Author/s:
Shanmuganathan, Thananjeyan

Title:
Framework for Designing Multi-Access Edge Computing Network

Date:
2020

Framework for Designing Multi-access Edge Computing Network

Thananjeyan Shanmuganathan

ORCID: 0000-0002-1385-8482

Submitted in partial fulfilment of the requirements of the degree of

Doctor of Philosophy

July 2020

Department of Electrical and Electronic Engineering

THE UNIVERSITY OF MELBOURNE

Framework for Designing Multi-access Edge Computing Network

Thananjeyan Shanmuganathan

Principal Supervisor: Prof. Ampalavanapillai Nirmalathas
Co-Supervisors: Prof. Elaine Wong
Dr. Chien Aun Chan

Abstract

Multi-access edge computing (MEC) is the next paradigm to support the enormous growth of diverse mobile applications that require high computational power, ultra-low latency, and high bandwidth. The user experience can be enhanced beyond the constrained resources limited by the mobile devices by offloading computation-intensive tasks to the MEC hosts. Since MEC hosts are deployed proximity to the end-users, mobility of users leads to multiple handovers in the mobile network, which leads to application migrations in the MEC network. Hence, there is a critical challenge in MEC to maintain the service continuity between the offloaded user application that is running on the MEC host and the mobile device when a user is moving from radio node to radio node. On the other hand, since a larger number of MEC hosts are going to be deployed within the radio access network, the energy efficiency of these hosts is another challenge for MEC service providers.

In this thesis, we design an energy-efficient MEC network through optimizing the resource allocation and MEC hosts selection problems by considering user movements. Our findings could help mobile operators in developing a real-time network resource orchestration system to reduce network costs while increasing the number of users based on users' mobility patterns. This thesis advances the state-of-the-art by making the following contributions:

1. Correlated user mobility model to produce user trajectories during the morning commute.
2. A utilitarian resource distribution algorithm to select suitable locations to deploy hosts and the right amount of resources for each MEC host

3. Energy-efficient server selection methodologies and energy-efficient virtual machine placement and migration processes to maximize the energy efficiency of the MEC hosts
4. An extended Balas-Geoffrion additive algorithm to select a suitable host based on cost minimization for MEC host selection problem
5. A shortest path-based methodology for host selection and user application migration problem to maximize the energy efficiency of the MEC network.

Declaration

This is to certify that

1. the thesis comprises only my original work towards the PhD,
2. due acknowledgement has been made in the text to all other material used,
3. the thesis is less than 100,000 words in length, exclusive of tables, maps, bibliographies and appendices.

Thananjeyan Shanmuganathan
July 2020

Preface

This thesis comprises only the original work completed during the PhD period of the student at the University of Melbourne, conducted under the supervision of Prof. Ampalavanapillai Nirmalathas, Prof. Elaine Wong and Dr. Chien Aun Chan. The supervisors contributed to insightful technical guidance, comments and discussions. The student independently completed the theoretical analysis, analytical models, simulations, and experiments.

The main contributions of the thesis are discussed in Chapters 3-6 and the published outcomes of the thesis are detailed in Chapter 1 Section 1.4. The student is the first and the primary author of all the publications. The student was responsible for the planning, execution, simulation, result analysis, and writing up the manuscripts. In this process, the student benefited greatly from the supervisors in meetings and discussions where they provided technical comments and assisted the student in revising the manuscripts. The suggestions and comments from the supervisors were invaluable.

The thesis has not been submitted for other qualifications. All the work towards the thesis was carried out after the enrolment in the degree. No third-party editorial assistance was provided in the preparation of the thesis. The student is jointly sponsored by the University of Melbourne and University Grant Commission, Sri Lanka.

Acknowledgement

I would like to express my sincerest gratitude to my principal supervisor, Prof. Ampalavanapillai Nirmalathas. With his depth of knowledge and profound thinking, Nirmalathas guided and supported me in every aspect to help me shape the research work in completing my PhD. Throughout my candidature, Nirmalathas encouraged and motivated me to keep going, assisted and inspired me with his expertise.

My sincere thanks go to my co-supervisor, Prof. Elaine Wong, for her encouragement and technical support on my research and timely feedback. My special thanks go to my co-supervisor, Dr. Chien Aun Chan, for his constant encouragement and help on my research. Over these years, Chien spared his valuable time in reviewing my work and providing insightful comments with patience.

My appreciation also goes to Prof. Pierluigi Mancarella, for chairing my committee and providing constructive feedback on my research progress.

I gratefully acknowledge the joint sponsorship from the University of Melbourne and the University Grant Commission, Sri Lanka. I would like to thank the University of Melbourne for providing me opportunities and resources to pursue my study at such a world-leading institution, and for the outstanding research environment, it has created. My PhD study at the University of Melbourne is made possible also with the help and enlightenment from the teachers and supervisors in my previous education.

At this phase of my PhD journey, my deepest gratitude goes to my mother Rathy Shanmuganathan, father Kanapathippillai Shanmuganathan, brother Janakan Shanmuganathan and sister Gaayathrie Shanmuganathan for their continuous support, affection and encouragements. I would like to thank all of my friends throughout my carrier for their support and encouragements. Last but not least, I am thankful to my wife Tharmika Thananjeyan for her endless love, devotion and understandings and my beloved daughter Thaaraghai Tharmika Thananjeyan who was born recently.

Thananjeyan Shanmuganathan

Melbourne, Australia

July 2020

Contents

1. Introduction 17
   1.1. Introduction of MEC 18
   1.2. Motivations: Challenges in the MEC network 19
   1.3. Thesis Outlines and Original Contributions 22
      1.3.1 Chapter 2 Literature Review 23
      1.3.2 Chapter 3 Deployment and Resource Distributions of MEC Hosts 23
      1.3.3 Chapter 4 Energy-Efficient MEC Hosts 24
      1.3.4 Chapter 5 Optimum MEC Host Selection 25
      1.3.5 Chapter 6 Mobility-aware Energy Optimization in MEC Host Selection 26
      1.3.6 Chapter 7 Conclusions and Future Directions 27
   1.4. Publications Arising from the Thesis 28
2. Literature Review 30
   2.1. Requirements for Next-Generation Networks 30
   2.2. Computation Offloading 31
   2.3. Multi-access Edge Computing 32
      2.3.1 Edge Computing Architectures 33
      2.3.2 Use Cases and applications of MEC 35
      2.3.3 Computation Offloading in MEC 36
      2.3.4 Deployment of the MEC Network 37
   2.4. User Mobility Predictions 39
   2.5. Energy Efficiency in MEC 41
3. Deployment and Resource Distributions of MEC Hosts 45
   3.1. Introduction 45
   3.2. Related Works in Literature 47
   3.3. System Models and Methodologies 50
      3.3.1 Correlated Mobility Model 50
      3.3.2 MEC System Model 51
      3.3.3 Hard Deadline Requirement 53
      3.3.4 Soft Deadline Requirement 54
      3.3.5 Resource Distribution Algorithm 56
   3.4. Simulation and Results 59
      3.4.1 Mobility Model 60
      3.4.2 Task Request Profiles 61
      3.4.3 MEC System Setup 62

3.4.4 Evaluation of the Utilitarian Algorithm 65
3.4.5 Hard Deadline vs Soft Deadline Requirement 67
3.4.6 Different Deployment Comparison 68
3.4.7 Task Acceptance Distribution Based on Mobility 69
3.5. Conclusions 69
4. Energy-Efficient MEC Hosts 72
4.1. Introduction 72
4.2. Related Works in Literature 73
4.3. System Models and Methodologies 75
4.3.1 Power Model 75
4.3.2 Energy-efficient Server Class Selection 77
4.3.3 Energy-efficient Processes 78
4.3.4 Virtual Machines Migration Process 79
4.3.5 Virtual Machine Placement Process 81
4.3.6 Physical Server Activation Process 82
4.3.7 Process Execution Triggers 85
4.4. Simulation and Results 86
4.4.1 Server Class Selection Evaluation 88
4.4.2 Energy-efficient Processes Evaluation 89
4.5. Conclusion 90
5. Optimum MEC Host Selection 93
5.1. Introduction 93
5.2. Related Works in Literature 94
5.3. System Models and Methodologies 95
5.3.1 User Application Modeling 96
5.3.2 Total Cost Minimization 98
5.3.3 Balas-Geoffrion Additive Algorithm 100
5.3.4 Extended Balas-Geoffrion Additive Algorithm 102
5.4. Simulation and Results 103
5.4.1 Benefit of Collaboration Between MEC Hosts 105
5.4.2 Detail of MEC Hosts Collaborations 106
5.4.3 Cost vs Number of Accomplished Tasks 107
5.5. Summary 110
6. Mobility-aware Energy Optimization in MEC Host Selection 112
6.1. Introduction 112
6.2. Related Works in Literature 113
6.3. System Models and Methodologies 114

6.3.1 System Models 114
6.3.2 Energy Modeling 116
6.3.3 Latency Model 117
6.3.4 User Mobility 117
6.3.5 Objective of Optimization 118
6.3.6 Feasible MEC Hosts Selection 119
6.4. Simulation and Results 122
6.4.1 Simulation Setup 122
6.4.2 Total energy consumption in MEC network 126
6.4.3 Power Consumption of Individual Users 127
6.5. Summary 132
7. Conclusions and Future Directions 134
7.1. Summary of Key Contributions 134
7.1.1 MEC Resource Allocation and Host Selection 134
7.1.2 Energy Efficiency in MEC Network 135
7.2. Future Research Directions 136
7.2.1 Dynamic Resource Allocation in MEC Host Selection 136
7.2.2 Optimisation of MEC network for Internet of Things (IoT) Devices 137
7.2.3 Vehicle to Vehicle Communication 138
7.2.4 Edge Intelligence 139
7.3. Conclusions 140

List of Figures

Figure 1.1 Introduction of multi-access edge computing network.....19
Figure 1.2 Computation offloading process of MEC.....20
Figure 1.3 Deployment options of MEC hosts 21
Figure 2.1 Impact of user mobility in MEC. 38
Figure 3.1 Estimated transit time of three representative users commute using public transport during morning peak hours.....48
Figure 3.2 Users' locations as a function of time: (a) random mobility, (b) correlated mobility 57
Figure 3.3 User trajectories: (a) random mobility, (b) correlated mobility 59
Figure 3.4 Total tasks offloading requests profile for selected MEC hosts.....59
Figure 3.5 Selected MEC hosts for results presentation.....60
Figure 3.6 Different Deployment scenarios (a) RN deployment (b) AP deployment.....62
Figure 3.7 Total accepted tasks of selected MEC hosts for different CPU resource allocations. 62
Figure 3.8 Selected MEC hosts and the CPU resources allocations for an instant 63
Figure 3.9 (a) resource allocations comparison for hard deadline requirement applications in AP deployments, **(b)** resource allocations comparison for hard deadline requirement applications in RN deployments.....64
Figure 3.10 (a) resource allocations comparison for soft deadline requirement applications in AP deployments, **(b)** resource allocations comparison for soft deadline requirement applications in RN deployments 65
Figure 3.11 Comparison of total task acceptance for RN and AP deployments.....66
Figure 3.12 Comparison of tasks accomplishments of hard deadline vs soft deadline. 67
Figure 3.13 Comparison of total tasks accepted in RN deployments for random mobility vs correlated mobility.....68
Figure 4.1 Proposed process to address energy-efficiency improvements.....77
Figure 4.2 (a) User trajectories of selected 20 users during morning peak hours (green circle indicates starting point and red circle indicates end points) **(b)** tasks requests profile of selected radio nodes based on correlated mobility (1,000 users). 84
Figure 4.3 Deployment of different classes of servers and the number of servers. Selected MEC hosts for analysis are highlighted in yellow. 85
Figure 4.4 Total power consumption of different classes of servers for different loads (i.e., VMs) 86
Figure 4.5 Total power consumption of different classes of servers deployed in H1. 87
Figure 4.6 Power consumption of different MEC hosts (ES- with energy savings enabled). 88
Figure 4.7 Total power consumptions of the MEC system 89
Figure 5.1 MEC service dependency graph of a user application.....96
Figure 5.2 Network topology graph of the MEC network.....97
Figure 5.3 Balas-Geoffrion algorithm terminology 101
Figure 5.4 Number of iterations required to find the optimal solution for Balas and modified Balas algorithms 103
Figure 5.5 Comparison of accepted requests in independent MEC hosts and collaborative MEC hosts methods 104
Figure 5.6 Details of accepted requests in collaborative MEC hosts method.....105

Figure 5.7 (a) selected MEC hosts **(b)** total costs; for first fifty offloading requests at time 100 s 106
Figure 5.8 (a) selected MEC hosts **(b)** total costs; for first seven hundred offloading requests at time 2400 s. 108
Figure 6.1 Shortest path problem formulation for MEC hosts selection and user application migration problem; (a) is the feasible MEC hosts in each period and (b) is the derived shortest path problem 121
Figure 6.2 users' location at different time 123
Figure 6.3 The total energy consumption of the MEC network and the rejected tasks percentage for Case 3. 124
Figure 6.4 power consumption variations and the task accomplishment latency of a representative user (10th user out of 1,000) 125
Figure 6.5 User trajectories of a representative user (10th user out of 1,000) with the time spent in each radio nodes 125
Figure 6.6 User trajectories of a representative user (100th user out of 1,000) with the time spent in each radio nodes 126
Figure 6.7 power consumption variations and the task accomplishment latency of a representative user (100th user out of 1,000) 129
Figure 6.8 Breakdowns of (a) energy and (b) latency of the representative user (100th user) for a day 130

List of Acronyms

AP	Aggregation point
AR	Augmented reality
CBD	Central business district
CPS	Cyber-physical systems
DVFS	Dynamic voltage and frequency scaling
EPON	Ethernet passive optical network
ETSI	European Telecommunications Standards Institute
ICT	Information communication technology
IoT	Internet of things
ITU	International Telecommunication Union
MCC	Mobile cloud computing
MD	Mobile device
MEC	Multi-access edge computing
NFV	Network functions virtualization
PS	Physical server
QoE	Quality of Experience
QoS	Quality of Service
RAN	Radio access network
RN	Radio node
RWP	Random waypoint
SDN	Software defined networking
TLW	truncated Lévy walk
ULL	Ultra-low latency
V2X	Vehicle to everything
VM	Virtual machine
VR	Virtual reality
XR	Extended reality

1 Introduction

Information and communication technologies continue to reshape people's lifestyle and daily activity. Introduction of smartphones drastically changed the way we communicate with each other on daily basis [1]. Smartphones transformed mobile devices from having basic communication functionality to powerful platforms supporting the converged delivery of communication, entertainment, and computing devices, disrupting a range of consumer technologies in the form of mobile computing [2]. With the increasing computing power of smartphones, rapid growth in mobile applications further fuelled the unprecedented success of smartphones as the pervasive and accessible platform of choice [3]. Mobile applications soon provide us with the ability to achieve things that were never possible, replaced popular gadgets to attain dominance, become part of our way of life, from home to work, from schools to universities, from business to government embracing the rapid adoption [4]. The future of mobile applications can be perceived from the prediction that revenue from mobile applications will nearly be more than doubled in 2023 as of 2019 [5].

On the other hand, the emergence of virtual reality, augmented reality and mixed reality technologies, often referred to as *'XR technologies'* [6], are currently being developed at a rapid pace to support the new mode of interactivity and being examined for adoption across a diverse cross section of applications [7]. The current market demand for XR technologies can be understood from the statistic that states global augmented reality and virtual reality market in the healthcare industry is expected to reach $10.82 billion by 2025, representing a remarkable 2019-2026 compound annual growth rate of 36.1% [8]. Such technologies are incorporated into mobile applications as well as mobile platforms being designed and developed natively to better support such technologies. Latest smartphones can be used to execute some of the XR applications such as Pokémon Go and Titans of space. In addition, external mobile devices such as Oculus Rift and Qualcomm's snapdragon XR smart viewer also available to enjoy the immersive experience of these technologies. Obviously, these technologies would demand even greater computing and storage capabilities in future mobile devices.

Due to the size constraint in mobile devices, computing and storage capacities and battery power are always limited [9]. Thus, the capacity to meet the rising demand for more mobile computing capabilities could potentially be limited by these constraints on the mobile device's capabilities. Computation offloading is a promising solution to solve the resource constraint issues in mobile devices.

In addition, industrial automation is undergoing a tremendous change with the introduction of the Internet of Things (IoT) and cyber-physical system (CPS) concepts in industrial application scenarios which leads the way to Industry 4.0 [10]. Applications such as factory automation (real-time control of machines and systems in fast production and manufacturing lines), process automation (monitoring and diagnostics of industrial elements), smart grid, autonomous driving and optimization of road traffic require very low latency and high reliability [11]. The above-mentioned requirement needs to be satisfied in computation offloading form IoT devices.

1.1. Introduction of MEC

The concept of computation offloading by taking advantage of scalable access to computational resources enabled by the cloud computing is used to address the inherent problems in mobile computing by using external computing resource providers other than the mobile device itself to host the execution of computationally intensive tasks from mobile applications [12] [13]. By exploiting the computing and storage capabilities of the cloud, computation-intensive applications can then be executed on mobile devices [14]. This introduces the concept of mobile cloud computing. Mobile cloud computing offers many advantages for mobile devices such as extended battery life, improving data storage capacity and processing power and improving reliability [15]. While the cloud computing offers significant computation resource, due to their location deep in the network, they are well-suited to applications with higher computation power requirement and less concerned about the excessive round-trip times to get tasks offloaded and executed due to latencies associated with the networking.

Despite the merits of mobile cloud computing, it faces inevitable problems such as long latency and backhaul bandwidth limitations due to the long-distance [14]. Therefore, mobile cloud computing is not suitable for computation offloading from mobile applications which require low latency and high bandwidth [16] [11]. However, stringent

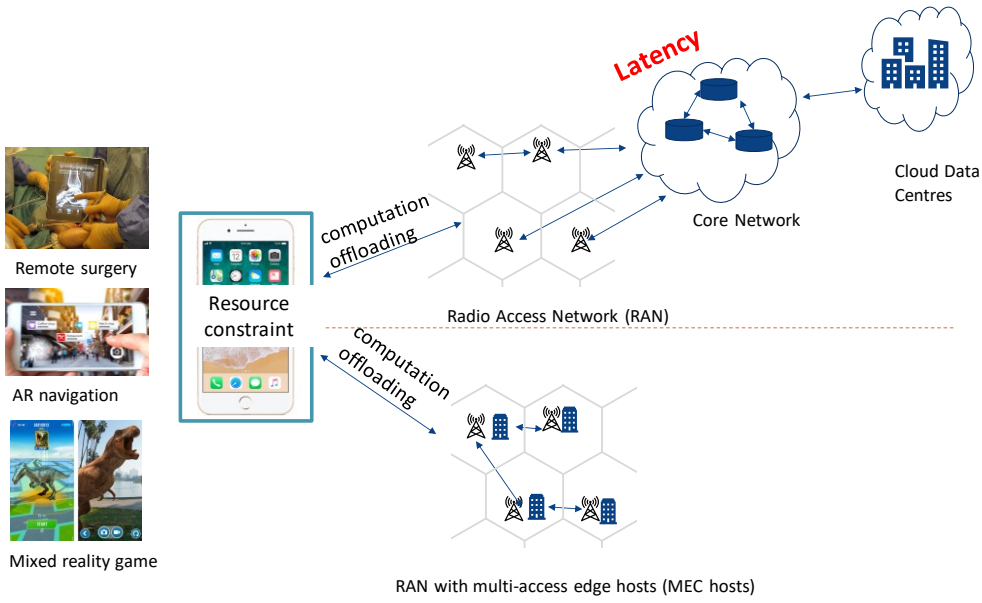

Figure 1.1 Introduction of multi-access edge computing network

latency requirements are of utmost importance for providing a pleasant immersive experience for end users in XR applications [11] [17].

Multi-access edge computing (MEC) is an emerging solution to address the above-mentioned challenges which is expected to be deployed with the fifth-generation (5G) mobile communication networks [18]. The benefits of MEC consist of low latency, proximity, high bandwidth, real-time radio network information and location awareness [19]. MEC hosts, which consists of virtualization infrastructure to provide compute, storage and network resources for mobile applications, are deployed within the radio access network (RAN). Figure 1.1 summarises the need for the MEC network where computing-intensive mobile applications with very low latency requirement drive the mobile network to add MEC support on the existing mobile communication network. Since MEC hosts with limited resources are deployed at RAN, it brings new challenges to MEC.

1.2. Motivations: Challenges in the MEC network

In order to design a MEC network, first of all, demands for MEC network need to be analyzed. One such demands is computation offloading from mobile applications to the MEC system. The following groups of mobile applications are expected to be the

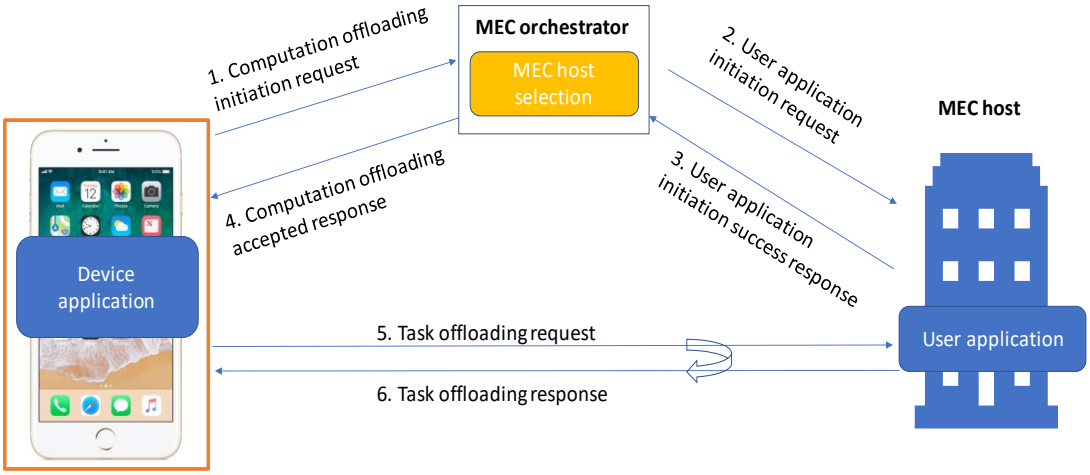

Figure 1.2 Computation offloading process of MEC

dominant mobile applications in future, creating a diversity of requirements and challenges for the mobile networks [20].

- **Enhanced mobile broadband (eMMB)**: Mobile Broadband addresses the human-centric use cases for access to multi-media content, services and data
- **Ultra-reliable and low latency communications (URLLC)**: This use case has stringent requirements of mobile applications for capabilities such as throughput, latency and availability.
- **Massive machine type communications (mMTC)**: This use case is characterized by a very large number of connected devices typically transmitting a relatively low volume of non-delay sensitive data.

To summaries, these new and emerging applications require ultra-reliable, very low latency and high bandwidth. Thus, the MEC networks in 5G are expected to support these requirements.

MEC computation offloading process

Figure 1.2 shows the procedure of initiating computation offloading in MEC. A device application suitable for execution in the MEC system will first send a request to the MEC orchestrator to initiate computation offloading from the device to the MEC system. The request will contain information about the application in terms of rules and requirements. Before selecting a MEC host for serving the request, the MEC orchestrator will examine information such as the required virtualized computing resources, latency requirement,

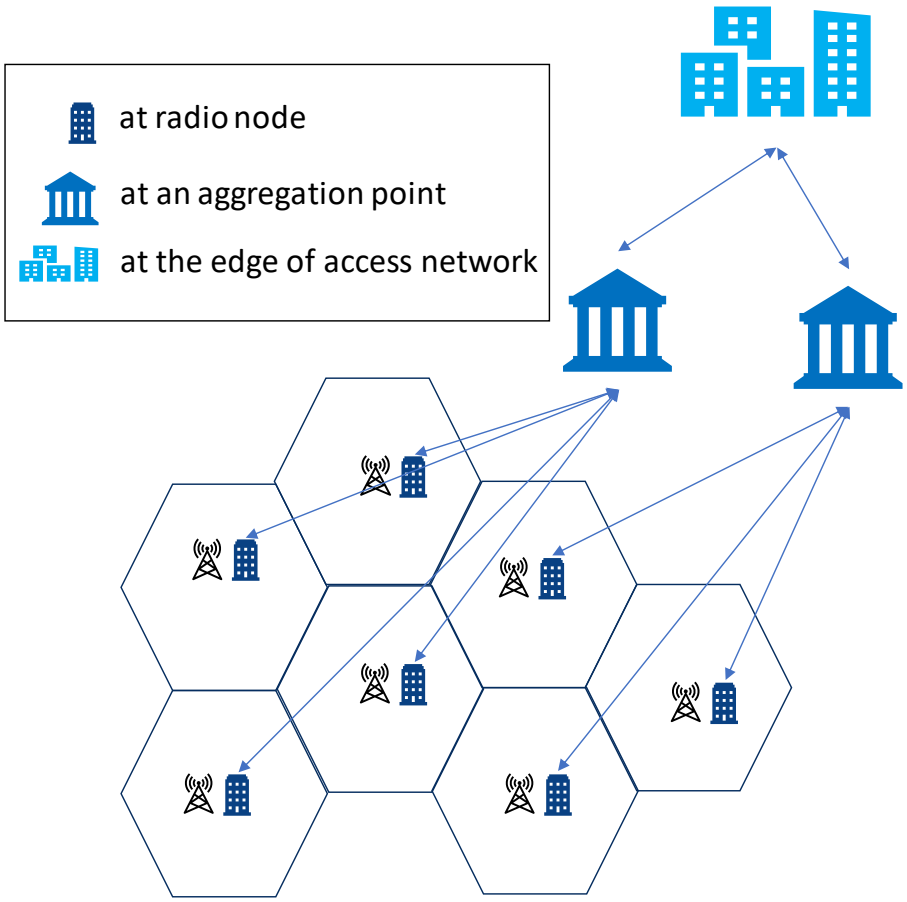

Figure 1.3 Deployment options of MEC hosts

required mobile edge services, connectivity or mobility requirement, and information about the resources currently available in the MEC system. It will then select one or more MEC hosts within the MEC system and triggers the creation of an application context, i.e., user application in the selected MEC host(s). Once the user application is created, the device application that is running on the mobile device can choose to offload the whole or part of the computational load to the user application running on the selected MEC host(s).

Deployment Challenges

MEC service providers need to select suitable locations to deploy MEC hosts in RAN to support the mobile application requirements. MEC hosts can be deployed at a radio node, or a data aggregation point of multiple radio nodes or at the edge of the access network [21]. Figure 1.3 shows the different deployment location selections for MEC service providers to select to deploy MEC hosts. Interestingly, one of these deployment

options can alone be used or a combination of different deployment options can also be considered for deployment to enhance MEC hosts collaborations. This leads to a critical challenge to MEC service providers to select suitable locations to deploy MEC hosts [22]. Once the locations are selected, service providers need to find out the right amount of resources to allocate in each MEC host. Allocating the right amount of resources in MEC host is also a critical challenge since it is much expensive to run a server in RAN than in a cloud data centre [23].

Once the amount of resources has been calculated, selecting suitable physical servers to deploy in the MEC host is another challenge. The total cost of ownership and the energy efficiency need to be considered in selecting suitable physical servers for the deployment. Even after deployment, the service provider's challenge is to make sure that the MEC hosts are operating in an energy-efficient manner to reduce the maintenance cost [23].

In addition to the above-mentioned challenges in the design and deployment of MEC hosts, user mobility imposes a major challenge in maintaining the computation offloading service continuity [24]. User mobility in the RAN may lead to multiple handovers in radio nodes to maintain the continuity of the mobile connection. Computation offloading service continuity is different from the mobile connection continuity. Connection continuity should be maintained by handing over to the new radio node whenever the user moves to a new radio node. On the other hand, service continuity in computation offloading does not always require moving offloaded user application to the nearest MEC host. Because service continuity can be maintained even the user application is hosted in a MEC host that is not nearest. In this case, traffic can be directed between MEC host and mobile device while maintaining the service continuity. This makes computation offloading handovers differ from connection handovers. Thus, selecting a suitable MEC host to provide offloading service to a mobile user is a new challenge for MEC service provider. However, user mobility can be predicted up to an accuracy of 93% [25] which can be used to select suitable MEC host.

1.3. Thesis Outlines and Original Contributions

The key to the focus of this thesis to investigate and develop a framework for design and optimisation of the MEC network considering the above-mentioned challenges in

MEC. The overall objective of the thesis is to design a multi-access edge computing network that

- performs computation offloading of mobile applications,
- delivers high quality of user experience in terms of latency.
- considers maximizing energy efficiency. and
- well-performs in resource constraint environment.

This chapter provides the overarching motivation for this research, by outlining key benefits as well as remaining challenges towards realising highly effective MEC network. The motivations to design a MEC network and the design challenges are discussed in detail and Chapter 1 also provides an introduction to the organization of the Thesis as well as the original contributions described in subsequent chapters. The rest of the thesis is arranged in 6 additional Chapters The detailed content of these chapters are described below.

1.3.1 Chapter 2 Literature Review

Chapter 2 analyses the current trend in MEC. The emerging and future requirement of mobile applications such as ultra-low latency and high data transfer are discussed. Then the possible solutions to satisfy such requirements are discussed. The need for the MEC network is explained with the challenges it introduces. Then the motivations for this thesis is presented with different possible questions to solve such challenges.

1.3.2 Chapter 3 Deployment and Resource Distributions of MEC Hosts

Chapter 3 focuses on selecting suitable locations to deploy MEC hosts and allocating suitable resources for each MEC host. A maximization problem is formulated to maximize the total number of tasks accomplished in the MEC system considering user mobility and resource limitations due to budget constraints. A correlated user mobility model is proposed to mimic user mobility during morning peak hour. Utilitarian resource distribution algorithm is presented to solve the maximization problem. The results show that the propose3d algorithms outperform the other methods in terms of maximizing the total number of tasks accomplished in the MEC system.

Original Contributions of Chapter 3 are listed as follows:

- Mobile applications were grouped based on the latency requirement. Two task deadline requirements were proposed for MEC services: hard deadline (i.e., for applications with crucial latency requirements such as medical applications) and soft deadline (i.e., for applications with flexible deadline requirements such as online gaming, augmented reality and virtual reality tour guides).
- Correlated mobility model is developed to produce user mobility traces during morning peak hours.
- Computation offloading request profiles were modelled for optimizing MEC hosts deployments and resource distributions.
- Utilitarian resource distribution algorithm is constructed to determine the ideal locations to deploy the MEC hosts and the amount of resources needed to be allocated in each MEC host in order to maximize the total number of tasks accomplished while users are moving in correlated mobility given that the resources in the MEC system are limited.
- Extensive simulations are provided to verify the performance of the proposed methods. It is observed that 60% of the total minimum required resources of the MEC system is adequate to satisfy 90% and 82% of the total tasks requests for hard deadline and soft deadline requirement applications, respectively.
- It is also observed that the deployment of MEC host at the radio node and aggregation point have a significant impact on the total tasks accepted profile of MEC services due to additional latency of the backhaul.

1.3.3 Chapter 4 Energy-Efficient MEC Hosts

In the previous chapter, we proposed solutions to select suitable locations to deploy MEC hosts and allocate compute resources in each MEC host. In Chapter 4, energy efficiency is considered in selecting suitable physical servers to deploy in MEC hosts and maintaining service continuity. Energy minimization problem without compromising the user's quality of service is formulated. A methodology based on computation offloading demand profile is proposed to select suitable physical servers. MEC host energy-efficient processes also proposed to minimize the energy consumption during the operations of

MEC host. The simulation results show significant improvement in energy efficiency in MEC hosts compared to traditional methods.

Original Contributions of Chapter 4 are listed as follows:

- A server selection methodology is proposed to select the suitable physical server class based on the task offloading request profile of the MEC host by considering energy efficiency
- An energy-efficient methodology that comprises of three processes to minimize the total energy consumption of the MEC host is also proposed. These three processes are (i) Virtual machine (VM) migration process to select the suitable VMs in a physical server (PS) to migrate in order to save energy or to keep the utilization of the PS below the threshold value. This process also keeps the PSs in the sleep state if there are any PSs in idle state; (ii) VM placement process to place the VMs in the queue in an energy-efficient way; (iii) PS activation process to activate (wake up) the PS from sleep state if it is required.
- A queueing trigger-based approach is then proposed to initiate the (ii) and (iii) in order to minimize both the queueing delay and energy consumption. The proposed methodologies were evaluated based on correlated mobility during morning peak hours and uniform workload.
- The results show that up to 34.32% of the energy can be saved by carefully selecting the physical servers in a MEC host and an average of 16.15% energy savings can be achieved in the MEC system using our proposed method during peak hours.

1.3.4 Chapter 5 Optimum MEC Host Selection

In the previous chapters, deployment and hardware selections are considered for MEC deployment. After the deployment, during computation offloading, selecting suitable MEC host to serve computation offloading request is a critical challenge for MEC service providers. In this chapter, MEC host selection based on cost minimization is considered. Extended Balas-Geoffrion algorithm is proposed to solve the minimization problem in

which an additive algorithm which reduces the time complexity to solve the problem. Simulation results show that the extended Balas algorithm reaches the optimal solution in terms of the number of iterations required to reach the solution.

Original Contributions of Chapter 5 are listed as follows:

- The MEC hosts selection problem is first formulated as a 1-0 integer program (binary programming) problem to minimize the total cost of provisioning the offloading services at the MEC host;
- The use of Balas-Geoffrion additive algorithm is extended to find the optimal solution for the special case of binary programming problems similar to the MEC hosts selection problem. The modifications are as follows:
- The strategy to select the free variable is modified to include the minimum coefficient in the objective function.
- The algorithm described is simplified to omit the tests that are not relevant to the context of this special case problems.
- The proposed algorithm minimizes the computation time complexity since only the addition and subtraction operations are employed to find the optimal solution;
- It is shown through simulations that the number of iterations to find the optimal solution in the proposed extended algorithm outperforms the original Balas-Geoffrion algorithm;
- The trade-off between the servicing costs and the number of tasks rejected is compared between the collaborative and independent methods.

1.3.5 Chapter 6 Mobility-aware Energy Optimization in MEC Host Selection

Energy optimization based on user mobility in MEC host selection is another critical challenge. In this chapter, the shortest path problem is formulated to minimize the total energy consumption in MEC host selection and user application migration problem. Computational intensity is proposed as an important metric to categorize user applications

based on its resource requirement. Results from extensive simulations show that MEC host selection can be pre-calculated based on the user mobility and available resources in the MEC network, where those can be predicted using machine learning and big data analytics.

Original Contributions of Chapter 6 are listed as follows:

- The total duration of computation offloading of the user application is divided into multiple time periods based on the time spent by the user in each default serving MEC host based on the user mobility predictions. This is because the user application migration is required during handovers.
- Energy efficiency in MEC hosts selection and user application migration problem considering the user mobility and latency requirements, which is currently lacking in the literature, was modelled as a shortest path problem.
- The computational intensity (CI) metric, which is important to describe the type of the application based on computation and data offloading requirement is then proposed. CI metric can be used to select suitable MEC hosts in user application deployment.
- The above-mentioned problem was analysed through extensive simulations and observe a predictable pattern in selecting suitable MEC hosts for user application deployment based on the requirement of the application.
- Results show that with accurate predictions of user mobility and the MEC network resource usage, MEC hosts selections can be precalculated to minimize offloading request response time.

1.3.6 Chapter 7 Conclusions and Future Directions

Chapter 7 summarises the research work, important insights and findings, and contributions reported in this thesis. Further, in this chapter, potential future directions based on the research activities covered in this thesis are provided for the benefit of anyone pursuing this research further.

1.4. Publications Arising from the Thesis

- S. Thananjeyan, C. A. Chan, E. Wong and A. Nirmalathas, "Deployment and Resource Distribution of Mobile Edge Hosts Based on Correlated User Mobility," in IEEE Access, vol. 7, pp. 148-159, 2019.
- S. Thananjeyan, C. A. Chan, E. Wong and A. Nirmalathas, "Energy-Efficient Mobile Edge Hosts for Mobile Edge Computing System," 2018 IEEE International Conference on Information and Automation for Sustainability (ICIAfS), Colombo, Sri Lanka, 2018, pp. 1-6.
- S. Thananjeyan, C. A. Chan, E. Wong and A. Nirmalathas, "Optimum Selection of Mobile Edge Computing Hosts Based on Extended Balas-Geoffrion Additive Algorithm," 2019 IEEE 2nd 5G World Forum (5GWF), Dresden, Germany, 2019, pp. 613-618.
- S. Thananjeyan, C. A. Chan, E. Wong and A. Nirmalathas, "Mobility-aware Energy Optimization in Hosts Selection for Computation Offloading in Multi-access Edge Computing," in IEEE Open Journal of the Communications Society, doi: 10.1109/OJCOMS.2020.3008485.

2 Literature Review

2.1. Requirements for Next-Generation Networks

Next-generation mobile networks are not only envisioned to enhance the traditional mobile broadband use cases but also aim to meet the requirements of new use cases, such as ultra-reliable and low latency communication and massive machine type communications [20]. Therefore, in addition to the classical mobile broadband traffic demands of high throughput and capacity, new requirements of achieving low latency and high reliability for many IoT use cases are very critical for next-generation networks such as 5G and 5G advance [11].

For example, the end-to-end latency should be around 1 millisecond for the tactile Internet [26] [27], and in the order of a few microseconds to a few milliseconds for industrial applications [10], and in the order of 100 microseconds for uploading or downloading in mobile networks. For example, some critical applications telesurgery in healthcare and traffic control in transportation [28] require near-real-time connectivity. The throughput requirements of these applications may vary widely from small amounts of data to large exchanges of media data transfer to and from the remote server. These requirements are mainly dependent on the application needs [11]. Furthermore, In some cases, applications such as autonomous automotive vehicles [29], extended reality applications including augmented and virtual reality (AR/VR) and robotic applications, which are essential for Industrial IoT (IIoT), may require both high data rates as well as ULL [30] [31]. In these cases, high data rates may be required for transferring live video streams from cameras that are used to control vehicles and robots to the remote servers for processing [32]. Hence, in such heterogeneous environments and applications, a mechanism to accommodate a diverse range of ULL requirements would be very helpful in implementing support for these applications [33].

2.2. Computation Offloading

On the other hand, computation offloading to mobile cloud computing is an existing paradigm to support resource-limited mobile devices. Computation offloading is a technique in which resource-constrained mobile devices fully or partially offload its computation-intensive tasks to a resource-rich cloud environment to be executed in the cloud environment and get the results back. Computation offloading is performed mostly to save energy which saves battery life in the mobile device, speed up the process of computation, or due to the incapability of the mobile device to process computation-intensive tasks [18]. Cyber foraging [34] is a pervasive computing technique, which enables computation offloading to enhance the capabilities of mobile devices, while significantly improving energy efficiency. Different cost models were proposed in the literature for resource monitoring and profiling [13] [35] [36] [37] of mobile applications. These cost models are then used to predict the cost, used for a cost-benefit analysis. Cost-benefit analysis is essential to evaluate the benefits of offloading [12] compared to execute in the local, i.e., in the mobile device. Based on the cost-benefit analysis, computation offloading decisions are made in mobile devices.

The experimental results to evaluate the effectiveness of computation offloading in [38] [39] validate that the computation offloading can save energy significantly in mobile devices. Hence, many mobile applications take advantages of computation offloading and remote processing to save energy in mobile devices. For example, computation offloading of a compiler optimization function in the image processing application [40] reduces up to 41% of the energy consumption of a mobile device. Further, using memory arithmetic unit and interface (MAUI) methodology to offload mobile game components [13] to servers in the cloud can save up to 27% of energy for typical games and 45% for computing-intensive games such as the chess game.

Computation offloading also reduces the running cost for compute-intensive tasks that take a long time to execute, in turn, a large amount of energy when performed on the resource-constraint devices. This leads to enormous computing-intensive mobile applications to be executed in cloud data centres using computation offloading technique. For example, cloud computing can be used for transcoding in real time [41], playing chess [42], or broadcasting multimedia services [43] in mobile devices. In these cases, all the

Table 2.1 Comparison of mobile edge computing and mobile cloud computing.

	Mobile edge computing	**Mobile cloud computing**
Resources	Data centres with limited resources (small scale data centre) [29]	Data centres with ample amount of resources (Large scale) [30]
Location	Co-located with wireless gateways, WI-FI routers or radio access network	Dedicated cloud data centres [31]
Distance	Proximity to end users (tens of hundreds of meters)	Far from end users (may across the continent)
Latency	Can support less than of milliseconds [32]	Can support larger than 100 milliseconds [33]
Use cases	Ultra-low latency and computation-intensive applications [34]	Latency-tolerant and computation-intensive application [35]

complex calculations are processed efficiently on the cloud using computation offloading technique.

In addition to the benefit in computation offloading, there are two major issues associated with radio access: power consumption and latency. These are the major bottlenecks in the usage of computation offloading in MCC in current mobile networks. In macro cellular systems, the power used for computation offloading from mobile devices is significant, especially if the mobile devices are in the edge of the cell. In some cases, this large power consumption may invalidate all potential benefits of computation offloading in terms of energy saving [14]. Thus, to avoid such a case, a possible way to reduce this power consumption is to bring computational resources proximity to the end users.

2.3. Multi-access Edge Computing

To reduce latency, core functions such as gateway nodes, along with the applications and services themselves, will need to be moved closer to the end users [16]. Software-defined networking (SDN) and network functions virtualization (NFV) are two trends in

Table 2.2 Comparison of edge computing architecture.

	MEC [34]	**Fog Computing [36]**	**Cloudlet [33]**
Proposed by	ETSI	Cisco	Prof. Satyanarayanan
Ownership	Mobile operators	Fog node owners	Local business
Location	RAN	In between mobile devices and data centres	In between mobile devices and data centers or directly in a device

mobile networking that are being considered for future mobile network architectures, which offer opportunities for lower latency in distributed topologies [44] [45].

Table 2.1 summarizes the comparison of the mobile edge computing and mobile cloud computing concepts in terms of available resources, location, distance, latency and use cases. To summaries, based on the characteristics of mobile edge computing, it is more suitable for ULL and computing intensive applications. Since mobile edge computing servers are deployed proximity to the end users, it has advantages of lower latency with constraint in computing resources.

2.3.1 Edge Computing Architectures

Three different edge computing architectures have been proposed in the literature: fog computing, cloudlet and multi-access edge computing. The basic variations in terms of ownership and locations are summarized in Table 2.2. All of these are similar concepts with very little variations in the usage scenarios and deployment location.

Fog computing is one of the edge computing paradigm originally proposed by Cisco with the intension of supporting mainly the future internet of things (IoT) applications [46] [47]. Fog computing uses edge devices such as switches and routers to carry out the computation offloading functionality in the wireless network, instead of separate servers in other architectures. The name fog computing is an analogy that the fog is closer to people than the cloud in nature. Correspondingly, the IoT device is closer to a fog

Table 2.3 Related works on use cases of MEC

Applications and use cases	**Related work**	**Key point**
AR/VR	[43]	Real-time processing, context-aware
Connected Vehicles	[44] [45] [46]	Traffic control, V2X communication
Video streaming and analysis	[47]	Redundant video avoidance
Internet of things	[48] [49] [50] [51] [52] [53]	Healthcare, smart grid, smart home, smart city

computing platform than to large-scale data centres located in the cloud, distance-wise. The proposed 2-tier architecture of fog computing deployment of IoT applications is not sufficient for the requirements of low latency, mobility, and location awareness [48].

On the other hand, the concept of Cloudlet is developed by an academic team from Carnegie Mellon University [49]. The cloudlet can be considered as an extension of the cloud in which a scale down version of a data centre is deployed in a box. It is self-managed, energy-efficient and simple to deploy on commercial sites such as a coffee shop or an office room. To further simplify the cloudlet deployment, one approach would be integrating cloudlet and WiFi access point into a single entity. On the other hand, since it can be deployed in different locations independently, managing installed cloudlets are challenging [50].

Multi-access edge computing (MEC) is proposed in [51] to provide communication, computing and storage facilities within the radio access network. MEC can yield sufficient capacities by harvesting the available compute resources at proximity to the users for performing computation-intensive and latency-critical computational tasks, that could not be effectively executed at the mobile device (MD) [52]. When MEC hosts are deployed in a hierarchical topology, the computational capabilities of MEC hosts vary depending on the deployment locations [53] due to the size and performance of the servers. For instance, more powerful servers can be deployed at the edge of the core

network than at a radio node. On the other hand, more network bandwidth can be allocated between the mobile device and the radio node than the mobile device and edge of the core network.

2.3.2 Use Cases and applications of MEC

Some of the experiment analysis from the literature related to the use cases and applications of MEC are summarized in Table 2.3. The MEC host with powerful processing ability can use local context information to open an enormous mobile applications opportunity which is very suitable for the AP/VR applications [54] and video analytics [55]. MEC network can play an important role in next-generation transportation technology including connected vehicles, V2X communication and automotive safety. Since MEC hosts are located proximity to the vehicles and can collect and analyze real-time data, it can provide roadside functionality such as traffic control and smart parking with ultra-low latency [56]. These can be achieved through sensor devices installed in smart city implementations [57].

MEC is capable of providing different services that would not be possible before [56]. For example, experiments conducted for the healthcare system in [58] proved that edge computing is faster and more energy-efficient than cloud computing. In addition, machine to machine (M2M) interactions in future networks can be enabled by deploying MEC hosts as gateways close to the smart devices [59]. It is also helpful to discover the incidence of abnormal events for time-critical events since it is much easier to exploit the contents generated by a large number of connected devices in the MEC network [60].

The work in [61] shows that the communication latency of offloading tasks can be reduced in the MEC network without affecting the network performance. The work carried out in [62] confirms that computation offloading in edge computing improves latency for highly interactive and compute-intensive applications such as augmented reality and cognitive assistance, in radio networks. Experimental results in [63] from radio networks show that computation offloading to cloudlets improves execution latency by 51% compared to offloading to the cloud.

2.3.3 Computation Offloading in MEC

The main objective of the researches in the literature focusing on the computation offloading decision is to minimize energy consumption at the mobile device while maximum tolerable latency constraint is satisfied, to minimize an execution delay, or to find an optimum trade-off between both the energy consumption and the execution delay.

The design objectives of the recent researches in [64] [65] [66] [67] are to minimize the computation delay in offloading. For instances, computation offloading of a multiuser video compression application is considered in [64] to minimize the latency in local compression and edge cloud compression considering partial offloading scenarios. The objective of minimizing execution delay in [67] is achieved by the one-dimensional search algorithm. The proposed algorithm finds an optimal offloading decision policy according to the current status of the mobile device such as buffer state, available processing powers and the conditions of the radio channel between the mobile device and the MEC host. When compared to the previous study, [68] reduces the application failures for the offloaded applications.

The different approaches are proposed in [69], [70] [71] for computation offloading decision making to minimize the total energy consumption at the mobile device while satisfying the maximum tolerable latency constraint of the application. A peer offloading framework is proposed in [69] for autonomous decision making. A Markov decision process (MDP) is formulated in [70] for the optimization problem. To solve the optimization problem, pre-calculated offline approach and an online learning approach are used. A further extension of [70] from a single-user to a multi-user scenario is considered in [71].

The papers in [68] [72] considers the energy consumption of mobile device based on dynamic voltage and frequency scaling (DVFS) and energy harvesting techniques as in [73] to minimize the energy consumption in computation offloading during the execution at the mobile device and data transmission over the radio network. Multi-user multi-channel environment is considered in the computation offloading decision making considering a trade-off between the energy consumption at the mobile device and the execution latency is proposed in [74] and [75]. These offloading decisions are made based on the weighting parameters assigned to preferences of minimizing energy consumption or execution delay.

2.3.4 Deployment of the MEC Network

The primary motivation of MEC deployment is to shift the cloud computing capability to the edges of the network to reduce the latency caused in the core network due to the congestion and propagation latency. Possibility of deployment at different hierarchical levels introduces new challenges to site selection problem of MEC hosts. This problem is different from the conventional base station site selection problems. since the optimal placement of MEC hosts is based on the computation resource demands, existing mobile network deployment and the deployment budget of the service providers [23]. Even though deploying MEC hosts at fewer locations can help to reduce the total rentals, this may lead to possible service degradations. The efficiency of a MEC system relies heavily on the workload demands and communication network architecture. In addition, MEC service providers should properly plan for the required hosts' density for supporting such service demands and infrastructure deployment cost and marketing strategies [23].

The site selection for MEC hosts deployment is the first step towards building up the MEC network. The MEC service providers should consider site rentals and computation demands to make the cost-effective MEC hosts site selection [23]. In general, MEC hosts with more resources should be deployed at regions with higher computation demands such as CBD and densely populated areas considering the deployment budget given. However, high site rentals are most likely in such areas.

MEC host selection

For users served by MEC system, a key design issue is to determine the suitable MEC host to serve the computation offloading request, i.e., either at the radio node or at the aggregation point or the edge of the core network. The server selection problem for a multiuser system with a single edge host and the central cloud is studied in [76]. A heuristic algorithm is proposed to maximize the total successful offloading probability by leveraging both the low communication latency due to the proximity of the MEC host and the low computation latency at the cloud data centre due to ample resource availability. In addition, MEC host selection problem over multiple hosts is explored in [77], which introduces a new challenge in the correlation between the amounts of the offloaded computation and selected MEC hosts. To minimize the total energy consumption of computation offloading, a congestion game is formulated and solved. A

computation offloading framework based on the semidefinite relaxation-based algorithm is proposed in [78].

Mobility management in MEC

Since MEC hosts are deployed proximity to the users, mobility of users may lead to multiple handovers, which may lead to migration of user application. Thus, mobility management in MEC is an important feature [79] which has been extensively studied for traditional heterogeneous cellular networks [80] [81] [82]. In these prior works, the connectivity probability or link reliability is used to model the user mobility. Based on such models, mobility management in the networks has been proposed to achieve high throughput and low error rate. However, these mobility management approaches cannot be directly applied for MEC networks, since they neglect the effects of the computational resources at edge servers in the hierarchical deployment.

Recent works in [83] [84] [85] [86] have made efforts to design a mobility-aware MEC network. Specifically, user mobility in [83] is modelled based on the inter-contact time and contact rate. Alternatively, the number of edge hosts that a user can access at a time was modelled by an independent homogeneous Poisson point process (HPPP) in [84].

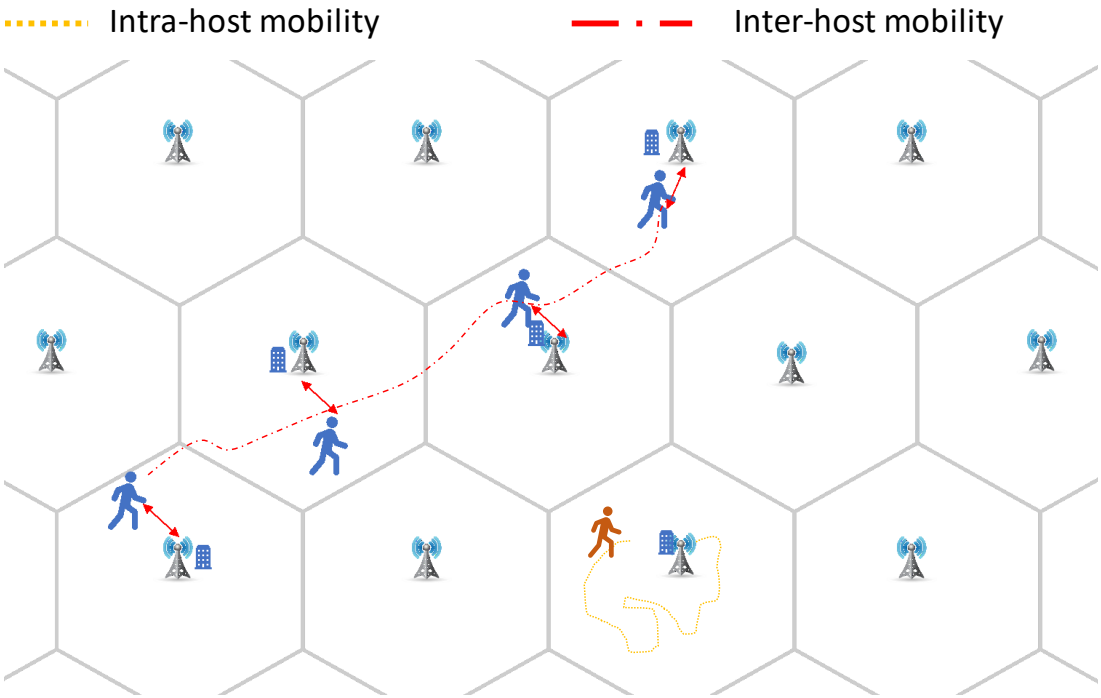

Figure 2.1 Impact of user mobility in MEC.

Based on this mobility model, a MDP problem is formulated to minimize the offloading cost which consists of mobile-energy consumption, latency and failure penalty. Two-dimensional spatial time mobility models were also proposed in [85] [86]. Thus, mobility predictions are a key factor to consider in designing a MEC network which is essential to maintain the service continuity.

2.4. User Mobility Predictions

Since different users move in different directions in the mobile network. A user could move to the coverage area of a different radio node that is still in the coverage of the serving MEC host creating a scenario of intra-host user mobility or to another radio node associated with a different MEC host creating a scenario of inter-host user mobility. In the case of the intra-host user mobility, the MEC system does not need to relocate the user application to maintain service continuity. Also, with the inter-host user mobility, if the service continuity can be maintained to meet the requirements of the application such as latency and bandwidth, a relocation of the user application from the serving host to a new target host is not required. In this situation, the traffic between the mobile device and the user application is then routed so that it reaches the intended destination as the user moves in the network. The above-mentioned cases are shown in Figure 2.1.

The availability of user movement traces that are collected from real-life user mobility or generated using simulation-based methods makes them possible to be analysed and explored in terms of the pattern of trajectories of MDs in mobile networks. Traditional mobile networks were mainly evaluated based on synthetic movement models, such as random waypoint (RWP) [87] or random walk models such as Brownian motion [88]. However, several research efforts such as [89] validated that user mobility is rarely random and that random models often fail to analyse the performance of mobile networks accurately.

On the other hand, most of the real traces were derived from bounded environments such as campuses or conferences using Bluetooth or Wi-Fi technologies [90] [91]. However, most of these realistic movement traces are neither scalable nor suitable to be used to measure the mobility of mobile users within a city scale. Location-based social networks (LBSN) are an alternative source of user mobility traces that are collected from online services [92]. In this method, users share their location information with their

friends by checking-into their visiting locations. Compared to the Bluetooth/Wi-Fi data, mobility traces that were collected using this method are reliable and scalable. Up till 2012, Gowalla (http://blog.gowalla.com/) application programming interface (API) allowed researchers to access their contents such as users' check-ins, social relationships and friend information. Thus, publicly available user mobility data is limited now. Map-driven mobility models extract movement features of real-world traces in order to reproduce scalable mobility traces using simulation methods synthetically. Further, the geometry of the map, i.e., the road network has a small impact on the mobility model, as shown in [93].

Simulation-based mobility models are alternative approaches to generate mobility traces synthetically [90]. Simulation-based mobility models are used for the evaluation of user-associated networking protocols due to the following reasons. First, a majority of real traces are environment-specific, i.e., they are collected in universities or conferences, are not scalable, and only confined within a small area. Second, they are not controllable and flexible to changing system parameters such as MD density and MD velocity. Third, publicly available traces are limited and often not available. These problems required researchers to seek alternatives such as the simulation-based modelling approach, in which the parameters of the mobility models vary according to the problem specifications. These models can be used to generate scalable and flexible mobility traces. Thus, synthetic models have been proposed to capture the movement patterns of nodes, e.g., MDs in a realistic way.

Recent literature reports that humans tend to perform Lévy walks [94] [95] [96] with heavy-tail flight distributions. A Lévy walk is a random walk whose step-lengths have a heavy-tailed probability distribution. Intuitively, the Lévy walks consist of many short flights and, occasionally, long flights; here, a flight is defined as the longest straight-line trip of a human from one location to another without a directional change or pause. The truncated Lévy walk mobility (TLW) [97] model is constructed to study the impact of heavy-tail statistical features on the performance of mobile networks.

On the other hand, in reality, human mobility tends to be correlated and strongly dependent on users' personal and social characteristics and behaviours as well as environmental parameters [91]. For example, an experimental analysis presented in [98] demonstrates that users frequently visit locations with which they have strong social ties.

Furthermore, mobile users tend to visit just a few locations where they spend the majority of their time [99].

2.5. Energy Efficiency in MEC

Energy efficiency in data centres is critical since it consumes much energy as 25,000 households. In addition, data centre spaces may consume up to 100 to 200 times as much electricity as standard office space [100]. In addition, the energy costs of typical data centres double every five years [101]. Therefore, electricity costs have become a significant expense for today's data centres [100] [102] with the steep increase in electricity use and rising electricity costs. Electricity costs, in some cases, may exceed the hardware purchasing cost [103]. On the other hand, data centre energy usage causes environmental problems [104] [105]. The global electricity usage by data centres was estimated at around 1.5% of the total worldwide electricity usage [106] in 2010. Due to these reasons the energy efficiency of cloud data centres has gained key importance in research [107].

Finally, it is important to note that servers consume a significant amount of energy even when they are running in the idle mode. Thus, large savings can be possible by appropriately turning off these servers if they are not currently required. This important aspect and workload intensity need to be considered to reduce data centre electricity usage. On the other hand, these power-saving techniques may reduce system performance, which leads to a trade-off balance between energy savings and high performance.

Different server power consumption modelling approaches are proposed based on the component-wise breakdown of server power [108] [109] [110] [111]. Server power is modelled as a sum of CPU and memory power consumption in [108]. Motherboard energy consumption is added in [109]. In addition, the power of sub-components such as disks, network peripherals is added in [110] [111]. Some researchers model server power consumption based on the assumption that server power consumption and CPU utilization has a linear relationship [112] [113] [114]. CPU is one of the largest power consumers of a server [102]. Processor power consumption can be modelled as static power consumption and dynamic power consumption at a higher level, similar to system power consumption. Processor power modelling based on statistical approaches is presented in

[115] [116] [117]. It has been observed that processors with a low utility will consume too large static power compared to dynamic power.

Since MEC hosts are data centres in small scales, each of these consumes considerably less energy than the cloud data centre. However, their dense deployment in MEC network raises a big concern on the system-wide energy consumption [23] in the MEC network. Therefore, it is very important to develop innovative energy-efficient techniques to achieve high energy efficiency in the MEC network [118].

Switching off/slow down the processing speed of edge hosts with light computation loads is one way to realize energy efficiency which is termed as dynamic right-sizing in the literature [119]. The predicted dynamic workload profile of MEC host is an important metric to consider in order to make an effective decision on dynamic rightsizing.

Geographical load balancing is another key technique for green data centres [120] [121] which can be applied for the MEC network as well. It uses the workload patterns of data centres and electricity prices to make workload routing decision among data centres. Dynamic right-sizing and VM management [122] [123] [124] [125] are required in implementing load balancing at edge servers. Energy savings are achieved by continuous consolidation of VMs according to current utilization of resources, virtual network topologies established between VMs and thermal state of computing nodes [122]. A greedy algorithm is proposed in [123] for VM placement problem which is an NP-hard. A VM allocation mechanism with spatial/temporal awareness is proposed in [124].

Migrating VMs during user mobility is another critical challenge in MEC. Minimizing system cost in VM migration is studied in [126] [127]. The VM migration problem was formulated as an MDP problem based on a random-walk mobility model in [126]. Lyapunov optimization techniques are used to solve the minimization problem of average transmission costs in [127]. Minimizing transmission delay in VM migration is analysed in [128] [129] [130]. A path selection problem between the user device and MEC host is modelled in [128] [129] as Markov decision process considering transmission delay and energy consumption in which user mobility is modelled as Manhattan mobility model. A mobility prediction based on real-world mobility trace is considered for VM placement problem in [131]. Two-dimensional random walk model is considered in [132] for VM

migration problem which is formulated as MDP and solved using linear programming reformulation.

To summaries, since the design and development of the MEC network is in the infant stage, there are critical challenges which need to be solved before the real-world implementation of the network. This thesis addresses the deployment and energy efficiency challenges in the MEC network as in the subsequent Chapters.

3 Deployment and Resource Distributions of MEC Hosts

In order to design multi-access edge computing network, computation offloading demand from the end-users needs to be analysed first. Then based on the demand, suitable locations need to be selected for MEC host deployment. This chapter analyses the computation offloading demands from mobile users based on their mobility pattern. Then, it proposes methodologies to select suitable locations to deploy MEC hosts and allocate computing resources.

3.1. Introduction

MEC hosts are entities that contain the computing and storage resources deployed in a distributed manner throughout the radio access network. Those are usually resource-constrained when compared to the cloud data centre and could potentially be costly due to the substantial number of distributed servers that are required to be deployed at various locations of the network. Furthermore, it is expected that mobile users in the future mobile networks will be offloading computing-intensive tasks to these hosts during their commute. Thus, the user's mobility will influence the service continuity of computation offloading in MEC due to the proximity to the end-users.

To incorporate the impact of mobility patterns, appropriate models of user mobility is required for the design of MEC. Many different user mobility models such as the random waypoint model [87] and Lévy walk models [97] have been proposed in the literature, mostly producing or emulating trajectories of a random walk. However, most of these user mobility models were based on real traces obtained from relatively small and bounded environments and are not suitable to design large scale MEC. Besides, real-life examples from the trajectory data from the Oyster Card system of London [133] and the vehicle-location data from the Massachusetts Bay Transportation Authority (MBTA's) Charlie Card for fare-payment [134] have shown that user mobility is correlated in morning and evening peak hours during which users move toward and leave the central business district (CBD), respectively, showing strong diurnal patterns.

Correlated movement of users during peak hours is expected to have a high impact on the service continuity of MEC networks. To the best of our knowledge, resource distribution of MEC systems based on user mobility in an environment with resource constraints has not yet been fully investigated in the literature, thus becomes the focus of this Chapter. In particular, how to select the best locations for MEC host deployment and how to distribute the available resources to the MEC hosts based on user mobility patterns during peak hours to maximize the number of offloaded tasks accomplished, have yet to be investigated.

In this Chapter, the maximization problem with regard to *how the total number of offloaded tasks accomplished in the MEC system can be maximized with the limited number of CPU resources available while considering user mobility* is formulated. In other words, how much CPU resources should be allocated to MEC hosts in order to maximize the total number of tasks accomplished in the system by considering the user mobility patterns when the total number of CPU resources are limited. Only the offloading requests during the morning peak hours are considered in this work. However, the resource distribution can be extended to reflect the offloading requests during other periods.

In summary, the main contributions in this Chapter are the following: Correlated user mobility model is developed to produce user mobility traces during morning peak hours. Task request profiles are developed to understand the user demands based on the deadline requirement of the application such as hard deadline and soft deadline. Utilitarian resource distribution algorithm is developed to identify the suitable locations to deploy MEC hosts and the amount of resources needed to be allocated in each MEC host. Extensive simulations are carried out to verify the proposed solutions.

The remaining sections of this Chapter are organized as follows. Section Related Works in Literature3.2 introduces the related work in resource allocations in MEC. Section 3.3 introduces the proposed correlated mobility model and resource distribution algorithms. Section 3.4 discusses the simulation and results of the proposed models and Section 3.5 provides a summary of the main Chapter.

3.2. Related Works in Literature

MEC hosts can be deployed at a radio node or at an aggregation point in which multiple radio nodes are connected and aggregated in the network hierarchy and/or at the edge of the core network [21]. Such deployment scenarios are subject to the cost budget, traffic mix and demand patterns, and the service provider's preferences. When a MEC host is deployed at an aggregation point or the edge of the core network, radio node information such as channel bandwidth allocation of the user and current traffic at the radio node may need to be retrieved from the radio node to the MEC host. On the other hand, the computing capabilities of MEC hosts vary depending on the deployment location [53] due to the size and the capacity of the servers. For instance, more powerful servers can be deployed at the edge of the core network than at an aggregation point.

Three main categories have been identified for use cases of MEC: consumer-oriented services, operator and third-party services, and network performance and quality of experience (QoE) improvement [21]. Consumer-oriented services are a direct benefit to end-users (e.g. XR applications such as AR/ VR application, gaming). Different use cases in consumer-oriented services have different requirements in terms of latency, bandwidth, and computing resources. In some use cases such as certain medical applications (e.g. real-time remote surgical operations, U-fall smart health care infrastructure [135]), the latency requirement is crucial, i.e., *hard deadline*. In other words, for an application with tasks having a hard deadline requirement, the tasks should be executed on or before the deadline requirement of the tasks. In many other cases such as AR and VR applications (e.g., VR visitor guideline applications of a museum) etc., the delay is relatively soft. Thus, the latency requirement is flexible in which the task is allowed to be executed beyond the deadline i.e. *soft deadline*. Due to a mix of different applications and their diverse requirements, the MEC system will need to support the offloading requests to meet both soft and hard deadlines.

Computation offloading in multi-access edge computing has been studied in recent literature. The performance of computation offloading was investigated in the context of IoT devices within a resource-intensive 3-D application[136]. A queueing theory-based approach was proposed in [137] to solve the multi-objective optimization including energy consumption, execution delay, and payment cost of computation offloading in edge computing. Furthermore, task caching and offloading problems were formulated and

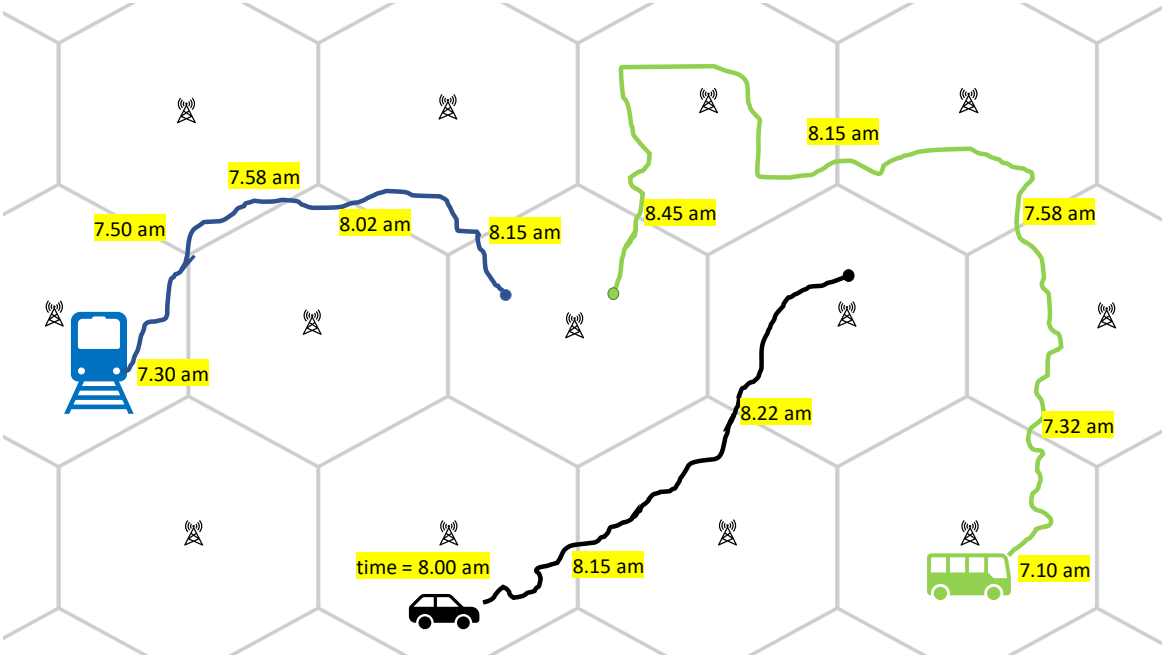

Figure 3.1 Estimated transit time of three representative users commute using public transport during morning peak hours.

solved as a mixed-integer problem in [138]. Joint computation offloading and resource allocation optimization were decomposed and then solved using the game-theoretical approach in [139]. However, the resource distributions considering the mobility of the users in a limited resource environment has not been considered in any of the previous studies.

On the other hand, user mobility is a critical factor in analyzing computation offloading demands. As the user moves further away from the serving host, there could be an increased latency in serving the user via the original serving host. Due to potential network congestion, it might become necessary to relocate the user application state or user application instance from the serving MEC host to the target MEC host in order to provide service continuity. User application relocation due to user mobility depends on MEC host deployment options and the network topology.

User application relocation failure may occur due to late relocation or early relocation or relocation to a MEC host with higher latency. Reducing the relocation failure rate is the key to improving the quality of experience (QoE) [79]. For instance, if a user commutes to work on the public transport while offloading tasks from a mobile device, the main concern is the possibility of a rather late relocation due to the MD's high velocity, which affects QoE of the user. Since user mobility is unavoidable in a mobile system, if

Algorithm 3.1 Algorithm to derive user trajectories based on correlated mobility

Input: Geographical area, CBD location(s)
Output: User mobility trajectories
Parameters:
For pause time: $p \in (0,1)$, $\alpha_p \in (0,1)$, $\beta_p > 0$, $l_p > 0$;
For flight length: $\alpha_f \in (0,1)$, $\beta_f > 0$, $l_f > 0$;
For direction change: $\alpha_d > 0$, $\beta_d > 0$;
For correlated coefficient $cc \in (0,1)$;
For speed: $max\_speed > min\_speed > 0$.

Begin:

- Choose a starting location
- Choose a destination location
- Simulate a direction φ uniformly at random
- Simulate a flight length x from $TS(\alpha_f , \beta_f , l_f)$
- Choose a speed v uniformly between *min_speed* and *max_speed*
- Move in direction φ a distance of x at the selected speed v
- **While** *Not Arrive at destination* **do**
    - With probability p simulate a random time from $TS(\alpha_p, \beta_p, l_p)$ and wait this amount of time.
    - Simulate x from $TS(\alpha_f , \beta_f , l_f)$
    - Choose a speed v uniformly between *min_speed* and *max_speed*
    - Simulate θ_s from $beta(\alpha_d, \beta_d)$
    - Derive the θ_d the direction to the destination from current location
    - Calculate direction $\theta = cc \times \theta_d + (1 - cc) \times \theta_s$
    - Move in direction θ a distance of x at the selected speed v

the user mobility information is available beforehand, then the MEC system can proactively predict the handover timing and guarantee seamless service continuity and smooth relocation of offloading tasks to optimal MEC host to maximise QoE.

Figure 3.1 shows an example of the prediction of handover timing for users commuting to work in the morning using different modes of transportation. The transit time in each radio node can be estimated by the accurate prediction of user mobility and the current locations by retrieving information of MDs and radio nodes. Hence, successful prediction of users' future locations is the main key to improving the QoE of MEC application

relocation. Mobility traces of users have been extensively analysed in order to gain insight into user mobility patterns and to forecast their next locations accurately. Recently, several user mobility prediction methods have been proposed to demonstrate user mobility. Researchers have found that user mobility can be predicted up to 93 % accuracy [25].

Recent literature reports that humans tend to perform Lévy walks [94] [95] with heavy-tail flight distributions. A Lévy walk is a random walk whose step-lengths have a probability distribution that is heavy-tailed. Intuitively, the Lévy walks consist of many short flights and, occasionally, long flights; here, a flight is defined as the longest straight-line trip of a user from one location to another without a directional change or pause. The truncated Lévy walk mobility (TLW) [97] model is constructed to study the impact of heavy-tail statistical features on the performance of mobile networks.

On the other hand, in reality, user mobility tends to be correlated and strongly dependent on users' personal and social characteristics and behaviours as well as environmental parameters [140]. For example, an experimental analysis presented in [98] demonstrates that users frequently visit locations with which they have strong social ties. Furthermore, mobile users tend to visit just a few locations where they spend the majority of their time [141].

3.3. System Models and Methodologies

As user mobility patterns tend to be correlated as discussed in the previous section, here, the correlated mobility of users is modelled with modifications from the smoothly truncated Lévy walk proposed in [96].

3.3.1 Correlated Mobility Model

The correlated mobility algorithm is shown in Algorithm 3.1. Each MD starts at a randomly selected location in the simulation area. It selects a random area around the CBD of the simulation area as the destination. Then it moves a flight length in a direction toward the destination at a randomly chosen speed between some predetermined minimum and maximum values. This speed indicates the mode of transportation. It stays for a pause time after each flight length and continues moving toward the destination. The

pause time reflects the time spent on traffic signals and traffic congestion. If it reaches the destination, it stays there. The direction of movement of each flight length is calculated as the weighted average (or correlated coefficient) of the destination direction and the random direction. The destination direction is the direction between the current location and the destination location. Flight lengths, pause times, and random directions are selected from the tempered stable distribution TS(α_f, β_f, l_f), tempered stable distribution TS(α_p, β_p, l_p), and the beta distribution beta(αd, βd), respectively, as in [96].

3.3.2 MEC System Model

Performance metrics such as latency, energy efficiency, throughput jitter, and Quality-of-Service (QoS) are key parameters for evaluating the effectiveness of a MEC system. These time-variable metrics can be measured within a defined time interval and described by a profile over time or summarized through maximum, minimum, mean, standard deviation or the value of given percentile [52]. Latency is measured as the time interval between any event and a consequent target effect. The Round-Trip Time (RTT) is the time taken for a request from a MD to go to the server at the MEC host, be updated or replied, and travel back to the same MD. RTT assumes ideal service capability conditions, i.e., RTT does not depend on the server computational load.

Service processing time (SPT) is the time taken by the server to process a user request, which depends on the computational load. SPT is the time taken by a task between arriving at the MEC host to being executed, which includes queueing and server execution times. As the computing resources are limited in the MEC hosts when compared to the ample resource availability in the cloud data centres, SPT is non-negligible. Service delivery time (SDT) is the time taken for a user request to reach the MEC host, be processed, and reach back at the MD. Thus, the SDT of a task is defined as:

$$\tau_{SDT}^{h} = \tau_{RTT}^{h} + \tau_{SPT}^{h} \tag{3.1}$$

where τ_{SDT}^{h} is SDT of the task, τ_{RTT}^{h} is RTT and $\tau_{SPT}^{h}$is the SPT. RTT, SPT, and SDT depend on the selected MEC host to offload the task.

The computation task of a user application can be represented by $T \triangleq (r^C, r^N, \tau^M)$ where r^C is required CPU cycles to accomplish the task in CPU cycles, r^N is the size of the input data in bits and τ^M is maximum tolerable latency or the deadline requirement of

the task in ms. It is assumed that only one MEC host is selected to execute an offloaded task to avoid data coordination among MEC hosts, i.e., parallel execution of an offloaded task in multiple MEC hosts is not considered in this Chapter.

The target service processing time (TSPT) is the maximum allowed SPT to accomplish the task. A task is considered as accomplished when the task is executed and delivered before the deadline requirement of the task, i.e., ($\tau_{SDT}^{h} \leq \tau^{M}$). TSPT is, therefore, calculated as

$$\tau_{TSPT}^{h} = \tau^{M} - \tau_{RTT}^{h} \tag{3.2}$$

The minimum required CPU resources of a task at the MEC host is the minimum average CPU resources needed to be allocated to accomplish the task. Thus,

$$r_{REQ}^{C,h} = \frac{r^{C}}{\tau_{TSPT}^{h}} \tag{3.3}$$

However, $r_{REQ}^{C,h}$ may not be available at the MEC host due to resource limitations. Thus, a different amount of CPU resources may be allocated to a task depending on the CPU resources available at the MEC host. The SPT of a task is calculated based on the average CPU resources allocated during SPT.

$$\tau_{SPT}^{h} = \frac{r^{C}}{r_{ALLOC}^{C,h}} \tag{3.4}$$

where $r_{ALLOC}^{C,h}$ is the CPU resource allocation (the average amount of CPU resource allocated during the SPT). Further, the total allocated CPU resources of a MEC host should not exceed the available CPU resource at the MEC host at a specific time, i.e., $\sum_{u} r_{ALLOC}^{C,h} \leq x_h$. The remaining CPU resources at the MEC host after the allocations at a specific time can be calculated as

$$r_{REM}^{C,h} = x_h - \sum_{u} r_{ALLOC}^{C,h} \tag{3.5}$$

The *minimum CPU resource requirement of the MEC host* is the minimum amount of CPU resources needed to be allocated to accomplish all the offloaded tasks at the MEC host. The *accepted utility of the MEC host* is the total number of tasks accepted by a MEC host for the allocated CPU resources over a time period, i.e., how many offloaded tasks of

hard deadline requirement applications have been executed and delivered on or before the deadline requirement of each task for the allocated CPU resources of a MEC host over a time period. Thus, if the minimum CPU resource requirement of the MEC host is allocated at a MEC host, then the accepted utility of the MEC host is 100%. It can be assumed that if the CPU resource allocation is increased, the total number of tasks accomplished will increase until all the tasks requested during the period are accepted. There will be no change in the total number of tasks accomplished beyond this allocation. Thus, allocating CPU resources to a MEC host beyond the minimum CPU resource requirement of the MEC host in a resource-constrained environment is not acceptable.

3.3.3 Hard Deadline Requirement

When a task is requested for offloading and the hard deadline requirement of the offloading task request can be met, the task will be accepted for processing at the MEC host. In other words, if the minimum required CPU resources of a task as in Eq. (3.3) are available at the MEC host, it is assumed that the same amount of resources will be allocated and that the task request will be accepted and processed at the MEC host. Otherwise, the task request will be rejected, and no CPU resource will be allocated.

$$r_{ALLOC}^{C,h} = \begin{cases} r_{REQ}^{C,h} \; if \;\; r_{REQ}^{C,h} \leq r_{REM}^{C,h} \\ 0 \; otherwise \end{cases} \tag{3.6}$$

Thus, the task acceptance profile (θ^h) of hard deadline requirement applications is

$$\theta^h = \begin{cases} 1 \; if \;\; r_{REQ}^{C,h} \leq r_{ALLOC}^{C,h} \\ 0 \; otherwise \end{cases} \tag{3.7}$$

θ^h will be 1 if the task can be accepted (thus accomplished), otherwise it will be 0. The total number of tasks accepted in each MEC host is $\sum_u \sum_n \theta^h$. The objective of the hard deadline requirement is to maximize the total number of tasks accepted in the MEC system, subject to the limitation of the total available CPU resources. Thus,

$$Max_{\{x_h\}} \sum_h \sum_u \sum_n \theta^h \tag{3.8}$$

Subject to

$$\sum_h x_h \leq R$$

where R is the total CPU resources available in the MEC system. Thus, our goal is to maximize the total number of hard deadline requirement tasks accepted in the MEC system by distributing the available total CPU resources to the MEC hosts (x_h) based on user mobility and the task offloading request patterns. The maximization problem in Eq. (3.8) solves the resource distribution problem that provides the best locations to deploy MEC hosts.

The *total tasks acceptance ratio* is the ratio of the total number of tasks accepted to the total number of task offloading requests in the hard deadline requirement case. Maximizing the total number of tasks accepted in Eq. (3.8) yields a maximization of the total tasks acceptance ratio. The total tasks acceptance ratio is a metric that can be used to evaluate the performance of the MEC system in the hard deadline requirement use case, i.e. the readiness of the system to support the hard deadline requirement of a MEC application. MEC service providers can use this metric to evaluate whether their MEC system supports the hard deadline requirement of the user application.

3.3.4 Soft Deadline Requirement

In some user applications such as the VR visitor guideline applications of a museum, the task completion deadline can be flexible, i.e., MEC system can serve the tasks on an extension to the required deadline. In case of soft deadline requirement applications, all the task offloading requests are queued for processing in the MEC host. If the minimum required CPU resources of a task as in Eq. (3.3) is available at the MEC host, it is assumed that the same amount of CPU resources will be allocated; otherwise, the remaining CPU resources will be allocated during the SPT of the task.

$$r_{ALLOC}^{C,h} = \begin{cases} r_{REQ}^{C,h} \; if \; r_{REQ}^{C,h} \leq r_{REM}^{C,h} \\ r_{REM}^{C,h} \; otherwise \end{cases} \tag{3.9}$$

Further SPT of soft deadline requirement tasks can be calculated based on the resources allocated in Eq. (3.9) and Eq. (3.4). The task service profile (μ^h) of soft deadline requirement application is

$$\mu^h = \begin{cases} 1 \; if \; \tau_{SPT}^h \leq \tau_{TSPT}^h \\ 0 \; otherwise \end{cases} \tag{3.10}$$

μ^h will be 1 if the task can be accomplished (thus serviced within the deadline requirement), otherwise it will be 0. The objective of the soft deadline requirement is to maximize the total number of tasks serviced in the MEC system. Thus,

$$Max_{\{x_h\}} \sum_h \sum_u \sum_n \mu^h \tag{3.11}$$

Subject to

$$\sum_h x_h \leq R$$

Algorithm 3.2 Utilitarian resource distribution algorithm.

Input: T^h_{PROF} - Total tasks request profile of proposed MEC hosts, R – total available CPU resources for allocations in the MEC system, k – number of step size

Output: $R^{C,h}_{ALLOC}$ - CPU resource allocations of each MEC host, R^C_{REM} – total remaining CPU resources in the MEC system

- Calculate $R^{C,h}_{REQ}$ - minimum CPU resource required to accomplish all the tasks in each MEC host.
- Set $R_{REM} = R$, $R^{C,h}_{ALLOC} = 0 \; \forall \; h$, $T^h_{ALLOC} = 0$ (the number of tasks accomplished in each MEC host for the resource allocation $R^{C,h}_{ALLOC}$)
- Let *step_resource* = $\frac{\min(R^{C,h}_{REQ})}{k}$
- **While** $R_{REM} > 0$ **do**
  - Set $R^{C,h}_{NEW} = R^{C,h}_{ALLOC}$ + *step_resource* $\forall \; h$
  - Calculate expected total number of tasks accomplished (T^h_{NEW}) based on deadline requirement, $R^{C,h}_{NEW}$ and T^h_{PROF}
  - Find the MEC host h^* which accomplish more tasks for step size increment (randomly select one if there are many) and add the new resource
    - **if** $R^{C,h^*}_{REQ} > R^{C,h^*}_{ALLOC}$ + *step_resource*
      - **then** *add_resource* = *step_resource*
    - **else** *add_resource* = $R^{C,h^*}_{REQ} - R^{C,h^*}_{ALLOC}$
    - $R_{REM} = R_{REM} - add\_resource$
  - If $R^{C,h}_{REQ} \leq R^{C,h}_{ALLOC} \; \forall \; h$ **break**;

where R is the total compute resource available in the MEC system. The *total task service ratio* is the ratio of the total number of tasks serviced within the deadline to the total number of tasks requested in the soft deadline requirement case. The total task service ratio metric can be used to evaluate the performance of the MEC system under different deployment scenarios.

3.3.5 Resource Distribution Algorithm

The MEC service provider may distribute the CPU resources based on the ratio of the demand in each MEC host to maximize the total number of tasks accomplished, in turn, to maximize the QoE in the MEC system. Demand might be calculated based on the mean or maximum of task request profiles of the MEC hosts. Resource allocations based on maximum demand capture the peak demands during the morning peak hours. On the other hand, resource allocations based on mean demands captures the average demand during the period. Our proposed utilitarian resource distribution algorithm is compared with these two resource allocation schemes.

However, a utilitarian resource distribution algorithm () has been proposed in this Chapter to solve the maximization problems in Eq. (3.8) and Eq. (11). MEC service providers can propose their preferred locations to deploy the MEC hosts based on the facilities available in the radio nodes. Then the total task request profile of each proposed MEC host locations can be estimated. MEC service providers can calculate the total available CPU resources for allocations in the MEC system based on their allocated budget. The above algorithm distributes the total CPU resources to the MEC hosts to maximize the total number of tasks accomplished.

The utilitarian resource distribution algorithm allocates CPU resources step by step in a MEC host that accomplishes more tasks than the other proposed MEC hosts during the same step increment of CPU resources. MEC service providers can select the number of step size (k) which is a constant. Then the step resource increment is calculated as the minimum of minimum CPU resource required in all MEC hosts divided by the selected step size. When the number of step size increases, step resource decreases which increase the accuracy of the resource allocation. On the other hand, it increases the time taken to find the solution.

The algorithm finds a MEC host that could accomplish more tasks than others during each step size increment of CPU resources. Then it adds the step size resource to the selected MEC host. The total task request profiles are used to calculate the total number of tasks accomplished for the given CPU resources of the MEC host. However, the CPU resource allocations as in (6) and (9) differ for hard deadline requirement applications and soft deadline requirement applications, respectively. Thus, the remaining resources at the MEC hosts also differ, which leads the total number of tasks accomplished to also differ.

If the total resource allocated in a MEC host exceeds the minimum required CPU resources of the MEC host, only the minimum required CPU resources will be allocated. If multiple MEC hosts could accomplish the same number of tasks for the step size increment of the CPU resources, a random MEC host is selected. The algorithm loops

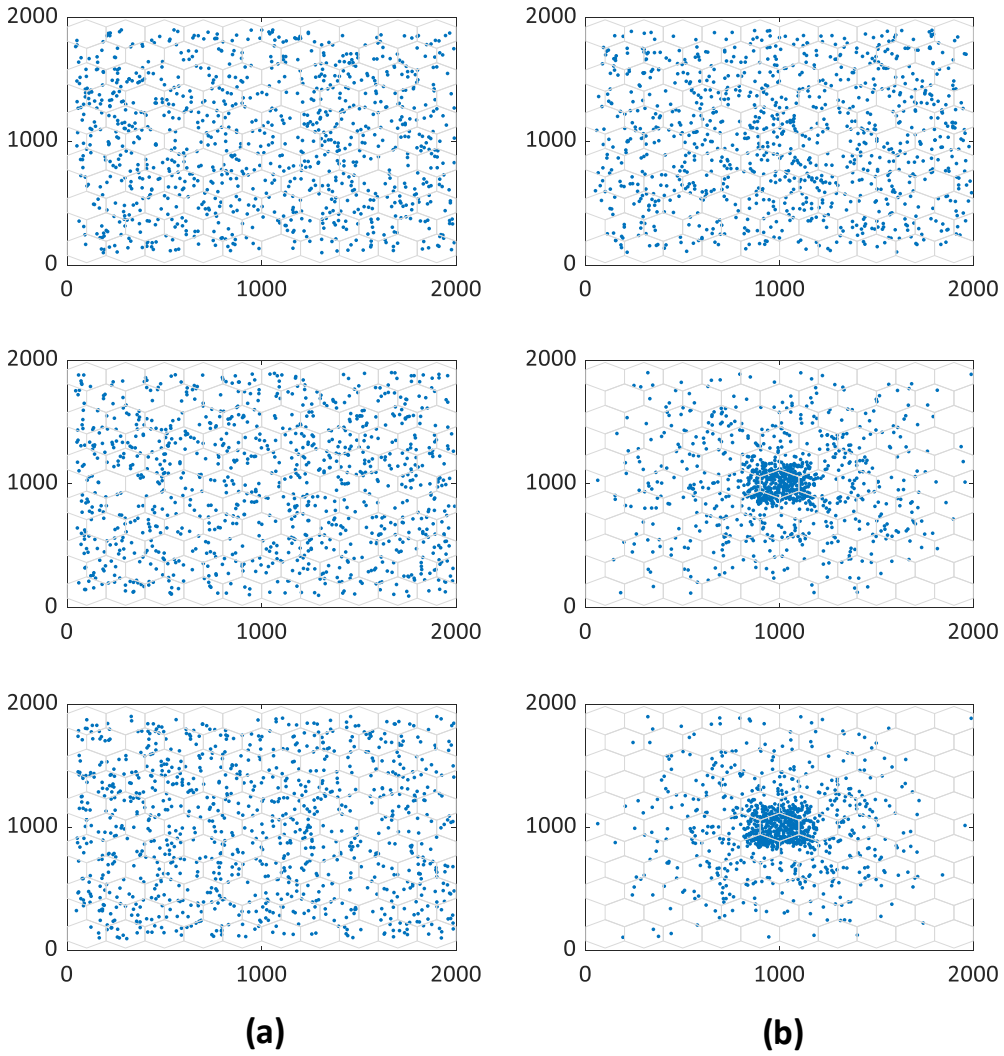

Figure 3.2 Users' locations as a function of time: (a) random mobility, (b) correlated mobility

through each step size until all the given resources are allocated or the minimum required resource of each MEC host is allocated. When the algorithm terminates, the output will be the resource allocations of each MEC hosts and the remaining resources. These results can be used by the service providers to solve the deployment and resource distribution problems of a new MEC system. On the other hand, if the required amount of CPU resources is readily available in the existing MEC system as per the results, then the MEC service providers can ensure the service continuity of the moving users. Thus, the results of the utilitarian algorithm can be used to evaluate the quality of service of the existing MEC system for the given mobility pattern and tasks offloading request pattern.

There may be some MEC hosts with no allocations, which indicates that the proposed locations are not selected for the MEC host deployment. In contrast, the resource distribution based on the mean or maximum of resources required selects all the proposed locations and then distributes the CPU resources. Thus, our utilitarian algorithm selects the best locations for the MEC host deployment and the amount of CPU resources needed to be allocated in the selected MEC hosts while maximizing the total number of tasks accomplished according to the given mobility and tasks offloading patterns.

Overall, the utilitarian algorithm divides the total available resources into multiple step resources to solve the maximization problem starting with a small resource constraint. It finds the local maximum of the optimization problem step by step by increasing the step resources. The local maximum of step by step process leads to the global maximum of the optimization problem. Thus, the original optimization problem is solved by achieving global maximization through step by step local maximization.

Utilitarian resource distribution algorithm depends on three inputs: total tasks request profile of proposed MEC hosts, total available CPU resources and the number of step size. Minimum CPU resource required ($R_{REQ}^{C,h}$) calculation depends on the number of proposed MEC hosts. The while loop in the algorithm depends on the number of step size. The expected total number of tasks (T_{NEW}^{h}) calculation depends on the number of proposed MEC hosts. Since T_{NEW}^{h} operation is inside the while loop, it is the predominant operation among the other operations in the algorithm. Thus the time complexity of the algorithm becomes $O(mn)$, where it depends on the number of MEC hosts and the number of step size.

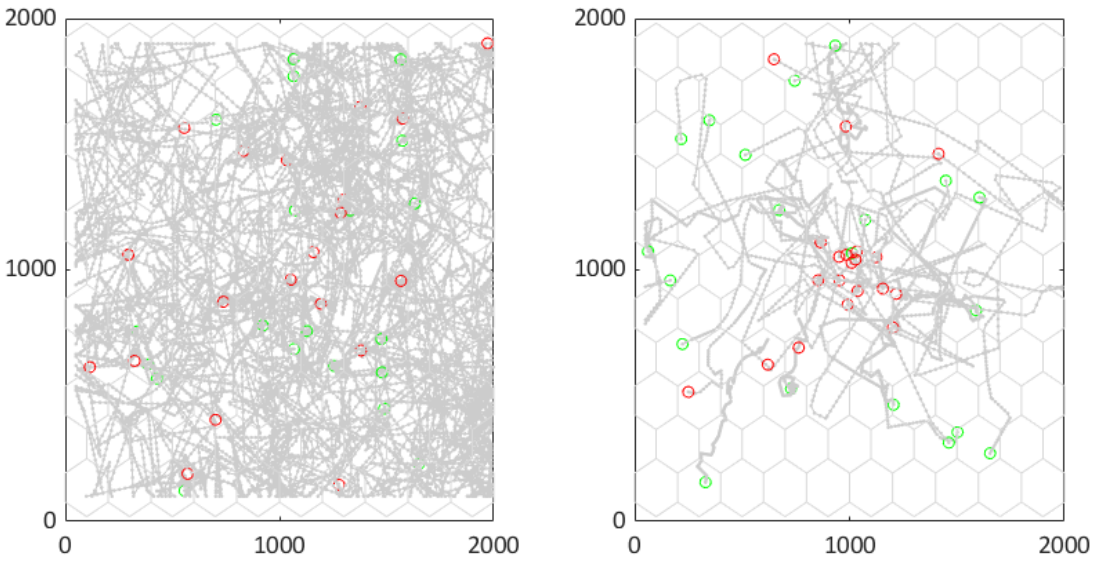

Figure 3.3 User trajectories: (a) random mobility, (b) correlated mobility

3.4. Simulation and Results

An urban area of 2 km × 2 km served by a mobile wireless network in which radio nodes are spaced 200m apart (picocells) is assumed in our simulation. Within that 4km² area, a total of 1000 users are assumed, each with a MD and moving in vehicles for an hour, i.e., 3600 seconds during the morning peak hour rush.

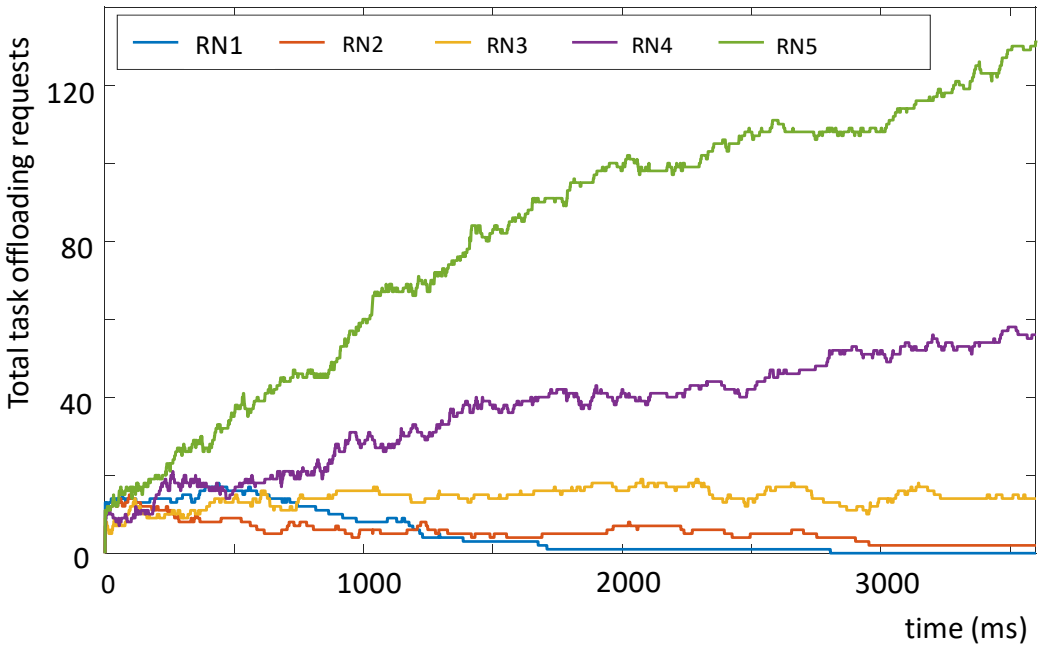

Figure 3.4 Total tasks offloading requests profile for selected MEC hosts

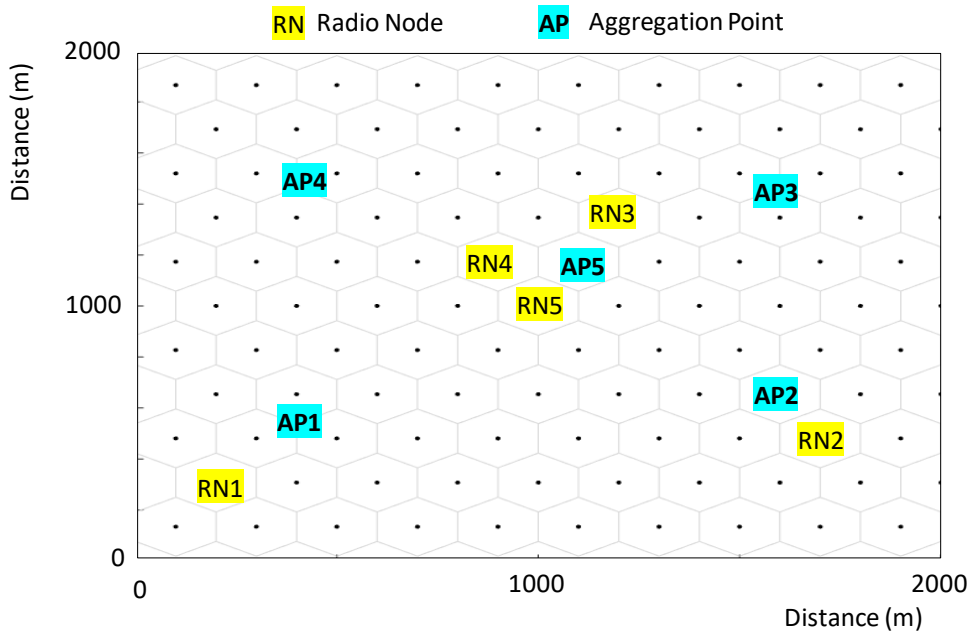

Figure 3.5 Selected MEC hosts for results presentation

3.4.1 Mobility Model

To test both the smoothly truncated Lévy walk (for random mobility) and the proposed correlated mobility model, the minimum and maximum speeds of mobility are chosen to be 5 kmph and 100 kmph, respectively. The different speeds correspond to different modes of transportation such as bus, taxi, train etc.

Figure 3.2 shows the users' locations as a function of time. Figure 3.2(a) illustrates random mobility (modelled using smoothly truncated Lévy walk [28]) and Figure 3.2(b) shows the correlated mobility (modelled using our proposed correlated mobility model). The users' trajectories (shown in grey) of a randomly selected 20 users (out of 1000 users in the simulation) for random and correlated mobility models are shown in Figure 3.3(a) and Figure 3.3(b), respectively. Note that the green circles represent the origins of the trajectories and the red circles represent the destinations of the trajectories. The user trajectories of the correlated mobility model better describe the morning peak hour mobility as presented in the Oyster data [6] and MBTA's Charlie fare-payment card data [7] as well. Figure 3.2 and Figure 3.3 are important to evaluate the effectiveness of our proposed utilitarian resource distribution algorithm that depends on task request profiles. Task request profiles vary according to user mobility and offloading demands.

3.4.2 Task Request Profiles

The uniform workload test as suggested in [52] is carried out for the resource distribution of the MEC system. This test assumes that every second, each MD will request to offload a computation task that requires 100 Mega CPU cycles with a 14 millisecond (ms) deadline. This latency requirement is chosen to minimize motion sickness of humans in VR/ AR applications that require the SDT to be below 20 ms. Further, 1 ms sensor sampling delay and 5 ms display refresh delay leaves only 14 ms for a computing and communication delay [142].

The task request profiles of the moving users that provide the total tasks requested at each radio node over a time period is an important metric that decides which MEC host serves the request. It is assumed that the MEC host nearest to the radio node serves the requests from the radio node. Five MEC hosts at the radio nodes are randomly selected to display the different variations in the MEC hosts evaluation. The selected MEC hosts located at the radio nodes and the aggregation point are shown in Figure 3.5.

Figure 3.4 shows the total number of tasks requested by moving users over the simulation period for selected radio nodes. This profile indicates how the offloading requests vary over time while users are on the move during the morning peak time based on our correlated mobility model and the uniform workload test. As time increases, users are moving toward the centre of the simulation area. It can be seen that the number of task requests at RN1 and RN2 decrease since these nodes are located in the outer fringes of the CBD. On the other hand, RN4 and RN5 increase since these are located near the CBD. The number of task requests in RN5 increases rapidly than RN4 as it is located at the centre of the area where most people are moving. The number of task requests in RN3 doesn’t vary much since the average number of users passing through RN3 during the morning peak hour doesn’t change much.

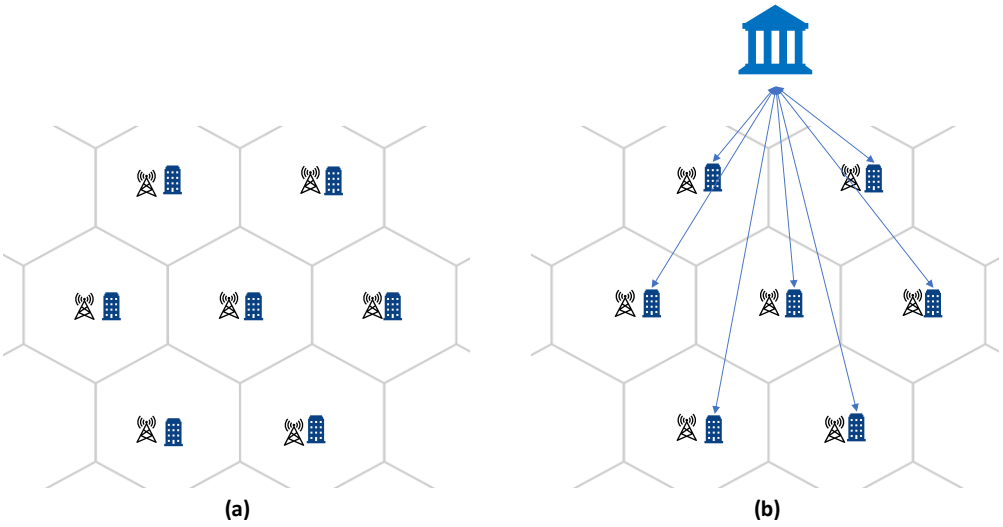

Figure 3.6 Different Deployment scenarios (a) RN deployment (b) AP deployment

3.4.3 MEC System Setup

Two different MEC host deployment scenarios were simulated: (i) only at radio nodes (ii) only at aggregation points. Figure 3.6(a) shows the scenario where MEC hosts are deployed at the radio nodes only. Figure 3.6(b) shows the scenario where MEC hosts are deployed at an aggregation point and how each radio node is connected to the MEC host. A 7 ms uploading delay in LTE wireless communication (communication delay between the radio node and MD) and 2 ms backhaul delay (communication delay between the radio node and the aggregation point) are assumed as in [143] and neglect the

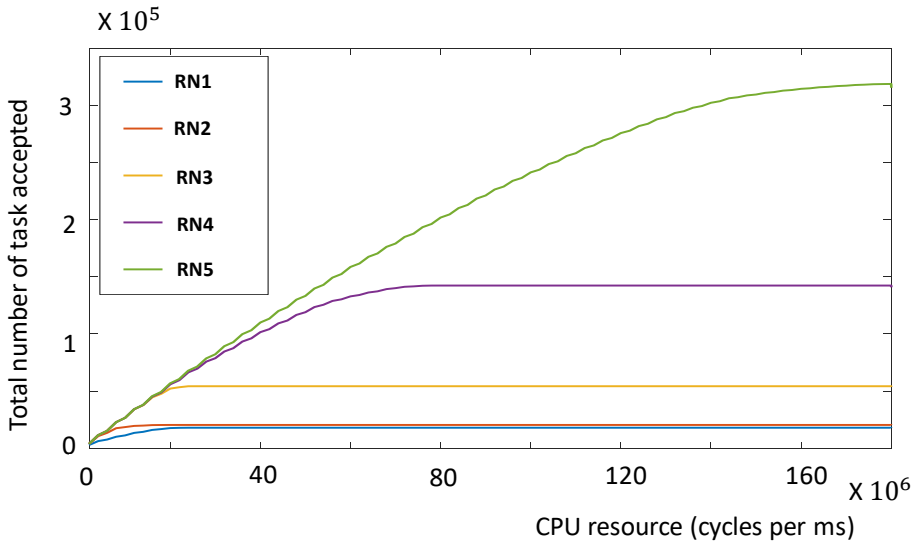

Figure 3.7 Total accepted tasks of selected MEC hosts for different CPU resource allocations.

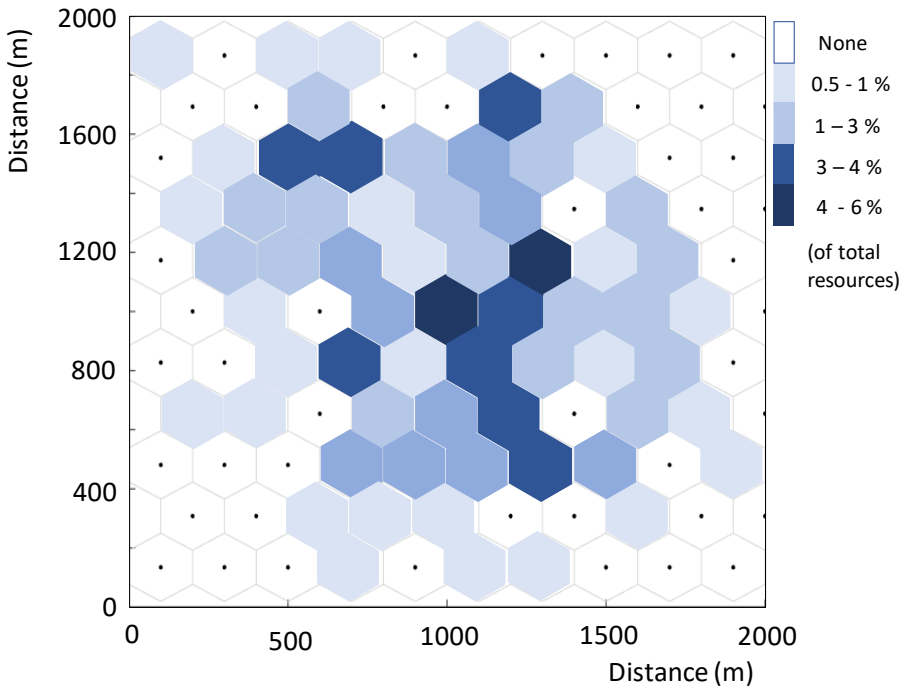

Figure 3.8 Selected MEC hosts and the CPU resources allocations for an instant

downloading time to receive the processed outcome as in [13]. CPU resources are allocated to meet the deadline requirement marginally ($\tau_{SDT}^{h} = \tau^{M}$) for each task, and first in first out scheduling is employed in each MEC host.

Figure 3.7 shows the accepted utility of the selected MEC hosts in terms of the total number of accepted tasks for different CPU resource allocations. An increment in CPU resources increases the total number of tasks accepted until the minimum resource required of the MEC host is reached. The total number of tasks accepted is constant beyond this allocation as all the offloading tasks requests are accepted in the MEC hosts. Thus, allocating additional resources beyond the minimum resource required of the MEC host should be avoided in a resource constraint environment.

Each user application will be relocated to the nearest MEC host while the MD is on the move based on the assumption that all the MEC hosts are capable enough to serve the user applications. As it is expected to migrate and pre-load the user applications to the target MEC host based on the correlated mobility predictions, the effect of relocation time is neglected, i.e., whenever a user moves to the coverage area of a new (target) MEC host other than the currently serving MEC host user application will be readily available to serve the user in the target MEC host [79]. However, there is no guarantee that the

offloaded task will be accomplished even the user application is relocated as this depends on the amount of resources available at the target MEC host. This scenario is captured in the task acceptance and task service profiles.

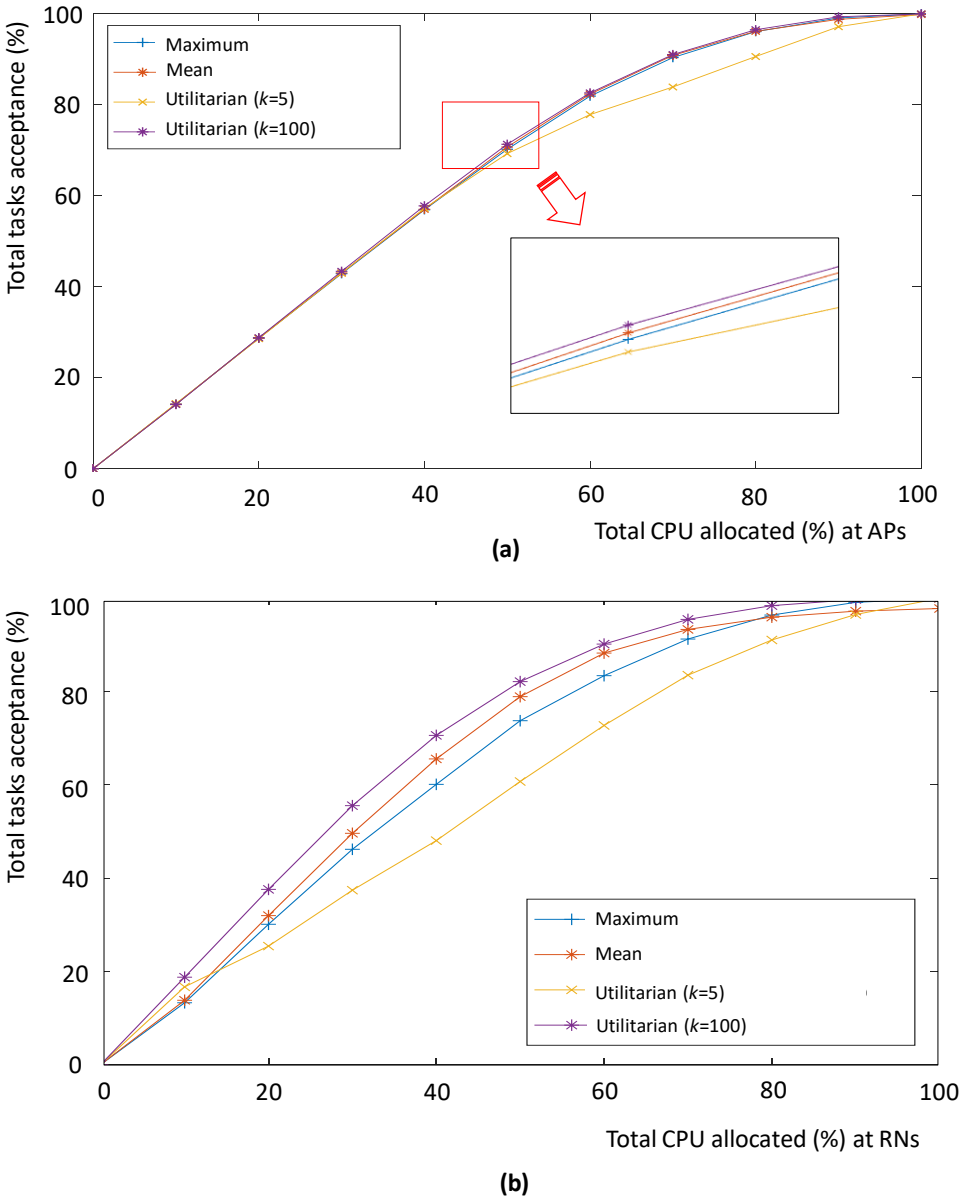

Figure 3.9 (a) resource allocations comparison for hard deadline requirement applications in AP deployments, **(b)** resource allocations comparison for hard deadline requirement applications in RN deployments

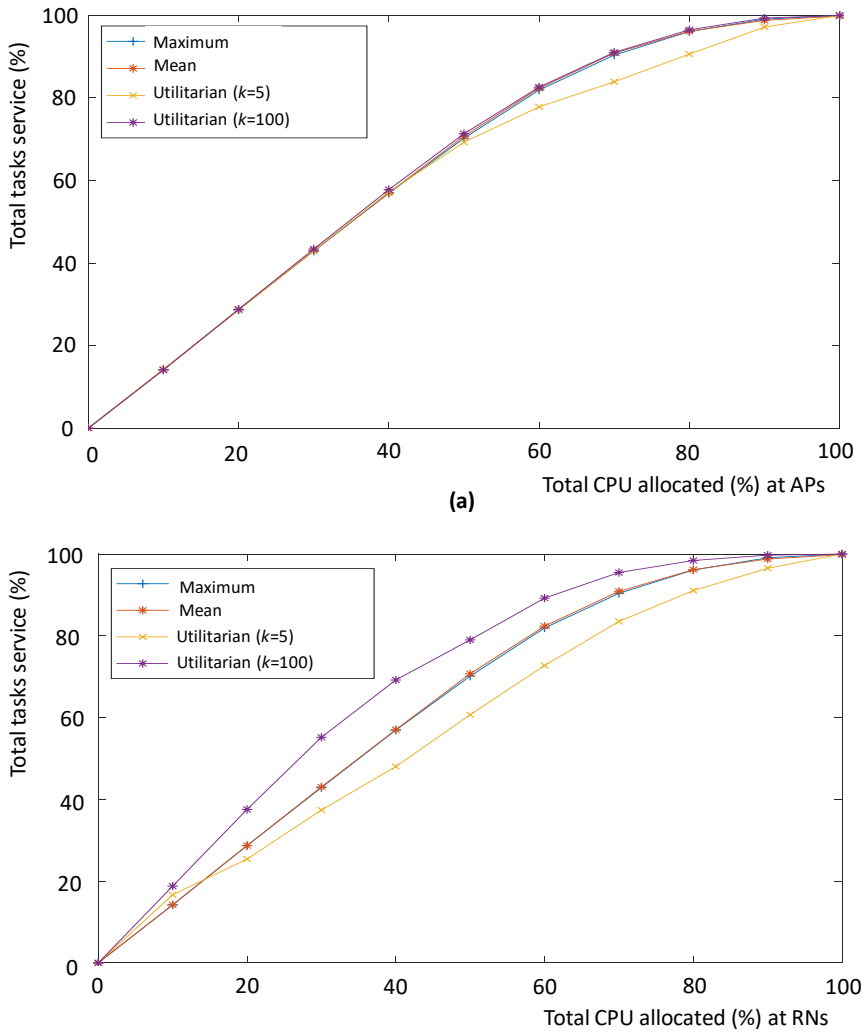

Figure 3.10 (a) resource allocations comparison for soft deadline requirement applications in AP deployments, **(b)** resource allocations comparison for soft deadline requirement applications in RN deployments

3.4.4 Evaluation of the Utilitarian Algorithm

The utilitarian resource distribution algorithm provides the selected locations for MEC host deployment from the proposed locations and the amount of CPU resources needed to be allocated in each MEC host. For instance, Figure 3.8 shows the selected locations for MEC hosts deployment and the amount of CPU resources needed to be allocated in the locations. This instance is plotted for the total CPU resources of 60% of the total minimum CPU resource as a result of the utilitarian algorithm. 32 % of the total CPU resources are allocated to the seven MEC hosts (out of 110 MEC hosts) located in

central CBD. Most of the MEC hosts located near to the centre of the CBD are allocated more CPU resources, while some of the outer MEC hosts are not considered for MEC host deployment. Figure 3.9(a) shows the total tasks acceptance ratio of hard deadline requirement case at the APs for different total CPU resources allocations at the MEC system. All the allocation methods provide almost the same results except the utilitarian allocation with less k value. Figure 3.9(b) compares the total tasks acceptance ratio of hard deadline requirement case at the RNs for different total CPU resources allocation at the MEC system. 72% and 90% of the total tasks are accepted with 40% and 60% of the total CPU resources, respectively. Total tasks acceptance ratio increases concavely with total CPU resource allocations.

Figure 3.10(a) and Figure 3.10(b) compare the total tasks service ratio of soft deadline requirement case at the APs and RNs, respectively, for different total CPU resources allocation at the MEC system. Total tasks service ratio also increases concavely with total CPU resource allocations. It is observed in both Figures (Figure 3.9 and Figure 3.10) that the performance of the utilitarian algorithm increases with a higher k value (smaller step size increment in CPU resources). Further, the utilitarian resource distribution algorithm with a higher k value outperforms other distribution methods and benchmarks the maximization problem. These graphs can be used to calculate the total number of CPU

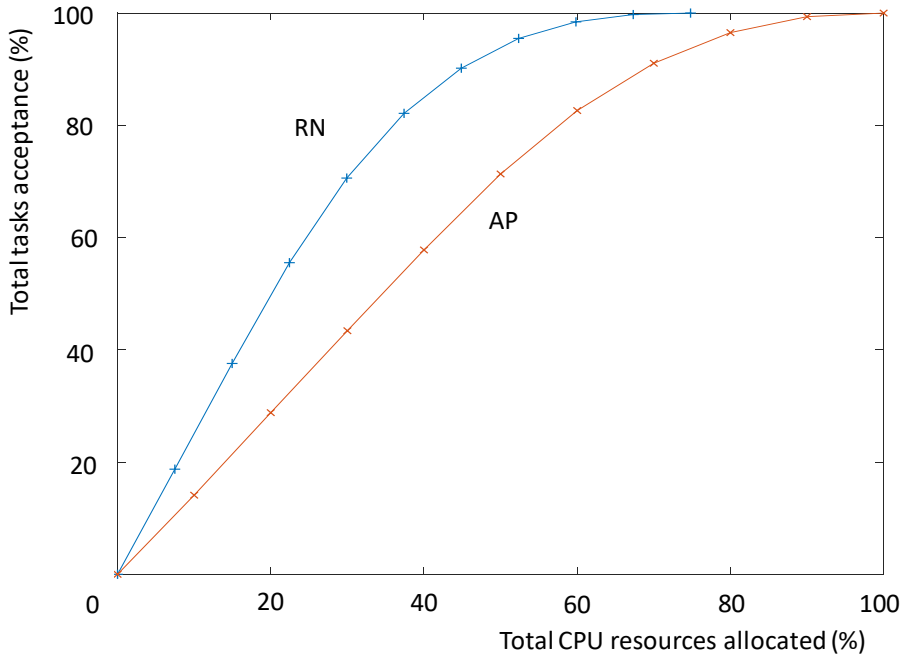

Figure 3.11 Comparison of total task acceptance for RN and AP deployments

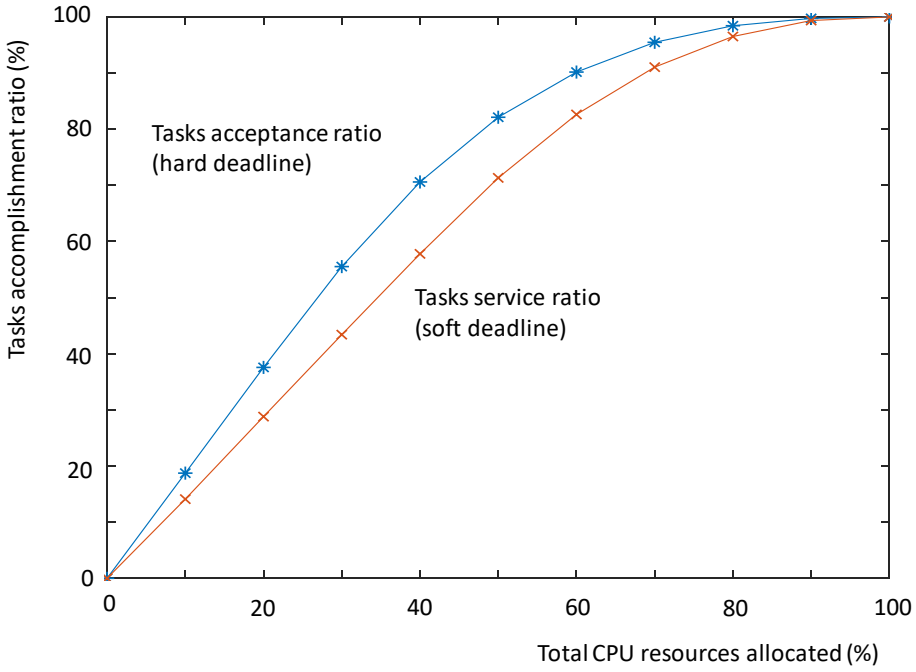

Figure 3.12 Comparison of tasks accomplishments of hard deadline vs soft deadline.

resources required for given total tasks accepted ratio or total tasks serviced ratio decided by the service providers based on the service level agreements with customers. On the other hand, based on the observations in the above Figures, the utilitarian algorithm doesn't have an added advantage in the APs deployment scenario. Since AP deployment scenario covers a larger area, user mobility is minimal than RN deployment scenario, i.e., the total task request profiles over the large geographical area remain more or less the same. Thus, advantage of using our proposed algorithm is minimal. Furthermore, it should be noted that for areas that have low mobility predictability, it is recommended that the MEC hosts be deployed at a higher layer of the network hierarchy, for example, aggregation point, so that the MEC hosts could cover relatively large areas and users compared to the deployment at radio node.

3.4.5 Hard Deadline vs Soft Deadline Requirement

The total tasks accepted ratio (hard deadline requirement case) is higher than the total tasks serviced ratio (soft deadline requirement case) for the same amount of total available CPU resources as shown in Figure 3.12, as the CPU resources are always allocated for all the tasks requested in the soft deadline requirement case; however, no CPU resource is

allocated if the task is rejected in hard deadline requirement case. Further, 60% of the total minimum required CPU resources ($\sum_h R_{REQ}^{C,h}$) are adequate to satisfy 90% and 82% of the total tasks requests over the simulation period in the hard deadline and soft deadline cases, respectively. These values depend on the task request profiles, specifically user mobility, the number of CPU cycles required, and deadline requirement of the requested tasks.

3.4.6 Different Deployment Comparison

The total number of tasks accepted is determined for the same total number of CPU resources available to compare the different deployment scenarios in the MEC system.

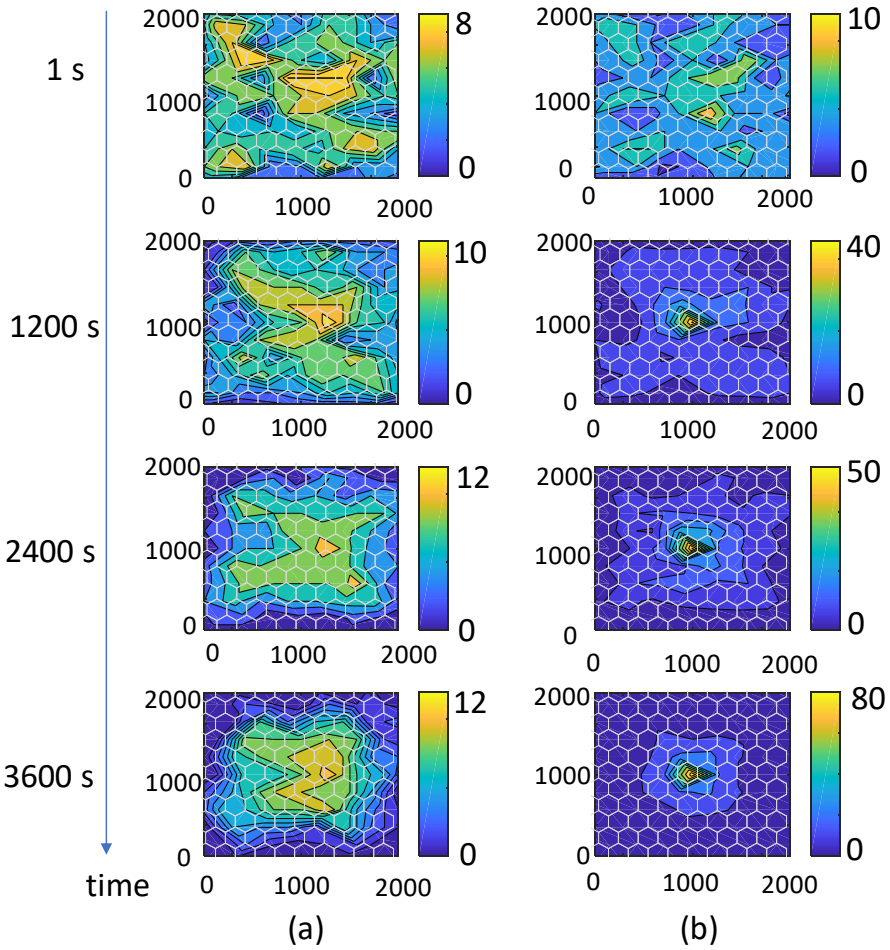

Figure 3.13 Comparison of total tasks accepted in RN deployments for random mobility vs correlated mobility

Figure 3.11 compares the total tasks acceptance ratio for RN and AP deployment scenarios. Deploying MEC hosts at the RN requires less total CPU resources than deploying at AP, as a 2 ms propagation delay between the RN and the AP influences the result. If the MEC host is deployed at the AP, the backhaul link will be used to transfer data, which will burden the backhaul link. Thus, the backhaul latency will be increased, which in turn increases the total number of CPU resources required to accomplish the tasks. This observation indicates that despite the deployment cost metrics, the capacity of the backhaul link is an important metric for the location selection (RN or AP) for MEC host deployment in a limited resource environment.

3.4.7 Task Acceptance Distribution Based on Mobility

Figure 3.13 compares the number of tasks accepted in RN deployment scenario during the morning peak hour between random mobility and correlated mobility for the resources allocated based on utilitarian distribution algorithm. It can be seen that the number of tasks accepted range remain unchanged between 0 and 10 for random mobility. On the other hand, the number of tasks accepted range increases from 0 - 10 to 0 - 80 in the correlated mobility scenario. This indicates that the utilitarian resource distribution algorithm allocates more resources near the centre locations, preparing to serve more tasks when users arrive at the centre location as a result of morning peak hour mobility. Further, it shows that the utilitarian resource distribution algorithm yields maximum results based on the given user mobility. Thus, tasks offloading request predictions based on user mobility play a vital role in allocating CPU resources in MEC hosts.

3.5. Conclusions

In this Chapter, computation offloading demands from mobile users are analysed to allocate resources in MEC network. A correlated mobility model is presented to capture user mobility during peak hours as mirrored by real data. User trajectories of spatial and temporal dimensions derived from the correlated mobility model and the task offloading request pattern of users are used to estimate the computation offloading demands in terms of total task request profiles of the MEC hosts. The computation offloading application is categorized as soft deadline and hard deadline based on the deadline requirement. The computation offloading demand of each MEC host is then used for the deployment and

resource distribution of MEC hosts using the utilitarian resource distribution algorithm. The utilitarian resource distribution algorithm proposed here provides a benchmark for the maximum number of total tasks accomplished while providing insight into the selected locations and the amount of CPU resources needed to be allocated in a resource-constrained environment.

As a typical example, the computation offloading demands of users moving during the morning peak hour is only considered. Simulation results show that 60% of the total minimum resource required is sufficient to satisfy 90% of the total task requests of hard deadline applications for users with correlated mobility and uniform workload. Further, a higher number of hard deadline requirement applications can be accomplished in the MEC host when compared to the soft deadline requirement case, as some of the task requests of the former are rejected due to resource limitations. Despite the deployment cost metric, deploying MEC hosts at the radio nodes based on the correlated mobility require less amount of both total computing resources and backhaul capacity. However, it requires better management of user mobility and the offloaded task migration. On the other hand, deploying MEC host at the aggregation points do not require frequent task migration or user mobility management, since it covers a larger geographical area for task offloading, simplifying the network management.

4 Energy-Efficient MEC Hosts

To realise MEC, suitable physical servers need to be selected with energy efficiency considerations and appropriate energy saving mechanisms need to be followed to minimize energy consumption. In this chapter, we investigate the energy efficiency of MEC deployment and evaluate the impact on the operational cost of the MEC hosts.

4.1. Introduction

Climate change is recognized as one of the key challenges the world is facing. The long-term goal of the Paris agreement at the recent Paris climate conference is to limit global warming to well below 20^0 C [144]. To achieve this goal, the International energy agency considers energy efficiency as the key enabler [145]. Gartner estimated that Information Communication Technology (ICT) sector produces up to 2% of the global CO_2 emissions in which data centres account for 23 % [146]. In addition, Cisco estimates that the data traffic from video streaming, real-time game consoles and mobile devices will be 82% of the total data traffic in 2021 and the total data traffic will be three folded of 2016 [147]. Hence, the energy consumption of the radio access networks together with the amount of information that needs to be stored, processed and transmitted by the power-hungry data centres are expected to increase dramatically over the next few years.

On the other hand, new and emerging mobile applications such as virtual reality applications require high computing power and very low latency. However, processing power and storage capacity of a mobile device are limited due to the size constraint of mobile devices. Computation offloading to MEC is considered as an emerging technology to support new requirements of mobile applications. In computation offloading, mobile devices can offload their compute-intensive tasks to the MEC host for execution. It is expected that a vast amount of MEC hosts will be deployed in the RAN to satisfy the demands of emerging mobile applications that require high computing power and very low latency.

Hence, one of the important challenges for a MEC service provider is to reduce the energy consumption of the MEC system. First, users' computation offloading request

patterns need to be analysed. Then, the selection of the ideal locations to deploy the MEC hosts needs to be able to satisfy the traffic demand. Then the required resources to be allocated to each MEC host, subject to budget constraints, need to be determined. As MEC hosts are deployed within the RAN, users' mobility patterns that tend to be correlated during peak hours [133] are important factors in deciding where to deploy the MEC hosts.

In Chapter 3, the deployment and resource distribution of MEC hosts were studied based on correlated mobility model. However, to the best of our knowledge, little study has been focusing on the energy efficiency of a MEC system in a resource constraint environment while maintaining a high quality of service in terms of latency for computing-intensive applications. In particular, how to maximize the energy efficiency of a MEC system while maximizing the number of offloaded tasks accomplished in a resource constraint environment by considering correlated mobility pattern, has yet to be investigated.

The contributions arising from this Chapter are as follows: (a) A physical server selection methodology is proposed to select suitable hardware for MEC host deployment; (b) energy-efficient processes are proposed to minimize the energy consumption during the MEC host operation and (c) comprehensive simulations are conducted with results showing the improvement in the energy efficiency of MEC hosts.

The remaining sections of this Chapter are organized as follows. Section 4.2 discusses the related work. Section 4.3 discusses the methodology and algorithm-based processes for energy-efficient MEC system. Then, Section 4.4 discusses the simulation and results. Finally, Section 4.5 concludes the Chapter.

4.2. Related Works in Literature

Computation tasks offloading of mobile devices in an edge computing environment could potentially improve the response time and reduces the energy consumption of mobile devices significantly [62]. Quality of experience in computation offloading of a virtual reality (VR) application is studied in three different resource limited edge cases [148] and Green VR is discussed to minimize the power consumption of the VR devices. Energy efficiency of base stations and cloud data centres are rigorously analysed in the literature. For example, wireless base stations are currently consuming more than 80 per

cent of the total mobile network (MN) power consumption and this number is expected to rise with the increases in data traffic and support of new services for 5G and the Internet of Things [149]. Cellular energy efficiency evaluation framework [150] and the mobile network energy efficiency assessment methodology and metrics [151] were also proposed. Network energy consumption of mobile applications was assessed [152] in which signalling traffic is revealed as a significant component in terms of network energy consumption.

On the other hand, the energy efficiency of a data centre is analysed thoroughly in the literature. Data collected from more than 5,000 servers showed that the servers operate only 10 – 50% of their full capacity most of the time [153]. Studies revealed that idle servers are expected to consume about 70% of their peak power [154]. Furthermore, energy efficiency measurement methodology for comparing the benefits of MEC vs. non-MEC is described in [52].

In terms of the allocation of VM, the VM placement in a datacentre can be considered as a bin packing problem, which is NP-hard. Heuristic approaches were proposed as potential solutions for VM placement [155]-[156]. However, virtual machine placement in MEC is different than the cloud because of the effect of user mobility. User mobility aware virtual network function placement has been studied in [157]. User mobility aware content caching was also investigated in the ultra-dense cellular network [158]. Markov Decision Process was used to select the communication path of the users in which dynamic resource allocation is conducted based on the mobility prediction in multi-access edge computing [130]. Energy efficiency in the MEC system was not considered in the above research.

Live migration of VMs has been investigated in the literature. A threshold-based migration was proposed in [17] with overload CPU threshold of 85% and underload CPU threshold of 50%. VMware distributed power management operates with the values of 81% and 45%, respectively [159]. An adaptive way to calculate the threshold values was proposed in [18]. Live migration has an impact on service degradation because it depends on the time taken to transfer the VM data, which in turn depends on the bandwidth allocated between the serving host and the target host.

4.3. System Models and Methodologies

Maximising energy efficiency is defined as minimizing the energy consumption to fulfil the same service without compromising the quality-of-service. Being energy efficient means the same task or service can be performed in a relatively lower energy costs compared to other solutions without sacrificing the quality-of-service. This involves weighing the higher initial cost of purchasing energy-efficient servers against the expected benefits of future cost savings when operating the servers. Thus, energy efficient MEC hosts selection considers the total cost of ownership of MEC host servers and energy minimization of the MEC hosts.

As power-hungry servers are going to be deployed at the MEC hosts within the RAN, selecting an existing radio node (cell site) for the deployment of MEC host would benefit in many ways. Both the MEC host and the base station at the radio node can share infrastructure and facilities such as cooling, feeder cables, which minimize the capital expenditure as well as operating expenditure. So, it is trivial that a MEC service provider will select a radio node to deploy MEC host to gain the above benefits. As the power consumptions of radio nodes have been thoroughly investigated in the literature, the *focus of this Chapter is on the power consumption of the MEC host system*.

4.3.1 Power Model

Computation offloading request of a user is treated as a user application in the MEC host. Each user application will be allocated a VM in a MEC host. Let $\mathbf{V} = \{1,2, \dots \mathrm{v}, \dots\}$ be the sets of virtual machines on the MEC host. Let $r_{ALLOC}^{C,t}$ be the allocated CPU cycles of a VM at a time instance. These allocated CPU cycles are calculated considering latency constraint of the application as explained in Section 3.3. Utility ($u(t)$) of the PS is the ratio of total CPU resources consumed by the VMs running on the physical server to the maximum CPU resource available at the PS:

$$u(t) = \frac{\sum_v r_{ALLOC}^{C,t}}{R_{MAX}} \tag{4.1}$$

where R_{MAX} is the maximum resources available at the PS and $\sum_v r_{ALLOC}^{C,t}$ is the total number of CPU cycles consumed by the VMs running on the PS. Hence, the power

consumption of a physical server is a function of the utility of the server [114] [23]. Power consumption (*P(u(t))*) of a PS can be formulated as follows:

$$P\big(u(t)\big) = k \times P_{MAX} + (1-k) \times P_{MAX} \times u(t) \tag{4.2}$$

where P_{MAX} is the power consumption of the PS at its maximum load [114]. k is the fraction of power consumed by the idle server. Typical value of *k* is 0.6 [14] [23]. In other words, 60% of the maximum power of the server is consumed even if the server is in the idle state. In addition, the total energy consumption of a PS during a period is

$$E = \int \big(P\big(u(t)\big) \times t\big)\, dt \tag{4.3}$$

The objective here is to minimize the total power consumption of the physical servers running in the MEC host without compromising the quality of service requirements. Thus, all the accepted tasks at the MEC host should be accomplished within the required deadline. In other words, the total power consumption of the MEC host is aimed to minimize, while allocating adequate CPU resources to the required VMs to satisfy the quality of service.

$$\min \sum_{s} P\big(u(t)\big) \tag{4.4}$$

subject to:

$$\sum_{t} r_{ALLOC}^{C,t} \leq r_{REQ}^{C} \tag{4.5}$$

$$\sum_{v} r_{ALLOC}^{C} \leq U_{MAX} \times R_{MAX} \tag{4.6}$$

where U_{MAX} is the threshold value of the utility of the physical server and $\sum_t r_{ALLOC}^{C,t}$ is the total number of CPU cycles allocated for the VM during the deadline. The first constraint in Eq. (4.5) is to ensure that adequate CPU resources are allocated to the VM during the execution of the task, thus, the quality of service is maintained. The second constraint in Eq. (4.6) is to ensure that the total available CPU resources of a PS in the MEC host do not exceed the total number of CPU resources consumed by all VMs running on that PS. $U_{MAX} \times R_{MAX}$ is the total available CPU cycles in a physical server. The typical utility threshold value, U_{MAX}, is 80%.

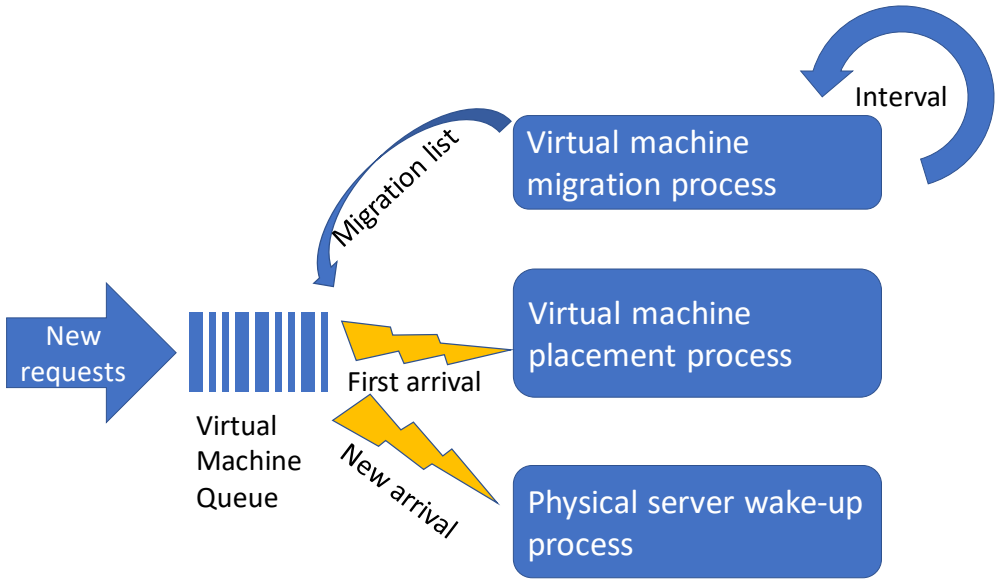

Figure 4.1 Proposed process to address energy-efficiency improvements

4.3.2 Energy-efficient Server Class Selection

Optimising the energy efficiency of the MEC hosts begins from the selection of physical servers based on the computation task offloading requests. User mobility pattern and user task offloading request pattern are used to calculate the task offloading request profile of each MEC host. Task offloading request profile is then used to calculate the computing resource demands at the MEC host. The key challenge of MEC service provider is to select the right server class, which minimizes the total average power consumption (i.e., total energy consumption) of the MEC system, i.e. to minimize the operational costs. Even though the ratio of the power consumption to the total CPU resources available of a large-scale server is lower than the small-scale server, the service provider needs to decide on the selection of server classes, which consume minimal power based on the task offloading resource request profiles at each MEC host. Further, as idle PS consumes significant power, it has been proposed to allow the idle PS to sleep in order to save power [160]. Thus, maintaining the utilization of a running server at a higher level by keeping the idle servers in sleeping state is regarded as a promising energy-efficient method.

In the proposed energy efficiency methodology, first, the task offloading request profile of each MEC host is estimated. Then the total average power consumption of a MEC host is calculated based on the estimated task offloading request profile for each

server classes. Then, the server class which minimizes the total average power consumption among other server classes is selected.

4.3.3 Energy-efficient Processes

The energy optimization procedure consists of three processes based on the single threshold energy optimization mechanism, which consumes less total power compared to other mechanisms [156]. Figure 4.1 shows the proposed energy optimization procedure. When an offloading request arrives at the MEC host, it is added to the virtual machine queue. When the first virtual machine request is added to the queue, the virtual machine placement process starts. When the queue's status changes from empty to non-empty, this process is triggered. The virtual machine placement process is to place all the offloading requests in the queue to the MEC host servers in an energy-efficient manner. Similarly, when a new request arrives at the queue, the physical server wake-up process starts. This process makes sure that there are enough resources available on the running servers. If not, it looks for a physical server to wake-up and adds its resources to the available resources. In addition to the above processes, the virtual machine migration process is to select virtual machines to migrate from one server to another as a result of the energy minimization process. This virtual machine migration process runs in a predefined interval. The output of this virtual machine process will be added to the virtual machines queue.

Algorithm 4.1 Virtual machine migration process

Input: *server_list* – a list of currently active physical servers.

Output: *migration_list* – a list of virtual machines that need to be migrated

Process

1. *tot_avail_resources* = *server_list.*availableResources()
2. **FOR EACH** *server* **IN** *server_list*
   a. *srv_util* = *server*.currentUtility()
   b. *srv_threshold* = *server*.Threshold()
   c. *srv_cur_resources* = *server.*currentResourceConsumption()
   d. **IF** *srv_util* > *srv_threshold*
      i. *bestFitvms* ← Find the best fit virtual machines that is running on the *server*
      ii. *migrationList*.add(*server*, *bestFitvm*)
      iii. *tot_avail_resources* = *tot_avail_resources* – *bestFitvms.totalAllocatedResources()*
   e. **IF** (*srv_util* < *srv_threshold)* **AND** (*srv_cur_resources* < *tot_avail_resources)*
      i. *migrationList*.add(*server*.runningVMs())
      ii. *tot_avail_resources* = *tot_avail_resources* - *srv_cur_resources*
   f. **IF** *server*.runningVMCount() == 0
   g. *server*.sleep()

4.3.4 Virtual Machines Migration Process

First, suitable virtual machines are selected for migration from the current running physical server to another physical server. In the virtual machine migration process, virtual machine selection is based on either the physical server is overloaded (i.e., above the threshold utilization) or it can be transferred to the sleep state to save energy. Then, the process places any idle physical servers to sleep state. As the virtual machine supervisor will be running on one of the physical servers, at least one physical server will be in the active state in each MEC host. The virtual machines selected for migration are then added to the virtual machine queue. The virtual machine migration process is shown in Algorithm 4.1. The VM queue consists of all the VMs needed to be placed in the

physical servers for execution. New arrival user application requests, i.e. new VM requests will also be placed in the VM queue.

The input to the virtual machine migration process is the list of currently active physical servers ordered by their utility in ascending order. The output will be the virtual machine migration list, i.e., the list of virtual machines needs to be migrated. The process starts with calculating the total available resources in currently active physical servers. Then, the process loops through each active server to find out the virtual machines to migrate. The current utility of the server is calculated. If the current utility exceeds the threshold level of the server, it searches for the best fit virtual machine(s) to migrate. If the best fit virtual machine(s) is removed from the server, the server utility will be just under the threshold value. In other words, best fit virtual machines are the virtual machines which is removed to keep server utility at maximum while maintaining the utility under the threshold value. Then it adds the selected best fit virtual machine(s) to the migration list.

If the current utility is below the threshold level and the amount of resources currently being used by the server is less than the total available resources, all the virtual machines in the server can be migrated to other servers. In other words, all the currently running virtual machines can be migrated to other servers, all the virtual machines will be added to the migration list. Further, the total available resources will be reduced by the amount of resources at the server. If no virtual machines are running on the server, the server can be placed in sleep mode to minimize energy consumption. After considering all the servers, the output of the process i.e., virtual machine migration list will be added to the virtual machine queue.

Algorithm 4.2 Virtual machine placement process

Input: *server_list* – a list of currently active physical servers,

vm_queue– a list of virtual machines in the queue

Output: successful placements of virtual machines

Process

WHILE *vm_queue*.count() >0

1. *vm_queue*.sortbyIncreasingResourceAllocated ()
2. *server_list*.sortbyIncreasingAvailableResources()
3. **FOR EACH** *vm* **IN** *vm_queue*
   a. *min_power_consumption* = MAX;
   b. **FOR EACH** *srv* **IN** *server_list*
   c. IF *vm*.resourceAllocated() < *srv*.resourceAvailable()
      1. *power_increment* = *srv*.extimatePower(*vm*. resourceAllocated())
      2. **IF** *power_increment* < *min_power_consumption*
         a. *srv.placeVM(vm);*
         b. *vm_queue.remove(vm);*
4. *vm_queue*.refresh()

4.3.5 Virtual Machine Placement Process

The virtual machine placement process places the ordered virtual machines in the queue to the physical servers in such a way that the power consumption is minimized. The virtual machines are allocated based on the required resources, sorted incrementally so that the number of tasks executed on the running physical servers can be maximized. The process searches all the currently running physical servers to find the best fit physical server, which has the resources to accommodate the required virtual machine (i.e., the physical server is not overloaded above the threshold by adding the new virtual machine) while consuming minimal power consumption. The virtual machine placement process algorithm is shown in Algorithm 4.2.

The list of currently running virtual machines and the virtual machine queue are the input for the algorithm. Virtual machine resources are allocated based on the resource requirement of the user application. It is assumed that all the physical servers have the

same idle power ratio k. Further, more virtual machines can be accommodated by placing the less resource required virtual machines first in a resource limited environment. Thus, ordering the virtual machine based on virtual machine resource allocations increases the number of tasks accomplished. Then the algorithm searches for the best server which executes the virtual machine with less power consumption. Power increment is estimated on each physical server, which indicates the additional power consumption by virtual machine by placing on the physical server. The physical server with less power increment will be selected for the placement of the virtual machine. Since active servers are sorted by available resources, virtual machines will be placed first in high utilized active servers, in turn, server utility is maximized. This will give room for under-utilized physical servers to be kept in sleep mode by the virtual machine migration process.

4.3.6 Physical Server Activation Process

The physical server activation process is responsible for activating (waking-up) the suitable servers that are in the sleeping state. If the demands of CPU resources increase while the currently running physical servers do not have adequate resources to handle the demands, the process will activate PSs to accommodate the demands. It selects physical servers to be wakened up in which the power consumption on the execution of the demands is minimized. The physical server activation process is shown in Algorithm 4.3.

Current active physical servers and list of servers that are in sleeping mode are the inputs. First, it calculates the total allocated resources of all the virtual machines in the queue and the total available resources on the currently running servers. In order to calculate the total available resources, server threshold values are considered, i.e., total available resources up to the threshold values. If the total allocated resources of virtual machines are higher than the total available resources on the currently running servers, then this process activates the suitable physical server(s) to satisfy the virtual machines' resource demands. Total power minimization is considered in selecting the server list

Algorithm 4.3 Physical server activation process.

Input: *server_list* – a list of physical servers in active state,

sleeping_list – a list of physical servers in sleeping state,

vm_queue – a list of virtual machines that need to be placed

Output: *short_resources – resource shortage*

list_activate – list of servers activated

Process

1. *vm_tot_resources* = *vm_queue*.totalAllocatedResource()
2. *srv_tot_avail_resources* = *serv_list*.totalAvailableResource()
3. **IF** *vm_tot_resources > srv_tot_avail_resources*
   - **a.** *req_resources = srv_tot_avail_resources - vm_tot_resources*
   - **b.** *sleeping_list.sortByResourcesDescending()*
   - **c.** *feasible_server = NULL;*
   - **d.** *short_resources = req_resources;*
   - **e.** **FOR EACH** *srv* **IN** *sleeping_list*
     - ***i.*** **IF** *srv.availableResources()>req_resources*
       1. *feasible_server = srv;*
       2. *short_resources = 0;*
     - **ii.** **ELSE**
       1. **IF** *feasible_server != NULL*
          - ***a.*** *list_activate.add(feasible_server);*
          - ***b.*** *feasible_server = NULL;*
       2. **IF** *short_resources > 0*
          - ***a.*** *list_activate.add(srv);*
          - ***b.*** *short_resources = short_resources - req_resources;*
       3. **ELSE**
          - ***a.*** **BREAK***;*
   - **f.** **FOR EACH** *server* **IN** *list_activate*
     - **i.** *server*.Activate()

among the sleeping server to activate. After selecting suitable servers, this process activates the servers from the sleeping state to activate state. All the above three processes

are running simultaneously to achieve the optimum energy efficiency of a MEC host.

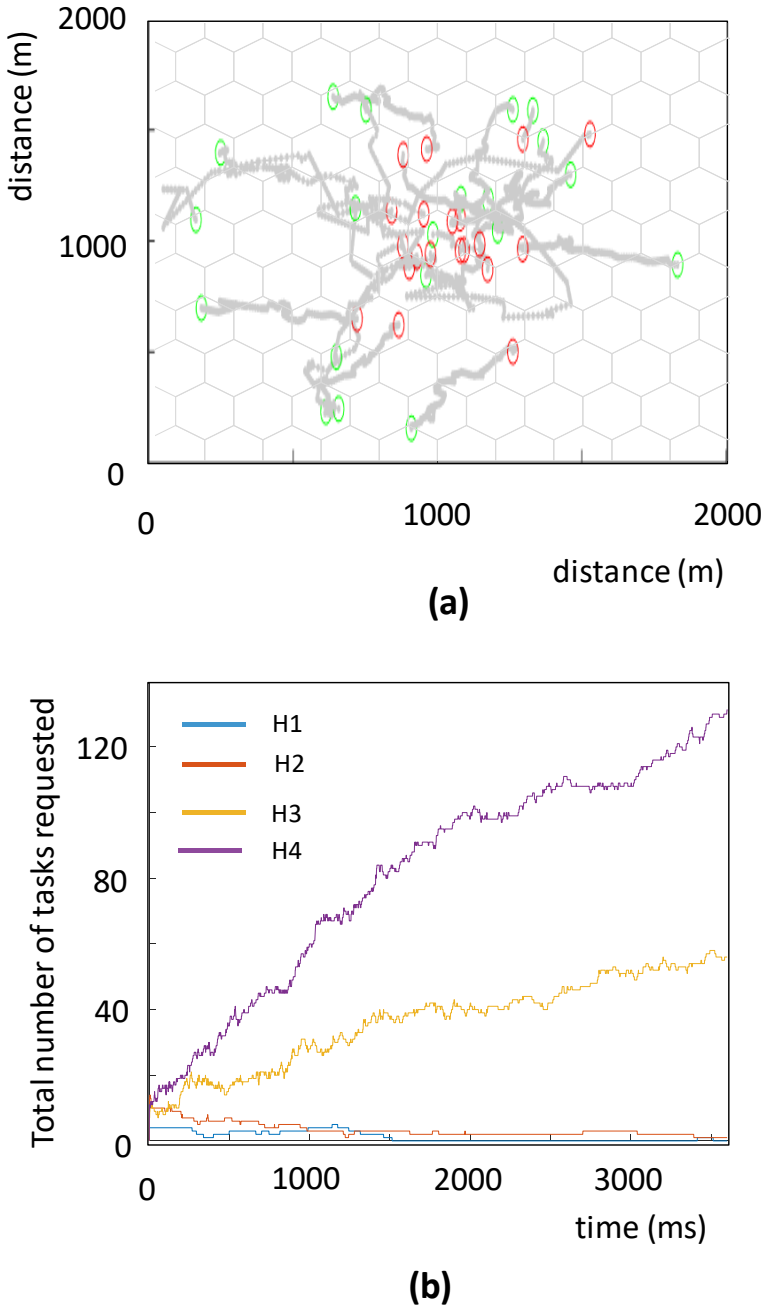

Figure 4.2 (a) User trajectories of selected 20 users during morning peak hours (green circle indicates starting point and red circle indicates end points) **(b)** tasks requests profile of selected radio nodes based on correlated mobility (1,000 users).

4.3.7 Process Execution Triggers

The virtual machine migration process could be performed within a fixed time interval, i.e., every 1 second. In order to optimize the number of running time of the other processes and minimize the queuing time of VMs, the virtual machine migration process could start running upon a queuing triggering event. The VM placement process could start running upon the first VM arrival event in the VM queue, i.e. when the VM queue state changes from empty to occupied. If there is a new VM request in the queue, the process starts to allocate CPU resources to the VM until the queue is empty. The PS activation process triggers upon the event of a new VM arrival at the queue to make sure that the required CPU resources are readily available to be occupied and thus, minimizing the queuing time. By minimizing the queuing time, the probability of meeting the deadline of the offloading requests could increase significantly. Further, the power consumed by the processes can be reduced by not running the processes when there is no VM request in the queue.

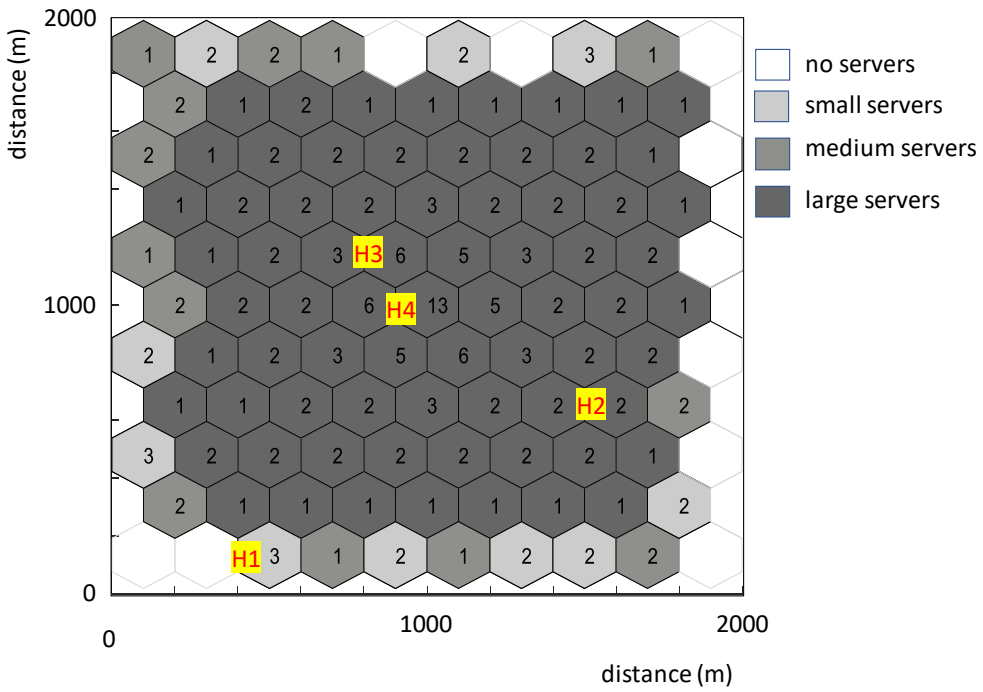

Figure 4.3 Deployment of different classes of servers and the number of servers. Selected MEC hosts for analysis are highlighted in yellow.

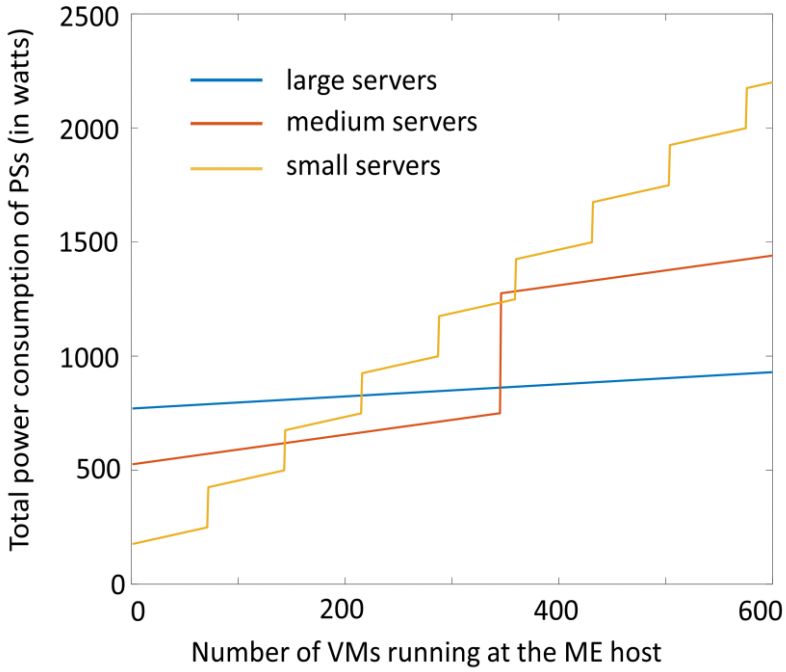

Figure 4.4 Total power consumption of different classes of servers for different loads (i.e., VMs)

Virtual machine supervisor is responsible to run all of these processes; virtual machine migration processes, virtual machine placement process and physical server activation process. Thus, there will be at least one physical server running in the MEC host, which hosts the virtual machine supervisor. Further, the virtual machine supervisor ensures that there is only one instance of the process running in each process.

4.4. Simulation and Results

In the simulations, a dense urban area of 2 km×2 km with 100 small radio nodes (picocells) deployed with a cell distance of 200m is implemented. The quality-of-service requirement of the MEC system is assumed to be 90%, i.e. 90% of the total offloaded tasks in the MEC system must be accomplished within the deadline requirement. Further, 1,000 users are assumed in correlated movement during morning peak hours, requesting for offloading tasks of a compute-intensive virtual reality application. Further, a uniform workload is used in which each mobile device (MD) unit requires 100 Mega CPU cycles of resources with a task deadline of 20 milliseconds in every second interval. Simulated data of user trajectories of morning peak hours of 20 users (out of 1,000) are shown in Figure 4.2(a) as mirrored with real data [133]. Figure 4.2(b) shows the total task

offloading request profiles of moving users in the selected MEC hosts. The selected MEC hosts for analysis are shown in Fig. 4.4. The utilitarian resource distribution algorithm is used for the deployment and resource distribution of MEC hosts in the simulation area. The utilitarian resource distribution algorithm selects the ideal locations to deploy MEC hosts and determines the amount of CPU resources required in each MEC host such that the total number of offloaded tasks accomplished in the MEC system is maximized by considering the correlated mobility patterns of mobile users. Further, based on the results in [156], a single threshold of 60% (i.e., j=60%) is chosen, which minimizes the total energy consumption among other algorithms with moderate service level agreement (SLA) violation [18]. MATLAB is used to simulate the proposed methodologies.

Three different Dell PowerEdge rack servers are selected: small server class (model - R230, total available CPU resources - 12 Giga CPU cycles per second, power consumption - 0.25kW), medium server class (R930, 57.6 Giga, 0.75kW) and large server class (R740, 207 Giga, 1.1kW) [161]. As observed in the datasheets, a large server has the minimum power to CPU cycle ratio. However, a service provider cannot simply deploy large servers in all MEC hosts.

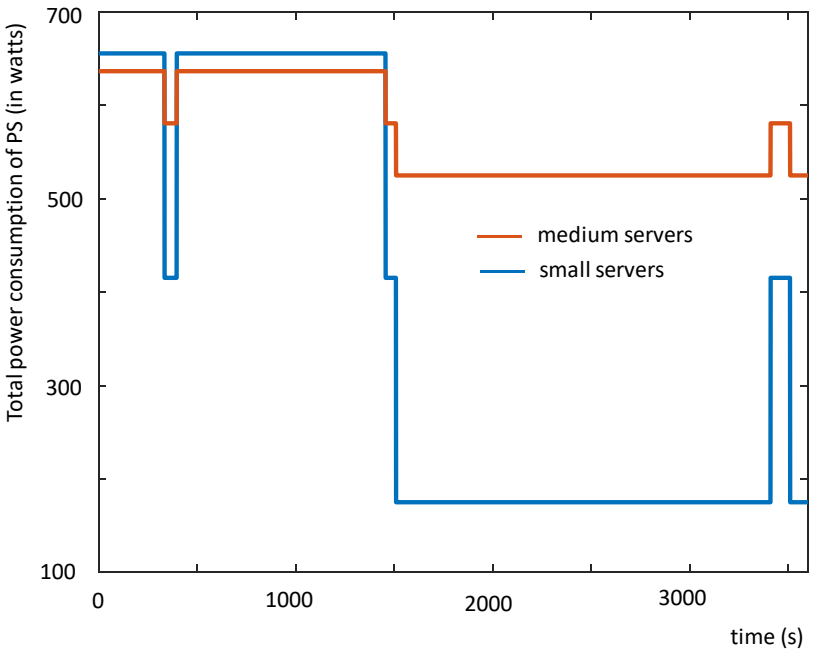

Figure 4.5 Total power consumption of different classes of servers deployed in H1.

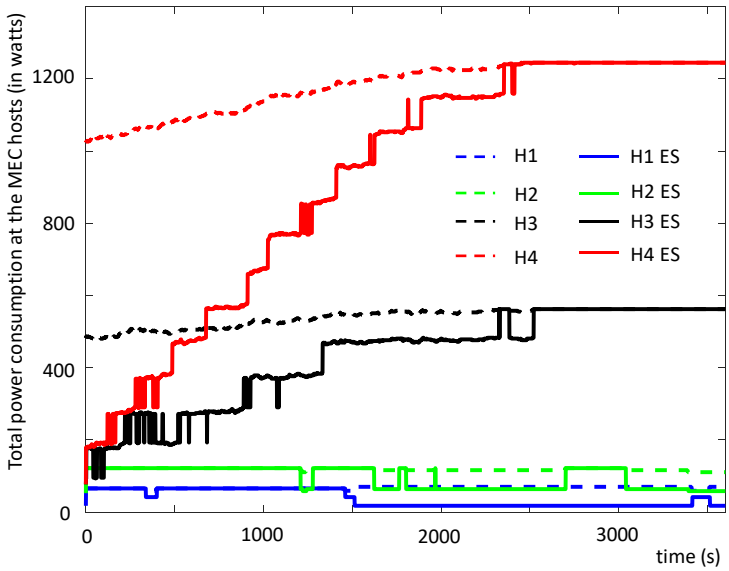

Figure 4.6 Power consumption of different MEC hosts (ES- with energy savings enabled).

Server Class Selection Evaluation

The server class selection should consider the task offloading request profile of the MEC host to minimize the total average power consumption (total energy consumption) of the system. Figure 4.4 shows the power consumption of three different MEC hosts that are running different classes of servers (a class of server per host) with different numbers of VM requests (user applications). As depicted in Figure 4.4, for less than 143 user applications (VMs), small servers are suitable and for the server load between 143 and 340 user applications, medium servers are suitable. Further, the power consumption of a MEC host is high and not comparable with the power consumption of a picocell radio node (i.e., approximately 10 watts) [162].

Thus, it is essential to select the classes of servers, which minimize the power consumption based on the task offloading request profile as described in the previous section. The class of servers and the required number of physical servers for the minimum total power consumption of the MEC system is shown in Figure 4.3. For some outer areas, radio nodes are not selected for the deployment of MEC hosts. As observed in Figure 4.3, 3 small servers are deployed at cell H1. The total average power consumption of 3 small servers at the maximum load is higher than the total average power consumption of a single medium server at maximum load. However, based on the task offloading request profile shown in Figure 4.2(b), the total average power (total energy) consumption of

three small servers with energy saving mechanisms are less than of a medium server as shown in Figure 4.5. Even though using a medium PS could result in lower energy consumption up to 1450s, the total energy consumption is higher than 3 small servers. This is because there are only a few task offloading requests during the time from 1450 s to 3500 s and thus the total average power consumption of a medium server in idle state is much higher than the total average power consumption of a small server in idle state (2 others in sleep state). By carefully selecting the servers in radio node H1, 706 kWh energy could be saved during an hour of morning peak hours, which is approximately 34.32% of the total energy (2.057 MWh calculated based on Eq. 4.3) consumed by the medium server during the same period.

4.4.2 Energy-efficient Processes Evaluation

The power consumption of selected MEC hosts is shown in Figure 4.6. It compares the power consumption of normal operation and the one with the energy saving process enabled. As users moving towards the centre area, the MEC hosts in the centre (H3 and H4) receive fewer task requests in the early part of the peak hour. Thus, the power saving processes are benefited in the early stage of the morning peak hours. The initial power consumption gap in H4 MEC host (deployed in the centre) indicates the power savings

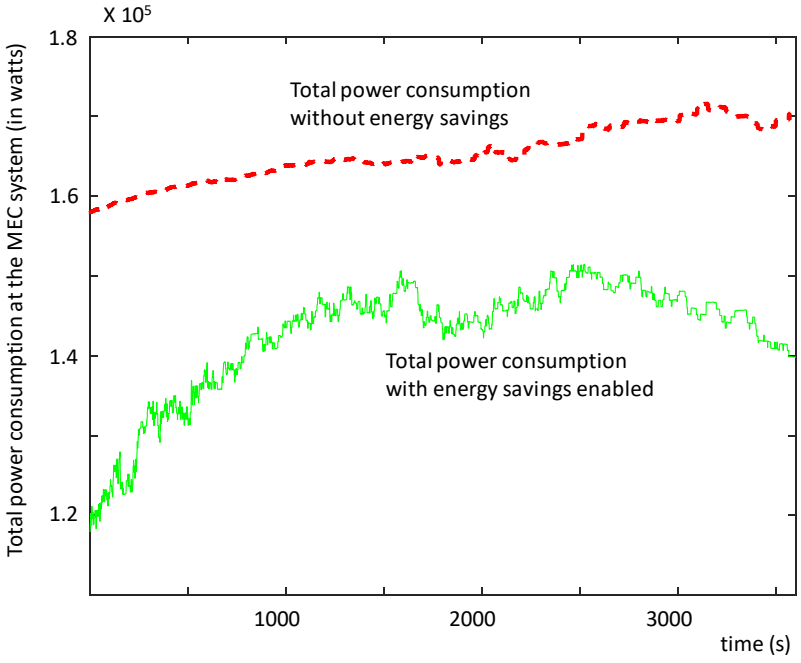

Figure 4.7 Total power consumptions of the MEC system

between the idle server and the sleep state enabled servers. On the other hand, the MEC hosts in the outer area (H1 and H2) receive fewer task requests in the latter part of the morning peak hours and energy-saving processes are beneficial in later stages of the morning peak hours.

The total power consumption of the MEC system during morning peak hours is shown in Figure 4.7. The total energy consumption of the MEC system during the morning peak hour is 541.5 MWh, calculated as the total energy consumption of all the PSs in all the MEC hosts based on Eq. (4.3) before our energy processes applied. Using our proposed energy saving processes, 87.45 MWh energy could be saved in the MEC system during a morning peak hour, which is translated to 16.15 % of the total energy consumption of the MEC system.

The server class selection method and energy-efficient processes can be applied to different user mobility patterns and different task offloading request patterns. This is because the server class selection method and the process of finding a PS with minimum power consumption in VM placement process are dependent on the task offloading request profile and the resource requirement of VMs, respectively. Thus, our proposed processes provide better results regardless of user mobility pattern and task offloading request pattern.

4.5. Conclusion

Energy savings of a multi-access edge computing host begins with careful selection of physical servers by utilizing the offloading task request profile context information based on users' mobility patterns and task offloading request patterns. As MEC hosts are deployed at the radio nodes, energy efficiency is a major challenge for MEC service providers in terms of capital expenditure and operational costs. In this chapter, a method for server class selection is proposed based on task offloading request profile of the MEC host. Then, energy saving algorithms for MEC host is also proposed based on the single threshold energy saving methodology. The proposed processes were evaluated based on correlated mobility and uniform workload. By carefully selecting the classes of servers based on the offloading request profile, the proposed algorithms could achieve an energy saving of up to 34.32% in a MEC host during the morning peak hour. In addition, based

on the proposed energy-efficient processes, an average energy saving of 16.15 % can be achieved in the MEC system during a morning peak hour.

5 Optimum MEC Host Selection

When the computation offloading request arrives at the MEC system, a suitable MEC host needs to be selected to serve the request. MEC service providers will desire to minimise the cost of providing offloading services to mobile users. In this Chapter, suitable MEC host selection based on cost minimisation to serve offloading requests is investigated.

5.1. Introduction

Extended reality based mobile applications are emerging and becoming more popular. Computation offloading of these applications comes with its own rules and requirements. All of the above-mentioned applications require very low latency to provide a better immersive experience to the end users and some of them require high network bandwidth for data transmission. Thus, the MEC deployment model of the application will also vary with applications. For instance, a personal virtual reality application may require one instance per user deployment model, while a virtual reality based museum navigation application may require one instance per MEC host and collaborative online gaming may require one instance on each MEC host. The requirement of virtualized resources such as compute, storage, network resources may also vary between applications. These applications may depend on different MEC services that are provided in the MEC system. For instance, an augmented reality application may consume the object identification service provided by the MEC system. Some MEC applications require connectivity to the local network, while some require connectivity to the Internet. Some others require persistent storage to store data permanently. Thus, satisfying these rules and requirements of mobile applications is one of the greater challenges faced by MEC service providers.

MEC hosts are deployed within the radio access network to provide virtualized resources such as compute, storage and network resources for computation offloading of mobile users. Deployment within the radio access network increases the context awareness and decreases the offloading network latency. MEC hosts can be deployed at a radio node, or the edge of the core network, or at an aggregation point that is between the radio node and the edge of the core network [21]. In order to maximize the utility of

the MEC hosts with limited resources, collaboration among MEC hosts is proposed in [163].

Since different applications can have different requirements and MEC hosts can be deployed in different network locations and collaboration among them is allowed, it is a critical challenge to select a suitable MEC host to serve the application. Thus, selecting the ideal MEC hosts to instantiate the user application by abiding user application's rules and requirements is a critical challenge for the MEC service providers. This challenge is known as the *MEC hosts selection problem* and solving this requires joint considerations of limitations in resources (bandwidth and computing) and the associated usage costs in order to minimize the overall network costs. However, to the best of our knowledge, *consideration of the cost of provisioning the MEC offloading services in the MEC hosts selection problem has yet to be fully investigated*. This chapter addresses the key question of how to minimize the total cost incurred by the MEC service providers in selecting the ideal MEC hosts to instantiate the user application in a collaborative environment without compromising latency requirements.

The remaining sections are organized as follows. Section 5.2 discusses the related work in MEC host selection. Section 5.3 discusses the methodology and our modifications to the Balas-Geoffrion additive algorithm. Then, Section 5.4 discusses the simulation setup and results that verify the effectiveness of the optimization and the MEC hosts collaborations. Finally, Section 5.5 concludes the chapter with a summary of the insights gained from our work.

5.2. Related Works in Literature

Related works in application placement algorithms can be found in various domains such as content distribution within a content delivery network [4] and cloud server management in the context of cloud computing [164]. In [165], the placement of content is studied in the context of content delivery networks and the cost is calculated based on the hop count.

Another body of existing work usually involves only two physical computing entities (i.e., the mobile device and the cloud) [166] [67]. The net utility that trades-off the energy saved by the mobile, subject to constraints on the communication delay, overall

application execution time, and component precedence order is defined in [166] to solve the linear optimization problem using real data measurements obtained from running multi-component applications in mobile phone and cloud. A power-constrained delay minimization problem in [67] by analysing the average delay of each task and the average power consumption at the mobile device and proposed an efficient one-dimensional search algorithm to find the optimal task scheduling policy.

Some existing works on application placement and scheduling in MECs have considered applications with two components, one running on the cloud (which can either be by the MEC or core cloud) and the other running on the mobile device [167] [168]. The hierarchical architecture of edge cloud is considered in [169] to enable aggregation of the peak loads across different tiers of cloud servers to maximize the amount of mobile workloads being served.

The user application is modelled in [170] as an application graph and the physical computing system as a physical graph with resource demands/ availabilities annotated on these graphs. A task placement algorithm of an edge computing orchestrator is also proposed in [171] that performs replication and placement of application components in a telecom-driven application-hosting infrastructure. The Nova scheduler of OpenStack [172] is an existing application placement architecture that selects suitable computing nodes to initiate the virtual machines. However, it is explicitly disconnected from the networking component and does not consider requirements such as latency, connectivity and mobility that originated from the application providers [171].

5.3. System Models and Methodologies

The user application instantiation process is described in Chapter 1. First, a device application sends an offloading request to MEC system to instantiate the computation offloading process. The MEC orchestrator is responsible to instantiate the user application in the most suitable MEC host(s) in the MEC system in response to the request from the mobile device. The MEC hosts selections should satisfy the rules and requirements of the request such as deployment mode, specific hardware requirement, required resources, latency, connectivity, and mobility requirements. Since user application may depend on different MEC services such as image processing service, object identification service or persistence storage service, it is a critical challenge for MEC service providers to select

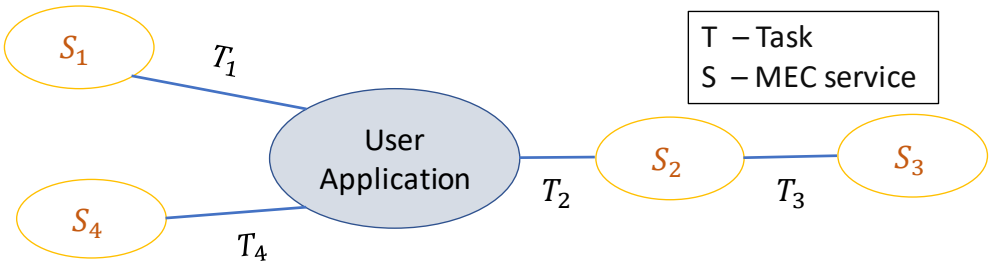

Figure 5.1 MEC service dependency graph of a user application

the suitable MEC hosts to provide above mentioned dependent services in an efficient manner while satisfying latency requirement of the user application. Once the user application is instantiated in the MEC hosts, the device application, that is running on the mobile device, collaborates with user application to offload the computation tasks regularly, until the computation offloading process ends.

On the other hand, hosting the offloading applications in MEC network incur costs to the MEC service providers, which includes computing resource usage cost and network bandwidth resource usage cost. Computing resources and network bandwidth resource usages may have separate cost structures. Further, hosting dependant MEC services in every MEC host is not feasible, since additional resources need to be allocated in each MEC hosts for dependant MEC services in each MEC host. It is important to note that hosting dependant MEC services in each MEC host will incur additional cost to the service providers and underutilization of resources. Hence, MEC services will be deployed at feasible and optimum network locations. The MEC service location is selected based on the service demand, network topology, resource availability and service provider's preference locations. In this research, the focus is on the cost associated with the offloading services, where MEC services are deployed in predefined locations. Thus, from the MEC service provider's point of view, the objective is to minimize the total cost incurred by the MEC system when provisioning the offloading services while satisfying the service requirements of the MEC application.

5.3.1 User Application Modeling

The MEC service dependencies of a user application can be represented in a graph. For instance, Figure 5.1 shows the service dependencies of a user application. The shown user application depends on the S_1, S_2 and S_4 MEC services. S_2 service depends on S_3

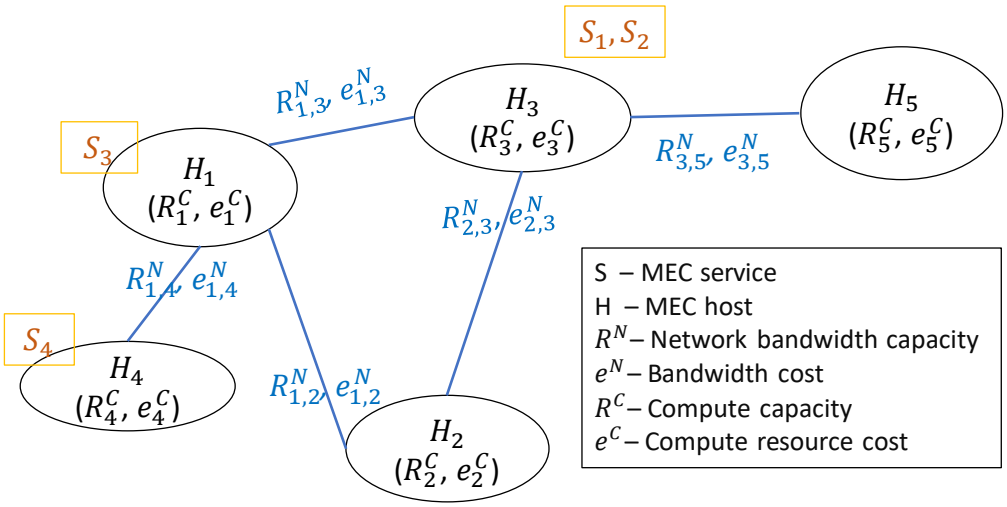

Figure 5.2 Network topology graph of the MEC network

service. For instance, object identification MEC service depends on classification MEC service. Vertices of the graph represent the MEC services and the edges represent the computation offloading requests within the services. Each MEC service request from the user application can be considered as a computational task offloading request from the user application. For instance, T_1 represents the computation task to the service S_1. One MEC service may depend on another MEC service as S_2 depends on S_3 service as in Figure 5.1. Interdependencies among MEC services can be grouped into one service in which user application depends on. Thus, this research focuses only on service dependencies of user application and not on the interdependencies of MEC services. Computation intensive tasks (T) can be represented by the size of the offloading data (q^N), task deadline (τ^{MAX}) and the number of CPU cycles required to complete the task (q^C), i.e., $T \triangleq (q^N, \tau^{MAX}, q^C)$.

On the other hand, MEC hosts deployment can be represented in the MEC network topology graph as shown in Figure 5.2. Vertices of the graph represent the MEC hosts, the MEC host capacity in terms of computing resources and the unit compute resource allocation cost. Edges of the graph represent the network capacities in terms of bandwidth resources and the unit bandwidth allocation cost. For instance, R_1^C, e_1^C represent the allocated CPU resource and the unit CPU allocation cost in MEC host H1, respectively. Similarly, $R_{2,3}^N, e_{2,3}^N$ are the allocated network bandwidth on the network link between MEC hosts H_2 and H_3 and the unit bandwidth allocation cost of the network link respectively. A complete topology graph of the MEC system can be generated from the

network topology graph in which there is a direct communication link between all pairs of MEC hosts. All the communication links in the complete topology graph also will have the network bandwidth capacity and the unit bandwidth allocation cost. It should be noted that both compute and bandwidth resource usages are associated with different costs in the complete graph. In addition, these costs are not affected by user mobility or channel interference, since these costs are in MEC access network.

5.3.2 Total Cost Minimization

Let MEC hosts in the MEC system be represented in a set $H = \{1,2, \dots h, \dots\}$. Let I $(I \subseteq H)$ be the filtered subset of the MEC hosts that could support the rules and requirements of the user applications such as virtualized resource requirement and special hardware requirement. Different MEC services might be required by the user application. Let $J(J \subseteq H)$ be the filtered subset of MEC hosts in which the required services of the user application are hosted. Let $x_{i,j}^N$ be the maximum amount of network bandwidth that can be allocated to the user application during the data transmission to be used between the MEC hosts i and j. This network bandwidth allocation is based on the network resource allocation policy of the MEC service provider and the available bandwidth. For instance, a maximum of 50% of the remaining bandwidth can be allocated to the incoming resource request. Thus, the minimum network latency is given by $\tau_{i,j}^N = \frac{q_{i,j}^N}{x_{i,j}^N}$, where $q_{i,j}^N$ is the data that needs to be transmitted between the MEC hosts. Further, the data transmission cost is $e_{i,j}^N * q_{i,j}^N$, where $e_{i,j}^N$ is the cost of transferring a unit of data between MEC hosts.

Let x_i^C be the maximum computing resource that can be allocated at the MEC host during the task execution. Noted that the computing resource allocation also depends on the MEC service provider's resource allocation policy and the resource availability at the MEC host with limited resources. Thus, the minimum computing latency in the servicing MEC host is $\tau_i^C = \frac{q_i^C}{x_i^C}$, where q_i^C is the required computing resources. The computing cost is thus $e_i^C * x_i^C$, where e_i^C is the unit allocation cost of the computing resources at MEC host i. The total service cost is the sum of the data transmission costs and computing

costs, i.e., $e_i^C * x_i^C + \sum_{j \in J}(e_{i,j}^N * q_{i,j}^N + e_j^C * x_j^C)$. It is assumed that each service is independent and can be consumed in parallel.

The objective of the MEC service provider is to minimize the total cost incurred by provisioning the offloading services while maintaining the quality-of-service (QoS):

$$\min \{ \sum_{i \in I} \alpha_i \, (e_i^C * x_i^C + \sum_{j \in J}(e_{i,j}^N * q_{i,j}^N + e_j^C * x_j^C)) \} \tag{5.1}$$

Subject to

$$\alpha_i(\tau_i^C + \max\{\tau_j^C + \tau_{i,j}^N\}) \leq \tau^{MAX}; \; \forall \, i \in I, \mathrm{j} \in \mathrm{J} \tag{5.2}$$

$$\sum_{i \in I} \alpha_i \geq m \text{ and } \alpha_i \in \{0,1\} \tag{5.3}$$

where $e_i^C \geq 0$, $e_{i,j}^N \geq 0$ $e_j^C \geq 0$, $x_i^C \geq 0$, $\mathrm{q}_{\mathrm{i,j}}^{\mathrm{N}} \geq 0$, $x_j^C \geq 0$ and α_i is a binary variable. If $\alpha_i = 1$, the MEC host is selected for instantiating the user application, otherwise, it is not selected. The offloaded computational task should be executed on or before the deadline requirement of the user application. Since each service is independent, the task execution time of each service is independent. Thus, the sum of task execution time at the user application and the maximum execution time of the MEC service execution times should be within the deadline requirement of the application as in Eq. (5.2). In addition to the latency constraints in Eq. (5.2), there is another constraint in Eq. (5.3): at least m MEC hosts shall be selected to instantiate the user application as per the deployment mode requirement of the application. Eq. (5.3) can be written as

$$-\sum_{i \in I} \alpha_i \leq -m \text{ and } \alpha_i \in \{0,1\} \tag{5.4}$$

The optimal (feasible) solution needs to satisfy Eqs. (5.1), (5.2) and (5.4). The above minimization problem can be represented in the standard form as follows:

$$\min\{ \sum_{i \in I} \alpha_i * z_i \} \tag{5.5}$$

Subject to

$$\sum \alpha_i * y_{i,k} + \beta_k = \gamma_k, \qquad k = 1,2 \tag{5.6}$$

where $z_i = e_i^C * x_i^C + \sum_{j \in J}(e_{i,j}^N * q_{i,j}^N + e_j^C * x_j^C)$ and slack variable of the constraint k is $\beta_k (\geq 0)$. The objective cost function is derived using cost functions of network and computing resources. Similarly, latencies are derived from the resource limitations in network bandwidth and computing resources. As all the cost components are non-negative, the total cost is also non-negative ($z_i \geq 0$).

Eqns. (5.1), (5.2) and (5.4) are considered as MEC hosts selection problem. The problem is dual feasible as $z_i \geq 0$. Further, by analyzing the problem, there is only one constraint with all nonpositive constant (-1) coefficients as in Eq. (5.4) and other with all nonnegative coefficient as in Eq. (5.2). In other words, it is a minimization problem with upper bound constraints with all positive coefficients and all negative constant coefficients.

5.3.3 Balas-Geoffrion Additive Algorithm

It is important to note that the solution of MEC hosts selection problem will be a 0-1 integer (binary) programming. Binary programming is NP-complete and is one of Karp's 21 NP-complete problems [173]. Techniques available for solving the 0-1 integer programming problem include algorithms of Glass, Balas, Glover, Lawler and Bell, Geoffrion, Lemke and Spielberg etc. as summarized in [174]. These additive algorithms are enumerative and developed for solving 0-1 binary programming problems. The general idea of the additive algorithm is to enumerate through some of all 2n possible solutions of a problem explicitly to find the best solution.

Bala's additive algorithm [175] with some modifications can be applied to solve the MEC host selection problem as it is dual feasible. The approach of Bala's algorithm that makes it efficient is that only some solutions are selected for enumeration. Geoffrion reformulated the additive algorithm by reducing the spatial complexity (storage) to improve the efficiency of the Balas algorithm [176]. The only operations required under the algorithm are additions and subtractions.

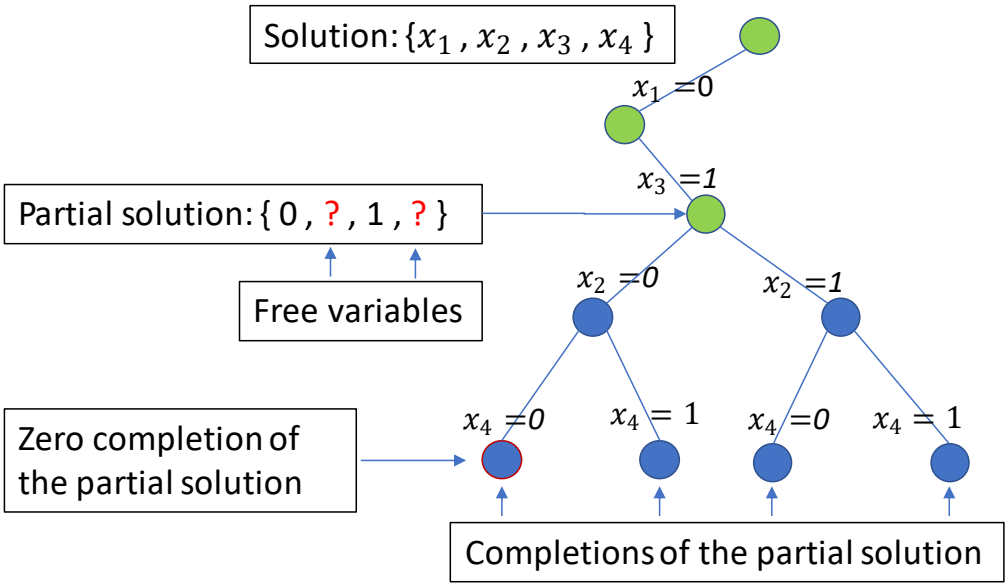

Figure 5.3 Balas-Geoffrion algorithm terminology

The computation complexity of addition and subtractions is Θ(n) whereas the computational complexity of multiplication and division is $\Theta(n^2)$. This shows the advantage of applying an additive algorithm in terms of computational complexity in solving the MEC selection problem. Another advantage of the algorithm is that it provides the near-optimal solution, even if the calculations stop before all the possible solutions are enumerated. [177].

As mentioned above, not all the solutions are to be explicitly enumerated, rather implicitly enumerated by considering groups of solutions together. To explain how groups of solutions will be defined, partial solution notation is used. A partial solution is defined as an assignment of binary values to a subset of variables. Any variable not assigned a value is called the free variable. For instance, x_1 $(= 0)$ and $x_3 (= 1)$ variables are assigned with values and x_2 and x_4 are free variables in the partial solution in Figure 5.3. The tree represents the order in which the variables are enumerated. In this instance, x_3 is assigned before x_2. Completions of a partial solution is defined as a solution that is determined together with the binary values of the free variables. There are four completions of the partial solution, since x_2 and x_4 can be assigned to 0 or 1. Thus, the solution may be one of the {0,0,1,0}, {0,1,1,0}, {0,0,1,1}, {0,1,1,1}. Zero completion of the partial solution is derived by assigning zeros for all free variables, i.e., {0,0,1,0} in this case.

Implicit enumeration involves generating a sequence of partial solutions and simultaneously considering all completions of each. As the calculations proceed, feasible solutions are discovered from time to time, and the best one yet found is kept in store as an incumbent. Now it may happen that for a given partial solution, the best feasible completion of the partial solution can be determined, i.e., feasible completion that minimizes the objective function among all feasible completions of the partial solution. If such a best feasible completion is better than the best-known feasible solution, assuming that one is known, then it replaces the later in store. Otherwise, the partial solution may have no feasible completion better than the incumbent. In either case, it means that one could fathom the partial solution.

All completions of a fathomed partial solution have been implicitly enumerated in the sense that they can be excluded from further consideration. Fathoming tests can be carried out to omit unnecessary iterations to find out the solutions. In addition, we need to make sure that no completion of a partial solution in the sequence ever duplicates a completion of a previous partial solution that was fathomed.

5.3.4 Extended Balas-Geoffrion Additive Algorithm

Even though Balas-Geoffrion algorithm can be applied to a general binary programming problem, it is not efficient in solving binary problems with upper bound constraints with all positive coefficients and all negative constant coefficients. The selection strategy of free variables and fathoming tests of Balas-Geoffrion algorithm is modified to improve the efficiency of the algorithm for the above-mentioned special case of binary programming problems. Bala's strategy was to choose the free variables, which would then result in the least infeasibility. As there is an upper bound constraint with all negative constant coefficients, least infeasibility calculations will end up listing all the free variables and must select one free variable randomly. Thus, the strategy is modified to select the free variables, which have the minimum coefficient in the objective function in order to guarantee the optimality of the problem.

Balas developed four tests as described in [174] to validate whether the given partial solution is fathomed or not. Based on the context of our problem, Tests 1 and 3 in [3] can be omitted. Because all the coefficients in each constraint are either positive or negative, there is no nonnegative coefficient for a free variable in all the constraints in Test 1 and

hence should be omitted. Similarly, in Test 3, if the slack variable of a partial solution is negative, it should be from one of the upper bound constraints with all positive coefficients. Thus, there is no way to improve it by converting it to be positive by assigning 1 to any free variables in that constraint.

If the algorithm terminates with the feasible solutions, then the computation offloading request will be accepted, otherwise, the offloading request will be rejected. The total requests accepted in the MEC system is defined as a metric of performance measurement to compare the collaborative MEC hosts method and non-collaborative (independent) MEC hosts method.

5.4. Simulation and Results

In order to evaluate the performance of MEC hosts collaborations, the MEC hosts selection problem is simulated involving an urban area of 2 km × 2 km served by a mobile wireless network in which radio nodes are spaced 200 m apart. A total of 1,000 users is assumed, each with a mobile device moving in vehicles for an hour, i.e., 3,600 seconds during the morning peak hour rush. Correlated mobility model created in Section 3.3.1 is used to produce the users' trajectories of morning peak hours in which users start from the outer suburbs of the city and then move towards the central business district. Different

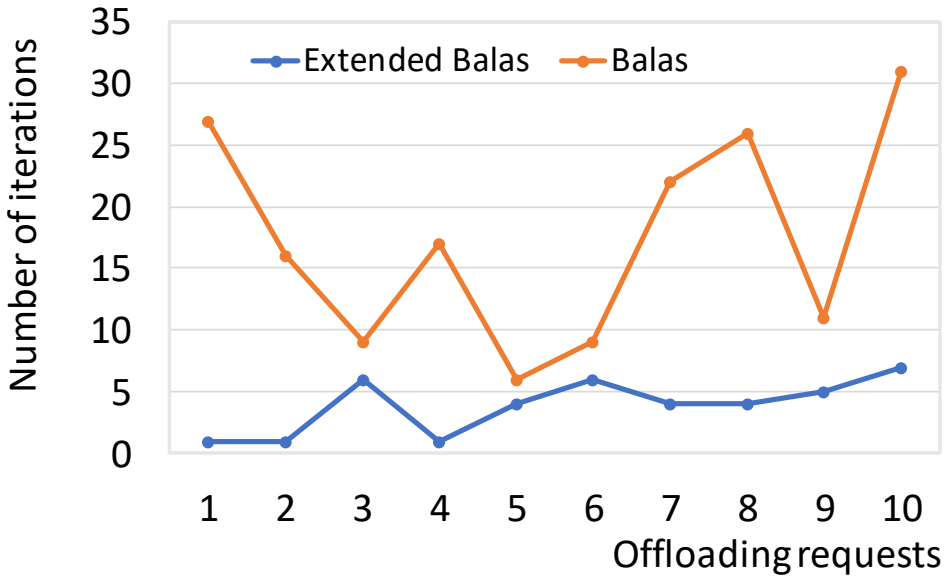

Figure 5.4 Number of iterations required to find the optimal solution for Balas and modified Balas algorithms

types of servers are deployed in the MEC hosts considering energy efficiency as in the Section 4.3.2. Electricity cost is a major operational cost for MEC host servers. The mean electricity cost is USD 78 per MWh as in [178]. The power consumption of servers based on server utilization is considered.

Further, different link capacities are assumed with different costs (1 Gbps, USD 0.05/ GB), 400 Mbps, USD 7/GB), (100 Mbps USD 0.09/GB) as in [178]. Link capacities are allocated according to the geographical distances between the MEC hosts. The amount of CPU resources distributed over the MEC hosts are based on utilitarian resource distribution algorithm as in the Section 3.3.5.

In previous chapters, MEC hosts are assumed to be independent. Thus, when a mobile device sends an offloading request, if the serving MEC host does not satisfy the requirements, the offloading request is rejected, otherwise, the request is accepted. In this chapter, MEC hosts collaboration is introduced in which user application can be instantiated in any suitable MEC hosts, not necessarily the serving MEC host.

As mentioned in Section III, Balas-Geoffrion algorithm is not efficient in solving the MEC host selection problem as shown in Figure 5.4. Because of the constant negative coefficients, Balas-Geoffrion selects a random value as the next free variable to iterate.

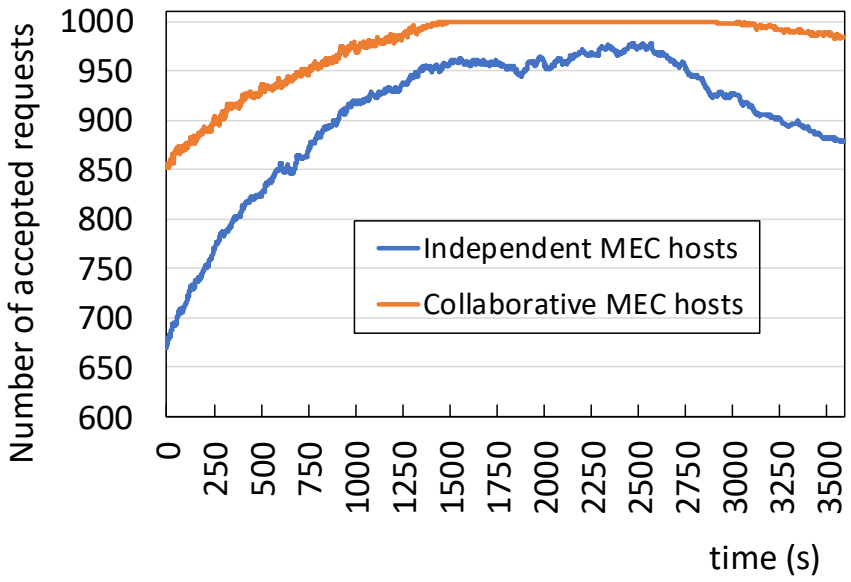

Figure 5.5 Comparison of accepted requests in independent MEC hosts and collaborative MEC hosts methods

Thus, the number of iterations to reach out the final solution is random in Balas-Geoffrion algorithm. On the other hand, the number of iterations is less in the extended algorithm. However, a study increment in the number of iterations can be found except 3 and 4 offloading requests. As shown in Figure 5.4, the extended algorithm outperforms Balas-Geoffrion algorithm in the number of iterations required to select the optimal solution in the MEC host selection problem.

5.4.1 Benefit of Collaboration Between MEC Hosts

Figure 5.4 shows the benefit of MEC hosts collaboration method compared to independent method during the morning peak hour. It shows the improvement in the total number of accepted requests in collaborative method compared to the independent MEC hosts method. All the offloading requests during the middle of the morning peak hours are accepted. During the initial stages and the later stages of morning peak hours, a maximum of 15% and 2% of the total requests in the collaborative method are rejected, respectively. Since most of the resources are deployed near to the CBD based on the utilitarian algorithm, more tasks re rejected during the morning peak hours, as most users start their commute from out of the CBD. The total number of accepted requests is improved from 87% in the independent method to 95% in the collaborative method.

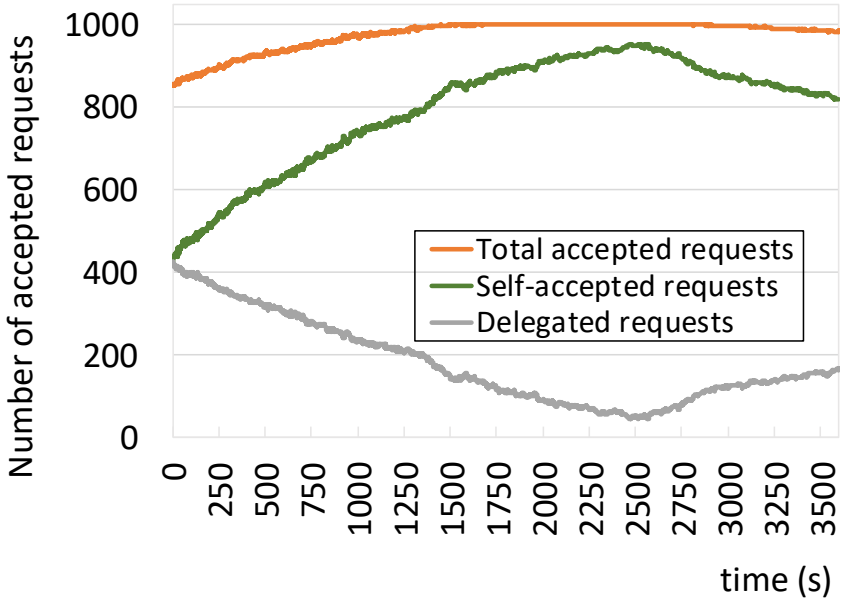

Figure 5.6 Details of accepted requests in collaborative MEC hosts method

According to our analysis, the link capacity is the limiting factor in causing the number of requests rejected in the MEC hosts collaboration method, i.e., if there is unlimited link capacity, all the requests will be accepted in the MEC hosts collaborative method.

5.4.2 Detail of MEC Hosts Collaborations

Figure 5.6 shows the details of accepted requests in the MEC host collaboration method. Self-accepted requests of the MEC host are the requests accepted by the MEC host that are requested by the devices in the serving coverage area of the MEC host.

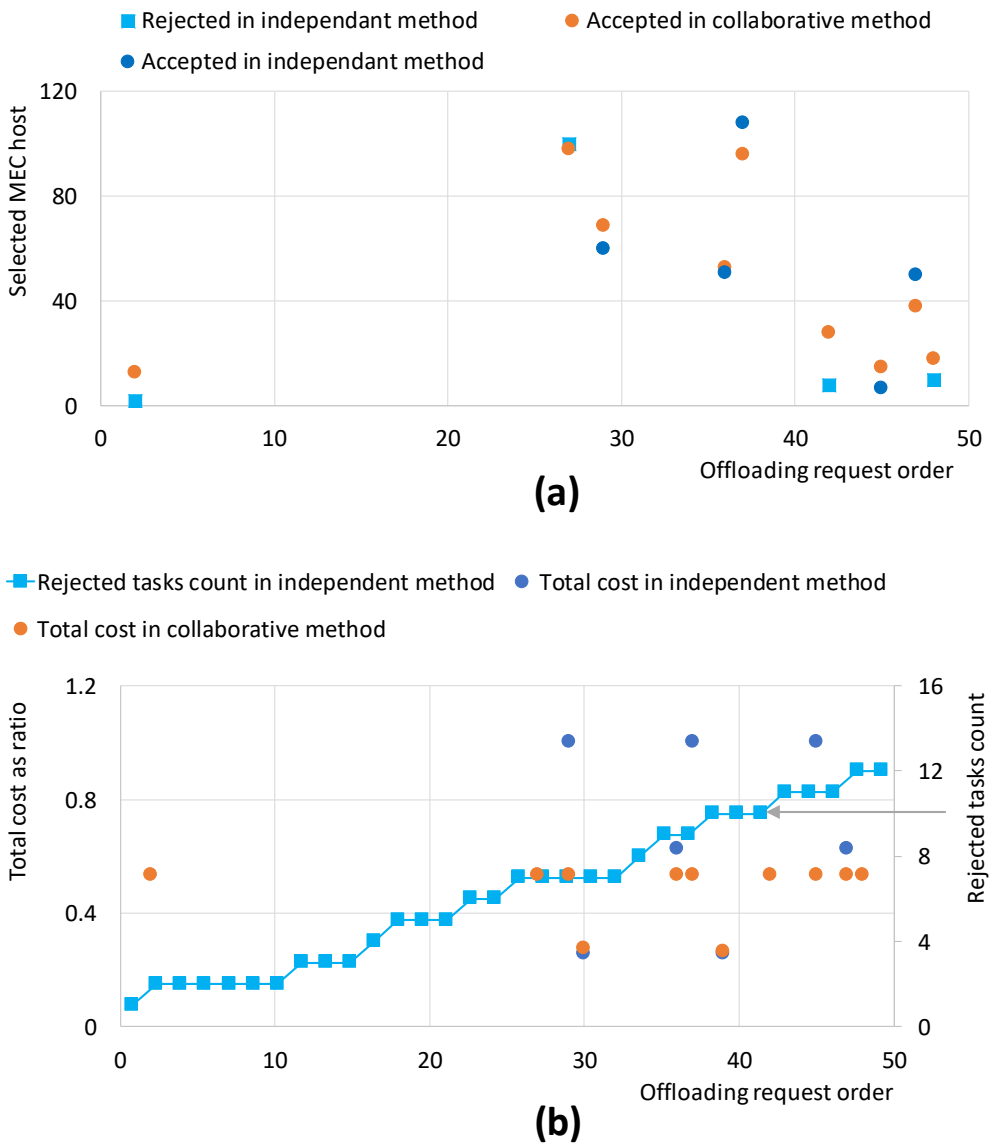

Figure 5.7 (a) selected MEC hosts **(b)** total costs; for first fifty offloading requests at time 100 s

Delegated requests of the MEC host are the requests transferred to delegated MEC host by the serving MEC host. Delegation may occur due to the resource limitations in the serving MEC host or it is cheaper to host the user application in the delegated MEC host than the serving MEC host. The total number of delegated requests are higher in the early part of the morning peak hours. This indicates that most of the tasks are migrated from the outer MEC hosts to the centre MEC hosts to be served since most of the users are located outside of the CBD during this time.

Once users start moving towards the CBD, a number of delegated tasks start to decrease and a number of self-accepted requests increases. When the users move very close to CBD, i.e., after time 2500 s, the number of delegated tasks increases and self-accepted requests decreases. This shows that tasks are being migrated to outer MEC hosts for centre MEC hosts to server more requests. However, the total number of accepted requests does not change much after time 1250 s.

5.4.3 Cost vs Number of Accomplished Tasks

As first-come-first-serve scheduling is utilised at the MEC orchestrator, offloading requests are processed in the sequential order they are requested. Figure 5.7(a) shows the deviations in MEC host selections of the first 50 offloading requests at time 100 s, i.e. only the variations in both independent and collaborative methods in MEC host selections are displayed in the figure. As observed, there are some requests rejected in independent method, for instance, 2nd, 27th and 42nd requests are being rejected. On the other hand, all the task requests are being accepted in the collaborative method. For the 29th request, 60th and 69th MEC hosts are being selected in independent and collaborative methods, respectively. This indicates that selecting 69th MEC host is cost-efficient in collaborative method than selecting 60th MEC host for the 29th request as can be seen in Figure 5.7(b). Further, the number of variations in MEC host selection increase with the number of offloading requests.

Figure 5.7(b) shows the total cost of providing offloading services for the requests. The cost is lower in collaboration method than the independent method except for 30th and 47th requests. The slight increment in the cost of both requests is due to that the selected MEC host is in higher utilization in the collaborative method than independent method. In other words, 29th request is being served at the 69th MEC host in the

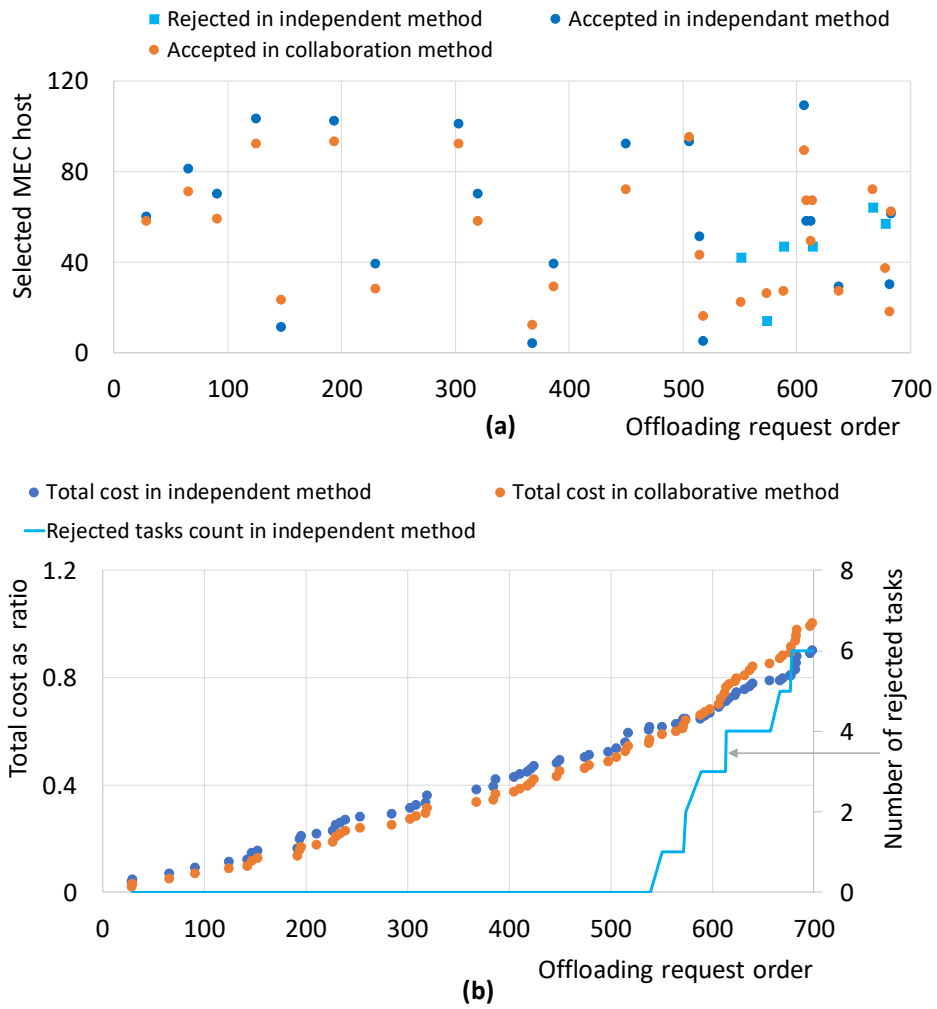

Figure 5.8 (a) selected MEC hosts **(b)** total costs; for first seven hundred offloading requests at time 2400 s.

collaborative method for cost efficiency. This will increase the utility of the 69th MEC host. When the 30th request is also being served at the 69th MEC host, the cost is slightly higher in the collaborative method than the independent method. On the other hand, the number of rejected tasks increases with the number of offloading requests in independent method while there is no rejection in the collaborative method. The number of tasks rejected tend to reduce the cost in the independent method since the MEC host utilization is less compared to the collaborative method.

Figure 5.8(a) shows the deviations in MEC host selections of the first 700 offloading requests at time 2,400 s. While all the first 700 tasks requests in the collaborative method are accepted, however, 6 offloading requests are rejected in the independent method as shown in the blue line in Figure 5.8(b). Further, the first task request rejection is at the

2nd request for 100 s as in Figure 5.7(a) and 551st request for 2400 s as in Figure 5.8(a). Since most of the users are outer area during the early part of the peak hour and fewer resources are allocated in the outer MEC hosts, the task is being rejected at the early stage, i.e., at time 100 s. On the other hand, since most users have moved to the inner area where most resources are allocated on the inner MEC hosts, all the requests are accepted, up to 550th requests, in mid of the morning peak hour, i.e. at time 2400s. In addition, the number of MEC hosts collaborations increases with the number of offloading requests increases as can be seen in Figure 5.7(a) and Figure 5.8(a).

Figure 5.8(b) shows the total cost of providing the offloading services, as the ratio of maximum cost, with the request arrival order in both methods at time 2400 s. It also shows the number of rejected offloading requests in independent method. During the early stage of the morning peak hours, i.e., at 100 s, 12 tasks are rejected within the first 50 requests. However, on the other hand, only 6 task requests were rejected within the first 700 requests during the mid of the peak hour, i.e., at 2400 s. This is a clear indication that our utilitarian resource distribution algorithm allocates more resources to the centre of the deployment to accept more task requests. Thus, maximizing the quality of service of the MEC service providers in terms of the number of tasks accepted while satisfying the latency requirement.

After the first three tasks are rejected by the independent method, the cost of instantiating the application in collaborative method increases as compared to the independent method from the 593rd offloading request onwards as shown in Figure 5.8(b). The cost gap further increases as the number of tasks rejected at the independent method increases. This implies that when the number of requests accepted in the collaborative method increases beyond the margin, the cost of providing the offloading services also increases as compared to the independent method. This is the trade-off between the cost of providing the offloading services to the quality of service in terms of the number of requests accepted. If the service providers want to increase the quality of service beyond the margin, then the cost increases.

5.5. Summary

The MEC hosts selection problem in a collaborative MEC system is a challenging task for the MEC service providers in order to satisfy users' QoS requirements while minimizing service costs. In this work, the MEC hosts selection problem is formulated as a special case of binary programming problems, which minimizes the total cost incurred by the MEC service providers without compromising on QoS requirements. An extended Balas-Geoffrion algorithm is developed to solve this problem as the additive algorithm minimizes the time complexity. Further, it provides the near-optimal solution, even if the calculations stop before all the possible solutions are enumerated. The findings in this chapter show that the extended algorithm outperforms the original Balas-Geoffrion algorithm in the number of iterations required to reach the optimal solution in the context of the special case of binary programming similar to MEC host selection problem. Furthermore, the trade-off between the increased costs of the collaboration methods to the number of tasks rejected in the independent method is analysed. Even though MEC hosts collaborative method increases the number of tasks accomplished, the cost of serving a request increases compared to independent method beyond a threshold margin. This trade-off will be helpful for MEC service providers to finalise their service level agreements with customers.

6 Mobility-aware Energy Optimization in MEC Host Selection

In MEC, user application migration needs to be considered taking into account of user mobility in order to provide service continuity. In this Chapter, the energy efficiency of MEC host selection and user application migration based on the type of applications in a hierarchical MEC network deployment is analysed.

6.1. Introduction

Mobile devices can offload their computationally intensive tasks to the MEC servers on the edge of the network, rather than utilizing the servers in the data centres. However, the deployment of MEC hosts within the radio access network can potentially introduce new challenges. While the computation offloading of the user application is in progress, i.e., the user application is running on the MEC host, the user may be moving around and be connected to a different radio node, i.e., handover. Regardless, the services still need to be provisioned seamlessly, i.e., the connectivity between the device application and the user application needs to be maintained. As the user moves away from the current serving MEC host, the data offloading latency between the device application and the user application is likely to extend.

A user may move within the coverage area of a MEC host, resulting in an intra-host user mobility scenario, or a user may move around radio nodes which are connected to entirely different MEC hosts, resulting in an inter-host user mobility scenario [179]. In the case of intra-host user mobility, the user application does not require migration to another MEC host. However, in the case of inter-host user mobility, the user application might have to be migrated to another MEC host in order to maintain the latency requirements of the offloading application.

The MEC host selection problem [51], discussed earlier in Chapter 5 becomes compounded due to the fact that user mobility introduces another important dimension to the MEC host selection problem, giving rise to the combined problem of *MEC hosts selection and user application migration*.

On the other hand, since MEC host servers are going to be deployed within the radio access network, operational costs will be very high compared to the operational costs at the dedicated cloud datacenters. For instance, operational cost at a MEC host located at the city centre of a dense urban town will be much higher. In addition, as presented in Chapter 4, maximizing energy efficiency, decreases the operational cost. Therefore, MEC service providers will also have to manage the energy efficiency of MEC hosts selection and user application migration problem as it offers significant cost-benefits. *To the best of our knowledge, minimizing total energy consumption in the MEC hosts selection and user application migration problem considering user application's latency requirement and user mobility has yet to be fully investigated*. This chapter focuses on the investigation of energy consumption minimization in MEC hosts selection and user application migration problem.

The remaining sections of this chapter are organized as follows. Section 6.2 introduces related work in different types of latency requirement of mobile application, application placement in edge computing and user mobility predictions. Section 6.3 models the system and formulate the minimization problem as the shortest path problem. Section 6.4 discusses the simulation setup and results of our proposed methods and Section 6.5 provides a summary of the main findings of this chapter.

6.2. Related Works in Literature

Mobile applications can be mainly categorized into the hard deadline and soft deadline applications depending on the maximum tolerable latency requirements [179]. The latency requirement of some applications such as remote medical surgery, remote desktop applications, industry automation, tele-control applications is very critical to those applications effectiveness [135], i.e., *hard deadline*. In other words, each task of a hard deadline requirement application should be executed on or before the preset deadline requirement. In use cases such as video analytical applications, AR applications, e.g., visitor guideline applications of a museum, etc., the delay is tolerable to a certain extent.

Thus, the latency requirement is flexible resulting in the application having a *soft deadline*.

In addition to specific target deadlines associated with the execution of applications, some existing work on application placement and scheduling in MECs have considered applications with two components, one running on the cloud which can be either by the MEC host or the cloud data centre and the other running on the mobile device [167] [168]. Another body of existing work usually involves only two physical computational entities, i.e., the mobile device and the cloud [166] [67]. Multi-component applications that can be deployed across one or multiple levels of MECs and core cloud(s) have been considered in [170], and a task placement algorithm of an edge computing orchestrator is also proposed in [171]. Machine learning approaches for computation offloading have been proposed in the literature [180] [181]. However, MEC network topology is not considered for application placement.

As discussed in detail in Chapter 3, mobility traces of users have been extensively analyzed in the literature in order to gain insights into humans' mobility patterns and to forecast their future locations accurately. Researchers have found that user mobility can be predicted up to 93 % [25]. Recent literature reports that human tend to perform Lévy walks [94] with heavy-tail flight distributions. However, in reality, user mobility tends to be correlated and strongly dependent on users' personal and social characteristics and behaviours as well as environmental parameters [140]. Correlated user mobility model is proposed to derive user trajectories of morning peak hour in [179]

6.3. System Models and Methodologies

6.3.1 System Models

The device applications running on the mobile devices offload their computation-intensive tasks to the MEC host through a 5G mobile network. Each device application can have many computational tasks to be completed within a certain delay constraint. Each computational task can be described in three terms: $T = \{r^C, r^N, \tau^M\}$. In a computational task, r^C is the computational resource required for accomplishing this task, which is quantified by the number of CPU cycles; r^N on the other hand denotes the

size of the input data for the computation, which include program codes, input files, etc., which is then quantified by the number of bits. τ^M is the maximum tolerable latency of the computational task, which is quantified by nanoseconds. These parameters are related to the nature of the applications and should be pre-specified by the MEC applications.

User applications on mobile devices can be grouped into three different case types, with each having distinct latency requirements and mobility patterns to formulate the models and algorithms discussed in the next section. Some augmented reality applications such as applications guiding visitors in museums and exhibitions, do not require stringent latency requirement and fixed in one location. These kinds of user applications are considered as Case 1 user applications that do not require hard deadline and no user mobility or mobility limited to within a MEC host (intra-host user mobility). User applications such as remote medical surgery have stringent latency requirement but with intra-host user mobility. These kinds of applications are considered as Case 2 user applications that have stringent deadline requirement with intra-host user mobility. Mobile applications such as remote desktop application while commuting has greater mobility as well as stringent latency requirement. These kinds of applications are considered as Case 3 applications which have stringent deadline requirement with inter-host user mobility.

A set of MEC hosts are denoted as $\boldsymbol{H} = \{1,2,..h..,H\}$. Each MEC host has a maximum compute capacity, measured in CPU cycles per second, and energy-efficiency metric, measured in joules per CPU cycle. A hierarchical MEC network deployment, as shown in Figure 1.2 is considered, deployed at the radio node, i.e., level 1, network aggregation points, i.e., level 2, and at the edge of the core network (i.e., level 3) [54]. $\boldsymbol{I} : \boldsymbol{I} \subseteq \boldsymbol{H}, i \in \boldsymbol{I}$ is denoted as the MEC hosts deployed at the radio nodes. One of these will be the default serving MEC host for the mobile device with application computation offloading, i.e., a mobile device will be always within the serving coverage area of a MEC host i. $\mathbf{J} : \mathbf{J} \subseteq \boldsymbol{H}, j \in \boldsymbol{J}$ and $\boldsymbol{J} = \{1,2,..,j,..J\}$ is denoted as the set of MEC hosts capable of hosting the user application considering the application's rules and requirements such as special type hardware requirement such as graphics processing unit (GPU).

6.3.2 Energy Modeling

The energy consumption of a MEC host is jointly determined by the usage of the CPU, storage, memory, and network interfaces. Since the CPU contribution is dominant among these factors in MEC host, it is the main focus in the literature [57]. Two models are widely used for the energy consumption: one model is based on the Dynamic voltage and frequency scaling (DVFS) [182] technique while the other model is based on an observation in [183] [154] that the server-energy consumption is linear to the CPU utilization ratio, which depends on the computation load. The energy consumption of the server at the MEC host based on CPU utilization is modelled as in Eq. (6.1), assuming a fixed running frequency, where E_{max} is the energy consumption for a fully utilized server, α is the fraction of the idle energy consumption and u denotes the CPU utilization ratio. Idle power consumption is mainly due to the power consumption in the power delivery and cooling infrastructure.

$$E_s = \alpha E_{max} + (1 - \alpha) E_{max} u \tag{6.1}$$

The total energy consumption of an offloaded computational task comprises two parts: (a) data offloading energy and (b) task computation energy. Data offloading energy consumption is due to the uploading and downloading energy of data. Since our research is focusing on minimizing the energy consumption in a MEC network, data offloading energy through the wireless network is not considered as it is the same for all the cases of MEC host selection. Data offloading energy consumption is $r^N e_{i,j}^N$, where $e_{i,j}^N$ is the energy efficiency of data offloading between MEC hosts i and j, measured in joules per bit. Similarly, task computation energy is $r^C e_j^C$, where e_j^C is the energy efficiency of task computation at the MEC host j, measured in joules per CPU cycle. Thus, the total energy consumption of the task is $r^N e_{i,j}^N + r^C e_j^C$.

The device application could offload multiple tasks, i.e., n number of tasks. The total energy consumption of application computation offloading in the MEC network is $E_j = (\overline{r^N} * e_{i,j}^N + \overline{r^C} * e_j^C) * n$, where $\overline{r^N}$ and $\overline{r^C}$ represent the average of the quantities over n number of tasks.

6.3.3 Latency Model

The total latency of the computation offloading comprises two parts: (a) data offloading latency and; (b) task computational latency. Data offloading latency is determined as $\frac{r^N}{x_{i,j}^N}$, where $x_{i,j}^N$ is the network bandwidth allocated to the user in bits per second during offloading of the task. Task computational latency is then $\frac{r^C}{x_j^C}$, where x_j^C is the compute resource allocated to the user in the host j, measured in CPU cycles per second. However, it is not always possible to host the user application in the same MEC host due to user mobility, which causes the latency variations.

6.3.4 User Mobility

Mobile users move around the MEC network during the application computation offloading. The movements may cause users moving into the coverage area of a different default serving MEC host. To consider the variations in resource availability and resource allocations in different MEC hosts, The time spent by each user under the coverage area of a MEC host is considered in time periods $p, p \in \mathbf{P}$, $\mathbf{P} = \{1,2,..,p,p+1,...P)$, where p and $p+1$ are consecutive time periods. The user application may be migrated to another MEC host due to the optimization process of the MEC system initiated by MEC network, even if the user stays under the coverage area of the same MEC host over a time period. However, in this chapter, it is assumed that only one MEC host is selected during a specific time period p and the user application is not migrated during this time period since the user will be under the coverage area of the same MEC host during that period p.

Thus, the offloading energy consumption during one time period is:

$$E_{p,j}^O = \left(\overline{r_p^N} * e_{p,i,j}^N + \overline{r_p^C} * e_{p,j}^C \right) * n_p \qquad (6.2)$$

where $p \in \mathbf{P}$, $j \in \mathbf{J}$ and n_p is the number of tasks offloaded during the period p. The parameters $\overline{r_p^C}$ and $\overline{r_p^N}$ are the average of the computational resource required and the average of the size of the input data of the offloaded tasks respectively. The parameters $e_{p,i,j}^N$ and $e_{p,j}^C$ are the energy efficiency of data offloading in joules per bit and the energy efficiency of task computation joules per CPU cycle respectively, during the time period p.

In addition to the offloading energy, migration energy due to the migration of the user application from one MEC host to another MEC host needs to be considered. As in [184], data offloading energy is dominant of all other energies involved in live migration. The migration energy consumption of the user application at the end of each period p is considered as:

Algorithm 6.1 Algorithm to calculate feasible MEC hosts for user application offloading for a user

$$E_{p,j}^{M} = r_{p,j}^{M} \, \mathrm{e}_{\mathrm{p,j}}^{\mathrm{M}} \tag{6.3}$$

where $r_{p,j}^{M}$ is data to be migrated in bits and $\mathrm{e}_{\mathrm{p,j}}^{\mathrm{M}}$ is migration energy efficiency in Joules per bit, where $p \in \mathbf{P}$, $j \in \mathbf{J}$ and $E_{p,j}^{M} = 0$ for $p = P$.

6.3.5 Objective of Optimization

Our objective is to minimize the total energy consumption of the application computation offloading during the morning peak hour:

$$\min \left\{ \sum_{p=1}^{P} \sum_{j=1}^{J} y_j \, (E_{p,j}^{O} + E_{p,j}^{M}) \right\} \tag{6.4}$$

where y_j is an indicator variable to indicate whether the MEC host j is selected for hosting the user application. If it is selected, $y_{\mathrm{j}} = 1 \; or \; 0$ otherwise. The solution to the problem in (4) is the set of MEC hosts selected in each period of times that, in turns, indicates the user application migration path.

Input: *user mobility, MEC network information*

Output: *feasible_hosts – feasible MEC hosts*

Process
Predict time periods **P**
FOR EACH period $p \in \mathbf{P}$
Get default connected host $i \in \boldsymbol{I}$
Get all possible MEC hosts $\boldsymbol{J}$
FOR EACH possible MEC host $j \in \boldsymbol{J}$
IF MEC hosts selection criteria is satisfied
Calculate energy using Eq. (6.2)
Add j to *feasible_hosts* with energy

The *computational intensity* (δ_p) for a time period p is defined as the ratio of average computational resource required to the average of the input data size over that period:

$$\delta_p = \frac{\overline{r_p^C}}{\overline{r_p^N}} \tag{6.5}$$

CI is a metric to describe the type of application based on computational resource requirement and data offloading requirement. A higher CI indicates computation-intensive tasks and a lower CI indicates bandwidth-intensive tasks.

6.3.6 Feasible MEC Hosts Selection

Algorithm 6.1 explains the procedure to get feasible MEC hosts for a user application. First, time spent under each default MEC serving host **P** is derived based on the user mobility prediction. Then for each period $p \in \mathbf{P}$, all the possible MEC hosts that satisfy the rule and requirement of the applications such as special hardware requirement is derived. For each possible MEC hosts check whether latency constraint is satisfied. If the latency constraint is satisfied, calculate the total energy consumption as in Eq. (6.2) and add to the list of feasible MEC hosts. Thus, Algorithm 6.1 outputs all the feasible MEC hosts during the application computation offloading duration.

Case 1 – User Applications with Soft deadline and Intra-host Mobility

Since the user does not move to a different serving MEC host during morning peak hours, the whole hour could be considered as one time period. Migration energy is assumed to be 0 in this context because of no migration. Hence, offloading energy consumption becomes the total energy consumption E_j. Thus, the objective function in Eq. (6.4) can be minimized to:

$$min\left\{\sum_{j=1}^{J} y_j E_j\right\} \tag{6.6}$$

In this context, we need to find the MEC host, starting from the default serving MEC host to all MEC hosts in the hierarchy level, which minimizes the total energy consumption for the user application. If a user application needs to be hosted in another MEC host j instead of the default serving host i, the total energy consumption at i should be greater than the total energy consumption at j:

$$E_i > E_j$$

$$\overline{r^C} * e_i^C > \overline{r^N} * e_{i,j}^N + \overline{r^C} * e_j^C$$

$$\frac{\overline{r^C}}{\overline{r^N}}(\delta) > \frac{e_{i,j}^N}{e_i^C - e_j^C} \tag{6.7}$$

Equation (6.7) reveals the MEC host selection criteria for Case 1. The parameter δ depends on the user application and the right-hand side of the Eq. (6.7) depends on the current condition of the MEC network. Algorithm 6.1 provides all the feasible MEC hosts. The MEC host with minimum total energy consumption among all the feasible MEC hosts is the solution for MEC host selection problem for Case 1.

Case 2 – User Applications with Hard Deadline and Intra-host Mobility

Since the requirement of the user application is a hard deadline, the maximum tolerable latency constraint is imposed onto the objective function in Eq. (6.6). The total latency, i.e., the sum of the data offloading latency and task computational latency, should be less or equal to the user application's maximum tolerable latency.

Therefore, the maximum tolerable latency constraint can be calculated as:

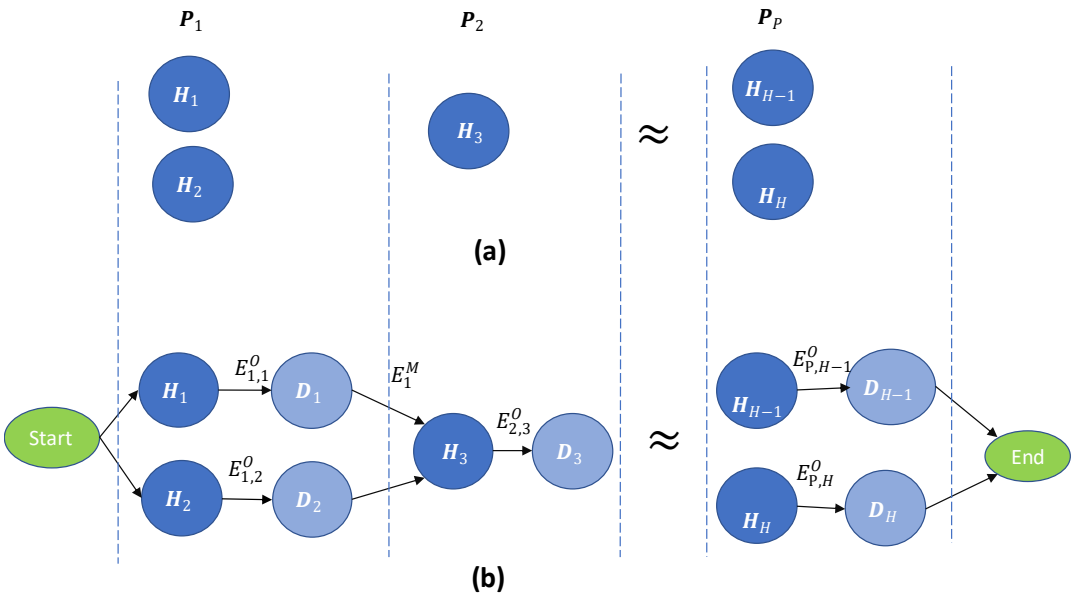

Figure 6.1 Shortest path problem formulation for MEC hosts selection and user application migration problem; (a) is the feasible MEC hosts in each period and (b) is the derived shortest path problem

$$\frac{\overline{r^N}}{x^N_{i,j}} + \frac{\overline{r^C}}{x^C_j} \leq \tau^M \tag{6.8}$$

By considering Eq. (6.7) and Eq. (6.8), it can be derived from the criteria as in Eq. (6.9) to select a MEC host to instantiate the user application:

$$\frac{\tau^M}{\overline{r^N}} \geq \frac{x^N_{i,j}}{x^C_j \left(e^C_i - e^C_j\right)} + \frac{1}{x^N_{i,j}} \tag{6.9}$$

Equation (6.9) reveals the MEC host selection criteria for Case 2. The left-hand side of the criteria in Eq. (6.9) is based on the type of user applications and the right-hand side of the criteria, i.e., resource allocations and energy efficiency metrics depending on the current condition of the MEC network. Selecting the MEC host with minimum total energy consumption among the feasible MEC hosts resulted from Algorithm 6.1 is the solution for Case 2.

Case 3 – User Applications with Hard deadline and Inter-host Mobility

While the user is moving to the coverage areas of different MEC hosts, resource allocations and energy efficiency metrics vary for each period, thus the total energy

consumptions. Since the hard deadline requirement, Eq. (6.9) is the MEC host selection criteria for Case 3 also. Algorithm 6.1 outputs all the feasible MEC hosts during the application computation offloading duration. However, a local minimum in each period does not lead to a global minimum because the migration energy is involved in between each period. This indicates that different combinations of MEC hosts between each period need to be considered rather than considering only local minimum in each period. This introduces the difficulty in finding the suitable sets of MEC hosts that minimize the total energy consumption while satisfying the maximum tolerable latency constraint.

To tackle this issue, feasible MEC hosts for computation offloading in each period can be found. For instance, Figure 6.1(a) shows the list of possible MEC hosts in each time period during the offloading process of the user application. For example, H_3 is the only feasible MEC host during the period P_2 that satisfies the rules and requirements of the user application including the maximum tolerable latency requirement.

Then by considering all possible combinations of MEC hosts between each period, the suitable MEC host in each period needs to be found to host the user application and migrate accordingly. To solve this issue, a directed graph from the MEC hosts selection and user application migration problem is derived. The MEC hosts are represented by nodes and the energy consumptions are represented by edges of the graph. A dummy MEC host node is added next to each of the MEC host node to incorporate task computation energy in the directed graph. Task computation energy at the MEC host is added to the edge of these nodes. Figure 6.1(b) shows the directed graph derived from the feasible MEC hosts. Further, the start and end nodes are added to convert this to the shortest path problem in the directed graph. Dijkstra's algorithm can be utilized to solve the shortest path problem since it is a directed graph with non-negative weights.

6.4. Simulation and Results

6.4.1 Simulation Setup

MEC hosts selection and user application migration problem is simulated in an urban area of 2 km × 2 km served by a mobile wireless network in which radio nodes are spaced 200 m apart. It is assumed a total of 1,000 users, each with a mobile device moving in

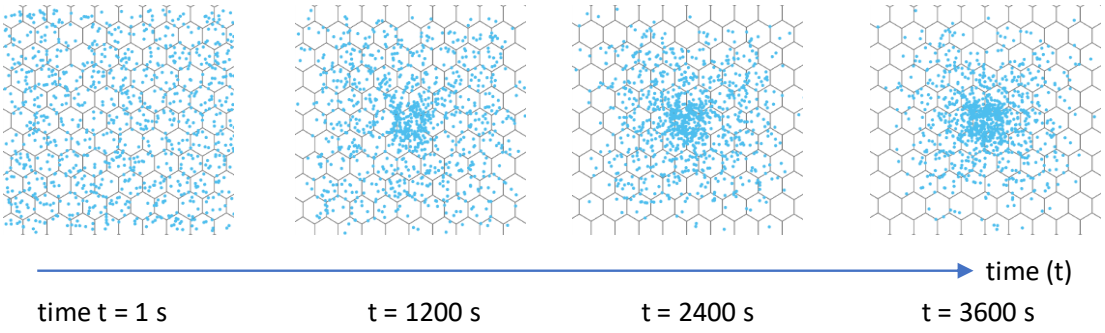

Figure 6.2 users' location at different time

vehicles for an hour, i.e., 3,600 seconds, during the morning peak hour rush. We used the correlated mobility model developed in Section 3.3.1 to produce the users' trajectories during morning peak hours in which users start from the outer suburbs of the city and then gradually move towards the central business district. From the user trajectories, we calculate the default serving MEC host and the time period spent under each MEC host. We assume that each device application offloads a computation-intensive task per second. To compare two applications and specify whether first application is more compute-intensive application or bandwidth-intensive application based on CI value, we need to have a reference value, i.e., fixed input data size or compute resource. We fix the input data size in this research. In other words, we fix the data offloading energy. The input data size and the maximum tolerable latency are fixed to 1 Mbits and 14 ms, respectively. The maximum tolerable latency is selected to cater to virtual reality applications [179] [142]. CI is varied to analyze the dynamics of energy efficiency in MEC hosts selection and user application migration problem.

Different types of servers are deployed in different levels of the MEC network. The MEC network is designed to be three levels. Level 1 MEC host is deployed at each radio node with less capability (CPU capacity 12 GHz with 250 watts power consumption per server) whereas level 2 (57.6 GHz, 600 watts) is the network aggregation point where the MEC hosts are deployed at. Level 3 (207 GHz, 1500 watts) is the edge of the core where a pool of servers with richer compute resources are deployed at. These MEC hosts locations are based on the results of the utilitarian resource distribution algorithm proposed in Section 3.3.5. Physical servers are selected by considering the energy efficiency selection process based on the methodology proposed in Section 4.3.2. The network uplink capacities between level 1 and level 2, and between level 2 and level 3 are 2.5 Gbps (XG-PON) and 10 Gbps (NG-PON2) respectively [185]. The corresponding energy efficiency metrics for the above links are 21.4 nano joules per bit and 25.2 n joules per bit as given in [186]. The maximum CPU resources and network transmission bandwidth at each level are 6 GHz (level 1), 12 GHz (level 2), 20 GHz (level 3) and 600 Mbps (level 1 to level 2) and 100 Mbps (level 2 to level 3), respectively. CPU resource allocations randomly vary between 5 GHz – 10 GHz, 7.5 GHz – 15 GHz and 10 GHz – 20 GHz in level 1, 2 and 3 MEC hosts, respectively. Network bandwidth allocations vary between 400 Mbps – 800 Mbps and 100 Mbps – 200 Mbps in the network between level 1 and level 2, and between level 2 and level 3, respectively. The resource allocation is maintained in such a way that when mobile users gradually move towards the CBD,

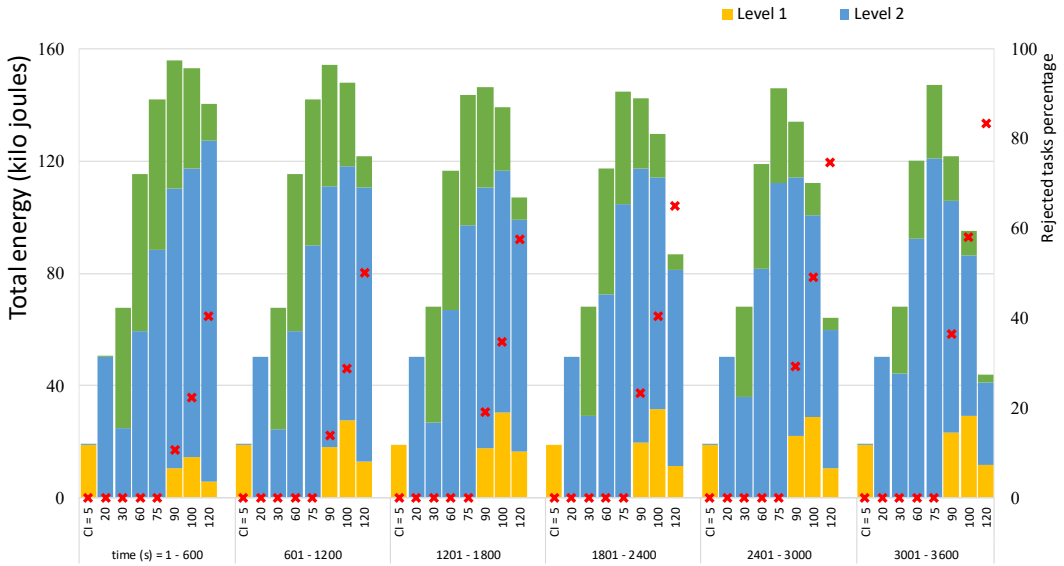

Figure 6.3 The total energy consumption of the MEC network and the rejected tasks percentage for Case 3.

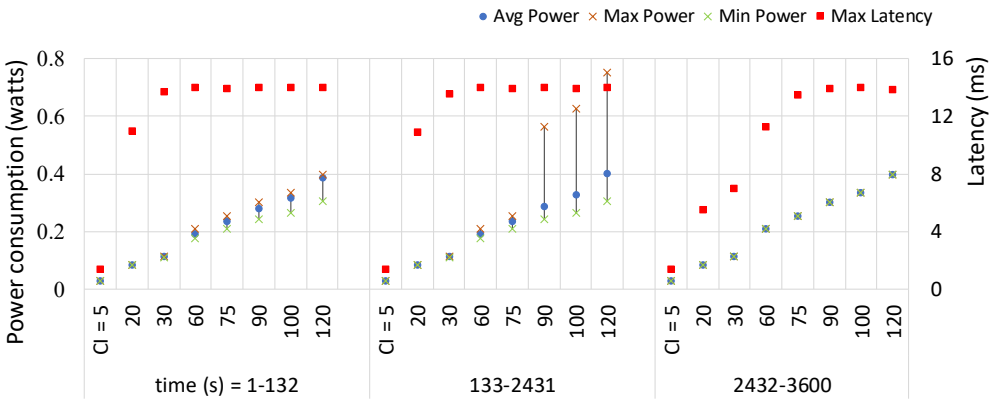

Figure 6.5 power consumption variations and the task accomplishment latency of a representative user (10th user out of 1,000)

resource allocations per application tends to skew toward the minimum value of the allocations.

In the case of intra-host mobility, i.e. Case 1 and Case 2, MEC host selection is trivial. Algorithm 6.1 outputs all the feasible MEC hosts. The MEC host with minimum total energy among the feasible MEC hosts will be selected for hosting the user application. The MEC host selection criteria for Case 1 and Case 2 are Eq. (6.7) and Eq. (6.9) respectively. The left-hand side of both conditions depend on the type of the application and right side depend on the current condition of the MEC network. Thus, accurate predictions of MEC network conditions improve the energy efficiency in MEC host selection in both cases. Since Case 3 is the most challenging case out of three cases, we analyze Case 3 in detail in the following subsections.

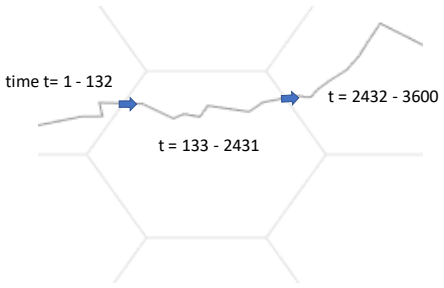

Figure 6.4 User trajectories of a representative user (10th user out of 1,000) with the time spent in each radio nodes

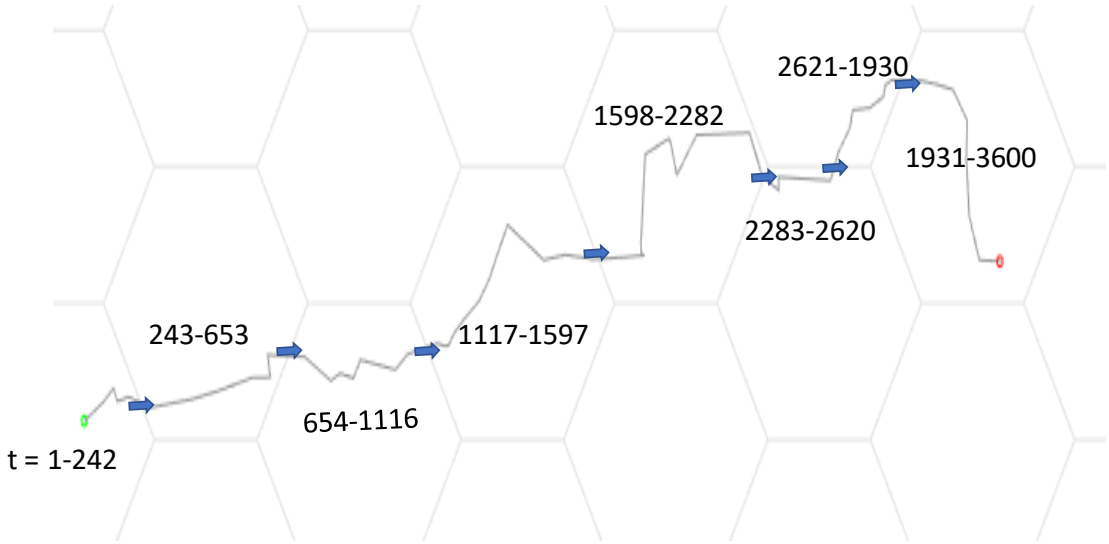

Figure 6.6 User trajectories of a representative user (100th user out of 1,000) with the time spent in each radio nodes

6.4.2 Total energy consumption in MEC network

(a) shows the user locations on the simulation area in different time instances during the morning peak hour. It can be seen that users start their commute randomly from their initial locations, mostly from their homes, then move towards the central business district (CBD) for their day-today activities like work, education etc., during the morning peak hour. At the end of the morning peak hour, most of the users stay at the CBD.

Figure 6.2(b) shows the total energy consumption of the offloaded MEC applications in the MEC network for selected CI values in every 10-minute time interval of the morning peak hour for Case 3. These CI values are selected to show the dynamics of MEC hosts selection and user application migration process.

For a CI value of 5, almost all of the offloading requests are handled by the MEC host at level 1 and the user application is being migrated to each MEC host at level 1 while the user is moving from the coverage area of a MEC host to another MEC host, similar to a moving personal cloud. This indicates that hosting low CI value (high bandwidth requirement) applications at level 1 provides energy efficiency. Hosting user applications at level 2 and level 3 are energy-efficient for CI=20 and CI=30, respectively. This implies that hosting computation-intensive applications at a higher level is energy efficient. If

user applications do not require hard deadline, which described in Case 1 in Section III E, then hosting user applications in level 3 is energy-efficient beyond CI value of 20.

However, for Cases 2 and 3 with hard deadline requirements, this trend changes. Since the latency requirement cannot be satisfied at level 3, for CI values between 30 to 75, selecting ME hosts at level 2 gives better results in terms of energy efficiency. The resource allocations and the energy efficiency of the MEC network during the period are the key factors in selecting which network level to host the offloaded tasks. Beyond CI value of 75, some of the applications need to be offloaded to level 1 to satisfy the latency requirement, even though hosting a high computation-intensive application in level 1 consumes very high energy. Most importantly, beyond the CI value of 75, offloading task requests are being rejected by the MEC network due to the inability to handle the hard latency requirements of the tasks during that period. The rejected ratio, therefore, increases with the CI value. Further, the rejected ratio increases with time, indicating that more requests are being rejected when users are moving towards the CBD.

When users move toward the CBD, the hosting of user applications tends to move toward the lower level. For instance, for a CI value of 30, 36% of the tasks were hosted in level 2 during the first period (t = 1 – 600), which then increases up to 65% of the tasks at the end of the time period (t = 3001 – 3600). Similarly, 9% of the accepted tasks were served at level 1 for a CI value of 100 during the start of the simulation. It then increases to 30% over time. On the other hand, hosting offloaded tasks at level 3 drops from 23% to 9% over time. This implies that when users are moving toward the CBD, user applications are being migrated to level 1 to satisfy the latency requirement.

6.4.3 Power Consumption of Individual Users

Figures 6.3 and 6.4 show the trajectories of two different users, their applications' power consumption and service delivery latencies for different CIs during their commute for Case 3. Figure 6.3(a) shows the trajectory of a representative user (10^{th} user) who moves through three different default MEC host coverage areas, with time spent on each coverage areas indicated in the figure. It can be observed that this user spent more time on the second MEC host during the user's commute.

Figure 6.3(b) shows the average power consumption of the selected user with different CI values analyzed for 200 different days. The average power consumption increases with

CI value, as expected. However, high skewness is visible during the 133 – 2431 period for CI 90, 100, 120 values. This is due to the reason that only a few requests were served in level 1 due to latency constraints, which leads to higher energy consumption. For instance, 4%, 49% and 25% of user applications are hosted in level 1, level 2 and level 3, respectively for a CI value of 90. The remaining 22% of the requests were dropped. The maximum latency in serving the user application increases with CI and stays within 14 ms, as the applications' hard deadline requirement is 14 ms.

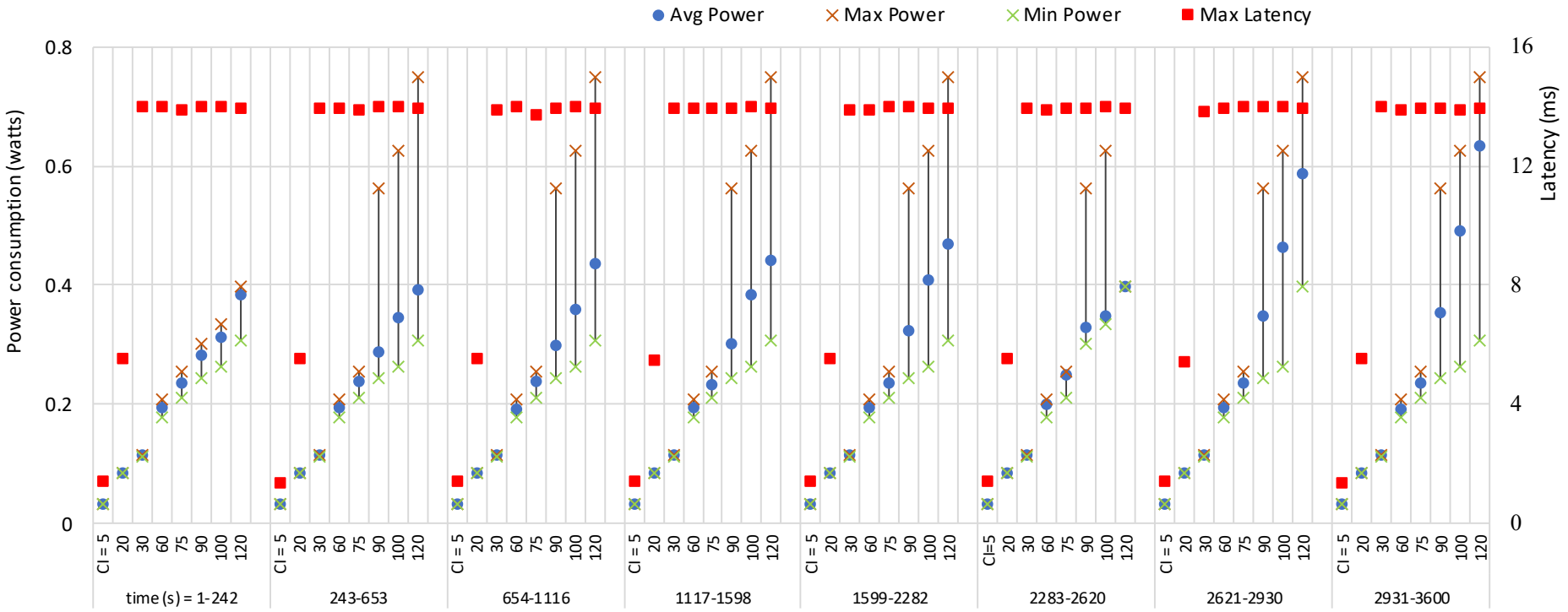

Figure 6.7 power consumption variations and the task accomplishment latency of a representative user (100th user out of 1,000)

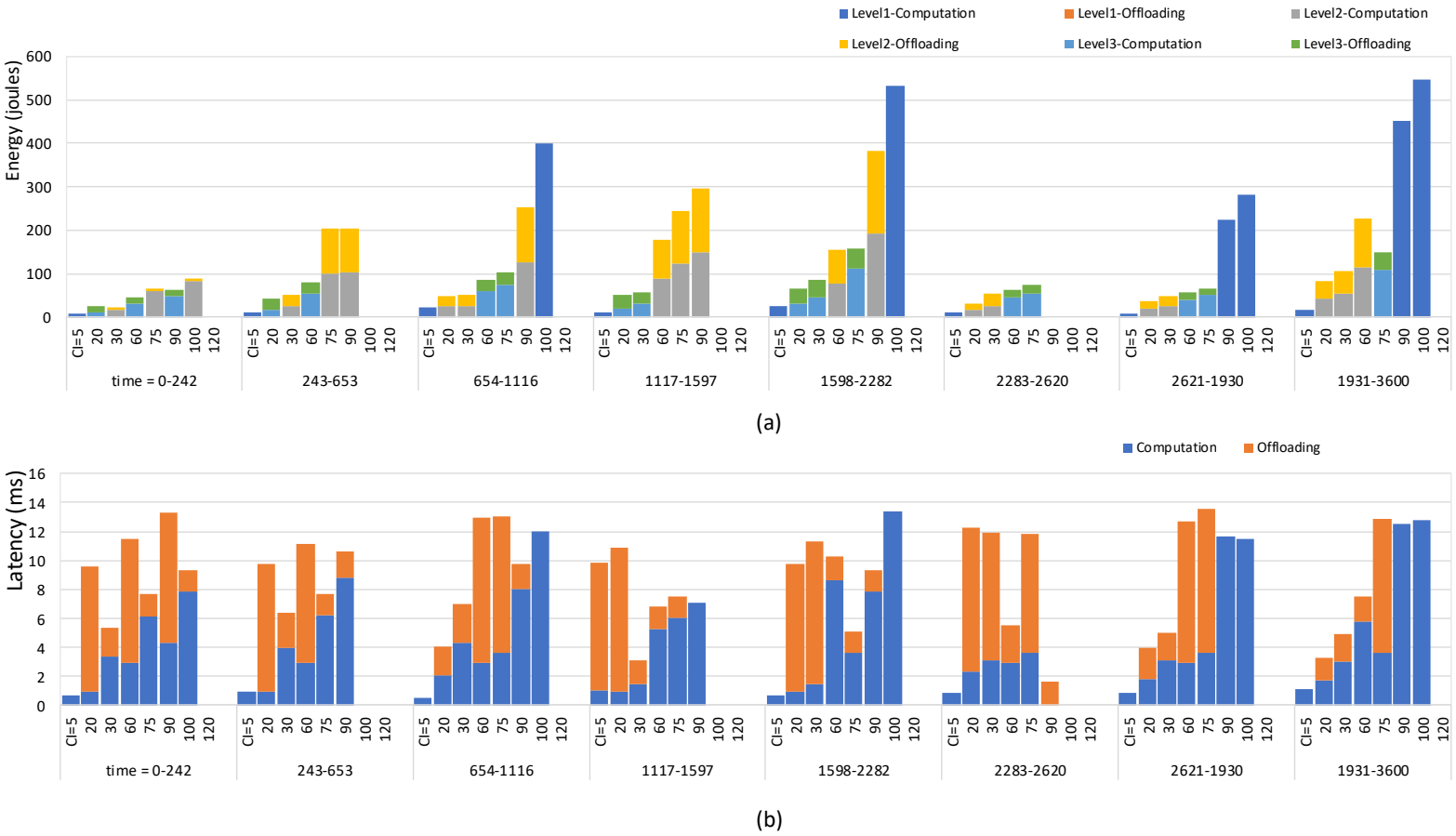

Figure 6.8 Breakdowns of (a) energy and (b) latency of the representative user (100th user) for a day

Similarly, Fig. 6.4(a) show the user trajectory of another representative user (100^{th} user) who is moving through eight different radio nodes and the time spent under the coverage of each radio node. High skewness can be seen for larger CI values during the morning peak hour similar to the above-selected user as in Figure 6.3. However, during the last two periods, i.e., 2621- 2930 and 1931 – 3600, higher power is consumed for larger CI values. For instance, 6% and 1 % are hosted for a CI value of 120 during the period 1931-3600 in level 1 and level 3, respectively.

Figure 6.8(a) shows the breakdown of energy consumption and Figure 6.8(b) shows the breakdown of latency for the representative user in a day. Only computation energy at level 1 is consumed for CI value of 5 since all the tasks are served at Level 1. When the CI value increases, energy consumption increases, while the level of the application being served is changing. For instance, CI value of 60 for the time period of 1117-1597, level 2 computation energy and level 2 offloading energy is consumed. Both of the energies are nearly equal. However, in the case of level 3, as in 654 - 1116 for the same CI value, computing energy is higher than the offloading energy. This is because more computing resources have to be allocated at level 3 to complete the task before the deadline requirement. This is proved by the total latency to execute the task is around 14 ms from Figure 6.8(b), which is the maximum tolerable latency.

In addition, some of the tasks such as CI values of 100 and 120 for the period 243 - 653 are rejected. This is because the MEC network is unable to satisfy the latency requirement of the application during that time period. Interestingly, when time increases, users move to CBD, higher value CI are being served at level 1 due to the maximum tolerable latency requirement. Most of the latency is due to offloading data to the MEC host Figure 6.8(b), which limits the task acceptance for accomplishment. To avoid this situation, adding more network bandwidth will improve the task acceptance ratio.

We found that both MEC hosts selection and user application migration can be predicted up to a high extent based on the CI values, as demonstrated by our results above. However, this highly depends on the mobility predictions and the resource availability prediction in the MEC network. Recent research such as in [187] uses a machine learning approach to predict the network demand. Since many research works have been focused on improving the prediction of human mobility from historical data of users within MEC network and MEC resource availability predictions, the shortest path problem can be pre-

calculated, which then leads to minimization of the response time to a user application offloading request and hence improving the energy efficiency of the MEC network.

6.5. Summary

Improving energy efficiency in MEC hosts selection and user application migration considering user mobility and latency requirement is a critical challenge for MEC service providers in a hierarchical network. In this chapter, we formulated the above-mentioned problem as a shortest path problem. We showed through simulations that by using Dijkstra's algorithm to solve the problem, computational intensity (δ) is an important metric in selecting the MEC host level to offload the user application. Our results showed that when CI increases, the benefits of selecting the top-level MEC host to host the user application in terms of energy efficiency increases but beyond a certain CI value, i.e., CI = 75, this trend changes due to maximum tolerable latency constraint.

Our research showed that the energy efficiency in MEC hosts selection and user application migration problem can be maximized with the help of machine learning and big data analytics to accurately predict user mobility patterns and MEC network resource availability. This, in turn, allows service providers to develop network resource orchestration algorithms to anticipate network loads and optimize their networks in real-time.

7 Conclusions and Future Directions

Multi-access edge computing is an emerging solution to support future mobile applications. Computation offloading in MEC network is the key enabler to enhance the capabilities of the mobile device beyond its limited capacities to support new and emerging applications that are computing-intensive and required ultra-low latency communications. Since MEC is still at its infant stage, it is very challenging to design and develop a MEC network especially to complement the existing 5G infrastructure.

7.1. Summary of Key Contributions

The main focus of this thesis is to propose potential solutions and techniques in designing a multi-access edge computing network that performs computation offloading of mobile applications, which delivers high quality-of-experience to the end users in terms of latency while considering the maximisation of network energy efficiency in a resource constraint environment. In a broader view, this thesis analyses the resource allocations and energy efficiency of host deployment in a MEC network. A summary of each contributing chapters is given below.

7.1.1 MEC Resource Allocation and Host Selection

In order to design and deploy MEC network within an existing mobile network, computation offloading demands of end users need to be derived. The resource requirement variations of different mobile applications are considered in deriving the demand of computation offloading in the MEC network. Since user mobility brings new challenges to the MEC network, a correlated user mobility model is presented in Chapter 3 to capture user mobility during peak hours as mirrored by real data. User trajectories of spatial and temporal dimensions derived from the correlated mobility model and the computational task offloading request pattern of users are used to estimate the

computation offloading demand of the MEC network. The utilitarian resource distribution algorithm proposed in Chapter 3 provides a benchmark for the maximum number of total tasks accomplished in the MEC network while providing insights into the selected host locations and the amount of CPU resources needed to be allocated in a resource-constrained environment.

Cost minimization of computation offloading without compromising the quality-of-service in a MEC network based on the service level agreement is another challenge for MEC service providers. Selecting suitable MEC host to serve offloading request is a key to minimize the serving cost of computation offloading in a resource constraint environment. In Chapter 5, an extended Balas-Geoffrion additive algorithm is developed to solve MEC host selection problem, which minimizes the total cost of computation offloading. Further, the algorithm minimizes the time complexity to find the optimum solution, since it uses only additions and subtractions to determine the solution. The findings in Chapter 5 show that the extended algorithm outperforms the original Balas-Geoffrion algorithm in the number of iterations required to reach the optimal solution for the special case of binary programming similar to MEC host selection problem. The cost of serving computation offloading increases beyond a threshold margin in terms of the number of rejected requests. This trade-off will be helpful for MEC service providers to finalize their service level agreements with their customers.

7.1.2 Energy Efficiency in MEC Network

Energy efficiency is an important consideration in designing MEC network to minimize the capital and operational expenditures of mobile operators. It starts with selecting suitable hardware devices to deploy in MEC hosts and implementing energy-efficient best practices in MEC network to minimize the energy consumption. In Chapter 4, a method for server class selection is proposed based on computation offloading demands in a MEC network. Then, energy saving algorithms for MEC host is also proposed based on the single threshold energy saving methodology. By carefully selecting the classes of servers based on the offloading demands, the proposed algorithms could achieve an energy saving of up to 34.32% in a MEC host during the morning peak hour. In addition, based on the proposed energy-efficient processes, an average energy saving of 16.15 % can be achieved in the MEC system during a morning peak hour using the proposed solution.

In addition to the energy-efficient MEC host deployments, improving energy efficiency in MEC hosts selection and user application migration is a critical challenge for MEC service providers in a hierarchical network. In Chapter 6, the above-mentioned problem is formulated as a shortest path problem. Computational intensity (CI) metric is proposed as an important metric in selecting the MEC host hierarchy level to offload the user application. Results suggested that the energy efficiency in MEC hosts selection and user application migration problem can be maximized with the help of machine learning and big data analytics to accurately predict user mobility patterns and MEC network resource availability. This, in turn, allows service providers to develop network resource orchestration algorithms based on proposed algorithms to anticipate network loads and optimize their networks in real-time.

7.2. Future Research Directions

The research work throughout this thesis is built up based on some valid assumptions to narrow down the scope of the thesis. However, by relaxing some of these assumptions, the thesis can be extended to cover more diverse use cases and research into the practicality of MEC network deployment issues. This opens different pathways for research activities in future. Yet, many research questions related to optimizing underlying MEC network to support a variety of applications are discussed in the following subsections.

7.2.1 Dynamic Resource Allocation in MEC Host Selection

The latency in uploading and downloading the offloading tasks through the wireless network is assumed to be constant in this thesis. In other words, wireless bandwidth resource allocation is assumed to be consistent throughout this thesis. However, this is not the case in a real wireless network. Hence, the work in this thesis can be extended to consider latency variations in the wireless network to jointly allocate resources in wireless and access network.

In detail, this thesis considers MEC host selection and user application migration problem by allocating bandwidth resource in the access network and computing resources in MEC hosts. However, this research work assumes the same amount of bandwidth

resource is allocated in the wireless network for all users. In reality, wireless bandwidth allocation could be varied based on the resource availability and the signal strength in the wireless network. Thus, the allocation of wireless network resource can be dynamically adjusted based on the resource availability in the radio access network or vice versa. For instance, if the end user is allocated a low bandwidth or the wireless connectivity of the end user is poor, the latency of data transmission through the wireless network is expected to increase. To compensate this increased latency in the wireless network, more resources will need to be allocated in the access network to satisfy the task deadline requirement. Thus, wireless network resource allocation and access network resource allocation can be jointly considered for the MEC host selection and user application migration problem.

Similar joint resource allocations for computation offloading were analyzed in previous literature for mobile cloud computing as in [70] [188] [189]. However, the main differences in MEC when compared to mobile cloud computing are impact of user mobility and limitations of available resources. Joint wireless bandwidth and computing resource allocations in MEC network gain attention of researchers recently [190] [191] [192]. User mobility in MEC network leads to another dimension of the resource allocation problems analysed recently.

7.2.2 Optimisation of MEC network for Internet of Things (IoT) Devices

The use case of computation offloading from user mobile devices is considered throughout this thesis, i.e., the accomplished results of the offloaded tasks should be returned to the same device. However, it might not be the case for some computation offloading applications. For instance, computation offloading from sensor network need not to be returned to the same sensor. The uploading data from the sensor network can be processed in an intermediate MEC host to extract the required information before uploading to the cloud data centre.

One of the use cases for future mobile applications is massive machine type communications (mMTC). This use case is characterized by a very large number of connected devices typically transmitting a relatively low volume of non-delay sensitive data, i.e. IoT devices. MEC can be used to process and aggregate the data generated by IoT services before they reach the cloud data centres. This will be important for network

scalability as the number of connections is expected to be huge and may be crucial for battery-powered IoT devices. A shorter transmission time between the device and the MEC host reduces drain on the battery and therefore can increase the life of the device and improve the business case.

Selecting a suitable MEC host to process and aggregate the IoT data is another challenge for MEC service providers. Since these data are non-delay sensitive and needs to be aggregated and uploaded to cloud data centres, it varies from the MEC host selection and user application migration problem analysed in this thesis.

Application of MEC network for IoT based computation offloading for Industry 4.0 is analysed in [193] and joint computation offloading and multi-user scheduling for NB-IoT is analysed in [194]. A cognitive data offloading approach for IoT is considered in [195] However, resource limitations and hierarchical deployment in MEC network is not considered in the above research. Thus, MEC host selection for computation offloading of MTC use case could be a future research direction that can be extended from this thesis.

7.2.3 Vehicle to Vehicle Communication

User mobility in this thesis is considered as correlated user mobility during the commute in the morning peak hours. The proposed correlated mobility model considers the CBD in the centre of the simulation area and morning peak hours. The same model can be extended to multiple CBDs or gathering areas and the evening peak hour mobility by changing the parameters in the model. In addition, mobility model can be extended to vehicular communication to understand the mobility of the vehicle in a bird's eye view.

Hence, the next-generation transportation systems are proposed to improve transportation safety and efficiency by incorporating wireless communication and informatics technologies in the transportation system. Vehicular communication networks enable vehicles to exchange vital information with other vehicles and external environments. Vehicle-to-vehicle communications can be used in supporting a variety of services such as road safety, traffic management, and entertainment. Vehicular communication networks face many technical challenges such as vehicle mobility, burst traffic, highly dynamic topology, vulnerable wireless links, very low latency communication [196]. Facilitating computation offloading in vehicular technologies

highly depend on very low latency and hard deadline requirements of the user applications, since vehicles could move at a high speed.

In this thesis, the correlated mobility model is proposed to predict user mobility during the morning peak hour, where most of the users are moving towards the central business district from outer areas. Proposed correlated mobility model can be extended to predict the mobility of the vehicles. In addition, computation offloading from mobile devices were considered in this thesis in which tasks are offloaded from a mobile device and the results are returned to the same device. However, in the vehicular communication scenario, tasks are offloaded from a vehicle and the results might be delivered to multiple vehicles. This additional constraint needs to be considered in selecting suitable MEC host to serve offloading request in vehicular communication, which can be extended from the MEC host selection and user application migration problem as analysed in this thesis.

Collaborative vehicular edge commuting network architectures and challenges are discussed in [197] to implement autonomous driving and augmented reality applications scenarios. In addition, different methodologies are proposed for computation offloading from vehicles to edge computing network in recent literature [198] [199].

7.2.4 Edge Intelligence

MEC network conditions such as network traffic are highly dynamic due to user mobility and edge server deployment. In addition, the user application usage pattern is also highly dynamic due to the availability of a vast variety of mobile applications. This thesis assumed that the network conditions and application usage pattern are known to the MEC system. However, in real networks, it is a challenge to predict these values.

Since MEC hosts are going to be deployed proximity to the end users, context awareness is a key feature of edge computing, which can be utilised to predict the behaviour of the users and network. These predictions can be used in vast variety of applications from healthcare to network optimization [200] [201]. The vast amount of user context data can be collected for analysing user behaviour history. Similarly, network context data can be saved to analyse the dynamics of the network under different times of day as well as different days and months. Big data analytical tools shall be used to analyse the context-awareness data to propose precise predictions. Further, machine learning methods such as deep reinforcement learning, federated learning and transfer

learning [201] can be used to predict the network conditions, user behaviours such as mobility and usage pattern on an ongoing basis [202]. In keeping the records of context awareness, blockchain approach can be used to save the history of user context securely and in a decentralised manner, which provides convenience and easy to access. For example, recent research such as in [187] uses a machine learning approach to predict the network demand which can be used to extend this thesis.

7.3. Conclusions

In summary, challenges in designing MEC network are analysed and solutions are proposed in this thesis to solve such challenges. Deployment and resource allocation related challenges such as selecting suitable locations to deploy hosts and choosing right-size resources for each host are analysed critically. Correlated mobility model for user mobility predictions and utilitarian resource distribution algorithms for resource allocations are proposed to solve such challenges. To maximize the energy efficiency of the MEC system, methodologies related to physical server selection and energy-efficient virtual machine placement processes are proposed. MEC service providers can utilize the deployment and resource distribution solutions and energy efficiency methodologies to design and develop their network in a sustainable way to empower a better-connected future.

In addition, selecting a suitable host to serve offloading request during computation offloading is a critical challenge to optimize the performance of the MEC system. Extended Balas-Geoffrion additive algorithm is proposed to minimize the total cost incurred by service providers in host selection. In addition, the shortest path problem based solution is also proposed to maximize energy efficiency in the system. MEC hardware system vendors can utilize the proposed host selection algorithms to implement the host selection process efficiently in their system. Therefore, this thesis proposes methodologies and solutions for critical challenges for service providers and hardware system vendors to design and develop MEC network.

References

[1] A. J. Reid, *The Smartphone Paradox: Our Ruinous Dependency in the Device Age*. Cham: Springer International Publishing, 2018.

[2] P. Zheng and L. Ni, *Smart Phone and Next Generation Mobile Computing.* Burlington: Elsevier Science, 2010.

[3] "How the smartphone has changed our world (for better or worse)," *Australian Financial Review*, Dec. 19, 2019. https://www.afr.com/technology/how-the-smartphone-has-changed-our-world-for-better-or-worse-20191205-p53h51 (accessed Jul. 17, 2020).

[4] "50% of teens feel addicted to their phones, poll says - CNN." https://edition.cnn.com/2016/05/03/health/teens-cell-phone-addiction-parents/ (accessed Jul. 17, 2020).

[5] "Mobile app revenues 2014-2023," *Statista*. https://www.statista.com/statistics/269025/worldwide-mobile-app-revenue-forecast/ (accessed Jun. 23, 2020).

[6] "Extended Reality XR | Immersive VR," *Qualcomm*, May 19, 2017. https://www.qualcomm.com/invention/extended-reality (accessed Jul. 16, 2020).

[7] "XR Accessibility User Requirements." https://www.w3.org/TR/xaur/#what-is-xr-used-for (accessed Jun. 23, 2020).

[8] R. and M. ltd, "Global Healthcare Augmented Reality and Virtual Reality Market by Technology, Offering, Device Type, Application, End-user, and Region 2019-2026: Trend Forecast and Growth Opportunity." https://www.researchandmarkets.com/reports/4876656/global-healthcare-augmented-reality-and-virtual (accessed Jul. 17, 2020).

[9] S. Guo, B. Xiao, Y. Yang, and Y. Yang, "Energy-efficient dynamic offloading and resource scheduling in mobile cloud computing," in *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, Apr. 2016, pp. 1–9, doi: 10.1109/INFOCOM.2016.7524497.

[10] M. Wollschlaeger, T. Sauter, and J. Jasperneite, "The Future of Industrial Communication: Automation Networks in the Era of the Internet of Things and Industry 4.0," *IEEE Industrial Electronics Magazine*, vol. 11, no. 1, pp. 17–27, Mar. 2017, doi: 10.1109/MIE.2017.2649104.

[11] P. Schulz *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, Feb. 2017, doi: 10.1109/MCOM.2017.1600435CM.

[12] N. Fernando, S. W. Loke, and W. Rahayu, "Mobile cloud computing: A survey," *Future Generation Computer Systems*, vol. 29, no. 1, pp. 84–106, Jan. 2013, doi: 10.1016/j.future.2012.05.023.

[13] E. Cuervo *et al.*, "MAUI: making smartphones last longer with code offload," in *Proceedings of the 8th international conference on Mobile systems, applications, and services - MobiSys '10*, San Francisco, California, USA, 2010, p. 49, doi: 10.1145/1814433.1814441.

[14] S. Barbarossa, S. Sardellitti, and P. Di Lorenzo, "Communicating While Computing: Distributed mobile cloud computing over 5G heterogeneous networks," *IEEE Signal Processing Magazine*, vol. 31, no. 6, pp. 45–55, Nov. 2014, doi: 10.1109/MSP.2014.2334709.

[15] H. T. Dinh, C. Lee, D. Niyato, and P. Wang, "A survey of mobile cloud computing: architecture, applications, and approaches," *Wirel. Commun. Mob. Comput.*, vol. 13, no. 18, pp. 1587–1611, Dec. 2013, doi: 10.1002/wcm.1203.

[16] R. Ford, M. Zhang, M. Mezzavilla, S. Dutta, S. Rangan, and M. Zorzi, "Achieving Ultra-Low Latency in 5G Millimeter Wave Cellular Networks," *IEEE Communications Magazine*, vol. 55, no. 3, pp. 196–203, Mar. 2017, doi: 10.1109/MCOM.2017.1600407CM.

[17] M. S. Elbamby, C. Perfecto, M. Bennis, and K. Doppler, "Toward Low-Latency and Ultra-Reliable Virtual Reality," *IEEE Network*, vol. 32, no. 2, pp. 78–84, Mar. 2018, doi: 10.1109/MNET.2018.1700268.

[18] 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, thirdquarter 2017, doi: 10.1109/COMST.2017.2705720.

[19] 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.

[20] ITU Radiocommunication Sector, "IMT Vision – Framework and overall objectives of the future development of IMT for 2020 and beyond." ITU, Sep. 2015, Accessed: Mar. 19, 2020. [Online]. Available: https://www.itu.int/dms_pubrec/itu-r/rec/m/R-REC-M.2083-0-201509-I!!PDF-E.pdf.

[21] ETSI, "Mobile edge computing (MEC): Technical requirements." ETSI Industry Specification Group (ISG), Mar. 2016, Accessed: Nov. 10, 2017. [Online]. Available: http://www.etsi.org/deliver/etsi_gs/MEC/001_099/002/01.01.01_60/gs_MEC002v010101p.pdf.

[22] 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, thirdquarter 2017, doi: 10.1109/COMST.2017.2682318.

[23] 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. PP, no. 99, pp. 1–1, 2017, doi: 10.1109/COMST.2017.2745201.

[24] H. Liu, F. Eldarrat, H. Alqahtani, A. Reznik, X. de Foy, and Y. Zhang, "Mobile Edge Cloud System: Architectures, Challenges, and Approaches," *IEEE Systems Journal*, vol. PP, no. 99, pp. 1–14, 2017, doi: 10.1109/JSYST.2017.2654119.

[25] C. Song, Z. Qu, N. Blumm, and A.-L. Barabasi, "Limits of Predictability in Human Mobility," *Science (New York, N.Y.)*, vol. 327, pp. 1018–21, Feb. 2010, doi: 10.1126/science.1177170.

[26] G. P. Fettweis, "The Tactile Internet: Applications and Challenges," *IEEE Vehicular Technology Magazine*, vol. 9, no. 1, pp. 64–70, Mar. 2014, doi: 10.1109/MVT.2013.2295069.

[27] M. Maier, M. Chowdhury, B. P. Rimal, and D. P. Van, "The tactile internet: vision, recent progress, and open challenges," *IEEE Communications Magazine*, vol. 54, no. 5, pp. 138–145, May 2016, doi: 10.1109/MCOM.2016.7470948.

[28] A. Nasrallah *et al.*, "Ultra-Low Latency (ULL) Networks: The IEEE TSN and IETF DetNet Standards and Related 5G ULL Research," *IEEE Communications Surveys Tutorials*, vol. 21, no. 1, pp. 88–145, Firstquarter 2019, doi: 10.1109/COMST.2018.2869350.

[29] S. Samii and H. Zinner, "Level 5 by Layer 2: Time-Sensitive Networking for Autonomous Vehicles," *IEEE Communications Standards Magazine*, vol. 2, no. 2, pp. 62–68, Jun. 2018, doi: 10.1109/MCOMSTD.2018.1700079.

[30] D. Delaney, T. Ward, and S. McLoone, "On Consistency and Network Latency in Distributed Interactive Applications: A Survey—Part I," *Presence*, vol. 15, no. 2, pp. 218–234, Apr. 2006, doi: 10.1162/pres.2006.15.2.218.

[31] F. Prinz, M. Schoeffler, A. Lechler, and A. Verl, "Dynamic Real-time Orchestration of I4.0 Components based on Time-Sensitive Networking," *Procedia CIRP*, vol. 72, pp. 910–915, Jan. 2018, doi: 10.1016/j.procir.2018.03.174.

[32] C. Bachhuber, E. Steinbach, M. Freundl, and M. Reisslein, "On the Minimization of Glass-to-Glass and Glass-to-Algorithm Delay in Video Communication," *IEEE Transactions on Multimedia*, vol. 20, no. 1, pp. 238–252, Jan. 2018, doi: 10.1109/TMM.2017.2726189.

[33] E. Gardiner, "The Avnu Alliance Theory of Operation for TSN-enabled Industrial Systems," *IEEE Communications Standards Magazine*, vol. 2, no. 1, pp. 5–5, Mar. 2018, doi: 10.1109/MCOMSTD.2018.8334911.

[34] R. Balan, J. Flinn, M. Satyanarayanan, S. Sinnamohideen, and H.-I. Yang, "The case for cyber foraging," in *Proceedings of the 10th workshop on ACM SIGOPS European workshop*, Saint-Emilion, France, Jul. 2002, pp. 87–92, doi: 10.1145/1133373.1133390.

[35] J. Flinn, SoYoung Park, and M. Satyanarayanan, "Balancing performance, energy, and quality in pervasive computing," in *Proceedings 22nd International Conference on Distributed Computing Systems*, Jul. 2002, pp. 217–226, doi: 10.1109/ICDCS.2002.1022259.

[36] B.-G. Chun, S. Ihm, P. Maniatis, M. Naik, and A. Patti, "CloneCloud: elastic execution between mobile device and cloud," in *Proceedings of the sixth conference on Computer systems*, Salzburg, Austria, Apr. 2011, pp. 301–314, doi: 10.1145/1966445.1966473.

[37] M. Darø Kristensen, "Scavenger: Transparent development of efficient cyber foraging applications," in *2010 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, Mar. 2010, pp. 217–226, doi: 10.1109/PERCOM.2010.5466972.

[38] R. N. Mayo and P. Ranganathan, "Energy Consumption in Mobile Devices: Why Future Systems Need Requirements–Aware Energy Scale-Down," in *Power-Aware Computer Systems*, Berlin, Heidelberg, 2005, pp. 26–40, doi: 10.1007/978-3-540-28641-7_3.

[39] A. Rudenko, P. Reiher, G. J. Popek, and G. H. Kuenning, "Saving portable computer battery power through remote process execution," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 2, no. 1, pp. 19–26, Jan. 1998, doi: 10.1145/584007.584008.

[40] U. Kremer, J. Hicks, and J. Rehg, "A Compilation Framework for Power and Energy Management on Mobile Computers," in *Languages and Compilers for Parallel Computing*, Berlin, Heidelberg, 2003, pp. 115–131, doi: 10.1007/3-540-35767-X_8.

[41] A. Garcia and H. Kalva, "Cloud transcoding for mobile video content delivery," in *2011 IEEE International Conference on Consumer Electronics (ICCE)*, Jan. 2011, pp. 379–380, doi: 10.1109/ICCE.2011.5722637.

[42] K. Kumar and Y. H. Lu, "Cloud Computing for Mobile Users: Can Offloading Computation Save Energy?," *Computer*, vol. 43, no. 4, pp. 51–56, Apr. 2010, doi: 10.1109/MC.2010.98.

[43] L. Li, X. Li, S. Youxia, and L. Wen, "Research on Mobile Multimedia Broadcasting Service Integration Based on Cloud Computing," in *2010 International Conference on Multimedia Technology*, Oct. 2010, pp. 1–4, doi: 10.1109/ICMULT.2010.5630979.

[44] P. K. Agyapong, M. Iwamura, D. Staehle, W. Kiess, and A. Benjebbour, "Design considerations for a 5G network architecture," *IEEE Communications Magazine*, vol. 52, no. 11, pp. 65–75, Nov. 2014, doi: 10.1109/MCOM.2014.6957145.

[45] V.-G. Nguyen, T.-X. Do, and Y. Kim, "SDN and Virtualization-Based LTE Mobile Network Architectures: A Comprehensive Survey," *Wireless Pers Commun*, vol. 86, no. 3, pp. 1401–1438, Feb. 2016, doi: 10.1007/s11277-015-2997-7.

[46] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog Computing and Its Role in the Internet of Things," in *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*, New York, NY, USA, 2012, pp. 13–16, doi: 10.1145/2342509.2342513.

[47] 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*, Hangzhou, China, Jun. 2015, pp. 37–42, doi: 10.1145/2757384.2757397.

[48] M. R. Anawar, S. Wang, M. Azam Zia, A. K. Jadoon, U. Akram, and S. Raza, "Fog Computing: An Overview of Big IoT Data Analytics," *Wireless Communications and Mobile Computing*, May 07, 2018. https://www.hindawi.com/journals/wcmc/2018/7157192/ (accessed Jul. 18, 2020).

[49] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The Case for VM-Based Cloudlets in Mobile Computing," *IEEE Pervasive Computing*, vol. 8, no. 4, pp. 14–23, Oct. 2009, doi: 10.1109/MPRV.2009.82.

[50] 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, doi: 10.1109/ACCESS.2017.2685434.

[51] "Multi-access Edge Computing (MEC); Framework and Reference Architecture." ETSI Industry Specification Group (ISG), Jan. 2019, Accessed: Feb. 06, 2019. [Online]. Available: https://www.etsi.org/deliver/etsi_gs/MEC/001_099/003/02.01.01_60/gs_MEC003v020101p.pdf.

[52] ETSI, "Mobile Edge Computing; Market Acceleration; MEC Metrics Best Practice and Guidelines." European Telecommunications Standards Institute, Jan. 2017, Accessed: Aug. 10, 2018. [Online]. Available: https://www.etsi.org/deliver/etsi_gs/MEC-IEG/001_099/006/01.01.01_60/gs_MEC-IEG006v010101p.pdf.

[53] T. Soyata, R. Muraleedharan, C. Funai, M. Kwon, and W. Heinzelman, "Cloud-Vision: Real-time face recognition using a mobile-cloudlet-cloud acceleration architecture," in *2012 IEEE Symposium on Computers and Communications (ISCC)*, Jul. 2012, pp. 000059–000066, doi: 10.1109/ISCC.2012.6249269.

[54] "Multi-access Edge Computing (MEC); Phase 2: Use Cases and Requirements." ETSI Industry Specification Group (ISG), Oct. 2018, Accessed: Feb. 06, 2019. [Online]. Available: https://www.etsi.org/deliver/etsi_gs/MEC/001_099/002/02.01.01_60/gs_MEC002v020101p.pdf.

[55] Y. Harchol, A. Mushtaq, V. Fang, J. McCauley, A. Panda, and S. Shenker, "Making edge-computing resilient," in *Proceedings of the 11th ACM Symposium on Cloud Computing*, New York, NY, USA, Oct. 2020, pp. 253–266, doi: 10.1145/3419111.3421278.

[56] 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, Oct. 2016, doi: 10.1109/MCE.2016.2590118.

[57] A. Ahmed and E. Ahmed, "A survey on mobile edge computing," in *2016 10th International Conference on Intelligent Systems and Control (ISCO)*, Jan. 2016, pp. 1–8, doi: 10.1109/ISCO.2016.7727082.

[58] A. V. Dastjerdi and R. Buyya, "Fog Computing: Helping the Internet of Things Realize Its Potential," *Computer*, vol. 49, no. 8, pp. 112–116, Aug. 2016, doi: 10.1109/MC.2016.245.

[59] C. Vallati, A. Virdis, E. Mingozzi, and G. Stea, "Mobile-Edge Computing Come Home Connecting things in future smart homes using LTE device-to-device communications," *IEEE Consumer Electronics Magazine*, vol. 5, no. 4, pp. 77–83, Oct. 2016, doi: 10.1109/MCE.2016.2590100.

[60] M. Sapienza, E. Guardo, M. Cavallo, G. La Torre, G. Leombruno, and O. Tomarchio, "Solving Critical Events through Mobile Edge Computing: An Approach for Smart Cities," in *2016 IEEE International Conference on Smart Computing (SMARTCOMP)*, St Louis, MO, USA, May 2016, pp. 1–5, doi: 10.1109/SMARTCOMP.2016.7501719.

[61] 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, Sep. 2017, doi: 10.1109/TNSM.2017.2706085.

[62] W. Hu *et al.*, "Quantifying the Impact of Edge Computing on Mobile Applications," in *Proceedings of the 7th ACM SIGOPS Asia-Pacific Workshop on Systems*, New York, NY, USA, 2016, p. 5:1–5:8, doi: 10.1145/2967360.2967369.

[63] Y. Gao, W. Hu, K. Ha, B. Amos, P. Pillai, and M. Satyanarayanan, "Are Cloudlets Necessary," 2015.

[64] J. Ren, G. Yu, Y. Cai, and Y. He, "Latency Optimization for Resource Allocation in Mobile-Edge Computation Offloading," *IEEE Transactions on Wireless Communications*, vol. 17, no. 8, pp. 5506–5519, Aug. 2018, doi: 10.1109/TWC.2018.2845360.

[65] M. Molina, O. Muñoz, A. Pascual-Iserte, and J. Vidal, "Joint scheduling of communication and computation resources in multiuser wireless application offloading," in *2014 IEEE 25th Annual International Symposium on Personal, Indoor, and Mobile Radio Communication (PIMRC)*, Sep. 2014, pp. 1093–1098, doi: 10.1109/PIMRC.2014.7136330.

[66] L. Yang, J. Cao, H. Cheng, and Y. Ji, "Multi-User Computation Partitioning for Latency Sensitive Mobile Cloud Applications," *IEEE Transactions on Computers*, vol. 64, no. 8, pp. 2253–2266, Aug. 2015, doi: 10.1109/TC.2014.2366735.

[67] J. Liu, Y. Mao, J. Zhang, and K. B. Letaief, "Delay-optimal computation task scheduling for mobile-edge computing systems," in *2016 IEEE International Symposium on Information Theory (ISIT)*, Jul. 2016, pp. 1451–1455, doi: 10.1109/ISIT.2016.7541539.

[68] Y. Mao, J. Zhang, and K. B. Letaief, "Dynamic Computation Offloading for Mobile-Edge Computing With Energy Harvesting Devices," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 12, pp. 3590–3605, Dec. 2016, doi: 10.1109/JSAC.2016.2611964.

[69] L. Chen, S. Zhou, and J. Xu, "Computation Peer Offloading for Energy-Constrained Mobile Edge Computing in Small-Cell Networks," *arXiv:1703.06058 [cs]*, Mar. 2017, Accessed: Oct. 11, 2017. [Online]. Available: http://arxiv.org/abs/1703.06058.

[70] M. Kamoun, W. Labidi, and M. Sarkiss, "Joint resource allocation and offloading strategies in cloud enabled cellular networks," in *2015 IEEE International Conference on Communications (ICC)*, Jun. 2015, pp. 5529–5534, doi: 10.1109/ICC.2015.7249203.

[71] W. Labidi, M. Sarkiss, and M. Kamoun, "Energy-optimal resource scheduling and computation offloading in small cell networks," in *2015 22nd International Conference on Telecommunications (ICT)*, Sydney, Australia, Apr. 2015, pp. 313–318, doi: 10.1109/ICT.2015.7124703.

[72] W. Zhang, Y. Wen, K. Guan, D. Kilper, H. Luo, and D. O. Wu, "Energy-Optimal Mobile Cloud Computing under Stochastic Wireless Channel," *IEEE Transactions on Wireless Communications*, vol. 12, no. 9, pp. 4569–4581, Sep. 2013, doi: 10.1109/TWC.2013.072513.121842.

[73] S. Ulukus *et al.*, "Energy Harvesting Wireless Communications: A Review of Recent Advances," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 3, pp. 360–381, Mar. 2015, doi: 10.1109/JSAC.2015.2391531.

[74] 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, Oct. 2016, doi: 10.1109/TNET.2015.2487344.

[75] M.-H. Chen, B. Liang, and M. Dong, "A semidefinite relaxation approach to mobile cloud offloading with computing access point," in *2015 IEEE 16th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, Jun. 2015, pp. 186–190, doi: 10.1109/SPAWC.2015.7227025.

[76] T. Zhao, S. Zhou, X. Guo, Y. Zhao, and Z. Niu, "A Cooperative Scheduling Scheme of Local Cloud and Internet Cloud for Delay-Aware Mobile Cloud Computing," *arXiv:1511.08540 [cs]*, Nov. 2015, Accessed: Nov. 30, 2017. [Online]. Available: http://arxiv.org/abs/1511.08540.

[77] Y. Ge, Y. Zhang, Q. Qiu, and Y.-H. Lu, "A Game Theoretic Resource Allocation for Overall Energy Minimization in Mobile Cloud Computing System," in *Proceedings of the 2012 ACM/IEEE International Symposium on Low Power Electronics and Design*, New York, NY, USA, 2012, pp. 279–284, doi: 10.1145/2333660.2333724.

[78] T. Q. Dinh, J. Tang, Q. D. La, and T. Q. S. Quek, "Offloading in Mobile Edge Computing: Task Allocation and Computational Frequency Scaling," *IEEE Transactions on Communications*, vol. 65, no. 8, pp. 3571–3584, Aug. 2017, doi: 10.1109/TCOMM.2017.2699660.

[79] "Mobile Edge Computing (MEC); End to End Mobility Aspects." ETSI Industry Specification Group (ISG), Oct. 2017, [Online]. Available: http://www.etsi.org/deliver/etsi_gr/MEC/001_099/018/01.01.01_60/gr_MEC018v010101p.pdf.

[80] D. Lopez-Perez, I. Guvenc, and X. Chu, "Mobility management challenges in 3GPP heterogeneous networks," *IEEE Communications Magazine*, vol. 50, no. 12, pp. 70–78, Dec. 2012, doi: 10.1109/MCOM.2012.6384454.

[81] A. Damnjanovic *et al.*, "A survey on 3GPP heterogeneous networks," *IEEE Wireless Communications*, vol. 18, no. 3, pp. 10–21, Jun. 2011, doi: 10.1109/MWC.2011.5876496.

[82] M. Kassar, B. Kervella, and G. Pujolle, "An overview of vertical handover decision strategies in heterogeneous wireless networks," *Computer Communications*, vol. 31, no. 10, pp. 2607–2620, Jun. 2008, doi: 10.1016/j.comcom.2008.01.044.

[83] C. Wang, Y. Li, and D. Jin, "Mobility-Assisted Opportunistic Computation Offloading," *IEEE Communications Letters*, vol. 18, no. 10, pp. 1779–1782, Oct. 2014, doi: 10.1109/LCOMM.2014.2347272.

[84] Y. Zhang, D. Niyato, and P. Wang, "Offloading in Mobile Cloudlet Systems with Intermittent Connectivity," *IEEE Transactions on Mobile Computing*, vol. 14, no. 12, pp. 2516–2529, Dec. 2015, doi: 10.1109/TMC.2015.2405539.

[85] K. Lee and I. Shin, "User Mobility Model Based Computation Offloading Decision for Mobile Cloud," *Journal of Computing Science and Engineering*, Sep. 2015. http://www.dbpia.co.kr (accessed Nov. 30, 2017).

[86] M. R. Rahimi, N. Venkatasubramanian, and A. V. Vasilakos, "MuSIC: Mobility-Aware Optimal Service Allocation in Mobile Cloud Computing," in *2013 IEEE Sixth International Conference on Cloud Computing*, Jun. 2013, pp. 75–82, doi: 10.1109/CLOUD.2013.100.

[87] C. Bettstetter, G. Resta, and P. Santi, "The node distribution of the random waypoint mobility model for wireless ad hoc networks," *IEEE Transactions on Mobile Computing*, vol. 2, no. 3, pp. 257–269, Jul. 2003, doi: 10.1109/TMC.2003.1233531.

[88] R. Groenevelt, E. Altman, and P. Nain, "Relaying in mobile ad hoc networks: The Brownian motion mobility model," *Wireless Netw*, vol. 12, no. 5, pp. 561–571, Oct. 2006, doi: 10.1007/s11276-006-6535-0.

[89] J. Yoon, M. Liu, and B. Noble, "Random waypoint considered harmful," in *IEEE INFOCOM 2003. Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Cat. No.03CH37428)*, Mar. 2003, vol. 2, pp. 1312–1321 vol.2, doi: 10.1109/INFCOM.2003.1208967.

[90] P. Pirozmand, G. Wu, B. Jedari, and F. Xia, "Human mobility in opportunistic networks: Characteristics, models and prediction methods," *Journal of Network and Computer Applications*, vol. 42, pp. 45–58, Jun. 2014, doi: 10.1016/j.jnca.2014.03.007.

[91] N. Aschenbruck, A. Munjal, and T. Camp, "Trace-based mobility modeling for multi-hop wireless networks," *Computer Communications*, vol. 34, no. 6, pp. 704–714, May 2011, doi: 10.1016/j.comcom.2010.11.002.

[92] T. Nguyen and B. K. Szymanski, "Using Location-Based Social Networks to Validate Human Mobility and Relationships Models," in *2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, Aug. 2012, pp. 1215–1221, doi: 10.1109/ASONAM.2012.210.

[93] F. Ekman, A. Keränen, J. Karvo, and J. Ott, "Working Day Movement Model," in *Proceedings of the 1st ACM SIGMOBILE Workshop on Mobility Models*, New York, NY, USA, 2008, pp. 33–40, doi: 10.1145/1374688.1374695.

[94] D. Brockmann, L. Hufnagel, and T. Geisel, "The scaling laws of human travel," *Nature*, vol. 439, no. 7075, pp. 462–465, Jan. 2006, doi: 10.1038/nature04292.

[95] M. C. González, C. A. Hidalgo, and A.-L. Barabási, "Understanding individual human mobility patterns," *Nature*, vol. 453, no. 7196, pp. 779–782, Jun. 2008, doi: 10.1038/nature06958.

[96] L. Cao and M. Grabchak, "Smoothly truncated levy walks: Toward a realistic mobility model," in *2014 IEEE 33rd International Performance Computing and Communications Conference (IPCCC)*, Dec. 2014, pp. 1–8, doi: 10.1109/PCCC.2014.7017071.

[97] I. Rhee, M. Shin, S. Hong, K. Lee, and S. Chong, "On the Levy-Walk Nature of Human Mobility," presented at the IEEE INFOCOM 2008 - The 27th Conference on Computer Communications, Apr. 2008, doi: 10.1109/INFOCOM.2008.145.

[98] S. Phithakkitnukoon, Z. Smoreda, and P. Olivier, "Socio-Geography of Human Mobility: A Study Using Longitudinal Mobile Phone Data," *PLOS ONE*, vol. 7, no. 6, p. e39253, Jun. 2012, doi: 10.1371/journal.pone.0039253.

[99] H. S. Dhillon, R. K. Ganti, F. Baccelli, and J. G. Andrews, "Modeling and Analysis of K-Tier Downlink Heterogeneous Cellular Networks," *IEEE Journal on Selected Areas in Communications*, vol. 30, no. 3, pp. 550–560, Apr. 2012, doi: 10.1109/JSAC.2012.120405.

[100] M. Poess and R. O. Nambiar, "Energy cost, the key challenge of today's data centers: a power consumption analysis of TPC-C results," *Proc. VLDB Endow.*, vol. 1, no. 2, pp. 1229–1240, Aug. 2008, doi: 10.14778/1454159.1454162.

[101] R. Buyya, C. Vecchiola, and S. T. Selvi, *Mastering Cloud Computing: Foundations and Applications Programming*, 1st ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2013.

[102] Y. Gao, H. Guan, Z. Qi, B. Wang, and L. Liu, "Quality of service aware power management for virtualized data centers," *Journal of Systems Architecture*, vol. 59, no. 4, pp. 245–259, Apr. 2013, doi: 10.1016/j.sysarc.2013.03.007.

[103] S. Rivoire, M. A. Shah, P. Ranganathan, C. Kozyrakis, and J. Meza, "Models and Metrics to Enable Energy-Efficiency Optimizations," *Computer*, vol. 40, no. 12, pp. 39–48, Dec. 2007, doi: 10.1109/MC.2007.436.

[104] K. Bilal, S. U. R. Malik, S. U. Khan, and A. Y. Zomaya, "Trends and challenges in cloud datacenters," *IEEE Cloud Computing*, vol. 1, no. 1, pp. 10–20, May 2014, doi: 10.1109/MCC.2014.26.

[105] B. Whitehead, D. Andrews, A. Shah, and G. Maidment, "Assessing the environmental impact of data centres part 1: Background, energy use and metrics," *Building and Environment*, vol. 82, pp. 151–159, Dec. 2014, doi: 10.1016/j.buildenv.2014.08.021.

[106] A. Andrae and P. M. Corcoran, "Emerging trends in electricity consumption for consumer ICT," Report, 2013. Accessed: Jul. 22, 2020. [Online]. Available: https://aran.library.nuigalway.ie/handle/10379/3563.

[107] M. Dayarathna, Y. Wen, and R. Fan, "Data Center Energy Consumption Modeling: A Survey," *IEEE Communications Surveys Tutorials*, vol. 18, no. 1, pp. 732–794, Firstquarter 2016, doi: 10.1109/COMST.2015.2481183.

[108] S. Roy, A. Rudra, and A. Verma, "An energy complexity model for algorithms," in *Proceedings of the 4th conference on Innovations in Theoretical Computer Science*, Berkeley, California, USA, Jan. 2013, pp. 283–304, doi: 10.1145/2422436.2422470.

[109] A. W. Lewis, N.-F. Tzeng, and S. Ghosh, "Runtime energy consumption estimation for server workloads based on chaotic time-series approximation," *ACM Trans. Archit. Code Optim.*, vol. 9, no. 3, p. 15:1–15:26, Oct. 2012, doi: 10.1145/2355585.2355588.

[110] R. Lent, "A model for network server performance and power consumption," *Sustainable Computing: Informatics and Systems*, vol. 3, no. 2, pp. 80–93, Jun. 2013, doi: 10.1016/j.suscom.2012.03.004.

[111] B. M. Tudor and Y. M. Teo, "On understanding the energy consumption of ARM-based multicore servers," in *Proceedings of the ACM SIGMETRICS/international conference on Measurement and modeling of computer systems*, Pittsburgh, PA, USA, Jun. 2013, pp. 267–278, doi: 10.1145/2465529.2465553.

[112] Y. Jin, Y. Wen, Q. Chen, and Z. Zhu, "An Empirical Investigation of the Impact of Server Virtualization on Energy Efficiency for Green Data Center," *The Computer Journal*, vol. 56, no. 8, pp. 977–990, Aug. 2013, doi: 10.1093/comjnl/bxt017.

[113] A. Beloglazov, J. Abawajy, and R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for Cloud computing," *Future Generation Computer Systems*, vol. 28, no. 5, pp. 755–768, May 2012, doi: 10.1016/j.future.2011.04.017.

[114] R. Basmadjian, N. Ali, F. Niedermeier, H. de Meer, and G. Giuliani, "A methodology to predict the power consumption of servers in data centres," in *Proceedings of the 2nd International Conference on Energy-Efficient Computing and Networking*, New York, New York, USA, May 2011, pp. 1–10, doi: 10.1145/2318716.2318718.

[115] F. Bellosa, “The benefits of event: driven energy accounting in power-sensitive systems,” in *Proceedings of the 9th workshop on ACM SIGOPS European workshop: beyond the PC: new challenges for the operating system*, Kolding, Denmark, Sep. 2000, pp. 37–42, doi: 10.1145/566726.566736.

[116] “Run-time power estimation in high performance microprocessors - IEEE Conference Publication.” https://ieeexplore.ieee.org/document/945389 (accessed Jul. 22, 2020).

[117] T. Li and L. K. John, “Run-time modeling and estimation of operating system power consumption,” *SIGMETRICS Perform. Eval. Rev.*, vol. 31, no. 1, p. 160, Jun. 2003, doi: 10.1145/885651.781048.

[118] X. Sun and N. Ansari, “Green Cloudlet Network: A Distributed Green Mobile Cloud Network,” *IEEE Network*, vol. 31, no. 1, pp. 64–70, Jan. 2017, doi: 10.1109/MNET.2017.1500293NM.

[119] M. Lin, A. Wierman, L. L. H. Andrew, and E. Thereska, “Dynamic right-sizing for power-proportional data centers,” in *2011 Proceedings IEEE INFOCOM*, Apr. 2011, pp. 1098–1106, doi: 10.1109/INFCOM.2011.5934885.

[120] M. Lin, Z. Liu, A. Wierman, and L. L. H. Andrew, “Online algorithms for geographical load balancing,” in *2012 International Green Computing Conference (IGCC)*, Jun. 2012, pp. 1–10, doi: 10.1109/IGCC.2012.6322266.

[121] H. Xu, C. Feng, and B. Li, “Temperature Aware Workload Managementin Geo-Distributed Data Centers,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 6, pp. 1743–1753, Jun. 2015, doi: 10.1109/TPDS.2014.2325836.

[122] A. Beloglazov and R. Buyya, “Energy Efficient Resource Management in Virtualized Cloud Data Centers,” in *2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, May 2010, pp. 826–831, doi: 10.1109/CCGRID.2010.46.

[123] X. Li, J. Wu, S. Tang, and S. Lu, “Let’s stay together: Towards traffic aware virtual machine placement in data centers,” in *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*, Apr. 2014, pp. 1842–1850, doi: 10.1109/INFOCOM.2014.6848123.

[124] L. Chen and H. Shen, “Consolidating complementary VMs with spatial/temporal-awareness in cloud datacenters,” in *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*, Apr. 2014, pp. 1033–1041, doi: 10.1109/INFOCOM.2014.6848033.

[125] Z. Han, H. Tan, R. Wang, G. Chen, Y. Li, and F. C. M. Lau, “Energy-Efficient Dynamic Virtual Machine Management in Data Centers,” *IEEE/ACM Trans. Netw.*, vol. 27, no. 1, pp. 344–360, Feb. 2019, doi: 10.1109/TNET.2019.2891787.

[126] S. Wang, R. Urgaonkar, T. He, M. Zafer, K. Chan, and K. K. Leung, “Mobility-Induced Service Migration in Mobile Micro-clouds,” in *2014 IEEE Military Communications Conference*, Oct. 2014, pp. 835–840, doi: 10.1109/MILCOM.2014.145.

[127] S. Wang, R. Urgaonkar, M. Zafer, T. He, K. Chan, and K. K. Leung, “Dynamic service migration in mobile edge-clouds,” in *2015 IFIP Networking Conference (IFIP Networking)*, May 2015, pp. 1–9, doi: 10.1109/IFIPNetworking.2015.7145316.

[128] Z. Becvar, J. Plachy, and P. Mach, “Path selection using handover in mobile networks with cloud-enabled small cells,” in *2014 IEEE 25th Annual International Symposium on Personal, Indoor, and Mobile Radio Communication (PIMRC)*, Sep. 2014, pp. 1480–1485, doi: 10.1109/PIMRC.2014.7136402.

[129] J. Plachy, Z. Becvar, and P. Mach, “Path selection enabling user mobility and efficient distribution of data for computation at the edge of mobile network,” *Computer Networks*, vol. 108, pp. 357–370, Oct. 2016, doi: 10.1016/j.comnet.2016.09.005.

[130] J. Plachy, Z. Becvar, and E. C. Strinati, “Dynamic resource allocation exploiting mobility prediction in mobile edge computing,” in *2016 IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, Sep. 2016, pp. 1–6, doi: 10.1109/PIMRC.2016.7794955.

[131] S. Wang, R. Urgaonkar, T. He, K. Chan, M. Zafer, and K. K. Leung, "Dynamic Service Placement for Mobile Micro-Clouds with Predicted Future Costs," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 4, pp. 1002–1016, Apr. 2017, doi: 10.1109/TPDS.2016.2604814.

[132] V. Di Valerio and F. Lo Presti, "Optimal Virtual Machines allocation in mobile femto-cloud computing: An MDP approach," in *2014 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, Apr. 2014, pp. 7–11, doi: 10.1109/WCNCW.2014.6934852.

[133] E. Badger, "A Day in The Life of 3 Million London Commuters, in 1 Minute," *CityLab*. http://www.theatlanticcities.com/commute/2013/02/day-life-3-million-london-commuters-1-minute/4739/ (accessed Oct. 29, 2017).

[134] J. Gordon, "Boston in Motion," *Jay Gordon*. http://jaygordon.net/bostonviz.html (accessed Mar. 17, 2018).

[135] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile Edge Computing: A Survey," *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 450–465, Feb. 2018, doi: 10.1109/JIOT.2017.2750180.

[136] G. Premsankar, M. D. Francesco, and T. Taleb, "Edge Computing for the Internet of Things: A Case Study," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 1275–1284, Apr. 2018, doi: 10.1109/JIOT.2018.2805263.

[137] L. Liu, Z. Chang, X. Guo, S. Mao, and T. Ristaniemi, "Multiobjective Optimization for Computation Offloading in Fog Computing," *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 283–294, Feb. 2018, doi: 10.1109/JIOT.2017.2780236.

[138] Y. Hao, M. Chen, L. Hu, M. S. Hossain, and A. Ghoneim, "Energy Efficient Task Caching and Offloading for Mobile Edge Computing," *IEEE Access*, vol. 6, pp. 11365–11373, 2018, doi: 10.1109/ACCESS.2018.2805798.

[139] J. Zhang, W. Xia, F. Yan, and L. Shen, "Joint Computation Offloading and Resource Allocation Optimization in Heterogeneous Networks With Mobile Edge Computing," *IEEE Access*, vol. 6, pp. 19324–19337, 2018, doi: 10.1109/ACCESS.2018.2819690.

[140] "Trace-based mobility modeling for multi-hop wireless networks," *Computer Communications*, vol. 34, no. 6, pp. 704–714, May 2011, doi: 10.1016/j.comcom.2010.11.002.

[141] C. Song, T. Koren, P. Wang, and A.-L. Barabási, "Modelling the scaling properties of human mobility," *Nature Physics*, vol. 6, no. 10, pp. 818–823, Oct. 2010, doi: 10.1038/nphys1760.

[142] M. S. Elbamby, C. Perfecto, M. Bennis, and K. Doppler, "Towards Low-Latency and Ultra-Reliable Virtual Reality," *arXiv:1801.07587 [cs, math]*, Jan. 2018, Accessed: Apr. 17, 2018. [Online]. Available: http://arxiv.org/abs/1801.07587.

[143] S. Andreev *et al.*, "Exploring synergy between communications, caching, and computing in 5G-grade deployments," *IEEE Communications Magazine*, vol. 54, no. 8, pp. 60–69, Aug. 2016, doi: 10.1109/MCOM.2016.7537178.

[144] Paris Agreement, "European Union," *Climate Action - European Commission*, Nov. 23, 2016. https://ec.europa.eu/clima/policies/international/negotiations/paris_en (accessed Aug. 07, 2018).

[145] International Energy Agency, "COP21." http://www.iea.org/cop21/ (accessed Aug. 07, 2018).

[146] "Gartner Says Data Centres Account for 23 Per Cent of Global ICT CO2 Emissions." https://www.gartner.com/newsroom/id/530912 (accessed Aug. 07, 2018).

[147] "Cisco Visual Networking Index: Forecast and Methodology, 2016–2021," *Cisco*. https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.html (accessed Aug. 07, 2018).

[148] E. Bastug, M. Bennis, M. Medard, and M. Debbah, "Toward Interconnected Virtual Reality: Opportunities, Challenges, and Enablers," *IEEE Communications Magazine*, vol. 55, no. 6, pp. 110–117, 2017, doi: 10.1109/MCOM.2017.1601089.

[149] “Power Model for Wireless Base Stations,” *imec*. https://www.imec-int.com/powermodel (accessed Aug. 17, 2018).

[150] G. Auer *et al.*, “Cellular Energy Efficiency Evaluation Framework,” in *2011 IEEE 73rd Vehicular Technology Conference (VTC Spring)*, May 2011, pp. 1–6, doi: 10.1109/VETECS.2011.5956750.

[151] ETSI STANDARD, “Environmental Engineering (EE); Assessment of mobile network energy efficiency.” European Telecommunications Standards Institute, Apr. 2015, Accessed: Aug. 10, 2018. [Online]. Available: https://www.etsi.org/deliver/etsi_es/203200_203299/203228/01.01.01_60/es_203228v010101p.pdf.

[152] C. A. Chan *et al.*, “Assessing network energy consumption of mobile applications,” *IEEE Communications Magazine*, vol. 53, no. 11, pp. 182–191, Nov. 2015, doi: 10.1109/MCOM.2015.7321989.

[153] L. A. Barroso and U. Hölzle, “The Case for Energy-Proportional Computing,” *Computer*, vol. 40, no. 12, pp. 33–37, Dec. 2007, doi: 10.1109/MC.2007.443.

[154] X. Fan, W.-D. Weber, and L. A. Barroso, “Power Provisioning for a Warehouse-sized Computer,” in *Proceedings of the 34th Annual International Symposium on Computer Architecture*, New York, NY, USA, 2007, pp. 13–23, doi: 10.1145/1250662.1250665.

[155] D. Gmach, J. Rolia, L. Cherkasova, G. Belrose, T. Turicchi, and A. Kemper, “An integrated approach to resource pool management: Policies, efficiency and quality metrics,” in *2008 IEEE International Conference on Dependable Systems and Networks With FTCS and DCC (DSN)*, Jun. 2008, pp. 326–335, doi: 10.1109/DSN.2008.4630101.

[156] A. Beloglazov and R. Buyya, “Adaptive Threshold-based Approach for Energy-efficient Consolidation of Virtual Machines in Cloud Data Centers,” in *Proceedings of the 8th International Workshop on Middleware for Grids, Clouds and e-Science*, New York, NY, USA, 2010, p. 4:1–4:6, doi: 10.1145/1890799.1890803.

[157] T. Taleb, M. Bagaa, and A. Ksentini, “User mobility-aware Virtual Network Function placement for Virtual 5G Network Infrastructure,” in *2015 IEEE International Conference on Communications (ICC)*, Jun. 2015, pp. 3879–3884, doi: 10.1109/ICC.2015.7248929.

[158] M. Chen, Y. Hao, M. Qiu, J. Song, D. Wu, and I. Humar, “Mobility-Aware Caching and Computation Offloading in 5G Ultra-Dense Cellular Networks,” *Sensors (Basel)*, vol. 16, no. 7, Jun. 2016, doi: 10.3390/s16070974.

[159] “VMware Distributed Power Management: Concepts and Usage.” https://www.vmware.com/techpapers/2008/vmware-distributed-power-management-concepts-and-1080.html (accessed Aug. 09, 2018).

[160] D. Meisner, B. T. Gold, and T. F. Wenisch, “PowerNap: Eliminating Server Idle Power,” in *Proceedings of the 14th International Conference on Architectural Support for Programming Languages and Operating Systems*, New York, NY, USA, 2009, pp. 205–216, doi: 10.1145/1508244.1508269.

[161] “Dell PowerEdge Servers | Dell Australia,” *Dell*. https://www.dell.com/en-au/work/shop/dell-poweredge-servers/sc/servers (accessed Aug. 23, 2018).

[162] A. GreenTouch, “GreenTouch Technical Solutions for Energy Efficient Mobile Networks Improving the nationwide energy efficiency in 2020 by more than a factor of 10000 in relation to the 2010 reference scenario,” 2015. /paper/GreenTouch-Technical-Solutions-for-Energy-Efficient-GreenTouch/e9201ba8848eec2643dc7c51971986ec28c35c32 (accessed Aug. 19, 2018).

[163] T. X. Tran, A. Hajisami, P. Pandey, and D. Pompili, “Collaborative Mobile Edge Computing in 5G Networks: New Paradigms, Scenarios, and Challenges,” *IEEE Communications Magazine*, vol. 55, no. 4, pp. 54–61, Apr. 2017, doi: 10.1109/MCOM.2017.1600863.

[164] M. Alicherry and T. V. Lakshman, “Network aware resource allocation in distributed clouds,” in *2012 Proceedings IEEE INFOCOM*, Mar. 2012, pp. 963–971, doi: 10.1109/INFCOM.2012.6195847.

[165] J. Kangasharju, J. Roberts, and K. W. Ross, "Object replication strategies in content distribution networks," *Computer Communications*, vol. 25, no. 4, pp. 376–383, Mar. 2002, doi: 10.1016/S0140-3664(01)00409-1.

[166] S. E. Mahmoodi, R. N. Uma, and K. P. Subbalakshmi, "Optimal Joint Scheduling and Cloud Offloading for Mobile Applications," *IEEE Transactions on Cloud Computing*, pp. 1–1, 2018, doi: 10.1109/TCC.2016.2560808.

[167] T. Taleb and A. Ksentini, "An analytical model for Follow Me Cloud," in *2013 IEEE Global Communications Conference (GLOBECOM)*, Dec. 2013, pp. 1291–1296, doi: 10.1109/GLOCOM.2013.6831252.

[168] R. Urgaonkar, S. Wang, T. He, M. Zafer, K. Chan, and K. K. Leung, "Dynamic service migration and workload scheduling in edge-clouds," *Performance Evaluation*, vol. 91, pp. 205–228, Sep. 2015, doi: 10.1016/j.peva.2015.06.013.

[169] L. Tong, Y. Li, and W. Gao, "A hierarchical edge cloud architecture for mobile computing," in *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, Apr. 2016, pp. 1–9, doi: 10.1109/INFOCOM.2016.7524340.

[170] S. Wang, M. Zafer, and K. K. Leung, "Online Placement of Multi-Component Applications in Edge Computing Environments," *IEEE Access*, vol. 5, pp. 2514–2533, 2017, doi: 10.1109/ACCESS.2017.2665971.

[171] V. Karagiannis and A. Papageorgiou, "Network-integrated edge computing orchestrator for application placement," in *2017 13th International Conference on Network and Service Management (CNSM)*, Nov. 2017, pp. 1–5, doi: 10.23919/CNSM.2017.8256008.

[172] "OpenStack Docs: Filter Scheduler." https://docs.openstack.org/nova/latest/user/filter-scheduler.html (accessed Jan. 15, 2019).

[173] R. M. Karp, "Reducibility among Combinatorial Problems," in *Complexity of Computer Computations: Proceedings of a symposium on the Complexity of Computer Computations, held March 20–22, 1972, at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, and sponsored by the Office of Naval Research, Mathematics Program, IBM World Trade Corporation, and the IBM Research Mathematical Sciences Department*, R. E. Miller, J. W. Thatcher, and J. D. Bohlinger, Eds. Boston, MA: Springer US, 1972, pp. 85–103.

[174] M. Rajib Arefin, T. Hossain, and M. Ainul Islam, "Additive Algorithm for Solving 0-1 Integer Linear Fractional Programming Problem," *Dhaka University Journal of Science*, vol. 61, Nov. 2013, doi: 10.3329/dujs.v61i2.17066.

[175] E. Balas, F. Glover, and S. Zionts, "An Additive Algorithm for Solving Linear Programs with Zero-One Variables," *Operations Research*, vol. 13, no. 4, pp. 517–549, 1965.

[176] A. Geoffrion, "Integer Programming by Implicit Enumeration and Balas' Method," *SIAM Rev.*, vol. 9, no. 2, pp. 178–190, Apr. 1967, doi: 10.1137/1009031.

[177] J. ter Wengel, "Allocations of Industries Given Generalized Distributional Constraints," in *Allocation of Industry in the Andean Common Market*, J. ter Wengel, Ed. Dordrecht: Springer Netherlands, 1980, pp. 109–131.

[178] L. Jiao, A. M. Tulino, J. Llorca, Y. Jin, and A. Sala, "Smoothed Online Resource Allocation in Multi-Tier Distributed Cloud Networks," *IEEE/ACM Transactions on Networking*, vol. 25, no. 4, pp. 2556–2570, Aug. 2017, doi: 10.1109/TNET.2017.2707142.

[179] S. Thananjeyan, C. A. Chan, E. Wong, and A. Nirmalathas, "Deployment and Resource Distribution of Mobile Edge Hosts Based on Correlated User Mobility," *IEEE Access*, vol. 7, pp. 148–159, 2019, doi: 10.1109/ACCESS.2018.2885119.

[180] X. Chen, H. Zhang, C. Wu, S. Mao, Y. Ji, and M. Bennis, "Optimized Computation Offloading Performance in Virtual Edge Computing Systems Via Deep Reinforcement Learning," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4005–4018, Jun. 2019, doi: 10.1109/JIOT.2018.2876279.

[181] S. Yu, X. Wang, and R. Langar, "Computation offloading for mobile edge computing: A deep learning approach," in *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, Oct. 2017, pp. 1–6, doi: 10.1109/PIMRC.2017.8292514.

[182] M. Wolf, "Chapter 5 - Processors and Systems," in *The Physics of Computing*, M. Wolf, Ed. Boston: Morgan Kaufmann, 2017, pp. 149–203.

[183] C.-C. Lin, P. Liu, and J.-J. Wu, "Energy-efficient Virtual Machine Provision Algorithms for Cloud Systems," in *2011 Fourth IEEE International Conference on Utility and Cloud Computing*, Dec. 2011, pp. 81–88, doi: 10.1109/UCC.2011.21.

[184] A. Strunk and W. Dargie, "Does Live Migration of Virtual Machines Cost Energy?," in *2013 IEEE 27th International Conference on Advanced Information Networking and Applications (AINA)*, Mar. 2013, pp. 514–521, doi: 10.1109/AINA.2013.137.

[185] I. A. Alimi, A. L. Teixeira, and P. P. Monteiro, "Toward an Efficient C-RAN Optical Fronthaul for the Future Networks: A Tutorial on Technologies, Requirements, Challenges, and Solutions," *IEEE Communications Surveys Tutorials*, vol. 20, no. 1, pp. 708–769, Firstquarter 2018, doi: 10.1109/COMST.2017.2773462.

[186] A. Vishwanath, F. Jalali, K. Hinton, T. Alpcan, R. W. A. Ayre, and R. S. Tucker, "Energy Consumption Comparison of Interactive Cloud-Based and Local Applications," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 4, pp. 616–626, Apr. 2015, doi: 10.1109/JSAC.2015.2393431.

[187] T. Subramanya, D. Harutyunyan, and R. Riggio, "Machine learning-driven service function chain placement and scaling in MEC-enabled 5G networks," *Computer Networks*, vol. 166, p. 106980, Jan. 2020, doi: 10.1016/j.comnet.2019.106980.

[188] M. H. Chen, M. Dong, and B. Liang, "Joint offloading decision and resource allocation for mobile cloud with computing access point," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Mar. 2016, pp. 3516–3520, doi: 10.1109/ICASSP.2016.7472331.

[189] S. Barbarossa, S. Sardellitti, and P. D. Lorenzo, "Joint allocation of computation and communication resources in multiuser mobile cloud computing," in *2013 IEEE 14th Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, Jun. 2013, pp. 26–30, doi: 10.1109/SPAWC.2013.6612005.

[190] N. Kiran, C. Pan, S. Wang, and C. Yin, "Joint resource allocation and computation offloading in mobile edge computing for SDN based wireless networks," *Journal of Communications and Networks*, vol. 22, no. 1, pp. 1–11, Feb. 2020, doi: 10.1109/JCN.2019.000046.

[191] C. Wang, C. Liang, F. R. Yu, Q. Chen, and L. Tang, "Joint computation offloading, resource allocation and content caching in cellular networks with mobile edge computing," in *2017 IEEE International Conference on Communications (ICC)*, May 2017, pp. 1–6, doi: 10.1109/ICC.2017.7996857.

[192] I. A. Elgendy, W. Zhang, Y.-C. Tian, and K. Li, "Resource allocation and computation offloading with data security for mobile edge computing," *Future Generation Computer Systems*, vol. 100, pp. 531–541, Nov. 2019, doi: 10.1016/j.future.2019.05.037.

[193] S. K. Datta and C. Bonnet, "MEC and IoT Based Automatic Agent Reconfiguration in Industry 4.0," in *2018 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*, Dec. 2018, pp. 1–5, doi: 10.1109/ANTS.2018.8710126.

[194] L. Lei, H. Xu, X. Xiong, K. Zheng, and W. Xiang, "Joint Computation Offloading and Multiuser Scheduling Using Approximate Dynamic Programming in NB-IoT Edge Computing System," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 5345–5362, Jun. 2019, doi: 10.1109/JIOT.2019.2900550.

[195] P. A. Apostolopoulos, E. E. Tsiropoulou, and S. Papavassiliou, "Cognitive Data Offloading in Mobile Edge Computing for Internet of Things," *IEEE Access*, vol. 8, pp. 55736–55749, 2020, doi: 10.1109/ACCESS.2020.2981837.

[196] K. Zhang, Y. Mao, S. Leng, Y. He, and Y. ZHANG, "Mobile-Edge Computing for Vehicular Networks: A Promising Network Paradigm with Predictive Off-Loading," *IEEE Vehicular Technology Magazine*, vol. 12, no. 2, pp. 36–44, Jun. 2017, doi: 10.1109/MVT.2017.2668838.

[197] R. Xie, Q. Tang, Q. Wang, X. Liu, F. R. Yu, and T. Huang, "Collaborative Vehicular Edge Computing Networks: Architecture Design and Research Challenges," *IEEE Access*, vol. 7, pp. 178942–178952, 2019, doi: 10.1109/ACCESS.2019.2957749.

[198] X. Huang, K. Xu, C. Lai, Q. Chen, and J. Zhang, "Energy-efficient offloading decision-making for mobile edge computing in vehicular networks," *J Wireless Com Network*, vol. 2020, no. 1, p. 35, Feb. 2020, doi: 10.1186/s13638-020-1652-5.

[199] S. Raza *et al.*, "An efficient task offloading scheme in vehicular edge computing," *Journal of Cloud Computing*, vol. 9, no. 1, p. 28, Jun. 2020, doi: 10.1186/s13677-020-00175-w.

[200] S. Singh and J. Singh, "Location Driven Edge Assisted Device and Solutions for Intelligent Transportation," in *Fog, Edge, and Pervasive Computing in Intelligent IoT Driven Applications*, IEEE, 2021, pp. 123–147.

[201] C. Chen, B. Liu, S. Wan, P. Qiao, and Q. Pei, "An Edge Traffic Flow Detection Scheme Based on Deep Learning in an Intelligent Transportation System," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–13, 2020, doi: 10.1109/TITS.2020.3025687.

[202] Z. Lv, D. Chen, and Q. Wang, "Diversified Technologies in Internet of Vehicles Under Intelligent Edge Computing," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–12, 2020, doi: 10.1109/TITS.2020.3019756.