main concepts in NFV," ETSI Ind. Spec. Group (ISG) Netw. Functions Virtualisation (NFV), Sophia-Antipolis Cedex, France, Dec. 2014.

[16] "ETSI GS NFV-REL 003 V1.1.1: Network functions virtualisation (NFV); reliability; report on models and features for end-to-end reliability," ETSI Ind. Spec. Group (ISG) Netw. Functions Virtualisation (NFV), Sophia-Antipolis Cedex, France, Apr. 2016.

[17] "ETSI GS NFV-REL 002 V1.1.1: Network functions virtualisation (NFV); reliability; report on scalable architectures for reliability management," ETSI Ind. Spec. Group (ISG) Netw. Functions Virtualisation (NFV), Sophia-Antipolis Cedex, France, Sep. 2015.

[18] J. Liu, Z. Jiang, N. Kato, O. Akashi, and A. Takahara, "Reliability evaluation for NFV deployment of future mobile broadband networks," *IEEE Wireless Commun.*, vol. 23, no. 3, pp. 90–96, Jun. 2016.

[19] M. Maier, M. Chowdhury, B. P. Rimal, and D. P. Van, "The tactile internet: Vision, recent progress, and open challenges," *IEEE Commun. Magazine*, vol. 54, no. 5, pp. 138–145, May 2016.

[20] B. Han, P. Hui, V. S. A. Kumar, M. V. Marathe, J. Shao, and A. Srinivasan, "Mobile data offloading through opportunistic communications and social participation," *IEEE Trans. Mobile Comput.*, vol. 11, no. 5, pp. 821–834, May 2012.

[21] 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 *Proc. ACM 8th International Conf. Mobile Syst. App. Serv. (MobiSys10)*, San Francisco, CA, USA, pp. 49–62, Jun. 2010.

[22] 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 *Proc. IEEE International Conf. Comput. Commun. (INFOCOM)*, Orlando, FL, USA, pp. 945–953, Mar. 2012.

[23] M. Whaiduzzaman, A. Naveed, and A. Gani, "MobiCoRE: Mobile device based cloudlet resource enhancement for optimal task response," *IEEE Trans. Services Comput.*, vol. 9, no. 99, pp. 1–15, May 2016.

[24] N. Fernando, S. W. Loke, and W. Rahayu, "Mobile cloud computing: A survey," *Future Generation Computer Systems*, vol. 29, no. 1, pp. 84–106, Jan. 2013.

[25] J. E. Costa and J. J. Rodrigues, *Mobile cloud computing: Technologies, services, and applications*. IGI Global, 2013.

[26] J. Oueis, E. Calvanese-Strinati, A. D. Domenico, and S. Barbarossa, "On the impact of backhaul network on distributed cloud computing," in *Proc. IEEE Wireless Commun. Netw. Conf. Workshops (WCNCW)*, Istanbul, Turkey, pp. 12-17, Apr. 2014.

[27] J. M. Chung *et al.*, "Adaptive cloud offloading of augmented reality applications on smart devices for minimum energy consumption," *KSII Trans. Internet Inf. Syst.*, vol. 9, no. 8, pp. 3090–3102, Aug. 2015.

[28] A. Al-Shuwaili *et al.*, "Joint uplink/downlink optimization for backhaul-limited mobile cloud computing with user scheduling," *IEEE Trans. Signal Inf. Process. Net.*, to appear, 2017. [Online]: Available: http://ieeexplore.ieee.org/document/7850968/.

[29] S. Sardellitti, G. Scutari, and S. Barbarossa, "Joint optimization of radio and computational resources for multicell mobile-edge computing," *IEEE Trans. Signal Inf. Process. Net.*, vol. 1, no. 2, pp. 89–103, Jun. 2015.

[30] O. Munoz, A. Pascual-Iserte, and J. Vidal, "Optimization of radio and computational resources for energy efficiency in latency-constrained application offloading," *IEEE Trans. Veh. Technol.*, vol. 64, no. 10, pp. 4738–4755, Oct. 2015.

[31] M. Molina, O. Muoz, A. Pascual-Iserte, and J. Vidal, "Joint scheduling of communication and computation resources in multiuser wireless application offloading," in *Proc. IEEE International Symposium Pers. Indoor Mobile Radio Commun. (PIMRC)*, Washington, DC, USA, pp. 1093-1098, Sep. 2014.

[32] S. Bohez *et al.*, "Mobile, collaborative augmented reality using cloudlets," in *Proc. Mobilware*, Bologna, Italy, pp. 45-54, Nov. 2013.

[33] "Ericsson white paper Uen Rev B: The real-time cloud: Combining cloud, NFV and service provider SDN," Ericsson AB, Stockholm, Sweden, Feb. 2014.

[34] D. Huang, P. Wang, and D. Niyato, "A dynamic offloading algorithm for mobile computing," *IEEE Trans. Wireless Commun.*, vol. 11, no. 6, pp. 1991–1995, Jun. 2012.

[35] A. P. Miettinen and J. K. Nurminen, "Energy efficiency of mobile clients in cloud computing," in *Proc. 2nd USENIX Conf. Hot Topics Cloud Comput.*, Boston, MA, USA, pp. 4-11, Jun. 2010.

[36] K. Kumar and Y. H. Lu, "Cloud computing for mobile users: Can offloading computation save energy?" *Computer*, vol. 43, no. 4, pp. 51–56, Apr. 2010.

[37] Y.-H. Kao, B. Krishnamachari, M.-R. Ra, and F. Bai, "Hermes: Latency optimal task assignment for resource-constrained mobile computing," in *Proc. IEEE International Conf. Comput. Commun. (INFOCOM)*, Kowloon, Hong Kong, pp. 1894-1902, Apr. 2015.

[38] M. Kamoun, W. Labidi, and M. Sarkiss, "Joint resource allocation and offloading strategies in cloud enabled cellular networks," in *Proc. IEEE International Conf. Commun. (ICC)*, London, UK, pp. 5529-5534, Jun. 2015.

[39] S. Khalili and O. Simeone, "Inter-layer per-mobile optimization of cloud mobile computing: a message-passing approach," *Transactions on Emerging Telecommunications Technologies*, Feb. 2016.

[40] X. Xiang, C. Lin, and X. Chen, "Energy-efficient link selection and transmission scheduling in mobile cloud computing," *IEEE Wireless Commun. Letters*, vol. 3, no. 2, pp. 153–156, Apr. 2014.

[41] Y. Yu, J. Zhang, and K. Ben Letaief, "Joint subcarrier and CPU time allocation for mobile edge computing," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Washington, DC, USA, pp. 1–6, Dec. 2016.

[42] W. Zhang, Y. Wen, K. Guan, D. Kilper, H. Luo, and D. O. Wu, "Energy-optimal mobile cloud computing under stochastic wireless channel," *IEEE Trans. Wireless Commun.*, vol. 12, no. 9, pp. 4569–4581, Sep. 2013.

[43] Y. Wang, M. Sheng, X. Wang, L. Wang, W. Han, Y. Zhang, and Y. Shi, "Energy-optimal partial computation offloading using dynamic voltage scaling," in *Proc. IEEE International Conf. Commun. Workshop (ICCW)*, London, UK, pp. 2695–2700, Sep. 2015.

[44] C. You, K. Huang, and H. Chae, "Energy efficient mobile cloud computing powered by wireless energy transfer," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 5, pp. 1757–1771, May 2016.

[45] X. Chen, "Decentralized computation offloading game for mobile cloud computing," *IEEE Trans. Parallel Distributed Sys.*, vol. 26, no. 4, pp. 974–983, Apr. 2015.

[46] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Trans. Netw.*, vol. 9, no. 99, pp. 1–14, Sep. 2015.

[47] R. Kaewpuang, D. Niyato, P. Wang, and E. Hossain, "A framework for cooperative resource management in mobile cloud computing," *IEEE J. Sel. Areas Commun.*, vol. 31, no. 12, pp. 2685–2700, December 2013.

[48] S. Sardellitti, G. Scutari, and S. Barbarossa, "Joint optimization of radio and computational resources for multicell mobile-edge computing," *IEEE Trans. Signal Inf. Process. Over Netw.*, vol. 1, no. 2, pp. 89–103, Jun. 2015.

[49] C. You, K. Huang, H. Chae, and B. H. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1397–1411, Mar. 2017.

[50] K. Zhang, Y. Mao, S. Leng, Q. Zhao, L. Li, X. Peng, L. Pan, S. Maharjan, and Y. Zhang, "Energy-efficient offloading for mobile edge computing in 5G heterogeneous networks," *IEEE Access*, vol. 4, pp. 5896–5907, Aug. 2016.

[51] J. Guo, Z. Song, and Y. Cui, "Energy-efficient resource allocation for multi-user mobile edge computing," in *Proc. IEEE International Conf. Commun. (ICC)*, to appear, 2017. [Online]. Available: arXiv:1611.01786. 2017.

[52] T. Zhao, S. Zhou, X. Guo, Y. Zhao, and Z. Niu, "A cooperative scheduling scheme of local cloud and internet cloud for delay-aware mobile cloud computing," in *Proc. IEEE Globecom Workshops*, San Diego, CA, USA, pp. 1–6, Dec. 2015.

[53] E. Bastug, M. Bennis, M. Médard, and M. Debbah, "Towards interconnected virtual reality: Opportunities, challenges and enablers," *IEEE Commun. Magazine*, to appear, Nov. 2016. [Online]. Available: ArXiv:1611.05356.

[54] G. Papagiannakis, G. Singh, and N. Magnenat-Thalmann, "A survey of mobile and wireless technologies for augmented reality systems," *Computer Animation and Virtual Worlds*, vol. 19, no. 1, pp. 3–22, 2008.

[55] M. Billinghurst, A. Clark, and G. Lee, "A survey of augmented reality," *Foundations and Trends® Human–Computer Interaction*, vol. 8, no. 2-3, pp. 73–272, 2015.

[56] G. Gerstweiler, E. Vonach, and H. Kaufmann, "Hymotrack: A mobile ar navigation system for complex indoor environments," *Sensors*, vol. 16, no. 1, p. 17, 2015.

[57] M. Tonnis, C. Sandor, C. Lange, and H. Bubb, "Experimental evaluation of an augmented reality visualization for directing a car driver's attention," in *Proc. ACM International Symposium on Mixed and Augmented Reality*, Washington, DC, USA, pp. 56-59, Oct. 2005.

[58] P. J. Bartie and W. A. Mackaness, "Development of a speech-based augmented reality system to support exploration of cityscape," *Transactions in GIS*, vol. 10, no. 1, pp. 63–86, 2006.

[59] V. Vlahakis, J. Karigiannis, M. Tsotros, M. Gounaris, L. Almeida, D. Stricker, T. Gleue, I. T. Christou, R. Carlucci, and N. Ioannidis, "Archeoguide: First results of an augmented reality, mobile computing system in cultural heritage sites," in *Proc. Conference on Virtual Reality, Archeology, and Cultural Heritage*, Glyfada, Greece, pp. 131-140, Nov. 2001.

[60] S. Boring, S. Gehring, A. Wiethoff, A. M. Blöckner, J. Schöning, and A. Butz, "Multi-user interaction on media facades through live video on mobile devices," in *Proc. SIGCHI Conf. Human Factors Comput. Syst.*, pp. 2721–2724, May 2011.

[61] W. Piekarski and B. Thomas, "Arquake: the outdoor augmented reality gaming system," *Communications of the ACM*, vol. 45, no. 1, pp. 36–38, 2002.

[62] C. Matysczok, R. Radkowski, and J. Berssenbruegge, "Ar-bowling: immersive and realistic game play in real environments using augmented reality," in *Proc. ACM SIGCHI International Conf. Advances Comput. Entertainment Tech.*, Valencia, Spain, pp. 269-276, Jun. 2005.

[63] M. López-Nores, J. J. Pazos-Arias, Y. Blanco-Fernández, M. Ramos-Cabrer, and A. Gil-Solla, "Delivering personalised m-commerce through cloud-based augmented reality on billboards," in *Proc. IEEE International Conf. Consumer Electronics (ICCE)*, Las Vegas, NV, USA, pp. 162–163, Jan. 2015.

[64] K. Ha, Z. Chen, W. Hu, W. Richter, P. Pillai, and M. Satyanarayanan, "Towards wearable cognitive assistance," in *Proc. International Conf. Mobile Syst. App. Serv.*, New Hampshire, USA, pp. 68–81, Jun. 2014.

[65] J. Bang, D. Lee, H. Lee, and W. Son, "Network assistance to localization and mapping for outdoor augmented reality in cellular network," in *International Conf. Platform Tech. Serv. (PlatCon)*, Jeju, Korea, pp. 1-4, Feb. 2016.

[66] D. Van Krevelen and R. Poelman, "A survey of augmented reality technologies, applications and limitations," *International J. of Virtual Reality*, vol. 9, no. 2, pp. 1–20, Jan. 2010.

[67] P. Simoens, T. Verbelen, and B. Dhoedt, "Mobile ar in the outdoors: watch the clouds," in *How to industrialize Wearable AR?* Ghent University, Department of Information Technology, 2012.

[68] B.-R. Huang, C. H. Lin, and C.-H. Lee, "Mobile augmented reality based on cloud computing," in *Proc. International Conference on Anti-Counterfeiting, Security and Identification (ASID)*, Taipei, Taiwan, pp. 1-5, Oct. 2012.

[69] M. Chen, C. Ling, and W. Zhang, "Analysis of augmented reality application based on cloud computing," in *Proc. International Congress Image Sig. Process. (CISP)*, Shanghai, China, pp. 569–572, Oct. 2011.

[70] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for VM-based cloudlets in mobile computing," *IEEE Pervasive Comput.*, vol. 8, no. 4, pp. 14–23, Dec. 2009.

[71] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *Proc. ACM International Symposium on Mixed and Augmented Reality*, Nara, Japan, pp. 225-234, Nov. 2007.

[72] T. Verbelen, P. Simoens, F. De Turck, and B. Dhoedt, "Cloudlets: Bringing the cloud to the mobile user," in *Proc. ACM Workshop Mobile Cloud Comput. Serv.*, Lake District, UK, pp. 29–36, Jun. 2012.

[73] L. Yu, S.-K. Ong, and A. Y. Nee, "A tracking solution for mobile augmented reality based on sensor-aided marker-less tracking and panoramic mapping," *Multimedia Tools and Applications*, vol. 75, no. 6, pp. 3199–3220, May 2016.

[74] C.-C. Wu, L.-P. Tung, C.-Y. Lin, B.-S. P. Lin, and Y.-C. Tseng, "On local cache management strategies for mobile augmented reality," in *Proc. IEEE International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, Boston, MA, USA, pp. 1–3, Jun. 2014.

[75] "CloudNFV," [Online]. Available: http://www.cloudnfv.com (accessed on: Jan. 2017).

[76] "Alcatel-CloudBand," [Online]. Available: https://networks.nokia.com/solutions/cloudband (accessed on: Jan. 2017).

[77] R. Cohen, L. Lewin-Eytan, J. S. Naor, and D. Raz, "Near optimal placement of virtual network functions," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Kowloon, Hong Kong, pp. 1346-1354, Apr. 2015.

[78] G. Scutari, F. Facchinei, and L. Lampariello, "Parallel and distributed methods for constrained nonconvex optimization-part I: Theory," *IEEE Trans. Signal Process.*, to appear, 2016. [Online]: Available: arXiv:1410.4754.

[79] G. Scutari, F. Facchinei, L. Lampariello, S. Sardellitti, and P. Song, "Parallel and distributed methods for nonconvex optimization–part II: Applications in communications and machine learning," *IEEE Trans. Signal Process.*, to appear, 2016. [Online]: Available: arXiv:1601.04059.

[80] Y. Shi, J. Cheng, J. Zhang, B. Bai, W. Chen, and K. B. Letaief, "Smoothed Lp-minimization for green cloud-RAN with user admission control," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 4, Apr. 2016.

[81] D. Gesbert, S. Hanly, H. Huang, S. S. Shitz, O. Simeone, and W. Yu, "Multi-cell MIMO cooperative networks: A new look at interference," *IEEE J. Sel. Areas Commun.*, vol. 28, no. 9, pp. 1380–1408, Dec. 2010.

[82] O. Simeone, A. Maeder, M. Peng, O. Sahin, and W. Yu, "Cloud radio access network: Virtualizing wireless access for dense heterogeneous systems," *J. Commun. Netw.*, vol. 18, no. 2, pp. 135–149, Apr. 2016.

[83] D. Tse and P. Viswanath, *Fundamentals of wireless communication*. New York, NY, USA: Cambridge University Press, 2005.

[84] A. Lapidoth, "Nearest neighbor decoding for additive non-Gaussian noise channels," *IEEE Trans. Information Theory*, vol. 42, no. 5, pp. 1520–1529, Sep. 1996.

[85] Y. Nesterov and A. Nemirovskii, *Interior-point polynomial algorithms in convex programming*. SIAM, 1994, vol. 13.

[86] T. Lipp and S. Boyd, "Variations and extension of the convex–concave procedure," *Optimization and Engineering*, pp. 1–25, Nov. 2015.

[87] S. Boyd and L. Vandenberghe, *Convex optimization*. New York, NY, USA: Cambridge University Press, 2004.

[88] H. Sun, X. Chen, Q. Shi, M. Hong, and X. Fu, "Learning to optimize: Training deep neural networks for wireless resource management," submitted, Sep. 2016.

[89] T. D. Burd and R. W. Brodersen, "Processor design for portable systems," *J. VLSI Signal Process. Syst.*, vol. 13, no. 2-3, pp. 203–221, Aug. 1996.

[90] E. Lagerspetz and S. Tarkoma, "Mobile search and the cloud: The benefits of offloading," in *Proc. IEEE International Conf. Pervasive Comput. Commun. (PerCom)*, St. Louis, MO, USA, pp. 117-122, Mar. 2011.

[91] A. LLC, "Amazon elastic compute cloud (EC2) website." [Online]. Available: https://aws.amazon.com/ec2/ (accessed on: Oct. 2016)

[92] N. Balasubramanian, A. Balasubramanian, and A. Venkataramani, "Energy consumption in mobile phones: A measurement study and implications for network applications," in *Proc. ACM SIGCOMM Internet Measurement Conference*, Chicago, IL, pp. 280–293, Nov. 2009.

[93] L. Cannelli, F. Facchinei, V. Kungurtsev, and G. Scutari, "Asynchronous parallel algorithms for nonconvex big-data optimization. part II: Complexity and numerical results," *Mathematical Programming*, submitted, Jan. 2017.

[94] 3GPP TR 36.814, "Technical specification group radio access network; Further advancements for E-UTRA, physical layer aspects," Mar. 2010.

[95] F. Renna, J. Doyle, V. Giotsas, and Y. Andreopoulos, "Query processing for the internet-of-things: Coupling of device energy consumption and cloud infrastructure billing," in *Proc. IEEE International Conf. Internet-of-Things Design and Implementation (IoTDI)*, Berlin, Germany, pp. 83-94, Apr. 2016.

[96] M. Tawarmalani and N. V. Sahinidis, "A polyhedral branch-and-cut approach to global optimization," *Mathematical Programming*, vol. 103, pp. 225–249, 2005.

[97] P. Rost, S. Talarico, and M. C. Valenti, "The complexity-rate tradeoff of centralized radio access networks," *IEEE Trans. Wireless Commun.*, vol. 14, no. 11, pp. 6164–6176, Nov. 2015.

[98] A. Gatherer, "Revisiting cloud RAN from a computer architecture point of view," *IEEE ComSoc Technology News (CTN)*, Jul. 2016.

[99] S. H. Lim, C. Feng, A. Pastore, B. Nazer, and M. Gastpar, "A joint typicality approach to algebraic network information theory," *IEEE Trans. Information Theory*, submitted, jun. 2016. [Online]: Available: ArXiv:1606.09548.