Latency and Reliability Aware Edge Computation Offloading in 5G Networks

Elie El Haber

A Thesis
In
The Concordia Institute
For
Information and Systems Engineering

Presented in Partial Fulfillment of the Requirements
For the Degree of
Doctor of Philosophy (Information and Systems Engineering)
Concordia University
Montréal, Québec, Canada

March 2022

Concordia University
School of Graduate Studies

This is to certify that the thesis prepared

By: **Elie El Haber**

Entitled: **Latency and Reliability Aware Edge Computation Offloading in 5G Networks**

and submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy (Information and Systems Engineering)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining commitee:

Dr. Gösta Grahne _____ Chair

Dr. Melike Erol-Kantarci _____ External Examiner

Dr. Dongyu Qiu _____ External to Program

Dr. Jamal Bentahar _____ Examiner

Dr. Roch Glitho _____ Examiner

Dr. Chadi Assi _____ Supervisor

Approved by _____
Dr. Abdessamad Ben Hamza, Chair

_____ April, 28, 2022 _____
Dr. Mourad Debbabi, Dean
Gina Cody School of Engineering and Computer Science

ABSTRACT

Latency and Reliability Aware Edge Computation Offloading in 5G Networks

Elie El Haber, Ph.D.
Concordia University, 2022

Empowered by recent technological advances and driven by the ever-growing population density and needs, the conception of 5G has opened up the expectations of what mobile networks are capable of to heights never seen before, promising to unleash a myriad of new business practices and paving the way for a surging number of user equipments to carry out novel service operations. The advent of 5G and networks beyond will hence enable the vision of Internet of Things (IoT) and smart city with its ubiquitous and heterogeneous use cases belonging to various verticals operating on a common underlying infrastructure, such as smart healthcare, autonomous driving, and smart manufacturing, while imposing extreme unprecedented Quality of Service (QoS) requirements in terms of latency and reliability among others. Due to the necessity of those modern services such as traffic coordination, industrial processes, and mission critical applications to perform heavy workload computations on the collected input, IoT devices such as cameras, sensors, and Cyber-Physical Systems (CPSs), which have limited energy and processing capabilities are put under an unusual strain to seamlessly carry out the required service computations. While offloading the devices' workload to cloud data centers with Mobile Cloud Computing (MCC) remains a possible alternative which also brings about a high computation reliability, the latency incurred from this approach would prevent from satisfying the services' QoS requirements, in addition to elevating the load in the network core and backhaul, rendering MCC an inadequate solution for handling the 5G services'

required computations. In light of this development, Multi-access Edge Computing (MEC) has been proposed as a cutting edge technology for realizing a low-latency computation offloading by bringing the cloud to the vicinity of end-user devices as processing units co-located within base stations leveraging the virtualization technique. Although it promises to satisfy the stringent latency service requirements, realizing the edge-cloud solution is coupled with various challenges, such as the edge servers' restricted capacity, their reduced processing reliability, the IoT devices' limited offloading energy, the wireless offloading channels' often weak quality, the difficulty to adapt to dynamic environment changes and to under-served networks, and the Network Operators (NOs)' cost-efficiency concerns. In light of those conditions, the NOs are consequently looking to devise efficient innovative computation offloading schemes through leveraging novel technologies and architectures for guaranteeing the seamless provisioning of modern services with their stringent latency and reliability QoS requirements, while ensuring the effective utilization of the various network and devices' available resources. Leveraging a hierarchical arrangement of MEC with second-tier edge servers co-located within aggregation nodes and macro-cells can expand the edge network's capability, while utilizing Unmanned Aerial Vehicles (UAVs) to provision the MEC service via UAV-mounted cloudlets can increase the availability, flexibility, and scalability of the computation offloading solution. Moreover, aiding the MEC system with UAVs and Intelligent Reflecting Surfaces (IRSs) can improve the computation offloading performance by enhancing the wireless communication channels' conditions. By effectively leveraging those novel technologies while tackling their challenges, the edge-cloud paradigm will bring about a tremendous advancement to 5G networks and beyond, opening the door to enabling all sorts of modern and futuristic services.

In this dissertation, we attempt to address key challenges linked to realizing the vision of a low-latency and high-reliability edge computation offloading in modern networks while exploring the aid of multiple 5G network technologies. Towards that end, we provide novel contributions related to the allocation of network and devices'

resources as well as the optimization of other offloading parameters, and thereby efficiently utilizing the underlying infrastructure such as to enable energy and cost-efficient computation offloading schemes, by leveraging several customized solutions and optimization techniques. In particular, we first tackle the computation offloading problem considering a multi-tier MEC with a deployed second-tier edge-cloud, where we optimize its use through proposed low-complexity algorithms, such as to achieve an energy and cost-efficient solution that guarantees the services' latency requirements. Due to the significant advantage of operating MEC in heterogeneous networks, we extend the scenario to a network of small-cells with the second-tier edge server being co-located within the macro-cell which can be reached through a wireless backhaul, where we optimize the macro-cell server use along with the other offloading parameters through a proposed customized algorithm based on the Successive Convex Approximation (SCA) technique. Then, given the UAVs' considerable ability in expanding the capabilities of cellular networks and MEC systems, we study the latency and reliability aware optimized positioning and use of UAV-mounted cloudlets for computation offloading through two planning and operational problems while considering tasks redundancy, and propose customized solutions for solving those problems. Finally, given the IRSs' ability to also enhance the channel conditions through the tuning of their passive reflecting elements, we extend the latency and reliability aware study to a scenario of an IRS-aided MEC system considering both a single-user and multi-user OFDMA cases, where we explore the optimized IRSs' use in order to reveal their role in reducing the UEs' offloading consumption energy and saving the network resources, through proposed customized solutions based on the SCA approach and the SDR technique.

Acknowledgments

The dedication, the hard work, the perseverance to make the dream come true and the vision into a reality no matter what, are embodied with my mentor and supervisor, Dr. Chadi Assi. I am thankful Dr. Assi for all your support, your understanding, your patience, your guidance, and the priceless lessons I learned from you throughout my Ph.D. years. They are going to stay with me throughout my life, which I will aim to make it as fruitful and as rewarding as yours. Thank you for everything.

A true friend is someone who is always there for you without asking for anything in return. Dr. Hyame Alameddine, thank you for all your unconditional support and guidance since my first day in Canada and throughout my Ph.D. years; You have helped me keep my eyes on the target, and keep going during my hard times. What an amazing opportunity it is with you to live the true meaning of friendship, while also being able to collaborate together on some of my thesis projects.

I am grateful for Dr. Tri Minh Nguyen for his help and guidance on the application of many optimization techniques, and for Mohamed Elhattab, Dr. Sanaa Sharafeddine, Dr. Wessam Ajib, and Dr. Dariush Ebrahimi for all the enlightening discussions, comments and valuable collaborations on the projects that I have completed throughout my Ph.D.

I would like to thank my committee members Dr. Jamal Bentahar, Dr. Roch Glitho, and Dr. Dongyu Qiu for their time, their valuable feedback and constructive comments. I would like to extend my appreciation to Dr. Melike Erol-Kantarci for accepting to serve as my external examiner.

Many thanks to Dr. Kim Khoa Nguyen, Dr. Brigitte Jaumard, and the Ciena research team whom I had a great pleasure to meet and work with. Thank you for all your kindness, support and for enriching my internship time at Ciena.

My deepest appreciation goes to all my friends and colleagues whom I was able to share special moments with, and engrave great memories that will stay with me forever. Dr. Ibrahim Sorkhoh, Mohamed Amine Arfaoui, Mohamed Elhattab, Ali Muhammad, Dr. Mohammad Ekramul Kabir, Joseph Antoun, Shirin Rezasoltani, and all the others who know themselves, thanks for all the advices, the laughs, and the good times. I am blessed to have you. Also, to my friends at the Lebanese university and Nymgo S.A. in Lebanon whom I shared great moments with, thanks for your support and motivation that have allowed me to reach this point.

I dedicate this thesis to my beloved mother and father, Afaf El Haber and Issam El Haber; You worked hard and did everything possible to raise and educate me with my siblings, despite all the odds. You are truly exceptional parents; I never felt anything but care and support in everything I did, thanks to the endless love that you have passed to me. Mother, thank you for your endless care and persistence in providing and doing everything possible for us and for your infinite support; Nothing would have been possible without it. Father, thank you for always believing in me and supporting me in everything I did, for all your guidance and persistent motivation to go to the next level and follow my dreams, and for always being there for me. My parents, I hope that this achievement pays but little of all that you did to me. Thank you for everything. I love you.

To my brother Michael El Haber and my sister Marie-Louise El Haber, the best siblings that I could have, my aunts Nawal Chalhoub and Marcelle Daher, and to my entire family, your love, warmth, and support have been the fuel that kept me going, nothing is more valuable than having you in my life. Thank you for being by my side and for handling me with my good and bad all these years.

Finally, I would like to thank Concordia University, NSERC, and Mitacs for their financial support which made my Ph.D. work possible.

To all the challenges, difficulties, and hardest moments,
thank you for bringing out the best.

Contents

List of Figures

List of Tables

Abbreviations

AoI	Age of information
AP	Access Point
AR	Augmented Reality
BCD	Block Coordinate Descent
BnB	Branch-and-Bound
BS	Base Station
CN	Challenged Network
CPS	Cyber-Physical System
CPU	Central Processing Unit
CSI	Channel State Information
D2D	Device-to-Device
dBm	Decibel Milliwatts
DC	Difference of Convex
DRL	Deep Reinforcement Learning
eMBB	enhanced Mobile Broadband
ETSI	European Telecommunications Standards Institute
FSPL	Free-Space Path Loss

GHz	Gigahertz
GtA	Ground-to-Air
HAP	High-altitude Platform
IaaS	Infrastructure as a Service
InP	Infrastructure Provider
IoT	Internet of Things
IRS	Intelligent reflective Surface
ITU	International Telecommunication Union
Kb	kilobits
LAP	Low-altitude Platform
LB	Lower Bound
LoS	Line of Sight
LTE	Long-Term Evolution
Mbps	Megabits per Second
MC	Macro-cell
MCC	Mobile Cloud Computing
MDP	Markov Decision Process
MEC	Multi-access Edge Computing
MHz	Megahertz
MI	Mixed Integer
MIMO	Multiple Input Multiple Output
Mj	Millijoules
mMTC	massive Machine Type Communication
ms	Milliseconds

NFV	Network Function Virtualization
NLP	Non-Linear Program
NO	Network Operator
NOMA	Non-Orthogonal Multiple Access
NS	Network Slicing
OFDMA	Orthogonal Frequency Division Multiple Access
PaaS	Platform as a Service
PSD	Positive Semi-Define
PSD	Positive Semi-define
QoE	Quality of Experience
QoS	Quality of Service
RAN	Radio Access Network
RAT	Radio Access Technology
RB	Resource Block
RHS	Right-hand Side
SC	Small-cell
SCA	Successive Convex Approximation
SDN	Software-Defined Networking
SDP	Semi-definite Program
SDR	Semi-definite Relaxation
SLA	Service Level Agreement
SOCP	Second-order Cone Program
SRE	Smart Radio Environment
TDMA	Time Division Multiple Access

UAV	Unmanned Aerial Vehicle
UE	User Equipment
URLLC	Ultra-reliable Low-latency Communication
V2V	Vehicle-to-Vehicle
VM	Virtual Machine
VNF	Virtual Network Function
VR	Virtual Reality

Chapter 1

Introduction

1.1 The Advent of 5G and Networks Beyond

Since the advent of mobile communication with the first generation in the early 80s, technological advancement has been on a constant rise, bringing quality enhancements and service improvements to cellular networks within each generation. 2G in the 90s brought enhancements to the quality of voice communication. 3G introduced basic data transmissions and allowed users to have primitive Internet access. With 4G, data packets communication has boomed, allowing users to have seamless Internet access with high data rates, enabling the streaming of HD videos, and opening the door to countless Internet-based businesses which flourished the economy. Evidently, this trend characterized by an introduction of new service requirements in one generation and their full solidification in the next, is not coming to a stop anytime soon.

Figure 1.1: The vertical markets of IoT and the horizontal integration between them [1].

In recent years, driven by the ever-growing population density around the cities and the consumer's need to have access to fast, safe, and smart modern services [9], a large widespread of billions of Internet-connected devices has been underway, causing a massive increase in the service demands and data load, culminating into the IoT paradigm [10]. In the age of IoT, various processes will be modernized and automated, and the data will be mostly produced, consumed, and communicated by things rather than people, where things such as sensor-enabled devices, wearables, home utilities, will become smart by exploiting their underlying technologies such as ubiquitous computing, embedded devices, and communication technologies [1, 11]. As Fig. 1.1 shows, IoT will realize the vision of smart city and cutting-edge verticals

such as smart healthcare, autonomous driving, and smart manufacturing with their modern services and use cases [1]. For instance, smart healthcare will enable the operation of real-time remote surgeries. On-body sensors will allow for the automatic detection and timely treatment of health issues. Camera surveillance systems will operate through the instantaneous capture and processing of images and videos. Street cameras and sensors will enable performing seamless traffic coordination. Smart parking will allow for the automatic detection of available parking spots. CPSs will be able to spontaneously carry out factory operations with real-time sensing and objects identification. Enabling those services will impose extremely challenging novel requirements with unprecedented stringency and heterogeneity, putting a huge strain on the established networks with their provisioned resources.

To realize this full-fledged smart and connected world, the next big evolution in cellular networks is the beyond 2020 mobile communication systems, i.e. the 5G and 6G networks [12], which is envisioned to contain the large boost in the data traffic and to cater for the unique service requirements that established networks are not prepared for.

1.2 The Introduction of Novel 5G Technologies

To enable the vision of 5G and 6G networks for supporting the massive load of the devices' requests and the unique requirements of the novel services, the research community and industry in recent years have put huge efforts into proposing new network architectures and technologies that are still being researched to overcome

their challenges and to realize their full vision.

Extending the radio spectrum to include Millimeter waves was proposed to expand the available wireless spectrum for accommodating more traffic. NOMA, massive MIMO, and small-cells were introduced in cellular networks to increase the spectrum efficiency, support more users load, and improve communication rates in the access network. UAVs were introduced to assist in extending the cellular networks' capability and coverage, and enhancing the wireless channel conditions, thanks to their numerous advantages, such as flexibility, low-cost, and LoS communication [7, 13]. IRSs were introduced into cellular networks owing also to their ability to enhance the wireless channel conditions and the communication rates in the access network through the use of their passive reflecting elements.

1.3 Characterization of the Modern 5G Services

The coexistence of human-centric and machine-type services in the age of IoT will make communication environments more diverse and complicated. While services in LTE networks mostly required high broadband in order to provide seamless QoE and high data rates for the end-user devices, 5G and 6G networks however must support unique service characteristics with extremely challenging and heterogeneous requirements, such as ultra-low end-to-end communication latency, ultra-reliability, and high availability, required by modern services and real-time applications [14, 15].

IoT services bring special technical requirements that were not present in traditional human-centric services, which constitutes a challenge to enable their support in

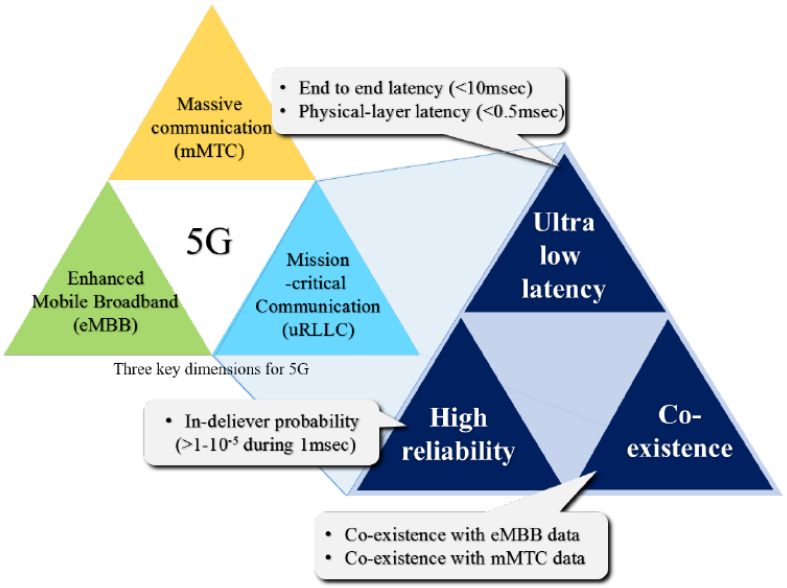


Figure 1.2: 5G service categories and key requirements for URLLC [2].

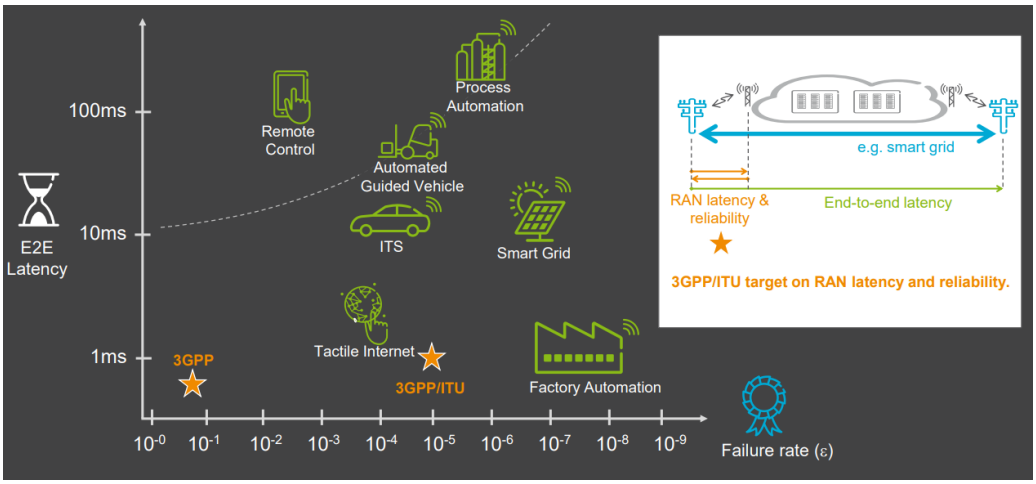


Figure 1.3: Latency and reliability service requirements in 5G networks [3].

5G networks and beyond [1]. Availability for everyone at different places simultaneously. Reliability of the communication network that must be resilient to failures for

realizing reliable information distribution. Reliability in both the software and hardware throughout all layers. Scalability that is designed from the ground up to enable extensible services and operations in the presence of diverse hardware platforms and communications protocols. Interoperability for handling many heterogeneous things that belong to different platforms.

Fig. 1.2 presents the three basic service categories in the 5G and 6G era as defined by the ITU, which impose requirements of an unprecedented level of heterogeneity and strictness [2]. eMBB services such as VR/AR, 3D video streaming, and high resolution gaming, which possess high requirements for bandwidth and data rates. mMTC services such as sensor networks, which are characterized by a high density of battery-limited devices that perform operations such as logging, metering, and measuring with the inability of transmitting the data over a long distance. URLLC services such as the ones presented in Fig. 1.3 like traffic coordination, autonomous driving, industrial processes, remote surgery, the tactile Internet, and other mission-critical applications, which need to carry real-time sensing and control tasks with an ultra-high reliability (up to 99.99%) and an ultra-low latency (up to 1 ms) requirements, in order for their mission to proceed safely, which are conflicting requirements since increasing reliability generally incurs a higher latency [16, 17, 18].

1.4 Latency and Reliability Aware Tasks Computation

In order to provide the intended functionality, most of the modern URLLC-type services such as the ones presented in Fig. 1.3, will require performing computational-intensive tasks on the collected data [19, 14] with stringent latency and reliability requirements. Consider for instance a scenario where a street sensor or a camera needs to perform a real-time processing on the captured data or video in order to detect and warn pedestrians passing on the street. However, those devices often have limited energy and capabilities, preventing them from affording local tasks computation. Thus, there is a need for a solution to enable latency and reliability aware tasks computation for the low-energy user devices.

1.5 The Traditional Mobile Cloud Computing Approach

As illustrated in Fig. 1.4, MCC has traditionally been the solution for enabling computation-intensive applications [4] such as mobile commerce, mobile learning, mobile healthcare, and mobile gaming, through the cloud services provided by Microsoft Azure and Amazon Web Service. MCC constituted an innovative solution for provisioning computation and storage services for energy-limited UEs, whereby the tasks are offloaded to cloud data centers through the cellular network and Internet backbone, leveraging the large-scale and massive available amounts of computational

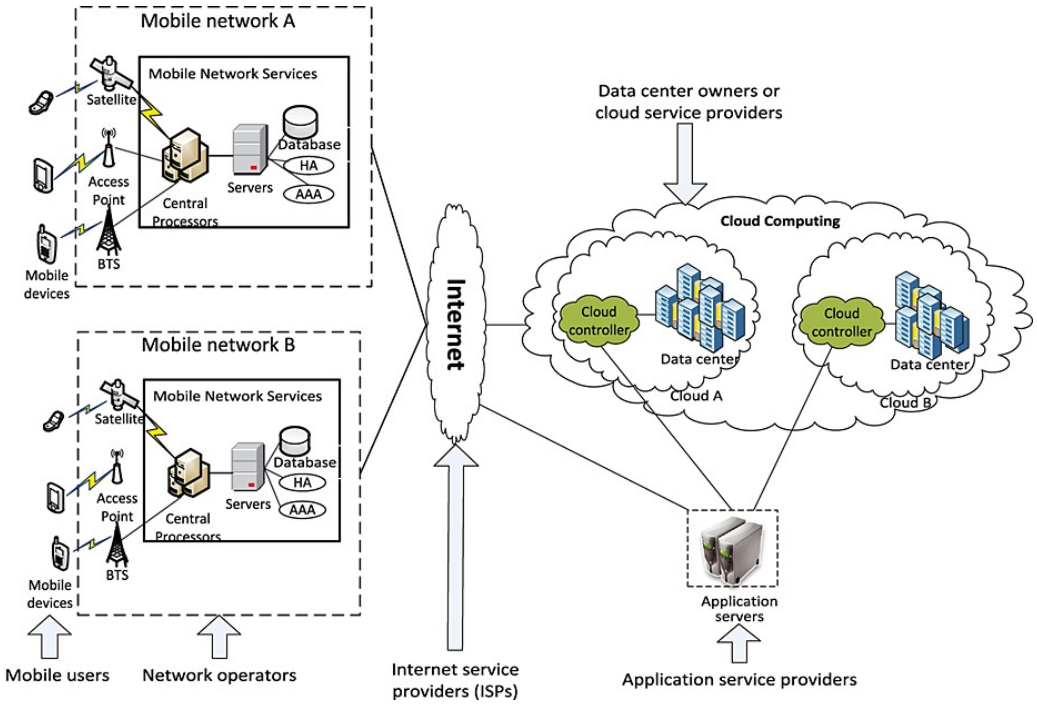


Figure 1.4: Mobile cloud computing architecture [4].

resources dedicated for tasks processing and data storage [4, 20]. MCC can extend the UEs' battery lifetime, where up to 45% of energy consumption can be saved, while improving their processing power and data storage capacity [4]. Moreover, due to the optimized cloud computing environment with the large-scale resources availability and the applied multiplexing techniques, MCC provides high levels of computation reliability, scalability, and flexibility.

However, MCC has several shortcomings, rendering it unable to be applied for modern latency-sensitive 5G services. First, MCC incurs a high end-to-end communication latency that would bypass the required threshold, due to the round-trip transmission having to traverse the Internet backbone. Second, MCC incurs a high

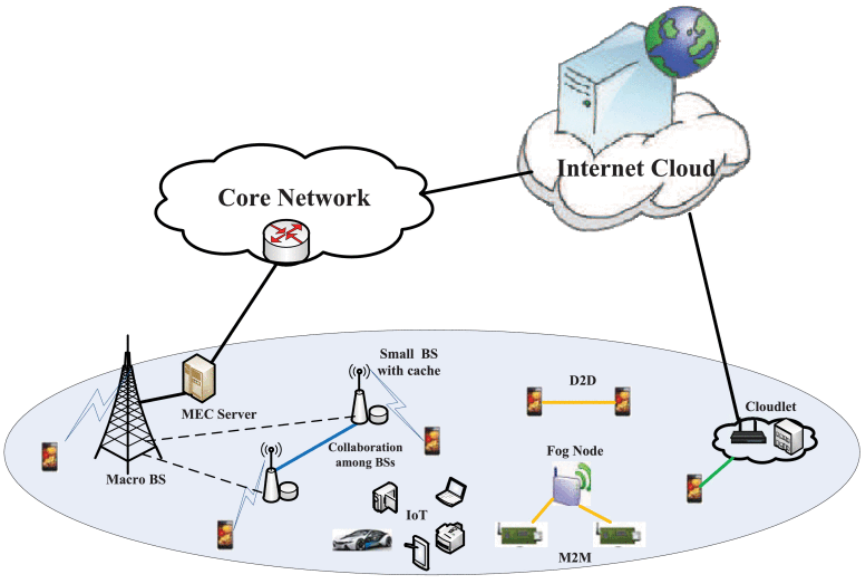


Figure 1.5: Architecture of Mobile edge-cloud [5].

data load in the network core and backhaul that consumes the communication bandwidth, which adds to the already existing load in the access network, decreasing the transmission rates and increasing the end-to-end latency even more. Therefore, MCC is rendered incapable of provisioning the tasks' computation required for operating the modern 5G services, necessitating a serious rethinking of the network architecture for enabling computation-intensive applications on the low-energy user devices, in a way that the envisioned latency and reliability sensitive services in 5G networks and beyond can be realized for enabling a smart and connected world.

1.6 The Emergence of The Edge-cloud Paradigm

Seeing the limitations of applying MCC to provide tasks computation for modern 5G services with their stringent latency and reliability requirements, the concept of edge-cloud computing was introduced. The notion of cloudlets first appeared in [21], and MEC was later defined by the ETSI as cloud-like computation and storage capabilities that are brought as small computation units (i.e. cloudlets) to the network edge in the vicinity of the end-user devices co-located within BSs and APs [19, 22]. The definition of MEC was later updated to multi-access edge computing, in order to allow for multiple RATs to be adopted apart from the cellular radio, such as Wi-Fi, Bluetooth, and WiMAX [23]. As it can be seen in Fig. 1.5, in the context of MEC, UEs can offload their tasks' workload to the edge server located nearby at the BS, while the cloud can still be utilized for computing the tasks that can afford a high end-to-end delay.

The efficient computation offloading of heterogeneous services on the network edge with MEC has been made possible by leveraging the following technologies. The virtualization technique allows for multiple independent software instances to run on the same physical server through VMs and containers for performing different independent computations with a guaranteed service isolation, while supporting the UEs' mobility through the seamless migration of the VMs and containers. By utilizing the virtualization technique for implementing network functions as software modules that can run on general-purpose hardware with NFV, and decoupling the management of the control plane from the data plane with SDN, the centralized

network configuration and management can be simplified towards achieving a low-cost, flexible, easy, and fast deployment of software-based modules on the network edge. Leveraging the technologies of virtualization, NFV, and SDN, which support the flexible provisioning, isolation, and dynamic assignment of network resources, NS extends the capability of MEC by enabling a multi-tenancy environment through the creation and co-existence of multiple slices (i.e. virtual networks) for provisioning heterogeneous services on the same physical infrastructure [10, 24].

The network edge computation through MEC provides the following benefits ever needed for supporting the computation of the end-user devices with stringent latency requirements in 5G networks and beyond. First, MEC can significantly reduce the communication latency due to the processing capability provided in the proximity of the end-user devices. Also, MEC allows for alleviating the traffic load and bottlenecks in the core and backhaul networks [21]. Moreover, due to the edge servers being deployed in the RAN near the UEs, context-awareness is possible with detailed context information about the users that can be obtained, ranging from network-level to device-level information, which allows for a more efficient allocation of the network resources. Numerous studies have validated the performance and effectiveness of the edge computing architecture brought by MEC for assisting the energy-limited user devices in computing their tasks, and thus it now constitutes a building block in 5G and 6G networks for supporting the latency-sensitive services [25, 26].

1.7 Challenges Hindering The Edge-cloud Realization

Despite the good amount of research done for enabling efficient computation on the network edge, still multiple challenges which must be overcome through well-designed computation offloading solutions, hinder the realization of MEC with its full benefits for provisioning cutting-edge services with their latency and reliability sensitive characteristics [14].

Edge servers usually suffer from a limited computing capacity as compared to cloud data centers, creating a strife on the resources from end-user devices, making them easily overloaded in response to a high load of offloading requests, which offsets MEC's latency benefits in the absence of well-designed load balancing and optimized offloading and resources allocation solutions. Also, moving the computation to the network edge takes out the benefits of reliability, scalability, and flexibility, typically provisioned by the cloud with the MCC approach. Moreover, the computation offloading solutions must effectively accommodate the low-energy characteristics of IoT devices. In addition, the often unfavorable conditions of the wireless access channels, caused by blockages and deep fading, will severely impact the performance of computation offloading and the achieved latency and reliability. Moreover, situations of emergency scenarios, under-served networks, and CNs, will inhibit the availability of the MEC infrastructure for conducting the offloading operations. Also, in a rapidly and spontaneously changing environment, edge cloudlets can suffer from the inability to dynamically scale and adapt their resources to respond to the environment

changes in case no resources from other tiers are utilized. Furthermore, edge servers need to maintain a high resources' utilization in order to maximize the support for the offloading requests and hence avoid resources over-provisioning and wastage, which necessitates the design of efficient solutions for the computation and communication resources allocation. Moreover, seeing the novel business practices enabled by modern networks such as the NOs leasing network resources from established InPs, the designed computation offloading solutions must take into consideration the cost-efficiency with respect to the NOs.

1.8 Exploring Novel 5G Technologies For Supporting The Edge-cloud

In light of the challenges hindering the full realization of the vision of MEC with its promised benefits, the industry and academia have been recently exploring the use of novel 5G technologies and architectures, for assisting MEC in enabling the seamless provisioning of computation offloading for the modern services with their stringent latency and reliability QoS requirements.

1.8.1 Multi-tier MEC and Heterogeneous Networks

The hierarchical arrangement of the edge-cloud nodes in multiple tiers has been an appealing architecture for lessening the performance issues caused by the limited edge cloudlets' capacity, which prevents them from effectively provisioning the offloading service in periods of high load. As Fig. 1.6 shows, in this architecture, lower-tier

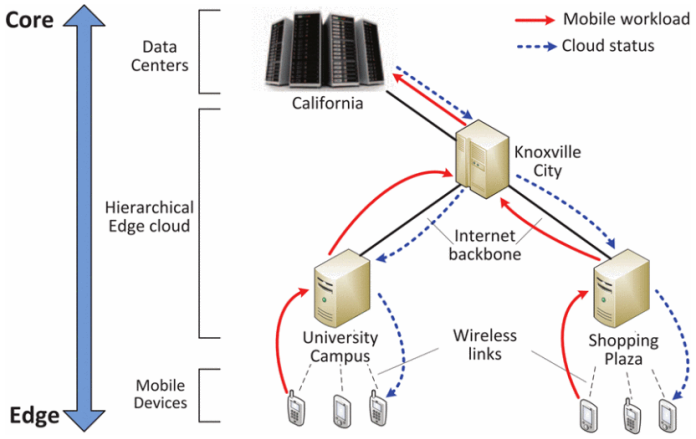


Figure 1.6: The hierarchical edge-cloud architecture [6].

cloudlets can migrate their workload to be computed on upper-tier counterparts, which typically have more powerful capabilities, and are co-located within nearby aggregation nodes and the core network, resulting in a limited number of hops as compared to the cloud [6, 27]. Formal analysis and results in [6] demonstrate the superiority of the hierarchical MEC system over a flat one in terms of the achieved latency, and hence its ability to serve a larger number of UEs more effectively.

By deploying low-power SCs overlaid within an MC near the end-user devices, the emergence of heterogeneous networks has allowed for improving the network capacity and the UEs' transmission rates, and thereby alleviating the bottleneck produced in the RAN in scenarios of high network load [28]. By leveraging a cloudlet co-located on the MC, a multi-tier MEC system can be formed, which allows energy-limited SC UEs to migrate their computational tasks to the MC cloudlet through the wireless backhaul, and therefore alleviating the load on the SC cloudlets, which reduces the computation latency overhead [29]. Therefore, such multi-tier MEC system where first-tier and second-tier cloudlets are co-located within the SCs and

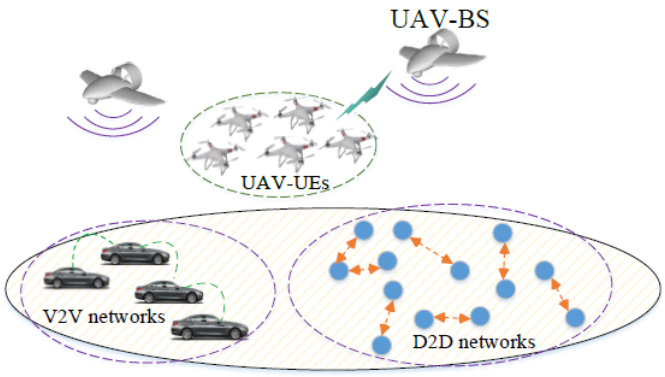


Figure 1.7: UAV-assisted terrestrial networks [7].

MC, respectively, will allow for provisioning the modern latency-sensitive 5G services in scenarios of high network load [6].

1.8.2 UAVs as Aerial Base Stations and Cloudlets

Seeing the unique advantages that they bring, the integration of low-altitude UAVs into cellular networks in a wide range of applications has been the subject of research over the past few years for helping to enhance the networks' capabilities [7, 30]. The key advantages of UAVs are their mobility, flexibility, and adaptive altitude, allowing them to swiftly move and change their position and altitude in order to enhance the coverage and transmission rates in cellular networks in response to environment changes, such as users' mobility, IoT devices' activation/deactivation, and load variations. Also, due to their high altitude, UAVs can deliver on-demand services in locations that may be restricted to ground BSs, and are able to establish LoS communication with ground devices and overcome network blockages and deep fading, which translates into a high capacity communication, achieving higher transmission

rates. In addition, due to their low-cost and easy planning and deployment, UAVs can be rapidly dispatched for a limited amount of time and later retired for dynamically adapting to the environment changes, which increases the levels of resiliency, flexibility, scalability, and cost-efficiency in the network. Due to their easy on-the-fly programmability, UAVs can also be configured to accomplish different tasks at their operational time.

There are several prospective applications for the deployment of UAVs in cellular networks [7]. UAVs can take the role of aerial-UEs for provisioning IoT services from the air, such as scenarios of surveillance, remote sensing, VR applications, traffic monitoring, and wind estimation [31]. As Fig. 1.7 shows, UAVs can also be deployed as aerial-BSs to assist in enhancing the connectivity of the end-user devices, and terrestrial networks such as D2D, V2V, densely deployed ground-BSs, and IoT networks, such as to avoid short communication ranges and alleviate interference problems between the ground devices, in addition to using air-to-air links to service other aerial-UEs. Also, aerial-BSs can provide better communication channels for the energy-limited IoT devices, allowing them to consume a lower communication energy. In addition, UAV-based aerial networks can benefit scenarios of public safety and under-served terrestrial networks, by enabling a fast, flexible, scalable, and reliable on-demand wireless communication to user devices when ground infrastructures are compromised or unavailable. Moreover, UAVs can be deployed in a fixed location as a flying backhaul to relay the communication between ground-BSs and the core network in scenarios of high load, and therefore alleviating the load in the ground backhaul network. Furthermore, by co-locating them with cloudlets, UAVs can be

utilized for provisioning edge computing capabilities in the air leveraging their unique advantages, and thereby significantly expanding the capabilities of the terrestrial MEC system in serving the offloading requests of the ground UEs [7, 32].

Although the introduction of UAVs in cellular networks brings many opportunities due to their unique advantages, multiple challenges need to be taken into consideration upon their adoption as aerial BSs and UAV-mounted cloudlets in cellular networks [7]. Unlike classical ground communication channels, the LoS and non-LoS GtA channels used for communicating with the UAVs are highly dependent upon the variable UAVs' position and altitude, and can also cause interference on the other air-to-air links. In addition, the UAVs' optimal positioning is a challenging task as it depends on many factors, such as the deployment environment, the locations of ground UEs, and the GtA channel which itself is a function of the UAVs' position. Moreover, UAVs have a limited amount of on-board energy which is used for mobility, communication, workload computation, and payloads purposes, leading to a limited operational time. Furthermore, UAVs are susceptible to node failure which will impact the computation reliability when they are mounted with cloudlets for conducting the computation offloading operation, and thereby inhibiting them from provisioning for reliability-sensitive mission-critical applications where unexpected disturbances must be minimized or completely prevented.

Thus, while observing the great benefit in leveraging UAVs for supporting the MEC system with UAV-mounted cloudlets in order to provide a low-latency and high-reliability computation offloading, there is a need to carefully optimize the UAVs'

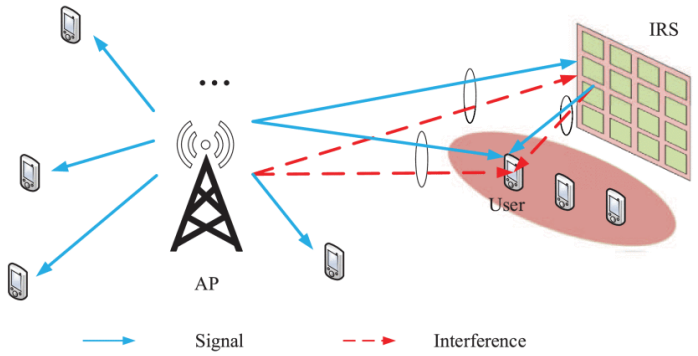


Figure 1.8: IRS-aided multi-user communication system [8].

positioning and the allocated computation and communication resources, while taking into account their limited energy and capability.

1.8.3 IRS-assisted Cellular and MEC-based Networks

Thanks to advancements in programmable meta-materials, IRSs have been recently introduced into cellular networks as a novel technology for passively enhancing the RAN wireless communications quality through exploiting a large number of low-cost low-energy reflecting elements [8]. To collaboratively enhance the signals' propagation environment, the amplitude and/or phase shift of the IRS elements' reflected signals can be tuned as has been demonstrated in [33, 34], and thereby improving the RAN communication rates such as in the case of the IRS-aided multi-user communication system illustrated in Fig. 1.7.

Since the IRSs can enhance the transmission rates and upload latencies in networks that have unfavorable wireless access channels, we envision the introduction of IRSs in MEC systems to play a significant role in reducing the UEs' offloading energy

consumption, and in alleviating the load on the MEC resources. Therefore, an IRS-aided MEC system would assist the UEs in accessing resources-rich MEC servers that are normally unreachable or require a high transmission energy due to low-quality channels, which would help in enabling modern 5G services with stringent latency and reliability requirements in networks with unfavorable channel conditions.

1.9 Challenges and Contributions

Interesting challenges and questions come up with regard to integrating MEC with the presented novel 5G technologies and architectures in cellular networks.

1. The allocation mechanism for the limited edge resources in a MEC system needs to guarantee maximizing their utilization and avoiding their wastage in order to maximize the amount of served offloading requests. Also, the allocation of the wireless communication resources needs to be carefully considered since it has a direct effect on the achieved latency and reliability, and thus creating a coupling between the computation and communication resources' allocation. Moreover, the decision on how to associate the UEs to AP cloudlets is coupled with the computational resources' allocation strategy, which complicates the offloading problem.

2. The NO leasing the second-tier cloudlet's resources from an InP for assisting the edge offloading service in a multi-tier MEC system, would generally incur a higher cost due to the lease price being dependent upon economical and other complex factors. In this case, selecting the AP that requires the least offloading

energy for a given UE may not be cost-efficient due to the potential overload of the residing cloudlet, requiring the utilization of the higher-cost second-tier cloudlet. Thus, there is a necessity for cost-efficient resource allocation strategies coupled with optimized UEs-to-AP mapping while accounting for the UEs' energy consumption.

3. The UEs in a MEC-based heterogeneous network would be forced to compute their tasks locally when the limited-capacity SC cloudlets are overloaded, incurring a higher energy consumption which could not be afforded, in addition to being impacted by SCs' inter-cell interference in the RAN which would negatively affect the offloading rates. Thus, there is a need for efficiently optimizing the association between the UEs and SC cloudlets, and for the efficient utilization of MC among all SC and MC UEs, while effectively utilizing the low-cost wireless backhaul and accounting for the resulting RAN interference.

4. When LoS communications cannot be established in a UAV-aided MEC system due to GtA channels fading and blockage such as in high urban environments, the channel state (which the optimized position depends on), cannot be known without knowing the UAVs' position, which makes the UAVs' positioning very challenging, necessitating the use of predictive techniques and probabilistic models. In addition, the allocation of UAV-mounted cloudlets' very limited resources must be carefully optimized such that the amount of served requests can be maximized. Also in such system, jointly optimizing the UAVs' positioning along with the UEs' offloading decisions and the allocated resources while accounting for the UAVs' limited energy, is very challenging.

5. When catering for the stringent latency and reliability requirements in a MEC system, increasing the computation reliability would require the redundancy of tasks' computation on multiple cloudlets which would in turn incur a higher latency, making it difficult to cater for both requirements. Also when the IRS is utilized to aid the MEC system with computation redundancy for the offloaded tasks, the judicious sharing of the IRS elements among the limited-energy UEs through the phase shifts' optimization, is a challenging problem.

This thesis aims at providing a deep understanding of the aforementioned challenges and their inter-dependency while proposing several novel solution approaches that efficiently address them under different network designs and assumptions. We present our main contributions in the following.

1. **Latency-aware Cost and Energy Efficient Computation Offloading in Multi-tier Edge-clouds** A hierarchical arrangement of the edge cloudlets has shown to be successful in expanding the limited capabilities of the MEC system for providing computation offloading for the energy-limited UEs [6]. However, a cost disparity between the edge tiers leads to cost-inefficient solutions from the NOs' perspective, in a scenario where the NO is leasing resources of a high-tier central cloudlet. Motivated by the lack of research in this area, we proceed to study a multi-tier MEC system that is provisioning latency-sensitive services with the support of a second-tier edge-cloud, where we jointly minimize the NO's computational cost and the UEs' energy consumption, by optimizing the UEs-to-AP association, the UEs' offloading decisions, the UEs' transmission power, and the allocated computation and uplink communication resources.

We model and mathematically formulate our mixed-integer non-convex program, and propose a BnB algorithm which uses exhaustive search to solve the problem optimally. Due to the BnB's complexity, we propose a low-complexity algorithm based on the SCA method to solve and obtain a high-quality solution, and also present an inflation-based algorithm for obtaining a polynomial-time and efficient solution. Numerical results show the performance and scalability of the proposed algorithms, demonstrate their efficiency, and uncover insights for helping the NOs better manage their resources following various configurations. To the best of our knowledge, this study is the first attempt at exploring the computation offloading problem for minimizing the NOs' costs in a multi-tier edge-cloud system as it applies in 5G networks.

2. **Latency-aware Macro-cell Assisted Edge-clouds Offloading in Heterogeneous Networks with Wireless Backhaul** Due to the interest of operating MEC in heterogeneous networks which have allowed network operators to enhance the spectral efficiency and support a large number of end-user devices, we extend the scenario in the previous problem to study the energy-efficiency of computation offloading in a network of small-cells with the second-tier edge server being co-located within the MC which can be reached through a wireless backhaul. Our main motivation is to explore the performance benefit of leveraging the second-tier MC cloudlet for computing the computational tasks of the small-cell users', while optimizing the allocation of the backhaul communication resources and accounting for the resulting RAN interference. We proceed to minimize the UEs' energy consumption, by optimizing the SC

UEs' partial offloading decision (local, SC-cloudlet, or MC-cloudlet), the MC UEs' offloading decision (local or MC-cloudlet), the UEs' transmission power, and the allocated computation and OFDMA communication resources, while respecting the UEs' latency deadline. We model and mathematically formulate the problem as a non-convex MI-NLP, and due to its complexity, we propose an iterative algorithm based on the SCA approach that provides an approximate solution to the original problem. Through numerical analysis, we perform simulations based on varying configurations, and demonstrate the performance and efficiency of our proposed solution. The significance of this study lies in considering a wireless backhaul, which is adopted in most cases due to its relatively low-cost and easier installation, while requiring interference management and the optimization of the communication resources' allocation.

3. **Latency and Reliability Aware Computation Offloading via UAV-mounted Cloudlets in IoT Networks** Due to MEC's rigidity and susceptibility to infrastructure failures, in addition to often being hindered by the weak wireless signals caused by obstacles and high network load in the RAN, which would prevent from provisioning flexible computation offloading with strict latency and reliability requirements, the usage of UAV-mounted cloudlets for improving MEC's performance has been studied, thanks to the UAVs' unique advantages in expanding the capabilities of cellular networks and enhancing the channel conditions. However, since UAV-mounted cloudlets may have failure rates that would disrupt mission-critical applications and other latency and reliability sensitive services in IoT networks, we proceed to investigate a

novel study for the provisioned reliability in a UAV-aided MEC system considering the UAV-mounted cloudlets' failure rates and tasks redundancy. We aim at maximizing the amount of served requests, by optimizing the UAVs' positions, the UEs-to-UAV associations, and the allocated computation and communication resources considering both LoS and non-LoS components, while guaranteeing the stringent latency and reliability requirements, and respecting the UAVs' available energy levels. The problem is divided into a planning problem for optimizing the long-term placement of UAVs, and an operational problem for making optimized offloading and resource allocation decisions with constrained UAVs' energy corresponding to the specific requests in a particular time slot. We formulate both problems associated with each phase as non-convex MI-NLPs, and due to their non-convexity, we perform customized conversions to transform them into approximate SOCPs, and then propose an efficient customized algorithm for solving the overall problem based on the SCA method. Further, we approach the problem considering the task partitioning model which will be prevalent in 5G networks. Through numerical analysis, we demonstrate the effectiveness of our proposed solution by comparing it to other baseline approaches, and study the achieved gains considering various scenarios.

4. **Latency and Reliability Aware Computation Offloading in IRS-aided Edge-clouds** Seeing the IRSs' ability to also enhance through the tuning of their passive reflecting elements the quality of the weak wireless signals in the RAN which often greatly impact MEC's offloading performance, we are

motivated to extend the latency and reliability aware study to a scenario of an IRS-aided MEC system. We explore the optimized use of the IRSs for enhancing the performance of the MEC system while considering the cloudlets' failure rates and computation offloading tasks redundancy for the purpose of minimizing the UEs' energy consumption. We start by studying a single-user network where we optimize the IRS elements' phase shift, the UE's offloading decision, and the UE's transmission power. Then, we extend the study to a multi-user network considering the OFDMA communication model, where we optimize the IRSs elements' phase shift, the UEs' offloading decisions, the UEs' transmission power, and the allocated servers' computational resources and OFDMA RBs. For each of the presented non-convex mathematical problem, we propose a customized sub-optimal solution based on the SCA approach and the SDR technique, where the problem is divided into multiple sub-problems that are solved separately in an alternating fashion. Numerical results are illustrated for the presented solutions, which demonstrate the energy reduction and the saving in network resources in various scenarios achieved by the optimized use of the IRSs, especially for offloading services with higher reliability requirements. Our work provides insights on leveraging IRS-aided APs for reducing the energy consumption of UEs that are requesting services with low-latency and high-reliability requirements such as mission-critical applications, for influencing the design of the MEC network parameters, and for reducing the load on the MEC resources.

1.10 Thesis Outline

The rest of the thesis is organized as follows. Chapter 2 studies the problem of the joint optimization for the NO's computational cost and the UEs' energy consumption in the context of a multi-tier edge-cloud system. Chapter 3 extends the energy-efficiency study to a heterogeneous network scenario with a wireless backhaul and an MC cloudlet deployed for assisting the MEC Offloading process. Chapter 4 explores the reliability-aware optimized use of UAV-mounted cloudlets for provisioning the MEC computation Offloading service with stringent latency and reliability guarantees through the use of tasks redundancy. Chapter 5 extends the latency and reliability study to an IRS-aided MEC system for enhancing the computation offloading performance in scenarios with unfavorable RAN channel conditions. Finally, Chapter 6 concludes the thesis and highlights potential research problems for future consideration.

Notations which are used throughout the thesis are independent from one chapter to another. Hence, some symbols may appear in different chapters and serve different purpose.

Chapter 2

Latency-aware Cost and Energy Efficient Computation Offloading in Multi-tier Edge-clouds[1]

Observing the limited capacity that edge servers possess which offsets their benefits in periods of high load, a hierarchical arrangement of the edge cloudlets has been studied, and has shown to be successful in expanding their capabilities. Yet, considering the emerging business models in 5G networks, the cost disparity between the edge tiers has been until now ignored, leading to cost-inefficient solutions from the NOs' perspective. In this work, we consider a NO that is leasing resources of a high-tier central cloudlet for computation offloading, where we jointly minimize the NO's computational cost and the UEs' energy consumption in a multi-tier MEC

[1]This chapter has been published in IEEE Transactions on Communication [35].

system, by optimizing the UEs-to-AP association, the UEs' offloading decisions, the UEs' transmission power, and the allocated computation and uplink communication resources, while respecting the UEs' latency requirement. We mathematically formulate our non-convex MI-NLP, and propose a BnB algorithm for obtaining the optimal solution. Due to the BnB's complexity, we propose a low-complexity algorithm based on the SCA approach to solve and obtain a high-quality solution, and also present an inflation-based algorithm for obtaining a polynomial-time and efficient solution. Numerical results show the performance and scalability of the algorithms, demonstrate their efficiency, and uncover insights for helping the NOs better manage their resources following various configurations.

2.1 Introduction

Due to the limited capability of edge cloudlets that prevents them from effectively provisioning the offloading service in periods of high load, recent studies have proposed a hierarchical arrangement of cloudlets in multiple edge-tiers, where upper tiers consist of more powerful cloudlets, and can receive migration requests from the lower-tier cloudlets in case of overload [6, 27]. Formal analysis and results in [6] demonstrate the superiority of a hierarchical MEC system over a flat one in terms of the achieved latency, and hence its ability to serve a larger number of UEs. In reality, the edge tiers can be provisioned on an existing network infrastructure, e.g. leveraging nearby aggregation nodes and the core network for cloudlets installation. In fact, this model brings a latency advantage due to the limited number of hops, in contrast

to existing solutions that solely rely on the assistance of a distant cloud data center for expanding the edge-cloud capabilities. A multi-tier MEC model can be leveraged by service-specific network slices, where the virtual computing resources can be hosted by a set of multi-tier cloudlets, and hence allowing each slice to efficiently contain the load of its users [36]. An example of a NS model on top of a hierarchical edge-cloud architecture is illustrated in Fig. 2.1, where three first-tier cloudlets and one second-tier cloudlet are provisioning virtual computation and communication resources for two slices that provide self-driving and healthcare services.

Meanwhile, it will be common for NOs within the emerging 5G business models to lease high-tier edge cloudlet and communication bandwidth resources from an InP through multi-tenancy [37, 38]. This allows the NOs to expand the capacity of their edge cloudlets in order to serve the offloaded requests especially in periods of high load, while imposing fees on utilizing the cloudlet and the communication channels. In fact, the NOs in most cases would be better off leasing cloudlet resources from established InPs in order to reduce their capital expenditure, instead of going through the hurdle of installing proprietary cloudlets, especially when the NOs face high requests load only at specific periods. However, the utilization of the leased cloudlet resources in such case will incur a higher cost on the NOs in many situations, since the lease price is dependent upon economical and other complex factors. For instance, due to its large capacity, the leased cloudlet has high capital and operational expenditures that the InP is trying to offset. Moreover, the price could go higher depending on the amount of resources leased for the NOs' customers, where the InP is trying to maximize their own profit [39]. However, the NOs have control

over optimizing the cost-efficiency and improving the multiplexing gains of its small-scale edge cloudlets, and therefore can lower their operational costs. Meanwhile, it is paramount for the NOs to maintain a minimal UEs' energy consumption, which ultimately depends on the tasks workload size and the wireless channel strength, and the UEs in many cases may be offloading multiple tasks in order to run their service. In such scenario, selecting the best AP in terms of channel strength that requires the least offloading energy may not be cost-efficient due to the potential overload of the residing cloudlet, necessitating the utilization of the higher-tier cloudlet which incurs increased costs. Thus, a non-optimal tasks-to-cloudlet mapping will result in an inefficient solution in terms of the NO's computational cost and the UEs' energy consumption.

In this chapter, motivated by the aforementioned challenges and by the computational cost being mostly ignored in the literature, we study the computation offloading problem in a multi-tier MEC system, where we jointly minimize the NO's computational cost and the UEs' energy consumption, by optimizing the offloading decision, the UEs' transmission power, and the allocated computation and communication resources, while guaranteeing the UEs' latency requirement. To the best of our knowledge, this is the first attempt at exploring the computation offloading problem for minimizing the NOs' cost in a multi-tier edge-cloud system as it applies in 5G networks.

2.1.1 Novel Contributions

The contributions of this chapter can be summarized as follows:

1. We model and mathematically formulate the problem of joint transmission power and computation/communication resources' allocation for computation offloading in multi-tier edge-clouds for minimizing the computational cost and UEs' energy consumption while meeting their latency deadline.

2. Due to its complexity, we transform the non-convex problem into a more tractable form and present a high-complexity BnB algorithm which uses exhaustive search to solve the problem optimally.

3. Due to the scalability and complexity issues of the BnB algorithm that quickly become excessive, we first transform the problem into a SOCP that can be efficiently solved via the interior point method [40], and then propose an efficient low-complexity algorithm based on the SCA method [41] which provides an approximate solution by iteratively solving until convergence.

4. We also propose a relax-continuous and inflation-based algorithm to solve the problem with more efficiency, overcoming the time bottleneck caused by the mixed-integer nature of the low-complexity SCA-based algorithm. This new algorithm is a more appealing approach and is able to run in a polynomial time.

5. We present numerical results that evaluate the performance of our proposed algorithms and demonstrate their efficiency based on various system configurations, and obtain new insights that will help the NOs decrease their costs and UEs' energy when serving the offloaded tasks.

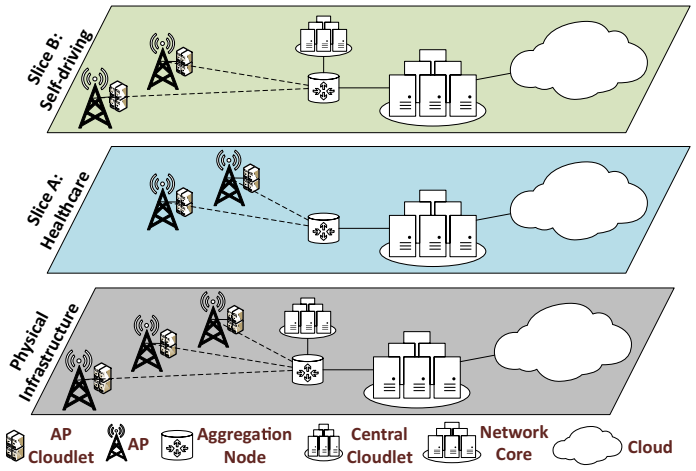


Figure 2.1: NS Applied in Hierarchical MEC.

The remainder of this chapter is structured as follows. Section 2.2 explores the related literature. Section 2.3 introduces the system model. Section 2.3.6 presents the mathematical formulation. In section 2.4, we present the BnB exhaustive search algorithm. In section 2.5, we propose two low-complexity SCA-based algorithms for obtaining approximate solutions on the original problem. In section 2.6, we evaluate the numerical results and demonstrate the efficiency of our solutions based on various parameters. Finally, section 2.7 concludes the chapter.

2.2 Literature Review

Research on problems addressing the design of a hierarchical edge-cloud that inspired this work has recently started. In [6], a hierarchical edge-cloud architecture is proposed, and its latency advantage is demonstrated over flat edge-cloud using formal analysis and simulation. [42] presents the hierarchical edge-cloud system as

an enabler for the Industry 4.0 and smart factory paradigms, where the computation and storage capacity highlighted are larger for upper tiers, and also describes existing challenges such as the infrastructure energy and cost. In [43], the authors studied the design of an edge-cloud network where the placement decision for cloudlets among the available sites is made (edge, aggregation, or core nodes). This work is motivated by the findings in [6], and is based on a two-tier hierarchical edge-cloud where edge cloudlets are connected to a central cloudlet for containing the loads.

The following are papers that studied computation offloading problems in a MEC system without considering a hierarchical edge-cloud. [44, 45, 46] considered a single-server MEC system. [44] explored the latency minimization problem for one user with multiple tasks connected to one MEC server subject to the mobile power consumption constraint, where the stochastic optimization problem is solved using an MDP approach. [45] studied the joint energy and latency minimization while optimizing the transmission power and tasks scheduling of one user connected to one MEC server. [46] used a game-theoretic approach to minimize the energy consumption and latency while optimizing the processing rate and offloading decision of one user (local, edge, or central cloud computation). [47, 48, 49] considered a multi-server MEC system. In [47], a stochastic optimization technique is employed to minimize the power consumption of multiple users that are sharing the radio spectrum, by optimizing the transmission power and allocated local and server computation. [48] used the theory of minority games to guarantee energy-efficient MEC servers by activating a subset of them while also guaranteeing users' latency requirement. [49], optimized the task allocation and local CPU frequency decisions for minimizing the

total latency and energy consumption of one multi-task user with a multi-channel access.

The following are among the few papers that studied cost-efficiency for computation offloading in a MEC system without considering a hierarchical edge-cloud. [50] studied the computation and transmission cost minimization problem for several users over a multi-heterogeneous cloudlets framework, subject to the latency constraint considering a fixed channel rate. [51] proposed a solution to offload MEC server's load to the cloud for load balancing, in order to minimize the total users' latency, MEC fixed cost, and cloud computational cost. [52] minimized the computational cost subject to the latency constraint in a multi-edge servers system, where servers can allocate computational resources or offload demands to other servers. [53] jointly minimized the users' energy, latency, and system utility cost while optimizing the offloading decision (local or cloud) and the allocation of communication resources. [54] adds improvement on [53], by considering a cloudlet-like unit at the AP (called CAP), adding a third offloading layer, where the allocation of computational resources on the CAP is also optimized.

The following works addressed computation offloading problems in a hierarchical edge-cloud without studying the cost efficiency. [27] aimed to maximize the NO's revenue using an auction-based approach, by optimizing the accepted bids and allocation of computation and communication resources, but without considering the difference of computational costs across tiers that results in 5G multi-tenancy scenarios. [55] proposed a context-aware hierarchical MEC system with cloud support,

where the layer and service-type for each edge server is first decided, and then the allocation of computation/content requests to one edge tier and server is optimized by performing a lookup, but without addressing the cost aspect. [56] addresses computation offloading in a hierarchical edge-cloud connected to a cloud data center, where the operational cost and network latency are minimized by optimizing the allocation of tasks on the MEC servers, while assuming fixed computation and bandwidth for all tasks, which makes the problem too simplistic.

In this work, compared to the existing studies, we go one step beyond and address a business model that will be common in 5G multi-tenancy scenarios, where we study the cost efficiency for a NO that is leasing resources from an InP for expanding the edge-cloud capability with response to the network load increase. Furthermore, in contrast to studies that either did not address the UEs' energy or considered a fixed transmission power and rate, we make sure to also tackle the UEs' energy aspect by optimizing the allocation of transmission power and the selection of APs taking into account the uplink wireless channel state which makes the transmission rates unknown in advance. Also, in contrast to studies that considered a cloud-assisted MEC system which would incur a high transmission latency, we consider a higher-tier central cloudlet as a support for the MEC system. To the best of our knowledge, this work is the first to address the minimization problem of cost and energy by jointly optimizing the UEs-to-AP association, UEs' transmission power, and the allocated cloudlets' resources in the context of computation offloading as it applies in the 5G multi-tenancy scenarios while utilizing a hierarchical edge-cloud.

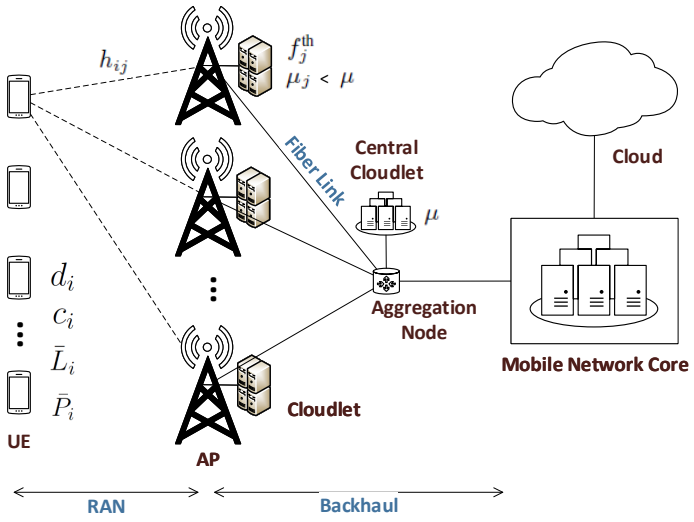


Figure 2.2: System Model.

2.3 System Model

2.3.1 MEC Model

Without a loss of generality, as depicted in Fig. 2.2, we consider a two-tier MEC system where the NO has a set of M first-tier cloudlet-enabled APs indexed by $\mathcal{M} = \{1, 2, ..., M\}$, and is leasing resources of a more powerful second-tier cloudlet from an InP that we call a central cloudlet. Within the considered RAN, N UEs indexed by $\mathcal{N} = \{1, 2, ..., N\}$ are requesting to offload one task as part of the service provided by the slice, and are assumed to have very limited capabilities, and hence cannot afford local computation. Each task $i \in \mathcal{N}$ is represented by the tuple $\{d_i, c_i, \bar{L}_i, \bar{P}_i\}$, concatenating the task input size d_i (Kb), the task computational demand c_i (CPU cycles/bit), the required latency threshold $\bar{L}_i$ (ms), and the maximum power budget $\bar{P}_i$ (dBm).

Furthermore, each AP cloudlet $j \in \mathcal{M}$ possesses some computation capability with capacity f_j^{th} (GHz), while the InP with its large resource-rich central cloudlet is assumed to serve all migrated requests with no fixed capacity, noting that the resources provisioned by the InP could be abstracted from different sources possibly from upper tiers. Our model can be thought of as having M field cloudlets supported by one shallow cloudlet in accordance with the principles of LTE-advanced backhaul network [27], noting that it can be extended for the case of multiple shallow cloudlets and third tier deep cloudlets. Also, our choice of one cloudlet is supported by the formal analysis in [6], showing that the employment of one cloudlet in the second-tier of a two-tier MEC system maximizes the resources' utilization efficiency.

2.3.2 Communication Model

We consider the uplink communication between the UEs and the APs to be accommodated on the whole spectrum of B Hertz. Each UE i will be allocated a fraction $\alpha_i \geq 0$ of bandwidth B for transmitting its task to only one AP j, where $\sum_{i \in \mathcal{N}} \alpha_i = 1$. We do not consider the downlink communication since the task output size is in general much smaller than the task input size, e.g. face recognition [57]. We note that APs are assumed to have small separating distances, and thus operate on a different licensed spectrum in order to avoid severe interference [47]. Since, we later aim at determining the association between UE i to AP j, $\forall\{i, j\} \in \{\mathcal{N}, \mathcal{M}\}$, let us introduce a decision binary variable x_{ij}, where $x_{ij} = 1$ implies that UE i is associated to AP j and $x_{ij} = 0$ otherwise. By denoting p_{ij} and h_{ij} as the transmission power and channel gain from UE i to AP j on the allocated $\alpha_i B$ spectrum, respectively,

the transmission rate achieved at UE i, computed in Mbps, can be written as

$$R_i(\boldsymbol{p}_i, \alpha_i) = \alpha_i B \sum_{j \in \mathcal{M}} \log_2 \left(1 + \frac{p_{ij} h_{ij}}{\alpha_i B N_0}\right) \tag{2.1}$$

where we assume a perfect CSI, and $\boldsymbol{p}_i = \{p_{ij} \geq 0, \forall j \in \mathcal{M}\}$ is the set of transmission power from UE i, and N_0 is the noise power spectral density. It is important to note that since UE i transmits to only one AP j, we impose the following constraints between $\boldsymbol{p}_i$ and $\boldsymbol{x}_i = \{x_{ij}, \forall j \in \mathcal{M}\}$ to govern a proper relationship of the involved variables

$$\sum_j x_{ij} = 1 \tag{2.2a}$$

$$p_{ij} \leq x_{ij} \bar{P}_i \tag{2.2b}$$

While it is obvious that (2.2a) regulates that UE i is associated to only one AP, (2.2b) restricts that when $x_{ij} = 0$, UE i does not transmit to AP j. We also assume that the first-tier APs can communicate with the central cloudlet through backhaul wired links with enough high bandwidth [43], and hence the task migration latency is assumed to be negligible [57, 58]. We note that a wireless backhaul communication can be utilized instead, as in [59]. However, this is out of scope of this work, and hence we leave that to be explored in a future work.

2.3.3 Latency Model

When task i is offloaded, an amount of resources $f_i \geq 0$, $\forall i \in \mathcal{N}$ (cycles/second) is allocated for its computation, knowing that task i must be computed only on one cloudlet, and thus f_i does not indicate where task i is actually computed. Thus, the time needed for the computation of task i on the host cloudlet, denoted as execution latency $L_i^{\text{ex}}(f_i)$, is:

$$L_i^{\text{ex}}(f_i) = \frac{d_i c_i}{f_i} \tag{2.3}$$

We note that allocating more resources f_i will decrease the execution latency, but this will drive the costs up, as will be later explained. The time needed for UE i to upload its task to AP j of length d_i Kb, depends upon the allocated transmission power and bandwidth, and is given by

$$L_i^{\text{u}}(\boldsymbol{p}_i, \alpha_i) = \frac{d_i}{R_i(\boldsymbol{p}_i, \alpha_i)} \tag{2.4}$$

It can be seen that allocating higher power and more bandwidth will decrease the transmission latency, but with a cost of an energy increase and a decrease in the available bandwidth, respectively.

2.3.4 Energy Model

The consumed energy (Mj) resulting from offloading task i, can be computed as the product between the total transmission power of UE i, e.g., $\sum_{j \in \mathcal{M}} p_{ij}$, and the time

to complete the upload. c.f. (2.4):

$$\mathcal{E}_i(\boldsymbol{p}_i, \alpha_i) = \frac{d_i \sum_{j \in \mathcal{M}} p_{ij}}{R_i(\boldsymbol{p}_i, \alpha_i)} \tag{2.5}$$

2.3.5 Cost Model

The computational cost depends on the resource unit cost μ_j at each AP j, and μ at the central cloudlet. It is important to note that the considered cost model acts as a deterrent for pushing tasks to be executed on the edge server whenever possible, due to the higher cost incurred on the NO for central cloudlet utilization as noted in Section 2.1, thus $\mu_j < \mu$. For instance, the considered central cloudlet unit cost μ is abstracted from the imposed resource price by the InP, the price of utilizing the communication link, and possibly other factors. The unit cost is computed in cent/gigahertz.

Let us denote $y_i = \{0, 1\}$ as a binary decision variable, where $y_i = 1$ means that task i is executed at the central cloudlet, and $y_i = 0$ means that task i is executed at the cloudlet associated with AP j such that $x_{ij} = 1$. The computational cost of offloading task i which is relative to the allocated resources for its execution at either cloudlet, is:

$$\mathcal{C}_i(\boldsymbol{x}_i, y_i, f_i) = y_i f_i \mu + \sum_{j \in \mathcal{M}} x_{ij}(1 - y_i) f_i \mu_j \tag{2.6}$$

It is important to note that μ can be written as $\mu(a) = a \max(\mu_j, \forall j \in \mathcal{M})$ assuming all μ_j are different, and therefore (2.6) can be rewritten as $\mathcal{C}_i(\boldsymbol{x}_i, y_i, f_i) = y_i f_i \mu(a) + \sum_{j \in \mathcal{M}} x_{ij}(1 - y_i) f_i \mu_j$ where a is a scalar value. Thus, It can be seen that a and μ_j

act as scalars for $y_i f_i \mu(a)$ and $\sum_{j \in \mathcal{M}} x_{ij}(1 - y_i) f_i \mu_j$. Hence, from a mathematical point of view, a and the ratio between $\mu_j, \forall j \in \mathcal{M}$ are the important factors instead of the exact values of μ and μ_j. We explore the effect of change in the ratio a later on in Section 2.6.

2.3.6 Problem Formulation

Our objective is to minimize the weighted sum of energy and cost, while respecting the UEs' latency requirement. This is done by optimizing for each UE i: the associated AP j, the transmission power to that AP, the allocated uplink communication resources, the assigned cloudlet, and the allocated amount of computation. We adopt a centralized approach where a computation unit utilized by the NO, e.g. located on the central cloudlet, is responsible for conducting the optimization program after acquiring the CSI and the UEs configurations through the control plane [19], while the offloading and transmission power decisions would be signaled back to the UEs after the optimization procedure is finished. We note that the energy sensitivity for each UE could also be signaled to the central unit, and can be then used to set the objective weights accordingly. Our problem can be applied to address a snapshot of the system at one time instance, where the set of UEs with offloading requests are positioned in certain locations with corresponding channel gains, noting that another snapshot can also be solved when the UEs' number and state change after a given amount of time.

By denoting $\boldsymbol{p} = \{\boldsymbol{p}_i, \forall i \in \mathcal{N}\}$, $\boldsymbol{\alpha} = \{\alpha_i \geq 0, \forall i \in \mathcal{N}\}$, $\boldsymbol{x} = \{\boldsymbol{x}_i, \forall i \in \mathcal{N}\}$,

$\boldsymbol{y} = \{y_i, \forall i \in \mathcal{N}\}$, $\boldsymbol{f} = \{f_i \geq 0, \forall i \in \mathcal{N}\}$, the joint transmission power and computation/communication resources' allocation for computation offloading problem $\mathcal{P}_1$ is, formulated as

$$\mathcal{P}_1 : \min_{\substack{\boldsymbol{p},\boldsymbol{\alpha},\boldsymbol{x},\\ \boldsymbol{y},\boldsymbol{f}}} \quad \sum_{i\in\mathcal{N}} (\gamma \mathcal{E}_i(\boldsymbol{p}_i, \alpha_i) + \beta \mathcal{C}_i(\boldsymbol{x}_i, y_i, f_i)) \tag{2.7a}$$

$$\text{s.t.} \quad \sum_{i\in\mathcal{N}} x_{ij}(1 - y_i) f_i \leq f_j^{\text{th}}, \forall j \in \mathcal{M} \tag{2.7b}$$

$$L_i^{\text{ex}}(f_i) + L_i^{\text{u}}(\boldsymbol{p}_i, \alpha_i) \leq \bar{L}_i, \forall i \in \mathcal{N} \tag{2.7c}$$

$$\sum_{i\in\mathcal{N}} \alpha_i = 1 \tag{2.7d}$$

$$(2.2), \forall \{i, j\} \in \{\mathcal{N}, \mathcal{M}\} \tag{2.7e}$$

$$x_{ij}, y_i, \in \{0, 1\}, \forall \{i, j\} \in \{\mathcal{N}, \mathcal{M}\},$$

$$p_{ij}, \alpha_i, f_i \geq 0, \forall \{i, j\} \in \{\mathcal{N}, \mathcal{M}\} \tag{2.7f}$$

where γ, β in (2.7a) are the weights used for signaling the significance of the energy and cost sub-objectives, respectively. Constraint (2.7b) makes sure the resources' capacity of each AP cloudlet is respected. Constraint (2.7c) is for respecting the required latency for each task. Constraint (2.7d) ensures the wireless communication resources' capacity is respected. It is worth noting that our model is sufficiently broad, since weights can be adjusted to emphasize a sub-objective over another, and possibly focusing on minimizing only the energy or cost. All mathematical symbols used thus far are summarized in Table 2.1.

It can be seen that the objective (2.7a) and constraint (2.7c) are non-convex. In addition, constraint (2.7f) implies that (2.7) is an integer optimization problem. In

Notation	Description
$\mathcal{N}$	Set of UEs
$\mathcal{M}$	Set of APs
d_i	Task input size (Kb)
c_i	Task computational demand (CPU cycles/bit)
L_i	Task required latency (ms)
P_i	UE's maximum power budget (dBm)
γ	Energy sub-objective weight
β	Cost sub-objective weight
f_j^{th}	Cloudlet j computation capability (GHz)
μ_j	Edge cloudlet j resource unit cost
μ	Central cloudlet resource unit cost
B	Radio spectrum bandwidth (MHz)
N_0	Noise power spectral density (dBm/MHz)
h_{ij}	Channel gain between UE i and AP j
$x_{ij} \in \{0, 1\}$	Indicates if UE i is offloading using AP j
$y_i \in \{0, 1\}$	Indicates the host cloudlet for task i
$p_{ij} \in \mathbb{R}^+$	Allocated transmission power for UE i on AP j
$\alpha_i \in \mathbb{R}^+$	Allocated fraction of the bandwidth
$f_i \in \mathbb{R}^+$	Allocated computational resources for task i

Table 2.1: Table of Notations

fact, the formulated problem (2.7) is a mixed-integer non-convex program, which is generally difficult to solve.

2.4 Optimal Solution Approach via Branch-and-Bound Method

In this section, we solve (2.7) optimally using BnB exhaustive search algorithm. We first equivalently transform problem (2.7) into a more tractable form using the well-known big-M technique, where $A \gg 1$ is the big-M constant, to facilitate the

difficulty of handling the binary-related objective function and constraints. Specifically, we introduce the new slack variables $u_{ij} \geq 0$ and $v_i \geq 0$ for $i \in \mathcal{N}, j \in \mathcal{M}$, to substitute the terms $x_{ij}(1-y_i)f_i$ and $y_i f_i$, respectively, in (2.6), and (2.7b). In fact, the new slack constraints together with the slack variables u_{ij} and v_i can be presented according to the big-M method as

$$u_{ij} \leq x_{ij} A \tag{2.8a}$$

$$u_{ij} \leq (1-y_i)A \tag{2.8b}$$

$$(x_{ij} + (1-y_i) - 2)A + f_i \leq u_{ij} \leq f_i \tag{2.8c}$$

$$v_i \leq y_i A \tag{2.8d}$$

$$(y_i - 1)A + f_i \leq v_i \leq f_i \tag{2.8e}$$

We remark that each constraint in (2.8) is now linear with respect to the involved variables. At this point, by denoting $\boldsymbol{u} = \{u_{ij}, \forall i \in \mathcal{N}, j \in \mathcal{M}\}$ and $\boldsymbol{v} = \{v_i, \forall i \in \mathcal{N}\}$, $\mathcal{P}_1$ can be equivalently transformed into the following form:

$$\mathcal{P}_{2a}: \min_{\substack{\boldsymbol{p},\boldsymbol{\alpha},\boldsymbol{x},\\ \boldsymbol{y},\boldsymbol{f},\boldsymbol{u},\\ \boldsymbol{v}}} \quad \sum_{i\in\mathcal{N}} \Big(\gamma \mathcal{E}_i(\boldsymbol{p}_i, \alpha_i) + \beta(v_i \mu + \sum_{j\in\mathcal{M}} u_{ij}\mu_j)\Big) \tag{2.9a}$$

$$\text{s.t.} \quad \sum_{i\in\mathcal{N}} u_{ij} \leq f_j^{\text{th}} \tag{2.9b}$$

$$u_{ij}, v_i \geq 0, \forall \{i,j\} \in \{\mathcal{N}, \mathcal{M}\} \tag{2.9c}$$

$$\text{(2.7c)–(2.7f), (2.8).} \tag{2.9d}$$

Next, by introducing slack variables $\boldsymbol{t} = \{t_{ij}, \forall i \in \mathcal{N}, j \in \mathcal{M}\}$, $\boldsymbol{\zeta} = \{\zeta_i \geq 0, \forall i \in$

$\mathcal{N}\}$, and η, we proceed to equivalently transform (2.9) into the following form:

$$\mathcal{P}_{2b}: \underset{\substack{\boldsymbol{p},\boldsymbol{\alpha},\boldsymbol{x},\boldsymbol{y}\\ \boldsymbol{f},\boldsymbol{u},\boldsymbol{v},\boldsymbol{t}\\ \boldsymbol{\zeta},\eta}}{max} \quad -\gamma \sum_{i\in\mathcal{N}} \frac{d_i}{\zeta_i} - \frac{1}{\eta} \tag{2.10a}$$

$$\text{s.t.} \quad \zeta_i \sum_{j\in\mathcal{M}} p_{ij} \leq \sum_{j\in\mathcal{M}} t_{ij} \tag{2.10b}$$

$$t_{ij} \leq \alpha_i B \log\left(1 + \frac{p_{ij}h_{ij}}{\alpha_i B N_0}\right) \tag{2.10c}$$

$$\frac{d_i c_i}{f_i} + \frac{d_i}{\sum_{j\in\mathcal{M}} t_{ij}} \leq \bar{L}_i \tag{2.10d}$$

$$\sum_{i\in\mathcal{N}} \Big(\beta(v_i\mu + \sum_{j\in\mathcal{M}} u_{ij}\mu_j)\Big) \leq \frac{1}{\eta} \tag{2.10e}$$

$$(2.7\text{d})\text{-}(2.7\text{f}),\ (2.8),\ (2.9\text{b}),\ (2.9\text{c}). \tag{2.10f}$$

The equivalence between (2.7) and (2.10) is proved in Appendix A.1. By observing (2.10), we note two important properties. First, when each of the term $\zeta_i, \forall i \in \mathcal{N}$, and η increases within its feasible domain, the objective function of (2.10) achieves a higher value. Second, when we fix the value of ζ_i, and η, (2.10) becomes a feasibility checking optimization problem of finding the solution $\{\boldsymbol{p}, \boldsymbol{\alpha}, \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{f}, \boldsymbol{u}, \boldsymbol{v}, \boldsymbol{t}\}$ that satisfies (2.10b)–(2.10f). Thus, we can employ the concept of monotonic optimization to customize the BnB-based algorithm in order to optimally solve (2.7). To proceed, let us denote $\boldsymbol{\Omega} = [\boldsymbol{\zeta}, \eta]^T$ as the set of variables $\boldsymbol{\zeta}$ and η. By following the definitions and facts in [60, Section III-B, pp. 5577], we define $\mathcal{C} = \big\{\boldsymbol{\Omega} \in \mathbb{R}^{N+1} | (2.10\text{b}) - (2.10\text{f})\big\}$ as the normal compact set and $\mathcal{D} = \big[\underline{\boldsymbol{\Omega}}, \bar{\boldsymbol{\Omega}}\big]$ as the box that contains all the feasible solutions related to $\boldsymbol{\Omega}$ in (2.10). Obviously, the LB is given by $\underline{\boldsymbol{\Omega}} = [\epsilon, \ldots, \epsilon]^T$, where ϵ can be chosen arbitrarily small, e.g., $\epsilon = 10^{-5}$. To compute the upper bound, we

can simply consider that

$$\zeta_i \leq \bar{\zeta}_i = \frac{B \log\left(1 + \frac{\max_{j \in \mathcal{M}}\{h_{ij}\}\bar{P}_i}{\epsilon B N_0}\right)}{P_i} \tag{2.11}$$

$$\eta \leq \bar{\eta} = \frac{1}{\beta(\mu + M \sum \mu_j)} \tag{2.12}$$

Thus, the upper bound is given by $\bar{\boldsymbol{\Omega}} = \left[\bar{\zeta}_1, \ldots, \bar{\zeta}_N, \bar{\eta}\right]^T$. Given the value of $\boldsymbol{\Omega} \in \mathcal{D}$, the problem of checking whether $\boldsymbol{\Omega} \in \mathcal{C}$ or not becomes the feasibility checking problem, which is given by

$$\text{find}\,\{\{\boldsymbol{p}, \boldsymbol{\alpha}, \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{f}, \boldsymbol{u}, \boldsymbol{v}\} | (2.10\text{b}) - (2.10\text{f})\} \tag{2.13}$$

Problem (2.13) is indeed a mixed-integer generalized convex feasibility checking problem since all the constraints (2.10b)-(2.10f) are convex with respect to the involved variables. Here, it is worth mentioning that (2.13) has the generalized convex characteristic due to the appearance of the generalized exponential cone constraint (2.10c). Thus, we remark that solving (2.13) in its current state is still challenging, due to the very high computation time resulting from solving problem (2.13) using a generalized convex solver such as FMINCON, in contrast to other standard convex programs such as the SOCP, which can be solved much more rapidly while achieving an accuracy of 99.99% [61]. Thus, we are motivated by the availability of the commercial solver MOSEK, which is capable of solving the MI-SOCP, to employ the conic approximation with controlled accuracy in [61] to rewrite constraint (2.10c) by a set of

second order cone inequalities as

$$\begin{aligned}
\kappa_{m+4}^{ij} &\leq \alpha_i + \frac{p_i h_{ij}}{BN_0} \\
\alpha_i + \kappa_1^{ij} &\geq \left\| \begin{bmatrix} \alpha_i - \kappa_1^{ij} & 2\alpha_i + \frac{t_{ij}}{B2^{m-1}} \end{bmatrix} \right\|_2 \\
\alpha_i + \kappa_2^{ij} &\geq \left\| \begin{bmatrix} \alpha_i - \kappa_2^{ij} & \frac{5\alpha_i}{3} + \frac{t_{ij}}{B2^{m}} \end{bmatrix} \right\|_2 \\
\alpha_i + \kappa_3^{ij} &\geq \left\| \begin{bmatrix} \alpha_i - \kappa_3^{ij} & 2\kappa_1^{ij} \end{bmatrix} \right\|_2 \\
\kappa_4^{ij} &\geq \kappa_2^{ij} + \frac{\kappa_3^{ij}}{24} + \frac{19\alpha_i}{72} \\
\alpha_i + \kappa_l^{ij} &\geq \left\| \begin{bmatrix} \alpha_i - \kappa_l^{ij} & 2\kappa_{l-1}^{ij} \end{bmatrix} \right\|_2 \quad \forall\ l \in \{5, ..., m+3\} \\
\alpha_i + \kappa_{m+4}^{ij} &\geq \left\| \begin{bmatrix} \alpha_i - \kappa_{m+4}^{ij} & 2\kappa_{m+3}^{ij} \end{bmatrix} \right\|_2
\end{aligned} \tag{2.14}$$

where $\kappa_m^{ij} \geq 0$ is a new slack variable and m is the parameter of the conic approximation technique, which can be chosen as $m = 4$ to attain the 99.99% accuracy as already noted. By replacing (2.14) into constraint (2.10c), we can rewrite (2.13) in a standard form of MI-SOCP and employ MOSEK to solve it. Towards this end, we conclusively state that solving for problem (2.10) can be interpreted as solving the following problem

$$\max \left\{ -\gamma \sum_{i \in \mathcal{N}} \frac{d_i}{\zeta_i} - \frac{1}{\eta} \middle| \boldsymbol{\Omega} \in \mathcal{C} \subset \mathcal{D} \right\} \tag{2.15}$$

To solve (2.15), we first check whether $\underline{\boldsymbol{\Omega}}$ is feasible or not. If feasible, we run the BnB-based algorithm that recursively branches the box $\mathcal{D}$ into smaller boxes, checks the feasibility of each new box, updates the new upper and lower bounds by the

Box Reduction—Bound Computation process, and disposes the boxes that do not contain the optimal solution. According to [62], the BnB algorithm exhaustively searches for all possible solutions and terminates after many iterations when the difference between the upper and lower bounds is arbitrary small and the global optimal solution is determined. The pseudocode of the proposed BnB algorithm is outlined in Algorithm 1, where we denote by ζ_n and $\mathcal{D}_n$ as the current best objective and the collection of all created boxes at iteration n, respectively. In the following, we provide brief descriptions of the protocol of Box Branching, Box Reduction, and Pruning and Bound Computation, noting that more details are presented in [60, Section III-B]:

- **Box Branching:** At each iteration, one of the boxes that achieves the maximum upper bound is selected to branch, and then is divided into two smaller boxes using a partition rule. We adopt here the commonly used rule of bisecting along the longest edge [62], which is proven to be exhaustive.

- **Box Reduction:** The purpose is to exclude certain portions of the obtained boxes that are of no more interest without loss of optimality. As noted in [60, Section III-B], a box $\mathcal{B} = [\boldsymbol{q}, \boldsymbol{s}]$ can be reduced only when $\boldsymbol{q} \in \mathcal{C}$ and $\boldsymbol{s} \in \mathcal{D} \setminus \mathcal{C}$, by using the method outlined in [62] to find a smaller box $\mathcal{B}' = [\boldsymbol{q'}, \boldsymbol{s'}]$, noting that if an optimal solution is contained in $\mathcal{B} \cap \mathcal{C}$, then it is guaranteed to be contained in $\mathcal{B}' \cap \mathcal{C}$ when $\mathcal{B}'$ is a valid reduction [62].

- **Bounding and Pruning:** For each of the obtained reduced boxes denoted as $\mathcal{B}' = [\boldsymbol{q'}, \boldsymbol{s'}]$, due to the monotonic increase of the objective, the lower

Algorithm 1 Proposed BnB algorithm.

Apply box reduction to $\mathcal{D}$ to obtain $\text{red}(\mathcal{D})$
$n = 1$; $\mathcal{B}_1 = \text{red}(\mathcal{D})$; $\mathcal{D}_1 = \{\mathcal{B}_1\}$;$\zeta_1 = LB(\mathcal{B}_1)$;
repeat
    Select a box to branch: $\mathcal{B}_n = \arg\max_{\mathcal{B}_i \subset \mathcal{D}_n} UB(\mathcal{B}_i)$;
    Branch $\mathcal{B}_n$ into two smaller boxes $\mathcal{B}_n^{(1)}$ and $\mathcal{B}_n^{(2)}$;
    for $j = 1:2$ **do**
        Compute LB set of $\mathcal{B}_n^{(j)}$, as $\underline{\mathrm{X}}_n^{(j)} = \left\{\underline{\boldsymbol{s}}_n^{(j)}\right\}$;
        if $\underline{\mathrm{X}}_n^{(j)}$ is feasible **then**
            Apply box reduction to $\mathcal{B}_n^{(j)}$ to obtain $\text{red}\left(\mathcal{B}_n^{(j)}\right)$;
            else $\underline{\mathrm{X}}_n^{(j)} = \emptyset$;
        Compute $LB\left(\text{red}\left(\mathcal{B}_n^{(j)}\right)\right)$ and $UB\left(\text{red}\left(\mathcal{B}_n^{(j)}\right)\right)$;
    Update the current best objective: $\zeta_{n+1} =$
$\max\left(LB\left(\text{red}\left(\mathcal{B}_n^{(1)}\right)\right), LB\left(\text{red}\left(\mathcal{B}_n^{(2)}\right)\right), \zeta_n\right)$;
    Update the set of boxes: $\mathcal{D}_{n+1} = \left\{\mathcal{D}_n, \mathcal{B}_n^{(1)}, \mathcal{B}_n^{(2)}\right\}$;
    Delete boxes that do not contain optimal solution:
$\mathcal{D}_{n+1} = \mathcal{D}_n \setminus \{\mathcal{B}_i | \zeta_{n+1} > UB(\text{red}(\mathcal{B}_i)), \forall i\}$;
    n=n+1;
until $|\max_{\mathcal{B}_i \subset \mathcal{D}_n} UB(\text{red}(\mathcal{B}_i)) - \zeta_n| \leq \epsilon$;

and upper bounds are computed by evaluating the objective at $\boldsymbol{q'}$ and $\boldsymbol{s'}$, respectively. Then, after computing the bounds for all boxes, the objective ζ_n is set as the maximum obtained lower bound, and finally in the pruning step, the boxes having an upper bound smaller than ζ_n will be removed.

2.5 Proposed Iterative Low-Complexity Algorithms

2.5.1 SCA-based MI-SOCP Algorithm

In the previous section, a BnB-based exhaustive search algorithm is customized to find a global solution of the mixed-integer non-convex problem (2.7). However, the complexity of this approach quickly becomes excessive when the problem size grows, i.e. the number of UEs and APs, since it is well-known that the exhaustive search algorithm often has an exponential computational complexity. Due to its in-applicability in practice, in this section, we propose to approach a solution of (2.7) with more pragmatic, efficient, and lower computational complexity algorithms. To achieve this goal, let us start by reusing the equivalent transformation based on the big-M method, c.f. (2.8), which results in an equivalent problem (2.9). The motivation for this step can be explained similar to Section 2.4. Then, with a slightly different introduction of the slack variables $\theta_i \geq 0, \forall i \in \mathcal{M}$, $\tau \geq 0$, and $t_{ij} \geq 0, \forall i \in \mathcal{M}, \forall j \in \mathcal{N}$, we can also equivalently rewrite (2.9) into the following

problem, denoted as $\mathcal{P}_2$:

$$\mathcal{P}_{3a}: \min_{\substack{\boldsymbol{p},\boldsymbol{\alpha},\boldsymbol{x},\\ \boldsymbol{y},\boldsymbol{f},\boldsymbol{u},\boldsymbol{v},\\ \boldsymbol{t},\boldsymbol{\theta},\tau}} \quad \sum_{i\in\mathcal{N}} (\gamma d_i \theta_i) + \tau \tag{2.16a}$$

$$\text{s.t.} \quad \sum_{j\in\mathcal{M}} p_{ij} \leq \theta_i \sum_{j\in\mathcal{M}} t_{ij} \tag{2.16b}$$

$$t_{ij} \leq \alpha_i B \log\left(1 + \frac{p_{ij} h_{ij}}{\alpha_i B N_0}\right) \tag{2.16c}$$

$$\frac{d_i c_i}{f_i} + \frac{d_i}{\sum_{j\in\mathcal{M}} t_{ij}} \leq \bar{L}_i \tag{2.16d}$$

$$\sum_{i\in\mathcal{N}} \Big(\beta(v_i \mu + \sum_{j\in\mathcal{M}} u_{ij}\mu_j)\Big) \leq \tau \tag{2.16e}$$

$$(2.7\text{d})\text{-}(2.7\text{f}),\ (2.8),\ (2.9\text{b}),\ (2.9\text{c}). \tag{2.16f}$$

where we denote $\boldsymbol{\theta} = \{\theta_i \geq 0, \forall i \in \mathcal{M}\}$. At this point, we observe that the underlying issues which make (2.16) difficult, are due to the existence of non-convex constraint (2.16b) and binary constraints (2.7f). Another subtle point, as analyzed in (2.14), is that (2.16c) appears in the form of generalized exponential cone. In the following, we will invoke the SCA-based framework to approximate (2.16) into a series of approximated MI-SOCP problems, where a modern dedicated solver such as MOSEK is available to solve efficiently. Then we develop an SCA-based MI-SOCP algorithm to solve for its solution.

Let us first concentrate on the non-convex constraint (2.16b). By quickly investigating (2.16b), we realize that the factor which causes the non-convexity of (2.16b) is because of the non-convex non-concave function $\theta_i \sum_{j\in\mathcal{M}} t_{ij}$ with respect to variables

θ_i and t_{ij}. By simple algebraic manipulations, we can easily rewrite (2.16b) as

$$\sum_{j\in\mathcal{M}} p_{ij} + 0.25\left(\theta_i - \sum_{j\in\mathcal{M}} t_{ij}\right)^2 - 0.25\left(\theta_i + \sum_{j\in\mathcal{M}} t_{ij}\right)^2 \leq 0 \tag{2.17}$$

Here, it appears that the above inequality contains a difference of convex (D.C.) functions with respect to all variables, which makes (2.17) non-convex. To handle this obstacle, we are motivated by the inner-approximation method in [63] to approximate function

$$g_i(\boldsymbol{\theta}, \boldsymbol{t}) = -0.25\left(\theta_i + \sum_{j\in\mathcal{M}} t_{ij}\right)^2 \tag{2.18}$$

by its upper-bounded convex function $G_i(\boldsymbol{\theta}, \boldsymbol{t}; \boldsymbol{\theta}^{(n)}, \boldsymbol{t}^{(n)})$ around the points $\boldsymbol{\theta}^{(n)}, \boldsymbol{t}^{(n)}$ as

$$\begin{aligned} G_i(\boldsymbol{\theta}, \boldsymbol{t}; \boldsymbol{\theta}^{(n)}, \boldsymbol{t}^{(n)}) = &-0.25\left(\theta_i^{(n)} + \sum_{j\in\mathcal{M}} t_{ij}^{(n)}\right)^2 \\ &- 0.5\left(\theta_i^{(n)} + \sum_{j\in\mathcal{M}} t_{ij}^{(n)}\right)\left(\theta_i - \theta_i^{(n)} + \sum_{j\in\mathcal{M}} t_{ij} - \sum_{j\in\mathcal{M}} t_{ij}^{(n)}\right) \end{aligned} \tag{2.19}$$

By replacing $g_i(\boldsymbol{\theta}, \boldsymbol{t})$ with $G_i(\boldsymbol{\theta}, \boldsymbol{t}; \boldsymbol{\theta}^{(n)}, \boldsymbol{t}^{(n)})$, we remark that all constraints of (2.16) except for (2.16c), can be converted into the conic form. In fact in (2.16d), we can easily introduce some slack variables f_i^{inv} and t_i^{inv} and equivalently rewrite (2.16d) in

a set of conic constraints as

$$f_i^{\text{inv}} + t_i^{\text{inv}} \leq \bar{L}_i \tag{2.20a}$$

$$\|[\sqrt{d_i c_i} \quad (f_i^{\text{inv}} - f_i)/2]\|^2 \leq (f_i^{\text{inv}} + f_i)/2 \tag{2.20b}$$

$$\|[\sqrt{d_i} \quad (t_i^{\text{inv}} - \sum t_{ij})/2]\|^2 \leq (t_i^{\text{inv}} + \sum t_{ij})/2 \tag{2.20c}$$

However, the resulting problem from this SCA-based method is a generalized convex mixed-integer program for which dedicated solvers are very limited. Besides, (2.16c) is indeed a convex constraint, so that applying an SCA-based method here is not necessary, due to its preserved convexity. Towards this end, we reuse the established derivation of conic approximation with controlled accuracy to represent (2.16c) by a set of conic constraints as in (2.14).

While recognizing that the conic approximation with controlled accuracy, as presented in (2.14) is available, we remark that this approximation is quite complicated due to the introduction of numerous slack variables and conic constraints in order to obtain high accuracy. This can be a burden when the problem size grows higher. Therefore, we propose a novel and more efficient approach to represent (2.16c) as a conic constraint. The following proposition, which derives the lower-bound concave approximate of the generic function $\alpha_i \log\left(1 + \frac{A_{ij}p_{ij}}{\alpha_i}\right)$, where $A_{ij} = h_{ij}/(BN_0)$, is in order

Proposition 1. *The function* $\ell(\alpha_i, p_{ij}) = \alpha_i \log\left(1 + \frac{A_{ij}p_{ij}}{\alpha_i}\right), \forall \alpha_i, p_{ij} \in \mathbb{R}^+$ *can be approximated by its lower-bounded quadratic function* $L(\alpha_i, p_{ij}; \alpha_i^{(n)}, p_{ij}^{(n)})$, *which is*

given by

$$\ell(\alpha_i, p_{ij}) \geq L(\alpha_i, p_{ij}; \alpha_i^{(n)}, p_{ij}^{(n)}) = \alpha_i^{(n)} \log\left(1 + \frac{A_{ij} p_{ij}}{\alpha_i}\right) + \left[\log\left(1 + \frac{A_{ij} p_{ij}^{(n)}}{\alpha_i^{(n)}}\right) - \frac{A_{ij} p_{ij}^{(n)}}{\alpha_i^{(n)} + A_{ij}}\right]\left(\alpha_i - \alpha_i^{(n)}\right) + \frac{\alpha_i^{(n)}(p_{ij} - p_{ij}^{(n)})}{1 + A_{ij} p_{ij}^{(n)}/\alpha_i^{(n)}} - \frac{L}{2}\left(p_{ij} - p_{ij}^{(n)} + \alpha_i - \alpha_i^{(n)}\right)^2 \quad (2.21)$$

where L is the Lipschitz constant of $\nabla \ell(\alpha_i, p_{ij})$.

By employing all approximations and representations from (2.14), (2.19), and (2.20), an approximated MI-SOCP of the mixed-integer non-convex problem (2.16), denoted by $\tilde{\mathcal{P}}^{(n)}$, can be formulated at the nth iteration as

$$\mathcal{P}_{3b}: \min_{\substack{\boldsymbol{p},\boldsymbol{\alpha},\boldsymbol{x},\\ \boldsymbol{y},\boldsymbol{f},\boldsymbol{f}^{\text{inv}},\boldsymbol{u},\boldsymbol{v},\\ \boldsymbol{\kappa},\boldsymbol{t},\boldsymbol{t}^{\text{inv}},\boldsymbol{\theta},\tau}} \quad \sum_{i\in\mathcal{N}} (\gamma d_i \theta_i) + \tau \quad (2.22a)$$

$$\text{s.t.} \quad \sum_{j\in\mathcal{M}} p_{ij} + 0.25\left(\theta_i - \sum_{j\in\mathcal{M}} t_{ij}\right)^2 + G_i(\boldsymbol{\theta}, \boldsymbol{t}; \boldsymbol{\theta}^{(n)}, \boldsymbol{t}^{(n)}) \leq 0 \quad (2.22b)$$

$$\text{(2.7d)-(2.7f), (2.8), (2.9b), (2.9c), (2.14), (2.16e), (2.20).} \quad (2.22c)$$

where we denote $\boldsymbol{f}^{\text{inv}} = \{f_i^{\text{inv}} \geq 0, \forall i \in \mathcal{N}\}$, $\boldsymbol{t}^{\text{inv}} = \{t_i^{\text{inv}} \geq 0, \forall i \in \mathcal{N}\}$. The pseudocode to solve problem $\mathcal{P}_2$ is given in Algorithm 2.

Convergence Analysis: The convergence of Algorithm 2 can be guaranteed by showing that the series of resulting objective is monotonically convergent. Let $\Gamma^{(n)}$

Algorithm 2 SCA-based MI-SOCP Algorithm.

1: **Initialize:**
2: $n = 0$;
3: Choose an initial point $\boldsymbol{\theta}^{(n)}, \boldsymbol{t}^{(n)}$;
4: **repeat**
5: Solve $\tilde{\mathcal{P}}^{(n)}$ to obtain the optimal solution at the nth iteration $\Omega^\star = \{\boldsymbol{p}^\star, \boldsymbol{\alpha}^\star, \boldsymbol{x}^\star, \boldsymbol{y}^\star, \boldsymbol{f}^\star, \boldsymbol{f}^{\text{inv}\star}, \boldsymbol{u}^\star, \boldsymbol{v}^\star, \boldsymbol{\kappa}^\star, \boldsymbol{t}^\star, \boldsymbol{t}^{\text{inv}\star}, \boldsymbol{\theta}^\star, \tau^\star\}$
6: Update $\boldsymbol{\theta}^{(n)} = \boldsymbol{\theta}^\star, \boldsymbol{t}^{(n)} = \boldsymbol{t}^\star$;
7: $n = n + 1$;
8: **until** Convergence of the objective of $\tilde{\mathcal{P}}^{(n)}$.

denote the optimal objective value and $\Omega^{(n)}$ denote the optimal solution set at the nth iteration of Algorithm 2. Due to the convex approximation in (2.19), the updating rules in Algorithm 2, c.f., Step 6, ensure that the solution set $\Omega^{(n)}$ is a feasible solution to problem $\mathcal{P}_2$ at step $n + 1$. This subsequently leads to the results of $\Gamma^{(n+1)} \leq \Gamma^{(n)}$, which means that Algorithm 2 generates a non-increasing sequence of objective function values. Due to the latency constraints, the sequence of $\Gamma^{(n)}, n = 1, 2, \ldots$ is bounded below and therefore, Algorithm 2 guarantees that the objective converges.

2.5.2 SCA-based SOCP Algorithm using Continuous Relaxation and Post-Processing Protocol

Although we can employ the modern solver MOSEK to iteratively solve a sequence of $\tilde{\mathcal{P}}^{(n)}$ for a sub-optimal solution of (2.16), it is important to mention that when the problem size grows larger, e.g. $N \geq 10$ and $M \geq 4$, the time consumed to solve one iteration of Algorithm 2 is extremely high. This is because the solver's capability is also limited to handle a large number of binary variables. To develop

a more appealing approach, we further consider the continuous relaxation of binary variables, e.g., $0 \leq x_{ij} \leq 1$ and $0 \leq y_i \leq 1, \forall i, j$. Consequently, the continuous relaxed version of (2.22), denoted as $\tilde{\mathcal{P}}_r^{(n)}$, becomes a SOCP (continuously convex) and can be solved within a polynomial time.

The iterative SCA-based SOCP algorithm under this continuous relaxation contains two steps: solving the relaxed problem and post-processing procedure. In the first stage, we repeat similar steps in Algorithm 2, but replace $\tilde{\mathcal{P}}^{(n)}$ by $\tilde{\mathcal{P}}_r^{(n)}$ to solve for a solution. The post-processing protocol is then employed to refine the obtained continuous values of x_{ij} and y_i into the binary state, which is feasible to (2.16). Here, we are inspired by the Inflation Algorithm in [64] to execute the rounding task. In particular, we consider the solution of the continuous relaxed problem $\tilde{\mathcal{P}}_r^{(n)}$ at convergence as an incentive measure to sequentially decide which binary value x_{ij} and y_i should take. We denote $\tilde{\boldsymbol{x}}$ and $\tilde{\boldsymbol{y}}$ as the solution achieved after the first stage. Intuitively, the achieved value of $\tilde{x}_{ij}$ is higher if the channel condition h_{ij} between the ith UE and the jth AP is better and the cost to offload through that link is smaller than the others. Based on this observation, we propose an iterative procedure to determine the set of binary variables based on the values of $\tilde{x}_{ij}$ and $\tilde{y}_i$. Initially, we assume there is no connection to the central cloudlet and there is no association between UEs and APs. The UE-AP association with the largest $\tilde{x}_{ij}$ will be set to 1 and the corresponding $1 - \tilde{y}_i$ will be rounded to 1 accordingly. The overall algorithm is given in Algorithm 3, where $\pi^{(m)}$ represents the obj value obtained at iteration m, and $\mathcal{R}_{\text{off}}^{(m)}$ is a set containing (i, j) pairs $\in (\mathcal{N}, \mathcal{M})$ at iteration m.

Algorithm 3 Relax-continuous and inflation based algorithm

1: Set $m := 0$, $\pi^{(m)}$ is significantly small, and initialize the sets $\mathcal{R}_{\text{off}}^{(m)} = \{(i,j) \in (\mathcal{N}, \mathcal{M})\}, \mathcal{U}_{\text{off}}^{(m)} = \{i \in \mathcal{N}\}$.
2: **repeat**
3: Set $m := m + 1$;
4: Solve $\tilde{\mathcal{P}}_r^{(n)}$ until convergence with $x_{i',j'} = 1, \forall \{(i',j')\} \notin \mathcal{R}_{\text{off}}^{(m-1)}$, and $y_{i'} = 0, \forall \{i'\} \notin \mathcal{U}_{\text{off}}^{(m-1)}$;
5: Update $\mathcal{R}_{\text{off}}^{(m)} = \mathcal{R}_{\text{off}}^{(m-1)} \setminus \left\{(i',j') \times i' = \arg\max_{i,j \in \mathcal{R}_{\text{off}}^{(m-1)}} \widetilde{x}_{i,j}\right\}, \mathcal{U}_{\text{off}}^{(m)} = \mathcal{U}_{\text{off}}^{(m-1)} \setminus \left\{i' = \arg\min_{i \in \mathcal{U}_{\text{off}}^{(m-1)}} \widetilde{y}_i\right\}$;
6: Solve $\tilde{\mathcal{P}}^{(n)}$ until convergence given $x_{i',j'} = 1, \forall \{(i',j')\} \notin \mathcal{R}_{\text{off}}^{(m)}$ and $y_{i'} = 0, \forall \{i'\} \notin \mathcal{U}_{\text{off}}^{(m)}$ and $x_{i,j} = 0, \forall \{(i,j)\} \in \mathcal{R}_{\text{off}}^{(m)}$ and $y_i = 1, \forall \{i\} \in \mathcal{U}_{\text{off}}^{(m)}$, denoted as $(\mathcal{P}^{int})$. If $(\mathcal{P}^{int})$ is feasible, set $\pi^{(m)}$ as the value of objective function achieved at the convergence. If not, set $\pi^{(m)} = \pi^{(0)}$.
7: **until** $\tilde{\mathcal{P}}_r^{(n)}$ starts to be infeasible or $(\mathcal{P}^{int})$ is feasible and $\pi^{(m)} > \pi^{(m-1)}$;
8: Solve $\tilde{\mathcal{P}}^{(n)}$ given $x_{i',j'} = 1, \forall \{(i',j')\} \notin \mathcal{R}_{\text{off}}^{(m-1)}$ and $y_{i'} = 0, \forall \{i'\} \notin \mathcal{U}_{\text{off}}^{(m-1)}$ and $x_{i,j} = 0, \forall \{(i,j)\} \in \mathcal{R}_{\text{off}}^{(m-1)}$ and $y_i = 1, \forall \{i\} \in \mathcal{U}_{\text{off}}^{(m-1)}$ to obtain $\boldsymbol{p}^\star, \boldsymbol{\alpha}^\star, \boldsymbol{x}^\star, \boldsymbol{y}^\star, \boldsymbol{f}^\star$;

2.5.3 Complexity Analysis

In this subsection, we discuss the complexity of our proposed algorithms:

1. **BnB algorithm:** The complexity here is extremely high since the number of boxes to be considered increases exponentially with the problem dimension. In addition, an MI-SOCP feasibility problem is solved in each iteration, which increases the complexity even more.

2. **SCA-based MI-SOCP Algorithm:** The overall complexity here depends mainly on that of solving the MI-SOCP problem in (2.22) which is indeed a combinatorial optimization problem. Specifically, there are NM binary variables x_{ij} and N binary variables y_i, resulting in 2^{NM+N} combinations for all

the binary variables. Given fixed values for $\boldsymbol{x}$ and $\boldsymbol{y}$, the constraints in problem (2.22) approximately consist of a total number of $(m+17)NM+38N+M+1$ variables. Thus, in each iteration, the worst-case complexity of Algorithm 2 can be written as $\mathcal{O}(2^{NM+N}(mNM))$.

3. **Relax-continuous and inflation based algorithm:** We first note that in the worst case, Algorithm 3 must iteratively solve and update the resulting parameters for the SOCP problem $(\tilde{\mathcal{P}}_r^{(n)})$ and $(\mathcal{P}^{int})$ for $(N-1)$ times. In each step, the complexity of solving $(\tilde{\mathcal{P}}_r^{(n)})$ and $(\mathcal{P}^{int})$ is approximately $\mathcal{O}(mNM)$, which is a polynomial time complexity as is the case for SOCP problems [65]. Thus, the overall complexity for Algorithm 3 is $\mathcal{O}(2(N-1)(mNM))$.

2.6 Numerical Results

In this section, we present numerical results based on simulations following different scenarios. For the majority of our simulations, we consider a small two-tier hierarchical MEC system with a set of $M = 3$ cloudlet-enabled APs connected to a central cloudlet and a set of $N = 5$ UEs with tasks to offload. Simulation parameters are presented in Table 2.2. The cycles/bit values is chosen such as to support generic use cases that can have an exact task computational demand around the used value, such as in [45, 47]. Note that d_i is also selected to support a broad range of applications, knowing that our solution can easily address larger tasks such as in [44]. Also, the change in the ratio between μ and μ_j will be explored later in this section, since it is the important factor as discussed in Subsection 2.3.5. Also, each channel gain h_{ij}

Notation	Description	Value
$\mathcal{N}$	Set of UEs	5
$\mathcal{M}$	Set of APs	3
d_i	Task input size	200 Kb
c_i	Task computational demand	600 CPU cycles/bit
P_i	UE's maximum power budget	30 dBm [44]
μ_j	Edge cloudlet resource unit cost	1 cent/gigahertz
μ	Central cloudlet resource unit cost	5 cent/gigahertz
B	Radio spectrum bandwidth	10 MHz [47]
N_0	Noise power spectral density	-120 dBm/MHz [44]

Table 2.2: Simulation Parameters

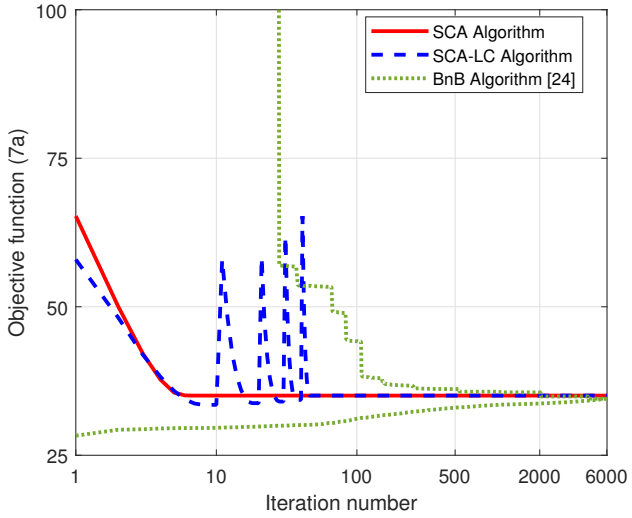


Figure 2.3: Convergence behavior between the BnB and SCA-based algorithms.

is independently randomly generated according to an exponential distribution with mean of 1, and the effect of path loss is omitted for simplicity. The objective weights were evenly selected with $\gamma = \beta = 0.5$. The convergence criteria of Algorithm 2 is established when ϵ, i.e. the difference of objective value between $\Gamma^{(n)}$ and $\Gamma^{(n+1)}$ of the approximated problem, is $\epsilon \leq 10^{-3}$.

In Fig. 2.3, we show the convergence performance for Algorithms 1, 2, and 3, where we consider a slightly smaller instance with $M = 3$ APs and $N = 4$ UEs, for the purpose of meeting the low-scalability of the BnB algorithm. As it can be seen, the SCA-based MI-SOCP algorithm (Algorithm 2) just takes a few iterations to stabilize. However, its overall runtime is high since the problem in each SCA iteration is a MI-SOCP. On the other hand, the SCA-based SOCP iterative algorithm (Algorithm 3) requires more iterations to converge, but with a much lower computation time since the per-iteration problem is a relaxed SOCP, which can be solved with low computational effort. We note the downhill and uphill effect of the SOCP iterative algorithm in the figure, which is caused by the convergence of each SOCP SCA iteration during the inflation process. We also observe the lower bound obtained by the SOCP iterative algorithm, but eventually it achieves almost the same objective as the MI-SOCP algorithm due to the small instance size used. In general, the MI-SOCP algorithm will achieve a better objective due to its superior approximation, while the SOCP iterative algorithm will be able to solve bigger instances, as will be later demonstrated. On the other hand, the BnB algorithm (Algorithm 1) was the slowest and obtained the optimal solution after few thousand iterations. The BnB algorithm has two curves for showing the progression of the lower bound starting from a very small value, and the upper bound starting from a very high value, until they coincide upon convergence. As it can be seen, the BnB algorithm outperformed the other algorithms in terms of the objective due to its non-approximate nature, while the low-complexity SCA algorithms outperformed the BnB algorithm in terms of the running time due to their scalability advantage.

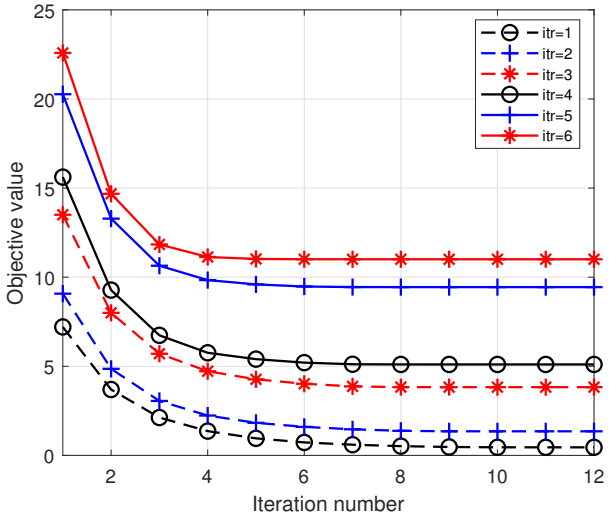


Figure 2.4: Convergence behavior for the SCA-based SOCP iterative algorithm.

In Fig. 2.4, we study the convergence behavior of the SCA-based SOCP iterative algorithm apart, for a latency threshold $\bar{L}_i = 100$ ms. In the figure, each line represents the convergence for a specific iteration of the algorithm. It can be seen that every next iteration achieves a higher objective due to the post-processing phase which rounds variables and fix them in their binary state. The first few iterations give a lower bound on the solution achieved by the SCA-based MI-SOCP algorithm, until ending with a solution which is an upper bound and is almost equal to the solution obtained by that algorithm.

In Fig. 2.5, we use the SCA-based MI-SOCP algorithm to study the variation of the objective value with respect to the UEs' latency threshold and the cloudlets computational capacity. Two observations can be made. First, increasing the latency threshold always leads to a decrease in the objective value. Second, the objective decreases even more whenever the edge cloudlets have higher resources' capacity.

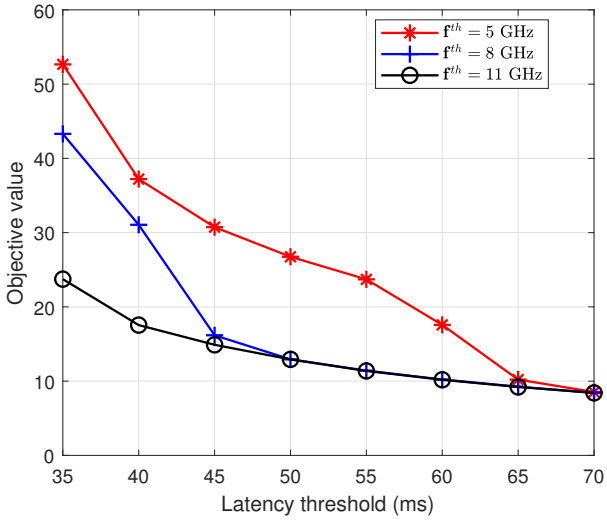


Figure 2.5: Obj value vs UEs' latency threshold and cloudlets' computational capacity.

This is visible through the three plots, where each one represents a given resources' capacity. Note here that the objective value represents the sum of the energy (in Mj) and the cost sub-objectives. The first observation can be explained by the fact that when the latency threshold is small, UEs have to consume more server resources in order to lower their task execution time to keep up with the new threshold, which will incur higher costs. In addition to that, more tasks may end up using the central cloudlet since edge cloudlets become more demanded and overloaded, which again comes at a higher cost due to the central cloudlet's higher computational cost. On the other hand, increasing the latency threshold will allow users to utilize less resources and hence lower their total cost. The second observation can be explained by the fact that when edge cloudlets have more computational capacity, more space is created for tasks execution with lower costs (as compared to the central cloudlet cost), which

lowers the need to migrate tasks to the central cloudlet. On the other hand, low resource capacities will rapidly congest the edge servers, and will force more tasks to be further migrated to the central cloudlet, which will incur a high execution cost and a higher objective value. It can be seen that for high latency ranges, the difference between the curves gets smaller, and sometimes they overlap. This is because when latency thresholds are high, adding more edge resources becomes less necessary and effective since fewer computational resources are needed, and hence it will have a lesser (and sometimes negligible) effect on the objective value. We conclude that NOs should increase the capacity of the edge cloudlets when possible, especially when users have stringent latency requirements, so that the central cloudlet is less utilized and the costs are decreased. On the other hand, for slices with loose latency requirements for their users, adding more edge resources may be unnecessary and may not decrease the costs as needed.

As presented in Fig. 2.6, we compare our SCA-based MI-SOCP algorithm in which the allocation of bandwidth is optimized (SCA-OB), with another variation of the same algorithm (SCA-EB) where bandwidth is equally allocated among UEs. We show this comparison for different latency threshold values $\bar{L}_i$. We use the same instance after varying UEs' data size randomly within the range of [200, 600] Kb. It can be seen that our SCA-based MI-SOCP algorithm always achieves a better objective regardless of the latency threshold, and that equally allocating the bandwidth will always lead to an inferior solution. Also, the gap decreases for higher latency thresholds. First, the reason for the superiority of SCA-OB is that each UE has a different task size, and hence will need a specific bandwidth chunk to make sure

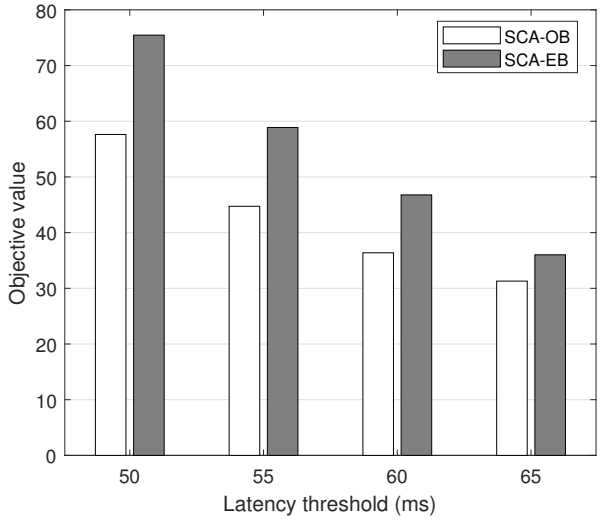


Figure 2.6: Optimized vs equal bandwidth allocation with respect to latency threshold.

its objective is minimized, which means that an equal allocation of the bandwidth in this case will lead to a sub-optimal solution. For instance, allocating an average channel bandwidth to a UE with a large task will not be enough and will achieve a higher objective due to the resulting increase in the upload latency relative to the large data size, which will in turn affect the energy consumption. We conclude that optimizing the bandwidth allocation along with the users' association and power transmission decisions is crucial for obtaining a high quality solution with a minimal energy, and omitting this optimization aspect will result in a sub-optimal solution.

In Fig. 2.7, we study the change in the objective value with respect to the change in the number of UEs in the network. We take advantage to compare the performance of the SCA-based MI-SOCP algorithm and that of the SOCP iterative algorithm in regard to this variation while also varying the latency threshold. Four observations

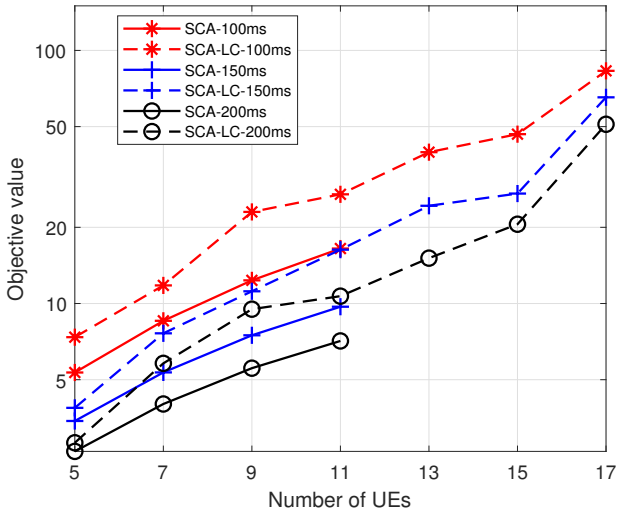


Figure 2.7: Objective vs number of UEs and their latency threshold.

can be made; first, the objective value always increases whenever the number of users grows in the network. Second, for the same latency threshold level, the MI-SOCP algorithm always outperforms the SOCP iterative algorithm regardless of the number of requests. Third, the SOCP iterative algorithm can better scale with the instance size and is able to solve the problem for instances that are beyond the MI-SOCP algorithm's capacity. Fourth, a lower objective is achieved for higher latency thresholds, as has been demonstrated earlier. The first observation is caused by the increase in the amount of consumed resources in the network in addition to the amount of energy consumed by the UEs. This increase also means more users are utilizing the central cloudlet, which incurs higher costs. The second observation is because of the MI-SOCP algorithm, which by nature achieves a better objective due to its ability of fully optimizing the solution containing binary variables. The third observation indicates the scalability advantage of the SOCP iterative algorithm that

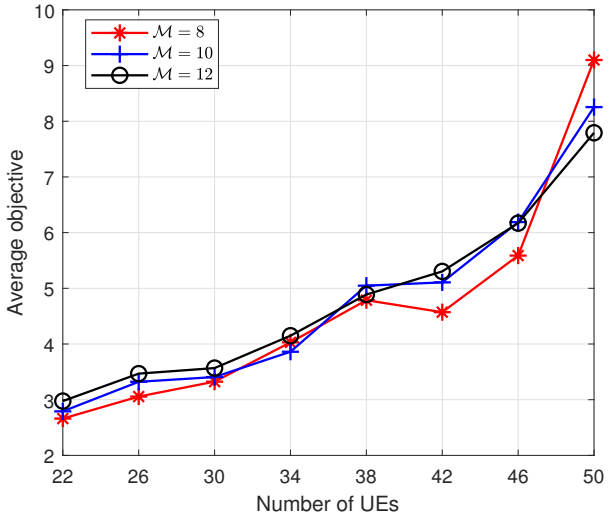
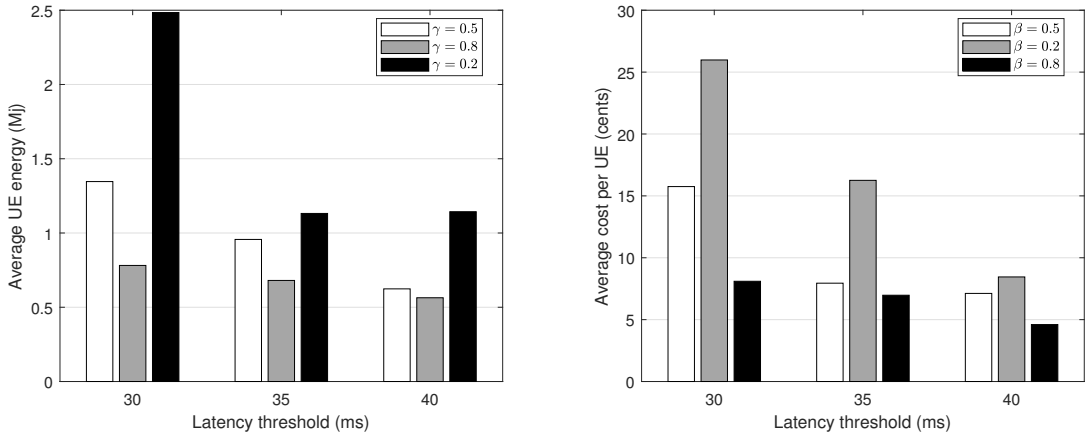


Figure 2.8: Variation of the average objective per UE with respect to instance size.

is able to solve bigger instances and networks with more number of UEs due to the continuous relaxation performed in that algorithm. Network operators should consider increasing the capacity of the edge cloudlets especially for APs which have a higher load. Also, adding more edge cloudlets could prove useful in order to account for the high UEs' requests in dense areas, which would decrease the costs by lowering the high utilization of the central cloudlet.

In Fig. 2.8, we study the effect of change in instance size on the average objective obtained for relatively bigger instances, utilizing the SOCP iterative algorithm exclusively, due to its superior scalability. We specifically vary the number of UEs and APs, and show the average objective achieved as the problem size grows. As it can be seen, the average objective achieved per UE increases whenever the number of UEs rises in the network. This is due to the increased competition on the computation and communication resources whenever new UEs enter the system, which



2.9.a Variation of the average UE energy with respect to the weights change.

2.9.b Variation of the average cost per UE with respect to the weights change.

Figure 2.9: Trade-off between energy and cost with respect to the change in obj weights γ and β while varying the latency threshold.

translates into a higher cost and energy since more tasks will be migrated to the central cloudlet, in addition to the decrease in the bandwidth availability which forces UEs to increase their transmission power in order to keep up with the required latency. Few exceptions occur when the average objective decreases due to particular devices positioned close to certain APs enter the network, requiring low transmission energy and decreasing the overall average energy in the process. It can also be seen that only a small difference is made when the number of APs changes, which is due to the high number of APs and cloudlets that already offer too many options for the existing UEs, noting that in general a significant addition of APs will offer users more resources in the network. We note that the SOCP iterative algorithm is scalable to support much higher numbers of UEs and APs in the network, but with the cost of a decreased solution quality, as compared to the MI-SOCP algorithm.

Next, we tackle the effect of the objective weights on the obtained overall solution. So far, we have been presenting the change in the objective, which is the summation of the normalized energy and cost multiplied by their respective weights $\gamma = \beta = 0.5$. Now, as presented in Fig. 2.9, we show how the change in the weights γ and β affects the energy and cost. Specifically, we show the average UE energy consumption and cost as they vary with respect to the objective weights for different values of the latency threshold. We show the results for each of the energy and cost in a separate figure, namely 2.9.a and 2.9.b. As it can be seen, a trade-off exists between the energy consumption and cost. We observe that when $\gamma = 0.8$ and $\beta = 0.2$, the average UE energy consumption decreases significantly for all latency threshold values, but this is accompanied by an increase in the average cost. On the other side, when $\gamma = 0.2$ and $\beta = 0.8$, the average costs per UE decreases, but with a cost of an increase in the average UE energy consumption. These observations are due to the fact that when putting a higher weight on the energy sub-objective, there will be a decrease in the average energy consumption, since the solution will be driven to focus more on associating UEs to APs with good channels, so that transmission energy can be decreased. This, however, will drive the costs up since the APs with good channels are not necessarily the best in terms of the available computational resources and their cost. On the other hand, putting more weight on the cost sub-objective will prioritize associating UEs to edge cloudlets with low unit costs, and hence will incur lower overall costs, but this on the other hand will drive the UEs' energy consumption up since this association will not be best in terms of the wireless channel, and thus UEs will be forced to increase their transmission power in order to keep up with the

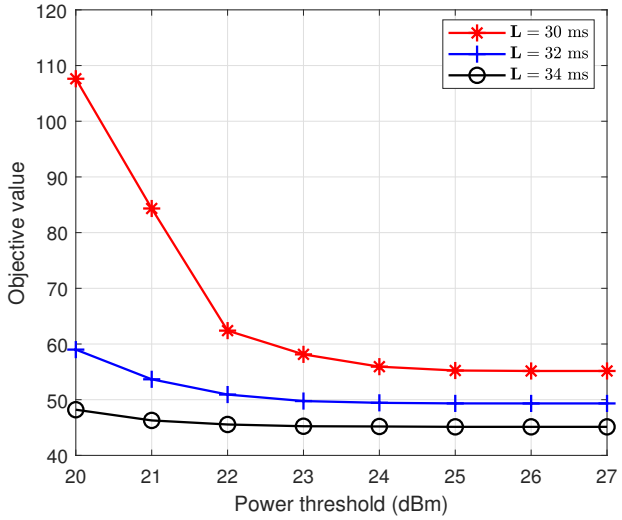


Figure 2.10: Objective value vs UEs' transmission power limit and latency threshold.

required latency, incurring a higher energy consumption in the process.

In Fig. 2.10, also by utilizing the SCA-based MI-SOCP algorithm, we show the variation of the objective value with respect to the power threshold. Here, we consider all UEs to have the same power limit in each run. It can be seen that increasing the power limit will decrease the objective. This is because when UEs are forced to utilize a low transmission power, the upload latency will increase, which in turn will increase their energy expenditure. We also observe that at one point, increasing the threshold becomes insignificant. This is because a power increase after a specific threshold will start to increase the energy consumption even though the upload latency becomes lower, and that is because the energy function in non-linear. It is important to note that for the sake of realizing an optimal energy consumption, transmission power have to be neither too low nor too high. Also, another reason for the higher objective is that limiting the transmission power could prevent UEs

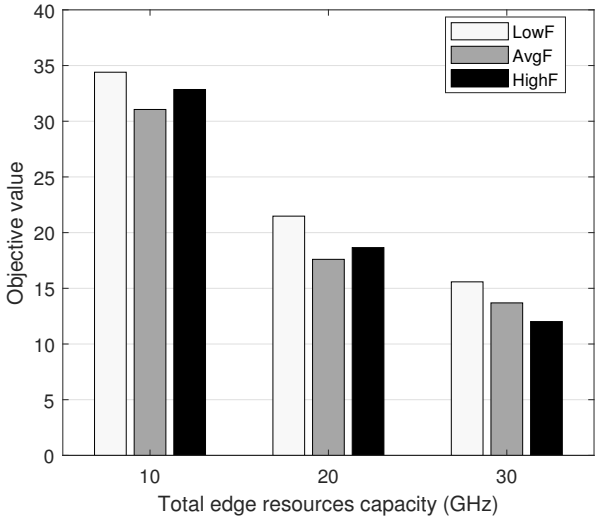


Figure 2.11: Comparing different approaches for distributing edge resources.

from utilizing APs with poorer channel state but which could provide more resources with lower costs. On the other hand, we observe that a lower objective is achieved when the latency threshold is increased, which is mainly due to the resulting decrease in computational costs as explained in Fig. 2.5. For that reason, we conclude that having a high transmission power threshold is always a better option and will not affect the objective negatively due to the energy being part of that objective which is being minimized.

We also studied a scenario where UEs are concentrated around one AP, e.g. AP with $i = 1$ (all UEs have their best channel with AP $i = 1$). We compared three approaches for distributing the available computational resources on the edge cloud. In LowF, scarce resources were given to AP 1 (10%); in AvgF, resources were evenly distributed among edge cloudlets; and in HighF, resources were concentrated mostly in AP 1 (80%). We compared the objective values resulting from these approaches

given by the SCA-based MI-SOCP algorithm and in three cases representing different values for the total edge cloudlet resources. The results of this study are given in Fig. 2.11. As it can be seen, giving scarce resources to the AP with high load will always lead to an inferior solution, and will be superseded by the other approaches. This is because the resources on AP $i = 1$ will not be enough for the surrounding UEs, and will be forced to either utilize another AP with a worse channel incurring more energy, or offload to the central cloudlet incurring a higher cost, where both cases translate into an increase in the objective. On the other hand, the other two approaches will offer more resources on AP 1 which will lower the objective value since both energy and cost can be decreased. An important insight can be realized here, which is that even though HighF may seem the best approach to follow, evenly allocating the resources will be a better approach in most cases. The reason is that most UEs prefer to utilize AP 1 which increases the demand on its resources and thus limits its resources availability and hence causes many UEs to further migrate their task to the central cloudlet. However, evenly distributing the resources gives a chance to the other UEs to utilize the other edge cloudlets and avoid the central cloudlet which would save costs. We conclude that evenly allocating resources among edge cloudlets appears to be the best approach to follow by the NOs.

In figure 2.12, we study the performance and scalability of the SOCP iterative algorithm while changing the number of UEs and APs using the same relatively big instance utilized in figure 2.8. As it can be seen, intuitively, the running time increases whenever the problem size grows, whether new UEs are introduced or new APs are added. However, the rate of change in the running time with respect to the

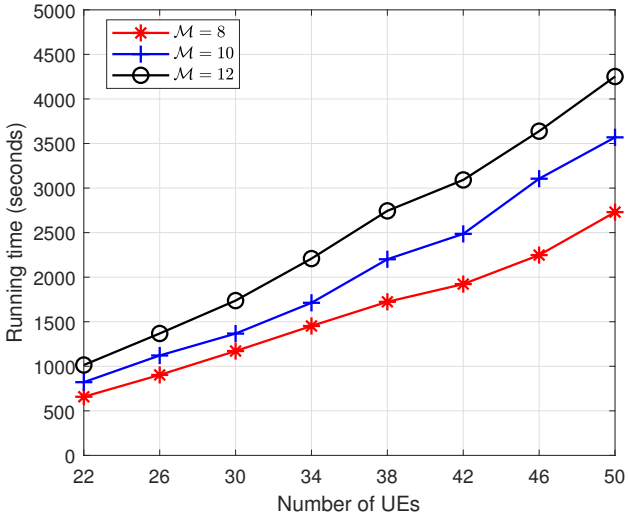


Figure 2.12: Runtime of the SOCP algorithm with respect to the instance size.

problem size is close to linear, in contrast to the exponential increase in the running time for the MI-SOCP and BnB algorithms, demonstrating the superior scalability of the SOCP iterative algorithm, but with a cost of a decrease in the solution quality as compared to the other algorithms. However, our SOCP algorithm still has some performance limitations in part due to its centralized nature, which is of utility for solving the offline problem, noting that a much lower runtime can be achieved when a fewer number of APs is considered by the NO. It is worth noting that this can also serve as a benchmark for other efficient methods.

We have discussed in Subsection 2.3.5 about the ratio a between the central cloudlet unit cost μ and the edge cloudlets unit cost μ_j in $\mu(a) = a \max(\mu_j, \forall j \in \mathcal{M})$ being the important aspect. In Fig. 2.13, we analyze the effect of varying the ratio a on the objective, where we consider $\mu_j = \mu_{j'} (j \neq j')\ \forall j, j' \in \mathcal{M}$ for simplicity, i.e. $\mu(a) = a\mu_j$. We observe an increase in the objective that is almost linear whenever

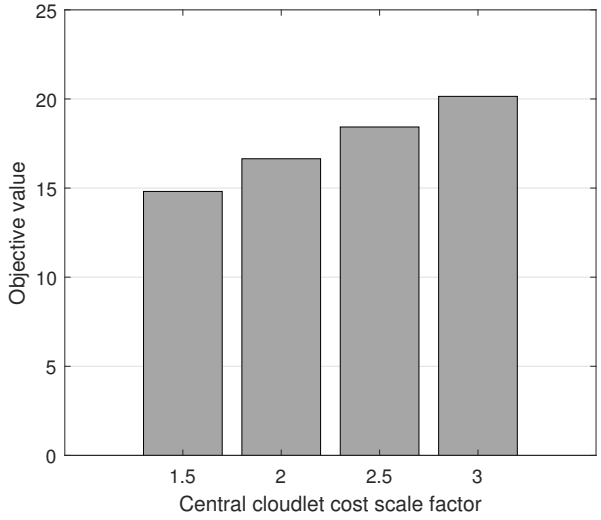


Figure 2.13: Variation of the objective value with respect to the unit cost ratio.

Solver \ $\mathcal{N}$		5	7	9
FMINCON	Runtime	1.10×10^3	4.15×10^3	1.34×10^4
	Obj value	8.429085	15.432419	18.964022
MOSEK	Runtime	22.160336	39.040830	59.512224
	Obj value	8.429118	15.433633	18.965948

Table 2.3: FMINCON vs MOSEK (objective value and running time).

the ratio a rises. This can be explained by the fact that when edge cloudlets are congested, some computational tasks must be migrated in particular cases to the central cloudlet no matter what the unit cost there is, since all UEs' requirements must be guaranteed. This means that, when the unit cost at the central cloudlet is higher, the effect will be directly proportional on the objective, since the total cost increase in this case is imminent.

In Table 2.3, we study the difference in objective and running time between MOSEK which solves the problem while utilizing the conic approximation in (2.14),

and FMINCON which considers the generalized convex constraint (2.16c). The purpose is to validate the accuracy of the conic approximation in (2.14), and study the running time difference between both approaches while varying the number of UEs. As it can be seen by looking at the objective obtained, MOSEK achieves an accuracy of 99.99% compared to running the original general convex problem through FMINCON, while accomplishing a huge decrease in the running time. It can be concluded that approximating a general exponential cone by a set of conic constraints as we have done in (2.14), is a great technique for achieving a huge performance increase.

2.7 Conclusion

In this chapter, we studied the cost and energy efficiency of computation offloading, and optimized the joint transmission power and computation/communication resources allocation in a two-tier MEC system with multiple connected UEs. We first solved the problem using a BnB exhaustive search approach to obtain the optimal solution, and also proposed an SCA-based algorithm to solve the approximate MISOCP of the original non-convex problem. Moreover, we presented a more scalable low-complexity algorithm based on the continuous relaxation. Through numerical results, we evaluated the performance of the proposed algorithms, we analyzed the results in varying scenarios, and also discovered new insights that help NOs better decide when and how to add resources to their network to lower their costs. To the best of our knowledge, this work is the first to study the problem of cost and energy efficiency in hierarchical edge-clouds as it fits with the 5G business models, and it

can be further extended in the future to include other special cases.

Chapter 3

Latency-aware Macro-cell Assisted Edge-clouds Offloading in Heterogeneous Networks with Wireless Backhaul[1]

Heterogeneous networks have allowed the NOs to enhance the spectral efficiency and support a large number of devices by deploying nearby small-cells. In this chapter, we study the problem of computation offloading in a MEC-enabled heterogeneous network with a low-cost wireless backhaul, where we minimize the total devices' energy consumption while respecting their latency deadline. We explore the benefit of

[1]This chapter has been published in IEEE Transactions on Network and Service Management [66].

leveraging the macro-cell cloudlet for computing small-cell users' tasks, where the allocation of backhaul wireless communication resources is optimized while accounting for the resulting RAN interference. We jointly optimize the partial offloading decision, transmission power, and the allocation of computation and radio access communication resources. We mathematically formulate our problem as a non-convex MI-NLP, and due to its complexity, we propose an iterative algorithm based on the SCA method that provides an approximate solution. Through numerical analysis, we perform simulations based on varying configurations, and demonstrate the performance and efficiency of our proposed solution.

3.1 Introduction

The introduction of heterogeneous networks has allowed NOs to overcome the bottleneck produced in the RAN, and thus improving the network capacity and UEs' transmission rates. This is achieved by deploying low-power SCs overlaid within an MC close to the end-users [28]. By leveraging a cloudlet co-located on the MC, a MEC-enabled heterogeneous network allows energy-limited UEs to offload their computational tasks with a low-latency overhead, and therefore enabling ubiquitous 5G services. However, in such case, a large number of SC UEs (SUEs) and MC UEs (MUEs) could overload the MC cloudlet and the backhaul links, increasing the overall latencies. Therefore, recent studies started considering MEC servers to be also co-located on the SCs, e.g. [29, 67, 68], allowing tasks computation in the SUEs' local network, and hence relieving the load on the MC cloudlet and the backhaul

links. Such model forms a multi-tier MEC system [6], where first-tier and second-tier cloudlets are co-located within the SCs and MC, respectively.

When computation offloading is addressed in a heterogeneous network with cloudlet-enabled SCs, the offloading decision usually includes local task computation, and offloading the task to a cloudlet that is co-located within the nearby SC. However, SC cloudlets typically have very limited capabilities, and they could easily become overloaded in the presence of a high connectivity and computation offloading demands. In such case, when the computation can be done only locally or offloaded to the AP-cloudlet, many UEs will be forced to compute their tasks locally when the SC-cloudlet's computation would violate their SLA, e.g. latency deadline. Consequently, more energy will be consumed due to the local device computation, which may seriously inhibit UEs from achieving potential energy savings, entailing a sub-optimal solution.

In this chapter, we investigate and answer the following question: what is the performance benefit brought by utilizing the MC cloudlet as a backup computation unit for serving the SUEs in addition to serving the MUEs in a MEC-based heterogeneous network? There has been interesting studies demonstrating the advantage of multi-tier over single-tier MEC. For instance, in [6], an upper-tier cloudlet behaves as a backup to contain the load of lower-tier cloudlets, which improves the system performance. In this work, we present a novel solution for realizing energy-efficient computation offloading for MEC-enabled heterogeneous networks, where we aim at minimizing the UEs' energy consumption while respecting their latency deadline and accounting for the resulting RAN interference. The significance of this study also

lies in considering a wireless backhaul, which is adopted in most cases due to its relatively low-cost and easier installation. This however requires interference management and the optimization of the communication resources' allocation, where OFDMA is considered as the resources' multiplexing scheme.

3.1.1 Novel Contributions

The contributions of this chapter can be summarized as follows:

1. We model and mathematically formulate the problem of minimizing the UEs' energy consumption in a MEC-based heterogeneous network, by optimizing the SUEs' partial offloading decision (local, SC-cloudlet, or MC-cloudlet), the MUEs' offloading decision (local or MC-cloudlet), the UEs' transmission power, and the allocated computation and communication resources in the SC RAN and backhaul, while respecting the UEs' latency deadline and managing the resulting interference.

2. Due to the problem being a non-convex MI-NLP, we transform it into a SOCP which is a more tractable form that can be solved efficiently via the interior point method [40]. We then propose an efficient low-complexity algorithm based on the SCA approach [41].

3. We present numerical results that evaluate the performance of our proposed algorithm and demonstrate its efficiency based on various system configurations.

The remainder of this chapter is structured as follows: section 3.2 presents an overview of the related work. In section 3.3, we present our system model and

mathematically formulate our non-convex problem. Section 3.4 presents our solution approach to transform the non-convex problem into a more tractable form, where the iterative SCA-based algorithm is presented. Section 3.5 presents numerical results for validating the efficiency of our solution where various simulations are performed. Finally, section 3.6 concludes the chapter.

3.2 Literature Review

We first present studies that addressed computation offloading in a two-tier MEC system. In [6], a hierarchical edge-cloud architecture is proposed, and its latency advantage is demonstrated over flat edge-cloud using formal analysis and simulation. They showed that there is a trade-off between the utilization of the central cloudlet and the incurred communication latency between the edge-cloud tiers. [69] considered a network with one AP where wireless communication resources and cloudlet/cloud computational resources are jointly allocated. [70] considered a multi-AP model with frequency reuse where the allocation of transmission power and computation and wireless access resources are jointly optimized, and backhaul rates are partitioned given a fixed capacity. Those works however assumed the cloud as the second-tier which would incur a high end-to-end latency.

The following studies addressed computation offloading in a heterogeneous network with one cloudlet situated on the MC where computational resources were allocated. In [59], the UEs' energy is minimized by optimizing the offloading decision and the OFDMA discrete RBs' allocation, considering UEs connected to SCs

and the MC, while assuming a fixed latency per data unit over the wireless backhaul. [58, 57, 71, 72, 73] considered a wired backhaul with negligible latency. In [58], the overall network revenue is maximized while optimizing the allocation of access communication resources as a continuous spectrum. [57] minimized the cost in terms of latency and energy where the allocation of RBs resources is optimized. [71] presented a distributed scheme for minimizing the total cost, where the offloading decision and the allocation of OFDMA RBs, UEs' transmission power, and computational resources are optimized. [72] and [73] explored the trade-off between UEs' energy and latency, where both metrics are jointly minimized by optimizing the UEs' offloading decision considering an OFDMA model. [73] in addition, considered the case of a single SC, and optimized the local UEs' computation and transmission power.

The following studies addressed computation offloading in a heterogeneous network where cloudlets are also co-located on the SCs. However, they did not address the option of migrating SC tasks to the MC cloudlet, and therefore did not consider the backhaul communication for data transfer. [29] minimized the UEs' latency with respect to the energy constraint, by optimizing the offloading decision and the computational resources' allocation, but without optimizing the UEs' transmission power and the allocation of wireless communication resources. [67] explored the MEC-aware association of UEs to cells with heterogeneous characteristics in the context of computation offloading, and its advantage is shown over communication-only association. [68] derived a probabilistic model to study the offloading decisions considering both communication and MEC computation performances and also explored the minimization of the mean latency. [74] considered a densely deployed small-cells

where a genetic algorithm is proposed to minimize UEs' energy by jointly optimizing the offloading decision, transmission power, and the allocation of computation and wireless communication resources. [75, 76] did not consider a cloudlet-enabled MC. [75] studied a stochastic optimization for a network where densely deployed SCs can offload their incoming tasks to each others in order to minimize the total offloading latency with respect to the SCs' energy constraint. [76] used the matching theory to minimize the total computation overhead in terms of latency and energy in a network of densely deployed SCs, by optimizing the UEs-to-SC association and offloading decision, transmission power, and the allocation of computation and OFDMA RBs in a distributed manner.

Related works that considered one MC cloudlet except [59] assumed a wired backhaul with negligible latency, which is much more costly in practice. [59] on the other hand, did not optimize the allocation of backhaul communication resources knowing that only one SC is considered. [59] assumed a fixed backhaul transmission latency that is proportional to the data size with a scaling factor. Moreover, works that considered SC cloudlets ignored the possibility of utilizing the MC cloudlet as a backup for SUEs tasks through backhaul communication. This would incur a sub-optimal solution in terms of UEs' energy consumption, as already discussed. However, this work is the first to explore the possibility of leveraging the MC cloudlet for migrating SUEs tasks to the MC-cloudlet over a wireless backhaul when SC-cloudlets become overloaded, in addition to serving the SUEs tasks through the MC cloudlet. Taking this novel aspect into account and for obtaining a highly efficient solution, we minimize the energy consumption of UEs in a MEC-enabled heterogeneous network by

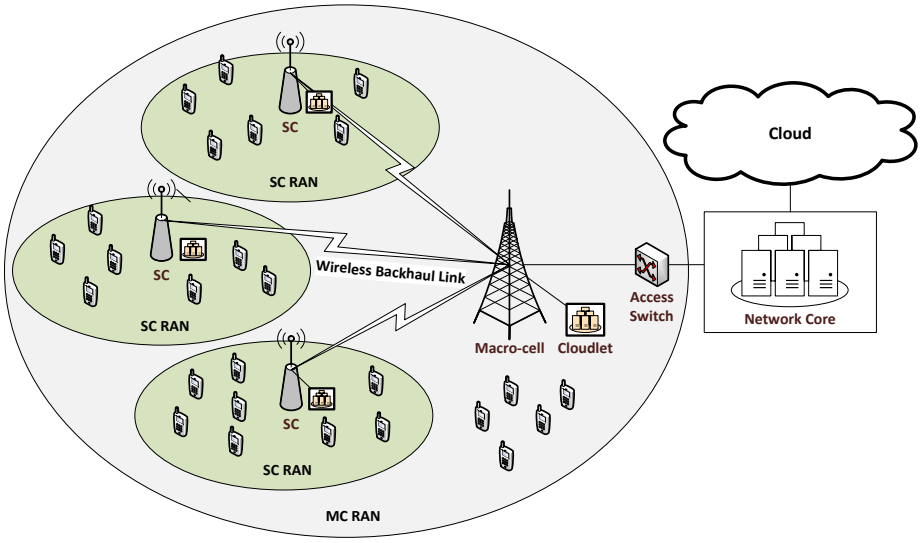


Figure 3.1: System model.

jointly optimizing the offloading decision, UEs' transmission power, and the allocation of computation and wireless communication resources in the RAN and backhaul while managing the resulting interference.

3.3 System Model

3.3.1 Spatial Model

As depicted in Figure 3.1, we consider a MEC-enabled heterogeneous network that consists of S single-antenna SCs indexed by $\mathcal{S} = \{1, ..., S\}$, which coexist within the coverage of one multi-antenna MC with index 0 where $\mathcal{M} = \{0 \cup \mathcal{S}\}$. Each SC $j \in \mathcal{S}$ has in its coverage a set of U UEs indexed by $\mathcal{U}_j = \{1, ..., U\}$, while the MC has in its coverage a set of U_0 UEs indexed by $\mathcal{U}_0 = \{1, ..., U_0\}$. For ease of presentation, we denote by $F = S + U_0$ as the number of units transmitting to the MC indexed by

$\mathcal{F} = \{\mathcal{S}; \mathcal{U}_0\}$. We do not consider the cloud data center in our model, since all UEs are assumed to have low-latency requirements.

In the subsequent sections, we consider that the UEs have tasks to compute. When they offload these tasks, they need to communicate wirelessly. The next section elaborates the computation model.

3.3.2 Computation Model

Each UE $i \in \mathcal{U}_j$ in the range of cell $j \in \mathcal{M}$ has a computational task represented by the tuple $\{d_{ij}, c_{ij}, \bar{L}_{ij}\}$, concatenating the Task input size d_{ij} (Kb), the task computational density c_{ij} (CPU cycles/bit), and the task required latency deadline $\bar{L}_{ij}$ (ms). Those parameters depend on the nature of the application, and can be estimated through task profilers [19]. We consider the computational tasks to be recurring, which means that the system would perform the optimization program and adapt the resources to serve the recurring offloading requests across a given period of time. The SCs and MC are equipped with cloudlets for supporting computation with limited capability, where we consider all SCs to have an associated cloudlet for convenience, noting that our model can be easily adapted to include SCs with no cloudlets. We consider the data-partitioning model [19], where a given task can be partitioned into sub-tasks and partially offloaded to multiple cloudlets and computed simultaneously, e.g. image/video processing. We assume for simplicity that the granularity in task partitioning has arbitrary precision, and that no overlap exists between any two sub-tasks [77]. We denote by s_{ijk} $\forall k \in \mathcal{K} = \{1, 2, 3\}$ an optimization variable indicating the percentage of task size d_{ij} that will be computed locally ($k = 1$), on

the associated SC cloudlet j ($k = 2$), and on the MC cloudlet ($k = 3$). In addition, the following constraints hold:

$$\sum_{k \in \mathcal{K}} s_{ijk} = 1 \ \forall j \in \mathcal{M}, i \in \mathcal{U}_j \tag{3.1a}$$

$$s_{i02} = 0 \ \forall i \in \mathcal{U}_0 \tag{3.1b}$$

where (3.1a) makes sure the whole task is computed across the three computation layers, and (3.1b) is to ensure the second sub-task size for MUEs is zero since they are not associated to a SC. In the next section, we elaborate the communication model.

3.3.3 Communication Model

We consider the uplink communication used by SUEs and MUEs for computation offloading in the SCs' and MC's RAN, and ignore the downlink communication, such as in [59, 57], knowing that the task output size is in general much smaller than the task input size. We consider a wireless backhaul for the communication between the SCs and MBS. Also, we consider an OFDMA system with assumed perfect CSI [19], where the wireless radio spectrum is separated into B RBs, indexed by $\mathcal{B} = \{1, ..., B\}$. We adopt a split-spectrum approach, i.e. $\mathcal{B} = \{\mathcal{B}_1; \mathcal{B}_2\}$, where $\mathcal{B}_1$ is the set of RBs dedicated for the communication between SUEs and their SCs, and $\mathcal{B}_2$ is the set of RBs dedicated for the communication with the MC, i.e. the backhaul transmission between the SCs and MC, and the communication between the MUEs and MC.

Next, we discuss the communication between the SUEs and SCs, and between

the MUEs (and SCs) and the MC, separately.

3.3.3.1 SUEs–SCs Communication

We denote by $\boldsymbol{x} = \{x_{ijb}, \forall j \in \mathcal{S}, i \in \mathcal{U}_j, b \in \mathcal{B}_1\}$ a binary decision variable indicating if RB b is assigned to SUE i for the communication with SC j. When task i is offloaded, multiple RBs can be assigned to SUE i for task transmission. We denote by $p_{ijb} \geq 0$ as the optimization variable indicating the power allocated for the transmission from SUE $i \in \mathcal{U}$ to SC $j \in \mathcal{S}$ on RB $b \in \mathcal{B}_1$ in dBm, $\boldsymbol{p}_b = \{p_{ijb}, \forall j \in \mathcal{S}, i \in \mathcal{U}_j\}$, $\boldsymbol{p} = \{\boldsymbol{p}_b, \forall b \in \mathcal{B}_1\}$. h_{ijlb} is the channel gain from SUE i to SC l on RB b which includes fading and path loss components. We assume the UEs' mobility to be low during the offloading duration, so h_{ijlb} is a constant. Thus, the uplink transmission rate for SUE i in the range of SC j on RB b $r_{ijb}(\boldsymbol{p}_b)$ and the achievable rate $r_{ij}(\boldsymbol{p})$ are defined as

$$r_{ijb}(\boldsymbol{p}_b) = \log\left(1 + \frac{p_{ijb}|h_{ijjb}|^2}{\sum_{l \in \mathcal{S} \setminus} \sum_{k \in \mathcal{U}_l} p_{klb}|h_{kljb}|^2 + N_0}\right) \tag{3.2a}$$

$$r_{ij}(\boldsymbol{p}) = \sum_{b \in \mathcal{B}_1} r_{ijb}(\boldsymbol{p}_b) \tag{3.2b}$$

respectively, where N_0 is the white noise power level. As it can be seen, the inter-cell interference perceived is caused by the SUEs' transmissions in the nearby SCs on the same RB b, knowing that frequency reuse is adopted among SCs to achieve high spectrum efficiency.

The achievable transmission rate $r_{ij}(\boldsymbol{p})$ for UE i in the range of cell j is used for uploading the chunks of the task that will be computed on either the SC or

MC or both. Thus, we denote by $\boldsymbol{r}^{\text{ue}} = \{r^{\text{ue}}_{ijk} \geq 0, \forall j \in \mathcal{S}, i \in \mathcal{U}_j, k \in \{2,3\}\}$ an optimization variable indicating the fraction of the UE achievable rate that is assigned for uploading sub-task s_{ijk} ($k \neq 1$) to the SC cloudlet ($k = 2$) and to the MC cloudlet ($k = 3$). By also denoting $\bar{\boldsymbol{P}} = \{\bar{P}_{ij}, \forall j \in \mathcal{S}, i \in \mathcal{U}_j\}$ as the maximum power budget for all SUEs in dBm, the following constraints are imposed to govern a proper relationship domain of the involved variables:

$$\sum_{i \in \mathcal{U}_j} x_{ijb} \leq 1 \quad \forall j \in \mathcal{S}, b \in \mathcal{B}_1 \tag{3.3a}$$

$$p_{ijb} \leq x_{ijb}\bar{P}_{ij} \quad \forall j \in \mathcal{S}, i \in \mathcal{U}_j, b \in \mathcal{B}_1 \tag{3.3b}$$

$$\sum_{b \in \mathcal{B}_1} p_{ijb} \leq \bar{P}_{ij} \quad \forall j \in \mathcal{S}, i \in \mathcal{U}_j \tag{3.3c}$$

$$r^{\text{ue}}_{ij1} = 0 \quad \forall j \in \mathcal{S}, i \in \mathcal{U}_j \tag{3.3d}$$

$$\sum_{k \in \mathcal{K}} r^{\text{ue}}_{ijk} \leq r_{ij}(\boldsymbol{p}) \quad \forall j \in \mathcal{S}, i \in \mathcal{U}_j \tag{3.3e}$$

where (3.3a) ensures orthogonal radio resources' allocation in the SCs RAN incurring no intra-cell interference, (3.3b) assures no transmission power on an RB that was not assigned to a given SUE, (3.3c) is for respecting the SUEs' power threshold, (3.3d) is to ensure the sub-task assigned for local computation does not receive an upload rate, and (3.3e) ensures the sum of the partitioned rates on UE i in the range of SC j respects its achievable rate.

3.3.3.2 Communication with the MC

Communication with the MC is needed by MUEs to offload their sub-tasks, and by SCs in case they are migrating sub-tasks from their SUEs. Without loss of generality, we consider Zero-forcing (ZF) receiver at the multi-antenna MC [78]. Thus, we assume that the number of antennas on the MC, denoted by N, is able to support simultaneous connections from the transmitting units $\mathcal{F}$ on a given RB b as in [79], i.e. $F \leq N$, incurring no interference among these units, noting that a system with $F > N$ can be studied in the future. By denoting $\rho_{ib} \geq 0$ as the optimization variable indicating the power allocated for the transmission from MUE (or SC) $i \in \mathcal{F}$ to the MC on RB $b \in \mathcal{B}_2$, $\boldsymbol{\rho}_i = \{\rho_{ib}, \forall b \in \mathcal{B}_2\}$, and $\boldsymbol{\rho} = \{\boldsymbol{\rho}_i, \forall i \in \mathcal{F}\}$, the uplink transmission rate for MUE (or SC) i on RB b $R_{ib}(\rho_{ib})$ and the achievable rate $R_i(\boldsymbol{\rho}_i)$ are defined as

$$R_{ib}(\rho_{ib}) = \log\left(1 + \frac{\rho_{ib}}{||\boldsymbol{w}_{ib}||^2 N_0}\right) \tag{3.4a}$$

$$R_i(\boldsymbol{\rho}_i) = \sum_{b \in \mathcal{B}_2} R_{ib}(\rho_{ib}) \tag{3.4b}$$

where $\boldsymbol{w}_{ib} \in \mathbb{C}^{N \times 1}$ is the ZF receive beamforming row vector obtained from matrix $\boldsymbol{w}_b \in \mathbb{C}^{F \times N}$, and $\boldsymbol{w}_b = (\boldsymbol{h}_b^T \boldsymbol{h}_b)^{-1} \boldsymbol{h}_b^T$ is the pseudo-inverse of the channel state matrix $\boldsymbol{h}_b \in \mathbb{C}^{N \times F}$ for RB b, which includes fading and path loss components.

This work is innovative for exploring the option of migrating SUE tasks to the MC cloudlet through the backhaul for decreasing the overall UEs' energy and latency. Thus, we make sure to optimize the backhaul communication through: 1) optimizing the transmission power $\boldsymbol{\rho}$ of the SCs on the RBs, and 2) optimizing the partitioning

of the achievable rate $R_i(\boldsymbol{\rho}_i)$ for SC $i \in \mathcal{F}$ on the migrated tasks through that SC. To achieve the latter goal, we denote by $\boldsymbol{r}^{\text{sc}} = \{r_{ij}^{\text{sc}} \geq 0, \forall j \in \mathcal{S}, i \in \mathcal{U}_j\}$ an optimization variable indicating the fraction of the achievable rate on SC j that is assigned for migrating the chunk of task i to the MC. By denoting $\bar{\boldsymbol{\rho}} = \{\bar{\rho}_i, \forall i \in \mathcal{F}\}$ as the maximum power budget for all MUEs (and SCs) in dBm, the following constraints are imposed to govern a proper relationship domain of the involved variables:

$$\sum_{b \in \mathcal{B}_2} \rho_{ib} \leq \bar{\rho}_i \quad \forall i \in \mathcal{F} \tag{3.5a}$$

$$\sum_{i \in \mathcal{U}_j} r_{ij}^{\text{sc}} \leq R_j(\boldsymbol{\rho}_j) \quad \forall j \in \mathcal{S} \tag{3.5b}$$

where (3.5a) allows respecting the MUEs (and SCs)' power threshold, and (3.5b) ensures that the sum of all partitioned rates on SC j does not exceed its achievable rate. Next, we model the latency and energy consumption resulting from the computation of a task i.

3.3.4 Latency Model

3.3.4.1 Local Computation Latency

We denote by f_{ij}^{loc} as the local computation capability in cycles/second for UE i in the range of cell j which can be different from other UEs. The resulting local computation latency for task i is defined as

$$L_{ij}^{\text{loc}}(s_{ij1}) = \frac{c_{ij} d_{ij} s_{ij1}}{f_{ij}^{\text{loc}}} \tag{3.6}$$

which directly depends on the local UE computation capability f_{ij}^{loc}.

3.3.4.2 Upload Latency

The upload latency for SUE and MUE $i \in \mathcal{U}_j$ incurred from task transmission to the associated cell j are given by

$$L_{ijk}^{\text{sue}}\left(s_{ijk}; r_{ijk}^{\text{ue}}\right) = \frac{d_{ij} s_{ijk}}{r_{ijk}^{\text{ue}}} \ \forall k \in \{2, 3\} \tag{3.7a}$$

$$L_{i}^{\text{mue}}\left(s_{i03}; \boldsymbol{\rho}_i\right) = \frac{d_{i0} s_{i03}}{R_i(\boldsymbol{\rho}_i)} \tag{3.7b}$$

respectively, where the upload latency depends on the corresponding transmission rate, which in turn depends on the allocated power for transmission on the assigned RBs (and on the resulting inter-cell interference for SUEs). It can be seen in (3.7a) that having a higher transmission rate (resulting from a higher power and more assigned RBs) will decrease the upload latency, but this will result in interference on SUEs located in other cells utilizing the same RBs. Therefore, these values must be jointly optimized for achieving the best objective. Also in (3.7b), transmission power on the assigned RBs will be optimized while respecting the power threshold.

In case a chunk of task i (i.e. sub-task s_{ij3}) is being migrated from the associated SC j to the MC through the backhaul, the migration latency in the backhaul which depends on the SC assigned rate r_{ij}^{sc} is considered, and is defined as

$$L_{ij}^{\text{mig}}\left(s_{ij3}; r_{ij}^{\text{sc}}\right) = \frac{d_{ij} s_{ij3}}{r_{ij}^{\text{sc}}} \tag{3.8}$$

We observe that optimizing the partitioned rate r_{ij}^{sc} is essential for making sure the end-to-end latency is respected for all tasks that are migrated to the MC cloudlet.

3.3.4.3 Cloudlet Computation Latency

We denote by $\boldsymbol{f}^{\text{sc}} = \{f_{ij}^{\text{sc}} \geq 0, \forall j \in \mathcal{S}, i \in \mathcal{U}_j\}$ and $\boldsymbol{f}^{\text{mc}} = \{f_{ij}^{\text{mc}} \geq 0, \forall j \in \mathcal{M}, i \in \mathcal{U}_j\}$ the optimization variables indicating the computation power allocated for executing sub-task s_{ij2} on the SC cloudlet and sub-task s_{ij3} on the MC cloudlet, respectively. The resulting latency for SC and MC cloudlet computation are given by

$$L_{ij}^{\text{sc}}(s_{ij2}; f_{ij}^{\text{sc}}) = \frac{c_{ij} d_{ij} s_{ij2}}{f_{ij}^{\text{sc}}} \tag{3.9a}$$

$$L_{ij}^{\text{mc}}(s_{ij3}; f_{ij}^{\text{mc}}) = \frac{c_{ij} d_{ij} s_{ij3}}{f_{ij}^{\text{mc}}} \tag{3.9b}$$

respectively. It can be seen that allocating more computational resources for a given sub-task will decrease its computation latency, but this will limit the available resources for executing other offloaded sub-tasks on the corresponding cloudlet, which in turn will increase their computation latency. Migrating a chunk of the SUE task to the MC cloudlet can constitute a great advantage when the SC cloudlet is overloaded. This is because the MC cloudlet can perform the sub-task computation and further reduce the chunk of task processed locally, which decreases the energy consumption while also reducing the latency thanks to the parallelism.

3.3.4.4 Latency Constraint

Since local computation and offloading are done simultaneously for the chunks of a task i, the end-to-end latency for the whole task depends on the end-to-end latency of the sub-task that finishes last. Thus, to respect the UE latency deadline, the latency of each sub-task must respect the task latency deadline. To achieve this, we impose the following constraints for task i in the range of cell j:

$$L_{ij}^{\text{loc}}(s_{ij1}) \leq \bar{L}_{ij} \quad \forall j \in \mathcal{M}, i \in \mathcal{U}_j \tag{3.10a}$$

$$L_{ij2}^{\text{sue}}(s_{ij2}; r_{ij2}^{\text{ue}}) + L_{ij}^{\text{sc}}(s_{ij2}; f_{ij}^{\text{sc}}) \leq \bar{L}_{ij} \; \forall j \in \mathcal{S}, i \in \mathcal{U}_j \tag{3.10b}$$

$$L_{ij3}^{\text{sue}}(s_{ij3}; r_{ij3}^{\text{ue}}) + L_{ij}^{\text{mig}}(s_{ij3}; r_{ij}^{\text{sc}}) + L_{ij}^{\text{mc}}(s_{ij3}; f_{ij}^{\text{mc}}) \leq \bar{L}_{ij}$$
$$\forall j \in \mathcal{S}, i \in \mathcal{U}_j \tag{3.10c}$$

$$L_{i}^{\text{mue}}(s_{i03}; \boldsymbol{\rho}_i) + L_{i0}^{\text{mc}}(s_{ij3}; f_{i0}^{\text{mc}}) \leq \bar{L}_{i0} \quad \forall i \in \mathcal{U}_0 \tag{3.10d}$$

where constraints (3.10a)–(3.10d) make sure the latency deadline is respected when: a UE sub-task is computed locally (3.10a), an SUE sub-task is offloaded to the SC cloudlet (3.10b), an SUE sub-task is migrated to the MC cloudlet (3.10c), and an MUE sub-task is offloaded to the MC cloudlet (3.10d).

3.3.5 Energy Model

Energy consumption for UE i in the range of cell j results from both local computation and task transmission. We next present both energy models.

3.3.5.1 Local Computation Energy

The energy resulting from local Computation for UE i in the range of cell j is defined as

$$E_{ij}^{\text{loc}}(s_{ij1}) = c_{ij} d_{ij} E_{ij}^{\text{cyc}} s_{ij1} \tag{3.11}$$

where E_{ij}^{cyc} represents the consumed energy per CPU cycle, which depends on the UE circuit architecture and can be obtained by the measurement method in [80].

3.3.5.2 Upload Energy

The energy consumption for SUE and MUE $i \in \mathcal{U}_j$ resulting from task transmission to cell j depends on the allocated transmission power and the assigned RBs, and is given by

$$E_{ij}^{\text{sue}}(\boldsymbol{s}_{ij}; \boldsymbol{p}) = \frac{(\sum_{b \in \mathcal{B}_1} p_{ijb})(s_{ij2} + s_{ij3}) d_{ij}}{r_{ij}(\boldsymbol{p})} \tag{3.12a}$$

$$E_{i}^{\text{mue}}(s_{i03}; \boldsymbol{\rho}_i) = \frac{(\sum_{b \in \mathcal{B}_2} \rho_{ib}) s_{i03} d_{ij}}{R_i(\boldsymbol{\rho}_i)} \tag{3.12b}$$

respectively.

3.3.6 Problem Formulation

Our objective is to minimize the total energy consumption for all UEs while respecting their latency deadline. This is done by optimizing the offloading decision for each UE i. In addition, the transmission power, wireless communication resources' allocation in the RAN and backhaul, and the computational resources' allocation

on the host cloudlet are all optimized. We note that another version of the problem could be to jointly minimize a weighted objective of UEs' energy and latency. However, we consider in our model the UEs to have a hard latency constraint that must be respected in order to meet their required QoS. Hence, lowering the latency past the required threshold would be unnecessary for the NO. By denoting $\boldsymbol{\chi}_1 = \{\boldsymbol{s}, \boldsymbol{x}, \boldsymbol{p}, \boldsymbol{\rho}, \boldsymbol{r}^{\text{ue}}, \boldsymbol{r}^{\text{sc}}, \boldsymbol{f}^{\text{sc}}, \boldsymbol{f}^{\text{mc}}\}$, the joint transmission power and computation/-communication resources' allocation for computation offloading problem in MEC-enabled heterogeneous network, denoted as $\mathcal{P}_1$, is formulated as

$$\mathcal{P}_1 : \min_{\boldsymbol{\chi}_1} \sum_{j\in\mathcal{M}}\sum_{i\in\mathcal{U}_j} E_{ij}^{\text{loc}}(s_{ij1}) + \sum_{j\in\mathcal{S}}\sum_{i\in\mathcal{U}_j} E_{ij}^{\text{sue}}(\boldsymbol{s}_{ij};\boldsymbol{p}) + \sum_{i\in\mathcal{U}_0} E_i^{\text{mue}}(s_{i03};\boldsymbol{\rho}_i) \tag{3.13a}$$

$$\text{s.t. } r_{ij3}^{\text{ue}} \le r_{ij}^{\text{sc}} \quad \forall j \in \mathcal{S}, i \in \mathcal{U}_j \tag{3.13b}$$

$$\sum_{i\in\mathcal{U}_j} f_{ij}^{\text{sc}} \le \bar{F}_j \quad \forall j \in \mathcal{S} \tag{3.13c}$$

$$\sum_{j\in\mathcal{M}}\sum_{i\in\mathcal{U}_j} f_{ij}^{\text{mc}} \le \bar{F}_0 \tag{3.13d}$$

$$(3.1),\ (3.3),\ (3.5),\ (3.10) \tag{3.13e}$$

$$x_{ijb} \in \{0,1\} \tag{3.13f}$$

$$s_{ijk}, p_{ijb}, \rho_{ib}, r_{ijk}^{\text{ue}}, r_{ij}^{\text{sc}}, f_{ij}^{\text{sc}}, f_{ij}^{\text{mc}} \ge 0 \tag{3.13g}$$

The objective function (3.13a) is minimizing the total energy consumption (in Mj) for all UEs. Constraint (3.13b) prevents the backhaul from being a bottleneck when migrating a task to the MC, i.e. the task assigned uplink rate in the backhaul should

be greater than or equal to its corresponding rate in the SC RAN. This constraint adds a major significance by ensuring transmission stability in the wireless backhaul. Constraints (3.13c) and (3.13d) are for respecting the computational resources' capacity, denoted by $\bar{F}_j$, on the SCs and MC cloudlet, respectively.

It is worth noting that this problem always has a feasible solution, since all UEs can compute their whole task locally but with a high energy consumption. Thus, all UEs will try to offload a chunk of their task to their associated cell cloudlet, noting that SUEs further have the possibility to migrate a chunk of the task to the MC cloudlet. In fact, deciding on the optimal communication resources' allocation while managing the resulting interference and the balance between the latency and power allocation while leveraging the MC cloudlet through backhaul communication for minimizing the energy consumption, is a highly challenging problem that we proceed to tackle. All mathematical symbols used thus far are summarized in Table 3.1.

It can be seen that the following terms in problem (3.13) are non-convex: the SC rate function (3.2b) used in constraint (3.3e), the latency functions (3.7) used in the latency constraints (3.10b)–(3.10d), and the energy functions (3.12) used in the objective (3.13a). In addition, constraint (3.13f) implies that (3.13) is an integer optimization problem. In fact, the formulated problem (3.13) is a non-convex MINLP, which is generally difficult to solve.

Notation	Description
$\mathcal{S}$	Set of SCs.
$\mathcal{M}$	Set of both the SCs and MC.
$\mathcal{U}_j$	Set of UEs in the range of cell j.
$\mathcal{F}$	Set of units transmitting to the MC.
$\mathcal{K}$	Computational nodes (local, SC and MC cloudlet).
$\mathcal{B}$	Set of RBs.
$\mathcal{B}_1$	Set of RBs allocated for the communication with the SCs.
$\mathcal{B}_2$	Set of RBs allocated for the communication with the MC.
d_{ij}	Task input size (Kb).
c_{ij}	Task computational density (CPU cycles/bit).
$\bar{L}_{ij}$	Task required latency deadline (ms).
f_{ij}^{loc}	Local computation power (GHz).
E_{ij}^{cyc}	Energy consumption per computation cycle (Joule).
F_j	Computational capability (GHz).
h_{ijlb}	Channel gain from SUE i in the range of SC j.
$\boldsymbol{w}_{ib}$	ZF receive beamforming vector.
N_0	White noise power level.
P_{ij} $(\bar{\rho}_i)$	Maximum transmission power for SUE i (dBm).
$x_{ijb} \in \{0,1\}$	Indicates if RB b is assigned to SUE i in the range of SC j.
$s_{ijk} \in \mathbb{R}^+$	Percentage of task size d_{ij} computed on node $k \in \mathcal{K}$.
$p_{ijb}(\rho_{ib}) \in \mathbb{R}^+$	Transmit power for SUE i on RB m (Watts).
$r_{ijk}^{\text{ue}} \in \mathbb{R}^+$	Fraction of the UE's rate for sub-task s_{ijk} (Mbps).
$r_{ij}^{\text{sc}} \in \mathbb{R}^+$	Fraction of the assigned SC's achievable rate (Mbps).
$f_{ij}^{\text{sc}} \in \mathbb{R}^+$	Allocated computational resources on the SC cloudlet (GHz).
$f_{ij}^{\text{mc}} \in \mathbb{R}^+$	Allocated computational resources on the MC cloudlet (GHz).

Table 3.1: Table of Notations

3.4 Proposed Iterative Low-Complexity Algorithm

Due to the high complexity of the proposed non-convex MI-NLP problem, in this section, we propose to approach a solution of (3.13) with a more pragmatic, efficient, and low computation complexity algorithm. We first reveal the factors that make the formulated problem $\mathcal{P}_1$ non-convex by separating the convex and non-convex

constraints. Then, we invoke the framework of SCA to convexify the non-convex constraints so that the transformed problem results in a series of approximated mixed integer convex problems.

3.4.1 Convex Approximation

We first note that constraint (3.5b) is recognized as a convex exponential cone due to the existence of the log function $R_{ib}(\rho_{ib})$, and can be easily approximated by a system of second order cone constraints, similar to the transformation done in ([81], Sec. III). This conversion, in addition to similar conic conversions that will be subsequently discussed, ensures the resulting problem is a standard Mixed-Integer Second-Order Cone Program (MI-SOCP), where a modern dedicated solver such as MOSEK [82] is available to solve the problem efficiently in each iteration.

The underlying issues which make (3.13) a non-convex MI-NLP problem and hence difficult to solve, are due to the existence of the following non-convex terms: function $r_{ij}(\boldsymbol{p})$ in (3.3e) which is non-convex with respect to $\boldsymbol{p}$, the latency functions in (3.10b)–(3.10d) which are non-convex with respect to their input variables, and functions $E_{ij}^{\text{sue}}(\boldsymbol{s}_{ij};\boldsymbol{p})$ and $E_{i}^{\text{mue}}(s_{i03};\boldsymbol{\rho}_i)$ in (3.13a) which are non-convex with respect to $\boldsymbol{s}_{ij},\boldsymbol{p}$ and $s_{i03},\boldsymbol{\rho}_i$, respectively.

In the following subsections, we will address the non-convex constraints, where we invoke the SCA-based framework to approximate (3.13) into a series of approximated MI-SOCP problems. Then, we develop an SCA-based MI-SOCP algorithm to iteratively solve until convergence.

3.4.1.1 Proposed approximation of (3.3e)

After some Algebraic manipulation, function $r_{ij}(\boldsymbol{p})$ can be rewritten as follows:

$$r_{ij}(\boldsymbol{p}) = \sum_{b\in\mathcal{B}_1} \underbrace{\log\left(\overbrace{\sum_{l\in\mathcal{S}\setminus}\sum_{k\in\mathcal{U}_l} p_{klb}|h_{kljb}|^2+N_0+p_{ijb}|h_{ijjb}|^2}^{\breve{\gamma}_{ijb}(\boldsymbol{p}_b)}\right)}_{\breve{r}_{ijb}(\boldsymbol{p}_b)} - \underbrace{\log\left(\overbrace{\sum_{l\in\mathcal{S}\setminus}\sum_{k\in\mathcal{U}_l} p_{klb}|h_{kljb}|^2+N_0}^{\hat{\gamma}_{jb}(\boldsymbol{p}_b)}\right)}_{\hat{r}_{jb}(\boldsymbol{p}_b)} \tag{3.14}$$

In order to resolve the DC form in constraint (3.3e), $r_{ij}(\boldsymbol{p})$ should be made concave. To handle that task, we are motivated by the inner-approximation method in [63] to approximate function $\hat{r}_{jb}(\boldsymbol{p}_b)$ by its upper-bounded convex function $\hat{R}_{jb}(\boldsymbol{p}_b;\boldsymbol{p}_b^{(n)})$ around the point $\boldsymbol{p}_b^{(n)}$ (at iteration n of the SCA algorithm) as

$$\hat{R}_{jb}(\boldsymbol{p}_b;\boldsymbol{p}_b^{(n)}) = \hat{r}_{jb}(\boldsymbol{p}_b^{(n)}) + \frac{\hat{\gamma}_{jb}(\boldsymbol{p}_b)-\hat{\gamma}_{jb}(\boldsymbol{p}_b^{(n)})}{\hat{\gamma}_{jb}(\boldsymbol{p}_b^{(n)})} \tag{3.15}$$

By replacing function $r_{ij}(\boldsymbol{p})$ by its approximate, constraint (3.3e) can now be written as

$$\sum_{k\in\mathcal{K}} r_{ijk}^{\mathrm{ue}} \leq \sum_{b\in\mathcal{B}_1}\left(\breve{r}_{ijb}(\boldsymbol{p}_b) - \hat{R}_{jb}(\boldsymbol{p}_b;\boldsymbol{p}_b^{(n)})\right) \tag{3.16}$$

which is now convex due to the existence of the log function $\breve{r}_{ijb}(\boldsymbol{p}_b)$, and constraint (3.16) can be easily linearized similar to constraint (3.5b).

3.4.1.2 Proposed approximation of (3.10b)–(3.10d)

By introducing slack variables $\boldsymbol{\theta} = \{\theta_{ijk} \geq 0, \forall j \in \mathcal{M}, i \in \mathcal{U}_j, k \in \{2,3\}\}$ and $\boldsymbol{\eta} = \{\eta_i \geq 0, \forall i \in \mathcal{U}_0\}$, constraints (3.10b)–(3.10d) can be equivalently written as

$$\frac{d_{ij}\theta_{ij2}^2}{r_{ij2}^{\mathrm{ue}}} + \frac{c_{ij}d_{ij}\theta_{ij2}^2}{f_{ij}^{\mathrm{sc}}} \leq \bar{L}_{ij} \quad \forall j \in \mathcal{S}, i \in \mathcal{U}_j \tag{3.17a}$$

$$\frac{d_{ij}\theta_{ij3}^2}{r_{ij3}^{\mathrm{ue}}} + \frac{d_{ij}\theta_{ij3}^2}{r_{ij}^{\mathrm{sc}}} + \frac{c_{ij}d_{ij}\theta_{ij3}^2}{f_{ij}^{\mathrm{mc}}} \leq \bar{L}_{ij}$$

$$\forall j \in \mathcal{S}, i \in \mathcal{U}_j \tag{3.17b}$$

$$\frac{d_{i0}\theta_{i03}^2}{\eta_i} + \frac{c_{i0}d_{i0}\theta_{i03}^2}{f_{i0}^{\mathrm{mc}}} \leq \bar{L}_{i0} \quad \forall i \in \mathcal{U}_0 \tag{3.17c}$$

$$\eta_i \leq R_i(\boldsymbol{\rho}_i) \tag{3.17d}$$

$$s_{ijk} - \theta_{ijk}^2 \leq 0 \tag{3.17e}$$

The equivalence between (3.10b) and (3.17a), (3.17e) is proved in Appendix A.2, noting that the equivalence between (3.10c), (3.10d) and their corresponding transformations can be similarly proved. Constraints (3.17a)–(3.17d) are convex, where constraints (3.17a)–(3.17c) can be easily converted to quadratic cones, and constraint (3.17d) can be easily linearized similar to constraint (3.5b). However, constraint (3.17e) is non-convex due to the existence of concave function $f(\theta_{ijk}) = -(\theta_{ijk})^2$ on the lesser side of the inequality, causing a DC form. In order to convexify (3.17e), we apply the inner-approximation method in [63] to approximate $f(\theta_{ijk})$ by its upper-bounded convex function $\tilde{f}(\theta_{ijk}; \theta_{ijk}^{(n)})$ around the point $\theta_{ijk}^{(n)}$ (at iteration n of the SCA

algorithm) as

$$\tilde{f}(\theta_{ijk};\theta_{ijk}^{(n)}) = -(\theta_{ijk}^{(n)})^2 - 2\theta_{ijk}^{(n)}(\theta_{ijk} - \theta_{ijk}^{(n)}) \tag{3.18}$$

By replacing function $f(\theta_{ijk})$ by its approximate, constraint (3.17e) can now be written as

$$s_{ijk} + \tilde{f}(\theta_{ijk};\theta_{ijk}^{(n)}) \leq 0 \tag{3.19}$$

where constraint (3.19) is now linear.

3.4.1.3 Proposed approximation of (3.13a)

By introducing slack variables $\boldsymbol{\zeta} = \{\zeta_{ij} \geq 0, \forall j \in \mathcal{S}, i \in \mathcal{U}_j\}$, $\boldsymbol{\beta} = \{\beta_{ij} \geq 0, \forall j \in \mathcal{S}, i \in \mathcal{U}_j\}$, $\boldsymbol{\alpha} = \{\alpha_i \geq 0, \forall i \in \mathcal{U}_0\}$, constraint (3.13a) can be equivalently written as

$$\sum_{j\in\mathcal{M}}\sum_{i\in\mathcal{U}_j} E_{ij}^{\text{loc}}(s_{ij1}) + \sum_{j\in\mathcal{S}}\sum_{i\in\mathcal{U}_j}\frac{d_{ij}\beta_{ij}^2}{\zeta_{ij}} + \sum_{i\in\mathcal{U}_0}\frac{d_{i0}\alpha_i^2}{\eta_i} \tag{3.20a}$$

$$\zeta_{ij} \leq \sum_{b\in\mathcal{B}_1}\left(\breve{r}_{ijb}(\boldsymbol{p}_b) - \hat{R}_{jb}(\boldsymbol{p}_b;\boldsymbol{p}_b^{(n)})\right) \tag{3.20b}$$

$$s_{ij2} + s_{ij3} \leq \frac{\beta_{ij}^2}{\sum_{b\in\mathcal{B}_1} p_{ijb}} \tag{3.20c}$$

$$s_{i03} \leq \frac{\alpha_i^2}{\sum_{b\in\mathcal{B}_2}\rho_{ib}} \tag{3.20d}$$

where (3.20b) is directly converted into its linear form following the approximation done in (3.16). The equivalence between (3.13a) and (3.20) can be proved similar to the proof in Appendix A.2. We observe that constraint (3.20a) is convex, where

(3.20a) can be easily converted to a quadratic cone. On the other hand, constraints (3.20c) and (3.20d) are non-convex due to the existence of the convex terms β_{ij}^2 and α_i^2 on the greater-than side of the inequality, rendering the constraints as a DC form.

With a slight abuse of notation x, these functions have the same form and can be represented by $g(x) = \frac{x^2}{y}$. To linearize constraints (3.20c) and (3.20d), we proceed to employ the inner-approximation method to approximate function $g(x)$ by its upper-bounded convex function $\tilde{g}(x, y; x^{(n)}, y^{(n)})$ around the points $x^{(n)}$ and $y^{(n)}$ as

$$\tilde{g}(x, y; x^{(n)}, y^{(n)}) = \frac{2x^{(n)}x}{y^{(n)}} - \frac{(x^{(n)})^2}{(y^{(n)})^2}y \tag{3.21}$$

By replacing the non-convex functions by their approximates, constraints (3.20c) and (3.20d) can now be written in their general form as

$$s_{ij2} + s_{ij3} \leq \tilde{g}\left(\beta_{ij}, \sum_{b\in\mathcal{B}_1} p_{ijb}; \beta_{ij}^{(n)}, \sum_{b\in\mathcal{B}_1} p_{ijb}^{(n)}\right) \tag{3.22a}$$

$$s_{i03} \leq \tilde{g}\left(\alpha_i, \sum_{b\in\mathcal{B}_2} \rho_{ib}; \alpha_i^{(n)}, \sum_{b\in\mathcal{B}_2} \rho_{ib}^{(n)}\right) \tag{3.22b}$$

We remark that constraints (3.22) are now linear.

3.4.2 SOCP Approximation and SCA-based Algorithm

The MI-SOCP approximation of problem (3.13) will still pause scalability limitations, preventing the SCA algorithm from being applied to big instances due to the mixed-integer nature of the problem, which is caused by constraint (3.13f). To solve that problem, we adopt a similar approach to [83], and relax the binary constraint for

variable $\boldsymbol{x}$ by introducing the following constraints:

$$0 \leq x_{ijb} \leq 1 \tag{3.23a}$$

$$0 \leq x_{ijb} - \tilde{h}(x_{ijb}; x_{ijb}^{(n)}) \leq \delta_{ijb} \tag{3.23b}$$

where $\boldsymbol{\delta} = \{\delta_{ijb} \geq 0, \forall j \in \mathcal{S}, i \in \mathcal{U}_j, b \in \mathcal{B}_1\}$ is a newly introduced slack variable. Constraint (3.23b) will force variable $\boldsymbol{x}$ to take a binary value with a penalty term added to the objective.

By denoting $\boldsymbol{\chi} = \{\boldsymbol{\chi}_1; \boldsymbol{\theta}, \boldsymbol{\eta}, \boldsymbol{\zeta}, \boldsymbol{\beta}, \boldsymbol{\alpha}\}$ and employing all the conic transformations and approximations from (3.15), (3.18), and (3.21), an approximated SOCP of the non-convex MI-NLP problem (3.13), denoted by $\tilde{\mathcal{P}}^{(n)}$, can be formulated at the nth iteration as

$$\mathcal{P}_1^{(n)} : \min_{\boldsymbol{\chi}_1} \ (3.20a) + A \sum_{j \in \mathcal{S}} \sum_{i \in \mathcal{U}_j} \sum_{b \in \mathcal{B}_1} \delta_{ijb} \tag{3.24a}$$

$$\text{s.t. } r_{ij3}^{\text{ue}} \leq r_{ij}^{\text{sc}} \quad \forall j \in \mathcal{S}, i \in \mathcal{U}_j \tag{3.24b}$$

$$\sum_{i \in \mathcal{U}_j} f_{ij}^{\text{sc}} \leq \bar{F}_j \quad \forall j \in \mathcal{S} \tag{3.24c}$$

$$\sum_{j \in \mathcal{M}} \sum_{i \in \mathcal{U}_j} f_{ij}^{\text{mc}} \leq \bar{F}_0 \tag{3.24d}$$

$$\text{(3.1), (3.3a)–(3.3d), (3.5), (3.10a), (3.16),}$$

$$\text{(3.17a)–(3.17d), (3.19), (3.20b), (3.22), (3.23)} \tag{3.24e}$$

$$s_{ijk}, p_{ijb}, \rho_{ib}, r_{ijk}^{\text{ue}}, r_{ij}^{\text{sc}}, f_{ij}^{\text{sc}}, f_{ij}^{\text{mc}} \geq 0 \tag{3.24f}$$

where $A > 0$ is the penalty parameter. The pseudocode for the corresponding SCA-based algorithm is given in Algorithm 4.

Algorithm 4 SCA-based SOCP Algorithm.

Initialize:
$n = 0$;
Choose an initial point $\boldsymbol{p}^{(n)}, \boldsymbol{\rho}^{(n)}, \boldsymbol{\theta}^{(n)}, \boldsymbol{\beta}^{(n)}, \boldsymbol{\alpha}^{(n)}$;
repeat
    Solve $\tilde{\mathcal{P}}^{(n)}$ to obtain the optimal solution at the nth iteration $\Omega^\star = \{\boldsymbol{s}^\star, \boldsymbol{x}^\star, \boldsymbol{p}^\star, \boldsymbol{\rho}^\star, \boldsymbol{r}^\star, \boldsymbol{f}^{\mathrm{sc}\star}, \boldsymbol{f}^{\mathrm{mc}\star}, \boldsymbol{\theta}^\star, \boldsymbol{\eta}^\star,$
    $\boldsymbol{\zeta}^\star, \boldsymbol{\beta}^\star, \boldsymbol{\alpha}^\star\}$;
    Update $\boldsymbol{p}^{(n)} = \boldsymbol{p}^\star, \boldsymbol{\rho}^{(n)} = \boldsymbol{\rho}^\star, \boldsymbol{\theta}^{(n)} = \boldsymbol{\theta}^\star, \boldsymbol{\beta}^{(n)} = \boldsymbol{\beta}^\star, \boldsymbol{\alpha}^{(n)} = \boldsymbol{\alpha}^\star$;
    $n = n + 1$;
until Convergence of the objective of $\tilde{\mathcal{P}}^{(n)}$.

Our centralized solution would be conducted by a central computation unit (e.g. located on the MC cloudlet) which would run Algorithm 4 after acquiring the CSI and the UEs' configurations through the control plane [19], where the offloading and transmission power decisions would be eventually signaled back to the UEs. It is worth noting that our solution can adapt to online scenarios by solving for the current state of the network when a change in the offloading requests occurs after a period of time. However, more efficient solutions can be developed to address online scenarios, which is out of the scope of this work. Also, our results can serve as a benchmark for other efficient methods, whether they are online or not.

Convergence Analysis: The convergence of Algorithm 4 can be guaranteed by showing that the series of the objective resulted from Algorithm 4 is monotonically convergent. Let $\Gamma^{(n)}$ denote the optimal objective value and $\Omega^{(n)}$ denote the optimal solution set at the nth iteration of Algorithm 4. Due to the convex approximations

in (3.15), (3.18), and (3.21), the updating rules in Algorithm 4, c.f., Step 7, ensure that the solution set $\Omega^{(n)}$ is a feasible solution to problem $\mathcal{P}_2$ at step $n+1$. This subsequently leads to the results of $\Gamma^{(n+1)} \leq \Gamma^{(n)}$, which means that Algorithm 4 generates a non-increasing sequence of objective function values. Due to the latency constraints, the sequence of $\Gamma^{(n)}, n = 1, 2, \ldots$ is bounded below and therefore, Algorithm 4 guarantees that the objective converges.

Complexity Analysis: The overall complexity of Algorithm 4 mainly depends on that of solving the SOCP problem in (3.24). The constraints in problem (3.24) approximately consist of a total number of $12SU + 6U0 + 2SUB_1 + U_0B_2$ variables and a number of $25SU + 14U_0 + SB_1 + 4SUB_1 + U_0B_2 + 3S + 1$ constraints of dimension $SU + U_0 + S$. Thus, in each iteration, the worst-case complexity of Algorithm 4 can be written as $\mathcal{O}(S^4U^3U_0^2B_1B_2)$. It can be seen that while providing an approximate solution on the original problem, Algorithm 4 has a polynomial complexity, which translates into a good scalability. We note that an overhead is added when computing the algorithm at the central controller, which needs to happen when a new set of offloading requests is introduced after a certain time. Those requests will be recurring for a certain period of time as noted in Subsection 3.3.2.

It is worth noting that in spite of applying the well known SCA method for Algorithm 4, our contribution in this aspect lies in the non-trivial approximations and conversions performed to eventually propose the SCA-based algorithm, which are all customized to match the specific constraints and objective of our problem.

Parameter	Value
RBs for SCs communication B_1	5
RBs for MC communication B_2	5
RB bandwidth	4 MHz
Task input size d_{ij}	[60;100] Kb
Task computational density c_{ij}	[800;1000] CPU cycles/bit
Cloudlet computational capacity $\bar{F}_j$	[4;6] GHz
UE power threshold $\bar{P}_{ij}$	30 dBm
UE latency deadline $\bar{L}_{ij}$	[30;50] ms

Table 3.2: Simulation Parameters

3.5 Numerical Results

In this section, we present numerical results based on various simulations that are done with MATLAB software on a computer with Intel i7 processor. The default instance represents a heterogeneous network that consists of a set of $S = 3$ cloudlet-enabled SCs connected to an MC, having 2 UEs in the range of each SC and the MC. The channel gain h_{ijlb} follows an exponential distribution with mean 1. Both the beamforming vector norm $||\boldsymbol{w}||^2$ and the noise power level are normalized to 1. All remaining system parameters are presented in Table 3.2, noting that we aim to address a general class of applications without focusing on a particular one. The convergence criteria of Algorithm 4 is established when ϵ, i.e. the difference of objective value between $\Gamma^{(n)}$ and $\Gamma^{(n+1)}$ of the approximated problem, is $\epsilon \leq 10^{-3}$.

In Fig. 3.2, we study the convergence behavior of the SCA-based iterative algorithm, where the objective represents the total UEs energy consumption in Mj. It can be seen that the algorithm starts with a very high objective value, and then the objective rapidly decreases in a few iterations until convergence where the objective

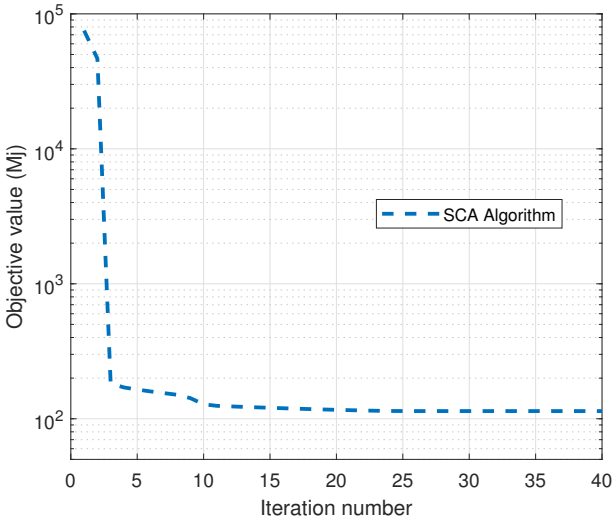


Figure 3.2: Convergence behavior for the SCA-based algorithm.

is equal to 114.022 Mj. This convergence behavior can be explained by the fact that the violation sub-objective starts with a very high value due to the big margin that exists initially between x_{ijb} and x_{ijb}^2 in (3.23b). Then, the effect of the violation sub-objective rapidly disappears when the values of x_{ijb} become closer to 1 and 0. After that, the focus is transferred to decreasing the energy sub-objective, which is done by updating the approximation points in each iteration until the algorithm convergence. Note that all subsequent results follow a convergence behavior that is similar to the one shown in Fig. 3.2.

In Fig. 3.3, we show a comparison between the proposed SCA algorithm and SCA-NBC which is similar to SCA algorithm, but no backhaul communication is performed for migrating SUEs' tasks. The objective value is presented as it varies with the SC cloudlets computational capacity for both cases. We use here the same

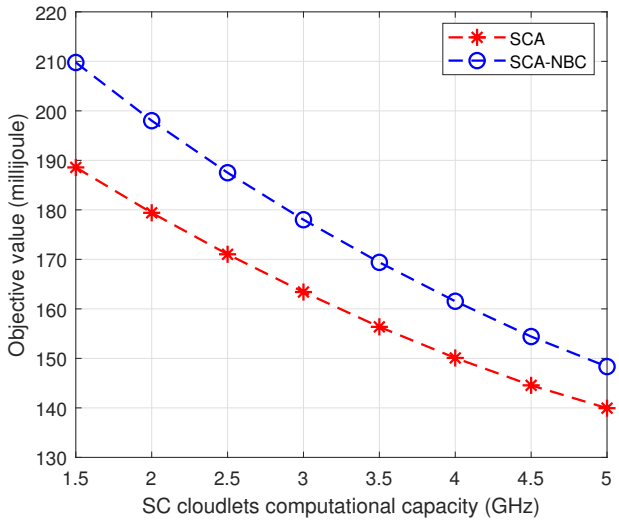


Figure 3.3: Comparison between the SCA and SCA-NBC where task migration to the MC is disabled

instance where two UEs coexist in the range of each of the three SCs. Three observations can be made here. First, the objective value decreases almost linearly with the increase of available computational capacity at the SC cloudlets. Second, the SCA algorithm always outperforms the SCA-NBC approach. Third, the margin between the two approaches reduces with the increase in the cloudlets computational capacity. The first observation is due to the increased availability of the computational resources. It allows for more task chunks to be offloaded and to be computed within the same latency bound, and thus decreasing the local computation energy. Also, lower computational capacity for SCs cloudlets leads to a higher computation latency for SUEs, forcing the upload latency to be smaller in order to keep up with the latency bound. This forces SUEs to increase their transmission power in order to meet that latency. Second, exploiting the wireless backhaul for migrating to the MC

cloudlet allows more task chunks to be offloaded from the local device (leveraging simultaneous sub-tasks transmission and computation) while respecting the latency deadline. Therefore, the local devices' energy will be decreased and thus affecting the total consumed energy. On the other hand, when SUEs' tasks cannot be migrated to the MC cloudlet, SUEs in this case are forced to compute their task locally to ensure the latency deadline is satisfied, but incurring more energy in the process. Third, as the availability of computational resources on the SC increases, limiting sub-tasks migration to the MC cloudlet becomes less detrimental, since more resources are already available for serving the tasks load with a decreased need to leverage the MC cloudlet. In this figure, we can see an improvement of up to 11.2% by leveraging the MC cloudlet computation capability for SUE users. It is worth noting that the improvement can become much more significant when 1) the local UEs' computation energy is of a higher magnitude, or when 2) the UEs have a more restricted latency requirement, since leveraging tasks partitioning and simultaneous transmission to the SC and MC would increase the chance of respecting the latency deadline while decreasing the load on the local device.

In Fig. 3.4, we study the variation of the objective value with respect to the SC cloudlets computation capacity and UEs' latency bound. Here, we make two observations. First, increasing the latency bound leads to a lower UEs' energy consumption. Second, decreasing the capacity of SC cloudlets will incur a higher UEs' energy consumption. First, increasing the latency bound allows UEs to occupy less computational resources and thus increasing the resources' availability on the SC and MC cloudlets, where more tasks chunks can then be offloaded. Also, this allows

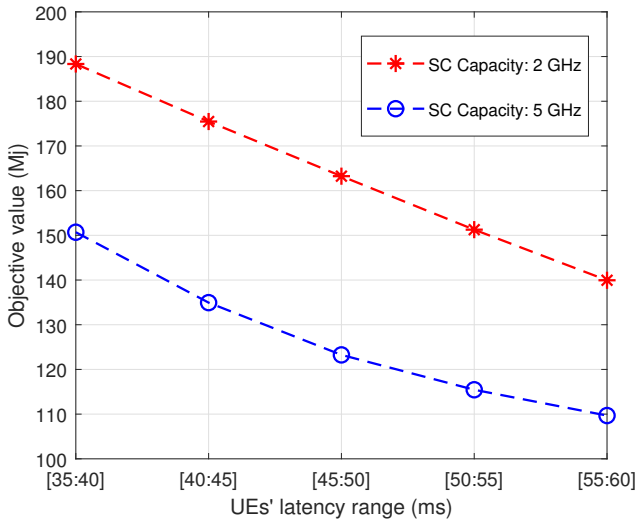


Figure 3.4: Variation of the objective value with respect to the cloudlets computation capacity and the UEs latency bound.

decreasing the SUEs' transmission power in a way that minimizes the overall consumed energy. However, when the latency bound is too low, devices will be more inclined to compute tasks chunks locally, which will incur a higher energy. On the other hand, the second observation can be explained similar to the discussion for Fig. 3.3.

In Fig. 3.5, we compare our SCA-based algorithm with two other approaches: the Non-Interference (SCA-NI) and the Worst-Case Interference (SCA-WCI). The SCA-NI considers the case where a given RB is not being used by any other SUE in the neighboring cells, so the interference on this block will amount to 0. This means that the obtained solution might be infeasible when considering the real interference, since the upload latency would increase and might bypass the latency deadline for some SUEs. On the other hand, SCA-WCI considers the highest possible interference

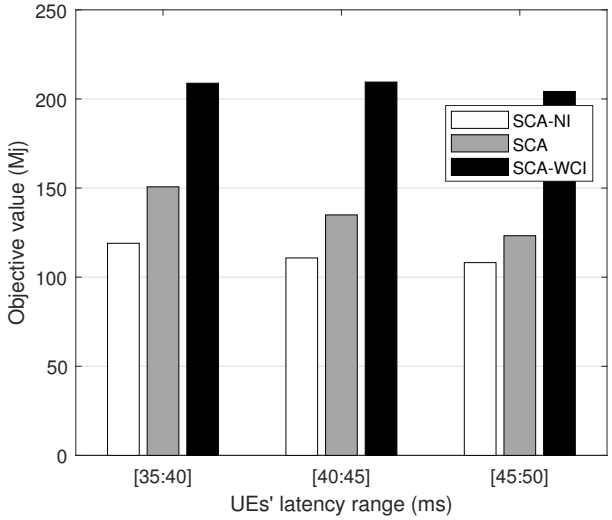
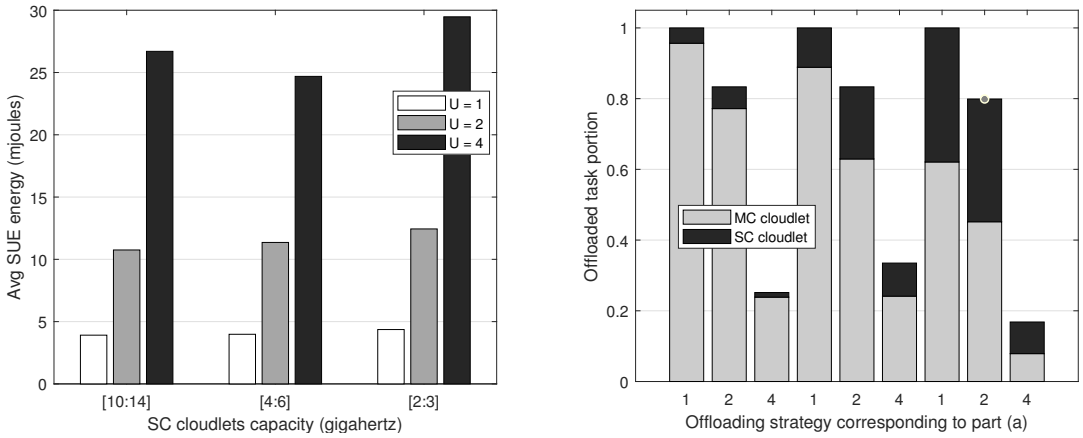


Figure 3.5: Comparison between the SCA and the BCI and WCI approaches with respect to the latency bound.

on each RB, which results from the maximum transmission power over the strongest channel in each neighboring cell. This means that the obtained solution overestimates the effect of interference, which would lower the resources utilization and push more tasks chunks to be computed locally and thus increasing the objective value. Both SCA-NI and SCA-WCI have lower complexity than the proposed algorithm, which is due to discarding the effect of inter-cell interference in equation (3.2a) making it a constant. As it can be seen, the BCI and WCI approaches give lower and upper bounds, respectively, on our approximate solution with a significant gap. Adopting the SCA-WCI approach allows the algorithm to have a better performance and scalability, but with a cost of a higher objective and a decreased solution quality. As it can also be seen, the objective value generally increases when the latency bound decreases as previously discussed, especially for the SCA-based algorithm. For the



3.6.a Variation of the average SUE energy with respect to the users density and cloudlets capacity.

3.6.b Variation of the offload strategy with respect to the users density and cloudlets capacity.

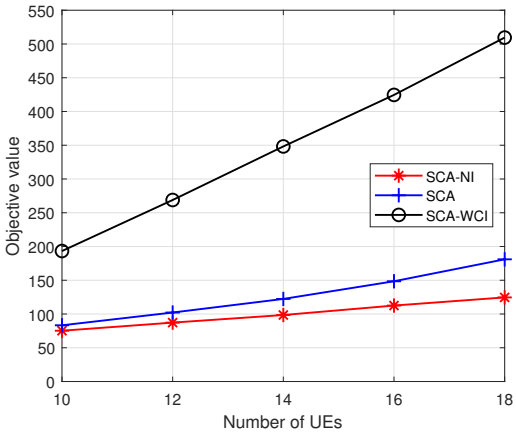
Figure 3.6: Effect of users density in the SCs on the average energy and offload strategy.

other relaxed approaches, they are less sensitive to latency bound changes. This is mainly because the interference amount is identical in both cases, causing the transmission rate to depend only on the signal strength, and hence causing it to change slightly. On the other hand, in the SCA case the amount of interference will also be diminished, adding more effect on the transmission rate and hence energy. Also, note that the change in the objective value will generally be more significant for bigger instances.

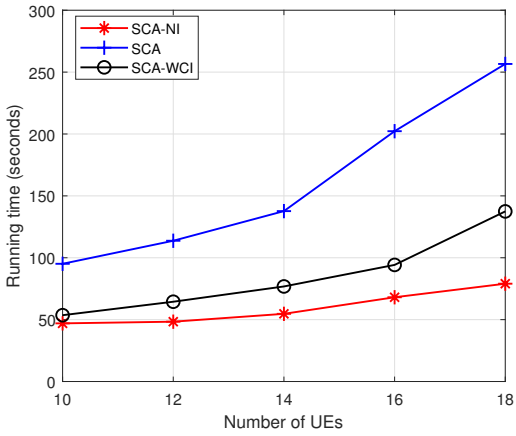
In Fig. 3.6, we study the effect that SC cloudlets capacity and SC users density have on the average energy and offloading strategy. Fig. 3.6.a shows a significant increase of the SUE average energy whenever increasing the number of users with the range of an SC. This is due to the increased competition causing elevated demand of computation and communication resources, causing SUEs to compute more task

chunks locally which incurs more energy. Also, the limited amount of resources increases the upload latency where SUEs have to consume more power to respect the latency bound, which increases the upload energy. In addition, it can be seen that limiting the availability of SC computational resources has little or no effect on the UEs' average energy consumption. This is because SUEs will be pushed to utilize the MC cloudlet which by itself does not incur additional upload energy in case resources are available. We note that limiting or prohibiting the utilization of the MC cloudlet would incur increased energy consumption for the SUEs as observed in 3.3. On the other hand, in Fig. 3.6.b, the corresponding offloading strategy is illustrated for each case, which represents the average strategy among UEs. As it can be seen, increasing the SUEs' density withing the SCs pushes SUEs to offload a fewer percentage of their task (e.g. from around 0.95 to 0.2 in the first case). This is because of the mentioned increased resources demand that forces SUEs to compute their task locally in order to respect the latency bound. In addition, it can be seen that limiting the SC computational resources generally pushes more task chunks to be computed on the MC cloudlet. As seen in Fig. 3.6.a, this does not have a significant effect on the average SUE energy consumption, but the change in the offloading strategy can be noticeable in this case, which indicates the advantage of utilizing the MC cloudlet through the wireless backhaul when resources are available there.

In Fig. 3.7, we study the scalability and performance of our SCA algorithm along with the two other SCA-NI and SCA-WCI simplified approaches. As can be seen in Fig. 3.7.a, the objective value clearly increases in the three approaches whenever the



3.7.a Variation of the objective with respect to the number of UEs.



3.7.b Variation of the algorithm running time with respect to the number of UEs.

Figure 3.7: Variation of the objective and algorithm running time with respect to the number of UEs.

number of UEs goes up. This is because more UEs have to compute their task and hence will consume energy, which results from either local device execution or from uploading the task to the associated SC through the access network. It can be seen that the variation of the objective value with respect to the number of UEs is close to linear. Also, the SCA-NI and SCA-WCI always give lower and upper bounds on the SCA solution, which is explained in Fig. 3.5. On the other hand, in Fig. 3.7.b, we study the performance of the three approaches while varying the number of UEs. It can be seen that the running time almost increases linearly with the number of UEs, which is due to the polynomial complexity of the SCA algorithm that is previously explained. However, the performance advantage of both SCA-NI and SCA-WCI can be clearly seen in that figure, where the running time is much lower than in that of the SCA-based algorithm. This is because both approaches are simplified versions

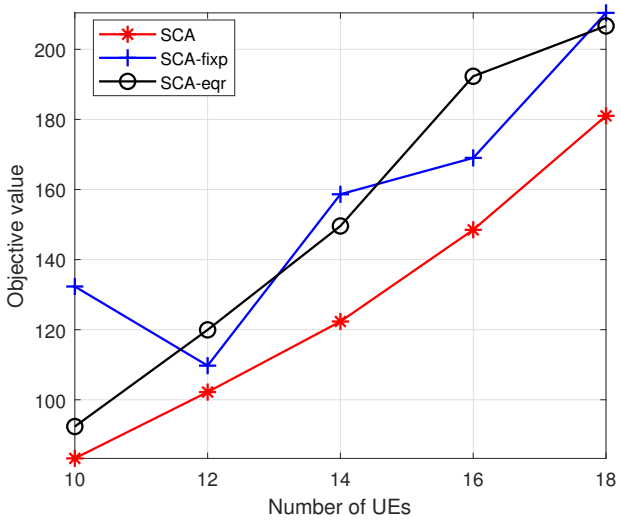


Figure 3.8: Comparison between the SCA algorithm and other simplified approaches.

that ignore the complexity of the interference aspect while computing the uplink rate. However, Fig. 3.7.a shows that utilizing SCA-WCI leads to a much worse solution. This is due to assuming the interference resulting from the highest possible power on the RB with the strongest channel in each SC. We note that the solution from the SCA-WCI approach can be evaluated within problem 3.24 which leads to a closer upper bound, noting that the solution from the SCA-NI might be infeasible for 3.24 as already discussed. We note that the NOs have to consider increasing SC cloudlets' capacity in order to accommodate the increased number of UEs' requests and help in decreasing the UEs energy.

In Fig. 3.8, we compare our SCA-based algorithm with two other approaches, namely SCA-fixp, where the transmission power for the UEs is fixed to 23 dBm, and SCA-eqr, where the SUEs' uplink transmission rate is divided equally among the tasks being offloaded to the SC cloudlet (i.e. r^{ue}_{ij2}), and the tasks being offloaded

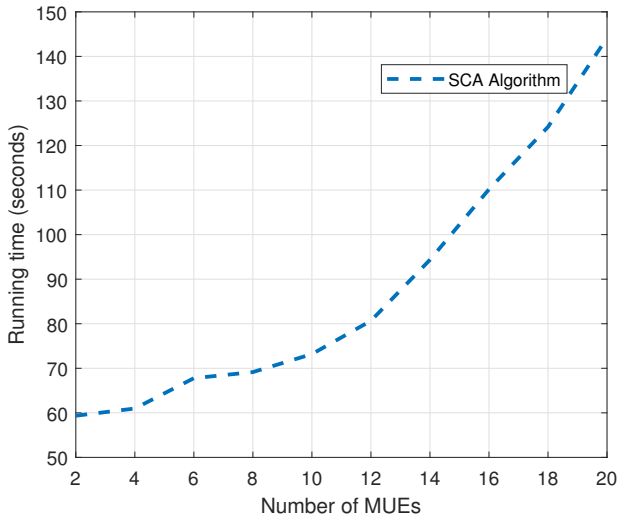


Figure 3.9: Variation of the algorithm running time with respect to the number of macro-cell users.

to the MC cloudlet ((i.e. r_{ij3}^{ue})). The comparison is done while varying the number of UEs in the network. As it can be seen, the SCA-based algorithm, including all the optimization parameters, leads to the best solution in all cases. On the other hand, SCA-fixp leads to a worse solution since the transmission power is not being optimized, and SCA-eqr also incurs an inferior solution since the UEs' uplink rate is not optimally partitioning for each of the data chunks being sent to the SC and MC cloudlet. This shows the importance of optimizing the mentioned optimization parameters in the problem, along with the allocation of computation and communication resources which must be optimized in order to obtain the best solution for the low-complexity SCA-based algorithm.

In Fig. 3.9, we study the algorithm scalability specifically by examining the runtime as it varies with respect to the number of MUEs. This is done while using

another instance with parameters matching the ones in Table 3.2. As it can be seen, there is a small difference in the running time when having a small number of MUEs. For bigger numbers, the increase in the running time with respect to the number of MUEs is almost linear. This means that the algorithm runtime increases almost linearly with respect to the addition of more and more number of MUEs, which reflects the polynomial complexity of the algorithm.

3.6 Conclusion

In this chapter, we studied the computation offloading problem in a MEC-enabled heterogeneous network, where we optimized the computation and wireless communication resources' allocation in the RAN and backhaul, considering SCs that can migrate SUEs' tasks for computation on the MC cloudlet through a wireless backhaul. Our objective was to minimize the total UEs' energy consumption resulting from local computation and computational tasks' transmission while respecting the UEs' latency deadline. We presented a low-complexity SCA-based algorithm for providing an approximate solution on the original non-convex problem, while achieving a high scalability. Through numerical results, we demonstrated the efficiency and superiority of our solution, and performed multiple simulations following different scenarios.

Chapter 4

Latency and Reliability Aware Computation Offloading via UAV-mounted Cloudlets in IoT Networks[1]

The rigidity of MEC and its susceptibility to infrastructure failure and weak wireless signals would prevent from effectively provisioning computation offloading with strict latency and reliability requirements. UAVs have been proposed for providing a flexible edge computing capability through UAV-mounted cloudlets, harnessing their unique advantages such as mobility, low-cost, and line-of-sight communication.

[1]This chapter has been published in IEEE Transactions on Communications [84].

However, UAV-mounted cloudlets may have failure rates that would impact mission-critical applications, necessitating a novel study for the provisioned reliability considering the UAV nodes' failure rate and tasks redundancy. In this work, we investigate the novel problem of UAV-aided latency and reliability aware computation offloading, which would enable modern IoT services with their strict requirements. We aim at maximizing the rate of served requests, by optimizing the UAVs' positions, the offloading decisions, and the allocated resources, while respecting the stringent latency and reliability requirements. To do so, the problem is divided into two phases; the first being a planning problem for optimizing the long-term placement of UAVs, and the second an operational problem to make optimized offloading and resource allocation decisions with constrained UAVs' energy. We formulate both problems associated with each phase as non-convex MI-NLPs, and due to their non-convexity, we propose a two-stage approximate algorithm where the two problems are transformed into approximate convex programs. Further, we approach the problem considering the task partitioning model which will be prevalent in 5G networks. Through numerical analysis, we demonstrate the efficiency of our solution considering various scenarios, and compare it to other baseline approaches.

4.1 Introduction

Despite the advantages brought by MEC which has been considered excessively in the literature for various problems seeing its efficiency in provisioning low-latency computation offloading, MEC has limitations which makes it difficult to cater for

the flexibility and reliability required by emerging services and use cases in various scenarios. Take for instance an automated factory where custom operations with real-time sensing and objects identification need to be spontaneously carried out by CPSs at some periods. A scenario where an unexpected increase in the number of user devices could occur. A system of cameras/sensors in a smart city that are installed for performing traffic coordination. A scenario with a CN characterized by network irregularities and intermittent or no connectivity, such as in the situations of disaster and emergency. An under-served area such as a rural or developing region where a dense network infrastructure is not provisioned for various costs reasons [85]. Or a network with weak wireless signals in the RAN caused by obstacles and high network load. In those situations, a ground MEC would be rigid, over-provisioned, affected by weak channels, and unable to scale and adapt to the dynamic and spontaneously changing environment.

Leveraging the advantages of UAVs which have already been studied as aerial-BSs in cellular networks, UAV-mounted cloudlets have been proposed for providing efficient edge computing for IoT services with stringent latency and reliability requirements [86, 87]. The motivation for using UAV-mounted cloudlets is their various advantages over terrestrial cloudlets in terms of low-cost, mobility, flexibility, scalability, and adaptive altitude [7, 13]. UAVs can be deployed on-demand for a specific duration when needed, in addition to having a higher probability of LoS communication, which increases the transmission rates [88]. Therefore, in 5G networks and beyond, it will be common to dispatch a set of UAV-mounted cloudlets for providing offloading services with stringent requirements, such as in the case of

emergency operations [89].

However, enabling MEC via UAV-mounted cloudlets is coupled with challenges such as the UAVs' placement difficulty, and limited resources and battery level, although several studies have addressed UAV-aided computation offloading problems where the UAVs' positions, offloading decisions, and resources allocation were optimized [90, 91]. Still, since UAVs need to support mission-critical applications where unexpected disturbances must be minimized or completely prevented, their susceptibility to node failure which will impact the computation reliability for the provided services must be taken into account [17]. Thus, a new measure for the reliability taking UAVs' failure rate must be considered, where redundancy can be exploited by computing the offloaded tasks on multiple nodes in parallel [92, 93, 10]. In this case, however, multiple challenges arise when it comes to the complexity of the computation and communication aspects and that of the UAVs' positioning, while considering the variations in the incoming requests and avoiding resources over-provisioning.

In this chapter, we investigate a novel UAV-aided latency and reliability aware computation offloading problem considering tasks redundancy and UAV failure rates for avoiding service disruptions and enabling services with stringent latency and reliability requirements for mission-critical applications in IoT networks. Our proposed problem is divided into two phases: a planning problem is first proposed for optimizing the UAVs' positions considering the long-term computation offloading where the task arrival is modeled using a Poisson process. Then, a second problem is proposed for the offloading and resources allocation decisions which are optimized for responding to the specific task requests present in the particular time slot where the

problem is being solved. Our solution aims at maximizing the rate of served IoT incoming offloading requests, while guaranteeing the tasks' required latency and reliability, and respecting the UAVs' available energy levels, by optimizing the UAVs' positions, the UEs-to-UAV association, and the allocated computation and communication resources considering both LoS and non-LoS components.

4.1.1 Novel Contributions

The contributions of this chapter can be summarized as follows:

1. We model the two problems and mathematically formulate them as non-convex MI-NLPs.

2. Due to the non-convexity of the proposed problems, we perform customized conversions to transform them into approximate SOCPs. We then propose an efficient customized algorithm for solving the overall problem based on the SCA method [41], which provides an approximate solution by iteratively solving until convergence.

3. We study the offloading problem considering the task partitioning model, where a task can be subdivided and computed in parallel on multiple nodes, which will be a prevalent model in 5G networks [19].

4. We demonstrate the effectiveness of our proposed solution through numerical results, and study the achieved gains for different use cases.

The remainder of this chapter is structured as follows. Section 4.2 presents the related work. Section 4.3 presents an illustrative example for our problem. Section

4.4 presents our system model and mathematically formulate the two non-convex problems. In Section 4.5, we propose a simplified solution approach and transform the non-convex problems into a more tractable form, where we present the iterative SCA-based algorithm. Section 4.6 presents the offloading problem adapted to the task partitioning model. Section 4.7 contains the numerical results for our proposed solution. Finally, section 4.8 concludes the chapter.

4.2 Literature Review

The following papers studied latency and reliability problems in the MEC computation offloading context. The authors in [94] studied the reliability problem for enabling industrial control processes through MEC, where different nodes and communication failure scenarios are considered, and a low-complexity algorithm was proposed for minimizing the operational costs. [47] minimized the servers' transmission power using a Lyapunov stochastic optimization technique, by studying the latency and reliability efficiency in a MEC system with a set of users communicating with ground servers. The authors in [95] studied the reliability in distributed edge clouds where a reliability indicator is introduced for each edge node, and computational tasks are then replicated to specific computing nodes. [77] studied the trade-off between latency and reliability in a MEC system, where the end-to-end latency and the failure probability of computation offloading is minimized while considering the transmission reliability resulting from the bit error rate, where task partitioning is also considered. The heuristic search, reformulation linearization technique, and

semi-definite relaxation approaches were used to solve the obtained non-convex problem. [92] optimized the placement of VNFs for minimizing the intra-communication latencies between them, where redundant computation paths were used for guaranteeing the services' reliability requirements. [96, 97] used a task allocation strategy to enhance the system reliability. While the authors in [96] utilized a solution approach that combines the power of the simulated annealing, genetic algorithm, with a fast problem specific local search heuristic algorithm, [97] addressed their difficult problem using the hybrid particle swarm optimization algorithm.

The following are papers that tackled important problems in the context of UAV-aided MEC computation offloading. [98] studied the deployment of one UAV for serving the computation offloading requests for a network of ground users. There, the UAV trajectory, the users' transmission power, and the computational resource allocation were optimized using an SCA-based algorithm that is used to solve the complex problem. In [99], the authors studied the use of a UAV-mounted cloudlet for supporting the ground MEC system in providing computation offloading service for stationary IoT devices, where the tasks latency and UAV's energy were minimized. This would be useful in scenarios where ground MEC servers could be sparse or the amount of requests could be very high. In [100], the authors studied the latency and energy performance of a UAV-aided edge/fog computing framework in a smart IoT community environment where a set of UAVs are used for providing augmented reality service for a set of ground users and IoT devices. There, the 3D placement and trajectory of two types of UAVs in addition to the devices' transmission power

were optimized. [101] studied the computation offloading problem to a set of UAV-mounted cloudlets where a low-complexity algorithm is proposed for minimizing the sum power of users and UAVs by optimizing the users' association, the resources' allocation, and UAVs' positioning. [90] minimized the devices' energy consumption by jointly optimizing the bit allocation and the UAVs' trajectory while guaranteeing the users' QoS requirements. [91] studied the computation rate maximization problem for a UAV-aided MEC wireless powered system while respecting the energy harvesting and the UAVs' speed constraints. Very few papers addressing UAV-aided computation offloading problems have considered both latency and reliability aspects. Furthermore, none have taken the UAV failure as a reliability measure for enabling services with stringent requirements where task redundancy is performed. The authors in [98, 99, 90] utilized the SCA approach to provide an efficient solution for their non-convex problem. The authors in [101] proposed a fuzzy c-means clustering-based algorithm to solve their 3 sub-problems iteratively. The authors in [91] proposed a two-stage algorithm and a three-stage alternative algorithm in order to solve their non-convex problem.

In our previous work [102], we provided a first and rather simplistic attempt to guarantee the latency and reliability of the offloaded tasks to a UAV-aided MEC system. Our previous work overlooked the long-term consideration of computation offloading and considered solving for one state of the tasks' requests. Further, we did not account for the non-LoS scenarios which limits the scope of the problem, and also did not consider the energy limitation of the UAVs. In this work, we extend the previous work by considering more realistic scenarios that comply with the modern

5G services in IoT networks. First, we propose a planning problem where the UAVs' positions are optimized considering the long-term knowledge of the average arrival of tasks. Then, a separate problem is solved for the offloading and resource allocation decisions with UAVs energy constraints, which targets the specific requests data in the particular time slot. Second, we place the UAVs considering both LoS and non-LoS components. Third, we account for multiple scenarios in which IoT devices can opt to fully or partially offload their tasks. We also explore the trade-off between latency and reliability, and between reliability and resource utilization in the studied system. The extension in this work would contribute towards enabling modern services with stringent latency and reliability requirements for mission-critical applications in IoT networks with no service disruptions.

4.3 Problem Description

We describe the problem of UAV-aided latency and reliability aware computation offloading in Fig. 4.1, with two IoT devices UE_1 and UE_2 that are requesting to offload their tasks T_1 and T_2 to two UAV-mounted cloudlets U_1 and U_2. The values for the computation capability F (GHz), reliability R, latency L (ms), are indicated in the figure. The task-to-UAV assignments are represented by two numbers denoting the allocated computational resources (GHz), and wireless bandwidth (MHz). The achieved latency is equal to the sum of upload and computation latencies, while the achieved reliability is equal to the reliability of the UAV where the task is offloaded.

In part (a), each task is assigned to the UAV having the better wireless channel

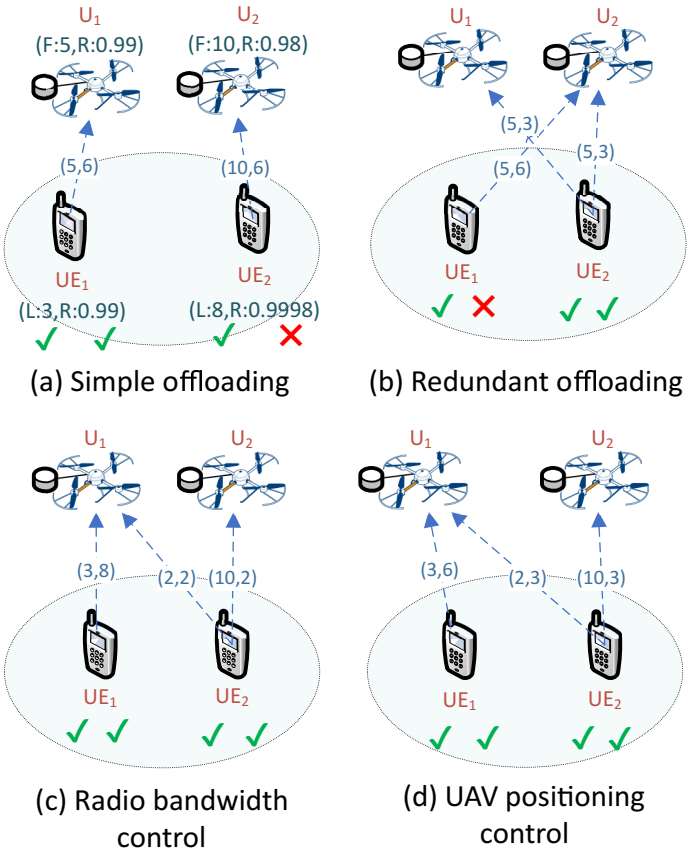


Figure 4.1: Illustrative example.

and is consuming their available computational resources while respecting the UAVs' energy limit, and the wireless bandwidth is equally divided among both tasks with 6 MHz each. While T_2 cannot be admitted due to is reliability not being guaranteed with U_2 ($0.98 < 0.9998$), T_1's reliability is met with U_1 ($0.99 = 0.99$). In part (b), redundancy is exploited by computing T_2 on both UAVs in parallel, where the achieved reliability for T_2 becomes $1 - (1 - R_{U_1})(1 - R_{U_2}) = 0.9998$. To also meet T_2's latency when offloading to U_1, T_2 has to consume 3 GHz from U_1. This however decreases the computational resources available for T_1 on U_1 which increases T_1's

computation latency, preventing its admissibility. Hence, T_1 is offloaded instead to U_2 which has more computational capacity that can meet its latency requirement. However, U_2's reliability is not enough for guaranteeing the reliability of T_1, and hence T_1 still cannot be admitted. In part (c), T_1 is associated back to U_1 where the transmission latency for T_1 is lowered by increasing the allocated wireless communication bandwidth from 6 to 8 MHz. This change would meet the latency requirement of T_1 which allows its admission. However, this decreases the available bandwidth for T_2 where we assume T_2's latency requirement is still respected. In part (d), we exploit the UAVs' positioning where U_1's position is adjusted to enhance the wireless channel of T_1 for decreasing its transmission latency by increasing its transmission rate, assuming T_2's latency is still guaranteed. Thus, the tasks' association, their allocated resources, and the UAVs' positioning, are decisions that influence the number of admitted tasks, so they must be optimized to maximize the system efficiency while preventing resource over-provisioning.

4.4 System Model

4.4.1 Spatial Model

As illustrated in Fig. 4.2, we consider a system where a set of N IoT devices denoted by $\mathcal{N} = \{1, 2, ..., N\}$ are provisioning their service by recurrently generating computation offloading requests across time $\mathcal{T}$ to a set of M UAV-mounted cloudlets denoted by $\mathcal{M} = \{1, 2, ..., M\}$. We assume that each UAV-mounted cloudlet can be modeled as a server, where the arrival of tasks from IoT device i to this server

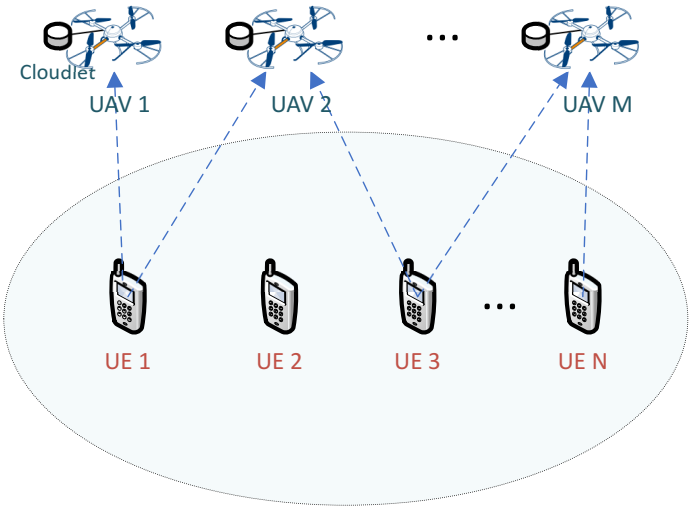


Figure 4.2: System model.

follows a Poisson process with arrival rate λ_i. Each task of UE i is represented by a tuple $\{d_i, c_i, \bar{L}_i, \bar{R}_i\}$, concatenating the task input size d_i (Kb), computational demand density c_i (CPU cycles/bit), required latency $\bar{L}_i$ (ms), and required reliability $\bar{R}_i$. We denote by o_{ij} a binary decision variable indicating if UE i's task is offloaded to UAV j, where tasks of UE i can be offloaded to multiple UAVs in parallel for meeting their required reliability. We adopt a three-dimensional Cartesian coordinate system measured in meters, where a UE i is located in the xy-plane at position $\bar{\boldsymbol{p}}_i = (\bar{x}_i, \bar{y}_i, 0)$. In addition, the position of each UAV j is optimized as $\boldsymbol{p}_j = (x_j, y_j, H)$ where H is a constant representing the UAV height.

Ground UEs need the GtA channel for offloading their tasks to the UAV cloudlets. Let $h_{ij}(\theta_{ij})$ denote the GtA channel power gain between UE i to UAV j. $h_{ij}(\theta_{ij})$ is

composed of both LoS and non-LoS links [88, 103], and can be defined as:

$$\begin{aligned} h_{ij}(\boldsymbol{p}_j, \theta_{ij}) &= P_{\text{LoS}ij}(\boldsymbol{p}_j)\theta_{ij}g_0 + (1 - P_{\text{LoS}ij}(\boldsymbol{p}_j))\theta_{ij}g_0\kappa \\ &= \hat{P}_{\text{LoS}ij}(\boldsymbol{p}_j)\theta_{ij}g_0 \end{aligned} \tag{4.1}$$

where g_0 is the path loss at a reference distance of 1 meter, and $\kappa < 1$ is the non-LoS additional attenuation factor. $P_{\text{LoS}ij}(\boldsymbol{p}_j)$ denotes the LoS probability and $\hat{P}_{\text{LoS}ij}(\boldsymbol{p}_j) = P_{\text{LoS}ij}(\boldsymbol{p}_j) + (1 - P_{\text{LoS}ij}(\boldsymbol{p}_j))\kappa$ can be interpreted as a regularized LoS probability by taking into account the effect of non-LoS occurrence with the additional attenuation factor κ. θ_{ij} represents the path-loss coefficient calculated as:

$$\theta_{ij} = \left(\sqrt{H^2 + ||\boldsymbol{p}_j - \bar{\boldsymbol{p}}_i||^2}\right)^{-2} \tag{4.2}$$

Also, $P_{\text{LoS}ij}(\boldsymbol{p}_j)$ depends on the environment and the elevation angle between the UAV and the UE and is calculated as [88]:

$$P_{\text{LoS}ij}(\boldsymbol{p}_j) = \frac{1}{1 + a\exp\left(-b\left(\arctan\left(\frac{H}{||\boldsymbol{p}_j - \bar{\boldsymbol{p}}_i||}\right) - a\right)\right)} \tag{4.3}$$

where (a, b) are environment-dependent, e.g. equal to (9.61, 0.16) in urban environments [104].

4.4.2 Communication Model

We assume the wireless radio spectrum is shared among all UAVs and GtA communications, with a bandwidth of B MHz. We do not consider the downlink communication, since the task output size is in general much smaller than the task input size [57]. We denote by b_{ij} a decision variable indicating the portion of the bandwidth allocated to UE i for communicating with UAV j. Thus, the achievable transmission rate in Mbps is expressed as:

$$R_{ij}(\boldsymbol{p}_j, \theta_{ij}, b_{ij}) = b_{ij} B \log_2 \left(1 + \frac{P_{ij} h_{ij}(\boldsymbol{p}_j, \theta_{ij})}{b_{ij} B N_0}\right) \tag{4.4}$$

where P_{ij} is the transmission power from UE i to UAV j, and N_0 is the noise power spectral density.

4.4.3 Latency and Reliability Models

Thus, the upload latency from UE i to UAV cloudlet j which depends on the UAV position $\boldsymbol{p}_j$ and the assigned wireless communication resources b_{ij}, can be computed as follows:

$$L^{\mathrm{u}}_{ij}(\boldsymbol{p}_j, \theta_{ij}, b_{ij}) = \frac{d_i}{R_{ij}(\boldsymbol{p}_j, \theta_{ij}, b_{ij})} \tag{4.5}$$

We consider the node reliability associated with the failure probability of each UAV cloudlet j [94, 95], where the server node failures are mainly caused by either software failures or hardware failures such as hard disk, memory and RAID controller failures [105, 106]. We denote by ϕ_j as the reliability indicator for UAV cloudlet j,

which can be obtained after estimating the occurrence probability of failure scenarios through statistical means based on the cloudlets' historical failure pattern and maintenance records, such as data indicating the mean time between failures and the mean time to repair for each node [107, 108]. This is in fact identical to how the failure probabilities are usually computed for server nodes in cloud data centers [109]. For guaranteeing the tasks' reliability, redundancies are performed where the task can be computed on multiple cloudlets in parallel. The achieved reliability for task $i \in \mathcal{N}$ is given by:

$$\Phi_i(\boldsymbol{o}_i) = 1 - \prod_{j \in \mathcal{M}} (1 - o_{ij}\phi_j) \tag{4.6}$$

Our approach to solve the proposed problem is divided into two parts. First, we solve a planning problem for optimizing the stationary positioning of UAVs across the offloading duration. This planning decision will be informed and guided by the long-term knowledge of the average units for the tasks' arrival and computation request. Then, we solve a second problem for optimizing the offloading and resource allocation decisions in a given time slot considering the UAVs' energy limit, where the optimized decisions are tailored for the specific UE requests at that particular time slot. Thus, in the following subsection, we propose to decouple the design of UAVs' location $\boldsymbol{p}$, $\boldsymbol{\theta}$ from the remaining variables. Then, after determining the long-term UAVs' positions, another problem is solved to optimize the other variables in a given time slot based on the accurate $\hat{P}_{\mathrm{LoS}ij}(\boldsymbol{p}_j)$.

4.4.4 UAV-aided MEC Planning Problem

In this subsection, we first formulate the UAV-aided MEC planning problem, where the UAVs' position is optimized along with the other variables.

It is observed that $R_{ij}(\boldsymbol{p}_j, \theta_{ij}, b_{ij})$ in (4.5) depends on the UAV position $\boldsymbol{p}_j$ not only via the UAV-UE distance θ_{ij}, but also via the regularized LoS probability $\hat{P}_{\mathrm{LoS}ij}(\boldsymbol{p}_j)$. This makes $R_{ij}(\boldsymbol{p}_j, \theta_{ij}, b_{ij})$ intractable for optimizing the UAVs' position. Thus, we denote by $\tilde{h}_{ij}(\theta_{ij}) = \theta_{ij} g_0$ the GtA channel gain between UE i and UAV j based on the FSPL model where $P_{\mathrm{LoS}ij}(\boldsymbol{p}_j) = 1$. Then, the achievable transmission rate from UE i to UAV j based on the FSPL model, is expressed as

$$\tilde{R}_{ij}(\theta_{ij}, b_{ij}) = b_{ij} B \log_2 \left(1 + \frac{P_{ij} \tilde{h}_{ij}(\theta_{ij})}{b_{ij} B N_0}\right) \tag{4.7}$$

Then, the upload latency from UE i to UAV cloudlet j is updated as follows:

$$\tilde{L}^{\mathrm{u}}_{ij}(\theta_{ij}, b_{ij}) = \frac{d_i}{\tilde{R}_{ij}(\theta_{ij}, b_{ij})} \tag{4.8}$$

Because the UAVs' optimized positions are static throughout the offloading duration, we consider the average quantities for the tasks' arrival and size. Thus, we consider each UAV-mounted cloudlet carrying the computation service to be modeled as an $M/M/1$ queue, where the computation latency on cloudlet j is given by:

$$\tilde{L}^{\mathrm{c}}_{j}(\boldsymbol{o}, \boldsymbol{\alpha}) = \frac{1}{\frac{F_j}{\bar{c}} - \sum_{i \in \mathcal{N}} o_{ij} \alpha_i \lambda_i} \tag{4.9}$$

where the service follows an exponential distribution with service rate of $\frac{F_j}{\bar{c}}$, and

$\bar{c} = \frac{\sum_{i \in \mathcal{N}} \lambda_i c_i}{\lambda_i N}$ is the average task computational demand that is normalized to the tasks' arrival rate λ_i.

We aim at maximizing the rate of admitted tasks load, while respecting the tasks' required latency and reliability, by optimizing the UAVs' static position $\boldsymbol{p}, \boldsymbol{\theta}$ jointly with the variables $\tilde{\boldsymbol{O}}, \tilde{\boldsymbol{B}}$. By denoting α_i as the portion of admitted load for each UE $i \in \mathcal{N}$, the UAV-aided latency and reliability aware MEC planning problem $\mathcal{P}_{1a}$ is formulated as:

$$\mathcal{P}_{1a}: \underset{\substack{\boldsymbol{p},\boldsymbol{\theta},\boldsymbol{\alpha},\\ \tilde{\boldsymbol{O}},\tilde{\boldsymbol{B}}}}{max} \quad \sum_{i \in \mathcal{N}} \alpha_i \lambda_i \tag{4.10a}$$

$$\text{s.t.} \quad \theta_{ij}^{-1} \geq (x_j - \bar{x}_i)^2 + (y_j - \bar{y}_i)^2 + H^2, \ \forall \{i,j\} \in \{\mathcal{N}, \mathcal{M}\} \tag{4.10b}$$

$$\Phi_i(\boldsymbol{o}_i) \geq \bar{R}_i, \ \forall i \in \mathcal{N} \tag{4.10c}$$

$$o_{ij} \left(\tilde{L}_{ij}^{\mathrm{u}}(\theta_{ij}, b_{ij}) + \tilde{L}_j^{\mathrm{c}}(\boldsymbol{o}, \boldsymbol{\alpha}) \right) \leq \bar{L}_i, \ \forall i \in \mathcal{N} \tag{4.10d}$$

$$\frac{F_j}{\bar{c}} \geq \sum_{i \in \mathcal{N}} o_{ij} \alpha_i \lambda_i, \ \forall j \in \mathcal{M} \tag{4.10e}$$

$$\sum_{j \in \mathcal{M}} o_{ij} \geq 1, \ \forall i \in \mathcal{N} \tag{4.10f}$$

$$\sum_{j \in \mathcal{M}} \sum_{i \in \mathcal{N}} b_{ij} \leq 1 \tag{4.10g}$$

$$\boldsymbol{p}_j, \theta_{ij}, b_{ij} \geq 0, o_{ij} \in \{0,1\} \tag{4.10h}$$

where (4.10a) maximizes the admitted tasks load. (4.10b) is the relaxed version of (4.2), knowing that (4.10b) becomes equality at optimality. Constraint (4.10c) guarantees the tasks' reliability requirement, and along with the objective will guarantee "just enough" reliability so that resources can be allocated for other tasks.

Constraint (4.10d) guarantees the total latency of each offloaded task replica (composed of both the upload and computation latencies) must respect the task latency deadline. Here, the task completion latency will depend on the latency of the last completed task replica. Constraint (4.10e) is the UAV cloudlets' queue stability constraint. Constraint (4.10f) makes sure a task load is offloaded to at least one cloudlet. Constraint (4.10g) makes sure the wireless radio bandwidth B is respected, and constraint (4.10h) is for the integrality and non-negative conditions.

It is observed that solving (4.10) is very difficult due to its non-convex nature that is caused by the existence of the non-convex non-concave terms: constraint (4.10b) which is non-convex with respect to θ_{ij}, constraint (4.10c) which is non-convex with respect to $\boldsymbol{o}_i$, and constraint (4.10d) which is non-convex with respect to θ_{ij}, b_{ij}. Also, the binary-related constraints in (4.10h) increase the complexity of (4.10) more. Here, the purpose of variables $\tilde{\boldsymbol{O}}, \tilde{\boldsymbol{B}}$ is to be jointly solved with $\boldsymbol{p}$, $\boldsymbol{\theta}$. After solving problem (4.10) and obtaining the solution $\boldsymbol{p}^\star, \boldsymbol{\theta}^\star$, the UAVs will move to the locations corresponding to $\boldsymbol{p}^\star$, where the real associated $\hat{P}_{\mathrm{LoS}ij}(\boldsymbol{p}_j)$ can be obtained.

4.4.5 UAV-aided MEC Offloading and Resources Allocation Problem

After the position of the UAVs is optimized in the planning phase, we present in this subsection the UAV-aided MEC offloading and resources allocation problem, where the variables $\boldsymbol{o}, \boldsymbol{a}, \boldsymbol{f}, \boldsymbol{b}$ will be optimized for the purpose of maximizing the admission rate of the offloading requests that are specific for the given time slots. Then, the

problem can be solved in subsequent time slots upon a change in the state of the corresponding computation offloading requests.

We denote by a_i a binary decision variable indicating if UE i's task is admitted in the network. When $o_{ij} = 1$, a portion $f_{ij} \geq 0$ of UAV cloudlet j's computational resources F_j (GHz) is allocated for computing UE i's task. Thus, the computation latency for UE i's task on cloudlet j depends on the allocated portion $f_{ij}F_j$ of the computational capacity, and is given by:

$$L^{\mathrm{c}}_{ij}(f_{ij}) = \frac{d_i c_i}{f_{ij} F_j} \tag{4.11}$$

There has been advances in the UAV battery performance and the energy replenishment techniques providing energy support and allowing the UAVs to fly for long periods [110, 111, 112, 113, 114]. However, there are still situations where the UAVs' flying and operational time is limited by their available onboard battery, which has been considered in existing studies. For instance, the authors in [115] imposed a limit on the traveling distance for the UAV due to the energy limitations where the collected scores are maximized, while in [116] the requests for UAVs' cached content were restricted by their available battery level, where each content request reduces the battery level by a specific amount.

Thus, in this work, we consider an energy limit of $\bar{E}_j$ (Joule) for each UAV $j \in \mathcal{M}$, which is assumed for tractability purposes, and is available for the UAV to conduct their service at the particular offloading period. Also, we note that $\bar{E}_j$ can be selected by the operator, such as the energy of UAV j is saved for subsequent offloading periods in case the UEs' request load is expected to rise. In our model, we

consider the energy consumption of each UAV j to be primarily influenced by the computation energy of the offloaded tasks on its cloudlet, noting that the hovering energy is ignored for the sake of simplicity since it is considered to be constant. The computation energy consumption of UAV j is defined as [117]:

$$E_j(\boldsymbol{f}_j) = \sum_{i \in \mathcal{N}} \gamma_j d_i c_i (F_j f_{ij})^2 \tag{4.12}$$

where γ_j is the effective switched capacitance of the CPU, which is determined by the CPU hardware architecture. We note in equation (4.12) that the energy resulting from computing each task i on UAV cloudlet j is the product of the power $\gamma_j (F_j f_{ij})^3$ and the time $d_i c_i / F_j f_{ij}$. In addition, f_{ij} is enough in (4.12) to indicate if task i is computed on UAV cloudlet j since otherwise zero resources $f_{ij} = 0$ will be allocated by the optimizer in that case.

At this point, for every collection of tasks requests in a given time slot, we proceed

to solve for the variables $\boldsymbol{o}, \boldsymbol{a}, \boldsymbol{f}, \boldsymbol{b}$ through the following problem:

$$\mathcal{P}_{1b}: \max_{\boldsymbol{o},\boldsymbol{a},\boldsymbol{f},\boldsymbol{b}} \quad \sum_{i\in\mathcal{N}} a_i \tag{4.13a}$$

$$\text{s.t.} \quad \Phi_i(\boldsymbol{o}_i) \geq a_i \bar{R}_i, \ \forall i \in \mathcal{N} \tag{4.13b}$$

$$o_{ij}\left(L^{\mathrm{u}}_{ij}(\boldsymbol{p}^{\star}_j, \theta^{\star}_{ij}, b_{ij}) + L^{\mathrm{c}}_{ij}(f_{ij})\right) \leq \bar{L}_i, \ \forall i \in \mathcal{N} \tag{4.13c}$$

$$o_{ij} \leq a_i, \ \forall \{i,j\} \in \{\mathcal{N}, \mathcal{M}\} \tag{4.13d}$$

$$a_i \leq \sum_{j\in\mathcal{M}} o_{ij}, \ \forall i \in \mathcal{N} \tag{4.13e}$$

$$\sum_{i\in\mathcal{N}} f_{ij} \leq 1, \ \forall j \in \mathcal{M} \tag{4.13f}$$

$$\sum_{j\in\mathcal{M}} \sum_{i\in\mathcal{N}} b_{ij} \leq 1 \tag{4.13g}$$

$$E_j(\boldsymbol{f}_j) \leq \bar{E}_j, \ \forall j \in \mathcal{M} \tag{4.13h}$$

$$f_{ij}, b_{ij} \geq 0, o_{ij}, a_i \in \{0,1\} \tag{4.13i}$$

where (4.13a) maximizes the rate of admitted tasks. Constraint (4.13b) guarantees the tasks' reliability requirement, where the RHS enforces the constraint only on admitted tasks. Constraint (4.13c) guarantees the total latency of each offloaded task replica must respect the task latency deadline. Constraint (4.13d) ensures a non-admitted task cannot be offloaded, and constraint (4.13e) ensures admitting a task that is offloaded to at least one UAV. Constraint (4.13f) respects the computational capacity of all UAV-mounted cloudlets. Constraint (4.13g) makes sure the wireless radio bandwidth B is respected, constraint (4.13h) respects the UAVs' available energy, and constraint (4.13i) is for the integrality and non-negative conditions.

It is worth noting that assigning continuous values for f_{ij} and b_{ij} (between 0 and 1), is considered a very safe assumption, because $f_{ij}F_j$ and $b_{ij}B$ return the amount of allocated cycles in Gigahertz, and the amount of allocated bandwidth in MHz, respectively. Thus, the integer values for the allocated number of cycles and number of hertz can be then obtained. All mathematical symbols used thus far are summarized in Table 4.1.

Similar to (4.10), problem (4.13) is non-convex due to the existence of the non-convex constraints (4.13b) and (4.13c). We note that constraint (4.13h) is convex and can be easily converted to a quadratic cone for more tractability. Problems (4.10) and (4.13) are non-convex and have binary constraints, which makes them general non-convex MI-NLP and hence very difficult to solve. Finding an optimal solution for (4.10) and (4.13) often requires a high-complexity exhaustive search algorithm, which is impractical. Thus, we present in the next section a low-complexity algorithm for solving (4.10) and (4.13) sequentially that attains a sub-optimal solution within a polynomial time.

4.5 Proposed Iterative Low-Complexity Algorithm

In this section, we transform the non-convex problems (4.10) and (4.13) into more tractable approximate forms, and propose an SCA-based iterative algorithm which provides a low-complexity approximate solution for the overall problem.

Notation	Description
$\mathcal{N}$	Set of UEs
$\mathcal{M}$	Set of UAVs
λ_i	Requests arrival rate for UE i (tasks/second)
d_i	Task input size (Kb)
c_i	Task computational density (CPU cycles/bit)
$\bar{L}_i$	Task latency deadline (ms)
$\bar{R}_i$	Task reliability requirement
$\bar{\boldsymbol{p}}_i = (\bar{x}_i, \bar{y}_i, 0)$	UE i's position
F_j	UAV computational capacity (GHz)
ϕ_j	UAV cloudlet reliability
g_0	Path loss at a reference distance of 1 meter
κ	Additional attenuation factor due to the non-LoS condition
P_{ij}	Transmission power from UE i to UAV j (dBm)
B	Radio spectrum bandwidth (MHz)
N_0	Noise power spectral density (dBm/MHz)
$\boldsymbol{p}_j = (x_j, y_j, H)$	UAV j's position
θ_{ij}	Path loss coefficient
$o_{ij} \in \{0, 1\}$	Indicates if the task of UE i is offloaded to UAV j
$a_i \in \{0, 1\}$	Indicates if the task of UE i is admitted to the system
$b_{ij} \in \mathbb{R}^+$	Portion of the allocated wireless radio bandwidth
$f_{ij} \in \mathbb{R}^+$	Portion of UAV j's resources assigned for the task of UE i

Table 4.1: Table of Notations

4.5.1 Convex Approximation

In this subsection, we convexify the non-convex constraints to approximate problems (4.10) and (4.13). We use the term "convexify" to indicate solving the non-convexity aspect of the particular constraint. This is done by making sure all terms on the left side of the "$\leq$" sign in the constraint are made convex, and the terms on the right side of the "$\leq$" sign are made concave. Those conditions apply to problems having either a maximization or a minimization objective [40].

First, we address constraint (4.10b) which is non-convex due to the existence of the convex function $f(\theta_{ij}) = (\theta_{ij})^{-1}$ on the greater side of the inequality, causing a Difference of Convex (DC) form [118]. Thus, we convexify (4.10b) by applying the inner-approximation method to approximate $f(\theta_{ij})$ by its upper-bounded convex function $\tilde{f}(\theta_{ij}; \theta_{ij}^{(n)})$ around the point $\theta_{ij}^{(n)}$ (at iteration n of the SCA algorithm) as:

$$\tilde{f}(\theta_{ij}; \theta_{ij}^{(n)}) = \frac{1}{\theta_{ij}^{(n)}} - \frac{(\theta_{ij} - \theta_{ij}^{(n)})}{(\theta_{ij}^{(n)})^2} \tag{4.14}$$

Constraint (4.10b) can now be written in a convex form as:

$$\tilde{f}(\theta_{ij}; \theta_{ij}^{(n)}) \geq (x_j - \bar{x}_i)^2 + (y_j - \bar{y}_i)^2 + H^2 \tag{4.15}$$

Second, constraint (4.10c) can be equivalently replaced by the following linear constraint:

$$\ln(1 - \bar{R}_i) \geq \sum_{j \in \mathcal{M}} o_{ij} \ln(1 - \phi_j) \tag{4.16}$$

The steps detailing the convexification of (4.10c) are laid out in Appendix A.3.

Finally, we address (4.10d). By introducing slack variables $\boldsymbol{\beta} = \{\beta_{ij} \geq 0, \forall i \in \mathcal{N}, j \in \mathcal{M}\}$, $\boldsymbol{\Gamma} = \{\Gamma_{ij} \geq 0, \forall i \in \mathcal{N}, j \in \mathcal{M}\}$, $\boldsymbol{\gamma} = \{\gamma_{ij} \geq 0, \forall i \in \mathcal{N}, j \in \mathcal{M}\}$,

$\boldsymbol{\delta} = \{\delta ij \geq 0, \forall i \in \mathcal{N}, j \in \mathcal{M}\}$, constraint (4.10d) can be equivalently rewritten as

$$o_{ij}\left(\beta_{ij} + \Gamma_{ij}\right) \leq \bar{L}_i \tag{4.17a}$$

$$\frac{d_i}{\gamma_{ij}} \leq \beta_{ij} \tag{4.17b}$$

$$\frac{1}{\delta_{ij}} \leq \Gamma_{ij} \tag{4.17c}$$

$$\gamma_{ij} \leq \tilde{R}_{ij}(\theta_{ij}, b_{ij}) \tag{4.17d}$$

$$\delta_{ij} \leq \frac{F_j}{\bar{c}} - \sum_{i \in \mathcal{N}} o_{ij}\alpha_i\lambda_i \tag{4.17e}$$

We observe that (4.17b) and (4.17c) are convex and can be easily converted to quadratic cones, while (4.17d) is a generalized exponential cone so it is also convex. However, constraints (4.17a) and (4.17e) along with (4.10e) are still non-linear. To address (4.17a), (4.17e), (4.10e), we apply the well known big-M technique with the constant $A \gg 1$. By introducing slack variables $\boldsymbol{u} = \{u_{ij} \geq 0, \forall i \in \mathcal{N}, j \in \mathcal{M}\}$ and $\boldsymbol{v} = \{v_{ij} \geq 0, \forall i \in \mathcal{N}, j \in \mathcal{M}\}$, the linear slack constraints are:

$$u_{ij} \leq o_{ij}A \tag{4.18a}$$

$$(o_{ij} - 1)A + \beta_{ij} + \Gamma_{ij} \leq u_{ij} \tag{4.18b}$$

$$u_{ij} \leq \beta_{ij} + \Gamma_{ij} \tag{4.18c}$$

$$v_{ij} \leq o_{ij}A \tag{4.18d}$$

$$(o_{ij} - 1)A + \alpha_i \leq v_{ij} \tag{4.18e}$$

$$v_{ij} \leq \alpha_i \tag{4.18f}$$

4.5.2 MI-SOCP Problem Transformation

After linearizing the non-convex constraints for problem (4.10), the approximated problem would obtain the generalized convex characteristic, due to the appearance of the generalized exponential cone constraint (4.17d), which makes it very challenging to solve. We can transform the problem into a MI-SOCP form which is a standard convex program that can be solved very efficiently. Thus, we proceed to employ the conic approximation with controlled accuracy in [61] to rewrite constraint (4.17d) by a set of second order cone inequalities as:

$$
\begin{aligned}
\kappa_{m+4}^{ij} &\leq b_{ij} + \frac{P_{ij}\tilde{h}_{ij}(\theta_{ij})}{BN_0} \\
b_{ij} + \kappa_1^{ij} &\geq \left\| \begin{bmatrix} b_{ij} - \kappa_1^{ij} & 2b_{ij} + \frac{\gamma_{ij}}{B2^{m-1}} \end{bmatrix} \right\|_2 \\
b_{ij} + \kappa_2^{ij} &\geq \left\| \begin{bmatrix} b_{ij} - \kappa_2^{ij} & \frac{5b_{ij}}{3} + \frac{\gamma_{ij}}{B2^{m}} \end{bmatrix} \right\|_2 \\
b_{ij} + \kappa_3^{ij} &\geq \left\| \begin{bmatrix} b_{ij} - \kappa_3^{ij} & 2\kappa_1^{ij} \end{bmatrix} \right\|_2 \\
\kappa_4^{ij} &\geq \kappa_2^{ij} + \frac{\kappa_3^{ij}}{24} + \frac{19b_{ij}}{72} \\
b_{ij} + \kappa_l^{ij} &\geq \left\| \begin{bmatrix} b_{ij} - \kappa_l^{ij} & 2\kappa_{l-1}^{ij} \end{bmatrix} \right\|_2 \forall\ l \in \{5, ..., m+3\} \\
b_{ij} + \kappa_{m+4}^{ij} &\geq \left\| \begin{bmatrix} b_{ij} - \kappa_{m+4}^{ij} & 2\kappa_{m+3}^{ij} \end{bmatrix} \right\|_2
\end{aligned}
\tag{4.19}
$$

where $\boldsymbol{\kappa}_m = \{\kappa_{ij} \geq 0, \forall i \in \mathcal{N}, j \in \mathcal{M}\}$ is a new slack variable, and m is the parameter of the conic approximation technique which can be chosen as $m = 4$ to attain a high accuracy.

4.5.3 SOCP Approximation and SCA-based Algorithm

In this subsection, we develop an approximate SCA-based SOCP algorithm to iteratively solve problems (4.10) and (4.13) until convergence.

The mixed-integer SOCP approximation of problem $\mathcal{P}_{1a}$ will still pause scalability limitations, preventing the SCA algorithm from being applied to big instances due to the mixed-integer nature of the problem, which is caused by the binary condition of variable $\boldsymbol{o}$. To solve that problem, we adopt a similar approach to [83], and relax the binary condition for variable $\boldsymbol{o}$ by introducing the following constraint:

$$0 \leq o_{ij} - o_{ij}^2 \leq \zeta_{ij} \tag{4.20}$$

where $\boldsymbol{\zeta} = \{\zeta_{ij} \geq 0, \forall i \in \mathcal{N}, j \in \mathcal{M}\}$ is a newly introduced slack variable. (4.20) is non-convex due to the concave function $g(o_{ij}) = -o_{ij}^2$ which renders the left side of (4.20) as a DC form. Thus, with $o_{ij}^{(n)}$ as the input point, we employ the SCA method to replace $g(o_{ij})$ by its first order Taylor approximate as:

$$\tilde{g}(o_{ij}; o_{ij}^{(n)}) = -(o_{ij}^{(n)})^2 - 2o_{ij}^{(n)}(o_{ij} - o_{ij}^{(n)}) \tag{4.21}$$

Constraint (4.21) will force variable $\boldsymbol{o}$ to take a binary value with a penalty term added to the objective. At this point, by employing the above approximations, an

approximated SOCP problem for problem $\mathcal{P}_{1a}$ is formulated at the n^{th} iteration as:

$$\mathcal{P}_1^{(n)}: \underset{\substack{\boldsymbol{p},\boldsymbol{\theta},\tilde{\boldsymbol{O}},\\ \tilde{\boldsymbol{B}},\boldsymbol{\beta},\boldsymbol{\Gamma},\boldsymbol{\gamma},\\ \boldsymbol{\delta},\boldsymbol{u},\boldsymbol{v},\boldsymbol{\kappa},\boldsymbol{\zeta}}}{max} \quad \sum_{i\in\mathcal{N}} \alpha_i \lambda_i - A \sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{M}} \zeta_{ij} \tag{4.22a}$$

$$\text{s.t.} \quad u_{ij} \leq \bar{L}_i \tag{4.22b}$$

$$\delta_{ij} \leq \frac{F_j}{\bar{c}} - \sum_{i\in\mathcal{N}} v_{ij}\lambda_i \tag{4.22c}$$

$$\frac{F_j}{\bar{c}} \geq \sum_{i\in\mathcal{N}} v_{ij}\lambda_i \tag{4.22d}$$

$$\boldsymbol{p}_j, \theta_{ij}, b_{ij} \geq 0, o_{ij} \in [0,1] \tag{4.22e}$$

$$\text{(4.10f)–(4.10g), (4.15), (4.16), (4.17b),}$$

$$\text{(4.17c), (4.18), (4.19), (4.21).} \tag{4.22f}$$

We note that constraints (4.13b) and (4.13c) of (4.13) can be transformed similar to constraints (4.10c) and (4.10d) of problem (4.10).

The pseudocode for the algorithm is outlined in Algorithm 5. Problem (4.22) is solved iteratively until the convergence of objective (4.22a). After finding the solution for $\boldsymbol{p}^\star$ and $\boldsymbol{\theta}^\star$, the UAVs will fly to the corresponding positions to obtain the accurate $\hat{P}_{\text{LoS}ij}(\boldsymbol{p}_j)$ at the optimal locations. Then, we find the optimal values for the other optimization variables by solving a similarly transformed SOCP form of problem (4.13). A central computation unit is considered for conducting the optimization program after acquiring the necessary parameters of the tasks and UAVs through the control plane, which can be done using a high speed link. We assume a negligible latency for transmitting the input parameters to the central controller, considering

its negligible size compared to the actual computational tasks data.

Algorithm 5 SCA-based SOCP Algorithm.

1: Set $n := 0$;
2: Initialize the starting point for $\boldsymbol{\theta}^{(n)}$;
3: **repeat**
4: Solve the SOCP problem (4.22) in order to obtain objective value $\omega_1^{(n)}$, and $\boldsymbol{p}^\star, \boldsymbol{\theta}^\star, \tilde{\boldsymbol{O}}^\star, \tilde{\boldsymbol{B}}^\star, \boldsymbol{\beta}^\star, \boldsymbol{\Gamma}^\star, \boldsymbol{\gamma}^\star, \boldsymbol{\delta}^\star, \boldsymbol{u}^\star, \boldsymbol{v}^\star, \boldsymbol{\kappa}^\star$;
5: Set $n := n + 1$;
6: Update $\boldsymbol{\theta}^{(n)} = \boldsymbol{\theta}^\star$;
7: **until** Objective (4.22a) convergence: $|\omega_1^{(n+1)} - \omega_1^{(n)}| = 0$.
8: Solve the transformed form of problem (4.13).

Convergence Analysis: The convergence of Algorithm 5 can be guaranteed by showing that the series of resulting objective is monotonically convergent. Let $\Omega^{(n)}$ denote the optimal solution set at the nth iteration of Algorithm 5. Due to the convex approximation in (4.14), the updating rule in Algorithm 5, c.f., Step 6, ensures that the solution set $\Omega^{(n)}$ is a feasible solution to problem $\mathcal{P}_{1a}$ at step $n + 1$. This subsequently leads to the results of $\omega_1^{(n+1)} \geq \omega_1^{(n)}$, which means that Algorithm 5 generates a non-decreasing sequence of objective function values. Due to the latency and reliability constraints, the sequence of $\omega_1^{(n)}, n = 1, 2, \ldots$ is bounded below and therefore, Algorithm 5 guarantees that the objective converges.

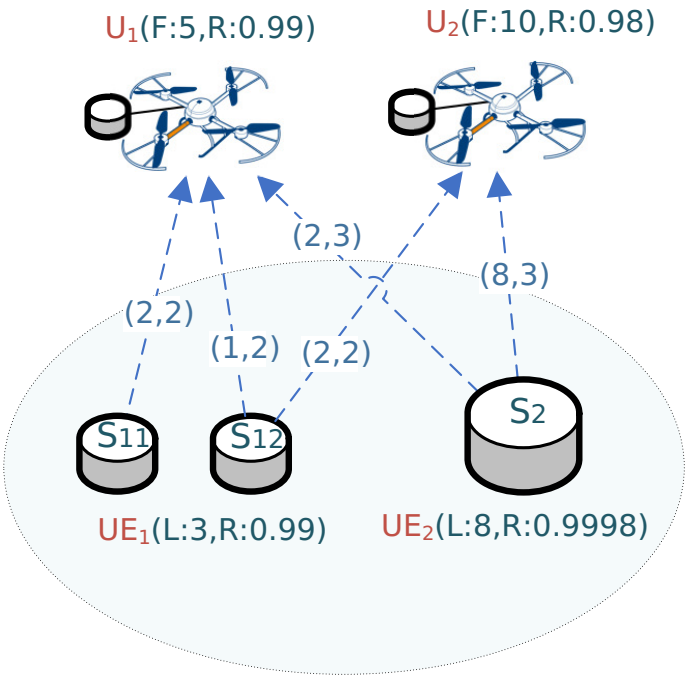


Figure 4.3: Task partitioning illustrative example.

4.6 Computation Offloading with Task Partitioning

Many services in 5G and future networks will require the offloading of tasks where input data can be partitioned, e.g. image/video processing [19]. Thus, in this section, we adapt our model and mathematical formulation to account for the data-partitioning model where a task can be partially offloaded and computed on multiple UAVs in parallel. The adopted task partitioning follows the partial computation offloading model, where a task can be subdivided into sub-tasks with random size, because no inter-dependency exists among the task parts, such as the case with image and video processing. In contrast, when inter-dependency among the sub-tasks

exists, other task modelings are usually adopted such as a task-call graph where the nodes in the graph represent the sub-tasks, and the directed edges represent the sub-tasks' dependency [19]. As highlighted in Fig. 4.3, the total task latency for U_1 can be decreased after T_1 is split into two sub-tasks and each sub-task is assigned to one or more UAV cloudlets. This is because a sub-task will have a smaller size which decreases its individual upload and computation latency. This model will also allow us to exploit the trade-off between latency and reliability, as will be later discussed.

4.6.1 Model Updates

We denote by $\mathcal{S}_i = \{1, 2, ..., S_i\}$ the set of equal-sized sub-tasks of UE i's task, and S_i is the number of sub-tasks that i's task can have. For enhancing the task reliability, sub-task k can be offloaded to multiple UAVs. For this reason, after specifying dimension S_i, o_{ikj} indicates if sub-task $k \in \mathcal{S}_i$ is offloaded to UAV j, and f_{ikj} indicates the portion of the allocated computational resources when $o_{ikj} = 1$. Thus, constraints (4.13d)-(4.13e) can be substituted by:

$$o_{ikj} \leq a_i, \ \forall \{i, k, j\} \in \{\mathcal{N}, \mathcal{S}, \mathcal{M}\} \tag{4.23a}$$

$$a_i \leq \sum_{j \in \mathcal{M}} o_{ikj}, \ \forall \{i, k\} \in \{\mathcal{N}, \mathcal{S}\} \tag{4.23b}$$

Then, the reliability constraint (4.6) can be replaced by:

$$\Phi_i(\boldsymbol{o}_i) = \prod_{k \in \mathcal{S}} \left(1 - \prod_{j \in \mathcal{M}} (1 - o_{ikj}\phi_j)\right) \tag{4.24}$$

Equation (4.24) hints that the number of sub-tasks would decrease the achieved reliability, but this will help in decreasing the attained latency since the sub-tasks can be offloaded in parallel.

Due to parallel offloading, the task latency will depend on the latency of the last completed sub-task. Thus, the total latency of each offloaded sub-task must respect the task latency. Hence, constraint (4.13c) can be replaced by:

$$\frac{d_i o_{ikj}}{r_{ikj} S_i} + \frac{d_i c_i o_{ikj}}{f_{ikj} F_j S_i} \leq \bar{L}_i \tag{4.25a}$$

$$\sum_{k \in \mathcal{S}} r_{ikj} \leq \tilde{R}_{ij}(\theta_{ij}, b_{ij}) \tag{4.25b}$$

where r_{ikj} is a decision variable denoting the fraction of the achievable rate $R_{ij}(\theta_{ij}, b_{ij})$ between UE i and UAV j that is allocated for offloading sub-task k. Constraint (4.25b) ensures respecting the task achievable rate.

Moreover, constraint (4.13h) can be replaced by:

$$E_j(\boldsymbol{f}_j) = \sum_{i \in \mathcal{N}} \sum_{k \in \mathcal{S}} \left(\gamma_j d_i c_i F_j^2 f_{ikj}^2 / S_i \right) \tag{4.26}$$

which captures the energy consumed by UAV cloudlet j for computing the offloaded sub-tasks.

Problem (4.10) can now be replaced by problem $\mathcal{P}_{2a}^{(n)}$ as:

$$\mathcal{P}_{2a}^{(n)}: \underset{\substack{\boldsymbol{p},\boldsymbol{\theta},\boldsymbol{\alpha},\\ \tilde{\boldsymbol{O}},\tilde{\boldsymbol{B}}}}{max} \quad \sum_{i\in\mathcal{N}} \alpha_i \lambda_i \tag{4.27a}$$

$$\text{s.t.} \quad \text{(4.10b), (4.10d), (4.10e).} \tag{4.27b}$$

$$\Phi_i(\boldsymbol{o}_i) \geq \bar{R}_i \tag{4.27c}$$

$$\sum_{j\in\mathcal{M}} o_{ijk} \geq 1, \ \forall i \in \mathcal{N}, k \in \mathcal{S} \tag{4.27d}$$

$$\sum_{j\in\mathcal{M}} \sum_{i\in\mathcal{N}} b_{ij} \leq 1 \tag{4.27e}$$

$$\boldsymbol{p}_j, \theta_{ij}, b_{ij} \geq 0$$

$$, o_{ij} \in \{0,1\} \tag{4.27f}$$

At this point, we solve for the remaining variables $\boldsymbol{o}, \boldsymbol{a}, \boldsymbol{f}, \boldsymbol{b}, \boldsymbol{r}$ through the following

problem:

$$\mathcal{P}_{2b}: \max_{\boldsymbol{o},\boldsymbol{a},\boldsymbol{f},\boldsymbol{b}} \quad \sum_{i\in\mathcal{N}} a_i \tag{4.28a}$$

$$\text{s.t.} \quad (4.23),\ (4.25). \tag{4.28b}$$

$$\Phi_i(\boldsymbol{o}_i) \geq a_i \bar{R}_i \tag{4.28c}$$

$$\sum_{i\in\mathcal{N}} \sum_{k\in\mathcal{S}} f_{ikj} \leq 1,\ \forall j \in \mathcal{M} \tag{4.28d}$$

$$\sum_{j\in\mathcal{M}} \sum_{i\in\mathcal{N}} b_{ij} \leq 1 \tag{4.28e}$$

$$E_j(\boldsymbol{f}_j) \leq \bar{E}_j,\ \forall j \in \mathcal{M} \tag{4.28f}$$

$$f_{ij}, b_{ij}, r_{ikj} \geq 0$$

$$o_{ij}, a_i \in \{0,1\} \tag{4.28g}$$

4.6.2 Proposed Algorithm Updates

Similar to problem (4.10), (4.27) is also a general non-convex MI-NLP due to the binary-related constraints in (4.27f), and to the existence of the non-convex non-concave terms: constraints (4.10b), (4.10d), and (4.27c). We also proceed to transform (4.27) into a more tractable form, and solve it using a low complexity SCA-based iterative algorithm.

We address (4.27c) which can be equivalently replaced by the following convex constraints through steps similar to those of constraint (4.10c) after introducing slack

variable $\alpha_{ik} \geq 0$:

$$\sum_{k \in \mathcal{S}} \ln(\alpha_{ik}) \geq a_i \ln(\bar{R}_i) + (1 - a_i) \ln(\epsilon) \tag{4.29a}$$

$$\ln(1 - \alpha_{ik}) \geq \sum_{j \in \mathcal{M}} o_{ikj} \ln(1 - \phi_j) \tag{4.29b}$$

Constraints (4.25a) and (4.25b) have an identical structure to constraints (4.13c) and (4.17d), respectively, and can be transformed similarly. By employing the above approximations, an approximated SOCP of problem (4.27) can be formulated at the n^{th} iteration as:

$$\mathcal{P}_2^{(n)}: \max_{\substack{\boldsymbol{p},\boldsymbol{\theta},\tilde{\boldsymbol{O}},\\ \tilde{\boldsymbol{A}},\tilde{\boldsymbol{F}},\tilde{\boldsymbol{B}},\tilde{\boldsymbol{R}},\\ \boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\Gamma},\boldsymbol{u},\boldsymbol{\kappa}}} \quad \sum_{i \in \mathcal{N}} \alpha_i \lambda_i - A \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{M}} \zeta_{ij} \tag{4.30a}$$

$$\text{s.t.} \quad \boldsymbol{p}_j, \theta_{ij}, b_{ij} \geq 0, o_{ij} \in [0, 1] \tag{4.30b}$$

$$(4.15),\ (4.17b),\ (4.17c),\ (4.18),\ (4.19)$$

$$(4.21),\ (4.22b)\text{-}(4.22d),\ (4.23),\ (4.27d)\text{–}(4.27e),\ (4.29). \tag{4.30c}$$

The pseudocode for the SCA-based algorithm is outlined in Algorithm 6.

4.7 Numerical Results

In this section, we study the design performance of the proposed solutions through simulations considering various scenarios. The main instance we use consists of $N = 15$ IoT devices that are randomly distributed in a 2-D area of 2x2 km^2 [117],

Algorithm 6 SOCP Algorithm for Task Partitioning.

1: Set $n := 0$;
2: Initialize the starting point for $\boldsymbol{\theta}^{(n)}$;
3: **repeat**
4: Solve the SOCP problem (4.30) in order to obtain objective value $\omega_2^{(n)}$, and $\boldsymbol{p}^\star, \boldsymbol{\theta}^\star, \tilde{\boldsymbol{O}}^\star, \tilde{\boldsymbol{A}}^\star, \tilde{\boldsymbol{F}}^\star, \tilde{\boldsymbol{B}}^\star, \tilde{\boldsymbol{R}}^\star, \boldsymbol{\alpha}^\star, \boldsymbol{\beta}^\star, \boldsymbol{\Gamma}^\star, \boldsymbol{\gamma}^\star, \boldsymbol{u}^\star, \boldsymbol{\kappa}^\star$;
5: Set $n := n + 1$;
6: Update $\boldsymbol{\theta}^{(n)} = \boldsymbol{\theta}^\star$;
7: **until** Objective (4.30a) convergence: $|\omega_2^{(n+1)} - \omega_2^{(n)}| = 0$.
8: Solve the transformed form of problem (4.28).

Parameter	**Value**
Task input size d_i	[30 70] Kb
Task computational demand c_i	[150 250] CPU cycles/bit
Task latency deadline $\bar{L}_i$	[15 20] ms
Task reliability requirement $\bar{R}_i$	[0.99 0.9955]
UE i's requests arrival rate λ_i	[5 10] tasks/second
UAV cloudlet capacity F_j	[1 1.5] GHz
UAV cloudlet reliability ϕ_j	[0.9955 0.9999]
UAVs height H	100 meters [103, 104]
Path loss at 1 meter g_0	-50 dB [91, 117]
Channel parameters (a, b, κ)	(10, 0.6, 0.2) [103]
UE Transmission power P_{ij}	30 dBm [117]
Radio spectrum bandwidth B	20 MHz
Noise power spectral density N_0	-174 dBm/Hz x 10^6 [90]

Table 4.2: Simulation Parameters

and a set of $M = 3$ UAV-mounted cloudlets. The system parameters are those presented in Table 4.2 unless otherwise specified.

In Fig. 4.4, we study the convergence behavior of the SCA algorithm for solving the planning problem (lines 1 to 7 of Algorithm 5). This is done for three different starting positions for the SCA approximation points $\boldsymbol{\theta}^{(n)}$ which reflect the UAVs'

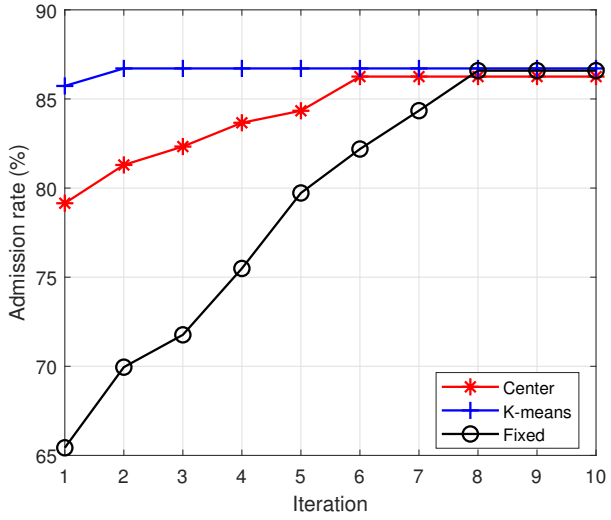


Figure 4.4: Convergence behavior of the SCA algorithm.

positions. When selecting the starting points using the K-means technique, we observe that the algorithm converges very quickly. When starting from the center of the xy plane, the SCA approach takes few more iterations to converge. Lastly, when the starting values of $\boldsymbol{\theta}^{(n)}$ are fixed such as θ_{ij} is the same $\forall\{i,j\} \in \{\mathcal{N}, \mathcal{M}\}$ (the UAVs' starting points are scattered), this leads to the lowest performance as the algorithm needs more iterations to converge. It is worth noting that the SCA algorithm generally requires more iterations to converge when UEs have stricter latency and reliability requirements.

In Fig. 4.5, we study the effect of reliability requirements and the bandwidth capacity on the achieved admission rate for the planning problem. The objective is to explore the bandwidth capacity required by the network in response to the UEs' reliability requirements for achieving a certain admission rate. As is apparent in Fig. 4.7, having more stringent reliability requirements decreases the achieved admission

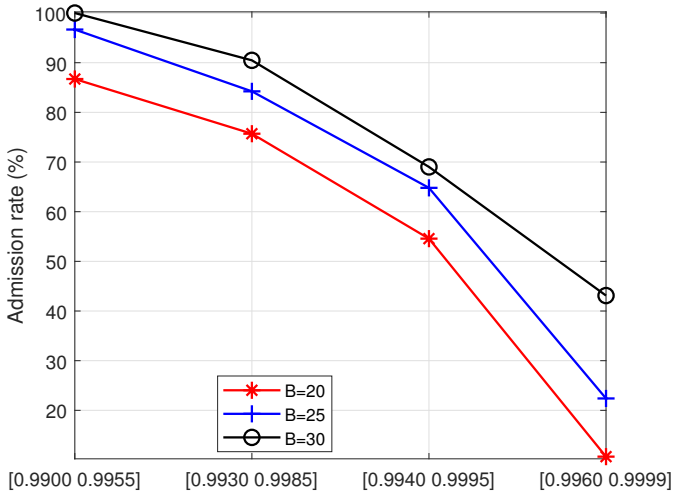
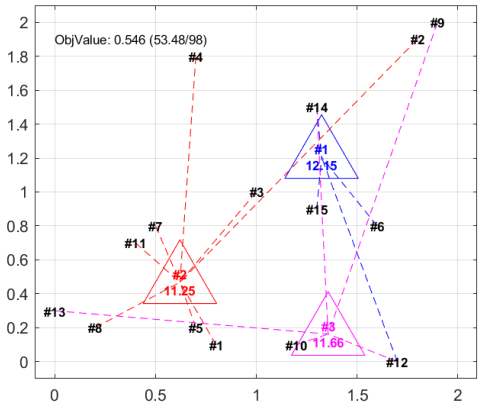


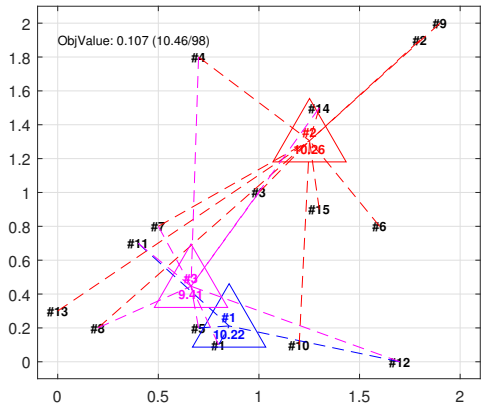
Figure 4.5: Objective vs UEs' reliability $\bar{R}_i$ and wireless bandwidth B.

rate, because UEs will need to replicate the transmission of their tasks to more UAVs, which utilizes more wireless radio bandwidth, and hence limits the resources available for the other tasks. Thus, it is observed that a trade-off exists between the ability to provision computation offloading with high reliability requirements, and the capability of the system to admit offloaded tasks due to the utilization of more network resources. In addition, it is shown that the NO can increase the admission rate when more system bandwidth is available, obviously because more tasks can be accommodated on the wireless spectrum.

In Fig. 4.6, we illustrate the optimized UAVs' positions and the UE-to-UAV associations within the xy plane for different classes of reliability requirements, where the numbers in the triangles represent the average cloudlet computation latency based on the offloaded requests. As can be seen in Fig. 4.6.a, UAVs are relatively positioned apart to cover the UEs in such a way that their transmission latency



4.6.a Reliability requirement: 0.9940-0.9995.



4.6.b Reliability requirement: 0.9960-0.9999.

Figure 4.6: Illustrating the UAVs' placement for different UE's reliability requirements.

can satisfy the latency deadline, knowing that in this case, UEs need to offload to only one UAV to satisfy their reliability requirement. In Fig. 4.6.b, the UEs have higher reliability requirements, resulting in the need to offload to multiple UAVs, occupying more computation and wireless resources, and hence fewer requests are admitted in the network. Here, it is important to note that the UAVs' positions have higher influence on the achieved latency because bandwidth resources are more utilized, which made UAV 1 in this case be positioned nearby UAV 3 for satisfying the latency requirements of the served UEs.

In Fig. 4.7, we compare the SCA approach in Algorithm 5 with two other baseline approaches: KM and EB, while studying the effect of change in the UEs' reliability requirement on the objective. In KM, problem (4.22) is slightly modified, such as

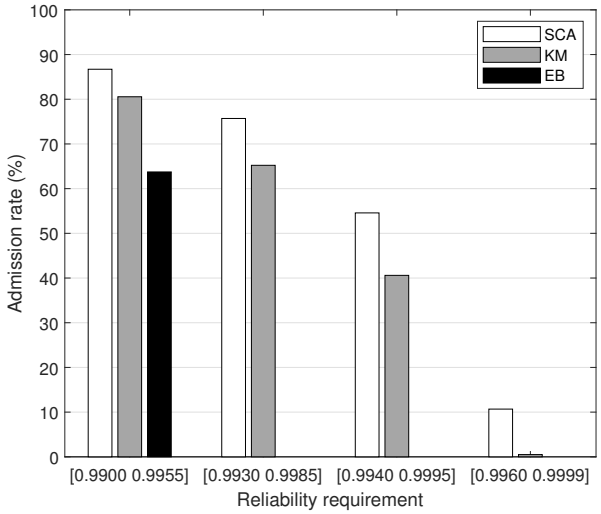


Figure 4.7: Objective obtained using the SCA algorithm vs baseline approaches.

the UAVs' positions are optimized using the K-means algorithm and provided as input to the problem for solving the remaining variables. In EB, the SCA approach of Algorithm 5 is invoked with the same decision variables except for $\tilde{\boldsymbol{B}}$ where the wireless bandwidth resources are equally allocated to the IoT devices from the available bandwidth B. The simulations are done for a task deadline $\bar{L}_i$ in the range of [25 30] ms, $\forall i \in \mathcal{N}$. Through Fig. 4.7, we observe that the SCA Algorithm always leads to a better admission rate when all variables are optimized. KM leads to a lower tasks admission rate because the UAVs' positions optimization through the K-means approach is not as efficient. We note that the solution gap increases and the improvement becomes more significant when the reliability requirements become stricter, which is because in the SCA approach the UAVs' positions will be pushed further for accommodating the latency requirements of farther UEs that are offloading to multiple cloudlets for satisfying their reliability. In fact, the transmission latency

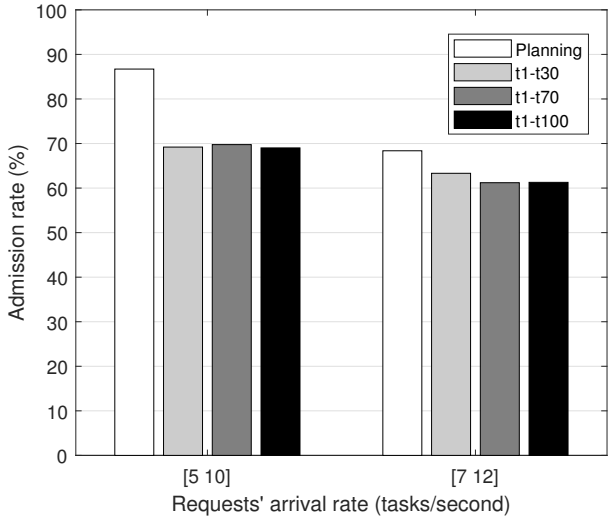


Figure 4.8: Planning problem objective vs Algorithm 5 objective for particular time slots.

here becomes more important for those UEs having to utilize more UAV resources because of their increased reliability requirement, which increases the computation latency. EB leads to an even lower admission rate and becomes infeasible when the UE's reliability requirements become stricter, which is because the transmission rate is drastically affected. Lastly, increasing the UEs' reliability requirements always decreases the admission rate since more computation and bandwidth resources have to be consumed, limiting the resources available for other tasks.

In Fig. 4.8, we study the achieved admission rate for the planning problem, versus the average admission rate obtained for the whole problem when applied to the first 30 time slots, first 70 time slots, and first 100 time slots of the offloading process (time slots requests are generated based on the UEs' Poisson arrival rates). The objective is to understand how does the admission rate obtained from the planning

problem compares with the solution of the second problem when applied for different offloading time spans. As it can be seen, the planning problem gives a bit more optimistic measure for the admitted load, compared to the actual admitted load for the requests during the offloading phase. This difference is actually due to considering the LoS communication and to the usage of the average task computational demand $\bar{c}$ in the planning problem, and therefore leading to a more optimistic admission rate. However, it is important to note that the corresponding admission rate (which depends on average planning values and not on the actual requests at hand) can be used to obtain an insight into the predicted admission rate during the actual offloading phase. Moreover, a term can be added in the objective to account for this difference and therefore obtain a more accurate prediction. However, it is essential to emphasize on that fact that the primary objective of the planning problem is to obtain the long-term optimized positioning of the UAVs based on average planning values, so that the other decisions can be optimized in the particular time slots after the UAVs' positions are determined. Another observation is that the achieved admission rate stays more or less the same when increasing the length of the offloading time-span, noting that the difference will be more significant when using instances with a higher number of UEs. In addition, it can be seen that a higher arrival rate for the requests leads to a lower admission rate, which is because additional requests will not be admitted when resources are fully utilized in the network, emphasizing the importance of efficiently optimizing the network resources.

In Fig. 4.9, we solve Algorithm 5 for the purpose of studying how both the UEs' latency requirements and cloudlets' capacity influence the achieved admission rate.

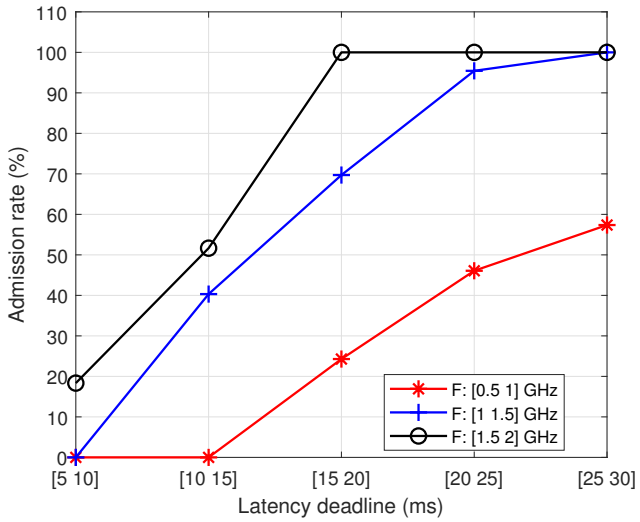


Figure 4.9: Objective obtained as it varies with the UEs' latency deadline and cloudlets' capacity.

The goal is to explore the extent at which computation capabilities need to be added for maintaining a certain admission rate considering different classes of IoT devices' latency deadline. As it can be seen, having stricter latency deadlines decreases the admission rate, since IoT devices have to consume more wireless and computational resources for meeting their requirements, and hence restricting the resources available for serving other tasks. In addition, increasing the available computational resources allows for a higher admission rate, obviously because more resources are available for serving the requested tasks, which reduces the computation latency and allows for meeting the tasks' deadline. It can be observed that at some point, for IoT devices with high enough latency requirements, the needed admission rate in the network can already be satisfied and adding more network resources will not be needed. Thus, it is important to plan network resources according to the expected IoT service category

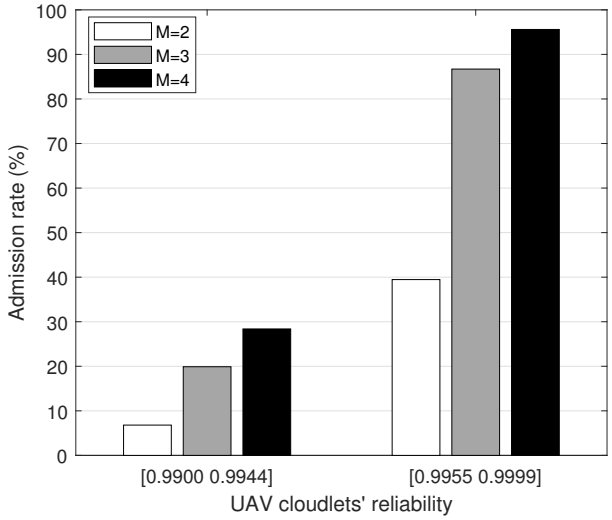


Figure 4.10: Objective obtained as it varies with the cloudlets' reliability and their number.

to avoid resources over-provisioning.

In Fig. 4.10, we solve Algorithm 5 for the purpose of studying the pattern through which the UAV cloudlets' reliability and their number can influence the admission rate. The goal is to provide some information into the number of utilized UAV cloudlets which can be dependent on their reliability, and which would affect the computation offloading efficiency. Here, we study two classes of UAV cloudlets' reliability. As it can be seen, when UAV cloudlets are more reliable, the admission rates are increased since the tasks can then be offloaded to fewer UAVs, which offers more computation and wireless resources for offloading other tasks. Here, it is apparent that when UAV cloudlets have low reliability, a lot more of them are needed to maintain the same admission rate due to the need to occupy a higher amount of resources. However, introducing more UAVs also requires the optimization of their positioning.

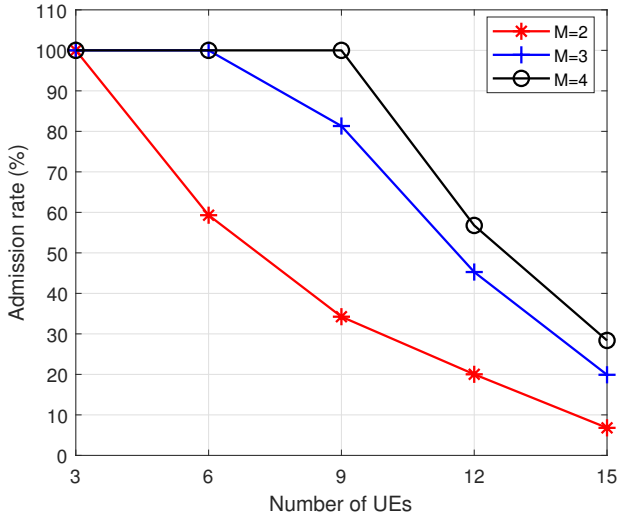


Figure 4.11: Objective value as it varies with the number of UEs and the UAV cloudlets.

Therefore, depending on their reliability, a certain number of UAV-mounted cloudlets can be utilized for responding to the IoT devices' specific requirements.

In Fig. 4.11, we study the solution of Algorithm 5 for exploring the effect of change in the number of IoT devices and the number of UAV cloudlets on the admission rate. Here, we consider UAV cloudlets with a reliability in the range of [0.9900 0.9944]. As it can be seen, when more UEs are added to the network, the admission rate decreases since only a limited number of requests can be served due to the limited amount of network resources. Here, it is shown that when more UAV cloudlets are added to the network, more tasks can be offloaded because of the increased amount of computational resources and the offloading options.

In Fig. 4.12, we present the performance for the offloading and resources allocation problem in terms of the admission rate as it varies with the energy limit per each

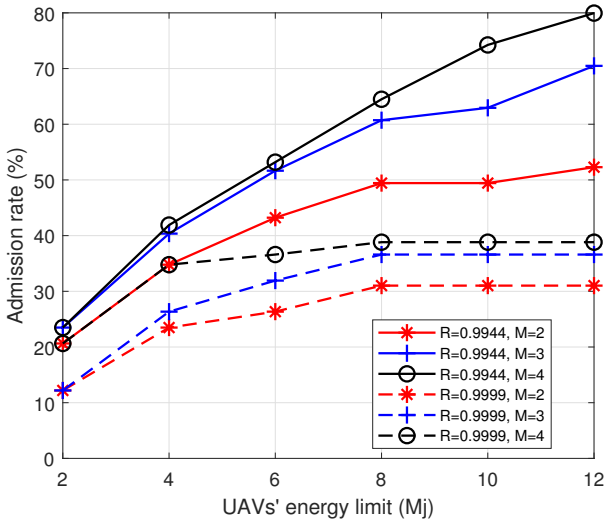
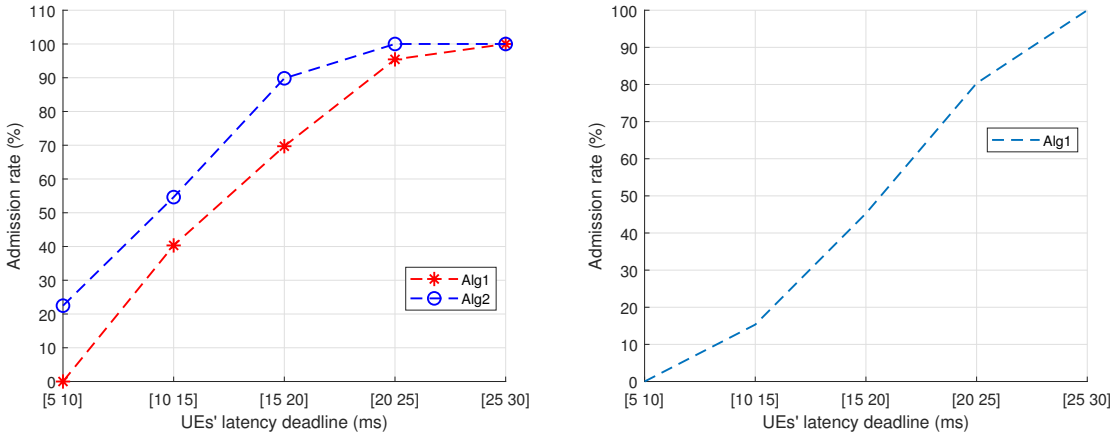


Figure 4.12: Objective value as it varies with the UAVs' energy level.

UAV. The results are presented for different number of UAVs as well as for different task reliability requirements (namely, $R_1 = 0.9944$ and $R_1 = 0.9999$), and the reliability of each UAV cloudlet is fixed at 0.9955. This implies that for tasks of reliability R_1, offloading to one UAV will be sufficient, while tasks of reliability R_2 require each two UAVs to attain the required reliability. As the figure shows, when the reliability requirement of the tasks is lower, the admission rate is higher, and as we increase the energy limit per UAV, the admission rate also increases. This is explained due to the direct relationship between the computing capacity and the energy consumption of the UAV. The higher the UAV energy budget, the more capacity can be utilized to carry out the computation and therefore, the better the task admission rate into the system. We also observe that more UAVs provide higher admission rate, since the system has more resources (computing resources) which helps in meeting the latency requirements of more tasks and hence admitting more UEs. Now, in the



4.13.a UEs' reliability in the range of 0.99-0.9955. 4.13.b UEs' reliability in the range of 0.9944-0.9999.

Figure 4.13: Algorithm 5 vs the task partitioning approach (Algorithm 6) with $S_i = 2\ \forall i \in \mathcal{N}$.

case of R_2, when the task reliability requirement is higher than what one UAV can offer, two UAVs become necessary to satisfy the constraint. Hence, more resources are demanded for the same load; i.e., twice the resources are required to satisfy the latency and reliability requirements of the task, which explains the lower admission rate. However, we observe that increasing the UAVs' energy budget beyond a certain limit does not lead to an increase in the admission rate. The reason being that while the provided energy is enough to utilize all the available computational resources, the computational resources themselves are fully consumed to satisfy the latency requirements of the already admitted tasks and hence not allowing more tasks into the system. In other words, the system become capacity limited rather than energy limited.

In Fig. 4.13, we study how the task partitioning in Algorithm 6 with $S = 2$

affects the obtained solution for different classes of UE's latency deadline. We use an instance that consists of 6 UEs and 2 UAV cloudlets where the UEs' task size is in the range of [70 110] Kb. As it can be seen in Fig. 4.13.a, a higher admission rate is achieved by Algorithm 6 for all latency deadline levels. This gain is due to the fact that with task partitioning, sub-tasks (having a smaller size) can be transmitted and computed in parallel on multiple UAV cloudlets, which decreases the tasks' computation latency, and hence less resources are needed, and more tasks can be served. It is also shown that the difference gap becomes less significant when latency deadlines become more loose, since the tasks here can consume less resources and more resources can be used for admitting other tasks. In Fig. 4.13.b, where UEs have higher reliability requirements, the admission rate obtained from Algorithm 5 decreases for most of the points, since task redundancy is utilized which decreases the resources available for admitting other tasks. In this case, however, the reliability achieved through equation (4.27c) in Algorithm 6 is too low with respect to the UEs' reliability requirement, and therefore Algorithm 6's solution becomes infeasible. This is due to the higher resources demand caused by the increased replication of the sub-tasks to more cloudlets, while having a limited number of 3 UAV cloudlets in addition to their limited resources (in this particular case, the solution will be feasible when more resources are added to the network). Thus, we observe that when splitting a task into multiple sub-tasks, more computation and wireless resources are needed to achieve the same reliability of the original task, and hence the NOs should consider increasing their available resources for the classes of applications where task partitioning is used. Thus, when considering task partitioning, a trade-off can be

observed between the offloading latency and the resources' utilization: task partitioning can achieve a lower offloading latency. On the other hand, more computation and wireless resources are required for maintaining the same task reliability.

4.8 Conclusion

We studied in this chapter the UAV-aided latency and reliability aware computation offloading problem in IoT networks, which would enable the provisioning of mission-critical smart services. We proposed two problems for optimizing the UAVs' positions, the UEs-to-UAV association, and the allocated resources considering redundancy of task computation for maximizing the admitted tasks' rate. This is done through a two-stage process where the planning problem optimizes the UAVs' position for the long-term offloading, and the computation offloading and resource allocation problem optimizes the other variables for the particular offloading instance considering the UAVs' available energy. We presented a solution approach for providing a low-complexity SCA-based algorithm for obtaining an approximate solution, and also approached the problem while considering task partitioning, which will be prevalent in 5G networks. Through numerical results, we demonstrated the solution efficiency and performed simulations for various scenarios, where the trade-off between reliability and resources utilization and between latency and reliability are explored. This work can offer valuable insights into provisioning computation offloading for modern services with stringent latency and reliability requirements through the aid of UAV-mounted cloudlets.

Chapter 5

Latency and Reliability Aware Computation Offloading in IRS-aided Edge-clouds[1]

Although MEC has allowed for computation offloading at the network edge, weak wireless signals in the RAN caused by obstacles and high network load are still preventing efficient edge computation offloading, especially for user requests with stringent latency and reliability requirements. IRS have been recently studied as a solution to enhance the quality of the signals in the RAN, where passive reflecting elements can be tuned to improve the uplink or downlink signals. Harnessing the IRS's potential in enhancing the performance of edge computation offloading, in this

[1]This chapter has been published in IEEE Communications Letters and submitted to IEEE Journal on Selected Areas in Communications - Special Issue on Multi-Tier Computing for Next Generation Wireless Networks.[119].

work, we study the use of an IRS-aided edge computing system for enabling 5G services with stringent latency and reliability computation offloading requirements, where the phase shifts of the IRS elements are optimized for the purpose of minimizing the UEs' energy consumption considering the cloudlets' failure rates and tasks redundancy. First, we consider a single-user network where the IRS elements' phase shift, the UE's offloading decision, and the UE's transmission power are optimized with the objective of minimizing the device's energy consumption. Then, in the second part of this chapter, we extend the study to a multi-user OFDMA network where we optimize the IRSs elements' phase shift, the UEs' offloading decisions, the UEs' transmission power, and the allocated servers' computational resources and OFDMA RBs. For each of the presented non-convex mathematical problems, we propose a customized sub-optimal solution based on the SCA approach and the SDR technique, where the problem is divided into multiple sub-problems that are solved separately in an alternating fashion. Numerical results are illustrated for the presented solutions, which demonstrate the energy reduction and network resources saving achieved by the optimized use of the IRSs especially for offloading services with a higher reliability requirement, and highlights on the IRSs' influence on the design of the MEC parameters.

5.1 Introduction

Although MEC has proved thus far to be a worthy technology for enabling the modern 5G services with their stringent requirements, there are still limitations concerning the RAN communication quality that are still hindering the full realization of MEC as a sustainable solution in 5G networks and beyond. Those limitations are the product of a poor communication environment caused by blockages, deep fading, or under-served areas, which cause the access channel quality to be degraded. Such situations cause a high latency to be incurred when conducting the offloading operation, which either forces the limited-capacity devices to spend a high offloading energy to satisfy the service requirements, or prevents the offloading procedure from happening altogether while wasting the utilization of edge resources and keeping them under-utilized. The problem is more aggravated when trying to satisfy the stringent latency and reliability requirements of the modern URLLC-type 5G services.

Multiple technologies have been explored so far in conjunction with MEC for enhancing the offloading performance, such as the utilization of UAVs, small cells, and the NOMA technology, where each have their advantages and limitations for their use in the context of edge computing. In order to tackle the communication channels impairments that prevent the URLLC-type applications from exploiting the MEC servers, the emerging IRS paradigm has been recently investigated, aiming to construct an SRE by converting the wireless propagation environment to an optimization variable that can be configured and controlled [33, 34]. Specifically, the IRS can passively and collaboratively enhance the RAN wireless communications quality

through exploiting a large number of low-cost and low-energy reflecting elements thanks to progress in programmable meta-materials, where the amplitude and/or phase shift of the elements' reflected signal can be tuned.

Now due to the observed difficulty in realizing low-energy MEC computation offloading for low-latency and high-reliability constrained services in a network with unfavorable wireless access channels, we envision the introduction of the IRS in the MEC system to play a significant role in reducing the overall devices' energy consumption while satisfying their strict requirements. Particularly, since the IRS can enhance the transmission rates and hence the upload latency, a load reduction on the MEC resources can be obtained which allows the devices to consume less offloading energy. However, in scenarios where the users need to offload to multiple cloudlets due to their reliability requirements, the judicious sharing of the IRS among the offloading users and the phase shifts optimization while minimizing the overall users' energy consumption is a challenging problem that we aim to tackle.

In this chapter, we investigate the problem of IRS-aided latency and reliability aware MEC computation offloading for the purpose of minimizing the UEs' energy consumption, considering cloudlets' failure rate and tasks redundancy where the task can be simultaneously offloaded to multiple cloudlets for guaranteeing its reliability. We start by studying a single-user network where we optimize the IRS elements' phase shift, the UE's transmission power, and the offloading decision while guaranteeing the task's stringent latency and reliability requirements. Then, in the second part of this chapter, we extend to a multi-user OFDMA network, where we jointly

optimize the IRS elements' phase shift, the offloading decisions, the UEs' transmission power, and the allocated servers' computational resources and OFDMA RBs. The considered IRSs' optimization would enable the UEs' access to resources-rich MEC servers which are normally unreachable or require a high UEs' transmission energy due to low-quality channels. Our work provides insights on leveraging IRS-aided APs for reducing the energy consumption of UEs that are requesting services with low-latency and high-reliability requirements such as mission-critical applications, for influencing the design of the MEC network parameters, and for reducing the load on the MEC resources.

5.1.1 Novel Contributions

The contributions of this chapter can be summarized as follows:

1. For each of the studied single-user and multi-user cases, we model the problem and mathematically formulate it as a non-convex MI-NLP program.

2. Due to the non-convexity of the proposed problem, we propose a customized sub-optimal solution based on the SCA approach and the SDR technique, where the problem is divided into multiple sub-problems that are solved separately in an alternating fashion.

3. We present numerical results considering various scenarios where the solution performance is demonstrated, and studies into the reduction in the UEs' energy consumption and network resources utilization through the optimized use of the IRSs are conducted.

5.2 Literature Review

We discuss first the main papers that explored the use of IRS for enhancing the wireless communication in the access network. Then, we present studies that focused on the use of IRS in the context of edge computing for enhancing the computation offloading performance. Finally, we present related work that studied reliability and latency problems considering a MEC system.

In [34], the authors studied the use of one IRS in one cell where multiple single-antenna UEs are communicating with a multi-antenna AP, with the objective of minimizing the AP's transmission power by optimizing the transmission beamforming and IRS phase shifts while respecting the users' required quality of service. In [33], the authors provided an overview of the use of IRS in wireless networks, and discussed its advantages and challenges and architecture. Also, numerical results are presented that show the performance improvement brought by the use of IRS.

In [120], the authors proposed a BCD-based algorithm for minimizing the latency of computation offloading considering a set of users communicating with a multi-antenna AP, where the computation and communication resources are optimized, and the IRS's impact on the performance improvement is demonstrated. In [121], the authors developed a solution for optimizing the IRS phase shifts for maximizing the operator's earnings in terms of the devices' payments while minimizing the users' weighted sum of latency and energy. In [122], the authors developed a solution for performing wireless power transfer in an OFDMA system through the use of IRS in an edge computing setting where the operator's energy consumption is minimized. In [123], the authors studied an IRS-aided MEC system where the IRS phase shift

is optimized for maximizing the number of offloaded bits from a set of users to one AP. The authors in [124] minimized the sum energy consumption through a BCD approach for a set of users communicating with a cloudlet-enabled AP where the IRS phase shift along with the offloading data, transmission power, and TDMA resources' allocation are optimized considering NOMA. The authors in [125] maximized the computation offloading rate in an IRS-aided MEC system to explore the offloading performance considering the TDMA and NOMA multiple access schemes.

In [47], the authors minimized the servers' transmission power by studying the latency and reliability efficiency in a MEC system for a set of users. The authors in [95] studied the reliability in distributed edge clouds where a reliability indicator is introduced for each edge node, and computational tasks are then replicated to specific computing nodes. [77] studied the trade-off between latency and reliability in a MEC system, where the end-to-end latency and the failure probability of computation offloading is minimized.

In contrast to the existing studies that considered IRS-aided MEC systems, in this work, we explore the use of multiple IRSs for helping to minimize the total energy consumption for a set of UEs while guaranteeing the tasks' stringent latency and reliability requirements. Here, redundancy is performed by simultaneously offloading the tasks to multiple servers to guarantee their computing reliability, where the optimized use of the IRSs which need to be judiciously shared in this scenario, are performed along with the offloading and resources allocation parameters. In this context, the IRS elements which influence the offloading latency can be optimized such as to reduce the load on the MEC resources and also to affect the design of the

MEC parameters for achieving a minimal energy consumption. Hence, the use of the IRSs is explored to cater for the limited devices' energy and capability, which would contribute towards enabling future mission-critical services in IoT networks.

5.3 Single-user case

In this section, we present our proposed mathematical problem for the single-user case in Subsection 5.3.1, and due to the non-convexity of the problem, we propose in Subsection 5.3.2 the customized solution approach discussed as follows. After performing the offloading decision, the problem is divided into two sub-problems that are solved iteratively. First, the transmission power initial value is used as an input for optimizing the IRS elements' phase shifts which is solved using a novel SCA-based technique that provides an approximate solution by iteratively solving until convergence after the problem is converted to a DC representation [41]. Then, the transmission power is optimized given the input values for the IRS elements' phase shifts through a novel SCA-based algorithm. The two sub-problems are solved in an alternating fashion until the convergence of the objective. Finally, Subsection 5.3.3 discusses the numerical analysis.

5.3.1 System Model and Problem Formulation

5.3.1.1 Spatial Model

As illustrated in Fig. 5.1, we consider an IRS-enabled edge computing system that consists of M APs denoted by $\mathcal{M} = \{1, 2, ..., M\}$ that are located within the reach

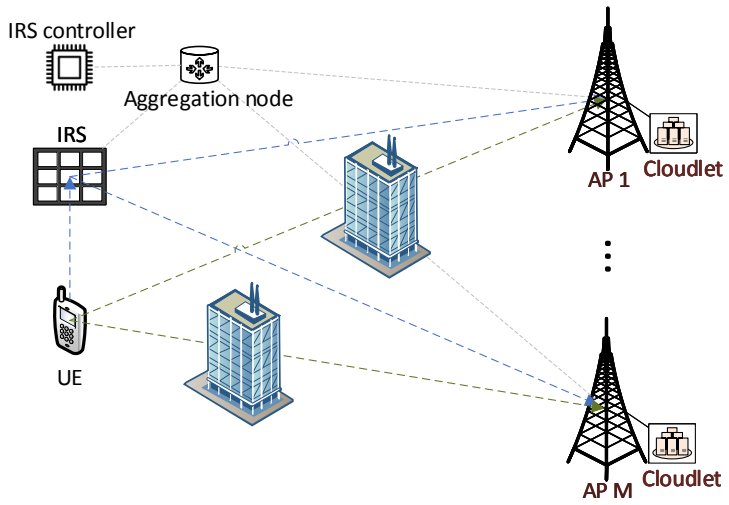


Figure 5.1: Single-user system model.

of a UE that needs to offload its computational task to carry its service. Each AP $j \in \mathcal{M}$ is equipped with a cloudlet of capacity F_j GHz for serving the UE's offloading request. Furthermore, each AP $j \in \mathcal{M}$ has a reliability denoted by ϕ_j which can be obtained after estimating the occurrence probability of failure scenarios through statistical means based on the cloudlets' historical failure pattern and maintenance records [108]. The node failures corresponding to the reliability ϕ_j are mainly caused by either software failures or hardware failures such as hard disk, memory and RAID controller failures [105]. An IRS comprised of N reflecting elements denoted by $\mathcal{N} = \{1, 2, ..., N\}$ is deployed to assist the UE's offloading to the APs, via generating passive beamforming coordinated by the IRS controller. We introduce a decision variable $x_j \in \{0, 1\}$ to indicate if the UE is offloading its task to cloudlet $j \in \mathcal{M}$. The UE's task is represented by the tuple $\{D, C, \bar{L}, \bar{R}\}$, concatenating the task input size D Kb, computational demand density C CPU cycles/bit, required latency $\bar{L}$ ms,

and required reliability $\bar{R}$.

5.3.1.2 Communication Model

We consider the uplink communication between the UE and the APs to be accommodated on the whole spectrum of B Hz, while the downlink communication is ignored since the task output size is in general much smaller than the task input size. The channel coefficients between the UE and AP j, the UE and the IRS, as well as the IRS and AP j, are denoted by h_j, $\boldsymbol{h}^{\mathrm{I}} = [h_1^{\mathrm{I}}, h_2^{\mathrm{I}}, ..., h_N^{\mathrm{I}}]$, $\boldsymbol{g}_j = [g_{j1}, g_{j2}, ..., g_{jN}]^T$, respectively, which are assumed to be perfectly estimated. We note that the IRS channels $\boldsymbol{h}^{\mathrm{I}}$ and $\boldsymbol{g}_j$ are assumed to be of a much better quality than the direct channel h_j. We set the amplitude reflection coefficient to 1 for all IRS reflection elements, and denote the phase shift coefficient vector by $\boldsymbol{\theta} = [\theta_1, \theta_2, ..., \theta_N]^T$, where $\theta_n \in [0, 2\pi)$ for all $n \in \mathcal{N}$. Then, the reflection-coefficient vector of the IRS is denoted by $\boldsymbol{v} = [e^{i\theta_1}, e^{i\theta_2}, ..., e^{i\theta_N}]^T$, where i represents the imaginary unit, and we define $\boldsymbol{\Theta} = \mathrm{diag}(\boldsymbol{v})$. By denoting $p \geq 0$ as the variable representing the UE's allocated transmission power out of the maximum power $\bar{P}$ and N_0 as the noise power spectral density, the UE's achieved rate for transmitting to AP j is:

$$R_j(\boldsymbol{\theta}, p) = B \log_2 \left(1 + \left(p|h_j + \boldsymbol{h}^{\mathrm{I}} \boldsymbol{\Theta} \boldsymbol{g}_j|^2 / BN_0\right)\right) \tag{5.1}$$

5.3.1.3 Reliability Model

For guaranteeing the task's reliability, redundancy is performed where the task can be offloaded and computed on multiple cloudlets in parallel. By denoting $\boldsymbol{x} = \{x_j, \forall j \in$

$\mathcal{M}\}$, the achieved task reliability is then represented by:

$$\Phi(\boldsymbol{x}) = 1 - \prod_{j \in \mathcal{M}} (1 - x_j \phi_j) \tag{5.2}$$

A higher reliability is achieved when the task is computed on more cloudlets, but this increases the consumption of bandwidth and computing resources. The considered server reliability and failure model are similar to the ones for cloud servers and were also considered in the context of edge computing [126].

5.3.1.4 Problem Formulation

We aim at minimizing the UE's offloading energy consumption while respecting the required task latency and reliability, by optimizing the offloading decision $\boldsymbol{x}$, the IRS phase shift $\boldsymbol{\theta}$, and the transmission power p. The IRS-aided latency and reliability aware MEC offloading problem $\mathcal{P}_1$, is formulated as:

$$\mathcal{P}_1 : \min_{\boldsymbol{x}, \boldsymbol{\theta}, p, \tau} \quad p\,(D/\tau) \tag{5.3a}$$

$$\text{s.t.} \quad R_j(\boldsymbol{\theta}, p) \geq x_j \tau, \ \forall j \in \mathcal{M} \tag{5.3b}$$

$$x_j\,(D/R_j(\boldsymbol{\theta}, p) + DC/F_j) \leq \bar{L}, \ \forall j \tag{5.3c}$$

$$\Phi(\boldsymbol{x}) \geq \bar{R} \tag{5.3d}$$

$$x_j \in \{0, 1\}, \theta_n \in [0, 2\pi), p \in [0, \bar{P}] \tag{5.3e}$$

Notation	Description
$\mathcal{M}$	Set of APs
d	Task input size (Kb)
c	Task computational density (CPU cycles/bit)
$\bar{L}$	Task latency deadline (ms)
$\bar{R}$	Task reliability requirement
F_j	Cloudlet computational capacity (GHz)
ϕ_j	Cloudlet reliability
h_j	Channel coefficient between the UE and AP j
$\boldsymbol{h}^{\mathrm{I}}$	Channel coefficient between the UE and the IRS
$\boldsymbol{g}_j$	Channel coefficient between the IRS and AP j
$\boldsymbol{v}$	IRS reflection-coefficient vector
B	Radio spectrum bandwidth (MHz)
N_0	Noise power spectral density (dBm/MHz)
$x_j \in \{0, 1\}$	Indicates if the UE's task is offloaded to cloudlet j
$\boldsymbol{\theta}$	Phase shift coefficient vector
$p \in \mathbb{R}^+$	UE's transmission power (watts)

Table 5.1: Table of Notations

where (5.3a) minimizes the UE's consumed energy, which is the product of the transmission power and the highest upload latency. (5.3b) sets τ to be the minimum transmission rate which is being maximized. (5.3c) makes sure the total UE's offloading latency on AP j respects the latency deadline. (5.3d) guarantees the required task reliability. Finally, (5.3e) is for the integrality and variable bounding conditions. All mathematical symbols used thus far are summarized in Table 5.1.

Problem (5.3) is non-convex due to the coupling between the phase shift $\boldsymbol{\theta}$ and transmission power p variables as well as the binary offloading indicator in (5.3c). In general, solving problem (5.3) optimally is very difficult, and there are no standard methods for providing such a solution.

5.3.2 Proposed Iterative Low-Complexity Algorithm

We propose a customized sub-optimal solution to solve (5.3).

A. Offloading Decision Optimization: First, we decouple the offloading decision optimization from other variables. We construct the set $\mathcal{S}$ with each $\hat{\mathcal{M}}^{(m)} \in \mathcal{S}$ representing a subset of APs $\mathcal{M}$ that satisfies (5.3d) which can later be omitted, i.e. $\Phi(\boldsymbol{x}^{(m)}) \geq \bar{R}$ where $\boldsymbol{x}_j \in \boldsymbol{x}^{(m)}$ is set to 1 if AP $j \in \hat{\mathcal{M}}^{(m)}$, and to 0 otherwise. Thus, given $\hat{\mathcal{M}}^{(m)}$, $\boldsymbol{\theta}$ and p are optimized as:

$$\mathcal{P}^{(m)}: \min_{\boldsymbol{\theta},p,\tau} \quad p\,(D/\tau) \tag{5.4a}$$

$$\text{s.t.} \quad R_j(\boldsymbol{\theta}, p) \geq \tau,\ \forall j \in \hat{\mathcal{M}}^{(m)} \tag{5.4b}$$

$$R_j(\boldsymbol{\theta}, p) \geq DF_j/(\bar{L}F_j - DC),\ \forall j \tag{5.4c}$$

$$\theta_n \in [0, 2\pi), 0 \leq p \leq \bar{P} \tag{5.4d}$$

which is still non-convex. Thus, we approach a customized solution for $\mathcal{P}^{(m)}$ by decoupling the optimization of the UE's transmission power from that of the IRS phase shift, which are later solved in an alternating fashion. Specifically, in each iteration, the IRS phase shift is first optimized given $\hat{p}$ which is optimized in the previous iteration. Then, the transmission power is again optimized given the just obtained $\hat{\boldsymbol{\theta}}$ as input.

B. IRS Phase Shift Optimization: For the given transmission power value $\hat{p}$,

the IRS phase shift $\boldsymbol{\theta}$ can be optimized as:

$$\mathcal{P}_{a1}^{(m)}: \max_{\boldsymbol{\theta},\tau} \quad \tau \tag{5.5a}$$

$$\text{s.t.} \quad R_j(\boldsymbol{\theta}, \hat{p}) \geq \tau, \ \forall j \in \hat{\mathcal{M}}^{(m)} \tag{5.5b}$$

$$R_j(\boldsymbol{\theta}, \hat{p}) \geq DF_j/(\bar{L}F_j - DC), \ \forall j \tag{5.5c}$$

$$\theta_n \in [0, 2\pi) \tag{5.5d}$$

where $\boldsymbol{\theta}$ is optimized such as the minimum transmission rate among the associated APs is maximized, which is represented by τ. We remark that the obtained problem would be a generalized convex problem due to the appearance of the generalized exponential cone constraints (5.5b) and (5.5c), which still makes the problem challenging due to the very high computation time resulting from solving the problem using a generalized convex solver such as FMINCON. In contrast, a standard convex program such as the Second Order Cone Programming (SOCP) can be solved much more rapidly by the commercial solver MOSEK while achieving an accuracy of 99.99% [61]. Thus, we are motivated to employ the conic approximation with controlled accuracy in [61], where constraints (5.5b) and (5.5c) can be rewritten by

a set of second order cone inequalities as

$$\kappa_{m+4}^{j} \leq 1 + \left(\hat{p}|h_j + \boldsymbol{h}^{\mathrm{I}}\boldsymbol{\Theta}\boldsymbol{g}_j|^2/BN_j\right) \tag{5.6}$$

$$\begin{aligned}
u_j &\geq \tau \\
u_j &\geq DF_j/(\bar{L}F_j - DC) \\
1 + \kappa_1^j &\geq \left\| \begin{bmatrix} 1 - \kappa_1^j & 2 + u_j/B2^{m-1} \end{bmatrix} \right\|_2 \\
1 + \kappa_2^j &\geq \left\| \begin{bmatrix} 1 - \kappa_2^j & 5/3 + u_j/B2^{m} \end{bmatrix} \right\|_2 \\
1 + \kappa_3^j &\geq \left\| \begin{bmatrix} 1 - \kappa_3^j & 2\kappa_1^j \end{bmatrix} \right\|_2 \\
\kappa_4^j &\geq \kappa_2^j + \kappa_3^j/24 + 19/72 \\
1 + \kappa_l^j &\geq \left\| \begin{bmatrix} 1 - \kappa_l^j & 2\kappa_{l-1}^j \end{bmatrix} \right\|_2 \ \forall\ l \in \{5, ..., s+3\} \\
1 + \kappa_{m+4}^j &\geq \left\| \begin{bmatrix} 1 - \kappa_{m+4}^j & 2\kappa_{m+3}^j \end{bmatrix} \right\|_2
\end{aligned} \tag{5.7}$$

where $\boldsymbol{u} = \{u_j \geq 0, j \in \hat{\mathcal{M}}^{(m)}\}$, $\boldsymbol{\kappa}_s = \{\kappa_s^j \geq 0, j \in \hat{\mathcal{M}}^{(m)}\}$, and s is the conic approximation parameter which can be chosen as $s = 4$ to attain the 99.99% accuracy. Also, by replacing the term $h_j + \boldsymbol{h}^{\mathrm{I}}\boldsymbol{\Theta}\boldsymbol{g}_j$ in (5.6) by $h_j + \boldsymbol{v}^H\boldsymbol{\Phi}_j$ with $\boldsymbol{\Phi}_j = \mathrm{diag}(\boldsymbol{h}^{\mathrm{I}})\boldsymbol{g}_j$, problem $\mathcal{P}_{a1}^{(m)}$ can be transformed to:

$$\mathcal{P}_{a2}^{(m)} : \max_{\boldsymbol{\theta}, \tau} \quad \tau \tag{5.8a}$$

$$\text{s.t.} \quad \kappa_{m+4}^{j} \leq 1 + \hat{p}/BN_j(\boldsymbol{v}^H\boldsymbol{\Phi}_j\boldsymbol{\Phi}_j^H\boldsymbol{v} + \boldsymbol{v}^H\boldsymbol{\Phi}_j h_j + h_j^H\boldsymbol{\Phi}_j^H\boldsymbol{v} + |h_j|^2) \tag{5.8b}$$

$$|v_n|^2 = 1,\ (5.7) \tag{5.8c}$$

By introducing a slack variable t, problem $\mathcal{P}_{a2}^{(m)}$ can be converted to a homogeneous QCQP as:

$$\mathcal{P}_{a3}^{(m)}: \underset{\boldsymbol{\theta},\tau}{max} \quad \tau \tag{5.9a}$$

$$\text{s.t.} \quad \kappa_{m+4}^{j} \leq 1+\hat{p}/BN_j\left(\bar{\boldsymbol{v}}^H\boldsymbol{R}_j\bar{\boldsymbol{v}}+|h_j|^2\right) \tag{5.9b}$$

$$|v_n|^2 = 1, \ (5.7) \tag{5.9c}$$

$$\boldsymbol{R}_j = \begin{bmatrix} \boldsymbol{\Phi}_j\boldsymbol{\Phi}_j^H & h_j\boldsymbol{\Phi}_j \\ h_j\boldsymbol{\Phi}_j^H & 0 \end{bmatrix} \forall j, \ \bar{\boldsymbol{v}} = \begin{bmatrix} \boldsymbol{v} \\ t \end{bmatrix}$$

We note that $\bar{\boldsymbol{v}}^H\boldsymbol{R}_j\bar{\boldsymbol{v}} = \text{tr}(\boldsymbol{R}_j\bar{\boldsymbol{v}}\bar{\boldsymbol{v}}^T)$. By introducing variable $\boldsymbol{V} = \bar{\boldsymbol{v}}\bar{\boldsymbol{v}}^T$, problem $\mathcal{P}_{a3}^{(m)}$ can be transformed to:

$$\mathcal{P}_{a4}^{(m)}: \underset{\boldsymbol{V}\succcurlyeq 0,\tau}{max} \quad \tau \tag{5.10a}$$

$$\text{s.t.} \quad \kappa_{m+4}^{j} \leq 1+\hat{p}/BN_j\left(\text{tr}(\boldsymbol{R}_j\boldsymbol{V})+|h_j|^2\right) \tag{5.10b}$$

$$\boldsymbol{V}_{n,n} = 1, \ \forall n \in \{1,2,...,N+1\} \tag{5.10c}$$

$$\boldsymbol{V} \succcurlyeq 0, \ \text{rank}(\boldsymbol{V}) = 1, \ (5.7) \tag{5.10d}$$

which is a non-convex SDP where (5.10d) indicates that $\boldsymbol{V}$ is a semi-definite matrix with a non-convex rank-one constraint, which makes solving the problem very challenging. One widely adopted technique is the SDR [127], where the rank-one constraint is relaxed, making the problem a convex SDP which can be easily solved before using the Gaussian randomization to construct a rank-one solution from $\boldsymbol{V}$ [127].

However, this technique does not guarantee obtaining a feasible or a rank-one solution to (5.10) especially when the problem size grows. Thus, we propose a novel SCA approach for solving (5.10) which is guaranteed to converge where the rank-one constraint is written in a DC form.

With $\sigma_r(\boldsymbol{V})$ defined as the r-th largest singular value of $\boldsymbol{V}$, the rank-one constraint for $\boldsymbol{V}$ indicates that $\sigma_1(\boldsymbol{V}) > 0$ and $\sigma_r(\boldsymbol{V}) = 0, \forall i \in \{2, 3, ..., N+1\}$. Thus, by defining the trace norm and spectral norm of $\boldsymbol{V}$ as $\mathrm{Tr}(\boldsymbol{V}) = \sum_{n=1}^{N+1} \sigma_r(\boldsymbol{V})$ and $||\boldsymbol{V}|| = \sigma_1(\boldsymbol{V})$, respectively, the rank-one constraint can be equivalently rewritten as the difference of these two convex norms as $\mathrm{Tr}(\boldsymbol{V}) - ||\boldsymbol{V}|| = 0$ with $\mathrm{Tr}(\boldsymbol{V}) > 0$. Hence, problem (5.10) can be converted into the following DC program:

$$\mathcal{P}_{a5}^{(m)}: \underset{\boldsymbol{V} \succcurlyeq 0, \tau}{min} \quad -\tau + (\mathrm{Tr}(\boldsymbol{V}) - ||\boldsymbol{V}||) \tag{5.11a}$$

$$\text{s.t.} \quad (5.7), (5.10b), (5.10c) \tag{5.11b}$$

which is still non-convex due to the concave term $-||\boldsymbol{V}||$ in (5.11a). Thus, we proceed to linearize $||\boldsymbol{V}||$ by substituting it with the term $\tilde{g}(\boldsymbol{V}; \boldsymbol{V}^{(m)}) = \left\langle \psi||\boldsymbol{V}^{(m)}||, \boldsymbol{V} \right\rangle$, where $\boldsymbol{V}^{(m)}$ is the solution obtained at the previous SCA iteration, and $\psi||\boldsymbol{V}^{(m)}||$ denotes the sub-gradient of spectral norm at point $\boldsymbol{V}^{(m)}$. We note that one sub-gradient of $||\boldsymbol{V}||$ can be efficiently computed as $\boldsymbol{q}_1\boldsymbol{q}_1^H$, where $\boldsymbol{q}_1$ is the vector corresponding to the largest singular value $\sigma_1(\boldsymbol{V})$. Thus, $\mathcal{P}_{a5}^{(m)}$ can be replaced by:

$$\mathcal{P}_{a6}^{(n,m)}: \underset{\boldsymbol{V} \succcurlyeq 0, \tau}{min} \quad -\tau + \Big(\mathrm{Tr}(\boldsymbol{V}) - \tilde{g}(\boldsymbol{V}; \boldsymbol{V}^{(m)})\Big) \tag{5.12a}$$

$$\text{s.t.} \quad (5.7), (5.10b), (5.10c) \tag{5.12b}$$

which is a standard convex SDP that is solved at each SCA iteration m until convergence given an initial $\boldsymbol{V}^{(0)}$. Specifically, we design a practical stopping criterion with $\mathrm{Tr}(\boldsymbol{V}) - \tilde{g}(\boldsymbol{V}; \boldsymbol{V}^{(m)}) < \epsilon$, where ϵ is a sufficiently small positive constant. After obtaining a feasible $\boldsymbol{V}$, the phase shift matrix $\boldsymbol{\theta}$ can be then easily recovered from $\boldsymbol{V}$.

C. Transmit Power Optimization: For the given values of $\hat{\boldsymbol{\theta}}$, the transmission power p can be optimized as:

$$\mathcal{P}_b^{(m)}: \min_{p,\tau} \quad p\,(D/\tau) \tag{5.13a}$$

$$\text{s.t.} \quad \kappa_{m+4}^{j} \leq 1 + \left(p|h_j + \boldsymbol{h}^{\mathrm{I}}\hat{\boldsymbol{\Theta}}\boldsymbol{g}_j|^2/BN_j\right) \tag{5.13b}$$

$$0 \leq p \leq \bar{P}, \ (5.7) \tag{5.13c}$$

which is a non-convex program due to (5.13a) being neither convex nor concave in general. Thus, we introduce slack variable δ which will equivalently replace (5.13a), and slack variable Λ, and equivalently define the following constraints:

$$\delta\tau \geq \Lambda^2 \tag{5.14a}$$

$$p - \Lambda^2 \leq 0 \tag{5.14b}$$

Here, (5.14a) is a quadratic conic convex constraint, while (5.14b) is non-convex due to the concave function $f(\Lambda) = -\Lambda^2$ which renders the left side of (5.14b) as a DC form. Thus, with $\Lambda^{(m)}$ as the input point, we employ the SCA method to replace

$f(\Lambda)$ by its first order Taylor approximate as:

$$\tilde{f}(\Lambda;\Lambda^{(m)}) = -(\Lambda^{(m)})^2 - 2\Lambda^{(m)}(\Lambda - \Lambda^{(m)}) \tag{5.15}$$

Problem $\mathcal{P}_b^{(m)}$ can then be replaced by:

$$\begin{aligned}
\mathcal{P}_b^{(n,m)}: \underset{p,\tau,\Lambda,\delta}{min} \quad & \delta && \text{(5.16a)}\\
\text{s.t.} \quad & \kappa_{m+4}^{j} \leq 1 + \left(p|h_j + \boldsymbol{h}^{\mathrm{I}}\hat{\boldsymbol{\Theta}}\boldsymbol{g}_j|^2/BN_j\right) && \text{(5.16b)}\\
& \delta\tau \geq \Lambda^2 && \text{(5.16c)}\\
& p + \tilde{f}(\Lambda;\Lambda^{(m)}) \leq 0 && \text{(5.16d)}\\
& 0 \leq p \leq \bar{P}, \ (5.7) && \text{(5.16e)}
\end{aligned}$$

which is a SOCP that is solved at each iteration m until convergence, with $\Lambda^{(m)}$ being updated to the optimal Λ.

The pseudocode for the overall proposed algorithm is outlined in Algorithm 7, which is solved for each set $\hat{\mathcal{M}}^{(m)} \in \mathcal{S}$, and the solution achieving the lowest objective $\delta^{(m)}$ is returned. The convergence of Algorithm 7 can be guaranteed by showing that the series of resulting objective is monotonically convergent. Due to the non-increasing sequence of objective function values in the SCA algorithms for $\mathcal{P}_{a6}^{(n,m)}$ and $\mathcal{P}_b^{(n,m)}$, the objective values $\delta_a^{(m)}$ and $\delta_b^{(m)}$ at step $i+1$ are guaranteed to be less than or equal to $\delta_a^{(m)}$ and $\delta_b^{(m)}$ at step m, which means that Algorithm 7 generates a non-increasing sequence of objective function values $\delta_b^{(m)}$, which guarantees the objective convergence.

Algorithm 7 Overall Algorithm for solving $\mathcal{P}^{(m)}$

1: Set $m = 1$, $p^{(m)} = 1$, construct $\boldsymbol{x}^{(m)}$ based on $\hat{\mathcal{M}}^{(m)}$;
2: **repeat**
3: Obtain $\boldsymbol{\theta}^{(m)}$ and the objective $\delta_a^{(m)}$ through SCA for problem $\mathcal{P}_{a6}^{(n,m)}$ given the input $\boldsymbol{x}^{(m)}$ and $\hat{p} = p^{(m)}$.
4: Obtain $p^{(m)}$ and the objective $\delta_b^{(m)}$ through SCA for problem $\mathcal{P}_b^{(n,m)}$ given the input $\boldsymbol{x}^{(m)}$ and $\hat{\boldsymbol{\theta}} = \boldsymbol{\theta}^{(m)}$.
5: Set $m = m + 1$;
6: **until** Convergence of objective $\delta_b^{(m)}$.

The overall complexity of Algorithm 7 depends mainly on that of solving the SDP problem (5.12) and the SOCP problem (5.16). Since the complexity of (5.16) is approximately $\mathcal{O}(sNM)$ which is a polynomial time complexity with s as the conic approximation parameter [61], the complexity of solving (5.12) is the dominant one. The order of complexity for an SDP problem with m SDP constraints which includes an $n \times n$ PSD matrix, is given by $\mathcal{O}(\sqrt{n}\log(1/\epsilon)(mn^3 + m^2n^2 + m^3))$ where $\epsilon > 0$ is the solution accuracy [128, Th. 3.12]. In our case, we have $n = N + 1$ and $m = M + N + 1$; but since N is much larger than M in general, the approximate computation complexity for solving (5.12) (and Algorithm 7), can be reduced to $\mathcal{O}(\log(1/\epsilon)N^{4.5})$.

5.3.3 Numerical Results

We study in this section the solution performance considering a UE that is positioned in the center of a 2-D circular area with a diameter of 200 meters, an IRS positioned within 10 meters of the UE, and $M = 5$ randomly distributed APs. The system parameters are those presented in Table 5.4 unless otherwise specified.

Parameter	Value
Task input size D	50 Kb
Task computational demand C	200 cycles/bit
Task latency deadline $\bar{L}$	30 ms
Task reliability requirement $\bar{R}$	[0.9955 0.9999]
UAV cloudlet capacity F_j	3 GHz
UAV cloudlet reliability ϕ_j	[0.9955 0.9999]
Path loss at 1 meter g_0	-30 dB [123]
Path loss exponents (α_1, α_2)	$(4, 2.2)$
Rician Factor	2
UE Transmission power Threshold P	30 dBm [124]
Radio spectrum bandwidth B	1 MHz [124]
Noise power spectral density N_0	-174 dBm/MHz [124]

Table 5.2: Simulation Parameters

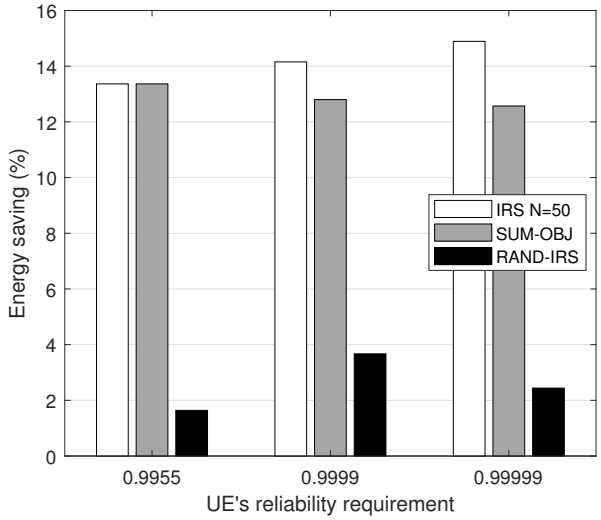


Figure 5.2: Energy gain incurred from the use of the IRS.

In Fig. 5.2, we present the UE energy saving achieved through the IRS use that is obtained from our solution with $\bar{L} = 30$ ms, and compare it to SUM-OBJ where the sum of transmission rates is maximized in (5.12), and RAND-IRS where

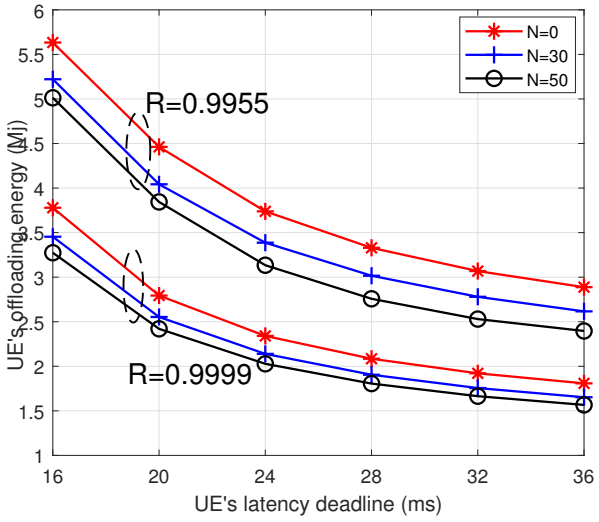


Figure 5.3: Objective value vs reliability requirement and number of IRS elements.

the IRS phase shifts are randomly configured. The proposed solution with $N = 50$ outperforms the other approaches, achieving an energy saving up to 15%, compared to a saving below 5% from a random IRS configuration. Also, the gain over SUM-OBJ is negligible when one communication channel is utilized, and becomes higher with the use of more APs. In addition, the energy saving becomes larger when the UE's reliability requirement is higher, since the IRS can better enhance the lower quality channels when more APs are utilized which reduces the UE's transmission power and hence the energy.

In Fig. 5.3, we study how the UE's energy consumption is influenced by the IRS size for different cases of latency and reliability. When the task has more strict latency deadline, the energy consumption is increased since a higher transmission power is needed to respect the latency bound. Also, a higher reliability requirement causes an increase in energy since the UE needs to compute its task on more cloudlets

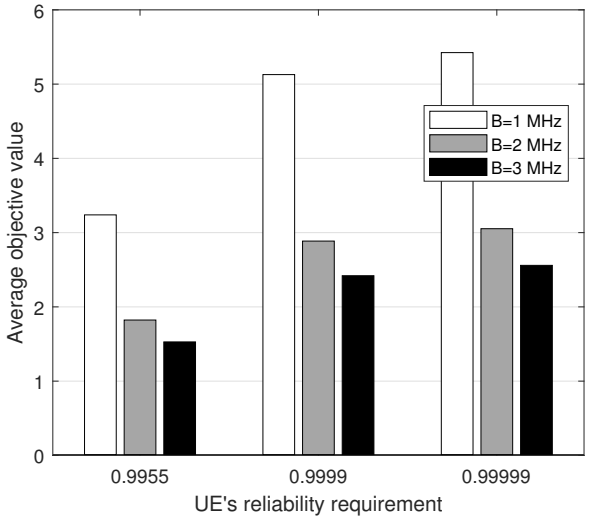


Figure 5.4: Objective value vs reliability requirement and radio bandwidth.

that have lower quality channels, forcing the UE to allocate more transmission power to meet the latency bound. Moreover, a larger IRS can significantly counteract the increase in UE energy consumption by selectively enhancing the channels' quality in response to more stringent service requirements. Here, it is observed that a higher channel gain is allocated through the IRS for the MEC server that is incurring a higher computation latency, while for homogeneous servers the IRS optimization is maximizing the minimum channel gain which would be the same without MEC. Also, we note the influence of the IRS on the offloading decision, such as allowing the use of a larger MEC server by selectively enhancing its channel, instead of offloading to a smaller MEC server which incurs a higher computing latency.

In Fig. 5.4, we study how a change in the allocated system bandwidth can influence the UE's energy consumption for different service reliability levels. As previously discussed, more energy is consumed when a more strict reliability is required

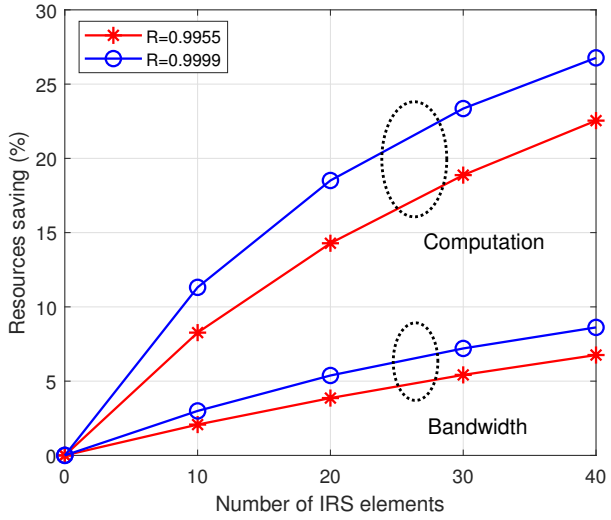


Figure 5.5: Resources saving vs number of IRS elements.

due to a higher transmission power needed for reaching more APs. Now, allocating more bandwidth can maintain the same levels of the needed transmission rate, which avoids the increase in energy consumption in response to a higher reliability requirement. Also, the energy decrease resulting from a higher system bandwidth is more prominent when the bandwidth is already small, and the rate of energy increase is lower in the third reliability case because some instances might require offloading to the same number of APs to meet their reliability.

In Fig. 5.5, we study the impact of the IRS size on the resources saving assuming the same UE energy consumption where the IRS is tailored towards resources saving, which can often be the operators' focus. With a larger IRS, a higher saving in computing and bandwidth resources is achieved. However, a higher saving in computing resources is possible in that case (close to 30%) compared to bandwidth resources (up to 10%), which can bring a great advantage to operators in saving energy and

costs related to computational resources. Also, the saving gains are more significant for services with higher reliability since the IRS can better improve on lower quality channels which can require less resources to meet the latency bound.

5.4 Multi-user case in OFDMA-based Networks

In this section, we extend our study to consider a multi-user OFDMA network. First, we present our proposed mathematical problem in Subsection 5.4.1, and due to the non-convexity of the problem, we propose in Subsection 5.4.2 the customized solution approach discussed as follows. First, the offloading decision is performed using a matching game algorithm. Then, the other parameters' design is performed through two sub-problems that are solved iteratively. First, the IRS elements' phase shifts are optimized through a novel technique based on the SCA approach, which provides an approximate solution by iteratively solving until convergence, after the sub-problem is converted to a DC representation. Then, the optimized values are used as inputs for the second sub-problem, where the UEs' transmission power and resource allocation parameters are also optimized through an SCA-based approach after the sub-problem is converted to a SOCP. The two sub-problems are solved in an alternating fashion until the objective convergence. Finally, Subsection 5.4.3 discusses the numerical analysis.

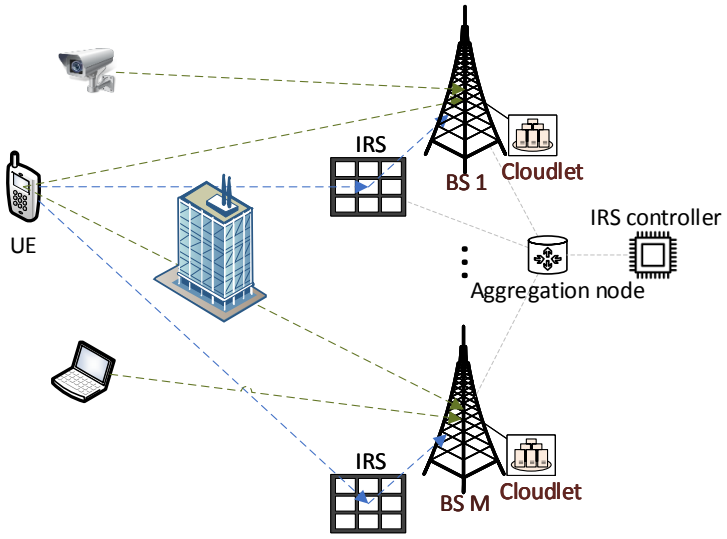


Figure 5.6: Multi-user OFDMA-based system model.

5.4.1 System Model and Problem Formulation

5.4.1.1 Spatial Model

As illustrated in Fig. 5.6, we consider an IRS-enabled edge computing system that consists of M access points (APs) denoted by $\mathcal{M} = \{1, 2, ..., M\}$ that are located within the reach of K IoT devices (UEs) denoted by $\mathcal{K} = \{1, 2, ..., K\}$, which need to offload their computational tasks to carry their service. Each AP $j \in \mathcal{M}$ is equipped with a cloudlet of computational capacity F_j (GHz) for serving the UEs' offloading requests. Furthermore, each cloudlet $j \in \mathcal{M}$ has a reliability denoted by ϕ_j which can be obtained after estimating the occurrence probability of failure scenarios through statistical means based on the cloudlets' historical failure pattern and maintenance records [108]. The node failures corresponding to the reliability ϕ_j are mainly caused by either software failures or hardware failures such as hard disk, memory and RAID

controller failures [105]. For each AP j, an IRS comprised of N reflecting elements denoted by $\mathcal{N}_j = \{1, 2, ..., N\}$ is deployed in order to assist the UEs' computation offloading, via generating passive beamforming coordinated by the IRS controller. We introduce decision variable $x_{kj} \in \{0, 1\}$ indicating if UE k is offloading its task to cloudlet $j \in \mathcal{M}$. The task of UE k is represented by a tuple $\{d_k, c_k, \bar{L}_k, \bar{R}_k\}$, concatenating the task input size d_k (Kb), computational demand density c_k (CPU cycles/bit), required latency $\bar{L}_k$ (ms), and required reliability $\bar{R}_k$.

5.4.1.2 Communication Model

We consider the uplink communication used by the UEs for computation offloading and ignore the downlink communication, since the task output size is in general much smaller than the task input size [57]. We consider an OFDMA system with assumed perfect channel state information (CSI), where the wireless radio spectrum is divided into B resource blocks (RBs), indexed by $\mathcal{B} = \{1, ..., B\}$. We denote by y_{kb} a binary decision variable indicating if RB b is assigned to UE k for the communication with the associated APs where multiple RBs can be assigned to UE k for task transmission. The channel coefficients on RB b between UE k and AP j, UE k and IRS j, as well as IRS j and AP j, are denoted by h_{kjb}, $\boldsymbol{h}^{\mathrm{I}}_{kjb} = [h^{\mathrm{I}}_1, h^{\mathrm{I}}_2, ..., h^{\mathrm{I}}_N]$, $\boldsymbol{G}_{jb} = [G_{j1}, G_{j2}, ..., G_{jN}]^T$, respectively, which are assumed to be perfectly estimated. We note that the IRS channels $\boldsymbol{h}^{\mathrm{I}}_{kjb}$ and $\boldsymbol{G}_{jb}$ are assumed to be of a much better quality than the direct channel h_{kjb}. We set the amplitude reflection coefficient to 1 for all IRS reflection elements, and denote the phase shift coefficient vector for IRS j by $\boldsymbol{\theta}_j = [\theta_{j1}, \theta_{j2}, ..., \theta_{jN}]^T$, where $\theta_{jn} \in [0, 2\pi)$ for all $j \in \mathcal{M}, n \in \mathcal{N}$. Then, the

reflection-coefficient vector of IRS j is denoted by $\boldsymbol{v}_j = [e^{i\theta_{j1}}, e^{i\theta_{j2}}, ..., e^{i\theta_{jN}}]^T$, where i represents the imaginary unit, and we define $\boldsymbol{\Theta}_j = \text{diag}(\boldsymbol{v}_j)$.

By denoting $\boldsymbol{y}_k = \{y_{kb}, \forall b \in \mathcal{B}\}$, the achieved transmission rate from UE k to AP j on RB b $R_{kjb}(\boldsymbol{\theta}_j)$, and the achievable rate $R_{kj}(\boldsymbol{\theta})$, are defined as:

$$R_{kjb}(p_k, \boldsymbol{\theta}_j) = W \log_2 \left(1 + \frac{p_k |h_{kjb} + \boldsymbol{h}^{\text{I}}_{kjb} \boldsymbol{\Theta}_j \boldsymbol{G}_{jb}|^2}{W N_0}\right) \tag{5.17a}$$

$$R_{kj}(\boldsymbol{y}_k, p_k, \boldsymbol{\theta}_j) = \sum_{b \in \mathcal{B}} y_{kb} R_{kjb}(p_k, \boldsymbol{\theta}_j) \tag{5.17b}$$

respectively, where p_k is a decision variable denoting UE k's transmission power on the assigned RBs out of the maximum power $\bar{P}_k$, W is the RB bandwidth, and N_0 is the white noise power level. We note that a large block fading is assumed, such as the channel fading is considered constant throughout the duration of the task transmission.

5.4.1.3 Latency and Reliability Models

The latency incurred from transmitting UE k's task to AP j, is given by:

$$L^{\text{u}}_{kj}(\boldsymbol{y}_k, p_k, \boldsymbol{\theta}_j) = \frac{d_k}{R_{kj}(\boldsymbol{y}_k, p_k, \boldsymbol{\theta}_j)} \tag{5.18}$$

When UE k offloads its task to cloudlet j, a portion $f_{kj} \geq 0$ of cloudlet j's computational resources F_j is allocated for computing UE k's task. Thus, the latency incurred from computing task k on cloudlet j depends on the allocated portion $f_{ij}F_j$

of the computational capacity, and is given by:

$$L_{kj}^{c}(f_{kj}) = \frac{d_k c_k}{f_{kj} F_j} \tag{5.19}$$

For guaranteeing the tasks' reliability, redundancies are performed where the task of each UE k can be offloaded and computed on multiple cloudlets in parallel. By denoting $\boldsymbol{x}_k = \{x_{kj}, \forall j \in \mathcal{M}\}$, the achieved reliability of task k is then represented by:

$$\Phi_k(\boldsymbol{x}_k) = 1 - \prod_{j \in \mathcal{M}} (1 - x_{kj}\phi_j) \tag{5.20}$$

A higher reliability can be achieved when task k is computed on more cloudlets, but this increases the consumption of network bandwidth and computing resources. The considered server reliability and failure model are similar to how they are usually computed for cloud servers and was also considered in the context of edge computing [126].

5.4.1.4 Problem Formulation

We aim at minimizing the total UEs' consumed energy while respecting the tasks' latency and reliability requirements and the network resources capacity, by optimizing the computation offloading matrix $\boldsymbol{x} = \{x_{kj}, \forall k \in \mathcal{K}, j \in \mathcal{M}\}$, the RBs association matrix $\boldsymbol{y} = \{y_{kb}, \forall k \in \mathcal{K}, b \in \mathcal{B}\}$, the UEs' transmission power vector $\boldsymbol{p} = \{p_k, \forall k \in \mathcal{K}\}$, the UEs' computational resources' allocation matrix $\boldsymbol{f} = \{f_{kj}, \forall k \in \mathcal{K}, j \in \mathcal{M}\}$, and the phase shift coefficient matrix $\boldsymbol{\theta} = \{\theta_{jn}, \forall j \in \mathcal{M}, n \in \mathcal{N}\}$. The latency and reliability aware IRS-aided edge computation offloading problem $\mathcal{P}_1$, is formulated

as follows:

$$\mathcal{P}_1 : \min_{\substack{\boldsymbol{x},\boldsymbol{y},\boldsymbol{p},\\ \boldsymbol{f},\boldsymbol{\theta},\boldsymbol{\tau}}} \quad \sum_{k\in\mathcal{K}} p_k(d_k/\tau_k) \tag{5.21a}$$

$$\text{s.t.} \quad R_{kj}(\boldsymbol{y}_k, p_k, \boldsymbol{\theta}_j) \geq x_{kj}\tau_k, \ \forall k, j \tag{5.21b}$$

$$x_{kj}\left(L^{\mathrm{u}}_{kj}(\boldsymbol{y}_k, p_k, \boldsymbol{\theta}_j) + L^{\mathrm{c}}_{kj}(f_{kj})\right) \leq \bar{L}_k \tag{5.21c}$$

$$\Phi_k(\boldsymbol{x}_k) \geq \bar{R}_k, \ \forall k \in \mathcal{K} \tag{5.21d}$$

$$\sum_{k\in\mathcal{K}} y_{kb} \leq 1, \ b \in \mathcal{B} \tag{5.21e}$$

$$\sum_{k\in\mathcal{K}} f_{kj} \leq 1, \ \forall j \in \mathcal{M} \tag{5.21f}$$

$$x_{kj}, y_{kb} \in \{0, 1\}$$

$$p_k \in [0, \bar{P}_k], f_{kj} \geq 0, \theta_{jn} \in [0, 2\pi) \tag{5.21g}$$

where (5.21a) minimizes the total UEs' consumed energy, which is the product of the transmission power and the highest upload latency. (5.21b) sets τ_k to be the minimum transmission rate for UE k which is being maximized. (5.21c) makes sure the total offloading latency of UE k's task on AP j composed of both the upload and computation latencies, respects the UE's latency deadline. (5.21d) is to guarantee task k's reliability requirement. (5.21e) ensures the orthogonal allocation of the communication resources in the APs' RAN. (5.21f) respects the computational capacity of all cloudlets. Finally, (5.21g) is for the integrality and variable bounding conditions. All mathematical symbols used thus far are summarized in Table 5.3.

Problem (5.21) is non-convex due to the coupling with respect to the phase shift

Notation	Description
$\mathcal{K}$	Set of UEs
$\mathcal{M}$	Set of APs
$\mathcal{B}$	Set of RBs
d_k	Task input size (Kb)
c_k	Task computational density (CPU cycles/bit)
$\bar{L}_k$	Task latency deadline (ms)
$\bar{R}_k$	Task reliability requirement
F_j	Cloudlet computational capacity (GHz)
ϕ_j	Cloudlet reliability
h_{kjb}	Channel coefficient between UE k and AP j on RB b
$\boldsymbol{h}^{\mathrm{I}}_{kjb}$	Channel coefficient between UE k and IRS j on RB b
$\boldsymbol{G}_{jb}$	Channel coefficient between AP j and its IRS on RB b
$\boldsymbol{v}_j$	IRS j's reflection-coefficient vector
B	Radio spectrum bandwidth (MHz)
N_0	Noise power spectral density (dBm/MHz)
$x_{kj} \in \{0,1\}$	Indicates if UE k's task is offloaded to cloudlet j
$y_{kb} \in \{0,1\}$	Indicates if RB b is assigned to UE k
$\boldsymbol{\theta}_j$	Phase shift coefficient vector for IRS j
$p_k \in \mathbb{R}^+$	UE k's transmission power (watts)
$f_{kj} \in \mathbb{R}^+$	Portion of allocated resources for UE k on cloudlet j (GHz)

Table 5.3: Table of Notations

$\boldsymbol{\theta}$ and RBs' allocation $\boldsymbol{y}$ as well as the binary computation offloading indicator $\boldsymbol{x}$ in (5.21c). In general, solving problem (5.21) optimally is very difficult, and there are no standard methods for providing such a solution. Thus, in the next sections, we propose a customized sub-optimal solution for solving the problem.

5.4.2 Proposed Iterative Low-Complexity Algorithm

In this section, we propose a sub-optimal solution in order to solve problem (5.21).

5.4.2.1 Offloading Decision Optimization

Fist, we decouple the computation offloading optimization from the other variables. We aim to define a policy for optimizing the UEs' offloading decision where the maximum average offloading latency from the UEs to the APs is minimized. We note that other policies for optimizing the offloading decisions can be followed by the NO. For the purpose of optimizing the computation offloading decisions, we consider the UEs to offload their requests through the average obtained rate with all RBs ($y_{kb} = 1 \ \forall (k,b) \in (\mathcal{K}, \mathcal{B})$) using the maximum available power ($p_k = \bar{P}_k$) on the direct signal with the APs without assistance from the IRS ($\boldsymbol{\theta}$ is omitted). Thus, equations (5.17b) and (5.18) can be replaced by:

$$\hat{R}_{kj}(\boldsymbol{x}_j) = \frac{\sum_{b\in\mathcal{B}} W \log_2\left(1 + \frac{\bar{P}_k |h_{kjb}|^2}{W N_0}\right)}{\sum_{k\in\mathcal{K}} x_{kj}} \tag{5.22a}$$

$$\begin{aligned}\hat{L}^{\mathrm{u}}_{kj}(\boldsymbol{x}_j) &= \frac{D}{\hat{R}_{kj}(\boldsymbol{x}_j)} \\ &= \frac{D \sum_{k\in\mathcal{K}} x_{kj}}{W \sum_{b\in\mathcal{B}} \log_2\left(1 + \frac{\bar{P}_k |h_{kjb}|^2}{W N_0}\right)}\end{aligned} \tag{5.22b}$$

Then, the computation offloading matrix $\boldsymbol{x}$ is optimized through problem $\mathcal{P}_a$ as:

$$\mathcal{P}_a : \min_{\boldsymbol{x}} \quad \gamma \tag{5.23a}$$

$$\text{s.t.} \quad x_{kj}\left(\hat{L}^{\mathrm{u}}_{kj}(\boldsymbol{x}_j) + L^{\mathrm{c}}_{kj}(\boldsymbol{x}_j)\right) \leq \gamma, \ \forall \ k, j \tag{5.23b}$$

$$\ln(1 - \bar{R}_k) \geq \sum_{j\in\mathcal{M}} x_{kj} \ln(1 - \phi_j) \tag{5.23c}$$

$$x_{kj} \in \{0, 1\} \tag{5.23d}$$

where the objective (5.23a) aims to minimize the maximum incurred offloading latency from each UE to each associated AP represented by γ in (5.23b). Constraint (5.23c) is the linear form of (5.21d) where the transformation steps are detailed in Appendix A.4. Problem $\mathcal{P}_a$ is a mixed-integer convex problem. One solution to make the problem more tractable and scalable, is to relax the binary condition of $\boldsymbol{x}$ where post-processing steps will be performed for fixing the variables into their binary form. However, this would provide a low-quality solution compared to the binary solution. Thus, we propose a customized many-to-many matching game algorithm for solving the problem.

First, we convert the game to a many-to-one matching game by forming the set $\mathcal{S}$ which includes all possible APs subsets, and by defining the matching function $\Omega : \{\mathcal{K}, \mathcal{S}, \succ_{\mathcal{K}}, \succ_{\mathcal{S}}\}$. The preference utility for each UE $k \in \mathcal{K}$ when it is assigned to subset $s \in \mathcal{S}$ is set to be the maximum offloading latency incurred from offloading to all APs $j \in s$ which needs to be minimized, and is defined as:

$$U_k(s) = \max \hat{L}^{\mathrm{u}}_{kj}(\boldsymbol{x}_j) + L^{\mathrm{c}}_{kj}(\boldsymbol{x}_j), \forall j \in s \tag{5.24}$$

The pseudocode for the proposed matching game algorithm for solving the offloading decision optimization sub-problem, is outlined in Algorithm 8. In each iteration, each UE $k \in \mathcal{K}$ calculates $minS(k)$ as the subset that provides the minimum offloading cost $minCost(k)$. Then, if $minCost(k)$ is less than the cost of offloading previously calculated, UE k deviates to subset $minS(k)$ which provides the minimum utility $U_k(s)$. The algorithm converges when no UE has deviated. Finally, the output Ω of Algorithm 8 is mapped to the computation offloading matrix $\boldsymbol{x}$ of problem $\mathcal{P}_a$.

The Ping-pong effect is a known effect where some UEs might be assigned back and forth together to the same subsets, preventing the matching game algorithm from converging. In order to avoid that problem, we define the set $Ex(k), \forall k \in \mathcal{K}$ which will include the previously assigned APs' subset anytime UE k deviates. Then, in future iterations, UE k cannot be assigned to any of the subsets included in $Ex(k)$.

Algorithm 8 UE's Assignment Matching Algorithm

Input: $U_k \ \forall k \in \mathcal{K}$
Initialization: $\Omega = \{\Omega(k)\}_{k\in\mathcal{K}} = \varnothing$, $bestCost = 100$, $Ex = \{Ex(k)\}_{k\in\mathcal{K}} = \varnothing$;
while true **do**
    for $k \in \mathcal{K}$ **do**
        Compute $U_k(s), \forall \{s \in \mathcal{S}\} \notin Ex(k)$
        Compute $(minS(k), minCost(k)) = \min(U_k)$
        if $minCost(k) < bestCost(k)$ **then**
            Set $bestCost(k) = minCost(k)$
            $Ex(k) = Ex(k) \cup \{\Omega(k)\}$
            Deviate to the new subset: $\Omega(k) = minS(k)$
    if no UE $k \in \mathcal{K}$ has deviated **then**
        Stop the execution.
Map Ω to $\boldsymbol{x}^\star$ and return the solution $\boldsymbol{x}^\star$.

Convergence Analysis: At each iteration, a UE $k \in \mathcal{K}$ can either preserve the same matching, or be matched with a new subset that has not been assigned to her/him before. After enough iterations, each UE $k \in \mathcal{K}$ will have either exhausted all possible matchings, or has maintained a matching that provides her/him with the best preference utility, and hence no more UEs deviation will occur. Thus, the algorithm converges to a stable matching [129].

5.4.2.2 IRS Phase Shift Optimization and Resources Allocation

After optimizing the computation offloading matrix $\boldsymbol{x}$ through Algorithm 8, the RBs association matrix $\boldsymbol{y}$, the UEs' transmission power vector $\boldsymbol{p}$, the UEs' computational resources' allocation matrix $\boldsymbol{f}$, and the IRS phase shift matrix $\boldsymbol{\theta}$, can be optimized through problem $\mathcal{P}_b$ as:

$$\mathcal{P}_b : \underset{\substack{\boldsymbol{y},\boldsymbol{p},\boldsymbol{f},\\ \boldsymbol{\theta},\boldsymbol{\tau}}}{min} \quad \sum_{k\in\mathcal{K}} p_k(d_k/\tau_k) \tag{5.25a}$$

$$\text{s.t.} \quad R_{kj}(\boldsymbol{y}_k, p_k, \boldsymbol{\theta}_j) \geq \tau_k, \ \forall k, j \tag{5.25b}$$

$$L^{\text{u}}_{kj}(\boldsymbol{y}_k, p_k, \boldsymbol{\theta}_j) + L^{\text{c}}_{kj}(f_{kj}) \leq \bar{L}_k, \ \forall k, j \in \hat{\mathcal{M}}_k \tag{5.25c}$$

$$\sum_{k\in\mathcal{K}} y_{kb} \leq 1, \ b \in \mathcal{B} \tag{5.25d}$$

$$\sum_{k\in\mathcal{K}} f_{kj} \leq 1, \ \forall j \in \mathcal{M} \tag{5.25e}$$

$$y_{kb} \in \{0,1\}$$

$$p_k \in [0, \bar{P}_k], f_{kj} \geq 0, \theta_{jn} \in [0, 2\pi) \tag{5.25f}$$

where $\hat{\mathcal{M}}_k$ represents the subset of APs that UE k is assigned to, e.g. where $\hat{x}_{kj} = 1$. Since problem $\mathcal{P}_b$ is still non-convex similar to $\mathcal{P}_1$, we approach a sub-optimal solution for $\mathcal{P}_b$ by decoupling the optimization of the IRS phase shift from that of the RBs' allocation and the UEs' transmission power.

IRS Phase Shift Optimization: For the given values of the computation offloading matrix $\hat{\boldsymbol{x}}$, the RBs' allocation $\hat{\boldsymbol{y}}$, the UEs' transmission power vector $\hat{\boldsymbol{p}}$, and the UEs' computational resources' allocation matrix $\hat{\boldsymbol{f}}$, the IRS phase shift $\boldsymbol{\theta}$

can be optimized as:

$$\mathcal{P}_{c1}: \underset{\boldsymbol{\theta},\boldsymbol{\tau},\boldsymbol{u}}{max} \quad \sum_{k\in\mathcal{K}} \tau_k \tag{5.26a}$$

$$\text{s.t.} \quad \sum_{b\in\hat{\mathcal{B}}_k} u_{kjb} \geq \tau_k, \ \forall (k,j) \in (\mathcal{K}, \hat{\mathcal{M}}_k) \tag{5.26b}$$

$$R_{kjb}(\hat{p}_k, \boldsymbol{\theta}_j) \geq u_{kjb}, \ \forall (k,j,b) \in (\mathcal{K}, \hat{\mathcal{M}}_k, \hat{\mathcal{B}}_k) \tag{5.26c}$$

$$\theta_{jn} \in [0, 2\pi) \tag{5.26d}$$

where $\hat{\mathcal{B}}_k$ represents the subset of RBs that UE k is assigned to (where $\hat{y}_{kb} = 1$), and $u_{kjb} \geq 0$ is a newly introduced slack variable to facilitate the transformation of the problem. The objective (5.25a) of problem $\mathcal{P}_{c1}$ is to maximize the sum of the minimum transmission rates from the UEs to the associated APs, which is represented by $\tau_k, \forall k \in \mathcal{K}$ in (5.26c). We remark that the obtained problem would be a generalized convex problem due to the appearance of the generalized exponential cone constraint (5.26c), which still makes the problem challenging due to the very high computation time resulting from solving the problem using a generalized convex solver such as FMINCON. In contrast, a standard convex program such as the SOCP can be solved much more rapidly by the commercial solver MOSEK while achieving an accuracy of 99.99% [61]. Thus, we are motivated to employ the conic approximation with controlled accuracy in [61], where constraint (5.26c) can be rewritten by a set of

second order cone inequalities as:

$$\kappa_{q+4}^{kjb} \leq 1 + \left(\hat{p}_k |h_{kjb} + \boldsymbol{h}_{kjb}^{\mathrm{I}} \boldsymbol{\Theta}_j \boldsymbol{G}_{jb}|^2 / W N_0\right) \tag{5.27}$$

$$\begin{aligned}
a + \kappa_1^{kjb} &\geq \left\| \begin{bmatrix} a - \kappa_1^{kjb} & 2 + u_{kjb}/W 2^{m-1} \end{bmatrix} \right\|_2 \\
a + \kappa_2^{kjb} &\geq \left\| \begin{bmatrix} a - \kappa_2^{kjb} & 5/3 + u_{kjb}/W 2^{m} \end{bmatrix} \right\|_2 \\
a + \kappa_3^{kjb} &\geq \left\| \begin{bmatrix} a - \kappa_3^{kjb} & 2\kappa_1^{kjb} \end{bmatrix} \right\|_2 \\
\kappa_4^{kjb} &\geq \kappa_2^{kjb} + \kappa_3^{kjb}/24 + 19/72a \\
a + \kappa_l^{kjb} &\geq \left\| \begin{bmatrix} a - \kappa_l^{kjb} & 2\kappa_{l-1}^{kjb} \end{bmatrix} \right\|_2 \quad \forall\ l \in \{5, ..., q+3\} \\
a + \kappa_{q+4}^{kjb} &\geq \left\| \begin{bmatrix} a - \kappa_{q+4}^{kjb} & 2\kappa_{q+3}^{kjb} \end{bmatrix} \right\|_2
\end{aligned} \tag{5.28}$$

where $\boldsymbol{\kappa}_l = \{\kappa_l^{kjb} \geq 0, k \in \mathcal{K}, j \in \hat{\mathcal{M}}_k, b \in \hat{\mathcal{B}}_k\}$, a is a constant with $a = 1$, and q is the conic approximation parameter which can be chosen as $q = 4$ to attain a high accuracy. By replacing the term $h_{kjb} + \boldsymbol{h}_{kjb}^{\mathrm{I}} \boldsymbol{\Theta}_j \boldsymbol{G}_{jb}$ in (5.27) by $h_{kjb} + \boldsymbol{v}_j^H \boldsymbol{\Phi}_{kjb}$ with $\boldsymbol{\Phi}_{kjb} = \mathrm{diag}(\boldsymbol{h}_{kjb}^{\mathrm{I}}) \boldsymbol{G}_{jb}$, problem $\mathcal{P}_{c1}$ can be transformed to:

$$\mathcal{P}_{c2}: \max_{\boldsymbol{\theta}, \boldsymbol{\tau}, \boldsymbol{u}} \quad \sum_{k \in \mathcal{K}} \tau_k \tag{5.29a}$$

$$\begin{aligned}
\text{s.t.} \quad & \kappa_{q+4}^{kjb} \leq 1 + \hat{p}_k / W N_0 (\boldsymbol{v}_j^H \boldsymbol{\Phi}_{kjb} \boldsymbol{\Phi}_{kjb}^H \boldsymbol{v}_j \\
& \quad + \boldsymbol{v}_j^H \boldsymbol{\Phi}_{kjb} h_{kjb} + h_{kjb}^H \boldsymbol{\Phi}_{kjb}^H \boldsymbol{v}_j + |h_{kjb}|^2)
\end{aligned} \tag{5.29b}$$

$$|v_{jn}|^2 = 1, (5.26\text{b}), (5.28) \tag{5.29c}$$

By introducing slack variable $\boldsymbol{t} = \{t_j, \forall j \in \mathcal{M}\}$, problem $\mathcal{P}_{c2}$ can be converted

to a homogeneous QCQP as:

$$\mathcal{P}_{c3}: \underset{\boldsymbol{\theta},\boldsymbol{\tau},\boldsymbol{u}}{max} \quad \sum_{k\in\mathcal{K}} \tau_k \tag{5.30a}$$

$$\text{s.t.} \quad \kappa_{q+4}^{kjb} \leq 1 + \hat{p}_k / W N_0 \left(\bar{\boldsymbol{v}}_j^H \boldsymbol{R}_{kjb} \bar{\boldsymbol{v}}_j + |h_{kjb}|^2 \right) \tag{5.30b}$$

$$|v_n|^2 = 1, (5.26\text{b}), (5.28) \tag{5.30c}$$

$$\boldsymbol{R}_{kjb} = \begin{bmatrix} \boldsymbol{\Phi}_{kjb}\boldsymbol{\Phi}_{kjb}^H & h_{kjb}\boldsymbol{\Phi}_{kjb} \\ h_{kjb}\boldsymbol{\Phi}_{kjb}^H & 0 \end{bmatrix} \forall j, \ \bar{\boldsymbol{v}}_j = \begin{bmatrix} \boldsymbol{v}_j \\ t_j \end{bmatrix}$$

We note that $\bar{\boldsymbol{v}}_j^H \boldsymbol{R}_{kjb} \bar{\boldsymbol{v}}_j = \text{tr}(\boldsymbol{R}_{kjb} \boldsymbol{V}_j)$ with $\boldsymbol{V}_j = \bar{\boldsymbol{v}}_j \bar{\boldsymbol{v}}_j^T$. Problem $\mathcal{P}_{c3}$ can be transformed to:

$$\mathcal{P}_{c4}: \underset{\boldsymbol{V},\boldsymbol{\tau},\boldsymbol{u}}{max} \quad \sum_{k\in\mathcal{K}} \tau_k \tag{5.31a}$$

$$\text{s.t.} \quad \kappa_{q+4}^{kjb} \leq 1 + \hat{p}_k / W N_0 \left(\bar{\boldsymbol{v}}_j^H \boldsymbol{R}_{kjb} \bar{\boldsymbol{v}}_j + |h_{kjb}|^2 \right) \tag{5.31b}$$

$$V_{j,n,n} = 1, \ \forall n \in \{1, 2, ..., N+1\} \tag{5.31c}$$

$$\boldsymbol{V}_j \succcurlyeq 0 \tag{5.31d}$$

$$\text{rank}(\boldsymbol{V}_j) = 1, \ (5.26\text{b}), (5.28) \tag{5.31e}$$

which is a non-convex SDP where (5.31d) and (5.31e) indicate that $\boldsymbol{V}_j$ is a semi-definite matrix with a non-convex rank-one constraint which makes solving the problem very challenging. One widely adopted technique is the SDR [127], where the rank-one constraint is relaxed making the problem a convex SDP which can be easily solved, and then the Gaussian randomization is used to construct a rank-one

solution from $\boldsymbol{V}_j$ [127]. However, this technique does not guarantee obtaining a feasible or a rank-one solution to (5.31) especially when the problem size grows. Thus, we propose a novel SCA approach for solving (5.31) which is guaranteed to converge where the rank-one constraint is written in a DC form.

With $\sigma_r(\boldsymbol{V}_j)$ defined as the r-th largest singular value of $\boldsymbol{V}_j$, the rank-one constraint for $\boldsymbol{V}_j$ indicates that $\sigma_1(\boldsymbol{V}_j) > 0$ and $\sigma_r(\boldsymbol{V}_j) = 0, \forall i \in \{2, 3, ..., N+1\}$. Thus, by defining the trace norm and spectral norm of $\boldsymbol{V}_j$ as $\text{Tr}(\boldsymbol{V}_j) = \sum_{n=1}^{N+1} \sigma_r(\boldsymbol{V}_j)$ and $||\boldsymbol{V}_j|| = \sigma_1(\boldsymbol{V}_j)$, respectively, the rank-one constraint can be equivalently rewritten as the difference of these two convex norms as $\sum_{j\in\mathcal{M}} (\text{Tr}(\boldsymbol{V}_j) - ||\boldsymbol{V}_j||) = 0$ with $\text{Tr}(\boldsymbol{V}_j) > 0$. Hence, problem $\mathcal{P}_{c4}$ can be converted into the following DC program:

$$\mathcal{P}_{c5}: \min_{\boldsymbol{V},\boldsymbol{\tau},\boldsymbol{u}} \quad -\sum_{k\in\mathcal{K}} \tau_k + \sum_{j\in\mathcal{M}} (\text{Tr}(\boldsymbol{V}_j) - ||\boldsymbol{V}_j||) \tag{5.32a}$$

$$\text{s.t.} \quad \text{(5.26b), (5.28), (5.31b), (5.31c)} \tag{5.32b}$$

which is still non-convex due to the concave term $-||\boldsymbol{V}_j||$ in (5.32a). Thus, we proceed to linearize $f(\boldsymbol{V}_j) = ||\boldsymbol{V}_j||$ by substituting it with the term $\tilde{f}(\boldsymbol{V}_j; \boldsymbol{V}_j^{(m)}) = \left\langle \psi||\boldsymbol{V}_j^{(m)}||, \boldsymbol{V}_j \right\rangle$, where $\boldsymbol{V}_j^{(m)}$ is the solution obtained at the previous SCA iteration n, and $\psi||\boldsymbol{V}_j^{(m)}||$ denotes the sub-gradient of spectral norm at point $\boldsymbol{V}_j^{(m)}$. We note that one sub-gradient of $||\boldsymbol{V}_j||$ can be efficiently computed as $\boldsymbol{q}_1\boldsymbol{q}_1^H$, where $\boldsymbol{q}_1$ is the vector corresponding to the largest singular value $\sigma_1(\boldsymbol{V}_j)$. Thus, $\mathcal{P}_{c5}$ can be replaced

by:

$$\mathcal{P}_{c6}^{(m)}: \underset{\boldsymbol{V},\boldsymbol{\tau},\boldsymbol{u}}{min} \quad \Gamma_c = -\sum_{k\in\mathcal{K}} \tau_k + \sum_{j\in\mathcal{M}} \Big(\mathrm{Tr}(\boldsymbol{V}_j) - \tilde{f}(\boldsymbol{V}_j; \boldsymbol{V}_j^{(m)})\Big) \tag{5.33a}$$

$$\text{s.t.} \quad \text{(5.26b), (5.28), (5.31b), (5.31c)} \tag{5.33b}$$

which is a standard convex SDP that is solved at each SCA iteration n until convergence given an initial $\boldsymbol{V}_j^{(0)}$. Specifically, we design a practical stopping criterion with $\sum_{j\in\mathcal{M}} \Big(\mathrm{Tr}(\boldsymbol{V}_j) - \tilde{f}(\boldsymbol{V}_j; \boldsymbol{V}_j^{(m)})\Big) < \epsilon$, where ϵ is a sufficiently small positive constant. After obtaining a feasible $\boldsymbol{V}_j, \forall j \in \mathcal{M}$, the phase shift matrix $\boldsymbol{\theta}_j$ can be then easily recovered from $\boldsymbol{V}_j, \forall j \in \mathcal{M}$. The algorithm pseudocode for solving the IRS phase shift optimization sub-problem at each iteration of the alternating optimization, is outlined in Algorithm 9.

Algorithm 9 IRS Phase Shift Optimization

1: **Initialize:**
2: $\hat{\boldsymbol{x}}$, $\hat{\boldsymbol{y}}$, $\hat{\boldsymbol{p}}$, $\hat{\boldsymbol{f}}$, and $n = 0$;
3: Choose an initial point $\boldsymbol{V}^{(m)}$;
4: **repeat**
5: Solve $\mathcal{P}_{c6}^{(m)}$ to obtain the optimal solution $\boldsymbol{\omega}_c^{(m)} = \{\boldsymbol{V}^\star, \boldsymbol{\tau}^\star, \boldsymbol{u}^\star\}$ and objective $\Gamma_c^{(m)}$ at the nth iteration.
6: Update $\boldsymbol{V}^{(m)} = \boldsymbol{V}^\star$;
7: $n = n + 1$;
8: **until** Convergence of the objective of $\mathcal{P}_{c6}^{(m)}$.
9: Recover the phase shift matrix $\boldsymbol{\theta}^\star$ from $\boldsymbol{V}^\star$.
10: Return the solution $\boldsymbol{\theta}^\star$ and the objective $\Gamma_c^\star = \Gamma_c^{(m)}$.

Convergence Analysis: The convergence of Algorithm 9 can be guaranteed by showing that the series of resulting objective is monotonically convergent. Due to the convex approximation for $\tilde{f}(\boldsymbol{V}_j; \boldsymbol{V}_j^{(m)})$, the updating rules in Algorithm 9, c.f.,

Step 6, ensure that the solution set $\boldsymbol{\omega}_c^{(m)}$ is a feasible solution to problem $\mathcal{P}_{c5}$ at step $n+1$. This subsequently leads to the results of $\Gamma_c^{(n+1)} \leq \Gamma_c^{(m)}$, which means that Algorithm 9 generates a non-increasing sequence of objective function values. Due to constraint (5.31b), the sequence of $\Gamma_c^{(m)}, n = 1, 2, \ldots$ is bounded below and therefore, Algorithm 9 guarantees that the objective converges.

Resources Allocation Optimization: For the given values of the phase shift $\hat{\boldsymbol{\theta}}$, the RBs' allocation $\boldsymbol{y}$, the UEs' transmission power $\boldsymbol{p}$ and the UEs' computational resources' allocation $\boldsymbol{f}$, can be optimized as:

$$\mathcal{P}_{d1}: \min_{\substack{\boldsymbol{y},\boldsymbol{p},\\ \boldsymbol{f},\boldsymbol{u}}} \quad \sum_{k\in\mathcal{K}} p_k(d_k/\tau_k) \tag{5.34a}$$

$$\text{s.t.} \quad \sum_{b\in\mathcal{B}} u_{kjb} \geq \tau_k, \ \forall (k,j) \in (\mathcal{K}, \hat{\mathcal{M}}_k) \tag{5.34b}$$

$$y_{kb} R_{kjb}(p_k, \hat{\boldsymbol{\theta}}_j) \geq u_{kjb}, \ \forall (k,j,b) \in (\mathcal{K}, \hat{\mathcal{M}}_k, \mathcal{B}) \tag{5.34c}$$

$$L_{kj}^{\mathrm{u}}(\boldsymbol{y}_k, p_k, \hat{\boldsymbol{\theta}}_j) + L_{kj}^{\mathrm{c}}(f_{kj}) \leq \bar{L}_k, \ \forall j \in \hat{\mathcal{M}}_k \tag{5.34d}$$

$$(5.25\text{d}), \ (5.25\text{e}) \tag{5.34e}$$

$$y_{kb} \in \{0,1\}, p_k \in [0, \bar{P}_k], f_{kj} \geq 0 \tag{5.34f}$$

Problem $\mathcal{P}_c$ is a mixed-integer non-convex program due to the binary condition of variable $\boldsymbol{y}$ in (5.34f), and to (5.34a) being neither convex nor concave in general. We note that the generalized exponential cone constraint (5.34c) can be rewritten into a set of second order cone inequalities similar to (5.26c) with $a = y_{kb}$, and the generalized convex constraint (5.34d) can be easily transformed into a second-order cone constraint. In order to convexify problem $\mathcal{P}_c$, we introduce slack variable

$\delta_k, \forall k \in \mathcal{K}$ which will equivalently replace (5.34a), and slack variable $\Lambda_k, \forall k \in \mathcal{K}$, and equivalently define the following constraints:

$$\delta_k \tau_k \geq \Lambda_k^2 \tag{5.35a}$$

$$p_k - \Lambda_k^2 \leq 0 \tag{5.35b}$$

Here, (5.35a) is a quadratic conic convex constraint, while (5.35b) is non-convex due to the concave function $g(\Lambda_k) = -\Lambda_k^2$ which renders the left side of (5.35b) as a DC form. Thus, with $\Lambda^{(m)}$ as the input point, we employ the SCA method to replace $g(\Lambda)$ by its first order Taylor approximate as:

$$\tilde{g}(\Lambda_k; \Lambda_k^{(m)}) = -(\Lambda_k^{(m)})^2 - 2\Lambda_k^{(m)}(\Lambda_k - \Lambda_k^{(m)}) \tag{5.36}$$

The mixed-integer SOCP approximation of problem $\mathcal{P}_{d1}$ will still pause scalability limitations, preventing the SCA algorithm from being applied to big instances due to the mixed-integer nature of the problem, which is caused by the binary condition of variable $\boldsymbol{y}$ in (5.38g). To solve that problem, we adopt a similar approach to [83], and relax the binary condition for variable $\boldsymbol{y}$ by introducing the following constraint:

$$0 \leq y_{kb} - \tilde{g}(y_{kb}; y_{kb}^{(m)}) \leq \zeta_{kb} \tag{5.37}$$

which is the approximated form of the original non-convex constraint, where $\boldsymbol{\zeta} = \{\zeta_{kb} \geq 0, \forall k \in \mathcal{K}, b \in \mathcal{B}\}$ is a newly introduced slack variable. Constraint (5.37) will force variable $\boldsymbol{y}$ to take a binary value with a penalty term added to the objective. At

this point, by employing the above approximations, an approximated SOCP problem for problem $\mathcal{P}_{d1}$ is formulated at the m^{th} iteration as:

$$\mathcal{P}_{d2}^{(m)}: \underset{\substack{\boldsymbol{y},\boldsymbol{p},\boldsymbol{f},\\ \boldsymbol{u},\boldsymbol{\Lambda},\boldsymbol{\zeta}}}{min} \quad \Gamma_d = \sum_{k\in\mathcal{K}} d_k\delta_k + A\sum_{k\in\mathcal{K}}\sum_{b\in\mathcal{B}}\zeta_{kb} \tag{5.38a}$$

$$\text{s.t.} \quad \sum_{b\in\mathcal{B}} u_{kjb} \geq \tau_k,\ \forall (k,j) \in (\mathcal{K}, \hat{\mathcal{M}}_k) \tag{5.38b}$$

$$\kappa_{q+4}^{kjb} \leq y(k,b) + \left(p_k|h_{kjb} + \boldsymbol{h}_{kjb}^{\mathrm{I}}\hat{\boldsymbol{\Theta}}_j\boldsymbol{G}_{jb}|^2/WN_0\right) \tag{5.38c}$$

$$L_{kj}^{\mathrm{u}}(\boldsymbol{y}_k, p_k, \hat{\boldsymbol{\theta}}_j) + L_{kj}^{\mathrm{c}}(f_{kj}) \leq \bar{L}_k,\ \forall j \in \hat{\mathcal{M}}_k \tag{5.38d}$$

$$p_k + \tilde{g}(\Lambda_k; \Lambda_k^{(m)}) \leq 0 \tag{5.38e}$$

$$\text{(5.25d), (5.25e), (5.28), (5.35a), (5.37)} \tag{5.38f}$$

$$y_{kb} \in [0,1], p_k \in [0, \bar{P}_k], f_{kj} \geq 0 \tag{5.38g}$$

where $A > 0$ is the penalty parameter for reinforcing the binary condition on variable $\boldsymbol{y}$. Here, $\boldsymbol{\Lambda}^{(m)}$ and $\boldsymbol{y}^{(m)}$ are being updated to the optimal $\boldsymbol{\Lambda}$ and $\boldsymbol{y}$, respectively, at each iteration of the SCA-based algorithm. The algorithm pseudocode for solving the resources' allocation optimization sub-problem at each iteration of the alternating optimization, is outlined in Algorithm 10.

Convergence Analysis: The convergence of Algorithm 10 can be guaranteed by showing that the series of resulting objective is monotonically convergent. Due to the convex approximation in (5.36), the updating rules in Algorithm 10, c.f., Step 6, ensure that the solution set $\boldsymbol{\omega}_d^{(m)}$ is a feasible solution to problem $\mathcal{P}_{d1}$ at step $n+1$. This subsequently leads to the results of $\Gamma_d^{(m+1)} \leq \Gamma_d^{(m)}$, which means that Algorithm 10 generates a non-increasing sequence of objective function values. Due

Algorithm 10 Resources Allocation Optimization

Initialize:
$\hat{\boldsymbol{x}}$, $\hat{\boldsymbol{\theta}}$, and $m = 0$;
Choose an initial point $\boldsymbol{\Lambda}^{(m)}$;
repeat
    Solve $\mathcal{P}_{d2}^{(m)}$ to obtain the optimal solution $\boldsymbol{\omega}_d^{(m)} = \{\boldsymbol{y}^\star, \boldsymbol{p}^\star, \boldsymbol{f}^\star, \boldsymbol{u}^\star, \boldsymbol{\Lambda}^\star\}$ and objective $\Gamma_d^{(m)}$ at the mth iteration.
    Update $\boldsymbol{\Lambda}^{(m)} = \boldsymbol{\Lambda}^\star$;
    $m = m + 1$;
until Convergence of the objective of $\mathcal{P}_{d2}^{(m)}$.
Return the solution $\{\boldsymbol{y}^\star, \boldsymbol{p}^\star, \boldsymbol{f}^\star\}$ and the objective $\Gamma_d^\star = \Gamma_d^{(m)}$.

to the latency constraints in (5.38d), the sequence of $\Gamma_d^{(m)}, m = 1, 2, \ldots$ is bounded below and therefore, Algorithm 10 guarantees that the objective converges.

The overall algorithm pseudocode for providing an efficient sub-optimal solution for problem $\mathcal{P}_1$, is outlined in Algorithm 11.

Algorithm 11 Sub-optimal Solution for Solving $\mathcal{P}_1$

Obtain $\boldsymbol{x}^\star$ through Algorithm 8.
Start with initial values: $\boldsymbol{y}^\star = 1$, $\boldsymbol{p}^\star = \bar{\boldsymbol{P}}$, and $\boldsymbol{f}^\star = 1$;
Set $i = 1$
repeat
    Obtain $\boldsymbol{\theta}^{(i)}$ and the objective $\Gamma_c^{(i)}$ through Algorithm 9 given the inputs $\hat{\boldsymbol{y}} = \boldsymbol{y}^{(i)}$, $\hat{\boldsymbol{p}} = \boldsymbol{p}^{(i)}$, and $\hat{\boldsymbol{f}} = \boldsymbol{f}^{(i)}$.
    Obtain $\{\boldsymbol{y}^{(i)}, \boldsymbol{p}^{(i)}, \boldsymbol{f}^{(i)}\}$ and the objective $\Gamma_d^{(i)}$ through Algorithm 10 given the input $\hat{\boldsymbol{\theta}} = \boldsymbol{\theta}^{(i)}$.
    Set $i = i + 1$;
until convergence of objective $\Gamma_d^{(i)}$.
Return the optimized solution $\boldsymbol{\omega}^\star = \{\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)}, \boldsymbol{p}^{(i)}, \boldsymbol{f}^{(i)}, \boldsymbol{\theta}^{(i)}\}$ and objective $\Gamma_d^{(i)}$.

Convergence Analysis: The convergence of Algorithm 11 can be guaranteed by showing that the series of resulting objective is monotonically convergent. Due to the non-increasing sequence of objective function values in Algorithm 9 and Algorithm

10, the objective values $\Gamma_c^{(m)}$ and $\Gamma_d^{(m)}$ at step $i+1$ are guaranteed to be less than or equal to $\Gamma_c^{(m)}$ and $\Gamma_d^{(m)}$ at step m, which means that Algorithm 11 generates a non-increasing sequence of objective function values $\Gamma_d^{(m)}$, which guarantees the objective convergence.

Complexity Analysis: The overall complexity of Algorithm 11 depends mainly on that of solving the SDP problem (5.33) in Algorithm 9, and the SOCP problem (5.38) in Algorithm 10. Since the complexity of (5.38) is approximately $\mathcal{O}(qKMB)$ which is a polynomial time complexity with q as the conic approximation parameter [61], the complexity of solving (5.33) is the dominant one. The order of complexity for an SDP problem with m SDP constraints which includes an $n \times n$ PSD matrix, is given by $\mathcal{O}(\sqrt{n}\log(1/\epsilon)(mn^3 + m^2n^2 + m^3))$ where $\epsilon > 0$ is the solution accuracy [128, Th. 3.12]. In our case, we have $n = N + 1$ and $m = KM + qKMB + N + 1$; but since $qKMB$ is the dominant term, the approximate computation complexity for solving (5.33) (and Algorithm 11), can be defined as $\mathcal{O}(\log(1/\epsilon)(qKMBN^3 + (qKMB)^2N^2 + (qKMB)^3))$.

5.4.3 Numerical Results

In this section, we study the design performance of the proposed solution through simulations. The main instance consists of UEs that are positioned in the center of a 2-D area of 400 meters^2. Instance A consists of $K = 20$ UEs that are positioned within the range of $M = 5$ randomly distributed APs each having a nearby IRS with $N = 20$ elements, and instance B consists of $K = 10$ UEs that are positioned within the range of $M = 3$ randomly distributed APs each having a nearby IRS with

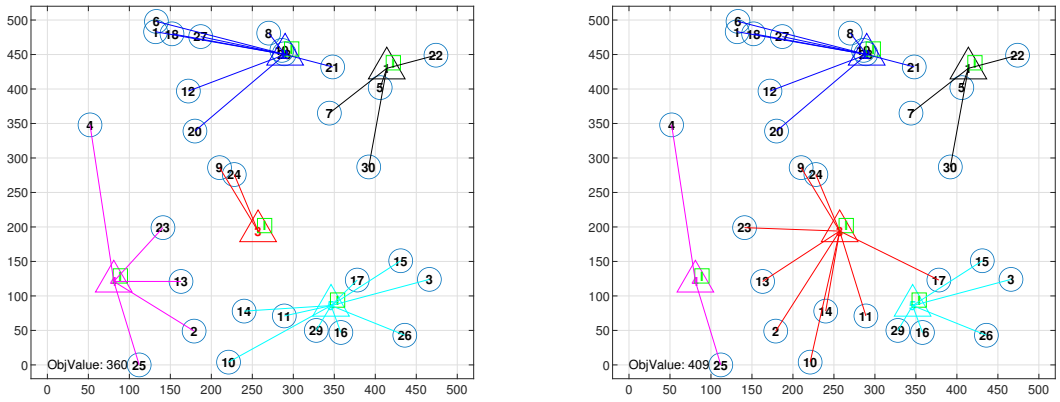
Table 5.4: Instance Parameters

Parameter	**Value**
Task input size D_k	50 kilobits
Task computational demand C_k	200 cycles/bit
UEs' transmission power threshold $\bar{P}_k$	$P = 30$ dBm [124]
Cloudlet capacity F_j	10 GHz
Cloudlet reliability ϕ_j	0.9955
Rician Factor	2
Path loss (PL) at 1 meter g_0	-30 dB[123]
PL exponents for the UE-AP link (IRS link)	4 (2.2)
RB Radio spectrum bandwidth W	1 MHz [124]
Noise power spectral density N_0	-174 dBm/Hz [124]

$N = 30$ elements. The system parameters are presented in Table 5.4.

In Fig. 5.7, we show the results obtained from Algorithm 8 for instance A with UEs' reliability requirement $\bar{R}_k = 0.9955$, where each UE needs to offload its task to one AP that achieves its minimum offloading latency. In Fig. 5.7.a, the assignment results for the UEs is shown where the servers have identical resources' capacity. However, in Fig. 5.7.b, the capacity of servers 4 and 5 has been reduced by half, which reflects the deviation of UEs 1 and 15 to AP 2. This change is due to the fact that with lower capacity, servers 4 and 5 become quickly congested achieving a higher computation latency, which prompts some UEs to deviate to other servers that are able to achieve a lower offloading latency.

In Fig. 5.8, we use instance A for comparing the sum offloading latencies obtained



5.7.a Servers with identical resources capacity ($F_j = 10$ GHz).

5.7.b Servers 4 and 5 have resources capacity $F_4 = F_5 = 5$ GHz.

Figure 5.7: Matching game UEs' offloading decisions results.

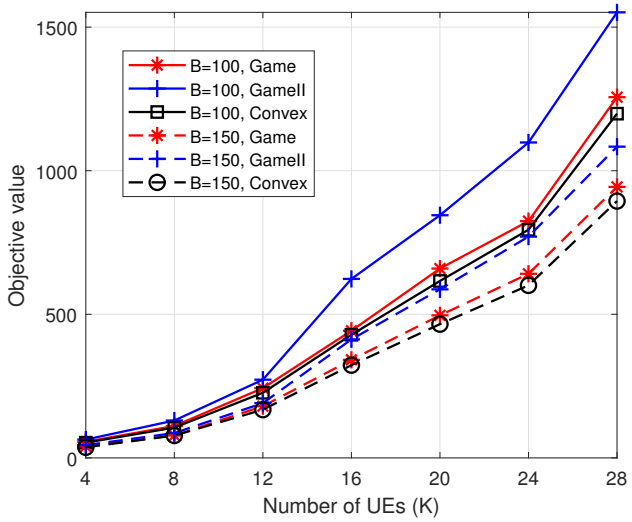


Figure 5.8: Matching game compared to other approaches.

from the matching game algorithm with that of Game2, where the cost of each UE is computed only using the channel gain. Also, we compare our algorithm to the

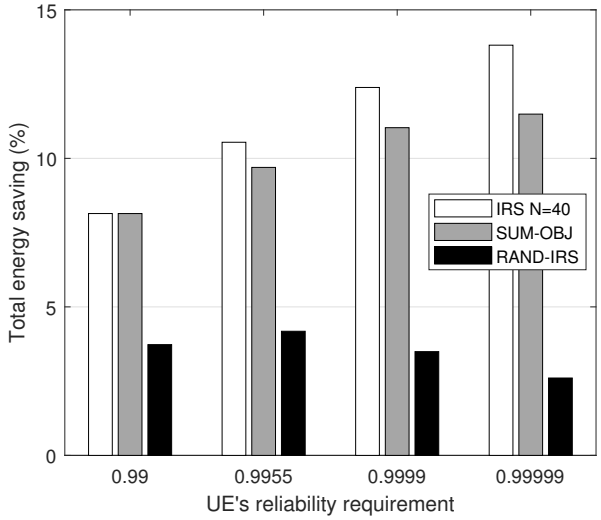


Figure 5.9: Energy gain incurred from the use of the IRSs.

mixed-integer convex problem $\mathcal{P}_a$. As it can be seen, the matching game algorithm achieves a relatively small gap when compared to the optimal solution, where the gap grows a little bigger when more UEs are added to the network. However, it is worth noting that the matching game algorithm has a much lower complexity and is able to execute large instances very quickly. Also, it can be seen that our matching game approach outperforms the other variation where the cost is calculated only using the UEs' channel gain.

In Fig. 5.9, we present the UEs' energy saving achieved through the IRS use that is obtained from our solution for instance A, and compare it to SUM-OBJ where the sum of transmission rates is maximized in (5.33), and RAND-IRS where the IRS phase shifts are randomly configured. As it can be seen, the proposed solution with $N = 40$ outperforms the other approaches, where the gain over SUM-OBJ is negligible when one communication channel is utilized, and becomes higher with the

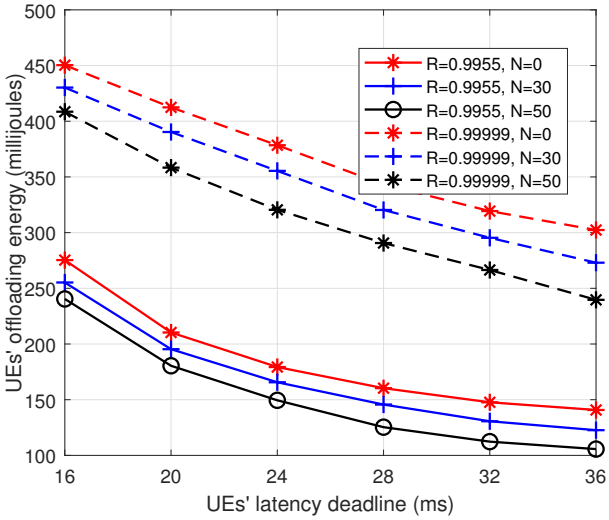


Figure 5.10: Objective value vs reliability requirement and number of IRS elements.

use of more APs. It is also observed that the energy saving becomes larger when the UEs' reliability requirement is higher. This is because when more APs are utilized, the IRSs can better enhance the lower quality channels, which reduces the UEs' transmission power and therefore their consumed energy.

In Fig. 5.10, we study how the IRS size influences the achieved total energy consumption for different classes of latency and reliability requirements, for instance A. It is observed that a higher energy consumption is incurred when the UEs have a more strict latency deadline, since the UEs in this case need to use a higher transmission power in order to satisfy their latency requirement. In addition, an increase in the energy consumption is caused by a higher reliability requirement, since the UEs need to offload their task on more cloudlets that have lower quality channels, which forces the UEs to allocate more transmission power to meet their latency bound. Moreover, it can be seen that when the operator uses a larger IRS, the increase in

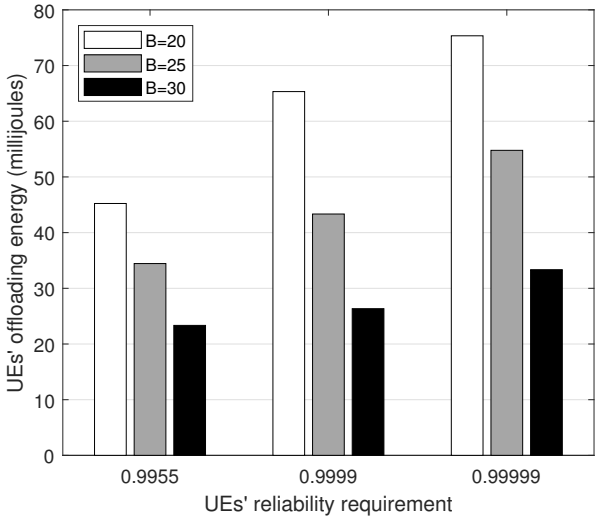


Figure 5.11: Objective value vs reliability requirement and number of resource blocks.

the UEs' transmission power and energy consumption can be significantly counteracted by selectively enhancing the channels' quality in response to more stringent service requirements.

In Fig. 5.11, we study using instance B how the UEs' energy consumption is affected by the available system bandwidth for the UEs for different service reliability levels. As can be seen, more energy is consumed when a more strict reliability is required, which is because the UEs' need to use a higher transmission power for reaching more APs. However, the availability of more RBs can help in maintaining the same levels of the needed transmission rate, which avoids the increase in energy consumption in response to a higher reliability requirement.

5.5 Conclusion

In this chapter, we studied an IRS-aided MEC system for enabling low-energy computation offloading for IoT services with stringent latency and reliability requirements, considering redundancy of tasks computation for minimizing the UEs' offloading energy consumption. First, we considered the single-user case where we proposed our non-convex problem for optimizing the IRS elements' phase shifts, the offloading decision, and the UE's transmission power. Then, we extended our study to the multi-user OFDMA network, where we presented our non-convex problem for optimizing the IRS elements' phase shifts, the offloading decisions, the UEs' transmission power, and the allocated servers' computational resources and OFDMA RBs. For each of the studied problems, we proposed our customized sub-optimal algorithm where the decisions are separately optimized in an alternating fashion using the SCA technique and the DC representation. Through numerical analysis, we demonstrated the energy reduction and saving in network resources that are possible through the optimized use of the IRSs especially for offloading services with higher reliability requirements, and we highlighted on the IRSs' influence on the design of the MEC parameters. Our work offers valuable insights into provisioning computation offloading for services with stringent latency and reliability requirements with the aid of deployed IRSs.

Chapter 6

Conclusion and Future Research Directions

This chapter concludes the presented thesis and highlights future research directions.

6.1 Conclusion

The concept of smart living in the age of IoT and smart city has recently emerged, and continues to gain significant interest towards modernizing all traditional operations and improving the quality of life through enabling innovative services leveraged by advanced information and communication technologies. A myriad of new business practices has been unleashed, paving the way for a surging number of UEs to carry out novel ubiquitous and heterogeneous use cases belonging to various verticals, such as smart healthcare, autonomous driving, and smart manufacturing, while imposing extreme unprecedented QoS such as stringent latency and reliability requirements.

Within the advancing 5G network technologies that aim to enable this smart and connected world, MEC has been proposed as a cutting edge solution for realizing computation offloading within the vicinity of end-user devices through the deployment of edge cloudlets, and therefore realizing the ever-needed low-latency computation for the low-energy end-user devices that is required to operate the modern 5G services. However, the realization of the latency and reliability aware computation offloading in the context of MEC, is coupled with various challenges, prompting the NOs to seek the assistance of novel technologies and architectures, in order to ensure the provisioning of the modern latency and reliability sensitive services. However, the integration of those technologies in the MEC system introduces new challenges and difficulties that must be addressed through novel techniques and solutions that are efficiently designed to fully leverage the available resources and infrastructure for the purpose of maximizing the efficiency of MEC computation offloading.

Throughout this thesis, we addressed several challenges linked to the realization of a latency and reliability aware edge computation offloading in 5G networks and beyond, with the aid of novel technologies and architectures while tackling their challenges. We first highlighted on the shortcomings of MEC that are preventing it from fully realizing its intended vision in light of various scenarios and network conditions. Then, we explored novel 5G technologies and techniques that can be integrated with the MEC system for assisting in overcoming its limitations and enhancing the computation offloading performance, namely, the multi-tier MEC architecture, UAV-mounted cloudlets, and the IRS-aided MEC system, which have incredible potentials to realize the ever-needed low-latency and high-reliability computation offloading in

light of various network conditions in the age of 5G and beyond. Then, we presented the challenges that we aim to tackle in order to fully leverage those technologies for aiding the MEC system in provisioning the computation offloading service, while minimizing the UEs' energy consumption and maximizing the network resources' utilization. Towards that end, we provided novel contributions related to the allocation of network and devices' resources as well as the optimization of other offloading parameters, and thereby efficiently utilizing the underlying infrastructure such as to enable energy and cost-efficient computation offloading schemes, by leveraging several customized solutions and optimization techniques.

In Chapter 2, we studied the problem of the joint optimization for the NO's computational cost and the UEs' energy consumption in the context of a multi-tier edge-cloud system with a deployed second-tier edge-cloud, where we optimized its use through proposed low-complexity algorithms, such as to achieve an energy and cost-efficient solution that guarantees the services' latency requirements. Due to the significant advantage of operating MEC in heterogeneous networks, we extended in Chapter 3 the energy-efficiency study to a network of small-cells with the second-tier edge server being co-located within the MC which can be reached through a wireless backhaul, where we optimized the macro-cell server use along with the other offloading parameters through a proposed customized algorithm based on the Successive Convex Approximation (SCA) technique.

Given the UAVs' considerable ability in expanding the capabilities of cellular networks and MEC systems, we explored in Chapter 4 the reliability-aware optimized use of UAV-mounted cloudlets for provisioning the MEC computation Offloading

service with stringent latency and reliability guarantees through the use of tasks redundancy. Towards that end, we studied the optimized positioning and use of UAV-mounted cloudlets for computation offloading through two planning and operational problems while considering tasks redundancy, and proposed customized solutions for solving those problems. Finally, given the IRSs' ability to also enhance the channel conditions through the tuning of their passive reflecting elements, we extended in Chapter 5 the latency and reliability study to an IRS-aided MEC system for enhancing the computation offloading performance in scenarios with unfavorable RAN channel conditions. We addressed both a single-user and multi-user OFDMA cases, where we explored the optimized IRSs' use in order to reveal their role in reducing the UEs' offloading energy consumption and saving the network resources, through proposed customized solutions based on the SCA approach and the SDR technique.

6.2 Future Research Directions

While we addressed several research challenges related to MEC in conjunction with the leveraged novel 5G network technologies, there still exists many future research directions that need to be tackled.

6.2.1 UAV-aided MEC Systems With The Aid of Second-tier High-altitude Platforms

After we leveraged UAV-mounted cloudlets to conduct the MEC operation for services with stringent latency and reliability requirements in Chapter 4, we envision to extend this aerial MEC system with HAPs that are co-located with cloudlets which can be utilized as second-tier computational units for supporting the LAP cloudlets with the offloading operations. In such case, the MEC system would greatly benefit from the air-to-air communication links between the cloudlet tiers for enabling low-latency migration of the UEs' tasks to be computed on the HAP cloudlets which would typically be of higher capacity. In this way, the MEC system would possess more capabilities that would allow for more UEs' request to be accommodated, where the optimized UEs' offloading decisions between the LAP and the HAP cloudlets, as well as the allocation of computation and wireless resources would be conducted in order to maximize the system efficiency.

6.2.2 Leveraging Machine Learning For Optimized IRS-aided MEC Networks with Load Variability

After we addressed the IRS-aided MEC computation offloading problem in Chapter 5, an interesting scenario that comes to light and is worth tackling is to consider the variability in the offloading requests across a time duration, where there is uncertainty in the arrival of UEs and their computation offloading pattern. Here, we observe that a DRL approach to solve the problem would form an effective solution for

accounting for the randomness in the offloading pattern, whereby the DRL agent learns to identify the request patterns and then take optimized decisions for the IRS elements' phase shifts and resources allocation, such as to minimize the total UEs' energy consumption in light of the existing network conditions.

6.2.3 Minimizing AoI For Computation Offloading in Edge-clouds

Seeing the concept of AoI that has recently emerged as a novel metric that measures the freshness of information for real-time applications which need to transmit status update packets to the destination node as timely as possible, there are modern services and applications which require performing intensive tasks' computations on the collected input before the real-time status information can be sent. In such case, seeing the limited capabilities of end-user devices, we observe the potential of a MEC system for minimizing the AoI by efficiently devising computation offloading and resources allocation strategies such as to accommodate the computation-intensive needs of the requesting services, and therefore guaranteeing the information freshness. In this case, it is also promising to explore a hierarchical MEC system and study its benefits in decreasing the AoI by leveraging the computation capacity of an upper tier cloudlet, and thereby accommodating a larger number of UEs' offloading requests for operating real-time applications.

Bibliography

[1] Ala Al-Fuqaha, Mohsen Guizani, Mehdi Mohammadi, Mohammed Aledhari, and Moussa Ayyash. Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE Communications Surveys & Tutorials*, 17(4):2347–2376, 2015.

[2] Hyoungju Ji, Sunho Park, Jeongho Yeo, Younsun Kim, Juho Lee, and Byonghyo Shim. Introduction to ultra reliable and low latency communications in 5g. *arXiv preprint arXiv:1704.05565*, 2017.

[3] Janne Peisa. 5g techniques for ultra reliable low latency communication. In *Proc. IEEE CSCN*, 2017.

[4] Hoang T Dinh, Chonho Lee, Dusit Niyato, and Ping Wang. A survey of mobile cloud computing: architecture, applications, and approaches. *Wireless communications and mobile computing*, 13(18):1587–1611, 2013.

[5] Shuo Wang, Xing Zhang, Yan Zhang, Lin Wang, Juwo Yang, and Wenbo Wang. A survey on mobile edge networks: Convergence of computing, caching and communications. *IEEE Access*, 5:6757–6779, 2017.

[6] Liang Tong, Yong Li, and Wei Gao. A hierarchical edge cloud architecture for mobile computing. In *IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*, pages 1–9. IEEE, 2016.

[7] Mohammad Mozaffari, Walid Saad, Mehdi Bennis, Young-Han Nam, and Mérouane Debbah. A tutorial on UAVs for wireless networks: Applications, challenges, and open problems. *IEEE communications surveys & tutorials*, 21(3):2334–2360, 2019.

[8] Qingqing Wu, Shuowen Zhang, Beixiong Zheng, Changsheng You, and Rui Zhang. Intelligent reflecting surface-aided wireless communications: A tutorial. *IEEE Transactions on Communications*, 69(5):3313–3351, 2021.

[9] Martin Curley. Keynote speakers: Open innovation 2.0 and digital technology; the new paradigm for prosperity and sustainable intelligent living. In *2016 3rd MEC International Conference on Big Data and Smart City (ICBDSC)*, pages xix–xxvii. IEEE, 2016.

[10] Tarik Taleb, Konstantinos Samdanis, Badr Mada, Hannu Flinck, Sunny Dutta, and Dario Sabella. On multi-access edge computing: A survey of the emerging 5G network edge cloud architecture and orchestration. *IEEE Communications Surveys & Tutorials*, 19(3):1657–1681, 2017.

[11] J Gold. What is the industrial IoT?[and why the stakes are so high]. *Network World*, 2018.

[12] NGMN Alliance. 5G white paper. *Next generation mobile networks, white paper*, 2015.

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

[14] Imtiaz Parvez, Ali Rahmati, Ismail Guvenc, Arif I Sarwat, and Huaiyu Dai. A survey on low latency towards 5g: Ran, core network and caching solutions. *IEEE Communications Surveys & Tutorials*, 20(4):3098–3130, 2018.

[15] Tiffany Yu-Han Chen, Lenin Ravindranath, Shuo Deng, Paramvir Bahl, and Hari Balakrishnan. Glimpse: Continuous, real-time object recognition on mobile devices. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*, pages 155–168. ACM, 2015.

[16] Mehdi Bennis, Mérouane Debbah, and H Vincent Poor. Ultrareliable and low-latency wireless communication: Tail, risk, and scale. *Proceedings of the IEEE*, 106(10):1834–1853, 2018.

[17] Emiliano Sisinni, Abusayeed Saifullah, Song Han, Ulf Jennehag, and Mikael Gidlund. Industrial internet of things: Challenges, opportunities, and directions. *IEEE Transactions on Industrial Informatics*, 14(11):4724–4734, 2018.

[18] Murtaza Ahmed Siddiqi, Heejung Yu, and Jingon Joung. 5G ultra-reliable low-latency communication implementation challenges and operational issues with IoT devices. *Electronics*, 8(9):981, 2019.

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

[20] Niroshinie Fernando, Seng W Loke, and Wenny Rahayu. Mobile cloud computing: A survey. *Future generation computer systems*, 29(1):84–106, 2013.

[21] Mahadev Satyanarayanan, Paramvir Bahl, Ramón Caceres, and Nigel Davies. The case for vm-based cloudlets in mobile computing. *IEEE pervasive Computing*, 8(4):14–23, 2009.

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

[23] Akhil Gupta and Rakesh Kumar Jha. A survey of 5g network: Architecture and emerging technologies. *IEEE access*, 3:1206–1232, 2015.

[24] Gopika Premsankar, Mario Di Francesco, and Tarik Taleb. Edge computing for the internet of things: A case study. *IEEE Internet of Things Journal*, 5(2):1275–1284, 2018.

[25] Milan Patel, Brian Naughton, Caroline Chan, Nurit Sprecher, Sadayuki Abeta, Adrian Neal, et al. Mobile-edge computing introductory technical white paper. *White paper, mobile-edge computing (MEC) industry initiative*, 29:854–864, 2014.

[26] Yun Chao Hu, Milan Patel, Dario Sabella, Nurit Sprecher, and Valerie Young. Mobile edge computing—a key technology towards 5G. *ETSI white paper*, 11(11):1–16, 2015.

[27] Abbas Kiani and Nirwan Ansari. Toward hierarchical mobile edge computing: An auction-based profit maximization approach. *IEEE Internet of Things Journal*, 4(6):2082–2091, 2017.

[28] Aleksandar Damnjanovic, Juan Montojo, Yongbin Wei, Tingfang Ji, Tao Luo, Madhavan Vajapeyam, Taesang Yoo, Osok Song, and Durga Malladi. A survey on 3GPP heterogeneous networks. *IEEE Wireless communications*, 18(3), 2011.

[29] Min Chen and Yixue Hao. Task offloading for mobile edge computing in software defined ultra-dense network. *IEEE Journal on Selected Areas in Communications*, 36(3):587–597, 2018.

[30] Zhenyu Zhou, Junhao Feng, Lu Tan, Yejun He, and Jie Gong. An air-ground integration approach for mobile edge computing in iot. *IEEE Communications Magazine*, 56(8):40–47, 2018.

[31] Lav Gupta, Raj Jain, and Gabor Vaszkun. Survey of important issues in uav communication networks. *IEEE Communications Surveys & Tutorials*, 18(2):1123–1152, 2015.

[32] Seng W Loke. The internet of flying-things: Opportunities and challenges

with airborne fog computing and mobile cloud in the clouds. *arXiv preprint arXiv:1507.04492*, 2015.

[33] Qingqing Wu and Rui Zhang. Towards smart and reconfigurable environment: Intelligent reflecting surface aided wireless network. *IEEE Communications Magazine*, 58(1):106–112, 2019.

[34] Qingqing Wu and Rui Zhang. Intelligent reflecting surface enhanced wireless network via joint active and passive beamforming. *IEEE Transactions on Wireless Communications*, 18(11):5394–5409, 2019.

[35] Elie El Haber, Tri Minh Nguyen, and Chadi Assi. Joint optimization of computational cost and devices energy for task offloading in multi-tier edge-clouds. *IEEE Transactions on Communications*, 67(5):3407–3421, 2019.

[36] NGMN Alliance. Description of network slicing concept. *NGMN 5G P*, 1, 2016.

[37] Jose Ordonez-Lucena, Pablo Ameigeiras, Diego Lopez, Juan J Ramos-Munoz, Javier Lorca, and Jesus Folgueira. Network slicing for 5G with SDN/NFV: Concepts, architectures, and challenges. *IEEE Communications Magazine*, 55(5):80–87, 2017.

[38] Konstantinos Samdanis, Xavier Costa-Perez, and Vincenzo Sciancalepore. From network sharing to multi-tenancy: The 5G network slice broker. *IEEE Communications Magazine*, 54(7):32–39, 2016.

[39] Pouria Sayyad Khodashenas, Cristina Ruiz, J Ferrer Riera, Jose Oscar Fajardo,

Ianire Taboada, Bego Blanco, Fidel Liberal, Javier Garcia Lloreda, Jordi Pérez-Romero, Oriol Sallent, et al. Service provisioning and pricing methods in a multi-tenant cloud enabled RAN. In *2016 IEEE Conference on Standards for Communications and Networking (CSCN)*, pages 1–6. IEEE, 2016.

[40] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[41] Barry R Marks and Gordon P Wright. A general inner approximation algorithm for nonconvex mathematical programs. *Operations research*, 26(4):681–683, 1978.

[42] Nhu-Ngoc Dao, Yunseong Lee, Sungrae Cho, Eungha Kim, Ki-Sook Chung, and Changsup Keum. Multi-tier multi-access edge computing: The role for the fourth industrial revolution. In *2017 International Conference on Information and Communication Technology Convergence (ICTC)*, pages 1280–1282. IEEE, 2017.

[43] Alberto Ceselli, Marco Premoli, and Stefano Secci. Mobile edge cloud network design optimization. *IEEE/ACM Transactions on Networking*, 25(3):1818–1831, 2017.

[44] Juan Liu, Yuyi Mao, Jun Zhang, and Khaled B Letaief. Delay-optimal computation task scheduling for mobile-edge computing systems. In *2016 IEEE international symposium on information theory (ISIT)*, pages 1451–1455. IEEE, 2016.

[45] Yuyi Mao, Jun Zhang, and Khaled B Letaief. Joint task offloading scheduling and transmit power allocation for mobile-edge computing systems. In *2017 IEEE wireless communications and networking conference (WCNC)*, pages 1–6. IEEE, 2017.

[46] Meng-Hsi Chen, Min Dong, and Ben Liang. Multi-user mobile cloud offloading game with computing access point. In *2016 5th IEEE International Conference on Cloud Networking (Cloudnet)*, pages 64–69. IEEE, 2016.

[47] Chen-Feng Liu, Mehdi Bennis, and H Vincent Poor. Latency and reliability-aware task offloading and resource allocation for mobile edge computing. In *2017 IEEE Globecom Workshops (GC Wkshps)*, pages 1–7. IEEE, 2017.

[48] Shermila Ranadheera, Setareh Maghsudi, and Ekram Hossain. Computation offloading and activation of mobile edge computing servers: A minority game. *IEEE Wireless Communications Letters*, 7(5):688–691, 2018.

[49] Thinh Quang Dinh, Jianhua Tang, Quang Duy La, and Tony QS Quek. Offloading in mobile edge computing: Task allocation and computational frequency scaling. *IEEE Transactions on Communications*, 65(8):3571–3584, 2017.

[50] Lin Wang, Lei Jiao, Dzmitry Kliazovich, and Pascal Bouvry. Reconciling task assignment and scheduling in mobile edge clouds. In *2016 IEEE 24th International Conference on Network Protocols (ICNP)*, pages 1–6. IEEE, 2016.

[51] Xiao Ma, Shan Zhang, Wenzhuo Li, Puheng Zhang, Chuang Lin, and Xuemin

Shen. Cost-efficient workload scheduling in cloud assisted mobile edge computing. In *2017 IEEE/ACM 25th International Symposium on Quality of Service (IWQoS)*, pages 1–10. IEEE, 2017.

[52] Wenjie Wang and Wei Zhou. Computational offloading with delay and capacity constraints in mobile edge. In *2017 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, 2017.

[53] Meng-Hsi Chen, Ben Liang, and Min Dong. Joint offloading decision and resource allocation for multi-user multi-task mobile cloud. In *2016 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, 2016.

[54] Meng-Hsi Chen, Ben Liang, and Min Dong. Joint offloading and resource allocation for computation and communication in mobile cloud with computing access point. In *IEEE INFOCOM 2017-IEEE Conference on Computer Communications*, pages 1–9. IEEE, 2017.

[55] Juyong Lee and Jihoon Lee. Hierarchical mobile edge computing architecture based on context awareness. *Applied Sciences*, 8(7):1160, 2018.

[56] Alessio Silvestro, Nitinder Mohan, Jussi Kangasharju, Fabian Schneider, and Xiaoming Fu. Mute: Multi-tier edge networks. In *Proceedings of the 5th Workshop on CrossCloud Infrastructures & Platforms*, pages 1–6. ACM, 2018.

[57] Chenmeng Wang, F Richard Yu, Chengchao Liang, Qianbin Chen, and Lun Tang. Joint computation offloading and interference management in wireless

cellular networks with mobile edge computing. *IEEE Transactions on Vehicular Technology*, 66(8):7432–7445, 2017.

[58] Chenmeng Wang, Chengchao Liang, F Richard Yu, Qianbin Chen, and Lun Tang. Computation offloading and resource allocation in wireless cellular networks with mobile edge computing. *IEEE Transactions on Wireless Communications*, 16(8):4924–4938, 2017.

[59] Ke Zhang, Yuming Mao, Supeng Leng, Quanxin Zhao, Longjiang Li, Xin Peng, Li Pan, Sabita Maharjan, and Yan Zhang. Energy-efficient offloading for mobile edge computing in 5g heterogeneous networks. *IEEE Access*, 4:5896–5907, 2016.

[60] Oskari Tervo, Le-Nam Tran, and Markku Juntti. Optimal energy-efficient transmit beamforming for multi-user MISO downlink. *IEEE Transactions on Signal Processing*, 63(20):5574–5588, 2015.

[61] Aharon Ben-Tal and Arkadi Nemirovski. On polyhedral approximations of the second-order cone. *Mathematics of Operations Research*, 26(2):193–205, 2001.

[62] Hoang Tuy, Faiz Al-Khayyal, and Phan Thien Thach. Monotonic optimization: Branch and cut methods. In *Essays and Surveys in Global Optimization*, pages 39–78. Springer, 2005.

[63] Hoang Tuy. Outer and inner approximation. In *Convex Analysis and Global Optimization*, pages 177–222. Springer, 1998.

[64] Yong Cheng, Marius Pesavento, and Anne Philipp. Joint network optimization and downlink beamforming for CoMP transmissions using mixed integer conic programming. *IEEE Transactions on Signal Processing*, 61(16):3972–3987, 2013.

[65] Farid Alizadeh and Donald Goldfarb. Second-order cone programming. *Mathematical programming*, 95(1):3–51, 2003.

[66] Elie El Haber, Tri Minh Nguyen, Chadi Assi, and Wessam Ajib. Macro-cell assisted task offloading in MEC-based heterogeneous networks with wireless backhaul. *IEEE Transactions on Network and Service Management*, 16(4):1754–1767, 2019.

[67] Mustafa Emara, Miltiades C Filippou, and Dario Sabella. MEC-aware cell association for 5g heterogeneous networks. In *2018 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, pages 350–355. IEEE, 2018.

[68] Chanwon Park and Jemin Lee. Mobile edge computing-enabled heterogeneous networks. *IEEE Transactions on Wireless Communications*, 20(2):1038–1051, 2020.

[69] Meng-Hsi Chen, Min Dong, and Ben Liang. Joint offloading decision and resource allocation for mobile cloud with computing access point. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3516–3520. IEEE, 2016.

[70] Ali Al-Shuwaili, Osvaldo Simeone, Alireza Bagheri, and Gesualdo Scutari. Joint uplink/downlink optimization for backhaul-limited mobile cloud computing with user scheduling. *IEEE Transactions on Signal and Information Processing over Networks*, 3(4):787–802, 2017.

[71] Jing Zhang, Weiwei Xia, Feng Yan, and Lianfeng Shen. Joint computation offloading and resource allocation optimization in heterogeneous networks with mobile edge computing. *IEEE Access*, 6:19324–19337, 2018.

[72] Laizhong Cui, Chong Xu, Shu Yang, Joshua Zhexue Huang, Jianqiang Li, Xizhao Wang, Zhong Ming, and Nan Lu. Joint optimization of energy consumption and latency in mobile edge computing for internet of things. *IEEE Internet of Things Journal*, 6(3):4791–4803, 2018.

[73] Jiao Zhang, Xiping Hu, Zhaolong Ning, Edith C-H Ngai, Li Zhou, Jibo Wei, Jun Cheng, and Bin Hu. Energy-latency tradeoff for energy-aware offloading in mobile edge computing networks. *IEEE Internet of Things Journal*, 5(4):2633–2645, 2017.

[74] Fengxian Guo, Heli Zhang, Hong Ji, Xi Li, and Victor CM Leung. An efficient computation offloading management scheme in the densely deployed small cell networks with mobile edge computing. *IEEE/ACM Transactions on Networking*, 26(6):2651–2664, 2018.

[75] Lixing Chen, Sheng Zhou, and Jie Xu. Computation peer offloading for energy-constrained mobile edge computing in small-cell networks. *IEEE/ACM Transactions on Networking*, 26(4):1619–1632, 2018.

[76] Quoc-Viet Pham, Tuan Leanh, Nguyen H Tran, Bang Ju Park, and Choong Seon Hong. Decentralized computation offloading and resource allocation for mobile-edge computing: A matching game approach. *IEEE Access*, 6:75868–75885, 2018.

[77] Jianhui Liu and Qi Zhang. Offloading schemes in mobile edge computing for ultra-reliable low latency communications. *IEEE Access*, 6:12825–12837, 2018.

[78] Quentin H Spencer, A Lee Swindlehurst, and Martin Haardt. Zero-forcing methods for downlink spatial multiplexing in multiuser MIMO channels. *IEEE Transactions on Signal Processing*, 52(2):461–471, 2004.

[79] Ti Ti Nguyen, Long Bao Le, and Quan Le-Trung. Computation offloading in MIMO based mobile edge computing systems under perfect and imperfect CSI estimation. *IEEE Transactions on Services Computing*, 14(6):2011–2025, 2019.

[80] Yonggang Wen, Weiwen Zhang, and Haiyun Luo. Energy-optimal mobile application execution: Taming resource-poor mobile devices with cloud clones. In *2012 proceedings IEEE Infocom*, pages 2716–2720. IEEE, 2012.

[81] Kien-Giang Nguyen, Le-Nam Tran, Oskari Tervo, Quang-Doanh Vu, and Markku Juntti. Achieving energy efficiency fairness in multicell MISO downlink. *IEEE Communications Letters*, 19(8):1426–1429, 2015.

[82] Erling D Andersen and Knud D Andersen. The mosek interior point optimizer for linear programming: an implementation of the homogeneous algorithm. In *High performance optimization*, pages 197–232. Springer, 2000.

[83] Tri Minh Nguyen, Wessam Ajib, and Chadi Assi. A novel cooperative non-orthogonal multiple access (NOMA) in wireless backhaul two-tier HetNets. *IEEE Transactions on Wireless Communications*, 17(7):4873–4887, 2018.

[84] Elie El Haber, Hyame Assem Alameddine, Chadi Assi, and Sanaa Sharafeddine. UAV-aided ultra-reliable low-latency computation offloading in future IoT networks. *IEEE Transactions on Communications*, 69(10):6838–6851, 2021.

[85] Mamta Narang, Simon Xiang, William Liu, Jairo Gutierrez, Luca Chiaraviglio, Arjuna Sathiaseelan, and Arvind Merwaday. UAV-assisted edge infrastructure for challenged networks. In *2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 60–65. IEEE, 2017.

[86] Fuhui Zhou, Rose Qingyang Hu, Zan Li, and Yuhao Wang. Mobile edge computing in unmanned aerial vehicle networks. *IEEE Wireless Communications*, 27(1):140–146, 2020.

[87] Latif U Khan, Ibrar Yaqoob, Nguyen H Tran, SM Ahsan Kazmi, Tri Nguyen Dang, and Choong Seon Hong. Edge-computing-enabled smart cities: A comprehensive survey. *IEEE Internet of Things Journal*, 7(10):10200–10232, 2020.

[88] Akram Al-Hourani, Sithamparanathan Kandeepan, and Simon Lardner. Optimal LAP altitude for maximum coverage. *IEEE Wireless Communications Letters*, 3(6):569–572, 2014.

[89] Giuseppe Faraci, Christian Grasso, and Giovanni Schembra. Fog in the clouds:

UAVs to provide edge computing to IoT devices. *ACM Transactions on Internet Technology (TOIT)*, 20(3):1–26, 2020.

[90] Seongah Jeong, Osvaldo Simeone, and Joonhyuk Kang. Mobile edge computing via a UAV-mounted cloudlet: Optimization of bit allocation and path planning. *IEEE Transactions on Vehicular Technology*, 67(3):2049–2063, 2017.

[91] Fuhui Zhou, Yongpeng Wu, Rose Qingyang Hu, and Yi Qian. Computation rate maximization in UAV-enabled wireless-powered mobile-edge computing systems. *IEEE Journal on Selected Areas in Communications*, 36(9):1927–1941, 2018.

[92] Hassan Hawilo, Manar Jammal, and Abdallah Shami. Network function virtualization-aware orchestrator for service function chaining placement in the cloud. *IEEE Journal on Selected Areas in Communications*, 37(3):643–655, 2019.

[93] David Öhmann, Meryem Simsek, and Gerhard P Fettweis. Achieving high availability in wireless networks by an optimal number of rayleigh-fading links. In *2014 IEEE Globecom Workshops (GC Wkshps)*, pages 1402–1407. IEEE, 2014.

[94] Peiyue Zhao and György Dán. Resilient placement of virtual process control functions in mobile edge clouds. In *2017 IFIP Networking Conference (IFIP Networking) and Workshops*, pages 1–9. IEEE, 2017.

[95] Albert Jonathan, Muhammed Uluyol, Abhishek Chandra, and Jon Weissman.

Ensuring reliability in geo-distributed edge cloud. In *2017 Resilience Week (RWS)*, pages 127–132. IEEE, 2017.

[96] Qin-Ma Kang, Hong He, Hui-Min Song, and Rong Deng. Task allocation for maximizing reliability of distributed computing systems using honeybee mating optimization. *Journal of Systems and Software*, 83(11):2165–2174, 2010.

[97] Peng-Yeng Yin, Shiuh-Sheng Yu, Pei-Pei Wang, and Yi-Te Wang. Task allocation for maximizing reliability of a distributed system using hybrid particle swarm optimization. *Journal of Systems and Software*, 80(5):724–735, 2007.

[98] Mushu Li, Nan Cheng, Jie Gao, Yinlu Wang, Lian Zhao, and Xuemin Shen. Energy-efficient UAV-assisted mobile edge computing: Resource allocation and trajectory optimization. *IEEE Transactions on Vehicular Technology*, 69(3):3424–3438, 2020.

[99] Zhe Yu, Yanmin Gong, Shimin Gong, and Yuanxiong Guo. Joint task offloading and resource allocation in UAV-enabled mobile edge computing. *IEEE Internet of Things Journal*, 7(4):3147–3159, 2020.

[100] Zhenjie Tan, Hua Qu, Jihong Zhao, Shiyu Zhou, and Wenjie Wang. UAV-aided edge/fog computing in smart IoT community for social augmented reality. *IEEE Internet of Things Journal*, 7(6):4872–4884, 2020.

[101] Zhaohui Yang, Cunhua Pan, Kezhi Wang, and Mohammad Shikh-Bahaei. Energy efficient resource allocation in UAV-enabled mobile edge computing

networks. *IEEE Transactions on Wireless Communications*, 18(9):4576–4589, 2019.

[102] Elie El Haber, Hyame Assem Alameddine, Chadi Assi, and Sanaa Sharafeddine. A reliability-aware computation offloading solution via UAV-mounted cloudlets. In *2019 IEEE 8th International Conference on Cloud Networking (CloudNet)*, pages 1–6. IEEE, 2019.

[103] Yong Zeng, Jie Xu, and Rui Zhang. Energy minimization for wireless communication with rotary-wing UAV. *IEEE Transactions on Wireless Communications*, 18(4):2329–2345, 2019.

[104] Weisen Shi, Junling Li, Wenchao Xu, Haibo Zhou, Ning Zhang, Shan Zhang, and Xuemin Shen. Multiple drone-cell deployment analyses and optimization in drone assisted radio access networks. *IEEE Access*, 6:12518–12529, 2018.

[105] Kashi Venkatesh Vishwanath and Nachiappan Nagappan. Characterizing cloud computing hardware reliability. In *Proceedings of the 1st ACM symposium on Cloud computing*, pages 193–204, 2010.

[106] Phillipa Gill, Navendu Jain, and Nachiappan Nagappan. Understanding network failures in data centers: measurement, analysis, and implications. In *Proceedings of the ACM SIGCOMM 2011 Conference*, pages 350–361, 2011.

[107] Qi Zhang, Mohamed Faten Zhani, Maissa Jabri, and Raouf Boutaba. Venice: Reliable virtual data center embedding in clouds. In *IEEE INFOCOM 2014-IEEE Conference on Computer Communications*, pages 289–297. IEEE, 2014.

[108] Long Qu, Maurice Khabbaz, and Chadi Assi. Reliability-aware service chaining in carrier-grade softwarized networks. *IEEE Journal on Selected Areas in Communications*, 36(3):558–573, 2018.

[109] Sara Ayoubi, Yanhong Zhang, and Chadi Assi. A reliable embedding framework for elastic virtualized services in the cloud. *IEEE Transactions on Network and Service Management*, 13(3):489–503, 2016.

[110] Boris Galkin, Jacek Kibilda, and Luiz A DaSilva. UAVs as mobile infrastructure: Addressing battery lifetime. *IEEE Communications Magazine*, 57(6):132–137, 2019.

[111] Azade Fotouhi, Haoran Qiang, Ming Ding, Mahbub Hassan, Lorenzo Galati Giordano, Adrian Garcia-Rodriguez, and Jinhong Yuan. Survey on UAV cellular communications: Practical aspects, standardization advancements, regulation, and security challenges. *IEEE Communications Surveys & Tutorials*, 21(4):3417–3442, 2019.

[112] Jie Ouyang, Yueling Che, Jie Xu, and Kaishun Wu. Throughput maximization for laser-powered UAV wireless communication systems. In *2018 IEEE International Conference on Communications Workshops (ICC Workshops)*, pages 1–6. IEEE, 2018.

[113] Mustafa Kishk, Ahmed Bader, and Mohamed-Slim Alouini. Aerial base station deployment in 6G cellular networks using tethered drones: The mobility and endurance tradeoff. *IEEE Vehicular Technology Magazine*, 15(4):103–111, 2020.

[114] Yujie Qin, Mustafa A Kishk, and Mohamed-Slim Alouini. Performance evaluation of UAV-enabled cellular networks with battery-limited drones. *IEEE Communications Letters*, 24(12):2664–2668, 2020.

[115] Emmanouil Fountoulakis, Georgios S Paschos, and Nikolaos Pappas. UAV trajectory optimization for time constrained applications. *IEEE Networking Letters*, 2(3):136–139, 2020.

[116] Emmanouil Lakiotakis, Pavlos Sermpezis, and Xenofontas Dimitropoulos. Joint optimization of UAV placement and caching under battery constraints in uav-aided small-cell networks. In *Proceedings of the ACM SIGCOMM 2019 Workshop on Mobile AirGround Edge Computing, Systems, Networks, and Applications*, pages 8–14, 2019.

[117] Qiyu Hu, Yunlong Cai, Guanding Yu, Zhijin Qin, Minjian Zhao, and Geoffrey Ye Li. Joint offloading and trajectory design for UAV-enabled mobile edge computing systems. *IEEE Internet of Things Journal*, 6(2):1879–1892, 2018.

[118] Le Thi Hoai An and Pham Dinh Tao. The DC (difference of convex functions) programming and DCA revisited with DC models of real world nonconvex optimization problems. *Annals of operations research*, 133(1):23–46, 2005.

[119] Elie El Haber, Mohamed Elhattab, Chadi Assi, Sanaa Sharafeddine, and Kim Khoa Nguyen. Latency and reliability aware edge computation offloading via an intelligent reflecting surface. *IEEE Communications Letters*, 25(12):3947–3951, 2021.

[120] Tong Bai, Cunhua Pan, Yansha Deng, Maged Elkashlan, Arumugam Nallanathan, and Lajos Hanzo. Latency minimization for intelligent reflecting surface aided mobile edge computing. *IEEE Journal on Selected Areas in Communications*, 38(11):2666–2682, 2020.

[121] Yang Liu, Jun Zhao, Zehui Xiong, Dusit Niyato, Chau Yuen, Cunhua Pan, and Binbin Huang. Intelligent reflecting surface meets mobile edge computing: Enhancing wireless communications for computation offloading. *arXiv preprint arXiv:2001.07449*, 2020.

[122] Tong Bai, Cunhua Pan, Hong Ren, Yansha Deng, Maged Elkashlan, and Arumugam Nallanathan. Resource allocation for intelligent reflecting surface aided wireless powered mobile edge computing in OFDM systems. *IEEE Transactions on Wireless Communications*, 20(8):5389–5407, 2021.

[123] Zheng Chu, Pei Xiao, Mohammad Shojafar, De Mi, Juquan Mao, and Wanming Hao. Intelligent reflecting surface assisted mobile edge computing for internet of things. *IEEE Wireless Communications Letters*, 10(3):619–623, 2020.

[124] Zhiyang Li, Ming Chen, Zhaohui Yang, Jingwen Zhao, Yinlu Wang, Jianfeng Shi, and Chongwen Huang. Energy efficient reconfigurable intelligent surface enabled mobile edge computing networks with NOMA. *IEEE Transactions on Cognitive Communications and Networking*, 7(2):427–440, 2021.

[125] Qingqing Wu, Wen Chen, Derrick Wing Kwan Ng, Lajos Hanzo, et al. Irs-aided wireless powered mec systems: Tdma or noma for computation offloading? *arXiv e-prints*, pages arXiv–2108, 2021.

[126] Nouha Kherraf, Sanaa Sharafeddine, Chadi M Assi, and Ali Ghrayeb. Latency and reliability-aware workload assignment in IoT networks with mobile edge clouds. *IEEE Transactions on Network and Service Management*, 16(4):1435–1449, 2019.

[127] Anthony Man-Cho So, Jiawei Zhang, and Yinyu Ye. On approximating complex quadratic optimization problems via semidefinite programming relaxations. *Mathematical Programming*, 110(1):93–110, 2007.

[128] Imre Pólik and Tamás Terlaky. Interior point methods for nonlinear optimization. In *Nonlinear optimization*, pages 215–276. Springer, Jan 2010.

[129] SM Ahsan Kazmi, Nguyen H Tran, Walid Saad, Zhu Han, Tai Manh Ho, Thant Zin Oo, and Choong Seon Hong. Mode selection and resource allocation in device-to-device communications: A matching game approach. *IEEE Transactions on Mobile Computing*, 16(11):3126–3141, 2017.

Appendix A

Proofs

A.1 Proof of Equivalence between (2.7) and (2.10)

To prove that (2.7) and (2.10) are equivalent, we must prove that at optimality, all the constraints (2.10b), (2.10c), and (2.10e) are active, e.g., all constraints occurs at equality. We prove this by contradiction. Assuming that at the optimal solution $\boldsymbol{p}^\star, \boldsymbol{\alpha}^\star, \boldsymbol{x}^\star, \boldsymbol{y}^\star, \boldsymbol{f}^\star, \boldsymbol{u}^\star, \boldsymbol{v}^\star, \boldsymbol{t}^\star, \boldsymbol{\zeta}^\star, \eta^\star$, some of the constraints in (2.10b), (2.10c), and (2.10e) achieves at strict inequality. Let us assume that the strict inequality occurs at constraint (2.10b) at index i_1 (or (2.10e)). At this point, we remark that when we can easily find another value $\tilde{\zeta}_{i_1} > \zeta^\star_{i_1}$ (or $\tilde{\eta} > \eta^\star$) so that (2.10b) at index i_1 (or (2.10e)) becomes equality and result in a higher objective function of problem (2.10). This contradicts our assumption of optimality. On the other hand, assuming that the strict inequality occurs at constraint (2.10c) at some index i_1, j_1. Similarly, we can easily find an upper-scaled value $\tilde{t}_{i_1 j_i} > t^\star_{i_1 j_i}$ to make (2.10c) become equality.

However, this subsequently makes (2.10b) at index i_1 become strict inequality, and again we can find a value $\tilde{\zeta}_{i_1} > \zeta^{\star}_{i_1}$ to make this constraint equality to result higher objective function. This also contradicts our assumption. Thus, at optimality, all the constraints (2.10b), (2.10c), and (2.10e) are active. This completes the proof.

A.2 Proof of Equivalence between (3.10b) and (3.17a), (3.17e)

The equivalence between (3.10b) and (3.17a), (3.17e) is proved if there exists a feasible solution taken from constraint (3.10b), it must also satisfy (3.17a), (3.17e). In addition, if there exists a feasible solution taken from constraints (3.17a), (3.17e), it must also satisfy (3.10b). Let us take first the case when a feasible solution to (3.17a), (3.17e) are $s^{\circ}_{i,j,2}, r^{\mathrm{ue},\circ}_{i,j,2}, f^{\mathrm{sc},\circ}_{i,j}, \theta^{\circ}_{i,j,2}$. In this case, since $s^{\circ}_{i,j,2}$ is not greater than $(\theta^{\circ}_{i,j,2})^2$ in (3.17e), this means that we can find $(\theta^{\circ}_{i,j,2})^2$ to replace into (3.17a) and hence (3.10b) is satisfied.

Now let us take the case when (3.10b) is satisfied and the feasible solution for (3.10b) is $s^{\circ}_{i,j,2}, r^{\mathrm{ue},\circ}_{i,j,2}, f^{\mathrm{sc},\circ}_{i,j}$. In this case, we can easily choose a value $\theta^{\circ}_{i,j,2} = \sqrt{s^{\circ}_{i,j,2}}$ so that (3.17a), (3.17e) are satisfied. This completes the proof.

A.3 Convexification Steps of (4.16)

First, we re-arrange the terms of (4.10c) as follows:

$$1 - \bar{R}_i \geq \prod_{j \in \mathcal{M}} (1 - o_{ij}\phi_j) \tag{A.1}$$

Then, after taking the natural logarithm of both sides and using its properties, constraint (A.1) can be equivalently rewritten as:

$$\ln(1 - \bar{R}_i) \geq \sum_{j \in \mathcal{M}} \ln(1 - o_{ij}\phi_j) \tag{A.2}$$

The RHS of (A.2) $\forall\ j \in \mathcal{M}$ is equals to $\ln(1 - \phi_j)$ if $o_{ij} = 1$, and to 0 if $o_{ij} = 0$. Thus, (A.2) can be replaced by the linear constraint (4.16).

A.4 Convexification Steps of (5.23c)

First, we re-arrange the terms of (5.21d) as:

$$1 - \bar{R}_k \geq \prod_{j \in \mathcal{M}} (1 - x_{kj}\phi_j) \tag{A.3}$$

Then, after taking the natural logarithm of both sides and using its properties, constraint (A.3) can be equivalently rewritten as:

$$\ln(1 - \bar{R}_k) \geq \sum_{j \in \mathcal{M}} \ln(1 - x_{kj}\phi_j) \tag{A.4}$$

The RHS of (A.4) $\forall\ j \in \mathcal{M}$ is equal to $\ln(1-\phi_j)$ if $x_{kj} = 1$, and to 0 if $x_{kj} = 0$. Thus, (A.4) can be replaced by the following linear constraint:

$$\ln(1-\bar{R}_k) \geq \sum_{j \in \mathcal{M}} x_{kj} \ln(1-\phi_j) \tag{A.5}$$