Provisioning of Edge Computing Resources for Heterogeneous IoT Workload

Nouha Kherraf

A Thesis

in

The Department

of

Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements for the Degree of

Master of Applied Science (Electrical and Computer Engineering) at

Concordia University

Montréal, Québec, Canada

March 2019

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: Nouha Kherraf

Entitled: **Provisioning of Edge Computing Resources for Heterogeneous IoT Workload**

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

Master of Applied Science (Electrical and Computer Engineering)

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

Signed by the Final Examining Committee:

_____ Chair
Dr. Otmane Ait Mohamed

_____ External Examiner
Dr. Jamal Bentahar

_____ Examiner
Dr. Otmane Ait Mohamed

_____ Supervisor
Dr. Chadi Assi

_____ Co-supervisor
Dr. Ali Ghrayeb

Approved by _____
Dr. W. E. Lynch, Chair
Department of Electrical and Computer Engineering

_____ 2019

_____
Amir Assif, Dean
Faculty of Engineering and Computer Science

Abstract

Provisioning of Edge Computing Resources for Heterogeneous IoT Workload

Nouha Kherraf

With the evolution of cellular networks, the number of smart connected devices have witnessed a tremendous increase to reach billions by 2020 as forecasted by Cisco, constituting what is known today as the Internet of Things (IoT). With such explosion of smart devices, novel services have evolved and invaded almost every aspect of our lives; from e-health to smart homes and smart factories, etc. Such services come with stringent QoS requirements. While the current network infrastructure (4G) is providing an acceptable QoE to the end users, it will be rendered obsolete when considering the critical QoS requirements of such new services. Hence, to deliver the seamless experience these services provide, MEC has emerged as a promising technology to offer the cloud capabilities at the edge of the network, and hence, meeting the low latency requirements of such services. Moreover, another QoS parameter that needs to be addressed is the ultra high reliability demanded by the IoT services. Therefore, 5G has evolved as a promising technology supporting ultra Reliable Low Latency Communication (uRLLC) and other service categories. While integrating uRLLC with MEC would help in realizing such services, it would however raise some challenges for the network operator. Thus, in this thesis, we address some of these challenges. Specifically, in the second chapter, we address the problem of MEC Resource Provisioning and Workload Assignment (RPWA) in an IoT environment, with heterogeneous workloads demanding services with stringent latency requirements. We formulate the problem as an MIP with the objective to minimize the resources deployment cost. Due to the complexity of the problem, we will develop a decomposition approach (RPWA-D) to solve the problem and study through different simulations, the performance of our approach. In chapter 3, we consider both ultra high reliability and low latency requirements of different IoT services, and solve the Workload Assignment problem (WA) in an IoT environment.

We formulate the problem as an MIP with the objective of maximizing the admitted workload to the network. After showing the complexity of the problem and the non scalability of the WA-MIP, we propose two different approaches; WA-D and WA-Tabu. The results show that WA-Tabu was the most efficient and scalable.

Acknowledgments

I would like to express my deepest gratitude to my supervisors Dr. Chadi Assi and Dr. Ali Ghrayeb, for their continuous support, motivation and guidance.

I would like to give a special thanks to my mother Hanane Larabi for her endless love and support, without you, this would have never been possible. I would also like to thank my father Abdelkarim Ouarie for encouraging me to step a leg in this path and pursue my dreams. To my sisters Amira, Achouak and Hala, thank you for cheering me up when I needed it. To my aunt Najwa Larabi, you have been a great support, thank you so much.

I would also like to take the opportunity to thank my friends and colleagues in Qatar, Algeria and Montreal for the wonderful times and interesting discussions we had over the past two years.

Last but not least, I would like to give a special thanks to my wonderful grandfather Abdallah Larabi, for his continuous support, prayers and encouragement.

Contents

List of Figures

List of Tables

List of Abbreviations

1G	First Generation
2G	Second Generation
3G	Third Generation
4G	Fourth Generation
5G	Fifth Generation
6G	Sixth Generation
APs	Access Points
AR	Augmented Reality
BSs	Base Stations
DALA	Delay Aware Load Assignment
eMBB	enhanced Mobile Broadband
ILP	Integer Linear Program
IoT	Internet of Things
ITS	Intelligent Transportation Systems
ITU-R	International Telecommunication Union - Radio communication sector
LAWA	Latency Aware Workload Assignment
LAWA-MIP	Latency Aware Workload Assignment - Mixed Integer Program
MCC	Mobile Cloud Computing
MEC	Mobile Edge Computing
MESD	Mobile Edge Servers Dimensioning
MIP	Mixed Integer Program
mMTC	massive Machine Type Communication

NFC	Near Field Communication
QoE	Quality of Experience
QoS	Quality of Service
RACS	Reliability Aware Candidate Selection
RFID	Radio Frequency Identification
RPWA	Resource Provisioning and Workload Assignment
RPWA-D	Resource Provisioning and Workload Assignment - Decomposition approach
RPWA-MIP	Resource Provisioning and Workload Assignment - Mixed Integer Program
SDN	Software Defined Networking
UE	User Equipment
uRLLC	ultra Reliable Low Latency Communication
VMs	Virtual Machines
WA	Worload Assignment
WA-D	Workload Assignment - Decomposition approach
WA-MIP	Workload Assignment - Mixed Integer Program
WA-Tabu	Workload Assignment - Tabu search-based algorithm

Chapter 1

Introduction

1.1 Evolution of cellular communication technologies

Over the past few decades, mobile wireless technologies have evolved to adapt to the ever emerging people's lifestyle. With the introduction of new cellular generation (G) every ten years, higher data rates and improved quality of service (QoS) are provided to the end users. It all started when the first generation (1G) was introduced in early 1980's [3]. The first generation was designed to support voice communication with a data rate up to 2.4kbps. In 1990's, the second generation (2G) was established to support short messages (SMS) and e-mail in addition to the voice communication with a data rate up to $64 - 144$kbps. The transmission data rate raised up to $2Mbps$ with the introduction of $3G$ in 2000 supporting enhanced voice and data capacity. This in return, opened the door for a wide variety of applications, such as live TV broadcasting, weather forecasts and video calls [4]. Along with these applications, a new wave of smart devices; such as tablets and smart phones were made available to the users. As technology advances, user expectations rise, and hence, as a response to this demand, 4G was launched to support up to 100Mbps data rate for users with high mobility and up to 1Gbps for users with local wireless access [5]. The fourth generation led to an increase in the number of smart devices trying to connect to the internet on a daily basis (e.g. smart watches, smart TV, smart cars, etc.). Further, as predicted by Cisco, the number of the devices connected to the internet is expected to exceed 50 billions by 2020 [6]. These connected devices constitute what is known today as the Internet of Things (IoT) [7].

1.2 IoT paradigm and 5G

The Internet of Things could be defined as a network of smart connected objects with the ability of collecting and sharing data, as well as reacting to changes in the environment. For many, the term "connected objects" is associated only to smart phones, tablets, laptops and PCs connected to the internet. However, in the current era of smart things, many other devices such as sensors and actuators are also connected to the internet [8]. As mentioned earlier, the number of the IoT devices is increasing dramatically and it is excepting to reach billions of devices in the near future. With such proliferation of devices, the IoT is transforming our environment into a smart one. For instance, in a smart hospital, the integration of IoT allows for intelligent drug/medecine control and patient tracking and monitoring. Moreover, integrating IoT in transportation systems realized the so called "Intelligent Transportation Systems (ITS)" allowing for traffic guidance, remote vehicle monitoring, vehicle and road coordination, etc. [9]. Due to the increase in the number of IoT devices and the amount of the data they generate, as well as the heterogeneity of this data along with their stringent QoS requirements (e.g. ITS has low latency $(10 - 100ms)$ and ultra high reliability (99.9999%) requirements), the current network infrastructure (4G) would fall short in supporting these challenges. This, in return, has raised research interest in the development of the next generation cellular networks (5G) [10].

The promise of 5G is to provide the end users with a better QoS and user experience through supporting higher data rates, better coverage and massive number of connected devices. Furthermore, according to the International Telecommunication Union-Radio communication sector (ITU-R), services promised by the fifth generation are categorized as 1) enhanced Mobile Broadband (eMBB); supporting high data rates for stable connections making it suitable for Augmented Reality (AR) and 360-degree video streaming applications, 2) ultra Reliable Low Latency Communications (uRLLC); that supports applications with low latency and ultra high reliability requirements such as factory automation ($0.25 - 10$ms latency and 99.999% reliability), and 3) massive Machine Type Communications (mMTC) which supports a massive number of connected devices [11]. In order to realize the delivery of such services categories offered by 5G and ensure a satisfactory QoS to the end users, some enabling technologies have emerged. Traditionally, the Mobile Cloud Computing was

the go to solution providing computing and storage services to mobile users remotely. Mobile Cloud Computing came as a direct response to address the limitations of the IoT devices. Even though IoT devices support real time applications (e.g.Augmented Reality (AR), Virtual Reality (VR), etc.), they fall short in executing such resource-hungry applications due to their limited storage and computing resources. By leveraging the remote cloud capabilities, mobile devices (e.g.smart phones) running real time applications offload their computing intensive tasks, either entirely or partly, to be processed at the remote cloud [12]. Although Mobile Cloud Computing (MCC) helped in the realization of real time applications, the offloading approach faces some challenges such as the long response times resulting from the large distance between the end users and the remote cloud server, as well as the high bandwidth requirements. With such shortcomings of the MCC, providing the service categories offered by 5G would not be possible. Hence, bringing the cloud's capabilities closer to the users would be a key technology to realize those services, which led to the emergence of Mobile Edge Computing (MEC) [13].

1.3 Mobile Edge Computing (MEC)

It is now clear that only depending on the remote cloud computing and storage capabilities is not sufficient for those latency critical applications. Hence, the concept of Mobile Edge Computing (MEC) has emerged promising better QoE to end users by bringing the remote cloud's computing and storage capabilities in a close proximity to the mobile users reducing the latency to milliseconds. The edge-cloud was first introduced by Satyanarayan in the form of a cloudlet, which was defined as a micro-cloud that connects the nearby mobile devices (e.g.laptops, tablets, etc.) via WiFi/Bluetooth [14] [12]. The cloudlet infrastructure consists of three layers; mobile devices, cloudlets and a centralized data-center. Leveraging the virtualization technologies, mobile users can offload their latency sensitive and computationally intensive tasks to the nearby cloudlets placed in local areas (e.g.coffee shops, train stations, etc) to be processed by applications running on virtual machines (VMs) hosted on the cloudlets [14]. Unlike cloudlets, mobile edge computing (MEC) has been introduced as a more sophisticated approach to allow all mobile users, not only specific

users, to access the cloud computing capabilities at the edge of the network. Mobile edge computing was first introduced by the European Telecommunications Standards Institute (ETSI) as edge servers collocated nearby base stations (BSs). The edge servers could refer to either base stations or datacenters collocated with base stations [15]. Mobile Edge Computing was then redefined as Multi-access Edge Computing to include more access technologies such as WiFi, Z-wave and fixed access technologies. One other advantage of MEC is its location and context awareness. This offers the service providers the opportunity to collect more information about their customers' location and interests, and hence, provide them with more personalized services improving their QoE. Another edge computing term that was introduced by Cisco is Fog Computing. Any equipment with storage and computing capabilities could be considered as a fog node, such as routers, base stations, access points, vehicles, etc. [14]. While there exist some differences between Multi-access edge nodes, fog nodes and cloudlets, these three terms have been used interchangeably in the literature to refer to nodes deployed at the edge of the network. Within the IoT enviroment, edge servers could be deployed to serve the great amount of data generated by the IoT devices by running different types of IoT applications. With such promising technology, integrating MEC with 5G in an IoT environment would raise new challenges. Particularly, the challenges could be categorized into the following: 1*) Edge servers placement:* since one of the goals of the adoption of MEC is to serve devices with low latency demands, misplacing the edge servers could result in long network delays and unbalanced distribution of workloads. Hence, studying different strategies to optimize the placement of edge servers is of great importance. Some work in the literature has addressed this problem [16] [17] [18] [19]. For instance the work in [18] addressed this problem within a smart city environment. They formulated the problem as a multi-objective MIP to minimize the access delays and balance workloads by minimizing the workload difference between edge servers. 2*) IoT applications provisioning:* since IoT devices generate tremendous workloads demanding different IoT services, edge servers host IoT applications of different types to serve them. Therefore, provisioning IoT applications would play a crucial role in ensuring a satisfactory QoS to the end users. Hence, different research has targeted the provisioning of these applications in terms of determining their allocated computing resources and the edge servers hosting them [20] [21] [22]. 3*) Task*

offloading and assignment: MEC has emerged allowing users to offload their computationally intensive tasks to the edge of the network to overcome the shortcomings of their devices. Furthermore, although MEC was a response to the long propagation delays in MCC, processing the tasks at the network edge would still incur some delays (e.g. access delays, network delays, processing delays, etc.). Given the heterogeneous low latency requirements of the IoT services, optimizing the assignment of the generated workloads from the IoT devices is extremely important to deliver a seamless experience to the users. Thus, extensive research has been done in this area [23] [24] [25] [26].

1.4 Contributions

In this thesis, we aim to study the aforementioned challenges, namely, edge servers deployment problem; that is deciding on the number and location of the edge servers, IoT applications placement problem; which is deciding on the number and placement of IoT applications as well as the amount of computing resources allocated to them, and the workload assignment problem; that is deciding on the mapping of the IoT devices generated workloads to the edge servers. Further, all these challenges are coupled with the stringent QoS requirements of the IoT services. We investigate these challenges in details and provide efficient approaches to address them. Below is a brief description of our contributions.

1.4.1 Optimized Provisioning of Edge Computing Resources with Heterogeneous Workload in IoT Networks

Managing the heterogeneity of the workload generated by IoT devices, especially in terms of computing and delay requirements, while being cognizant of the cost to network operators requires an efficient dimensioning of the MEC-enabled network infrastructure. Hence, in this chapter, we study and formulate the problem of MEC Resource Provisioning and Workload Assignment for IoT services (RPWA) as a Mixed Integer Program (MIP) to jointly decide on the number and the location of edge servers and applications to deploy, in addition to the workload assignment. Given its complexity, we propose a decomposition approach to solve it which consists of decomposing RPWA

into the Delay Aware Load Assignment (DALA) sub-problem and the Mobile Edge Servers Dimensioning (MESD) sub-problem. We analyze the effectiveness of the proposed algorithm through extensive simulations and highlight valuable performance trends and trade-offs as a function of various system parameters.

1.4.2 Heterogeneous workload assignment in an MEC-IoT environment for uRLLC

Along with the dramatic increase in the number of IoT devices, different IoT services with heterogeneous QoS requirements are starting to see the light with the aim of making the current society smarter and more connected. In order to deliver such services to the end users, the network infrastructure has to accommodate the tremendous workload generated by the smart devices and their heterogeneous and stringent latency and reliability requirements. This would only be possible with the emergence of uRLLC promised by 5G. MEC has emerged as an enabling technology to help with the realization of such services by bringing the remote cloud computing and storage capabilities closer to the users. However, integrating uRLLC with MEC would require the network operator to efficiently map the generated workloads to MEC nodes along with resolving the trade-off between the latency and reliability requirements. Thus, we study in this chapter the problem of Workload Assignment (WA) and formulate it as a Mixed Integer Program (MIP) to decide on the assignment of the workloads to the available MEC nodes. Due to the complexity of the WA problem, we decompose the problem into two subproblems; Reliability Aware Candidate Selection (RACS) and Latency Aware Workload Assignment (LAWA-MIP). We evaluate the performance of the decomposition approach and propose a more scalable approach; Tabu meta-heuristic (WA-Tabu). Through extensive numerical evaluation, we analyze the performance and show the efficiency of our proposed approach under different parameters.

The rest of the thesis is organized as follows, we present our first and second contributions in chapter 2 and 3 respectively. We then conclude and present future work in chapter 4.

Chapter 2

Optimized Provisioning of Edge Computing Resources with Heterogeneous Workload in IoT Networks

2.1 Introduction

The number of Internet-connected devices (e.g., smart phones, tablets, smart cameras, industrial sensors, connected cars, smart traffic lights, etc.) is expected to exceed 50 billions by 2020 [27], hence inadvertently realizing the paradigm of the so-called Internet of Things/Everything (IoT/E). Given the immense proliferation of IoT devices, continuous advancements and development of data analytics and networking will empower them with enhanced real-time capabilities [28]. For instance, advances in Radio Frequency Identification (RFID) and Near Field Communication (NFC) technologies made real-time monitoring of almost every entity in a supply chain possible, from inventory tracking to after-sales services. Moreover, sensor technologies have enabled real-time monitoring and processing of traffic flows and vehicles' information in an intelligent traffic ecosystem. The integration of IoT devices in the healthcare domain enabled tracking patients and monitoring

their vital signs [29]. However, owing to the shear volume of data these devices generate, they fall short in terms of computing capacity and storage [30], hence restricting their capabilities and hindering the performance of the applications they can support. Traditionally, the cloud has been the go-to solution for providing storage and computing resources to process the vast amount of data generated by IoT devices and to execute the necessary analytics [31, 32]. However, offloading the workload generated by these devices to a remote cloud infrastructure for processing is subject to high communication delays and energy consumption which will eventually violate the latency requirements of real-time IoT applications [23, 33].

Mobile Edge Computing (MEC) has emerged as a new paradigm to overcome the aforementioned challenges. By leveraging a distributed range of computing and storage resources deployed in close proximity to User Equipment (UE), MEC provides the IoT devices the opportunity to offload and run their workload on a wide range of IoT applications deployed on edge servers, hence, supplying them with varying Quality of Service (QoS) requirements [34]. Unlike the traditional centralized cloud, edge servers are collocated with 4G/5G cellular Base Stations (BSs) [35] deployed at the edge of the network. Recently, the term MEC has been redefined as Multi-access Edge Computing, extending its applicability to include new connectivity options such as WiFi, Z-wave and fixed access technologies and, hence, enabling the support of a wider variety of devices and use cases [14]. Another paradigm that has been introduced by Cisco is fog computing which also brings the cloud's intelligence and processing capabilities to the edge of the network. Fog, however, offers a multi-layer cloud computing architecture where fog nodes are deployed in different network tiers (i.e., small base stations, vehicles, wifi access point, and user terminals) [36] [37]. The terms fog and edge computing have been used interchangeably in the literature [38]. Further, edge computing capabilities can be enabled at business premises and accessed through wifi Access Points (APs) and are typically known as cloudlets [37]. IoT devices access these edge resources by connecting via an existing wireless access network technology, therefore providing faster response times and saving bandwidth by reducing the load on the network core [23, 33].

In order to enable MEC capabilities, current network infrastructure should be dimensioned to support the deployment of edge servers. However, the cost of such deployment is one of the major obstacles facing network operators given the massive number and spatial spread of IoT devices

which are expected to be served within tolerable/low delays [33, 39]. In addition, as the offloaded IoT workloads are required to be processed by different types of applications, usually running on Virtual Machines (VMs) hosted on the edge servers, the decision on the number of instances and the computing resources to assign to each of them becomes challenging and has a direct impact on the response time achieved. Finally, as many IoT devices may be requiring the edge servers capabilities at the same time, efficient and dynamic assignment of their workloads to the hosted applications is required. Since collectively addressing these challenges is a difficult task, the authors of [23] assumed already deployed cloudlets (e.g. edge servers) and addressed the problem of IoT applications placement and workload assignment. The problem of joint application placement on fog nodes and data stream routing from IoT devices to them has been considered recently in [33] with both bandwidth and delay performance guarantees. Task offloading to cloudlets interconnected through a metropolitan wide area network has also been studied in [40], where tasks require virtual functions for their processing. In addition, the optimal placement of cloudlets in a metropolitan area network has been addressed in [41] with the objective of balancing the workload among the deployed cloudlets.

Unlike the work in the literature, we envision an environment with a large number of IoT devices requesting a set of delay-sensitive services (e.g., smart cities, connected cars, industrial control, environmental monitoring, etc.) [42, 43] that can be offered by a wide range of applications. We address the *MEC Resource Provisioning and Workload Assignment for IoT services (RPWA)* problem which consists of jointly solving: 1) *The MEC dimensioning sub-problem* which consists of deciding on the number and the placement of edge servers; 2) *The IoT applications placement sub-problem* which aims at determining the number and the placement of different types of applications' instances to deploy on the edge servers, in addition to deciding on the computing resources (i.e., CPU shares) to allocate to each of them; 3) *The workload assignment sub-problem* which proposes the assignment of the workload generated by IoT devices to the required type of application instance hosted on a suitable edge server and able to achieve its required response time. We formulate the RPWA problem as a Mixed Integer Program (RPWA-MIP) with the objective of minimizing the edge servers deployment cost. As we prove its NP-Hardness, we exploit the inter-dependency existing between the three aforementioned sub-problems composing it, and propose

a decomposition approach (RPWA-D) to address its different aspects in a more efficient and scalable strategy. Hence, we divide the RPWA problem into two sub-problems: 1) *The Delay Aware Load Assignment (DALA) sub-problem* which solved the workload assignment sub-problem while deciding on the number and the computing resources to assign to the applications to deploy; and 2) *The Mobile Edge Servers Dimensioning (MESD) sub-problem* which solves the MEC dimensioning sub-problem while deciding on the placement of the applications that needs to be deployed (i.e., provided by DALA) on the provisioned edge servers. Through extensive numerical evaluation, we explore and analyze the different trade-offs existing between the edge servers deployment cost and the workloads of variable QoS requirements which can be served. Under varying parameters, we show that our proposed decomposition approach is efficient, scalable and provide comparable results to those provided by the RPWA-MIP.

The remainder of the paper is organized as follows. Section 2.2 presents the literature review. Section 2.3 introduces the system model. Section 2.4 defines and formulates the RPWA problem. Section 2.5 presents and explains the proposed decomposition approach. Our numerical evaluation is depicted in Section 2.6. We conclude in Section 2.7.

2.2 Literature review

The new concept of MEC has stimulated much research work in the past few years, of which we are going to survey a prime selection of the most closely related.

2.2.1 Task offloading and resource allocation in MEC

The authors of [44] have recently presented a survey on exploiting MEC technologies for the realization of IoT applications. The authors in [45] considered the cloudlets placement problem in a Software Defined Networking (SDN)-based IoT network with a focus on minimizing the average cloudlet access delay. They assumed that the SDN control plane manages the routing of IoT devices' requests by commanding a set of SDN-enabled APs. They proposed an enumeration-based algorithm that finds the optimal placement of the cloudlets by evaluating all possible combinations. To reduce the complexity, they devised another ranking-based near-optimal algorithm for the cloudlets

placement. The authors, however, did not consider the cloudlets deployment cost.

The work in [46] accounted for the static and dynamic design of an edge cloud network, while respectively considering the absence and the presence of user mobility. Thus, they presented a column generation approach to determine the sites on which the cloudlets have to be installed. Then, they determined the BSs to cloudlets assignment. Finally, the authors addressed the resource allocation problem in terms of determining the placement of each VM required by an end device with respect to its mobility conditions and latency requirement. Although the authors in [46] took into account the cloudlets deployment cost, they did not consider the sharing of VMs between multiple end devices.

The authors in [47] developed an Integer Linear Program (ILP) model for the placement of IoT application services. Unlike [46], the authors considered that VMs could be shared by multiple IoT devices. They solved the model iteratively by considering a new objective at each iteration. These objectives not only addressed the cost efficiency of operating the network (i.e., minimizing the number of active computation nodes and gateways, maximizing the number of admitted applications requests, etc.) but also accounted for the latency reduction (minimizing the hop count). The work in [22] considered the data placement of IoT applications in a fog infrastructure with the objective to minimize the network latency. The authors developed a divide and conquer heuristic in which the fog nodes were weighted and partitioned, resulting in a data placement sub-problem for each partition. Each sub-problem was then solved using algorithms implemented in the simulator iFogSim. However, in both works [47] [22], specific latency requirement for each IoT application was overlooked.

Unlike [22, 47], the authors in [33] jointly considered the IoT applications latency and bandwidth requirements. They tackled the applications' requests assignment problem in a fog infrastructure and developed a provisioning model that is responsible for applications' data routing and assignment to the fog nodes. However, they considered that each application has specific hardware requirements, preventing it from being able to be served by any of the fog nodes as some of them may be deprived from the requested hardware resources. The authors in [23] tackled both the resource allocation and IoT application requests assignment problems in an edge infrastructure. They accounted for the limited computing resources of the cloudlets in addition to both network

and computation delays of the application requests. They assumed that any IoT application has a specific latency requirement and, unlike [33], can be hosted on top of a VM deployed on any cloudlet. Within this framework, the authors were aiming at minimizing the response time of the applications' requests. Hence, they developed an algorithm that sequentially solves the assignment and resource allocation problems.

The problem of optimal placement of cloudlets in a metropolitan area network, where cloudlets are assumed to be collocated with APs, has been addressed in [41]. Given the complexity of the problem, the authors presented two heuristic solutions. They also considered the users to cloudlet assignment problem to minimize the response time, and noted that routing traffic normally to the closest cloudlet may not always yield a satisfactory solution in terms of response times and, thus, it is necessary to balance the workload among the deployed cloudlets. Task offloading to cloudlets hosting virtual network functions has been considered in [40], and more recently, in [48] for MEC in software defined ultra dense networks.

The resource allocation problem was also considered in the central cloud by the authors of [49,50]. In [49], the authors considered a cloud resource provisioning scheme to reduce the price the customers have to pay for leasing cloud resources using stochastic optimization. However, in [50], the authors presented a bidding strategy to decide on the customers to serve with the objective of maximizing the cloud provider profit while minimizing the amount of Service Level Agreement Violation (SLA). While in both works the cloud infrastructure was considered as available, we aim in this work at planning and dimensioning the edge cloud infrastructure.

2.2.2 Novelty of our work

To the best of our knowledge, the work in the literature focused on solving at most two of the aforementioned subproblems; MEC dimensioning, IoT placement and workload assignment. Further, most of the work mentioned above assumed a homogeneous workload or a single user equipment. While the work in [23] considered the heterogeneity of applications' requests in an IoT environment, they, however, did not address the MEC dimensioning problem and assumed a network where the edge servers are already placed. The novelty of our work is, therefore, manifested by jointly solving the three above mentioned problems and considering a large scale IoT environment

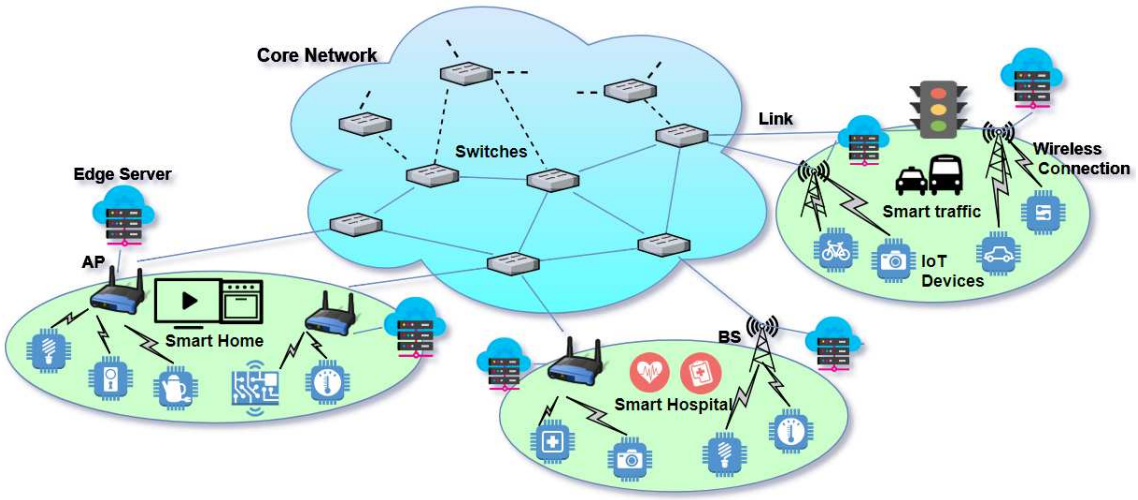


Figure 2.1: MEC-enabled smart environment.

with heterogeneous applications' requests.

2.3 System Model

2.3.1 MEC-enabled Smart Environment

The system model we target resembles the scenario depicted in Fig. 2.1. A metropolitan area encompassing a massive number of Internet-enabled devices generating requests to diverse IoT applications to support services such as smart health, intelligent transportation, and environmental monitoring. We assume a multi-access edge cloud where IoT applications are hosted on edge servers, fog nodes or cloudlets, that are accessible through a set of WiFi APs (or cellular BSs). We also assume a backbone network infrastructure available to interconnect the APs to each other. Hence, some APs may not need to host edge servers or may host an edge server but not support a particular IoT application. This enables a graceful, rather than ubiquitous, cost-effective deployment of edge servers. We further assume that a wide range of devices spatially distributed throughout the smart environment generate workload by requesting given IoT services provided by corresponding application types hosted on selected edge servers. This can lead to scenarios where a given device can access applications on edge servers co-located with APs other than its serving AP. Therefore,

the various edge servers hosting IoT applications should be accessible by devices from various locations via the backbone network infrastructure. Formally, the network is abstracted as a graph $G(N, E)$, where N is the set of nodes; $N = L \cup R$ is composed of a set of APs (or BSs) dispersed at various accessible locations ($l \in L$) in the network, and R represents the set of backbone network equipment (e.g., routers, switches, etc.). All nodes in the network are interconnected with a set E of communication links. Each IoT device connects to the closest AP, and can utilize services provided by IoT applications deployed at either an edge server attached to its serving AP or an edge server attached to another AP that is accessible through the network backbone. Let M be the set of edge servers, each with a computing capacity c_m. An edge server m once deployed at AP l will incur a location-dependent deployment cost π_m^l. Let A be the set of all IoT applications; an application $a \in A$ will be hosted on a VM running on one of the edge servers. Each application provides certain functions for a particular IoT service and, hence, is subject to workload generated from IoT devices that are subscribed to this particular service; for example, a smart camera may request its video streams to be processed by an application providing rendering or surveillance functionality. Applications are classified into different types T (i.e., video processing, face recognition, etc.) and, thus, we define $\mu_a^t \in \{0, 1\}$ to denote that an application a is of type t. In addition, each application requires minimum computing resources p_{min}^a to run and efficiently handle the computation of the minimum load of the IoT devices accessing it [51]. In order to avoid assigning all the computing resources of an edge server to one IoT application, we denote by p_{max}^a the maximum computing resources that can be assigned to an IoT application $a \in A$. More precisely, p_{min}^a and p_{max}^a are the minimum and maximum computing resources that can be assigned to the VM hosting the IoT application a. Further, given that some services require ultra-low latency (e.g., tactile Internet or industrial control), each application of type $t \in T$ is subject to a maximum allowable response time δ_t to ensure satisfactory QoS. The aggregate workload requesting the service provided by applications of type t generated from the devices located within the coverage area of AP deployed at location l is assumed to follow a Poisson process, with an arrival rate λ_l^t (requests/sec), where each request has a given average computing size w_t (e.g., CPU cycles needed to complete the task). This workload can be processed by any hosted application of the requested type t. Further, a given application may receive workload requests from different locations in the network. The higher the intensity of

the offloaded workload on a particular application, the more computing resources this application should be provided in order to keep the processing and queuing delays of the corresponding tasks low. Thus, we denote by p_a (measured in cycles/second), the computing capacity assigned to the IoT application a through the VM hosting it. Further, given that an application hosted on a VM running on an edge server could be shared by multiple loads coming from different locations $l \in L$ following a Poisson process, and the load computation time is exponentially distributed, each application is modeled as an M/M/1 queue [52], with an aggregate arrival rate of IoT requests and a service rate based on the processing capacity p_a of each application. Finally, as mentioned earlier, a request can be either processed by its home edge server (the edge server that is co-located with its serving AP l) or processed by a different edge server at AP deployed at location l'. Therefore, we denote by $h_l^{l'}$ the network delay representing the delay incurred by routing the workload request from the serving AP at location l to the assigned edge server at location l'.

2.3.2 Problem Description

Providing efficient IoT devices workload offloading and processing requires upgrading existing networks to become MEC-enabled. Thus, based on the proposed system model (Section 2.3.1) and while accounting for stringent latency requirement of the offloaded workload, we study and explore three interleaving challenges intrinsic for a successful cost-efficient dimensioning of current networks to become MEC-enabled. Hence, we exploit the following sub-problems:

1) *The MEC dimensioning sub-problem* which consists of deciding on the number and the placement of edge servers. Such decision is affected by the workload to be offloaded from the different existing locations and their required response times. Further, the computing capacities of the edge servers to deploy at different locations may also be variable and subject to the workload which it will be processing. Hence, as edge servers of variable computing capacities may be available for deployment, the MEC dimensioning sub-problem seeks at choosing those with enough computing capacity to deploy at specific locations. Such decision has a direct impact on the deployment cost which increases with the edge servers computing resources.

2) *The IoT applications placement sub-problem* which aims at determining the number and the placement of different types of applications' instances to deploy on the edge servers. Such decision

is coupled to the size of the workload requesting a specific application type from each location and its latency requirement. For instance, as each application instance can only be deployed on exactly one edge server, the choice of the edge server that will host it will affect the response time of the workload that it will be processing. Such response time includes the network delay (e.g., if the application was deployed on an edge server collocated to an AP/BS other than its serving AP/BS), in addition to the processing and queuing delays. These latter are respectively dependent on the processing capacity assigned to the application and the amount of workloads assigned to this specific application. Thus, the IoT applications placement sub-problem is also required to decide on the processing capacity p_a to assign to the application instance to deploy. Note that, the choice of p_a limits the choice of edge servers that can host the application instance, as some of them may not have enough computing resources left to host it. In fact, the computing capacity of edge servers should be optimally allocated to different types of applications in order to successfully meet the delay requirements of the assigned workloads. Further, the processing resources (p_a) assigned to the application instances to deploy impacts the deployment cost as more edge servers may be needed to accommodate those applications.

3) *The workload assignment sub-problem* which consists of assigning the workload generated by IoT devices to the required type of application instance hosted on a suitable edge server, able to meet its response time. It is important to note that assigning the workloads to the closest edge server may not yield an optimal or feasible solution. In fact, the closest edge server collocated to the serving AP/BS may not be hosting the required type of application. Alternatively, the processing capacity assigned to the required application hosted on the edge server may not be enough to serve the workload with respect to its latency requirement. Hence, as we note the inter-dependencies between the three aforementioned sub-problems, we define a joint problem entitled *MEC Resource Provisioning and Workload Assignment for IoT services (RPWA)* which we mathematically formulate next.

2.4 RPWA - A MIXED INTEGER PROGRAM (RPWA-MIP)

2.4.1 Problem Definition

Definition 1. *Given $G(N, E)$, a set M of edge servers, a set A of IoT applications of different types, and a set of IoT devices requesting their workloads to be offloaded and processed by a specific application type within a determined response time δ_t, determine the lowest cost deployment of edge servers and IoT applications in $G(N, E)$, the processing capacity to allocate to the deployed applications, in addition to the assignment of workloads to these latter with respect to their latency requirements.*

2.4.2 Problem Formulation

Network Inputs	
$G(N, E)$	Network of N nodes where $N = L \cup R$ and E links connecting them.
L	Set of locations mounted with APs/BSs.
R	Set of backbone network equipment.
M	Set of edge servers to be deployed in $G(N, E)$.
A	Set of IoT applications to be hosted in $m \in M$.
T	Set of applications' types.
$c_m \in \mathbb{R}^+$	Processing capacity of edge server $m \in M$.
$\pi_l \in \mathbb{Z}^+$	Setup cost of an edge server at location $l \in L$.
$k \in \mathbb{Z}^+$	Cost of a unit of processing capacity.
$\mu_a^t \in \{0, 1\}$	Parameter which depicts that application $a \in A$ is of type $t \in T$ (1) or not (0).
$\delta_t \in \mathbb{R}^+$	Maximum allowable response time when utilizing an application of type $t \in T$.
$p_{min}^a \in \mathbb{R}^+$	Minimum processing capacity required by application $a \in A$.
$p_{max}^a \in \mathbb{R}^+$	Maximum processing capacity that can be assigned to application $a \in A$.
$\lambda_l^t \in \mathbb{Z}^+$	Arrival rate of requests for an application of type $t \in T$ generated by IoT devices associated to AP/BS at location $l \in L$.
$w_t \in \mathbb{Z}^+$	Average number of CPU cycles per request for an application of type t .
$h_l^{l'} \in \mathbb{R}^+$	Network delay of a request from its home edge server at $l \in L$ to its assigned edge server at $l' \in L$.

Table 2.1: Parameters of RPWA-MIP.

Table 2.1 delineates the parameters used throughout the formulation of RPWA-MIP presented

below. We define a variable x_m^l to determine whether edge server $m \in M$ is deployed at location $l \in L$.

$$x_m^l = \begin{cases} 1 \text{ if edge server } m \in M \text{ is deployed at location } l \in L, \\ 0 \text{ otherwise.} \end{cases}$$

Our objective is to minimize the edge servers deployment cost while meeting the delay requirements of the workloads requesting to be processed by a specific type of IoT applications:

$$Minimize \sum_{l \in L} \sum_{m \in M} x_m^l (\pi_l + k c_m) \tag{2.1}$$

In order to meet our objective, several constraints, that we elucidate in the following, have to be respected. Let y_a^{lm} be a decision variable that determines whether application $a \in A$ is placed on edge server $m \in M$ deployed at location $l \in L$.

$$y_a^{lm} = \begin{cases} 1 \text{ if IoT application } a \text{ is placed on } m \text{ at location } l, \\ 0 \text{ otherwise.} \end{cases}$$

In order to simplify some of the constraints, we define σ_a^l to determine whether application a is placed at location l.

$$\sigma_a^l = \begin{cases} 1 \text{ if IoT application } a \text{ is placed at location } l, \\ 0 \text{ otherwise.} \end{cases}$$

z_{lt}^a is another decision variable that indicates whether the workload generated by IoT devices at location $l \in L$ demanding an application of type $t \in T$ are mapped to application $a \in A$.

$$z_{lt}^a = \begin{cases} 1 \text{ if workload generated by devices at location } l \text{ and} \\ \text{demanding an application of type } t \text{ is mapped to } a, \\ 0 \text{ otherwise.} \end{cases}$$

Further, let $p_a \in \mathbb{R}^+$ specifies the amount of computing resources assigned to application $a \in A$.

Hence, the RPWA problem constraints can be classified as follows:

Placement of edge servers

First, we need to guarantee that each edge server $m \in M$ is deployed on at most one location $l \in L$ (Eq.(2.2)), that is: $\sum_{l \in L} x_m^l = 1$ if edge server m is deployed at location l, and 0 otherwise. Further, Eq.(2.3) guarantees that each location can host at most one edge server; hence, $\sum_{m \in M} x_m^l = 1$ if location l is hosting one edge server, and 0 otherwise.

$$\sum_{l \in L} x_m^l \leq 1 \quad \forall m \in M \tag{2.2}$$

$$\sum_{m \in M} x_m^l \leq 1 \quad \forall l \in L \tag{2.3}$$

Placement of applications on edge servers

Once the edge servers are deployed, we provision IoT applications and assign computing resources to them. Eq.(2.4) ensures that each application $a \in A$ is placed on at most one edge server $m \in M$. Thus, if application a is placed on an edge server m, then $\sum_{l \in L} \sum_{m \in M} y_a^{lm} = 1$, and 0 otherwise. Further, each application's computing resource is guaranteed a minimum value p_{min}^a (Eq.(2.5)), and limited to a maximum value p_{max}^a (Eq.(2.6)).

$$\sum_{l \in L} \sum_{m \in M} y_a^{lm} \leq 1 \quad \forall a \in A \tag{2.4}$$

$$p_a \geq \sum_{m \in M} \sum_{l \in L} y_a^{lm} p_{min}^a \quad \forall a \in A \tag{2.5}$$

$$p_a \leq \sum_{m \in M} \sum_{l \in L} y_a^{lm} p_{max}^a \quad \forall a \in A \tag{2.6}$$

Constraint (2.7) guarantees that all IoT applications provisioned on an edge server $m \in M$ do not exceed the capacity c_m of m. Constraint (2.8) guarantees that each application $a \in A$ can only be placed on a deployed edge server $m \in M$.

$$\sum_{l \in L} \sum_{a \in A} p_a y_a^{lm} \leq c_m \quad \forall m \in M \tag{2.7}$$

$$y_a^{lm} \leq x_m^l \quad \begin{matrix} \forall l \in L \\ \forall m \in M \\ \forall a \in A \end{matrix} \tag{2.8}$$

Further, (2.9) ensures that if $y_a^{lm} = 1$ then $\sigma_a^l = 1$.

$$\sigma_a^l = \sum_m y_a^{lm} \quad \begin{matrix} \forall l \in L \\ \forall a \in A \end{matrix} \tag{2.9}$$

Workload assignment

Once applications are placed, the workload generated by devices associated to an AP/BS at location $l \in L$ is assigned to an IoT application of the requested type $t \in T$. Thus, constraint (2.10) ensures that z_{lt}^a gets a value, for some a, when $\lambda_l^t > 0$; H is a big integer number and $\lambda_l^t << H$.

$$\sum_{a \in A} z_{lt}^a \geq \frac{\lambda_l^t}{H} \quad \begin{matrix} \forall l \in L \\ \forall t \in T \end{matrix} \tag{2.10}$$

Constraint (2.11) guarantees that the generated load from location l requesting application of type t is mapped to at most one IoT application a:

$$\sum_{a \in A} z_{lt}^a \leq 1 \quad \begin{matrix} \forall l \in L \\ \forall t \in T \end{matrix} \tag{2.11}$$

In addition, we need to make sure that each load is mapped to an application $a \in A$ providing the same requested type t:

$$z_{lt}^a \leq \mu_a^t \lambda_l^t \quad \begin{matrix} \forall t \in T \\ \forall l \in L \\ \forall a \in A \end{matrix} \tag{2.12}$$

Note that Eq.(2.12) guarantees that $z_{lt}^a = 0$ if there are no requests for an application of type t coming from location l, i.e., $\lambda_l^t = 0$. Constraint (2.13) ensures that requests are only mapped to applications that are deployed on an edge server $m \in M$ placed at some location $l \in L$ and constraint (2.14) prevents hosting an application a on an edge server m if it is not processing any load.

$$z_{lt}^a \leq \sum_{m \in M} \sum_{l' \in L} y_a^{l'm} \quad \begin{matrix} \forall l \in L \\ \forall t \in T \\ \forall a \in A \end{matrix} \tag{2.13}$$

$$\sum_{m \in M} \sum_{l \in L} y_a^{lm} \leq \sum_{l \in L} \sum_{t \in T} z_{lt}^a \quad \forall a \in A \tag{2.14}$$

Delay Constraints

Offloaded IoT traffic experiences delays consisting of access delays from IoT device to the serving AP (or BS), network delays if traffic is to be routed from the serving AP to the AP where the edge server is hosted, and finally server delays incurred at the edge server of the receiving application and that constitutes queuing and processing delays. We assume negligible access delays and focus on the network delays and server delays. To guarantee delay requirements for each IoT service provided by applications of type t, we constrain in Eq.(2.15) the total delay experienced by the traffic to a given target response time δ_t.

$$2d_n^{l,t} + d_s^{l,t} \leq \delta_t \quad \begin{matrix} \forall l \in L \\ \forall t \in T \end{matrix} \tag{2.15}$$

where $d_n^{l,t}$ represents the network delays experienced by a workload generated from location l requesting an application of type t and assigned to an IoT application hosted in a remote location, and is given by:

$$d_n^{l,t} = \sum_{a \in A} \sum_{l' \neq l} h_l^{l'} z_{lt}^a \sigma_a^{l'} \quad \begin{matrix} \forall l \in L \\ \forall t \in T \end{matrix} \tag{2.16}$$

$d_s^{l,t}$ depicts the server delay that constitutes the queuing and processing delays that the workload generated from location l experiences at the edge server of the receiving application of type t. As mentioned earlier, given the workload assigned from various locations l' to each application, we model an application a as an M/M/1 queuing system with aggregate request arrival rate of $\sum_{l' \in L} z_{l't}^a \lambda_{l'}^t$ and service rate of $\frac{p_a}{w_t}$, where p_a is the processing capacity in cycles per second assigned to application a and w_t is the average request size in cycles. Therefore,

$$d_s^{l,t} = \sum_{a \in A} z_{lt}^a \left(\frac{1}{\frac{p_a}{w_t} - \sum_{l' \in L} z_{l't}^a \lambda_{l'}^t} \right) \quad \begin{matrix} \forall l \in L \\ \forall t \in T \end{matrix} \tag{2.17}$$

Eq. (2.15) after some manipulation can be rewritten as:

$$\begin{aligned} \sum_{a \in A} z_{lt}^a \frac{p_a}{w_t} - \sum_{a \in A} \sum_{l' \in L} z_{lt}^a z_{l't}^a \lambda_{l'}^t \geq \\ \frac{1}{\delta_t - 2 \sum_{a \in A} \sum_{l' \neq l} h_l^{l'} z_{lt}^a \sigma_a^{l'}} \quad \begin{matrix} \forall l \in L \\ \forall t \in T \end{matrix} \end{aligned} \tag{2.18}$$

To maintain a stable queue at the application, we force the service rate to be greater than the arrival rate as per below:

$$\frac{p_a}{w_t} - \sum_{l' \in L} z^a_{l't} \lambda^t_{l'} \geq 0 \quad \substack{\forall a \in A \\ \forall t \in T} \tag{2.19}$$

The above model is a mixed integer non-linear program. Appendix A.1 presents linearization of Eq.(2.7) and Eq.(2.18).

2.4.3 Complexity Analysis

The RPWA-MIP formulation is complex and the model is clearly hard to solve even for a small network (Section 2.6). In fact, the complexity of the RPWA problem can be highlighted through the complexity of the different sub-problems it solves. For instance, the MEC dimensioning and the IoT applications placement sub-problems combined can be proven as NP-Hard via a reduction from the capacitated facility location problem [53] (known to be NP-Hard) where the facilities are the edge servers to be deployed at locations $l \in L$ and the customers are the applications to place on the deployed edge servers. Further, the NP-Hard generalized assignment problem [54] can be reduced to the workload assignment sub-problem where the workloads constitute the items that need to be assigned to bins representing the IoT applications. Hence, the workload assignment sub-problem is also NP-Hard. Thus, the RPWA problem is NP-Hard as it combines three NP-Hard sub-problems. Given its complexity, we present in the following an efficient decomposition approach to solve it.

2.5 RPWA-D: A Decomposition Approach

As addressing the MEC dimensioning, the IoT applications placement and the workload assignment sub-problems jointly is challenging, we investigate the inter-dependency tightening these three sub-problems together in the aim of exploring a more efficient approach to solve them. Thus, we first notice that the workload assignment sub-problem highly couples the MEC dimensioning sub-problem and the IoT applications placement sub-problem which makes it difficult to address each of these three sub-problems independently. By investigating the workload assignment sub-problem, we observe that the placement of the applications on edge servers has a direct impact on

the network delay experienced by the workload assigned to the application. Further, the decision on the computing resources to allocate to each application is dependent on the size of the workload it will be processing and affects the server delay of this latter. In fact, the response time (δ_t) of the IoT service requested by the offloaded workloads is to be met as per Eq.(2.15). Hence, we re-evaluate the response time constraint in Eq.(2.15) given that it is a critical constraint linking the three aforementioned sub-problems. Eq.(2.15) is rewritten as follows:

$$d_s^{l,t} \leq \delta_t - 2d_n^{l,t} \quad \substack{\forall l \in L \\ \forall t \in T} \tag{2.20}$$

By substituting $d_s^{l,t}$ given in Eq.(2.17) into Eq.(2.20), we obtain:

$$\sum_{a \in A} z_{lt}^a \left(\frac{1}{\frac{p_a}{w_t} - \sum_{l' \in L} z_{l't}^a \lambda_{l'}^t} \right) \leq \delta_t - 2d_n^{l,t} \quad \substack{\forall l \in L \\ \forall t \in T} \tag{2.21}$$

From (2.21), we notice that if $d_n^{l,t}$ is known, it will be easy to decouple the workload assignment sub-problem from the IoT applications placement and MEC dimensioning sub-problems as the placement of the Iot applications and edge servers will no longer affect the response time requirement (δ_t). To resolve this, we assume a maximum network delay experienced by the load ($d_n^{max} = \max\ (h_l^{l'}),\ \forall l,l' \in L$), to guarantee that any obtained solution will meet the response time requirement. Although this assumption simplifies the decomposition of our problem, it may result in over-provisioning the edge servers as it assumes that some loads incur higher network delays from what they actually experience. Eq.(2.21) can then be rewritten as follows:

$$\sum_{a \in A} z_{lt}^a \left(\frac{1}{\frac{p_a}{w_t} - \sum_{l' \in L} z_{l't}^a \lambda_{l'}^t} \right) \leq \delta_t' \quad \substack{\forall l \in L \\ \forall t \in T} \tag{2.22}$$

where $\delta_t' = \delta_t - 2d_n^{max}$ and represents the maximum server delay allowed to meet the response time requirement. Eq.(2.22) shows that given the maximum network delay, one can decide on the computing resources p_a to assign to each application a as well as the workload mapped to this application. Determining the number of needed applications of the same type and the computing resources (p_a) to assign to each of those applications based on a *maximum* server delay (δ_t'), which may be experienced by all workloads requesting the same application type, simplifies the IoT applications

placement sub-problem to simply respecting the edge servers capacities. In fact, the applications and the edge servers can then be placed at any location without incurring any violation to the response time requirement. Hence, since applications placement on edge servers is dependent on the computing capacities of these edge servers and their number, consequently, the total deployment cost is affected. Thus, the MEC dimensioning sub-problem can take care of the IoT applications placement, if fed with those to be deployed under the objective of minimizing the deployment cost. The workload assignment sub-problem can, hence, take care of deciding on the number and the computing resources to be assigned to the applications to deploy. Thus, the decisions initially taken by the IoT applications placement sub-problem are divided between the two other sub-problems (i.e., MEC dimensioning and workload assignment sub-problems). Therefore, in the following, we decouple the RPWA problem and decompose it into two different sub-problems:

1) The Delay Aware Load Assignment (DALA) sub-problem that decides on the workload assignment to the IoT applications while determining the number of needed applications and their computing resources with respect to the response time requirement.

2) The Mobile Edge Servers Dimensioning (MESD) sub-problem which determines the placement of the applications and edge servers.

2.5.1 The Delay Aware Load Assignment (DALA)

Definition 2. *Let $G(N, E)$ denotes the network, R_t represents the set of all workloads demanding the service of an IoT application of type t (each $r \in R_t$ denotes a load with rate λ_r requests/sec for application of type t). Let A_t depicts the set of all IoT applications of type t, maximize the fraction of workloads that can be admitted to the network while determining the number of applications to be deployed in $G(N, E)$ and the assignment of the admitted workloads to those applications with respect to the response time requirement.*

We delineate in Table 2.2 the parameters used throughout our formulation. The remaining ones are as defined in Table 2.1. As the main objective of the RPWA problem is to minimize the deployment cost of edge servers, a possible objective for the DALA sub-problem is to minimize the sum of computing resources assigned to the applications to deploy, as it will reduce the number of edge servers needed to host those applications and hence, minimize the deployment cost. While

Network Inputs		
A_t		Set of applications of type t.
R_t		Set of all workloads demanding the service of an application of type t.
λ_r	$\in \mathbb{Z}^+$	Arrival rate of requests of workload $r \in R_t$.
w_t	$\in \mathbb{Z}^+$	Average number of computing cycles demanded by one request $r \in R_t$.
$\delta'_t \in \mathbb{R}^+$		Maximum tolerated server delay by an application of type t.

Table 2.2: Parameters of the DALA-MIP.

this is a possible objective and comes aligned with the definition of the RPWA problem, it requires admitting all the workloads; this might lead to infeasibility for the same instances which are can be solved by the RPWA-MIP. This is because DALA considers the maximum network delay (d_n^{max}) when searching for a solution, which tightens up the server delay (Eq.2.22) and hence, the solution space. Thus, to obtain a solution for the same instances of RPWA, we define DALA under the objective of maximizing the fraction of the admitted workloads. Hence, we define $\alpha_r \in [0 - 1]$ to depict the fraction of the load that can be admitted to the network and we depict the objective of DALA in Eq.(2.23).

$$\text{Maximize} \quad \sum_{r \in R_t} \lambda_r \alpha_r \tag{2.23}$$

This objective is subject to several constraints, as elaborated in the sequel. We introduce the decision variable $\rho^a \in \{0, 1\}$ to depict whether an application $a \in A_t$ is assigned at least one workload to process.

$$\rho^a = \begin{cases} 1 \text{ if application } a \text{ is used,} \\ 0 \text{ otherwise.} \end{cases}$$

We define a new variable $z_r^a \in \{0, 1\}$ to specify whether a workload $r \in R_t$ is mapped to application $a \in A_t$ as follows:

$$z_r^a = \begin{cases} 1 \text{ if workload } r \text{ is mapped to application } a, \\ 0 \text{ otherwise.} \end{cases}$$

In addition, we define $p_a \in \mathbb{R}^+$ to depict the computing resources to be allocated to application $a \in A_t$. An application is used and should be deployed in the network if at least one workload is mapped to it as specified in Eq.(2.24).

$$\rho^a \geq z_r^a \quad \begin{matrix} \forall a \in A_t \\ \forall r \in R_t \end{matrix} \tag{2.24}$$

A workload $r \in R_t$ can be mapped to and processed by exactly one application as determined by Eq.(2.25).

$$\sum_{a \in A_t} z_r^a \leq 1 \quad \forall r \in R_t \tag{2.25}$$

A workload $r \in R_t$ should be processed by an application $a \in A_t$ that has enough computing resources to process it without violating the server delay requirement as depicted in Eq.(2.26).

$$d_s^r \leq \delta_t' \quad \forall r \in R_t \tag{2.26}$$

where d_s^r represents the server delay (processing and queuing delays) and is determined by (2.27).

$$d_s^r = \sum_{a \in A_t} z_r^a \left(\frac{1}{\frac{p_a}{w} - \sum_{r' \in R_t} z_{r'}^a \lambda_{r'} \alpha_{r'}} \right) \quad \forall r \in R_t \tag{2.27}$$

Further, the queue of each application should remain stable. That is, the average service rate of the IoT application $a \in A_t$ should be larger than the aggregate average arrival rates of all workloads mapped to application a as given in Eq.(2.28).

$$\frac{p_a}{w} - \sum_{r' \in R_t} z_{r'}^a \lambda_{r'} \alpha_{r'} > 0 \quad \forall a \in A_t \tag{2.28}$$

Finally, the computing resources assigned to an application should be at least equal to its minimum required computing capacity as depicted in Eq.(2.29) and at most equal to $p_m ax^a$ as specified in Eq.(2.30)

$$p_a \geq p_{min}^a \rho^a \quad \forall a \in A_t \tag{2.29}$$

$$p_a \leq p_{max}^a \rho^a \quad \forall a \in A_t \tag{2.30}$$

Eq.(2.26) and Eq.(2.28) are non linear and can be easily linearized (Appendix A.2). We note that DALA is a MIP that can run in multiple threads where each thread finds the solution for one application type.

2.5.2 The Mobile Edge Servers Dimensioning (MESD)

Definition 3. *Given a network $G(N, E)$, a set A of IoT applications and a set M of edge servers, determine the number and the placement of edge servers in $G(N, E)$, in addition to the placement of the applications on the deployed edge servers such that the total deployment cost is minimized.*

Network Inputs	
$\bar{A}$	Set of applications to deploy.
$p_a \in \mathbb{R}^+$	Computing resources of application $a \in A$.

Table 2.3: Parameters of the MESD-IP.

Table 2.3 presents the parameters used. The remaining parameters are as defined in Table 2.1. In order to formulate the MESD sub-problem, we define the decision variable $x_m^l \in \{0, 1\}$ to specify whether the edge server $m \in M$ is deployed at location $l \in L$ as follows:

$$x_m^l = \begin{cases} 1 \text{ if edge server } m \text{ is deployed at location } l, \\ 0 \text{ otherwise.} \end{cases}$$

In addition, we introduce a decision variable $y_a^m \in \{0, 1\}$ to indicate whether application $a \in \bar{A}$ is placed on edge server $m \in M$ as follows:

$$y_a^m = \begin{cases} 1 \text{ if application } \ a \text{ is placed on edge server } m, \\ 0 \text{ otherwise.} \end{cases}$$

The objective of the MESD sub-problem is to minimize the deployment cost of the edge servers (Eq.(2.31)).

$$Minimize \sum_{l \in L} \sum_{m \in M} x_m^l (\pi_l + k c_m) \tag{2.31}$$

Subject to the following constraints. First, an edge server can be placed on at most one location Eq.(2.32).

$$\sum_{l \in L} x_m^l \leq 1 \ \ \forall m \in M \tag{2.32}$$

Similarly, a location $l \in L$ can host at most one edge server $m \in M$ as depicted in Eq.(2.33).

$$\sum_{m \in M} x_m^l \leq 1 \ \ \forall l \in L \tag{2.33}$$

In addition, an application can be deployed on exactly one edge server as specified in Eq.(2.34).

$$\sum_{m \in M} y_a^m = 1 \ \ \forall a \in \bar{A} \tag{2.34}$$

Finally, Eq.(2.35) guarantees that the edge servers capacities are not violated and that the applications are only hosted on placed edge servers.

$$\sum_{a \in \bar{A}} y_a^m p_a \leq c_m \sum_{l \in L} x_m^l \ \ \forall m \in M \tag{2.35}$$

2.5.3 Decomposition Algorithm

We design and implement the RPWA decomposition solution (RPWA-D) to solve the RPWA problem. Our proposed decomposition approach captures the collaboration between the DALA and the MESD sub-problems. Thus, we depict in Fig.2.2 a flowchart detailing the steps of RPWA-D. In fact, RPWA-D accounts for the independency that exists between the applications types. For instance, the assignment of the workloads requesting the service provided by a specific type of applications and the computing resources to be assigned to those applications have no impact on the same decisions made for any other type of applications. Hence, several instances of the DALA-MIP are executed as parallel threads in order to provide a workloads assignment and application computing resources determination for each type of applications. Thus, RPWA-D pre-processes the simulation data by categorizing it by the type of application it is requesting. It then instantiate a thread to execute the DALA-MIP for each application type. Once the execution of all the DALA-MIP threads is finalized, their solutions are processed to capture the application instances that need

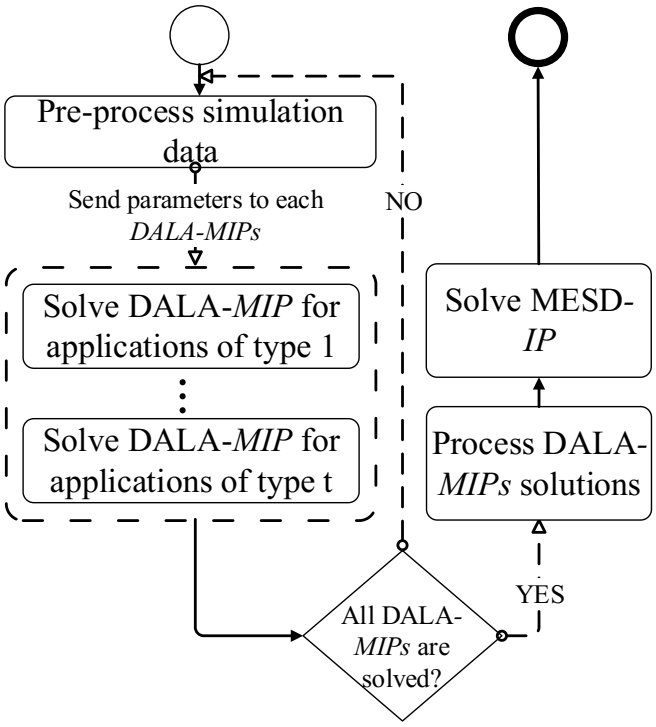


Figure 2.2: Flowchart of RPWA-D.

to be deployed on edge servers (i.e., the application instances that were set as used by the DALA-MIP threads ($\rho_a = 1$)), in addition to their assigned computing resources (p_a). Those latter are supplied to MESD-IP as parameters. The MESD-IP is then executed in order to decide on the number and the location of edge servers to place, in addition to the placement of the applications which need to be deployed. Thus, with the help of the DALA-MIP, the MESD-IP provides the minimum edge servers deployment cost.

2.6 Numerical Evaluation

We conduct an extensive empirical study to evaluate the performance of the RPWA-MIP against our decomposition approach (RPWA-D). We also explore the efficiency of RPWA-D under varying parameters and study the trade-off between the edge servers deployment cost and the percentage of the workloads that can be admitted. Our numerical evaluation is conducted using IBM ILOG CPLEX Optimization Studio version 12.8.

2.6.1 Evaluation setup

In our tests, we consider networks of different sizes where we vary the number of locations hosting APs/BSs (L) and the number of edge servers that can be collocated to the existing APs/BSs (M). However, we consider that each edge server $m \in M$ has a computing capacity $c_m = 6GHz$ (unless stated otherwise) and a deployment cost normalized to 8 units. In addition, we perform our tests under different number of available IoT applications (variable A) of 4 different types ($T = 4$) that belong to the same industry vertical and hence, may require the same QoS (i.e., response time). Thus, we delineate in Table 2.4 the different industry verticals accounted for in our tests, and present the range of their required response times in addition to the one used to set the value of δ_t. Note that, we consider that each IoT application $a \in A$ requires a minimum processing capacity of $p^a_{min} = 1.7GHz$ and can consume up to $p^a_{max} = 1.9GHz$ of computing resources. We account for IoT devices scattered at different locations within the considered network. Each of these IoT devices is connected to an AP/BS available at its location and is requesting the service of a determined type of application. We evaluate variable arrival rates (variable λ^t_l) of the IoT devices requests per second for different application types spread at different locations of the network. However, we assume an average of $w_t = 2 \times 10^6$ CPU cycles/request for a defined application type $t \in T$.

Industry Vertical	**Allowable response times (ms)**	**Applied response time (δ_t) (ms)**
Tactile Internet	1 - 10	5
Factory Automation	0.25 - 10	10
Smart Grids	3 - 20	20
Intelligent transportation Systems (ITS)	10 - 100	50
Tele Surgery	≤ 250	110

Table 2.4: QoS of different industry verticals [1, 2].

2.6.2 RPWA-MIP vs. RPWA-D

We first study the performance of RPWA-D and compare it against RPWA-MIP in terms of optimality (e.g., deployment cost) and scalability (e.g., execution time) as we vary the number of locations in the network. In fact, increasing the number of locations depicts an increase in the workloads as more IoT devices requests originating from the added locations are to be accounted for.

Hence, we consider a maximum of $M = 10$ edge servers and $A = 4$ applications of $T = 4$ different types belonging to a factory automation vertical ($\delta_t = 10ms$) that can be deployed in the network. In addition, we assume $\lambda_l^t = 60$ requests/sec originating from each location $l \in L$ and requesting the service of an application of type $t \in T$. The maximum network delay is set to $4ms$. Our results are presented in Table 2.5.

Instance $< L, M, T, A >$	**RPWA-MIP**		**RPWA-D**		
	Cost (units)	*Execution Time (ms)*	*Cost (units)*	*Execution Time (ms)*	*Load admitted (%)*
L=5, M=10, T=4, A=4	16	69	16	57	100
L=7, M=10, T=4, A=4	16	90	16	60	100
L=15, M=10, T=4, A=4	-	-	16	90	50
L=15, M=10, T=4, A=12	-	2days+	32	661	100

Table 2.5: Evaluation of RPWA-MIP vs. RPWA-D.

1) Optimality Gap: Table 2.5 depicts that for a small number of locations ($L = 5$ and $L = 7$), RPWA-D is able to provide the optimal cost of 16 units supplied by RPWA-MIP while admitting all the workloads. Note that, this cost remained constant for $L = 5$ and $L = 7$ given that the deployed edge servers when $L = 5$ had enough free computing resources to accommodate the additional applications needed to handle the extra workloads generated from the two added locations (when $L = 7$). In other words, no extra edge servers were needed to be deployed. As we increase the number of locations to $L = 15$, RPWA-MIP fails to give a feasible solution as it is constrained by admitting all the generated workloads and the fixed number of applications ($A = 4$) that can be deployed. However, RPWA-D admitted 50% of the workload with a cost of 16 units. As $A = 4$ applications were not sufficient for RPWA-MIP to provide a solution, we performed a final test where we increased the number of applications from $A = 4$ to $A = 12$ for the same number of locations (e.g., $L = 15$). However, this resulted in a large increase of the size of the problem and RPWA-MIP failed to give a solution even after 2 days of execution. However, RPWA-D was able to admit all the loads.

2) Execution Time: In order to evaluate the scalability of the proposed methods, we delineates in Table 2.5 the execution time of RPWA-MIP and RPWA-D. As the size of the problem increases with the increase of the number of locations and the number of applications, finding a solution for the problem becomes more challenging, and hence, the runtime of both methods increases exponentially. However, Table 2.5 clearly shows that the increase of the runtime of RPWA-D is at a slower pace than that of RPWA-MIP and remains in the order of milliseconds (661 ms) while the runtime of RPWA-MIP exceeded the 2 days without providing a solution ($L = 15$ and $A = 12$). This proves that RPWA-D is much more scalable than RPWA-MIP.

2.6.3 Evaluation of RPWA-D

Given the non-scalability of RPWA-MIP, we focus in the following on studying the impact of varying network parameters on the deployment cost using RPWA-D while highlighting several trade-offs existing between the evaluated system parameters. Thus, unless stated otherwise, we consider a network consisting of $L = 10$ different locations, and a maximum of $M = 5$ edge servers that can be deployed in it. The maximum network delay is set to $1.5ms$.

1) Impact of varying the number of IoT applications for different industry verticals: We first investigate the impact of varying the number of available applications on the admitted workloads given IoT services belonging to different industry verticals [42, 43]. Thus, we consider four application types ($T = 4$) for each of the industry verticals depicted in Table 2.4 and represented by their maximum allowable response time ($5ms \leq \delta_t \leq 110ms$). We evaluate the percentage of admitted workload for each of them as we vary the maximum number of applications that can be deployed in the network ($4 \leq A \leq 12$). Fig.2.3 illustrates an increase of the percentage of admitted workload with the increase of the number of applications. For instance, when considering Tactile Internet ($\delta = 5ms$), the admitted workload almost doubled when the number of applications went from $A = 4$ to $A = 8$. Similar observation can be deducted for the other presented types of industry verticals. This infers that, even though enough edge servers computing resources are available, a low number of applications was not able to admit all the workloads even when all the applications are deployed and assigned the maximum processing resources they can acquire (p^a_{max}). This shows that the maximum computing resources allocated to each application is not enough to meet the required

QoS (response time), if assigned more workloads. In fact, an increase of the server delay will be observed with the increase in the workloads assigned to an application. Furthermore, Fig.2.3 depicts that when the response time increases (e.g., considering all the industry verticals), the percentage of admitted workload increases for a fixed number of applications. For instance, when $A = 12$, IoT applications utilizing Tactile Internet ($\delta_t = 5ms$) admitted 57.4% of the generated workloads while those used for factory automation ($\delta_t = 10ms$) were able to admit 94.3%. This trend continues as the response time increases to reach 100% of admission with the applications employed for Tele-Surgery. This can be explained by the fact that when the response time is relaxed, the workloads can tolerate higher queuing and processing delays which allows the applications to be able to meet the QoS requirement for a bigger fraction of their assigned load.

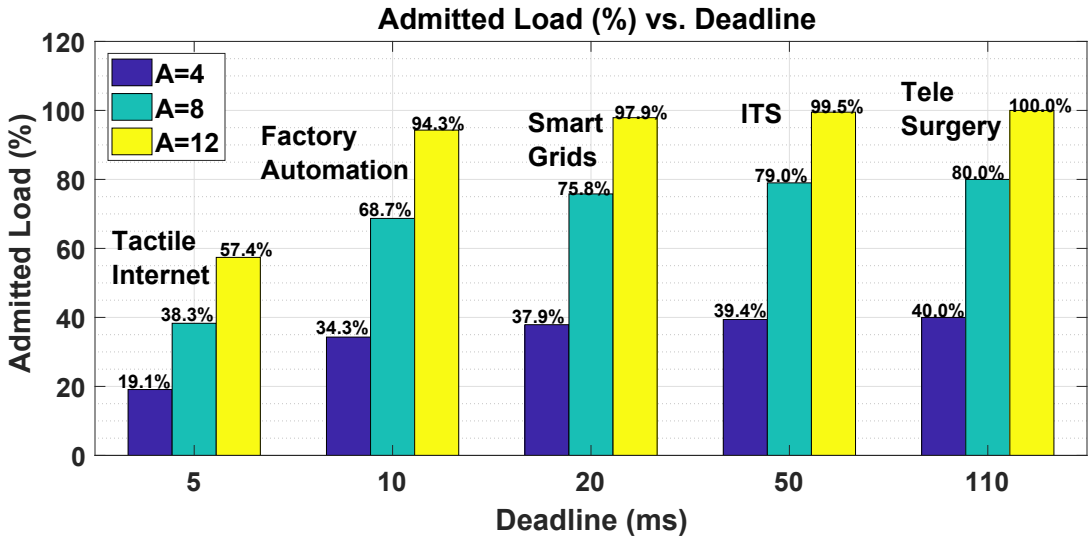


Figure 2.3: Admission rate under varying number of applications.

2) Impact of varying workloads for different industry verticals: We evaluate the impact of the workload increase on the admission rate for different industry verticals. Thus, we consider $A = 12$ applications that can be deployed in the network and we vary the workload by varying the value of 205 requests/sec $\leq \lambda_l^t \leq$ 265 requests/sec. Fig.2.4 depicts that as the generated workloads per location per type increase for a given industry vertical, the percentage of admitted ones decreases, given that more workloads will be contending for the same computing resources (applications), which are not sufficient to serve all of them while meeting the required response time, since they will be

experiencing higher queuing delay. However, this increased queuing delay can be tolerated if the response time increased. More precisely, one can note that as the workload for smart grid applications increased by 22.6% (from 205 requests/sec to 265 requests/sec), the admitted workload decreased by 6.4% showing that when all the computing resources (applications) are being consumed, the network fails to cope with the increased workloads and hence, less load is admitted. Further, when $\lambda_l^t = 265$ requests/sec, the admission rate increases with the increase of the response time to reach 95.5% for less latency sensitive applications such as those used in Tele-surgery.

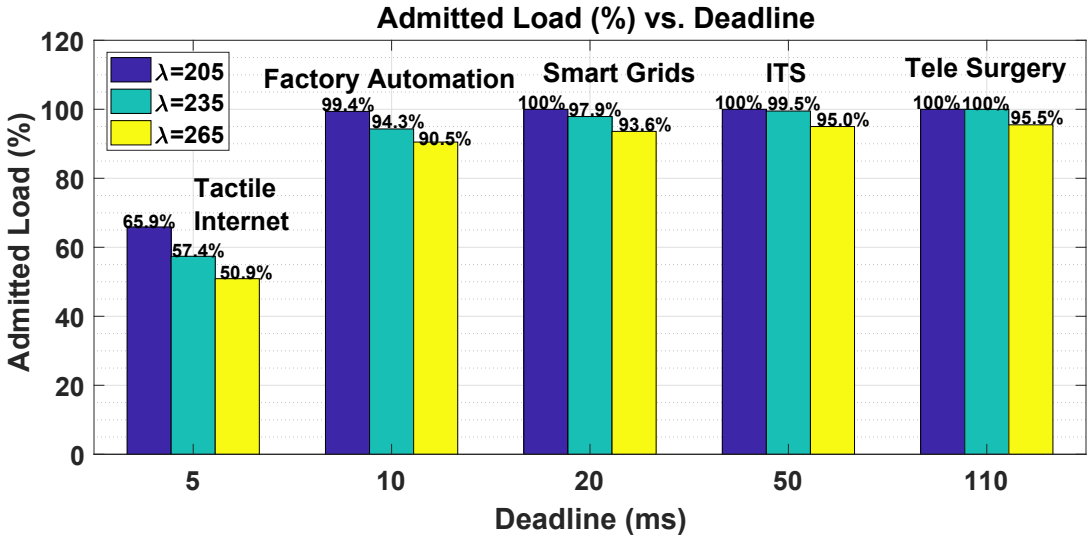


Figure 2.4: Admission rate under varying workloads.

3) Impact of varying the network size: We consider a factory automation industry where the allowed response time of its application is fixed to $\delta_t = 10ms$. We evaluate the trade-off existing between the number of locations and the number of applications in a network where at most $M = 5$ edge servers can be deployed. We account for $\lambda_l^t = 235$ requests/sec generated from each location for each of the $T = 4$ types of available applications.

Fig. 2.5 demonstrates that for the same number of applications, the admission rate decreases with the increase of the number of locations. This is explained by the fact that as the network size increases, more workloads are generated from the additional locations making the existing applications experience more congestion which leads to having their allocated computing resources fall short in serving the additional workloads within the allowed delay. However, as the number of applications increases (for a specific number of locations), more computing resources become available

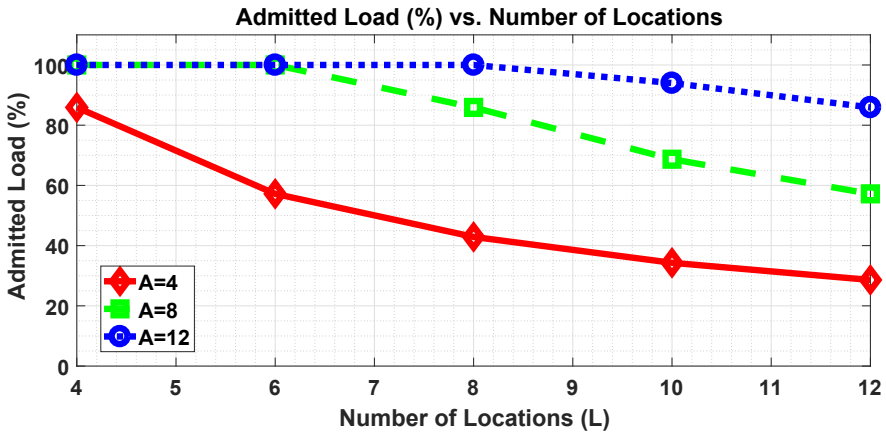


Figure 2.5: Admitted rate over varying number of locations and applications.

to process the additional workloads which explains the increase in the admission rate.

4) Impact of varying the generated workloads and applications: Under the same simulation setup mentioned in the previous test (Section 2.6.3(3)), we provide a large scale test in order to demonstrate the feasibility of the RPWA-D approach. Thus, we increase the workload size for each type of application generated from every location by varying the value of λ_l^t between 150 requests/sec and 550 requests/sec. As we consider $L = 10$ locations and $T = 4$ types of applications, the total workload generated from all locations targeting all application types varied between 6000 requests/sec and 22000 requests/sec. In this test scenario, We vary the number of applications that can be placed on edge servers from $A = 4$ to $A = 12$ to study its impact on the admission rate. Fig. 2.6 illustrates that for the same number of applications, the admission rate decreases with the increase of the generated workload. This can be interpreted that the provisioned processing capacities for a given number of applications cannot accommodate growing workload demand unless more resources are made available. As such, we can note that when the number of applications increases, the admission rate increases for the same amount of workload λ_l^t.

5) Impact of varying the edge servers capacities: We explore the relation between the edge servers capacities and the network utilization for different generated workloads. The network utilization is defined as the ratio between the total computing resources used by the deployed applications $\sum_{a \in A} p_a$ and the total computing capacity available by the deployed edge servers $\sum_{m \in M} c_m$

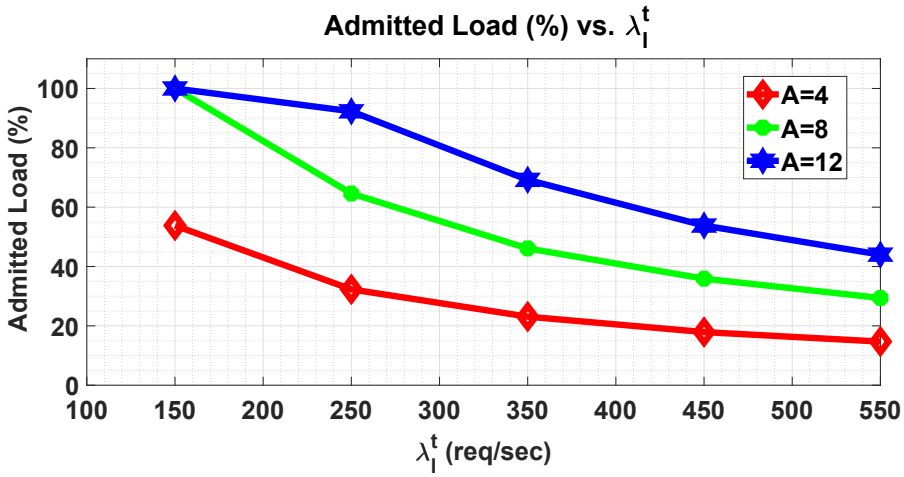


Figure 2.6: Admission rate over varying workload and applications.

$(\frac{\sum_{a\in A} p_a}{\sum_{m\in M} c_m})$. Hence, we consider a factory automation industry ($\delta_t = 10ms$) of $L = 10$ locations, $M = 5$ edge servers that can be deployed to host up to $A = 12$ applications of $T = 4$ different types. Fig.2.7 depicts the percentage of network utilization as we increase the capacities of the edge servers ($4\text{Ghz} \leq c_m \leq 6.8\text{GHz}$) and arrival rates of the requests ($\lambda_l^t \in \{150, 200\}$ requests/sec).

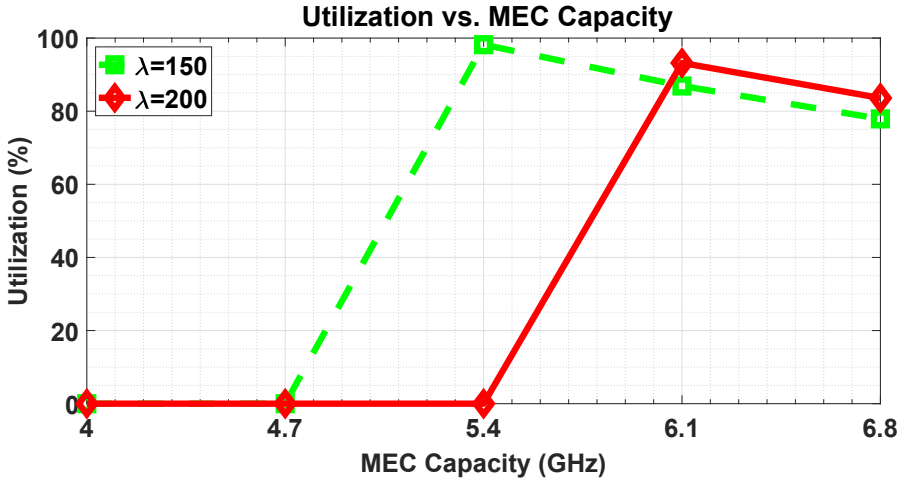


Figure 2.7: Network utilization under variable MEC capacities.

Fig.2.7 delineates that 0% of the network resources are utilized when $c_m \in \{4, 4.7\}$ GHz and $\lambda_l^t \in \{150, 200\}$ requests/sec as RPWA-D fails to give a feasible solution since the total computing resources specified by the DALA-MIP and required to process the maximum amount of workload exceed those offered by all the edge servers that can be placed by the MESD-IP. The same behavior

is observed when $c_m = 5.4$GHz and $\lambda_l^t = 200$ requests/sec. However, for a lower arrival rate of $\lambda_l^t = 150$ requests/sec, the network was fully utilized as the total computing resources provided by the edge servers were just enough to accommodate all the required applications determined by the DALA-MIP. As the value of c_m continues to increase, we observe that the network utilization decreases given that $\sum_{a \in A} p_a$ stabilized once all the workload is admitted and $\sum_{m \in M} c_m$ increases.

6) Deployment cost evaluation under varying network delay: We evaluate the impact of the increase of the maximum network delay on the edge servers deployment cost. Thus, we consider a smart grid system ($\delta_t = 20ms$) of $L = 10$ locations, $M = 10$ edge servers of varying configurations of computing capacity $c_m \in \{3, 5, 7, 9, 11\}\ GHz$. We vary the setup cost between $\pi_l \in [5 - 20]$ unit cost and we set the cost of 1Ghz to be $k = 1$ unit cost. We consider $A = 20$ applications of $T = 4$ different types to process a workload generated at an arrival rate of $\lambda_l^t = 400$ requests/sec and vary the maximum network delay between $2ms$ and $10ms$.

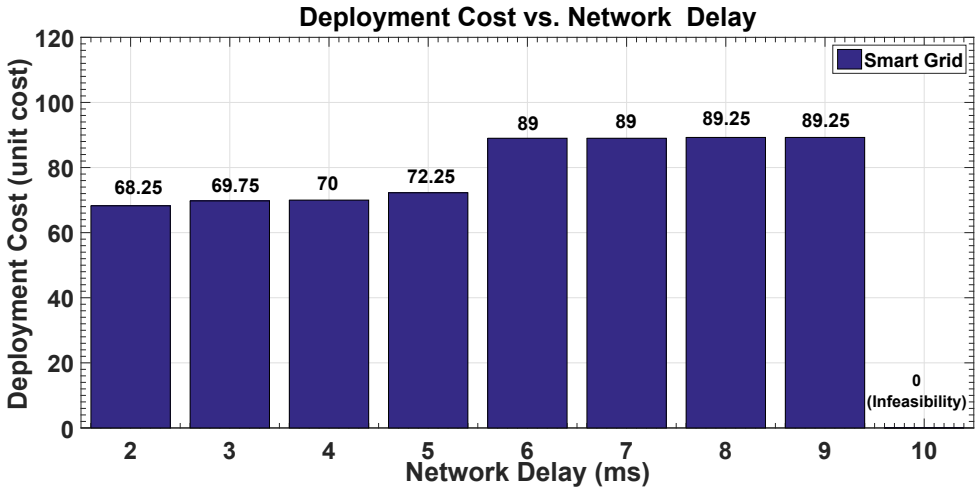


Figure 2.8: Deployment cost under varying network delays.

In fact, increasing the network delay leads to very strict delay limit at the edge server to completely process requests by the hosted applications. As a result, more computing resources are required to be allocated to the applications when increased network delays are considered. Given that each of these applications has an upper bound on the maximum computing resources it can be assigned (p_{max}^a), more applications will be needed to process all the workload. This will require deploying

more edge servers to be able to host these applications, which will eventually increase the deployment cost as shown in Fig.2.8. Note that, for a network delay of $10ms$, no solution was found as the maximum server delay remaining to process the workload within the response time is $0ms$ ($\delta'_t = \delta_t - 2d_n^{max} = 20 - 2 * 10 = 0ms$) (Section 2.5). This, in fact, shows the importance of MEC in responding to the delay sensitive requirements of new emerging services.

7) Deployment cost evaluation under varying workload: Finally, we evaluate the impact of the increase of the generated workload on the deployment cost. We consider the same simulation setup described in Section 2.6.3 (6) and we vary the workload arrival rate $\lambda_l^t \in [250 - 650]\ requests/sec$. Our results depicted in Fig.2.9 show that the deployment cost increases with the increase of the workload as more edge servers need to be provisioned to handle such increase. However, one can note that the deployment cost remains the same for a load of 450, 550 and 650 requests/sec. With a more detailed evaluation of the cause of such result, we noticed that for all the aforementioned generated workloads, the total amount of admitted load remained stable at 17820 requests/sec with a deployment of all the $A = 20$ applications which explains that the number of available applications and the maximum processing capacity p_{max} they can be allocated limited the admission rate which stabilized the deployment cost.

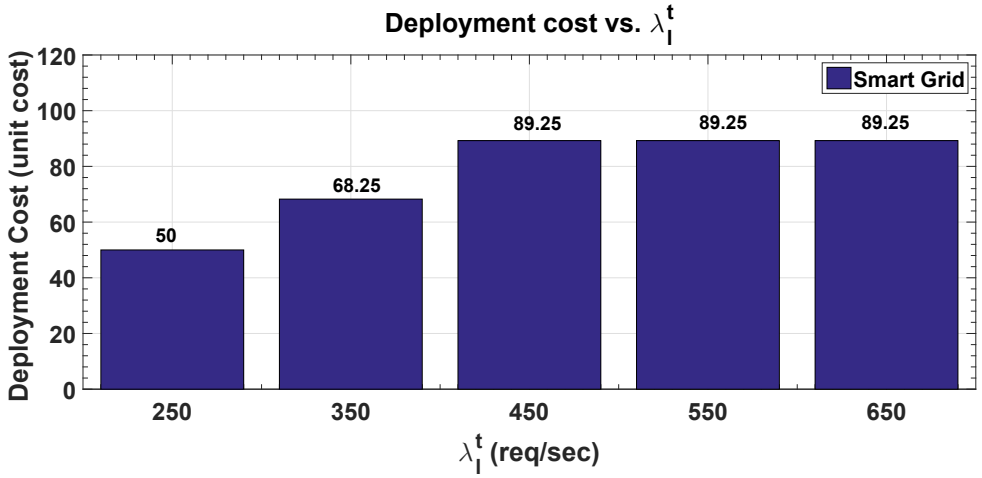


Figure 2.9: Deployment cost under varying workload.

2.6.4 Comparison of RPWA-D with existing work

As mentioned earlier, we are not aware of any existing literature that addressed the joint problem of MEC dimensioning, IoT application placement, and workload assignment. Moreover, most of the existing work did not account for various types of applications. As a result and to evaluate the efficiency of our proposed decomposition approach RPWA-D, we consider only one type of applications ($T = 1$) and compare against the work in [41] that addressed the placement of cloudlets (edge servers) and user (load) assignment in a Wireless Metropolitan Area Network WMAN. In order to apply their Density Based Clustering solution to our work, we use M/M/1 queue as the cloudlet model and assign deadlines to users' requests. Fig. 2.10 shows the simulation results of our proposed approach (RPWA-D) against the method proposed in [41] and referred to as DBC. Our simulation setup consists of $M = 5$ edge servers and $L = 10$ locations. We account for 150 users, each generating requests within the range of $[50 - 100]$ requests/sec. The value for λ_l^t in RPWA-D is calculated based on the total number of user requests generated from location l in the DBC algorithm. Our test regroups different vertical industries with varying response times $\{5, 10, 20, 50, 110\}ms$. As illustrated in Fig.2.10, using our proposed scheme RPWA-D, the ad-

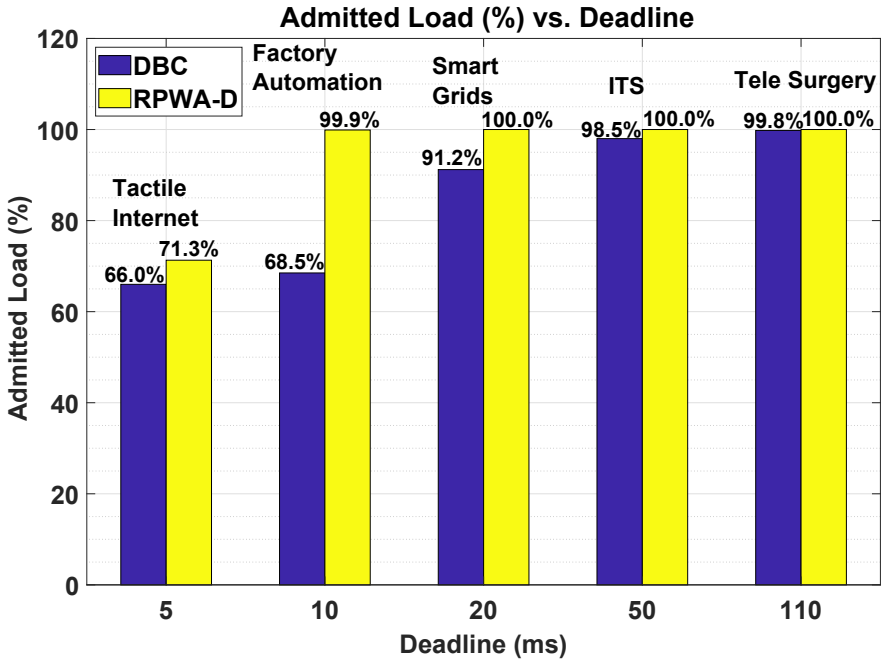


Figure 2.10: Admitted rate over varying deadline.

mission rate can be increased by approximately 5.3%, 31.4%, 8.8%, 1.5% and 0.2% than the DBC algorithm for each of the presented vertical industries respectively. This is explained by the fact that DBC places cloudlets (edge servers) based on the most dense locations while in our approach an optimal edge servers placement is obtained. Further, in the DBC approach, a workload is either admitted or rejected completely, while, our approach allows a fraction of the load generated from one location to be admitted leading to more efficient and flexible workload assignment.

2.7 Conclusion

In this chapter, we studied the RPWA problem that jointly solves the MEC dimensioning, IoT applications placement and workload assignment sub-problems. To the best of our knowledge, this is the first work to address the three aforementioned sub-problems jointly. We mathematically formulated the RPWA problem as a MIP with the objective of minimizing the deployment cost while respecting the IoT applications' maximum allowable response times. Given the showed NP-Hardness of RPWA and the non scalability of the RPWA-MIP, we presented a decomposition approach (RPWA-D) to efficiently solve it. Unlike RPWA-MIP that considers admitting all the workloads, RPWA-D studies the percentage of the workloads that can be admitted given the available computing resources. Hence, RPWA-D can be used to study the trade-off between the increase of the workload admission rate and the extra computing resources needed to process the additional workload. Through extensive simulations, we verified the efficiency of RPWA-D under varying parameters and network conditions. Our proposed decomposition approach serves as a tool for network operators to develop cost effective strategies for edge network planning and design.

Chapter 3

Heterogeneous workload assignment in an MEC-IoT environment for uRLLC

3.1 Introduction

The Internet of Things (IoT) paradigm has emerged as the future Internet since 1999 [55]. It is now evolving to be an intrinsic part of our daily lives, where everyday objects are embedded with transceivers, protocols and microcontrollers turning them into smart things that can communicate with each other [56]. In the current era of smart things, novel use cases (e.g. Tactile Internet, Intelligent Transportation Systems (ITS), Tele-surgery, etc.) are contributing to the evolution of smart cities. Such cities exploit those applications to enhance the quality of life by providing seamless services (i.e., e-health care, intelligent transportation, etc) to end users. However, in order to offer the immersive experience these applications promise, low latency (within milliseconds) and high reliability (less than 10^{-5} packet loss rate) are required [57]. Such requirements would only be possible with the emergence of ultra-reliable low latency communications (uRLLC) promised by the fifth generation mobile communication system ($5G$).

The terms reliability and latency are broad terms that encompass different meaning based on their definitions. For instance, latency could be defined as the end-to-end latency caused by transmission, queuing, and processing delays [58]. Further, Reliability - in general - denotes the probability of successfully transmitting the packets over a period of time. However, it could also be

defined as the availability; which is the probability that a specific service is available [59]. Hence, it is indispensable to properly define both terms. For the sake of clarity, in the rest of the paper the term $latency$ is defined as the end-to-end latency and $reliability$ as the availability; for instance, availability of 99.999% means that the service is available to the end users 99.999% of the time [60].

Within this uRLLC framework, a network has to be designed upon three building blocks; $i)Scale$: accommodation of the huge number of devices and the volume of data they produce $ii)Risk$: robustness towards uncertainty and sudden system changes and $iii)Tale$: dealing with the heterogeneity and randomness of latency and reliability requirements [58]. Building such network is plausible by leveraging technologies such as spatial diversity, machine learning, network coding, and others. [58]. One of the rather appealing enablers is Mobile Edge Computing (MEC).

MEC is an emerging computation paradigm that was introduced to overcome the high latency and low reliability of communication resulting from the large distance between the end users and the cloud [61]. In an MEC infrastructure, the cloud capabilities are brought closer to the IoT devices (e.g. smart cameras, industrial sensors, etc.) by equipping the nearby 4G/5G Base Stations (BSs) or WiFi Acess Points (APs) with computing and storage capabilities, creating an MEC node. Traditionally, the MEC nodes host Virtual Machines (VMs) running different IoT applications to serve the IoT devices. Hence, IoT devices can offload their computationally intensive workloads to the MEC nodes located at a close proximity to them, rather than the distant cloud, and thus, reducing latency and ensuring a more reliable network [62].

Provisioning a smart city with uRLLC backed with MEC would ensure a seamless delivery of services and a satisfactory Quality of Experience (QoE) for the end users. However, a critical aspect of this adoption is considering the various underlying trade-offs in a uRLLC system. Some of these trade-offs could arise between latency, reliability, energy consumption, spectral efficiency, SNR, etc,. For instance, a fundamental trade-off, where intensive research has been done, is between latency and energy consumption in which the device would consume more energy by periodically checking for packet delivery; the more frequent checks, the lower the latency but the higher energy consumption [58].

One of the less tackled trade-offs is between the latency and reliability. For instance, different IoT devices (smart sensors, smart cameras, etc.) generate a shear volume of data to be offloaded

to the MEC nodes to provide a specific type of service (Tactile Internet, Process Automation, etc.). Since each service has both reliability and latency requirements, offloading the workload to an MEC node that satisfies the latency requirement does not guarantee achieving the required reliability and vice versa. Hence, a decision needs to be made on which MEC node the workload should be processed to satisfy and optimize both requirements. Moreover, the generated workload is subject to different failure scenarios either through accessing the network; communication link failures (due to jamming and equipment failure) or being processed on an MEC node; MEC node failure (DoS attacks, hardware failure) [63], causing the service to be unavailable to the end user. Therefore, achieving the ultra high reliability demanded by the IoT services coupled with the aforementioned failure scenarios may require repeated transmissions which would incur a higher latency [58]. Some work has been done to address the reliability and latency in mobile edge computing either jointly or separately. For instance, the authors in [63] considered the probability of occurrences of failure scenarios for both communication and MEC nodes. Further, the work in [61] addressed the problem of workload assignment considering the latency and transmission reliability. However, to the best of our knowledge, no work has considered the reliability of the individual MEC nodes along with the specific reliability and delay requirements of the IoT services.

In this work, we consider a large number of IoT devices generating workloads demanding ultra high reliability and low latency services. We further consider the reliability of the individual MEC nodes. We address the *Workload Assignment (WA)* problem in a smart environment, which aims to assign the generated workloads from the IoT devices to IoT applications of the same requested type hosted on MEC nodes. We formulate the *WA* problem as a Mixed Integer Program (WA-MIP) with the objective of maximizing the admitted load with respect to the latency and reliability requirements. We prove that the WA-MIP is NP-Hard, and hence, we propose an efficient decomposition approach (WA-D) that first solves the resiliency problem and then the latency problem. Thus, WA-MIP is decomposed into two sub-problems, 1) The *Reliability Aware Candidate Selection (RACS)* sub-problem which is tackled by a heuristic search to determine the set of potential MEC nodes satisfying the ultra high reliability requirements. 2) The *Latency Aware Workload Assignment (LAWA)* sub-problem which is formulated as a MIP that takes the solution of RACS as an input and determines the optimal workload assignment with respect to the latency requirements.

Through extensive numerical evaluation we prove that the WA-D is more scalable than the WA-MIP. However, it showed unscalability for a very large network. Therefore, we propose a Tabu-search-based meta-heuristic approach (WA-Tabu) to solve the WA problem.

The remainder of this paper is organized as follows. Section 3.2 presents the related work, the system model is introduced in 3.3, the problem is formulated as a MIP in 3.4, the proposed WA-D approach is discussed in 3.5 and the WA-Tabu in 3.6 . The numerical evaluation and conclusion are presented in 3.7 and 3.8 respectively.

3.2 Related work

One of the rather disruptive advancements in the IoT industry is the introduction of 5G and its promise of uRLLC. Some work has been done on the feasibility of 5G and uRLLC within the context of IoT. Specifically, the authors in [64] studied the use of uRLLC in factory automation and proved its feasibility in factory automation with latency of sub milliseconds and failure rate of 10^{-9}. Further, the work in [65] studied the feasibility of 5G mm-wave as an enabler to Connected Autonomous Vehicles (CAV) applications. It was concluded that the 5G mm-wave satisfied the latency requirements of safety-critical applications and achieved high data rates sufficient for real-time applications (e.g. video streams processing for in-vehicle infotainment system). Further, extensive work has been done exploring the possibilities and use cases of MEC as a key technology for the inception of uRLLC in an IoT environment. For instance, the authors in [14] presented a detailed survey on MEC and its integral part in the development of 5G. The authors explored the various MEC use cases and challenges within the context of IoT and smart cities. Some of the challenges the authors presented was the Service Orchestration; optimizing the synergies between the different entities in an edge network (MEC nodes dimensioning, applications placements and workload assignment), as well as Service Enhancements; improving the users' Quality of Experience (QoE) and achieving Resiliency.

3.2.1 Latency in MEC & IoT infrastructure

Many research has focused on workload offloading in an MEC infrastructure with an emphasis on the end-to-end latency. For instance, Xiang et al. in [66] considered a network where multiple users are offloading their workload to the geographically distributed MEC nodes. They proposed a latency aware offloading framework that minimizes the total response time incurred by the users' workload. Further, the work done in [67] discussed the provisioning of resources (edge servers and applications) as well as the workload assignment, with the objective of minimizing the cost with respect to latency requirements of different industry verticals. A more realistic model is when latency is coupled with another system design parameter. For example, in [68], the authors considered the trade-off between energy consumption and latency by addressing workload offloading problem in an IoT environment. Their proposed framework optimizes the energy consumption and the system utility while respecting the latency requirement.

3.2.2 Reliability in MEC & IoT infrastructure

Unlike latency, little research has considered the reliability issues in MEC and IoT context, let alone considering the trade-off between reliability and latency together.

In [61] the tradeoff between latency and reliability was studied. The problem was formulated to jointly minimize the end to end latency and the failure probability of offloading tasks to MEC nodes. Further, the authors only considered the transmission reliability(offloading failure probability), and only one user with one task to be offloaded. The task is partitioned into subtasks, where each is transmitted using the whole channel bandwidth in a sequential manner. It was concluded that the higher channel quality, the better achieved reliability. On the other hand, the authors in [63] formulated an ILP to minimize the operational cost of placing Virtual Process Control Functions (VPFs) on MEC nodes with respect to capacity and resiliency constraints for different failure scenarios. The failures are due to either MEC node failure or communication link failure. Each failure scenario is assigned a probability based on historical data. Due to the complexity of the problem, the authors developed an iterative algorithm that uses the generalized Benders Decomposition and linear relaxation to reduce the search space.

3.2.3 Novelty of our work in comparison to the literature

In the aforementioned work, most of the authors often considered the latency requirements and overlooked the reliability demands. However, the few works that considered the reliability; communication link failure or MEC node failure, considered a single user task and its end to end delay [61], or the latency was neglected [63]. Further, in both works [61] and [63], the specific service reliability and latency requirements were not considered and they were not tailored towards an IoT context. To the best of our knowledge, our work is the first to consider the workload assignment problem in a densely populated, MEC-enabled IoT environment with multiple workloads/IoT devices while considering both ultra high reliability and low latency requirements of the IoT services, and the availability of the MEC nodes.

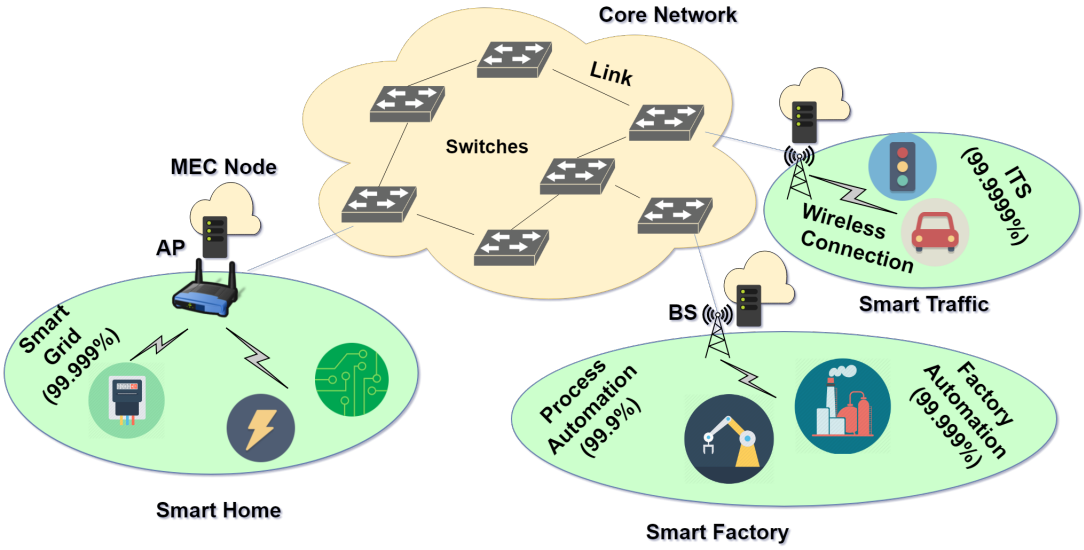


Figure 3.1: IoT enabled smart environment

3.3 System Model

We depict our system model in Figure 3.1 which consists of a smart environment where different IoT devices are spatially distributed and connected to the nearby WiFi APs (or cellular BSs). The IoT devices generate a tremendous workload by requesting IoT services granted by IoT applications of different types and capabilities hosted on MEC nodes. Serving the IoT devices does

not only depend on their requested services' types, but also on the service's latency and reliability requirements. In what follows, we formally explain the network, reliability and latency models.

3.3.1 Network model

Formally, the network is represented by a graph $G(N, E)$ where E is the set of communication links connecting a set of nodes N. $N(= L \cup R)$ is composed of APs/BSs dispersed at different locations $l \in L$, and the backbone network equipment R (routers, switches, etc.). We assume a set M of MEC nodes deployed nearby the APs/BSs, and hence, we denote by π_m^l that MEC node $m \in M$ is located at $l \in L$. IoT devices are connected to their nearby WiFi APs or cellular BSs located at locations $l \in L$, and request IoT services provided by IoT applications hosted on MEC nodes. Therefore, a set A of IoT applications are hosted on Virtual Machines (VMs) running on the MEC nodes. Each application $a \in A$ is assigned p_a processing capacity to process the workload generated by IoT devices. Further, we denote by σ_a^m which MEC node $m \in M$ is hosting application $a \in A$. Moreover, each application $a \in A$ is providing a specific type of IoT service requested by IoT devices. We use T to denote the set of types of the IoT applications. In addition, the parameter μ_a^t represents whether IoT application $a \in A$ is of type $t \in T$. For the sake of simplicity, we assume that each MEC node $m \in M$ is hosting one instance of each IoT application providing service of type $t \in T$, and hence, all MEC nodes are capable of supporting all IoT applications' types. For instance, in a smart hospital environment, IoT devices are collecting various environmental data (temperature, humidity, etc.) to regulate the surroundings and vital data to monitor the patients. In order to deliver such functionalities, the IoT devices would offload their data to the MEC nodes requesting it to be processed by IoT applications providing 1) temperature monitoring and 2) patient tracking services. Hence, all MEC nodes would be running two applications, each providing one of the previously mentioned IoT services.

3.3.2 Reliability and Latency model

The IoT devices offload their workload to the MEC nodes to be processed by IoT applications providing a service of type t. For simplicity and without loss of generality, we consider the aggregate demand generated by all IoT devices located at $l \in L$ and requesting IoT application of

type $t \in T$. This aggregate demand is assumed to follow a Poisson process with an arrival rate of λ_l^t $(requests/sec)$, and has an average computing size of w_t $(CPU cycles)$ per request [66] [67]. Given that the smart environment is backed with uRLLC, different IoT services with stringent latency and reliability figures are granted by applications providing the same service type $t \in T$ to ensure a seamless experience and a satisfactory QoS. For instance, in order to provide the ultimate experience promised by ITS, latency has to be as low as $10 - 100ms$ and an almost guaranteed reliability of 99.9999% [57]. Consequently, we denote by δ_t and r_t the maximum allowable response time when using an IoT application of type t, and the minimum reliability required by the requested service of type $t \in T$ respectively. Within this framework, we model each IoT application as $M/M/1$ queues with an average arrival rate of IoT devices' requests (λ_l^t) and a service rate determined by p_a and w_t. Further, we consider scenarios where the IoT devices' workload may not be assigned to the MEC node at which it was generated (home node), but it is redirected to another MEC node. This could be due to insufficient computing capacity, failure to satisfy the latency or reliability requirements. Hence, we define $h_l^{l'}$ to depict the network delay incurred from redirecting the load from its home MEC node at location l to the MEC node at location l'. Therefore, the total delay experienced by a workload offloaded and processed by an application running on an MEC node is calculated as in Eq.(3.1) which will be explained in details in section 3.4.

$$Delay_{total} = 2(h_l^{l'}) + \frac{1}{ServiceRate - ArrivalRate} \tag{3.1}$$

Similarly, some failure scenarios could result in the IoT workload not being processed or transmitted. These failures could be due to transmission links failure resulting from jamming, denial of service attacks or Hardware Failure (MEC node failures) which could happen due to equipment error, cyber attacks, etc [63]. Some work has already taken care of the transmission failures [61], and in our work, we consider the MEC nodes failure scenarios. Given that the IoT applications are hosted on VMs running on the MEC nodes, a failure of the MEC node would result in the failure of executing the IoT application. In other words, the IoT applications inherit the reliability of their hosting MEC nodes. Hence, we assign a reliability θ_m for each MEC node $m \in M$ to depict its availability. This probability is assigned based on historical data of the average repair time and time

between failures [69]. In order to achieve the required service reliability demanded by the workload, we replicate the load and assign it to one or more applications hosted by one or more MEC nodes such that the required reliability is met. Therefore, the overall achieved reliability of a certain workload is the probability that at least one MEC node that can accept it, is available as shown in Eq.(3.2). The derivation of Eq.(3.2) will be shown in section 3.4.

$$Reliability_{acheived} = 1 - \prod_{m \in M} (1 - \theta_m) \tag{3.2}$$

Hence, the more MEC nodes accepting the workload replicas, the higher the achieved reliability. This is demonstrated in the following illustrative example.

Illustrative example

Consider five locations; l_1, l_2, l_3, l_4, l_5 in a smart environment where workloads are generated requesting different IoT services. Specifically, workload generated from l_1 is requesting Tele-surgery services, while the workloads generated from l_2 and l_3 are requesting Process automation services. The arrival rate (λ_l^t) for each generated load from location $l \in L$ requesting IoT service of type $t \in T$ is given below:

$$\begin{array}{c} \\ t_1 \\ t_2 \end{array} \begin{array}{c} \begin{array}{ccccc} l_1 & l_2 & l_3 & l_4 & l_5 \end{array} \\ \begin{pmatrix} 100 & 0 & 0 & 0 & 0 \\ 0 & 250 & 40 & 0 & 0 \end{pmatrix} \end{array}$$

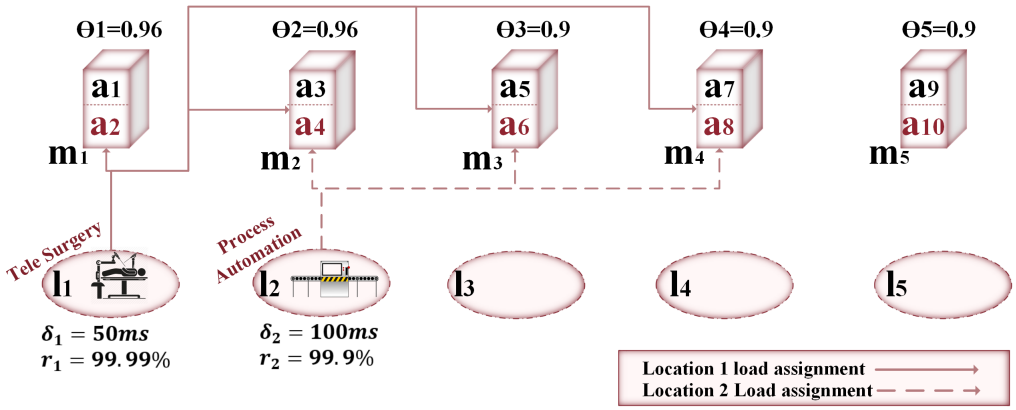
To ensure a satisfactory quality of service, the workload from each location has to be completed within a specific window of time and it has to be served with an ultra high reliability. The response times and reliabilities of the IoT services are listed in Table 3.1. Particularly, workload requesting tele-surgery service has to be processed within $50ms$ and guaranteed an availability of 99.99%, and as for the Process Automation service, a response time of $100ms$ and a reliability of 99.9% are required. To accommodate these requirements, we consider five MEC nodes m_1, m_2, m_3, m_4, m_5 each with reliability θ_m of 0.96, 0.96, 0.9, 0.9, and 0.9 respectively. Each MEC node hosts two IoT applications, each providing tele-surgery and process automation services. Specifically, let a_1, a_3, a_5, a_7, a_9 be the applications providing Tele-surgery services while process automation

services are provided by a_2, a_4, a_6, a_8 and a_{10}. Moreover, let m_1, m_2, m_3, m_4 and m_5 be located at l_1, l_2, l_3, l_4 and l_5 respectively. For the sake of simplicity we assume that the network delay is $1.5ms$ for all MEC nodes. In addition, we assume that the service rate $\frac{p_a}{w_t}$ $(requests/sec)$ of all applications of type tele-surgery is $150requests/sec$ and that of applications of type process automation is $300requests/sec$. In this example, we start off by considering only two workloads are generated from locations; l_1 and l_2 as shown in Figure 3.2(a). With the aforementioned assumptions, workload generated from l_1 could not be sent to only one MEC node as none satisfies its required reliability individually. Consequently, the workload has to be replicated to multiple MEC nodes to achieve its required reliability. The achieved reliability resulting from sending workload generated from l_1 replicas to MEC nodes m_1, m_2, m_3 and m_4 is 0.99998 satisfying its requested service reliability requirements (0.9999). In fact, the workload generated from l_1 could be assigned to any combination of MEC nodes satisfying its reliability according to Eq.3.2. Moreover, the maximum total delay incurred by the workload when processed by one of the MEC nodes that it is sent to, according to Eq.3.1, is:

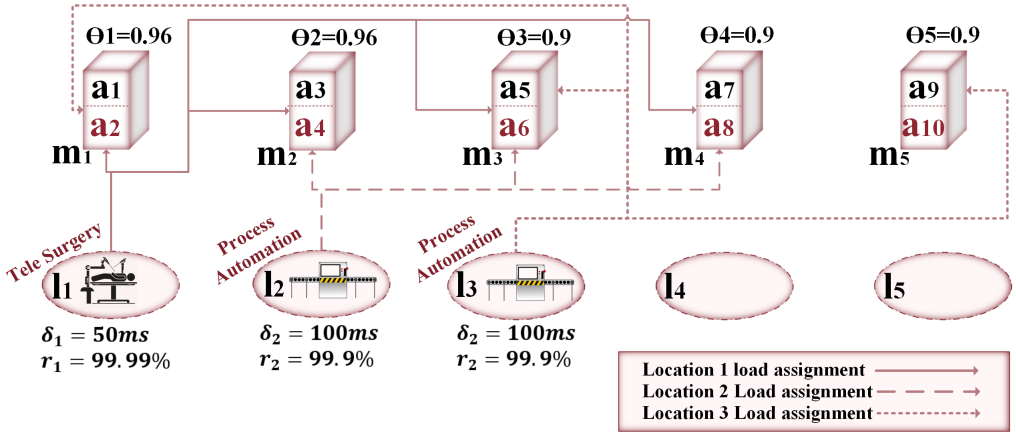
$$2(1.5ms) + \frac{1}{150 - 100} = 23ms \leq 50ms$$

Hence, the workload generated at l_1 is assigned to m_1, m_2, m_3 and m_4. Similarly, the workload generated from l_2 could be sent to any combination of MEC nodes that satisfies its reliability and latency requirements (e.g.$\{m_2, m_3, m_4\}$, $\{m_1, m_2, m_3\}$, etc.). Let m_2, m_3 and m_4 be the MEC nodes that the load is assigned to since the maximum total delay incurred by the workload is $23ms$ ($\leq 100ms$) and the achieved reliability is 0.9996 (≥ 0.999).

Now, we consider another load that is generated from location l_3 requesting process automation service as shown in Figure 3.2(b). The requested service reliability (0.999) would be met by replicating the load and mapping it to any combination of MEC nodes that satisfies its required reliability such as $\{m_2, m_3, m_4\}$, $\{m_1, m_2, m_3\}$ and $\{m_1, m_3, m_5\}$. However, since $\{m_2, m_3, m_4\}$ has already a load assigned to it from l_2 requesting the same service process automation, assigning the new workload generated from l_3 to the same MEC nodes would incur an additional queuing delay at the applications providing service of type Process automation running on those MEC nodes. Therefore, the total delay incurred by the workload generated at l_3 if assigned to $\{m_2, m_3, m_4\}$



(a) Scenario 1



(b) Scenario 2

Figure 3.2: Illustrative Example

would be given by

$$2(1.5ms) + \frac{1}{300 - (250 + 40)} = 103ms \not\leq 100ms$$

Similarly, this workload can not be assigned to any combination of MEC nodes that contains m_2 or m_4 as it would incur $3ms$ additional network delay which would result in a total delay of $103ms$ violating the latency requirements ($100ms$). Thus, the workload from l_3 is replicated and sent to $\{m_1, m_3, m_5\}$ which satisfies the latency and reliability requirements with a total delay of $100ms$ and an achieved reliability of 0.9996.

IoT service	Reliability (%)	Allowable response time (ms)	Used δ_t (ms)
Factory-automation	99.999	0.25-10	10
Smart Grids	99.999	3-20	20
ITS	99.9999	10-100	30
Tele-surgery	99.99	<=250	50
Process-automation	99.9	50-100	100

Table 3.1: IoT QoS requirments for different industry verticals

From this example, it can be seen that determining an optimal workload assignment is challenging where the objective is to satisfy most of the users' requests along with their low latency and ultra high reliability requirements.

3.4 The uRLLC-aware workload assignment problem

3.4.1 Problem Definition

Definition 4. *Given $G(N, E)$, a set M of deployed MEC nodes, a set A of IoT applications of different types hosted on the given MEC nodes, and a set of IoT devices requesting their generated workloads to be offloaded and processed by a specific application type within a determined response time δ_t and a minimum reliability requirement r_t, determine the optimal assignment of the generated workloads to IoT applications that maximizes the admitted load, with respect to their latency and reliability requirements.*

3.4.2 Problem Formulation

Table 3.2 shows the parameters used throughout the formulation of the WA-MIP presented below.

We define a variable $x_l^t \in [0, 1]$ to determine the fraction of load generated from location l and requesting service of type t, that can be admitted to the network, and hence our objective becomes:

$$Maximize \sum_{l \in L} \sum_{t \in T} \lambda_l^t . x_l^t \tag{3.3}$$

Network Inputs	
$G(N, E)$	Network of N nodes where $N = L \cup R$ and E links connecting them.
L	Set of locations where APs/BSs are mounted.
R	Set of backbone network equipment.
M	Set of MEC nodes in $G(N, E)$.
A	Set of IoT applications hosted on $m \in M$.
T	Set of IoT applications' types.
$\pi_m^l \in \{0, 1\}$	Parameter which depicts that MEC node $m \in M$ is deployed at location $l \in L$ (1) or not (0).
$\mu_a^t \in \{0, 1\}$	Parameter which depicts that application $a \in A$ is of type $t \in T$ (1) or not (0).
$\sigma_a^m \in \{0, 1\}$	Parameter to depict that application $a \in A$ is hosted on MEC node $m \in M$ (1) or not (0).
$\delta_t \in \mathbb{R}^+$	Maximum allowable response time required by an application providing service of type $t \in T$.
$r_t \in \mathbb{R}^+$	Minimum required reliability of IoT service of type $t \in T$.
$\theta_m \in \mathbb{R}^+$	Reliability of MEC node $m \in M$.
$p^a \in \mathbb{R}^+$	Processing capacity of application $a \in A$ hosted on $m \in M$.
$\lambda_l^t \in \mathbb{Z}^+$	Arrival rate of requests for an application of type $t \in T$ generated by IoT devices located at $l \in L$.
$w_t \in \mathbb{Z}^+$	Average number of CPU cycles per request for an application of type t .
$h_l^{l'} \in \mathbb{R}^+$	Network delay of a request from its home MEC node at $l \in L$ to its assigned MEC node at $l' \in L$.

Table 3.2: Parameters of WA-MIP.

That is to maximize the percentage of admitted load subject to the reliability and latency constraints. To realize our objective, we introduce a binary decision variable $z_{lt}^a \in \{0, 1\}$ to determine if a workload generated from location $l \in L$ requesting IoT service of type $t \in T$ is mapped to application $a \in A$ that is hosted on a MEC node.

$$z_{lt}^a = \begin{cases} 1 & \text{if generated workload at location } l \text{ demanding} \\ & \text{service of type } t \text{ is mapped to application } a, \\ 0 & \text{otherwise.} \end{cases}$$

Further, a new decision variable $y_{lt}^m \in \{0, 1\}$ is introduced to determine if workload generated from

location $l \in L$ requesting service of type $t \in T$ is mapped to MEC node $m \in M$.

$$y_{lt}^{m} = \begin{cases} 1 & \text{if generated workload at location } l \text{ demanding} \\ & \text{service of type } t \text{ is assigned to MEC node } m, \\ 0 & \text{otherwise.} \end{cases}$$

Further, we declare $r_{lt}^{m} \in [0, 1]$ to depict the achieved reliability when sending the workload generated at location $l \in L$ requesting IoT application of type $t \in T$ to MEC node $m \in M$. Hence, the following constraints are considered:

1) **Workload Assignment:**

We need to make sure that whenever there is a generated load ($\lambda_l^t > 0$), it is mapped to an IoT application $a \in A$; i.e., $z_{lt}^{a} = 1$

$$\sum_{a \in A} z_{lt}^{a} \geq \frac{\lambda_l^t}{H} \quad \begin{matrix} \forall l \in L \\ \forall t \in T \end{matrix} \tag{3.4}$$

H is a large integer number.

Similarly, the load λ_l^t is mapped to an MEC node $m \in M$; $y_{lt}^{m} = 1$

$$\sum_{m \in M} y_{lt}^{m} \geq \frac{\lambda_l^t}{H} \quad \begin{matrix} \forall l \in L \\ \forall t \in T \end{matrix} \tag{3.5}$$

Furthermore, the generated load is mapped to an application providing the same requested service type. This is ensured by Eq.(3.6).

$$z_{lt}^{a} \leq \mu_a^t . \lambda_l^t \quad \begin{matrix} \forall l \in L \\ \forall t \in T \\ \forall a \in A \end{matrix} \tag{3.6}$$

Eq.(3.4)-(3.6) together ensure that loads would always be assigned to IoT applications and if there exists no load, there will be no assignment.

Moreover, Eq.(3.7) and (3.8) ensure that whenever a load λ_l^t is mapped to MEC node $m \in M$ ($y_{lt}^{m} = 1$), the load is also mapped to an application $a \in A$ hosted on the same m ($z_{lt}^{a} = 1$), and vice versa.

$$z_{lt}^{a} \leq \sum_{m \in M} y_{lt}^{m} . \sigma_a^m \quad \begin{matrix} \forall l \in L \\ \forall t \in T \\ \forall a \in A \end{matrix} \tag{3.7}$$

$$y_{lt}^{m} \leq \sum_{a \in A} z_{lt}^{a} . \sigma_a^m \quad \begin{matrix} \forall l \in L \\ \forall t \in T \\ \forall m \in M \end{matrix} \tag{3.8}$$

2) **Reliability Constraints:**

In order to admit a workload requesting service of type t to an MEC node m, its requested service requirements should be met. Hence, the required reliability r_t has to be satisfied. Considering both the service required reliability r_t and the reliability of the MEC node θ_m, one of two possible outcomes would occur. The first possibility is that for an MEC node $m \in M$ with reliability θ_m, the service required reliability r_t is achieved, that is $\theta_m \geq r_t$. Hence, if the workload is sent to that MEC node, the achieved reliability would solely depend on the reliability of the one MEC node processing it. Therefore, Eq.3.9 ensures that the achieved reliability is at least the required reliability.

$$r_{lt}^{m} \geq r_t \quad \begin{matrix} \forall t \in T \\ \forall l \in L \\ \forall m \in M \end{matrix} \tag{3.9}$$

Where r_{lt}^{m} is the achieved reliability and is defined as: $r_{lt}^{m} = y_{lt}^{m}.\theta_m$.

When considering all MEC nodes in the network, Eq.3.9 becomes:

$$\sum_{m} r_{lt}^{m} \geq r_t \quad \begin{matrix} \forall t \in T \\ \forall l \in L \end{matrix} \tag{3.10}$$

On the other hand, the other possibility would be that the required service reliability is not met, that is non of the MEC nodes in the network could satisfy the required reliability (i.e $\theta_m < r_t$). In this case, the workload is replicated and sent to multiple MEC nodes taking advantage of the independency of their reliabilities. Hence, the failures of one MEC node would not influence the availability of the other MEC nodes. This means that replicating and sending the workload to multiple MEC nodes would increase the overall achieved reliability as it would become dependent on the reliability of the MEC nodes that can accept the workload and its replicas. Hence, the new achieved reliability becomes $r_{lt}^{m} = 1 - \prod_{m \in M}(1 - y_{lt}^{m}.\theta_m)$. Eq.(3.11) makes sure that the IoT service required reliability is guaranteed:

$$1 - \prod_{m \in M} (1 - y_{lt}^{m}.\theta_m) \geq r_t \quad \begin{matrix} \forall l \in L \\ \forall t \in T \end{matrix} \tag{3.11}$$

Thus, a decision needs to be made on which subset of MEC nodes will satisfy the required reliability of the requested service.

3) **Latency Constraints:**

The offloaded workload from IoT devices incurs different types of delays. These delays could be due to accessing the network (access delay), redirecting the workload from the home MEC node to another node (network delay) and queuing and processing delays (system delays). In this work, for the sake of simplicity, we consider the access delays to be negligible. Hence, the total delay incurred by the offloaded workload is represented solely by the system and network delays.

We use $d^{mlt}_{network}$ to depict the network delay experienced by workload generated from location l requesting service of type t to be transfered to MEC node m and is given by:

$$d^{mlt}_{network} = \sum_{l' \neq l} h^{l'}_{l}\, y^{m}_{lt}.\pi^{l'}_{m} \quad \begin{matrix}\forall l \in L \\ \forall t \in T \\ \forall m \in M\end{matrix} \tag{3.12}$$

Further, we define d^{alt}_{system} to represent the system delay. Given that each IoT application is modeled as M/M/1 queue with an average arrival rate of $\sum_{l' \in L} z^{a}_{l't}.x^{t}_{l'}.\lambda^{t}_{l'}$ and service rate of $\frac{p_a}{w_t}$, the System Delay is given by:

$$d^{alt}_{system} = z^{a}_{lt}\left(\frac{1}{\frac{p_a}{w_t} - \sum_{l' \in L} z^{a}_{l't}.x^{t}_{l'}.\lambda^{t}_{l'}}\right) \quad \begin{matrix}\forall l \in L \\ \forall t \in T \\ \forall a \in A\end{matrix} \tag{3.13}$$

To avoid congestion at the application, the service rate should be greater than the arrival rate as in Eq.(3.14).

$$\frac{p_a}{w_t} - \sum_{l' \in L} z^{a}_{l't}.x^{t}_{l'}.\lambda^{t}_{l'} \geq 0 \quad \begin{matrix}\forall a \in A \\ \forall t \in T\end{matrix} \tag{3.14}$$

Combining both delays together, the total delay D^{ma}_{lt} incurred by offloading a workload to application a hosted on an MEC node m is:

$$D^{ma}_{lt} = 2d^{mlt}_{network} + d^{alt}_{system} \quad \begin{matrix}\forall l \in L \\ \forall t \in T \\ \forall m \in M \\ \forall a \in A\end{matrix} \tag{3.15}$$

And finally, in order to meet the delay requirements δ_t of each IoT service provided by an application of type t, we have:

$$D^{ma}_{lt} \leq \delta_t \quad \begin{matrix}\forall t \in T \\ \forall a \in A \\ \forall l \in L \\ \forall m \in M\end{matrix} \tag{3.16}$$

The above model is a mixed integer non-linear program. Appendix B presents the linearization of Eqs.(3.11), (3.14) and (3.16).

3.4.3 Complexity Analysis

The WA is a MIP (WA-MIP) which is complex and hard to solve. Its NP-Hardness can be easily proven by a reduction from the Generalized Assignment Problem (GAP) (known to be NP-Hard), where the workloads represent the items to be assigned to bins (MEC nodes) [54, 70]. Given its complexity, we devise two different approaches to solve it.

3.5 WA-D approach

Solving the workload assignment problem with respect to both reliability and latency requirements is challenging. In order to solve the problem, we exploit the independency of the reliability and latency requirements and decompose the problem into two subproblems; the Reliability Aware Candidate Selection subproblem (RACS) and the Latency Aware Workload Assignment subproblem (LAWA).

3.5.1 RACS heuristic

Given the heterogeneity of the MEC nodes reliabilities, not all the MEC nodes can admit the workloads coming from the different locations. Hence, for all workloads generated from different locations demanding the same service type t, a set S_t of potential MEC nodes candidates is generated. The heuristic algorithm starts by determining the set of requested types by the generated workloads. It then forms a set of combinations of MEC nodes. To avoid generating all possible combinations ($2^M - 1$), the size of each combination ranges between one MEC node and a predefined number N of MEC nodes. We choose N based on a worst case scenario; that is when all MEC nodes have the lowest possible reliability θ_m and a generated load requesting a service having the highest required reliability r_t. Hence, N is the minimum number of MEC nodes needed to satisfy the service with the highest required reliability. Each combination is represented in binary to simplify the computation and to depict which MEC nodes are in the set; 0 for $m \notin set$ and 1 otherwise. For each requested type, the achieved reliability is computed for each combination according to Eq.(3.11). If the achieved reliability is $\geq r_t$, the combination is added to S_t. The i^{th} element in the set S_t is denoted by S_i^t, and represents either a potential MEC node or a subset of them, and hence,

$S_i^t \in S_t$. We denote by I the set of all elements belonging to S_t. Further, each element $i \in I$ is weighted and all are sorted ascendingly according to the weighing function defined in Eq.3.17. A pre-defined number of subsets $i \in I$ with the minimum weight are selected for each S_t and passed to the LAWA MIP.

$$\begin{aligned} &W(MEC\ nodes\ subset) = \\ &\quad w_1\left(reliability(MEC\ nodes\ subset) - r_t\right) \\ &\quad + w_2\left(|MEC\ nodes\ subset|\right) \\ &\quad s.t:\ w_1 + w_2 = 1 \end{aligned} \tag{3.17}$$

In other words, the weighting function in Eq.(3.17) ensures using the available resources (MEC nodes) efficiently by selecting the elements that precisely satisfy the required reliability of a specific service of type t requested by a load. For instance, consider two elements in S_t with the same number of MEC nodes and a load requesting a service of type t requiring a reliability of 0.999. Mapping the load to the first element would achieve a reliability of 0.9999 and mapping it to the second element would achieve a 0.999999 reliability. Hence, according to Eq.(3.17), the load should be assigned to the first subset. Thus, a predefined number of elements, minimizing the difference between the required reliability and achieved reliability and consisting of the lowest number of resources (MEC nodes), are selected from each S_t.

3.5.2 LAWA MIP

Given the set S_t of the potential MEC nodes for the generated workloads from different locations demanding service of type t obtained from the RACS heuristic, the LAWA MIP determines the optimal candidate S_i^t for each generated load λ_l^t. The optimal candidates are chosen to maximize the fraction of admitted load while satisfying the workloads' latency requirements. Formally, we use the decision variable $x_l^t \in [0, 1]$ as defined in section 3.4.2 to determine the fraction of admitted load. The LAWA objective is as depicted in Eq.(3.18).

$$Maximize \sum_{t \in T} \sum_{l \in L} \lambda_l^t . x_l^t \tag{3.18}$$

that is to maximize the percentage of admitted load subject to latency constraints. In order to meet our objective, we define g_i^{tl} to depict whether the i^{th} element in the set S_t is selected to admit workload λ_l^t or not.

$$g_i^{tl} = \begin{cases} 1 \text{ if the } i^{th} \text{ element in } S_t \text{ is selected,} \\ 0 \text{ otherwise.} \end{cases}$$

Further, we use the decision variable $z_{lt}^a \in \{0, 1\}$ as defined in section 3.4.2 to determine if workload generated from location l demanding service of type $t \in T$ is mapped to application $a \in A$.

We consider the following constraints.

Eq.(3.19) makes sure that at most one element should be selected from S_t.

$$\sum_i g_i^{tl} \leq 1 \quad \begin{matrix} \forall t \in T \\ \forall l \in L \end{matrix} \tag{3.19}$$

We need to make sure that whenever there is a generated workload demanding service of type t ($\lambda_l^t > 0$), it is mapped to an IoT application $a \in A$ hosted on an MEC node ($z_{lt}^a = 1$).

$$\sum_{a \in A} z_{lt}^a \geq \frac{\lambda_l^t}{H} \quad \begin{matrix} \forall t \in T \\ \forall l \in L \end{matrix} \tag{3.20}$$

Furthermore, the generated load is mapped to an application providing the same requested service type. This is ensured by Eq.(3.21).

$$z_{lt}^a \leq \mu_a^t . \lambda_l^t \quad \begin{matrix} \forall t \in T \\ \forall a \in A \\ \forall l \in L \end{matrix} \tag{3.21}$$

Eqs. (3.20) and (3.21) together ensure that loads would always be assigned to IoT applications and if there exists no load, there will be no assignment.

Moreover, we need to make sure that the workload λ_l^t is assigned to an application $a \in A$ that is hosted on an MEC node $m \in M$ that is in the i^{th} element in S_t. This is ensured by Eq.(3.22).

$$z_{lt}^a \leq \sum_{i \in I} \sum_{m \in S_i^t} g_i^{tl} . \sigma_a^m \quad \begin{matrix} \forall t \in T \\ \forall l \in L \\ \forall a \in A \end{matrix} \tag{3.22}$$

Whenever an element i (subset of MEC nodes) is selected $g_i^{lt} = 1$, the load is assigned to all

applications of type t hosted on the MEC nodes in S_i^t. This is ensured by Eq.(3.23).

$$\sum_{i \in I} |S_i^t| g_i^{tl} \leq \sum_{a \in A} z_{lt}^a \quad \begin{matrix} \forall t \in T \\ \forall l \in L \end{matrix} \tag{3.23}$$

Further, we need to make sure that each individual MEC node in the selected subset S_i^t meets the delay requirements δ_t of workload λ_l^t. This is verified by Eq.(3.24).

$$2\left(\sum_{l' \neq l} h_l^{l'} . z_{lt}^a . \sigma_a^m . \pi_m^{l'}\right) + z_{lt}^a \left(\frac{1}{\frac{p_a}{w_t} - \sum_{l' \in L} z_{l't}^a . x_{l'}^t . \lambda_{l'}^t}\right) \leq \delta_t \quad \begin{matrix} \forall t \in T \\ \forall a \in A \\ \forall l \in L \\ \forall m \in M \end{matrix} \tag{3.24}$$

To avoid congestion, the service rate should be greater than the arrival rate. This is ensured by Eq.(3.25).

$$\frac{p_a}{w_t} - \sum_{l' \in L} z_{l't}^a . x_{l'}^t . \lambda_{l'}^t \geq 0 \quad \begin{matrix} \forall a \in A \\ \forall t \in T \end{matrix} \tag{3.25}$$

As WA-D requires to solve an MIP, this makes it challenging to solve the problem. Hence, in the following section we propose a meta-heuristic approach to accelerate the performance of WA-D. Eqs.(3.24) and (3.25) are non-linear and their linearization is presented in Appendix B.

3.6 WA-Tabu

The Tabu search-based algorithm consists of the following components:

(1) **Initial solution:** The WA-Tabu starts by an initialization step of constructing an initial solution for the workload assignment to the MEC nodes. This is done by first generating a set of potential candidates S_t for each requested IoT service type $t \in T$ as described in section 3.5.1. The load assignment is then performed by selecting the best subset of MEC nodes for each workload. The selection is based on maximizing the weighting function given in Eq.(3.26) that is conditioned to satisfy the latency requirements of the workload's requested

service.

$$\begin{aligned} W(Mec\ nodes\ subset) = w_1(min_{x_l^t}) \\ - w_2\left(reliability(MEC\ nodes\ subset) - r_t\right) \\ - w_3\left(|MEC\ nodes\ subset|\right) \end{aligned} \tag{3.26}$$

$$s.t:\ w_1 + w_2 + w_3 = 1$$

Where $min_{x_l^t}$ is the fraction of the load that a subset can admit, determined by the MEC node admitting the least fraction of load. In other words, the best subset of MEC nodes is the one satisfying the latency requirement of the workload's requested service, maximizing the percentage of admitted load, minimizing the difference between the required and the achieved reliability, and using the lowest number of resources (MEC nodes). The priority of load assignment is then calculated for each load based on the priority function given in Eq.(3.27) and the load with the highest priority is assigned. The process repeats until all loads are considered.

$$\begin{aligned} P(\lambda_l^t) = W(Best\ MEC\ nodes\ subset) \\ - W(2^{nd}\ Best\ MEC\ nodes\ subset) \end{aligned} \tag{3.27}$$

(2) **Neighborhood Solutions:** Given the initial load assignment, the algorithm searches for improving the workload assignment in the neighborhood of the current solution based on the weighting function defined in Eq.(3.26). Within this context, a neighborhood is defined as any solution that involves shifting a workload from the MEC nodes subset that is assigned to, to another. In order to reduce the search space, we consider shifting loads from the most loaded subsets to the least. Hence, if an improved workload assignment is found, the initial load assignment is updated. If, however, no improving assignments were found, the algorithm finds the first non-improving assignment by allowing shifting a workload to the first subset yielding less weight. This raises the chances of reaching a global maximum.

(3) **Tabu list:** A tabu move is defined as shifting the load assignment of a workload from the more loaded subsets to the less loaded. Once the shift is performed, the tabu move is added to the tabu list where it is not considered for the next $tabuListSize$ iterations. This prevents the workload from cycling back to its original subset before allowing other possible moves to be considered. Further, choosing a solution with a lower weight than the current solution is also considered as a tabu move.

(4) **Aspiration Criterion:** In certain scenarios, we allow the violation of the tabu status of moves if the move gives a better solution than the best solution found so far.

(5) **Stopping Criteria:** The algorithm iterates until:

- A maximum number of iterations is reached.
- All the loads are admitted.

The pseudocode for the Tabu-search is shown in Algorithm (1).

3.7 Numerical Evaluation

In this section, we compare the performance of WA-MIP, WA-D and WA-Tabu through extensive numerical evaluation. Further, we evaluate the efficiency of our proposed WA-Tabu approach under varying different parameters.

3.7.1 Experimental setup

To evaluate our algorithms, we consider a network with $L = 25$ locations (unless stated otherwise) where at each location an MEC node is deployed. Each MEC node has a reliability θ_m that is randomly generated between $[0.9 - 0.96]$ [71]. Further, from each location a workload is generated with an average arrival rate λ_l^t taking random values between $[70 - 300] requests/sec.$ The generated workloads request different types of IoT services with various latency and reliability requirements. Hence, we consider $T = 4$ types of IoT services corresponding to the industry verticals and their QoS requirements presented in Table 3.1, which yields an aggregate load of

Algorithm 1 WA-Tabu

Input:
$y_{lt}^m current, x_l^t current$: initial solution
$tabuMove$: $(l, t, subset_{original}, subset_{new})$
$TabuList$: holds tabu moves
$TabuListSize$: indicates size of $TabuList$
$y_{lt}^m best, x_l^t best$: indicates the best Assignment so far
while stopping criteria is not met
  $firstImrpovAssign \leftarrow getFirstImprovAssign()$
  if ($firstImprovAssign$ is found and $\notin tabuList$)
    $y_{lt}^m current, x_l^t current \leftarrow firstImrpovAssign$
  else
    $firstNonImprovAssign \leftarrow$
      $getFirstNonImrpovAssign()$
    if ($firstNonImprovAssign \notin tabuList$)
      $y_{lt}^m current, x_l^t current \leftarrow$
        $firstNonImrpovAssign$
  end if
  if $\sum x_l^t current > \sum x_l^t best$
    $y_{lt}^m best, x_l^t best \leftarrow y_{lt}^m current, x_l^t current$
  end if
  Add $tabuMove$ to $tabuList$
  if $tabuList$ is full
    $remove$ first element added to $tabuList$
  end if
  i$ter++$
end while

$[7k - 30k] requests/sec$. In addition, for each of the requested types, we assume an average computing size w_t generated randomly between 1×10^6 and $2 \times 10^6 CPUcycles$ per request. In order to accommodate the generated workload, we consider applications providing different IoT services hosted on the MEC nodes. We then assume that all the MEC nodes support all types of applications, and hence, the number of applications A is equal to $T \times M$. Each of the applications is assigned computing resources p_a chosen randomly within the range of $[1.7 - 1.9] GHz$. Furthermore, since some of the generated loads might migrate to different MEC nodes other than their home MEC nodes, we assume the network delay to be generated between 1 and $2ms$ at random. Moreover, since the loads could be replicated and offloaded to a subset of MEC nodes satisfying the QoS requirements of its requested type, we choose the size of the set S_t of the potential subsets that a load could be assigned to, to be between 200 and 900 (unless stated otherwise). All our numerical evaluations are averaged over 5 sets. The WA-MIP and the WA-D are evaluated using IBM ILOG CPLEX Optimization Studio v.12.8.

3.7.2 WA-MIP vs. WA-D vs. WA-Tabu

We first compare the performance of our proposed solutions; WA-MIP, WA-D and WA-Tabu in terms of optimality (total admitted load) and scalability (CPU run time). To do so, we vary the network size by increasing the number of locations L, the number of MEC nodes M and the number of applications A. In fact, increasing the number of locations in the network implies increasing the aggregate generated load. Thus, more MEC nodes subsets are needed to accommodate the added load, and hence, increasing I (the size of S_t) as the size of the network increases for both WA-D and WA-Tabu. Further, we consider that the applications are of $T = 4$ different types belonging to smart grid industry vertical with $\delta_t = 20ms$ and $r_t = 99.999\%$. The evaluation results are presented in Table 3.3.

- **Scalability:** From Table 3.3 it is shown that as the size of the network increases, the execution time (CPU run time) increases exponentially for the WA-MIP. This behavior continues until it fails to give a solution when the size of the network is $L = M = 20$, $T = 4$ and $A = 80$. On the other hand, WA-D proved to be more scalable compared to the WA-MIP. This can be

Instance <L, M, T, A, I>	Execution Time (sec)			Admitted Load (%)		
	WA-MIP	WA-D	WA-Tabu	WA-MIP	WA-D	WA-Tabu
<5, 5, 4, 20, 10>	0.025	0.020	0.017	100%	100%	100%
<8, 8, 4, 32, 50>	0.99	0.36	0.047	100%	100%	100%
<11, 11, 4, 44, 100>	2.1	1.31	0.13	100%	100%	100%
<14, 14, 4, 56, 200>	20	2.37	0.40	100%	100%	100%
<17, 17, 4, 68, 300>	288.4	15.2	0.87	100%	100%	100%
<20, 20, 4, 80, 500>	Out of Mem.	27.07	2.40	-	100%	100%
<23, 23, 4, 92, 700>	-	20.1mins	4.61	-	100%	100%

Table 3.3: WA-MIP vs. WA-D vs. WA-Tabu.

seen from the execution time of the WA-D where it increases exponentially as the size of the network increases, but at a slower rate compared to the WA-MIP. It can be observed that the WA-D gave a solution when the network size was $L = M = 25,\ T = 4$ and $A = 92$, but the run time jumped to $20.1mins$. Alternatively, the WA-Tabu proved to be the most scalable as its execution time increases linearly as the size of the network increases.

- **Optimality:** As can be seen from Table 3.3, the algorithm WA-MIP was able to accept all the load up to the network size $L = M = 17,\ T = 4$ and $A = 68$. It failed however, to give a feasible solution for the last two instances. On the other hand, the algorithms WA-D and WA-Tabu were able to admit all the generated load which yields to an optimality gap of 0% in the considered instances.

3.7.3 WA-D vs. WA-Tabu

In the previous section, we showed that WA-MIP is not scalable. Further, we showed that WA-D is more scalable than WA-MIP, and WA-Tabu algorithm is the most scalable for the chosen values of I in Table 3.3. In this section, we further evaluate the performance of the WA-D and WA-Tabu. We thus select the instance from Table 3.3 with the network size $L = M = 17,\ T = 4$ and $A = 68$ to investigate the performance under varying the size of S_t. We vary the size of S_t between $(5 - 300)$. Our evaluation is in terms of execution time and total admitted load. The results are shown in Figures 3.3 and 3.4.

From Figure 3.3, we observe that for small values of I ($I = 5,\ I = 20$), WA-D fails to give a feasible solution after running for a couple of hours as the LAWA-MIP was hard to solve. While on the other hand, WA-Tabu admits 90.9% and 98.5% of the load for $I = 5$ and $I = 20$ respectively.

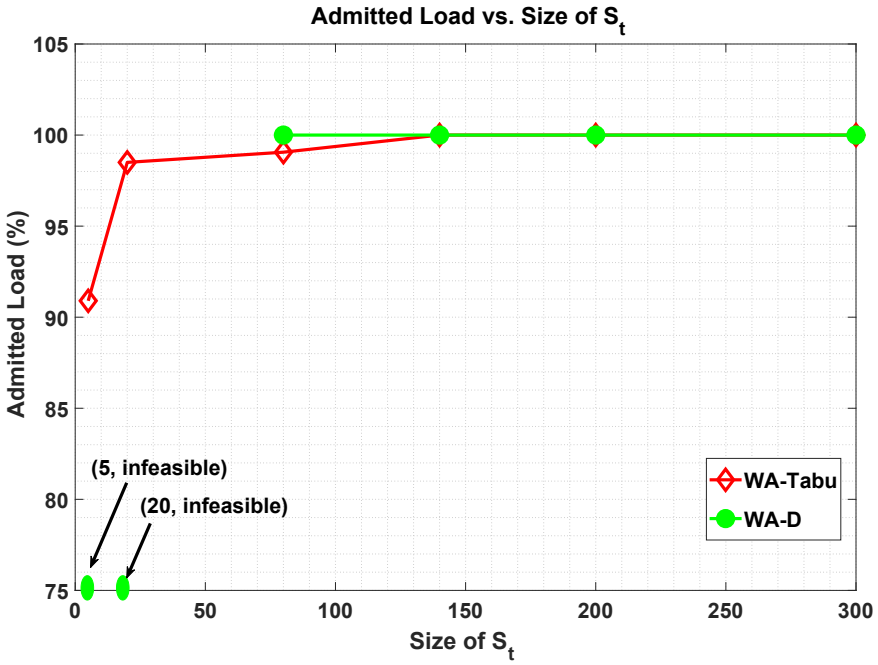


Figure 3.3: Admitted load under varying size of S_t

Further, as the size of I increases, the admission rate increases for both WA-D and WA-Tabu. This is explained by the fact that increasing the size of S_t means increasing the number of potential subsets of MEC nodes that the generated loads could be assigned to, and hence, admitting more load. Moreover, for $I = 80$, WA-D performed slightly better than the WA-Tabu in terms of the total admitted load with a difference less than 1%. In addition, for $I > 80$, both algorithms admit 100% of the total generated load.

Moving to Figure 3.4, it can be observed that the CPU run time for WA-D decreases dramatically as the size of I increases. This is due to the fact that increasing I makes it easier for the WA-D to find a solution as more potential candidates (subsets) becomes available to it, and hence, the lower the execution time. More interestingly, the CPU run time for WA-Tabu algorithm increases slightly as I increases up to $I = 80$ where it starts decreasing until I is equal to 140, then it starts increasing again. The first increasing behavior is because WA-Tabu iterates over the potential subsets to construct the initial solution and find neighboring solutions. Hence, increasing the size of I would increase the run time. When the size of I exceeds 80, the initial solution of the WA-Tabu gives a 100% admitted load, and hence, the algorithm terminates before iterating over the subsets in

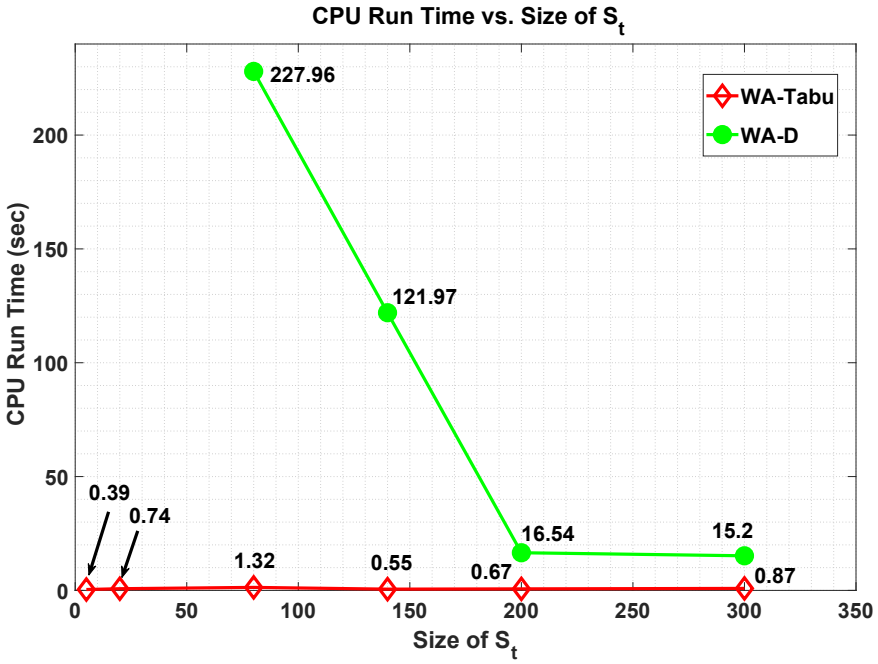


Figure 3.4: Execution time under varying I

S_t to improve the initial solution as the stopping criterion is met. Finally, the execution time hardly increases for $I > 140$ as the WA-Tabu would only iterate over the subsets to construct the initial solution.

3.7.4 Performance evaluation of WA-Tabu:

Varying the workloads for different industry verticals and its impact on the admission rate

We vary the workloads and study the impact on the admission rate for different industry verticals with different requirements. Thus, we increase the generated workload and choose the values of λ_l^t to be $\{100,\ 200,\ 300\}$ requests/sec. The results are presented in Figure(3.5). It can be seen from the figure that for each industry vertical, as the generated workload per location l per IoT service type t increases, the admission rate decreases. This is due to the fact that as λ_l^t increases, more load is requesting to be processed by the same available resources $a \in A$, which are not sufficient to accommodate the newly generated load with the given QoS requirements. Thus, this results in admitting less load. Moreover, for the same value of λ_l^t, the admitted load increases as the latency and reliability requirements become less strict. For instance, for $\lambda_l^t = 200$, the total admitted load

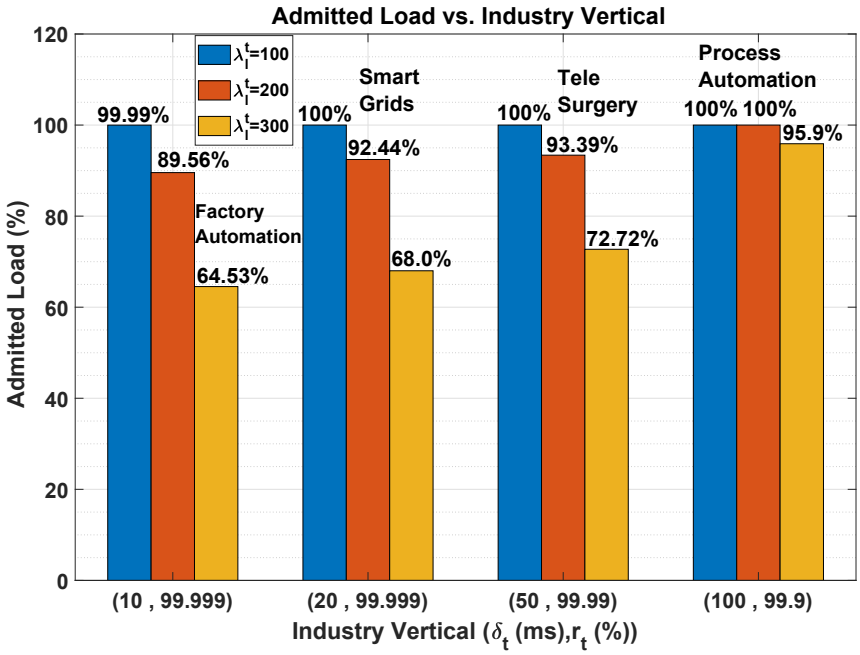


Figure 3.5: Admission rate under varying workloads

for the industry vertical factory automation with the QoS requirements ($\delta_t = 10ms, r_t = 99.999\%$) is 89.56%. While on the other hand, the admitted load for the smart grid industry vertical with the QoS requirements ($20ms, 99.999\%$) increased to 92.44%. The admission rate keeps increasing to reach 100% for the industry vertical process automation with the least strict QoS requirements ($100ms, 99.9\%$).

Varying the network delay for ITS industry vertical and its impact on the admission rate

We evaluate the impact of increasing the network delay on the admission rate for the Intelligent transportation systems (ITS) industry vertical with the latency requirements ($\delta_t = 30ms$). We thus vary the network delay between 6 and $16ms$. The results are depicted in Figure(3.6). It can be observed from the figure that as the network delay increases, the admission rate decreases. For instance, the total admitted load decreased by almost 17% when the network delay went from $12ms$ to $14ms$. This behavior is due to the fact that increasing the network delay leads to a more limited system delay at the MEC nodes to process the requests within the deadline($30ms$). Further, for the network delay $16ms$, no load was admitted as no system delay remained to process the workload

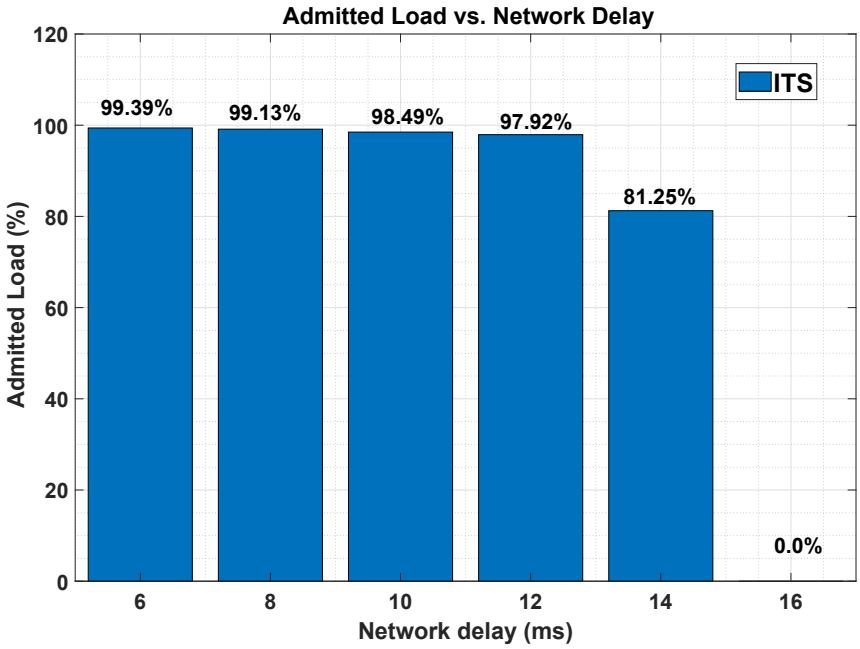


Figure 3.6: Admission rate under varying the network delay

within the maximum allowable response time.

Varying the required reliability for different response times and its impact on the admission rate

We consider the reliability requirements for different IoT industry verticals and overlook their latency requirements. We then study the impact of varying the required reliability (r_t) on the admission rate for two different deadlines ($10, 100ms$). Our results depicted in Figure(3.7) show that for a specific maximum allowable response time (δ_t), as the required reliability becomes more strict, the admission rate decreases. In fact, increasing the reliability requirements is coupled with replicating the workload to more MEC nodes, which leads to an increase in the queuing delay at the MEC nodes. Hence, the available resources (applications) would not be sufficient to completely process the workload. For instance, for $\delta_t = 10ms$, the admission rate decreased from 98.9% to 85.9% when the required reliability went from 99.9% to a more strict value of 99.9999%. Moreover, it can be seen that for a less strict deadline ($100ms$), more system delay at the MEC nodes remains to process the workload, and hence, the admission rate increases as compared to $10ms$.

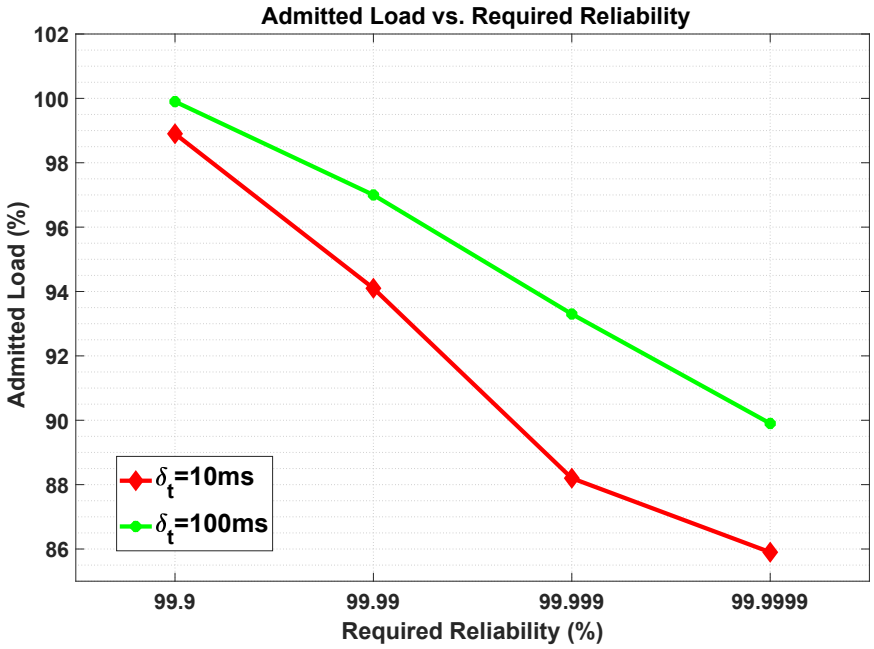


Figure 3.7: Admission rate under varying the required reliability

RACS-heuristic vs. random candidate selection

As part of our evaluation, we explore different strategies for the selection of the subsets composing S_t. Particularly, as discussed in section 3.5.1, we use the RACS heuristic as our selection methodology to select the subsets based on minimizing the achieved reliability with respect to the reliability required, and the used resources (number of used MEC nodes). Alternatively, we devise another selection strategy that randomly selects the subsets. We evaluate the performance of the WA-Tabu algorithm for different industry verticals under both strategies in terms of admission rate and resource utilization. The results are shown in Figures (3.8) and (3.9).

From Figure (3.8), it is observed that for the same selection strategy, the admission rate increases as the QoS requirements become less strict from one industry vertical to the other. More interestingly, for a given industry vertical, the random selection strategy gives a higher admission rate compared to the RACS method, with an insignificant difference between 0 and 7%. While on the other hand, the difference in the resource utilization is remarkable (between 12 and 24%) in favor of the RACS heuristic as demonstrated in Figure (3.9). More precisely, for the industry vertical

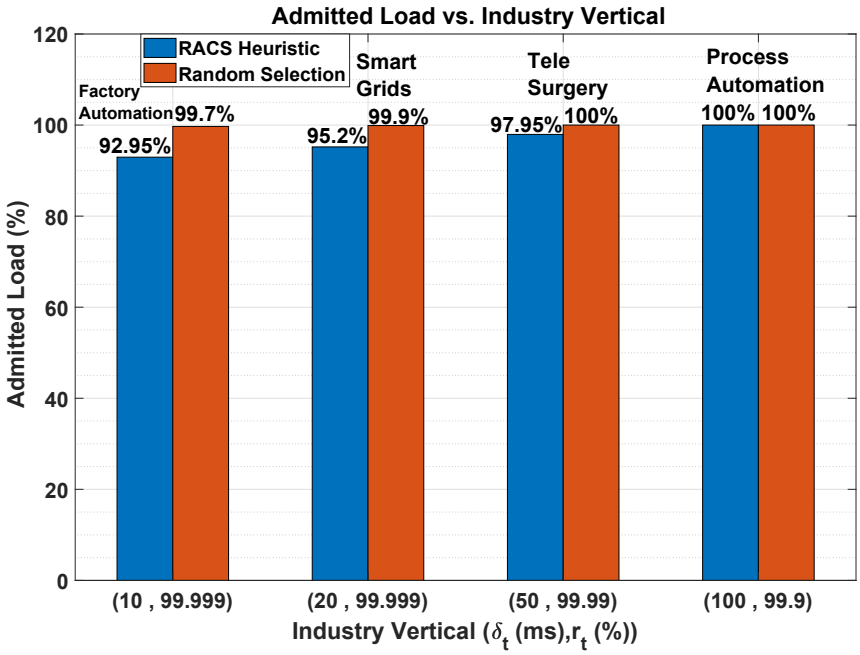


Figure 3.8: Admission rate for different subsets selection strategies

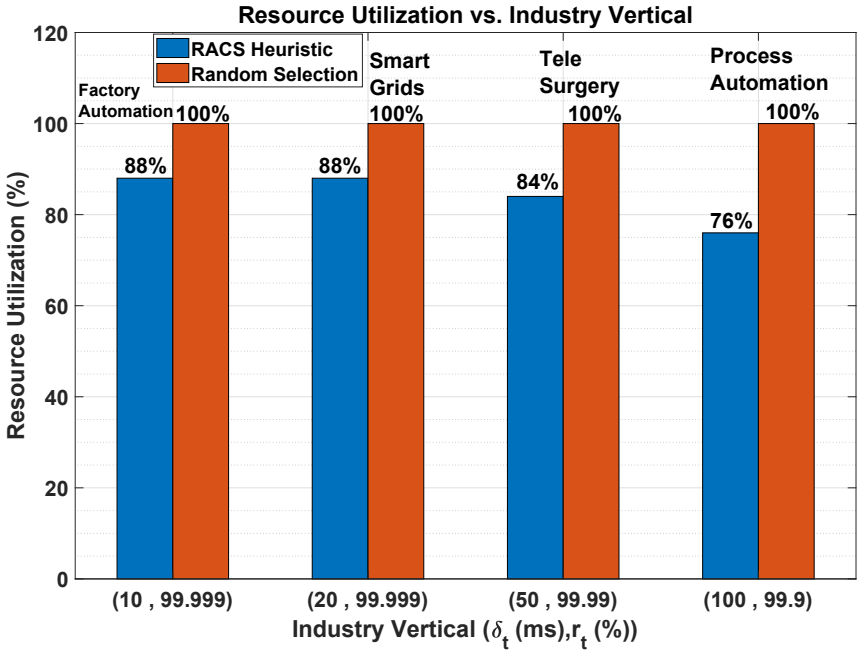


Figure 3.9: Resource utilization for different subsets selection strategies

factory automation with the the most strict QoS requirements, the WA-Tabu with the random selection strategy obtained a 7% more admitted load than the RACS heuristic. However, it utilized 12%

more resources to admit the load. Moreover, for the industry vertical process automation with the least stringent QoS requirements, the resource utilization with the RACS heuristic was significantly lower than the random selection with a difference of 24%, while both yield the same admitted load (100%). In fact, the random selection method provides more diverse subsets composing S_t which yields to less common MEC nodes between the subsets, and hence, the more admitted load. However, this diversity increases the probability of using more resources, and hence, the higher resource utilization. One can note that there is a trade-off between admitting more load and utilizing less resources. Moreover, from the network operator perspective, using the RACS strategy would allow saving energy, as he/she could shut down the unused MEC nodes. Another scenario where the RACS selection method would be more favorable is when an additional load is generated in the network, the unused resources could then be utilized to admit it.

3.8 Conclusion

In this chapter, we studied the WA problem and evaluated different approaches to solve it. We first mathematically formulated the problem as a MIP with the objective of maximizing the total admitted load to the network with respect to the IoT services QoS requirements. We then proved the non-scalability of the WA-MIP in addition to the NP-Hardness of the WA problem. Therefore, we developed the WA-D that decomposes the problem into two subproblems; namely, RACS and LAWA-MIP. The decomposition approach showed a significant improvement over scalability as compared to WA-MIP. Although WA-D proved to be more scalable, it is not scalable enough for very large networks. Thus, a meta-heuristic approach (WA-Tabu) was developed to solve the problem for larger networks with heterogeneous QoS requirements efficiently. Through extensive simulations under various parameters, we evaluated the performance of our proposed approach (WA-Tabu). Our proposed WA-Tabu aids the network operators to use the available resources in a network efficiently to serve the maximum number of end users in an IoT environment.

Chapter 4

Conclusion & Future Work

4.1 Conclusion

MEC has emerged as an enabling paradigm to help in the realization of 5G. The integration of MEC and 5G in an IoT environment would offer the end users a truly immersive experience. However, this integration would raise some challenges have gained a lot of attention in the literature. In this thesis, we studied closely the different challenges facing the new emerging technologies (i.e.,5G, MEC, uRLLC, and IoT) and proposed efficient approaches to solve them. Specifically, in chapter 2, we studied the Resource Provisioning Workload Assignment (RPWA) problem that jointly tackles the MEC nodes and IoT applications provisioning along with the workload assignment problems. We considered a smart environment backed with MEC, with myriad IoT devices generating heterogeneous workloads requesting different IoT services with stringent latency requirements, corresponding to different industry verticals. We proposed an efficient decomposition approach (RPWA-D) to solve the problem. In RPWA-D we decomposed the RPWA problem into two subproblems; Delay Aware Load Assignment (DALA) subproblem to tackle the workload assignment along with determining the needed IoT applications resources to process the load, and the Mobile Edge Servers Dimensioning (MESD) subproblem to address the placement of edge servers and the IoT applications. Through extensive simulations, the results showed that our proposed solution approach can be efficiently used by network operators to design and plan cost effective networks. We further extended this work in our second contribution (chapter 3) to consider the uRLLC

services offered by 5G and address the workload assignment (WA) problem in a smart environment backed with MEC. Within such an environment, enormous number of IoT devices are generating heterogeneous workloads requesting different IoT services with heterogeneous QoS requirements (i.e. low latency and ultra high reliability). We studied and formulated the WA problem as an MIP, and proposed two different approaches to solve it; WA-D and WA-Tabu. In WA-D, we decomposed the problem into Reliability Aware Candidate Selection (RACS) subproblem to generate the set of potential subsets of MEC nodes satisfying the availabiliy requirements of the IoT services. In the second subproblem Latency Aware Workload Assignment (LAWA), the results of RACS are used to determine the optimal assignment of each generated load satisfying its service latency requirement. As the decomposition approach required solving an MIP, we proposed a Tabu search-based meta heuristic (WA-Tabu) to accelerate its performance. Through our numerical evaluation, WA-Tabu proved to be the most scalable and could be leveraged by network operators to serve the end users and provide them with a satisfactory QoE and QoS.

4.2 Future Work

As a future work, we plan to extend our work in the second contribution (chapter 3) to include: 1) the provisioning of MEC nodes; deciding on the number and placement of MEC nodes to be deployed at different locations in the network, 2) the placement of IoT applications resources; deciding on the number of the applications, which MEC nodes to host them and the amount of resources allocated to them, along with 3) the workload assignment. Another future direction could be the dynamic workload assignment as an extension to our second contribution. The workload assignment problem that we considered in 3 was based on the assumption that the generated workloads were already known. This assumption could be reformed to consider a dynamic environment with an online arrival of workloads. One rather interesting research direction is the adoption of flying base stations in 5G to provide network coverage in locations where the network infrastructure is scarce. This concept is anticipated to be fully utilized with the introduction of 6G to not only provide network coverage, but also serve as computing servers [72]. Hence, provisioning the flying base stations within 6G would be an interesting research direction.

Appendix A

Linearization of RPWA problem

A.1 Linearization of RPWA-MIP constraints

Linearization of Eq.(2.7)

Eq.(2.7) is non linear and can be linearized by declaring a new decision variable $\theta_a^{lm} \in \mathbb{R}^+$ such that:

$$\theta_a^{lm} = p_a y_a^{lm} \quad \begin{smallmatrix} \forall l \in L \\ \forall m \in M \\ \forall a \in A \end{smallmatrix} \tag{A.1}$$

Eq.(2.7) can then be replaced by the following equations:

$$\sum_{l \in L} \sum_{a \in A} \theta_a^{lm} \leq c_m \quad \forall m \in M \tag{A.2}$$

$$\theta_a^{lm} \leq H y_a^{lm} \quad \begin{smallmatrix} \forall l \in L \\ \forall m \in M \\ \forall a \in A \end{smallmatrix} \tag{A.3}$$

$$\theta_a^{lm} \leq p_a \quad \begin{smallmatrix} \forall l \in L \\ \forall m \in M \\ \forall a \in A \end{smallmatrix} \tag{A.4}$$

$$\theta_a^{lm} \geq p_a - (1 - y_a^{lm}) H \quad \begin{smallmatrix} \forall l \in L \\ \forall m \in M \\ \forall a \in A \end{smallmatrix} \tag{A.5}$$

$$\theta_a^{lm} \geq 0 \quad \begin{smallmatrix} \forall l \in L \\ \forall m \in M \\ \forall a \in A \end{smallmatrix} \tag{A.6}$$

Where $H = p^a_{max}$ **Linarization of Eq.(2.18)**

The total delay in Eq.(2.18) is given by:

$$2\sum_{a\in A}\sum_{l'\neq l} h^{l'}_l z^a_{lt}\sigma^{l'}_a + \sum_{a\in A} z^a_{lt}\left(\frac{1}{\frac{p_a}{w_t} - \sum_{l''\in L} z^a_{l''t}\lambda^t_{l''}}\right) \leq \delta_t \; {\scriptstyle \forall l\in L \atop \forall t\in T} \tag{A.7}$$

Eq.(A.7) is non linear and can be linearized by declaring a new decision variable ζ^a_{lt} such that:

$$\zeta^a_{lt} \geq z^a_{lt}\left(\frac{1}{\frac{p_a}{w_t} - \sum_{l''\in L} z^a_{l''t}\lambda^t_{l''}}\right) \quad {\scriptstyle \forall l\in L \atop \forall a\in A \atop \forall t\in T} \tag{A.8}$$

Hence, Eq.(A.7) becomes:

$$\underbrace{2\sum_{a\in A}\sum_{l'\neq l} h^{l'}_l z^a_{lt}\sigma^{l'}_a}_{d^{lt}_n} + \underbrace{\sum_{a\in A}\zeta^a_{lt}}_{d^{lt}_s} \leq \delta_t \; {\scriptstyle \forall l\in L \atop \forall t\in T} \tag{A.9}$$

Eq.(A.9) is non linear and can be linearized by declaring a new decision variable $\psi^{al'}_{lt} \in \{0,1\}$ such that:

$$\psi^{al'}_{lt} = z^a_{lt}\sigma^{l'}_a \quad {\scriptstyle \forall l,l'\in L \atop \forall t\in T \atop \forall a\in A} \tag{A.10}$$

Eq.(A.9) can then be replaced by the following equations:

$$\underbrace{2\sum_{a\in A}\sum_{l'\neq l} h^{l'}_l \psi^{al'}_{lt}}_{d^{lt}_n} + \underbrace{\sum_{a\in A}\zeta^a_{lt}}_{d^{lt}_s} \leq \delta_t \; {\scriptstyle \forall l\in L \atop \forall t\in T} \tag{A.11}$$

$$\psi^{al'}_{lt} \leq z^a_{lt} \quad {\scriptstyle \forall l,l'\in L \atop \forall t\in T \atop \forall a\in A} \tag{A.12}$$

$$\psi^{al'}_{lt} \leq \sigma^{l'}_a \quad {\scriptstyle \forall l,l'\in L \atop \forall t\in T \atop \forall a\in A} \tag{A.13}$$

$$\psi^{al'}_{lt} \geq z^a_{lt} + \sigma^{l'}_a - 1 \quad {\scriptstyle \forall l,l'\in L \atop \forall t\in T \atop \forall a\in A} \tag{A.14}$$

Eq.(A.8) is non linear and can be linearized by rewriting it as:

$$\zeta_{lt}^{a}\frac{p_a}{w_t} - \sum_{l''\in L}\zeta_{lt}^{a} z_{l''t}^{a}\lambda_{l''}^{t} \geq z_{lt}^{a} \quad \begin{matrix}\forall l\in L\\ \forall a\in A\\ \forall t\in T\end{matrix} \tag{A.15}$$

and then declaring a new decision variable $\beta_{lt}^{al''}$ such that:

$$\beta_{lt}^{al''} = \zeta_{lt}^{a} z_{l''t}^{a} \quad \begin{matrix}\forall l,l''\in L\\ \forall t\in T\\ \forall a\in A\end{matrix} \tag{A.16}$$

Hence, Eq.(A.15) can be replaced by the following:

$$\zeta_{lt}^{a}\frac{p_a}{w_t} - \sum_{l''\in L}\beta_{lt}^{al''}\lambda_{l''}^{t} \geq z_{lt}^{a} \quad \begin{matrix}\forall l\in L\\ \forall a\in A\\ \forall t\in T\end{matrix} \tag{A.17}$$

$$\beta_{lt}^{al''} \leq H z_{l''t}^{a} \quad \begin{matrix}\forall l,l'\in L\\ \forall t\in T\\ \forall a\in A\end{matrix} \tag{A.18}$$

$$\beta_{lt}^{al'''} \leq \zeta_{lt}^{a} \quad \begin{matrix}\forall l,l'\in L\\ \forall t\in T\\ \forall a\in A\end{matrix} \tag{A.19}$$

$$\beta_{lt}^{al''} \geq \zeta_{lt}^{a} - (1 - z_{l''t}^{a})H \quad \begin{matrix}\forall l,l'\in L\\ \forall t\in T\\ \forall a\in A\end{matrix} \tag{A.20}$$

$$\beta_{lt}^{al''} \geq 0 \quad \begin{matrix}\forall l,l'\in L\\ \forall t\in T\\ \forall a\in A\end{matrix} \tag{A.21}$$

Where H is a big integer number.

Eq.(A.17) is still non linear and can be linearized using the McCormick envelopes method. This method involves introducing a new decision variable γ_{lt}^{a} such that:

$$\gamma_{lt}^{a} = \zeta_{lt}^{a} p_a \quad \begin{matrix}\forall l\in L\\ \forall t\in T\\ \forall a\in A\end{matrix} \tag{A.22}$$

In addition, since ζ_{lt}^{a} and p_a are bounded by:

$$\because 0 \leq \zeta_{lt}^{a} \leq \frac{H}{p_{min}^{a}} \tag{A.23}$$

$$\because 0 \leq p_a \leq p_{max}^{a} \tag{A.24}$$

Thus, Eq.(A.17) can be replaced with the following equations:

$$\frac{\gamma_{lt}^{a}}{w_t} - \sum_{l'' \in L} \beta_{lt}^{al''} \lambda_{l''}^{t} \geq z_{lt}^{a} \quad \begin{matrix} \forall l \in L \\ \forall a \in A \\ \forall t \in T \end{matrix} \tag{A.25}$$

$$\gamma_{lt}^{a} \geq 0 \quad \begin{matrix} \forall l \in L \\ \forall a \in A \\ \forall t \in T \end{matrix} \tag{A.26}$$

$$\gamma_{lt}^{a} \geq \frac{H}{p_{min}^{a}} p_a + \zeta_{lt}^{a} p_{max}^{a} - \frac{H}{p_{min}^{a}} p_{max}^{a} \quad \begin{matrix} \forall l \in L \\ \forall a \in A \\ \forall t \in T \end{matrix} \tag{A.27}$$

$$\gamma_{lt}^{a} \leq \frac{H}{p_{min}^{a}} p_a \quad \begin{matrix} \forall l \in L \\ \forall a \in A \\ \forall t \in T \end{matrix} \tag{A.28}$$

$$\gamma_{lt}^{a} \leq \zeta_{lt}^{a} p_{max}^{a} \quad \begin{matrix} \forall l \in L \\ \forall a \in A \\ \forall t \in T \end{matrix} \tag{A.29}$$

A.2 Linearization of DALA-MIP constraints

Linearization of Eq.(2.26)

Eq.(2.26) is non linear and can be rewritten as in Eq.(A.30)

$$\sum_{a \in A_t} z_r^a \frac{p_a}{w} - \sum_{a \in A_t} \sum_{r' \in R_t} z_r^a z_{r'}^a \lambda_{r'} \alpha_{r'} \geq \frac{1}{\delta'} \ \forall r \in R_t \tag{A.30}$$

Eq.(A.30) can the be linearized by declaring three new decision variables: $\nu_r^a \in \mathbb{R}^+$ such that:

$$\nu_r^a = z_r^a p_a \quad \begin{matrix} \forall a \in A_t \\ \forall r \in R_t \end{matrix} \tag{A.31}$$

and $\tau_{rr'}^a \in \{0, 1\}$ such that;

$$\tau_{rr'}^a = z_r^a z_{r'}^a \quad \begin{matrix} \forall a \in A_t \\ \forall r, r' \in R_t \end{matrix} \tag{A.32}$$

$\gamma_{rr'}^a \in [0 - 1]$ such that:

$$\gamma_{rr'}^a = \tau_{rr'}^a \alpha_r' \quad \begin{matrix} \forall a \in A_t \\ \forall r, r' \in R_t \end{matrix} \tag{A.33}$$

Eq.(A.30) can then be replaced by the following equations:

$$\nu_r^a \leq z_r^a p_{max}^a \quad \begin{matrix} \forall a \in A_t \\ \forall r \in R_t \end{matrix} \tag{A.34}$$

$$\nu_r^a \leq p_a \quad \begin{matrix}\forall a \in A_t \\ \forall r \in R_t\end{matrix} \tag{A.35}$$

$$\nu_r^a \geq p_a - p_{max}^a(1 - z_r^a) \quad \begin{matrix}\forall a \in A_t \\ \forall r \in R_t\end{matrix} \tag{A.36}$$

$$\tau_{rr'}^a \leq z_r^a \quad \begin{matrix}\forall a \in A_t \\ \forall r,r' \in R_t\end{matrix} \tag{A.37}$$

$$\tau_{rr'}^a \leq z_{r'}^a \quad \begin{matrix}\forall a \in A_t \\ \forall r,r' \in R_t\end{matrix} \tag{A.38}$$

$$\tau_{rr'}^a \geq z_r^a + z_{r'}^a - 1 \quad \begin{matrix}\forall a \in A_t \\ \forall r,r' \in R_t\end{matrix} \tag{A.39}$$

$$\gamma_{rr'}^a \leq \tau_{rr'}^a \quad \begin{matrix}\forall a \in A_t \\ \forall r,r' \in R_t\end{matrix} \tag{A.40}$$

$$\gamma_{rr'}^a \leq \alpha_r \quad \begin{matrix}\forall a \in A_t \\ \forall r,r' \in R_t\end{matrix} \tag{A.41}$$

$$\gamma_{rr'}^a \geq \alpha_r - (1 - \tau_{rr'}^a) \quad \begin{matrix}\forall a \in A_t \\ \forall r,r' \in R_t\end{matrix} \tag{A.42}$$

$$\sum_{a \in A_t} \frac{\nu_r^a}{w} - \sum_{a \in A_t} \sum_{r' \in R_t} \gamma_{rr'}^a \lambda_{r'} \geq \frac{1}{\delta'} \quad \forall r \in R_t \tag{A.43}$$

Linearization of Eq.(2.28)

Eq.(2.28) is non linear and can be linearized by declaring new decision variable $x_r^a \in [0,1]$ such that:

$$x_r^a = z_r^a \alpha_r \quad \begin{matrix}\forall a \in A_t \\ \forall r \in R_t\end{matrix} \tag{A.44}$$

Eq.(2.28) can then be replaced by the following equations:

$$x_r^a \leq z_r^a \quad \begin{matrix}\forall a \in A_t \\ \forall r \in R_t\end{matrix} \tag{A.45}$$

$$x_r^a \leq \alpha_a \quad \begin{matrix}\forall a \in A_t \\ \forall r \in R_t\end{matrix} \tag{A.46}$$

$$x_r^a \geq \alpha_r - (1 - z_r^a) \quad \begin{matrix}\forall a \in A_t \\ \forall r \in R_t\end{matrix} \tag{A.47}$$

$$\frac{p_a}{w} - \sum_{r \in R_t} x_r^a \lambda_r > 0 \quad \forall a \in A_t \tag{A.48}$$

Appendix B

Linearization of WA problem

B.1 Linearization of Eq.(3.11)

Eq.(3.11) is not linear and can be linearized by first rearranging the terms as follows:

$$1 - r_t \geq \prod_{m \in M} (1 - y_{lt}^m . \theta_m) \quad \begin{matrix} \forall t \in T \\ \forall l \in L \end{matrix} \tag{B.1}$$

and then taking the natural logarithm of both sides, we get:

$$\ln(1 - r_t) \geq \ln \left(\prod_{m \in M} (1 - y_{lt}^m . \theta_m) \right) \quad \begin{matrix} \forall t \in T \\ \forall l \in L \end{matrix} \tag{B.2}$$

Using the natural logarithm properties, Eq.(B.2) becomes:

$$\ln(1 - r_t) \geq \sum_{m \in M} \ln \left((1 - y_{lt}^m . \theta_m) \right) \quad \begin{matrix} \forall t \in T \\ \forall l \in L \end{matrix} \tag{B.3}$$

Now, Eq.(B.3) can be linearized by observing the two possible outcomes of its right hand side:

$$\ln(1 - y_{lt}^m . \theta_m) = \begin{cases} \ln(1 - \theta_m) \text{ if } y_{lt}^m = 1, \\ 0 \text{ if } y_{lt}^m = 0. \end{cases}$$

Hence, the right hand side of Eq.(B.3) could be rewritten as in Eq.(B.4).

$$\sum_{m\in M} y_{lt}^{m}(\ln(1-\theta_m)) \tag{B.4}$$

Thus, Eq.(3.11) can then be replaced by Eq.(B.5).

$$\ln(1-r_t) \geq \sum_{m\in M} y_{lt}^{m} \ln\left((1-\theta_m)\right) \quad \begin{matrix}\forall t\in T\\ \forall l\in L\end{matrix} \tag{B.5}$$

B.2 Linearization of Eqs.(3.16) and (3.24)

Equations (3.16) and (3.24) are not linear and can be linearized by rewriting them as follows:

$$z_{lt}^{a}\left[2\left(\sum_{l'\neq l} h_l^{l'}.\sigma_a^m.\pi_m^{l'}\right)+ \left(\frac{1}{\frac{p_a}{w_t}-\sum_{l''\in L} z_{l''t}^{a}.x_{l''}^{t}.\lambda_{l''}^{t}}\right)\right] \leq \delta_t \quad \begin{matrix}\forall l\in L\\ \forall t\in T\\ \forall a\in A\\ \forall m\in M\end{matrix} \tag{B.6}$$

$$\frac{z_{lt}^{a}}{\frac{p_a}{w_t}-\sum_{l''\in L} z_{l''t}^{a}.x_{l''}^{t}.\lambda_{l''}^{t}}\left(2.\frac{p_a}{w_t}\sum_{l'\neq l} h_l^{l'}.\sigma_a^m.\pi_m^{l'}- 2.\sum_{l'\neq l} h_l^{l'}.\sigma_a^m.\pi_m^{l'}.\sum_{l''\in L} z_{l''t}^{a}.x_{l''}^{t}.\lambda_{l''}^{t}+1\right) \leq \delta_t \quad \begin{matrix}\forall l\in L\\ \forall t\in T\\ \forall a\in A\\ \forall m\in M\end{matrix} \tag{B.7}$$

$$\left(z_{lt}^{a}.2.\frac{p_a}{w_t}\sum_{l'\neq l} h_l^{l'}.\sigma_a^m.\pi_m^{l'}- 2.\sum_{l'\neq l} h_l^{l'}.\sigma_a^m.\pi_m^{l'}.\sum_{l''\in L} z_{lt}^{a}.z_{l''t}^{a}.x_{l''}^{t}.\lambda_{l''}^{t}+z_{lt}^{a}\right) \leq \delta_t.\frac{p_a}{w_t}-\sum_{l''\in L} z_{l''t}^{a}.x_{l''}^{t}.\lambda_{l''}^{t} \quad \begin{matrix}\forall l\in L\\ \forall t\in T\\ \forall a\in A\\ \forall m\in M\end{matrix} \tag{B.8}$$

Eq.(B.8) is nonlinear and can be linearized by declaring two new decision variables $\beta_{ll''t}^{a} \in$

$\{0,1\}$ and $\psi^a_{l''t} \in [0,1]$ such that:

$$\psi^a_{l''t} = z^a_{l''t}.x^t_{l''} \quad \begin{matrix}\forall l'' \in L \\ \forall t \in T \\ \forall a \in A\end{matrix} \tag{B.9}$$

$$\beta^a_{ll''t} = z^a_{lt}.\psi^a_{l''t} \quad \begin{matrix}\forall l,l'' \in L \\ \forall t \in T \\ \forall a \in A\end{matrix} \tag{B.10}$$

Eq.(B.8) can then be replaced by the following equations:

$$\begin{aligned}\Bigg(z^a_{lt}.2.\frac{p_a}{w_t} \sum_{l' \neq l} h^{l'}_l.\sigma^m_a.\pi^{l'}_m - \\ 2.\sum_{l' \neq l} h^{l'}_l.\sigma^m_a.\pi^{l'}_m.\sum_{l'' \in L} \beta^a_{ll''t}.\lambda^t_{l''} + z^a_{lt} \Bigg) \\ \leq \delta_t.\frac{p_a}{w_t} - \sum_{l'' \in L} \psi^a_{l''t}.\lambda^t_{l''} \quad \begin{matrix}\forall l \in L \\ \forall t \in T \\ \forall a \in A \\ \forall m \in M\end{matrix}\end{aligned} \tag{B.11}$$

$$\psi^a_{l''t} \leq z^a_{l''t} \quad \begin{matrix}\forall l'' \in L \\ \forall t \in T \\ \forall a \in A\end{matrix} \tag{B.12}$$

$$\psi^a_{l''t} \leq x^t_{l''} \quad \begin{matrix}\forall l'' \in L \\ \forall t \in T \\ \forall a \in A\end{matrix} \tag{B.13}$$

$$\psi^a_{l''t} \geq x^t_{l''} + z^a_{l''t} - 1 \quad \begin{matrix}\forall l'' \in L \\ \forall t \in T \\ \forall a \in A\end{matrix} \tag{B.14}$$

$$\psi^a_{l''t} \geq 0 \quad \begin{matrix}\forall l'' \in L \\ \forall t \in T \\ \forall a \in A\end{matrix} \tag{B.15}$$

$$\beta^a_{ll''t} \leq z^a_{lt} \quad \begin{matrix}\forall l,l'' \in L \\ \forall t \in T \\ \forall a \in A\end{matrix} \tag{B.16}$$

$$\beta^a_{ll''t} \leq \psi^a_{l''t} \quad \begin{matrix}\forall l,l'' \in L \\ \forall t \in T \\ \forall a \in A\end{matrix} \tag{B.17}$$

$$\beta^a_{ll''t} \geq z^a_{lt} + \psi^a_{l''t} - 1 \quad \begin{matrix}\forall l,l'' \in L \\ \forall t \in T \\ \forall a \in A\end{matrix} \tag{B.18}$$

$$\beta^a_{ll''t} \geq 0 \quad \begin{matrix}\forall l,l'' \in L \\ \forall t \in T \\ \forall a \in A\end{matrix} \tag{B.19}$$

B.3 Linearization of Eqs.(3.14) and (3.25)

Eqs.(3.14) and (3.25) are nonlinear and can be linearized by replacing them by Eqs.(B.12), (B.13), (B.14), (B.15) and (B.20).

$$\frac{p_a}{w_t} - \sum_{l' \in L} \psi^a_{l't}.\lambda^t_{l'} \geq 0 \quad \begin{smallmatrix} \forall t \in T \\ \forall a \in A \end{smallmatrix} \tag{B.20}$$

Bibliography

[1] M. A. Lema, A. Laya, T. Mahmoodi, M. Cuevas, J. Sachs, J. Markendahl, and M. Dohler, "Business case and technology analysis for 5g low latency applications," *IEEE Access*, vol. 5, pp. 5917–5935, 2017.

[2] P. Schulz, M. Matthe, H. Klessig, M. Simsek, G. Fettweis, J. Ansari, S. A. Ashraf, B. Almeroth, J. Voigt, I. Riedel *et al.*, "Latency critical iot applications in 5g: Perspective on the design of radio interface and network architecture," *IEEE Communications Magazine*, vol. 55, no. 2, pp. 70–78, 2017.

[3] A. Gupta and R. K. Jha, "A survey of 5g network: Architecture and emerging technologies," *IEEE access*, vol. 3, pp. 1206–1232, 2015.

[4] V. R. Atukuri and M. P. RamaKrishna Mathe, "Network evolution in 3g/4g: Applications and security issues," *International Journal of Computer Science and Information Technologies*, vol. 2, no. 6, pp. 2835–2837.

[5] M. A. Albreem, "5g wireless communication systems: Vision and challenges," in *2015 International Conference on Computer, Communications, and Control Technology (I4CT)*. IEEE, 2015, pp. 493–497.

[6] D. Evans, "The internet of things how the next evolution of the internet is changing everything," Cisco Internet Business Solutions Group (IBSG), Tech. Rep., apr 2011.

[7] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications," *IEEE communications surveys & tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.

[8] S. Madakam, R. Ramaswamy, and S. Tripathi, "Internet of things (iot): A literature review," *Journal of Computer and Communications*, vol. 3, no. 05, p. 164, 2015.

[9] S. Chen, H. Xu, D. Liu, B. Hu, and H. Wang, "A vision of iot: Applications, challenges, and opportunities with china perspective," *IEEE Internet of Things journal*, vol. 1, no. 4, pp. 349–359, 2014.

[10] N. Saxena, A. Roy, B. J. Sahu, and H. Kim, "Efficient iot gateway over 5g wireless: A new design with prototype and implementation results," *IEEE Communications Magazine*, vol. 55, no. 2, pp. 97–105, 2017.

[11] P. Popovski, K. F. Trillingsgaard, O. Simeone, and G. Durisi, "5g wireless network slicing for embb, urllc, and mmtc: A communication-theoretic view," *IEEE Access*, vol. 6, pp. 55 765–55 779, 2018.

[12] R. S. Somula and R. Sasikala, "A survey on mobile cloud computing: Mobile computing+ cloud computing (mcc= mc+ cc)," *Scalable Computing: Practice and Experience*, vol. 19, no. 4, pp. 309–337, 2018.

[13] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2322–2358, 2017.

[14] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, "On multi-access edge computing: A survey of the emerging 5g network edge cloud architecture and orchestration," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1657–1681, 2017.

[15] D. Sabella, A. Vaillant, P. Kuure, U. Rauschenbach, and F. Giust, "Mobile-edge computing architecture: The role of mec in the internet of things," *IEEE Consumer Electronics Magazine*, vol. 5, no. 4, pp. 84–91, 2016.

[16] Z. Xu, W. Liang, W. Xu, M. Jia, and S. Guo, "Efficient algorithms for capacitated cloudlet placements," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 10, pp. 2866–2880, 2016.

[17] H. Xiang, X. Xu, H. Zheng, S. Li, T. Wu, W. Dou, and S. Yu, "An adaptive cloudlet placement method for mobile applications over gps big data," in *2016 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2016, pp. 1–6.

[18] S. Wang, Y. Zhao, J. Xu, J. Yuan, and C.-H. Hsu, "Edge server placement in mobile edge computing," *Journal of Parallel and Distributed Computing*, 2018.

[19] Q. Fan and N. Ansari, "Cost aware cloudlet placement for big data processing at the edge," in *2017 IEEE International Conference on Communications (ICC)*. IEEE, 2017, pp. 1–6.

[20] J. Santos, T. Wauters, B. Volckaert, and F. De Turck, "Resource provisioning for iot application services in smart cities," in *2017 13th International Conference on Network and Service Management (CNSM)*. IEEE, 2017, pp. 1–9.

[21] M. Taneja and A. Davy, "Resource aware placement of iot application modules in fog-cloud computing paradigm," in *2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*. IEEE, 2017, pp. 1222–1228.

[22] M. I. Naas, L. Lemarchand, J. Boukhobza, and P. Raipin, "A graph partitioning-based heuristic for runtime iot data placement strategies in a fog infrastructure," in *SAC 2018: Symposium on Applied Computing*, 2018.

[23] Q. Fan and N. Ansari, "Application aware workload allocation for edge computing based iot," *IEEE Internet of Things Journal*, 2018.

[24] ——, "Towards workload balancing in fog computing empowered iot," *IEEE Transactions on Network Science and Engineering*, 2018.

[25] R. Deng, R. Lu, C. Lai, T. H. Luan, and H. Liang, "Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 1171–1181, 2016.

[26] A. Yousefpour, G. Ishigaki, R. Gour, and J. P. Jue, "On reducing iot service delay via fog offloading," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 998–1010, 2018.

[27] R. James *et al.*, "The internet of things: a study in hype, reality, disruption, and growth," *Raymond James US Research, Technology & Communications, Industry Report*, 2014.

[28] I. Lee and K. Lee, "The internet of things (iot): Applications, investments, and challenges for enterprises," *Business Horizons*, vol. 58, no. 4, pp. 431–440, 2015.

[29] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[30] S. Yi, C. Li, and Q. Li, "A survey of fog computing: concepts, applications and issues," in *Proceedings of the 2015 Workshop on Mobile Big Data*. ACM, 2015, pp. 37–42.

[31] E. Cuervo, A. Balasubramanian, D.-k. Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl, "Maui: making smartphones last longer with code offload," in *Proceedings of the 8th international conference on Mobile systems, applications, and services*. ACM, 2010, pp. 49–62.

[32] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Transactions on Networking*, vol. 24, no. 5, pp. 2795–2808, 2016.

[33] R. Yu, G. Xue, and X. Zhang, "Application provisioning in fog computing-enabled internet-of-things: A network perspective," *IEEE International Conference on Computer Communications (INFOCOM)*, 2018. [Online]. Available: http://www.public.asu.edu/~ruozhouy/docs/infocom-18-paper.pdf

[34] ETSI. ((2014)) Mobile-edge computing – introductory technical white paper. [Online]. Available: https://portal.etsi.org/portals/0/tbpages/mec/docs/mobile-edge_computing_-_introductory_technical_white_paper_v1%2018-09-14.pdf

[35] K. Dolui and S. K. Datta, "Comparison of edge computing implementations: Fog computing, cloudlet and mobile edge computing," in *Global Internet of Things Summit (GIoTS), 2017*. IEEE, 2017, pp. 1–6.

[36] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1628–1656, 2017.

[37] S. Wang, X. Zhang, Y. Zhang, L. Wang, J. Yang, and W. Wang, "A survey on mobile edge networks: Convergence of computing, caching and communications," *IEEE Access*, vol. 5, pp. 6757–6779, 2017.

[38] A. Munir, P. Kansakar, and S. U. Khan, "Ifciot: Integrated fog cloud iot: A novel architectural paradigm for the future internet of things." *IEEE Consumer Electronics Magazine*, vol. 6, no. 3, pp. 74–82, 2017.

[39] B. P. Rimal, D. P. Van, and M. Maier, "Mobile-edge computing versus centralized cloud computing over a converged fiwi access network," *IEEE Transactions on Network and Service Management*, vol. 14, no. 3, pp. 498–513, 2017.

[40] M. Jia, W. Liang, and Z. Xu, "Qos-aware task offloading in distributed cloudlets with virtual network function services," in *Proceedings of the 20th ACM International Conference on Modelling, Analysis and Simulation of Wireless and Mobile Systems*. ACM, 2017, pp. 109–116.

[41] M. Jia, J. Cao, and W. Liang, "Optimal cloudlet placement and user to cloudlet allocation in wireless metropolitan area networks," *IEEE Transactions on Cloud Computing*, 2017.

[42] M. Maternia, S. E. El Ayoubi, M. Fallgren, P. Spapis, Y. Qi, D. Martín-Sacristán, Ó. Carrasco, M. Fresia, M. Payaró, M. Schubert *et al.*, "5g ppp use cases and performance evaluation models," *see https://5g-ppp. eu/wp-content/uploads/2014/02/5G-PPP-use-cases-and-performance-evaluation-modeling_v1. 0. pdf*, 2016.

[43] G. T. 22.864, "Feasibility study on new services and markets technology enablers - network operation - stage 1," 2016.

[44] P. Porambage, J. Okwuibe, M. Liyanage, M. Ylianttila, and T. Taleb, "Survey on multi-access edge computing for internet of things realization," *arXiv preprint arXiv:1805.06695*, 2018.

[45] L. Zhao, W. Sun, Y. Shi, and J. Liu, "Optimal placement of cloudlets for access delay minimization in sdn-based internet of things networks," *IEEE Internet of Things Journal*, 2018.

[46] A. Ceselli, M. Premoli, and S. Secci, "Mobile edge cloud network design optimization," *IEEE/ACM Transactions on Networking*, vol. 25, no. 3, pp. 1818–1831, 2017.

[47] X. Dos Santos, J. Rafael, T. Wauters, B. Volckaert, and F. De Turck, "Resource provisioning for iot application services in smart cities," in *CNSM2017, the 13e International Conference on Network and Service Management*, 2017, pp. 1–9.

[48] M. Chen and Y. Hao, "Task offloading for mobile edge computing in software defined ultra-dense network," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 3, pp. 587–597, 2018.

[49] S. Chaisiri, B.-S. Lee, and D. Niyato, "Optimization of resource provisioning cost in cloud computing," *IEEE Transactions on Services Computing*, vol. 5, no. 2, pp. 164–177, 2012.

[50] Y. Choi and Y. Lim, "Optimization approach for resource allocation on cloud computing for iot," *International Journal of Distributed Sensor Networks*, vol. 2016, p. 23, 2016.

[51] I. Bolodurina and D. Parfenov, "Development and research of models of organization distributed cloud computing based on the software-defined infrastructure," *Procedia Computer Science*, vol. 103, pp. 569–576, 2017.

[52] D. Zeng, L. Gu, S. Guo, Z. Cheng, and S. Yu, "Joint optimization of task scheduling and image placement in fog computing supported software-defined embedded system," *IEEE Transactions on Computers*, vol. 65, no. 12, pp. 3702–3712, 2016.

[53] L.-Y. Wu, X.-S. Zhang, and J.-L. Zhang, "Capacitated facility location problem with general setup cost," *Computers & Operations Research*, vol. 33, no. 5, pp. 1226–1241, 2006.

[54] S. O. Krumke and C. Thielen, "The generalized assignment problem with minimum quantities," *European Journal of Operational Research*, vol. 228, no. 1, pp. 46–55, 2013.

[55] J. Pontin, "Bill joy's six webs," 2005.

[56] K. Avijit and R. Chinnaiyan, "Iot for smart cities," 2018.

[57] G. Americas, "New services & applications with 5g ultra-reliable low latency communications," 5G Americas, Tech. Rep., 2018.

[58] M. Bennis, M. Debbah, and H. V. Poor, "Ultra-reliable and low-latency wireless communication: Tail, risk and scale," *arXiv preprint arXiv:1801.01270*, 2018.

[59] H. Ji, S. Park, J. Yeo, Y. Kim, J. Lee, and B. Shim, "Introduction to ultra reliable and low latency communications in 5g," *Computing Research Repository (CoRR) abs/1704.05565*, 2017.

[60] D. Öhmann, M. Simsek, and G. P. Fettweis, "Achieving high availability in wireless networks by an optimal number of rayleigh-fading links," in *Globecom Workshops (GC Wkshps), 2014*. IEEE, 2014, pp. 1402–1407.

[61] J. Liu and Q. Zhang, "Offloading schemes in mobile edge computing for ultra-reliable low latency communications," *Ieee Access*, vol. 6, pp. 12 825–12 837, 2018.

[62] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing?a key technology towards 5g," *ETSI white paper*, vol. 11, no. 11, pp. 1–16, 2015.

[63] P. Zhao and G. Dán, "Resilient placement of virtual process control functions in mobile edge clouds," in *IFIP Networking*, 2017.

[64] O. N. Yilmaz, Y.-P. E. Wang, N. A. Johansson, N. Brahmi, S. A. Ashraf, and J. Sachs, "Analysis of ultra-reliable and low-latency 5g communication for a factory automation use case," in *Communication Workshop (ICCW), 2015 IEEE International Conference on*. IEEE, 2015, pp. 1190–1195.

[65] S. M. Khan, M. Chowdhury, M. Rahman, and M. Islam, "Feasibility of 5g mm-wave communication for connected autonomous vehicles," *arXiv preprint arXiv:1808.04517*, 2018.

[66] X. Sun and N. Ansari, "Latency aware workload offloading in the cloudlet network," *IEEE Communications Letters*, vol. 21, no. 7, pp. 1481–1484, 2017.

[67] N. Kherraf, H. A. Alameddine, S. Sharafeddine, C. Assi, and A. Ghrayeb, "Optimized provisioning of edge computing resources with heterogeneous workload in iot networks," *IEEE Transactions on Network and Service Management*, 2019.

[68] Z. Wei and H. Jiang, "Optimal offloading in fog computing systems with non-orthogonal multiple access," *IEEE Access*, vol. 6, pp. 49 767–49 778, 2018.

[69] L. Qu, M. Khabbaz, and C. Assi, "Reliability-aware service chaining in carrier-grade softwarized networks," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 3, pp. 558–573, 2018.

[70] M. Yagiura and T. Ibaraki, "The generalized assignment problem and its generalizations."

[71] L. Qu, C. Assi, K. Shaban, and M. Khabbaz, "Reliability-aware service provisioning in nfv-enabled enterprise datacenter networks," in *2016 12th International Conference on Network and Service Management (CNSM)*. IEEE, 2016, pp. 153–159.

[72] F. Tariq, M. Khandaker, K.-K. Wong, M. Imran, M. Bennis, and M. Debbah, "A speculative study on 6g," 2019.