UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

Traffic control for energy harvesting virtual small cells via reinforcement learning

by

Dagnachew Azene Temesgene

UNIVERSITAT POLITÈCNICA DE CATALUNYA

PHD THESIS

Traffic Control for Energy Harvesting Virtual Small Cells via Reinforcement Learning

Author:
Dagnachew Azene Temesgene

Director:
Dr. Paolo Dini

Tutor:
Prof. Miquel Soriano

A thesis submitted in fulfillment of the requirements for the degree of International Doctor of Philosophy in the

Department of Network Engineering

Barcelona, June 2020

UNIVERSITAT POLITÈCNICA DE CATALUNYA

Abstract

Department of Network Engineering

Doctor of Philosophy

Traffic Control for Energy Harvesting Virtual Small Cells via Reinforcement Learning

by Dagnachew Azene TEMESGENE

Due to the rapid growth of mobile data traffic, future mobile networks are expected to support at least 1000 times more capacity than 4G systems. This trend leads to an increasing energy demand from mobile networks which raises both economic and environmental concerns. Energy costs are becoming an important part of OPEX by Mobile Network Operators (MNOs). As a result, the shift towards energy-oriented design and operation of 5G and beyond systems has been emphasized by academia, industries as well as standard bodies. In particular, Radio Access Network (RAN) is the major energy consuming part of cellular networks. To increase the RAN efficiency, Cloud Radio Access Network (CRAN) has been proposed to enable centralized cloud processing of baseband functions while Base Stations (BSs) are reduced to simple Radio Remote Heads (RRHs). The connection between the RRHs and central cloud is provided by high capacity and very low latency fronthaul. Flexible functional splits between local BS sites and a central cloud are then proposed to relax the CRAN fronthaul requirements via partial processing of baseband functions at the local BS sites. Moreover, Network Function Virtualization (NFV) and Software Defined Networking (SDN) enable flexibility in placement and control of network functions. Relying on SDN/NFV with flexible functional splits, network functions of small BSs can be virtualized and placed at different sites of the network. These small BSs are known as virtual Small Cells (vSCs). More recently, Multi-access Edge Computing (MEC) has been introduced where BSs can leverage cloud computing capabilities and offer computational resources on demand basis.

On the other hand, Energy Harvesting (EH) is a promising technology ensuring both cost effectiveness and carbon footprint reduction. However, EH comes with challenges mainly due to intermittent and unreliable energy sources. In EH Base Stations (EHBSs), it is important to intelligently manage the harvested energy as well as to ensure energy storage provision. Consequently, MEC enabled EHBSs can open a new frontier in energy-aware processing and sharing of processing units according to flexible functional split

options. The goal of this PhD thesis is to propose energy-aware control algorithms in EH powered vSCs for efficient utilization of harvested energy and lowering the grid energy consumption of RAN, which is the most power consuming part of the network. We leverage on virtualization and MEC technologies for dynamic provision of computational resources according to functional split options employed by the vSCs.

After describing the state-of-the-art, the first part of the thesis focuses on offline optimization for efficient harvested energy utilization via dynamic functional split control in vSCs powered by EH. For this purpose, dynamic programming is applied to determine the performance bound and comparison is drawn against static configurations. The second part of the thesis focuses on online control methods where reinforcement learning based controllers are designed and evaluated. In particular, more focus is given towards the design of multi-agent reinforcement learning to overcome the limitations of centralized approaches due to complexity and scalability. Both tabular and deep reinforcement learning algorithms are tailored in a distributed architecture with emphasis on enabling coordination among the agents. Policy comparison among the online controllers and against the offline bound as well as energy and cost saving benefits are also analyzed.

Acknowledgements

I would like to express my appreciation to all the people who have supported and encouraged me throughout the thesis journey.

First of all, I would like to express my sincere gratitude to my supervisors Dr. Paolo Dini and Dr. Marco Miozzo for their advice, patience, guidance and enthusiastic encouragement throughout the course of the PhD. Their continuous guidance and support helped me to be motivated and grow professionally.

I would also like to express my appreciation to Prof. Deniz Gündüz for his advice during the secondment period of the PhD research. I am also grateful to Dr. Antonio De Domenico and Prof. Michele Rossi for their time and valuable feedback throughout the thesis review process. I would like to thank my friends and colleagues, Nicola Piovesan and Hoang Duy Trinh, for their collaboration, technical discussions and all the good times together. Many thanks to SCAVENGE project colleagues for their support and cooperation throughout the project.

Last but not least, I would like to express my heartfelt gratitude to my family for their moral support during all these years and many thanks for all my friends living in Ethiopia and abroad for their help and encouragements.

Contents

List of Figures

List of Tables

Abbreviations

5G	**5**-th **G**eneration
AI	**A**rtificial **I**ntelligence
BB	**B**ase **B**and
BBU	**B**ase **B**and **U**nit
BSs	**B**ase **S**tations
CAPEX	**CAP**ital **EX**penditures
CRAN	**C**loud **R**adio **A**ccess **N**etwork
DP	**D**ynamic **P**rogramming
DQN	**D**eep **Q**-**N**etwork
DDRL	**D**istributed **D**eep **R**einforcement **L**earning
DRL	**D**eep **R**einforcement **L**earning
EH	**E**nergy **H**arvesting
EHBS	**E**nergy **H**arvesting **B**ase **S**tation
FQL	**F**uzzy **Q**-**L**earing
FIS	**F**uzzy **I**nference **S**ystem
FRAN	**F**og **R**adio **A**ccess **N**etwork
GOPS	**G**iga **O**perations **P**er **S**econd
HetNets	**Het**erogenous **Net**works
HCRAN	**H**eterogenous **C**loud **R**adio **A**ccess **N**etwork
ICT	**I**nformation and **C**ommunication **T**echnology
IoT	**I**nternet of **T**hings
IT	**I**nformation **T**echnology
LTE	**L**ong **T**erm **E**volution
MAC	**M**edia **A**ccess **C**ontrol
MBS	**M**acro **B**ase **S**tation

MDPs	**M**arkov **D**ecision **P**rocesses
MEC	**M**ulti-access **E**dge **C**omputing
ML	**M**achine **L**earning
MNOs	**M**obile **N**etwork **O**perators
MRL	**M**ulti-agent **R**enforcement **L**earning
NFV	**N**etwork **F**unction **V**irtualization
OPEX	**OP**erational **EX**penditures
PHY	**PHY**sical layer
PV	**P**hoto **V**oltaic
QL	**Q**-**L**earing
QoS	**Q**uality of **S**ervice
RAN	**R**adio **A**ccess **N**etwork
RAT	**R**adio **A**ccess **T**echnology
RL	**R**enforcement **L**earning
RRH	**R**adio **R**emote **H**ead
RRM	**R**adio **R**esource **M**anagement
SBSs	**S**mall **B**ase **S**tations
SOC	**S**tate **O**f **C**harge
SCs	**S**mall **C**ells
SDN	**S**oftware **D**efined **N**etworking
TD	**T**emporal **D**ifference
UE	**U**ser **E**quipment
VNF	**V**irtual **N**etwork **F**unction
vSCs	**v**irtual **S**mall **C**ells

Chapter 1

Introduction

1.1 Scenario and Motivation

Mobile data traffic is growing exponentially. It is estimated that overall mobile data will grow to 77 exabytes by 2022, a seven fold increase from 2017 [10]. In order to meet the growing demand from various end-devices, e.g., smartphones, tablets and IoT devices, future mobile networks are expected to support at least 1000 times more capacity than LTE systems [11]. This, in turn, leads to an increasing energy demand from mobile networks which raises both environmental and economic concerns. A study by Digital Power Group [12] estimated that the global ICT system consumed about 1500 TWh of electricity annually, which is approximately 10% of annual global electricity generation. In addition, ICT represents about $2-4\%$ of carbon footprint from human activity, which is equivalent to the emission from aviation industry. Moreover, fueled by the growing mobile traffic demand, the energy consumption of ICT will reach about 51% of worldwide electricity consumption in 2030 [13]. As a result, energy consumption expenditure is becoming an important portion of OPerational EXpenditures (OPEX) by Mobile Network Operators (MNOs). This trend leads to about 6% annual reduction in MNOs' average revenue per unit [14]. Hence, sustainable design and operation of mobile networks is included in the road map towards future mobile networks, known as 5G and beyond. The shift towards energy-oriented 5G and beyond systems has been emphasized by academia, industries as well as standardization bodies [15, 16].

Radio Access Network (RAN) is the major energy consuming part of cellular network [17]. Recently, to increase the RAN energy and radio resources' efficiency, an architecture known as Cloud Radio Access Network (CRAN) [18–20] has been proposed. CRAN, leveraging on advances in cloud computing, aims to process the mobile network Base Band (BB) functions by a centralized pool of Base Band Units (BBUs) while Base Stations (BSs) are reduced to Remote Radio Heads (RRHs). The connection between

the RRHs and the BBU pool is provided by high capacity and low latency fronthaul. The centralized processing proposed in CRAN ensures simplified BSs at cell sites and more efficient resource utilization. However, the high capacity and very low latency fronthaul requirements are challenges in CRAN. To relax these fronthaul requirements, *flexible functional splits* between local BS sites and a central BBU pool [1, 21] have been proposed. Hence, part of the BB processes are executed at the local BS sites, while maintaining many of the centralization advantages of CRAN. Flexible functional splits enable the feasibility of other cost efficient and flexible fronthaul technologies, e.g., mmWave and microwave [22]. Moreover, Network Function Virtualization (NFV) enables softwarized implementation of traditional network functions on general purpose computing hardware as virtual functions and Software Defined Networking (SDN) can be used for the management of these functions [23]. Leveraging on SDN/NFV with flexible functional splits, network functions of small BSs (e.g., BB functions) can be virtualized and placed at different sites of the network. These small BSs are known as virtual Small Cells (vSCs). More recently, Multi-access Edge Computing (MEC) has been introduced to enable the convergence of IT and telecommunication [24]. Thanks to MEC, BSs can leverage cloud computing capabilities and offer their computational resources on demand basis.

Energy Harvesting (EH) is a promising technology ensuring both cost effectiveness and carbon footprint reduction [25]. EH involves powering mobile network constituent elements, e.g., BSs, with renewable energy supplies. However, EH has challenges due to intermittent and unreliable energy sources. Energy arrival process is random, and hence, energy storage and intelligent energy allocation scheme are required to ensure uninterrupted operation. Hence, in a network of Energy Harvesting Base Stations (EHBSs), it is important to intelligently manage the harvested energy and to ensure energy storage provision. Consequently, MEC enabled EHBSs can open a new frontier in energy-aware processing and sharing of BB processing units according to flexible functional split options. The vSCs that solely rely on EH sources can opportunistically use the BB processing units available at the MEC server. This is particularly important since the power consumption due to BB processing has a significant share in the total power consumption breakdown of small BSs. As a result, MEC-enabled energy-aware placement of BB processes according to functional split options is a promising technique to enable energy efficient operation of EH powered vSCs. *This thesis focuses on the study of control algorithms for efficient utilization of harvested energy in vSCs with EH capabilities.* The work presented in this thesis has been carried out with in project number 4 of SCAVENGE Innovative Training Network funded by the European Union in the framework of the H2020 Marie Skodowska Curie Action [26].

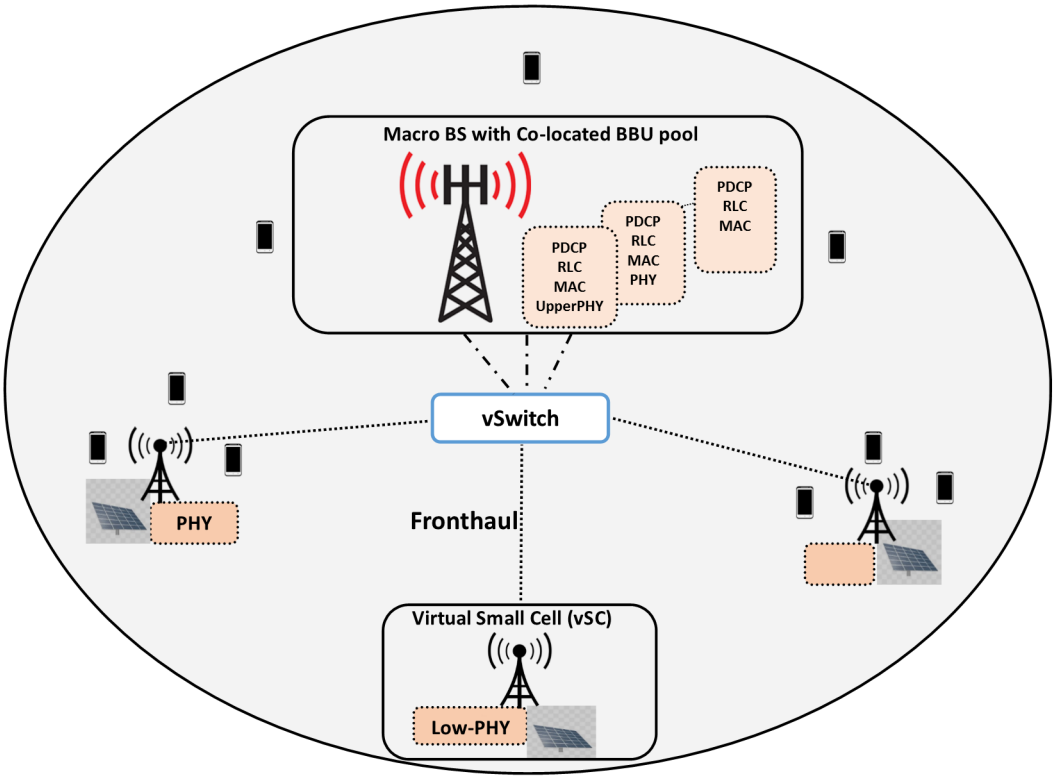


FIGURE 1.1: Network Scenario

1.2 Problem Statement

We consider a two-tier network scenario that is composed of a Macro BS (MBS) with co-located BBU pool and vSCs as shown in Fig. 1.1. The MBS with BBU pool is responsible for providing baseline coverage, mobility support and BB processing resources. The MBS site is fully supplied by the grid, thus assuring reliable communication and computing. The vSCs are deployed in hot-spots for capacity enhancement without overlapping coverage [27]. They are completely powered by solar panel and equipped with rechargeable batteries. The vSCs are fully or partially dependent on the MBS for their BB processing. Different functional split options are targeted where BB functions of the vSCs can be executed locally at the vSC or remotely using the resources of the MBS.

In addition, MEC enables BSs to leverage cloud computing capabilities. In MEC deployments, multi-tier MEC servers are co-located at BSs that have different computational and transmission capabilities (i.e., MBSs are high-power and high-computing nodes) [28]. In our scenario, we are interested in the case that the MBS hosts a MEC server to support the vSCs via computational offloading of some of their network functions. Hence, the vSCs opportunistically use the central BBU pool at the MBS, acting as the

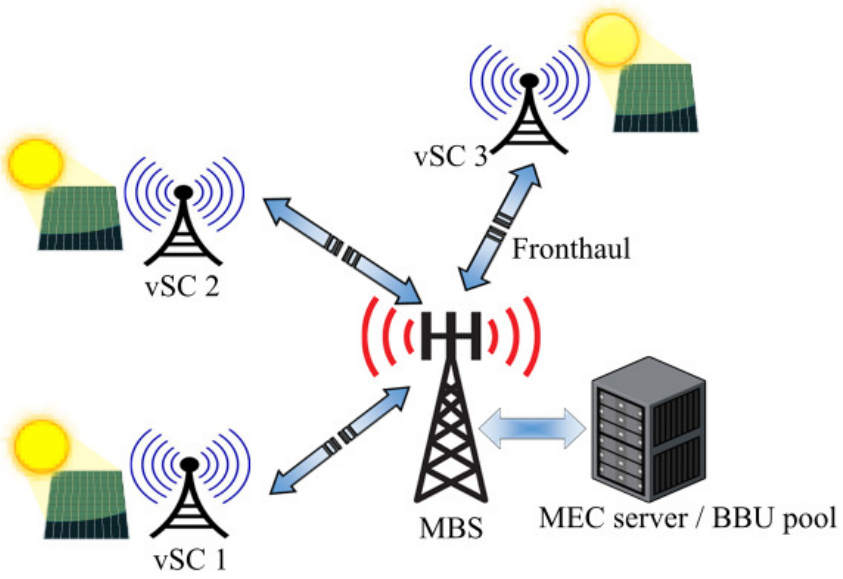


FIGURE 1.2: Reference architecture

MEC server, for full or partial BB processing requirements. For doing so, a standardized interface (e.g., Openflow [29]), can be used to implement the interactions between the MBS and vSCs. The proposed MEC-enabled architecture is shown in Fig. 1.2. This architecture jointly with SDN and NFV paradigms [30] enables automated network management, flexibility and cost reductions. It allows the vSCs' BB functions to be decoupled from proprietary hardware-dependent implementations and to be executed in different hardware resource of the network. In this regard, 3GPP has defined different functional splits between the distributed and the centralized unit [31], i.e., the vSCs and the MBS respectively. These split options along with a conventional eNodeB architecture are shown in Fig 1.3. The network functions without virtualized implementation are known as Physical Network Functions (PNF). The baseline macro or the standard CRAN split is also known as PHY-RF split.

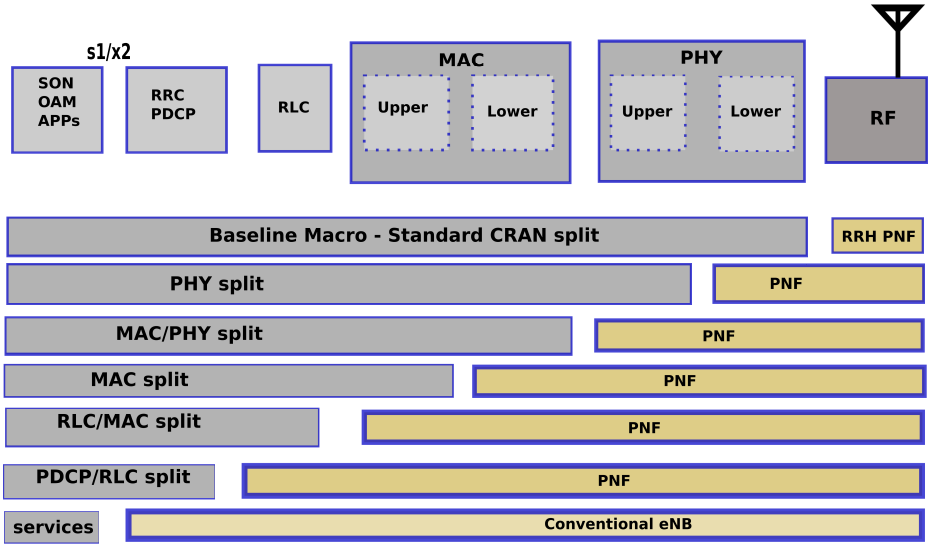


FIGURE 1.3: LTE functional split options [1]

The vSCs are solely powered by EH and batteries. Hence, it is importance to intelligently control the utilization of the harvested energy due to the randomness of energy arrival process. A dynamic control decision involves consideration of many factors including energy reserves, energy arrival forecast, traffic demand and system drop rate constraints. Hence, at each time-granularity of operation, the vSC can be controlled to operate in one of the possible operation modes, i.e., the functional split options. These operation modes can range from the execution of all the BB functions locally at vSCs or offloading part of the BB functions to the central BBU pool/ MEC server to switching off the vSCs.

1.3 Objectives

The objective of the thesis is to study the energy savings that can be achieved by dynamic control of BS modes of operation in softwarized RANs supplied by EH sources. This involves dynamic decision scheme for the placement of BB processes of EHBSs, either locally using their resources or remotely on a cloud/MEC resource. Dynamic configuration decisions have to take into account the traffic demand, energy arrival and available energy reserve while minimizing grid energy consumption and traffic drop rate. As mentioned in Section 1.1, we rely on SDN/NFV tools as enablers for the movement of virtual network functions.

To achieve the above goal, the following objectives have been studied in the thesis:

1. **Identifying Research Gaps** (**O**1): It involves the study of state-of-the-art to identify the research gaps in different softwarized RAN architectures, the issue of integrating EH in these architectures as well as the control methods. This objective also involves identification of BS power consumption models, traffic models and EH source models which are the building blocks for the realization of the other objectives.

2. **Performance Bound Assessment** (**O**2): It involves performance bound study of dynamic selection of functional split options in EH vSCs that rely on centralized BBU pool for part of their BB processing. The bound should ensure the constraints on the system drop rate and must consider the traffic request, energy arrival and energy reserve information.

3. **Online Algorithms** (**O**3): It involves studying online algorithms for dynamic functional split control. This implies removing some assumption made in **O**2 and modeling more realistic network scenarios. The algorithms are designed taking into account the partial observable information on the traffic demand, energy arrival and battery states guaranteeing constraints on the system drop rate. Various online distributed control methods are explored with special emphasis on temporal difference and deep reinforcement learning approaches.

4. **Policies Comparison** (**O**4): This objective involves the comparison among the policies obtained by online distributed controllers in terms of their characteristics as well as network performance. Moreover, the online controller policies are evaluated with respect to the performance bound in **O**2. In addition, the energy and cost savings obtained by different online control policies are estimated.

1.4 Methodology

For the completion of **O**1, we begin with a literature review to identify the gaps in the state-of-the-art including the issues of embedding EH in softwarized RAN architectures, available power consumption, traffic and EH sources models. We identified traffic and EH models that can be incorporated to our study. In addition, we have made an adaptation to flexible power consumption model of BSs in order to estimate the power consumption required by each BB functions / power consumption requirement per functional split options. The review of the state-of-the-art including the system models adopted are described in detail in Chapter 2.

To realize **O**2, we have applied an offline optimization using Dynamic Programming (DP), in particular shortest path search, to determine performance bound of dynamic

functional split control. The bound assessment study is described in detail in Chapter 4. The results of this offline optimization are used as performance bounds against which other online control policies can be compared with.

Regarding both **O**3 and **O**4, our focus is on online optimization approaches for dynamic functional split control in a vSCs solely powered by EH. As opposed to **O**2, here we rely on only partial information for a dynamic selection of functional split options considering the traffic request, energy reserve and energy arrivals. Hence, approximate DP methods, in particular Reinforcement Learning (RL) methods, will be suitable tools to derive these polices. In this regard, a study of the application of RL methods, in particular Q-Learning (QL) and SARSA algorithms in a single EH vSC scenario is described in Chapter 5. Extending the online optimization approach described in Chapter 5 to a scenario involving multiple vSCs is explained in Chapter 6. In this case, multi-agent temporal difference RL methods are applied with different levels of coordination among agents to derive control policies of efficient harvested energy utilization. To cope with the limitations of temporal difference RL for applications involving continuous states, Fuzzy Q-Learning (FQL) based controller has also been proposed. Moreover, due to the limitations of multi-agent tabular RL methods, i.e., QL, SARSA, FQL, when working with large state-action spaces, the more recent deep RL method is applied as an approach to coordinate the policies of agents via local state information exchange without facing practically infeasible state-action tables. The deep RL based controller design and its performance are presented in Chapter 7.

1.5 Outline of the thesis

This section gives a brief summary of the contents of the following chapters.

Chapter 2

This chapter presents the state-of-the-art of softwarized RAN with EH capabilities spanning the different RAN architecture proposals and the application of optimization tools in these architectures. Moreover, research gaps are identified regarding the integration of EH in softwarized RAN architectures. In addition, BS power consumption models, traffic demand models as well as EH sources models, which are the building blocks for realizing the thesis objectives, are described.

The contents of this chapter are published in the following paper:

D. A. Temesgene, J. Nunez-Martnez and P. Dini, "Softwarization and Optimization for Sustainable Future Mobile Networks: A Survey," in IEEE Access, 2017

Chapter 3

This chapter provides a brief background on the optimization tools used in the thesis. Hence, a brief explanation on DP, temporal difference RL, multi-agent RL as well as deep RL are given.

Chapter 4

In this chapter, the study of performance bound for optimal placement of functional split options for vSCs with EH capabilities is presented. The chapter describes the optimization problem and the optimal solution with comparison against static policies. In addition, modification of the optimization objective to include energy sharing between vSCs and MBS is also explained with policy characterization.

The contents of this chapter are published in the following papers:

D.A. Temesgene, N. Piovesan, M. Miozzo and P.Dini, "Optimal Placement of Baseband Functions for Energy Harvesting Virtual Small Cells," in Proc. IEEE VTC2018-Fall

N. Piovesan, D.A. Temesgene, M. Miozzo, and P. Dini. "Joint Load Control and Energy Sharing for Autonomous Operation of 5G Mobile Networks in Micro-Grids." in IEEE Access, 2019

Chapter 5

This chapter includes the study of temporal difference RL methods for the dynamic selection of functional split options without relying on a-priori knowledge. The case of singe vSC agent control via QL and SARSA is explained with training analysis and network performance evaluation.

The contents of this chapter are published in the following paper:

D.A. Temesgene, M. Miozzo and P.Dini, "Dynamic Functional Split Selection in Energy Harvesting Virtual Small Cells Using Temporal Difference Learning", in Proc. IEEE PIMRC 2018

Chapter 6

This section extends the temporal difference learning methods to a multi-agent optimization scenario, i.e., a network of multi-vSCs. Hence, distributed QL and FQL methods are tailored to the network scenario. Moreover, the training analysis, the benefit of fuzzification with respect to QL and network performance evaluations

of these distributed controllers are explained. Finally, the policy characteristics of these distributed online methods are evaluated against the offline bounds and the validation of the proposed polices are provided.

The contents of these chapter are published in the following paper:

D.A. Temesgene, M. Miozzo and P.Dini, " Dynamic Control of Functional Splits for Energy Harvesting Virtual Small Cells: a Distributed Reinforcement Learning Approach", in Elsevier Computer Communications, 2019

Chapter 7

In this chapter, distributed deep RL based controller design for dynamic functional split is explained. The architecture of deep RL based controller, the training analysis as well as its policy characteristics are included. Moreover, the performance of the deep RL based policies are evaluated against the distributed temporal difference online controller policies described in Chapter 6.

The contents of this chapter are included in the following paper:

D.A. Temesgene, M. Miozzo, D. Gündüz and P.Dini, "Distributed Deep Reinforcement Learning for Functional Split Control in Energy Harvesting Virtualized Small Cells", submitted to IEEE Transactions on Sustainable Computing, 2020

Chapter 8

In this chapter, a brief summary of the thesis results in form of conclusions is given. Moreover, future research directions are also highlighted.

Chapter 2

State of the Art and Beyond

2.1 Introduction

In order to cope with the growing mobile data traffic, a new generation of cellular network known as 5G is envisioned. A closer look at the requirements of 5G reveals that the successful deployment of 5G will require architectural evolution in cellular networks and specifically in the RAN segment. The main 5G requirements, as compared with 4G, are [11, 32, 33]:

- 1000 times more mobile data volume;
- 100 times more user data rate;
- 1000 times more number of connected devices;
- 1/10 reduction in energy consumption;
- 1/5 reduction in end to end latency;
- 1/5 lower cost of network management;
- 10 times longer battery life and
- 1/1000 reduction in service deployment times.

Hence, satisfying these diverse requirements calls for a paradigm shift in the design of cellular networks, RAN in particular. To this end, cellular networks are including new paradigms, technologies and tools. We have identified that softwarization and densification paradigms, EH and optimization tools are three key pillars to meet the main 5G goals. Softwarization and densification paradigms are

needed to meet the demands of high capacity, cost reduction, improved agility, lower latency and service deployment times. Furthermore, EH ensures sustainable and cost effective operation of cellular networks. The specific advantages of softwarization, densification and EH should be combined in timely and optimal way according the system requirements. For this reason, the applications of network optimization tools are crucial for lowering network management costs and to enable network wide intelligence and automation.

In the following, we give a brief introduction to the three pillars, namely softwarization, EH and optimization tools in the context of RAN in Section 2.2. Then a more detailed survey of RAN architectures is given in Section 2.3. In addition, the application of EH in RAN as well as the role of optimization tools are presented in Sections 2.4 and 2.5, respectively.

2.2 The Three Pillars

This section gives a brief introduction to the three pillars of sustainable mobile networks design and operation namely softwarization, EH and optimization tools.

2.2.1 Softwarization and Densification in RAN

Softwarization of mobile networks is mainly facilitated by the adoption of cloud computing, SDN, and NFV technologies. Cloud computing is a model for delivering computing services on demand basis over the Internet [34, 35]. In cloud computing, resources such as processing, data storage and networking are provided to end-users on demand basis. SDN, on the other hand, advocates for separation of the control and data planes [36–38]. Canonically, SDN comprises of a central controller that manages the data plane switching elements to activate the desired switching policies. Therefore, the switching elements apply the rules stated by the centralized controller and the interface between the controller and such forwarding elements can be, for instance, a standardized interface e.g., Openflow [38]. SDN through separating control and data planes and by adopting centralized control enables many advantages such as programmability, automation, and significant cost reductions due to lower complexity in the switching elements. On the other hand, NFV enables softwarized implementation of network functions on a general purpose hardware [37, 39]. Therefore, NFV decouples the network functions from

proprietary hardware-dependent implementations and enables the use of a hardware resource for many network functions. NFV has many advantages ranging from improved scalability and flexibility to significant cost reduction via sharing of a hardware resource for many network functions. It is important to note that SDN and NFV are independent but complementary technologies and the adoption of both in a network architecture will maximize their advantages in terms of improved flexibility, scalability, and cost reduction. Moreover, cellular networks are increasingly becoming dense due to the deployment of multi-tier BSs in the same coverage area. Densification significantly increases the offered capacity to end-users through frequency reuse. Such multi-tier BSs are already being deployed by MNOs to meet the growth in demand and more deployment is expected in the coming years. Dense deployment requires the management of a large number of resources. As a result, softwarization is required in dense deployments than legacy ones as an enabler for automated network management, flexibility and cost reductions.

2.2.2 EH in RAN

While densification is important to meet high capacity demand, it poses challenges in the energy consumption of the network. These challenges come from three directions. These are:

- Densification results in high demand of electrical power to ensure uninterrupted operations. Such high demand for energy increases the cost of operation of a network.
- Dense deployment of BSs also increases CO_2 emissions from mobile networks operation. Environmental footprints need to be reduced to limit the impact on climate change.
- Most of the BSs, also known as Small Cells (SCs), are expected to be deployed in hot spots. This makes grid power supply access to these BSs difficult sometimes. Moreover, RAN accounts for around 80% of energy demand from cellular networks [17].

These reasons drive the adoption of EH in RAN. EH is a solution to power communication nodes from natural or man-made activities by scavenging energy physically or chemically [40]. The use of EHBSs is multi-fold. Its potential advantages are:

- Significant reduction of operational costs, since the only incurred cost is due to EH hardware and site access. Once installed, the energy obtained is free.
- It minimizes the carbon footprint from the operation of network infrastructure. This enables MNOs to be aligned with sustainability goals, regulations and directions.
- EHBSs are independent of grid power access, which can improve MNOs network planning. As a result, MNOs need to consider only network performance related parameters prior to installation.

These factors lead to EHBSs deployment and massive usage of them is expected in the future 5G networks. With EH, energy is a intermittent resource and it is interesting to study the interaction and harmonization of EH with other RAN architecture paradigms, namely softwarization and densification.

2.2.3 Optimization tools in RAN

The vision of future mobile network, sketched in the above, correspond to a highly heterogeneous and complex system, including several types of end-devices with diverse QoS requirements, multiple cell layers working in different radio technologies and spectrum bands, and dynamic reconfiguration and placement of network functions. A general 5G network architecture platform that incorporates multiple technologies including massive MIMO, cognitive radio, device-to-device (D2D) communication, small cell networks, cloud, NFV and SDN is proposed in [41]. The proposed architecture gives insight into the complexity of the 5G. Such a complexity needs an intelligent, timely and automated control to balance many, often conflicting goals, and to ensure efficient deployment and utilization of the available spectrum, energy and computational resources. Hence optimization tools such as Machine Learning (ML) and DP are required in order to analyze the environment and take the appropriate decisions. In fact, ML and DP may include an end-to-end knowledge of the system to achieve a proactive optimization, able to exploit the huge amount of data available and to even incorporate additional dimensions, such as the characterization of end-user experience and behavior, the energy consumed and harvested. ML is a technique of detecting patterns in datasets automatically [42]. ML is applied to those kinds of problems where writing an explicit computer program is impossible or extremely challenging due to the complexity of patterns that need to be detected. One of such systems with complex

datasets is the cellular network. Because of the diverse nature of cellular network systems, which results in diverse sets of data, applying analytical modeling and performance prediction is challenging. Besides, DP (or RL as its approximation) algorithms help to solve complex sequential decision making processes by minimizing a certain cost function.

2.3 RAN Architectures

This section gives a more detailed explanation of the recent RAN architectures and paradigms namely network densification and softwarized RAN architectures.

2.3.1 Network Densification

Heterogeneous Networks (HetNets) is a paradigm that supports the co-existence of several BS types in the cellular network architecture each having their own parameters such as transmission power and coverage area [25]. HetNets were primary adopted as a means to enhance the capacity of cellular networks in areas of high traffic loads, such as indoor and areas of coverage holes around cell edges. Multi-tier cellular networks encompassing multiple types of BSs that co-exist in the coverage area of a HetNet deployment.

The BSs in a HetNet can be classified as MBSs and low power BSs, i.e., SCs. The low power BSs are usually further classified as pico, femto and relay cells [2], as shown in Fig. 2.1. Pico cells are typically deployed in hotspots to enhance the performance of the network in the selected areas. They are deployed in a planned manner by operators and they are open to all users in their coverage area. Hence, pico cells are similar to regular BSs with the only difference being lower transmission power, and hence, coverage area. On the other hand, femto cells are deployed for indoor environments in unplanned manner (i.e., deployed by customers) [2, 43]. Depending on whether femtocells allow access to all User Equipments (UEs) or restrict some UEs from access, they are classified as open or closed respectively. Femto cells can also be configured in a hybrid mode by granting access to some terminals with lower priority. Femto cells typically utilize end-users backhaul, such as cables or Digital Subscriber Lines (DSL). Relay cells, on the other hand, are used to boost the signal from BS to users in selected locations. Hence, they are deployed mainly for throughput improvement and coverage extension in areas of coverage holes. HetNets may also include multi Radio Access

Technology (RAT) deployments, where other RATs such as Wi-Fi are used for offloading traffic [43]. Inter-tier interference and coverage holes created by closed femto cells are identified as the main challenges in HetNets [2, 43].

Conventional HetNets are evolving into the notion of ultra-dense HetNets. Ultra-dense deployment trend is characterized by having many number of over-lapping BSs. Some authors even claim that in ultra-dense HetNets, the number of cells exceeds the number of active UEs [44, 45]. In such deployments, due to the high density of SCs, idle mode of operations, when there are no active users, proper propagation models and mobility management mechanisms are necessary. In ultra-dense HetNets, SCs can be deployed either in co-channel mode or non-co-channel mode [44]. In co-channel deployment, SCs and macro cells share the same frequency band whereas in non-co-channel deployment, a different frequency band is used for SCs. Moreover, the authors in [44] show that, typically macro cells use around 1 - 2 GHz licensed spectrum, whereas SCs can use unlicensed 5GHz band. Such deployments help to reduce the interference between macro and SC tiers and improve planning of cellular networks. This is due to very large number of SCs in ultra-dense deployments and co-channel existence with the baseline macro cells will be extremely challenging due to severe interference. The European METIS project identify ultra-dense networks as one of 5G architecture enablers. The project also identifies that the utilization of higher frequency spectrum, use of multi-RATs and switch on/off scheduling of SCs are essential techniques in ultra-dense deployments [33].

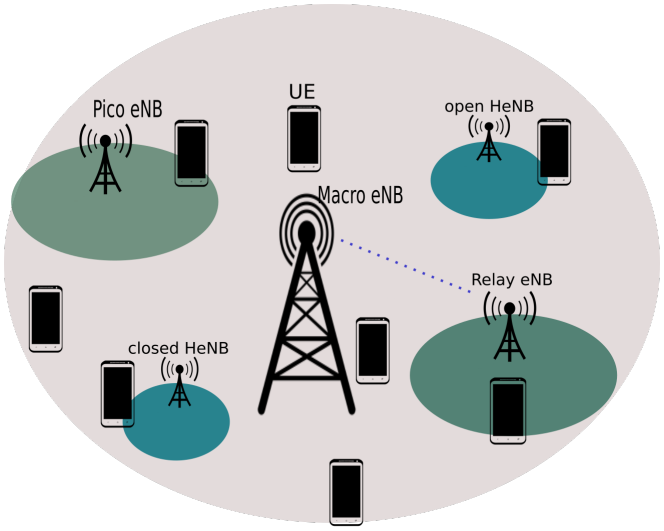


FIGURE 2.1: HetNet architecture [2]

2.3.2 CRAN

CRAN is an architecture that integrates cloud computing technology into the RAN of cellular networks. In CRAN, the BB functions are processed in a centralized BBU pool or central cloud [18–20]. Therefore, the BSs at the cell sites are reduced to simple RRHs. The various RRHs and the central cloud are connected via fronthaul links. The connectivity between central cloud and core of the network is provided by transport network. In its original proposal of CRAN, the RRH and central cloud connectivity is defined to be optical fiber fronthaul [18]. CRAN has a lot of advantages ranging from simplified BSs at cell sites to centralized processing at the cloud. Cloud operation also leads to a more efficient utilization of available resources. CRAN also enables decoupling of processing and transmission and opens a room to apply data plane cooperation techniques, such as CoMP [18, 46, 47].

Some of the challenges in CRAN arise from the fronthaul connectivity between RRH and central cloud. Optical fiber fronthaul provides the required high capacity and very low latency connectivity between RRH and central cloud. However, using optical fiber fronthaul for every RRH is not scalable and flexible due to high cost and unavailability [36]. Heterogenous CRANs (HCRANs) are extension of CRAN architectures to include the presence of High-Power Nodes (HPNs) for control plane functions and coverage [21]. Data transmission is handled by RRHs. HCRANs are similar in many aspects to CRAN, but they are slightly different since HPNs are interfaced to the central cloud for mitigation of cross-tier interference by cooperative techniques at the cloud [21]. HCRANs partially alleviate the fronthaul problem in CRANs by decoupling the control signaling from data. All the control signaling is delivered via HPNs and the fronthaul only carries data. In addition to reduced fronthaul requirements, HCRANs also open a room for multi-tier cooperation advantages [48].

Based on the ETSI NFV use case for mobile BSs [49], the authors in [1] propose flexible functional splits between RRH and central BBU pool and identify the requirements of fronthaul capacity at various functional split options. In flexible functional split architecture, part of the BB processing of SCs is done locally at SCs and the remaining BB processing takes place at the central BBU pool. Flexible functional splits enable the relaxation of the fronthaul capacity and latency constraints without sacrificing the centralization gains offered by the CRAN. The functional split between RRH and central cloud can occur at various layers ranging from PHY to PDCP-RLC, as shown in Fig. 1.3.

2.3.3 Fog Radio Access Networks (FRANs)

FRANs are proposed to embed the advantages of fog computing to alleviate the aforementioned limitations of CRAN, such as capacity constrained fronthaul, latency and heavy burden at central cloud [50]. In FRAN, shown in Fig. 2.2, in addition to the central cloud of CRAN, edge devices such as RRHs and UEs can be used for local signal processing, cooperative radio resource management and content storage [46, 50]. Such paradigm of using edge devices for computing and storage is called fog computing [10, 51–53] and it helps to partly address the fronthaul and central cloud heavy burdens in CRAN. Fog computing is gaining increasing interest recently as a complementary technology to alleviate cloud computing drawbacks. This is evidenced by initiatives such as OpenFog [54] and ETSI-MEC [55] to define an architecture of fog/MEC as a complement to cloud computing technology. One major functionality in FRANs is edge caching. Edge devices, such as RRHs and UEs, can be used as local storage of certain contents to reduce the latency arising from accessing the contents from far servers. Various edge caching strategies in FRANs are presented in [50].

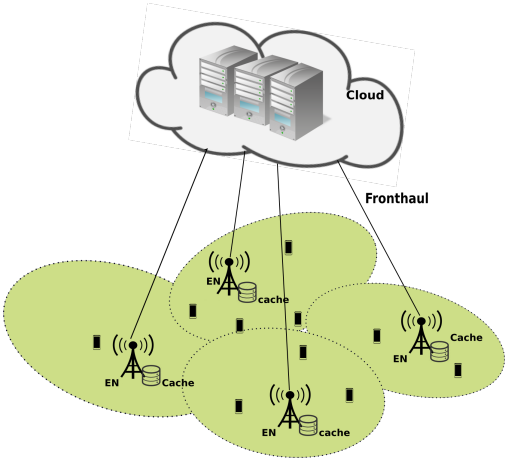


FIGURE 2.2: FRAN architecture [3]

2.3.4 SDN/NFV/Network Slicing based RAN architectures

Here, we classify RAN proposals in three categories. The first category deals with proposals that apply SDN to RAN. These works mainly focus on decoupling the RAN control from the RAN data plane with the aim of centralizing the RAN control plane functions. In the second category, namely NFV applied to RAN, we included novel proposals that primarily focus on virtualizing RAN control and data plane functions and increasing the flexibility in allocating RAN network

functions among different mobile network segments. Finally, the third category summarizes the work conducted on end-to-end slicing, that is, the allocation of logical networks including network and IT resources. It is important to note that the three categories can potentially include both SDN and NFV technologies, but they are categorized under SDN or NFV or slicing based on their main proposed novelties.

2.3.4.1 SDN applied to RAN

One of the pioneers in this category is SoftRAN [4], shown in Fig. 2.3. SoftRAN proposes a centralized Radio Resource Management (RRM) control plane function of many BSs by abstracting them as virtual big BS in a central SDN controller. In SoftRAN, all the radio resources belonging to a region are abstracted in a 3D resource grid with time-frequency-location indices. SoftRAN also proposes to distribute part of RRM control functions that can be efficiently implemented locally by individual BSs. SoftMobile [56] identifies the need for principle based evolution of the control plane in heterogeneous wireless networks and the need for abstraction in the control plane to achieve optimal performance. The authors in [57] propose a software defined RAN platform called FlexRAN. In FlexRAN, south and northbound Application Programming Interfaces (APIs) with master-agent controller architecture are proposed. Moreover, FlexRAN is designed to be used as an implementation platform to evaluate SDN applications in RAN. Various use cases and practical evaluations show the feasibility of the platform for RAN.

Improvement of CRAN by applying SDN is proposed in [58]. The work identifies the limitation in CRAN due to its one-to-one mapping of RRH and BBU in central cloud and introduces a SDN based flexible mapping architecture called Software Defined Fronthaul (SDF). SDF improves flexibility and also guarantees seamless mobility and improved resource utilization. The CONCERT architecture in [59] defines a converged edge infrastructure for future cellular networks. In CONCERT, SDN is applied to improve the CRAN architecture by ensuring flexibility and dynamic reconfiguration of radio resources to meet the traffic requirements. This architecture consists of a data plane of all physical resources that are interconnected via software defined switches and a control plane that is implemented by a conductor that orchestrates and virtualizes the resources in the data plane. The proposal enables a convergence of both cloud computing and mobile communications through software defined abstraction and management of the RAN resources.

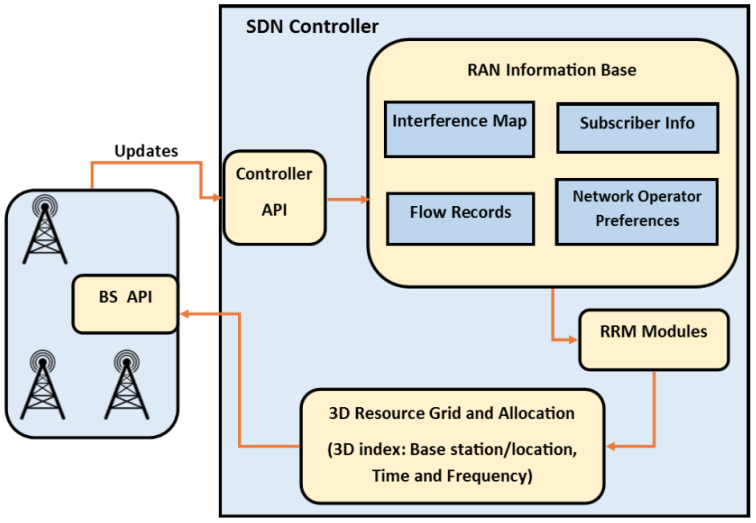


FIGURE 2.3: SoftRAN architecture [4]

2.3.4.2 NFV applied to RAN

On the other hand, some of the existing works on 5G RAN architecture focus mainly on the application of NFV principles. The ETSI NFV use case for virtualization of mobile base station [49] is one example. The authors in [49] propose RAN virtualization use case in a CRAN architecture, where the BBU functions can be executed in a Network Function Virtualization Infrastructure (NFVI) environment such as a data center. Based on this ETSI use case, the authors in [60] propose an architecture for LTE RAN virtualization. In this proposal, virtual radio processing units are responsible for executing LTE data plane stacks and a controller for interfacing virtual RAN to the core network and other RANs is devised. Similarly, the authors in [61] devise a NFV architecture for HCRAN that encompasses virtualization of radio and computing resources of both intra and inter RAN infrastructures. They emphasize on the need for transparency among virtual BSs of the same physical infrastructure. An architecture for virtualizing both RAN and core network functions with NFV is proposed in [62]. Here RRHs are directly connected to the core of the network and all BBU and core network functions are executed on Virtual Machines (VMs) on demand basis. On the other hand, the authors in [63] identify the key issues of NFV in the context of 5G and propose a network overlay concept of decoupling physical address from virtual addresses as well as isolation of traffic in virtual networks. With SDN and network overlay as enabling technologies for NFV implementation in 5G, the authors in

[63] identify candidate network functions that can be executed as virtual network functions.

In [64], a two layer architecture of radio and network cloud that also integrates dense deployments with NFV is proposed. In this architecture, the edge infrastructure is designed to perform RAN lower layer (i.e. PHY& MAC) functions and other higher level functions are defined in the cloud infrastructure. Cloud scalability is achieved by NFV and the notion of separation of coverage and capacity is emphasized in the proposal. A similar proposal called RAN as a Service or RANaaS [36] is proposed to ensure flexible functional split between central cloud, as in CRAN, and distributed operation, as in conventional mobile networks. The proposal aims to take advantage of flexible implementation of virtualized RAN functions where some functionalities remain to be executed at the BSs whereas, less delay stringent functions are placed centrally.

2.3.4.3 Network Slicing

With the advent of vertical industries (i.e., eHealth, automotive, robotics), the deployment of multiple end-to-end logical networks with different requirements on top of the mobile physical infrastructure, including the RAN segment, becomes a necessity. The management and orchestration of these virtualized (logical) networks encompassing network and IT resources is also referred to as network slicing. Network slices are E2E logical networks running on top of a physical (or even virtual) infrastructure [65]. The internals of each network slice are managed and orchestrated in an independent way by each individual vertical sector, hence being flexible enough to accommodate the different technical and business demands [66]. An important property to satisfy among network slices running in parallel is isolation. Isolation is required:

- To meet the particular per-slice KPIs,
- To attain per-slice security in the sense that attacks in one slice should not affect other slices, and
- To enable each vertical to manage each network slice in a self-contained manner.

SDN/NFV are enabling technologies for network slicing. The authors in [66] propose network slicing as a service model for mapping various requirements to service

models of operators, with the ultimate goal of enabling provision of customized end-to-end networks as a service. For the case of Mobile Virtual Network Operators (MVNOs), authors in [67] devise a broker architecture for enabling infrastructure providers to dynamically allocate portion of their offered capacity to tenants, e.g., MVNOs. The proposed broker architecture aims to allocate resources from infrastructure providers to enable demand driven allocation of their offered capacity. Thus, it supports multi-tenancy on the same physical infrastructure. The authors in [5] took a further step in slicing by proposing a network store for providing end users with a slice, according to demands, via programs that allocate and deploy necessary software and network elements. The network store architecture is shown in Fig. 2.4. It works in the same analogy of Apple's App Store and Google's Play Store, to motivate third parties to deploy network applications.

Two key challenges to enable resources sharing are resource allocation and isolation among multiple slices [68, 69]. Resource allocation mechanisms decide on how to embed a virtual wireless network onto the physical infrastructure, i.e., deciding on which resources, nodes and links to be picked and optimized based on requirement constraints from service providers [68]. Unlike wired networks, resource allocation mechanism in virtualized wireless environment is challenging due to the inherent nature of wireless communication such as interference, mobility, radio channel variability and roaming. In addition, the authors in [68] show that due to the variability in the number of end users and aggregate mobile traffic in a certain area, resource allocation mechanisms should be dynamic to avoid over and under provisioning and in some cases to ensure that the allocated virtual resources do not exceed the underlying physical substrate capacity. Once proper resource allocation is made, ensuring isolation among slices is necessary. Isolation is preventing degradation in the performance of a slice due to changes in another slice, addition of a new slice or removal of a slice [69]. Isolation in the context of wireless networks is complex due to a shared and broadcast nature of wireless medium and associated factors such as mobility, variation in RATs and interference. Both challenges, i.e., resource allocation and isolation are coupled and in some cases isolation can be translated as maintenance of the allocated resources. If, for example, the allocation mechanism ensures that there is no resource overlapping, then isolation among slices is implied. Softwarization technologies, namely SDN and NFV, in addition to being enablers for slicing, can also be applicable to alleviate the aforementioned challenges of slicing. These ideas are highlighted in [69] where the potential of SDN through decoupling of hardware and control logic can be exploited to enable per slice control plane and centralized management. In

addition, NFV enables location independent implementation of network functions according to specific scenarios. Hence, NFV helps to ease allocation and isolation issues [69].

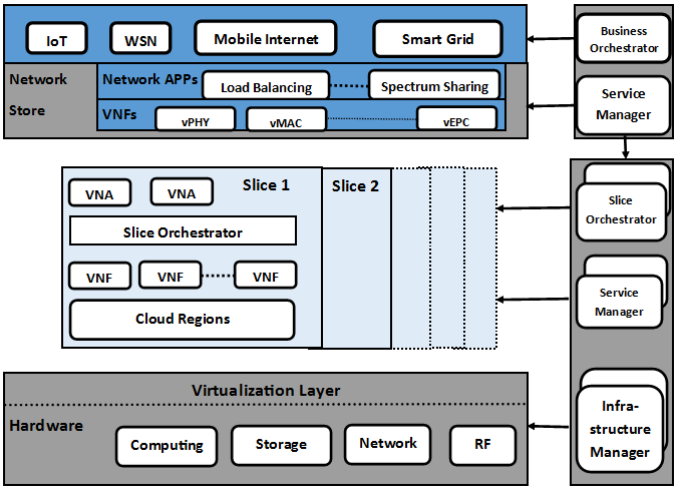


FIGURE 2.4: Creation of slices with network store [5]

2.4 EH in RAN

The common natural phenomena for scavenging energy from are sunlight and wind, but other sources such as motion, vibration and electromagnetic radiation can also be used as sources of energy to decrease the dependency of the network from the power grid. Earlier works of adopting renewable energies for cellular networks, e.g., [70], aim at regions where power supply from the grid is difficult and unreliable. But recently EH is also studied to be feasible for urban scenarios. This is mainly due to the trend of densification in RAN leading to many BSs with low transmission power requirements. This, in turn, results in lower initial investment for initial site access and harvesting hardware, that leads to low OPEX in the long term [25]. Therefore, the interest to partially or completely power BSs with energy harvesters is increasing [25]. The benefits of powering BSs with energy harvesters is not only lower OPEX, but also reduction of CO_2 emissions [25].

A system model for an EH cellular networks with local BS on/off strategy is analyzed in [71]. In this model, each BS decides when to switch on/off regardless of the state of other BSs. This model characterizes self-powered BSs by their availability metrics and optimal regions where BSs exhibit similar performance, as those with reliable energy sources. In addition to self-powering of BSs, hybrid powered BS deployments are an alternative. Hybrid powered BSs are characterized

by both grid and renewable energy supplies and the goal of system wide design is to maximize the utilization of harvested energy when they are available in order to decrease the total grid energy consumption [72].

The advances in renewable energy technologies, e.g., solar panels and wind turbines, are also paving the way for increasing interests in EHBSs. Renewable energy technologies, particularly solar panels and wind turbines are expected to enhance in efficiency and their costs will continue to decline [73]. This trend is favorable for MNOs to use EH and achieve both sustainable and cost efficient goals. Solar panels and wind turbines are the two promising EH technologies due to sufficient EH rates, longer life times with very low maintenance costs and mature industry eco-systems producing them [74]. Moreover, solar and wind turbines are complementary energy sources and hybrid supply of BSs with solar panels and wind turbines is an interesting solution to balance the challenge due to energy arrival dynamics. For instance, solar energy output is high in summer seasons, while, in many areas, wind turbine power output peaks during winter months [74]. Due to these reasons, BSs that are completely powered by renewables may combine hybrid sources of wind and solar panels.

In order to deploy EHBSs, finding optimal trade-off among network performance, density of BSs and grid power consumption is a challenge that requires careful investigation [74]. Hence, network deployment guidelines are needed to balance such trade-offs. For instance, the authors in [74] conclude that it is effective to use a higher number of EHBSs than using BSs with high EH capacity, i.e, large solar panel size and/or large diameter of wind turbine. Moreover, it is shown that when the density of EHBSs is too large or too small, the battery capacity has little impact on network performance and grid consumption reduction. The authors in [75] propose FreeNet, a joint spectrum and EH network architecture. FreeNet is promised to be useful for scenarios, such as emergency communications and rural broadband provisions, but due to dynamic nature of both spectrum and energy resources, their joint design is challenging.

Some literature also noted that EHBSs can also be part of smart grid architecture as both consumers and producers of energy [76, 77]. These ideas will open a new paradigm of integrating cellular and smart grid systems with dynamic and price aware strategies. The joint integration provides additional revenues for MNOs and performance gains for smart grid operators. The energy management concept in traditional cellular networks, where its main goal was only the reduction of power

consumption, is evolved with EH to include not only energy efficiency goals but also sustainability and smart grid awareness objectives [77].

While many literature study the issues of EH in wireless networks, including the challenge of having intelligent energy management by applying methods of ML and optimization (will be described in Section 2.5), most of the works are based on the HetNet architectures. Studies of EH applied to other architectures such as CRAN and FRAN are not widely available. EH with CRAN is studied in [78]. Here, the authors propose a green CRAN architecture where the RRHs of CRAN are powered by renewable energies and algorithms for sustainable resource management to ensure QoS and sustained operation of RRHs are proposed. In [79], a virtual distributed load balancing scheme that balances the trade-off between QoS and green energy availability is studied. The algorithm is designed to be applied on the central controller of the SDN based RAN architecture and is shown to sacrifice little QoS, such as increase in latency, for a higher proportion of energy benefit. In addition, the energy fluctuation due to EH and the mechanism of energy cooperation among RRHs in HCRAN is highlighted in [61]. However, it is a more generic inclusion of EH in HCRAN without specific EH related details. On the other hand, the authors in [80] define an architecture for EH with FRANs. They identify two critical policies, which require careful design in EH MEC systems: (i) offloading policy to determine how much work is offloaded to a central cloud and (ii) an auto-scaling policy to determine how much server capacity is provisioned.

An inherent nature of EH communication is the intermittent and unreliable nature of the energy sources. In order to cope with these challenges, EH communication systems require new protocols, planning, cooperation schemes and possibly techniques of using hybrid sources of energy and real-time interaction with smart grid.

2.5 Role of Optimization Tools

Future RANs will have to handle diverse QoS and QoE requirements of thousands of heterogeneous devices, multiple slices/services on the same infrastructure, multi-RATs, dynamic energy availability, possible interaction with the smart grid as well as dynamic reconfiguration decisions where to perform certain network functions. Handling these decisions optimally and automatically requires a control platform with tools such as ML and DP. In what follows, we focus on three application areas, namely, optimization in EHBSs, optimization in CRAN and general learning frameworks.

2.5.1 Optimization in EHBSs

As a result of dense deployment of BSs combined with the increasing significance of energy sustainability, intelligent energy management in EHBSs has been the focus of many studies in the research community. Most of these literature analyze hierarchical multi-tier networks (HetNets), with an intelligent switching on/off scheduling of BSs. The authors in [81] apply DP to determine the optimal switch on/off policy in a two-tier HetNets with baseline MBS and hot-spot deployed SCs. The solution shows the performance bound of an intelligent switch on/off policy when all the system dynamics information are known a-priori. Minimizing the grid energy consumption for hybrid powered BSs is also studied in [82]. Here, the authors applied a two stage DP methods designed to achieve energy saving gains while maintaining probability of blocking. The authors in [83] apply a ski-rental framework based on-line algorithm for optimal switch on/off scheduling for minimizing the operational costs of a network composed of self powered BSs. The work in [84] gives a broad scope on RL based methods for energy efficient management of mobile networks powered by renewables with characterization of solar/PV sources. Moreover, the authors in [85] studied sleep mode coordination between BSs powered by EH and grid energy using DP. However, the DP based solution is shown to entail high computational complexity.

On the other hand, the authors in [86] apply RL, in particular QL algorithm, to optimize the harvested energy utilization. The work is based on distributed QL where each renewable powered BS takes autonomous decision whether to switch on/off according to energy arrival, energy storage and traffic demand. In addition, multi-armed bandit based distributed learning is studied in [87] to allow each SC to learn its own energy-efficient policy. The authors in [88, 89] applied layered learning for system wide harvested energy allocation through decomposition of the problem into two layers. The first layer, based on RL, is in charge of local control at each SCs and the second layer, based on artificial neural networks, ensures network wide coordination among the SCs. The authors in [90] applied deep RL methods for minimization of energy consumption in HetNets through optimal activation of subset of SCs while maintaining the desired QoS. In particular, they have applied Actor-Critic RL methods where deep neural networks are used as policy and value function approximators. Moreover, renewable energy allocation in edge computing devices with EH is studied in [80]. Here, the authors propose RL based on-line solutions for offloading and auto-scaling in edge computing devices that are powered by EH. On the other hand, the authors in [91] proposed

a RL based energy controller for a SC powered by EH, battery and smart grid by considering battery aging effects. This work is based on FQL and is shown to provide significant extension to the life time of a SC battery. However, this work is limited to a single SC and a coordinated energy management among BSs within a mobile network remains an open issue. Even though many literature exist for attaining intelligent energy management algorithm in RANs with EH, most of them are on a single SC scenario and literature on multi-SCs scenario focus only on switch on/off policies. Therefore, there is a need of more investigations to integrate EH and flexible functional split options in MEC-enabled RAN. Here, we claim that functional splits provide insights into considering more configuration options of SCs, in addition to switch on/off and enable higher grid energy savings.

It is important to note that the application of optimization tools in EHBSs is highly dependent on the underlying mobile traffic load and energy arrival models. However, accurate representation of traffic load and energy arrival information is not straightforward, since both entail spatial and temporal variations. Hence, spatio-temporal analysis of both mobile traffic and energy arrival data are necessary. To this end, the authors in [9] analyze aggregate mobile traffic data of a number of operator owned BSs in an urban scenario to visualize and represent the temporal variation of the mobile traffic load in specific geographical positions. They have developed a model that combines both time and location information for analyzing large scale cellular network data. They provide a spatial distribution of cellular traffic in terms of geographical traffic density. The spatial distribution shows the traffic generation at different times of the day in a certain geographical area. The result reveals strong correlation between temporal and spatial characteristics. Moreover, the temporal distribution analysis provides the traffic pattern in different time scales such as hourly, daily and weekly basis. In addition, their results show that such an urban coverage area exhibits five mobile traffic patterns, namely residential, office, entertainment, transport and comprehensive, with a model that also captures the daily mobile traffic variation in different hours both in week days and weekends. However, cellular network data is of a large volume with variety of information, which makes it big data. Nonetheless, an extensive big-data analysis of cellular network data including their spatio-temporal distribution is missing in the literature. One of the challenges for a lack of large-scale mobile traffic data analytics is the lack of data that captures realistic traces of cellular network activities, e.g, service based requests. Most of the data are operator owned with limited availability for research.

Moreover, energy arrival also exhibits spatio-temporal variation. Hence, accurate models for representing harvested energy with a reasonable time granularity are required for successful application of optimization tools. For this reason, the authors in [92] developed a model that can be used as a tool to determine the amount of harvested solar power for small form-factor devices, including SBSs. The model is based on extensive database of solar irradiation for many years and it gives a reasonable estimate of expected energy arrival considering time of the day, month and status of the day. In addition, the authors in [93] also propose a solar irradiance model that can capture the small time-scale (in order of minutes) fluctuations. However, both papers exploit only the temporal variation of solar energy arrival. On the other hand, the authors in [94] develop a solar irradiance model considering the spatio-temporal variation of solar energy arrival. The model, by exploiting the spatio-temporal correlation, is proved to be of high accuracy. However, it is developed for large-scale power utility applications. In addition to solar, other ambient energy sources, such as wind, also exhibit spatio-temporal variation. For instance, the authors in [95] propose a model of wind energy forecasting by exploiting the spatio-temporal correlation of large data set of wind speed and direction. However, most of the literature focus on large-scale power system applications. Models of EH sources that focus on EH communication devices such as SCs require further investigation. A key input for developing such models is data. As a result, the National Renewable Energy Laboratory (NREL) maintains national solar radiation database [96] that provides solar radiation data of US territories for 30 or more years. Such open database encourages further investigation in the field of energy source modeling and forecasting and need to be adopted by other regions as well as for other energy sources.

2.5.2 Optimization in CRAN

Optimization tools are necessary for CRAN planning and placement of virtual network functions. The authors in [97] investigate the BBU pool placement problem in CRAN with the goal of minimizing deployment costs with constraints of processing capacity, synchronization (latency) and traffic demands. The studied cellular network is served by certain number of BBU pools and RRHs. The investigation leads to NP-hard integer programming problem and an algorithm base on local search is proposed. A similar approach is also taken in [98], where virtual network function placement and assignment problems are investigated. The placement problem deals with which of the available network nodes can be used as

BBU servers, whereas, the assignment problem deals with which subset of RRHs are assigned to a specific BBU server. These problems are formally formulated as binary integer linear programming problem. The ultimate goal here is minimizing server cost, fronthaul costs and latency. In addition, an approximate and faster algorithm is proposed in cases when the size of the network is large and many instances of RRH and BBU servers are possible. The authors in [99] adopt a different approach to tackle the problem of splitting BB functions between distributed units and central units. A graph based model of BB transceiver structure is proposed by assigning weights corresponding to computational and front-hauling costs. An optimal splitting problem is then converted to graph clustering problem and the work uses genetic algorithm to solve the problem imposing a path delay constraint. Genetic algorithms are also applied for dynamic RRH to BBU mapping challenge in CRAN [100]. The proposal aims to achieve dynamic BBU - RRH reconfiguration according to traffic conditions. It is shown that such dynamic mapping between RRHs and BBUs results in enhanced QoS.

Most of the literature focus on enabling dynamic RRH-BBU mapping in CRAN. However, the application of optimization tools in CRAN is far beyond. With the notion of flexible functional split and fog/edge nodes, not only the dynamic mapping between RRH-BBU is necessary, but also where to place certain network functions optimally, considering QoS demands and available resources including energy. This involves network wide data collection, processing the collected data and applying optimization tools to decide on, where and when to deploy certain network functions as well as maintenance of their life cycles. In this regard, the authors in [101] propose an optimal flexible functional split option selection scheme for a CRAN architecture with Radio Remote Unites (RRUs) supplied by renewables. They have shown that optimal functional split selection problem can be formulated as a convex optimization and propose an online heuristic algorithm that does not rely on future energy arrival information. However, the study in [101] is focused only on throughput maximization and the case of multi-RRUs is not explored. In conclusion, there is a gap in the literature in integrating EH and flexible functional split options in MEC-enabled RAN.

2.5.3 General learning frameworks

Some proposals emphasize the importance of a general purpose learning framework for multi-purpose optimization goals. Such platforms can be used to generate single purpose RRM policy by learning on the collected network wide data. The

learning platforms should account for noisy nature of data collection and ensure prediction of what measurements are to be expected. An example of network-wide ML based cognition platform is proposed in [102]. The authors identified how Generative Deep Neural Networks (GDNNs) are suited for network wide cognition and full self-organization capability. They also pointed out that the application of un-supervised learning, using GDNN from input network wide measurements, can result in high level abstraction representation of the input measurement data. As a result, supervised and reinforcement learning applications for specific tasks can be developed using such high level abstractions. The proposed framework is called COgnition BAsed NETworkS (COBANETS) which involves cognitive network nodes with an infrastructure for learning, modeling and optimization. The framework is an example of the need for combination of ML and softwarization, in particular SDN, as a re-configuration tool. COBANETS motivates its design on three recent trends; (i) recent ML advances, particularly in the unsupervised deep learning field, which is suitable for unlabeled data as the cellular network data is massive and usually unlabeled, (ii) recent research advances in re-configuration tools namely SDN and NFV and (iii) the advancement in the computational power in todays state-of-the-art processors.

The authors in [103] also emphasize the need for a shift from single purpose RRM algorithms to general purpose RRM framework that is capable of generating control policies automatically. In particular, they propose a general RL based RRM framework, that can be applied to generate and update control policies for various resource management objectives, based on experience, namely data gathered by nodes in the network. Similarly a network wide intelligent architecture, which combines the benefits of SDN and ML for self-automation is described in [104]. Their design is based on the novel knowledge-plane idea proposed by [105] and applied to SDN-enabled networks. Their proposal transforms the network-wide collected data into knowledge by leveraging ML, hence resulting a knowledge-defined networking paradigm. However, these general learning frameworks are faced with many obstacles that hinders their practical application. These challenges range from data collection, appropriate format of data representation, identifying domain specific structures including spatio-temporal correlations and lack of real-time test-beds for experimental evaluation.

2.6 BS Power Consumption Model

In this section, we explain the power consumption model adopted for a scenario described in Section 1.2. The BS power requirement is usually modeled as linearly dependent on the load with an additional baseline load independent component [106]. However, our scenario considers vSCs powered by EH which are relying on the MBS resources for full or part of their BB processing requirements. Hence, the power consumption requirement of each vSCs as well as the MBS are dependent on the functional split options selected. As a result, a more flexible power model for estimating the relative power consumption requirements of BB functions is required. A flexible BS power consumption model is proposed in [6, 7]. In these works, more detailed power model of BS components and sub-components are proposed, relying on the split of BSs into a number of components and sub-components as shown in Fig. 2.5. According to the model, the main sub-components of a BS are the digital BB, the analog RF, the Power Amplifier (PA) and the power system (power conversion and cooling). The model in [6, 7] is a general flexible power model of BSs and provides the power consumption in Giga Operation Per Second (GOPS). Technology dependent GOPS to Watt conversion factor is applied to determine the power consumption in Watts.

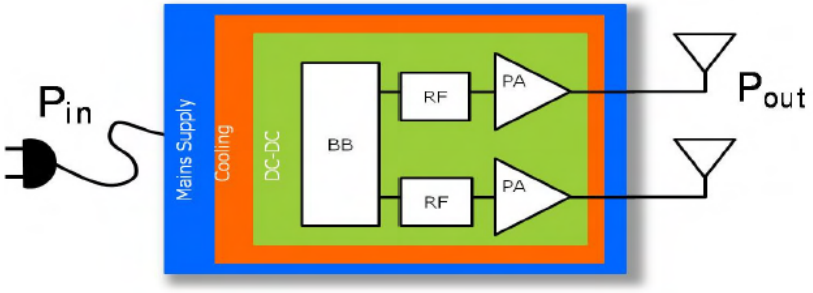


FIGURE 2.5: BS components in the power models of [6, 7]

It is important to have an estimation of power consumption requirements of different functional splits in our scenario. Moreover, based on the analysis in [6, 7] regarding the BB tasks with significant energy requirements, we have considered three functional split options as targets. These are: PHY-RF, UpperPHY-LowerPHY and MAC-PHY splits. Hence, based on the general model descriptions in [6, 7], we have mapped the various BB processing tasks of the three functional split options to their power requirement estimations. The main BB tasks associated with these functional split options are shown in Fig. 2.6.

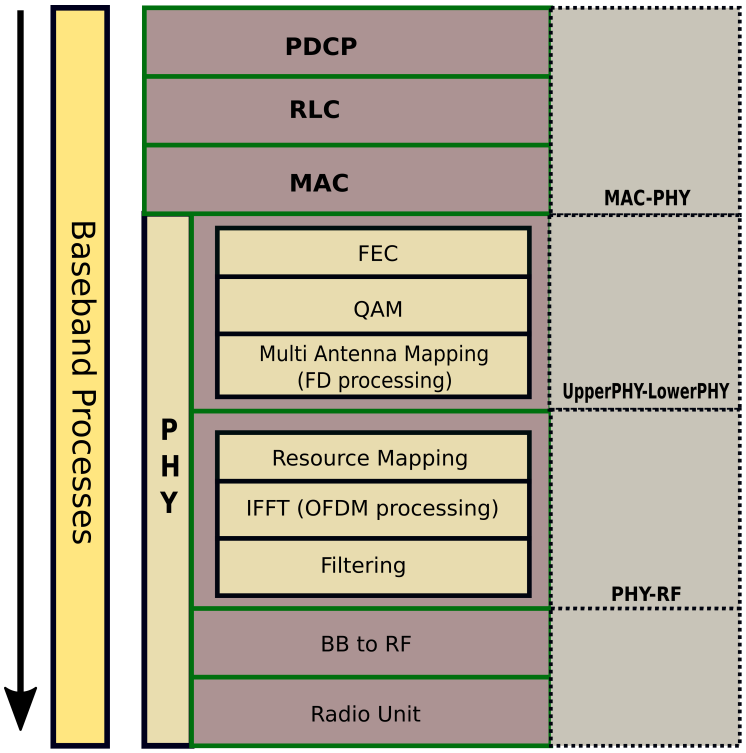


FIGURE 2.6: BB functions with the considered functional split options

The total BS power consumption is given by:

$$P_{\mathrm{BS}} = P_{\mathrm{BB}} + P_{\mathrm{RF}} + P_{\mathrm{PA}} + P_{\mathrm{overhead}} \tag{2.1}$$

where P_{BB} is the power consumption due to baseband processing, P_{RF} is the power consumption due to RF, P_{PA} is the power consumption by the power amplifier and P_{overhead} is the overhead power consumption, e.g., cooling system.

The BB power consumption, P_{BB}, is generally computed as:

$$P_{\mathrm{BB}} = P_{\mathrm{BB1}} + P_{\mathrm{BB2}} \tag{2.2}$$

More in detail, P_{BB1} is given by:

$$P_{\mathrm{BB1}} = [P_{\mathrm{CPU}} + P_{\mathrm{OFDM}} + P_{\mathrm{filter}}] \tag{2.3}$$

where P_{CPU} is the idle mode power consumption, P_{OFDM} is the power consumption due to OFDM processes and P_{filter} is the power consumption due to filtering. In addition, P_{BB2}, is given by:

$$P_{\mathrm{BB2}} = [P_{\mathrm{FD}} + P_{\mathrm{FEC}}] \tag{2.4}$$

where P_{FD} is the frequency domain processing power consumption and P_{FEC} is the power consumption due to FEC processes. The power consumption values mainly depends on bandwidth, number of antennas and the load fraction. In particular, P_{FD} and P_{FEC} are dependent on the traffic load. The power dependence on these factors can be both linear and exponential [6]. The BB power consumption of the vSC depends on the adopted functional split option, in particular it is given as:

$$P_{\mathrm{BB}}^{\mathrm{vSC}} = \begin{cases} 0, & \text{if vSC is in PHY-RF} \\ P_{\mathrm{BB1}}, & \text{if vSC is in UpperPHY-LowerPHY} \\ P_{\mathrm{BB1}} + P_{\mathrm{BB2}}, & \text{if vSC is in MAC-PHY} \end{cases} \tag{2.5}$$

The power consumption of the MBS is determined by (2.1). The power consumption of MBS includes two components: the power required for serving UEs attached to the MBS and an additional power required by its MEC server for the BB processing of vSCs in the same mobile cluster, which depends on the functional split option selected by each vSC. For instance, when the vSCs are in PHY-RF split mode, the vSC power consumption model does not include the corresponding P_{BB}, since the BB processing takes place at the MBS site. Instead, the corresponding P_{BB} term is added to the MBS. On the other hand, in MAC-PHY split mode, the vSC power consumption includes the BB power consumption term and is given in (2.1). Considering the aforementioned model description, the power consumed by the MBS is computed as:

$$P_{\mathrm{m}} = P_{\mathrm{BS}}^{\mathrm{MBS}} + \sum_{i \in \mathcal{G}} P_{\mathrm{BB}}^{i} \tag{2.6}$$

where $P_{\mathrm{BS}}^{\mathrm{MBS}}$ is the power consumption of the MBS computed as in (2.1), P_{BB}^{i} is the baseband power consumption of the i-th vSC and $\mathcal{G}$ is the set containing the vSCs in PHY-RF or UpperPHY-LowerPHY split modes.

Chapter 3

Reinforcement Learning

3.1 Introduction

RL is a learning paradigm that relies on learning by interacting with the environment without an exemplary supervision [8]. It gives the ability to learn behaviors online and adapting automatically to the temporal dynamics of the system. It is a well known framework of solving a problem of learning from interactions to achieve a goal, also known as a sequential decision making problem. In RL, the learning agent senses the state of the environment, takes actions and transits in to a new state. The environment returns a scalar feedback known as reward which represents the immediate impact of the particular action. Hence, RL is applied in creating autonomous systems that can improve themselves through agent-environment interaction experience. In many applications of RL, the learning process is formulated in a centralized fashion where a single entity takes actions, e.g. a robot in a factory control, a base station in mobile networks. However, centralized formulation of RL has its own challenges mainly due to large state-action spaces limiting its scalability. An alternative approach is a decentralized/distributed formulation of RL which is also known as multi-agent RL.

In what follows, we explain briefly a single-agent RL framework in Section 3.2. DP and temporal difference algorithms including QL, SARSA and FQL are briefly introduced in Sections 3.2.3 and 3.2.4 respectively. A more recent class of RL, known as Deep Reinforcement Learning (DRL) is also presented in Section 3.3 and Section 3.4 gives a brief explanation about multi-agent RL.

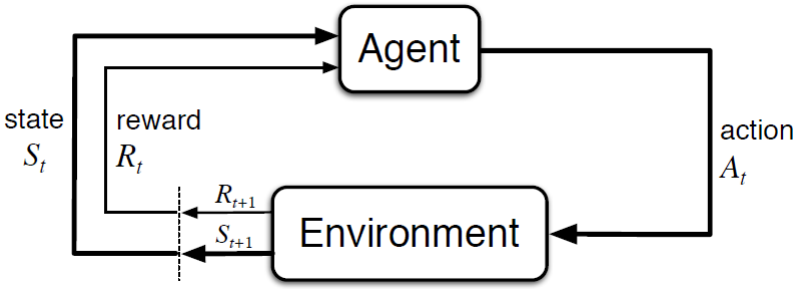


FIGURE 3.1: Agent-environment interaction in MDP [8]

3.2 Single-Agent RL

These classes of problems are formalized in terms of Markov Decision Processes (MDPs). MDPs are classical formalization of sequential decision making, where actions influence not just immediate rewards, but also subsequent situations or states, and through those, future rewards. Thus MDPs involve the notion of delayed reward and the need to trade off between immediate and delayed reward. In MDP frameworks, the learner /decision maker is usually called the agent. The thing it interacts with, comprising everything outside the agent, is called the environment. These interactions happen continually, i.e. the agent selecting actions and the environment responding to these actions and presenting new situations to the agent. The environment also gives rise to rewards, special numerical values/feedback that the agent seeks to maximize over time through its choice of actions. A diagram of agent-environment interactions is shown in Fig. 3.1. More specifically, the agent and environment interact at each of a sequence of discrete time steps, $t = 0, 1, 2, 3, ...$ At each time step t, the agent receives some representation of the environments state, $S_t \in S$, and on that basis selects an action, $A_t \in A(s)$. One time step later, in part as a consequence of its action, the agent receives a numerical reward, $R_{t+1} \in R$, and finds itself in a new state, S_{t+1}. The MDP and agent together thereby give rise to a sequence or trajectory that begins like this: $S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, R_3, ...$

In finite MDP, the sets of states S, actions A and reward R have finite number of elements. As a result, both the immediate reward R_t and the state S_t have well defined probability distributions dependent only on the preceding state and actions, given in (3.1).

$$p(s', r|s, a) = Pr\{S_t = s', R_t = r|S_{t-1} = s, A_{t-1} = a\} \tag{3.1}$$

The function p defines the dynamics of the MDP, i.e. mapping states to probabilities. In MDPs, the transition probabilities given by p in (3.1) completely characterize the environment's dynamics. That is, the probability of each state S_t and reward R_t depends only on the immediately preceding state S_{t-1} and action A_{t-1}. Hence, the state must include information about all aspects of the past agent-environment interaction. These states are said to have Markov property.

MDP framework provides a considerable abstraction of the problem of goal-directed learning from interaction. It proposes that any problem of learning goal-directed behavior can be reduced to three signals, which are:

- choices made by the agent (the actions),
- the basis on which the choices are made (the states), and
- a signal to define the agent's goal (the rewards).

Consequently, RL problem is defined in terms of states, actions and rewards. Through the RL learning process, according to the current state, the agent executes a certain action and receives an immediate reward and as a result of the action, its environment will evolve to a new state. It is important to note that in RL, the rewards can be delayed. Hence, it is a sequential decision making process with the goal of maximizing cumulative reward.

3.2.1 Returns

The RL goal is to maximize the cumulative reward the agent receives in the long run. This goal is defined formally as maximizing the *expected return*, where the return, denoted G_t, is defined as some specific function of the reward sequence. In the simplest case, the return is defined as the sum of the rewards (3.2).

$$G_t = R_{t+1} + R_{t+2} + R_{t+3} + ... + R_T \tag{3.2}$$

where T is a final time step. This definition is valid for application in which there is a natural notion of final time step, i.e. when agent-environment interaction breaks naturally into sub-sequences known as *episodes* and the last state is called the terminal state. The tasks in these setup are known as episodic tasks. On the other hand, in many cases the agent-environment interaction does not break

naturally into identifiable episodes, but goes on continually. These are known as continuing tasks. The return formulation in (3.2) is problematic for continuing tasks since $T = \infty$ which makes the return be infinite. As a result, we need to introduce the concept of discounting. Accordingly, the agent tries to select actions so that the sum of the discounted rewards it receives over the future is maximized. In particular, it chooses A_t to maximize the expected discounted return defined in (3.3).

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + ... = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \tag{3.3}$$

where γ is a parameter, $0 \leq \gamma \leq 1$, is called the discount rate and controls how strongly the agent takes future rewards into account. For instance, if γ approaches 1, the return objective takes future rewards into account more strongly, i.e. the agent is more farsighted.

3.2.2 Policies and Value Functions

RL algorithms involve estimating value functions. Value functions are functions of states (or state-action pairs) that estimate *how good* it is for the agent to be in a given state or *how good* it is for the agent to perform a given action in a given state. The level of *how good* is defined in terms of future rewards that can be expected, i.e. expected return. The expected return depends on what actions the agent will take. Hence, value functions are defined with respect to particular ways of acting, known as policies. Formally, a *policy* is a mapping from state to probabilities of selecting each possible action. If an agent is following policy π at time t, then $\pi(a|s)$ is the probability that $A_t = a$ if $S_t = s$.

The value function of a state s under a policy π, denoted as $v_\pi(s)$, is the expected return when starting in s and following π thereafter. Value function can be defined formally in (3.4).

$$v_\pi(s) = \mathbb{E}_\pi[G_t|S_t = s], \text{for all } s \in S \tag{3.4}$$

where $\mathbb{E}[.]$ denotes the expected value of a random variable given that the agent follows policy π, t is any time step and G_t is defined in (3.3). The function v_π is known as *state-value function* for policy π. Another important formulation of value function is to estimate the value of taking action a in state s under a policy

π, denoted as $q_\pi(s, a)$, as the expected return starting from s, taking the action a and thereafter following policy π. The function q_π is known as the *action-value function* for policy π and is formally defined in (3.5).

$$q_\pi(s, a) = \mathbb{E}_\pi[G_t | S_t = s, A_t = a] \tag{3.5}$$

The value functions v_π and q_π can be estimated from experience. For instance, if an agent following policy π maintains an average of the actual returns following each state encountered, then the average will converge to the value of the state as the number of times the state is encountered approaches infinity. These estimation methods are known as Monte Carlo methods. However, if there are many states, it may not be practical to keep separate averages for each state individually.

A fundamental property of value functions is that they satisfy recursive relationships. For any policy π and any state s, the following consistency condition holds between the value of s and the value of its possible successor states:

$$\begin{aligned} v_\pi(s) &= \mathbb{E}_\pi[G_t | S_t = s] \\ &= \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1} | S_t = s] \\ &= \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r|s, a)[r + \gamma \mathbb{E}_\pi[G_{t+1} | S_{t+1} = s']] \\ &= \sum_a \pi(a|s) \sum_{s', r} p(s', r|s, a)[r + \gamma v_\pi(s')], \text{for all } s \in S \end{aligned} \tag{3.6}$$

The expression in (3.6) is a sum over all values of the three variables, a, s' and r. For each triple, we compute the probability, $\pi(a|s)p(s', r|s, a)$ and weight the quantity in brackets by that probability and sum over to get an expected value. Equation (3.6) is known as the Bellman equation for v_π and expresses the relationship between the value of a state and the values of successor states. The Bellman equation is the basis of a number of methods to compute, approximate and learn the value of a policy v_π.

3.2.3 Dynamic Programming

DP is a collection of algorithms that can be applied to compute the optimal policies given a perfect model of the environment as a MDP [8, 107]. DP algorithms have only limited application in RL due to their assumption of a perfect model and computational expenses. DP algorithms are obtained by turning the Bellman

equation in (3.6) into update rules for improving approximations of the desired value functions.

3.2.3.1 Policy Iteration

It is one of the most widely used method for estimating an optimal policy π_* and its corresponding value v_*. Starting from an arbitrarily policy π, policy iteration method involves a series of policy evaluation to estimate v_π and improvement to get a better policy π'. These sequence guarantees that each policy is a strict improvement over the previous one unless it is already the optimal one. Since a finite MDP has only a finite set of policies, this process converges to an optimal policy and optimal value function in a finite number of iterations. The procedure of policy iteration is shown in Algorithm 1.

Algorithm 1 Policy Iteration for estimating π_*

1. Initialize $V(s) \in \mathbb{R}$ and $\pi(s) \in A(s)$ arbitrarily for all $s \in S$
2. Policy Evaluation

for $\delta > \theta$ (a small positive number) **do**:
  $\delta \leftarrow 0$
  for each $s \in S$ **do**:
    $v \leftarrow V(s)$
    $V(s) \leftarrow \sum_{s',r} p(s', r|s, \pi(s))[r + \gamma V(s')]$
    $\delta \leftarrow max(\delta, |v - V(S)|)$
  end for
end for

3. Policy Improvement

policy-stable $\leftarrow$ *true*
for each $s \in S$ **do**:
  old-action $\leftarrow \pi(s)$
  $\pi(s) \leftarrow argmax_a \sum_{s',r} p(s', r|s, a)[r + \gamma V(s')]$
  if *old-action* $\neq \pi(s)$ **then**:
    policy-stable $\leftarrow false$
  end if
  if *policy-stable* **then**:
    stop and return V and π
  else
    go to step 2
  end if
end for

3.2.3.2 Value Iteration

The main drawback of policy iteration method is that it involves policy evaluation, which is an iterative computation requiring multiple sweeps through the

state set. As an improvement to policy iteration, policy evaluation steps can be truncated without losing the convergence guarantees of policy iteration. When, policy evaluation is truncated after just one sweep (one update for each state), the algorithm is known as value iteration. Value iteration is simply the Bellman equation in (3.6) turned to an update rule. The procedure for value iteration is shown in Algorithm 2.

Algorithm 2 Value Iteration for estimating π_*

Algorithm parameter: a small threshold $\theta > 0$ determining the accuracy of estimation
Initialize $V(s)$ arbitrarily for all $s \in S$
for $\delta > \theta$ **do**:
  $\delta \leftarrow 0$
  for each $s \in S$ **do**:
    $v \leftarrow V(s)$
    $V(s) \leftarrow max_a \sum_{s',r} p(s', r|s, a)[r + \gamma V(s')]$
    $\delta \leftarrow max(\delta, |v - V(S)|)$
  end for
end for
Output a deterministic policy π such that
$\pi(s) = argmax_a \sum_{s',r} p(s', r|s, a)[r + \gamma V(s')]$

Even though, we have seen the notion of optimal value functions and optimal policies, in practice this rarely happens. In many applications, optimal policies can be generated only with extreme computational cost. Even if we have an accurate model of the environment's dynamics, it is usually not possible to compute an optimal policy due to computational power available to the agent. Moreover, in addition to computation, memory availability is another constraint. A large amount of memory is required to build-up approximations of value functions, policies and models. In cases of small, finite state sets, it is possible to form approximations using tables with one entry for each state (or state-action pair). These methods are called tabular methods. Hence, we are forced to settle for approximations in RL problems. The online nature of RL makes it possible to approximate optimal policies in ways that put more effort into learning to make good decisions for frequently encountered states, at the expense of less effort for infrequently encountered states. As such, RL is a way of approximately solving MDPs.

3.2.4 Temporal Difference RL

Temporal Difference (TD) learning is a combination of Monte Carlo ideas and DP ideas. That means, TD learning involves learning directly from raw experience without a model of the environment's dynamics and updating estimates based

in part on other learned estimates, without waiting for a final outcome. Hence, TD methods involve estimating the values of state-action pairs by making an update at each time-step without waiting for a final return [8]. TD learning involves the balance between choosing a random action to discover new knowledge (exploration) or choosing an action with maximum q value (exploitation). This is controlled by the policy used during the learning phase. An example and widely used exploration/exploitation policy is ϵ-greedy which chooses a random action with probability ε or an action with maximum q value with probability $1 - \varepsilon$, where ε is known as exploration rate parameter. The algorithms in TD learning can be an on-policy or off-policy methods. In on-policy algorithms, learning an optimal policy is done using the current estimate of the optimal policy where as, in an off-policy methods, learning to approximate the optimal behavior is done independently of the current policy being followed. Here, we specifically describe QL and SARSA algorithms which belong to TD off-policy and on-policy algorithm classes, respectively.

3.2.4.1 QL: Off-Policy TD Control

QL is an off-policy RL algorithm that can learn the optimal values for each state-action pairs, also known as Q values. As long as all state-action pairs are visited and continued to be updated, QL guarantees an optimal behavior regardless of the specific policy being followed throughout the learning phase. The equation for updating the Q-values is given by (3.7). The procedure of QL algorithm is shown in Algorithm 3.

$$Q(S_t, A_t) = Q(S_t, A_t) + \alpha(R_{t+1} + \gamma \max_{\boldsymbol{a}} Q(S_{t+1}, a) - Q(S_t, A_t)) \tag{3.7}$$

where α is the learning rate, γ is the discount factor, A_t is the current action, R_{t+1} is the immediate reward, S_t and S_{t+1} are the current and the next state respectively. The policy still has an effect as it determines which state-action pairs are visited and updated. However, all that is required for correct convergence is that all values of state-action pairs continue to be updated. The QL algorithm is shown in procedural form in Algorithm 3.

Algorithm 3 QL Algorithm

Initialize $Q(S, A) \forall S \in \boldsymbol{S}, A \in \boldsymbol{A}$ arbitrarily
for each episode **do**:
  Initialize S_t
  for each step, t, of episode **do**:
    Choose A_t from S using policy derived from Q (e.g., ε-greedy)
    Take action A_t, get reward R_{t+1} and next state S_{t+1}
    update Q-value using (3.7)
    $S_t = S_{t+1}$
  end for
end for

3.2.4.2 SARSA: On-Policy TD Control

On-policy methods involve estimating the action value function $q_\pi(s, a)$ for the current behavior policy π and for all states s and actions a. In these methods, the transitions from state-action pair to state-action pair are used to learn the values of state-action pairs. Hence, every element of the transition, $(S_t, A_t, R_{t+1}, S_{t+1}, A_{t+1})$ are used in updating the values of state-action pairs, shown in (3.8). This gives rise to the name SARSA for the algorithm. In SARSA, the update of the state-action pair values is done after observing a transition from one state-action pair to the next state-action pair and this transition is dependent on the policy used to select the actions at each time step. The procedure of SARSA algorithm is shown in Algorithm 4.

$$Q(S_t, A_t) = Q(S_t, A_t) + \alpha(R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)) \tag{3.8}$$

where α is the learning rate, γ is the discount factor, A_t and A_{t+1} are the current and next actions respectively, r_t is the immediate reward, S_t and S_{t+1} are the current and the next state respectively.

Algorithm 4 SARSA Algorithm

Initialize $Q(S, A) \forall S \in S, A \in \boldsymbol{A}$ arbitrarily
for each episode **do**:
  Initialize S_t
  Choose A_t from S using policy derived from Q
  for each step, t, of episode **do**:
    Take action A_t, get reward R_{t+1} and next state S_{t+1}
    Choose A_{t+1} from S_{t+1} using the policy derived from Q
    update $Q(S_t, A_t)$ using (3.8)
    $S_t = S_{t+1}$
    $A_t = A_{t+1}$
  end for
end for

3.2.4.3 Fuzzy Q-Learning

FQL involves the use of both Fuzzy Inference Systems (FISs) and QL. FIS is the process of mapping a set of input control signals to a set of output actions through fuzzy rules [108]. FIS are mainly applicable in systems that cannot be represented by explicit mathematical models through approximation of system knowledge in a similar way to human perception and reasoning. The design of FIS involves the following steps [108]:

1. *Fuzzification of the crisp input signals*: crisp values are the exact values as read from sensors or measurements. Fuzzification of crisp input signals is done by defining the fuzzy sets and membership functions of the input signals. Hence, the state space is partitioned into various fuzzy sets through membership functions. Each fuzzy set is associated with linguistic terms such as "high" or "low". The most common membership functions are triangular and trapezoidal.

2. *Defining the rule base*: this step involves defining the behavior of the controller in terms of control actions using linguistic variables defined in the first step. This corresponds to a series of "if ... then" rules for each combinations of fuzzy sets of input signals. For deriving these rules, expert knowledge and experience is generally used.

3. *Defuzzification*: this step reverts back the results from the fuzzy rule base in step 2) back to crisp mode and activates the output action.

FIS elements are shown in Fig. 3.2. The limitations of FIS arise from the rule base definition in step 2). FIS requires an expert knowledge to define the consequents of each rule (each combination of input signals and fuzzy sets). However, most of the consequents can not be easily deduced using previous knowledge/experience which can result in lower performance of the FIS. To overcome these challenges, FIS is applied in combination with RL, in particular QL, to learn the consequents of each rule.

Even though QL is one of the most widely used RL algorithms, it can be inefficient for large state-action spaces and cannot be directly applied in problems involving continuous state-action spaces. In such cases, fine grained discretization of the state-action space helps, but at a cost of an exponential increase in the state space, which makes the learning process slow. In order to overcome these limitations,

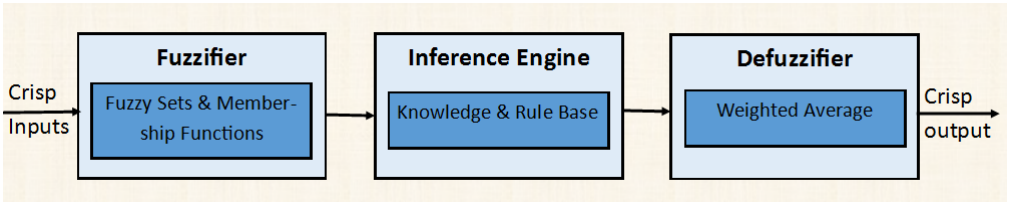


FIGURE 3.2: FIS elements

fuzzy functions approximation can be used with QL and achieve a more smooth action transition in response to a smooth change in states, without the need for fine grained discretization. FQL allows to integrate the benefits of FIS in QL: provide good approximations of the Q-function and enable the use of QL in continuous state spaces [109]. In FQL, let $\boldsymbol{X}$ be the crisp set of inputs defining the state of a learning agent. Crisp values are the exact values of the inputs without any form of pre-processing. The process of converting the crisp values to fuzzy values is known as fuzzification. Fuzzification is done according to the degree of membership determined from membership functions. Each fuzzy rule corresponds to a state and its firing strength defines the degree to which the agent is in that state. Unlike FIS, in FQL, rules do not have fixed consequents.

The consequents of each rule are learned through exploration/exploitation algorithm. The resulting FIS will have competing actions for each rule and each rule with have the following form:

if $\boldsymbol{X}$ is X_i then $A[i,1]$ with $q[i,1]$
or $A[i,j]$ with $q[i,j]$
.
.
.
or $A[i,k]$ with $q[i,k]$,

where $A[i,k]$ is the k^{th} possible action in rule i and $q[i,k]$ its corresponding q value.

Each fuzzy rule is corresponding to a state. A state X_i is defined as: (x_1 is $X_{i,1}$ and x_2 is $X_{i,2}$ and .... x_n is $X_{i,n}$), where $X_{i,j}$, $i = 1, ..., n$ and $j = 1, ..., n$ are the fuzzy sets corresponding to the membership functions of each crisp inputs x_i. The procedure of FQL algorithm is shown in Algorithm 5.

Algorithm 5 FQL Algorithm

Initialize q-values $q(i,k)$ for each rule i and number of possible actions A_k

Observe the crisp input state $\boldsymbol{X}_t$

Select action A_i from the number of possible actions for each rule i according to ϵ-greedy policy:

$A_i = argmax_j\ q(i,j)$ with probability $1-\epsilon$,

$A_i = random\ A_j, j = 1,...k$ with probability ϵ

Determine the global action $A(\boldsymbol{X}_t)$ for the state $\boldsymbol{X}_t$ using (3.9)

Estimate the corresponding Q value $Q(\boldsymbol{X}_t, A)$ using (3.10)

Take action $A(\boldsymbol{X}_t)$ and observe the new state $\boldsymbol{X}_{t+1}$

Get the reward r_t

Estimate the value of the new state $v(\boldsymbol{X}_{t+1})$ using (3.11)

Calculate the error signal ΔQ using (3.12)

Update q-values using (3.13)

$$A(\boldsymbol{X}_t) = \frac{\sum_{i=1}^{i=N} w_i(\boldsymbol{X}_t)A_i}{\sum_{i=1}^{i=N} w_i(\boldsymbol{X}_t)} \tag{3.9}$$

where $w_i(\boldsymbol{X}_t)$ is the firing strength of rule i which is determined by the membership functions of crisp input $\boldsymbol{X}_t$ using fuzzy *and* operation and A_i is the corresponding action/consequent of rule i from the exploration/exploitation policy.

$$Q(\boldsymbol{X}_t, A) = \frac{\sum_{i=1}^{i=N} w_i(\boldsymbol{X}_t)A_i q(i, A_i)}{\sum_{i=1}^{i=N} w_i(\boldsymbol{X}_t)} \tag{3.10}$$

where $q(i, A_i)$ is the q-value associated with rule i and its selected action A_i.

$$v(\boldsymbol{X}_{t+1}) = \frac{\sum_{i=1}^{i=N} w_i(\boldsymbol{X}_{t+1})A_i q(i, A_{max})}{\sum_{i=1}^{i=N} w_i(\boldsymbol{X}_{t+1})} \tag{3.11}$$

where $w_i(\boldsymbol{X}_{t+1})$ is the firing strength of rule i evaluated from the new state $\boldsymbol{X}_{t+1}$ and $q(i, A_{max})$ is the maximum q-value for rule i.

$$\Delta Q = r_t + \gamma v(\boldsymbol{X}_{t+1}) - Q(\boldsymbol{X}_t, A) \tag{3.12}$$

where γ is the discount factor.

$$\Delta q(i, A_i) = \alpha \Delta Q \frac{w_i(\boldsymbol{X}_t)}{\sum_{i=1}^{i=N} w_i(\boldsymbol{X}_t)} \tag{3.13}$$

where α is the learning rate.

3.3 Deep Reinforcement Learning

Tabular RL methods, e.g., QL, work through mapping each state to a value. Hence, each state-action pair need to be explored. As a result, these RL methods can be inefficient for large state-action spaces and cannot be directly applied in problems involving continuous state-action spaces. In such cases, fine grained discretization of the state-action space can be applied, e.g., FQL, but at a cost of an exponential increase in the state-action space. Learning via tabular RL methods may become infeasible due to the need to store large state-action table and a very slow learning process of exploring each state-action pairs. Hence, it is important to learn to generalize similar states to similar values as well as inferring the values of new states from the already explored ones. Recently, neural networks have been used for approximating the Q-values of state-action pairs [110]. The methods are known as DRL and combine deep neural networks with RL. In particular, Deep Q-Networks (DQN) use deep neural networks as value approximation functions [110, 111]. In what follows we briefly describe deep neural networks and the foundations of deep Q-network.

3.3.1 Deep Neural Network

Neural Networks are models of computation inspired by the neurons in human brains. A neural network can be represented as a direct graph whose nodes correspond to neurons and edges correspond to links between them. Hence, each neuron receives as input a weighted sum of the outputs of the neurons connected to its incoming edges. The recent advances in computing power, algorithms and big data drive the effectiveness of neural networks with multiple layers between input and output layers. This paradigm is known as deep learning and the neural network architecture is called deep neural network.

Deep learning relies on a function $f : x \rightarrow y$ parametrized with $\theta \in \mathbb{R}^{n_\theta}$: $y = f(x;\theta)$. A deep neural network is characterized by a succession of multiple processing layers. Each layer consists in a non-linear transformation and the sequence of these transformation leads to learning different levels of abstraction [111]. A simplified neural network with one fully connected hidden layer is shown in Fig. 3.3. The first layer is given the input features x in the form of a column

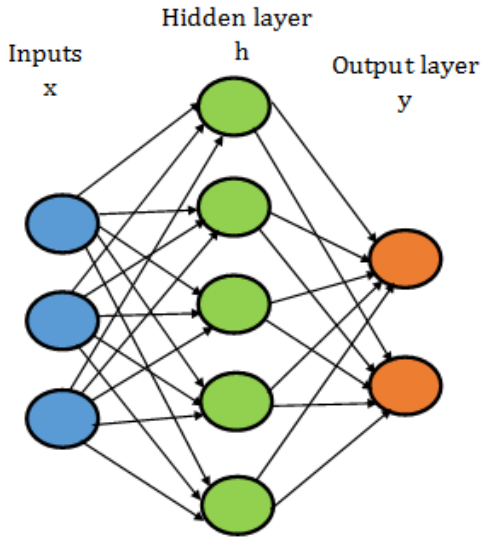


FIGURE 3.3: Simple neural network with one hidden layer

vector of size n_x. The values of the next hidden layer are a transformation of these values by a non-linear parametric function, which is essentially a matrix multiplication by W_1 of size $n_h \times n_x$ plus a bias term b_1 of size n_h followed by a non-linear transformation as shown in (3.14).

$$h = A(W_1 \cdot x + b_1) \tag{3.14}$$

Where A is the activation function. These activation functions are non-linear and consequently the transformation at each layer is non-linear. The hidden layer h can in turn be transformed to other sets of values up to the last transformation that provides the output values y. For the case of one layer neural network, the output y is given as (3.15):

$$y = (W_2 \cdot h + b_2) \tag{3.15}$$

Where W_2 is of size $n_y \times n_h$ and b_2 is of size n_y. The layers of neural network are trained to minimize the empirical error $I_S[f]$. The most widely used method for optimizing the parameters of a neural network is based on a gradient descent via the back propagation algorithm [112]. In this case, at every iteration, the algorithm changes its parameters set θ so as to fit the desired function given in (3.16).

$$\theta \leftarrow \theta - \alpha \nabla_\theta I_S[f] \tag{3.16}$$

Where α is the learning rate.

These type of neural networks are known as feed forward networks. Recently,

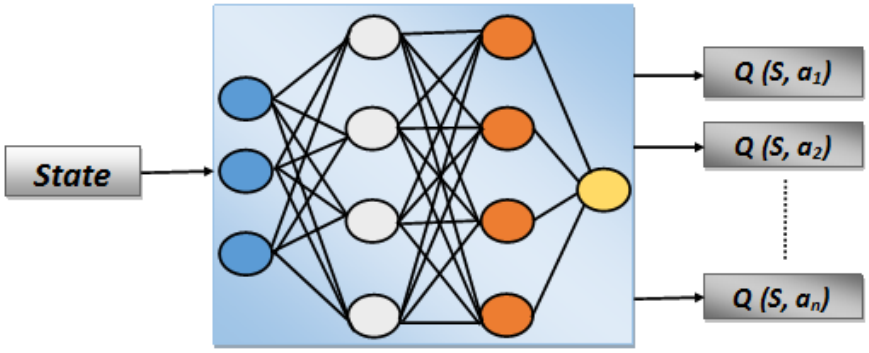


FIGURE 3.4: DQN Framework

many different types of neural network layers have appeared beyond the simple feed forward networks. Each variation provides specific advantages depending on the application at hand. The two most widely used types are convolutional layers which are well suited to images and recurrent layers which are particularity well suited for sequential data. In addition, there is a growing trend of using an increasing number of layers.

3.3.2 Deep Q-Network (DQN)

DQN uses neural networks as estimators of Q-values for state-actions pairs. Neural networks are widely known as universal function approximators and combining them with RL has recently been shown to be a promising paradigm with many successful applications. Earlier RL algorithms, e.g., QL, are limited in practice due to the tabular nature of Q-value updates which limits the scalability of the algorithms in case of problems involving continuous / large state-action pairs. DQN allows us to work with such complex problems through Q-value estimations without a need for a large and impractical look-up tables. The main idea behind DQN algorithm is shown in Fig. 3.4.

Algorithm 6 DQN Procedure

Initialize replay memory D of capacity M
Initialize action-value neural network Q with random weights $\boldsymbol{\theta}$
Initialize target action-value neural network $\hat{Q}$ with weights $\boldsymbol{\theta}^{-} = \boldsymbol{\theta}$
for each episode **do**:
  Initialize $\boldsymbol{S}_t$ - observation from the scenario
  for each step, t, of episode **do**:
    with probability ϵ select a random action a_t, otherwise $a_t = \pi(\boldsymbol{S}_t)$
    Take action a_t, get reward r_t and observe next state $\boldsymbol{S}_{t+1}$
    Store transition ($\boldsymbol{S}_t$, a_t, r_t, $\boldsymbol{S}_{t+1}$) in D
    Sample a random mini batch of K transitions ($\boldsymbol{S}_j$, a_j, r_j, $\boldsymbol{S}_{j+1}$) from D
    Set target value, $y_j = r_j + \gamma \times \max_{\hat{a}} \hat{Q}(\boldsymbol{S}_{j+1}, \hat{a}; \boldsymbol{\theta}^{-})$
    Perform a gradient descent step on $(y_j - Q(\boldsymbol{S}_j, a_j; \boldsymbol{\theta}))^2$ with respect to $\boldsymbol{\theta}$
    reset $\hat{Q} = Q$ every C steps
  end for
end for

As recommended in [110], there are some techniques usually applied to help the convergence of DQN algorithm. Among these, storing previous experiences of states, actions, reward and next state in memory buffer and randomly sampling mini-batch of this experience for training the neural network is proved to be the most effective way to ensure convergence by avoiding temporal correlation effects. This technique is known as experience replay. The training procedure of DQN with experience replay is shown in Algorithm 6.

3.4 Multi-agent Reinforcement Learning

Single Agent RL approach in systems involving many actors is prone to challenges mainly arising from scalability and long convergence phase. In systems with multiple actors, e.g. mobile networks with many BSs, a centralized decision making might experience long convergence and training phases due to high number of state and action spaces. Hence, a distributed approach is more feasible in such systems via sharing the problem among multiple agents and learning jointly towards a common goal. This approach is known as multi-agent RL (MRL). MRL can also harness new benefits from sharing experience, e.g. by communication, teaching or imitation. Experience sharing can help agents with similar goals learn faster and reach better performance. For instance, the agents can exchange information

using communication [113], skilled agents can act as teachers for others [114] or the learner may watch and imitate skilled agents [115]. Thanks to the decentralized nature of MRL set-up, speed-up can also be realized via parallel computation. MRL is also robust, i.e. when one or more agents fail in the system, the remaining agents can take over some of their tasks. In addition, in MRL design, insertion of new agents into the system is relatively easier leading to the a high degree of scalabiltiy.

However, defining a good MRL goal is a challenge due to the correlation among the agents' environment and each agent objective can not be maximized independently. Non-stationarity arises in MRL since all the agents in the system are learning simultaneously. This lead to a moving target dilemma, i.e. the best policy changes as the other agents' polices change [116]. The learning process in MRL is further complicated since exploration is required to obtain information not only about the environment, but also about the other agents in order to adapt their behavior. However, too much exploration can destabilize the other agents, thereby making the learning task more difficult for the exploring agent. Hence the agents decision depends also on the the decision taken by the other agents. As a result, the learning agents need to be coordinated. Coordination typically involves consistently breaking ties between equally good joint actions or strategies. Coordination is needed both in cooperative settings as well as in self-interested agents if the lack of coordination affects all the agents negatively.

In fully cooperative setting, as the one investigated in this work, the common return or reward can be jointly maximized. In this approach, the agents have the same reward function and the learning goal is to maximize the common discounted return. An example of MRL algorithm for cooperative setting is Distributed QL [117]. In this case, each agent maintains a local policy and local Q-function which depends only on its action. These local Q-values are updated only when the update leads to an increase in the Q-value, thus ensuring that the local Q-value always captures the maximum of the joint-action Q-values. However, the algorithm is limited only in deterministic problems with non-negative reward functions. An other approach involves correlated and different agents' returns which cannot be maximized independently. Specifying a good MRL goal is challenging since it should incorporate stability of the learning dynamics as well as adaptation to the changing behavior of other learning agents. Stability ensures convergence to a stationary policy. This requires the agents' strategies to eventually converge to a coordinated equilibrium like the Nash equilibria. However, the connection between Nash equilibra and the performance of stochastic dynamic games is unclear [118].

For ensuring adaption, Rationality [119] as well as the concept of no-regret [120] are defined as criteria of an agent behavior in the environment where policies of other agents change.

Empirical coordination techniques among the agents are also proposed in [121]. In these solutions, the convergence rate of RL algorithms are increased using a heuristic function for selecting actions in order to guide the exploration of the state-action space efficiently. Moreover, heuristically accelerated approach, which was proposed as a mechanism to improve the training phase of a single-agent RL, has been extended to MRL [122]. The external heuristic aids in coordinating the agents and can be defined to have a centralized view of the effect of the agents' action on the overall environment through adopting a hierarchical architecture. This hierarchical approach is known as layered learning and has been applied to energy management control in renewable powered mobile networks in [89].

In this thesis, the definition of a common system wide return as well as information exchange among agents as ways of mitigating the coordination issue are presented in detail in Chapters 6 and 7.

Chapter 4

Performance Bound Study

4.1 Introduction

This chapter focuses on determining the performance bounds of an optimal selection functional split options for networks of vSCs that are solely powered by EH. The functional splits give insights into considering more operation modes of BSs, in addition to switch on/off. DP, in particular, shortest path search is used to determine the optimal functional split options considering traffic requirements and available energy budget. In addition, this chapter considers dynamic functional split control with energy sharing. In this case, the vSCs are interconnected to the MBS to form a micro-grid that enables them to share excess energy to the MBS. Hence, the energy inflow is managed based on a *harvest-store-share* approach where a harvested energy is consumed by the vSC and any excess is stored in the battery for later use. Whenever, the battery reaches its maximum capacity, the excess energy is shared with the MBS to reduce grid energy consumption. The main goals of this chapter are:

- Formulating the energy management of vSCs and a MBS with co-located BBU pool as an offline optimization problem targeting three functional split options, namely MAC-PHY, UpperPHY-LowerPHY and PHY-RF;
- Applying a DP algorithm to find the optimal placement of functional split options considering the traffic demand, energy reserve and forecasted energy arrival. In particular shortest path search is applied for solving the optimization problem;

- Presenting the performance of dynamic placement of functional split options through numerical results with comparison against static configurations. Hence, these results can serve as performance bounds for online optimization approaches;
- Integration of distributed EH and storage systems in RANs and theoretical formulation of the joint load control and energy sharing optimization problem;
- Design of a joint optimal traffic and computational load control plus energy sharing method;
- Dimension of the EH and storage systems, characterization of the network performance with energy saving and cost analysis for different EH and storage design approaches.

In the following, we describe the network scenario in Section 4.2. The system model is explained in Section 4.3. The optimization problem statement and the optimal solutions are described in Section 4.4. The simulation scenario and results are explained in Section 4.5. Moreover, the modified problem statement to include energy sharing and numerical results are shown in Section 4.6. Finally, the conclusions of the chapter are presented in Section 4.7.

4.2 Network Scenario

We consider a CRAN-like architecture composed of a MBS with co-located BBU pool, acting as MEC server, and vSCs that are deployed in hot-spots for capacity enhancement. The MBS is relying on grid power whereas all the vSCs are powered by EH and batteries. The BB functions of the vSCs are executed as virtual network functions. The network scenario is described in detail in Section 1.2. These virtual network functions can be executed locally at the vSCs own resources or remotely using the BBU pool resources at the MBS. The decision on where to execute the BB functions depends on the available energy budget at the vSCs. The architecture considered is depicted in Fig. 1.2. The objective of this study is to determine the performance bounds of dynamic functional split selection policy. The functional split options that can be applied for the vSCs are given in [1]. Considering the potential centralization gains and effects on energy, we have selected the following functional split options as targets:

- PHY-RF – all the BB processing is done centrally at the BBU pool at MBS site;
- UpperPHY-LowerPHY – the LowerPHY layer processing is done by the vSC whereas UpperPHY and above are executed by the BBU pool at MBS site;
- MAC-PHY – the whole PHY layer processing takes place at vSCs whereas MAC and above layers are done at the central BBU pool.

4.3 System Model

A vector $\boldsymbol{A}^t \triangleq [A_1^t, A_2^t, \ldots, A_N^t]$ denotes the modes of operation of the N vSCs at time slot t where A_n^t is the mode of n^{th} vSC. Each single element A_i^t is defined as:

$$A_i^t = \begin{cases} 0, & \text{if } i^{\text{th}} \text{ vSC is switched off} \\ 1, & \text{if } i^{\text{th}} \text{ vSC is in PHY-RF split mode} \\ 2, & \text{if } i^{\text{th}} \text{ vSC is in UpperPHY-LowerPHY mode} \\ 3, & \text{if } i^{\text{th}} \text{ vSC is in MAC-PHY mode} \end{cases} \tag{4.1}$$

The energy harvested by each vSC at time slot t is denoted by $\boldsymbol{H}^t \triangleq [H_1^t, H_2^t, \ldots, H_N^t]$ whereas the energy stored by each vSC at time slot t is denoted by $\boldsymbol{B}^t \triangleq [B_1^t, B_2^t, \ldots, B_N^t]$, where H_n^t and B_n^t are the energy harvested and stored by the n^{th} vSC respectively. The traffic load experienced by each vSC is denoted by the vector $\boldsymbol{L}^t \triangleq [L_1^t, L_2^t, \ldots, L_N^t]$ where L_n^t denotes the traffic load of n^{th} vSC. The power consumption model for estimating the energy requirements of vSCs and MBS is described in Section 2.6.

4.3.1 EH and Demand Profiles

Hourly EH traces from a solar source are generated using the SolarStat tool [92]. The tool is based on a Markov model that provides accurate statistics per month basis by processing hourly energy arrival data of 20 years. EH traces are generally bell-shaped with a peak around midday, whereas the EH during the night is negligible. Moreover, as discussed in [92], high variability of the harvested energy may occur during the day, even for the summer months. As a result, although the energy inflow pattern can be known to a certain extent, intelligent and adaptive

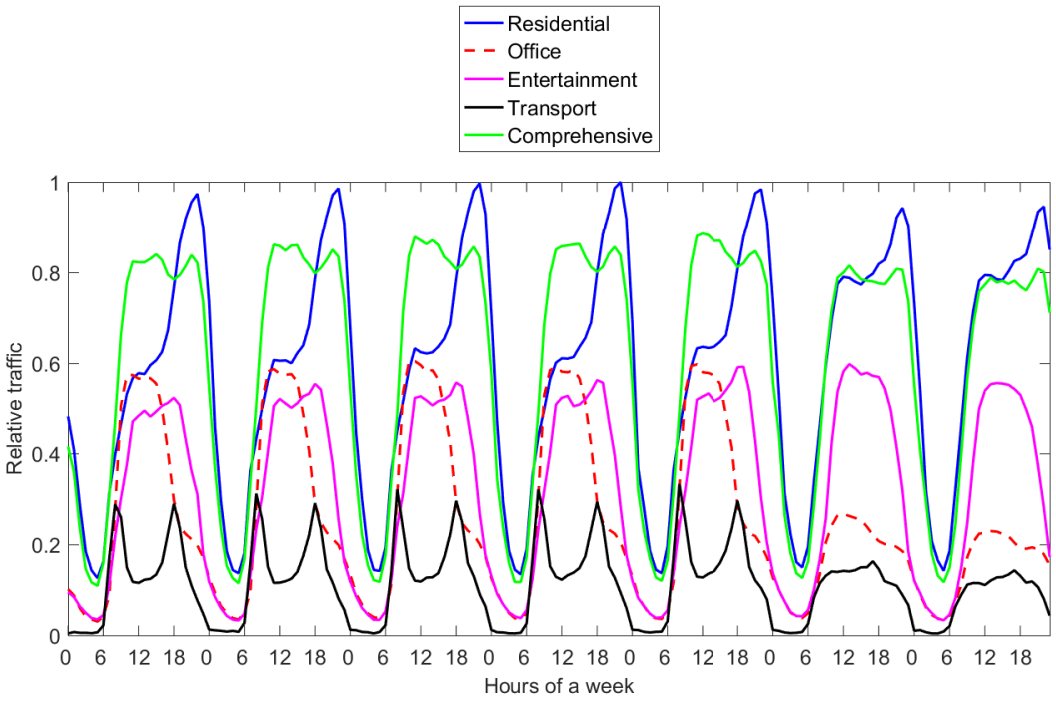


FIGURE 4.1: Typical weekly traffic profiles in different functional regions [9]

algorithms that make their decisions based on current and past inflow patterns, as well as predictions of future energy arrivals, have to be designed.

For the traffic demand profile, the UEs have been classified as in [106] in *heavy* and *ordinary* users according to their amount of requested traffic. Moreover, we use the traffic load profile obtained in [9] as the average amount generated by the users. In addition, based on the average traffic generated by the users, traffic variability is added following a normal distribution using standard deviation from measurements of real mobile traffic traces [123]. The traffic demand of each UEs in a cycle are dimensioned based on traffic profiles presented in [9], which are derived from time, location and frequency information of thousands of cellular towers. The analysis in [9] demonstrates that the urban mobile traffic usage can be described by mainly five basic time domain patterns that corresponds to different functional regions, i.e., residential, office, transportation, entertainment and comprehensive areas. Moreover, these traffic profiles are also characterized by weekdays and weekend variation. The relative magnitudes of the traffic in these regions as well as both weekday and weekend variations are depicted in Fig. 4.1, as described in [9].

In particular, we are considering residential and office profiles which are the most common use cases for urban deployment scenarios. An example of a normalized

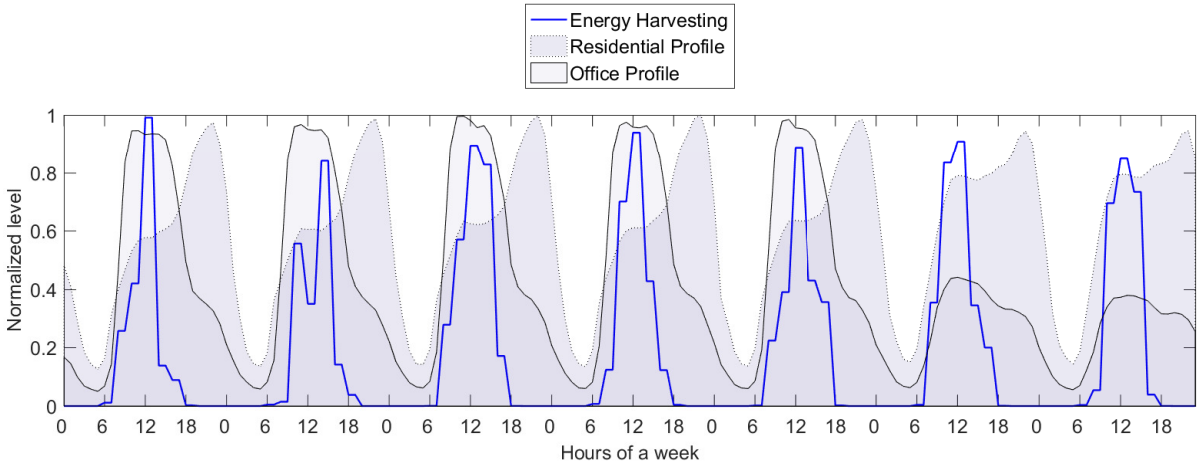


FIGURE 4.2: Typical weekly normalized energy harvesting, office traffic profile and residential traffic profile

EH trace, the residential traffic profile and the office traffic profile for both week and weekend days is shown in Fig. 4.2. The figure shows that EH and residential traffic profile peaks occur at different hours of the day i.e. EH peak occurs around noon where as traffic demand peak occurs during the evening emphasizing the need for an intelligent energy management policy to maximize the utilization of harvested energy.

4.4 Optimal Solution

4.4.1 Problem Statement

An intelligent energy management decision is a sequential process that selects the optimal configurations of the N vSCs based on the traffic demand, the energy reserve and energy arrivals. The objective is to minimize both the grid energy consumption and the amount of dropped traffic due to the vSCs in the OFF mode. Since the vSCs are solely powered by EH, the grid energy consumption of the system is equivalent to the energy consumption of the MBS. The decision process evolves in cycles along with the traffic demand and the energy arrival variations. At each cycle t, the task of the centralized controller is to select the optimal mode of each vSC among the four options given by (4.1). Hence the goal is to minimize the total weighted cost of both grid energy consumption at MBS and the traffic demands that cannot be satisfied due to vSCs in the OFF mode.

For a finite time horizon, K, it is modeled as a DP problem:

$$\begin{aligned} &\min_{\{\boldsymbol{A}^t\}_{t=1,\dots,K}} \sum_{t=1}^{K} f(\boldsymbol{A}^t) \\ &\text{subject to } B^t_{(i)} > B_{\text{th}} \ \forall i. \end{aligned} \tag{4.2}$$

where B_{th} is the battery threshold level adopted to prevent damages to the storage devices and K is the time horizon and $f(\boldsymbol{A}^t)$ is the cost function at time step t, which is defined as:

$$f(\boldsymbol{A}^t) = \omega_1 \cdot \mathrm{P_m}(\boldsymbol{A}^t) + \omega_2 \cdot \mathrm{D}(\boldsymbol{A}^t) \tag{4.3}$$

where $\mathrm{P_m}(\boldsymbol{A}^t)$ and $\mathrm{D}(\boldsymbol{A}^t)$ are respectively the grid power consumption and the traffic drop rate of the system, given the modes of operation of the vSCs. The grid energy consumption, $\mathrm{P_m}(\boldsymbol{A}^t)$, is equivalent to the power consumption by MBS and is determined based on (2.1), as shown in Section 2.6. The traffic drop rate, $\mathrm{D}(\boldsymbol{A}^t)$, is the ratio of the total amount of traffic demand that cannot be served by the system in the time step t. Additionally, each battery at the vSCs has to be maintained in the proper State Of Charge (SOC) (i.e, above the battery level threshold B_{th}) to avoid a rapid reduction of its lifetime [124]. Finally, the weights ω_1 and ω_2 determine the balance between two objectives and $\omega_1 \geq 0$, $\omega_2 \geq 0$, $\omega_1 + \omega_2 = 1$. In what follows, we describe the optimal solution for the optimization problem shown in (4.2). The solution rely on a-priori knowledge of the sequential decision making process, i.e., energy arrival and traffic request information are known in advance.

4.4.2 Graphical Representation

The problem of finding the optimal modes of operation at each time slot t is represented as a graph. In the graph, a single node (N^i_t) represents a possible combination of different modes of the vSCs. These combinations result in different grid power consumption, system drop rate and energy storage levels of vSCs. In Fig. 4.3, an example of graph for a system with a single vSC is depicted. At first time step ($t = 1$), the vSC can be in one of the four possible modes: switch off, PHY-RF split mode, UpperPHY-LowerPHY split mode and MAC-PHY split mode. Moving one time step ahead, EH and traffic demands are also evolving. Hence, each node (N^i_t) generates four possible child nodes corresponding to the

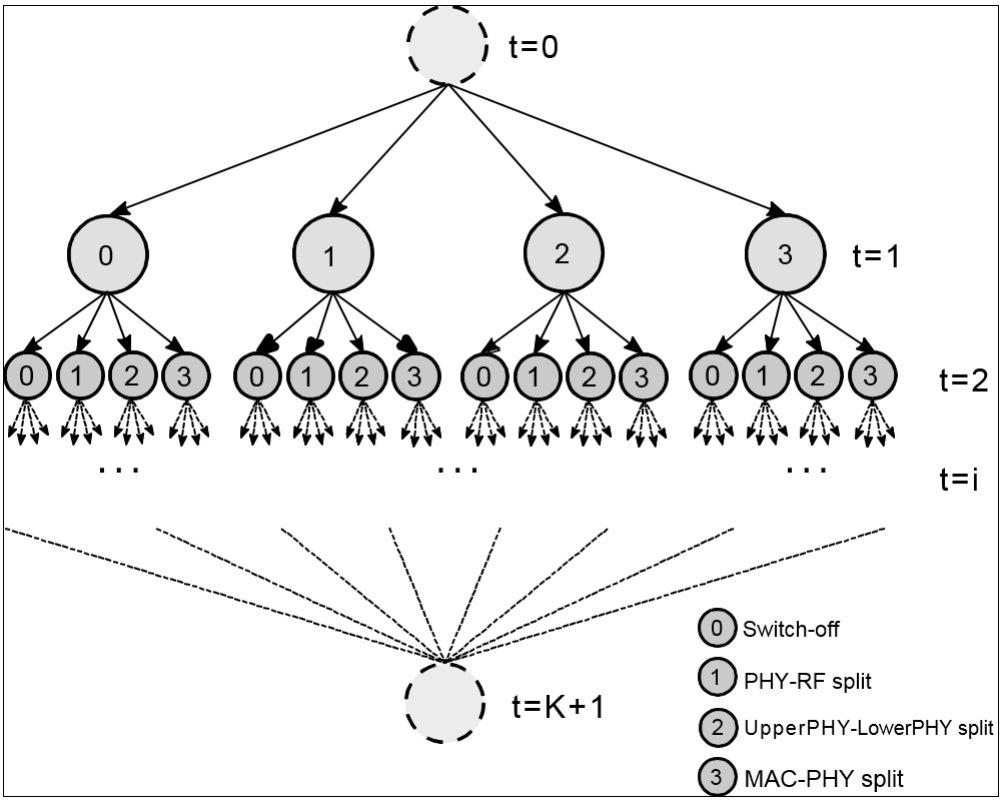


FIGURE 4.3: Graphical representation of sequential functional split options a vSC

four possible operating modes at cycle $t+1$, $(N_{t+1}^{j}), j = 1, ..., 4$. The number of such possible combinations keeps on evolving until reaching the time horizon K, leading to the maximum number of possible paths at time instant K.

At cycle $t+1$, the battery level corresponding to the child nodes is calculated based on:

$$\boldsymbol{B}^{t+1} = \min\left(\boldsymbol{B}^{t} + \boldsymbol{H}^{t} - \mathbf{P}_{\text{vSC}}(\mathbf{A}^{t})\Delta_{t}, B_{\text{cap}}\right) \tag{4.4}$$

where B_{cap} is the maximum capacity of the battery in kWh, Δ_t is the duration of a slot, $\mathbf{P}_{\text{vSC}}(\mathbf{A}^t)$ is a vector representing the power consumption of the vSCs which depends on their modes of operation as described in Section 2.6. The cost function (4.3) is used to compute the cost associated to each arc connecting two nodes. Two artificial nodes have been added at time step $t = 0$ and $t = K+1$, to have a single initial node and a single terminal node. The cost associated to the arcs connecting the artificial nodes are set to zero. The cost associated to each arc of the graph can be interpreted as its length. Hence, the optimization problem in (4.2) is equivalent to finding the shortest path from the initial node at time $t = 0$ to the terminal node at time $t = K+1$.

4.4.3 Shortest Path Search

We consider the Label Correcting Algorithm [107] for finding the shortest path. The exploration of the graph is done in a depth-first approach by sequentially discovering shorter paths from the starting node to the intermediate nodes until reaching the destination. We define the variable d_i, called *label of* i, as the length of the shortest path to the node i, OPEN as the list of nodes to be explored and UPPER as the last found minimum-length path. Initially both UPPER and d_i are set to ∞. Throughout the exploration process, the length of the shorter path found so far is maintained in d_i. If a new path is found with shorter length to i, the algorithm considers whether the labels d_j of the child nodes j can be corrected by setting d_j to $d_i + a_{ij}$, where a_{ij} is the $arc(i,j)$. The nodes that are candidates to be included in the shortest path are maintained in the list OPEN. Nodes that result in a path length longer than UPPER or those that cannot satisfy the battery and system constraints are excluded from this candidate list. The steps of the algorithm are shown in Algorithm 7. This exploration policy is relatively faster and requires lower memory by avoiding to explore the whole graph [107]. This is especially advantageous for a tree-like problem such as the one tackled in this work.

Algorithm 7 Shortest Path Search Algorithm

initialize OPEN with possible states at time t
while OPEN is not empty **do**
  remove a node N_t^i
  compute $\boldsymbol{B}^{t+1,j}, j = 1, .., 4^N$, for all $\boldsymbol{A}^{t+1}$ using (4.4)
  for each node N_{t+1}^j child of N_t^i **do**
    $a_{ij} = f(\boldsymbol{A}_j^{t+1})$
    if $d_i + a_{ij} < \min\{d_j, \text{UPPER}\}$
                and $\boldsymbol{B}_j^{t+1} > B_{\text{th}}$ **then**
      $d_j \leftarrow d_i + a_{ij}$
      set N_t^i parent of N_{t+1}^j
      if $t \neq K$ **then**
        place N_{t+1}^j in OPEN (if not already)
      else
        UPPER $= d_i + a_{ij}$
      end if
    end if
  end for
end while

4.5 Numerical Results

4.5.1 Simulation Scenario

According to the traffic model defined in Section 4.3.1, user activities are categorized based on [125] as heavy users with an activity of 900 MB/hr and ordinary users with an activity of 112.5 MB/hr. Moreover, 90 UEs per vSC are considered and out of these, the number of active UEs vary according to the traffic profiles shown in Section 4.3.1 depending on the vSC deployment area, i.e, residential or office. The solar energy traces are generated using the SolarStat tool [92] for the city of Los Angeles. For the PV modules, we have considered the commercial Panasonic N235B. These panels have single cell efficiencies as high as 21.1%, which ranks them amongst the most efficient solar modules delivering about 186 W/m^2. The solar panel size and battery capacity are dimensioned based on the criteria that the vSC can be fully recharged on a typical winter day. The simulation parameters and reference power consumption values are given in Table 4.1. The BB static power consumption figures are composed of P_{CPU}, P_{OFDM} and P_{filter} and the load dependent components are P_{FD} and P_{FEC}.

TABLE 4.1: Simulation Parameters.

Parameter	Value
Transmission power of macro cell (dBm)	43
Transmission power of vSC (dBm)	38
UEs per vSC	90
Heavy users ratio	0.5
Solar panel size (m^2)	4.48
Battery capacity (kWh)	2
B_{th}	20%
$P_{RF_{vSC}}$	2.6 W
$P_{PA_{vSC}}$	71.4 W
$P_{RF_{MBS}}$	9.18 W
$P_{PA_{MBS}}$	1100 W
GOPS to W conversion factor	8
$P_{BB_{static_{vSC}}}$	440 GOPS
$P_{BB_{load-dependent_{vSC}}}$	60 GOPS
$P_{BB_{static_{MBS}}}$	630 GOPS
$P_{BB_{load-dependent_{MBS}}}$	215 GOPS
$P_{overhead_{vSC}}$	0.0%
$P_{overhead_{MBS}}$	10.0%

Simulations of 3 vSCs in residential and office area profiles are considered. Moreover, A time horizon of 21 hours (i.e., $K = 21$) is selected, as it represents a

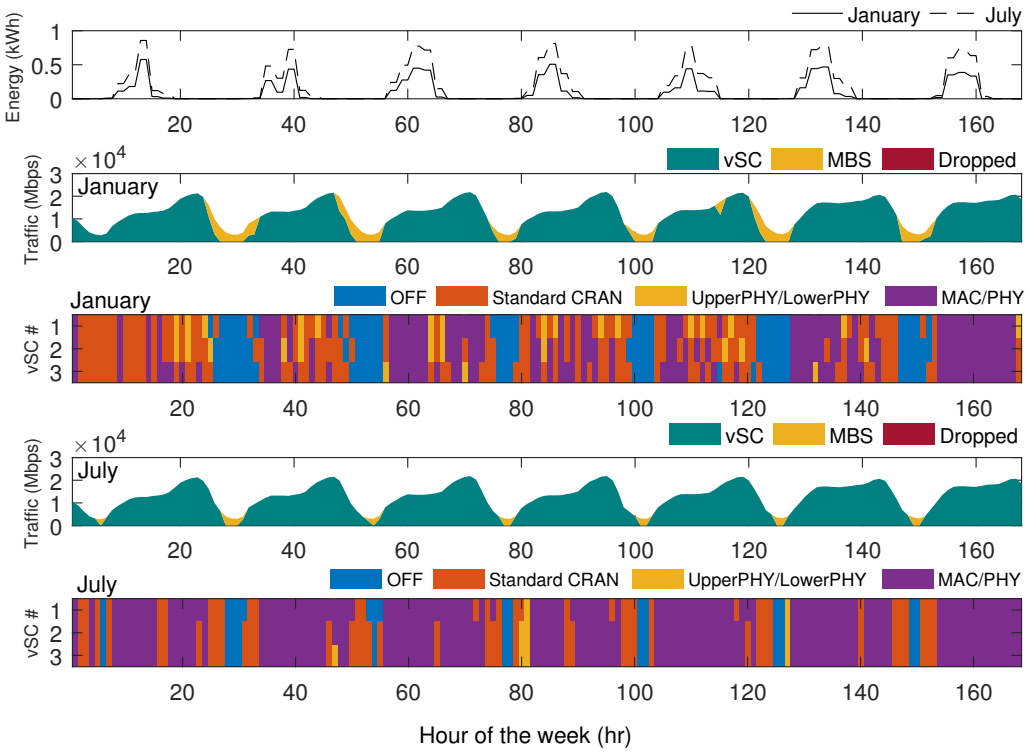


FIGURE 4.4: Optimal functional splits placement results in a residential area scenario for a week of January and July (hour 0 to hour 168; Monday from 0 - 23 hr). The traces show the amount of harvested energy, the amount of mobile traffic handled by vSCs and MBS and operative mode of each vSCs for both January and July weeks. The standard CRAN split is equivalent to PHY-RF split.

reasonable balance between algorithm performance and complexity [81]. In addition, equal weights of energy and system drop rate are used in the cost function (4.3), i.e., $\omega_1 = \omega_2 = 0.5$ to impose equal importance on the two objectives.

4.5.2 Optimal Functional Split Configurations

The result of optimal functional splits placement for a scenario involving 3 vSCs with 90 users per vSC in a residential area is shown in Fig. 4.4. Heavy users ratio of 50% is considered. The policy is able to decide the placement of the baseband functions in accordance with the available energy, forecasted harvested energy and traffic demands. Hence the result shows that, most of the user traffic is handled by the vSCs without dropped traffic. Moreover, for a January week, the PHY-RF and MAC-PHY are the most chosen split options during daytime and peak traffic periods where as switching off occurs during very low traffic hours. The average PHY-RF and MAC-PHY selection rate is 37% each and switching off rate

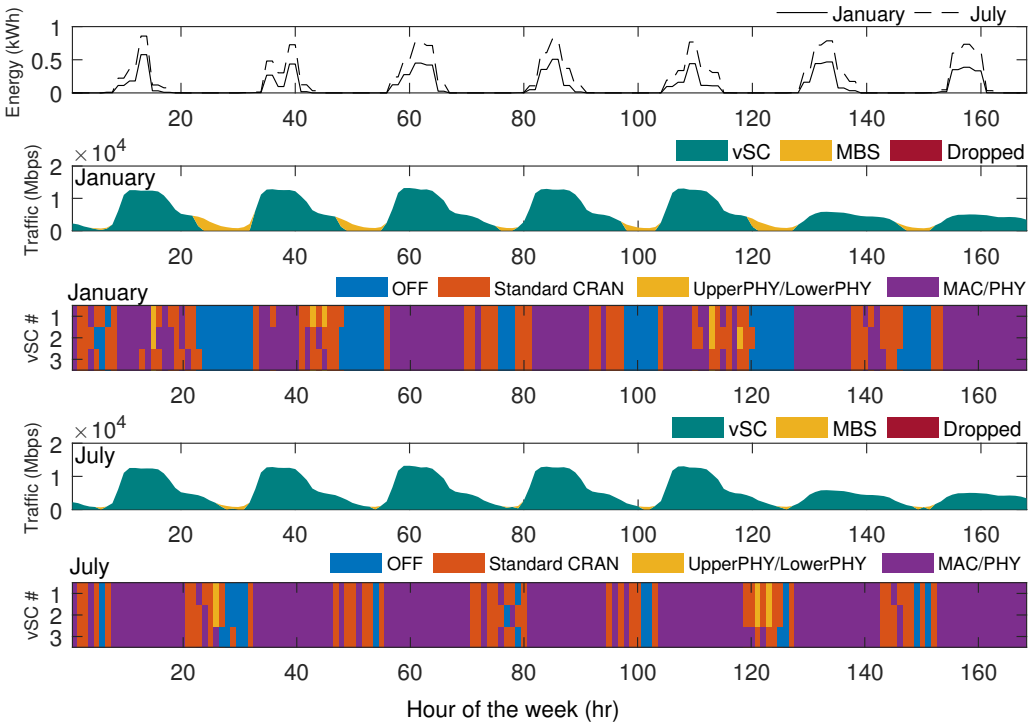


FIGURE 4.5: Optimal functional splits placement results in an office area scenario for a week of January and July (hour 0 to hour 168; Monday from 0 - 23 hr). The traces show the amount of harvested energy, the amount of mobile traffic handled by vSCs and MBS and operative mode of each vSCs for both January and July weeks. Standard CRAN split is equivalent to the PHY-RF split.

is averaged at 20%. The results of the simulation for a week of July shows that MAC-PHY split mode is the most selected option. Average MAC-PHY selection rate is 67%, where as switching off rate is averaged at 8%. This confirms that due to high energy availability, the vSCs are performing most of the baseband processes by themselves which, in turn, further reduces the grid energy consumed by the MBS.

The result of optimal functional split placement for a scenario of 3 vSCs deployed in an office area with 90 users per vSC is shown in Fig. 4.5. Heavy users ratio of 50% is considered. An office area traffic profile is characterized by relatively lower traffic peak both in weekdays and weekends. In addition, the peak traffic hours are different than the residential traffic profile and more aligned with the solar energy arrivals. The result shows that the dynamic placement of the baseband functions enable the vSCs to offload the MBS for most of the users traffic without any drop. In addition, MAC-PHY and PHY-RF operative modes are the most selected options in January during peak traffic periods. The average MAC-PHY

and PHY-RF selection rate is 47% and 28% respectively. Average switch off rate is at 23% and occurs during very low traffic periods, i.e., during night. In July, MAC-PHY is the most selected operative mode with an average selection rate of 68% and occurring during peak traffic periods, whereas, switch off occurs during very low traffic hours with average rate of 7.3%. This is due to the higher energy income in July. The high selection rate of MAC-PHY split results in a further reduction of grid energy consumption, since most of the baseband processes are performed locally by vSCs.

4.5.3 Comparison with static policies

This subsection provides a comparative analysis of the proposed optimal solutions with static configurations. In the static configurations, a vSC is kept in the same mode/functional split as long as the battery level is above the threshold, otherwise it is switched off. The results of these static policies for a scenario with 3 vSCs deployed in a residential area with 90 users each and a heavy user ratio of 50% for a week of January and July are shown in Table 4.2.

All the static policies are not capable to reach the traffic drop rate performance of the optimal bound. Moreover, the performance gap can vary importantly with respect to the policy. The PHY-RF policy shows smaller outage against both UpperPHY-LowerPHY and MAC-PHY policies. This is mainly due to the low energy consumption of the vSCs in PHY-RF mode since they do not perform any BB operation. Both UpperPHY-LowerPHY and MAC-PHY polices experience high drop rates and grid energy consumption for the case of January. This arises from the inability of the static polices to maintain the operation during the peak traffic periods that occur in the evening hours when the EH income is almost zero, which in turn overloads the MBS with the UEs from vSCs in the off mode. The static policies in general are sub-optimal in terms of grid energy consumption, where savings of up to 24 KWh and 29 KWh can be achieved with the optimal policy respectively for a week of January and July.

4.6 Joint Load Control and Energy Sharing

In this section, we focus on the study of an optimal dynamic control of functional split options with energy sharing. In particular, we investigate the case where vSCs

TABLE 4.2: Policy Comparisons

Policy	Grid energy consumption (KWh)		Average drop rate (%)	
	January	July	January	July
Optimal	149.51	133.23	0	0
PHY-RF	170.01	162.48	2.35	1.5
UpperPHY-LowerPHY	173.76	153.64	16.43	5
MAC-PHY	173.56	151.40	17.10	5

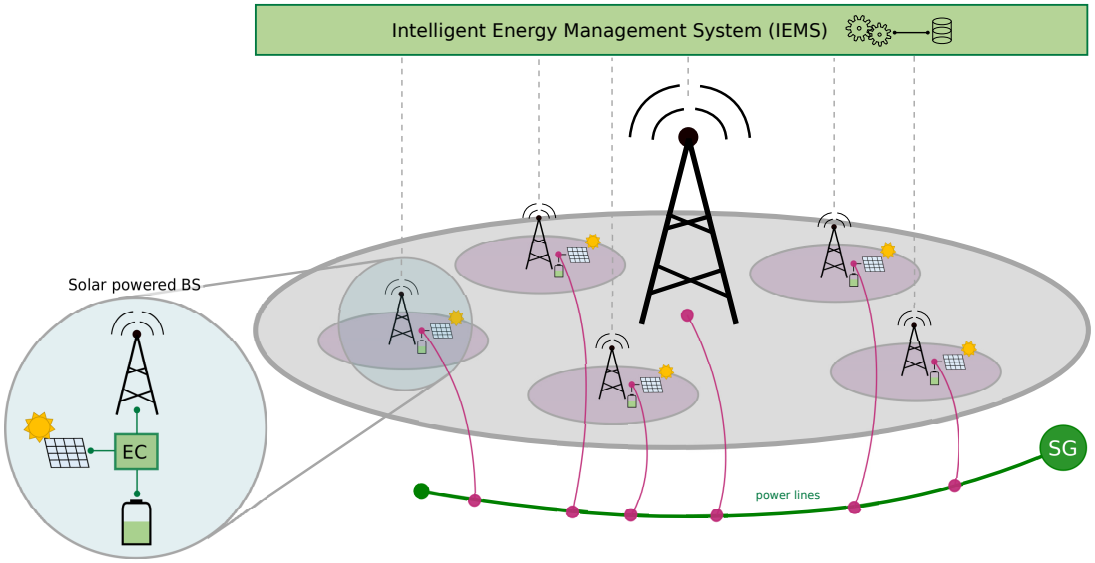


FIGURE 4.6: Diagram illustrating the reference framework, including the RAN with multiple tiers, the intelligent energy management system, the power line connections. In left-bottom side, a simplified scheme of the solar-powered BS is shown.

can share excess energy with the MBS when their batteries reach the maximum capacity.

4.6.1 Control Architecture and Problem Statement

We consider a two-tier scenario where the connection between the MBS and vSCs can be enabled via power lines and low resistance losses (i.e., the amount of energy lost in the conductor in form of heat) are guaranteed by the short distances between the vSCs and the MBS. In this scenario, a central control entity named Intelligent Energy Management System (IEMS) is located at the MBS and it is in charge of managing the operative states of the vSCs with the goal of achieving efficient utilization of the harvested energy. In detail, the IEMS opportunistically decides where to execute the BB functions of the vSCs. The operational states of the vCSs depend on the dynamics of the traffic demands and the EH arrivals. Therefore, the IEMS is in charge of predicting the evolution of those two processes to prevent

vSCs blackout during periods with low renewable energy arrivals and high traffic demands. The Energy Controller (EC) is an entity located at the vSC site that implements the decision taken by the IEMS. The full control architecture is shown in Fig. 4.6.

We modify the offline optimization problem introduced in Section 4.4.1 to include the possibility that vSCs can share excess energy, i.e., when the energy storage is full, to the MBS. The system model is described in Section 4.3 and the problem statement with the cost function is given in Section 4.4.1. The amount of harvested energy that exceeds the battery capacity is denoted by $\boldsymbol{X}^t \triangleq [X_1^t, X_2^t, \ldots, X_N^t]$ and it is calculated as:

$$\boldsymbol{X}^{t+1} = \max\left(\boldsymbol{B}^t + \boldsymbol{H}^t - \boldsymbol{P}^t\Delta_t - B_{\text{cap}}, 0\right) \tag{4.5}$$

The operative state of the vSCs at time t is defined by $\boldsymbol{A}^t = [A_1^t, A_2^t, \ldots, A_N^t]$. At each cycle, the intelligent energy management system decides the optimal configurations of each vSC to serve the traffic demand in that area. The general optimization problem formulation is shown in (4.2) and the cost function formulated in (4.3) is modified to include energy sharing as follows:

$$f(\boldsymbol{A}^t) = \omega_1 \cdot \mathrm{E_m}(\boldsymbol{A}^t) + \omega_2 \cdot \mathrm{D}(\boldsymbol{A}^t) \tag{4.6}$$

where $\mathrm{E_m}(\boldsymbol{A}^t)$ and $\mathrm{D}(\boldsymbol{A}^t)$ are respectively the normalized grid energy consumption and the traffic drop rate in the cluster, given the operative modes of the vSCs and the time step t. The optimization objective is to minimize the energy drained from the grid and reduce traffic drop rate. The grid energy consumption at time t is equivalent to the difference between the MBS energy consumption and the excess energy shared by the vSCs. The normalized grid energy consumption is then computed as:

$$\mathrm{E_m}(\boldsymbol{A}^t) = \max\left(\frac{\mathrm{P_m}(\boldsymbol{A}^t)\Delta_t - \sum_{i=1}^{N} X_i^t}{\mathrm{P_m^{MAX}}\Delta_t}, 0\right) \tag{4.7}$$

where $\mathrm{P_m}(\boldsymbol{A}^t)$ is the grid power consumption of the MBSs given the operative modes of the vSCs and the time step t, whereas $\mathrm{P_m^{MAX}}$ is the power consumption of the MBS at full load. The reader can refer to Section 2.6 for the power consumption model description. Due to the very low effect of UpperPHY-LowerPHY split on the energy consumption as shown in the results of Section 4.5.2, we target the following two functional split configuration options based on [31]:

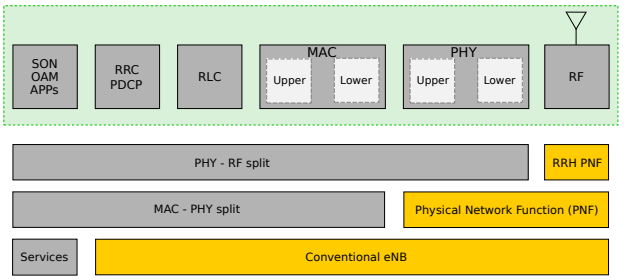


FIGURE 4.7: The different implementations of the functional split configuration options for a small BS, including Standard PHY-RF and MAC-PHY split. The conventional eNodeB (eNB) configuration is also shown for comparison.

- PHY-RF split: all the protocols, Physical (PHY) and above layers, are implemented at the MEC server located in the MBS site. Hence, the vSCs behaves as a Radio Frequency (RF) transceiver, used only for signal transmission and reception;
- MAC-PHY split: PHY layer processing takes place at the SBS, in addition to RF functions. Medium Access Control (MAC) and above layer functions are executed by the MEC server at the MBS site.

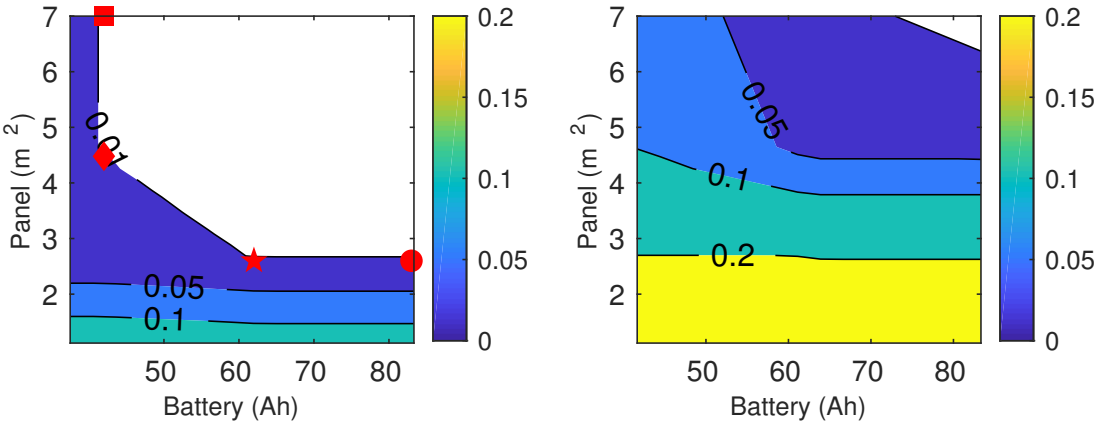
Fig. 4.7 shows the target functional split options compared to the conventional eNodeB architecture. Each operative mode corresponds to different computational load for the vSCs and MBSs, which in turn, corresponds to different energy consumption estimates, as described in Section 2.6.

It is worth highlighting here that in PHY-RF split, the vSCs are executing only the RF functionalities and the other upper layer functions, including PHY and MAC, are executed at the MBS site. This is similar to a CRAN architecture. Therefore, this architecture relies on a re-configurable fronthaul since the bandwidth and latency requirements become more stringent when more functions are placed in the centralized unit [1].

In this case, the state of the i^{th} vSC at time t is defined as:

$$A_i^t = \begin{cases} 0 & \text{if the } i\text{-th vSC is OFF} \\ 1 & \text{if the } i\text{-th vSC is in PHY-RF split mode} \\ 2 & \text{if the } i\text{-th vSC is in MAC-PHY split mode} \end{cases} \tag{4.8}$$

EH sources and demand profiles are given in Section 4.3.1.



(A) Optimal LC-ES (B) Naive

FIGURE 4.8: Contour plot of the traffic drop rate of LC-ES and the naive algorithm. Different colors indicate traffic drop rate regions, whose maximum outage is specified in the color map in the right hand side of the plot. The white filled region indicates a traffic drop rate smaller than 1%.

4.6.2 Optimal Load Control with Energy Sharing

In this section, we introduce the Load Control with Energy Sharing (LC-ES) algorithm, which provides the optimal functional split selection of vSCs at every time instant, given the temporal evolution of the EH arrival and traffic load processes.

We represent the optimization problem in (4.2) as a graph. A node i at time t in the graph (V_i^t) represents a possible combination of states of the vSCs in the cluster. Each combination returns a different level of the batteries of the vSCs. The procedure of the shortest path search algorithm for finding the optimal configurations are detailed in Section 4.4.3.

4.6.3 Numerical Results

We considered a simulation scenario as described in 4.5.1. Moreover, energy arrivals and aggregated downlink traffic have been generated according to the realistic models described in Section 4.3.1. In particular, the city of Los Angeles has been used for generating the solar harvested energy traces. The adopted power consumption model is described in Section 2.6. The results provided in what follows are averaged among ten different independent realizations of both energy arrivals and traffic processes.

Fig. 4.8 shows the contour plots of the traffic drop rate of the system, during the month of December (the worst in terms of harvested energy). Different colors

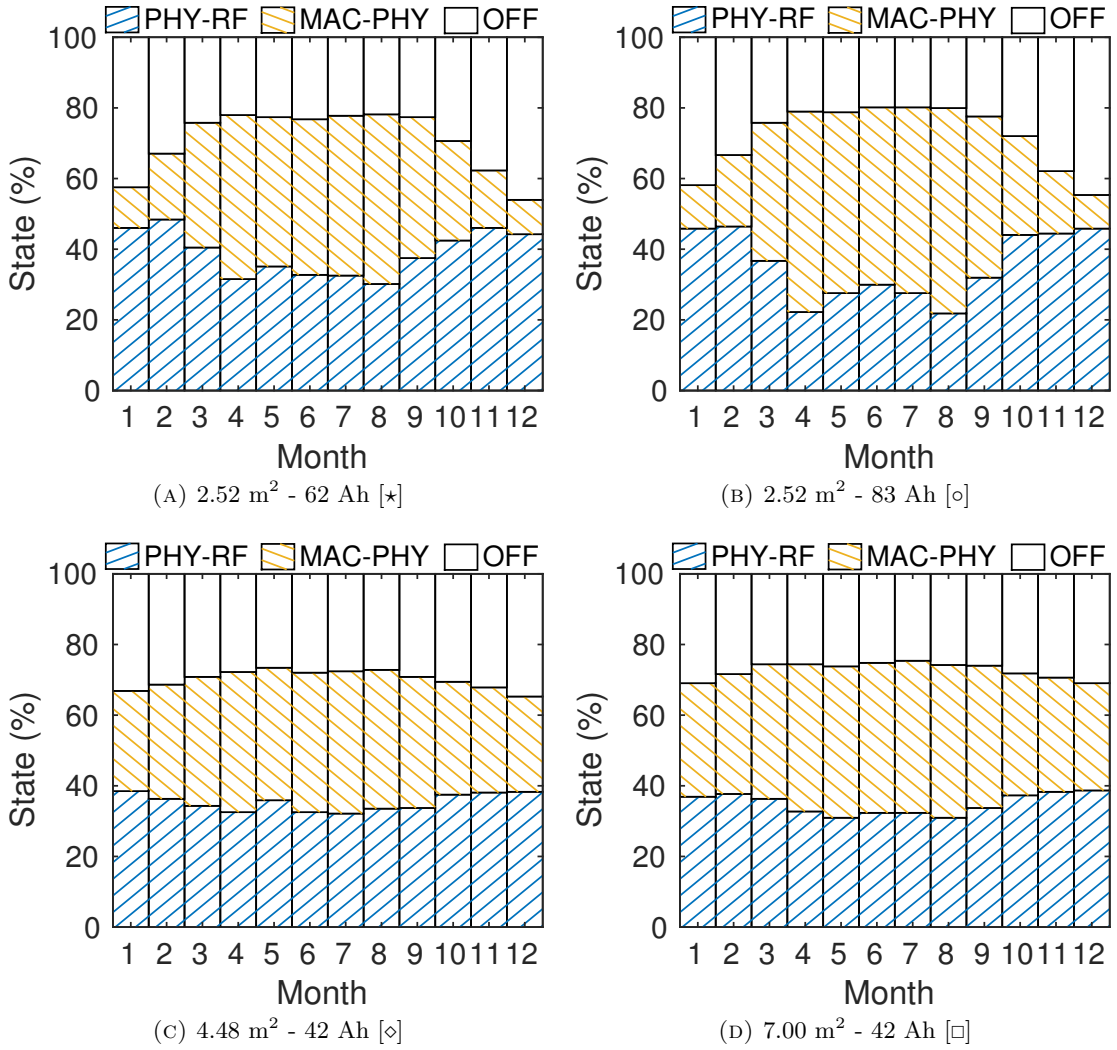
are used to indicate traffic drop rate regions (maximum values are specified in the associated color map). The white filled area indicates the parameter region where the traffic drop is smaller than 1%. Our optimal analysis is compared with naive approaches according to the network scenario. In naive approach, when the battery level is above the threshold the vSC is configured in the PHY-RF split mode, otherwise is in OFF state.

Taking 1% as our design parameter, all the points on the boundary of the white filled region are equally good. It is evident that optimal LC-ES allows the network to work with smaller sizes of the harvesting/storage system compared to the naive approaches. These results confirm that an intelligent energy management system is essential for an efficient use of the renewable energy resource and its installation in town facilities. The analysis in the following parts of the section considers various harvesting/storage design approaches corresponding to the different points laying in the boundary of the white filled area of Fig. 4.8 and labeled with star, circle, square and diamond.

4.6.3.1 vSCs Operative State Configuration

In Fig. 4.9, we report the different choices of the operative states of the vSCs by the optimal LC-ES across the different months of the year. The graphs refer to the selected harvesting/storage dimensions. It is shown that, vSCs offload the MBS for longer periods during the summer months, as expected. The length of the active periods and the choice of the operational mode depends on the specific harvesting/storage dimension. The vSC active period in the star deployment ranges between 54% (44% in PHY-RF mode and 10% in MAC-PHY, respectively, in December) and 78% (30% in PHY-RF mode and 48% in MAC-PHY, respectively, in August) of time; in the circle deployment between 55% (48% in PHY-RF and 9% in MAC-PHY, in December) and 80% (28% in PHY-RF, 52% in MAC-PHY, in August); in the diamond deployment between 65% (38% in PHY-RF, 27% in MAC-PHY, in December) and 73% (34% in PHY-RF, 39% in MAC-PHY, in August). Finally, the square deployment has an active period that ranges between 69% (39% in PHY-RF, 30% in MAC-PHY, in December) and 75% (32% in PHY-RF, 43% in MAC-PHY, in July) of the time.

PHY-RF split is the most chosen configuration option in the winter months, whereas MAC-PHY is the most prevalent in the summer months. A higher energy inflow allows the vSCs to locally perform their BB processing, whereas in winter,



(A) 2.52 m^2 - 62 Ah [⋆] (B) 2.52 m^2 - 83 Ah [○]

(C) 4.48 m^2 - 42 Ah [◇] (D) 7.00 m^2 - 42 Ah [□]

FIGURE 4.9: States selected by the LC-ES algorithm per month (in percentage) for different deployment sizes.

the lower energy arrivals force the vSC to offload the BB processing to the MBS and, hence, PHY-RF is the dominant operative mode. We observe that moving to the left side of the contour plot (i.e., bigger solar panels and smaller batteries) leads to longer activity during winter months and slightly shorter in summer.

4.6.3.2 Shared Energy Assessment

In Fig. 4.10 we show the energy shared and used by the MBS for every month of the year for different deployment sizes. The graphs are collected considering the selected harvesting/storage design approaches. For comparison purposes, we also indicate the amount of shared energy using the naive approaches.

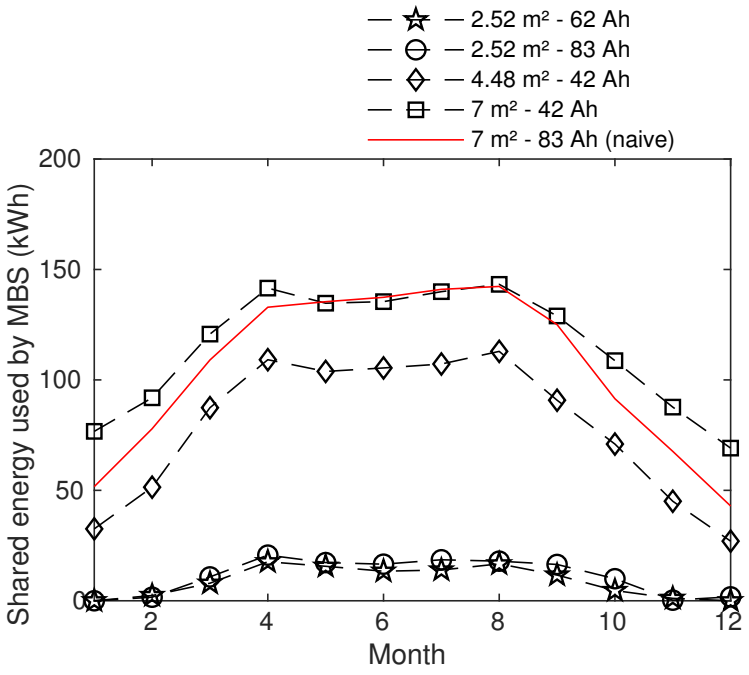


FIGURE 4.10: Energy shared by the vSCs and used by the MBS per month when considering different deployment sizes.

We observe a general trend of sharing a bigger amount of energy during the summer months, when a higher solar energy inflow occurs and, hence, a higher probability of exceeding the battery capacity of the vSCs.

4.6.4 Energy Savings and Cost Analysis

TABLE 4.3: Energy savings and costs for different deployment dimensions

Algorithm	Configuration P.(m^2)	B. (Ah)	Energy (kW) consumption [1yr]	Costs ($) CAPEX	OPEX [1yr]	cost [5yrs]	cost [10yrs]
Grid connected	-	-	10,264	0	2,155	10,775	21,550
LC-ES	⋆ 2.52	62	8,584 (-16%)	1,695	1,802	10,705 (-0.7%)	19,715 (-9%)
	○ 2.52	83	8,515 (-17%)	1,891	1,788	10,831 (-0.5%)	19,771 (-8%)
	◇ 4.48	42	7,335 (-29%)	2,359	1,540	10,059 (-7%)	17,759 (-18%)
	□ 7.00	42	5,373 (-48%)	3,464	1,128	9,104 (-16%)	14,744 (-32%)

In Table 4.3 we provide an energy and economic comparison between our LC-ES method and a scenario in which all the BSs are always active and supplied by the power grid. We refer to the latter as *grid-connected*. For the considered network scenario, we report the grid energy consumption, the CAPEX, the 1-year OPEX and the monetary cost (i.e., CAPEX + OPEX) due to the harvesting/storage add-on after 5 and 10 years, for the different panel and battery dimensions introduced in the previous section. The values between brackets indicate the savings with respect to the grid-connected scenario. We consider a cost of 1.17 \$/W for the solar panel (which also includes the installation cost) and 131 \$/kWh for the

battery. Moreover, the energy purchased from the grid has a cost of 0.21 $/kWh in our calculation.

The additional harvesting/storage hardware jointly with the LC-ES method allows reducing the grid power consumption for the different deployments considered. The energy savings range between 16% and 48% , depending on the harvesting/storage size. Carbon footprint and OPEX are decreased accordingly. In particular, monetary cost savings range between 0.5% and 16% after 5 years of operation. The results at 10 years show higher savings, ranging between 8% and 32%.

Table 4.3 provides useful insights on the energy and cost savings and permits the MNOs to design their harvesting/storage systems by considering the tradeoff between dimensions and economic cost. In general, the harvesting/storage system with the lowest CAPEX is not the most economically convenient option in the long run. In particular, the deployment size shown in the left side of Fig. 4.8 (i.e., square) is the most economically convenient option. Using these sizes, LC-ES is achieving higher savings by maintaining the vSCs operative for shorter periods in summer months and sharing more energy with the MBS compared to the other harvesting/storage configurations.

As a final remark, we can assume that higher revenues and savings can be achieved during the lifetime of the network in a near future considering that: i) equipment hardware is designed to be always more energy efficient, ii) the actual market trends show a decreasing cost of the solar panels and batteries, and increasing prices of the grid energy, iii) future RANs will be ultra-dense and longer offloading periods may occur due to the higher number of vSCs.

4.7 Conclusions

In this chapter, we have proposed an optimal functional split placement of EH vSCs that rely on central BBU pool for part of their BB processing. In particular, three functional split options namely, PHY-RF, UpperPHY-LowerPHY and MAC-PHY, have been targeted. A grid energy consumption minimization problem is stated and DP, more specifically shortest path search algorithm, is applied to determine the optimal functional split configurations. Simulation results show that dynamic functional split options placement with optimal control serves the traffic demand with significant energy saving, and hence lower OPEX, with respect to static functional split policies. Therefore, the obtained performance bounds

represent an encouraging starting point for the evaluation of more sophisticated online optimization techniques.

The chapter has also presented an optimal traffic and computational load control method with energy sharing to efficiently use the renewable energy coming from distributed sources and to facilitate the off-grid operation of the RAN. The proposed approach permits to move and execute some of the transmission functions of the vSCs at the MBS site. Energy exceeding the battery capacity is managed to be used by the MBS operations and further reduce the energy drained from the power grid. Software simulations demonstrate that an intelligent renewable energy management is essential to reduce the harvesting/storage dimensions with respect to naive approaches and leads to high energy and cost savings for a MNO.

Chapter 5

The Case of a Single vSC

5.1 Introduction

In Chapter 4, we have studied the performance bounds of dynamic selection of functional split options in an offline manner. The bounds are determined by solving a grid energy consumption minimization problem, based on a-priori knowledge of the system dynamics (traffic and energy arrivals) subject to battery constraints. This chapter focuses on proposing an online algorithm for the dynamic functional splits selection in vSCs with EH capabilities relying in part on central BBU pool co-located at MBS site.

The main contributions of this chapter are:

- Applying RL based on-line algorithms to find the optimal placement of functional split options for a scenario involving a single vSC and a MBS with co-located BBU pool. The approach considers the traffic demand, energy reserve and energy arrivals. In particular, TD learning approach is used and both QL and SARSA algorithms are applied;
- Presenting the performance of the RL based placement of functional split options through numerical results with comparison against offline performance bounds proposed in Chapter 4.

In what follows, we present the network model in Section 5.2. RL based energy management algorithm is described in Section 5.3. The results are analyzed in Section 5.4 and finally the conclusions of the chapter are written in Section 5.5.

5.2 Network Model

We consider a network scenario involving a vSC and MBS with BBU pool as described in Sections 1.2 and 4.2. Considering the potential centralization gains and following the results in Section 4.4, we have selected the following functional split options as targets in this work (shown in Fig. 5.1):

- PHY-RF – all the BB processing is done centrally at the BBU pool;
- MAC/PHY – the whole PHY layer processing takes place at vSCs, whereas MAC and above layers are done at the central BBU pool.

As a result, the vSC can be in one of the following three operative modes: PHY-RF/MAC-PHY/switch-off. At each time step, the RL based algorithm decides the optimal operative mode of the vSC by considering the traffic demand, energy arrival and energy reserve. The details of the RL algorithms applied are given in Chapter 3.

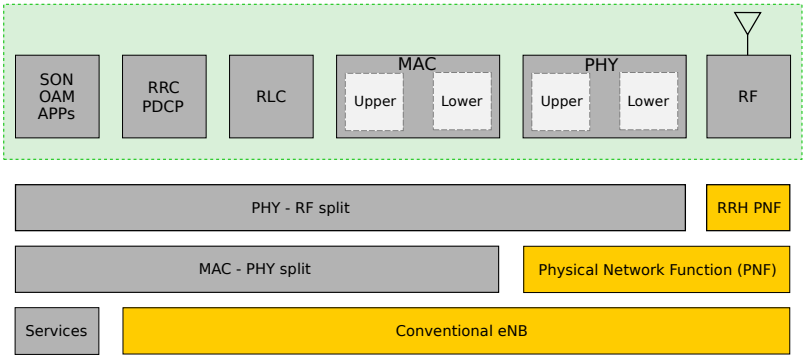


FIGURE 5.1: Functional split options considered in the scenario

The power model for estimating the power consumed by both MBS and vSCs in different functional split mode is described in Section 2.6. Moreover the energy arrival and demand profiles are explained in Section 4.3.1.

5.3 Algorithm

5.3.1 RL Based Energy Management

Formally, the RL framework is defined in terms of states, actions and rewards. For our network model, the objective of the RL based controller is to learn energy

management policies through interaction with the environment. The controller decides the operative mode of the vSC at each time step based on the traffic load, energy arrival and energy storage information. Let S_t be the state of the system at time t, the controller chooses an action A_t from action set $\boldsymbol{A}$, which is one of the operative modes of the vSC. As a result of this action, the environment returns an immediate reward r_t. Based on this r_t, the Q-value, $Q(S_t, A_t)$, which represents the value, i.e., how good it is to take a specific action in a given state, will be updated. This process of selecting a specific action and updating the Q-value continues sequentially for each time step. The controller selects the action at time step t based on the specific RL algorithm it applies. The goal of such algorithms is to estimate the Q-values for each state-action pair in order to learn optimal values in the long-term. In this work, we have applied algorithms that belong to the class called TD learning. Specifically both SARSA and QL RL procedures are applied for dynamic functional split placement policies. The TD class of RL methods are explained in Section 3.2.4. Moreover, the procedures of QL and SARSA based training are shown in Chapter 3, algorithms 3 and 4 respectively.

5.3.2 Algorithm Details

This section introduces the state, actions and reward functions that are defined for the dynamic functional split selection. The state, action and reward definitions are the same for both SARSA and QL based controllers. However, they differ on how they update Q-values as shown in Algorithm 3 and 4.

States: The state is dependent on the energy storage level, the normalized traffic load and the harvesting condition. Hence at time step, t, the state is given by (5.1).

$$S_t = \{H_t, B_t, \rho_t\} \tag{5.1}$$

Where H_t is the harvesting condition (e.g. daylight and night hours), B_t and ρ_t are the normalized battery status and traffic load which are quantized into 4 and 5 levels, respectively. This quantization levels are sufficient for capturing state variations in state representations.

Actions: The set of possible actions are the possible operative mode of the vSC. Hence, the action set are switch-off, PHY-RF split mode and MAC-PHY split mode.

Reward: The reward function determines the immediate reward the controller acquires as a result of taking a specific action. Since our goal is to maximize the harvested energy utilization (minimize the grid energy consumption by MBS), the reward definition should reflect this objective. The reward function is given as in (5.2).

$$r_t = \begin{cases} (1/(a+\rho_t)) - 1/(b \times B_t), & A_t \text{ is switch-off} \\ (c \times \rho_t) - (1/B_t), & A_t \text{ is PHY-RF} \\ (d \times \rho_t) - (1/B_t), & A_t \text{ is MAC-PHY} \end{cases} \tag{5.2}$$

Where ρ_t and B_t are the normalized traffic load and battery level at time step t respectively. The rationale behind the reward function is:

- It is desirable to switch-off during very low traffic periods and to operate in one of the split modes (PHY-RF or MAC-PHY) otherwise. Hence the immediate reward has inverse relationship with the traffic load for switching-off action whereas it is directly proportional to the traffic load for both PHY-RF and MAC-PHY modes;
- For all actions, there is an immediate penalty which is inversely proportional to the battery level. This helps to avoid the level of battery from falling into very low levels;
- The constants a, b, c and d are used to emphasize the reward according to the desired behavior. For example, choosing higher d than c implies higher immediate reward for MAC-PHY action than for PHY-RF action.

5.4 Numerical Results

The simulation scenario considered is the same as shown in Section 4.5.1. In what follows, we explain the training phase as well as the policy characteristics with comparison against the offline bound shown in Section 4.4.

5.4.1 Training phase

The training of the TD learning algorithms is performed on a residential, office and transport area traffic profiles. The traffic load at vSC is generated by 90 UEs with 50% heavy users ratio. After simulations involving different values of the training parameters, the values given in Table 7.1 are selected as the best combinations.

Similarly, for the reward function given in (5.2), the values of the constants a, b, c and d are chosen to be 1.15, 2, 10 and 20 respectively, as a result of a simulative evaluation. The training is done starting from January, by treating a month as one episode of training. The policy used for action selection during the training phase is an ε-greedy policy that can explore random action with probability of ε. The value of ε is multiplied by an ε-discount factor after each episode. The training phase battery status of the vSC in residential area are given in Fig. 5.2 for QL and SARSA algorithms, for one seed of training. It can be noted that during initial hours of training, the algorithm is unstable and the battery reaches very low level in many occasions. The training is performed with different seeds to add randomness in the energy arrival process. On average over 10 different seeds, the QL achieves stability after about 1750 hours of training whereas SARSA reaches stability after about 2000 hours of training. This is the average training duration computed for 10 seeds and can be interpreted as the duration after which the algorithm can be deployed for operation while improving its performance. Maintaining the battery level above the threshold is an important stability requirement since a wrong decision that results in a lower than threshold battery level can have negative consequences such as, damage in the storage system [126]. For this reason, the battery level is a good indicator to determine the stability of the algorithm.

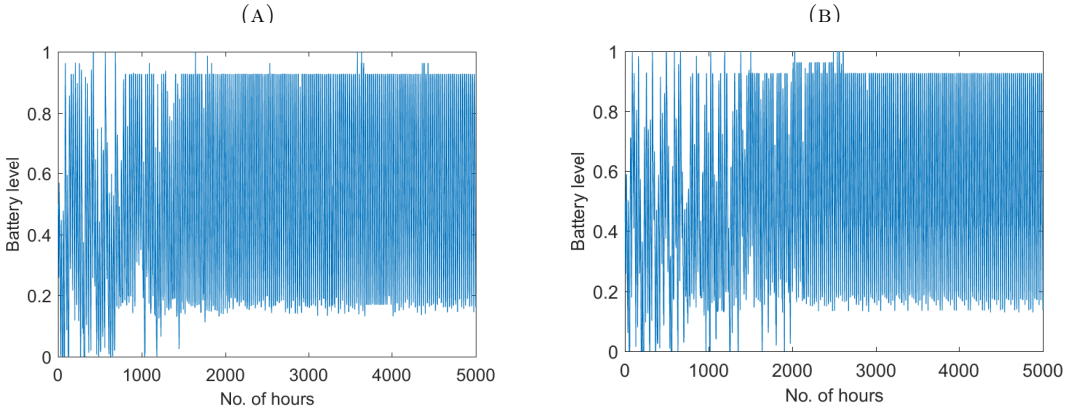


FIGURE 5.2: Battery level during the training phase: (a) QL (b) SARSA

TABLE 5.1: Training parameters values

Parameter	QL	SARSA
α	0.8	0.8
γ	0.9	0.9
ε	0.5	0.65
ε-discount	0.1	0.2

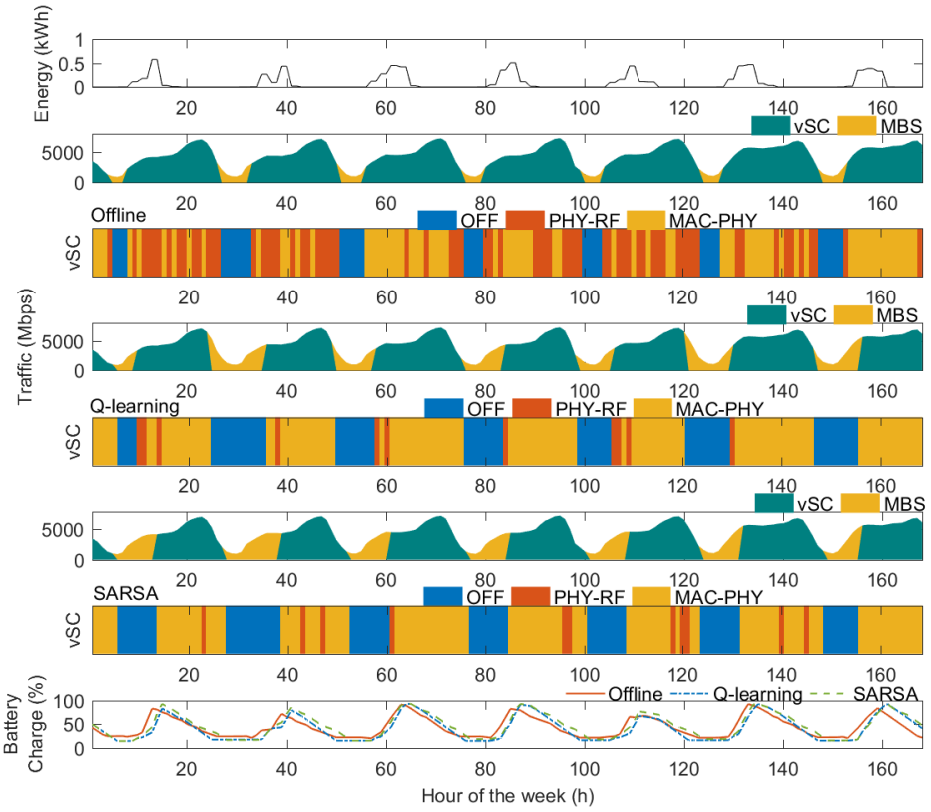


FIGURE 5.3: Functional split selection results in a residential area scenario for a week of January (hour 0 to hour 168; Monday from 0 - 23 hr). The traces show the amount of harvested energy, the amount of mobile traffic handled by vSC and MBS and operative mode of the vSC for a January week for offline, QL and SARSA policies.

5.4.2 Policy Characteristics and Comparison

This section describes the policies obtained by QL and SARSA and compares them with the offline optimal policy described in Section 4.4. The offline optimization is based on DP, in particular shortest path search, to determine the optimal policy for each hour of operation by taking into account the battery level, energy arrival and traffic requests. The offline optimization is used as a bound to evaluate how the performance of the online approaches is close to the optimal bound.

For both SARSA and QL, the training is performed for one year and the evaluation is done for a year of operation after the training. The training and evaluations are done for ten different seeds. A sample of the output of the policies for a week of January and for a residential traffic profile is shown in Fig. 5.3. As it is shown, both QL and SARSA are able to switch-off during low traffic periods as it is the case also for the optimal offline policy. However, the optimal offline policy and the

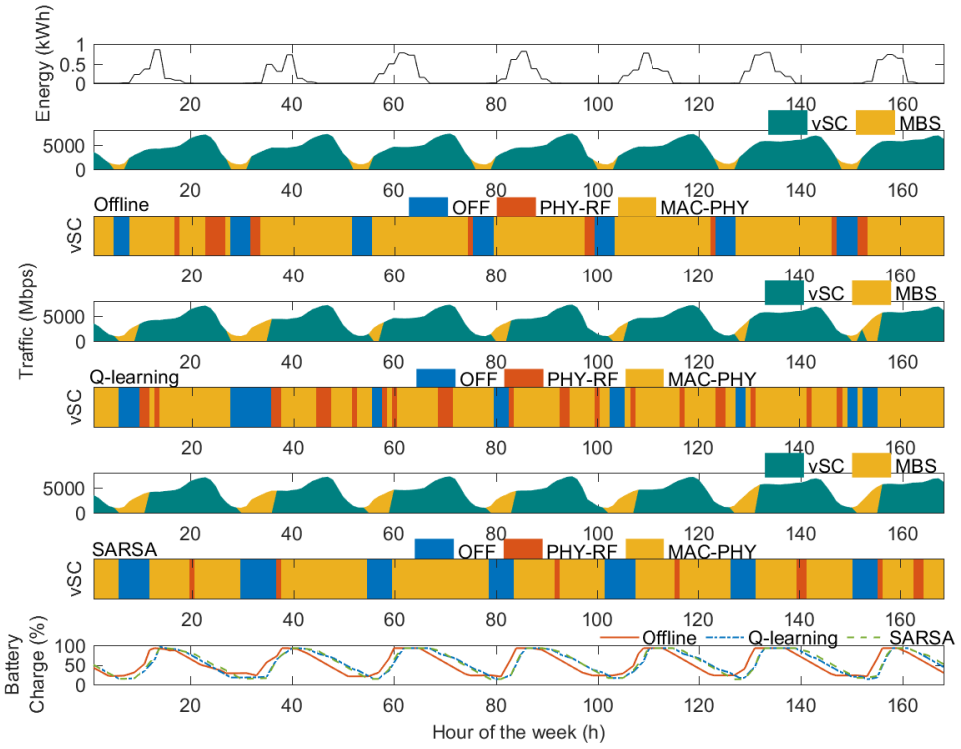


FIGURE 5.4: Functional split selection results in a residential area scenario for a week of July (hour 0 to hour 168; Monday from 0 - 23 hr). The traces show the amount of harvested energy, the amount of mobile traffic handled by vSC and MBS and operative mode of the vSC for a July week for offline, QL and SARSA policies.

TD policies differ in the selection of the operative modes. Both SARSA and QL choose the MAC-PHY split mode for most hours of operation, whereas the offline approach chooses the MAC-PHY and PHY-RF splits evenly. A sample output of the policies for July week and residential traffic profile is shown in Fig. 5.4. This month is characterized by high energy income. As a result, it can be observed that the policies adjust the operative mode decisions accordingly. All the three policies switch-off during very low traffic periods and all the three policies select MAC-PHY mode predominantly. In addition, SARSA shows higher switch-off rate with almost negligible PHY-RF selection mode whereas QL maintains relatively lower switch-off rate by selecting PHY-RF mode during some hours of operation. Residential profile is shown here as it represents traffic peaks during night when energy arrival is zero. Hence it is the most representative profile to evaluate the performance of the TD learning policies.

The policies output for a year of operation for three traffic profiles, namely residential, office and transport area are shown in Table 5.2. The results are the averages of ten different simulations running over one year of operation. As it

is shown, both QL and SARSA are able to consume a grid energy which is very close to the amount of grid energy consumed by the offline optimal policy. For a year of operation, the average annual grid energy consumption by QL policy is only 2.5%, 1.9% and 1.6% higher than the amount consumed by the offline policy, in residential, office and transport areas respectively. Moreover, for SARSA, the respective residential, office and transport area average annual grid energy consumption are 4.3%, 2.6% and 2.3% higher than the annual consumption by the offline policy. Furthermore, both QL and SARSA are able to achieve a total offloaded traffic of more than 90% with respect to the total offloaded traffic by the offline policy. The TD learning policies also approximate the optimal offline policy in terms of predominant operative mode selection rate. In all the three profiles, MAC-PHY is an operative mode with highest selection rate by the offline policy and this is also reflected by the TD learning policies. In MAC-PHY split mode, most of the BB processes takes place at vSC utilizing the harvested energy. Hence, the high MAC-PHY split selection rate helps the TD learning policies to achieve close to the optimum annual grid energy consumption. The TD learning polices tend to have relatively higher switch-off rate than the offline policy. This shows the conservative nature of the proposed online policies as compared to the offline approach.

It can also be noted that SARSA policy shows relatively higher PHY-RF split rate than QL. This explains the higher grid energy consumption by SARSA than QL. Hence, for all traffic profiles, QL performs better both in terms of grid energy consumption and offloaded traffic. This can be attributed to the nature of Q-values update in QL. As an off-policy method, QL learns the optimal behavior by choosing the maximum return that can be gained from one state-action transition to the next one, regardless of what the current policy might choose.

Finally, it is worth noting here that single agent TD learning algorithms are proved to perform optimally given the system model [127]. However, the two proposals in this work rely on different models with respect to the one used to solve the offline optimization problem. In particular, the reward function used by SARSA and QL algorithms presents minor modifications to the cost function of the offline optimization, which results in the different performance as presented in this section.

TABLE 5.2: Policy Comparisons (R = Residential, O = Office, T = Transport, OP = optimal-offline, QL , S = SARSA)

Profile	Policy	Grid energy (KWh)	Offloaded traffic (10^7Mbps)	Switch-off rate (%)	PHY-RF rate (%)	MAC-PHY rate (%)
R	OP	6780	3.47825	18.26	19.89	61.84
	QL	6952	3.1838	24.3	10.3	65.4
	S	7074	3.1	26	15. 02	58.97
O	OP	6572	1.450	26.55	1.148	72.29
	QL	6700	1.397	26.88	5.41	67.71
	S	6748	1.364	25.77	9.4	64.82
T	OP	6507	0.680	26.63	0.26	73.1
	QL	6611	0.573	30.29	1.5	68.2
	S	6661	0.531	34.6	6.72	58.61

5.5 Conclusions

In this chapter, an online optimization approach based on TD learning and more specifically QL and SARSA algorithms have been applied to determine the optimal functional split configurations. In particular, three operative modes namely, PHY-RF split mode, MAC-PHY split mode and switching off have been targeted. Such online approaches are evaluated and compared with respect to an offline optimal policy. Simulation results prove that the two proposed methods perform close to the optimal bounds and confirm the validity of our approach. In addition, QL is observed to perform better than SARSA both in the amount of annual grid energy consumption and offloaded traffic.

Chapter 6

The Case of Multi-vSCs

6.1 Introduction

In Chapter 5, RL based algorithm for dynamic placement of functional split options have been proposed. It is based on TD learning methods, namely QL and SARSA [8], for on-line learning of control policies of a vSC powered by EH with flexible operative modes. In this case, RL allows learning of optimal strategy through interaction with the system environment for achieving system wide objectives, i.e., efficient utilization of the harvested energy. However, when considering the case of various vSCs operating simultaneously, RL is expected to face more problems. Centralized solutions might experience long convergence and training phases due to the high number of state/action pairs needed to model the environment. Alternatively, a distributed approach, i.e, multi-agent RL, may allow to reduce the complexity by dividing the problem among the multiple agents. Nevertheless, multi-agent systems can experience issues due to the conflicting interests of the agents [128]. In fact, when each vSC is allowed to learn the best energy management policy independently, there is a risk that its actions affect other vSCs' policies, which in turn would have a negative effect on the overall performance of the network, e.g., system drop rate. Hence, multi-agent RL based strategy should ensure coordination among the learning agents, i.e., the vSCs, towards achieving system wide gains.

This chapter proposes multi-agent RL based on-line algorithms for dynamic selection of functional split options in MEC-enabled RAN with EH capabilities. Both FQL and QL based on-line algorithms are proposed with performance comparisons and evaluation against an off-line bound. Coordination among the multiple agents is favored by broadcasting system level information to the independent learners.

A comparison with an implementation of the learning algorithms without coordination is also analyzed.

The main contributions of this chapter may be summarized in the following items:

- *Grid Energy Minimization Problem Statement*: we formulate a network wide grid energy and traffic drop rate minimization problem for the proposed network scenario.
- *Coordinated Multi-agent RL Solutions*: we propose multi-agent RL controllers to solve the optimization problem. In particular, distributed FQL and QL based solutions are tailored for our purposes, including different levels of coordination among the vSCs.
- *Characterization of the Learning Algorithms*: we analyze the complexity and the convergence of the proposed learning algorithms by giving insights of the hyperparameter setup in both simulative training and run-time scenarios. We study the effects of the quantization of states and actions on the stability and system performance. We characterize the selected policy of the coordinated solutions with respect to the off-line bound.
- *Network Performance Evaluation*: we evaluate the network performance (in terms of energy consumption and traffic drop rate) by our multi-agent RL solutions with different levels of coordination and compare them against the off-line performance bound and static solutions.

The considered network scenario and the system model are described in Section 6.2. In addition, the design of distributed RL based controllers are shown in Section 6.3. The numerical results are analyzed in Section 6.4 and finally the conclusions are drawn in Section 6.5.

6.2 Network Scenario and System Model

A two-tier network architecture consisting of a MBS and co-located BBU pool and hot-spot deployed vSCs is considered. The network scenario is described in detail Sections 1.2 and 4.2. The vSCs are deployed in hot-spot manner for capacity enhancement and they do not overlap in coverage [27]. The vSCs in our scenario can opportunistically operate in one of the functional split configuration options, which are based on [31] and are explained in Section 5.2. These two functional

split options have been selected based on their impact on the energy consumption of vSCs following the results in Section 4.4. The functional split options are depicted in Section 5.2, Fig. 5.1 along with the conventional eNodeB architecture. Each operative mode corresponds to different computational load for the vSCs and MBSs, which in turn, corresponds to different energy consumption models, as will be described in Section 2.6.

In the following, we provide the system model for the dynamic control of functional split options for multi-vSCs scenario. We consider a two-tier mobile network composed of clusters of one MBS with co-located BBU pool and N vSCs. The system evolves in time slots based on the variation of the traffic demand and the energy arrivals. Recalling the definitions used in Section 4.3, The traffic load at slot t generated by the users in the coverage of the vSCs is defined as $\boldsymbol{L}^t \triangleq [L_1^t, L_2^t, \ldots, L_N^t]$ where L_n^t is the traffic load at the n^{th} vSC. The traffic load experienced by the MBS in slot t is defined as ρ^t. The energy harvested by the vSCs in slot t is defined by $\boldsymbol{H}^t \triangleq [H_1^t, H_2^t, \ldots, H_N^t]$ while the battery states are denoted by $\boldsymbol{B}^t \triangleq [B_1^t, B_2^t, \ldots, B_N^t]$ where H_n^t and B_n^t are the harvested energy and the battery state of the n^{th} vSC. Moreover, the battery states evolve according to the following relation:

$$\boldsymbol{B}^{t+1} = \min\left(\boldsymbol{B}^t + \boldsymbol{H}^t - \boldsymbol{P}^t \Delta_t, B_{\text{cap}}\right) \tag{6.1}$$

where $\boldsymbol{P}_t \triangleq [P_1^t, P_2^t, \ldots, P_N^t]$ and P_n^t is the power consumed by the n^{th} vSCs in slot t (and described in Section 2.6), B_{cap} is the maximum battery capacity and Δ_t is the duration of one time slot. The operative state of the vSCs in slot t is defined by $\boldsymbol{A}^t = [A_1^t, A_2^t, \ldots, A_N^t]$ and the mode of n^{th} vSC in time slot t, A_n^t, is given by:

$$A_n^t = \begin{cases} 0 & \text{if the } n\text{-th vSC is OFF} \\ 1 & \text{if the } n\text{-th vSC is in PHY-RF split mode} \\ 2 & \text{if the } n\text{-th vSC is in MAC-PHY split mode} \end{cases} \tag{6.2}$$

At each slot, intelligent decisions are made to determine the optimal configuration of the vSCs based on their battery state, energy arrival information and the traffic demand. The network wide sequential decision making problem is defined by a MDP as $\boldsymbol{X}^{t+1} = f(\boldsymbol{X}^t, \boldsymbol{A}^t, \boldsymbol{w}^t)$, where $\boldsymbol{X}^t = [X_1^t, X_2^t, \ldots, X_N^t]$ is the state of the vSCs in slot t, $\boldsymbol{A}^t = [A_1^t, A_2^t, \ldots, A_N^t]$ is the control action and $\boldsymbol{w}^t = [w_1^t, w_2^t, \ldots, w_N^t]$ is the random disturbance of the environmental variables. In particular, we define each state X_i^t, with $i = 1, ..., N$, as $X_i^t = (H_i^t, B_i^t, L_i^t, \rho^t)$.

The optimization objective is to minimize both the energy consumption from the grid and the traffic demands that cannot be satisfied due to vSCs in the OFF mode. As shown in Section 4.4.1, the optimization goal at each decision slot t will be to minimize the total weighted cost over a finite time horizon formulated as follows:

$$\begin{aligned} \text{P1:} \min_{\{\boldsymbol{A}^t\}_{t=1,\dots,K}} & \sum_{t=1}^{K} f(\boldsymbol{A}^t) \\ \text{subject to } & B_i^t > B_{\text{th}} \ \forall i. \end{aligned} \tag{6.3}$$

where B_{th} is the battery threshold level and K is the time horizon or the number of times the energy control is applied; $f(\boldsymbol{A}^t)$ is the weighted cost function in slot t, defined in Section 4.4.1 as:

$$f(\boldsymbol{A}^t) = \omega_1 \cdot \mathrm{E_m}(\boldsymbol{A}^t) + \omega_2 \cdot \mathrm{D}(\boldsymbol{A}^t) \tag{6.4}$$

where $\mathrm{E_m}(\boldsymbol{A}^t)$ and $\mathrm{D}(\boldsymbol{A}^t)$ are respectively the normalized grid energy consumption and the traffic drop rate in the cluster, given the operative modes of the vSCs in slot t. The grid energy consumption in the slot t is equivalent to the energy consumption at the MBS site, including the MEC server used for computational offloading. The grid energy consumption is then computed as:

$$\mathrm{E_m}(\boldsymbol{A}^t) = \mathrm{P_m}(\boldsymbol{A}^t)\Delta_t \tag{6.5}$$

where $\mathrm{P_m}(\boldsymbol{A}^t)$ is the power consumption of the MBS given the operative modes of the vSCs. The details on the power consumption models are described in Section 2.6. The traffic drop rate in slot t, $\mathrm{D}(\boldsymbol{S}^t, t)$, is the ratio of the total amount of traffic demand that cannot be served by the system in the slot t. The weights ω_1 and ω_2 determine the balance between the two objectives and $\omega_1 \geq 0$, $\omega_2 \geq 0$, $\omega_1 + \omega_2 = 1$. In this work, we will consider $\omega_1 = \omega_2 = 0.5$ to impose equal importance on the two objectives, but the results can easily be generalized to arbitrary weights.

A centralized off-line solution of this problem is proposed in Chapter 4 using DP and with a priori knowledge of the environmental variables. The problem of finding optimal configuration options is represented as a graph and stated as a shortest path search, while label correcting method is used to explore the graph and find the the shortest path. Those obtained results are considered as system performance bounds and are used as a benchmark to the control methods proposed

in this chapter. Here, we propose an on-line solution based on multi-agent RL. In particular, we use approximated DP methods, known as TD learning, to determine optimal policies [8]. Our proposal is based on distributed and coordinated decision making: i.e., each vSCs take actions by itself, which makes it scalable with the number of vSCs. In order to coordinate the decision making process, we rely on communicating system wide information, e.g., traffic load at MBS, to each vSCs. Section 6.3 describes the proposed QL and FQL based control solutions to the sequential decision making process described here. The power consumption model is described in Section 2.6. Moreover, the EH and traffic demand profiles are described in Section 4.3.1.

6.3 Distributed Control Methods

In this section, we introduce our distributed and coordinated multi-agent RL solutions and we focus on the details about both FQL and QL based controllers design. In the distributed QL and FQL controllers, each vSC are modeled as QL and FQL agent, respectively, and the agents decisions are coordinated via broadcasting MBS traffic load information to each vSC. For detailed background on RL and multi-agent RL including the algorithms of QL and FQL, the reader can refer to Chapter 3.

6.3.1 QL controllers

In what follows, we describe the definition of states, reward and actions for QL based controller for solving our MDP.

1. *States:* According to the system model defined in Section 6.2, the state of the i^{th} vSC modeled as QL agent, $X_i^t = (H_i^t, B_i^t, L_i^t, \rho^t)$ is composed of the energy arrival H_i^t, the battery level at the vSC, B_i^t, the traffic load at MBS, ρ^t, and the traffic load at vSC, L_i^t.

   The values of each state variables are all normalized and quantized into 5 levels. Hence, the QL based controller has $5 \times 5 \times 5 \times 5 = 625$ states. Since our optimization objective is system-wide where as the vSCs are taking actions in distributed fashion, the coordination among vSCs is achieved by including the MBS traffic load information in the states of the controller at each vSCs. In this way, vSCs action selection can be coordinated towards minimizing

the MBS load which, in turn, is equivalent to minimizing the grid energy consumption.

2. *Actions:* The set of possible actions are the possible operative modes of the vSCs, $\boldsymbol{A}^t$. The action set for the i^{th} vSC are switching off, PHY-RF split mode, or MAC-PHY split mode. Hence, the action set for the whole QL solution is a combination of the three operative modes of each vSC.

3. *Reward:* The reward function determines the immediate reward the controller acquire as a result of taking a specific action. The optimization goal is to minimize the power drained from the grid while reducing traffic drop rate as given by (6.4). Hence, the reward function can be formulated as:

$$r_t = 1 - (\omega_1 \cdot \mathrm{E_m}(\boldsymbol{A}^t) + \omega_2 \cdot \mathrm{D}(\boldsymbol{A}^t)) \tag{6.6}$$

where $\mathrm{E_m}(\boldsymbol{A}^t)$ and $\mathrm{D}(\boldsymbol{A}^t)$ are respectively the normalized grid energy consumption and the traffic drop rate in the cluster, given the operative modes of the vSCs and the time step t.

Each vSC agent applies QL procedure as shown in Algorithm 3.

6.3.2 FQL controllers

FQL allows to integrate the benefits of FIS in QL: provide good approximations of the Q-function and enable the use of QL in continuous state spaces [109]. In FQL, let $\boldsymbol{X}$ be the crisp set of inputs defining the state of a learning agent. Crisp values are the exact values of the inputs without any form of pre-processing. The process of converting the crisp values to fuzzy values is known as fuzzification. Fuzzification is done according to the degree of membership determined from membership functions. In our scenario, we define the membership functions for the traffic load, energy arrival and battery as well as actions and reward functions, as follows.

1. *Membership functions and fuzzy rules:* The crisp input state of i^{th} vSC X_i^t is defined in Section 6.2 as $X_i^t = (H_i^t, B_i^t, L_i^t, \rho^t)$. Trapezoidal and triangular membership functions are used for the traffic load at MBS, traffic load at vSCs, energy arrival and battery level of vSCs. In particular, 5 fuzzy sets are defined with linguistic terms "Very Low", "Low", "Medium", "High" and "Very High" as shown in Fig. 6.1. Hence, the fuzzification step involves mapping the input X_i^t into 5 fuzzy sets for traffic load at MBS, traffic load

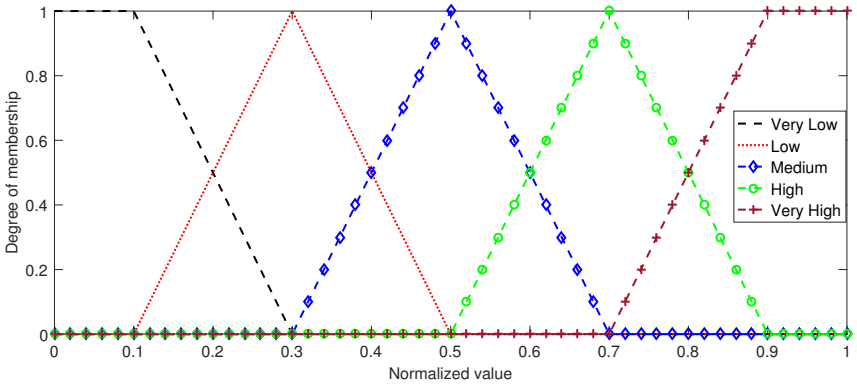


FIGURE 6.1: Membership functions of MBS traffic load, vSC traffic load, energy harvesting and battery level

at each vSCs, energy arrivals and battery level of each vSCs. Hence, there are 625 rules corresponding to every combination of fuzzy sets in the FQL. Similarly as for QL-based controller design, MBS traffic load information is included in the fuzzy rules definition of each vSCs controller to achieve coordination among vSCs towards a common optimization goal of minimizing grid energy consumption while avoiding system outages.

2. *Actions:* The set of possible actions are the possible operative mode of each vSC. An action for each rule is determined by using the exploration/exploitation policy as shown in step 2 of Algorithm 5. After computing the firing strength of each rule using membership functions, the global action is computed as a weighted sum of each action and the corresponding firing strength using (3.9). This defuzzification method is commonly applied in zero order Sugeno fuzzy systems [129] and is known to be computationally efficient. In our controllers, since the set of actions are limited (3 operative modes of each vSC), the crisp output obtained by the defuzzification method using (3.9) is converted to a nearest integer, which corresponds to an operative mode of a vSC.

3. *Reward:* The reward function is the same as defined for QL control in (6.6).

The procedure of FQL algorithm is shown in Algorithm 5.

6.3.3 Control Without Coordination

This section presents the control methods where each vSCs take actions independently without any system wide information. In this case, as opposed to the QL

and FQL methods described above, the vSCs did not have the load level of the MBS and this is not considered in their decision making process. As a result the state/rules of i^{th} vSC in un-coordinated methods are given by:

$$X_i^t = H_i^t, B_i^t, L_i^t \tag{6.7}$$

Therefore, the un-coordinated methods have $5 \times 5 \times 5 = 125$ rules/states for FQL and QL methods respectively. We call these methods as Un-Coordinated FQL (U-FQL) and Un-Coordinated QL (U-QL). The actions and reward definitions of U-FQL and U-QL methods are the same to the actions and rewards defined above for both FQL and QL controls.

6.4 Numerical Results

We considered a simulation scenario as described in 4.5.1. Moreover, energy arrivals and aggregated downlink traffic have been generated according to the realistic models described in Section 4.3.1.

6.4.1 Off-line Training

In this section we analyze the behavior of the system when the training is performed off-line. In particular, we considered one year as an episode with time granularity of one hour, since it allows to achieve a correct dimensioning of the solar power system for cellular BSs, as shown in [130]. Therefore, each training epoch has 8640 decision slots. Hence, every hour the agents choose actions corresponding to one of the possible operative modes of the vSCs with the goal of minimizing grid energy consumption, while reducing system drop rate. The training phase requires calibrating the parameters of the algorithm that have the strongest impact appropriately. These parameters are the learning rate (α), the exploration parameter (ϵ) and discount factor (γ). Moreover, we also adopt a discount process on these parameters in order to guide the exploration toward the stability. In particular, we are applying an exploration discount factor of 0.5 at the beginning of each epoch until the agent reaches minimum level of exploration, which is equivalent to 3%.

The cumulative reward of FQL and QL methods for a system composed of 3 vSCs and a MBS with in a residential area traffic profile are shown in Fig. 6.2. The

cumulative reward is normalized with respect to a cumulative reward bound which is determined off-line using DP [131], as shown in Chapter 4. As it can be seen from Fig. 6.2a, FQL based control can achieve a cumulative reward very close to the optimal bound (more than 97%). In addition, the choice of training parameters affects the convergence of the FQL control. The cumulative reward is shown to be sensitive to the exploration and learning rate parameter choices as it can reach the 85% level in the case of $\alpha = 0.01$ and $\epsilon = 0.5$. The cumulative reward of QL based control in residential area is shown in Fig. 6.2b. In the best case, the cumulative reward obtained by QL (94%) is close to the optimal bound but lower with respect to FQL.

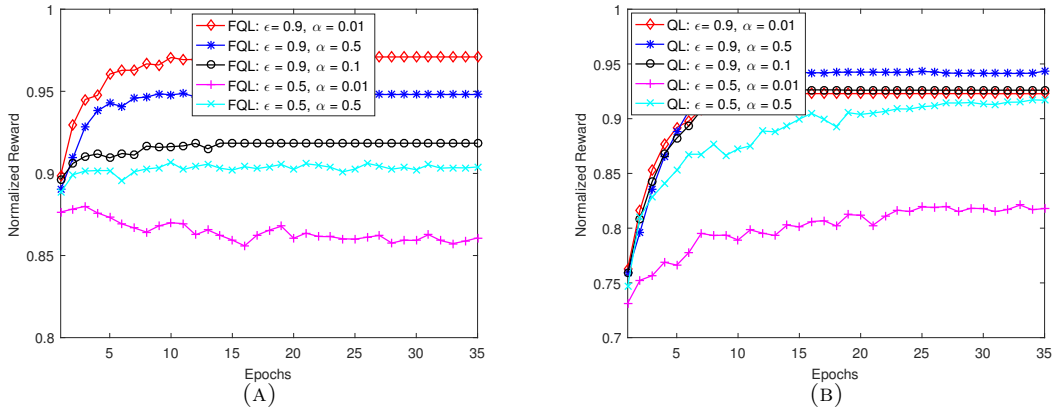


FIGURE 6.2: Cumulative reward in residential area corresponding to different training parameters: (a) FQL (b) QL

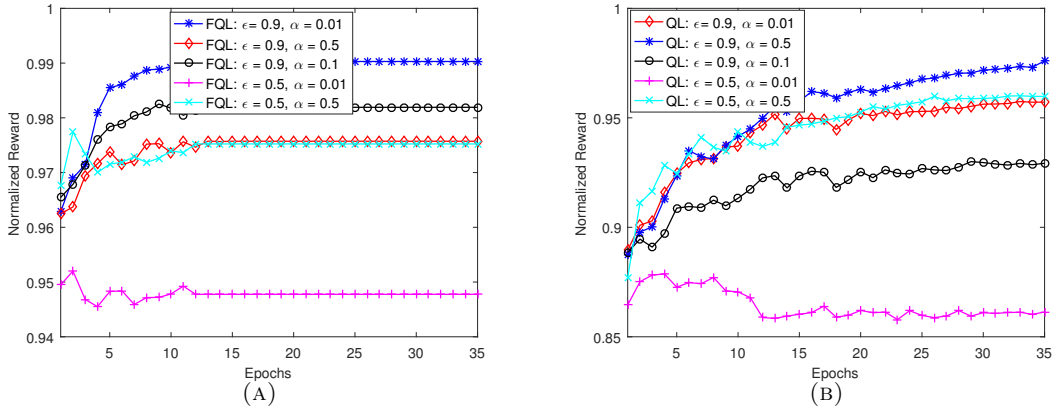


FIGURE 6.3: Cumulative reward in office area corresponding to different training parameters: (a) FQL (b) QL

The normalized cumulative reward for a system of 3 vSCs deployed in an office area is shown in Fig. 6.3. These results show that at the best case, FQL and QL controls in an office area are able to gain a cumulative reward of about 99% and 97% with respect to the optimal bound, respectively. The results in Fig. 6.2 and

Fig. 6.3 also show that, FQL based control is able to accumulate rewards faster than QL. In residential scenario, the FQL method is able to get around 95% of the reward in less than 5 epochs where as QL requires about 15 epochs to reach the same level of cumulative reward. For an office scenario, FQL achieves a cumulative reward of about 97% in less than 5 epochs, whereas QL requires about 20 epochs to reach 96% level. Moreover, higher initial exploration rate is important for both FQL and QL, since it enables to explore more actions randomly during the initial phase of the training. Thus, the agent in the vSC has already discovered a higher number of rules-actions/state-actions pairs for FQL and QL, respectively, which, in turn, help to avoid entering local optima.

The same analysis has been also applied for scenarios with 5, 7, 10, 12 and 15 vSCs. The maximum cumulative reward obtained by both QL and FQL based controllers in residential and office area are shown in Fig. 6.4. The results show that FQL is able to accumulate higher reward compared to QL, 35% more with 15 vSCs. It is to be noted that, the maximum cumulative reward is decreasing as the number of vSCs increases. This is due to the higher load in the system injected by the vSCs which generates higher system drop rate and, in turn, reduces the immediate reward. Moreover, as the number of vSCs increases, conflicts among the actions of the agents can emerge which can impact the immediate reward obtainable. In addition, the cumulative reward is higher in an office area. In fact, the peak of traffic in the residential profile occurs during the early night (11 pm), as shown in Fig. 4.2, when the energy income is low, thus forcing the agents to switch-off or choose actions with more computational offloading to MBS.

Finally, the maximum cumulative reward gap between FQL and QL increases with the number of vSCs, as can be seen in Fig. 6.4a and Fig. 6.4b. This highlights the better suitability of FQL control especially in a network of higher number of vSCs.

6.4.2 Policy Characteristics

The policy behavior of both FQL and QL based controls for a system with 3 vSCs are evaluated with respect to the off-line policy. The off-line solution, described in Chapter 4, is based on DP and aimed at determining the performance bound of dynamic functional split placement when system dynamics information are known a-priori. The policy behaviors of an off-line, FQL and QL based controls for 3 vSCs for an average winter day are depicted in Fig. 6.5 and Fig. 6.6 for residential and

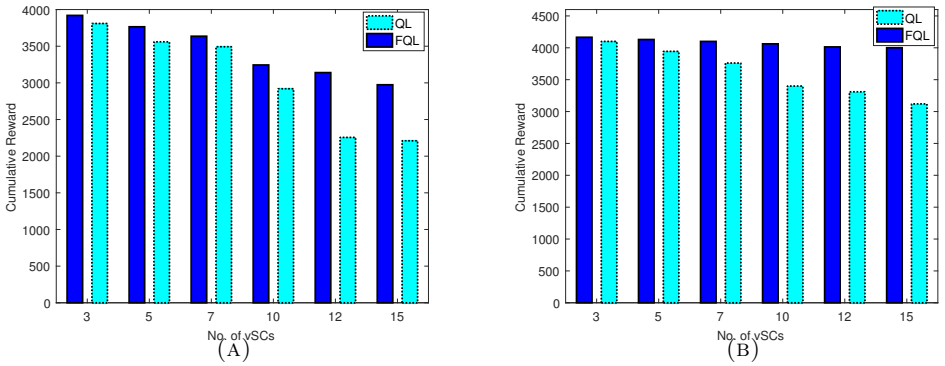


FIGURE 6.4: Maximum cumulative reward obtained by FQL and QL for 3, 5, 7, 10, 12 and 15 vSCs (a) Residential (b) Office

office area, respectively. Moreover the functional split selection behaviors for an average summer day are shown in Fig. 6.7 and Fig. 6.8 for off-line, FQL and QL controls in residential and office area, respectively. These results show that both FQL and QL polices usually switch-off most of the vSCs during very low traffic periods, as done by the off-line policy. However, the polices substantially differ in their respective functional split options selection when switched on. In this regard, QL is adopting a more conservative approach by selecting more PHY-RF split option as compared to the other solutions. In fact, FQL has a more similar behavior with respect to the off-line solution thanks to its higher flexibility in policy selection for both MAC-PHY and PHY-RF splits. In residential area and on average winter day, the MAC-PHY selection rates are 51%, 46% and 23% for off-line, FQL and QL controls respectively. For average summer day, the residential area MAC-PHY selection rate rises to 77%, 68% and 34% in off-line, FQL and QL solutions respectively. On the other hand, on average winter day in office area, the MAC-PHY selection rates are 62%, 50% and 34% for off-line, FQL and QL controls respectively. For average summer day, the office area MAC-PHY selection rate rises to 81%, 70% and 44% in off-line, FQL and QL solutions respectively. Hence, the FQL policy is able to have higher adaptation to the energy income of the seasons. It is also interesting to note that for an office area, the off-line solution configuration is predominantly between switch-off and MAC-PHY split. This can be clearly seen in Fig. 6.8, where the off-line solution has 0% PHY-RF selection rate on an average summer day.

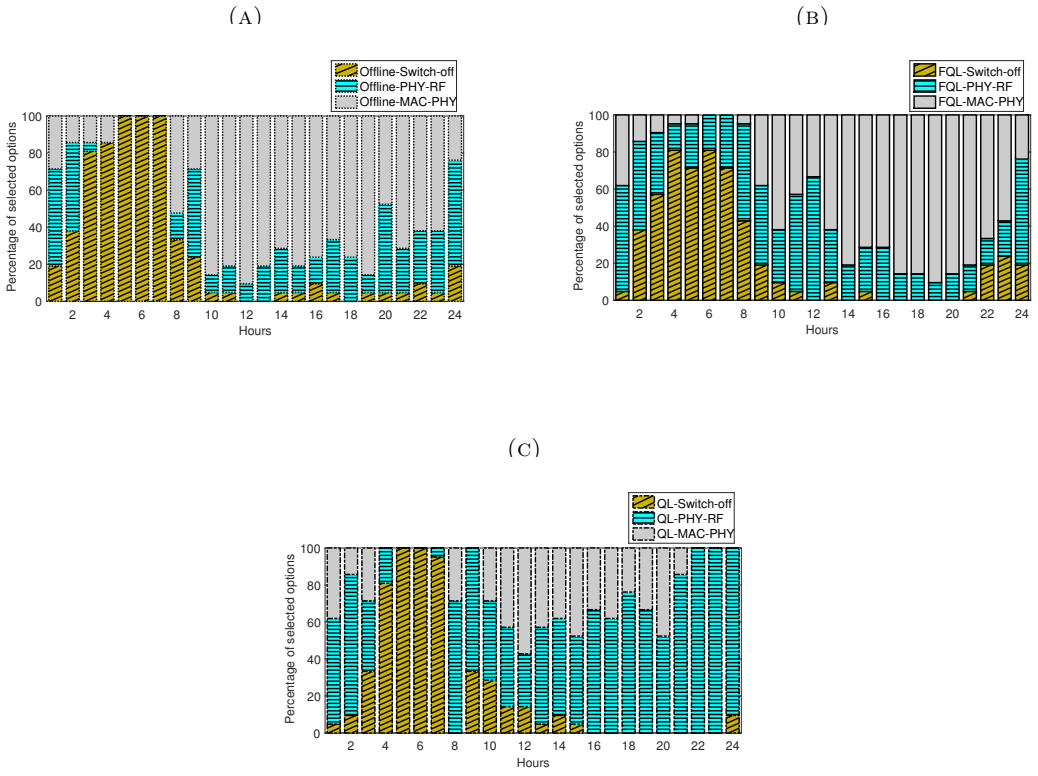


FIGURE 6.5: Average residential area winter day policy characteristics of: (a) Off-line (b) FQL (c) QL

6.4.3 Network Performance

This section evaluates the performance of the polices obtained in the training phase in terms of grid energy consumption and system drop rate parameters for a year of operation. Hence, the same energy arrival and traffic demand profiles used in the training phase are used for the evaluation of the network performance. First we compare the performance of the policies against the off-line bound in a scenario with 3 vSCs. Moreover, as a comparison bench mark, we considered uncoordinated solutions, i.e., U-FQL and U-QL presented in Section 6.3.3, and a greedy approach. The Greedy (G) approach works by keeping the vSC in PHY-RF mode all the time as long as the level of the battery is above a certain threshold (B_{th}). We call this static configuration as G-PHY-RF. The network performance results are shown in Table 6.1.

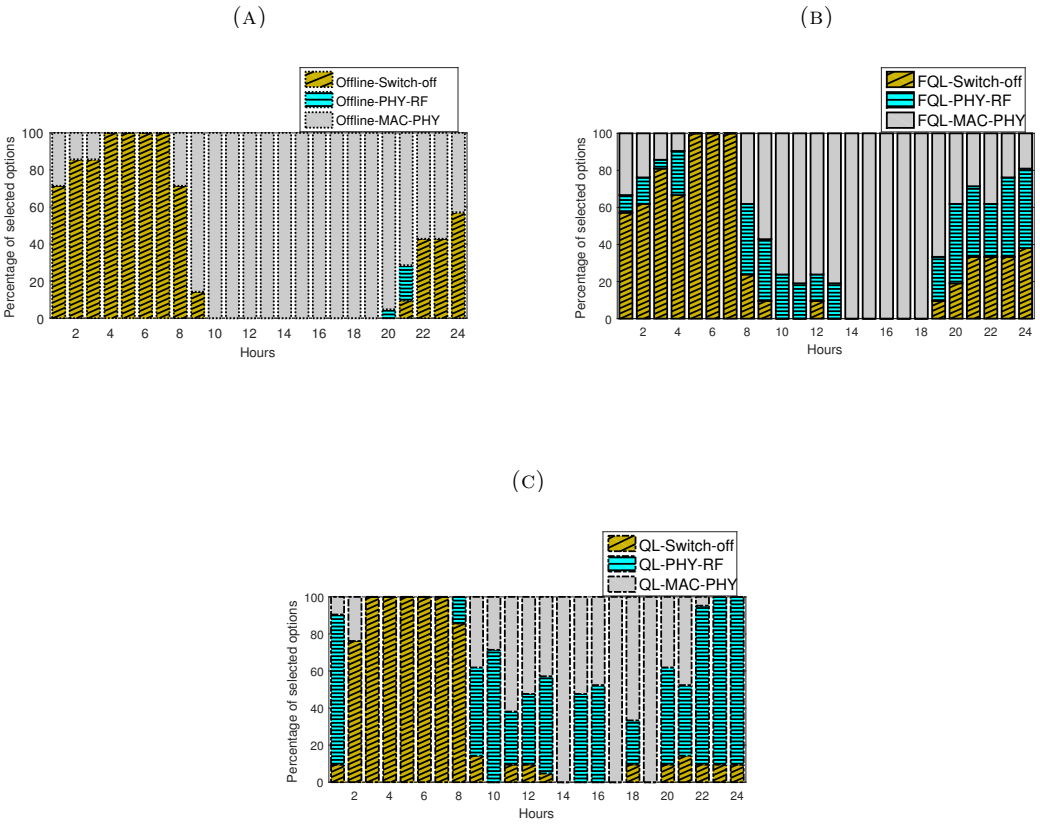


FIGURE 6.6: Average office area winter day policy characteristics of: (a) Off-line (b) FQL (c) QL

TABLE 6.1: Comparisons with respect to the off-line bound - 3 vSCs

Algorithm	Grid energy consumption (KWh)		Average drop rate (%)	
	Residential	Office	Residential	Office
Off-line	7403	6744	0.3	0.02
FQL	7695 (+3.9%)	7076 (+4.9%)	0.9	0.05
QL	8150 (+10%)	7289 (+8.1%)	0.5	0.03
U-FQL	8000 (+8.0%)	7487 (+11%)	1.2	0.1
U-QL	8202 (+10.8%)	7688 (+14%)	0.7	0.1
G-PHY-RF	8320 (+11.2%)	8232 (+18.1%)	3.3	0.8

These results show that FQL is able to achieve very low grid energy consumption which is only 4% to 5% higher than the energy consumption value obtained by the off-line bound for both office and residential profiles. On the other hand, QL policy consumes relatively higher grid energy which is about 8% to 10% higher than the

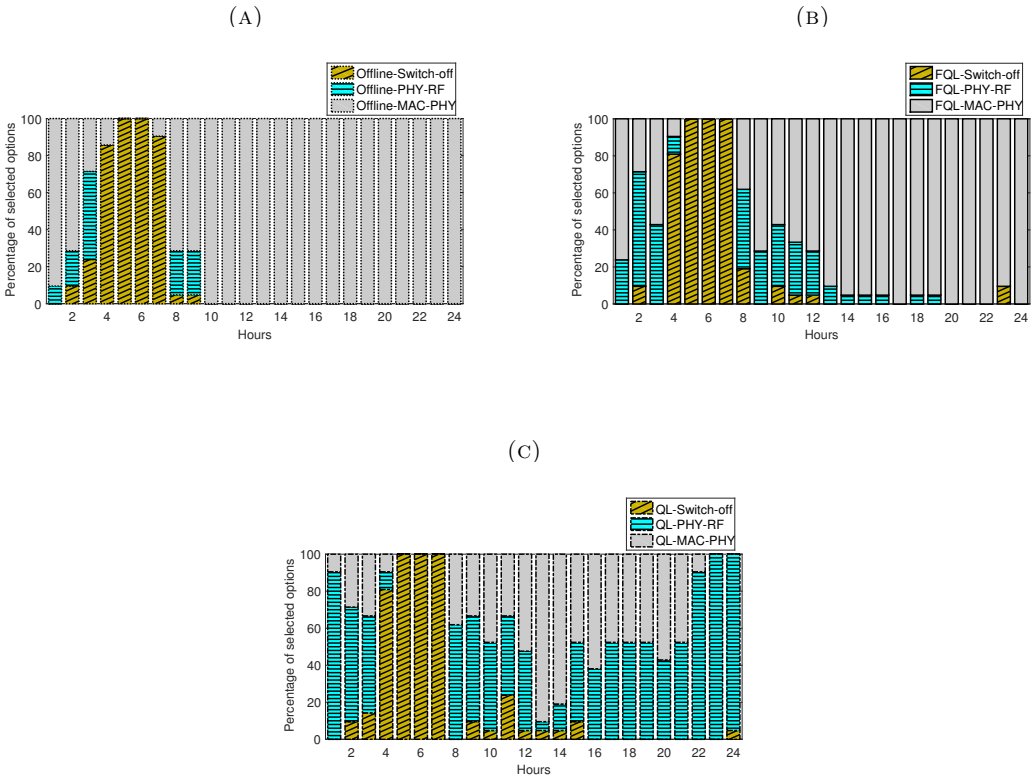


FIGURE 6.7: Average residential area summer day policy characteristics of: (a) Off-line (b) FQL (c) QL

off-line policy, for both office and residential area traffic. This can also be deduced from the policy behaviors in Fig. 6.5, Fig. 6.6, Fig. 6.7 and Fig. 6.8, which show that FQL has higher MAC-PHY selection rate than QL. In particular, the FQL policy shows adaptation to a higher energy income in summer months by increasing the selection rate of MAC-PHY split. This behavior is also observed from the off-line solution. With higher MAC-PHY selection rate, more energy saving can be achieved since most of the BB processing functions are performed locally at vSCs. The results in Table 6.1 also show that the policies without coordination, i.e. U-FQL and U-QL, exhibit lower performance than their coordinated counterparts, i.e. FQL and QL, respectively.

The grid energy consumption performances of FQL and QL based controllers in residential and office area for a year of operation for higher number of vSCs are shown in Fig. 6.9. Due to high computational demand of the off-line solution, we could not show the off-line bound results for vSCs higher than 3. Moreover, the traffic drop rate performances of both FQL and QL controls in residential and office area scenario are shown in Fig. 6.10. These results show that FQL policy performs better both in grid energy consumption and average drop rate in both residential

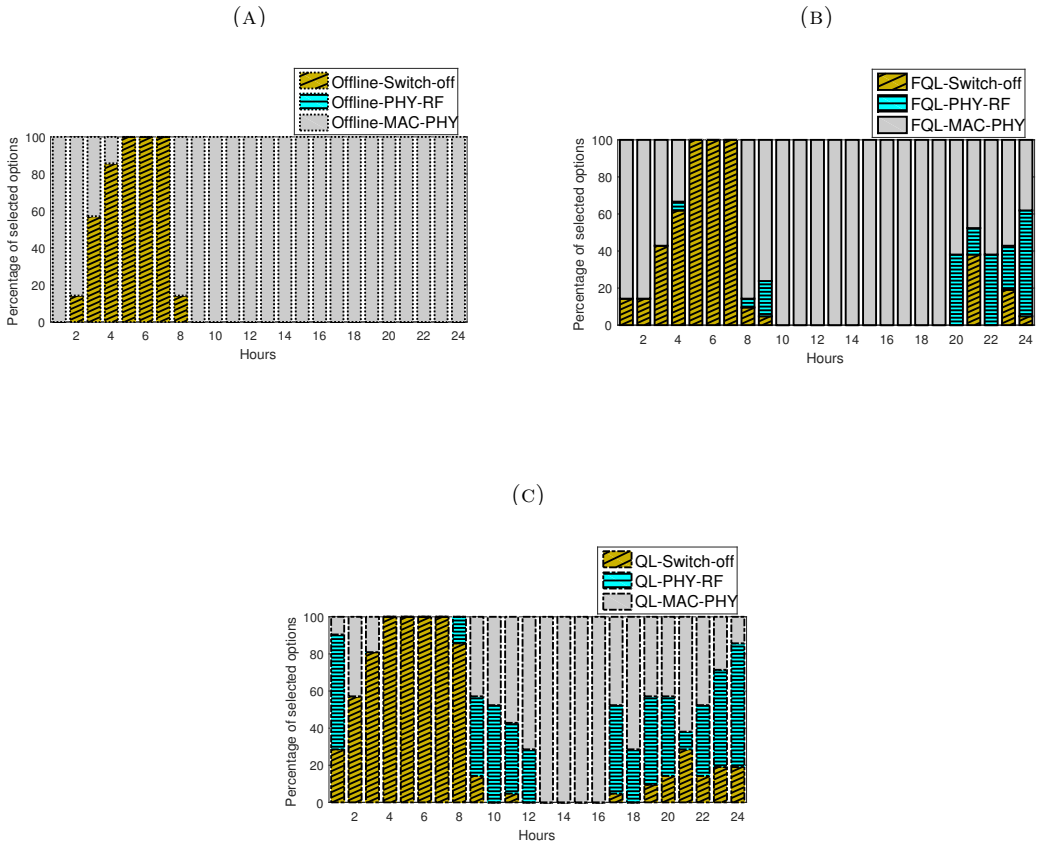


FIGURE 6.8: Average office area summer day policy characteristics of: (a) Off-line (b) FQL (c) QL

and office profiles for higher number of vSCs. The performance gap between FQL and the other solutions grows with the number of vSCs (7, 10, 12 and 15 vSCs). In residential area traffic, the FQL controller achieves an energy saving of up to 12% and an average drop rate of up to 10% less than the QL control. Moreover, FQL controller is able to achieve energy saving of up to 17% and average drop rate of up to 8% less than QL control in an office area traffic profile. The better performance by the FQL is aligned with the higher cumulative rewards obtained by the FQL controller as shown in Fig. 6.4. In addition, the results of uncoordinated solutions, i.e., U-FQL and U-QL, are shown for comparison. The solutions without MBS traffic load information, i.e. U-QL and U-FQL, have lower performances than the proposed QL and FQL counterparts both in energy consumption and average drop rate. In addition, the static configuration policy i.e., G-PHY-RF, presents the lowest performance, below than the RL based methods in both grid energy saving and average drop rate.

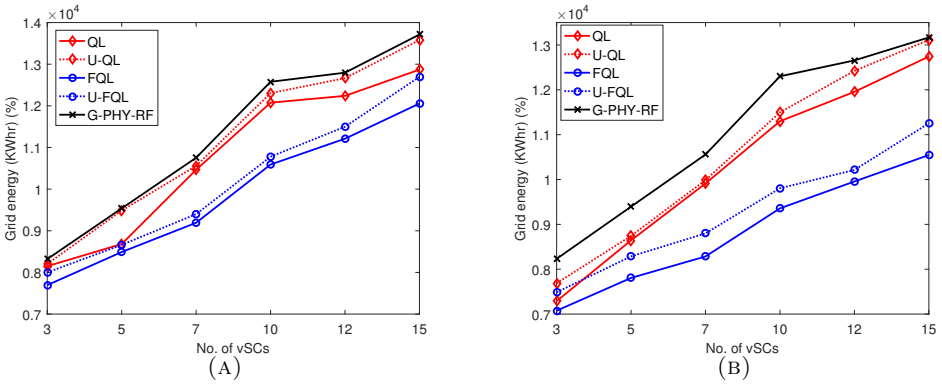


FIGURE 6.9: Grid energy consumption comparison among FQL, QL, U-FQL, U-QL and G-PHY-RF solutions: (a) Residential (b) Office

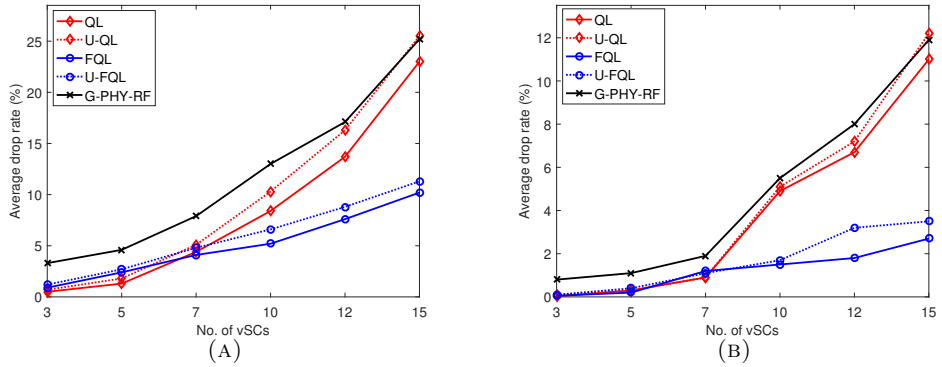


FIGURE 6.10: Average drop rate comparison among FQL, QL, U-FQL, U-QL and G-PHY-RF solutions: (a) Residential (b) Office

6.4.4 Policy Validation

In this section we evaluate the behavior of the system in real deployment scenario with a training performed off-line with simulation. In detail, we will validate the proposed FQL and QL based controllers using a new environment, which is characterized by an energy arrival and traffic demand profiles which are different from the environment used for simulated training. In particular, different EH traces and a slight variation in the traffic demand is applied in the new environment with respect to the environment used for the simulated training. In this case we are using the algorithms with pre-trained Q-values and with an exploration rate of 5%. The validation of the policies along with the training environment policy evaluation for 3, 5, 7, 10, 12 and 15 vSCs for a year of operation are shown in Table 6.2. The validation results in Table 6.2 show that both FQL and QL are able to adapt their behaviors to the new validation environment. This is confirmed by both grid energy and average drop rate performances that are very close to the corresponding policy evaluation results. These results give an insight that using

TABLE 6.2: Policy validation results
(R-T: Residential Training, O-T: Office Training, R-V: Residential Validation, O-V: Office Validation)

No. of vSCs	Algorithm	Grid energy consumption (KWh)				Average drop rate (%)			
		R - T	O - T	R - V	O - V	R - T	O - T	R - V	O - V
3	FQL	7695	7076	7757	7211	0.9	0.05	0.8	0.02
	QL	8150	7289	8330	7495	0.5	0.03	0.6	0.03
5	FQL	8490	7806	8527	7805	2.4	0.2	3	0.18
	QL	8682	8644	8903	8639	1.3	0.3	1.4	0.35
7	FQL	9193	8285	9189	8330	4.1	1.2	4.8	1.37
	QL	10466	9912	10482	9885	4.4	0.9	4.7	1.1
10	FQL	10591	9357	10606	9367	5.2	1.5	5.4	1.5
	QL	12076	11299	12177	11291	8.4	4.9	8.8	5.1
12	FQL	11211	9956	11222	10019	7.6	1.8	8.4	2.0
	QL	12240	11955	12216	11967	13.7	6.7	14. 1	6.7
15	FQL	12056	10546	12182	10623	10.2	2.7	10.6	3.2
	QL	12869	12742	12917	12711	23	11	23.4	11.1

simulated trained Q-values / rules-actions consequents for QL / FQL respectively, continuously exploring new actions in the new environment and updating the corresponding Q-tables is a viable approach in real deployment scenarios.

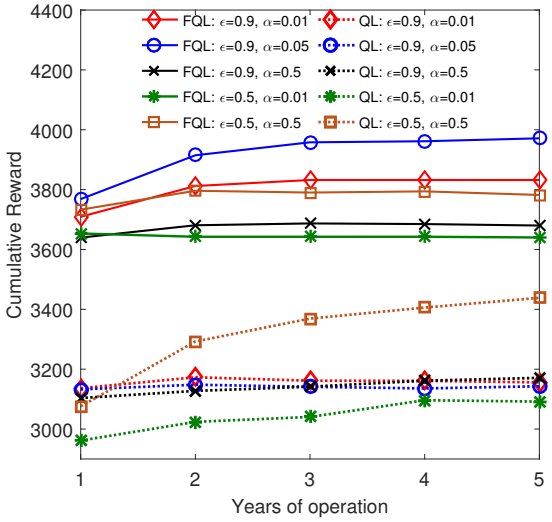


FIGURE 6.11: Cumulative reward for run-time training of FQL and QL in residential area for 3 vSCs

6.4.5 Run-time training

Here, we perform the evaluation of the proposed FQL and QL based controls in run-time deployment scenario without pre-training, i.e., the vSCs are learning on the job while they are in operation. In this case, all the Q-values / rules-actions

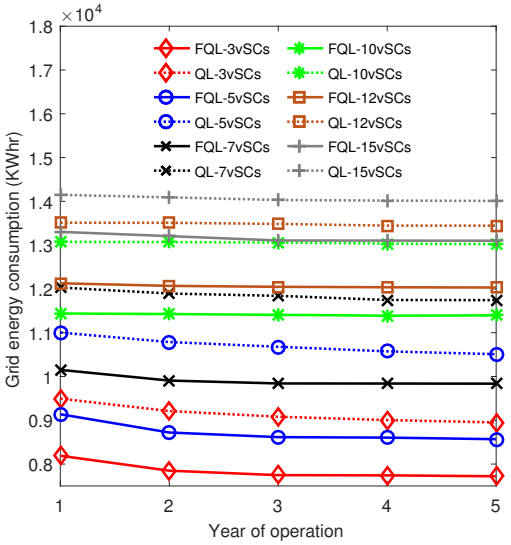


FIGURE 6.12: Grid energy consumption of run-time FQL and QL controls in residential scenario for 3, 5, 7, 10, 12 and 15 vSCs

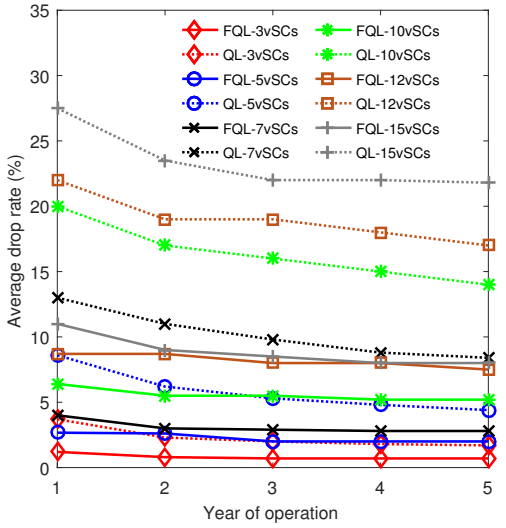


FIGURE 6.13: Average drop rate of run-time FQL and QL controls in residential scenario for 3, 5, 7, 10, 12 and 15 vSCs

consequences are initialized to 0 for QL and FQL, respectively. An exploration/-exploitation strategy is used for the learning of vSCs. In order to determine the effect of the learning parameters on run-time FQL and QL performances, we have compared the on-line training for different sets of learning rate (α) and exploration (ϵ) parameters for 3 vSCs. These results are shown in Fig. 6.11. The results show that FQL is able to gain higher cumulative rewards than QL starting from the first year of operation.

As a result, it is more suitable for run-time training of the vSCs than QL. Moreover,

lower values of the learning rate are better for FQL, whereas QL requires relatively higher learning rate. The exploration rate parameter shown in Fig. 6.11 are initial exploration rates, which are continuously discounted as the training progresses until reaching the minimum level of exploration, which is set at 5%.

The grid energy consumption performances of both FQL and QL controls for a run-time training and operation are shown in Fig. 6.12. Moreover the average drop rate performances of run-time controls are shown in Fig. 6.13. These results show that FQL policy is more suitable for run-time application, as shown both in terms of grid energy consumption and average drop rate.

Run-time FQL is able to gain an energy saving ranging from 10% to 17% with an average drop rate of 2.5% to 13% less, than run-time QL, in the first year of deployment. However, compared to policy validation results based on pre-trained agents, shown in Table 6.2, the run-time training and operation results shown in Fig. 6.12 and Fig. 6.13 display lower performances. For FQL controller, the validation results of pre-trained agents shown in Table 6.2, achieve energy saving ranging from 5% to 9.5% with a drop rate of 0.4% to 1% less, than the run-time results in Fig. 6.12 and Fig. 6.13. Moreover, for QL, the validation results in Table 6.2, show an energy saving ranging from 7% to 19% with a drop rate of 3% to 12% less, than the run-time results.

Hence, to get closer to the optimization goals, it is better to initialize vSCs' agents with some knowledge prior to their deployment. This can be in the form of simulated training of the vSCs, as shown in Section 6.4.1. As a result, training of the vSCs in a simulative environment prior to their deployment and allowing them to explore new knowledge while in operation is a more appropriate approach in real deployments.

6.5 Conclusions

In this chapter, we have investigated the joint grid energy and traffic drop rate minimization problem and proposed multi-agent RL to solve it for a scenario where the computational processes of the vSCs powered solely by EH and batteries may be executed on a grid-connected central MEC-server at the MBS. Distributed FQL and QL on-line algorithms are tailored for our purposes. Coordination among the multiple agents is achieved by broadcasting system level information (i.e. the traffic load at MBS) to the independent learners. Finally, we have evaluated the

network performance (in terms of energy consumption and traffic drop rate) by our multi-agent RL solutions with different levels of coordination and compared them against the off-line performance bound and static solutions. Our results confirm that coordination via broadcasting may achieve higher system level gains than un-coordinated solutions and cumulative rewards closer to the off-line bounds. Moreover, our analysis permits to evaluate the benefits of continuous state/action representation for the learning algorithms in terms of faster convergence, higher cumulative reward and more adaptation to changing environments.

Chapter 7

The Case of Multi-vSCs: Deep RL Approach

7.1 Introduction

Multi-agent RL based online algorithms for dynamic placement of functional split options in MEC-enabled RAN with EH capabilities have been proposed in Chapter 6. The methods are based on distributed RL, in particular QL and FQL. Coordination among the learning agents is favored via broadcasting system-wide information, i.e. the traffic load of MBS. In this Chapter, instead we propose a coordination scheme based on the exchange of specific local state information among the agents. Hence, we increase the amount of knowledge shared in order to reduce conflicting behaviors, and, in turn, converge to stationary policies with higher system wide gains. The main challenge with this approach is the exponential increase in the state space dimension, which may slow down the learning process and even jeopardize its convergence. Tabular multi-agent RL methods (e.g., QL, FQL) work through mapping each state to a value; hence, each state-action pair need to be properly explored. In problems with continuous state variables, as ours, this mapping is performed through quantization (QL) or fuzzy inference systems (FQL) and may result in large state-action spaces, which slow down the learning process. For instance, the solutions proposed in Chapter 6 relies on broadcasting system level information and have 4 state variables corresponding to energy, battery, local traffic load and system traffic load. Let z denote the level of quantization, then the state size of each agent in Chapter 6 is z^4. For more coordination, if vSCs exchange battery state information, the number of states will be multiplied by a factor of z^N, where N is the number of vSCs in the system. This implies an increment in the size of the states from 625 in Chapter 6 to a range from 78125 to $1.9073486 \times e^{13}$, for $z = 5$, in a scenario of 3 and 15 vSCs, respectively. Therefore, tabular RL methods can not be applied on such a large state space.

Artificial neural networks can be used to approximate the Q-values of state-action pairs [110], known as DRL, described in Chapter 3. DRL allows working with large state and action spaces through Q-value estimations without the need for large and impractical look-up tables. Here, we propose a Distributed DRL (DDRL) for the dynamic selection of functional split options in MEC-enabled RAN with EH capabilities, where each vSC is modeled as a DRL-based agent that takes decisions with coordination among other vSC agents. As opposed to tabular multi-agent RL, DDRL allows to coordinate the policies of the learning agents via local state information exchange among them without facing practically infeasible state-action tables.

The main contributions of this chapter may be summarized in the following items:

- *Coordinated DDRL Solutions*: We propose a multi-agent RL solution that can handle the prohibitively large state space in an efficient manner to optimally leverage flexible functional split options at the vSCs with the goals of minimizing the grid energy consumption and the amount of dropped traffic
- *Characterization of the Learning Algorithms*: We analyze the complexity and convergence properties of the proposed DDRL algorithms, and provide insights on the hyperparameter setup in both simulative training and policy validation. We also describe the spatio-temporal behavior of the selected DDRL policies.
- *Network Performance Evaluation*: We evaluate the performance (in terms of energy consumption and traffic drop rate) of the proposed DDRL solution with local state coordination, and compare them against the multi-agent FQL benchmark of Chapter 6 with system level coordination, as well as the offline bound of Chapter 4.
- *Cost Analysis*: We estimate the energy and cost savings that can be expected after 5 and 10 years of operation through DDRL and FQL controllers as compared to a system only relying on grid connection

In what follows, we describe briefly the scenario and system model considered in Section 7.2. The proposed DDRL solution is explained in Section 7.3. The numerical results and cost saving estimates are given in Section 7.4. Finally, we draw conclusions in Section 7.5.

7.2 Network Scenario and System Model

The network scenario, the system model as well as the optimization problem are explained in Chapter 6, Section 6.2. Recall that, the traffic loads at time slot t generated by the users in the coverage of the vSCs are denoted by $\boldsymbol{L}^t \triangleq [L_1^t, L_2^t, \ldots, L_N^t]$, where L_n^t is the traffic load at the n^{th} vSC. The energy harvested by the vSCs in slot t are denoted by $\boldsymbol{H}^t \triangleq [H_1^t, H_2^t, \ldots, H_N^t]$, while the battery states are denoted by $\boldsymbol{B}^t \triangleq [B_1^t, B_2^t, \ldots, B_N^t]$, where H_n^t and B_n^t are the harvested energy and the battery state of the n^{th} vSC, respectively. In addition, in order to capture the evolution of the traffic requests and energy arrivals, the hour of the day and the month are defined as h^t and m^t, respectively for time slot t.

The network wide sequential decision making problem is defined by a MDP as $\boldsymbol{X}_{t+1} = f(\boldsymbol{X}_t, \boldsymbol{A}^t, \boldsymbol{L}^t, \boldsymbol{H}^t)$, where $\boldsymbol{X}_t \triangleq [X_1^t, X_2^t, \ldots, X_N^t]$ are the states of the vSCs in slot t, $\boldsymbol{A}^t \triangleq [A_1^t, A_2^t, \ldots, A_N^t]$ are the control actions/modes of the vSCs and $(\boldsymbol{L}^t, \boldsymbol{H}^t)$ are the environmental random variables (i.e., traffic and EH stochastic processes). In particular, we define each state X_i^t, with $i = 1, ..., N$, as $X_i^t = (h^t, m^t, L_i^t, \boldsymbol{B^t})$. Hence, the state of the i^{th} vSC in slot t is represented by the battery levels of each vSCs $\boldsymbol{B^t}$, its traffic load L_i^t, the month of operation m^t and the hour of the day h^t. As stated in Section 6.2, the optimization objectives are to minimize the total weighted cost of grid energy consumption and system traffic drop rate over a finite time horizon. An off-line solution of this problem is proposed in Chapter 4 using DP with a priori knowledge of the environmental variables. The offline solution results are considered as system performance bounds. Moreover, Chapter 6 introduces an online solution based on tabular multi-agent RL, namely QL and FQL, that are used as a benchmark to the control methods proposed in this chapter.

In this chapter, we propose an on-line solution based on multi-agent DRL, without assuming the explicit knowledge of the system statistics governing the underlying random processes. In particular, we introduce distributed DRL agents in which neural networks are used as value approximation functions [110] to determine the optimal actions $\boldsymbol{A}^t$. Our proposal is based on distributed and coordinated decision making, i.e., each vSC takes its own action based on its state, which makes it scalable with the number of vSCs. In order to coordinate the decision making process, we rely on local state information exchange among vSCs; in particular, we assume that all the vSCs know the battery levels of all the other vSCs. The communication among vSCs may have additional overhead in terms of latency

and signaling. However, considering the time granularity of the decision making process (i.e., 1 hour), these effects are neglected in this study. The power consumption model is described in Section 2.6. Moreover, the EH and traffic demand profiles are described in Section 4.3.1.

7.3 Distributed Deep RL

In this section, we introduce the design of DDRL-based controllers where each vSC is modeled as a DRL agent taking decisions in coordination with other vSC agents. The details of the DDRL controllers including the states, actions and reward as well as the procedure followed by each DRL agent are included. We adopted a distributed approach where each vSC is modeled as a DRL-based agent that takes actions in coordination with other vSCs. This helps to avoid the exponentially increasing number of actions. For instance, if there are 3 possible operative modes/actions per vSC, a centralized approach will require the exploration of 3^N actions where N is the number of vSCs. This is a huge action space, e.g., the number of possible actions in centralized approach will reach $14,348,907$ for 15 vSCs. The distributed design ensures scalability while maintaining the complexity of the controllers to a reasonable level. In our DDRL implementation, coordination among the agents is enabled by the exchange of battery state information among agents as well as via the design of global reward where each vSC receives the same system level feedback.

7.3.1 DDRL control

In this section, the details of our DDRL based controller including the states, actions and reward definitions are given.

States: According to the system model defined in Section 7.2, the state of the i^{th} vSC at time slot t is defined as $X_i^t = (h^t, m^t, L_i^t, \boldsymbol{B^t})$, where h^t denotes the hour of the day, m^t denotes the month, L_i^t denotes the average traffic load experienced by the vSC and $\boldsymbol{B^t}$ denotes the vector of battery states of all the vSCs. The values of input the traffic load and the battery state variables, namely L_i^t and $\boldsymbol{B^t}$ are all normalized with respect to their maximum. On the other hand, for cyclic inputs h^t and m^t, sinusoidal transformation is applied [132]. Hence, the hour values ranging from $[0, 23]$ and months from $[0, 11]$ are transformed into sinusoidal values between $[-1, 1]$ and their cyclic properties are maintained. These state variables,

after normalization, are the input of the neural network that is used to estimate the Q values of all the actions at that state. Since the battery levels of all the vSCs are part of the states, the size of the input to the neural network of each vSC agent is dependent on the number of vSCs and is given as $3 + N$, where N is the number of vSC agents.

Actions: The set of possible actions are the possible operative modes of the vSCs, $\boldsymbol{A}^t$. The action set for the i^{th} vSC are switching off, PHY-RF split mode, or MAC-PHY split mode. Hence, the action set for the whole DDRL solution is a combination of the three operative modes of each vSC.

Reward: The reward function determines the immediate reward each DRL-agent, i.e., vSCs, acquires as a result of taking a specific action. The optimization goal is to minimize the power drained from the grid while reducing the traffic drop rate, as given by (6.4). Hence, the reward function can be formulated as:

$$r_t = 1 - (\omega_1 \cdot \mathrm{E_m}(\boldsymbol{A}^t) + \omega_2 \cdot \mathrm{D}(\boldsymbol{A}^t)) \tag{7.1}$$

where $\mathrm{E_m}(\boldsymbol{A}^t)$ and $\mathrm{D}(\boldsymbol{A}^t)$ are, respectively, the normalized grid energy consumption and the traffic drop rate, given the operative modes of the vSCs. In our DDRL implementation, each agent receives the same system level reward signal determined based on system drop rate and grid energy consumption. System level reward design is chosen to enable coordinated learning.

In order to improve the convergence behaviors of the DRL algorithm, we will use an experience buffer of capacity M, denoted by D in which we store previous experiences of the agents consisting of state, action, reward and next state tuples. We then randomly sample mini-batches from this experience buffer for training the neural networks of each DRL agents. This technique is known as *experience replay*. The neural networks used for estimating the Q-values are initialized with random weights, $\boldsymbol{\theta}$, and an ϵ-greedy policy maps the input states to actions; where actions are chosen randomly with probability of ϵ, i.e., *exploration*, otherwise an action with the highest Q-value is taken, i.e., *exploitation*. Each vSC agent applies a procedure shown in Algorithm 8.

Algorithm 8 DDRL based control

Initialize replay memory D_i of capacity M

Initialize action-value function, Q_i, with random weights $\boldsymbol{\theta}_i$

for each episode **do**:

Initialize $X_i^t = (h^t, m^t, L_i^t, \boldsymbol{B^t})$ - observation from the scenario

for each step, t, of episode **do**:

With probability ϵ select a random action A^t

Otherwise $A^t = \max_A Q_i(X_i^t, A; \boldsymbol{\theta}_i)$

Take action A^t, get reward r_t and observe next state X_i^{t+1}

Store transition (X_i^t, A_i^t, r_t, X_i^{t+1}) in D_i

Sample a random mini-batch of K transitions (X^j, A^j, r_j, X^{j+1}) from D_i

Set target value, $y_j = r_j + \gamma \max_{\hat{A}} Q_i(X^{j+1}, \hat{A}; \boldsymbol{\theta}_i)$

Perform a gradient descent step on $(y_j - Q_i(X^j, A^j; \boldsymbol{\theta}_i))^2$ with respect to $\boldsymbol{\theta}_i$

end for

end for

7.4 Numerical Results

We considered a simulation scenario as described in 4.5.1. Moreover, energy arrivals and aggregated downlink traffic have been generated according to the realistic models described in Section 4.3.1.

7.4.1 Training

In this section we analyze the behavior of the system when the training is performed off-line. In particular, we considered one year as an episode with time granularity of one hour, since it allows to achieve a correct dimensioning of the solar power system for cellular BSs, as shown in [130]. Hence, every hour the agents choose actions corresponding to one of the possible operative modes of the vSCs, with the goal of minimizing the weighted sum of grid energy consumption and traffic drop rate. As mentioned in 7.3.1, the size of the state set for each of the DRL agents is $N+3$, where N denotes the number of vSCs. An important step prior to feeding these inputs to the neural networks of each vSC agent are normalization and transformation, as described in Section 7.3.1. Through simulation trials and evaluation, we have selected a 3-layer dense neural network architecture with 256, 128 and 64 neurons, respectively. Moreover, *ReLu* activation function is used for hidden layers and *linear* activation is applied at the output layer. In addition,

Stochastic Gradient Descent (SGD) optimizer with momentum and learning rate decay is used for the training.

For each DRL agent, we have employed an experience replay buffer of size of $M = 2000$ and a mini-batch size of 32 are chosen for the training. The training procedure each DRL agent, i.e., each vSC, with experience replay is shown in Algorithm 8. Given that an episode lasts 1 year and actions are taken every hour, there are 8640 time steps within one episode of training. During training, there is random sampling of mini-batch of experiences from the replay memory to perform forward and back propagation steps in every hour prior to taking actions. After the mini-batch pass, the selection of the action is dependent on the ϵ-greedy policy. Each agent can select random action (exploration) or an action with maximum Q-value (exploitation) based on the exploration rate parameter ϵ used during training. In typical RL application, it is usually recommended to start with higher level of exploration and slowly decay exploration rates as the training progresses. The parameters used for the training of each DRL agent in our DDRL implementation are summarized in Table 7.1.

TABLE 7.1: Training Parameters.

Parameter	Value
learning rate (α)	0.01
initial exploration (ϵ)	0.9
discount factor (γ)	0.9
learning rate decay	0.01
exploration decay	0.9
optimizer	SGD
mini-batch size	32
number of layers	3

The cumulative rewards during the training procedure of vSC agents are shown in Fig. 7.1 for residential and office area traffic profiles. The training of each DRL agent is performed during 45 episodes. As it is shown in Fig. 7.1, the scenarios with higher number of vSCs require relatively longer training phase to reach stability and higher rewards. Moreover, the maximum cumulative reward reached at convergence decreases as the number of vSCs increases. This arise mainly due to the non-stationarity of the environment reflecting the conflicts among vSC policies. Finally, we notice that office profile results in a faster training phase and more stable behavior at convergence. In office scenario, traffic and energy harvesting phenomena are synchronous and facilitate an easier learning process compared to the residential scenario, in which the higher values of traffic demands appear during evening hours.

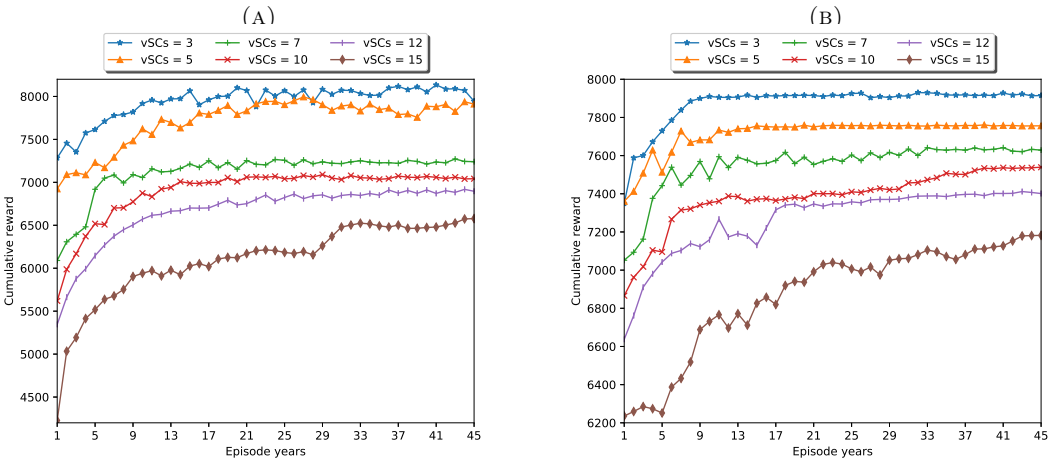


FIGURE 7.1: Cumulative reward vs number of vSCs: (a) Residential, and (b) Office traffic profiles

7.4.2 Policy Characteristics

In this section, we investigate the characteristics of the DDRL policies adopted by the vSCs. In DDRL, each agent chooses its policy in coordination with other agents with a common goal of minimizing the grid energy consumption and the dropped traffic rate simultaneously. As a result, even though the agents' policies can be different, the ultimate feedback or reward signal each agent acquires is the same, as provided in (7.1). This setup helps agents to jointly learn their own policies towards the direction of achieving a system wide goal. In addition, FQL policies are also shown here for comparison. The FQL controller design and training procedures are described in in Chapter 6. It is important to note that, the FQL controller state space consists of the vSC's battery level, energy arrival and vSC's and MBS's traffic load levels. Therefore, agents in FQL are coordinated only via the MBS's traffic load and they are not aware of others vSCs' battery conditions. For the case of 3 vSCs, both DDRL and FQL solutions are compared against an offline solution based on DP and described in Chapter 4. Due to the offline model computational complexity, we could not show this comparison for higher number of vSCs.

The average hourly winter and summer day policies for a scenario of 3 vSCs in residential area by the offline, FQL and DDRL controllers are shown in Fig.s 7.2 and 7.3, respectively. We have selected a day in December for winter, i.e., worst EH, and a day in August for summer, i.e., best EH. It can be observed that both FQL and DDRL are able to approximate the behavior of the optimal offline policy. A further investigation shows that DDRL better approximates the offline

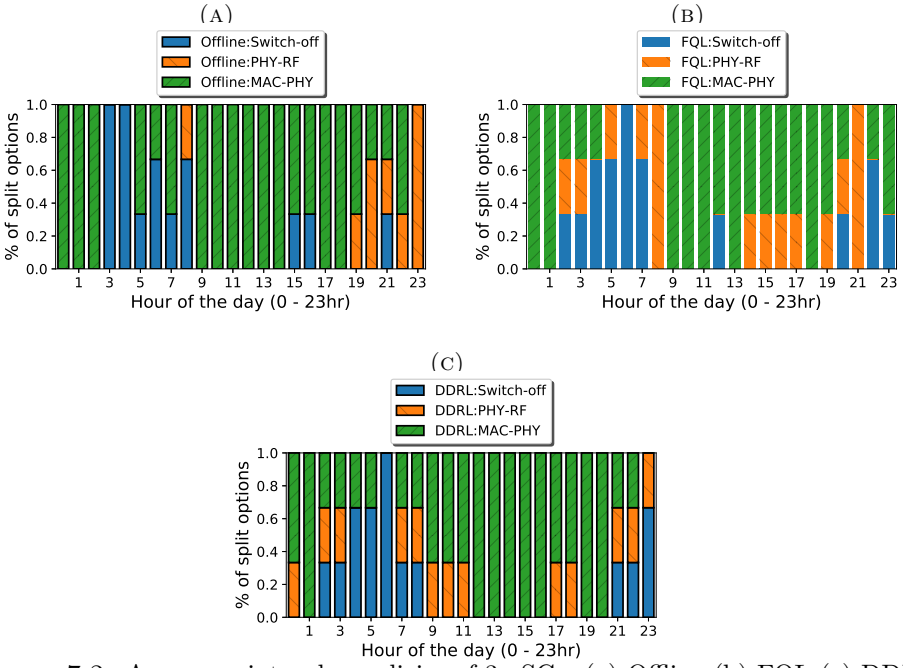


FIGURE 7.2: Average winter day policies of 3 vSCs: (a) Offline (b) FQL (c) DDRL

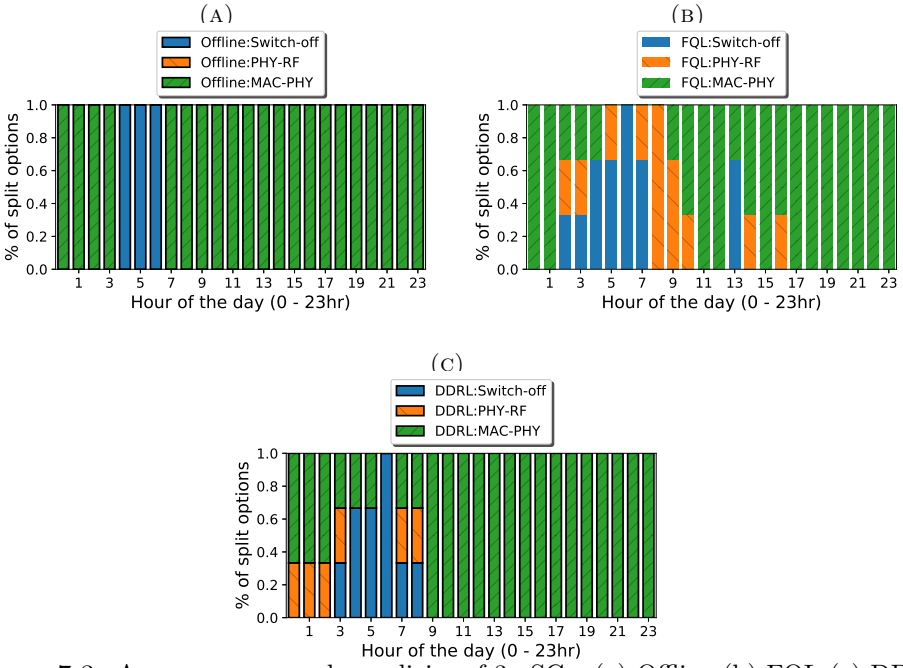
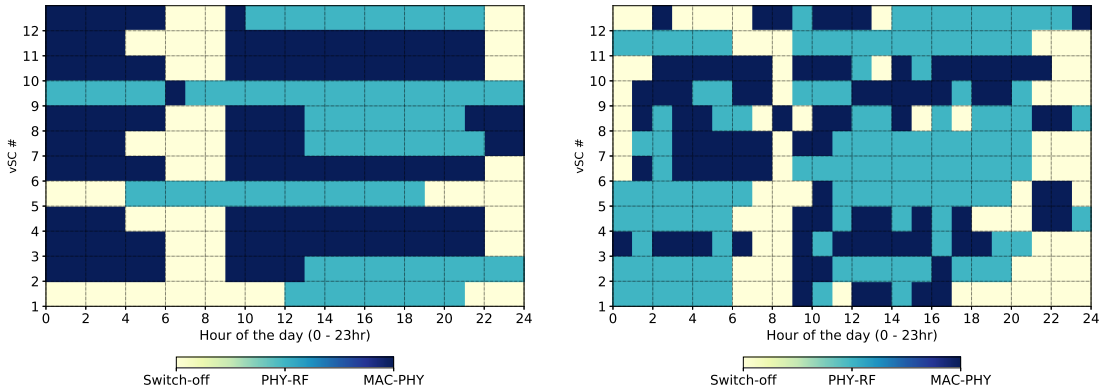


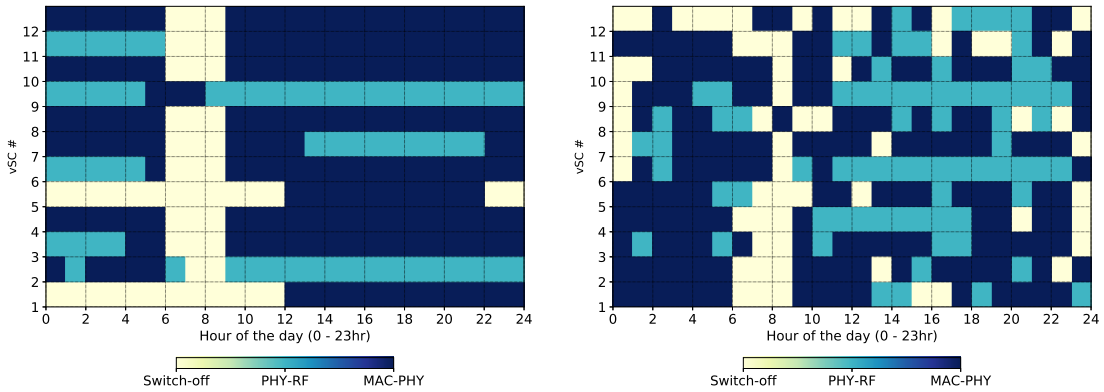
FIGURE 7.3: Average summer day policies of 3 vSCs: (a) Offline (b) FQL (c) DDRL

policy in terms of PHY-RF and MAC-PHY selection rates. On average, during December, MAC-PHY selection rates were 56%, 61% and 67% for the FQL, DDRL and offline policies whereas, during August, these rates increase to 65%, 78% and 87% respectively. It is interesting to observe from Fig. 7.3 that the offline policy shows no PHY-RF selection during August, thanks to the higher energy income, and indicating the benefit of processing most of the baseband functions locally



(A) (B)

FIGURE 7.4: Typical winter day policies with 12 vSCs: (a) DDRL (b) FQL



(A) (B)

FIGURE 7.5: Typical summer day policies with 12 vSCs: (a) DDRL (b) FQL

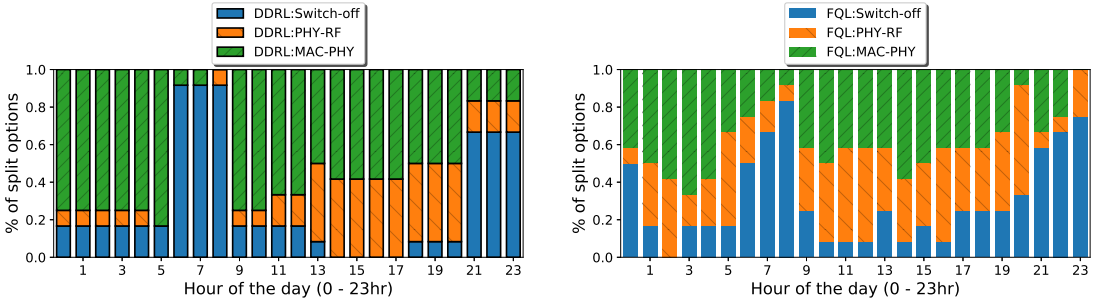
at vSCs. On the other hand, DDRL and FQL result in 8% and 17% PHY-RF selection, respectively, in August.

Table 7.2 shows the average winter and summer DDRL policy characteristics of the vSCs in residential area, averaged over the months of December and August. It can be seen that the winter policies are characterized by relatively higher switch-off rate than the corresponding summer policies. Moreover, the summer polices have higher MAC-PHY selection rate. This shows the adaptation to the months with higher energy income, i.e., by selecting MAC-PHY strategy, so that the vSCs execute most of the baseband processes locally, in turn saving more grid energy. On average, summer policies have 20.37% higher MAC-PHY and 18.82% lower switch-off rate than winter day policies. An example of daily policies by DDRL and FQL for a network setup of 12 vSCs in residential area are shown in Fig.s 7.4 and 7.5, respectively. The figures show that, in both DDRL and FQL, the vSCs learn relatively different policies with different PHY-RF, MAC-PHY and switch-off rates. The behaviors of DDRL and FQL policies differ in

TABLE 7.2: Average Policy Characteristics in Residential Area
(off: Switch-off, P-R: PHY-RF, M-P: MAC-PHY)

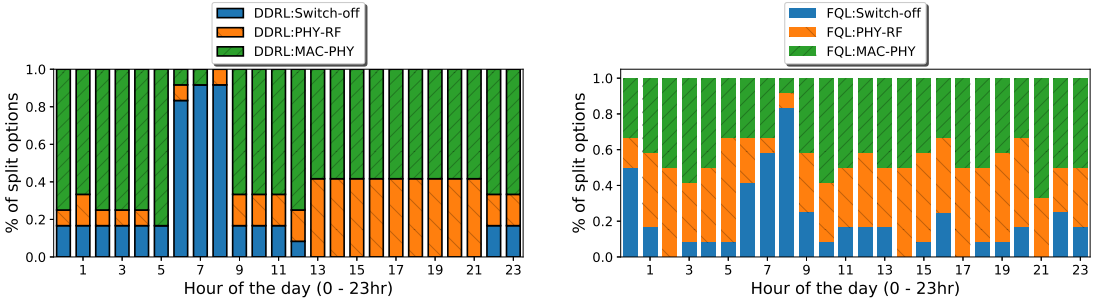
no. of vSCs	Winter			Summer		
	off(%)	P-R(%)	M-P(%)	off(%)	P-R(%)	M-P(%)
3	20.8	18.1	61.1	13.9	8.3	77.8
5	32.7	23.0	44.3	15.7	17.5	66.8
7	33.9	26.9	39.2	15.2	28.0	56.8
10	33.6	24.7	41.7	16.4	21.5	62.1
12	38.4	20.3	41.3	18.1	23.9	58.0
15	36.7	26.2	37.1	17.6	25.1	57.3

their respective selection rates of operative modes. For instance, for a scenario of 12 vSCs, the winter policies' MAC-PHY selection rates are 41% and 32% by DDRL and FQL, respectively. Whereas, during summer, the MAC-PHY rates increase to 58% and 49%, respectively, for DDRL and FQL. The FQL policy is characterized by a relatively higher PHY-RF selection rate. For a scenario of 12 vSCs, winter PHY-RF rates are 20% and 44%, while summer PHY-RF rates are 23% and 26%, by DDRL and FQL, respectively. This shows the relatively better flexibility of the DDRL policies according to the seasonal energy incomes. FQL results in less aggressive energy policies for the vSCs via selecting more PHY-RF modes, thereby relying more on the MBS for BB processing, which leads to higher grid energy consumption (as shown in Section 7.4.3). This is due to the fact that FQL controllers rely only on the MBS traffic load information for coordination. Therefore, each vSC does not have information about the battery state of the others, and hence, it is encouraged to remain switched on in order to reduce the load on MBS and select PHY-RF mode, which is the operative mode with lower energy consumption. Moreover, from the policy characteristics shown in Fig.s 7.4 and 7.5, the DDRL agents tend to have more stable behavior, i.e., vSCs prefer to stay in a single operative mode for longer time-slots than the FQL agents. This will help in limiting the overhead in the SDN/NFV framework for moving the virtual network functions as well as to reduce the frequency of operative mode changes in a deployed infrastructure. In order to assess this behavior, we have shown the average day policies of FQL and DDRL during winter and summer months in Fig.s 7.6 and 7.7, respectively, for a scenario of 12 vSCs. These behaviors are computed by averaging the vSC policies observed in each day during December and August. As shown in Fig.s 7.6 and 7.7, the DDRL results in policies that exhibit similar behavior within certain time-slots, whereas FQL policies are characterized by relatively higher variation within a day.



(A) (B)

FIGURE 7.6: Average winter day policies of 12 vSCs: (a) DDRL (b) FQL



(A) (B)

FIGURE 7.7: Average summer day policies of 12 vSCs: (a) DDRL (b) FQL

7.4.3 Network Performance

In this section, we evaluate the performance of the trained agents in terms of annual network grid energy consumption and average traffic drop rate. As a comparison benchmark, we have also analyzed the performance of FQL policies with similar simulation scenario. In addition, for the case of 3 vSCs, the network performance of both DDRL and FQL are evaluated against the offline bound studied in Chapter 4.

Table 7.3 shows the performance of DDRL and FQL polices compared with the offline bound of 3 vSCs. Both DDRL and FQL polices perform close to the offline bound in both residential and office scenarios with only $1.4-2.1\%$ and $4.8-5.3\%$ increase in annual grid energy consumption, respectively. DDRL performs closer to the offline bound and this can be confirmed by the cumulative reward plot shown in Fig. 7.8, where it can be seen that DDRL accumulates up to 97% of the rewards obtained by the offline policy whereas FQL accumulates up to 94%.

TABLE 7.3: Comparisons with respect to the off-line bound - 3 vSCs

Algorithm	Grid energy consumption (KWh)		Average drop rate (%)	
	Residential	Office	Residential	Office
Off-line	6775	6712	0.0	0.0
DDRL	6874 (+1.4%)	6857 (+2.1%)	0.0	0.0
FQL	7136 (+5.3%)	7037 (+4.8%)	0.0	0.0

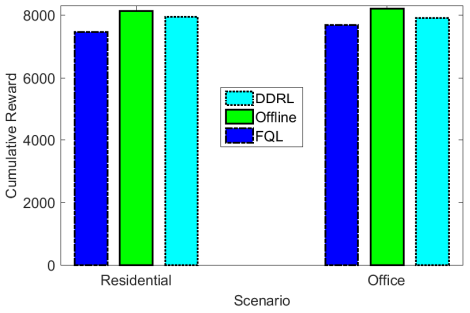
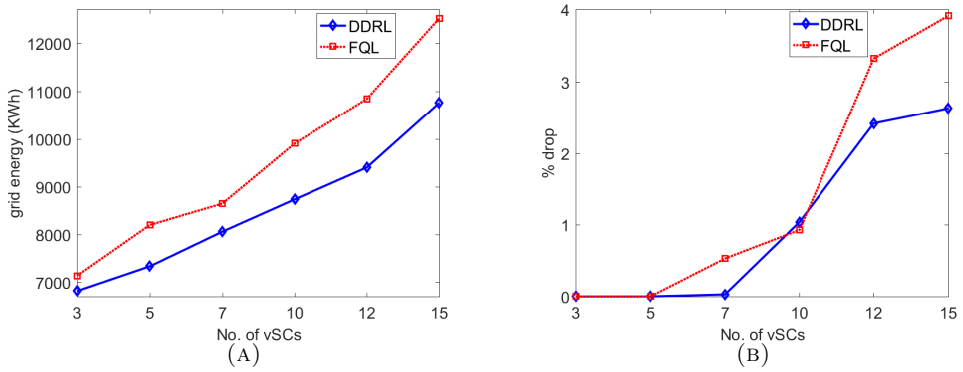


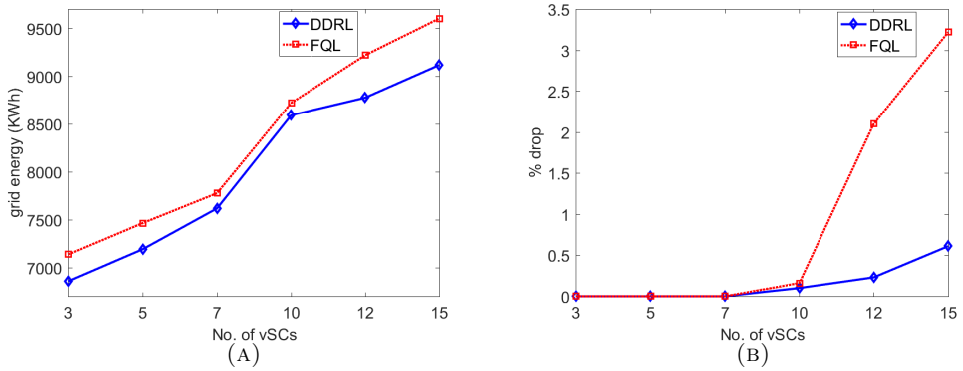
FIGURE 7.8: Cumulative reward comparison among the policies obtained by offline optimization, DDRL and FQL

The network grid energy consumption in one year of operation and the system drop rate comparison between DDRL and FQL controllers in residential and office areas for higher number of vSCs are shown in Fig. 7.9 and Fig. 7.10, respectively. The results show the better performance obtained by DDRL as compared to FQL controllers. More than 13% and 5% reduction in annual grid energy consumption is achieved with DDRL compared to FQL, in residential and office scenarios, respectively. Moreover, the DDRL control results in up to 2.6% and 1.3% less traffic drop rate than FQL in office and residential scenarios, respectively. The better performance by DDRL compared to FQL is also evident from the cumulative reward achieved by the correspondent controllers in the same simulation scenarios. The maximum cumulative rewards obtained by DDRL and FQL in residential and office area traffic profiles are shown in Fig. 7.11. It shows the relatively higher cumulative reward gained by DDRL, which translates to better network performance, i.e., lower grid energy consumption and system drop rate, as justified by Fig. 7.9 and 7.10. Moreover, the gap in cumulative reward is increasing with the number of vSCs, which implies that DDRL is able to reach better coordination among the agents



(A) (B)

FIGURE 7.9: Network performance comparison between DDRL and FQL in residential profile: (a) Grid Energy Consumption (KWh) (b) Average drop rate (%)



(A) (B)

FIGURE 7.10: Network performance comparison between DDRL and FQL in office profile: (a) Grid Energy Consumption (KWh) (b) Average drop rate (%)

(A) (B)

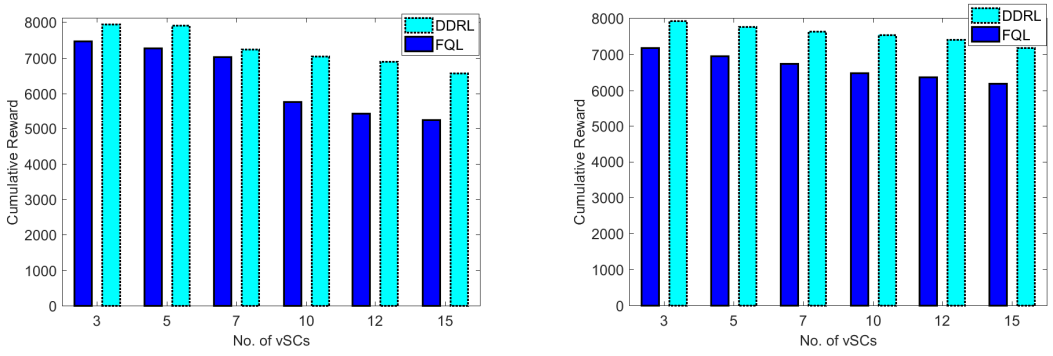


FIGURE 7.11: Maximum cumulative reward: (a) Residential and, (b) Office

7.4.4 Policy Validation

Here, we evaluate the behavior of the system in real deployment scenario after an offline training. In detail, we will validate the proposed DDRL based controllers using a new environment, which is characterized by an energy arrival and traffic

demand profiles which are different from the environment used for training. In this case we are using the pre-trained model with an exploration rate of 5%.

The validation of the policies along with the training environment policy evaluation for 3, 5, 7, 10, 12 and 15 vSCs for a year of operation are shown in Table 7.4. The validation results in Table 7.4 show that DDRL agents are able to adapt their behaviors to the new validation environment. This is confirmed by both grid energy and average drop rate performances that are very close to the corresponding policy evaluation results. On average, only 0.5% to 1.3% variation is observed in annual grid energy consumption results of policy evaluation and validation, in both residential and office profiles with no relevant changes in the system drop rate of the new environment. These results give an insight that using offline trained model and continuous exploration in the new environment is a viable approach for deployment.

TABLE 7.4: Policy validation results
(R-T: Residential Training, O-T: Office Training, R-V: Residential Validation, O-V: Office Validation)

No. of vSCs	Grid energy consumption (KWh)				Average drop rate (%)			
	R - T	O - T	R - V	O - V	R - T	O - T	R - V	O - V
3	6874	6857	6909	6891	0.00	0.00	0.00	0.00
5	7331	7191	7412	7287	0.00	0.00	0.00	0.00
7	8058	7617	8161	7677	0.03	0.00	0.10	0.10
10	8736	8591	8850	8422	1.04	0.10	1.61	0.90
12	9404	8775	9508	8694	2.42	0.23	2.38	0.93
15	10760	9116	10938	9162	2.63	0.61	2.71	1.03

7.4.5 Energy Savings and Cost Analysis

In Table 7.5, we compared the RL based controllers, i.e., FQL and DDRL, with a scenario in which all vSCs and MBS are supplied by grid power, refereed as Grid Connected (G-C). Both CAPEX and OPEX and the total costs of operation for 5 and 10 years duration are estimated. We consider a cost of 1.17\$/W for solar panels including installation, and 131\$/KWh for energy storage costs [133]. The energy purchasing price from the grid is set to 0.21\$/KWh [134].

As it is shown in Table 7.5, both FQL and DDRL controllers provide significant energy savings, reaching up to 54% and 61% for FQL and DDRL, respectively, compared to the G-C solution. In terms of cost savings, FQL provides 11% and 32% cost reduction during 5 and 10 years of operation, respectively. For DDRL

controller, the cost savings rise to 17% and 39% during 5 and 10 years of operation, respectively. Moreover, DDRL provides more energy and cost reduction than FQL controllers. These results are encouraging as they show that powering mobile networks with renewable energy sources with an intelligent control are not only environmental friendly, but also cost effective solutions.

TABLE 7.5: Energy savings and costs

vSCs	Algorithm	Energy (kW) consumption [1yr]	Costs ($) CAPEX	OPEX [1yr]	cost [5yrs]	cost [10yrs]
3	G-C	11334	0	2380	11900	23801
	FQL	7136	2541	1498	10033	17526
	DDRL	6814	2541	1430	9695	16850
5	G-C	14051	0	2950	14753	29507
	FQL	8199	4235	1721	12843	21452
	DDRL	7331	4235	1539	11932	19630
7	G-C	16769	0	3521	17607	35214
	FQL	8647	5929	1815	15008	24087
	DDRL	8057	5929	1691	14388	22848
10	G-C	20845	0	4377	21887	43774
	FQL	9910	8470	2081	18875	29281
	DDRL	8736	8470	1834	17642	26815
12	G-C	23562	0	4948	24740	49480
	FQL	10844	10164	2277	21550	32936
	DDRL	9404	10164	1974	20038	29912
15	G-C	27638	0	5803	29019	58039
	FQL	12510	12705	2627	25840	38976
	DDRL	10760	12705	2259	24003	35301

7.5 Conclusions

In this chapter, we have investigated the joint grid energy and traffic drop rate minimization problem and proposed multi-agent DRL solution. Coordinated learning among multiple agents is enabled via the exchange of the agents' battery state information. We have evaluated the network performance, in terms of the grid energy consumption and traffic drop rate for the proposed DDRL controller, and compared the results with an offline solution proposed in Chapter 4 and a tabular multi-agent RL solutions proposed in Chapter 6. The results have confirmed that limited coordination among the agents via the exchange of battery states achieve higher system level gains and cumulative rewards closer to the off-line bounds,

while requiring limited computational complexity. Extensive numerical results using traffic and EH data have confirmed that the proposed DDRL strategy ensures higher network performance, better adaptation to a changing environment, and higher cost savings with respect to the benchmark scheme.

Chapter 8

Conclusions and Future Works

The exponential growth of mobile traffic demand drives the deployment of multi-tier BSs as a means of enhancing capacity. Dense deployment of BSs is one of the main reasons behind the rapidly increasing electrical power consumption by mobile networks. This, in turn, leads to both environmental and economic concerns. Fueled by the growing mobile traffic demand, the energy consumption of ICT will reach about 51% of worldwide electricity consumption in 2030 [13]. Hence, sustainable design and operation of mobile networks is included in the road map towards future mobile networks, known as 5G and beyond.

In mobile network, RAN is the major energy consuming part [17]. To enhance RAN's efficiency, CRAN is proposed to enable centralized processing of BB functions and simplification of BSs to RRHs. However, the need for high capacity and very low latency fronthaul is a major drawback in CRAN. To relax the fronthaul requirements, part of the BB processes can be executed at the local BS sites while maintaining many of the centralization advantages of CRAN, i.e., flexible functional split between local BS sites and a central BBU pool. Moreover, NFV enables the execution of network functions on general purpose computing hardware as virtual network functions and SDN is emerging as a tool to realize the management and control of these functions. Flexible functional split with NFV/SDN opens a possibility that network functions of SBSs can be virtualized and placed at different sites of the network. These SBSs, known as vSCs, enable flexibility in resource allocation and management.

Moreover, to mitigate both environmental and cost issues, the research community have been studying the application of EH technology in mobile networks, including various control algorithms aiming at efficient utilization of the harvested energy.

Most of these research focus on online switch on/off control of SBSs in multi-tier deployments, known as HetNets. The recent softwarization trend in mobile networks including the introduction of MEC opens a new frontier in energy-aware processing and sharing of BB processing units according to flexible functional split options. This thesis provides novel contributions in energy-aware control of EH powered softwarized/MEC enabled RANs leveraging functional split options. Both offline control based on DP to determine the performance bound and online control methods based on multi-agent RL have been proposed and evaluated in realistic scenarios including residential and office area traffic profiles and different seasons of the year. The algorithms effectiveness have been analyzed with respect to the theoretical bound, naive controllers and with systems powered solely by the electric grid. In addition, energy and cost saving benefits of the proposed methods are estimated.

8.1 Summary of Results

The objective of the thesis is to study energy-aware control algorithms for mobile networks that are partly powered by EH and rely on MEC for flexible functional split between local sites (vSCs) and central cloud co-located with MBS, acting as a MEC server. Both DP and RL based algorithms are designed and their effectiveness is evaluated under different scenarios. The main parts of the thesis are:

- Chapters 1, 2 and 3: Introduction to the thesis objectives, the state-of-the art and the theoretical background.
- Chapter 4: Presents the study on performance bound using offline optimization for dynamic selection of functional split options.
- Chapters 5 and 6: Present online RL based controllers design and performance evaluation for both single and multi-vSCs scenarios.
- Chapter 7: Presents distributed deep RL based approach for dynamic control of functional split options in multi-vSCs scenario.

In what follows, the contributions and conclusions of the main technical parts of the thesis are summarized.

8.1.1 Performance Bound Study

The thesis contribution starts by studying the performance bound of dynamic functional splits control via offline optimization. DP is applied based on the assumption that the environment dynamics are known a-priori. In particular, three functional split options namely, PHY-RF, UpperPHY-LowerPHY and MAC-PHY, have been targeted. A grid energy consumption and traffic drop rate minimization problem is stated as DP problem and shortest path search algorithm is applied to determine the optimal functional split configurations. Simulation results show that dynamic control of functional split options serves the traffic demand with significant energy saving, and hence lower OPEX, with respect to static functional split policies. Therefore, the obtained performance bounds represent an encouraging starting point for the evaluation of more sophisticated online optimization techniques.

We have also presented an optimal control of functional split options with energy sharing to efficiently use the renewable energy coming from distributed sources and to facilitate the off-grid operation of the RAN. The proposed approach permits to move and execute some of the BB functions of the vSCs at the MBS site. Energy exceeding the battery capacity of the vSCs has been used for the MBS operations in order to further reduce the energy drained from the power grid. Software simulations demonstrate that an intelligent renewable energy management is essential to reduce the harvesting/storage dimensions with respect to naive approaches and leads to high energy and cost savings for a MNO.

8.1.2 Online RL Based Control

In these regard, the PhD thesis contributes on the study on online optimization algorithms for energy-aware control of functional split options in CRAN-like architecture. First, single agent RL based energy management is introduced via the application of QL and SARSA algorithms. Three operative modes namely, PHY-RF split mode, MAC-PHY split mode and switching off have been targeted. These online approaches are evaluated and compared with respect to an offline optimal policy. Simulation results prove that the two proposed methods perform close to the optimal bounds. In addition, it is observed that QL is observed to perform better than SARSA both in the amount of annual grid energy consumption and offloaded traffic.

In addition, after stating the limitations encountered by a single-agent RL approach, we have proposed multi-agent RL based energy engagement in a scenario where the computational processes of the vSCs powered solely by EH and batteries may be executed on a grid-connected MEC-server at the MBS. Distributed FQL and QL algorithms are tailored for our purposes. Coordination among the agents is achieved by broadcasting system level information (i.e., the traffic load at MBS) to each of the vSC agents. Finally, we have evaluated the network performance (in terms of energy consumption and traffic drop rate) by the multi-agent RL solutions with different levels of coordination and compared them against the offline performance bound and static solutions. Our results confirm that coordination via broadcasting may achieve higher system level gains and cumulative rewards closer to the offline bound. Moreover, our analysis permits to evaluate the benefits of continuous state/action representation for the learning algorithms in terms of faster convergence, higher cumulative reward and more adaptation to changing environments.

8.1.3 Distributed Deep RL Based Control

The last part of the thesis contribution focuses on the design of distributed deep RL control for EH vSCs that rely on MEC server at the MBS for full/partial BB processing. After stating the limitations of tabular multi-agent RL methods in dealing with continuous or large state/action spaces, a more robust and scalable control using neural networks as value function approximators, i.e. DQN, have been proposed. The proposed approach relies on a distributed architecture where each vSC is modeled as a DRL agent, i.e., DDRL, and the decisions of each vSC is coordinated via the exchange of local state information, i.e., the battery states of each vSC. The resulting controllers training, policy characteristics and performance are evaluated. Moreover, a comparison between multi-agent FQL and DDRL based approaches has been made. The results prove that DDRL controllers achieve higher cumulative rewards, more stability and higher network performance with respect to the benchmark multi-agent FQL controllers. In addition, energy and cost saving estimations are provided to give insights on long-term benefits of RL based controllers with respect to a system fully powered by the grid.

8.2 Future Works

The thesis objective is to investigate energy-aware control methods for CRAN-like mobile architectures that rely on EH for part of their operation. Both offline and online control methods were studied and the policies characteristics as well as their potential advantage in terms of energy saving are highlighted. Based on the conclusions drawn above, in what follows we summarize future research directions that are identified through the course of the PhD study.

8.2.1 Energy Sharing

Sharing of excess energy to the MBS is briefly studied in Section 4.6. The results were promising in ensuring efficient utilization of the harvested energy and further reducing the grid energy consumed by the network. However, energy cooperation methods can be extended to include sharing among vSCs as well as sharing between each vSC and MBS. This will ensure further gains in efficient utilization of energy and to make the whole system more self-reliant. Including energy cooperation entails higher complexity in solving the optimization problem both in offline and online approaches. Hence, it requires a proper trade-off between algorithm complexity and practical gains that can be harnessed via energy cooperation. In addition, with advances in smart grid, energy price will be dynamic depending on the cost of production and on the expected demand. As such, investigating the integration of mobile networks to the smart grid and evaluating decision-making solutions to find the best energy-purchasing policies for the BSs is a promising research direction. These study should take into account: (i) the current and forecast renewable energy inflow, (ii) the current and forecast traffic load, and (iii) the future evolution of the energy prices. In this scenario, BSs will act as both producers and consumers to the smart grids.

8.2.2 Alternative EH Systems

This thesis is based on a distributed EH system where each SBS are powered by harvesting hardware with rechargeable batteries. It is also important to investigate alternative EH system design approaches where EH and storage can be done at one site while enabling the allocation of energy to different parts of the network in energy efficient manner, i.e., centralized EH system. Evaluation of the relative

benefits of distributed and centralized EH systems including short-term and long-term energy and cost saving benefits is important. Moreover, it is interesting to study energy-aware control algorithms by considering more than one EH sources, e.g., solar and wind. In this regard, open-data initiatives for EH sources data, e.g., solar irradiance and wind data [135], and incorporating them into data-driven forecast models can be a promising research direction.

8.2.3 Control Methods for Ultra-dense Scenario

It is expected that the next generation of mobile networks will be ultra-dense, having as many SBSs as mobile users. Automated and energy-aware control of these dense networks is a very complex problem given the high number of system variables. Due to the complexity and the dynamism of this scenario, the design of online control solutions becomes crucial to reach close to the optimal performance without compromising on service quality. In this regard, centralized solutions may suffer from complexity due to large state/action spaces whereas multi-agent solutions will suffer from non-stationarity due to the learning of many agents simultaneously. Hence, it is important to investigate the application of novel DRL approaches in ultra-dense scenario. These methods can be policy based DRL methods that can learn stochastic policies without value estimations as well as deep recurrent Q-networks which can be effective to capture and generalize based on the evolution of the system dynamics. Moreover, in ultra-dense scenario, completely distributed control architecture may not be feasible approach due to the non-stationarity of the polices learned by the agents and require very strong coordination techniques. On the other hand, the centralized approach suffers from a huge-number of action space and high computational complexity. Hence, investigating hybrid control architectures for ultra-dense scenario is a promising future research direction. In addition, a study on the trade off between complexity of the control algorithms and practicability is of paramount importance to find feasible solutions. This involves estimating the energy and computational processing requirements of the control algorithms and incorporating them in the evaluation of the algorithms' performance.

8.2.4 Joint Energy and Service Aware Network Management

Dynamic and flexible service deployment is identified as one of the requirements for 5G and beyond networks. To this end, network slicing as well as MEC are identified as enablers to ensure fast service deployment times. Hence, the integration of energy-aware control methods with network slicing to ensure both intra-slice and inter-slice efficiency is essential. Different services will have diverse requirements. For instance, mobile broadband services require high data-rates and low latency whereas machine type communications require reliable, low latency and resilient communication links. In this regard, enabling both energy and service aware placement of virtual functions with in a network infrastructure is promising and requires further investigation. In addition, networks can involve multi-tier architecture having local, edge and cloud level deployments. As a result, it is important to consider more functional split options and the inclusion of service deployment times and service specific requirements in the optimization objective in order to design controllers that can place virtual network functions on local/edge/cloud resources in service-aware and energy efficient way. To this end, ML techniques for decomposition of a problem into a hierarchy of sub-problems are applicable for handling the complex optimization objectives. For instance, hierarchical RL [136] is a method to decompose a RL problem into a hierarchy of sub-tasks where higher level parent tasks invoke lower level tasks as if they were actions. This is advantageous to reduce computational complexity since reusable sub-tasks can be learned and provided independently. Moreover, it is important to consider the fronthaul constraints in the movement of virtual network functions among different network sites.

8.2.5 Edge Intelligence

Edge computing has been proposed to push cloud services to the proximity of end-users. The closer proximity between computing and information generation sources has many benefits, e.g., lower latency, energy efficiency and context awareness. On the other hand, we have recently seen many applications of AI in many fields, including telecommunications. These intelligent applications rely on data sources. Moreover, due to mobile computing and IoT, the network edge is becoming a huge source of data. Hence, pushing the AI potential to the source of data, i.e., to the edge, has given rise to an *edge intelligence* paradigm [137]. Edge intelligence aims at unlocking AI insights by using the resources at the edge

of the network. This paradigm opens new frontiers in enabling intelligent network applications in resource constrained environment. Therefore, it is promising to harness the potential of edge intelligence for network control applications in 5G and beyond systems. The multi-agent RL based control algorithms proposed in this thesis align with this emerging paradigm, with vSCs acting as the edge devices. Edge intelligence empowers edge devices to collect, generate communicate and analyze data in near real-time. To this end, edge intelligence gives a platform to extend the control methods proposed in the thesis via decentralized implementation of both energy and service-aware control algorithms by employing learning methods that are tailored for resource constraint environment (at the edge of the network), e.g., vSCs in ultra-dense deployments. This enables to design context-aware control solutions with near real-time granularity of decision making. Computational complexity is one of the challenges to extend the control solutions proposed in this thesis towards ultra-dense scenarios. Edge intelligence helps to alleviate this problem via leveraging decentralized learning techniques. For instance, federated learning [138] allows training a model on a server by aggregating only locally computed updates while leaving the data generated at the edge devices. This allows decentralized implementation, reduction in communication overhead as well as to preserve privacy. Due to the multi-tenancy ecosystem of future ultra-dense networks, it will be essential to maintain privacy of the generated data while ensuring optimization with in a network infrastructure shared by many tenants. Other techniques include gradient compression for reducing the communication overhead [139] and transfer learning [140] which is suitable for edge devices since it has greatly reduced resource demand.

Bibliography

[1] Small Cell Forum. Small cell virtualization functional splits and use cases. 2016. Available at: https://scf.io/en/documents/159_-_Small_cell_virtualization_functional_splits_and_use_cases.php.

[2] Aleksandar Damnjanovic, Juan Montojo, Yongbin Wei, Tingfang Ji, Tao Luo, Madhavan Vajapeyam, Taesang Yoo, Osok Song, and Durga Malladi. A survey on 3GPP heterogeneous networks. *IEEE Wireless communications*, 18(3), 2011.

[3] Ravi Tandon and Osvaldo Simeone. Harnessing cloud and edge synergies: toward an information theory of fog radio access networks. *IEEE Communications Magazine*, 54(8):44–50, August 2016. ISSN 0163-6804. doi: 10.1109/MCOM.2016.7537176.

[4] Aditya Gudipati, Daniel Perry, Li Erran Li, and Sachin Katti. SoftRAN: Software Defined Radio Access Network. In *Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking*, HotSDN '13, pages 25–30, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-2178-5. doi: 10.1145/2491185.2491207.

[5] Navid Nikaein, Eryk Schiller, Romain Favraud, Kostas Katsalis, Donatos Stavropoulos, Islam Alyafawi, Zhongliang Zhao, Torsten Braun, and Thanasis Korakis. Network store: Exploring slicing in future 5g networks. In *Proceedings of the 10th International Workshop on Mobility in the Evolving Internet Architecture*, pages 8–13. ACM, 2015.

[6] C. Desset, B. Debaillie, V. Giannini, A. Fehske, G. Auer, H. Holtkamp, W. Wajda, D. Sabella, F. Richter, M. J. Gonzalez, H. Klessig, I. Gdor, M. Olsson, M. A. Imran, A. Ambrosy, and O. Blume. Flexible power modeling of LTE base stations. In *2012 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 2858–2862, 2012.

[7] B. Debaillie, C. Desset, and F. Louagie. A flexible and future-proof power model for cellular base stations. In *2015 IEEE 81st Vehicular Technology Conference (VTC Spring)*, pages 1–7, May 2015. doi: 10.1109/VTCSpring.2015.7145603.

[8] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[9] Fengli Xu, Yong Li, Huandong Wang, Pengyu Zhang, and Depeng Jin. Understanding mobile traffic patterns of large scale cellular towers in urban environment. *IEEE/ACM transactions on networking (TON)*, 25(2):1147–1161, 2017.

[10] Cisco Systems. Fog Computing and the Internet of Things: Extend the Cloud to Where the Things Are. 2016. Available at: https://www.cisco.com/c/dam/en_us/solutions/trends/iot/docs/computing-overview.pdf.

[11] 5G Infrastructure PPP. 5G Vision-The 5G Infrastructure Public Private Partnership: the next generation of communication networks and services. 2015. Available at: https://5g-ppp.eu/wp-content/uploads/2015/02/5G-Vision-Brochure-v1.pdf.

[12] Mark P Mills. The cloud begins with coal. *Digital Power Group. Online at: http://www. tech-pundit. com/wp-content/uploads/2013/07/Cloud_Begins_With_Coal. pdf*, 2013.

[13] Anders Andrae and Tomas Edler. On global electricity usage of communication technology: trends to 2030. *Challenges*, 6(1):117–157, 2015.

[14] Albrecht Fehske, Gerhard Fettweis, Jens Malmodin, and Gergely Biczok. The global footprint of mobile communications: The ecological and economic perspective. *IEEE communications magazine*, 49(8):55–62, 2011.

[15] ETSI ES 203 228. Environmental engineering (ee);assessment of mobile network energy efficiency. April 2017.

[16] Study on Energy Efficiency Aspects of 3GPP Standards. 2017. 3GPP 21.866.

[17] Gunther Auer, Vito Giannini, Claude Desset, Istvan Godor, Per Skillermark, Magnus Olsson, Muhammad Ali Imran, Dario Sabella, Manuel J Gonzalez, Oliver Blume, et al. How much energy is needed to run a wireless network? *IEEE Wireless Communications*, 18(5), 2011.

[18] China Mobile. C-RAN, the road towards green ran. *White Paper*, 2012.

[19] Shao-Chou Hung, Hsiang Hsu, Shao-Yu Lien, and Kwang-Cheng Chen. Architecture harmonization between cloud radio access networks and fog networks. *IEEE Access*, 3:3019–3034, 2015.

[20] Osvaldo Simeone, Andreas Maeder, Mugen Peng, Onur Sahin, and Wei Yu. Cloud radio access network: Virtualizing wireless access for dense heterogeneous systems. *Journal of Communications and Networks*, 18(2):135–149, 2016.

[21] Mugen Peng, Yuan Li, Jiamo Jiang, Jian Li, and Chonggang Wang. Heterogeneous cloud radio access networks: A new perspective for enhancing spectral and energy efficiencies. *IEEE Wireless Communications*, 21(6):126–135, 2014.

[22] Andreas Maeder, Massissa Lalam, Antonio De Domenico, Emmanouil Pateromichelakis, Dirk Wubben, Jens Bartelt, Richard Fritzsche, and Peter Rost. Towards a flexible functional split for cloud-RAN networks. In *Networks and Communications (EuCNC), 2014 European Conference on*, pages 1–5. IEEE, 2014.

[23] 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*, 28(6):18–26, Nov 2014. ISSN 0890-8044. doi: 10.1109/MNET.2014.6963800.

[24] Yun Chao Hu, Milan Patel, Dario Sabella, Nurit Sprecher, and Valerie Young. Mobile edge computing A key technology towards 5G. *ETSI white paper*, 11(11):1–16, 2015.

[25] Giuseppe Piro, Marco Miozzo, Giuseppe Forte, Nicola Baldo, Luigi Alfredo Grieco, Gennaro Boggia, and Paolo Dini. HetNets powered by renewable energy sources: Sustainable next-generation cellular networks. *IEEE Internet Computing*, 17(1):32–39, 2013.

[26] *Sustainable Cellular Networks Harvesting Ambient Energy (SCAVENGE)*, 2016. http://www.scavenge.eu/about/.

[27] NGMN Alliance. Ngmn kpis and deployment scenarios for consideration for imt2020. *Final Deliverable*, 2016. Available at: https://www.ngmn.org.

[28] Chanwon Park and Jemin Lee. Mobile edge computing-enabled heterogeneous networks. *arXiv preprint: arXiv:1804.07756*, 2018.

[29] Adrian Lara, Anisha Kolasani, and Byrav Ramamurthy. Network innovation using openflow: A survey. *IEEE communications surveys & tutorials*, 16(1): 493–512, 2014.

[30] NEC. NFV C-RAN for Efficient RAN Resource Allocation.

[31] 3GPP. TS 38.801. E-U; Study on new radio access technology: Radio access architecture and interfaces v14, 2017.

[32] Nisha Panwar, Shantanu Sharma, and Awadhesh Kumar Singh. A survey on 5G: The next generation of mobile communication. *Physical Communication*, 18:64–84, 2016.

[33] Heinz Droste, Gerd Zimmermann, Makis Stamatelatos, Neiva Lindqvist, Omer Bulakci, Josef Eichinger, Venkatkumar Venkatasubramanian, Uwe Dotsch, and Hugo Tullberg. The METIS 5G architecture: A summary of METIS work on 5G architectures. In *Vehicular Technology Conference (VTC Spring), 2015 IEEE 81st*, pages 1–5. IEEE, 2015.

[34] Brian Hayes. Cloud computing. *Communications of the ACM*, 51(7):9–11, 2008.

[35] Qi Zhang, Lu Cheng, and Raouf Boutaba. Cloud computing: state-of-the-art and research challenges. *Journal of internet services and applications*, 1 (1):7–18, 2010.

[36] Peter Rost, Carlos J Bernardos, Antonio De Domenico, Marco Di Girolamo, Massinissa Lalam, Andreas Maeder, Dario Sabella, and Dirk Wübben. Cloud technologies for flexible 5g radio access networks. *IEEE Communications Magazine*, 52(5):68–76, 2014.

[37] Yong Li and Min Chen. Software-defined network function virtualization: A survey. *IEEE Access*, 3:2542–2553, 2015.

[38] Adrian Lara, Anisha Kolasani, and Byrav Ramamurthy. Network innovation using openflow: A survey. *IEEE communications surveys & tutorials*, 16(1): 493–512, 2014.

[39] Jianmin Zhang, Weiliang Xie, and Fengyi Yang. An architecture for 5G mobile network based on SDN and NFV. 2015.

[40] Meng-Lin Ku, Wei Li, Yan Chen, and KJ Ray Liu. Advances in energy harvesting communications: Past, present, and future challenges. *IEEE Communications Surveys & Tutorials*, 18(2):1384–1412, 2016.

[41] Akhil Gupta and Rakesh Kumar Jha. A survey of 5g network: Architecture and emerging technologies. *IEEE access*, 3:1206–1232, 2015.

[42] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[43] Shu-Ping Yeh, Shilpa Talwar, Geng Wu, Nageen Himayat, and Kerstin Johnsson. Capacity and coverage enhancement in heterogeneous networks. *IEEE Wireless Communications*, 18(3), 2011.

[44] David López-Pérez, Ming Ding, Holger Claussen, and Amir H Jafari. Towards 1 Gbps/UE in cellular systems: Understanding ultra-dense small cell deployments. *IEEE Communications Surveys & Tutorials*, 17(4):2078–2101, 2015.

[45] Mahmoud Kamel, Walaa Hamouda, and Amr Youssef. Ultra-dense networks: A survey. *IEEE Communications Surveys & Tutorials*, 18(4):2522–2545, 2016.

[46] Mugen Peng, Shi Yan, Kecheng Zhang, and Chonggang Wang. Fog-computing-based radio access networks: issues and challenges. *IEEE Network*, 30(4):46–53, 2016.

[47] Mamta Agiwal, Abhishek Roy, and Navrati Saxena. Next generation 5G wireless networks: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 18(3):1617–1655, 2016.

[48] Po-Han Huang, Hsu Kao, and Wanjiun Liao. Hierarchical cooperation in heterogeneous cloud radio access networks. In *Communications (ICC), 2016 IEEE International Conference on*, pages 1–6. IEEE, 2016.

[49] ETSI. Network functions virtualisation (NFV); use cases. 2013. Available at: https://www.etsi.org/deliver/etsi_gs/NFV/001_099/001/01.01.01_60/gs_NFV001v010101p.pdf.

[50] Mugen Peng and Kecheng Zhang. Recent advances in fog radio access networks: Performance analysis and radio resource allocation. *IEEE Access*, 4: 5003–5009, 2016.

[51] Subhadeep Sarkar, Subarna Chatterjee, and Sudip Misra. Assessment of the Suitability of Fog Computing in the Context of Internet of Things. *IEEE Transactions on Cloud Computing*, 2015.

[52] Subhadeep Sarkar and Sudip Misra. Theoretical modelling of fog computing: a green computing paradigm to support iot applications. *Iet Networks*, 5(2): 23–29, 2016.

[53] Fatemeh Jalali, Kerry Hinton, Robert Ayre, Tansu Alpcan, and Rodney S Tucker. Fog computing may help to save energy in cloud computing. *IEEE Journal on Selected Areas in Communications*, 34(5):1728–1739, 2016.

[54] OpenFog Consortium Architecture Working Group. OpenFog architecture overview. *White Paper*, 2016. Available at: http://www.microraindrop.com/eng/documents/OpenFog_Architecture_Overview_whitepaper.pdf.

[55] Yun Chao Hu, Milan Patel, Dario Sabella, Nurit Sprecher, and Valerie Young. Mobile edge computing - A key technology towards 5G. *ETSI white paper*, pages 1–16, 2015.

[56] T. Chen, H. Zhang, X. Chen, and O. Tirkkonen. SoftMobile: control evolution for future heterogeneous mobile networks. *IEEE Wireless Communications*, 21(6):70–78, December 2014. ISSN 1536-1284. doi: 10.1109/MWC.2014.7000974.

[57] Xenofon Foukas, Navid Nikaein, Mohamed M. Kassem, Mahesh K. Marina, and Kimon Kontovasilis. FlexRAN: A Flexible and Programmable Platform for Software-Defined Radio Access Networks. In *Proceedings of the 12th International on Conference on Emerging Networking EXperiments and Technologies*, CoNEXT '16, pages 427–441, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4292-6. doi: 10.1145/2999572.2999599.

[58] M. Y. Arslan, K. Sundaresan, and S. Rangarajan. Software-defined networking in cellular radio access networks: potential and challenges. *IEEE Communications Magazine*, 53(1):150–156, January 2015. ISSN 0163-6804. doi: 10.1109/MCOM.2015.7010528.

[59] J. Liu, T. Zhao, S. Zhou, Y. Cheng, and Z. Niu. CONCERT: a cloud-based architecture for next-generation cellular systems. *IEEE Wireless Communications*, 21(6):14–22, December 2014. ISSN 1536-1284. doi: 10.1109/MWC.2014.7000967.

[60] Peter Rost, Ignacio Berberana, Andreas Maeder, Henning Paul, Vinay Suryaprakash, Matthew Valenti, Dirk Wübben, Armin Dekorsy, and Gerhard Fettweis. Benefits and challenges of virtualization in 5G radio access networks. *IEEE Communications Magazine*, 53(12):75–82, 2015.

[61] M. Peng, Y. Li, Z. Zhao, and C. Wang. System architecture and key technologies for 5g heterogeneous cloud radio access networks. *IEEE Network*, 29 (2):6–14, March 2015. ISSN 0890-8044. doi: 10.1109/MNET.2015.7064897.

[62] A Al-Quzweeni, Taisir EH El-Gorashi, L Nonde, and Jaafar MH Elmirghani. Energy efficient network function virtualization in 5g networks. In *Transparent Optical Networks (ICTON), 2015 17th International Conference on*, pages 1–4. IEEE, 2015.

[63] Sherif Abdelwahab, Bechir Hamdaoui, Mohsen Guizani, and Taieb Znati. Network function virtualization in 5g. *IEEE Communications Magazine*, 54 (4):84–91, 2016.

[64] Patrick Kwadwo Agyapong, Mikio Iwamura, Dirk Staehle, Wolfgang Kiess, and Anass Benjebbour. Design considerations for a 5g network architecture. *IEEE Communications Magazine*, 52(11):65–75, 2014.

[65] Xin Li, Mohammed Samaka, H Anthony Chan, Deval Bhamare, Lav Gupta, Chengcheng Guo, and Raj Jain. Network slicing for 5G: Challenges and opportunities. *IEEE Internet Computing*, 21(5):20–27, 2017.

[66] Xuan Zhou, Rongpeng Li, Tao Chen, and Honggang Zhang. Network slicing as a service: enabling enterprises' own software-defined cellular networks. *IEEE Communications Magazine*, 54(7):146–153, 2016.

[67] Konstantinos Samdanis, Xavier Costa-Perez, and Vincenzo Sciancalepore. From network sharing to multi-tenancy: The 5g network slice broker. *IEEE Communications Magazine*, 54(7):32–39, 2016.

[68] Chengchao Liang and F Richard Yu. Wireless network virtualization: A survey, some research issues and challenges. *IEEE Communications Surveys & Tutorials*, 17(1):358–380, 2015.

[69] Matías Richart, Javier Baliosian, Joan Serrat, and Juan-Luis Gorricho. Resource slicing in virtual wireless networks: A survey. *IEEE Transactions on Network and Service Management*, 13(3):462–476, 2016.

[70] Subodh Paudel, Jagan Nath Shrestha, Fernando J Neto, Jorge AF Ferreira, and Muna Adhikari. Optimization of hybrid pv/wind power system for remote telecom station. In *Power and Energy Systems (ICPS), 2011 International Conference on*, pages 1–6. IEEE, 2011.

[71] Harpreet S Dhillon, Ying Li, Pavan Nuggehalli, Zhouyue Pi, and Jeffrey G Andrews. Fundamentals of heterogeneous cellular networks with energy harvesting. *IEEE Transactions on Wireless Communications*, 13(5):2782–2797, 2014.

[72] Tao Han and Nirwan Ansari. On optimizing green energy utilization for cellular networks with hybrid energy supplies. *IEEE Transactions on Wireless Communications*, 12(8):3872–3882, 2013.

[73] L Honorio. Efficiency in electricity generation. eurelectric: Union of the electric industry. *VGB Powertech, Brussels, Belgium*, 2003.

[74] Yuyi Mao, Yaming Luo, Jun Zhang, and Khaled B Letaief. Energy harvesting small cell networks: feasibility, deployment, and operation. *IEEE Communications Magazine*, 53(6):94–101, 2015.

[75] Nirwan Ansari and Tao Han. Freenet: Spectrum and energy harvesting wireless networks. *IEEE Network*, 30(1):66–71, 2016.

[76] Davide Zordan, Marco Miozzo, Paolo Dini, and Michele Rossi. When telecommunications networks meet energy grids: cellular networks with energy harvesting and trading capabilities. *IEEE Communications Magazine*, 53(6):117–123, 2015.

[77] Hussein Al Haj Hassan, Alexander Pelov, and Loutfi Nuaymi. Integrating cellular networks, smart grid, and renewable energy: Analysis, architecture, and challenges. *IEEE access*, 3:2755–2770, 2015.

[78] Deyu Zhang, Zhigang Chen, Lin X Cai, Haibo Zhou, Ju Ren, and Xuemin Shen. Resource allocation for green cloud radio access networks powered by renewable energy. In *Global Communications Conference (GLOBECOM), 2016 IEEE*, pages 1–6. IEEE, 2016.

[79] Tao Han and Nirwan Ansari. A traffic load balancing framework for software-defined radio access networks powered by hybrid energy sources. *IEEE/ACM Transactions on Networking (TON)*, 24(2):1038–1051, 2016.

[80] Jie Xu and Shaolei Ren. Online learning for offloading and autoscaling in renewable-powered mobile edge computing. In *Global Communications Conference (GLOBECOM), 2016 IEEE*, pages 1–6. IEEE, 2016.

[81] Nicola Piovesan and Paolo Dini. Optimal Direct Load Control of Renewable Powered Small Cells: A Shortest Path Approach. *Internet Technology Letters*, 2017.

[82] Jie Gong, John S Thompson, Sheng Zhou, and Zhisheng Niu. Base station sleeping and resource allocation in renewable energy powered cellular networks. *IEEE Transactions on Communications*, 62(11):3801–3813, 2014.

[83] Gilsoo Lee, Walid Saad, Mehdi Bennis, Abolfazl Mehbodniya, and Fumiyuki Adachi. Online ski rental for ON/OFF scheduling of energy harvesting base stations. *IEEE Transactions on Wireless Communications*, 16(5):2976–2990, 2017.

[84] Marco Miozzo. Energy sustainability of next generation cellular networks through learning techniques (Doctoral dissertation). 2018.

[85] Sheng Zhou, Jie Gong, and Zhisheng Niu. Sleep control for base stations powered by heterogeneous energy sources. In *2013 International Conference on ICT Convergence (ICTC)*, pages 666–670. IEEE, 2013.

[86] Marco Miozzo, Lorenza Giupponi, Michele Rossi, and Paolo Dini. Distributed Q-learning for energy harvesting heterogeneous networks. In *2015 IEEE International Conference on Communication Workshop (ICCW)*, pages 2006–2011. IEEE, 2015.

[87] Hocine Ameur, Moez Esseghir, and Lyes Khoukhi. Fully distributed approach for energy saving in heterogeneous networks. In *2016 8th IFIP international conference on New Technologies, Mobility and Security (NTMS)*, pages 1–6. IEEE, 2016.

[88] M. Miozzo and P. Dini. Layered learning radio resource management for energy harvesting small base stations. In *2018 IEEE 87th Vehicular Technology Conference (VTC Spring)*, pages 1–6, June 2018. doi: 10.1109/VTCSpring.2018.8417657.

[89] M. Miozzo, N. Piovesan, and P. Dini. Coordinated load control of renewable powered small base stations through layered learning. *IEEE Transactions on Green Communications and Networking*, pages 1–1, 2019. ISSN 2473-2400. doi: 10.1109/TGCN.2019.2938860.

[90] YE Junhong and Ying Jun Zhang. DRAG: Deep reinforcement learning based base station activation in heterogeneous networks. *IEEE Transactions on Mobile Computing*, 2019.

[91] M. Mendil, A. De Domenico, V. Heiries, R. Caire, and N. Hadjsaid. Battery-Aware Optimization of Green Small Cells: Sizing and Energy Management.

IEEE Transactions on Green Communications and Networking, 2(3):635–651, Sep. 2018. ISSN 2473-2400. doi: 10.1109/TGCN.2018.2829344.

[92] Marco Miozzo, Davide Zordan, Paolo Dini, and Michele Rossi. Solarstat: Modeling photovoltaic sources through stochastic markov processes. In *2014 IEEE International Energy Conference (ENERGYCON)*, pages 688–695. IEEE, 2014.

[93] Dimitra Politaki and Sara Alouf. Stochastic models for solar power. In *European Workshop on Performance Engineering*, pages 282–297. Springer, 2017.

[94] Joshua D Patrick, Jane L Harvill, and Clifford W Hansen. A semiparametric spatio-temporal model for solar irradiance data. *Renewable Energy*, 87:15–30, 2016.

[95] Le Xie, Yingzhong Gu, Xinxin Zhu, and Marc G Genton. Short-term spatio-temporal wind power forecast in robust look-ahead power system dispatch. *IEEE Transactions on Smart Grid*, 5(1):511–520, 2014.

[96] National Solar Radiation Data Base. Available at: http://rredc.nrel.gov/solar/old_data/nsrdb/.

[97] Sheng Xu and Shaowei Wang. Efficient algorithm for baseband unit pool planning in cloud radio access networks. In *Vehicular Technology Conference (VTC Spring), 2016 IEEE 83rd*, pages 1–5. IEEE, 2016.

[98] Rashid Mijumbi, Joan Serrat, Juan-Luis Gorricho, Javier Rubio-Loyola, and Steven Davy. Server placement and assignment in virtualized radio access networks. In *Network and Service Management (CNSM), 2015 11th International Conference on*, pages 398–401. IEEE, 2015.

[99] Jingchu Liu, Sheng Zhou, Jie Gong, Zhisheng Niu, and Shugong Xu. Graph-based framework for flexible baseband function splitting and placement in c-ran. In *Communications (ICC), 2015 IEEE International Conference on*, pages 1958–1963. IEEE, 2015.

[100] M Khan, RS Alhumaima, and HS Al-Raweshidy. Quality of service aware dynamic bbu-rrh mapping in cloud radio access network. In *Emerging Technologies (ICET), 2015 International Conference on*, pages 1–5. IEEE, 2015.

[101] Liumeng Wang and Sheng Zhou. Flexible functional split and power control for energy harvesting cloud radio access networks. *IEEE Transactions on Wireless Communications*, 2019.

[102] Michele Zorzi, Andrea Zanella, Alberto Testolin, Michele De Filippo De Grazia, and Marco Zorzi. Cognition-based networks: A new perspective on network optimization using learning and distributed intelligence. *IEEE Access*, 3:1512–1530, 2015.

[103] Francesco D Calabrese, Li Wang, Euhanna Ghadimi, Gunnar Peters, and Pablo Soldati. Learning radio resource management in 5g networks: Framework, opportunities and challenges. *arXiv:1611.10253*, 2016.

[104] Albert Mestres, Alberto Rodriguez-Natal, Josep Carner, Pere Barlet-Ros, Eduard Alarcón, Marc Solé, Victor Muntés-Mulero, David Meyer, Sharon Barkai, Mike J Hibbett, et al. Knowledge-defined networking. *ACM SIGCOMM Computer Communication Review*, 47(3):2–10, 2017.

[105] David D Clark, Craig Partridge, J Christopher Ramming, and John T Wroclawski. A knowledge plane for the internet. In *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 3–10. ACM, 2003.

[106] G Auer, O Blume, V Giannini, I Godor, MA Imran, Y Jading, E Katranaras, M Olsson, D Sabella, P Skillermark, et al. EARTH Deliverable D2. 3: Energy efficiency analysis of the reference systems, areas of improvements and target breakdown. *Project Deliverable D*, 2, 2013.

[107] Dimitri P Bertsekas. *Dynamic programming and optimal control*, volume 1. Athena scientific Belmont, MA, 1995.

[108] Lefteri H Tsoukalas and Robert E Uhrig. Fuzzy and neural approaches in engineering. 1997.

[109] Pierre Yves Glorennec. Fuzzy q-learning and dynamical fuzzy q-learning. In *Fuzzy Systems, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the Third IEEE Conference on*, pages 474–479. IEEE, 1994.

[110] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

[111] V. Franois-Lavet, P. Henderson, R. Islam, M. G. Bellemare, and J. Pineau. *An Introduction to Deep Reinforcement Learning*. 2018. ISBN null. Available at: https://ieeexplore.ieee.org/document/8585411.

[112] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[113] Ming Tan. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the tenth international conference on machine learning*, pages 330–337, 1993.

[114] Jeffery A Clouse. Learning from an automated training agent. In *Adaptation and learning in multiagent systems*. Citeseer, 1996.

[115] Bob Price and Craig Boutilier. Accelerating reinforcement learning through implicit imitation. *Journal of Artificial Intelligence Research*, 19:569–629, 2003.

[116] Lucian Buşoniu, Robert Babuška, and Bart De Schutter. Multi-agent reinforcement learning: An overview. In *Innovations in multi-agent systems and applications-1*, pages 183–221. Springer, 2010.

[117] Martin Lauer and Martin Riedmiller. An algorithm for distributed reinforcement learning in cooperative multi-agent systems. In *In Proceedings of the Seventeenth International Conference on Machine Learning*. Citeseer, 2000.

[118] Yoav Shoham, Rob Powers, and Trond Grenager. If multi-agent learning is the answer, what is the question? *Artificial intelligence*, 171(7):365–377, 2007.

[119] Michael Bowling and Manuela Veloso. Rational and convergent learning in stochastic games. In *International joint conference on artificial intelligence*, volume 17, pages 1021–1026. Lawrence Erlbaum Associates Ltd, 2001.

[120] Michael Bowling. Convergence and no-regret in multiagent learning. In *Advances in neural information processing systems*, pages 209–216, 2005.

[121] Reinaldo AC Bianchi, Murilo F Martins, Carlos HC Ribeiro, and Anna HR Costa. Heuristically-accelerated multiagent reinforcement learning. *IEEE transactions on cybernetics*, 44(2):252–265, 2013.

[122] Reinaldo AC Bianchi and Ramón López de Mantaras. Case-based multiagent reinforcement learning: Cases as heuristics for selection of actions. In *ECAI*, pages 355–360, 2010.

[123] H. D. Trinh, L. Giupponi, and P. Dini. Mobile traffic prediction from raw data using lstm networks. In *2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, pages 1827–1832, Sep. 2018. doi: 10.1109/PIMRC.2018.8581000.

[124] Languang Lu, Xuebing Han, Jianqiu Li, Jianfeng Hua, and Minggao Ouyang. A review on the key issues for lithium-ion battery management in electric vehicles. *Journal of power sources*, 226:272–288, 2013.

[125] Gunther Auer, Oliver Blume, Vito Giannini, Istvan Godor, M Imran, Ylva Jading, Efstathios Katranaras, Magnus Olsson, Dario Sabella, Per Skillermark, et al. D2. 3: Energy efficiency analysis of the reference systems, areas of improvements and target breakdown. *EARTH*, 20(10), 2010.

[126] Languang Lu, Xuebing Han, Jianqiu Li, Jianfeng Hua, and Minggao Ouyang. A review on the key issues for lithium-ion battery management in electric vehicles. *Journal of power sources*, 226:272–288, 2013.

[127] Christopher J.C.H. Watkins and Peter Dayan. Technical note: Q-learning. *Machine Learning*, 8(3):279–292, May 1992. ISSN 1573-0565. doi: 10.1023/A:1022676722315.

[128] L. Buoniu, R. Babu ŝka, and B. D. Schutter. A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 38(2):156–172, March 2008. ISSN 1094-6977. doi: 10.1109/TSMCC.2007.913919.

[129] Pontus Bergsten, Rainer Palm, and Dimiter Driankov. Observers for takagi-sugeno fuzzy systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 32(1):114–121, 2002.

[130] A. P. Couto da Silva, D. Renga, M. Meo, and M. Ajmone Marsan. The impact of quantization on the design of solar power systems for cellular base stations. *IEEE Transactions on Green Communications and Networking*, 2(1): 260–274, March 2018. ISSN 2473-2400. doi: 10.1109/TGCN.2017.2762402.

[131] D. A. Temesgene, N. Piovesan, M. Miozzo, and P. Dini. Optimal Placement of Baseband Functions for Energy Harvesting Virtual Small Cells. In *2018 IEEE 88th Vehicular Technology Conference (VTC-Fall)*, pages 1–6, Aug 2018. doi: 10.1109/VTCFall.2018.8690929.

[132] Anthony Adams and Peter Vamplew. Encoding and decoding cyclic data. *The South Pacific Journal of Natural and Applied Sciences*, 1998.

[133] Ran Fu, David J Feldman, and Robert M Margolis. Us solar photovoltaic system cost benchmark: Q1 2018. Technical report, National Renewable Energy Lab. (NREL), United States, 2018.

[134] *Average Energy Prices*, 2019 (accessed Oct., 2019). Available at: http://bit.ly/averagepricesLA.

[135] *EC: Strategic Energy Technologies Information System - EMHIRES datasets*. Available at: https://setis.ec.europa.eu/EMHIRES-datasets.

[136] T. P. Le, N. A. Vien, and T. Chung. A deep hierarchical reinforcement learning algorithm in partially observable markov decision processes. *IEEE Access*, 6:49089–49102, 2018. ISSN 2169-3536. doi: 10.1109/ACCESS.2018.2854283.

[137] Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo, and J. Zhang. Edge intelligence: Paving the last mile of artificial intelligence with edge computing. *Proceedings of the IEEE*, 107(8):1738–1762, Aug 2019. ISSN 1558-2256. doi: 10.1109/JPROC.2019.2918951.

[138] X. Wang, Y. Han, C. Wang, Q. Zhao, X. Chen, and M. Chen. In-edge ai: Intelligentizing mobile edge computing, caching and communication by federated learning. *IEEE Network*, 33(5):156–165, Sep. 2019. ISSN 1558-156X. doi: 10.1109/MNET.2019.1800286.

[139] Yujun Lin, Song Han, Huizi Mao, Yu Wang, and William J Dally. Deep gradient compression: Reducing the communication bandwidth for distributed training. *arXiv preprint arXiv:1712.01887*, 2017.

[140] Ragini Sharma, Saman Biookaghazadeh, Baoxin Li, and Ming Zhao. Are existing knowledge transfer techniques effective for deep learning with edge devices? In *2018 IEEE International Conference on Edge Computing (EDGE)*, pages 42–49. IEEE, 2018.