
UNIVERSITÁ DEGLI STUDI DI PADOVA

Department of Information Engineering

---

Ph.D. COURSE IN: INFORMATION ENGINEERING

CURRICULUM: INFORMATION and COMMUNICATION TECHNOLOGY

Series: XXXII

# Core Network Management Procedures for Self-Organized and Sustainable 5G Cellular Networks

**Supervisor**: Prof. Michele Rossi

**Coordinator**: Prof. Andrea Neviani

**Ph.D. student**: Thembelihle Dlamini

---

Academic Year 2018-2019

# Abstract

This *thesis* investigates resource management procedures, within the *Multi-access Edge Computing (MEC)* paradigm, to obtain energy savings and guarantee Quality of Service (QoS) in Mobile Networks (MNs). Here, we enable energy savings within green-aware network apparatuses (i.e., communication and computing facilities) through the application of learning and control techniques, together with energy management procedures (BS sleep mode, VM soft-scaling, tuning of transmission drivers). In this study, we consider the MEC deployment scenarios suggested by ETSI and mobile operators for our system models.

Firstly, we investigate energy-saving strategies within a remote site fully powered by only green/renewable energy (solar and wind). Here, we consider a single Base Station (BS) co-located with the MEC server, i.e., the BS is empowered with computing capabilities. To address the energy consumption problem within the remote site, we propose an online algorithm for edge network management. The algorithm make use of a Long Short-Term Memory (LSTM) neural network for estimating the short-term future traffic load and harvested energy, and control theory, specifically the Limited Lookahead Control (LLC) principles, for foresighted optimization. It also make use of energy management procedures, i.e., BS sleep modes and Virtual Machine (VM) soft-scaling (the reduction of computing resources per time instance). To obtain the energy savings and guarantee QoS, per time instance, the algorithm considers the future BS loads, onsite green energy available and then provisions edge network resources based on the learned information.

Secondly, we study the energy consumption problem within an environment where BSs are densely-deployed, i.e., similar to an urban or semi-urban scenario. This work extend the energy consumption problem from a single BS case to multiple BSs. Here, each BS is powered by hybrid energy supplies (solar and power grid) and also empowered with computation capabilities (each BS is co-located with a MEC server). Towards edge system management, we propose a controller-based network architecture for man-

aging energy harvesting (EH) BSs empowered with computation capabilities where on/off switching strategies allow BSs and VMs to be dynamically switched on/off, depending on the traffic load and the harvested energy forecast, over a given look-ahead prediction horizon. To solve the energy consumption minimization problem in a distributed manner, the controller partitions the BSs into clusters based on their location; then, for each cluster, it minimizes a cost function capturing the individual communication site energy consumption and the users' QoS. To manage the communication sites, the controller performs online supervisory control by forecasting the traffic load and the harvested energy using a LSTM neural network, which is utilized within a LLC policy to obtain the system control actions that yield the desired trade-off between energy consumption and QoS.

Finally, we investigate the energy consumption problem within a virtualized MEC server placed in proximity to a group of BSs. To address this challenge, we consider a computing-plus-communication energy model, within the MEC paradigm, where we focus on the communication-related energy cost in addition to the energy drained due to computing processes. Towards server management, an online algorithm based on traffic engineering and MEC Location Service is proposed. To obtain the energy savings and QoS guarantee, we jointly launch an optimal number of VMs for computing and transmission drivers coupled with the location-aware traffic routing for real-time data transfers. In order to efficiently provisioned edge system resources, we forecast the server workloads and harvested energy by using a LSTM neural network and the output is then used within the LLC-based algorithm.

Our numerical results, obtained through trace-driven simulations, show that the proposed optimization strategies (algorithms) leads to a considerable reduction in the energy consumed by the edge computing and communication facilities, promoting energy self-sustainability within the MN through the use of green energy.

# Dedication

*Dedicated to my family*

# Acknowledgements

Words are often less to reveal one's deep regards. An understanding of work like this is never an outcome of a single person. I would like to take this opportunity to express my profound sense of gratitude to my supervisor Professor Michele Rossi and fellow researcher Ángel Fernández Gambín, for their guidance and editorial comments during the course of my studies.

I would like to thank God, who has given me the strength to work on this research, despite of daily challenges, and guided me to work on the right path of life. Without his grace this would never been a success. I am also very grateful to my family, Mrs. N. Shabangu-Dlamini, Uncle, Aunt and church members for keeping me in their prayers so that I can be able to complete this work.

Lastly, this work received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No. 675891. I want to thank and wish the best of luck to my fellow researchers with whom I worked with within the SCAVENGE project.

# Contents

# List of Tables

# List of Figures

# List of Acronyms

| | |
|---|---|
| **5G** | Fifth Generation |
| **API** | Application Programmable Interface |
| **ARCES** | Automated Resource Controller for Energy-aware Server |
| **BS** | Base Station |
| **CDR** | Call Detail Record |
| **CNN** | Convolutional Neural Network |
| **CPU** | Central Processing Unit |
| **DETA-R** | Dynamic and Energy-Traffic-Aware algorithm with Random behavior |
| **DVFS** | Dynamic Voltage and Frequency Scaling |
| **EB** | Energy Buffer |
| **EH** | Energy Harvesting |
| **EM** | Energy Manager |
| **ENAAM** | Energy Aware and Adaptive Management |
| **EPC** | Evolved Packet Core |
| **ES** | Energy Saving |
| **ETSI** | European Telecommunications Standards Institute |
| **FTP** | File Transfer Protocol |
| **GP** | Geometric Programming |
| **HTTP** | HyperText Transfer Protocol |

| | |
|---|---|
| **ICT** | Information and Communication Technology |
| **IP** | Internet Protocol |
| **IRS** | Iterative-based Resource Scheduler |
| **ITS** | Intelligent Transport System |
| **LLC** | Limited Lookahead Control |
| **LS** | Location Service |
| **LSTM** | Long Short-Term Memory |
| **LTE** | Long Term Evolution |
| **MEC** | Multi-access Edge Computing |
| **ML** | Machine Learning |
| **MN** | Mobile Network |
| **MPC** | Model Predictive Control |
| **NF** | Network Function |
| **NFV** | Network Function Virtualization |
| **NIC** | Network Interface Card |
| **OS** | Operating System |
| **QoE** | Quality of Experience |
| **QoS** | Quality of Service |
| **RAN** | Radio Access Network |
| **RMSE** | Root Mean Square Error |
| **RNIS** | Radio Network Information Services |
| **RNN** | Recurrent Neural Network |
| **SDN** | Software Defined Network |
| **SMS** | Short Message Service |

| | |
|---|---|
| **TCP** | Transmission Control Protocol |
| **TIM** | Telecom Italia Mobile |
| **UE** | User Equipment |
| **VLAN** | Virtual Local Area Network |
| **VM** | Virtual Machine |
| **VMM** | Virtual Machine Monitor |
| **VNF** | Virtualized Network Function |

# Chapter 1
# Introduction

## 1.1 Context

The evolution towards a softwarized Evolved Packet Core (EPC) is the driving force towards overcoming the challenges observed in current Mobile Networks (MNs) and set the way for high data rate and ultra-low latency 5G networks. Such changes avails the possibility of running Network Functions (NFs) in software, instead of proprietary hardware devices, and also it permit the possibility of dynamically scaling the network resources for a more robust network management in 5G and beyond as network complexity is reduced. Softwarization and virtualization of resources and services will provide the mechanism for network management, i.e., flexibility and adaptability will be guaranteed [1][2]. This is largely reliant on the use of virtualization technologies and virtualized platforms.

Different approaches have been investigated towards the state-of-the-art EPC architecture where vendors and researchers proposed the (i) grouping of the EPC functions [3][4][5], (ii) running the virtualized functions on clouds [6][7], (iii) partitioning the network resources into network slices [8][9][10][11], which refers to an isolated set of (programmable) resources to enable NFs and services, and (iv) redesigning the network to be based on Network Function Virtualization (NFV) technology [12][13]. A detailed summary of the architecture proposals and approaches, similarities and differences, their contributions towards an energy efficient MN and limitations, is presented in [2].

The architectural evolution involving redesigning the MN based on NFV is currently appealing towards 5G and beyond, as it allows the virtualization of the mobile NFs and then placing them within the access network. This is motivated by the expected data explosion in the volume, variety and velocity, generated by pervasive mobile devices and the Internet of Things [14] at the network edge (i.e., in close proximity to mobile

devices, sensors, actuators and connected things). This is coupled with the demand for stringent latency, requiring high computation resources which are not available at the end-user device. As a remedy, *Multi-access Edge Computing (MEC)* has recently emerged to enable ultra-low latency, distributed intelligence and location-aware data processing at the network edge [12][13]. Undoubtedly, offloading to a powerful computation resource-enriched MEC server located closer to mobile users is an ideal solution. Specifically, by offloading data- and computing-intensive tasks to the nearby MEC-enabled Base Stations (BSs), resource-constrained mobile devices benefit from reduced energy consumption and task completion time. Although this new approach is not intended to replace the cloud-based infrastructure, it expands the cloud by increasing the computing and storage resources available at the network edge. In order to reap the full benefits of the virtualized infrastructure, the NFV technology shall be combined with intelligent mechanisms for handling network resources management.

As suggested by the European Telecommunications Standards Institute (ETSI) [12][15] and mobile operators [16], the Virtualized Network Functions (VNFs) can be deployed at the BS, i.e., the BS site is empowered with computing capabilities (MEC server is co-located with the BS), or placed at an aggregation point (a point in close proximity to a group of BS) for edge network management. Some of the NFs that formerly only existed in the EPC are migrated to the network edge resulting into significant savings in cost, latency, round trip time (RTT), traffic download time and caching efficiency. Placing intelligent nodes at the network edge enables the entire system to benefit from more effective executive processes, primarily because the delay involved in reaching the remote cloud is eliminated. Furthermore, local service management and access control policies can be defined.

In light of the dense deployment pattern that is foreseen in 5G systems [17], the expected dense deployment of MEC servers and BSs raises concerns related to *energy consumption*. Specifically, energy drained in BSs is due to the always-on approach and in MEC servers it is due to the computing and communication processes associated with: (i) the running Virtual Machines (VMs) [18][19]; (ii) the communication within the server's Virtual Local Area Network (VLAN) [20], and the presence of transmission drivers (*fast* tunable optical drivers for data transfer) within the MN infrastructure [21][22]. Towards greener MNs, there arise a need for efficient edge network management procedures that will yield the trade-off between energy savings and the guarantee of Quality of Service (QoS).

The idea of using *renewable* or *green* energy sources to energize the computing and

communication systems (network apparatuses), coupled with energy management procedures, is being considered towards the minimization of energy consumption, carbon footprint, operational expenses (OPEX) in terms of annual electricity bills, and the dependence on the conventional electricity grid in future MNs [2][23]. From mobile operators perspective, reducing electrical energy consumption is not only a matter of being green and responsible, it is also very much an economically important issue. The motivation towards green energy usage is due to the fact that the components in solar- and wind-based systems are usually modular, which makes the design, expansion, and installation of these types of systems for the BS sites very practical and feasible. This result in EH-powered BSs (EH BSs) and MEC (EH-MEC) systems. The use of renewable energy, coupled with the integration of MEC and BSs, can help extend network coverage to areas where the electrical infrastructure cannot reach, or assist during the case of a natural disaster scenario as the network can work in isolation assuming the presence of the EPC application in the MEC server, where the conventional electricity grid may become unavailable. Here, the assumption is that green energy is either practically free or at least cheaper than grid power, neglecting the capital cost of the EH systems (e.g., solar panels, wind turbines) as it depends on the return of investment time. Despite the use of green energy for powering network systems, the integration of MEC and EH BS systems brings about new challenges related to energy consumption, and resource scheduling. Thus, proper energy management strategies are required, as effective operation of MEC is contingent upon this.

This *thesis* attempts to put forward research on dynamic resource management towards energy saving and QoS guarantee, within the MEC paradigm, for green-aware network apparatuses. Here, different MEC deployment scenarios are considered. In addition, we use open source mobile traffic and harvested energy traces for developing traffic-oriented network management procedures. In this research work, we identify and quantify the source of energy consumption within a communication site (or computing platform) and then propose online-based algorithms for green-aware dynamic resource allocation and edge network management. The online algorithms use Machine Learning (ML) tools, specifically the Long Short-Term Memory (LSTM) neural network [24] [25], to forecast the short-term traffic load and energy harvested (solar and/or wind energy), and then employing control-theorectic techniques (specifically the Limited Lookahead Control (LLC) principles for foresighted optimization [26][27][28]) and heuristics to obtain the best control input that yields the best system behavior, per time instance.

We *argue* that forecasting and foresighted optimization, over a given lookahead hori-

zon, can be considered for dynamic resource management in order to obtain Energy Savings (ESs) and guarantee QoS in MNs. We support this stance through trace-driven simulations for each considered MEC deployment scenario. In the remainder of this chapter, we discuss the main research objectives, outline the structure of the thesis and summarize the contributions of this work (we summarize the publications included in this dissertation).

Through out the thesis, we consider the BS as the default edge node and refer to the combination of an edge node and associated MEC server as an *edge system*. The edge network infrastructures consists of a two-layer architecture [29]: (i) edge node and (ii) heterogeneous end devices which lacks computation capabilities. The edge node is a networking component that can be characterized by a small to medium-size computing capability and aims to provide extra computing, storage, and networking resources. In the envisaged MEC scenarios, the edge node can embed complex functionality and become the key components in meeting the increasingly stringent application performance requirements, especially those concerning latency, as they are deployed closer to heterogeneous end devices. The end devices generate traffic that is either delay sensitive (locally processed workloads) or delay tolerant (i.e., standard workloads that is forwarded to remote cloud for processing).

## 1.2 Research Objectives

The overall objective of this thesis is to develop dynamic resource management procedures for edge system management, i.e., management of BSs and virtualized computing environments (MEC servers), and then validating them through trace-driven simulations. Through dynamic resource management procedures, we minimize the overall energy consumption in the communication site over time, i.e., the consumption related to the BS transmission and the MEC server activities, and also guarantee QoS. The ESs are obtained by *jointly* applying BS power saving modes (sleep modes), VM soft-scaling, i.e., the reduction of computing resources per time instance, and in addition, switching on/off the transmission drivers (*fast* tunable optical drivers) within the computing node. A parallel objective is to *green* the consumption of a network and this is explored through two areas:

- *Green Energy*: we partially or fully replace the powering of the edge systems through conventional energy sources with renewable energy sources such as solar

and wind energy. At each time instance, we select the appropriate renewable energy source taking into account current and forecast traffic loads.

- *Energy Storage*: we maximize the benefit from the integration of renewable energy systems by using energy storage devices, hereby termed *Energy Buffers (EBs)*. Also, the EB level is reported to the authorized MEC application using the pull transfer mode (e.g., File Transfer Protocol (FTP) [30]). That is, the MEC application pulls the energy report[1] from the Energy Manager (EM). The EM is an entity responsible for selecting the appropriate renewable energy source to fulfill the EB depending on the weather, and for monitoring the energy levels in the system.

While many of the targets have been studied, this thesis is unique in that it attempts to promote the use of green energy, development of traffic-oriented network solutions, application of learning and control techniques to edge systems, at the same time balancing energy savings and QoS. As a result, it produces several new insights on how and when energy management procedures can be beneficial within the MEC paradigm. For our simulations, *Python* is used as the programming language.

## 1.3 Manuscript Outline and thesis contributions

The outline of this manuscript is graphically shown in Figure 1.1. Chapter 2 discuss the literature review useful for all the remaining chapters. Chapters 3 investigate dynamic resource management procedures within a single BS site fully powered by green energy and Chapter 4 partially extends the single BS optimization case into multiple BS optimization (here, the BS system is partially powered by green energy) where the edge network management is handled by a controller. The chapters can be read following one another. Chapter 5 considers the energy consumption due to the computing and communication activities within the MEC server, and then investigate how energy consumption can be minimized. This chapter can be read alone. Finally, Chapter 6 draws the conclusions and describes some possible future research directions.

[1]In this work, we use open source datasets for solar and wind energy in order to emulate the energy reports. The solar traces are obtained from one solar farm located in Armenia and the wind traces are from one wind farm located in Belgium. For time slot scale not matching the dataset granularity, the data is aggregated to match the used time slot scale.

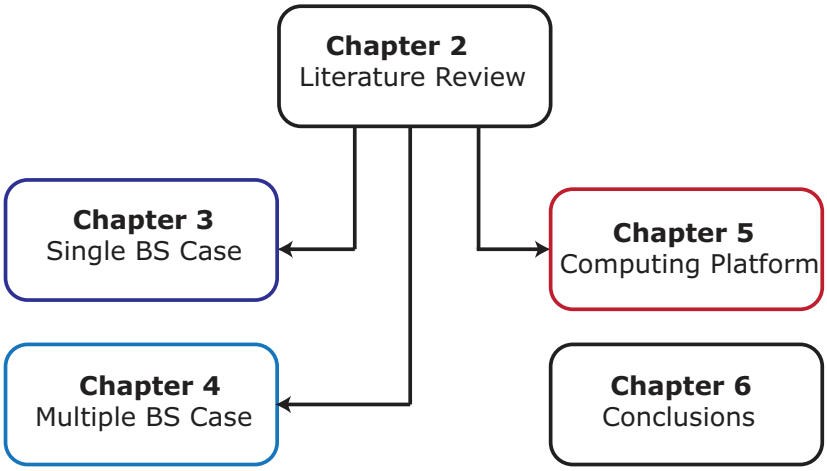



Figure 1.1: Manuscript outline showing each chapter target.

In order to deliver on the objectives outlined in Section 1.2, the main contributions of this thesis are summarized as follows.

Chapter 2 presents the background information related to our research.

- Section 2.1 presents the virtualization technologies and tools for the softwarization of MN, followed by NFV-based architectural EPC proposals. Here, the enabling technologies are discussed and they include Software Defined Network (SDN) and NFV, and the tools are: *hypervisor*, the environment allowing the computing resources (i.e., VMs) to run on top of the virtualization layer, and the *docker engine*, an environment for developing and running applications and also creating a loosely isolated environment called a *container*.

- The work related to energy efficient network management procedures is presented in Section 2.2. This work include sleep modes strategies in MNs, energy savings in virtualized computing platforms through soft-scaling, and pattern forecasting along with foresighted optimization techniques.

- Works related to optimization in MEC (edge) systems using green energy is discussed in Section 2.3. Here, we discuss work incorporating renewable energy within the MEC paradigm, the optimization techniques used toward energy consumption minimization. This is important, especially in light of the dense deployment pattern that is foreseen in 5G systems, whereby MEC servers are expected to be densely deployed to provide ultra-low latency services.

- Lastly, Section 2.4 presents optimization work that make use of mobile traf-

fic traces obtained from the mobile operator. Here, we review previous works analyzing the spatio-temporal evolution of mobile traffic and their application towards efficient and self-adaptable network solutions.

Chapter 3 presents research on online-based resource management achieved through prediction and soft-scaling of computing resources within remote site (or single off-grid BS site).

- In Section 3.2, the system model is presented. The use of open source mobile traffic load traces is explained and the energy consumption model is defined. In addition, the next time slot EB level energy model is formulated.

- The optimization problem to obtain energy savings through short-term traffic load and harvested energy predictions along with energy management procedures is formulated in Section 3.3. Here, a weighted cost function is formulated consisting of two functions, *one* weighs the energy consumption due to BS transmission and computation (MEC server) activities, and the *other* accounts for the QoS cost (i.e., we push the computation load to be entirely processed by the local MEC server).

- Section 3.4 presents a traffic load and energy harvesting prediction method, and an online algorithm for managing the remote site. In this section, the LSTM neural network is used to predict the short-term future traffic loads and harvested energy. Then, a state-space behavior of the control system is constructed and we also introduce the online control key concepts. To solve the energy consumption problem, an online algorithm based on control theory and heuristics is proposed for enabling dynamic resource management within the edge system through foresighted optimization. The online algorithm finds the best control input that yield the desired energy savings and system behavior.

- The proposed online resource management procedure is evaluated in Section 3.5. Numerical results, obtained with real-world energy and traffic traces, demonstrate that the proposed algorithm achieves energy savings between 56% and 66% on average, with respect to the case where no energy management techniques are applied, and hold the server utilization between 30% and 96% over time, with an average of 75%.

Chapter 4 presents research based on online supervisory control and resource management for EH BS sites empowered with computation capabilities. This chapter partially extends the work of Chapter 3, and in this chapter we consider an ultra-dense BS environment, similar to urban or semi-urban, in contrast to the single off-grid BS case, and the edge management is enabled by an *controller*. In addition, a hybrid energy supply is considered, solar energy as the main source and the power grid as a backup.

- The considered MEC deployment scenario and the energy consumption model is presented in Section 4.2. In this scenario, we consider that each EH BS is co-located with the MEC server, forming a cluster/group that is placed in proximity to the controller. In addition, the next time slot EB level energy model is formulated.

- Section 4.3 presents the optimization problem to obtain energy savings through short-term traffic load and harvested energy predictions, along with energy management procedures for a *single* communication site. Similar to Chapter 3, a weighted cost function is presented. Then, a traffic load and energy harvesting prediction method and an online management algorithm is proposed for single BS management. The online algorithm makes use of learning and control techniques, towards energy savings and the guarantee of QoS.

- Section 4.4 extends the work of Section 4.3 by considering the energy savings for multiple communication sites. We formulate an optimization problem to obtain energy savings through short-term traffic load and harvested energy predictions, clustering, along with energy management procedures for the clustered BS sites. In addition, we formulate the location-based (distance measure) clustering strategy. To solve the energy consumption minimization problem in a *distributed manner*, the online controller partitions the BSs into clusters based on their location; then, for each cluster, it minimizes a cost function capturing the individual communication site energy consumption and the users' QoS. The network impact is used as a performance metric for handling load balancing within the clustered BSs.

- The proposed online controller-based algorithm performance is evaluated in Section 4.5. Numerical results, obtained with real-world energy and traffic load traces, demonstrate that the proposed algorithm achieves energy savings between 57% and 69%, on average, for the single communication site case, and

a gain ranging from 9% to 16% on energy savings is observed when clustering is applied, with respect to the allocated maximum per-VM loads of 5 MB and 10 MB. The energy saving results are obtained with respect to the case where no energy management techniques are applied, either in one BS or single cluster.

Chapter 5 presents research related to adaptive resource allocation towards energy management within a virtualized computing platform (i.e., the MEC server). This work first address the short comings of the energy consumption model, i.e., the energy consumption include the overall computing-plus-communication energy consumption instead of assuming that the overall operational energy expenditure is related to solely the computation process. Here, we focus on the integration of communication-related energy consumption by considering the tuning of the transmission drivers.

- The system model and the computing-plus-communication energy consumption model is presented in Section 5.2. Here, we assume that the MEC server is virtualized, cache-enabled and Transmission Control Protocol (TCP)/Internet Protocol (IP) offload-enabled (partial computation is performed at the network adapter). The computing node is assumed to be deployed at an aggregation point and powered by hybrid energy sources: on-grid power for back-up and green energy (solar) as the main. The EB level update formula is defined.

- The optimization problem is presented in Section 5.3. We consider a computing-plus-communication energy model, within the MEC paradigm, formulating a constrained optimization problem. Then, we transform the non-convex function to a convex one using the Geometric Programming (GP) theory. The main goal is to minimize the overall energy consumption, under hard per-task delay constraints (i.e., QoS), through the joint consideration of VM soft-scaling and the tuning of transmission drivers, coupled with the location-aware traffic routing.

- In Section 5.4, we put forward a combination of a traffic engineering- and MEC Location Service-based online server management algorithm with EH capabilities for autoscaling and reconfiguring the computing-plus-communication resources. To obtain energy savings, we forecast the short-term future server workload and harvested energy, using LSTM network, allocating power using the GP concept, CVXOPT toolbox[2] and approximations. Then, an online algorithm using foresighted (control theory) optimization and heuristics is employed. The online

[2]CVXOPT is a free software package for convex optimization based on the Python programming language.

algorithm obtain the best control action that will adjust the computing system behavior at each time instance, with negligible computational overhead.

- Section 5.5 presents some selected numerical results of the proposed algorithm. Numerical results, obtained with real-world energy and server workload traces, demonstrate that the proposed algorithm achieves energy savings of 69%, on average, with an energy consumption ranging from 31% to 45% at high per-VM reconfiguration cost and from 21% to 25% at low per-VM reconfiguration cost, with respect to the case where no energy management techniques are applied.

Chapter 6 provides conclusions and future research directions.

# Chapter 2
# Background Information

There is growing awareness to the fact that the communication sector uses significant amount of energy [14]. This is especially true for wireless, and in particular for the BSs of cellular networks, where energy costs make up a large part of the operating expenses of mobile operators. In addition, in future MNs the BSs will be empowered with computation capabilities to enable local workload offloading (computation) and the provision of ultra-low latency services. This requires efficient energy management procedures. To handle the energy consumption challenge, we consider adaptive power control mechanisms for tuning the BS energy consumption in proportion to the traffic demand and the use of alternative energy sources (solar, wind). As EH technologies advance, energizing the network apparatuses (edge systems) with green energy (e.g., solar and/or wind) is a promising solution to reduce the dependence from on-grid power due to their location, reliability, carbon footprint and cost. Considering the ultra-dense deployment of BSs and the dynamic characteristics of MNs, energy saving is of great importance in green networks in order to exploit the benefits of sustainable networking and computing within the edge network.

Green networking and computing is a broad research area and it generally involves the combination of some of the following objectives:

- Minimizing the amount of energy drained/purchased from the power grid.
- Minimizing the total energy consumption of a network or part of the network.
- Minimizing the amount of the carbon footprint in order to support the energy requirements.
- Minimizing the network complexity through network softwarization and virtualization (currently being pursued in MNs).

In order to achieve the aforementioned objectives, green networking and computing research relies on several methods including:

- On-demand resource management, i.e., autoscaling and reconfiguration of the computing resources, together with the tuning of transmission drivers.
- Selectively switching on/off network elements to save energy.
- Integrating EH systems to harness the energy that energize the edge systems. This enables energy self-sufficiency and sustainability in MNs (this is expected in future MNs).
- Using content caching to store contents closer to mobile users in order to reduce the amount of traffic on a network (this is outside the scope of this thesis)

This chapter presents literature review on the EPC architectural evolution, energy management procedures in MNs focusing on green networking and computing, the use of renewable energy within MEC paradigm, and the use of mobile datasets towards traffic-oriented procedures for edge network management (i.e., dynamic BS and MEC server management mechanisms design). In Section 2.1, we explain the softwarization and virtualization tools, followed by NFV-based architectural proposals as an example of the EPC architectural evolution. Then, Section 2.2 looks at the literature related to energy management procedures in BSs, virtualized computing platforms, and the mathematical tools (the control-theoretic and ML methods) for online resource management. Section 2.3 covers literature specifically related to optimization in MEC systems using green energy. The importance of real MN traffic load traces for network optimization is discussed in Section 2.4.

## 2.1 Softwarization of Mobile Network Functions

Softwarization and virtualization of resources and services are undoubtedly among the main drivers of 5G and beyond 5G MNs, as they will provide flexibility and adaptability, and also facilitate network maintenance and the update of all NFs. Towards this end, researchers from industry and academia have presented different proposals towards the EPC architectural evolution (*see* [2] for more details about EPC architectural proposals). Their contributions results into fragmented inputs, with a unified goal of having an energy efficient EPC architecture for 5G networks. The outcome are

designs/proposals that are somehow overlapping in terms of the functions being softwarized, technologies used, and so forth. Here, we discuss the enabling technologies and tools in subsection 2.1.1, followed by the NFV-based architectural proposals in subsection 2.1.2, as it is one of the appealing architectures towards meeting the latency and data processing requirements at the network edge.

### 2.1.1 Virtualization Technologies and Tools

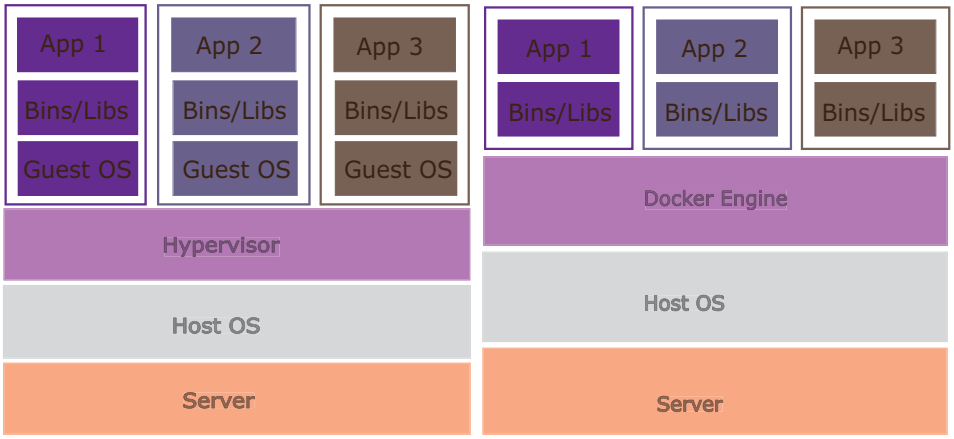
SDN and NFV are the emerging virtualization technologies that will enable flexibility and agility in MNs. SDN involves the separation of the control plane of the network devices from the data plane, allowing a centralized approach for networking control that provides simplification and global optimization for the routing and switching of network packets [31][32][33]. It also advocates for open Application Programmable Interfaces (APIs) and a programmatic approach to networking. In addition, it is an essential element of 5G that enables fast service provisioning (and de-provisioning) as well as the optimal use of the underlying transport infrastructure. Through its support for a programmable network, SDN-based architectures can also be implemented to ensure that end-to-end paths are provisioned efficiently to maximize the transport from the mobile User Equipment (UE) to the data center or edge services that they are likely to consume.

NFV is a complementary movement to SDN, leveraging virtualization to take proprietary physical hardware that is non-portable and hard to manage, and convert the NFs into virtualized software-only versions [33][34]. These VNFs can be quickly moved around as needed, and scaled up or down dynamically. NFV-based architectures support the ability to provision network slicing services flexibly, and this will likely take the form of data centers spread across the network from core to edge to reduce the latency of these services and for load distribution.

The combination of SDN and NFV is critical in achieving flexible network topology and the realization of 5G targets such as 1000-times higher system capacity; 100-times increase in data rates (10-Gb/s speeds); connectivity enablement for 100-times more devices; latency reduction from 5 ms to 1 ms; and energy savings. Both technologies limit the use of specialized hardware devices as they have been the limiting factor towards MNs evolution and the fast deployment of new services within the mobile space. Also, the technologies can co-exist within the same network, where SDN employs a centralized approach on switching and routing elements [3], and NFV migrate the NFs out of dedicated hardware into software that is imported into general purpose hardware.

These technologies are expected to facilitate the ease of efficient network management. The *key* to virtualization is that it enables *on-demand* or *utility* computing, a just in time resource provisioning model in which computing resources such as Central Processing Unit (CPU), memory, and disk space are made available to applications only as needed [26].

The tools used in virtualized computing platforms consists of the *hypervisor* and *docker engine*, and they are illustrated in Fig. 2.1. Virtualization making use of the hypervisor is referred to as `hypervisor-based` virtualization, with the VMs as the computing resources, and for the environment using docker engine is referred to as `container-based` virtualization, with the *containers* as the computing resources.




(a) Hypervisor-based virtualization (b) Container-based virtualization

Figure 2.1: An illustration showing the structure of VMs on top of the hypervisor (2.1a) and containers on top of the docker engine (2.1b).

The *hypervisor* is the software that provides the environment in which the VMs operate. A Hypervisor-based virtualization simply isolates the Operating System (OS) and applications from the underlying computer hardware [35]. This abstraction allows the underlying "host machine" hardware to independently operate one or more VMs as "guest machines" (also referred to as guest VMs), allowing them to share the system's physical computing resources, such as processing time, memory space, network bandwidth, etc. A new agnostic OS is generated to manage the underlying resources. Since the hypervisor sit between the actual physical hardware and the guest OS, it is also referred to as Virtual Machine Monitor (VMM). In Fig. 2.1a, we observe that each VM has a virtual OS of its own and the hypervisor provides the VMs with a platform to manage and execute multiple guest OS, and allows host computers to share their resources among them. A drawback of VMs is that they can be slow to boot.

*Docker engine* is a software technology written in the `Go` programming language and it avails the environment for developing and running applications. It runs natively on Linux systems (in recent Linux kernels), where it uses Linux kernel features like `namespaces`, to provides a private, restricted view on certain system resources within a container (i.e., a form of sandboxing), and `control` groups (cgroups), a technology that limits an application to a specific set of resources (i.e., provides resource management for groups of processes), to create a loosely isolated environment called a container. Mainly, it provides tooling, that is, software packaging tools that can package an application and its dependencies in a virtual container that can run on any Linux server, and a platform to manage the containers life-cycle. *Containers* are abstraction units for isolating applications and their dependencies, that can run in any environment. They can run on the same machine, on top of the docker engine, sharing the OS kernel with other containers. Fig. 2.1b shows that containers only package up the user space, and not the kernel or virtual hardware, like a VM does. Each container gets its own isolated user space to allow multiple containers to run on a single host machine. We observe that the entire OS level architecture is being shared across them. The only parts that are created from scratch are the `Bins` and `Libs`. Despite the process isolation and lightweight character, container-based virtualization is less secure and more vulnerable compared to hypervisor-based virtualization.

*Currently*, the most commonly used virtualization tool is the one illustrated in Fig. 2.1a (key platforms include VMware vSphere, Microsoft Hyper-V, Citrix Xenserver or KVM). The tool also enable live VM migration (one of the key features of virtualization), which is a process of duplicating and transmitting the VM memory image over the network, in virtualized data centers, without or least service interruption [36][37]. With the continuous advancements in virtualization technologies and tools, the future looks different as containers will probably co-exist with hypervisors. The use of containers, running on top of the docker engine, speeds up innovation, requires less space and can be deployed across different platforms, and hypervisors, with VMs on top, allows running multiple applications on multiple VMs. The combination of the virtualization tools can be beneficial as operators can not be restricted to one infrastructure, instead they can simply develop applications once and then run them on any infrastructure [38]. For example, the Unikernel, as a hybrid solution, promises to combine the advantages of the aforementioned tools [39].

Determining which is the most suitable tool (hypevisor- or container-based) to be used is a *challenge*. Mainly, this depends on the specific scenario and from that it re-

quires a thorough analysis of the scenario design and performance requirements, along with a meticulous analysis of the benefits and drawbacks associated with the use of one tool rather than another. The quantitative comparison of different virtualization techniques is presented in [40]. In light of the significant impact that software virtualization has in the MEC paradigm, in this work we focus on the hypervisor-based virtualization, where VMs are deployed on the MEC server for handling computation activities (delay sensitive workloads), due to the fact that (i) VMs are secure and (ii) hypervisor-based VMs can benefit from a well-established management mechanisms [35].

**Discussion**: The virtualization tools can play a role towards Energy Efficiency (EE) improvement within MNs. For example, from the hypervisor-based virtualization, the hypervisor can report resource usage to the orchestrator in order to trigger system automated sleep mode states and also to implement policies provided by management and orchestration entity, which includes power management and power stepping [35]. On the other hand, to enable the next level of flexibility in MNs, operators might have to turn to more recent innovations in the data center world, containers and cloud-native capabilities. Container-based virtualization can be considered as an alternative to hypervisor-based virtualization, as containers demand less memory space, portable (allow applications to be separated from the underlying infrastructure) and lightweight, and have a shorter start-up time which translates to low latency and power consumption as confirmed experimentally in [18]. They also provide lower overheads compared to VMs, isolation from infrastructure and other tenants [33], speed and agility to allow applications to be tested and deployed quickly.

### 2.1.2 NFV-based Mobile Network Evolution

The use of *cloud-based approach* with NFV platform has been proposed towards EPC architectural evolution. This type of architecture enable dynamic deployment of edge networks, the scaling of NFs, network monitoring, and load management. In addition, it avails the possibility of intelligently pooling capacity of resources when required.

In [6], the key elements of the architecture are 1) a data-driven network intelligence for optimizing network resources usage and planning, and 2) relaying and nesting techniques: to support multiple devices, group mobility and nomadic hotspots. The EPC is virtualized into three parts, namely; i) *control* plane entity (CPE), which is responsible for authentication, mobility management, radio resource control and non-access stratum (NAS) and access stratum (AS) integration, ii) the *user* plane entity (UPE), acting

as a gateway, mobility anchor and over-the-air (OTA) security provisioner. Lastly, iii) the *network intelligence* (NI) plane is for the extraction of actionable insights from big data, orchestration or required services and functionalities (e.g., traffic optimization, caching, etc.). The realization of the network cloud can be achieved by enabling virtual function instances to be hosted in data centers when needed. The use of virtualization techniques will enable quick deployment and scalability of CPE and UPE functions. For example, in case of a natural disaster, with this technology the local data center maybe unable to cope with the traffic upsurge, therefore, additional capacity can be sourced quickly from other data centers. A different strategy is employed in [7]. In that paper, the EPC architectural proposal simply abstracts the EPC network functions, decomposing and allowing them to run as software instances (virtual machines), on standard servers. This allows service providers to customize services and policies to design networks in new ways, to reduce costs and simplify operations.

The aforementioned architecture proposals differ from one another. SDN is integrated with NFV in [6] to provide network control and to host the network intelligence, and in [7] only the NFV platform is available for enabling network services provision. Moreover, the architecture proposed in [7] is commercially available. Their contributions towards energy efficiency is as follows: (i) both make use of NFV and cloud computing platforms, and this avail the possibility of dynamically scaling resources based on demand as presented in [6]; (ii) through the information centric approach (collection of user-centric, network-centric and context-centric data), intelligent algorithms, mainly network optimization tools, can be applied to the aggregated data in order to provide useful outlook for network planning and resource management.

Following the ETSI MEC reference architecture in [41], a MEC NFV-based architecture is proposed. Here, new APIs are opened, availing hosting environments for both mobile operators and external players, which can make use of the access network related information for their services. This architecture consists of the *infrastructure* plane, the *control application* plane and the *management* plane. In addition, there is an *orchestration and management* plane hosting MEC management activities. The hosting environment consists of hardware resources, a virtualization infrastructure (virtual computation, storage, and network resources) and a set of associated management services for MEC applications. Within the COMBO project [42], a new functional architecture is proposed, where a new element called the universal access gateway (UAG) is introduced. The EPC gateways, serving gateway (SGW) and packet data network gateway (PGW), are virtualized and moved into the UAG. They are located in a central

office closer to end-users so that they can access the national IP network and reach the Internet sooner, thus enhancing latency and saving transport resources. By placing the mobile gateways closer to end-users, all traffic that does not need specific treatment is delivered locally to the remote cloud. A proper functional integration is of great importance to offer virtual resources at the network edge, while effectively adapting to the actual network load.

## 2.2 Energy Efficient Network Management Procedures

We first discuss literature review related to BS sleep modes techniques in subsection 2.2.1, followed by literature review related to energy savings in virtualized computing platforms (i.e., works related to autoscaling of servers or VMs in computing platforms) in subsection 2.2.2. Finally, in subsection 2.2.3, we review the mathematical tools that we use in this work for forecasting and foresighted optimization.

### 2.2.1 Sleep-modes strategies in MNs

The dramatic growth in mobile data has spurred the dense deployment of small cell BSs to enhance spectrum efficiency and increase network capacity. Although small cell BSs consumes less power compared to macro BSs, the overall power consumption of a large number of small cell BSs is phenomenal. Two large trends appear in the literature to address the energy efficiency challenge: (i) search for more energy efficient transmission devices and technologies; (ii) new technology proposals aimed at improving energy consumption in BSs, such as sleep mode, as it is known that switching on/off the BS transmission power during low traffic demand can yield significant energy savings [43][44][45]. In this work, we consider the second (ii) trend as we focus on the dynamic switching on/off of BS(s) and trend (i) is outside the scope of this work. Therefore, we only consider the literature related to BS sleep mode strategies, from a single operator or multi-operators perspective, highlighting the main applied techniques that are related to our research work.

In order to minimize the energy consumption in MNs, some trade-offs in the optimization problem are expected where performance metrics are also introduced. The Quality of Experience (QoE) is included as a trade-off metric in [46], where a dynamic programming switching algorithm is put forward. In addition, the coverage probability and the BS state stability parameter, i.e., the number of on/sleep state transitions, is considered. For instance, a set of BSs switching patterns engineered to provide full

network coverage at all times, while avoiding channel outage, is presented in [47]. The QoE is also affected by the UE position due to channel propagation phenomena. To this respect, in [48] the selection of the BS to be switched off is taken so as to minimize the impact on the UEs' QoE, according to the distance from the handed off BS. Moreover, QoS has been widely used as a trade-off metric [49][50]. A trade-off between delay cost and power consumption is present in [51]. Here, an online algorithm based on the Lyapunov optimization technique, mainly the drift-plus-penalty [52], is proposed for the joint workload offloading decisions and the BS sleeping decisions within an ultra-dense MN where each BS is co-located with a MEC server. The optimal BS activation and offloading strategy, at each time slot, is obtained in an iterative manner (heuristics) using the Gibbs sampling technique [53], in each distributed BS node. The obtained performance achieves a close-to-optimal as future information is not considered.

When a BS is switched off, the whole traffic load is allocated to the neighboring active BS(s) in which orthogonal resource allocation helps mitigate interference. To support sleep modes, neighboring BSs must be capable of serving the offloaded traffic. To achieve this, proper user association strategies are required. A user association mechanism that maximize energy efficiency in the presence of sleep modes are addressed in [45]. Here, a downlink HetNet scenario is considered, where the EE is defined as the ratio between the network throughput and the total energy consumption. Since this leads to a rather complex integer optimization problem, the authors propose a Quantum particle swarm optimization algorithm to obtain a sub-optimal solution. In [54], a framework to characterize the performance (outage probability and spectral efficiency) of cellular systems with sleeping techniques and user association rules is presented. In that paper, the authors devise a user association scheme where a user selects its serving BS considering the maximum expected channel access probability. This strategy is compared against the traditional maximum SINR-based user association approach and is found superior in terms of spectral efficiency when the traffic load is in-homogeneous.

The use of clustering algorithms have been proposed as a way of switching off BSs to reduce the energy consumption within MNs. A centralized and distributed algorithms group BSs exhibiting similar traffic profiles over time is presented in [55]. Then, in [56], a dynamic switching on/off mechanism that locally groups BSs into clusters based on location and traffic load is proposed. The optimization problem is formulated as a non-cooperative game aiming at minimizing the BS energy consumption and the time required to serve their traffic load. Simulation results show energy costs and load reductions, while also providing insights of when and how the cluster-based coordi-

nation is beneficial. However, the dynamic BS switching on/off strategies, towards energy savings where clustering is adopted, may have an impact on the network due to the traffic load that is offloaded to the neighboring BSs. To avoid this, the BS to be switched off must be carefully identified within a BS cluster. In [57], the *network impact* is used to identify the BS to be switched off within a cluster, one at a time, with no significant network performance degradation.

A marketing approach to promote the opportunistic utilization of the unexploited small cell BS capacity in ultra-dense heterogeneous networks (HetNets) is presented in [43]. There, an offloading mechanism is proposed, where the operators lease the capacity of a small cell network owned by a third party in order to switch off their BSs (Macro BSs) and maximize their energy efficiency, when the traffic demand is low. The allocation of the small cell resources among a set of competing operators is mathematically formulated as an auction problem. Then, authors in [44], present a comprehensive energy management model employing a BS switching on/off mechanism, within a BS system powered by green energy. The model considers user mobility, different green energy harvesting rates, weather conditions, energy storage with self-discharge effect, and switching on/off frequency. The authors propose two algorithms: the first decides which BSs are to be active based on the minimum energy cost, i.e., the energy price per time period, while the second one determines the active BSs by first prioritizing the minimum power consumption of the system, and then the energy cost.

Towards the effort of reducing the energy consumption through BS sleep modes, it is observed in the literature that most of the existing works consider clusters of BSs from a *single* mobile operator perspective, where some functions of the BS can be switched off and then the remaining active BSs handle the upcoming traffic. A new mechanism is presented in [58] which exploits the coexistence of multiple BSs from different mobile operators in the same area. An intra-cell roaming-based infrastructure-sharing strategy is proposed, followed by a distributed game-theoretic switching-off scheme that takes into account the conflicts and interaction among the different operators. Then, the work of [59] investigate the energy and cost efficiency of multiple HetNets (i.e., each HetNet is composed of eNodeBs (eNBs) and small cell BSs from one operator) that share their infrastructure and also are able to switch off part of it. Here, a form of roaming-based sharing is also adopted, whereby the operator can roam its traffic to a rival operator during a predefined period of time and area. An energy efficient optimization problem is formulated and solved using a cooperative greedy heuristic algorithm. Regarding the cost efficiency, the cooperation and cost sharing decisions

among the operators are modeled using a Shapley Value based bankruptcy game.

To enable power transmission adaptation and load balancing across BSs, traffic load information (operator mobile traffic datasets) obtained from the EPC can be utilized to extract relevant demand patterns to design dynamic BS management mechanisms [60][61]. Optimization based on mobile traffic datasets, obtained from multiple network elements described in [61], will make it possible to minimize the amount of time it takes to steer traffic on a real time basis, thus provisioning the network resources for computing and communication. In [60], a greedy algorithm is used to estimate the energy savings, through dynamic BS operation, based on real cellular traffic traces and actual BS location, within an urban environment. Then, in [62] datasets usage show the potential of dynamically switching on/off BSs around a stadium (soccer field), as some of the BSs experience low traffic load during a large event.

In the effort of greening future MNs, BSs are expected to be powered by green energy sources. On the other hand, NFV technology is expected to improve energy consumption by enabling the BS sleep modes, i.e., scaling down the NFs during low traffic periods [35] or at low EB levels. The work of [63] models dynamic switching on/off strategy for a BS energized by only solar energy. It considers the case of two BSs and proposes a solution based on a robust Bayesian technique assuming complete information on the network traffic. Based on this work, it is observed that an integrated view of the networking and solar energy issues is lacking. Along the lines of MN softwarization, a distributed user association scheme that makes use of the SoftRAN concept for traffic load balancing via the RAN Controller (RANC) is proposed in [64]. Here, the user association algorithm runs on the RANC and the users report their downlink data rates via the associated BS to the RANC, where the traffic loads from individual users and BSs are measured. The algorithm enhance the network performance by reducing the average traffic delivery latency in BSs as well as to reduce the on-grid power consumption by optimizing the green energy usage.

Finally, an integrated view of networking (small cell BSs usage) and solar energy issues that are important for a cost-effective green energy network planning are presented in [65]. Here, the relationship between installing a solar harvesting system to power a BS and the energy management under varying demand is investigated. The authors present a solar installation planning model by explicitly modeling solar panels, batteries, inverters and charge controllers, as well as the cellular network demand and energy management. They found that the solar installation and the energy management of the BSs are so coupled that even the order in which these technologies are

introduced can have a major impact on the network cost and performance.

### 2.2.2 Energy savings in virtualized platforms through soft-scaling

To address 5G use cases in a more energy efficient way, visibility into power usage is required for developing energy management policies in virtualized computing platforms (i.e., MEC servers or data centers). In virtualized computing platforms, the energy consumption is related to the computing and communication processes. To minimize energy consumption within such environments, energy saving studies have involved autoscaling (the scaling up/down of servers/VMs [66][67][68][69]), VM migration [70] and soft resource scaling [71]. With the advent of NFV, it is expected that the NFV framework [35][72] will exploit the benefits of virtualization technologies to significantly reduce the energy consumption of large scale network infrastructures. In addition, the EE control framework [73] shall provide the control sequence and procedures for controlling and managing EE, within self-managed automated edge systems.

Algorithms for the dynamic switching on/off the servers have been proposed as a way of minimizing energy consumption in computing platforms. In [66], at the beginning of each time slot computing resources are provisioned depending on the expected server workloads via a reinforcement learning-based resource management algorithm, which learns on-the-fly the optimal policy for dynamic workload offloading and the autoscaling of servers. By adopting this approach, the number of turned on physical servers is minimized. However, server consolidation typically does not consider the contribution of network traffic exchange to the overall energy consumption of the computing infrastructure. Then, in [70], the CPU utilization thresholds are used to identify over-utilized servers. Hence, migration policies, enabled by the live VM migration method [36], are applied for moving the VM between physical nodes (servers). The VMs are only moved to hosts that will accept them without incurring high energy cost, i.e., without any increase in the CPU utilization. Subsequently, the idle servers are switched off.

Virtualization provides a promising approach for consolidating multiple online services onto few computing resources within a enterprise data center. By dynamically provisioning VMs, consolidating the workload, and switching servers on/off as needed, service providers can maintain the desired QoS while achieving higher server utilization and EE. In [26], a dynamic resource provisioning *framework* for a virtualized computing environment is presented and it is experimentally validated on a small server cluster that provides online services. The control objective is to maximize the profit gener-

ated by the server system through the reduction of power consumption and service-level agreement violations. A predictive control for dynamic resource allocation in enterprise data centers is presented in [74], to reduce resource over-provisioning (i.e., CPU allocation) for enterprise applications. In that paper, predictive and feedback control-based techniques were used to periodically determine the amount of resources needed for each server. The prediction algorithms were based on auto-regressive (AR) model, a combined of Analysis of variance (ANOVA) and AR model (denoted by ANOVA-AR), and multi-pulse (MP) mode. In addition, a set of CPU utilization traces were used within the optimization problem.

Along the same lines of VM soft-scaling, in [67], a traffic engineering-based adapative approach is presented with an aim of minimizing energy consumption induced by computing, communication and reconfiguration costs of virtualized clouds within a data center. An iterative method is used to obtain the energy savings within a server that transmit wireless to clients. Then, in [68], the computing-plus-communication is also considered towards a goal of saving energy through adaptive transmission rate for a Fog node (virtualized node). In [69], an automated server provisioning algorithm that aims to meet workload demand while minimizing energy consumption in data centers is presented. Here, energy-aware server provisioning is performed by taking into account trade-offs between cost, performance, and reliability. Lastly, in [71], a power management approach that exploits hardware power scaling, i.e., the dynamic power management strategies using Dynamic Voltage and Frequency Scaling (DVFS), and software-based methods, called *VirtualPower*, is presented. Here, due to the low power management benefits obtained from hardware scaling, a *soft resource scaling* mechanism is proposed whereby the scheduler shortens the maximum resource usage time for each VM, i.e., the time slice allocated for using the underlying physical resources.

Although the aforementioned references goal is to minimize the energy consumption within computing environments, none has considered the switching on/off the transmission (optical) drivers towards ESs within such environments except for the works of [21] and [22]. Both works show that having the least number of data transmission drivers (i.e., *fast* tunable lasers) can yield significant amount of ESs. The computing node (router) is assumed to be equipped with a set of tunable lasers working in parallel. In [21], a joint scheduling and routing algorithm for guaranteeing the QoS for packet transmission within Wavelength Division Multiplexing (WDM) network is presented towards energy and bandwidth efficiency by cutting down the number of tunable lasers and wavelengths required. Then, in [22], a cross-networks framework for EE improve-

ment from WDM wired networks to 5G wireless networks is proposed. This work extends the work of [21] by including the wireless component and then considering the transmitted bits per unit energy consumption in the WDM networks. Even though these works, [21][22], are not along the direction of MEC, but they propose the tuning of the transmission drivers as *one* of the ES strategies within the MN infrastructure. Thus, ESs within the MEC server can be *jointly* achieved by launching an optimal number of VMs for computing and transmission drivers for real-time data transfers.

In *summary*, the need for jointly managing networking and computing components will be even more evident in future MNs, which will extensively adopt virtualization techniques. To this aim, it is expected that future computing platforms will exploit different types of VMs, ranging from the ones devoted to classical computing tasks to elements running VNFs.

### 2.2.3 Pattern forecasting along with foresighted optimization

In the paradigm of supervisory control for managing MNs, online forecasting using ML techniques and the LLC method can yield the desired system behavior when taking into account the environmental inputs, i.e., BS traffic load, server workloads and energy to be harvested. Next, we briefly review the mathematical tools that we use in this research work, namely the LLC method [26][27][28] and LSTM neural networks [24] [25].

Control-theoretic algorithms and the LLC method have been used to obtain control actions that optimize the system behavior, forecast by a mathematical model, over a limited look-ahead prediction horizon. The LLC is conceptually similar to Model Predictive Control (MPC) [26][75]. It differs in the following ways: (i) LLC work in a discrete domain whereas MPC deals with systems operating in a continuous input and output domain; (ii) LLC problems must be solved quickly, given the dynamics of server/data center workload, whereas MPC problems are usually computationally expensive and suited for slow-changing processes.

An online supervisory control scheme based on LLC policies is proposed in [76]. Here, after the occurrence of an event, the next control action is determined by estimating the system behavior a few steps into the future using the currently available information as inputs. The control actions exploration is performed using a search tree assuming that the controller knows all future possible states of the process over the prediction horizon. In another paper [27], an online control framework for resource management in switching hybrid systems is proposed, where the system's control inputs are finite. The relevant parameters of the operating environment, e.g., workload

arrival, are estimated and then used by the system to forecast future behavior over a look-ahead horizon. From this, the controller optimizes the predicted system behavior following the specified QoS through the selection of the system controls. The work of [26] make use of the LLC to address a resource provision problem within virtualized environments. The optimization problem is posed as a profit maximization problem under uncertainty and the LLC formulation models the cost of control. To address the uncertainty over the workload arrival, the Kalman filter is used. As the workload varies within the enterprise system, a hierarchical LLC structure is proposed to achieve fast operation, wherein the control problem is decomposed into a set of smaller sub-problems and solved in a cooperative fashion by multiple controllers.

To model time-series datasets, the LSTM network is used as it is able to handle the long-term dependencies due to its inherent capability of storing past information and then recalling it. In [77], a distributed LSTM online method based on the particle filtering algorithm is presented with an aim of investigating the performance of online training of LSTM architectures in a distributed network of nodes. An LSTM based model for variable length data regression is proposed, and then put into a non-linear state-space form to train the model in an online fashion. Financial and real life datasets are used for performance evaluation, and it is observed that the distributed online approach yields the same results that are obtained in the centralized case, when considering the mean square errors as the performance measure.

The application of LSTM network is extended to include Intelligent Transport Systems (ITSs) in [78]. A new ITS edge analytics architecture that makes use of deep learning techniques that either runs on the mobile devices or on the intra-vehicle processors for data analytics is presented. A combination of LSTM networks and deep Convolutional Neural Networks (CNNs) is adopted, i.e., CNN-LSTM network, for path selection in autonomous vehicles, whereby the CNN is used for feature extraction, and then the extracted information is fed into LSTM networks for driving path selection.

Forecasting server workloads using LSTM network can be beneficial for dynamic resource scaling and power consumption in cloud computing datacenters. In [79], a forecasting model using the LSTM network for predicting future data center workloads is proposed, and then the results are fed into the *resource manager* for decision making, which either involves scaling up or down the computing resources (servers in this case). HyperText Transfer Protocol (HTTP) traces were used for evaluating the algorithm and from the mean square error results it is observed that the LSTM based forecasting model yield better results when compared with the methods from [80] and [81].

**Discussion**: The combination of the NFV framework [35][72] and EE control framework [73] can potentially deliver significant energy savings. Within the NFV framework, the hypervisor can utilize power management policies defined by the network management entity. To optimize power management schemes, the policies and algorithms must take into account the processor frequencies and running applications. On the other hand, the EE control framework will facilitate coordinated actions for maximizing system wide energy efficiency gains in self-managed automated energy efficiency control processes. It make use of the following key functions: (i) EE Policy Management (it translates the policy information into configurations at the EE optimization entities within the network), (ii) EE Control and Coordination (ability to control and coordinate the power saving operations across all the relevant elements in the network, equipment and site levels) and (iii) EE Profiles Management (monitors, collects, processes, stores and provides EE related information and statistics). In addition, the use of forecasting (e.g., LSTM), control-theoretic algorithms and foresighted optimization can deliver significant energy savings MNs.

## 2.3 Renewable energy in MEC

Recently, renewable energy sources such as solar and wind energy have emerged as viable and promising energy sources for various Information and Communication Technology (ICT) systems due to the advancement of EH techniques. This is important, especially in light of the dense deployment pattern that is foreseen in 5G systems. The work of [82] observed that solar energy is more suitable for workloads with high peak-to-mean ratio (PMR) and wind energy fits better for workloads with small PMR. This avails the development of proper strategies for renewable energy provisioning for edge servers with the objective of eliminating any chance of energy shortage. This can be achieved by selecting the appropriate renewable energy source at each time instance taking into account current and forecast traffic loads. Since MEC servers are small-scale data centers (or micro data centers), each of which consumes less energy than conventional cloud data centers, it is expected that powering the MEC infrastructure with green energy sources will reduce the overall network energy consumption.

Along the lines of green networks within the MEC paradigm, authors in [83] proposed a framework for jointly performing load balancing, admission control and energy purchase among a network of EH-powered BSs with the goal of minimizing the com-

putation delay and data traffic drops (i.e., increasing the locally computed workloads). To solve this problem, an online algorithm (distributed) is proposed leveraging the Lyapunov optimization with perturbation technique. The algorithm makes decisions using only current information as inputs. In [84], a new EE design principle for the BS (co-located with the MEC server) to minimize its energy consumption, while ensuring self-sustainable computation at the mobile devices (through wireless power transfer (WPT)), is investigated using the Lagrangian duality method. A multi-user MEC system consisting of a multi-antenna access point and multiple users is assumed. Each mobile device is equipped with two antennas, one for WPT and the other for computational offloading. The antennas operate over different frequency bands such that WPT and computational offloading can be performed simultaneously, without mutual interference. Users rely on their harvested wireless energy to execute the latency-sensistive computational tasks either via local computing or (possibly partial) offloading to the MEC server. The optimal policy under energy harvesting constraints is obtained leveraging the Lagrange duality [85] and the ellipsoid methods [86].

A Lyapunov optimization technique [52] based on channel state indicator and energy level in EB is used to obtain dynamic offloading policies for EH powered mobile devices in [87]. In [66], the challenge of incorporating renewables into MEC is investigated. Here, a joint offloading and edge server provisioning problem is formulated as a Markov decision process (MDP). To overcome the *curse of dimensionality* in MDP when the state space is large [88], a post-decision state based learning algorithm is proposed.

**Discussion**: In view of the significant carbon footprint of grid power as well as the increasing electricity prices, renewable energy harvested from wind and/or solar radiation is embraced as a major or even sole power supply for edge systems, thanks to the recent advancements of energy harvesting techniques [89]. To address the energy consumption in 5G MNs in a more energy efficient way, within the MEC paradigm and green-aware networks, visibility into power usage is required for developing power management policies in virtualized computing platforms and communication systems. Special attention is required when using green energy together with sleep modes in BSs empowered with computing capabilities due to the unpredictable nature of green energy. To fully utilize the harvested energy, it is desirable to incorporate the green energy utilization as a performance metric in traffic load balancing strategies [64][83], instead of using the traffic load (e.g., network impact [57]). This work advocates for

the integration of EH systems in edge systems and the use of forecasting, in order to make energy management decisions in a forward-looking fashion.

## 2.4 Mobile traffic datasets analysis

Understanding data traffic demands and user behavior is crucial to the evaluation of strategies addressing the problem of high bandwidth usage and scalability of network resources, with improvements in the offered services. Based on the mobile traffic demands per user and user pattern, mobile operators can timely plan network resource allocation and set better subscription plans. For many MN services, the traffic load demand exhibits a diurnal behavior [90], thus it suffices to forecast the short-term traffic loads in order to capture the daily behavior of subscribers.

From networking perspective, the understanding and characterization of traffic consumption within the network can pave the way towards more efficient and user-oriented networking solutions. This can be achieved through the use of historical mobile traffic traces (called *Call Detail Records (CDRs)*) obtained from mobile operators, specifically in the EPC network. The availability of such open source traces has triggered research efforts from industrialists and academicians, who are actively seeking for networking solutions, targeting the conception, design and management of 5G networks. Adequate solutions are envisioned to considerably improve the overall performance of the network at lowest possible costs.

In [91], anonymized datasets generated by smartphone subscribers (i.e., they are collected at the core network of a 3G network) are used to characterize and model real mobile data traffic demands. The motivation of using smartphones generated datasets comes from the fact that they are mainly used for data compared to voice calls. From the usage patterns analyses, identical patterns on different days of the week were observed. Then, the authors proposed a traffic generator model that reproduces real traffic demands, and benchmarked it with a sample test set and statistical tools to observe its performance. The outcome is a synthetic measurement-based mobile data traffic generator capable of imitating traffic-related activity patterns. The work of [92] characterize the usage of YouTube in MNs using a large dataset containing HTTP requests collected from a mobile operator. Specifically, they consider video content and then study how video content popularity trends can be classified in order to provide an insight into video access patterns of users. Based on their analysis, it is recommended that caching be adopted to improve network efficiency, as content

requests can be locally handled.

During crowded events (e.g., concerts, soccer games), MNs face voice and data traffic volumes that are often orders of magnitude higher than what they face during normal days. To address this, as large/special events are known in advance, mobile operators deploy portable BSs for temporary increasing network capacity and free Wi-Fi access points for offloading traffic from cellular networks. Despite of this, large events still present significant challenges for mobile operators looking to reduce dropped call events, energy consumption and improve Internet speeds. In [62], traffic traces are used to characterize large events/venues and then providing some insights about the challenges involved in the planning, design and deployment of wireless networks. When considering the eNBs covering the large venue, it is observed that the traffic usage patterns is non-uniform over time. This avails the opportunity of dynamically adjusting the radio power and code allocation. The work of [93] investigates network performance degradation in MNs during crowded events using real-world voice and data traces collected from one mobile operator. Based on their results, it is suggested that radio resource allocation tuning and opportunistic connection sharing (the aggregating of traffic from multiple devices into a single cellular connection) can improve the network performance without incurring any costs related to infrastructure changes, as large events are known beforehand. Then, in [94], user content consumption habits are analyzed, as a function of time and place, to determine digital consumption hot spots in the network. Here, datasets containing Internet data exchange (traffic traces) between the mobile device and the wireless network are used, considering a normal day temporal traffic traces and one special day (final game for a soccer tournament). From their analysis, it is noted that content consumption must be taken into account in conjunction with mobility patterns in order to distribute content services closer to the users (i.e., caching contents closer).

If we consider traffic load variation during the day within a industrial zone, we can observe that during the night not all BSs are required in order to accommodate the minor traffic demand remaining in the area. Accordingly, a BS power control strategy, with switch-off possibility, over the set of BSs surrounding the industrial zone, would be very beneficial from an energy consumption point of view. Towards this goal, mobile data traffic constitutes a very valuable source of information in this context. In [60], the use of mobile datasets towards the development of a dynamic BS management mechanism and understanding of the technical challenges that arise in implementing it is explored. Here, real cellular traffic traces (voice traffic) and BS location information

is employed to observe users pattern for one week. Based on their analysis, they find that during weekdays about 30 percent of the time the traffic is less than 10 percent of the peak and during weekends and holidays, this low activity period increases to 43 percent of the time.

**Discussion**: The use of open source datasets or trace-driven simulations can speed-up the development of traffic-oriented network management solutions. In addition, they can bring mobile networking research efforts a step closer to real-world systems, at a time in which it takes longer to develop networking solutions, from their early research stage, to their actual implementations, in industry. By providing a realistic evaluation environment that captures real-world usage patterns, it allows to assess possible real-world energy savings resulting from such load-adaptive network management strategies. However, due to the non-availability of traffic load traces from mobile operators, researchers rely on open source datasets with limited information to try and address some of the MN challenges[1]. The combination of harvested energy[2] and mobile traffic datasets can help provide insights about opportunities for green-aware dynamic resource allocation within communication sites, i.e., BSs empowered with computation capabilities. This thesis is along this direction. In our work, we focus on power control strategies over BSs and resource allocation in computing platforms using the knowledge learned from mobile traffic load and energy traces.

[1]Mobile traffic datasets are mainly available through open (mobile) data challenges initiated by mobile operators, e.g., the Data for Development (D4D) challenge by Orange and the Telecom Italia Mobile (TIM) Big Data Challenge. To preserve confidentiality and security of mobile users information, some of the dataset information is hashed/anonymized.

[2]This type of datasets (solar energy, wind energy) are readily available as open source and they contain significant relevant information.

# Chapter 3
# Single BS Site Optimization

The results of this chapter are the subject of the following published paper:
**T. Dlamini**, A. F. Gambin, D. Munaretto, M. Rossi, "Online Resource Management in Energy Harvesting BS Sites through Prediction and Soft-Scaling of Computing Resources", in *Proceedings of the 2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, Bologna, Italy, September 2018.

## 3.1 Introduction

The wireless and mobile telecommunication networks plays a significant role in improving the global economy and the way people share information and knowledge, due to their capacity to provide radio coverage over a wide geographic area. This, in particular, is practical for remote/rural telecommunication applications where, through the installation of BS, the development of the wireless and mobile telecommunication networks can be achieved. However, continuous operation cannot be achieved in remote/rural areas as the electricity might be unreliable or even very costly to extend the electricity grid connection to such areas for the provisioning of communication services. Over the years, the alternative has been the use of diesel generators to power remote/rural BS sites. Despite of such, the idea of using diesel generators as a primary or back-up power supply has become less favorable due to the challenges related to reliability, availability, high operational and maintenance costs, and their significant environmental impacts [95][96]. Hence, methods using renewables coupled with sustainable energy storage solutions are now receiving more attention than before (e.g.,

the LG Uplus project[1]).

Future MNs are expected to leverage the integration of MEC and EH BS, i.e., BSs equipped with EH equipments, towards service provision and energy self-sufficiency in remote/rural areas. Providing reliable and stable grid power supply in remote areas and hazardous location can be costly and even infeasible since construction and operation of transmission lines are often prohibitive, and grid-tied servers can violate environmental quality regulations in areas that are ecologically sensitive [97]. This motivates the need for energizing edge systems with green energy in order to help extend network coverage to remote areas, minimize the carbon footprint and their dependence on the power grid [2][23]. The green-powered BSs (also empowered with computation capabilities) are more suitable in national parks, when considering environmental impact, as most governments do not approve cable wiring in such areas. However, the integration of MEC and EH BS systems brings about new challenges related to energy consumption, and resource scheduling. In addition, bringing computing and storage services on the BS for offloading some workload requires special attention, as resources are limited at the edge.

In literature, energy savings in BSs have been studied to minimize the BS power consumption by enabling sleep modes at low traffic load periods [57][60][62]. For instance, if a BS has not harvested sufficient energy, its transmission power can be tuned to be in proportion to the energy in its local energy storage and, for low traffic load periods, some of the BS functions can be deactivated. ESs within the virtualized computing platform (i.e., MEC server) are also of great importance. It is known that the power drawn by the server consists of an *idle* component and a *dynamic* component, which is the power consumed by the physical resources when working on behalf of some VMs. In [18], it is shown that power consumption increases with a growth in the number of virtual entities (e.g., VMs) that are allocated to the physical core, and in [19], it is further experimentally shown that increasing the number of VMs also increases power consumption in virtualized platforms, when taking into account the CPU usage only. From the obtained results, [18][19], the authors observed that the locus of energy consumption for component of VNFs is the VM instance where the VNF is instantiated/executed. Thus, the computing power demand depends on the number of VMs as well as the locally computed workload.

In this work, focusing on the BS and MEC server energy consumption, we pro-

[1]This project involves the deployment of a solar-powered long-term evolution (LTE) BS and it is found here: `http://www.koreatimes.co.kr/www/news/tech/2016/06/133_207882.html`

pose online-based energy management procedures that employ a forecasting/prediction method (i.e., LSTM neural network) that is used with foresighted optimization which is achieved through the use of control theory techniques (i.e., the LLC principles) and heuristics. Here, we forecast the short-term traffic load and harvested energy, and then use the results in the online algorithm for edge system management. Within the communication site, two energy saving strategies to mitigate the power consumption are used and they are: VM soft-scaling and BS *sleep modes*. The proposed optimization strategy reads to a considerable reduction in the energy consumed by edge system for computing and communication activities, enabling mobile services to off-grid sites under limited energy budget and predicted traffic loads.

### 3.1.1 Related work

Control-theoretic and ML methods for resource management at the edge have been successfully applied to various problems, e.g., task scheduling, bandwidth allocation, network management policies, etc. The works of [27] and [76] study online resource management procedures, where control-theoretic methods are used. A generic online control framework for resource management in switching hybrid systems is presented in [27], where the system's control inputs are finite. Then, in [76], a supervisory online control scheme based on LLC policies is presented. Both works estimate the environmental input, e.g., workload arrival, that is used by the system to forecast future system behavior over a given lookahead horizon. From this, the controller optimizes the predicted system behavior following the specified QoS through the selection of the system control inputs. The work of [26] uses the LLC to solve a resource allocation within a data center that is offering online services and the problem is posed as a one sequential optimization. A Kalman filter is used to estimate workload arrival rate. The authors in [66] presents a reinforcement learning-based resource management algorithm to incorporate renewable energy into a MEC platform. At the beginning of the time slot the servers are consolidated, i.e., the number of turned on physical servers are minimized, using the learned optimal policy for dynamic workload offloading and the autoscaling (or right-sizing). Our work differs from [66], as we minimize the number of active VMs instead of server consolidation, and also we use a forecasting method instead of only relying on the available current information for decision making. Moreover, this work differs from the aforementioned works related to forecasting as we use a LSTM neural network for forecasting traffic load and harvested (solar and wind) energy.

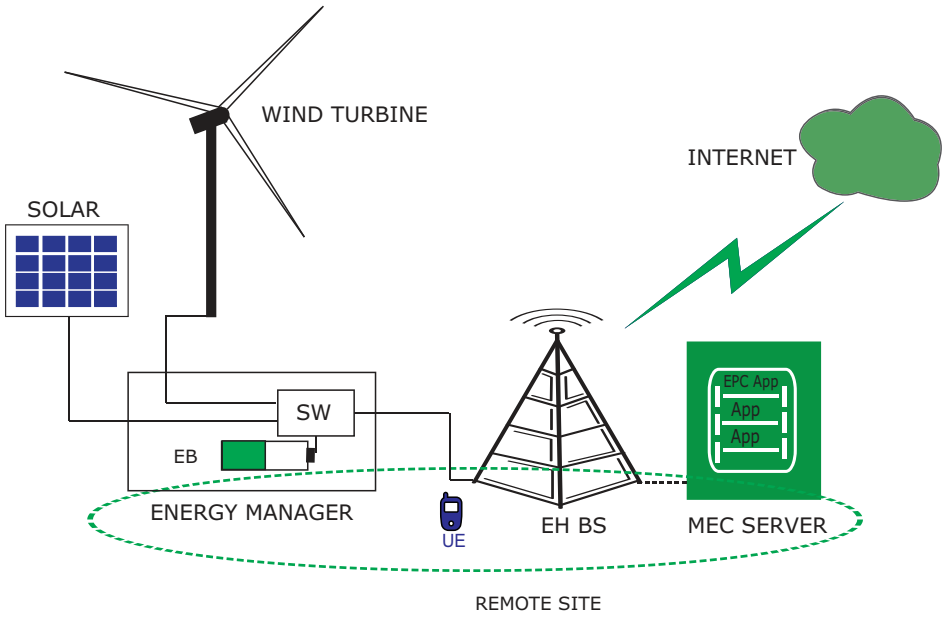



Figure 3.1: EH BS co-located with a MEC server. The electromechanical switch (SW) is responsible for aggregating the energy sources to fulfill the energy required to power the BS site.

### 3.1.2 Objectives and Contributions

The main contributions of this work are as follows:

1) we estimate the short-term future traffic load and harvested energy in BSs, by using the LSTM neural network [25], coupled with forecasting knowledge from [24];

2) We develop an online algorithm for edge network management based on control theory. The main goal is to enable ES strategies within the remote site through a joint consideration of BS sleep modes and VM soft-scaling, following the energy efficiency requirements of a virtualized infrastructure from [35]. The proposed management algorithm is called Energy Aware and Adaptive Management (ENAAM) and is hosted in the MEC server, i.e., ENAAM application. The ENAAM application considers the future BS loads, onsite green energy in the EB and then provisions edge network resources based on the learned information.

## 3.2 System Model

The considered deployment scenario is illustrated in Fig. 3.1. As a major deployment method of MEC suggested by ETSI [12], we consider a setup where a BS is co-located

with a virtualized MEC server, forming a communication site *termed* remote site. Both share the available energy stored in the EB. The solar modules and wind turbine combine their effort to power the edge system. Following the motivation from the introduction, we only consider an *off-grid* BS co-located with the MEC server. We assume that uplink and downlink transmissions operate on orthogonal channels and focus on the downlink traffic since energy consumption associated with downlink transmissions predominates the total BS energy consumption [98]. The MEC server accounts for $M$ virtual machines as total computation resources, and it is cache-enabled, i.e., some contents can be accumulated locally. The EM is responsible for selecting the appropriate renewable energy source to fulfill the EB depending on the weather, and for monitoring the energy levels in the system. Moreover, we consider a discrete-time model, whereby time is discretized as $t = 1, 2, \ldots$, and time slots have a constant duration $\tau$. For data communication from the remote site to the remote cloud, the system uses a microwave backhaul, as fast roll-out over large distances makes microwave an ideal rural/remote backhaul solution [99]. The list of the symbols that are used in this chapter is reported in Table 3.1.

### 3.2.1 Traffic Load and Energy Consumption

Traffic load traces have been obtained using real MN data from the Big Data Challenge organized by Telecom Italia Mobile (TIM) [100]. The open source dataset is a result of users interaction within the TIM MN for the city of Milan during the month of November 2013, whereby each interaction generates a CDR file. The considered TIM dataset refers to standard traffic such as Short Message Service (SMS), Calls and Internet browsing, and they are not yet a representative of future applications that require processing at the edge[2].

In this work, according to [92], we assume that 80% of the traffic from this dataset requires processing at the edge, whereas the remaining 20% pertains to standard, delay tolerant, flows[3]. The daily traffic load profile requiring computation at the BS, $\mathrm{L}(t)$, *see* blue curve in Fig. 3.2, is obtained by accounting for 80% of the aggregated CDR data. The normalized BS load at time slot $t$ is approximated as $\varphi(t) = \mathrm{L}(t)/\mathrm{L}_{\max}$, where $\mathrm{L}_{\max}$ represents the maximum load that can be served. Among this load, $\gamma(t) \in [0, 1]$

[2]This datasets are used due to the difficulties in obtaining relevant open source datasets containing computing requests.

[3]This assumption is based on the Pareto principle (also known as the 80-20 rule), a phenomenon that can be observed in many real-life situation.

Table 3.1: Notation: list of symbols used in the analysis.

| **Symbol** | **Description** |
|---|---|
| **Input Parameters** | |
| $M$ | maximum number of VM hosted in the MEC server, indexed by $m$ |
| $\tau$ | time slot duration |
| $L(t)$ | BS traffic load profile in time slot $t$ |
| $L_{\max}$ | maximum traffic load that can be served |
| $\Gamma(t)$ | standard (non MEC) traffic at time $t$ |
| $\varphi(t)$ | normalized BS traffic load at $t$ |
| $f$ | operating frequency |
| $\theta_0$ | BS load independent energy consumption or operation energy |
| $\theta_{\text{bh}}$ | microwave backhaul transmission energy |
| $\beta_{\max}$ | maximum energy buffer capacity |
| $\beta_{\text{up}}, \beta_{\text{low}}$ | upper and lower energy buffer thresholds |
| **Variables** | |
| $\gamma(t)$ | locally processed load (computation workload) at time $t$ |
| $\theta(\zeta, \gamma, t)$ | total energy consumption of the remote site at time $t$ |
| $\theta_{\text{tx}}(t)$ | total downlink BS transmission power at time $t$ |
| $\theta_{\text{bs}}(t)$ | BS energy consumption due to communication at time $t$ |
| $\theta_{\text{mec}}(\gamma, t)$ | server consumption due to computation activities at time $t$ |
| $\zeta(t)$ | BS switching status indicator at $t$ |
| $I(t)$ | number of VM to be active in time slot $t$ |
| $\beta(t)$ | energy buffer level in slot $t$ |
| $H(t)$ | harvested energy profile in slot $t$ |
| $Q(t)$ | purchased grid energy in slot $t$ |

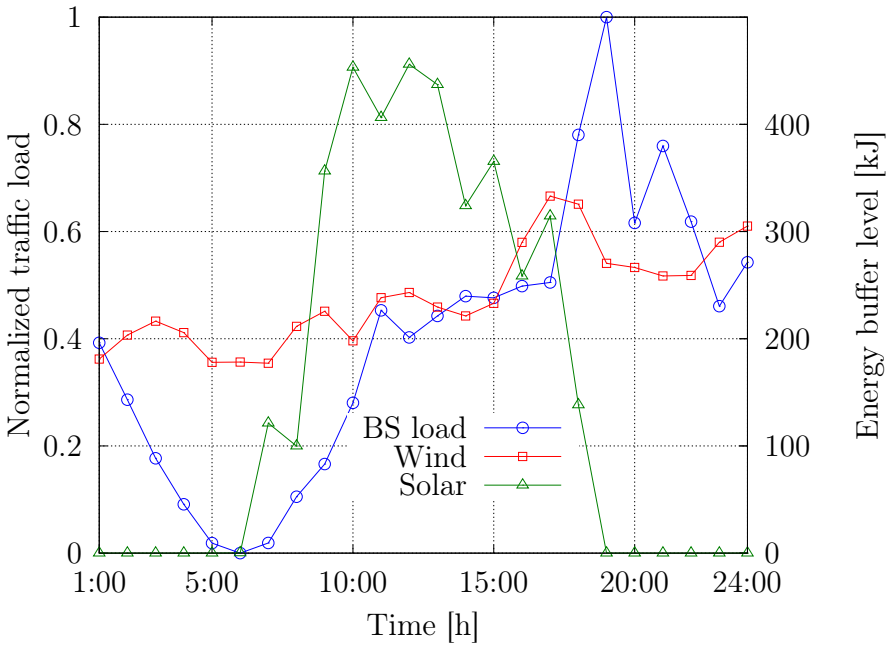



Figure 3.2: Example traces for harvested solar and wind energy, and normalized traffic load in the BS.

is processed locally and the rest $\Gamma(t) = \varphi(t) - \gamma(t) \in [0, 1]$ is handled by the remote cloud. Moreover, a low traffic threshold $L_{\text{low}}$ is defined to be used in the ENAAM algorithm. Note that $\gamma(t)$ is a decision variable, as we shall see, $\gamma(t)$ is ideally set to 1 for those time slots where the BS has enough energy, i.e., all the delay sensitive traffic is processed at the edge. $\gamma(t)$ will be set to a smaller value otherwise.

The total energy consumption ([J]) of the remote site is here obtained as the combination of the energy consumed by the BS and by the co-located MEC server operating at a frequency $f$ ([Hz]), with server maximum utilization factor $\gamma(t)$ in time slot $t$. The following model is inspired by [37] and [101], by additionally tuning the BS static energy and the server utilization factor, to scale the server dynamic energy consumption in proportion to the expected load to be processed locally:

$$\theta(\zeta, \gamma, t) = \theta_{\text{bs}}(t) + \theta_{\text{mec}}(\gamma, t)\,, \tag{3.1}$$

where $\theta_{\text{bs}}(t)$ is the BS energy consumption due to communication activities and $\theta_{\text{mec}}(\gamma, t)$ is the computation energy at the server due to computing processes.

### BS energy consumption

It is defined as: $\theta_{\text{bs}}(t) = \zeta(t)\theta_0 + \theta_{\text{tx}}(t) + \theta_{\text{bh}}$, where $\zeta(t) \in \{\varepsilon, 1\}$ is the BS switching status indicator (1 for *active mode* and $\varepsilon$ for *power saving mode*), $\theta_0$ is a constant

value (load independent), representing the operation energy which includes baseband processing, radio frequency power expenditures, etc. The constant $\varepsilon \in (0, 1)$ accounts for the fact that the baseband energy consumption can be scaled down as well whenever there is no or little channel activity, into a power saving mode. $\theta_{\text{tx}}(t)$ represents the total downlink transmission (load dependent) power from the BS to the served user(s). Since we assume a noise-limited channel and the guarantee of low latency requirements at the edge, to obtain $\theta_{\text{tx}}(t)$ we use the downlink transmission model in [51]. $\theta_{\text{bh}}$ is the microwave backhaul transmission energy, which is here assumed to be constant.

**MEC server energy consumption**

It is defined as follows: $\theta_{\text{mec}}(\gamma, t) = \theta_{\text{idle}} + \gamma(t)\theta_{\text{dyn}}(t)$, where $\theta_{\text{idle}}$ is the server load-independent operational component, and $\theta_{\text{dyn}}(\cdot)$ is the maximum energy amount that is consumed by the server when it operates at full power. Although omitted for the sake of notation compactness, $\theta_{\text{idle}}$ and $\theta_{\text{dyn}}(\cdot)$ depend on the MEC server computation frequency $f$. Also, $\theta_{\text{dyn}}(\cdot)$ is linearly scaled with respect to the load $\gamma(t)$, assuming that computation resources can be tuned. Finally, the number of virtual machines that shall be active in time slot $t$ to serve the offered load is here obtained as $\text{I}(t) = \texttt{round}(\gamma(t)M)$, where $\texttt{round}(\cdot)$ rounds the argument to the nearest integer.

### 3.2.2 Energy Patterns and Storage

The energy buffer of Fig. 3.1 is characterized by its maximum energy storage capacity $\beta_{\max}$, and power charging/discharging and leaking losses are not assumed. At the *beginning* of each time slot $t$, the EM provides the energy level report to the authorized MEC server application, thus the EB level $\beta(t)$ is known, enabling the provision of the required computation resources, i.e., VMs. Here, a pull transfer mode (e.g., FTP [30]) is assumed, where the MEC application pulls the energy report from the EM.

The amount of harvested energy $H(t)$ in time slot $t$ for the remote site is obtained from open source solar traces within a solar panel farm located in Armenia [102] (*see* green curve in Fig. 3.2), and also wind traces within a wind farm located in Belgium [103] (*see* red curve in Fig. 3.2). The data is aggregated to match our time slot duration (1 h). The datasets are a result of daily environmental records, considering solar panel orientation, measured and forecast wind speed, temperature, wind power, and pressure values. The solar panels and wind turbines are assumed to be placed in an area free from surrounding obstructions (e.g., buildings, shades). In this work, $H(t)$

is obtained by first scaling the datasets to fit the EB capacity $\beta_{\max}$ of 490 kJ, and then selecting the wind energy as a source during the solar energy off-peak periods in order to achieve steady operation. Thus, the available EB level $\beta(t+1)$ for the *off-grid* BS in time slot $t+1$ is calculated as follows:

$$\beta(t+1) = \beta(t) + H(t) - \theta(\zeta, \gamma, t) \tag{3.2}$$

where $\beta(t)$ is the energy level in the battery at the beginning of time slot $t$ and $\theta(\zeta, \gamma, t)$ is the energy that is used during the time slot for computation and communications processes, *see* Eq. (3.1). For decision making in the MEC server, a lower battery threshold is defined, $\beta_{\text{low}}$, with $0 < \beta_{\text{low}} < \beta_{\max}$, to steer how the energy management algorithm provisions the required edge network resources. Our optimization in Section 3.4 makes sure that $\beta(t)$ never falls below $\beta_{\text{low}}(t)$ low due to its transmission and computing activities within a time slot.

## 3.3 Problem Formulation

In this section, we formulate an optimization problem and it is defined in subsection 3.3.1.

### 3.3.1 Optimization Problem

As green energy budget is unpredictable and hence unknown at the beginning of time slot $t$, the edge system uses a forward-looking strategy towards workload offload and autoscaling computing resources decisions. In this work, we aim at minimizing the overall energy consumption in the remote site over time, i.e., consumption related to the BS and MEC server, by applying BS power saving modes and VM soft-scaling, i.e., tuning the number of active virtual machines. To achieve this, we define two cost functions: F1) $\theta(\zeta, \gamma, t)$, which weighs the energy consumption due to transmission (BS) and computation (MEC server); and F2) a quadratic term $(\varphi(t) - \gamma(t))^2$, which accounts for the QoS cost. In fact, F1 tends to push the system towards energy efficient solutions, i.e., where $\gamma(t) \to 0$ and $\zeta(t) \to \varepsilon$. Instead, F2 favors solutions where the load is entirely processed by the local MEC server, i.e., where $\gamma(t) \to \varphi(t)$. A weight $\alpha \in [0, 1]$, is utilized to balance the two objectives F1 and F2. The corresponding weighted cost function is defined as:

$$J(\zeta, \gamma, t) \triangleq \overline{\alpha}\theta(\zeta(t), \gamma(t), t) + \alpha(\varphi(t) - \gamma(t))^2\,, \tag{3.3}$$

where $\overline{\alpha} \triangleq 1 - \alpha$. Hence, over time horizon, $t = 1, \ldots, T$, the following optimization problem is defined:

$$\begin{aligned}
\textbf{P1} \quad &: \quad \min_{\boldsymbol{\zeta}, \boldsymbol{\gamma}} \sum_{t=1}^{T} J(\zeta, \gamma, t) \\
&\text{subject to:} \\
\text{C1} \quad &: \quad 0 < \gamma(t) \leq 1, \quad t = 1, \ldots, T \\
\text{C2} \quad &: \quad \zeta(t) \in \{\varepsilon, 1\}, \quad t = 1, \ldots, T \\
\text{C3} \quad &: \quad \mathrm{I}(t) \geq b, \quad t = 1, \ldots, T \\
\text{C4} \quad &: \quad \beta(t) \geq \beta_{\text{low}}, \quad t = 1, \ldots, T
\end{aligned} \tag{3.4}$$

where vectors $\boldsymbol{\zeta}$ (switching status) and $\boldsymbol{\gamma}$ (utilization factor) contain the control actions for the considered time horizon $1, 2, \ldots, T$, i.e., $\boldsymbol{\zeta} = [\zeta(1), \zeta(2), \ldots, \zeta(T)]$ and $\boldsymbol{\gamma} = [\gamma(1), \gamma(2), \ldots, \gamma(T)]$. Constraint C1 specifies the server utilization factor bounds, C2 specifies the BS operation status, C3 forces the required number of VMs, $\mathrm{I}(t)$, to be always greater than or equal to a minimum number $b \geq 1$: the purpose of this is to be always able to handle mission critical communications. C4 makes sure that the EB level is always above or equal to a preset threshold $\beta_{\text{low}}$, to guarantee energy self-sustainability over time.

To solve P1 in Eq. (3.4), we leverage the use of LLC [27][76] and heuristics. Once P1 is solved, the control action to be applied in the edge system at time $t$ is $\varsigma(t) \triangleq (\zeta(t), \gamma(t))$.

## 3.4 Remote Site Management

In this section, a traffic load and energy harvesting prediction method, and an online management algorithm are proposed to solve the previously stated problem P1. In subsection 3.4.1, we discuss the ML tool used to predict the short-term future traffic loads and harvested energy, and then in subsection 3.4.2, we solve P1 by first constructing the state-space behavior of the control system, where online control key concepts are introduced. Finally, the algorithm for managing the remote site is presented in subsection 3.4.3.

### 3.4.1 Traffic Load and Energy Prediction

Future MNs are expected to learn the diverse characteristics of users behavior, as well as renewable energy source(s) variations, in order to autonomously and dynamically

Table 3.2: LSTM Prediction Model Steps

| **Modeling steps** |
| --- |
| Step 1: load and normalize the dataset |
| Step 2: split dataset into training and testing |
| Step 3: reshape input to be [samples, time steps, features] |
| Step 4: create and fit the LSTM network |
| Step 5: make predictions |
| Step 6: calculate performance measure |

determine the desired system configurations. The network elements are expected to rely on sophisticated learning and decision making procedures, for an efficient network management. The use of ML techniques constitute a promising solution for network management and ESs in MNs [104][105]. In this work, we consider a time slot duration of 1 h and perform time series prediction, i.e., we obtain the 1 h-ahead estimates of $\hat{\mathrm{L}}(t+1)$ and $\hat{H}(t+1)$, by using an LSTM network developed in Python using Keras deep learning libraries (Sequential, Dense, LSTM) where the network has a visible layer with 1 input, a hidden layer of 4 LSTM blocks or neurons, and an output layer that makes a single value prediction. This type of recurrent neural network uses back-propagation through time and memory blocks for regression [25]. The dataset is split as 67% for training and 33% for testing. The network is trained using 100 epochs (2,600 individual training trials) with batch size of 1. As for the performance measure of the model, we use the Root Mean Square Error (RMSE). The prediction steps are outlined in Table 3.2, and Figs. 3.3 and 3.4 show the prediction results that will be discussed in subsection 3.5.2.

### 3.4.2 Edge System Dynamics

Here, we propose an online control technique where the actions governing system operation are obtained by optimizing its forecasted behavior, described by a mathematical

model, for the specified QoS criteria over a limited lookahead prediction horizon. The control action that drives the system towards the desired behavior is chosen from a finite discrete set, i.e., the control input is selected from a sequence of feasible control actions, per time instance. As a specific case study, we apply our control method to manage the energy consumption within the remote site, taking into account the forecasted environmental input, i.e., traffic load and harvested energy, over the lookahead horizon.

We denote the system state vector at time $t$ by $x(t) = (\mathrm{I}(t), \beta(t))$, which contains the number of active VMs, and the EB level. $\varsigma(t) = (\zeta(t), \gamma(t))$ is the input vector, i.e., the control action that drives the system behavior at time $t$. The system evolution is described through a discrete-time state-space equation, adopting the LLC principles [27][76]:

$$x(t+1) = \Phi(x(t), \varsigma(t)) \,, \tag{3.5}$$

where $\Phi(\cdot)$ is a behavioral model that captures the relationship between the current system $x(t)$, the control input $\varsigma(t)$, and the next state $x(t+1)$. Note that this relationship accounts for:

1) the amount of energy drained $\theta(\zeta, \gamma, t)$ and the harvested $H(t)$, which together lead to the next buffer level $\beta(t+1)$ through Eq. (3.2).

2) The traffic load $L(t)$, from which we compute the offered load $\varphi(t)$, that together with the control $\gamma(t)$ leads to $\mathrm{I}(t+1)$ (once a control policy is specified).

The remote site management algorithm (ENAAM application), acts as a *controller* that finds the best control action vector to the system in an iterative manner. For each time slot $t$, the best control action $\varsigma^*(t)$ is the one minimizing the weighted sum $J(\zeta, \gamma, t)$. This control action amounts to setting the BS radio mode $\zeta^*(t)$, i.e., either active or power saving, and the number of instantiated VMs, $\mathrm{I}^*(t)$, which directly follows from $\gamma^*(t)$.

An observation is in order. State $x(t)$ and control $\varsigma(t)$ are respectively measured and applied at the beginning of time slot $t$, whereas the offered load $L(t)$ and the harvested energy $H(t)$ are accumulated during the time slot and their value becomes known only by the end of it. This means that, being at the beginning of time slot $t$, the system state at the next time slot $t+1$ can only be estimated, which we formally write as:

$$\hat{x}(t+1) = \Phi(x(t), \varsigma(t)) \,. \tag{3.6}$$

For these estimations we use the forecast values of load $\hat{L}(t)$ and harvested energy $\hat{H}(t)$, from the LSTM forecasting module. The controller optimizes the forecast behavior as per the specified QoS requirements by selecting the best control inputs to apply to the system. The key ideas behind the controller decision making is discussed next.

**Controller decision-making**: The controller is obtained by estimating the relevant parameters of the operating environment, that is, in our case its the BS load $\hat{L}(t)$ and the harvested energy $\hat{H}(t)$, and subsequently using them to forecast the future system behavior through Eq. (3.6) over a look-ahead time horizon of $T$ time slots. The control actions are picked by minimizing $J(\zeta, \gamma, t)$, see Eq. (3.3). At the beginning of each time slot $t$ the following process is iterated:

- Future system states, $\hat{x}(t+k)$, for a prediction horizon of $k = 1, \ldots, T$ steps are estimated using Eq. (3.3). These predictions depend on past inputs and outputs up to time $t$, on the estimated load $\hat{L}(\cdot)$ and energy harvesting $\hat{H}(\cdot)$ processes, and on the control $\varsigma(t+k)$, with $k = 0, \ldots, T-1$.
- The sequence of controls $\{\varsigma(t+k)\}_{k=0}^{T-1}$ is obtained for each step of the prediction horizon by optimizing the weighted cost function $J(\cdot)$.
- The control $\varsigma^*(t)$ corresponding to the first control action in the sequence with the minimum total cost is the applied control for time $t$ and the other controls $\varsigma^*(t+k)$ with $k = 1, \ldots, T-1$ are discarded.
- At the beginning of the next time slot $t+1$, the system state $x(t+1)$ becomes known and the previous steps are repeated.

*Finally*, since the control actions are taken after exploring only a limited prediction horizon, yielding a limited number of possible operating states, we must ensure the stability of the edge system. For this, we rely on the notion that a system is said to be stable under control, if for any state, it is always possible to find a control input that forces it closer to the desired state or within a specified neighborhood of it [27]. This implies that the controller can eventually achieve the desired QoS goal. In addition, we further assume that since both the lookahead horizon and the number of control inputs is small, the computational overhead is negligible, and this has been confirmed through experimental validation in [28].

**Algorithm 1:** ENergy Aware and Adaptive Management (ENAAM)

**Input:** $x(t)$ (current state)

**Output:** $\varsigma^*(t) = (\zeta^*(t), \gamma^*(t))$

01: Initialization of variables

$\mathcal{S}(t) = \{x(t)\}$, $\text{Cost}(x(t)) = 0$

02: **for** $k = 1, \ldots, T$ **do**

- forecast the traffic load $\hat{\text{L}}(t+k-1)$

- forecast the harvested energy $\hat{\text{H}}(t+k-1)$

- $\mathcal{S}(t+k) = \emptyset$

03: **for all** $x \in \mathcal{S}(t+k-1)$ **do**

04: **for all** $\varsigma = (\zeta, \gamma) \in \mathcal{A}(t+k-1)$ **do**

05: $\hat{x}(t+k) = \Phi(x(t+k-1), \varsigma)$

06: $\text{Cost}(\hat{x}(t+k)) = J(\zeta, \gamma, t+k-1) + \text{Cost}(x(t+k-1), \varsigma)$

07: $\mathcal{S}(t+k) = \mathcal{S}(t+k) \cup \{\hat{x}(t+k)\}$

**end for**

**end for**

**end for**

08: **Find** $\hat{x}_{\min} = \text{argmin}_{\hat{x} \in \mathcal{S}(t+T)} \text{Cost}(\hat{x})$

09: $\varsigma^*(t) :=$ control leading from $x(t)$ to $\hat{x}_{\min}$

10: **Return** $\varsigma^*(t)$

### 3.4.3 The ENAAM Algorithm

Let $t$ be the current time. $\hat{\text{L}}(t+k-1)$ is the forecast load in slot $t+k-1$, with $k = 1, \ldots, T$, i.e., over the prediction horizon. For the control to be feasible, we have $\hat{\varphi}(t+k-1) = \hat{\text{L}}(t+k-1)/L_{\max}$ and $\underline{\gamma} \leq \gamma \leq \hat{\varphi}(t+k-1)$, where $\underline{\gamma}$ is the smallest $\gamma$ such that $\texttt{round}(\underline{\gamma}M) = b$. For the buffer state, we heuristically set $\zeta(t+k-1) = \varepsilon$ if $\beta(t+k-1) < \beta_{\text{low}}$ *or* $L(t+k-1) < L_{\text{low}}$, and $\zeta(t+k-1) = 1$ otherwise ($\beta_{\text{low}}$ and $L_{\text{low}}$ are preset thresholds). For slot $t+k-1$, the feasibility set $\mathcal{A}(t+k-1)$ contains

the control pairs $(\zeta, \gamma)$ that obey these relations.

The algorithm is specified in algorithm 1 as it uses the technique in [27]: the search starts (line 01) from the system state at time $t$, $x(t)$, and continues in a breadth-first fashion, building a tree of all possible future states up to the prediction depth $T$. A cost is initialized to zero (line 01) and is accumulated as the algorithm travels through the tree (line 06), accounting for predictions, past outputs and controls. The set of states reached at every prediction depth $t + k$ is referred to as $\mathcal{S}(t + k)$. For every prediction depth $t + k$, the search continues from the set of states $\mathcal{S}(t + k - 1)$ reached at the previous step $t+k-1$ (line 03), exploring all feasible controls (line 04), obtaining the next system state from Eq. (5.10) (line 05), updating the accumulated cost as the result of the previous accumulated cost, plus the cost associated with the current step (line 06), and updating the set of states reached at step $t + k$ (line 07). When the exploration finishes, the action at time $t$ that leads to the best final accumulated cost, at time $t + T$, is selected as the optimal control $\varsigma^*(t)$ (lines 08, 09, 10). Finally, for line 04, we note that $\gamma$ belongs to the continuous set $[\underline{\gamma}, \hat{\varphi}(t + k - 1)]$. To implement this search, we quantized this interval into a number of equally spaced points, obtaining a search over a finite set of controls.

## 3.5 Performance Evaluation

In this section, we show some selected numerical results for the scenario of Section 3.2. The parameters that were used for the simulations are listed in Table 3.3.

### 3.5.1 Simulation setup

We consider a single offgrid BS co-located with a MEC server within a coverage area of 40 m. In addition, we use a virtualized server with specifications from [106] for a VMware ESXi 5.1-ProLiant DL380 Gen8 that operates at $f = 1,600$ MHz. Our time slot duration is set to 1 h and the time horizon to $T = 2$ time slots.

### 3.5.2 Numerical results

Some example prediction results are shown in Figs. 3.3 and 3.4 for the traffic load and harvested energy, respectively, reaching an RMSE performance of 0.42 MB. Quite good accuracies are also obtained for the prediction of the harvested energy (RMSE of

Table 3.3: System Parameters.

| **Parameter** | **Value** |
|---|---|
| Low traffic threshold, $L_{\text{low}}$ | 4 MB |
| Maximum load, $L_{\text{max}}$ | 15 MB |
| Operating power, $\theta_0$ | 10.6 W |
| Microwave backhaul power, $\theta_{\text{bh}}$ | 50 W |
| Maximum number of VMs, $M$ | 27 |
| Minimum number of VMs, $b$ | 3 |
| Idle power, $\theta_{\text{idle}}$ | 30 W |
| Dynamic maximum power, $\theta_{\text{dyn}}(t)$ | 472.3 W |
| Energy storage capacity, $\beta_{\text{max}}$ | 490 kJ |
| Lower energy threshold, $\beta_{\text{low}}$ | 30% of $\beta_{\text{max}}$ |

0.38 kJ over a range of about 450 kJ). The measured accuracy is deemed good enough for the proposed optimization.

Figs. 3.5 and 3.6 show the energy savings achieved over time for $\alpha = 0$ and $\alpha = 0.5$ respectively, when on-demand and energy-aware edge resource provisioning is enabled (i.e., BS sleep modes and VM soft-scaling), in comparison with the case where they are not applied. Our remote site management algorithm (ENAAM) is benchmarked with another one that heuristically selects the amount of traffic that is to be processed locally, $\gamma(t)$, depending on the expected load behavior. It is named Dynamic and Energy-Traffic-Aware algorithm with Random behavior (DETA-R). Both ENAAM and DETA-R are aware of the predictions in the future time slots, see subsection 5.4.1, however, DETA-R provisions edge resources using a heuristic scheme. DETA-R heuristic works as follows: if the expected load difference is $\hat{\text{L}}(t+1) - \hat{\text{L}}(t) > 0$, then $\gamma(t)$ in the current time slot $t$ is randomly selected in the range $[0.6, 1]$, otherwise, it is picked evenly at random in the range $(0, 0.6)$.

In Fig. 3.6 ($\alpha = 0.5$), the ENAAM scheme achieves ES of about 56% and DETA-R of 29% on average. As expected, this shows a reduction in energy savings compared to when $\alpha = 0$. This is due to the balance between the emphasis on energy savings and

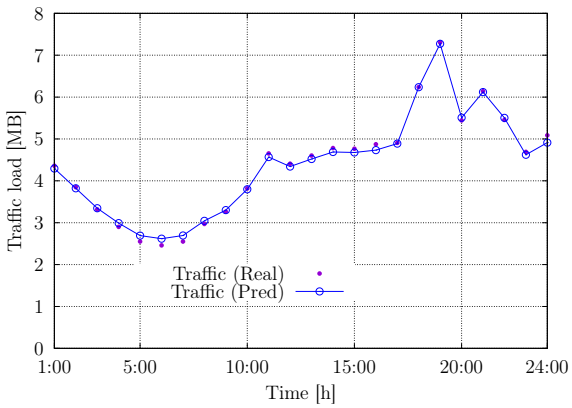



Figure 3.3: One-step ahead predicted BS load

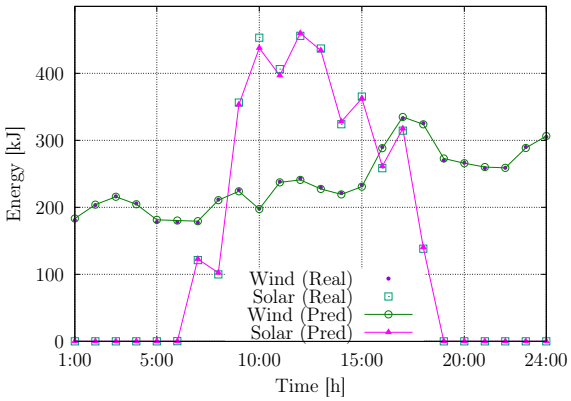



Figure 3.4: One-step ahead predicted energy

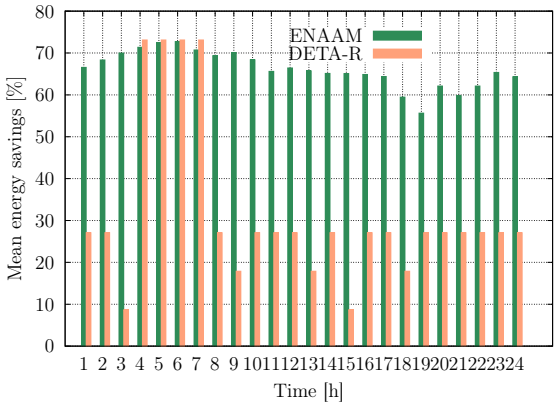



Figure 3.5: Hourly energy savings for $\alpha = 0$.

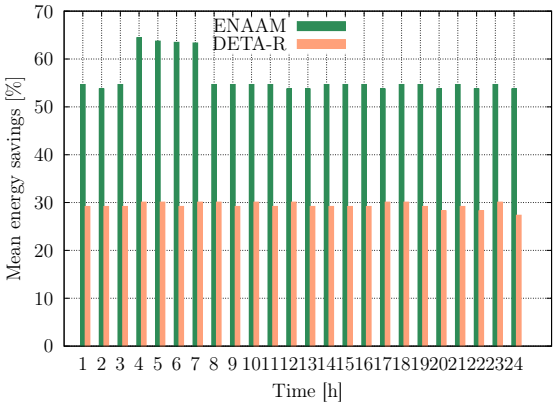



Figure 3.6: Hourly energy savings for $\alpha = 0.5$.

QoS, i.e., locally computed tasks, within the remote site. The evolution of ES with respect to $\alpha$ is presented in Fig. 3.7. As expected, a drop in energy savings is observed when QoS is prioritized, i.e., $\alpha \to 1$, as in this case the BS energy consumption is no longer considered.

Finally, Fig. 3.8 shows the MEC server utilization over time, i.e., the selected control $\gamma(t)$. For $\alpha = 0.5$, the server utilization is about 76% for ENAAM and 95% for DETA-R on average. A low server utilization can be observed in the performance of ENAAM between 4 h and 7 h due to an expected low traffic load in the system. This indicates that ENAAM has load adaptation capabilities, which are much desirable and lead to substantial energy savings (see again Fig. 3.6 between 4 h and 7 h).

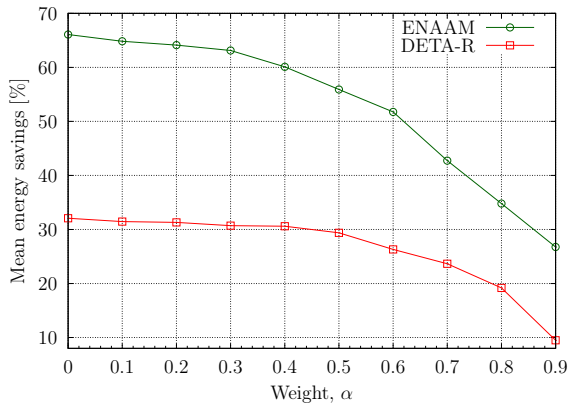



Figure 3.7: Energy savings *vs* the optimization weight $\alpha$.

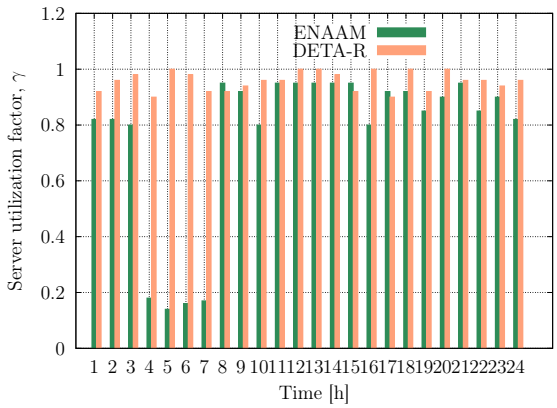



Figure 3.8: MEC server utilization ($\alpha = 0.5$)

## 3.6 Conclusion

In summary, we have envisioned a renewable-powered remote site for extending network coverage and promoting energy self-sustainability within mobile networks. The BS at the remote site is endowed with computation capabilities for guaranteeing low latency to mobile users, offloading their workloads. The combination of the energy saving methods, namely, BS sleep modes and VM soft-scaling, for the remote site helps to reduce its energy consumption. An edge energy management algorithm based on forecasting, control theory and heuristics, is proposed with the objective of saving energy within the remote base station, possibly making the BS system self-sustainable. Numerical results, obtained with real-world energy and traffic traces, demonstrate that the proposed algorithm achieves energy savings between 56% and 66% on average, with respect to the case where no energy management techniques are applied, and to hold the server utilization between 30% and 96% over time, with an average of 75%.

# Chapter 4
# Multiple BS Sites Optimization

The results of this chapter are the subject of the following published paper:
**T. Dlamini**, A. F. Gambin, D. Munaretto, M. Rossi, "Online Supervisory Control and Resource Management for Energy Harvesting BS Sites Empowered with Computation Capabilities", *Journal on Wireless Communications and Mobile Computing, vol. 2019*, February 2019.

The contents of this chapter partly extend the following published paper:
**T. Dlamini**, A. F. Gambin, D. Munaretto, M. Rossi, "Online Resource Management in Energy Harvesting BS Sites through Prediction and Soft-Scaling of Computing Resources", in *Proceedings of the 2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, Bologna, Italy, September 2018.

## 4.1 Introduction

The convergence of communication and computing (MEC [13]) within the mobile space poses new challenges related to energy consumption, as BSs are densely-deployed to maximize capacity, yet traffic varies during the day, and also empowered with computing capabilities to minimize latency. To cope with energy consumption challenges in BSs, previous studies have put forward BS sleep modes [43][44][45][60][62] as BSs are dimensioned for the expected maximum capacity, yet traffic varies during the day. In addition, energy savings within the virtualized computing platform are of great importance as virtualization can also lead to energy overheads. Therefore, a clear understanding and a precise modeling of the server energy usage can provide a funda-

mental basis for server operational optimizations. The experimental results in [18][19] show that the locus of energy consumption for the VNF components is the VM instance where the VNF is instantiated and executed. Thus, for a given expected traffic load, the energy consumption can be minimized by launching an optimal number of VMs (i.e., VM *soft-scaling*), together with BS power saving methods, i.e., BS sleep modes.

Along these lines, we propose a controller-based network architecture for managing EH BSs empowered with computation capabilities where on/off switching strategies allow BSs and VMs to be dynamically switched on/off, depending on the traffic load and the harvested energy forecast, over a given lookahead prediction horizon. To solve the energy consumption minimization problem in a distributed manner, the *controller* partitions the BSs into clusters based on their location, then for each cluster, it minimizes a cost function capturing the individual communication site energy consumption and the users' QoS. For communication sites management, the controller performs online supervisory control by forecasting the traffic load and the harvested energy using a LSTM neural network [25], which is utilized within a LLC policy (a predictive control approach [27]) to obtain the system control actions that yields the desired trade-off between energy consumption and QoS.

The proposed optimization strategy leads to a considerable reduction in the energy consumed by the edge computing and communication facilities, promoting self-sustainability within the MN through the use of green energy. This is achieved under the controller guidance, which makes use of forecasting, clustering, control theory and heuristics.

### 4.1.1 Related work

Following the emerging concept of green networking and computing, the realization of powering the BS sites (i.e., sites empowered with computing capabilities) using sustainable solutions has received significant attention, at the sametime the importance of energy saving procedures arises. Towards ES, the work of [51] proposed a joint computation offloading and BS sleeping decisions in order to maximize the QoS while keeping the energy consumption low. This work considers the dense deployment of BSs, within the MEC paradigm, and then employ the Lyapunov optimization technique (the drift-plus-penalty) towards network optimization. However, the consideration of autoscaling the computing resources, i.e., VM soft-scaling, is not taken into account. Then, towards traffic-oriented network solutions (in dense environments), real cellular traffic traces and actual BS location is used in the work of [60], where a greedy algorithm

estimate the energy savings through dynamic BS operation. In another paper, the usage of mobile datasets show the potential of dynamically switching on/off BSs around a stadium (soccer field), as some of the BSs experience low traffic load during a large event [62]. Both works, [60][62], use only current traffic load information with no forecasting and foresighted optimization consideration.

It is worth noting that the aforementioned works do not consider clustering algorithms towards minimizing the energy consumption within MNs. To this end, only the works of [55][56][57] have used clustering algorithms for switching off BSs. Despite of the efforts being made toward energy savings, the works are usually disjoint, i.e., it is either the clustering algorithm is used in densely-deployed BSs lacking computation capabilities or no clustering algorithm used, i.e., the BSs empowered with computation capabilities simply cooperate with their neighbors (see [51]). Therefore, a joint consideration of energy savings is essential in clustered EH BSs empowered with computation capabilities. In this chapter, we group BSs based on their location (or distance measures) similarity and then enable energy savings within the clustered BSs. The consideration of BSs that cooperate (no clustering case) towards energy saving is part of our future research work.

### 4.1.2 Objectives and Contributions

Our contributions is as follows:

1) we introduce the use of virtualization with the aim of investigating how VMs can be soft-scaled based on the forecasted server workloads, as VMs are the source of energy consumption in computing environments.

2) We put forward the edge controller-based architecture for small cell BSs management, as one of the future trends for small cells [16] in 5G MNs.

3) We reconsider the BS sleeping control mechanism under the new MEC paradigm, which has not been sufficiently covered in the literature. In addition, we use a clustering method for enabling energy savings within the MN.

4) We estimate the short-term future traffic load and harvested energy in BSs, by using LSTM neural network [24].

5) We develop an online supervisory control algorithm for the radio access (edge) network management based on a predictive method, specifically the LLC method,

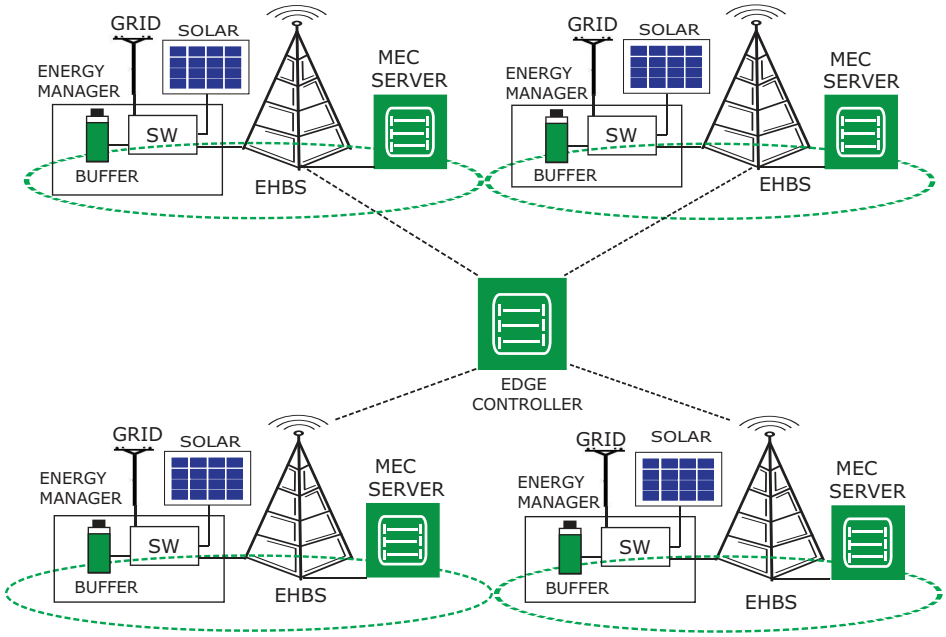



Figure 4.1: Edge network topology. The electromechanical switch (SW) aggregate the energy sources to fulfill the energy required to power the BS site.

along with clustering and energy management procedures. The main goal is to enable ES strategies within the access network, BS sleep modes and VM soft-scaling, following the energy efficiency requirements of a virtualized infrastructure from [35]. The proposed management algorithm is called ENergy Aware and Adaptive Management (ENAAM) and is hosted in the edge controller. The ENAAM algorithm considers the future BS traffic load, onsite green energy in the EB and then provisions access network resources, per communication site, based on the learned information, i.e., energy saving decisions are made in a forward-looking fashion.

## 4.2 System Model

As a major deployment of MEC suggested by ETSI [12] and in line with current trends for future mobile networks as suggested by prominent network operators (e.g., Huawei Technologies [16]), the considered scenario is illustrated in Fig. 3.1. It consists of a densely-deployed MN featuring $N$ BSs and co-located cache-enabled MEC servers. Each MEC server hosts $M$ VMs. Each communication site, i.e., the BS and the co-located MEC server, is empowered with EH capabilities through a solar panel and an EB that enables energy storage. Energy supply from the power grid is also

available as a back up. Moreover, the EM is an entity responsible for selecting the appropriate energy source and for monitoring the energy level of the EB. All BSs communicate with a centralized entity called the *edge controller*, which is responsible for managing the access network apparatuses. The energy level information is reported periodically to the edge controller through the pull file transfer mode procedure (e.g., FTP [30]). Moreover, we consider a discrete-time model, whereby time is discretized as $t = 1, 2, \ldots$, and each time slot $t$ has a fixed duration $\tau$. The list of symbols that are used in the chapter is reported in Table 4.1.

### 4.2.1 Traffic Load and Energy Consumption

Mobile traffic volume exhibits temporal and spatial diversity, and also follows a diurnal behavior [107]. Therefore, traffic volume at individual BSs can be estimated using historical mobile traffic datasets. In this work, real MN traffic load traces obtained from the Big Data Challenge organized by TIM [100] are used to emulate the computation workload[1]. Specifically, the used data was collected in the city of Milan during the month of November 2013, and it is the result of users interaction within the TIM MN, based on CDR files for a day considering four BS sites representing the traffic load profiles. A CDR file consists of SMS, Calls and Internet records with timestamps.

To understand the behavior of the mobile data, we have applied the X-means clustering algorithm [108] to classify the load profiles into several categories. In our numerical results, each BS $n = 1, 2, \ldots, N$ is assigned a load profile $L_n(t)$, which is picked at random as one of the four clusters (each cluster represents a typical BS load profile) in Fig. 4.2. $L_n(t)$ consists of computation workloads $\Gamma_n(t)$ ([MB]) and standard workloads $\Gamma'_n(t)$ ([MB]). According to [92], we assume that 80% of $L_n(t)$ is *delay sensitive* and, as such, requires processing at the edge, i.e., $\Gamma_n(t) = 0.8L_n(t)$, whereas the remaining 20% pertains to standard flows, *delay tolerant* traffic, i.e., $\Gamma'_n(t) = L_n(t) - \Gamma_n(t)$.

The total energy consumption ([J]) for the communication site $n$ at time slot $t$ is formulated as follows, inspired by [37], [51], [101] and [109]:

$$\theta_{\mathrm{tot},n}(t) = \theta_{\mathrm{BS},n}(t) + \theta_{\mathrm{MEC},n}(t) + \theta_{\mathrm{TX},n}(t)\,, \tag{4.1}$$

where $\theta_{\mathrm{BS},n}(t)$ is the BS energy consumption term, $\theta_{\mathrm{MEC},n}(t)$ is the MEC server consumption term due to computation activities, and $\theta_{\mathrm{TX},n}(t)$ represents the data trans-

[1]In fact, the dataset is not a *true* representative of future applications that require processing at the edge, but contains data that is exchanged with the purpose of communication. We nevertheless use it due to the difficulties in finding open datasets containing computing requests.

Table 4.1: Notation: list of symbols used in the analysis.

| Symbol | Description |
|---|---|
| **Input Parameters** | |
| $N$ | number of BSs, indexed by $n$ |
| $M$ | maximum number of VMs hosted by each MEC server |
| $L_n(t)$ | BS $n$ traffic load profile in time slot $t$ |
| $\Gamma_n(t)$ | workload handled by the MEC server at BS $n$ at time slot $t$ |
| $\Gamma'_n(t)$ | standard (non MEC) traffic at time slot $t$ |
| $\theta_0$ | BS load independent energy consumption |
| $f_{\max}$ | maximum processing rate for VM $m$ |
| $\theta_m^{\text{ov}}(t)$ | energy overheads incurred when turning on/off VMs at time slot $t$ |
| $\theta_{\text{idle},m}(t)$ | static energy consumed by VM $m$ in the idle state |
| $\theta_{\max,m}(t)$ | maximum energy consumed by VM $m$ at maximum rate |
| $\gamma_m(t)$ | workload fraction to be computed by the $m$-th VM at time slot $t$ |
| $\gamma^{\max}$ | maximum computation load per-VM |
| $\Delta$ | maximum per-slot and per-VM allowed processing time |
| $\beta_{\max}$ | maximum energy buffer capacity |
| $\beta_{\text{up}}, \beta_{\text{low}}$ | upper and lower energy buffer thresholds |
| **Variables** | |
| $\theta_{\text{tot},n}(t)$ | total energy consumption for the communication site $n$ |
| $\theta_{\text{BS},n}(t)$ | BS $n$ energy cost at time slot $t$ |
| $\theta_{\text{MEC},n}(t)$ | server consumption due to computation activities |
| $\theta_{\text{TX},n}(t)$ | data transmission cost between the BS and the MEC server at time slot $t$ |
| $\zeta_n(t)$ | BS $n$ switching status indicator at time slot $t$ |
| $M(t)$ | number of VMs to be active in time slot $t$ |
| $\theta_{\text{load}}(t)$ | total wireless transmission power at time slot $t$ |
| $f_m(t)$ | instantaneous processing rate |
| $\theta_m^{\text{op}}(t)$ | energy consumption of VM $m$ operation at time slot $t$ |
| $B_n(t)$ | total amount of load that is served by the BS site at time slot $t$ |
| $\beta_n(t)$ | energy buffer level in time slot $t$ |
| $H_n(t)$ | harvested energy profile in time slot $t$ |
| $Q_n(t)$ | purchased grid energy in time slot $t$ |

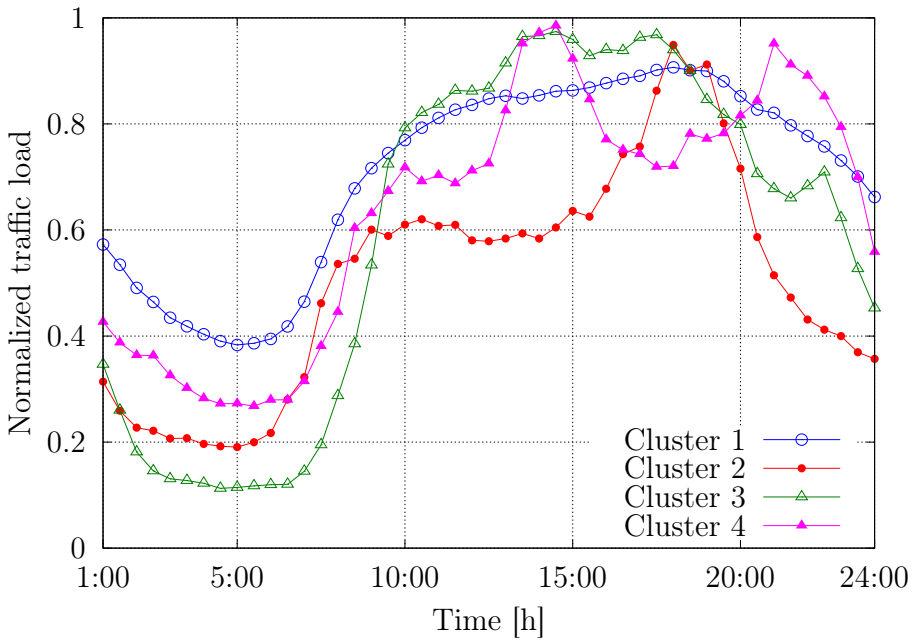



Figure 4.2: Example traces for normalized BS traffic loads. The used dataset has been split into four representative clusters.

mission energy consumption between the BS and the MEC server.

**BS energy consumption**: It is defined as: $\theta_{\mathrm{BS},n}(t) = \zeta_n(t)\theta_0 + \theta_{\mathrm{load}}(t)$, where $\zeta_n(t) \in \{\varepsilon, 1\}$ is the BS switching status indicator (1 for *active mode* and $\varepsilon$ for *power saving mode*), $\theta_0$ is a constant value (load independent), representing the operation energy which includes baseband processing, radio frequency power expenditures, etc. The constant $\varepsilon \in (0, 1)$ accounts for the fact that the baseband energy consumption can be scaled down as well whenever there is no or little channel activity, into a power saving mode. $\theta_{\mathrm{load}}(t)$ represents the total wireless transmission (load dependent) power to meet the target transmission rate from the BS to the served user(s) and to guarantee low latency at the edge. Since we assume a noise-limited channel and the guarantee of low latency requirements at the edge, $\theta_{\mathrm{load}}(t)$ is obtained by using the transmission model in [51]. Here, we neglect the imbalance of traffic volumes in uplink and downlink, and also we do not account for the switching energy cost for the BS mode transition [109] due to the fact that future BS functions will be virtualized [1].

**MEC server energy consumption**: It depends on the number of VMs running in time slot $t$, named $M(t) \leq M$, and on the CPU frequency that is allotted to each virtual machine. Specifically, VMs are instantiated on top of the physical CPU cores, and

each VM is given a share of the host server CPU, memory and network input/output interfaces. The CPU is the main consumer of energy in the server [70] due to the VM-to-CPU share mapping. Hence, in this work we focus on the CPU utilization only. With $f_m(t) \in [0, f_{\max}]$ we mean the instantaneous processing rate [110], expressed in bits per second that are computed, and $f_{\max}$ is the maximum processing rate for VM $m$. In this work, $f_m(t)$ is set within a finite set $\mathcal{F} = \{f_0, f_1, \ldots, f_{\max}\}$ where $f_0 = 0$ represents zero speed of the VM (e.g., deep sleep or shutdown). At any given time $t$, the total energy consumption of a virtualized server, with $M(t)$ running VMs, is

$$\theta_{\mathrm{MEC},n}(t) = \sum_{m=1}^{M(t)} \left(\theta_m^{\mathrm{op}}(t) + \theta_m^{\mathrm{ov}}(t)\right) , \tag{4.2}$$

where $\theta_m^{\mathrm{op}}(t)$ is the energy consumption of VM $m$ operation and $\theta_m^{\mathrm{ov}}(t) \geq 0$ is the energy cost incurred through the turning on/off the VM, i.e., $\theta_m^{\mathrm{ov}}(t) > 0$ only when VM $m$ is switched on/off and it is zero otherwise. $\theta_m^{\mathrm{op}}(t)$ is obtained using the linear relationship between the CPU utilization contributed by VM $m$ and the energy consumption, from [110] and [67]:

$$\theta_m^{\mathrm{op}}(t) = \theta_{\mathrm{idle},m}(t) + \alpha_m(t)(\theta_{\mathrm{max},m}(t) - \theta_{\mathrm{idle},m}(t)) , \tag{4.3}$$

where $\theta_{\mathrm{idle},m}(t)$ represents the *static* energy drained by VM $m$ in the idle state, and $\theta_{\mathrm{max},m}(t)$ is the *maximum* energy it drains. The quantity, $\alpha_m(t)(\theta_{\mathrm{max},m}(t) - \theta_{\mathrm{idle},m}(t))$, represents the *dynamic* energy component, where $\alpha_m(t) = (f_m(t)/f_{\max})^2$ [27] is a load dependent factor. Note that $\alpha_m(t)$ and $f_m(t)$ are deterministically related as $f_{\max}$ is a constant. $\theta_m^{\mathrm{ov}}(t)$ is obtained from [67] as a constant and is typically limited to a few hundreds of mJ per $\mathrm{MHz}^2$.

Conventionally, for each BS site, the hypervisor, i.e., the software that provides the environment in which the VMs operate, is in charge of allocating $f_m(t)$ and the workload fraction to be computed by the $m$-th VM, named $\gamma_m(t)$. In our setup, we have $\sum_{m=1}^{M(t)} \gamma_m(t) \leq \Gamma_n(t)$, where equality is achieved when the workload is fully served by the $M(t)$ VMs. We also note that, in practical application scenarios, the maximum per-VM computation load to be computed is generally limited up to an assigned value, named $\gamma^{\max}$. Motivated by the energy efficient requirements from [35], i.e., the hypervisor's ability to accept and implement policies from a management entity, in this research work, the *edge controller* usage is pursued. Here, the edge controller determines the $f_m(t)$ value that will yield the desired or expected processing time, $\mu_m(t) = \gamma_m(t)/f_m(t)$, considering the workload $\gamma_m(t)$ allotted to VM $m$. $\mu_m(t)$ must

be less than or equal to the maximum per-slot and per-VM processing time (in seconds), named $\Delta$, i.e., $\mu_m(t) \leq \Delta$. Note that $\Delta$ is also the server's response time, i.e., the maximum time allowed for processing the total computation load.

We remark that, as a result of the allocation procedure that is developed in this work, for any BS site $n$, the processing rates $f_m(t)$ shall be found, similar to [67], as $f_m(t) \triangleq \gamma_m(t)/\Delta$. Then, the total amount of load that is served by the BS site may be set as: $B_n(t) = \sum_{m=1}^{M(t)} \gamma_m(t) \leq \Gamma_n(t)$. The objective of the considered optimization is to find the operating mode for the BS (either "on" or "power saving"), the number of VMs $M(t)$ that are to be allocated and, for each of them, the processing rate $f_m(t)$. In doing so: 1) the amount of delay sensitive load that is not served at the edge, $\Gamma_n(t) - \sum_{m=1}^{M(t)} \gamma_m(t)$, shall be minimized, while exploiting as much as possible the energy harvested from the solar panels, so that the mobile network will be energetically self-sufficient, and 2) the load is computed in a time shorter than or equal to $\Delta$.

**Data transmission energy consumption**: We assume that the inter-communication between the BS and the MEC server is bi-directional and symmetric. Hence, under steady-state operating conditions, for the communication site $n$, $\theta_{\mathrm{TX},n}(t)$ is obtained as, by using the VM migration hint from [111],

$$\theta_{\mathrm{TX},n}(t) = \theta_{\mathrm{idle}}(t) + \theta_{\mathrm{data}}(t)\, B_n(t) \tag{4.4}$$

where $\theta_{\mathrm{idle}}(t)$ (fixed value in J) is the energy drained by the network interfaces in idle mode over a time slot $t$, $\theta_{\mathrm{data}}$ (fixed value in J/byte) is the cost of exchanging one byte of data between the MEC server and the BS per time slot $t$, and $B_n(t)$ is the amount of data exchanged. These parameters, $\theta_{\mathrm{idle}}(t)$ and $\theta_{\mathrm{data}}(t)$, are obtained from [111]. Note that $B_n(t)$ also corresponds to the amount of data to be processed at the MEC server in bytes.

### 4.2.2 Energy Patterns and Storage

The energy buffer is characterized by its maximum energy storage capacity $\beta_{\max}$. At the *beginning* of each time slot $t$, the EM provides the energy level report to the edge controller through the local MEC server, thus the EB level $\beta_n(t)$ is known, enabling the provision of the required computation resources, i.e., the VMs. The energy level report/file from the EM to the MEC server is transferred using the pull mode procedure (e.g., FTP) [30].

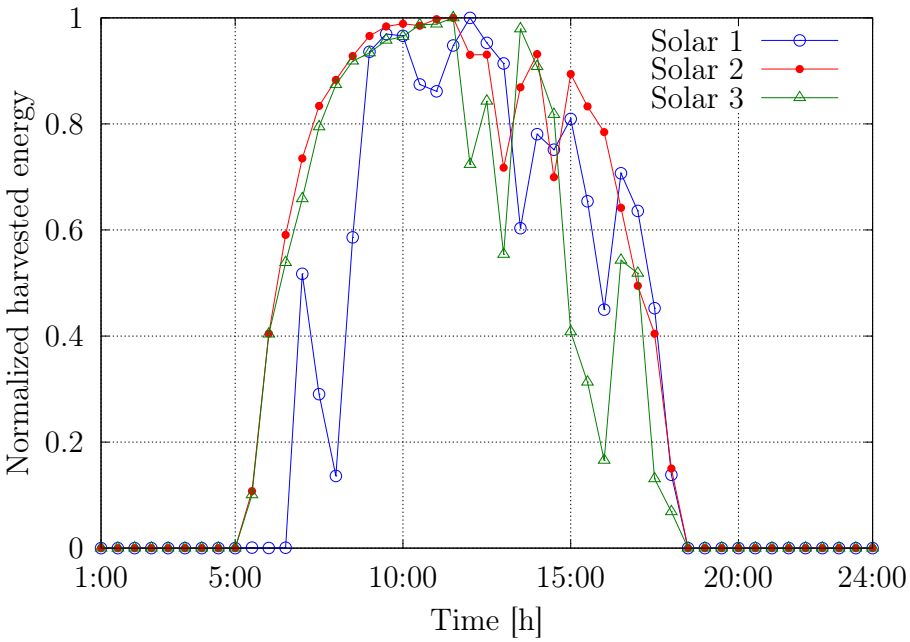



Figure 4.3: Example traces for harvested solar energy.

In this work, the amount of harvested energy $H_n(t)$ in time slot $t$ in the communication site $n$ is obtained from open source solar traces within a solar panel farm located in Armenia [102] (see Fig. 4.3). The dataset is the result of daily environmental records and the data is aggregated to match our time slot duration (30 min). In our numerical results, $H_n(t)$ represents a daily solar radiation record for three different areas. From the three solar profiles, each communication site energy profile is picked at a random to represent the daily energy harvested and then scaled to fit the EB capacity $\beta_{\max}$ of 490 kJ. Thus, the available EB level $\beta_n(t+1)$ at the beginning of time slot $t+1$ is calculated as follows:

$$\beta_n(t+1) = \beta_n(t) + H_n(t) - \theta_{\text{tot},n}(t) + Q_n(t), \tag{4.5}$$

where $\beta_n(t)$ is the energy level in the battery at the beginning of time slot $t$, $\theta_{\text{tot},n}(t)$ is the energy consumption of the communication site over time slot $t$, see Eq. (4.1), and $Q_n(t) \geq 0$ is the amount of energy purchased from the power grid. We remark that $\beta_n(t)$ is updated at the beginning of time slot $t$ whereas $H_n(t)$ and $\theta_{\text{tot},n}(t)$ are only known at the end of it.

For decision making in the edge controller, the received EB level reports are compared with the following thresholds: $\beta_{\text{low}}$ and $\beta_{\text{up}}$, respectively termed the lower and the upper energy threshold with $0 < \beta_{\text{low}} < \beta_{\text{up}} < \beta_{\max}$. $\beta_{\text{up}}$ corresponds to the desired energy buffer level at the BS and $\beta_{\text{low}}$ is the lowest EB level that any BS should ever

reach. The suitable energy source at each time slot $t$ is selected based on the forecast expectations, i.e., the expected harvested energy $\hat{H}(t)$. If $\hat{H}(t)$ is enough to reach $\beta_{\text{up}}$, no energy purchase is needed. Otherwise, the remaining amount up to $\beta_{\text{up}}$, i.e., $Q_n(t) = \beta_{\text{up}} - \beta_n(t)$ is bought from the electrical grid. Our optimization framework in following section makes sure that $\beta_n(t)$ never falls below $\beta_{\text{low}}$ and guarantees that $\beta_{\text{up}}$ is reached at every time slot.

## 4.3 Optimization for a Single Communication Site

In this section, we formulate an optimization problem to obtain *energy savings* through short-term traffic load, harvested energy predictions, along with energy management procedures for a *single* communication site. The optimization problem is defined in subsection 4.3.1, and the communication site management procedures are presented in subsection 4.3.2.

### 4.3.1 Problem Formulation

At the beginning of each time slot $t$, the edge controller receives the energy level report $\beta_n(t)$ from each EM (via the MEC application responsible for energy profiles in the MEC server), using the pull mode file transfer. Here, we aim at minimizing the overall energy consumption in the communication site over time, i.e., the consumption related to the BS transmission activity and the MEC server, by applying BS power saving modes and VM soft-scaling, i.e., tuning the number of active virtual machines. To achieve this, we first consider the optimization for a single communication site. We define two cost functions as:

**F1)** $\theta_{\text{tot},n}(t)$, which weighs the energy consumption due to transmission (BS) and computation (MEC server);

**F2)** a quadratic term $(\Gamma_n(t) - B_n(t))^2$, which accounts for the QoS cost.

In fact, F1 tends to push the system towards self-sustainability solutions, i.e., $\zeta_n(t) \to \varepsilon$. Instead, F2 favors solutions where the delay sensitive load is entirely processed by the local MEC server, i.e., $B_n(t) \to \Gamma_n(t)$. A weight $\eta \in [0, 1]$, is utilized to balance the two objectives F1 and F2. The corresponding (weighted) cost function is defined as:

$$J(\zeta, \alpha, t) \triangleq \overline{\eta}\theta_{\text{tot},n}(\zeta_n(t), \{\alpha_m(t)\}, t) + \eta(\Gamma_n(t) - B_n(t))^2\,, \tag{4.6}$$

where $\overline{\eta} \triangleq 1-\eta$, with $\{\alpha_m(t)\}$ we mean the sequence of factors $\alpha_1(1), \alpha_2(1), \ldots, \alpha_{M(t)}(1)$. Hence, letting 1 be the current time slot and $T$ be the time horizon, the following optimization problem is formulated over time slots $1, \ldots, T$:

$$\begin{aligned}
\mathbf{P1} \quad &: \quad \min_{\boldsymbol{\zeta}, \boldsymbol{\alpha}} \sum_{t=1}^{T} J(\zeta, \alpha, t) \qquad (4.7)\\
&\text{subject to:}\\
\text{C1} \quad &: \quad \zeta_n(t) \in \{\varepsilon, 1\},\\
\text{C2} \quad &: \quad b \leq M(t) \leq M,\\
\text{C3} \quad &: \quad \beta_n(t) \geq \beta_{\text{low}},\\
\text{C4} \quad &: \quad 0 \leq f_m(t) \leq f_{\max},\\
\text{C5} \quad &: \quad 0 \leq \gamma_m(t) \leq \gamma^{\max},\\
\text{C6} \quad &: \quad \mu_m(t) \leq \Delta, \quad t = 1, \ldots, T\,,
\end{aligned}$$

where $m = 1, \ldots, M(t)$ (VM index), vectors $\boldsymbol{\zeta}$ (BS switching status in time slots $1, \ldots, T$) and $\boldsymbol{\alpha}$ (load dependent factor) contain the **control actions** for the considered time horizon, per communication site, i.e., $\boldsymbol{\zeta} = [\zeta(1), \zeta(2), \ldots, \zeta(T)]$ and $\boldsymbol{\alpha} = [\{\alpha_m(1)\}, \{\alpha_m(2)\}, \ldots, \{\alpha_m(T)\}]$. Constraint C1 specifies the BS operation status (either *power saving* or *active*), C2 forces the required number of VMs, $M(t)$, to be always greater than or equal to a minimum number $b \geq 1$: the purpose of this is to be always able to handle mission critical communications. C3 makes sure that the EB level is always above or equal to a preset threshold $\beta_{\text{low}}$, to guarantee *energy self-sustainability* over time. Note that this constraint may imply that in certain time slots the BS is to be switched off, although the workload may be non-negligible. When managing a single BS site (the formulation in this section), this implies that the load will not be served, but this fact may be compensated for when multiple communication sites are jointly managed, e.g., handing off the workload to another, energy richer, BS. This is dealt with in Section 4.4. Furthermore, C4 and C5, bound the maximum processing rate and workloads of each running VM $m$, with $m = 1, \ldots, M(t)$, respectively. Constraint C6 represents a hard-limit on the corresponding per-slot and per-VM processing time.

To solve P1 in Eq. (4.7), we leverage the use of LLC principles [27][76] and heuristics, obtaining the controls $\varsigma(t) \triangleq (\zeta(t), \{\alpha(t)\})$ for $t = 1, \ldots, T$. Note that Eq. (4.7) can iteratively be solved at any time slot $t \geq 1$, by just redefining the time horizon as $t' = t, t+1, \ldots, t+T-1$.

### 4.3.2 Communication Site Management

In this subsection, a traffic load and energy harvesting prediction method and an online management algorithm are proposed to solve the previously stated problem P1. Firstly, we discuss the prediction of the future (short-term) traffic load and harvested energy processes, and secondly, we solve P1 by first constructing the state-space behavior of the control system, where online control key concepts are introduced. Finally, the algorithm for managing the single communication site is presented.

#### Traffic load and energy forecasting

ML techniques constitute a promising solution for network management and energy savings in cellular networks [104][105]. In this work, given a time slot duration of $\tau = 30\,\text{min}$, we perform time series prediction, i.e., we obtain the $T = 3$ estimates of $\hat{L}_n(t)$ and $\hat{H}_n(t)$, by using an LSTM neural network developed in Python using Keras deep learning libraries (Sequential, Dense, LSTM) where the network has a visible layer with one input, one hidden layer of four LSTM blocks or neurons, and an output layer that makes a single value prediction. This type of recurrent neural network uses back-propagation through time for learning and memory blocks for regression [25]. The dataset is split as 67% for training and 33% for testing. The network is trained using 100 epochs (2,600 individual training trials) with batch size of one. As for the performance measure of the model, we use the RMSE. In this work, prediction steps similar to Table 3.2 are adopted, and Fig. 4.4 and Fig. 4.5 show the prediction results that will be discussed in subsection 4.5.2.

#### Edge System Dynamics

We denote the system state vector at time $t$ by $\boldsymbol{x}(t) = (M(t), \beta_n(t))$, which contains the number of active VMs, $M(t)$, and the EB level, $\beta_n(t)$, for the BS site $n$. $\boldsymbol{\varsigma}(t) = (\zeta(t), \{\alpha_m(t)\})$ is the input vector, i.e., the control action that drives the system behavior at time $t$. The system evolution is described through a discrete-time state-space equation, adopting the LLC principles [27] [76]:

$$\boldsymbol{x}(t+1) = \Phi(\boldsymbol{x}(t), \boldsymbol{\varsigma}(t))\,, \tag{4.8}$$

where $\Phi(\cdot)$ is a behavioral model that captures the relationship between $(\boldsymbol{x}(t), \boldsymbol{\varsigma}(t))$, and the next state $\boldsymbol{x}(t+1)$. Note that this relationship accounts for

1) the amount of energy drained $\theta_{\text{tot},n}(t)$, that harvested $H_n(t)$ and that purchased from the power grid $Q_n(t)$, which together lead to the next buffer level $\beta_n(t+1)$ through Eq. (4.5);

2) The traffic load $L_n(t)$, from which we compute the server workloads $\Gamma_n(t)$, that leads to $M(t)$ and to the control $\boldsymbol{\varsigma}(t)$.

The network management algorithm in the edge controller, the ENAAM algorithm, finds the best control action vector for the communication site, following a *model predictive control approach*. Specifically, for each time slot $t$, problem (4.7) is solved, obtaining control actions for the whole time horizon $t, t+1, \ldots, t+T-1$. The control action that is applied at time $t$ is $\boldsymbol{\varsigma}^*(t)$, which is the first one in the retrieved control sequence. This control amounts to setting the BS radio mode according to $\zeta^*(t)$, i.e., either active or power saving, and the number of instantiated VMs, $M^*(t)$, along with their obtained $\{\alpha_m^*(t)\}$ values (see remarks 1 and 2 below). This is repeated for the following time slots $t+1, t+2, \ldots$.

**Remark 1 (Role of prediction):** state $\boldsymbol{x}(t)$ and control $\boldsymbol{\varsigma}(t)$ are respectively measured and applied at the beginning of time slot $t$, whereas the offered load $L_n(t)$ and the harvested energy $H_n(t)$ are accumulated during the time slot and their value becomes known only by the end of it. This means that, being at the beginning of time slot $t$, the system state at the next time slot $t+1$ can only be estimated, which we formally write as:

$$\hat{\boldsymbol{x}}(t+1) = \Phi(\boldsymbol{x}(t), \boldsymbol{\varsigma}(t)) \,, \tag{4.9}$$

the same applies to the subsequent time slots in the optimization horizon $t+2, t+3, \ldots, t+T-1$. For these estimations we use the forecast values of load $\hat{L}_n(t)$ and harvested energy $\hat{H}_n(t)$, from the LSTM forecasting module.

**Remark 2 (VM number and workload allocation):** a remark on the provisioned VMs per time slot per-MEC server, $M(t)$, is in order. Specifically, the number of active VM (i.e., the VM computing cluster) depends on the predicted load, $\hat{L}_n(t+1)$, where the expected server workload is $\hat{\Gamma}_n(t+1) = 0.8\hat{L}_n(t+1)$. Each VM can compute an amount of up to $\gamma^{\max}$. Then, an estimate of the number of virtual machines that shall be active in time slot $t$ to serve the predicted server workloads is here obtained as: $M(t) = \left\lceil (\hat{\Gamma}_n(t+1)/\gamma^{\max}) \right\rceil$, where $\lceil \cdot \rceil$ returns the nearest upper integer. We heuristically split the workload among virtual machines by allocating a workload

$\gamma_m(t) = \gamma^{\max}$ to the first $M(t) - 1$ VMs, $m = 1, \ldots, M(t) - 1$, and the remaining workload $\gamma_m(t) = \hat{L}_n(t+1) - (M(t)-1)\gamma^{\max}$ to the last one $m = M(t)$.

**Controller decision-making:** the controller is obtained by estimating the relevant parameters of the operating environment, i.e., the BS load $\hat{L}_n(t)$ and the harvested energy $\hat{H}_n(t)$, and subsequently using them to forecast the future system behavior through Eq. (4.9) over a look-ahead time horizon of $T$ time slots. The control actions are picked by minimizing $J(\zeta, \alpha, t)$, see Eq. (4.6). At the beginning of each time slot $t$ the following process is iterated:

**1)** Future system states, $\hat{\boldsymbol{x}}(t+k)$, for a prediction horizon of $k = 1, \ldots, T$ steps are estimated using Eq. (4.9). These predictions depend on past inputs and outputs up to time $t$, on the estimated load $\hat{L}_n(\cdot)$ and energy harvesting $\hat{H}_n(\cdot)$ processes, and on the control $\boldsymbol{\varsigma}(t+k)$, with $k = 0, \ldots, T-1$.

**2)** The sequence of controls $\{\boldsymbol{\varsigma}(t+k)\}_{k=0}^{T-1}$ is obtained for each step of the prediction horizon by optimizing the weighted cost function $J(\cdot)$, see Eq. (4.6).

**3)** The control $\boldsymbol{\varsigma}^*(t)$ corresponding to the first control action in the sequence with the minimum total cost is the applied control for time $t$ and the other controls $\boldsymbol{\varsigma}^*(t+k)$ with $k = 1, \ldots, T-1$ are discarded.

**4)** At the beginning of the next time slot $t+1$, the system state $\boldsymbol{x}(t+1)$ becomes known and the previous steps are repeated.

### The ENAAM algorithm

Let $t$ be the current time. $\hat{L}_n(t+k-1)$ is the forecast load in slot $t+k-1$, with $k = 1, \ldots, T$, i.e., over the prediction horizon. For the control to be feasible, we need $\underline{\Gamma}_n(t) \leq B_n(t) \leq \hat{\Gamma}_n(t+k-1)$, where $\underline{\Gamma}_n(t)$ is the smallest $\Gamma$ such that $\texttt{round}(\hat{\underline{\Gamma}}_n(t+1)/\gamma^{\max}) = b$. For the buffer state, we heuristically set $\zeta(t+k-1) = \varepsilon$ if either $\beta_n(t+k-1) < \beta_{\text{low}}$ or $L_n(t+k-1) < L_{\text{low}}$, and $\zeta(t+k-1) = 1$ otherwise ($\beta_{\text{low}}$ and $L_{\text{low}}$ are preset low thresholds for the EB and the BS load, respectively). For slot $t+k-1$, the feasibility set $\mathcal{A}(t+k-1)$ contains the control pairs $(\zeta(t), \{\alpha_m(t)\})$ that obey these relations.

The algorithm is specified in Algorithm 2 as it uses the technique in [27]: the search starts (line 01) from the system state at time $t$, $\boldsymbol{x}(t)$, and continues in a breadth-first fashion, building a tree of all possible future states up to the prediction depth $T$. A cost

**Algorithm 2:** ENergy Aware and Adaptive Management (ENAAM)

**Input:** $\boldsymbol{x}(t)$ (current state)

**Output:** $\boldsymbol{\varsigma}^*(t) = (\zeta^*(t), \{\alpha_m^*(t)\})$

01: Initialization of variables

$\mathcal{S}(t) = \{\boldsymbol{x}(t)\}$, $\text{Cost}(\boldsymbol{x}(t)) = 0$

02: **for** $k = 1, \ldots, T$ **do**

- forecast the load $\hat{L}_n(t+k-1)$

- forecast the harvested energy $\hat{\text{H}}_n(t+k-1)$

- $\mathcal{S}(t+k) = \emptyset$

03: **for all** $\boldsymbol{x} \in \mathcal{S}(t+k-1)$ **do**

04: **for all** $\boldsymbol{\varsigma} = (\zeta, \{\alpha_m(t)\}) \in \mathcal{A}(t+k-1)$ **do**

05: $\hat{\boldsymbol{x}}(t+k) = \Phi(\boldsymbol{x}(t+k-1), \boldsymbol{\varsigma})$

06: $\text{Cost}(\hat{\boldsymbol{x}}(t+k)) = J(\zeta, \alpha, t+k-1)$

$+\text{Cost}(\boldsymbol{x}(t+k-1), \boldsymbol{\varsigma})$

07: $\mathcal{S}(t+k) = \mathcal{S}(t+k) \cup \{\hat{\boldsymbol{x}}(t+k)\}$

**end for**

**end for**

**end for**

08: **Find** $\hat{\boldsymbol{x}}_{\min} = \text{argmin}_{\hat{\boldsymbol{x}} \in \mathcal{S}(t+T)} \text{Cost}(\hat{\boldsymbol{x}})$

09: $\boldsymbol{\varsigma}^*(t) :=$ control leading from $\boldsymbol{x}(t)$ to $\hat{\boldsymbol{x}}_{\min}$

10: **Return** $\boldsymbol{\varsigma}^*(t)$

is initialized to zero (line 01) and is accumulated as the algorithm travels through the tree (line 06), accounting for predictions, past outputs and controls. The set of states reached at every prediction depth $t+k$ is referred to as $\mathcal{S}(t+k)$. For every prediction depth $t+k$, the search continues from the set of states $\mathcal{S}(t+k-1)$ reached at the previous step $t+k-1$ (line 03), exploring all feasible controls (line 04), obtaining the next system state from Eq. (4.9) (line 05), updating the accumulated cost as the result

of the previous accumulated cost, plus the cost associated with the current step (line 06), and updating the set of states reached at step $t+k$ (line 07). When the exploration finishes, the initial action (at time $t$) that leads to the best final accumulated cost, at time $t+T-1$, is selected as the optimal control $\boldsymbol{\varsigma}^*(t)$ (lines 08, 09, 10). Finally, for line 04, we note that $\Gamma_n$ belongs to the continuous set $[\underline{\Gamma}_n, \hat{L}_n(t+k-1)]$. To implement this search, we quantized this interval into a number of equally spaced points, obtaining a search over a finite set of controls.

**ENAAM complexity:** the computation complexity of the algorithm is $O(N_x N_\varsigma T)$, where $N_x \triangleq |\boldsymbol{x}(t)|$ and $N_\varsigma \triangleq |\boldsymbol{\varsigma}(t)|$ respectively represent the number of system states and the number of feasible actions at time $t$. Note that state and action space are respectively quantized into $N_x = M \times N_\beta$ and $N_\varsigma = 2 \times M \times N_\alpha$ levels, where $M$ is the number of virtual machines, $N_\beta$ is the number of quantization levels for the energy buffer and $N_\alpha$ is the number of quantization levels for the load variable $\alpha_m(t)$. Such quantization facilitates the search in Algorithm 2. Note that exhaustive search would entail a complexity of $O((N_x N_\varsigma)^T)$.

## 4.4 Multiple Communication Sites

In this section, we extend the work from section 4.3 by considering the energy savings for *multiple* communication sites. We formulate an optimization problem to obtain energy savings through short-term traffic load and harvested energy predictions, clustering, along with energy management procedures for the clustered BS sites. The problem formulation for multiple communication sites is described in subsection 4.4.1, then cluster formation is discussed in subsection 4.4.2, and the edge management procedure for each cluster, enabled by the edge controller, is presented in subsection 4.4.3.

### 4.4.1 Problem Formulation

Our objective is to improve the overall energy savings of the network by clustering BSs based on their location (or distance measures) similarity, and then optimizing the energy savings within each cluster by employing the single optimization case described in Section 4.3. From an energy efficiency perspective, in a cluster of BS nodes, one BS (or more) might have a preference of switching off, by first offloading its (their) traffic load to its (their) neighboring BS(s) that have enough spare capacity for handling extra traffic load, and then switching off. The whole offloaded traffic load from the BS,

denoted by BS $n$, is allocated to the neighboring cluster member (active BS) in which orthogonal resource allocation helps mitigate intra-cluster interference, such that the selected neighboring BS, denoted by BS $n'$, is allocated the incremental load, denoted by $L_{nn'}(t) \triangleq L_n(t)$. Whenever a BS is switched off, it should maintain service to its users via a re-association process in order to offload the users to the neighboring active BS having extra resources for handling upcoming extra traffic load. The re-association process involves notifying the connected users to try and connect to neighboring BSs with extra resources.

In the view of the above, we consider that all BSs are grouped into sets of clusters $\mathcal{O} = \{O_1, \ldots, O_{|\mathcal{O}|}\}$. Here, a given cluster $O_i \in \mathcal{O}$, with $i = 1, \ldots, |\mathcal{O}|$, consists of a set of BSs that coordinate with the controller. The clustering mechanism is discussed in subsection 4.4.2. For each cluster $O_i \in \mathcal{O}$, we aim to minimize the energy consumption, i.e., the consumption due to BS transmission and the running VMs in the servers, using BS power saving modes and VM soft-scaling per active cluster member. To do so, we define a cost function which captures the individual communication site energy consumption and its QoS. The (weighted) cost for each cluster member, BS $n \in O_i$, is redefined as:

$$J_n(\zeta, \alpha, t) \triangleq \overline{\eta}\theta_{\text{tot},n}(\zeta_n(t), \{\alpha_m(t)\}_n, t) + \eta(\Lambda_n(t) - B_n(t))^2\,, \tag{4.10}$$

where $\zeta_n(t)$ is the activity status of BS $n$ (either *power saving* or *active*), $\{\alpha_m(t)\}_n$ is the set of factors for the allocated VMs at BS $n$. Moreover, $\Lambda_n(t) \leftarrow L_n(t)$ if BS $n$ only handles its own traffic, whereas $\Lambda_n(t) \leftarrow L_n(t) + \Delta L_n(t)$, in case one (or multiple) BSs are switched off in time slot $t$ and its (their) traffic is redirected (handed off) to BS $n$. The computation of $\Delta L_n(t)$ is addressed in subsection 4.4.3. The per cluster cost $\Upsilon_{O_i}(\boldsymbol{\zeta}_i, \boldsymbol{\alpha}_i, t)$ is the aggregated cost of all cluster members, $\Upsilon_{O_i}(\boldsymbol{\zeta}_i, \boldsymbol{\alpha}_i, t) = \sum_{\forall n \in O_i} J_n(\zeta, \alpha, t)$. Hence, over time horizon, $t = 1, \ldots, T$, the following optimization problem is defined:

$$\begin{aligned}
\mathbf{P2} \quad &: \quad \min_{\mathcal{E}} \sum_{\forall O_i \in \mathcal{O}} \Upsilon_{O_i}(\boldsymbol{\zeta}_i, \boldsymbol{\alpha}_i, t) \\
&\text{subject to:} \\
\text{C1} - \text{C6} \quad &: \quad \text{from Eq. (4.7)}, \\
\text{C7} \quad &: \quad |O_i| \geq 1, \forall\, O_i \in \mathcal{O}, \\
\text{C8} \quad &: \quad O_i \cap O_j = \emptyset, \forall\, O_i, O_j \in \mathcal{O}, O_i \neq O_j,
\end{aligned} \tag{4.11}$$

where $\mathcal{E} \triangleq \{\boldsymbol{\zeta}_i, \boldsymbol{\alpha}_i\}$ is the collection of variables to be reconfigured for all the BS clusters (the whole MN), for all time slots $t = 1, \ldots, T$. As for the constraints, C7

and C8 ensure that each BS is part of only one cluster. Solving **P2** in Eq. (4.11) involves BS clustering, the forecasting method from subsection 4.3.2, a heuristic rule for the selection of which BSs have to be switched off, and the ENAAM algorithm (see algorithm 2). Once **P2** is solved, the control action to be applied at time $t$, per cluster $O_i$, corresponds to the elements in $\{\boldsymbol{\zeta}_i, \boldsymbol{\alpha}_i\}$ that are associated with the first time slot 1 in the optimization horizon. As above, Eq. (4.11) can iteratively be solved at any time slot $t \geq 1$, by just redefining the time horizon as $t' = t, t+1, \ldots, t+T-1$.

### 4.4.2 Cluster Formation

Clustering algorithms have been proposed as a way of enabling energy saving mechanisms in BSs, where groups of inactive BSs or BSs with low loads are switched off. With the advent of EH BSs, the BSs with $\beta_n(t) < \beta_{\text{low}}$ can be switched off, while still guaranteeing the QoS through the other active BSs. That is, within each formed cluster, the controller tries to minimize the cost function, which captures the trade-off between the energy efficiency and the QoS of each cluster member. The key step in clustering is to identify similarities or distance measures between BSs in order to group BSs with similar characteristics. In this work, we use the location of the BSs as it defines the relative neighborhood (the distance measures) with the other BSs. Using the location of the BSs and the distance between the BSs, we obtain a distance-based similarity matrix $\boldsymbol{W}^d$. In addition, we assume that the network topology is static during the clustering algorithm execution.

Next, we detail the clustering measure that we use to obtain the similarities between BSs based on location, followed by the distance-based clustering algorithm.

**Relative neighborhood based on BS adjacency and Gaussian similarity**: Similar to [56], we model the MN as a graph $G = (\mathcal{N}, E)$, where $\mathcal{N}$ represents the set of BSs, while the set $E$ contains the edges between any two BSs. There is an edge $(n, n') \in E$ if and only if $n$ and $n'$ can mutually receive each other's transmission. In this case, we say that $n$ and $n'$ are neighbors. We use a parameter $r_{nn'}$ to characterize the presence of a link between nodes, where $r_{nn'} \in \{0, 1\}$. Let $y_n$ be the coordinates of BS $n \in \mathcal{N}$ in the Euclidean space. The relative neighborhood of BS $n$ is defined by the nearness of the BSs in its $e_d$-radio propagation space (or neighborhood):

$$\mathcal{Z}_n = \{n' \text{ s.t. } \|y_n - y_{n'}\| \leq e_d\}. \tag{4.12}$$

If $n' \in \mathcal{Z}_n$ we say that BSs $n$ and $n'$ are neighbors, and we set $r_{nn'} = 1$, otherwise

$r_{nn'} = 0$. The links between the vertices in $\mathcal{N}$ are weighted based on their similarities. Based on the distance between BS $n$ and $n'$, we can classify the BSs based on their location using the Gaussian similarity measure [56] (a classification kernel function used in machine learning), which is defined as:

$$w_{nn'}^{d} = \begin{cases} \exp\left(\dfrac{-\|y_n - y_{n'}\|^2}{2\sigma_d^2}\right) & \text{if} \quad \|y_n - y_{n'}\| \leq e_d, \\ 0 & \text{otherwise,} \end{cases} \tag{4.13}$$

where $2\sigma_d^2$ adjust the impact of the neighborhood size. In Eq. (4.13), we assume that the BSs located far from each other have low similarities, compared to those that are close to each other, as those that are close are more likely to cooperate with each other. The distance-based similarity matrix $\boldsymbol{W}^d$ is formed using $w_{nn'}^d$ as the $(n, n')$-th entry.

**Distance-based clustering**: The BS clustering is performed after obtaining the similarity matrix $\boldsymbol{W}^d$ of the MN graph $G = (\mathcal{N}, E)$. Given the matrix $\boldsymbol{W}^d$, we employ a centralized clustering method, specifically the K-means [112], as the matrix provides the full location knowledge. K-means partitions the set of nodes into clusters in which each node belongs to the cluster with the nearest mean distance. In addition, the value of $K$, i.e., the number of clusters ($|O_i|$), is known prior and is a design parameter. This algorithm requires knowledge of all the BS locations, thus, it is categorized as a centralized method. In our case, this process does not incur any computation delay as the edge controller is assumed to have high computation capabilities.

### 4.4.3 Edge Network Management

Our aim is to implement and validate an LLC framework for dynamic resource provisioning in multiple communication sites with the goal of achieving energy savings within the access network through BS sleep modes and VM soft-scaling. Given the formation of clusters, load and energy forecasting, our next goal is to developed a mechanism for solving **P2** (Eq. (4.11)) where each cluster of BSs adjust its transmission parameters and its computing cluster entities based on the forecast information. In order to minimize the per cluster cost function, we introduce the notion of *network impact* which is discussed the next, followed by the edge management procedure description and its complexity analysis.

**Network Impact**: The dynamic BS switching off strategies may have an impact on the network due to the traffic load that is offloaded to the neighboring BSs. To avoid this, the BS to be switched off must be carefully identified within a BS cluster. To determine whether a particular BS can be switched off or not, we follow the work done in [57]. As an example, we consider one cluster $O_i$, together with its cluster members $n \in O_i$, then from it we choose one BS, BS $n$, where BS $n$ neighbors set is denoted by $\mathcal{N}_n$. Note that the BS $n' \in \mathcal{N}_n$ is the BS to which the traffic load will be offloaded to after turning off BS $n$. Also, BS $n$ can only be switched off if there exists a neighboring BS $n'$ that satisfies the following feasibility constraint [57]:

$$L_{n'}(t) + L_{nn'}(t) \leq 1, \quad n' \in \mathcal{N}_n, \tag{4.14}$$

where $L_{n'}(t)$ is the original BS $n'$ traffic load and $L_{nn'}(t)$ is the incremental traffic load from BS $n$ (the switched off BS) to BS $n'$ (the neighboring BS). We recall that the load $L_{n'}(t)$ is normalized with respect to the maximum load that a BS can sustain, so the inequality in Eq. (4.14) means that it is feasible for BS $n'$ to take the extra load from BS $n$. To quantify how the incremental system load affects the overall network load due to the switching off process, we introduce the notion of *network impact*. For every BS $n$ within cluster $O_i$, $i = 1, \ldots, K$, its *network impact* due to the offloaded system load onto one of the neighboring BSs is defined as:

$$I_n(t) = \max_{n' \in \mathcal{N}_n} [L_{n'}(t) + L_{nn'}(t)], \forall\, n \in O_i. \tag{4.15}$$

Here, the maximum network impact value $I_n(t)$ over the neighboring BSs is considered as a measure for each BS towards switching off and generating extra traffic loads for its neighboring BSs. In this work, considering cluster $O_i$, we switch off the BS $n^*$ that has the least network impact, i.e.,

$$n^* = \operatorname*{argmin}_{n \in O_i} I_n(t). \tag{4.16}$$

The BS that takes the load from $n^*$ is selected as the BS $n'$ that minimizes $L_{n'}(t) + L_{n^*n'}(t)$ over the set of active BSs that are on within the cluster $O_i$. For BS $n'$, we then set $L_{n'}(t) \leftarrow L_{n'}(t) + L_{n^*n'}(t)$. This procedure is sequentially repeated for all the cluster members until there is no active BS whose neighbors satisfy the feasibility condition of Eq. (4.14). Note that here, we focus only on which BS to switch off, as for the BS turning on state, we assume that the *commitment time* (time configured so that the BS automatically wakes up without external triggers) is a system parameter that is pre-configured when the BS is switched off.

**Edge management procedure**: Here, we propose a distributed edge network management procedure that makes use of the ENAAM algorithm (see subsection 4.3.2). The decision making criterion only depends on the BS information and on its neighboring BSs, thus, the BS switching off decision can be localized within each cluster. To decide which BSs shall be switched off, we follow a sequential decision process. While this is heuristic, it allows coping with the high complexity associated with an optimal (all BSs are jointly assessed) allocation approach. The edge management procedure is as follows.

For each BS cluster $O_i$, with $i = 1, \ldots, K$, do:

**1)** Initialize an allocation variable $\Delta L_n(t) = 0$ for all BSs $n \in O_i$. Compute $I_n(t)$, using Eq. (4.15), for all BSs $n$ and obtain the BS with the least *network impact* $n^*(t)$, using Eq. (4.16). Switch off BS $n^*(t)$ and assign its load to the neighboring BS $n' \in O_i$ that minimizes $L_{n'}(t)+\Delta L_{n'}(t)+L_{n^*n'}(t)$. Update the extra allocation for BS $n'$ as $\Delta L_{n'}(t) \leftarrow \Delta L_{n'}(t) + L_{n^*n'}(t)$. Recompute $I_n(t)$ for all the BSs that are still on and identify the next BS that can be switched off, i.e., the one with the *least network impact*. This procedure is repeated until none of the BSs in the cluster verifies Eq. (4.14). At this point, we have identified all the BSs $n^*$ that shall be switched off in $O_i$.

**2)** For each active BS $n' \in O_i$, the ENAAM algorithm is executed using $L_{n'}(t) + \Delta L_{n'}(t)$, where $\Delta L_{n'}(t) = 0$ if BS $n'$ does not take extra load, whereas it is greater than zero otherwise. Note that, $\Delta L_{n'}(t)$ corresponds to the total traffic that is handed over to BS $n'$, possibly from multiple nearby BSs.

**Edge network management complexity:** the algorithm is independently executed for each cluster and the corresponding time complexity is obtained as follows. Considering the action **Step 1**, from above, the time complexity associated with the computation of the BS having the least network impact is linear with the size of the cluster $|O_i|$. Once that is computed, the complexity associated with updating the load allocation for the active BSs is $|O_i|-1$, which leads to a total complexity of $|O_i|(|O_i|-1) = O(|O_i|^2)$. Moreover, such process is iterated for each BS that is switched off. In the worst case, where all the BSs but one are switched off, the final complexity of step 1 is $O(|O_i|^3)$. As for **Step 2**, from above, the computation complexity depends on the ENAAM algorithm, which is independently executed by each *active* BS. Thus, in the worse case (no BSs are switched off), the total aggregated complexity is: $O(|O_i|N_x N_\varsigma T)$, which

is linear in all variables, namely, number of cluster members, number of BS states, number of actions and time horizon $T$.

## 4.5 Performance Evaluation

In this section, we show some selected numerical results for the scenario of Section 4.2. The parameters that were used for the simulations are listed in Table 4.2.

Table 4.2: System Parameters.

| **Parameter** | **Value** |
|---|---|
| Total BSs, $N$ | 24 |
| Max. number of VMs, $M$ | 27 |
| Min. number of VMs, $b$ | 1 |
| Time slot duration, $\tau$ | 30 min |
| Operating power, $\theta_0$ | 10.6 W |
| Energy overheads for switching VM, $\theta_m^{\text{ov}}(t)$ | 0.05 J/MHz$^2$ |
| Max. computation workload per VM, $\gamma^{\max}$ | $\{5, 10\}$ MB |
| Max. allowed processing time, $\Delta$ | 0.8 s |
| Energy cons. of network interfaces, $\theta_{\text{idle}}$ | 3 J |
| Cost of exchanging one unit of data, $\theta_{\text{data}}$ | 6 J/byte |
| Processing rate set, $\mathcal{F}$ | $\{0, 4, 8, 12, 16, 20\}$ |
| Static energy consumed by VM, $\theta_{\text{idle},m}(t)$ | 4 J |
| Max. energy cons. by VM at $f_{\max}$, $\theta_{\max,m}(t)$ | 10 J |
| Energy storage capacity, $\beta_{\max}$ | 490 kJ |
| Lower energy threshold, $\beta_{\text{low}}$ | 30% of $\beta_{\max}$ |
| Upper energy threshold, $\beta_{\text{up}}$ | 70% of $\beta_{\max}$ |
| Low traffic threshold, $L_{\text{low}}$ | 4 MB |

### 4.5.1 Simulation Setup

We consider multiple BSs, each one co-located with a MEC server and a coverage radius of 40 m. In addition, we use a virtualized server with specifications from [106] for a VMware ESXi 5.1-ProLiant DL380 Gen8. Our time slot duration $\tau$ is set to 30 min and the time horizon is set to $T = 3$ time slots.

### 4.5.2 Numerical Results

**Pattern forecasting:** we show real and predicted values for the traffic load and harvested energy over time in Figs. 4.4 and 4.5, where we track the one-step predictive mean value at each step of the online forecasting routine. Then, Table 4.3 shows the average RMSE of the normalized harvested energy and traffic load processes, for different time horizon values, $T \in \{1, 2, 3\}$. Note that the predictions for $H(t)$ are more accurate than those of $L(t)$ (confirmed by comparing the average RMSE), due to differences in the used dataset granularity. However, the measured accuracy is deemed good enough for the proposed optimization.

Table 4.3: Average prediction error (RMSE) for harvested energy and traffic load processes, both normalized in $[0, 1]$.

| | $T = 1$ | $T = 2$ | $T = 3$ |
|---|---|---|---|
| $L(t)$ | 0.037 | 0.042 | 0.048 |
| $H(t)$ | 0.011 | 0.016 | 0.021 |

**Single communication site:** Figs. 4.6 and 4.7 are computed with $\eta = 0$ using Cluster 1 and Solar 1 as traffic load and harvested energy profiles for each BS (see Figs. 4.2 and 4.3). Moreover, $\gamma^{\max} = 5$ MB and 10 MB, respectively. They show the mean energy savings achieved over time when on-demand and energy-aware edge resource provisioning is enabled (i.e., BS sleep modes and VM soft-scaling), in comparison with the case where they are not applied. Our edge network management algorithm (ENAAM) is benchmarked with another one that heuristically selects the amount of traffic that is to be processed locally, $B_n(t) \leq \Gamma_n(t)$, depending on the expected load behavior. It is named Dynamic and Energy-Traffic-Aware algorithm with Random behavior (DETA-R). Both ENAAM and DETA-R are aware of the predictions in

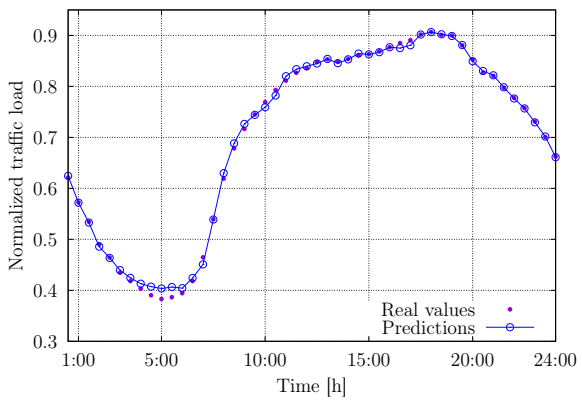



Figure 4.4: One-step ahead predictive mean value for $L(t)$.

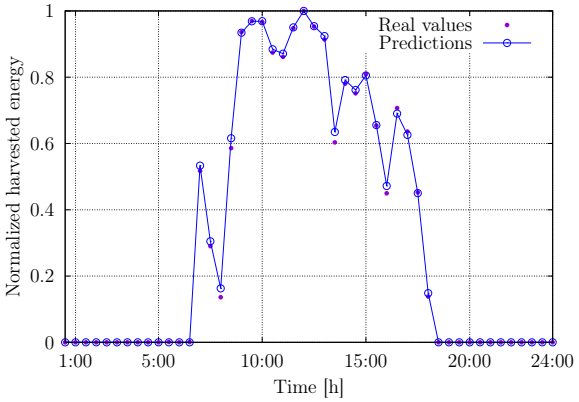



Figure 4.5: One-step ahead predictive mean value for $H(t)$.

future time slots (see Section 4.3.2), however, DETA-R provisions edge resources using a heuristic scheme. DETA-R heuristic works as follows: if the expected load difference is $\hat{L}(t+1) - \hat{L}(t) > 0$, then the normalize workload to be processed by BS $n$ in the current time slot $t$, $B_n(t)$, is randomly selected in the range $[0.6, 1]$, otherwise, it is picked evenly at random in the range $(0, 0.6)$.

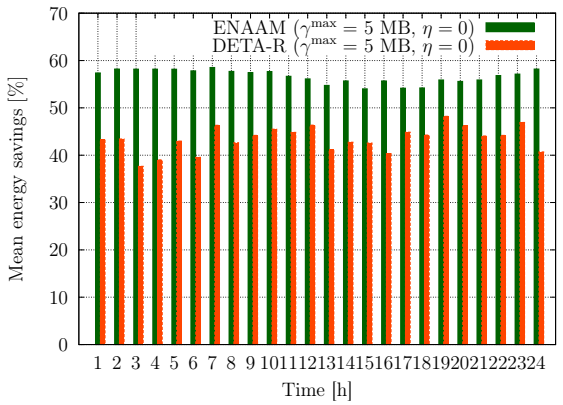



Figure 4.6: Mean energy savings for $\eta = 0$ and $\gamma^{\max} = 5$ MB.

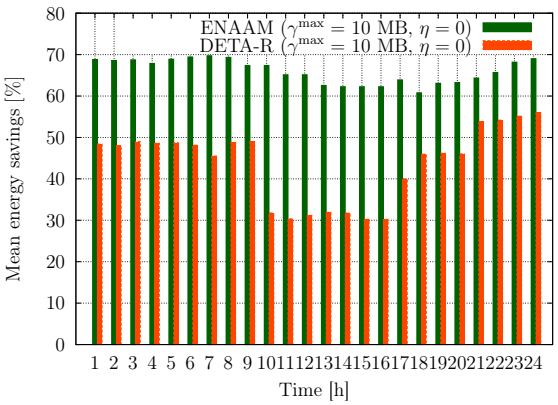



Figure 4.7: Mean energy savings for $\eta = 0$ and $\gamma^{\max} = 10$ MB.

Average results for the ENAAM scheme show energy savings of 69% ($\gamma^{\max} = 10$ MB) and 57% ($\gamma^{\max} = 5$ MB), while DETA-R achieves 49% ($\gamma^{\max} = 10$ MB) and 43% ($\gamma^{\max} = 5$ MB) on average, where these savings are with respect to the case where *no energy management* is performed, i.e., the network is dimensioned for maximum expected capacity (maximum value of $\theta_{\text{tot},n}(t)$, with $M = 27$ VMs, $\forall\, t$). The results show that the maximum load allocated to each VM, $\gamma^{\max}$, has an impact towards energy savings. An increase in energy savings is observed when $\gamma^{\max} = 10$ MB due to

the fact that the number of VMs demanded per time slot is reduced, when compared to the allocation of $\gamma^{\max} = 5$ MB.

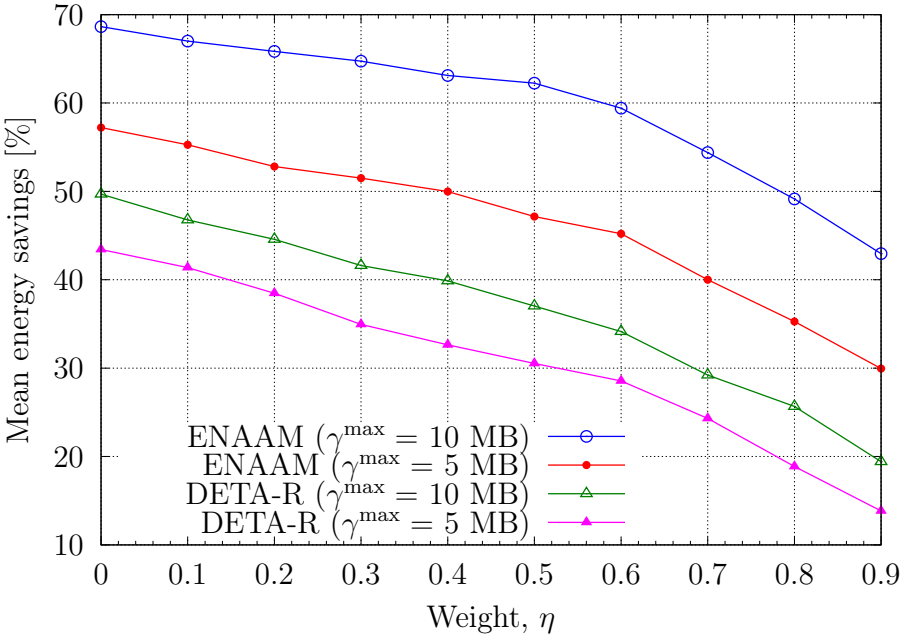



Figure 4.8: Energy savings *vs* weight $\eta$ (single BS case).

The ESs evolution with respect to $\eta$ is presented in Fig. 4.8, taking into account the load allocated to each VM, $\gamma^{\max}$. The results were obtained using Cluster 1 and Solar 1 as traffic load and harvested energy profiles (*see* Fig. 4.2 and Fig. 4.3). As expected, a drop in energy savings is observed when QoS is prioritized, i.e., $\eta \to 1$, as in this case the BS energy consumption is no longer considered. It can be observed that ENAAM achieves a 50% (or above) from $\eta = [0, 0.4]$ when $\gamma^{\max} = 5$ MB and from $\eta = [0, 0.7]$ when $\gamma^{\max} = 10$ MB. This shows that the higher the load allocated to each VM, the lesser the energy that is drained, as few VMs are running. DETA-R operates at below 50% for all $\eta$ and $\gamma^{\max}$ values.

**Multiple communication sites:** Figs. 4.9 and 4.10 present the mean energy savings achieved with respect to the cluster size and the weight $\eta$, using all the traffic load and harvested energy profiles from Fig. 4.2 and Fig. 4.3. Each BS randomly picks its own traffic load and harvested energy profile at the beginning of the optimization process. Here, to select the BS to be switched off, we use the management procedure of section 4.4.3. As for DETA-R, a BS is randomly selected to evolve its operating mode to power saving mode and offload its load to a nearby BS (in this case, the least loaded neighboring BS is selected), without taking into account its network impact measure.

Fig. 4.9 shows the average energy savings obtained when clustering is adopted, i.e., here, the cluster size is increased from $|O_i| = 1$ to 10 and $\eta = 0$. The obtained energy savings are with respect to the case where all BSs are dimensioned for maximum expected capacity (maximum value of $\theta_{\text{tot},n}(t)$, with $M = 27$ VMs, $\forall t, \forall n \in O_i$). It should be noted that the energy savings increase as the size of the cluster grows, thanks to the load balancing among active BSs, which cannot be implemented in the single communication site scenario (i.e., when BSs are independently managed).

Then, Fig. 4.10 shows the average energy savings with respect to $\eta$, when the cluster size is set to an intermediate case ($|O_i| = 6$). Again, here the energy savings are obtained with respect to the case where all the BSs are dimensioned for maximum capacity. As expected, there is a drop in the energy savings achieved as the value of $\eta$ increases, as QoS is prioritized. It can be observed that ENAAM achieves a value of 50% or above when $\eta = [0, 0.8]$ (at $\gamma^{\max} = 10$ MB) and when $\eta = [0, 0.6]$ (at $\gamma^{\max} = 5$ MB). DETA-R achieves value above 50% or above when $\eta = [0, 0.4]$ (at $\gamma^{\max} = 10$) and $\eta = [0, 0.1]$(at $\gamma^{\max} = 5$ MB).

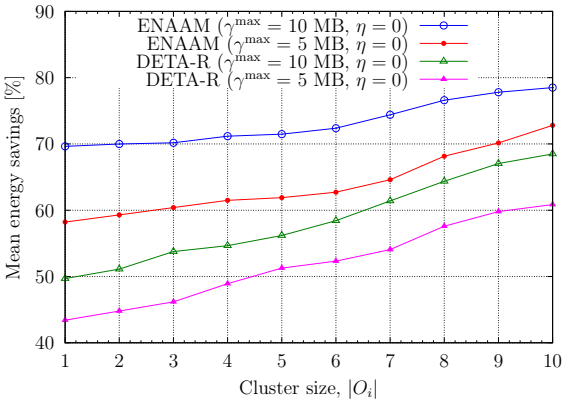



Figure 4.9: Energy savings *vs* cluster size.

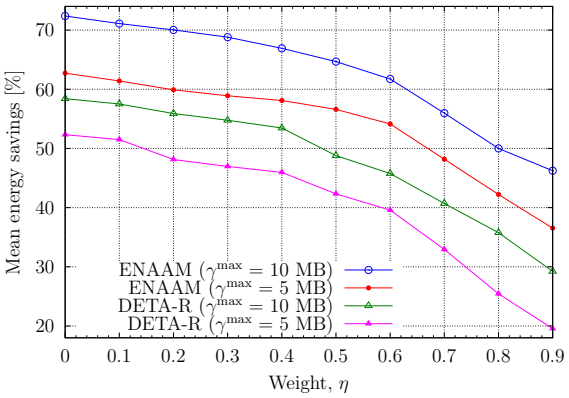



Figure 4.10: Energy savings *vs* $\eta$ for $|O_i| = 6$.

Comparing Figs. 4.8 and 4.10, an average gain of 9% on the energy savings is observed when clustering is applied, by considering the mean energy savings with respect $\eta$ achieved with ENAAM for both cases. From Fig. 4.9 we see that this gain can be as high as 16% for ENAAM with $\gamma^{\max} = 5$ MB (red curve) and bigger for the DETA-R approach. These results support the notion that performing a clustering-based optimization is beneficial thanks to the additional cooperation within each neighborhood of BSs. This cooperation allows to switch off more BSs through load balancing, increasing the energy savings while still controlling the users' QoS.

## 4.6 Conclusion

In *summary*, we have envisioned an edge network where a group of BSs are managed by a controller, for ease of BS organization and management, and also a mobile network where the edge apparatuses are powered by hybrid supplies, i.e., using green energy in order to promote energy self-sustainability and the power grid as a backup. Within the edge, each BS is endowed with computation capabilities to guarantee low latency to mobile users, offloading their workloads locally. The combination of energy saving methods, namely, BS sleep modes and VM soft-scaling, for single and multiple BS sites helps to reduce the mobile network's energy consumption. An edge energy management algorithm based on forecasting, clustering, control theory and heuristics, is proposed with the objective of saving energy within the access network, possibly making the BS system self-sustainable. Numerical results, obtained with real-world energy and traffic load traces, demonstrate that the proposed algorithm achieves energy savings between 57% and 69%, on average, for the single communication site case, and a gain ranging from 9% to 16% on energy savings is observed when clustering is applied, with respect to the allocated maximum per-VM loads of 5 MB and 10 MB. The energy saving results are obtained with respect to the case where no energy management techniques are applied, either in one BS or single cluster.

# Chapter 5
# Virtualized Platform Optimization

The results of this chapter are the subject of the following published paper:
**T. Dlamini**, A. F. Gambin, "Adaptive Resource Management for a Virtualized Computing Platform within Edge Computing", in *Proceedings of the IEEE International Conference on Sensing, Communication and Networking (SECON) workshop on Edge Computing for Cyber Physical Systems (CyberEdge)*, Boston, MA, USA, June 2019, [Online]. Available: arXiv:1906.05008.

## 5.1 Introduction

With the growing concern on the considerable energy drained by computing platforms (e.g., micro or large data centers), research efforts are targeting means towards greening them in order to achieve higher EE. On one hand, the integration of EH into the edge (or computing) system [2][23] is foreseen as a way of minimizing the carbon footprint and the dependence on the power grid. On the other hand, server virtualization is emerging as the prominent approach to consolidate multiple enterprise applications from multiple platforms to a *single* server, with an objective to save energy. In addition, the provisioning of computing resources based on demand is made possible. This allows operators to maintain the desired QoS while achieving higher EE. Despite of this, little understanding has been obtained about the potential overheads related to energy consumption within the virtualized platform [113]. Therefore, understanding the MEC server's energy cost or processes is of great importance towards the development of energy management procedures.

The energy drained (in the MEC server) is due to the computing-plus-communication processes and it is associated with (i) the running VMs [18][19] and (ii) the communi-

cation within the server's virtual network [20]. It is observed in the literature that most of the existing energy saving studies have involved the scaling up/down the number of computing nodes/servers or VMs [66][67][68][69], enabling VM migration to under utilized servers [70] and shortening the access time to physical resources [71]. Hence, the proposed energy models (a summary of the proposed models is found in [113]) and the overall operational expenditure of the computing node is usually related to the computation process (i.e., the running VMs), overlooking the communication processes within the server.

The communication-related energy cost has been considered in the following works: energy consumption within a server that transmit wireless to mobile clients [68] and rate adaptation within vehicular network connecting wireless to Fog nodes (nodes with virtualized computing platforms) [67]. The works considered only the intra-communications energy cost within the computing platforms. Also, the works of [21] and [22] considers the communication-related energy consumption within a MN infrastructure, i.e., within a computing node (router) where the transmission drivers are switched on/off towards energy savings. It is worth observing that both works, [21][22], are not along the direction of MEC, but they propose the tuning of the transmission drivers as *one* of the ES strategies within the MN infrastructure.

Thus, energy savings within the MEC server can be *jointly* achieved by launching an optimal number of VMs for computing, and transmission drivers *coupled* with the location-aware traffic routing for real-time data transfers. In addition, a significant amount of power is consumed even when the server is idle [19][114] (with no data transfer activities), with data links active. This opens an opportunity for tuning even the Network Interface Card (NIC) so that the energy drained is always zero when there is no data transfer.

In this chapter, to identify and quantify the server's energy consumption due to the computing-plus-communications processes, we focus on the virtualized MEC server and propose an online algorithm for managing the MEC server. This work differs from the work done in Chapter 3 and 4 as the MEC server is placed in proximity to a BS cluster, and not one co-located for each BS. Moreover, here we focus on the integration of communication-related energy consumption by considering the tuning of the transmission drivers, which is a novel concept within the MEC paradigm. The proposed optimization strategy is able to reduce the energy consumption under the guidance of the online resource controller and the energy management procedures.

### 5.1.1 Related work

For several years, great effort has been devoted to study energy savings in computing environments with the aim of minimizing the energy consumption. Procedures for the dynamic on/off switching of servers/VMs have been proposed as a way of minimizing energy consumption in computing platforms. A novel post-decision state based learning algorithm for server provisioning at the network edge is presented in [66]. This work incorporates renewable energy. Then, in [67], an iterative algorithm obtains the number of VMs to be provisioned within a node that transmit to clients wireless. The work of [68] consider a vehicular scenario where vehicles connect wireless to Fog nodes and then develop an adaptive scheduler, which computes on-the-fly the solutions of both the resource reconfiguration and consolidation problems. For this purpose, the primal-dual algorithm is used. In addition, in [69], an automated server provisioning system that aims to meet workload demand while minimizing energy consumption in data centers is presented. Here, the optimization problem is approximated through a linear programming formulation and solved using an efficient fast polynomial-time method. To handle energy in cloud platforms, the work of [70] present an autonomic and energy-aware mechanisms for self-managing the resources. A modified best fit decreasing (BFD) algorithm is used for the dynamic consolidation of VM resource partitions. Then, the works of [22] and [21] tries to address the communication-related energy cost by employing heuristics and traffic matrix decomposition techniques.

### 5.1.2 Objective and Contributions

The main contributions of this chapter are as follows:

- we consider a scenario, where MEC and EH are combined into a single system located close to a BS cluster, towards energy self-sustainability in MNs. The EH-MEC system is equipped with solar panels for EH and an EB for energy storage.

- We consider a computing-plus-communication energy model within the MEC paradigm, formulating a constrained optimization problem. Due to the non-linear behavior of the rate-vs-power relationship, the optimization problem is non-convex. To solve it, we convexify the function by using GP [115] and then employing the CVXOPT toolbox[1] and approximations.

[1]M. Andersen and J. Dahl. CVXOPT: Python Software for Convex Programming, 2019. [Online].

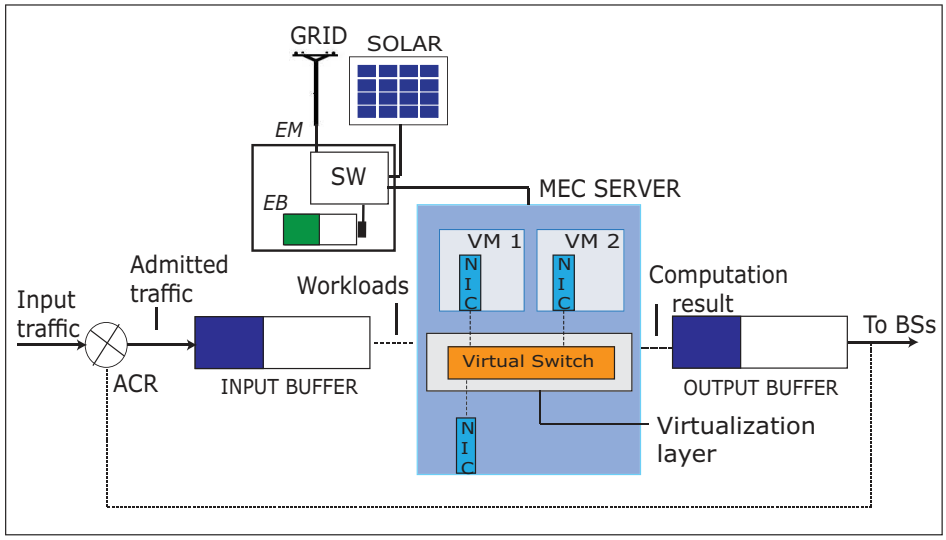



Figure 5.1: Virtualized computing system powered by hybrid energy sources: on-grid power and green energy. The electromechanical switch (SW) is for aggregating the energy sources to fulfill the energy required to power the computing platform.

- We forecast the short-term future server workload and harvested energy, by using a LSTM neural network [25], to enable foresighted optimization.
- Lastly, we develop an online controller-based algorithm *called* Automated Resource Controller for Energy-aware Server (ARCES) for the MEC server management based on LLC theory [28] and energy management procedures. The main goal is to minimize the overall energy consumption, under hard per-task delay constraints (i.e., QoS), through the *joint* consideration of VM soft-scaling and the tuning of transmission drivers coupled with the location-aware traffic routing. To the best of our knowledge, this is a novel concept within the MEC paradigm. ARCES considers future server workloads, onsite green stored energy, and target BS (based on the Location Service (LS) [116]), and then enable ES procedures.

## 5.2 System Model

As a major deployment of MEC [12], the considered network scenario is illustrated in Fig. 5.1. It consists of a cache-enabled, TCP/IP offload-enabled (*partial* computation at the network adapter), virtualized MEC server hosting $M$ VMs and it is assumed

Available: https://cvxopt.org/

to be deployed at an aggregation point [2][12], i.e., a point in proximity to a group of BSs interconnected to the MEC server for computation offloading. The MEC node is assumed to be equipped with higher computational and storage resources compared to the end-user device. The server clients are assumed to be mobile users moving in groups and they are represented by the Reference Point Group Mobility Model (RPGM [117]). Their current locations are known through the LS API [116], in the MEC platform, which is a service that supports UE's location retrieval mechanism, and then passing the information to the authorized applications within the server. The computing site is empowered with EH capabilities through a solar panel and an EB that enables energy storage. Energy supply from the power grid is also available for backup. The EM is an entity responsible for selecting the appropriate energy source and for monitoring the energy level of the EB. The virtualized Access Control Router (ACR) of Fig. 5.1 acts as an access gateway, responsible for routing, and it is locally hosted as an application. Moreover, we consider a discrete-time model, whereby time is discretized as $t = 1, 2, \ldots$, and each time slot $t$ has a fixed duration $\tau$. The list of the symbols that are used in this chapter is reported in Table 5.1.

### 5.2.1 Server Workload and Energy Consumption

For many MN services, the workload demand exhibits a diurnal behavior, thus it suffices to forecast the short-term server workload (using historical datasets [2][69]) and then enable dynamic resource management within the server. In this work, anonymized real server workload traces[2] obtained from [118] are used due to the difficulties in obtaining relevant open source datasets containing computing requests. A trace file consist of the file size, session duration, total number of packets and average transmission rate, over one day. In our numerical results, we use the total number of packets, denoted by $L_{\text{in}}(t)$ ([bits]), to represent the buffered (or admitted) computation workload at the input buffer at time slot $t$ (*see* red curve in Fig. 5.2). In addition, we assume that the upper-bounded input/output (I/O) queue's of Fig. 5.1 are loss-free and they implement the First-In First-Out (FIFO) service discipline, thus $L_{\text{in}}(t) = L_{\text{out}}(t)$, where $L_{\text{out}}(t)$ is the amount of the aggregate computation result stored at the output buffer.

The total energy consumption ([J]) for the virtualized computing platform is for-

[2]For the purpose of emulating the MEC server workload we make use of server related traces as they involve the computing and communication processes within the server.

Table 5.1: Notation: list of symbols used in the analysis.

| **Symbol** | **Description** |
|---|---|
| | **Input Parameters** |
| $M$ | maximum number of VM hosted on the server |
| $L_{\text{in}}(t)$ | buffered computation workload at time slot $t$ |
| $L_{\text{out}}(t)$ | aggregate computation result at time slot $t$ |
| $f_{\max}$ | maximum processing rate for VM $m$. |
| $\theta_{\text{dyn},m}(t)$ | the *dynamic* energy component of VM $m$ |
| $\theta_{\text{idle},m}(t)$ | idle energy consumed by VM $m$ |
| $\theta_{\max,m}(t)$ | VM $m$ maximum energy drained at max. rate |
| $\lambda_{\max}$ | maximum allocated load per VM |
| $\theta_{\text{idle}}^{\text{TOE}}(t)$ | energy drained by the NIC with no data transfer |
| $Y$ | maximum number of optical drivers |
| $B_{\max}$ | maximum energy buffer capacity |
| $B_{\text{up}}, B_{\text{low}}$ | upper and lower energy buffer thresholds |
| | **Variables** |
| $\theta_{\text{MEC}}(t)$ | total MEC energy at time slot $t$ |
| $\theta_{\text{COMP}}(t)$ | the energy drained due to computation processes |
| $\theta_{\text{COMM}}(t)$ | MEC server's intra-communications energy cost |
| $f_m(t)$ | instantaneous processing rate |
| $\theta_{\text{CPU}}(t)$ | energy drained by active VMs at time slot $t$ |
| $M(t)$ | number of VMs to be active in time slot $t$ |
| $\theta_{\text{SC}}(t)$ | energy drained due to VM switching at time slot $t$ |
| $\theta_{\text{TOE}}(t)$ | TCP/IP offload induced energy at the NIC |
| $\alpha_m(t)$ | load dependent factor |
| $\lambda_m(t)$ | workload allotted to VM $m$ at time slot $t$ |
| $\theta_{\max}^{\text{TOE}}(t)$ | maximum energy drained by the NIC at time slot $t$ |
| $\theta_{\text{VLAN}}(t)$ | energy drained due to the VMs communication links |
| $\theta_{\text{WCOM}}(t)$ | energy drained by optical drivers time slot $t$ |
| $Y(t)$ | number of laser (optical) drivers at time slot $t$ |
| $B(t)$ | energy buffer level in time slot $t$ |
| $H(t)$ | harvested energy profile in time slot $t$ |
| $E(t)$ | purchased grid energy in time slot $t$ |

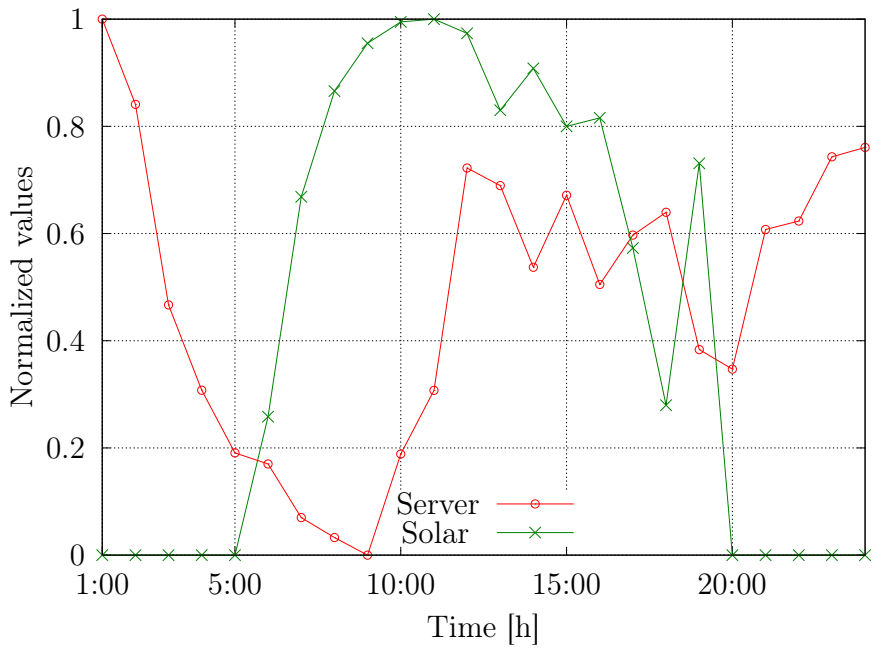



Figure 5.2: Example traces for server workloads and harvested solar energy.

mulated as follows, inspired by [68] and the virtualization knowledge from [20]:

$$\theta_{\mathrm{MEC}}(t) = \theta_{\mathrm{COMP}}(t) + \theta_{\mathrm{COMM}}(t)\,, \tag{5.1}$$

where $\theta_{\mathrm{COMP}}(t)$ is the energy drained due to computation and $\theta_{\mathrm{COMM}}(t)$ due to intra-communications processes in the MEC server at time slot $t$.

**Computing energy**: It is defined as: $\theta_{\mathrm{COMP}}(t) = \theta_{\mathrm{CPU}}(t) + \theta_{\mathrm{SC}}(t) + \theta_{\mathrm{TOE}}(t)$, where $\theta_{\mathrm{CPU}}(t)$ is the energy drained due to the running VMs, w.r.t CPU utilization, and $\theta_{\mathrm{SC}}(t)$ is the energy drained due to VM switching the processing rates $f_m(t) \in [0, f_{\max}]$. $f_{\max}$ [(bit/s)] is the maximum processing rate for VM $m$. $\theta_{\mathrm{TOE}}(t)$ is the energy induced by the TCP/IP offload on the network interface card (NIC, e.g., TCP/IP checksum offload). In practice, the VMs are instantiated on top of the CPU cores and each VM processes the currently allotted task by managing its own local virtualized computing resources, thus we model the processing rates to be between $f_0 = 0$ (represents zero speed of the VM, e.g., deep sleep or shutdown) and $f_{\max}$. Here, we assume that real-time processing of computation workloads is performed in parallel over the VMs interconnected by a power-limited and rate-adaptive switched VLAN.

Considering that $\theta_{\mathrm{CPU}}(t)$ is related to the number of VMs running in time slot $t$, named $M(t) \leq M$, and on the CPU frequency that is allotted to each VM, $\theta_{\mathrm{CPU}}(t)$ is obtained using the linear relationship between the CPU utilization contributed by VM

$m$, and the energy drained is:

$$\theta_{\mathrm{CPU}}(t) = \sum_{m=1}^{M(t)} \theta_{\mathrm{idle},m}(t) + \theta_{\mathrm{dyn},m}(t), \tag{5.2}$$

where $\theta_{\mathrm{idle},m}(t)$ represents the *static* energy drained by VM $m$ in the idle state, and the quantity $\theta_{\mathrm{dyn},m}(t) = \alpha_m(t)(\theta_{\mathrm{max},m}(t) - \theta_{\mathrm{idle},m}(t))$ represents the *dynamic* energy component of VM $m$, where $\alpha_m(t) = (f_m(t)/f_{\mathrm{max}})^2$ [28] is a load dependent factor and $\theta_{\mathrm{max},m}(t)$ is the *maximum* energy that VM $m$ can drain.

Next, we remark that the VM switching cost $\theta_{\mathrm{SC}}(t)$ depends on the frequency reconfiguration, i.e., the transition from $f_1(t)$ (current processing rate for VM $m$) to $f_2(t)$ (the next processing rate), as an example. In short, the energy cost depends on the absolute processing rate gap, $|f_2(t) - f_1(t)|$. Thus, $\theta_{\mathrm{SC}}(t)$ is defined as [68]:

$$\theta_{\mathrm{SC}}(t) = \sum_{m=1}^{M(t)} \kappa_e (f_2(t) - f_1(t))^2, \tag{5.3}$$

where $\kappa_e$ is the the per-VM reconfiguration cost caused by a unit-size frequency switching. Typically, $\kappa_e$ is limited to a few hundreds of mJ per $(\mathrm{MHz})^2$.

At this regard, we put forward the following: at the beginning of time slot $t$, the online resource controller adaptively allocates the available virtual resources and thus determines the VMs demanded, $M(t)$, the workload allotted to VM $m$, denoted by $\lambda_m(t)$, and $f_m(t)$ for VM $m$ that will yield the desired or expected processing time, $\chi_m(t) = \lambda_m(t)/f_m(t)$. Note that $L_{\mathrm{in}}(t) = \sum_{m=1}^{M(t)} \lambda_m(t)$. Moreover, in practical application scenarios, the maximum per-VM computation load to be computed is generally limited up to an assigned value, named $\lambda_{\mathrm{max}}$. Lastly, the VM provisioning and workload allocation is discussed in Section 5.4, and $f_m(t) \triangleq \lambda_m(t)/\Delta$.

Along the same lines of computation, advancement in TCP/IP Offload Engine (TOE) technology enables *partial* computation in the server's NIC [119], i.e., some TCP/IP processing (e.g., checksum computation) is offloaded to a specialized hardware on the network adapter, relieving the host CPU from the overhead of processing TCP/IP. Thus, $\theta_{\mathrm{TOE}}(t)$ is obtained by using the fact that it is data volume dependent and then determined using the workload volume received. Due to the lack of an existing TOE energy model, we rely on the performance measure for the Broadcom (Fibre) 10 Gbps NIC [119], which is considered here as an example of a TCP/IP offload-capable device. Then, $\theta_{\mathrm{TOE}}(t)$ is obtained as:

$$\theta_{\mathrm{TOE}}(t) = \zeta(t)\, \theta_{\mathrm{idle}}^{\mathrm{TOE}}(t) + \theta_{\mathrm{max}}^{\mathrm{TOE}}(t), \tag{5.4}$$

where $\theta_{\mathrm{idle}}^{\mathrm{TOE}}(t) > 0$ is the energy drained by the TOE when powered, with all links connected without any data transfer. This motivates the idea of tuning even the NIC

so that the energy drained is always zero when there is no data transfer. For this, we have $\zeta(t) = (0, 1)$ as the NIC switching status indicator (1 for active state and 0 for idle state). $\theta_{\max}^{\text{TOE}}(t) = \frac{L_{\text{in}}(t)}{\eta}$ is the maximum energy drained by the TOE. $\eta$ is a fixed value measured in [Gbit/J].

**Communication energy**: The communication-related energy consumption within the virtualized MEC server is defined as: $\theta_{\text{COMM}}(t) = \theta_{\text{VLAN}}(t) + \theta_{\text{WCOM}}(t)$, where $\theta_{\text{VLAN}}(t)$ is the energy drained due to the communication links (to-and-from each VM), and $\theta_{\text{WCOM}}(t)$ is the energy drained due to the number of transmission (optical) drivers used for the data transfer to target BS(s).

The energy cost due to communication within the VLAN is obtained by using the Shannon-Hartley exponential analysis. Here, we assume that each VM $m$ communicates with the resource controller through a dedicated reliable link, that operates at the transmission rate of $r_m(t)$ [(bit/s)]. Thus, the energy needed for sustaining the two-way $m^{\text{th}}$ link is defined as, inspired by [120]:

$$\theta_{\text{VLAN}}(t) = 2 \sum_{m=1}^{M(t)} P_m(r_m(t))(\lambda_m(t)/r_m(t)), \tag{5.5}$$

where $P_m(r_m(t)) = \Gamma_m(2^{r_m(t)/W_m} - 1)$ is the power drained by the $m^{\text{th}}$ communication link and $\Gamma_m = \frac{W_m \times N_0^{(m)}}{g_m}$. $N_0^{(m)}$(W/Hz) is the noise spectral power density, $W_m$ is the bandwidth, and $g_m$ is the (non-negative) gain of the $m^{\text{th}}$ link. In practical application scenarios, the maximum per-slot communication rate within the intra-VLAN is generally limited up to an assigned value $r_{\max}$. Thus, the following hard constraint must hold: $\sum_{m=1}^{M(t)} r_m(t) \leq r_{\max}$.

Before proceeding, we consider the *two-way* per-task execution delay ([s]). We have the $m = \{1, \ldots, M(t)\}$ link connection delays, each denoted by $\Omega_m(t) = \lambda_m(t)/r_m(t)$, and $\chi_m(t) \leq \Delta$, where $\Delta$ is the maximum per-slot and per-VM processing time ([s]). At this regard, we note that $\Delta$ is also the server's response time, i.e., the maximum time allowed for processing the total computation load and it is fixed in advance regardless of the task size allocated to VM $m$. Since parallel real-time processing is assumed in this work, the overall communication equates to $2\,\Omega_m(t) + \Delta$. Therefore, the hard per-task delay constraint on the computation time is: $\max\{2\,\Omega_m(t)\} + \Delta \leq \tau_{\max}$, where $\tau_{\max}$ is the maximum tolerable delay, which is fixed in advance.

Finally, $\theta_{\text{WCOM}}(t)$ depends on the number of laser (optical) drivers, named $Y(t) \leq Y$ ($Y$ is the total number of them), that are required for transferring $\ell_y(t) \in L_{\text{out}}(t)$ in time slot $t$ ($\ell_y(t)$ is the downlink traffic volume ([bits] of the driver at slot $t$). $L_{\text{out}}(t)$

is accumulated over a fixed period of time to form a *batch* at the output buffer. At this regard, we note that a large number of drivers yield large transmission speed while at the same time resulting into high energy consumption [21]. Therefore, the energy consumption can be minimized by launching an optimal number of drivers for the data transfer. Moreover, for every *mobile client* who offloaded their task into the MEC server associated with the radio nodes, i.e. BSs, its location and the computation result is known through the UE subscription procedure (i.e., through the LS), thus enabling the location-aware traffic routing and obtaining $Y(t)$.

The energy drained during the data transmission process consists of the following: a constant energy for utilizing each fast tunable driver denoted by $O_{\text{opt},y}(t)$ ([J/s]), the target transmission rate $r_0$ [bits/s] and $L_{\text{out}}(t)$. Thus, the energy is, inspired by [22][121]:

$$\theta_{\text{WCOM}}(t) = \sum_{y=1}^{Y(t)} \frac{O_{\text{opt},y}(t)\, l_y(t)}{r_0}, \tag{5.6}$$

where the parameter $Y(t)$ is obtained using the total number of target BSs as $Y(t) = \left\lceil \frac{1}{\alpha} \cdot (\frac{\omega(t)+1}{\omega(t)})^2 \right\rceil$ (see [22]), where $\omega(t) = \sqrt{\frac{\Upsilon}{\sigma N_{\text{BS}}(t)}}$. $\alpha \in (0, 1]$ is a controllable factor that determines the delay constraint of optical networks, $\sigma$ ([ms]) is the reconfiguration cost for tuning the transceivers, $N_{\text{BS}}(t)$ is an integer value representing the total number of target BSs at time slot $t$, and $\Upsilon$ is the number of time slots at which the computed workload is accumulated at the output buffer. $\alpha, \sigma$, and $\Upsilon$ are fixed values. $L_{\text{out}}(t)$ is equally distributed over the $Y(t)$ drivers.

### 5.2.2 Energy Patterns and Storage

The energy buffer of Fig. 5.1 is characterized by its maximum energy storage capacity $B_{\max}$, and power charging/discharging and leaking losses are not assumed. At each time slot $t$, the EM provides the energy level report to the MEC server, through the pull mode procedure (e.g., FTP [30]), thus the EB level $B(t)$ is known, enabling the provision of the required computation and communication resources, i.e., the VMs and laser drivers.

In this work, the amount of harvested energy $H(t)$ in time slot $t$ is obtained from open-source solar traces within a solar panel farm located in Armenia [102] (*see* green curve in Fig. 5.2), where the dataset time scale matches our time slot duration (1 min). The dataset is the result of daily environmental records for a place assumed to be free from surrounding obstructions (e.g., buildings, shades). In our numerical results, $H(t)$ is obtained by picking one day data from the dataset and then scaling the solar energy

to fit the EB capacity $B_{\max}$ of 490 kJ. Thus, the available EB level $B(t+1)$ at the beginning of time slot $t+1$ is calculated as follows:

$$B(t+1) = B(t) + H(t) - \theta_{\text{MEC}}(t) + E(t), \tag{5.7}$$

where $B(t)$ is the energy level in the battery at the beginning of time slot $t$, $\theta_{\text{MEC}}(t)$ is the energy consumption of the computing platform over time slot $t$, see Eq. (5.1), and $E(t) \geq 0$ is the amount of energy purchased from the power grid. We remark that $B(t)$ is updated at the beginning of time slot $t$ whereas $H(t)$ and $\theta_{\text{MEC}}(t)$ are only known at the end of it.

For decision making by the resource controller, the received EB level reports are compared with the following thresholds: $B_{\text{low}}$ and $B_{\text{up}}$, respectively termed the lower and the upper energy threshold with $0 < B_{\text{low}} < B_{\text{up}} < B_{\max}$. $B_{\text{up}}$ corresponds to the desired energy buffer level and $B_{\text{low}}$ is the lowest EB level that the MEC server should ever reach. The suitable energy source at each time slot $t$ is selected based on the forecast expectations, i.e., the expected harvested energy $\hat{H}(t)$. If $\hat{H}(t)$ is enough to reach $B_{\text{up}}$, no energy purchase is needed. Otherwise, the remaining amount up to $B_{\text{up}}$, i.e., $E(t) = B_{\text{up}} - B(t)$ is bought from the electrical grid. Our optimization framework in the next section makes sure that $B(t)$ never falls below $B_{\text{low}}$ and guarantees that $B_{\text{up}}$ is reached at every time slot.

## 5.3 Problem Formulation

In this section, we formulate an optimization problem and it is defined in subsection 5.3.1.

### 5.3.1 Optimization Problem

Reliable server management procedures are an essential requirement when powering an edge system with green energy. This ensures that the mobile users and telecommunication operators do not experience any service outages. To guarantee this, on per time slot basis, in this work the online controller adaptively schedules the communication and computing resources, at the same time receiving the energy level report from the EM. The goal is to minimize the overall resulting communication-plus-computing energy, i.e., the energy consumption related to the MEC server's VMs and transmission drivers. To achieve this, for $t = 1, \ldots, T$, where $T$ is the optimization horizon, we

define the optimization problem as:

$$
\begin{aligned}
\mathbf{P1} \quad &: \quad \min_{\mathcal{E}} \textstyle\sum_{t=1}^{T} \theta_{\mathrm{MEC}}(t) \\
&\text{subject to:} \\
\mathrm{C1} \quad &: \quad d \leq M(t) \leq M, \\
\mathrm{C2} \quad &: \quad B_{\mathrm{low}} \leq B(t) \leq B_{\max}, \\
\mathrm{C3} \quad &: \quad 0 \leq f_m(t) \leq f_{\max}, \\
\mathrm{C4} \quad &: \quad 0 \leq \lambda_m(t) \leq \lambda_{\max}, \\
\mathrm{C5} \quad &: \quad \chi_m(t) \leq \Delta, \\
\mathrm{C6} \quad &: \quad \textstyle\sum_{m=1}^{M(t)} r_m(t) \leq r_{\max}, \\
\mathrm{C7} \quad &: \quad \max\{2\,\Omega_m(t)\} + \Delta \leq \tau_{\max},
\end{aligned}
\tag{5.8}
$$

where $\mathcal{E} \triangleq \{M(t), \{\alpha_m(t)\}, \{P_m(t)\}, \{\lambda_m(t)\}, \zeta(t), Y(t)\}$ is the set of objective variables to be configured at slot $t$ in the MEC server, for the computing-plus-communication processes. Regarding the constraints, C1 forces the required number of VMs, $M(t)$, to be always greater than or equal to a minimum number $d \geq 1$: the target of this is to be always able to handle mission critical communications. C2 makes sure that the EB level is always above or equal to a preset threshold $B_{\mathrm{low}}$, to guarantee *energy self-sustainability* over time. C3 and C4, bound the maximum processing rate and workloads of each running VM $m$. Constraint C5 represents a hard-limit on the corresponding per-slot and per-VM processing time. Furthermore, C6 bounds the aggregate communication rate sustainable by the VLAN to $r_{\max}$ and C7 forces the server to process the offloaded tasks within the set value $\tau_{\max}$.

From P1, we note that $\theta_{\mathrm{MEC}}(t)$ consists of a non-convex component, i.e., Eq. (5.5), while the others are convex and non-decreasing. Then, Eq. (5.5) can be convexified into a convex function using GP concept [115], by introducing alternative variables and approximations. In this, we introduce fixed parameters (i.e., $\mu_m, \nu_m$) and approximations. Dropping the index $t$ for convenience, we let $r_m = 2\,\lambda_m/(\tau_{\max} - \Delta)$. We obtain $P_m(r_m)$ in terms of $\lambda_m$, by rearranging the Shannon-Hartley expression and substituting the value of $r_m$, as: $\hat{P}_m(r_m) = \frac{((2\,\lambda_m/(\tau_{\max}-\Delta)) - \nu_m\,W_m)\ln 2}{\mu_m W_m} + \ln(N_0^{(m)}) - \ln g_m$. From the Shannon-Hartley expression, we simply observed the presence of the *log-sum-exp* function as it has been proven to be convex in [85] and recall that $P_m(r_m) = \exp(\hat{P}_m(r_m))$.

To solve P1, we leverage the use of LLC [28], GP [115], and heuristics, obtaining the feasible system control inputs $\psi(t) = (M(t), \{\alpha_m(t)\}, \{P_m(t)\}, \{\lambda_m(t)\}, \zeta(t), Y(t))$, that yield the best system behavior within $T$.

## 5.4 Resource Controller Design and Server Management

In this subsection, a server workload and energy harvesting forecasting method, and an online resource management algorithm are proposed to solve the previously stated problem P1. In subsection 5.4.1, we discuss the LSTM neural network used to predict the short-term future server workloads and harvested energy, then in subsection 5.4.2, we solve P1 by using LLC principles, GP theory, and heuristics, and lastly, in subsection 5.4.3 we put forward the ARCES algorithm.

### 5.4.1 Server Workload and Energy Prediction

In order to estimate the system workload over the prediction horizon $T$, we perform time series prediction, i.e., we obtain $T = 3$ estimates of $\hat{\mathrm{L}}(t+1)$ and $\hat{H}(t+1)$, by using an LSTM network developed in Python using TensorFlow deep learning libraries (Keras, Sequential, Dense, LSTM), with a hidden layer of 4 LSTM neurons, and an output layer that makes a single value prediction. The dataset is split as 67% for training and 33% for testing. As for the performance measure of the model, we use the RMSE. In this work, prediction steps similar to Table 3.2 are adopted, and Fig. 5.3 shows the prediction results that will be discussed in subsection 5.5.2.

### 5.4.2 Edge System Dynamics

We denote the system state vector at time $t$ by $u(t) = (M(t), Y(t), B(t))$, which contains the number of active VMs, $M(t)$, transmission drivers, $Y(t)$, and the EB level, $B(t)$. The input vector $\psi(t) = (M(t), \{\alpha_m(t)\}, \{P_m(t)\}, \{\lambda_m(t)\}, \zeta(t), Y(t))$ drives the MEC server behavior (handles the joint VM soft-scaling and the tuning of transmission drivers) at time $t$. Note that $\{P_m^*(t)\}$ is obtained with CVXOPT, and $\{\lambda_m^*(t)\}$ is obtained by following *remark* 1.

The system behavior is described by the discrete-time state-space equation, adopting the LLC principles [27][28]:

$$u(t+1) = \phi(u(t), \psi(t)), \tag{5.9}$$

where $\phi(\cdot)$ is a behavioral model that captures the relationship between $(u(t), \psi(t))$, and the next state $u(t+1)$. Note that this relationship accounts for the amount of energy drained $\theta_{\mathrm{MEC}}(t)$, that harvested $H(t)$ and that purchased from the electrical

grid $E(t)$, which together lead to the next buffer level $B(t+1)$ through Eq. (5.7). The online resource management algorithm, ARCES, finds the best control action vector that yields the desired energy savings within the computing environment. Specifically, for each time slot $t$, problem Eq. (5.8) is solved, obtaining control actions for the prediction horizon $T$. The control action that is applied at time $t$ is $\psi^*(t)$, which is the first one in the retrieved control sequence. This control amounts to setting the number of instantiated VMs, $M^*(t)$ (along with their obtained $\{\alpha_m^*(t)\}$, $\{P_m^*(t)\}$, $\{\lambda_m^*(t)\}$ values), NIC status to either active or not, $\zeta^*(t) \in (0,1)$, and the optimal transmission drivers, $Y^*(t)$. The entire process is repeated every time slot $t$ when the controller can adjust the behavior given the new state information.

Since the actual values for the system input cannot be measured until the next time instant when the controller adjusts the system behavior, the corresponding system state for $t+1$ can only be estimated as:

$$\hat{u}(t+1) = \phi(u(t), \psi(t))\,. \tag{5.10}$$

For these estimations we use the forecast values of load $\hat{L}_{\text{in}}(t)$ and harvested energy $\hat{H}(t)$, from the LSTM forecasting module.

**Remark 1 (VM provisioning and load distribution):** a remark on the provisioned VMs at slot $t$, $M(t)$, is in order. The number of active VMs depends on the forecasted server workload, $\hat{L}_{\text{in}}(t+1)$, and each VM can compute an amount of up to $\lambda_{\max}$ (considering that virtualization technologies specify the minimum and maximum amount of resources that can be allocated per VM [122]). Then, the projected number of VMs that shall be active in slot $t$ to serve the forecasted server workloads is hereby obtained as: $M(t) = \left\lceil (\hat{L}_{\text{in}}(t+1)/\lambda_{\max}) \right\rceil$, where $\lceil \cdot \rceil$ returns the nearest upper integer. We heuristically split the workload among VMs by allocating a workload $\lambda_m(t) = \lambda_{\max}$ to the first $M(t)-1$ VMs, $m = 1, \ldots, M(t)-1$, and the remaining workload $\lambda_m(t) = \hat{L}_{\text{in}}(t+1) - (M(t)-1)\lambda_{\max}$ to the last one. This load distribution is motivated by the shares feature [122] that is inherent in virtualization technologies. This enables the resource scheduler to efficiently distribute resources amongst contending VMs, thus guaranteeing the completion of the computation process within the expected time.

**Algorithm 3:** ARCES Pseudocode

**Input:** $u(t)$ (current state)

**Output:** $\psi^*(t)$ (control input vector)

01: Parameter initialization

$\mathcal{S}(t) = \{u(t)\}$

02: **for** ($n$ within the prediction horizon of depth $T$) **do**

- $\hat{L}_{\text{in}}(t+n)$:= forecast the workload

- $\hat{\text{H}}(t+n)$:= forecast the energy

- $\mathcal{S}(t+n) = \emptyset$

03: **for** (each $u(t)$ in $\mathcal{S}(t+n)$) **do**

- generate all reachable states $\hat{u}(t+n)$

- $\mathcal{S}(t+n) = \mathcal{S}(t+n) \cup \{\hat{u}(t+n)\}$

04: **for** (each $\hat{u}(t+n)$ in $\mathcal{S}(t+n)$) **do**

- calculate the corresponding $\theta_{\text{MEC}}(\hat{u}(t+n))$

**end for**

**end for**

**end for**

05: - obtain a sequence of reachable states yielding minimum energy cost

06: $\psi^*(t)$ := control leading from $u(t)$ to $\hat{u}_{\min}$

07: **Return** $\psi^*(t)$

### 5.4.3 The ARCES Algorithm

In order to obtain the best control action that will adjust the computing system behavior at time $t$, with negligible computational overhead, the controller explores the prediction horizon of comprising discrete states and comes up with the feasibility action set, and from it the control input that yields the minimum energy cost is selected as $\psi^*(t) = (M^*(t), \{\alpha_m^*(t)\}, \{P_m^*(t)\}, \{\lambda_m^*(t)\}, \zeta^*(t), Y^*(t))$.

The algorithm pseudocode is outlined in Algorithm 3 and it follows the technique from [28]. Starting from the *initial state*, the controller constructs, in a breadth-first fashion, a tree comprising all possible future states up to the prediction depth $T$. The algorithm proceeds as follows: A search set $\mathcal{S}$ consisting of the current system state is initialized (line 01), and it is accumulated as the algorithm traverse through the tree (line 03), accounting for predictions, accumulated workloads at the output buffer, past outputs and controls. The set of states reached at every prediction depth $t+n$ is referred to as $\mathcal{S}(t+n)$ (line 02). Given $u(t)$, we first estimate the workload $\hat{L}_{\text{in}}(t+n)$ and harvested energy $\hat{H}(t+n)$ (line 02), and generate the next set of reachable control actions by applying the input workload and energy harvested (line 03). The energy cost function corresponding to each generated state $\hat{u}(t+n)$ is then computed (line 04). Once the prediction horizon is explored, a sequence of reachable states yielding minimum energy consumption is obtained (line 05). The control action $\psi^*(t)$ corresponding to $\hat{u}(t+n)$ (the first state in this sequence) is provided as input to the system while the rest are discarded (line 06). The process is repeated at the beginning of each time slot $t$.

## 5.5 Performance Evaluation

This section present some selected numerical results of the ARCES algorithm for real server workloads. The parameters that were used for the simulations are listed in Table 5.2.

### 5.5.1 Simulation Setup

As one of the MEC deployment scenarios, we assume that the MEC server is placed at an aggregation point where BSs in proximity can offload their computation workload following the random real-valued arrival process. Our time slot duration $\tau$ is set to 1 min and the time horizon is set to $T = 3$ time slots.

### 5.5.2 Numerical Results

In Fig. 5.3, we show real and predicted values for the server workloads (Server) and harvested energy (Solar) over time. We track the one-step predictive mean value at each step of the online forecasting routine. The obtained average prediction error (RMSE) for the server workloads and harvested energy processes, both normalized in [0,1] for

Table 5.2: System Parameters.

| **Parameter** | **Value** |
|---|---|
| Max. number of VMs, $M$ | 10 |
| Min. number of VMs, $d$ | 1 |
| Time slot duration, $\tau$ | 1 min |
| Idle state energy for VM $m$, $\theta_{\text{idle},m}(t)$ | 10 J |
| Max. energy for VM $m$, $\theta_{\max,m}(t)$ | 60 J |
| per-VM reconfiguration cost, $\kappa_e$ | 0.005 J/(MHz)$^2$ |
| TOE in idle state, $\theta_{\text{idle}}^{\text{TOE}}(t)$ | 13.1J |
| Max. allowed processing time, $\Delta$ | 0.8 s |
| Processing rate set, $\{f_m(t)\}$ | $\{0, 50, 70, 90, 105\}$ |
| Bandwidth, $W_m$ | 1 MHz |
| Max. number of drivers, $Y$ | 6 |
| Max. tolerable delay, $\tau_{\max}$ | 2 s |
| Noise spectral density, $N_0^{(m)}$ | -174 dBm/Hz |
| Max. VM $m$ load, $\gamma^{\max}$ | 5 Mbit |
| Driver energy, $O_{\text{opt},y}(t)$ | 1 J/s |
| Target transmission rate, $r_0$ | 1 Mbps |
| Energy storage capacity, $\beta_{\max}$ | 490 kJ |
| Lower energy threshold, $\beta_{\text{low}}$ | 30% of $\beta_{\max}$ |
| Upper energy threshold, $\beta_{\text{up}}$ | 70% of $\beta_{\max}$ |

$T \in \{1, 2, 3\}$, are $L_{\text{in}}(t) = \{0.017, 0.019, 0.021\}$ and $H(t) = \{0.038, 0.039, 0.039\}$. Note that the predictions for $L_{\text{in}}(t)$ are more accurate than those of $H(t)$ (confirmed by comparing the average RMSE), due to differences in the used dataset granularity. However, the measured accuracy is deemed good enough for the proposed optimization

Our online server management algorithm (ARCES) is benchmarked with another one, named Iterative-based Resource Scheduler (IRS), which is inspired by the iterative

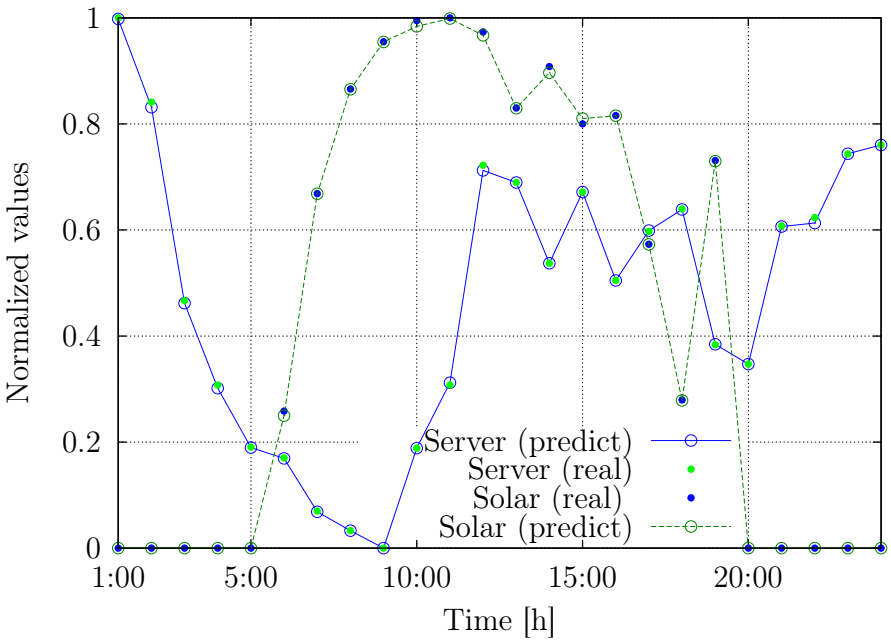



Figure 5.3: Forecast mean value for $L(t)$ and $H(t)$.

approach from [68]. In IRS, the optimum computing-plus-communication parameters are obtained in an iterative manner: at the end of each cycle, convergence conditions are used to determine if the found solution is acceptable or the optimization process should continue. The observed conditions are as follows: (i) to ensure that the total load $L_{\text{in}}(t)$ has been fully allocated with accuracy $\frac{\sum_{m=1}^{M(t)} \lambda_m(t) - L_{\text{in}}(t)}{L_{\text{in}}(t)} \leq \epsilon$, with $\epsilon = 0.01$; (ii) to verify if the selected working rate $f_m(t)$ is able to cope with the input load $L_{\text{in}}(t)$, guaranteeing that the computation processing time is within the server's response time limit $\Delta$, i.e., $L_{\text{in}}(t) \leq f_m(t)\Delta$ . The average energy savings obtained by ARCES are

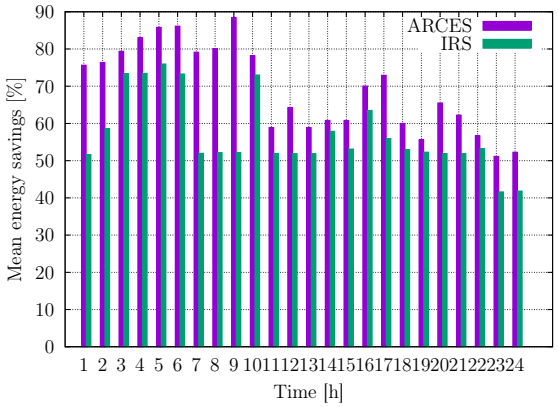



Figure 5.4: Mean energy savings within the MEC server.

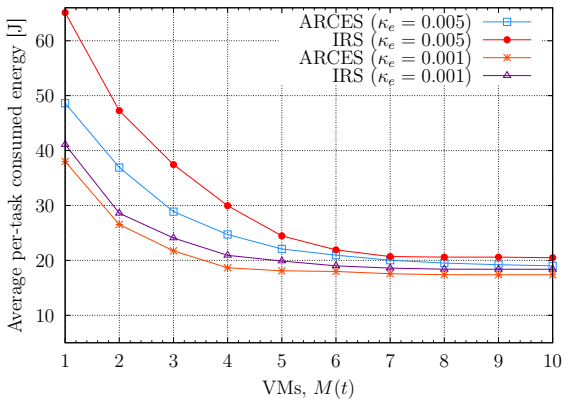



Figure 5.5: Average per-task consumed energy vs VMs.

shown in Fig. 5.4. The average results for ARCES ($\kappa_e = 0.005, \Gamma_m = 0.5\,\text{mW}, \eta = 1.4$ Gbit/J) show energy savings of 69%, while IRS achieves 56% on average, in both cases with respect to the case where no energy management procedures are applied; i.e., the MEC server provisions the computing resources for maximum expected computation workload (maximum value of $\theta_{\text{MEC}}(t)$, with $M = 10$, $\forall t$). As expected, the highest energy savings peak is observed at 9 h as the aggregate computation requests/workload was at its lowest with an expected increase in the computation workload and harvested solar energy in the near future. The effectiveness of the joint VM soft-scaling and tuning of transmission drivers, coupled with foresighted optimization is observed in the obtained numerical results.

In Fig. 5.5, we show the effects of the per-VM reconfiguration cost on $\theta_{\text{MEC}}(t)$ at $\kappa_e = \{0.001, 0.005\}$ ($\Gamma_m = 0.5\,\text{mW}, \eta = 1.4$ Gbit/J), taking into account the performance of ARCES when compared with IRS for $M(t) = 1, \ldots, M$. It can be observed that $\theta_{\text{MEC}}(t)$ increases with large $\kappa_e$ only for small values of $M(t)$, and as $M(t)$ increases the energy consumption decreases for large $\kappa_e$. Moreover, ARCES leads to an energy consumption reduction with respect IRS from 25% to 7% (case of $\kappa_e = 0.005$) and from 7% to 5% (case of $\kappa_e = 0.001$). When ARCES is compared with the case where no energy management is applied (maximum value of $\theta_{\text{MEC}}(t)$, with $M = 10$, $\forall t$), the obtained energy reduction ranges from 45% to 31% (case of $\kappa_e = 0.005$) and from 25% to 21% (case of $\kappa_e = 0.001$). These numerical results confirm that jointly autoscaling the available computing-plus-communication resources within the computing platform provides remarkable energy savings. These results conforms to our expectations from [120].

## 5.6 Conclusion

In this summary, we have envisioned a hybrid-powered MEC server placed in proximity to a BS cluster for handling the offloaded computation workload. Moreover, the use of green energy promotes energy self-sustainability within the network. We have considered a computing-plus-communication energy model, within the *MEC* paradigm, and then have put forward a combination of a traffic engineering- and MEC Location Service-based online server management algorithm with *EH* capabilities, called Automated Resource Controller for Energy-aware Server (ARCES), for autoscaling and reconfiguring the computing-plus-communication resources. The main goal is to minimize the overall energy consumption, under hard per-task delay constraints (i.e.,

QoS). ARCES jointly performs (i) a short-term server demand and harvested solar energy forecasting, (ii) VM soft-scaling, workload and processing rate allocation and lastly, (iii) switching on/off of transmission drivers (i.e., *fast* tunable lasers) coupled with the location-aware traffic scheduling. Numerical results, obtained with real-world energy and server workload traces, demonstrate that the proposed algorithm (ARCES) achieves energy savings of 69%, on average, with an energy consumption ranging from 31%-45% at high per-VM reconfiguration cost and from 21%-25% at low per-VM reconfiguration cost, with respect to the case where no energy management techniques are applied.

# Chapter 6
# Conclusions

## 6.1 Summary

This final chapter draws the conclusion of the entire dissertation. The common thread which connects all the chapters is certainly the strong emphasis on the use of green energy for purposes of energy self-sustainability, forecasting and foresighted optimization, together with BS sleep mode, VM soft-scaling, tuning of transmission drivers, for making ES decisions in a forward-looking fashion. In addition, the use of online algorithms is adopted and we have used some of the common mathematical tools.

The aim of the thesis was to put forward research on dynamic resource management procedures for purposes of energy savings and QoS guarantee, within the MEC paradigm. Towards this, different MEC deployment scenarios suggested by ETSI and mobile operators were considered as our system model examples. To improve energy savings and guarantee QoS, the LSTM neural network is used for forecasting the traffic load and harvested energy, and control-theoretic techniques, together with heuristics, are used for obtaining the best system control input, per time instance. The obtained numerical results uncover the capabilities of MN traffic load and energy traces towards the development of traffic-oriented edge network management solutions where the combination of forecasting and foresighted optimization help in obtaining the control actions that drives the system towards the desired system behavior.

In Chapter 3, the integration of MEC and EH BS for computing and communication services provision in remote/rural areas is pursued. Here, an off-grid BS empowered with computation capabilities is considered and its is fully powered by green energy (solar and wind). For data communication from the remote site to the remote cloud, the system uses a microwave backhaul. Towards energy savings and QoS guarantee, we dynamically provision computing and communication resources using an online

LLC-based algorithm that use the forecasted traffic load and harvested energy as input.

Chapter 4 was devoted to the development of a robust distributed algorithm for managing a group of EH BSs empowered with computation capabilities. Towards this goal, we consider a hybrid edge computing architecture where computing servers are co-located with each BS, and a centralized controller (placed at a point within range to a set of BSs) is utilized to manage them, deciding upon the allocation of their computing and transmission resources. To manage the communication sites, the controller partitions the BSs into clusters based on their location; then, for each cluster it performs online supervisory control by forecasting the traffic load and the harvested energy using a LSTM neural network, then employ foresighted optimization towards obtaining the control inputs that will drive the edge system towards the desired system behavior, per time instance.

Finally, in Chapter 5, we addressed the shortcomings of the MEC server energy consumption model and then presented a traffic engineering- and MEC Location Service-based online server management algorithm with EH capabilities. The computing platform is energized by solar (main supply) and power grid (back up). To improve energy savings, we provisioned the computing and communication resources, i.e., VMs and transmission drivers, using the forecasted server workload and harvested energy, and clients current location information. The online algorithm uses the LLC principles and heuristics towards obtaining the desired edge system control inputs, per time instance.

## 6.2 Future Research Directions

As this is a relatively new research area, there are many possible future research directions and they are outlined as follows.

**Container-based virtualization within a MEC server**: The thesis is based on the use of VMs within computing platforms. However, due to the ascendency of containers and the enormous effort that open-source communities have made to continually improve their management frameworks over the years, the consideration of containers as computing resources needs to be considered. In addition, their energy consumption needs to be quantified. The use of containers in virtualized computing platforms will reduce the energy drained as they produce lower overheads than VMs. The combination of container-based virtualization and green-based load balancing is an open problem that needs to investigated towards energy savings.

**Green-based traffic load balancing**: As EH technologies advance, powering the

edge apparatuses with green energy (e.g., solar and/or wind) is a promising solution to save on-grid power due to their location, reliability, carbon footprint and cost. To fully utilize the harvested energy, it is desirable to incorporate the green energy utilization as a *performance metric* in traffic load balancing strategies, instead of using the traffic load (e.g., network impact). With green-based traffic loading, MNs may enable BSs with sufficient green energy to serve more traffic while reducing the traffic loads of BSs consuming on-grid power. This problem is still open as the obtained energy savings can be compared with the work done in Chapter 4.

**Alternative forecasting methods**: The research work found in this thesis make use of the LSTM neural network for forecasting the traffic load and harvested energy. To observe its performance the RMSE is used. Since forecasting requires accurate predictions, therefore, there is a need for another forecasting method, e.g., CNN for multi-step time series forecasting, in order to determine the most accurate method. This problem is still open as CNN is mostly used for handling image recognition.

**Distributed control for densely-deployed EH BSs**: In Chapter 4, we have considered an environment where a centralized entity, an edge controller, that manages a group of BSs empowered with computation capabilities. Even though the energy minimization problem was solved in a distribute manner, there exists a gap where we consider BSs that cooperates using distributed algorithms (no centralized entity). Here, the BSs act as multi-agents that interact with each other over an information exchange network.

**Computation peer offloading for energy-constraint edge system**: The research presented in this thesis assumed the ability compute the offloaded workload within a single BS site (BS co-located with MEC server). In order to handle uneven computation workloads in the network when some of the BS sites are on sleep mode, cooperation among BSs via workload peer offloading can be exploited to avoid computational delays at overloaded BSs, as computing resources are always limited. This problem is still open for investigation using the scenario of Chapter 4 and 5 in order to maximize the long-term system wide performance (i.e., minimizing latency) while taking into account the available energy in the EB.

**Content caching**: Concerning the work of Chapter 5, we have assumed the computing platform is cache-enabled and neglecting the energy cost contributed by this process. This requires special attention in order to have a complete computing-plus-communication energy cost within a cache-enabled MEC server. Thus, the energy consumption problem within the server needs to be further investigated.

# List of Publications

This thesis summarizes the work developed over three years where the main outcomes are the following articles.

**Journal papers:**

**[J1] T. Dlamini**, A. F. Gambin, D. Munaretto, M. Rossi, "Online Supervisory Control and Resource Management for Energy Harvesting BS Sites Empowered with Computation Capabilities", *Journal on Wireless Communication and Mobile Computing*, vol. 2019, February 2019.

**[J2] T. Dlamini**, M. Rossi, and D. Munaretto, "Softwarization of Mobile Network Functions towards Agile and Energy Efficient 5G Architectures: A Survey", *Journal on Wireless Communication and Mobile Computing*, vol. 2017, November 2017.

**Conference papers:**

**[C1] T. Dlamini**, A. F. Gambin, "Adaptive Resource Management for a Virtualized Computing Platform within Edge Computing", in *IEEE International Conference on Sensing, Communication and Networking (SECON) workshop on Edge Computing for Cyber Physical Systems (CyberEdge)*, Boston, MA, USA: IEEE, June 2019, [Online]. Available: arXiv:1906.05008.

**[C2] T. Dlamini**, A. F. Gambin, D. Munaretto, M. Rossi, "Online Resource Management in Energy Harvesting BS Sites through Prediction and Soft-Scaling of Computing Resources", in *Proceedings of the 2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, Bologna, Italy: IEEE, September 2018.

**Book Chapter:**

**[B1] T. Dlamini**, "Softwarization in Future Mobile Networks and Energy Efficient Networks", Mobile Computing, IntechOpen, London, UK, November 2019

# References


[1] "Virtualization for small cells: Overview," Small Cell Forum, Draycott, England, Tech. Rep., Jun 2015.

[2] T. Dlamini, M. Rossi, and D. Munaretto, "Softwarization of Mobile Network Functions towards Agile and Energy Efficient 5G Architectures: A Survey," *Wireless Communications and Mobile Computing*, 2017.

[3] J. Xin, L. E. Li, V. Laurent, and J. Rexford, "Softcell: Scalable and flexible cellular core network architecture," in *ACM Conference on Emerging Networking Experiments and Technologies*, Santa Barbara, USA, Dec. 2013.

[4] H. Hawilo, A. Shami, M. Mirahmadi, and R. Asal, "NFV: state of the art, challenges, and implementation in next generation mobile networks (vEPC)," *IEEE Network*, vol. 28, no. 6, pp. 18–26, 2014.

[5] C. Yang, Z. Chen, B. Xia, and J. Wang, "When ICN meets C-RAN for HetNets: an SDN approach," *IEEE Communications Magazine*, vol. 53, no. 11, pp. 118–125, 2015.

[6] 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, 2014.

[7] "Unlock New Business Opportunities With vEPC," VMware, USA, Tech. Rep., Sep. 2016.

[8] "5G Mobile Communications Systems for 2020 and beyond," Fifth generation mobile promotion forum, Japan, Tech. Rep., Jul. 2016.

[9] A. Nakao, D. Ping, and K. Masayuki, "Flare: Deeply programmable network node architecture." EU-Japan Collaboration Project, University of

Tokyo, Japan. [Online]. Available: http://www.5gsummit.org/berlin/docs/slides/Aki-Nakao.pdf.

[10] "M-CORD Open Reference Solution Paves the Way for 5G Innovation. [Online]. Available: http://opencord.org/tag/m-cord/."

[11] A. Nakao, "Software-defined data plane enhancing SDN and NFV," *IEICE Transactions on Communications*, vol. 98, no. 1, pp. 12–19, 2015.

[12] S. Kekki, W. Featherstone, Y. Fang, P. Kuure, A. Li, A. Ranjan, D. Purkayastha, F. Jiangping, D. Frydman, G. Verin, K. Wen, K. Kim, R. Arora, A. Odgers, L. M. Contreras, and S. Scarpina, "MEC in 5G Networks," ETSI, Sophia-Antipolis, France, Tech. Rep., Jun 2018.

[13] M. Patel, Y. Hu, P. Hédé, J. Joubert, C. Thornton, B. Naughton, J. R. Ramos, C. Chan, V. Young, S. J. Tan, D. Lynch, N. Sprecher, T. Musiol, C. Manzanares, U. Rauschenbach, S. Abeta, L. Chen, K. Shimizu, A. Neal, P. Cosimini, A. Pollard, and G. Klas, "Mobile edge computing introductory technical white paper," ETSI, Sophia-Antipolis, France, Tech. Rep., Sep 2014.

[14] "Cisco visual networking index: Forecast and Methodology, 2017–2022," Cisco, Tech. Rep., Mar. 2019.

[15] B. Gabriel, "Mobile edge computing Use Cases and Deployment Options," Heavy Reading, Tech. Rep., Jul. 2016.

[16] "Five Trends to Small Cells 2020," Huawei Technologies, Helsinki, Finland, Tech. Rep., Feb 2016.

[17] N. Bhushan, J. Li, D. Malladi, R. Gilmore, D. Brenner, A. Damnjanovic, R. Sukhavasi, C. Patel, and S. Geirhofer, "Network densification: the dominant theme for wireless evolution into 5G," *IEEE Communications Magazine*, vol. 52, no. 2, pp. 82–89, 2014.

[18] R. Morabito, "Power Consumption of Virtualization Technologies: An Empirical Investigation," in *IEEE International Conference on Utility and Cloud Computing (UCC)*, Limassol, Cyprus, Dec. 2015.

[19] Y. Jin, Y. Wen, and Q. Chen, "Energy efficiency and server virtualization in data centers: An empirical investigation," in *IEEE Conference on Computer Communications Workshops (INFOCOM Workshops)*, Orlando, USA, Mar. 2012.

[20] M. Portnoy, *Virtualization essentials*. John Wiley & Sons, 2012.

[21] B. Wu, S. Fu, X. Jiang, and H. Wen, "Joint Scheduling and Routing for QoS Guaranteed Packet Transmission in Energy Efficient Reconfigurable WDM Mesh Networks," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 8, pp. 1533–1541, 2014.

[22] S. Fu, H. Wen, J. Wu, and B. Wu, "Cross-Networks Energy Efficiency Tradeoff: From Wired Networks to Wireless Networks," *IEEE Access*, vol. 5, pp. 15–26, 2017.

[23] D. Zordan, M. Miozzo, P. Dini, and M. Rossi, "When telecommunications networks meet energy grids: cellular networks with energy harvesting and trading capabilities," *IEEE Communications Magazine*, vol. 53, no. 6, pp. 117–123, 2015.

[24] R. Hyndman and G. Athanasopoulos, *Forecasting: principles and practice*. OTexts: Melbourne, Australia, 2013.

[25] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[26] D. Kusic, J. O. Kephart, J. E. Hanson, N. Kandasamy, and G. Jiang, "Power and Performance Management of Virtualized Computing Environments Via Lookahead Control," in *International Conference on Autonomic Computing*, Chicago, USA, Jun. 2008.

[27] S. Abdelwahed, N. Kandasamy, and S. Neema, "Online control for self-management in computing systems," in *IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, Ontario, Canada, May 2004.

[28] J. P. Hayes, "Self-Optimization in Computer Systems via On-Line Control: Application to Power Management," in *Proceedings of the First International Conference on Autonomic Computing*, Washington,USA, May 2004.

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

[30] "Charging Data Rececord (CDR) file format and transfer (3GPP TS 32.2.297)," ETSI, Sophia-Antipolis, France, Tech. Rep., Aug 2016.

[31] G. Lu, C. Guo, Y. Li, Z. Zhou, T. Yuan, H. Wu, Y. Xiong, R. Gao, and Y. Zhang, "ServerSwitch: A Programmable and High Performance Platform for Data Center Networks," in *USENIX conference on Networked systems design and implementation*, Boston, USA, Apr. 2011.

[32] B. A. A. Nunes, M. Mendonca, X. N. Nguyen, K. Obraczka, and T. Turletti, "A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks," *IEEE Communications Surveys Tutorials*, vol. 16, no. 3, pp. 1617–1634, 2014.

[33] "Software-Defined and Cloud-Native Foundations for 5G Networks," InterDigital, Denver, USA, Tech. Rep., May 2019.

[34] M. Ciosi, D. Clarke, C. Cui, J. Benitez, U. Mivhel, K. Ogaki, and M. Fukui, "Introductory White Paper: Network Functions Virtualization," in *SDN and OpenFlow World congress*, Darmstadt, Germany, Jun. 2012.

[35] "Network Functions Virtualisation (NFV): Hypervisor Domain," ETSI, Sophia-Antipolis, France, Tech. Rep., Jan 2015.

[36] M. Nelson, B.-H. Lim, and G. Hutchins, "Fast Transparent Migration for Virtual Machines," in *Proceedings of the Annual Conference on USENIX Annual Technical Conference*, Berkeley, USA, Apr. 2005.

[37] L. Haikun, X. Cheng-Zhong, J. Hai, G. Jiayu, and L. Xiaofei, "Performance and Energy Modeling for Live Migration of Virtual Machines," in *Proceedings of the 20th International Symposium on High Performance Distributed Computing*, California, USA, Jun. 2011.

[38] "Docker for the Virtualization Admin," Docker Inc., San Francisco, USA, Tech. Rep. [Online]. Available: https://www.docker.com/what-container

[39] A. Madhavapeddy and D. J. Scott, "Unikernels: Rise of the Virtual Library Operating System," *Queue*, vol. 11, no. 11, pp. 30–44, 2013.

[40] R. Morabito, V. Cozzolino, A. Y. Ding, N. Beijar, and J. Ott, "Consolidate IoT Edge Computing with Lightweight Virtualization," *IEEE Network*, vol. 32, no. 1, 2018.

[41] "Vision on Software Networks and 5G SN WG," The 5G Infrastructure Public Private Partnership (5G-PPP) consortium, Portugal, Tech. Rep., Jan. 2017.

[42] "A Universal Access Gateway for Fixed and Mobile Network Integration," COnvergence of fixed and Mobile BrOadband access/aggregation networks (COMBO) consortium, France, Tech. Rep., Apr. 2016.

[43] A. Bousia, E. Kartsakli, A. Antonopoulos, L. Alonso, and C. Verikoukis, "Multi-objective auction-based switching-off scheme in heterogeneous networks: To bid or not to bid?" *IEEE Transactions on Vehicular Technology*, vol. 65, no. 11, pp. 9168–9180, 2016.

[44] B. Z. Dongsheng Han and Z. Chen, "Sleep Mechanism of Base Station Based on Minimum Energy Cost," *Wireless Communications and Mobile Computing*, vol. 2018, 2018.

[45] Y. Zhu, Z. Zeng, T. Zhang, L. An, and L. Xiao, "An energy efficient user association scheme based on cell sleeping in LTE heterogeneous networks," in *International Symposium on Wireless Personal Multimedia Communications (WPMC)*, Sydney, Australia, Sep. 2014.

[46] Y. Yuan and P. Gong, "A QoE-orientated base station sleeping strategy for multi-services in cellular networks," in *International Conference on Wireless Communications & Signal Processing (WCSP)*, Nanjing, China, Oct. 2015.

[47] F. Han, Z. Safar, and K. R. Liu, "Energy-efficient base-station cooperative operation with guaranteed QoS," *IEEE Transactions on Communications*, vol. 61, no. 8, pp. 3505–3517, 2013.

[48] A. Bousia, A. Antonopoulos, L. Alonso, and C. Verikoukis, "Green distance-aware base station sleeping algorithm in LTE-Advanced," in *IEEE International Conference on Communications (ICC)*, Ottawa, Canada, Jun. 2012.

[49] S. Cai, L. Xiao, H. Yang, J. Wang, and S. Zhou, "A cross-layer optimization of the joint macro and picocell deployment with sleep mode for green communications," in *IEEE Wireless and Optical Communication Conference (WOCC)*, Chongqing, China, May 2013.

[50] Y. Zhu, Z. Zeng, T. Zhang, and D. Liu, "A QoS-Aware Adaptive Access Point Sleeping in Relay Cellular Networks for Energy Efficiency," in *IEEE Vehicular Technology Conference (VTC Spring)*, Seoul, Korea, May 2014.

[51] L. Chen, S. Zhou, and J. Xu, "Energy Efficient Mobile Edge Computing in Dense Cellular Networks," in *IEEE International Conference on Communications (ICC)*, Paris, France, May 2017.

[52] M. J. Neely, "Stochastic Network Optimization with Application to Communication and Queueing Systems," *Synthesis Lectures on Communication Networks*, vol. 3, pp. 1–211, 2010.

[53] C. Robert and G. Casella, *Monte Carlo statistical methods*. Springer Science and Business Media, 2013.

[54] H. Tabassum, U. Siddique, E. Hossain, and M. J. Hossain, "Downlink performance of cellular systems with base station sleeping, user association, and scheduling," *IEEE Transactions on Wireless Communications*, vol. 13, no. 10, pp. 5752–5767, 2014.

[55] H. Zhang, J. Cai, and X. Li, "Energy-efficient base station control with dynamic clustering in cellular network," in *IEEE International Conference on Communications and Networking (CHINACOM)*, Guilin, China, Aug. 2013.

[56] S. Samarakoon, M. Bennis, W. Saad, and M. Latva-aho, "Dynamic Clustering and ON/OFF Strategies for Wireless Small Cell Networks," *IEEE Transactions on Wireless Communications*, vol. 15, no. 3, pp. 2164–2178, 2016.

[57] E. Oh, K. Son, and B. Krishnamachari, "Dynamic Base Station Switching-On/Off Strategies for Green Cellular Networks," *IEEE Transactions on Wireless Communications*, vol. 12, no. 5, pp. 2126–2136, 2013.

[58] A. Antonopoulos, E. Kartsakli, A. Bousia, L. Alonso, and C. Verikoukis, "Energy-efficient infrastructure sharing in multi-operator mobile networks," *IEEE Communications Magazine*, no. 5, pp. 242–249, 2015.

[59] M. Oikonomakou, A. Antonopoulos, L. Alonso, and C. Verikoukis, "Evaluating cost allocation imposed by cooperative switching off in multi-operator shared HetNets," *IEEE Trans. Veh. Technol*, 2017.

[60] E. Oh, B. Krishnamachari, X. Liu, and Z. Niu, "Toward dynamic energy-efficient operation of cellular network infrastructure," *IEEE Communications Magazine*, vol. 49, no. 6, 2011.

[61] "Charging management; Charging Data Record (CDR) parameter description (3GPP TS 32.2 .298 version 13.5.0 Release 13)," ETSI, France, Tech. Rep., Oct. 2016.

[62] J. Erman and K. K. Ramakrishnan, "Understanding the super-sized traffic of the super bowl," in *Proceedings of the 2013 conference on Internet measurement conference*, Barcelona, Spain, Oct. 2013.

[63] J. Rubio, A. Pascual-Iserte, J. del Olmo Alòs, and J. Vidal, "Dynamic base station switch on/off strategies for sustainable wireless networks," in *IEEE International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, Toronto, Canada, Jun. 2014.

[64] T. Han and N. Ansari, "A traffic load balancing framework for software-defined radio access networks powered by hybrid energy sources," *IEEE/ACM Transactions on Networking*, vol. 24, no. 2, pp. 1038–1051, 2016.

[65] M. D'Amours, A. Girard, and B. Sansò, "Planning Solar in Energy-managed Cellular Networks," 2018. [Online]. Available: http://arxiv.org/abs/1809.07835

[66] J. Xu and S. Ren, "Online Learning for Offloading and Autoscaling in Renewable-Powered Mobile Edge Computing," in *IEEE Global Communications Conference (GLOBECOM)*, Washington, USA, Dec. 2016.

[67] M. Shojafar, N. Cordeschi, and E. Baccarelli, "Energy-efficient Adaptive Resource Management for Real-time Vehicular Cloud Services," *IEEE Transactions on Cloud Computing*, 2016.

[68] M. Shojafar, N. Cordeschi, D. Amendola, and E. Baccarelli, "Energy-saving adaptive computing and traffic engineering for real-time-service data centers," in *EEE International Conference on Communication Workshop (ICCW)*, London, UK, Jun. 2015.

[69] B. Guenter, N. Jain, and C. Williams, "Managing cost, performance, and reliability tradeoffs for energy-aware server provisioning," in *Proceedings IEEE INFOCOM*, Shanghai, China, Apr. 2011.

[70] 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, 2012.

[71] R. Nathuji and K. Schwan, "VirtualPower: coordinated power management in virtualized enterprise systems," in *ACM SIGOPS symposium on Operating systems principles*, Washington, USA, Oct. 2007.

[72] "ETSI GS MEC: Mobile Edge Computing (MEC); Framework and Reference Architecture," ETSI, France, Tech. Rep., Mar. 2016.

[73] "Study on Energy Efficiency Aspects of 3GPP (3GPP TR 21.866 - Release 14)," ETSI, France, Tech. Rep., Sep. 2016.

[74] W. Xu, X. Zhu, S. Singhal, and Z. Wang, "Predictive Control for Dynamic Resource Allocation in Enterprise Data Centers," in *IEEE/IFIP Network Operations and Management Symposium NOMS*, San Jose, USA, Apr. 2006.

[75] J. Maciejowski, *Predictive Control with Constraints*. Prentice Hall, 2002.

[76] S.-L. Chung, S. Lafortune, and F. Lin, "Limited lookahead policies in supervisory control of discrete event systems," *IEEE Transactions on Automatic Control*, vol. 37, pp. 1921–1935, 1992.

[77] T. Ergen and S. S. Kozat, "Online Training of LSTM Networks in Distributed Systems for Variable Length Data Sequences," *IEEE Transactions on Neural Networks and Learning Systems*, 2017.

[78] A. Ferdowsi, U. Challita, and W. Saad, "Deep Learning for Reliable Mobile Edge Analytics in Intelligent Transportation Systems," 2017. [Online]. Available: arXivpreprintarXiv:1712.04135

[79] J. Kumar, R. Goomer, and A. K. Singh, "Long Short Term Memory Recurrent Neural Network (LSTM-RNN) Based Workload Forecasting Model For Cloud Datacenters," *Procedia Computer Science*, vol. 125, pp. 676–682, 2018.

[80] J. J. Prevost, K. Nagothu, B. Kelley, and M. Jamshidi, "Prediction of cloud data center networks loads using stochastic and neural models," in *6th International Conference on System of Systems Engineering*, Albuquerque, USA, Jun. 2011.

[81] J. Kumar and A. K. Singh, "Dynamic resource scaling in cloud using neural network and black hole algorithm," in *Fifth International Eco-friendly Computing and Communication Systems (ICECCS)*, Bhopal, India, Dec. 2016.

[82] M. Lin, Z. Liu, A. Wierman, and L. L. Andrew, "Online algorithms for geographical load balancing," in *International Green Computing Conference (IGCC)*, San Jose, USA, Jun. 2012.

[83] J. Xu, H. Wu, L. Chen, C. Shen, and W. Wen, "Online Geographical Load Balancing for Mobile Edge Computing with Energy Harvesting," 2017.

[84] X. Cao, F. Wang, J. Xu, R. Zhang, and S. Cui, "Joint Computation and Communication Cooperation for Mobile Edge Computing," *arXiv preprint arXiv:1704.06777*, 2017.

[85] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[86] S. Boyd, "Ellipsoid method, Stanford Univ., Stanford, CA, USA." [Online]. Available: http://stanford.edu/class/ee364b/lectures/ellipsoidmethodnotes.pdf

[87] 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, 2016.

[88] R. Sutton and A. Barto., *Reinforcement learning: An Introduction*. MIT press, 1998.

[89] S. Ulukus, A. Yener, E. Erkip, O. Simeone, M. Zorzi, P. Grover, and K. Huang, "Energy Harvesting Wireless Communications: A Review of Recent Advances," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 3, pp. 360–381, 2015.

[90] E. M. R. Oliveira and A. C. Viana, "From routine to network deployment for data offloading in metropolitan areas," in *Eleventh Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, Singapore, Singapore, jun 2014.

[91] E. M. R. Oliveira, A. C. Viana, K. P. Naveen, and C. Sarraute, "Measurement-driven mobile data traffic modeling in a large metropolitan area," in *IEEE International Conference on Pervasive Computing and Communications (PerCom)*, St. Louis, USA , mar 2015.

[92] F. B. Abdesslem and A. Lindgren, "Large scale characterisation of YouTube requests in a cellular network," in *Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks*, Sydney, Australia, Jun. 2014.

[93] S. M. Zubair, J. Lusheng, L. A. X., P. Jeffrey, V. Shobha, and W. Jia, "A First Look at Cellular Network Performance During Crowded Events," *SIGMETRICS Perform. Eval. Rev.*, 2013.

[94] S. Hoteit, S. Secci, Z. He, C. Ziemlicki, Z. Smoreda, C. Ratti, and G. Pujolle, "Content Consumption Cartography of the Paris Urban Region Using Cellular Probe Data," in *Proceedings of the First Workshop on Urban Networking*, Nice, France, oct 2012.

[95] A. M. Aris and S. Bahman, "Sustainable Power Supply Solutions for Off-Grid Base Stations," *Energies*, vol. 8, pp. 10 904–10 941, 2015.

[96] M. H. Alsharif, R. Nordin, and M. Ismail, "Energy optimisation of hybrid off-grid system for remote telecommunication base station deployment in Malaysia," *EURASIP Journal on Wireless Communications and Networking*, 2015.

[97] C. Li, Y. HU, L. Liu, J. Gu, M. Song, X. Liang, J. Yuan, and T. Li, "Towards sustainable in-situ server systems in the big data era," in *ACM/IEEE 42nd Annual International Symposium on Computer Architecture*, Portland, USA, jun 2015.

[98] G. Auer, V. Giannini, C. Desset, I. Godor, P. Skillermark, M. Olsson, M. A. Imran, D. Sabella, M. J. Gonzalez, O. Blume, and A. Fehske, "How much energy is needed to run a wireless network?" *IEEE Wireless Communications*, vol. 18, no. 5, pp. 40–49, 2011.

[99] "Ericsson Microwave Outlook: Trends and Needs in the microwave Industry," Ericsson, Sweden, Tech. Rep., Oct. 2016.

[100] Open Big Data Challenge. [Online]. Available: https://dandelion.eu/datamine/open-big-data/

[101] P. S. Yu, J. Lee, T. Q. S. Quek, and Y. W. P. Hong, "Traffic Offloading in Heterogeneous Networks With Energy Harvesting Personal Cells-Network Throughput and Energy Efficiency," *IEEE Transactions on Wireless Communications*, vol. 15, no. 2, pp. 1146–1161, 2016.

[102] "Solar Radiation Measurement Data." [Online]. Available: https://energydata.info/dataset/armenia-solar-radiation-measurement-data-2017

[103] "Wind-power Generation Data." [Online]. Available: http://www.elia.be/en/grid-data/power-generation/wind-power

[104] Mingzhe Chen and Ursula Challita and Walid Saad and Changchuan Yin and Mérouane Debbah, "Machine Learning for Wireless Networks with Artificial Intelligence: A Tutorial on Neural Networks," *IEEE Wireless Communications*, Oct 2017. [Online]. Available: https://arxiv.org/abs/1710.02913

[105] C. Jiang and H. Zhang and Y. Ren and Z. Han and K. C. Chen and L. Hanzo, "Machine Learning Paradigms for Next-Generation Wireless Networks," *IEEE Wireless Communications*, vol. 24, no. 2, pp. 98–105, 2017.

[106] "Standard Performance Evaluation Corporation," SPEC, Virginia, USA, Tech. Rep., May 2013. [Online]. Available: https://www.spec.org/virt_sc2013/results/res2013q2/

[107] C. Peng, S.-B. Lee, S. Lu, H. Luo, and H. Li, "Traffic-driven Power Saving in Operational 3G Cellular Networks," in *Proceedings of the 17th Annual International Conference on Mobile Computing and Networking*, Nevada, USA, sep 2011.

[108] D. Pelleg, A. W. Moore *et al.*, "X-means: Extending K-means with efficient estimation of the number of clusters," in *Proceedings of the Seventeenth International Conference on Machine Learning (ICML)*, San Francisco, USA, Jun 2000.

[109] J. Wu, Y. Bao, G. Miao, S. Zhou, and Z. Niu, "Base-Station Sleeping Control and Power Matching for Energy-Delay Tradeoffs With Bursty Traffic," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 5, pp. 3657–3675, 2016.

[110] K. Li, "Performance Analysis of Power-Aware Task Scheduling Algorithms on Multiprocessor Computers with Dynamic Voltage and Speed," *IEEE Transactions on Parallel and Distributed Systems*, vol. 19, no. 11, pp. 1484–1497, 2008.

[111] C. Canali, L. Chiaraviglio, R. Lancellotti, and M. Shojafar, "Joint Minimization of the Energy Costs From Computing, Data Transmission, and Migrations in Cloud Data Centers," *IEEE Transactions on Green Communications and Networking*, vol. 2, no. 2, pp. 580–595, 2018.

[112] O. Maimon and L. Rokach, *Data Mining and Knowledge Discovery Handbook*. Springer, 2010.

[113] C. Gu, H. Huang, and X. Jia, "Power Metering for Virtual Machine in Cloud Computing-Challenges and Opportunities," *IEEE Access*, vol. 2, pp. 1106–1116, 2014.

[114] R. Shea, H. Wang, and J. Liu, "Power consumption of virtual machines with network transactions: Measurement and improvements," in *IEEE INFOCOM-IEEE Conference on Computer Communications*, Toronto, Canada, Apr. 2014.

[115] W.-C. Ho, L.-P. Tung, T.-S. Chang, and K.-T. Feng, "Enhanced component carrier selection and power allocation in LTE-advanced downlink systems," in *2013 IEEE WCNC*, Shanghai, China, apr 2013.

[116] "Mobile Edge Computing (MEC): Location API," ETSI, Sophia-Antipolis, France, Tech. Rep., Jul 2017.

[117] B. Fan and H. Ahmed, "A survey of mobility models," *Wireless Adhoc Networks*, vol. 206, pp. 147–176, 2011.

[118] "150 Megabit Ethernet anonymized packet traces." [Online]. Available: http://mawi.wide.ad.jp/mawi/ditl/ditl2009/

[119] S. Ripduman, R. Andrew, A. W. Moore, and M. Kieran, "Characterizing 10 Gbps network interface energy consumption," in *IEEE 35th Conference on Local Computer Networks (LCN)*, Colorado, USA, oct 2010.

[120] N. Cordeshi, M. Shojafar, and E. Baccarelli, "Energy-saving self-configuring network data centers," *Computer Networks*, vol. 57, no. 17, pp. 3479–3491, 2013.

[121] Chen, Lixing and Zhou, Sheng and Xu, Jie, "Computation peer offloading for energy-constrained mobile edge computing in small-cell networks," *IEEE/ACM Transactions on Networking*, vol. 26, no. 4, pp. 1619–1632, 2018.

[122] M. Cardosa, M. R. Korupolu, and A. Singh, "Shares and utilities based power consolidation in virtualized server environments," in *IFIP/IEEE International Symposium on Integrated Network Management*, New York, USA, jun 2009.