Machine Learning-Enabled Radio Resource Management for Next-Generation Wireless Networks

by

Medhat Elsayed

Thesis submitted to the University of Ottawa
in partial fulfillment of the requirement for the degree of

Doctor of Philosophy in Electrical and Computer Engineering

Abstract

A new era of wireless networks is evolving, thanks to the significant advances in communications and networking technologies. In parallel, wireless services are witnessing a tremendous change due to increasingly heterogeneous and stringent demands, whose quality of service requirements are expanding in several dimensions, putting pressure on mobile networks. Examples of those services are augmented and virtual reality, as well as self-driving cars. Furthermore, many physical systems are witnessing a dramatic shift into autonomy by enabling the devices of those systems to communicate and transfer control and data information among themselves. Examples of those systems are microgrids, vehicles, etc. As such, the mobile network indeed requires a revolutionary shift in the way radio resources are assigned to those services, i.e., Radio Resource Management (RRM).

In RRM, radio resources such as spectrum and power are assigned to users of the network according to various metrics such as throughput, latency, and reliability. Several methods have been adopted for RRM such as optimization-based methods, heuristics and so on. However, these methods are facing several challenges such as complexity, scalability, optimality, ability to learn dynamic environments. In particular, a common problem in conventional RRM methods is the failure to adapt to the changing situations. For example, optimization-based methods perform well under static network conditions, where an optimal solution is obtained for a snapshot of the network. This leads to higher complexity as the network is required to solve the optimization at every time slot. Machine learning constitutes a promising tool for RRM with the aim to address the conflicting objectives, i.e., Key Performance Indicator (KPI)s, complexity, scalability, etc.

In this thesis, we study the use of reinforcement learning and its derivatives for improving network KPIs. We highlight the advantages of each reinforcement learning method under the studied network scenarios. In addition, we highlight the gains and trade-offs among the proposed learning techniques as well as the baseline methods that rely on either optimization or heuristics. Finally, we present the challenges facing the application of reinforcement learning to wireless networks and propose some future directions and open problems toward an autonomous wireless network.

The contributions of this thesis can be summarized as follows. First, reinforcement learning methods, and in particular model-free Q-learning, experience large convergence time due to the large state-action space. As such, deep reinforcement learning was employed to improve generalization and speed up the convergence. Second, the design of the state and reward functions impact the performance of the wireless network. Despite the simplicity of this observation, it turns out to be a key one for designing autonomous wireless systems. In

particular, in order to facilitate autonomy, agents need to have the ability to learn/adjust their goals. In this thesis, we propose transfer in reinforcement learning to address this point, where knowledge is transferred between expert and learner agents with simple and complex tasks, respectively. As such, the learner agent aims to learn a more complex task using the knowledge transferred from an expert performing a simpler (partial) task.

To

my beloved parents,
and my lovely wife, Menna Eldaly

Acknowledgements

I would like to express my deepest gratitude to my supervisor Professor Melike Erol-Kantarci for her outstanding guidance, support, and motivation. Her insight and knowledge into the subject steered me through this research. Without her guidance and support, this dissertation would not have been possible.

I would like to express my sincere appreciation to Professor Halim Yanikomeroglu for his insightful comments during our research collaboration, which is an essential contribution to this thesis. His guidance and questions opened my eyes to new research directions.

I would like also to thank Professor Burak Kantarci and Professor Karin Hinzer for their valuable guidance and comments during our research collaboration. Many thanks to the committee members for providing an extremely valuable feedback.

Last but not least, I would like to thank all NETCORE lab members. It is a privilege to work with an excellent team.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

Acronym	Meaning
3GPP	Third Generation Partnership Program
5G	Fifth-Generation
ABSF	Almost-Blank SubFrame
AI	Artificial Intelligence
AoD	Angle of Departure
AR	Augmented Reality
AWGN	Additive White Gaussian Noise
BSDC	Best SINR association with DBSCAN
CBR	Constant Bit Rate
CQI	Channel Quality Indicator
CRM	Community Resilience Microgrid
CSI	Channel State Information
CUD	Critical User Device
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
DID	Data Intensive Device
DIRA	Distributed Iterative Resource Allocation
DM-DQN	Delay Minimizing Deep Q-Network
DMDQ	Delay Minimizing Deep Q-learning
DMQ	Delay-Minimization Q-learning
DQL	Deep Q-learning
DQLD	Deep Q-learning with DBSCAN
eCCDF	empirical Complementary Cumulative Distribution Function
eMBB	enhanced Mobile Broad-Band
eNB	evolved NodeB
ESN	Echo-State Network
FDD	Frequency Division Duplex

FR-2	Frequency Range 2
gNB	5G NodeB
HARQ	Hybrid Automatic Repeat Request
HetNet	Heterogeneous Network
IB-ICI	Inter-Beam Inter-Cell Interference
IoT	Internet-of-Things
ITU	International Telecommunications Union
KPI	Key Performance Indicator
KPPF	K-means clustering with PPF
LoS	Line of Sight
LR-Q	Latency-Reliability using Q-learning
LRT-Q	Latency-Reliability-Throughput Improvement using Q-Learning
LSTM	Long Short Term Memory
MAC	Medium Access Control
MCD	Mission-Critical Device
MCS	Modulation and Coding Scheme
MDP	Markov Decision Process
MGD	Micro-Grid Device
MIMO	Multiple Input Multiple Output
MISO	Multiple Input Single Output
ML	Machine Learning
MLP	multi-layer perceptron
mMTC	massive Machine Type Communications
NLoS	Non-LoS
NOMA	Non-Orthogonal Multiple Access
OFDM	Orthogonal Frequency Division Multiplexing
OFDMA	Orthogonal Frequency Division Multiple Access
PCP	Poisson Cluster Process
PDR	Packet Drop Rate
PF	Proportional Fairness
PLR	Packet Loss Rate
PPF	Priority-based Proportional Fairness
QAM	Quadrature Amplitude Modulation
QCI	Quality Class Indicator
QoS	Quality of Service
QPSK	Quadrature Phase Shift Keying
RAN	Radio Access Network
RB	Resource Block

RBG	Resource Block Group
RLC	Radio Link Control
RNN	Recurrent Neural Network
RR	Round Robin
RRM	Radio Resource Management
RSRP	Reference Signal Received Power
SARSA	State-Action-Reward-State-Action
SBS	Small-cell Base Station
SIC	Successive Interference Cancellation
SINR	Signal-to-Interference-plus-Noise Ratio
TBS	Transport Block Size
TD	Temporal Difference
TDD	Time Division Duplex
TMQ	Throughput-Maximizing Q-Learning
TQL	Transfer Q-learning
TTI	Transmission Time Interval
TvITM	Transfer via Inter-Task Mapping
UE	User Equipment
UMa	Urban Macro
UNB	Users of eNB
UPA	Uniform Power Allocation
uRLLC	ultra Reliable and Low Latency Communication
USRP	Universal Software Radio Peripheral
VR	Virtual Reality

List of Symbols

$\aleph$	Interval to update the target neural network with the main neural network
α	Learning rate of Q-learning
β	Control parameter
$\boldsymbol{v}$	Steering vector
$\boldsymbol{w}$	Beamforming vector
$\boldsymbol{x}$	Input feature vector of a neural network
$\boldsymbol{y}$	Output vector of a neural network
δ	User-cell association indicator
F	Number of antenna elements
ϵ	Exploration probability
η	Pathloss exponent
Γ	SINR
γ	Discount factor
Γ^{QoS}	SINR QoS requirement
Γ_{th}	Threshold SINR
μ	Service rate
Ω	Training interval of a neural network

ω	Bandwidth
$\overline{C}$	Average transmission rate
ϕ_a	Action mapping function of TvITM method
ϕ_q	Q-value mapping function of TvITM method
ϕ_s	State mapping function of TvITM method
π	Stochastic policy of agent
ψ	Activation function of a neuron
τ	Iteration step of a reinforcement learning agent
$\mathscr{A}$	Action space
$\mathscr{S}$	State space
θ	Angle of departure
$\varkappa$	Antenna spacing of the antenna array
ϱ	Channel complex gain
ξ	Packet size
ζ	Number of paths of a mm-wave channel
A	Random variable of the action
a	Action
B	Number of beams
b	Beam index
C	Transmission rate
D	Delay
d	Euclidean distance between a user and its base station
D^{harq}	HARQ retransmission delay

D^{QoS}	Delay QoS requirement
D^{q}	Queuing delay
D^{tr}	Transmission delay
D_0	Delay target
E	Number of experts
e	Expert index
G	Number of gNBs
g	gNB index
h	Channel coefficient
I	Interference
i	User index
J	Number of base stations
j	Base station index (eNB, gNB, or small cell)
K	Number of RBGs
k	RBG index
L	Number of learners
l	Learner index
M	Number of SBSs
m	SBS index
M_A	Moment generating function
N	Number of users
N_0	Noise power spectral density
o	Output function of a neuron

p	Transmission power
P_{max}	Maximum transmit power
PL_{dB}	Path loss (in dB)
q	Quality value function
R	Random variable of the reward
r	Reward function
S	Random variable of the state
s	State
s'	Next state
T	Simulation time
t	Time step (TTI)
x	RBG allocation indicator
z	Pre-activation function of a neuron
$\mathfrak{B}$	Set of beams
$\mathfrak{E}$	Set of experts
$\mathfrak{G}$	Set of gNBs
$\mathfrak{J}$	Set of base stations
$\mathfrak{K}$	Set of RBGs
$\mathfrak{L}$	Set of learners
$\mathfrak{M}$	Set of SBSs
$\mathfrak{N}$	Set of users
$\mathfrak{q}$	CQI value

Chapter 1

Introduction

1.1 Motivation

Future wireless networks are expected to support a multitude of services. According to the International Telecommunications Union (ITU), Fifth-Generation (5G) network services can be classified into three service types: enhanced Mobile Broad-Band (eMBB), ultra Reliable and Low Latency Communication (uRLLC), and massive Machine Type Communications (mMTC) [2]. Heterogeneous devices of different quality of service demands will require intelligent and flexible allocation of network resources in response to network dynamics. For instance, a highly reliable and low-latency network is needed to enable rapid transfer of messages between connected autonomous vehicles. At the same time, the same physical infrastructure is expected to serve users with high-quality video demand or even mobile augmented/virtual reality entertainment applications. Next-generation wireless networks, i.e., 5G and the upcoming 6G, are expected to accommodate diverse use cases. In particular, the heterogeneous traffic coming from mobile, vehicular, smart grid and tactile domains, calls for efficient utilization of network resources to maintain quality of service demands of each application. In addition, resource efficiency, reliability, and robustness are becoming more stringent for 5G and beyond networks.

Furthermore, current wireless networks employ mathematical models to represent the wireless system and evaluate its performance, which do not capture realistic situations accurately. In addition, optimization of wireless resources poses significant challenges in computation time, complexity, and energy consumption. Combined with the increasing and heterogeneous traffic, the mathematical models are likely to fail in capturing the stringent QoS requirements of next-generation wireless networks [3].

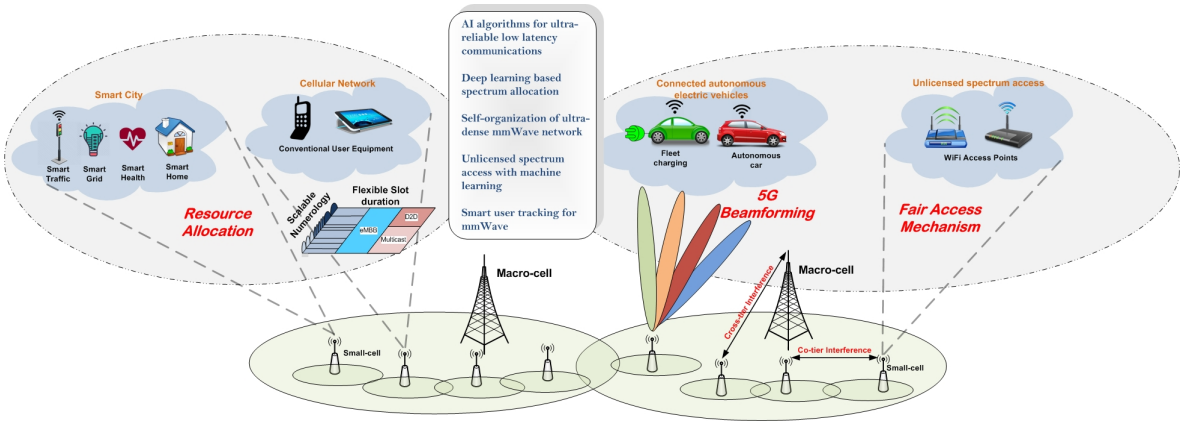

Figure 1.1: ML-enabled future wireless network and services.

To meet this, next-generation networks must incorporate a paradigm shift in network resource optimization, in which efficient and intelligent resource management techniques have to be employed. Machine Learning (ML) stands as a promising tool to intelligently manage network's resources such that network efficiency, reliability, robustness goals are achieved and quality of service demands are satisfied. In particular, ML presents a key advantage in managing network resources in a flexible and agile operation. This provides the network with more autonomy that reduces the computational and time expenses of manual configuration and maintenance. On the same line, ML can be adopted for real-time analysis and dynamic control that both provides flexibility and reduces human intervention. Indeed, the opportunities that arise from learning environment's parameters under varying behavior of the wireless channel, positions ML-enabled 5G and 6G superior to preceding generations of wireless networks. Fig. 1.1 highlights some wireless problems and applications that can leverage the potential of ML. Despite these opportunities there are certain challenges that need to be addressed such as convergence time, computational complexity, adaptability to network dynamics, etc.

1.2 Thesis Contributions

This thesis aims to investigate the potential of ML, and more specifically reinforcement learning methods, as developed for the task of RRM in current and future generations of wireless networks. In particular, we demonstrate the effectiveness, in addition to studying several challenges of applying distributed reinforcement learning methods for RRM. The proposed methods demonstrate a very good ability in achieving, as well as balancing,

several network KPIs such as throughput, latency, reliability, fairness, and load balancing. Furthermore, we address several challenges of reinforcement learning and its variants such as space and time complexity, design of state and reward functions, performance with respect to optimization-based schemes, and knowledge transfer. The following lines detail and highlight our contributions and challenges of applying reinforcement learning to the wireless domain.

- **Improving KPIs of Wireless Networks:** Improving the KPIs of wireless networks, such as throughput, latency, reliability, fairness, and load balancing, is the main goal of using ML methods. In this thesis, we used several techniques to address several objectives and to balance the conflicting trade-offs. In particular, reinforcement learning is used in sections 3.2 and 4.4 to improve aggregate throughput in both LTE and 5G networks, respectively. Furthermore, in sections 3.3, 3.4, and 3.5, we employ reinforcement and deep reinforcement learning for improving latency under different network scenarios. Finally, multi-objective RRM has been addressed in sections 4.2, 4.3, and 4.5, where a network serving uRLLC and eMBB users is considered. The aim is to address the trade-off stemming from the coexistence of uRLLC and eMBB users and satisfy latency, reliability, and throughput requirements.

- **Convergence and Complexity:** Despite the performance gain associated with reinforcement learning, it experiences a well-known convergence problem. In particular, in each iteration of Q-learning, a Q-value of state-action pair is updated. Although this leads to better discrimination, it also incurs large convergence time due to the need to visit as many state-action pairs through exploration. A straightforward solution to this problem is to use a function approximator for the calculation of the Q-values. As such, deep reinforcement learning has been a natural choice in our work to speedup reinforcement learning's convergence. In sections 3.4 and 3.5, we used deep Q-learning to improve latency of mission-critical services and tactile internet, respectively. Furthermore, deep Q-learning is used in section 4.5 to perform multi-objective resource allocation in mm-wave network, where latency and reliability of uRLLC were addressed. We show that deep reinforcement learning outperforms the convergence of tabular methods, in addition to achieving better policy that improves the targeted KPIs.

- **Design of Reinforcement Learning:** The design of state and reward functions of the reinforcement learning method have a direct impact on the performance results. In contrast to single-objective reinforcement learning, we propose a multi-objective reward function in sections 4.2 and 4.3 for RRM with the aim to improve throughput,

latency, and reliability in 5G network that covers uRLLC and eMBB users. Furthermore, section 4.5 addresses the improvement of Quality of Service (QoS) requirements of uRLLC and eMBB users using Long Short Term Memory (LSTM)-based deep Q-learning, in which the design of the reward function is crafted to capture the QoS of the users. As such, the sections demonstrate the importance of designing state and reward functions that capture the desired goal of the system. However, manual design constitutes a challenge, and a more flexible solution is needed to facilitate autonomy in the network. We address this using transfer in reinforcement learning as presented in section 5 and explained in the next item.

- **Knowledge Transfer - Autonomous Systems:** Most ML methods are inspired by human behaviour, in which an ambitious goal is to have an agent that is not only able to perform well under a predefined goal, but is also able to set goals for itself. Despite the success of reinforcement learning methods, yet the manual design of the reward function constitutes an impairment in the way toward a fully autonomous system. As a first step, we investigate the performance of transfer learning in the domain of reinforcement learning in chapter 5. In particular, we study how to transfer knowledge between an expert agent, that performs a simple task, to a learner agent that performs a more complex but related task. The task of the agent is user-cell association in mm-wave network, whereas the task of the learner is joint user-cell association and selection of the number of beams in a beamforming scenario. The work shows the advantages of using transfer in reinforcement learning under certain network scenarios in comparison to the Q-learning method.

1.3 Publications

[P01] M. Elsayed, Melike Erol-Kantarci, "System and Method for Joint Power and Resource Allocation Using Reinforcement Learning", US Patent Filed on 30 October 2020.

[J05] K. Shimotakahara, M. Elsayed, M. Erol-Kantarci and K. Hinzer, "Mobile Communications-Enabled Smart Grid Cosimulator System Design," *IEEE Systems Journal*, vol. 15, no. 2, pp. 2677-2686, June 2021.

[J04] M. Elsayed, M. Erol-Kantarci and H. Yanikomeroglu, "Transfer Reinforcement Learning for 5G New Radio mmWave Networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 5, pp. 2838-2849, May 2021.

[J03] M. Elsayed, M. Erol-Kantarci, B. Kantarci, L. Wu and J. Li, "Low-Latency Communications for Community Resilience Microgrids: A Reinforcement Learning Approach,"

IEEE Transactions on Smart Grid, vol. 11, no. 2, pp. 1091-1099, March 2020.

[J02] M. Elsayed and M. Erol-Kantarci, "AI-Enabled Future Wireless Networks: Challenges, Opportunities, and Open Issues," *IEEE Vehicular Technology Magazine*, vol. 14, no. 3, pp. 70-77, September 2019.

[J01] K. Shimotakahara, **M. Elsayed**, K. Hinzer and M. Erol-Kantarci, "High-Reliability Multi-Agent Q-Learning-Based Scheduling for D2D Microgrid Communications," *IEEE Access*, vol. 7, pp. 74412-74421, June 2019.

[C08] M. Elsayed and M. Erol-Kantarci, "Radio Resource and Beam Management in 5G mmWave Using Clustering and Deep Reinforcement Learning," in *IEEE Global Communications Conference*, pp. 1-6, December 2020.

[C07] M. Elsayed, K. Shimotakahara and M. Erol-Kantarci, "Machine Learning-based Inter-Beam Inter-Cell Interference Mitigation in mmWave," in *IEEE International Conference on Communications (ICC)*, pp. 1-6, June 2020.

[C06] K. Shimotakahara, **M. Elsayed**, K. Hinzer and M. Erol-Kantarci, "Integrated Power and Device-to-Device (D2D) Communications Simulator for Future Power Systems," in *IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pp. 1-5, November 2019.

[C05] M. Elsayed and M. Erol-Kantarci, "Reinforcement Learning-Based Joint Power and Resource Allocation for URLLC in 5G," in *IEEE Global Communications Conference (GLOBECOM)*, pp. 1-6, December 2019.

[C04] M. Elsayed and M. Erol-Kantarci, "AI-Enabled Radio Resource Allocation in 5G for URLLC and eMBB Users," in *IEEE 5G World Forum (5GWF)*, pp. 590-595, November 2019.

[C03] M. Elsayed and M. Erol-Kantarci, "Deep Reinforcement Learning for Reducing Latency in Mission Critical Services," in *IEEE Global Communications Conference (GLOBECOM)*, pp. 1-6, February 2018.

[C02] M. Elsayed and M. Erol-Kantarci, "Deep Q-Learning for Low-Latency Tactile Applications: Microgrid Communications," in *IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*, pp. 1-6, December 2018.

[C01] M. Elsayed and M. Erol-Kantarci, "Learning-Based Resource Allocation for Data-Intensive and Immersive Tactile Applications," in *IEEE 5G World Forum (5GWF)*, pp. 278-283, November 2018.

1.4 Organization of the Thesis

The rest of this thesis is organized as follows.

Chapter 2 provides a background on RRM and a literature review on different methods employed for RRM.

In chapter 3, we study the problem of RRM in LTE networks with the use of reinforcement and deep reinforcement learning methods. In particular, the RRM tasks considered are resource block allocation, user-cell association, and power allocation. We demonstrate the advantages and challenges of using the proposed methods under various traffic scenarios and network architectures. Various network KPIs are targeted, such as latency, throughput, and load balancing.

In chapter 4, we turn the page to 5G networks, and we present several advancements with the use of reinforcement learning methods. Traffic heterogeneity, in addition to more stringent QoS requirements are addressed with the proposed solutions. Furthermore, we study the use of reinforcement learning for clustering and beamforming in mm-wave networks.

Chapter 5 introduces transfer in reinforcement learning as a solution to speed up the convergence of conventional Q-learning methods that rely on tabular approaches. We demonstrate the performance of transfer in reinforcement learning under different network scenarios and we study its complexity and convergence proprieties compared to different machine learning algorithms.

Finally, a summary of this work and some challenges and future directions are presented in chapter 6.

Chapter 2

Background and Literature Review

2.1 Background

This section provides an overview of RRM and ML methods that are used throughout this thesis. Section 2.1.1 presents the fundamentals and requirements of RRM in LTE and 5G wireless networks. Section 2.1.2 presents background information on the three machine learning methods used, namely reinforcement learning, deep reinforcement learning, and transfer in reinforcement learning.

2.1.1 Radio Resource Management

Next-generation wireless systems are expected to serve massive connectivity between heterogeneous users such as human users, machines, vehicles, etc. With the heterogeneity of these users, they pose diverse QoS requirements. In addition, network dynamics such as users mobility, fading characteristics, and traffic variations calls for efficient utilization of network resources. RRM is the process of assigning wireless network resources, e.g., spectrum and power, to network users in order to achieve high quality wireless communication [4]. In this thesis, we consider RRM functionalities that are concerned with spectrum allocation, power allocation, interference management, user-cell association, and mm-wave beam management that are applied to LTE and 5G wireless networks.

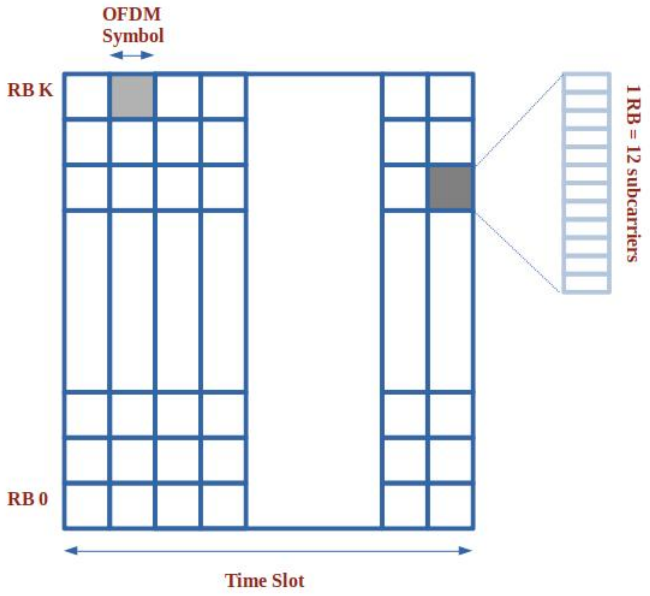

Figure 2.1: Resource grid of LTE/5G networks.

Spectrum Allocation

To schedule access to the time frequency radio resources across users, LTE and 5G have adopted Orthogonal Frequency Division Multiple Access (OFDMA) scheme. In OFDMA, time and frequency resources are organized in a grid, namely resource grid. The time direction of the grid is divided into a number of Orthogonal Frequency Division Multiplexing (OFDM) symbols, whereas the frequency direction is divided into a number of orthogonal subcarriers as shown in Fig. 2.1. Every 12 contiguous subcarriers are grouped to form one Resource Block (RB). The RB spans a time slot in the time direction that consists of multiple OFDM symbols. If short cyclic prefix is used, a time slot consists of 7 OFDM symbols, whereas if extended cyclic prefix is used, a time slot consists of 6 OFDM symbols. Therefore, a RB constitutes the minimum unit of allocation to a user. In addition, a RB is allocated to one user per cell, hence avoiding intra-cell interference. Since the time slot is the minimum duration of a transmission, it is denoted as Transmission Time Interval (TTI). Therefore, resource allocation can be casted as the assignment process of the number and positions of RBs to users in the network each TTI.

LTE supports two duplexing methods: Frequency Division Duplex (FDD) and Time

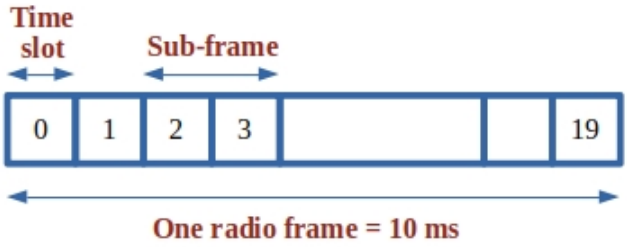

Figure 2.2: LTE type-1 frame structure.

Division Duplex (TDD). In FDD, separate frequency bands are used to enable simultaneous uplink and downlink communication. Fig. 2.2 presents LTE type-1 frame structure used for FDD operation. In particular, a radio frame of 10 msec duration is sub-divided into 10 sub-frames of 1 msec duration each. Furthermore, each sub-frame consists of two time slots, i.e., two TTIs. In FDD, the whole radio frame is used for both uplink and downlink communication through allocation of separate frequency bands.

On the other hand, in TDD, the radio frame is divided into two equal portions, where uplink and downlink communication is multiplexed, i.e., uplink and downlink get 5 msec each for transmission.

Power Control

Inter-cell interference constitutes a significant impairment to having a reliable wireless communication. In the downlink, a cell with high transmission power can cause severe inter-cell interference to a cell-edge user that belongs to an adjacent cell. Furthermore, in the uplink, close users that belong to different cells and are using the same RBs can cause inter-cell interference. Therefore, efficient RRM is needed to mitigate inter-cell interference in order to achieve reliable communication. One approach to lessen interference is to use efficient power control schemes, i.e., to adjust cell and users transmission powers.

LTE uses power control schemes in order to limit the transmission power of cells and users, hence mitigates inter-cell interference. Closed-loop power control is used in the

uplink. In particular, the base station can perform no power control, channel inversion, or fractional power control. Under no power control, all users are free to allocate their power in the uplink which might lead to high spectral efficiency but low battery efficiency and poor fairness [5]. Channel inversion, on the other hand, results in the same received power for all users, which achieves fairness. Between the extremes, fractional power control can be used to balance fairness, spectral efficiency, and energy-efficiency.

User-Cell Association

User-cell association is used to determine the cell that a user should associate with before data transmission commences. It plays a key role in interference mitigation, load balancing, spectrum efficiency, and energy efficiency [6]. Users are associated to cells based on their demands, their distance from the cell, and their channel quality [4].

Beamforming and User-Clustering

Several technologies are evolving in 5G and upcoming 6G wireless networks such as mm-wave, beamforming, and Non-Orthogonal Multiple Access (NOMA). In particular, mm-wave is enabling communication over higher bands, which are less congested and provide larger bandwidths. Furthermore, beamforming facilitates the communication with focused power gains in the direction of the user. NOMA, in addition, improves network's performance by multiplexing users' signals in the power domain. With these technologies, future wireless networks can provide better network capacities to the covered users. However, many challenges still exist. On one hand, with the use of power domain NOMA, user-beam association is needed to construct and assign better beams to users. Therefore, user clustering stands as an important decision in mm-wave networks employing power domain NOMA. On the other hand, as mentioned previously, power allocation plays a key role in mm-wave networks. In particular, power allocation is needed to mitigate inter-beam interference and to improve network throughput.

2.1.2 Machine Learning Methods

Over the past decade, the huge growth in data across many different fields resulted in big data challenge which amplified the need for intelligent data analysis schemes. Various machine learning methods emerged, such as deep learning, and they have been used along with traditional machine learning methods to cope with the big data problem. Recently

they have been adopted in wireless networks. In this section we give a brief overview of the widely used techniques adopted for the wireless network.

Before delving more into machine learning, it is worth mentioning a few words on the definition of machine learning and its difference with Artificial Intelligence (AI). The Turing test, proposed by Alan Turing [7], was designed to answer the question "Can a machine think?". The test identifies machine intelligence if a human cannot tell whether the written responses come from a person or from a computer. With this, AI can be defined as the technology which enables a machine to mimic human behavior. In order for the machine to pass Turing test, it has to have several capabilities such as the ability to communicate in a human language to interpret the test (i.e., natural language processing), the ability to hear and store information (i.e., knowledge representation), the ability to draw conclusions (i.e., reasoning), the ability to adapt to new situations (i.e., machine learning), etc [8]. As such, machine learning is a subset of AI which allows the machine to learn from past data to perform a specific task. Indeed, in this thesis, we use the terms machine learning and AI interchangeably.

Machine learning schemes can be classified into four main categories: Supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning. These four categories differ in the way the algorithm is being trained [9]. In supervised learning, the training is performed initially by some labeled data. The labeled data represents a set of inputs with their corresponding outputs, known beforehand. Therefore, supervised learning algorithms are well-suited to applications with historical data. Feature extraction and classification have been applied to several signal processing problems. In classification, the task is to identify which set of categories a new observation belongs to. In contrast, unsupervised learning algorithms aim to infer features in the data, thus inferring the implied structure. Semi-supervised learning algorithms use both labeled and unlabeled data. Finally, reinforcement learning uses data from the implementation instead of historical data. The aim of reinforcement learning is to improve the performance of an agent in a certain task using feedback from the environment. As such, the agent's goal is to predict the next action to take to earn the biggest final reward. Reinforcement learning is unsupervised, however, the way of learning is different than other unsupervised learning techniques. Rather than learning the structure of some data, reinforcement learning tries to explore the best actions in the medium of operation. Hence, the ability to capture the environment through feedback and perform actions makes reinforcement learning suitable for problems involving a series of decisions, i.e., following a policy of actions according to observed environment's state.

The following sections provide an overview of the main methods used in this work. These methods are reinforcement learning, deep learning, deep reinforcement learning,

and transfer in reinforcement learning.

Reinforcement Learning

The problem of reinforcement learning is a straightforward framing of learning from interaction between a decision-maker (i.e., an agent) and its environment [10]. Figure 2.3 presents a conceptual diagram of the operation of tabular methods of reinforcement learning. In particular, the elements of reinforcement learning constitute an agent that interacts with its environment by making action decisions and receiving a reward signal. The agent observes a state that characterizes its environment. The state of the environment should compactly retain relevant information of the environment through immediate and past sensations. A state signal that satisfies this condition is said to have Markov property. Therefore, the reinforcement learning problem can be cast as a Markov Decision Process (MDP) with the four-element tuple: {states, actions, transition probabilities, and reward function}. In particular, at each iteration τ, the agent receives some representation of the environment's state $S_\tau \in \mathfrak{S}$, where $\mathfrak{S}$ is the set of possible states. Afterwards, the agent selects an action $A_\tau \in \mathcal{A}(S_\tau)$, where $\mathcal{A}(S_\tau)$ is the set of possible actions available in state S_τ. At the next iteration step $(\tau+1)$, the agent receives a reward value $R_{\tau+1}$ in response to the taken action and the environment's state changes to $S_{\tau+1}$. Furthermore, the transition probability, $p(s'|s,a) = Pr\{S_{\tau+1} = s'|S_\tau = s, A_\tau = a\}$, defines the probability that the environment's state changes from $S_\tau = s$ to $S_{\tau+1} = s'$ when the agent performs action $A_\tau = a$.

The ultimate goal of a reinforcement learning's agent is to identify the best policy that maximizes its total expected reward as follows:

$$\max_{\pi(s)} \mathbb{E}[R_{\tau+1} + \gamma R_{\tau+2} + \gamma^2 R_{\tau+3} ... | S_\tau = s, A_\tau = a], \tag{2.1}$$

where $0 \leq \gamma \leq 1$ is a discount factor that reduces the contribution of future rewards in addition to maintaining a stability in computations. $\pi(s)$ is a policy that defines the optimal action at state s. In particular, the mapping from state S_τ to action A_τ is performed by following a stochastic policy $\pi(a|s) = Pr\{A_\tau = a|S_\tau = s\}$. As such, the goal of a reinforcement learning agent is to seek a policy that maximizes its total expected discounted reward over the long run. To achieve that, a value function is used to quantify how good is a certain policy given a state-action pair. An action-value function can be defined as follows:

$$q_\pi(s,a) = \mathbb{E}_\pi[R_{\tau+1} + \gamma R_{\tau+2} + \gamma^2 R_{\tau+3} ... | S_\tau = s, A_\tau = a], \tag{2.2}$$

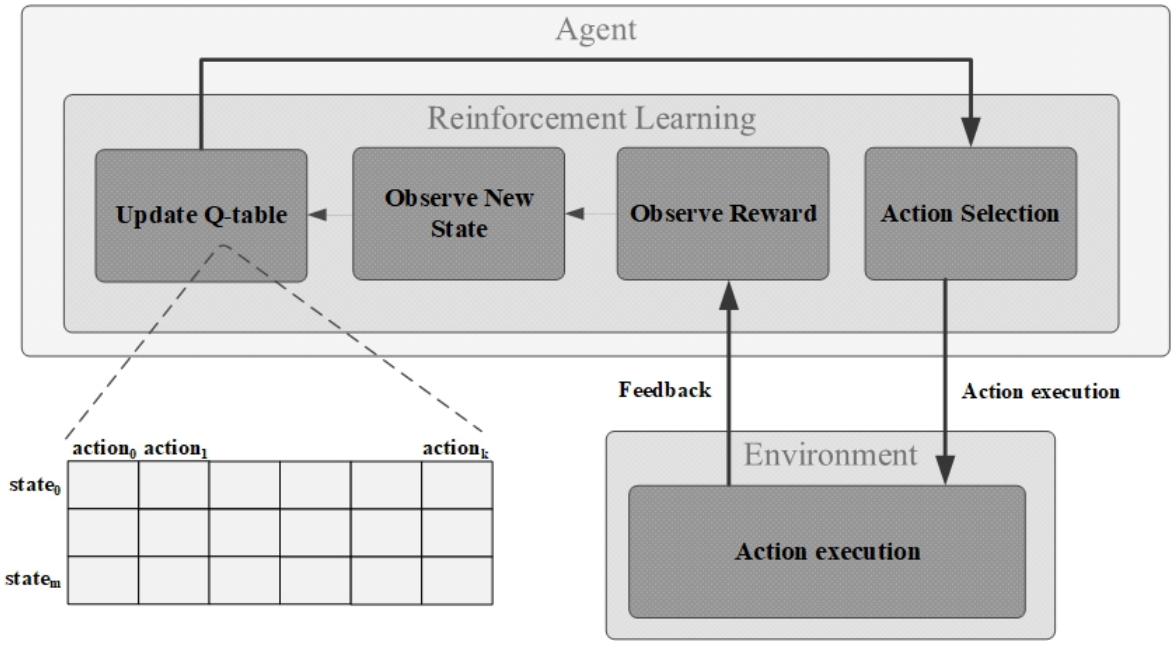

Figure 2.3: Conceptual diagram of Q-learning operation.

where $q_\pi(s, a)$ is the action-value (i.e., quality value or Q-value) of policy π when starting at s^{th} state and taking a^{th} action. The optimal value function can be computed through a brute-force method which becomes intractable for large state-action space. Instead, temporal difference methods, such as Q-learning and State-Action-Reward-State-Action (SARSA), are used to compute an estimation of the value function. In Q-learning, the following update rule is used to approximate an agent's policy:

$$q_\tau(S_\tau, A_\tau) \leftarrow q_\tau(S_\tau, A_\tau) + \alpha[R_{\tau+1} + \gamma \max_{a' \in \mathcal{A}_{\tau+1}} q_{\tau+1}(S_{\tau+1}, a') - q_\tau(S_\tau, A_\tau)], \tag{2.3}$$

where $q_\tau(S_\tau, A_\tau)$ represents a Quality-value (Q-value), and $\max_{a' \in \mathcal{A}_{\tau+1}} q_{\tau+1}(S_{\tau+1}, a')$ computes an approximate of the Q-value at the next state $S_{\tau+1}$ under the best action. On the other hand, SARSA uses the following update rule as follows:

$$q_\tau(S_\tau, A_\tau) \leftarrow q_\tau(S_\tau, A_\tau) + \alpha[R_{\tau+1} + \gamma q_{\tau+1}(S_{\tau+1}, a\tau + 1) - q_\tau(S_\tau, A_\tau)]. \tag{2.4}$$

It is worth mentioning that the design of the state and the reward functions can influence the goal and outcome of the reinforcement learning agent. Appendix A presents detailed guidelines on how to design the state and the reward functions of a reinforcement learning agent.

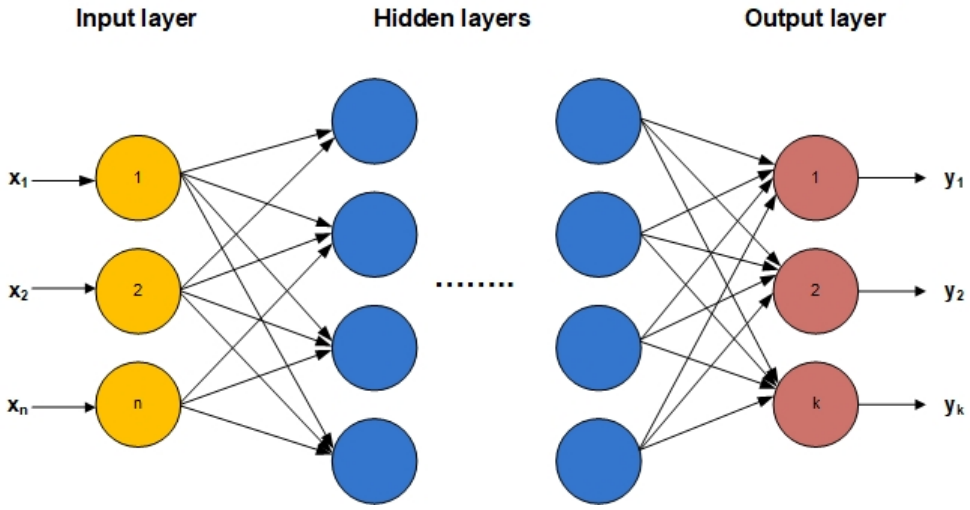

Figure 2.4: A typical structure of a neural network.

Deep Learning

Besides reinforcement learning, neural networks have been recently used in the state-of-the-art wireless network research. Neural networks are designed to mimic the structure of neurons in the human brain. Fig. 2.4 presents a typical structure of a neural network. In particular, a neural network consists of three types of layers: input layer, output layer, and hidden layers. Each layer comprises a set of artificial neurons that perform certain mathematical function, namely neuron activation function. In the input layer, a set of neurons are used to perform pre-processing on an input feature vector $\boldsymbol{x}$. Furthermore, another set of neurons at the output layer are used to produce the outcomes, $\boldsymbol{y}$, of the neural network. Neurons in a certain layer are connected to the neurons in the preceding layer, where each connection has a weight. In the training phase, the weights are adjusted according to the training dataset, where the training dataset provides a set of inputs and their expected outputs, i.e., labels.

Neuron's Activation Function

The neuron is the basic unit of a neural network, where it performs a certain mathematical function on its input information. Fig. 2.5 presents a typical structure of a neuron, where $\boldsymbol{x}$ represents input feature vector, $\boldsymbol{w}$ represents vector of weights of a neuron's pre-activation function ($z(\boldsymbol{x})$), b represents a bias value, and $\psi(\boldsymbol{x})$ represents the neuron's activation function. Furthermore, $\boldsymbol{w}$ and b are denoted by hyper-parameters of the neuron. As such, the combined processing of the neuron consists of pre-activation and activation. In

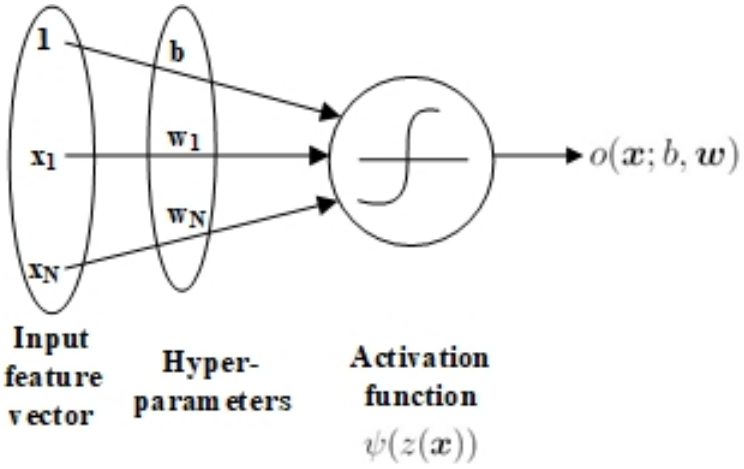

Figure 2.5: Structure of a neuron.

particular, the pre-activation is performed as follows:

$$z(\boldsymbol{x}) = b + \sum_{i=1}^{N} w_i x_i. \tag{2.5}$$

Therefore, the output of a neuron is represented as follows:

$$o(\boldsymbol{x}; b, \boldsymbol{w}) = \psi(z(\boldsymbol{x})) = \psi(b + \sum_{i=1}^{N} w_i x_i), \tag{2.6}$$

where $o(\boldsymbol{x}; b, \boldsymbol{w})$ represents the output of the neuron, and $\psi(z(\boldsymbol{x}))$ represents the activation function of the neuron.

The activation function of an artificial neuron can take several forms such as linear, sigmoid, hyperbolic tangent, and rectified linear activation functions. Fig. 2.6 presents example plots of the aforementioned activation functions.

Architectures of Neural Networks

Neural networks can be structured in different forms such as feedforward, convolutional, or recurrent neural network. A neural network with one hidden layer is a shallow neural network while a neural network with multiple hidden layers is a deep neural network. Furthermore, deep neural networks can have different forms such as feedforward, convolutional, recurrent.

In a feedforward neural network, information flows in one direction as shown in Fig. 2.4. In contrast, a deep recurrent neural network incorporates feedback connections among

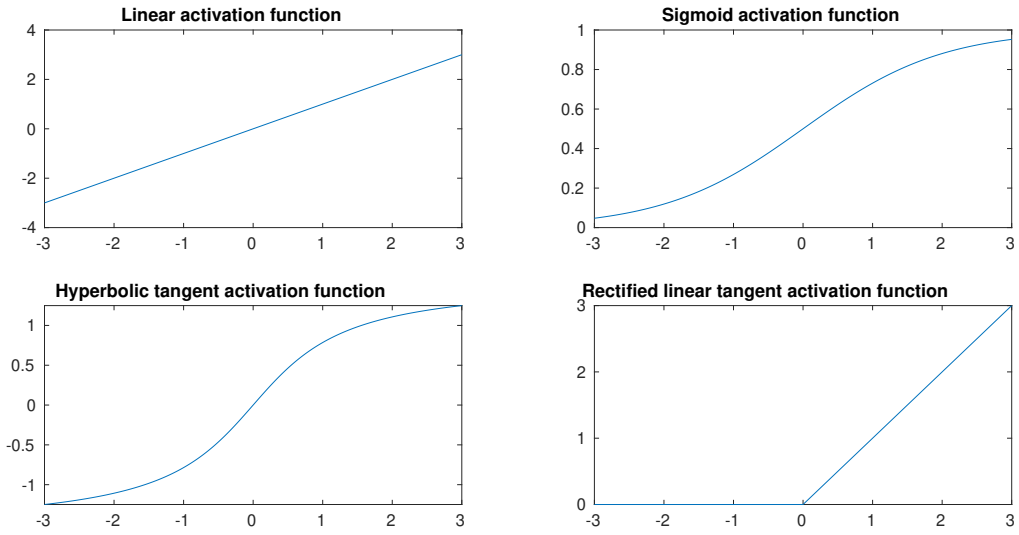

Figure 2.6: Examples of activation functions of a neuron.

layers as shown in Fig. 2.7. In particular, the next state of a neuron relies on input features as well as the current state. The feedback connections allow the deep recurrent neural network to infer relations in long sequential information, i.e., more efficient at generalization. Therefore, the activation function of a recurrent neural network's neuron can be written as follows:

$$o_\tau(\boldsymbol{x}) = f(o_{\tau-1}(\boldsymbol{x}), \boldsymbol{x}_\tau; b, \boldsymbol{w}), \tag{2.7}$$

where $o_\tau(\boldsymbol{x})$ and $o_{\tau-1}(\boldsymbol{x})$ are the output of the neuron at τ^{th} and $(\tau-1)^{th}$ iteration steps, respectively.

LSTM Neural Network

LSTM is a Recurrent Neural Network (RNN) architecture that is used to process entire sequences of data. In particular, LSTM is used to infer dependencies among long sequences of information. It can be used in tasks such as connected hand writing and speech recognition. An LSTM cell consists of an input gate, an output gate, and a forget gate as shown in Fig. 2.8, where σ represents a sigmoid function and tanh represents a hyperbolic tangent function. As such, LSTM is considered the most generic architecture of a recurrent neural network.

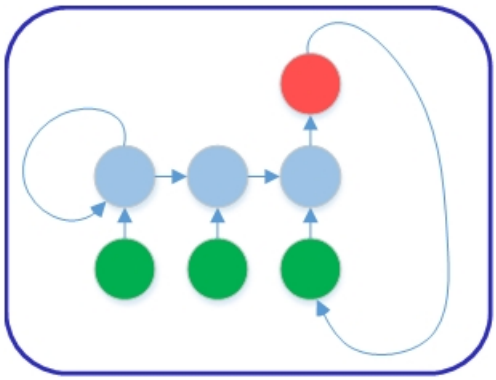

Figure 2.7: Feedback connections in a recurrent neural network.

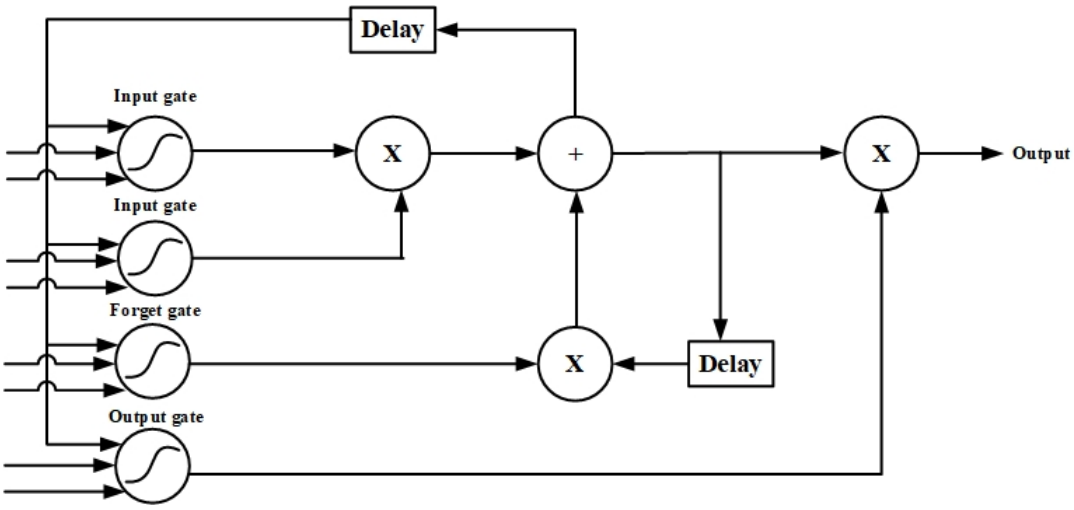

Figure 2.8: A conceptual diagram of a cell of an LSTM neural network [1].

Deep Reinforcement Learning

Tabular methods of reinforcement learning is known to suffer from long convergence due to the need of large sample space of experiences [11]. In particular, conventional reinforcement learning uses a table to store Q-values, namely Q-table, which leads to high complexity and large convergence time. This is because one Q-value is updated each iteration. As such, deep reinforcement learning methods have been used to improve the convergence of conventional reinforcement learning by introducing a deep neural network that is used for Q-value estimation. Moreover, the introduction of the deep neural network allows for learning correlations among input sequences.

One drawback of reinforcement learning is its tendency to diverge when using a non-linear function approximator such as neural network. To solve this, in [11], the use of an experience replay memory, in which network training is performed by sampling random experiences, is proposed. An experience is defined as $e = \{s, a, r, s'\}$. This experience can then be used in training the LSTM to refine the Q-value estimation at each iteration. Fig. 2.9 presents a conceptual diagram of a deep reinforcement learning method. In this thesis, the environment is considered to be the wireless environment, from which a feedback signal is transmitted to the agent to compute a reward and a next state signals. To maintain low complexity, the training of the main neural network is performed every multiple TTIs. In particular, a batch of experience samples is drawn randomly from the experience replay memory. The batch is fed to the target neural network in order to compute a sequence of reference responses. These responses constitute the labels used to train the main neural network. In addition, the target neural network is initially loaded with the weights of the main network. However, the update of the target neural network's weights are done every multiple TTIs to maintain stability [12].

Transfer in Reinforcement Learning

The idea behind transfer learning is to exploit the knowledge learned about one task to improve generalization in another task [1]. Indeed, humans can re-utilize their learned knowledge from a previous task in solving new tasks more rapidly or with better solutions [13]. This reduces the need for a large number of training samples, which is a common problem in reinforcement learning. For example, Temporal Difference (TD) methods, such as Q-learning, suffer from slow convergence due to the need of a large number of training samples of experience, commonly collected via trial-and-error approach over large number of iterations [12]. Hence, the key motivation to transfer learning in TD methods is to reduce the amount of samples needed for learning the target task, and to reduce the

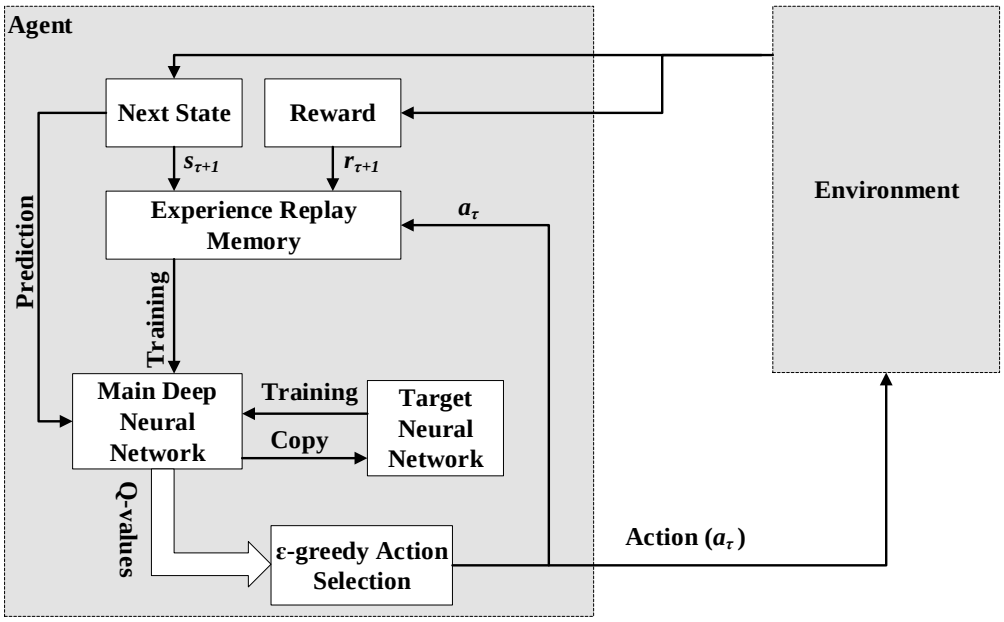

Figure 2.9: Architecture of a deep reinforcement learning method.

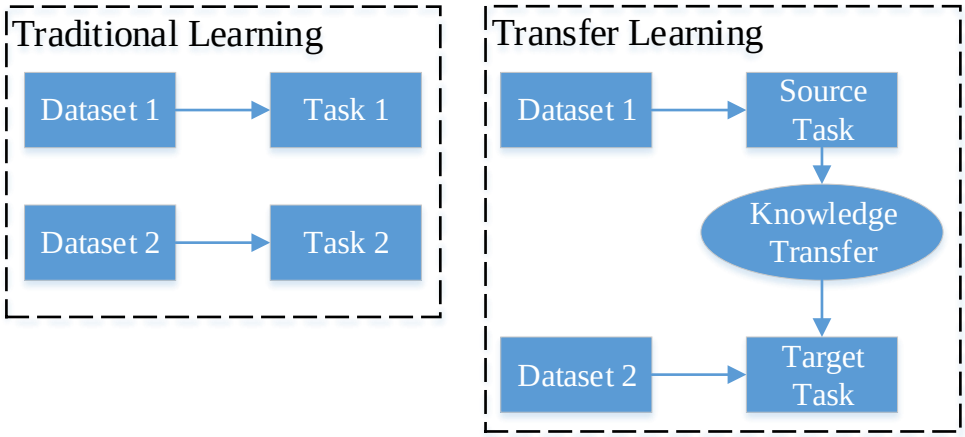

Figure 2.10: Conceptual explanation of the difference between traditional and transfer reinforcement learning.

convergence time. Furthermore, deep reinforcement learning can be considered as another technique to improve convergence, where efficient representations of the environment are drawn from high-dimensional input data that are further used to generalize over past experiences [11]. However, deep reinforcement learning generalizes over a localized domain (i.e., same knowledge domain), whereas transfer reinforcement learning aims at transferring knowledge across domains.

Fig. 2.10 presents a conceptual comparison between traditional and transfer reinforcement learning approaches. In particular, transfer learning considers knowledge transfer across tasks. Such knowledge transfer can transcend fixed or different domains. With fixed domain, the state-action spaces of the source and the target tasks are equivalent, whereas the objective, represented by the reward function, might differ. Transfer with different domains constitute different state-action spaces for source and target tasks [14, 15].

2.2 Related Work

ML is gaining a significant attention in the area of RRM for wireless networks. More specifically, the use of ML to improve the performance of wireless networks has increased

significantly in the last decade. This section surveys the most recent works on ML-enabled wireless networks. In particular, we focus on the works that employ reinforcement learning, deep learning, deep reinforcement learning and transfer learning. For more comprehensive surveys, the reader is referred to the works in [16–20]. Table 2.1 presents a summary on algorithms adopted for wireless networks. We start by providing a survey on conventional methods of optimizing network parameters, which includes heuristics, optimization-based methods, and conventional data science methods. Afterwards, we present the works with learning-based methods, which include reinforcement learning, deep learning, deep reinforcement learning, Bayesian reinforcement learning, transfer learning, federated learning, and imitation and meta-learning.

Table 2.1: Summary of research works on ML-enabled wireless networks.

Method	**Work**	**Network**	**Objective**	**Task**
Conventional Methods	[21]	mm-wave-NOMA	sum-rate	clustering
	[22]	mm-wave-NOMA	sum-rate	user clustering and NOMA power allocation
	[23]	mm-wave	sum-rate	user-cell association
	[24]	mm-wave femto-cell	sum-rate	user-cell association and resource allocation
	[25]	mm-wave	sum-rate	user-cell association
	[26]	vehicular networks	reliability and knowledge	power and resource block allocation
	[27]	UAVs	fairness	UAV's trajectory, user scheduling and bandwidth allocation
Reinforcement Learning	[28]	femto-cell networks	sum-rate	power allocation

Reinforcement Learning	[29]	femto-cell Universal Software Radio Peripheral (USRP)	sum-rate	power allocation
	[30]	D2D networks	sum-rate	spectrum and power allocation
	[31]	macro- and small-cell networks	energy-efficiency and QoS of users	traffic offloading
	[32]	UAVs	sum-rate	placement of UAVs
	[33]	UAVs	sum-rate	downtilt angle of ground stations
	[34]	mm-wave networks	rate	dynamic rate selection for energy harvesting, dynamic power allocation for heterogeneous network, and distributed resource allocation
	[35]	5G vehicular network	sum-rate	TDD uplink/downlink multiplexing
	[36]	heterogeneous networks with renewable energy sources	energy-efficiency	user scheduling and resource allocation
Deep Learning and Deep Reinforcement Learning	[37]	multi-cell network	energy-efficiency and sum-rate	power allocation
	[38]	massive Multiple Input Multiple Output (MIMO) mm-wave networks	spectral efficiency	beam alignment

Deep Learning and Deep Reinforcement Learning	[39]	virtual reality network on UAVs	quality of experience and delay	resource allocation and transmitted image size
	[40]	UAVs	energy-efficiency	deployment of UAVs and user association
	[41]	virtual reality network	breaks-in-presence	user-cell association
	[42]	green heterogeneous networks	energy-efficiency	networking, caching, and computing resources
	[43]	cognitive radio networks	sum-rate	packet transmission scheduling
	[44]	5G mm-wave networks	sum-rate	beamforming, power control, and interference management
	[45]	energy harvesting networks	sum-rate	power allocation
	[46]	vehicular networks	rate and packet delivery probability	spectrum allocation
	[47]	ultra-dense networks	spectrum efficiency, energy-efficiency and fairness	resource allocation
	[48]	multi-cell network	sum-rate	power allocation
	[49]	5G networks	sum-rate	resource block allocation
	[50]	multi-carrier NOMA	maximization of sum rate and maximization of minimal rate	spectrum and power allocation

	[51]	industrial wireless networks	reliability, latency, and throughput	resource block allocation
Bayesian Reinforcement Learning	[52]	D2D network	rate	spectrum, transmission mode, and power allocation
	[53]	mm-wave networks	increase number of covered users while meeting their QoS	beam direction
Transfer Learning	[54]	small-cell networks	rate	user-cell association
	[55]	general Radio Access Network (RAN)	average rate and fairness	downlink power control and edgeless connectivity
Federated Learning	[56]	general RAN	delay	parameters of federated learning model
	[57]	general RAN	convergence time of federated learning	resource allocation and selection of the users that contribute to the global model
	[58]	vehicle-to-everything network	sum-rate, latency and reliability	transmission mode selection, resource block allocation, and power allocation
Imitation and Meta Learning	[59]	D2D network	minimum data rate	spectrum and power allocation
	[60]	general RAN	sample efficiency	pruning policy
	[61]	UAVs	delay	trajectory of UAVs

2.2.1 Conventional Optimization Methods

In [21], the authors present a method for clustering the users in a mm-wave-NOMA system with the objective of maximizing the sum-rate. In particular, the authors use agglomerative hierarchical machine learning-based clustering technique to perform the automatic identification of the optimal number of clusters. The main advantage of agglomerative clustering is that specifying the number of clusters is not needed in constrast to conventional clustering techniques such as K-means clustering. Furthermore, the proposed technique uses a cosine similarity as the metric to identify users that can be grouped in the same cluster.

In [22], the authors aim to maximize the sum rate of a mm-wave-NOMA system by solving user clustering and NOMA power allocation. The authors utilize the correlation features of the user channels to develop a K-means clustering algorithm. They also drive a closed form solution for the optimal NOMA power allocation within a cluster.

The authors in [23] aim to maximize the throughput of an ultra-dense mm-wave network with multi-connectivity by solving the user-cell association problem. Using multi-label classification technique, the authors investigate three approaches for user-cell association: binary relevance, ranking by pairwise comparison, and random k-lebelsets.

In [24], the authors consider clustering, user-cell association, and resource allocation in mm-wave femto-cell networks. The solution is performed by difference of two convex functions programming with the aim to cluster femtocells and femto-users based on having the most Line of Sight (LoS) connectivity in order to provide maximum system rate.

The work in [25] addressed the improvement of user performance in a mm-wave network by considering association of users to multiple base stations. In particular, the user association problem is transformed into a multi-label classification problem by treating users and base stations as samples and classes, respectively, which is then transformed into a series of single-label classification. As such, the authors propose two multi-label classification algorithms, namely ranking by pairwise comparison and random k-labelsets that solve the user association without the knowledge of channel state information. Furthermore, in order to reduce the need for large sample space, the authors formulate the single-label classification as a Markov random field, where a novel feature extraction method that utilizes geographical location and topological information is adopted.

Age of information in vehicular networks was addressed in [26]. In particular, resource allocation in vehicular networks becomes a challenging process due to the highly dynamic behaviour of the network conditions. As such, the authors aim to balance the trade-off between maximizing reliability, which is achieved by minimizing the probability that age of

information exceeds a predefined threshold, and maximizing the knowledge about network dynamics. To achieve this, the authors propose an online decentralized solution based on Gaussian process regression for power and resource block allocation.

In [27], the authors propose to use UAVs as a relay to decode-and-forward signals transmitted from a base station to a ground user. In particular, fairness is achieved by maximizing the minimum throughput among all users. As such, a low-complexity algorithm based on the alternating direction method of multipliers was proposed to jointly optimize the UAV trajectory, user scheduling and bandwidth allocation.

2.2.2 Reinforcement Learning

Co-tier interference management was addressed in [28] by employing an online Q-learning algorithm for power control for improving aggregate network rate. In particular, three algorithms were proposed: Centralized, femto-based distributed and subcarrier-based distributed power control using Q-learning. Each algorithm works in two different paradigms: Independent and cooperative learning. In [29], we developed a USRP-based platform for addressing interference in femtocell networks. In particular, distributed independent and cooperative Q-learning approaches were used to perform an online power allocation with the aim to maximize aggregate femtocell rate while maintaining QoS of users of the macrocell.

Q-learning is used in [30] to address the problem of joint spectrum and power allocation in D2D networks. The proposed algorithm aimed at improving network capacity compared to maximum power allocation and random spectrum allocation.

In [31], a macro-cell base station learns to make traffic offloading decisions to small-cell base stations with the aim to improve energy-efficiency while maintaining QoS experienced by the mobile users. The traffic offloading problem is modelled as a discrete time MDP. Furthermore, a centralized Q-learning algorithm was developed to address the problem of curse of dimensionality.

Recently, deployment and placement of UAVs have been active area of research. In [32], the authors address the problem of 3D positioning of aerial base station to assist ground stations for covering mobile users with enhanced quality of service. Due to users' mobility, the network topology gradually changes which impacts the quality of service of users. Furthermore, the positioning algorithm will need more time to re-learn the network. As such, an agile and fast learning algorithm is needed. Authors propose to use a Q-learning algorithm to find an efficient placement of aerial stations to maximize throughput of the network. The difference between current and previous QoS (i.e., throughput) constitute the

agent's reward. This motivates the agent to improve its decision to gain positive rewards, hence improving the network throughput. After the training phase of Q-learning, it has been shown that the algorithm can adapt to small changes in the network more rapidly.

In [33], the authors consider mobility management of UAVs with the aim of improving UAVs' connectivity while maintaining a reasonable throughput performance for ground users. In particular, a model-free reinforcement learning, i.e., Q-learning, is used to learn how to tune the downtilt angles of ground base stations with the help of Reference Signal Received Power (RSRP) and ground user's capacity, whereas UAV's trajectory is assumed to be known beforehand.

The work in [34] studies RRM for mm-wave networks under unknown channel state information. The authors address three resource allocation problems: dynamic rate selection for energy harvesting, dynamic power allocation for heterogeneous network, and distributed resource allocation. As such, an online reinforcement learning approach that is modeled as a contextual uni-modal multi-armed bandit is used to learn the radio resource allocation policy.

In [35], the authors address throughput maximization in a 5G vehicular network that works with TDD frame structure. In TDD, uplink and downlink communication is multiplexed in time domain. In particular, a reinforcement learning algorithm is used to control TDD configuration, i.e., uplink/downlink ratio, while taking into account network and traffic load conditions.

The work in [36] addresses the problem of user scheduling and resource allocation in heterogeneous networks with renewable energy resources with the aim to maximize energy-efficiency of the network. As network environment is stochastic and the state-action space is continuous, the authors propose to use a model-free reinforcement learning algorithm, namely actor-critic reinforcement learning. As such, the actor generates stochastic actions, whereas the critic is used to criticize the actor's policy.

2.2.3 Deep Learning and Deep Reinforcement Learning

The work in [37] aims at maximizing system's energy-efficiency and sum rate through power control mechanism. To achieve this, the authors propose a neural network solution that is trained with a large training set generated through a branch-and-bound method in an offline setup, hence reducing the complexity of the algorithm. The trained neural network was able to achieve the optimal power allocation.

The work in [38] considers beam alignment in massive MIMO mm-wave networks. A neural network approach is adopted and trained offline to predict the beam distribution

vector using partial beams. Results show the effectiveness of the approach in terms of improving the total training time slots and the spectral efficiency.

The work in [39] considers a network of virtual reality users communicating using UAVs. UAVs behave as relays that receive images on the uplink and forwarding them on the downlink using LTE licensed and unlicensed bands, respectively. Furthermore, the quality of images can be adjusted to change the transmitted data size in order to fit the resources allocated. Therefore, dynamic resource allocation is required to both improve the quality of experience of users and to meet the delay requirements of virtual reality applications. To achieve this, the authors employ a deep Echo-State Network (ESN) algorithm. ESN is a type of recurrent neural network with sparsely connected hidden layers. ESN aims to learn the weights of the output layer only, while weights of hidden layers are fixed and randomly assigned. This in turn increases the speed of learning over conventional recurrent neural networks.

In [40], the problem of jointly optimizing UAV deployment, user association, and power efficiency is addressed to meet the illumination and communication requirements of users. The problem is solved using an deep learning algorithm that combines gated recurrent units with convolutional neural networks with the aim to achieve energy-efficiency.

The authors in [41] use deep ESN and federated learning to address the problem of breaks in presence for wireless virtual reality users. In order to reduce the breaks in presence, a solution that considers the virtual reality application, transmission delay, video quality, and users' awareness of the environment is studied. As such, the authors develop a federated ESN learning algorithm, in which each base station trains its machine learning algorithm locally using data collected from users' locations and orientations.

The work in [42] proposes a framework that enables orchestration of networking, caching and computing resources with the aim to improve green heterogeneous wireless networks. In particular, deep reinforcement learning was proposed to allocate networking, caching, and computing resources dynamically.

In [43], the authors address the problem of low packet transmission efficiency of Internet-of-Things Internet-of-Things (IoT) users in cognitive radio networks. In particular, a transmission scheduling mechanism based on Q-learning was developed to learn the appropriate strategy of transmitting packets to maximize system throughput. Furthermore, to address the problem of curse of dimensionality, a stacked auto-encoder deep learning algorithm is adopted to map the relation between the states and the actions, i.e., Q-values.

In [44], the authors formulate the joint design of beamforming, power control, and interference management in 5G mm-wave networks as a non-convex optimization prob-

lem. A solution using deep reinforcement learning was proposed to address the sum rate maximization with feasible complexity.

Energy harvesting networks require the exchange of the state of the energy harvesting process among nodes. In [45], the authors study the design of a decentralized power control technique that does not require the exchange of state information among nodes in order to maximize throughput in energy harvesting networks. To achieve this, the authors propose a mean-field multi-agent deep reinforcement learning algorithm for optimal power allocation. The algorithm is distributed and implemented in each node without the need to exchange information with other nodes in the network. Furthermore, the authors show that the obtained policies converge to the stationary Nash equilibrium.

The work in [46] addresses the spectrum sharing problem in vehicular networks. In particular, the spectrum used by a vehicle-to-infrastructure is shared with a vehicle-to-vehicle communication links. Multi-agent reinforcement learning was adopted, where each vehicle-to-vehicle link is modeled as a separate agent that uses a fingerprint-based deep Q-network to solve the spectrum sharing problem with the aim to improve vehicle-to-infrastructure capacity and vehicle-to-vehicle payload delivery probability.

The authors in [47] consider a multi-objective resource optimization in an ultra-dense network with the aim to balance the trade-off among spectrum efficiency, energy-efficiency and fairness. In particular, a Q-learning algorithm is used to generate the training samples of a model-driven deep neural network. Q-learning is designed to consider limited channel state information, hence reducing the need for massive labeling data. In addition, the deep neural network consists of a series of alternating direction method of multipliers. The approach show a rapid convergence, in addition to avoiding the need of random initialization of the neural network.

In [48], the authors aim at improving the network-wide capacity of a multi-cell scenario using deep reinforcement learning. In particular, a deep-Q-full-connected-network is used to learn the multi-cell power allocation to maximize the overall capacity of the network. The proposed solution is compared to water-filling and Q-learning power allocation, which demonstrates a significant speedup in convergence.

Radio resource scheduling in 5G networks with different numerologies is investigated in [49]. In particular, a numerology-agnostic deep reinforcement learning framework is proposed for resource block allocation. The algorithm was designed to motivate the maximization of the throughput, by improving modulation and coding scheme, while maintaining fairness among users.

Resource allocating in multi-carrier NOMA systems is considered in [50] with the use of deep reinforcement learning. In particular, the authors propose an attention-based deep

reinforcement learning algorithm to perform joint channel and power allocation in order to improve NOMA system performance represented by two metrics, i.e., maximization of sum rate and maximization of minimal rate.

The work in [51] studies industrial wireless networks that incorporate uRLLC users with interference, reliability, latency, and throughput requirements. In particular, the authors propose a deep reinforcement learning to perform resource allocation for the uRLLC in a decentralized setup.

2.2.4 Bayesian Reinforcement Learning

In [52], D2D pairs aim to select their transmission channels, modes, base stations, and power levels without relying on a decision from the base station. The problem is formulated as a Bayesian coalition formation game and a Bayesian reinforcement learning was used to maximize the long-term rewards of D2D pairs.

The authors in [53] consider beamforming in mm-wave networks with the aim to increase the number of covered users while satisfying their QoS requirements. In particular, the authors model the problem of finding the optimal beam direction as a multi-armed bandit. In addition, Thomspon sampling method is used to solve the multi-armed problem and provide faster convergence.

2.2.5 Federated, Transfer, and Meta Learning

User-cell association between small-cells and users was addressed in [54] through a collaborative learning approach, namely imitation learning. In particular, a neural Q-learning algorithm was proposed to enable a user to predict its reward function, hence to select its serving base station, by exploiting the similarities with its neighboring users. The algorithm demonstrate a speedup in convergence compared with conventional user-cell association without imitation learning.

The work in [55] presents the design of a reinforcement learning framework for RRM. The framework consists of a learner that learns an RRM policy from the network, and a set of distributed actors that execute the policy of the learner to generate a stream of experience. One realization of the framework is modeled using three components: neural-fitted Q-iteration, ensemble learning, and transfer learning. Furthermore, this realization was evaluated for two RRM problems: downlink power control and edgeless connectivity. Meanwhile, in [62] a learning approach, based on neural network and Q-learning, is used

to perform resource block allocation for latency and packet drop rate minimization. This approach aims at learning the best scheduling rule, such as fairness at every iteration.

The work in [56] addresses the convergence improvement of federated learning. In particular, the training of a federated learning model involves the cooperation between the base station and the users, where users generate a local model and send its parameters to the base station, which integrate them and generate a global model. As such, the computation and communication latencies impact the performance of federated learning. The authors address this by formulating a delay minimization problem and prove that it is a convex function of the learning accuracy. Then, they use a bisection search algorithm to the obtain the optimal solution. Meanwhile, the work in [57] addresses the convergence time problem by the appropriate selection of the users that contribute to the global model with their local model's parameters. In addition, the authors consider wireless resource allocation. As such, the joint learning problem, i.e., resource allocation and user selection, is formulated as an optimization problem with the aim to minimize the convergence time of federated learning. Furthermore, neural networks were used to improve the accuracy of the global model by estimating the parameters of the local model of the users that are not allocated any resource blocks for transmission.

In [58], the authors address the strict QoS requirements of vehicle-to-everything communication. In particular, the authors aim to maximize the capacity of the network while meeting the latency and reliability requirements. To achieve this, they propose a deep reinforcement learning algorithm that aims to learn transmission mode selection, resource block allocation, and power control for the vehicular network. Furthermore, the authors propose a federated deep reinforcement learning to address the limitation of deep reinforcement learning when it is trained locally.

In [59], resource allocation in D2D communication is considered to maximize the minimum data rate among D2D pairs. In particular, imitation learning is used to learn a good auxiliary prune policy to speed up the branch-and-bound algorithm for joint channel and power allocation. Furthermore, a mixed training strategy that involves a deep neural network was proposed to improve the generalization of the of the imitation learning.

In [60], imitation learning is used to improve sample efficiency and feasibility problem in learning techniques applied to radio resource management. In particular, the goal was to learn the optimal pruning policy in the branch-and-bound algorithm using a transfer via self-imitation method. Furthermore, the method is used to quickly adapt to new tasks with few additional unlableled training set.

In [61], the authors study the optimal design of a UAV's trajectory to cover users with dynamic traffic load. In particular, a meta-reinforcement learning algorithm was proposed

to optimize the trajectory of UAVs while considering the uncertainty and dynamic traffic load of the users. The meta-learning approach is used to tune the parameters of the reinforcement learning with the aim to provide on time service to ground users.

2.2.6 Research Gaps

The work in this thesis studies the methods of model-free reinforcement learning in detail to demystify its advantages and limits as applied to the area of wireless networks. We present reinforcement learning, deep reinforcement learning, and transfer in reinforcement learning for current and next generation wireless networks, where we address the improvement of several network KPIs in addition to presenting the challenges and complexities of the proposed methods. Unlike the previous works in the literature, we conclude that knowledge-driven methods such as transfer in reinforcement learning constitute a potential candidate for a fully autonomous wireless network. As such, we present transfer in reinforcement learning to address interference management in wireless networks. Finally, we discuss the challenges and the open issues in knowledge-driven learning.

Chapter 3

Reinforcement Learning for LTE Networks

3.1 Introduction

The works presented in this chapter focus on the use of model-free reinforcement learning to improve the performance of LTE networks employed for different traffic and network conditions, and RRM tasks. In particular, we address throughput, latency, and reliability improvement using reinforcement and deep reinforcement learning. Different applications have been chosen to reflect the studied KPIs. In particular, two application types have been used throughout this chapter, tactile and microgrid communication. The Tactile application demonstrate the need to improve the throughput of tactile users, whereas the microgrid application demonstrates the need to improve latency of microgrid users. Furthermore, the coexistence of users belonging to each application with traditional user equipments calls for fairness among users.

This chapter is organized as follows. In section 3.2, we present the use of reinforcement learning for resource block allocation with the aim to balance throughput of both tactile and traditional user equipments. In section 3.3, community resilience microgrids are presented, where reinforcement learning is used for resource block allocation with the aim to achieve low latency for microgrid users while maintaining fairness across the network. With the help of deep reinforcement learning, resource block allocation and user-cell association were addressed for mission critical and microgrid users in sections 3.4 and 3.5, respectively.

3.2 Q-learning based Resource Allocation for Data Intensive and Immersive Tactile Applications

The immersive tactile applications that are emerging in the entertainment, education and health industries are anticipated to be available for mobile users in the close future. These applications are data-intensive and delay-sensitive due to the nature of information that is being exchanged. With today's mobile networks, the throughput and latency challenges are the major roadblocks for mobile users. In this work, we propose a resource allocation technique with the aim of increasing throughput and reducing latency of Data Intensive Device (DID)s [63]. We consider the coexistence of DIDs with traditional UEs on a two-tier, densely deployed network of Small-cell Base Station (SBS)s and evolved NodeB (eNB)s. We propose a Q-learning-based resource allocation scheme, namely, Throughput-Maximizing Q-Learning (TMQ) that learns the efficient resource allocation of both SBSs and eNB. We show that by combining the two-tier network model and careful design of TMQ's reward, the algorithm can improve multiple network metrics simultaneously. The proposed technique is compared with the well-known Proportional Fairness (PF) algorithm in terms of average throughput, delay, and fairness. Simulation results show significant improvement in throughput, 80% reduction in delay, and 6% increase in fairness.

3.2.1 System Model

We consider a two-tier network of one eNB, $M \in \mathcal{M}$ SBSs, and N_m users per SBS as shown in Fig. 3.1. Let $J \in \mathcal{J}$ be the set of base stations (including eNB and SBSs) and N_j is the number of users attached to j^{th} base station. Two types of users are considered: DIDs, which can be haptics gadgets, Virtual Reality (VR)/Augmented Reality (AR) devices, mobile ultrasound, etc, and User Equipment (UE)s are conventional users of the mobile network such as smart phones. All nodes comply with LTE release-12 downlink and uplink communication [64]. In particular, each frame consists of 10 subframes of 1 milli-second duration. LTE resource grid consists of a number of RBs, where the RB is a collection of frequency subcarriers that spans two time slot duration (i.e., time slot = 0.5 msec). Each multiple contiguous RBs are combined to form one Resource Block Group (RBG). Let K be the number of RBG available for allocation. The RBG allocation process to attached users is performed each TTI (TTI = one subframe). Furthermore, power allocation is equal among RBGs and Almost-Blank SubFrame (ABSF) is used to minimize cross-tier interference. In particular, each tier (e.g., SBSs/eNBs) performs its uplink transmission in different subframes in an interleaved manner.

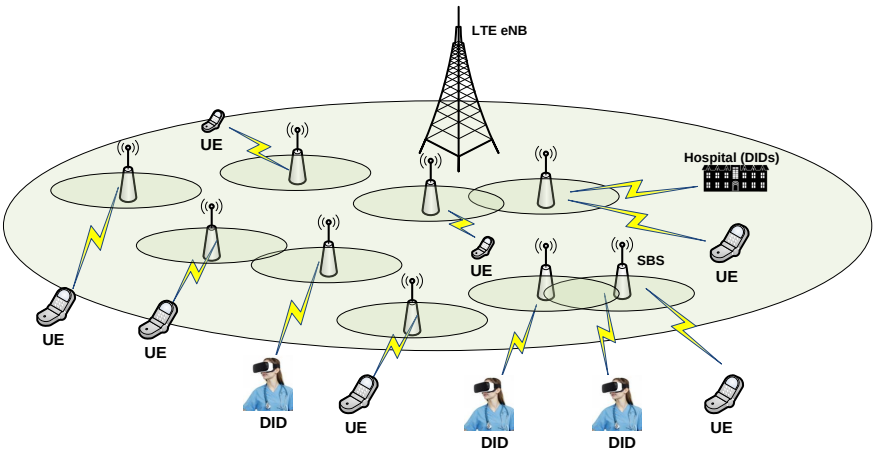

Figure 3.1: Data-intensive and tactile application users over small cell wireless networks.

3.2.2 Problem Formulation

We aim to improve the throughput of DID users by addressing the resource block allocation problem using Q-learning. The proposed algorithm based on a multi-agent Q-learning is performed by the base stations of each tier (i.e., on both eNB and SBSs). In particular, the rate between i^{th} user and j^{th} base station (i.e., $(i,j)^{th}$ link) can be formulated as:

$$C_{i,j} = \sum_{k=1}^{K} \omega_k \ \log_2 \Big(1 + \frac{x_{i,j,k} p_{i,j,k} \ h_{i,j,k}}{\omega_k \ N_0 \ + \sum\limits_{\substack{i' \in N_j \\ i' \neq i}} x_{i',j,k} \ p_{i',j,k} \ h_{i',j,k}}\Big), \tag{3.1}$$

where, $C_{i,j}$ denotes the total rate of i^{th} user attached to j^{th} base station. $x_{i,j,k}$ is a RB allocation indicator, where $x_{i,j,k} = 1$ denotes that k^{th} RB is allocated to i^{th} user that is attached to j^{th} base station. ω_k is the bandwidth of k^{th} RB. N_0 is the Additive White Gaussian Noise (AWGN) single-sided power spectral density. $p_{i,j,k}$ is the transmission power and $h_{i,j,k}$ is the channel coefficient on link (i,j) at k^{th} RB. $p_{i',j,k}$ and $h_{i',j,k}$ are the transmission power and the channel coefficient of i' interfering user, respectively.

The resource allocation can be formulated as an optimization problem that aims to maximize network rate:

$$\underset{x_{i,j,k}}{\text{Maximize}} \quad \sum_{j=1}^{J} \sum_{i=1}^{N_j} C_{i,j}. \tag{3.2}$$

However, optimization-based approaches are considered fit for centralized controller which does not scale well with the dynamic nature of the network. Therefore, we propose to use a

multi-agent reinforcement learning approach, in particular Q-learning. Indeed, Q-learning has the potential to reduce the computational complexity by searching for the optimal solution in an iterative approach. Furthermore, the proposed Q-learning algorithm is distributed and works independently on each base station. The detailed description of the proposed algorithm is presented in the following sections with a highlight on its main features.

3.2.3 Throughput-Maximizing Resource Allocation using Q-learning (TMQ)

The proposed algorithm, TMQ, utilizes Q-learning to maximize the throughput of DID and UE users. The agents running Q-learning are eNBs and SBSs. In particular, eNB performs TMQ to allocate RBs to its attached SBSs, whereas SBSs perform TMQ to allocate RBs to its attached users. Each agent estimates the link quality by utilizing the Channel State Information (CSI) feedback from its users.

TMQ is a multi-agent distributed Q-learning algorithm. TMQ's action, $a_{i,j,t}$, is defined as the RB allocation of i^{th} user attached to j^{th} base station at t^{th} TTI. As the number of users increases, the action-space dimension will increase significantly which leads to a curse-of-dimensionality problem. For instance, an SBS covering 10 users and performing allocation on 50 RBs will have 10^{50} actions to choose among. Instead, allocation can be performed in a group of contiguous RBs, namely RBG. In order to reduce complexity and improve the convergence, we consider a RBG of size 10 RBs. This significantly reduces the action-space to become $N^{(K/10)}$, where N is the number of users and K is the total number of RBs.

The reward function is formulated to maximize the rates of both DID and UE users as follows:

$$r_j = \beta\, r_d + (1 - \beta)\, r_u, \tag{3.3}$$

where r_j is the reward of j^{th} base station and β is a parameter to control the priority between DID and UE users. r_d and r_c are the rewards of DID and UE users which are defined as follows:

$$r_d = \left(\frac{2}{\pi}\right) \tan^{-1}(C_d), \tag{3.4}$$

$$r_u = \left(\frac{2}{\pi}\right) \tan^{-1}(C_u), \tag{3.5}$$

where C_d and C_u are the rates of DID and UE users, respectively. The reward function in (3.3) aims at maximizing both the DID and UE rates while giving higher priority to the critical load by increasing the parameter β. The Q-value is updated as in (2.4) [9].

The performance of TMQ is compared to PF, where PF allocates RBs to the users that have maximum relative channel conditions, with an intent to have fairness on the long-run [5].

3.2.4 Performance Evaluation

Our simulations are performed using Matlab LTE toolbox. Our settings incorporate one eNB with a radius of 800 meters, covering 10 SBSs, each with 50 meters radius [65]. In all simulation figures, a fixed number of 5 UEs per SBS is considered, while number of DIDs spans the range $[4 : 2 : 12]$. The Third Generation Partnership Program (3GPP) pathloss model is used [66]: $PL_{dB} = 128.1 + 37.6 \log(d)$, where d is the distance in Km between the base station and the user. Shadowing is drawn from a log-normal distribution of zero-mean and 8 dB variance while penetration loss is set to 20 dB [67] and noise is set to 5 dBm. We consider two traffic types. For devices running tactile applications we adopt the Beta distribution as defined in 3GPP for MTC [68], and the traditional user traffic is modeled as Poisson distribution with inter-arrival time 5 ms. Simulations are performed and averaged for 5 runs. Table 3.1 summarizes the simulation settings.

The performance is analyzed in terms of average throughput, average packet delay, average queuing delay, and fairness. We define delay as the total transmission delay from a user to an eNB, starting from the packet generation time. The queuing delay is the aggregate waiting time the packet experiences throughout its transmission (i.e., waiting in the device queue, and waiting in SBS queue).

Fig. 3.2 and 3.3 present the average and peak throughput versus number of DIDs. Fig. 3.2a and 3.3a show the throughput of DIDs. It can be seen that TMQ outperforms PF both in average and peak throughput. Meanwhile, TMQ improves the throughput of DIDs without compromising the throughput of UEs. As seen in Fig. 3.2b and 3.3b, UE throughput is also higher than the case with PF.

Fig. 3.4 presents the average packet delay in milli-seconds versus the number of DIDs. As seen from the figure, TMQ achieves the lowest total packet delay. On the other hand, Fig. 3.5 shows the average queuing delay experienced by both algorithms. The queuing delay is a direct outcome of the scheduling time, where it constitutes the time the user waits for getting a RB allocation from its base station. This result reflects that most of the packet delay comes from the scheduling time, which was significantly improved using

Table 3.1: Simulation settings

Physical layer	
Bandwidth	10 MHz
Modulation Schemes	Quadrature Phase Shift Keying (QPSK), 16-Quadrature Amplitude Modulation (QAM), 64-QAM
Number of RBs	50
Resource Block Groups	10
eNB power transmit	46 dBm
SBS power transmit	20 dBm
Pathloss model	3GPP $PL_{dB} = 128.1 + 37.6 \log(d)$
Penetration loss	20 dB
Noise figure	5 dB
Shadowing	$\sim$ LOGN(0, 8 dB)
Network	
Number of eNBs	1
Number of SBSs per eNB	10
Number of DIDs per SBS	4:2:10
Number of UEs per SBS	5
eNB radius	800 m
SBS radius	50 m
Min distance between SBSs	30 m
Traffic	
Traffic arrival model	DIDs: Beta [68] UEs: Poisson
Packet mean Inter-arrival time	5 milli-seconds
Packet size	Exponential (mean = 25 Bytes)
TMQ	
α (Learning rate)	0.5
γ (Discount factor)	0.9
β (Priority weight of DIDs)	0.9
ϵ (Exploration probability)	0.2
Simulation time	500 TTIs
Confidence Interval	95%

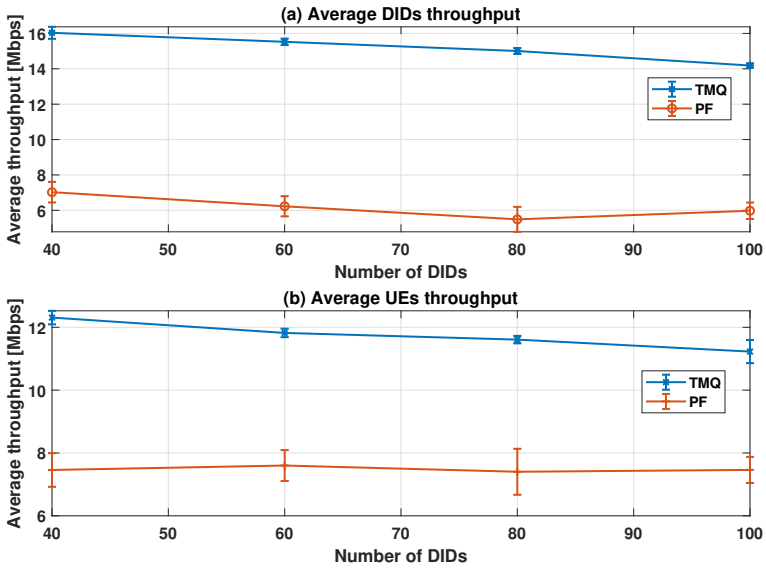

Figure 3.2: Average throughput for (a) DIDs and (b) UEs (10 SBS, 5 UEs per SBS)

the TMQ algorithm. As seen from the figures, TMQ achieves 80% decrease in delay for the highly-dense scenario (i.e., number of DIDs = 100). In addition, both algorithms do not incur any outage. It is worth noting that the achieved delay is still higher than the QoS requirements of tactile applications. Therefore, we devote the next section to studying latency requirement of delay-sensitive applications, where a microgrid scenario is considered.

To study fairness of TMQ, Jain's fairness index is plotted in Fig. 3.6. As the figure shows, TMQ outperforms PF, which is a direct product of applying a reward function that maintains fairness between DIDs and UEs.

Finally, the results presented here have converged after 200 TTIs (i.e., 200 msec). This is considered a high convergence time for tactile applications. However, this time could be saved if the training is performed offline. On the other hand, in an online setup, a more advanced technique is needed to address the adaptability of the algorithm. In section 5, we present how the speed of training and network adaptability can be boosted with the use of transfer in reinforcement learning.

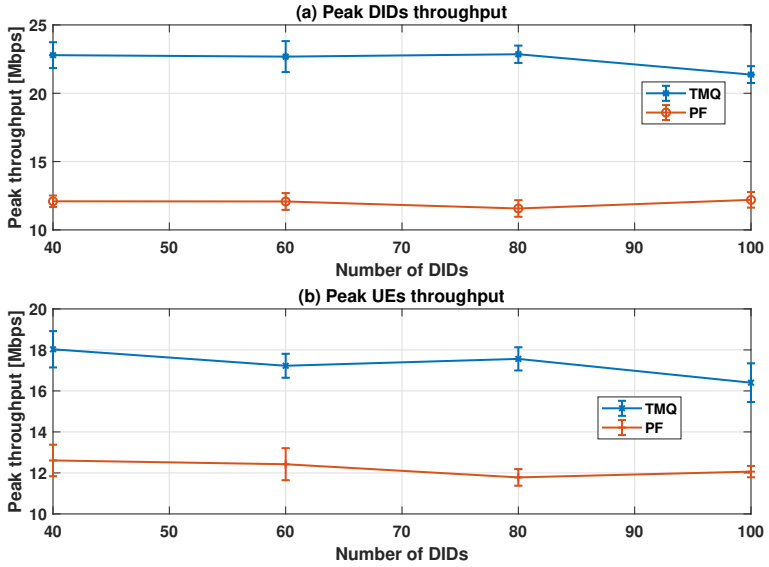

Figure 3.3: Max-user throughput for (a) DIDs and (b) UEs (10 SBS, 5 UEs per SBS)

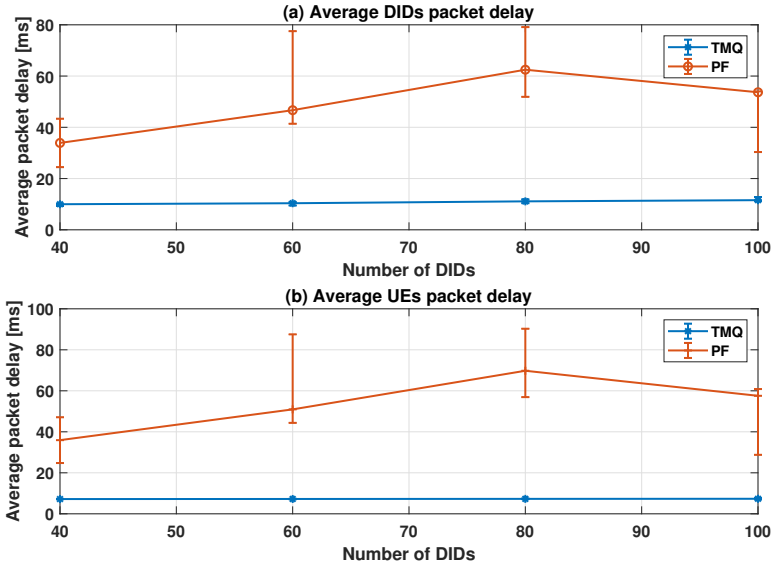

Figure 3.4: Average packet delay [ms] for (a) DIDs and (b) UEs (10 SBS, 5 UEs per SBS)

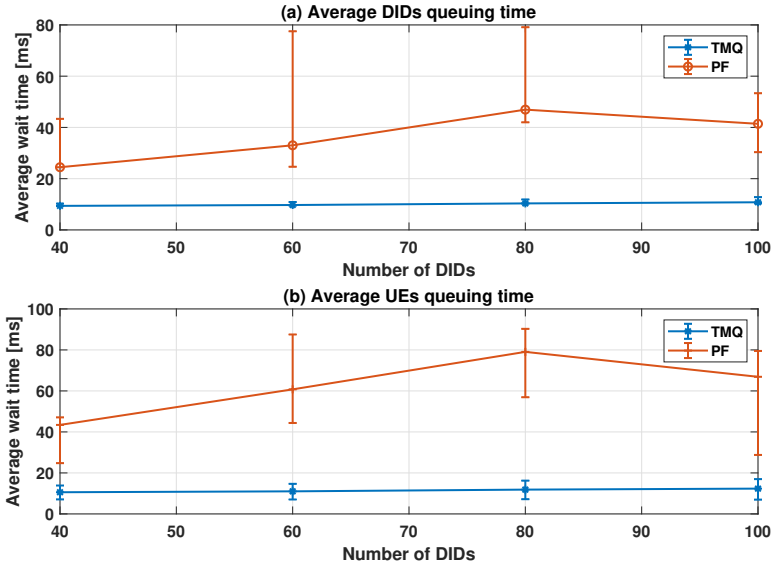

Figure 3.5: Average queuing time [ms] for (a) DIDs and (b) UEs (10 SBS, 5 UEs per SBS)

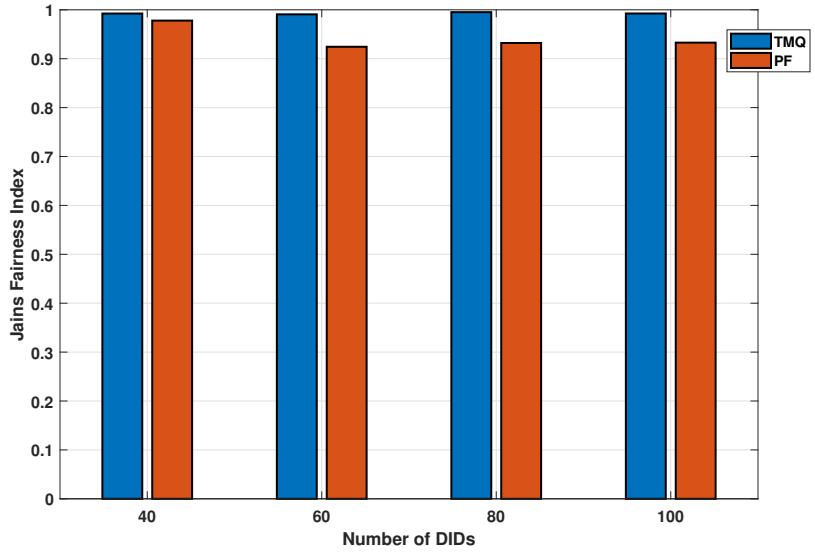

Figure 3.6: Jain's fairness index (10 SBS, 5 UEs per SBS)

3.3 Low-latency Communications for Community Resilience Microgrids: A Reinforcement Learning Approach

The reward signal of a reinforcement learning algorithm constitutes a utility/objective for guiding the system towards the desired performance. In the previous section, TMQ prevailed in achieving higher throughout and lower latency. However, the latency achieved was standing above the QoS requirement of the tactile application. In this section, we address the problem of resource allocation with the aim to minimize network latency and improved fairness, where a Community Resilience Microgrid (CRM) is considered. Indeed, the primary control mode of CRMs, which operates on a millisecond-level, constitutes a typical example of a service with stringent latency requirements.

Q-learning-based resource allocation algorithm, namely Delay-Minimization Q-learning (DMQ), is proposed. A salient feature of DMQ is its ability to capture network dynamics of CRM without *a priori* information. In addition, the proposed algorithm is fully distributed which promotes independent learning, and lowers signaling overhead in the network. Moreover, the design of the reward function achieves both low-latency and high fairness among Micro-Grid Device (MGD)s and UEs. Finally, the integrated design of Q-learning and two-tier small cell network allows for great flexibility in deployment and fast network adaptability. Performance results show 33% and 66% latency reduction for microgrid load when compared to previously proposed Distributed Iterative Resource Allocation (DIRA), which is an optimization based solution, and the traditional PF algorithm, respectively. Meanwhile, a significant improvement in throughput and fairness is also achieved by the proposed scheme which makes DMQ tailored for the connected microgrids of the future smart grid. It is worth mentioning that our approach can be adopted to other latency critical applications.

3.3.1 Community Resilience Microgrid

An increasing frequency of catastrophic weather events has been observed recently in the United States and globally, which has brought serious social and economic impacts. A critical issue associated with such catastrophic events is the availability of electricity for recovery efforts [69]. The smart grid is expected to heal itself under extreme circumstances [70]. In response to this, CRMs have been sought for enhancing resilient electricity supply to critical loads in a community during such disruption events. A CRM is a microgrid that is

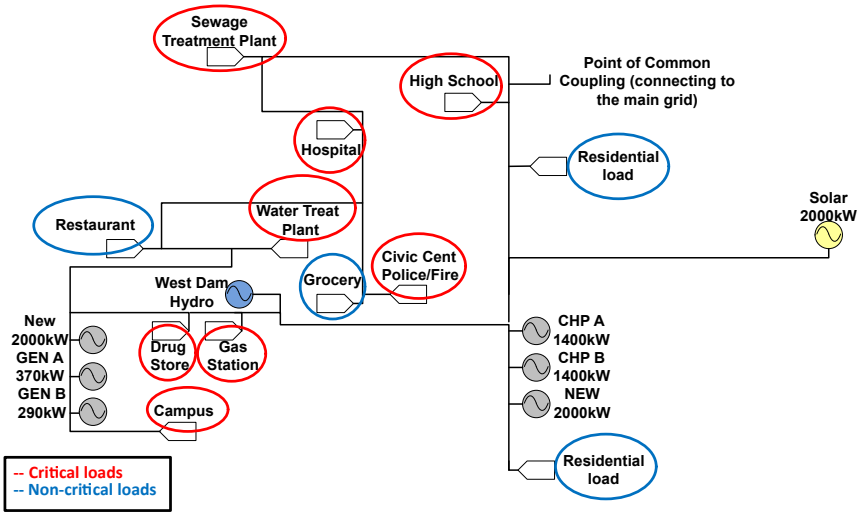

Figure 3.7: Conceptual design of a CRM with critical and non-critical loads.

expected to supply electricity uninterruptedly during the damage phase and initial recovery period of a resiliency event. As shown in Fig. 3.7, a CRM includes multiple distributed energy resources and critical loads that are owned/controlled by different entities within a clearly defined electrical boundary, which are connected via primary distribution lines owned by a local regulated power company [71–73]. However, CRMs, as complex networked systems, exhibit unique structure and bring new challenges for the operation and control. The methods used for control of standalone microgrids, such as droop control [74], need to be tailored to CRMs. Specifically, as CRMs present a variety of dynamical behaviors ranging from minutes and hours to milliseconds, such as real-time uncertain loads and renewable generation outputs, maintaining reliable operation of CRMs in terms of voltage and frequency stability calls for real-time response and control. Consequently, a three-level hierarchical control architecture, including hour-to-minute-level tertiary control, second-level secondary control, and millisecond-level primary control, is usually deployed to realize secure and cost-effective operation and coordinate multiple partners in CRMs.

The key of the hierarchical control strategy is to effectively integrate the three control levels at different timescales. In summary, different control levels would have distinct communication delay tolerance, and an efficient RRM and user scheduling approach is needed to optimally customize communication traffics, resource allocation, and delays of different needs. In here, we focus on the primary control as it poses the most stringent latency requirements. For example, in [75], latency requirements of substation automation is stated to be less than 100 ms. Our simulation results verify the suitability of the proposed

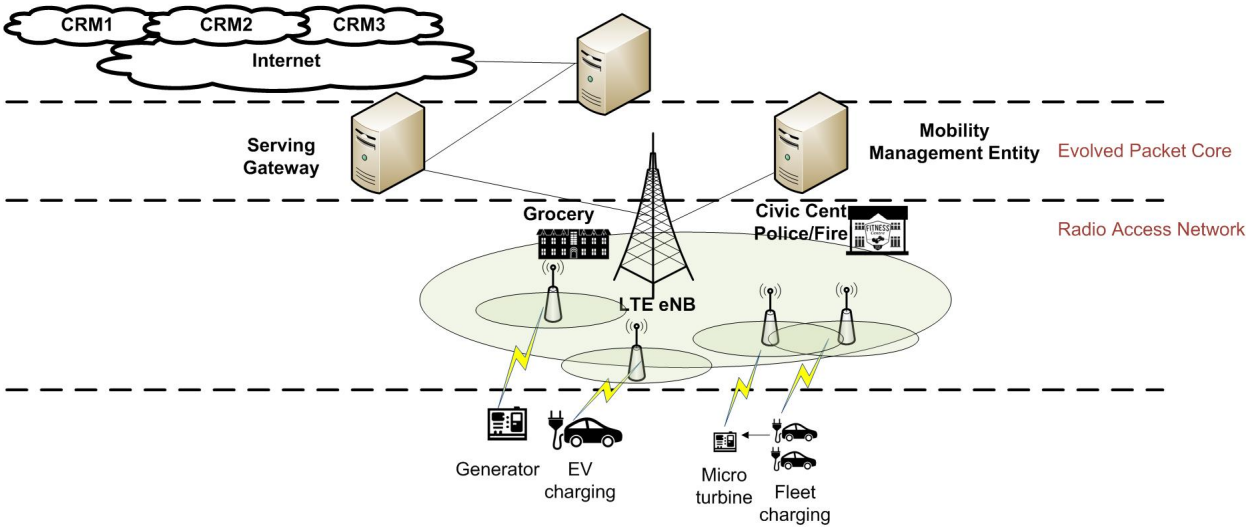

Figure 3.8: A minimalist illustration of CRM communications over small cell wireless networks. A two-tier wireless network of an eNB underlaid with SBS covering users and the evolved packet core.

algorithm by achieving latency values less than 50 ms under the worst-case scenario.

3.3.2 System Model

Our network model considers a two-tier network of an eNB underlaid with $M \in \mathcal{M}$ SBSs covering N_m user devices. Let $J \in \mathcal{J}$ be the set of base stations that includes eNBs and SBSs. In general, users can be classified as either MGDs such as smart meters, micro-phasor measurement units, etc, or conventional LTE UEs such as smart phones, tablets, etc. All nodes follow the downlink and uplink communication according to LTE release-12 standard. Let K be the number of RBGs available for allocation. Each base station performs a RBG allocation every TTI, where RBG is the unit of allocation that spans one TTI in the time direction. The problem at hand translates to a RBG allocation, whereas power allocation per RBG is considered to fixed.

In Fig. 3.8, all nodes conform to FDD with single antenna transmission. To remove cross-tier interference, we decompose the uplink (downlink) transmissions of both tiers, which is explained as follows. Uplink (downlink) of the links (users-SBS) and (SBS-eNB) use interleaving subframes to transmit their data. For example, the uplink communication of users uses subframes (1, 3, 5, .. $2i + 1$, .., $2n + 1$), while the uplink communication of

SBS uses subframes (0, 2, 4, ...i.,...$2n$). Although this succeeds to remove the cross-tier interference, co-tier interference still remains due to the dense deployment of SBSs, which causes users attached to one SBS and lying in the range of other SBS to cause interference in the adjacent cells.

Resource allocation process is performed by identifying the best RBG in time and frequency domains for active users in the network. Both the eNB and individual SBSs perform resource allocation to allocate RBGs to their attached users each TTI. In each TTI, the users report their scheduling request to their attached base station (i.e., users report to SBSs, and SBSs report to eNB). The base station performs the resource allocation algorithm and informs the users with the allocated RBGs to use in the next TTI. Performing the resource allocation on the two-tier network (i.e., eNB, and SBSs) reduces the burden on the eNB, as well as facilitates small cells for capacity and coverage improvement.

The wireless channel between nodes can be prone to multiple fading sources. We use the 3GPP pathloss model following [76–78]. $PL_{dB} = 128.1 + 37.6\log(d)$, where PL_{dB} is pathloss in dB, and d is distance between the base station and the user in km [66]. The shadowing effect is modeled as a log-normal distribution with zero-mean and 10 dB variance. Furthermore, we use the traffic model proposed by 3GPP TR 37.868 for MTC [68], where the traffic follows Beta distribution.

3.3.3 Problem Formulation

Delay on the link between i^{th} user and j^{th} base station (i.e., link (i, j)) can be formulated as follows:

$$D_{i,j} = D^{tr}_{i,j} + D^{q}_{i,j}, \tag{3.6}$$

where $D^{tr}_{i,j}$ is transmission delay, and $D^{q}_{i,j}$ is queuing delay on link (i, j). Eq. (3.7) formulates the transmission delay on link (i, j), where $\xi_{i,j}$ is packet size and $C_{i,j}$ is transmission rate. Delay and transmission rate can be formulated as follows:

$$D^{tr}_{i,j} = \frac{\xi_{i,j}}{R_{i,j}}, \tag{3.7}$$

$$C_{i,j} = \sum_{k=1}^{K} x_{i,j,k}\, c_{i,j,k}, \tag{3.8}$$

and

$$c_{i,j,k} = \omega_k\, \log_2\Big(1 + \frac{x_{i,j,k}\, p_{i,j,k}\, h_{i,j,k}}{\omega_k\, N_0 \; + \sum\limits_{\substack{i' \neq i \\ i' \in \mathcal{N}_j}} x_{i',j,k}\, p_{i',j,k}\, h_{i',j,k}}\Big), \tag{3.9}$$

where $c_{i,j,k}$ is rate on k^{th} RBG, $\mathcal{N}_j$ is the set of users of j^{th} base station, $x_{i,j,k}$ represents RBG allocation indicator, ω_k is bandwidth of k^{th} RBG, N_0 is AWGN single-sided power spectral density, $p_{i,j,k}$ is transmit power of i^{th} user on k^{th} RBG, $h_{i,j,k}$ is channel coefficient of k^{th} RBG, and $p_{i',j,k}$ is transmit power of i' interfering user on k^{th} RBG of j^{th} base station.

3.3.4 Delay minimization using Q-learning (DMQ)

DMQ is a decentralized algorithm where multiple agents (i.e., SBSs/eNB) aim at learning a sub-optimal decision policy by taking actions and computing feedback from the environment. The algorithm is represented by the tuple $\{s, a, r, s'\}$. The convergence point is reached when each agent learns a policy that maximizes its reward over infinite time horizon. This can be realized by having a Quality value (i.e., Q-value) representing the agents' reward over iterations. Hence, the optimal decision would be actions corresponding to the maximum reward (i.e., max Q-value). DMQ is defined as follows:

Agents: The macro-cell / small-cell base stations (i.e., eNB/SBSs) form the set of agents.

States: Agents perform a search to find the best resource allocation vector for its attached users. To this end, the number of states is limited to one.

Actions: Each base station (eNB/SBS) performs RBG-to-user mapping for its attached users on uplink. Hence, we denote $a_{j,t}$ as the decision of j^{th} base station at t^{th} TTI regarding the set of RBGs allocated to its attached users. Consequently, the dimension of the action space is N^K, where K is the total number of RBGs in a subframe and N is the number of users. The use of a RBG instead of RB as the allocation unit reduces the action space and helps the algorithm converge fast. Lastly, ϵ-greedy is used to account for action-space exploration.

Reward: We define the reward function as follows:

$$r_j(s_{j,t}, a_{j,t}) = \beta\, r_{j,c} + (1 - \beta)\, r_{j,n}, \tag{3.10}$$

where, β is a scalar weight to control priorities of individual loads (i.e., traffic from MGD and UE), $r_{j,c}$ and $r_{j,n}$ are defined as follows:

$$r_{j,c} = \left(\frac{-2}{\pi}\right) \tan^{-1}(D_{j,c}), \tag{3.11}$$

$$r_{j,n} = \left(\frac{-2}{\pi}\right) \tan^{-1}(D_{j,n}), \tag{3.12}$$

Algorithm 3.1 Delay minimization using Q-learning DMQ

Initialization: Q-Table $\leftarrow$ 0, ϵ, and T (Simulation time).
while $t < T$ **do**
  if $rand \leq \epsilon$ **then**
    // Random action selection
    $a_{j,t} \leftarrow$ arg rand$\{\mathcal{A}_j\}$
  else
    // Select action using greedy policy
    $a_{j,t} \leftarrow \arg\max_{a'_j \in \mathcal{A}_j} q(s_{j,t}, a'_j)$
  end if
  Calculate the reward using (3.10).
  Update $q(s_{j,t}, a_{j,t})$ using (2.4).
  Advance time t.
end while

where, $D_{j,c}$ and $D_{j,n}$ are the average delays of critical and non-critical loads, respectively. This function rewards the critical load delay with a positive reward as long as the achieved average delay is low. At the same time, it aims at minimizing delay of non-critical loads in order to maintain fairness among users.

Q-value: The Q-Value is updated as in (2.4).

The algorithm works in a two-tier scheduling approach on both the eNB and SBSs. That is, the eNB represents the first tier agent, and its attached SBSs are considered as its *environment*. Each SBS constitutes the second tier agent, with its attached users as the environment. SBS/users report their channel state information to eNB/SBS, respectively, on the uplink transmission. The channel state information enables the eNB/SBS to estimate the link quality of the allocated RBs thanks to the Channel Quality Indicator (CQI), Signal-to-Interference-plus-Noise Ratio (SINR), and total delay of the previous packet included in the channel state information. Algorithm 3.1 presents the Q-learning steps performed by each j^{th} agent. Algorithm 3.1 is repeated for the entire simulation time (T), during which the algorithm either performs exploration (i.e., random action selection) or exploitation (i.e., select the action with the maximum Q-value). The same algorithm runs on both the eNB and SBSs, where the eNB is responsible of scheduling SBSs on the uplink.

3.3.5 Baseline 1: Proportional Fairness

PF is a well-known algorithm that aims to give priority to the user having the maximum relative channel condition. The utility function is formulated as

$$i^* = \arg\max_{i \in \mathcal{N}_j} \frac{(C_{i,j,k}(t))}{(\overline{C}_{i,j,k}(t))}, \tag{3.13}$$

where $C_{i,j,k}(t)$ is the instantaneous rate of i^{th} user attached to j^{th} base station on k^{th} RBG at t^{th} TTI, $\overline{C}_{i,j,k}(t)$ is the moving average rate of i^{th} user [79], [80], and i^* is the user achieving the highest relative channel conditions. The moving average rate can be computed as in [81]:

$$\overline{C}_{i,j,k}(t+1) = \begin{cases} (1 - \frac{1}{t_w})\ \overline{C}_{i,j,k}(t) + \frac{1}{t_w}\ C_{i,j,k}(t), & i^* = i, \\ (1 - \frac{1}{t_w})\ \overline{C}_{i,j,k}(t), & i^* \neq i, \end{cases} \tag{3.14}$$

where t_w is the length of a history window.

3.3.6 Baseline 2: Distributed Iterative Resource Allocation (DIRA)

We compare our proposed scheme with an optimization-based solution that targets delay-sensitive users similar to our work [82]. To make DIRA comparable to our scheme, we slightly modify the original algorithm to consider only RBG allocation and omit power allocation. Furthermore, to have a fair comparison, we run DIRA on both network tiers (i.e., at the eNB and SBSs). To the best of our knowledge, the literature lacks an algorithm that both considers two-tier architecture and targets low-latency while tackling the resource block allocation problem. Therefore, DIRA is chosen to compare our results to a baseline solution that aims to provide low latency. Resource block allocation with DIRA can be

formulated as follows:

$$\max_{x_{i,j,k}} \quad \sum_{j=1}^{J} \sum_{i=1}^{N_j} \sum_{k=1}^{K} x_{i,j,k}\, c_{i,j,k} \tag{3.15}$$

subject to:

$$\sum_{k=1}^{K} x_{i,j,k}\, p_{i,j,k} \leq P_{max}, \forall j, i \tag{3.15a}$$

$$p_{i,j,k} \geq 0, \forall i, k \tag{3.15b}$$

$$\sum_{k=1}^{K} x_{i,j,k}\, c_{i,j,k} \geq C_0, \forall i \in MGDs, \forall j \tag{3.15c}$$

$$\sum_{i=1}^{N_j} x_{i,j,k} \leq 1, \forall j, i \tag{3.15d}$$

$$x_{i,j,k} \in 0, 1, \forall j, i, k \tag{3.15e}$$

where C_0 is the aggregate spectral efficiency threshold of MGDs in each base station, P_{max} is the maximum transmission power of i^{th} user. Eq. (3.15) aims to maximize the aggregate network rate through RBG allocation. Eq. (3.15a) limits the power allocation of each i^{th} user to P_{max} on all of its RBGs. Eq. (3.15c) guarantees a minimum achievable spectral efficiency, C_0, to each i^{th} user. Eq. (3.15d) and (3.15e) guarantee that each RB can only be assigned to one user within each cell. Following the same derivation methodology presented in [82], the following formula can be obtained:

$$H_{i,j,k} = (1 + \hat{\nu}_{i,j}) r_{i,j,k} - \theta_{i,j}\, p_{i,j,k} - (1 + \hat{\nu}_{i,j})\, \frac{1}{\ln(2)} \left(\frac{p_{i,j,k}\, h_{i,j,k}}{p_{i,j,k}\, h_{i,j,k} + I_{i,j,k}} \right), \tag{3.16}$$

where $I_{i,j,k} = p_{i',j,k} h_{i',j,k} + \sigma^2$ is the interference on link (i, j) on RB k.

$$\hat{\nu}_{i,j} = \begin{cases} \nu_{i,j}, & \forall i, j \in \text{MGDs}, \\ 0, & \text{otherwise}, \end{cases} \tag{3.17}$$

where $\nu_{i,j}$, and $\theta_{i,j}$ are Lagrangian multipliers obtained using the subgradient method. Hence, k^{th} RBG is assigned to the user with the largest $H_{i,j,k}$ as follows:

$$\hat{x}_{i^*,j,k} = 1|_{i^* = \max_i H_{i,j,k}}, \quad \forall j, k, \tag{3.18}$$

where $\hat{x}_{i^*,j,k}$ is the RBG allocation decision to selected user i^*.

3.3.7 Performance Evaluation

We use the LTE system toolbox in Matlab to design a discrete-level simulator for our network setup. Table 3.2 summarizes simulation settings used in the evaluation of the proposed and baseline algorithms. The simulation considers one eNB covering 20 SBSs, where eNB and SBS radii are 800m and 50m [65, 65], respectively. The pathloss model is 3GPP model with penetration loss is 20 dB, and noise figure is 9 dB [67]. DMQ uses a learning rate α of 0.5, a discount factor γ of 0.9, and ϵ of 0.8 [83]. All results are averaged over 5 testing runs, where each run is 500 subframes (i.e., 500 msec). A 95% confidence interval is provided in all our simulation results.

Fig. 3.9 presents the average packet delay versus the number of MGDs with 10 SBSs and 5 UEs per SBS. DMQ achieves the lowest transmission latency for both MGDs and UEs. Although the delay increases with the increase in the number of MGDs, as expected, DMQ still achieves the lowest delay compared to the other algorithms. Fig. 3.10 presents the average queuing delay for MGDs and UEs. This accounts for time that the packets have to wait until the RBGs allocated to them become available. DMQ achieves the lowest queuing delay, with some degradation when increasing the number of MGDs. However, it still has the lowest delay trend. It is also observed that most of the end-to-end delay is due to queuing delay.

Fig. 3.11 presents the average throughput versus number of MGDs. The results show that DMQ outperforms DIRA and PF. However, increasing the number of MGDs/SBS degrades DMQ's throughput. The main reason behind this is that DMQ's main aim is to decrease the end-to-end latency of MGDs while maintaining fairness among MGDs and UEs. Therefore, as can be seen in Fig. 3.9, both MGDs and UEs delays are decreased, whereas this comes on the price of higher throughput degradation, especially in dense scenarios. In Fig. 3.12, we show the top-10 users' throughput, which again shows a better performance of DMQ. Yet, throughput of DMQ is impacted by the number of MGDs more than the other algorithms. Once again, for denser networks, throughput results converge since the available resources are limited.

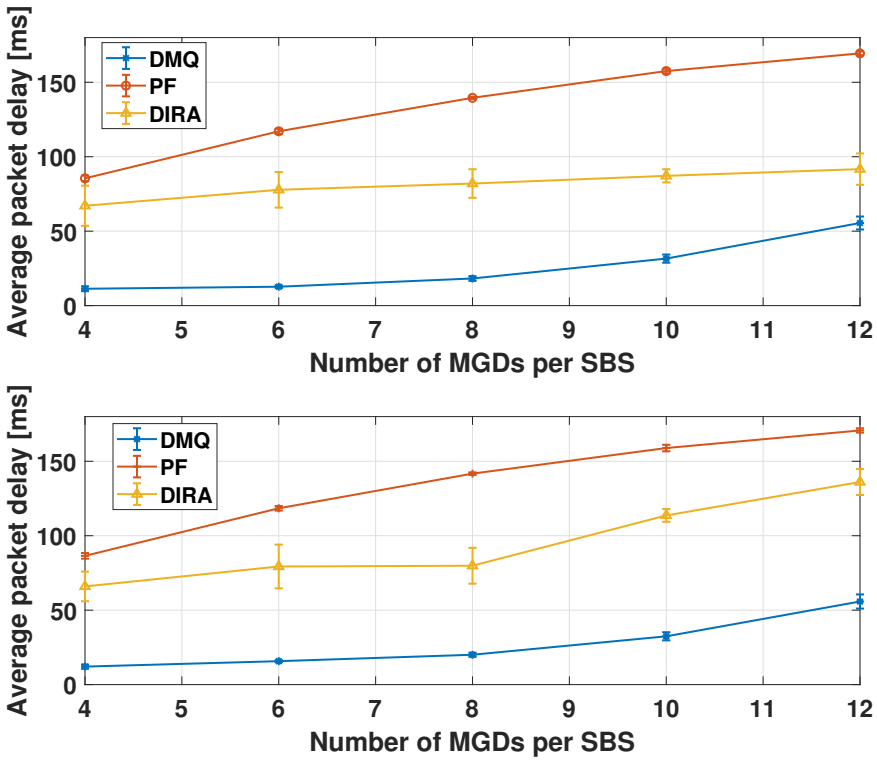

Figure 3.9: Average packet delay [ms] for (top) MGDs and (bottom) UEs vs number of MGDs; number of SBS is 10 and number of UEs is 50.

Table 3.2: Simulation settings

Network	
TTI	1 msec
Resource allocation algorithms	DMQ, PF and DIRA.
eNB radius	800 m
SBS radius	50 m
Min distance between SBSs	30 m
Number of eNBs	1
Number of SBSs per eNB	10
Number of MGDs per SBS	4:2:12
Number of UEs per SBS	5
Speed of users	Fixed positions
MGDs Traffic model	Beta ($a = 3, b = 4$) [68]
UEs Traffic model	Poisson
Packet mean Inter-arrival time	5 milli-seconds.
Packet size	Exponential (mean = 25 Bytes)
Transmission bandwidth	10 MHz
Number of RBs	50 (12 subcarriers / RB)
Number of RBGs	5 (10 RBs/RBG)
eNB Tx power	40 dBm [84]
SBS Tx power	20 dBm [84]
Pathloss model	3GPP $PL_{dB} = 128.1 + 37.6 \log(d)$
Penetration loss	20 dB
Noise figure	9 dB
Shadowing	$\sim$ LOGN(0, 10(dB))
Proportional Fairness	
t_w (window)	2
DMQ	
Learning rate (α)	0.5
Discount factor (γ)	0.9
Exploration probability (ϵ)	0.8
Priority weight of MGDs (β)	0.9
DIRA	
C_0	9 bps/Hz [82]

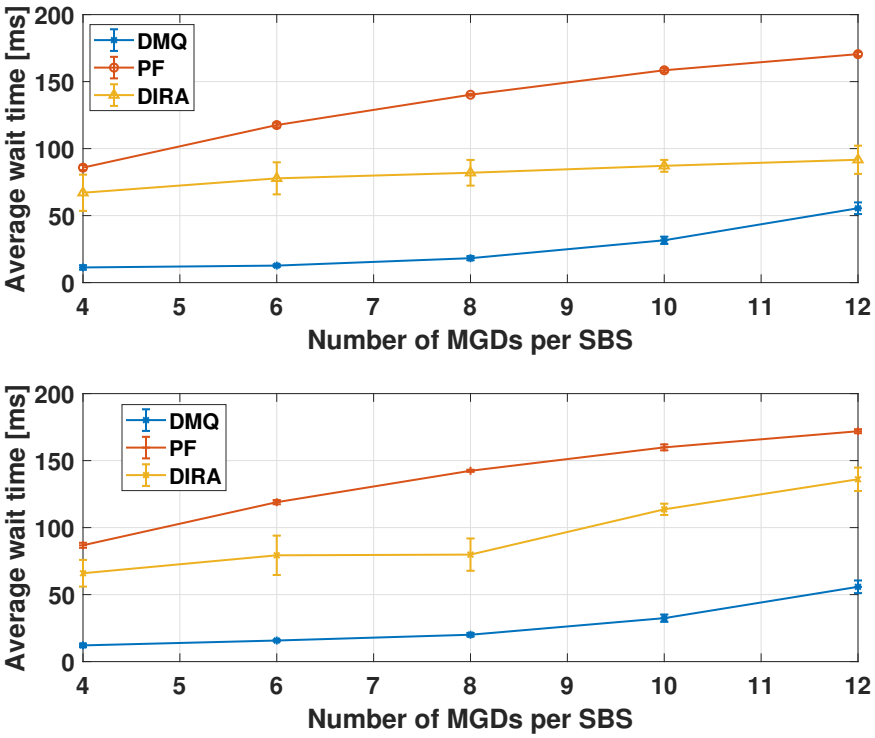

Figure 3.10: Average queuing delay [ms] for (top) MGDs and (bottom) UEs vs number of MGDs; number of SBS is 10, and number of UEs is 50.

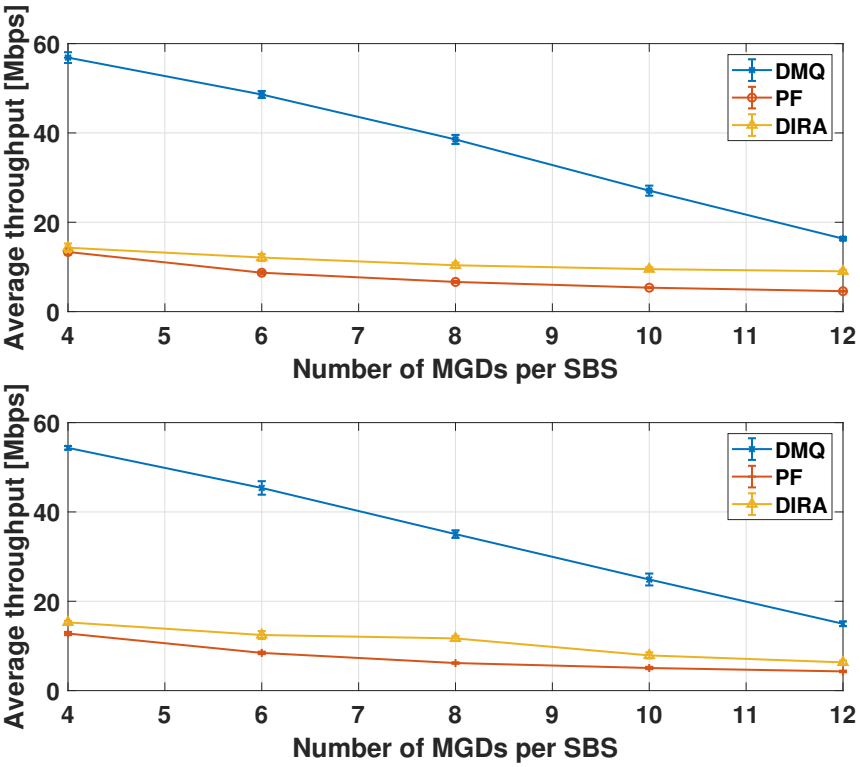

Figure 3.11: Average throughput [Mbps] for (top) MGDs and (bottom) UEs vs number of MGDs; number of SBS is 10, and number of UEs is 50.

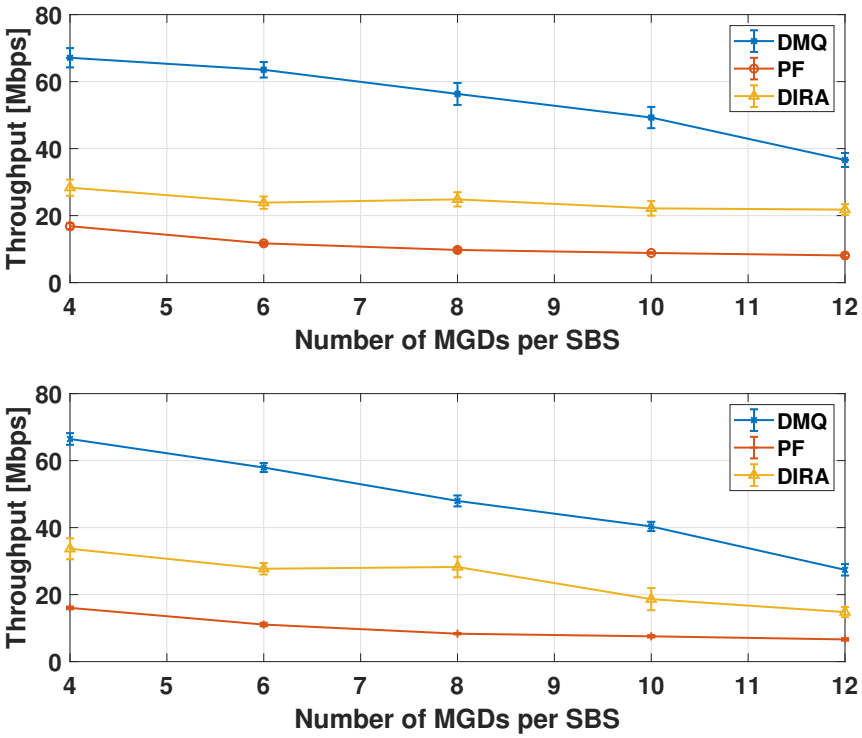

Figure 3.12: Top-10 throughput [Mbps] for (top) MGDs and (bottom) UEs vs number of MGDs; number of SBS is 10, and number of UEs is 50.

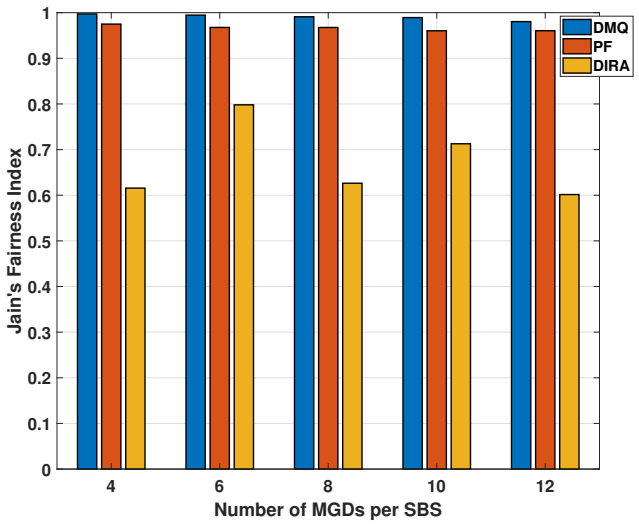

Figure 3.13: Jain's fairness index vs MGDs; number of SBS is 10, and number of UEs is 50.

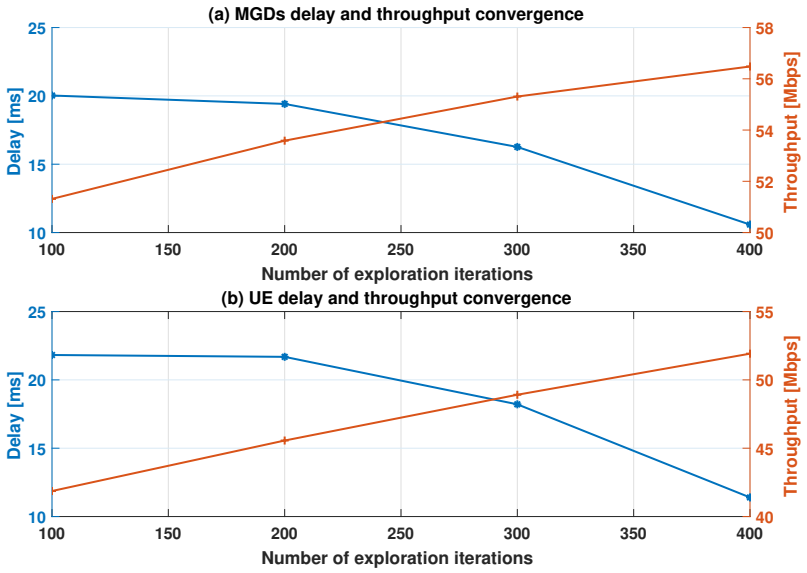

Figure 3.14: Average delay and throughput convergence for (a) MGDs, and (b) UEs vs number of exploration iterations (in TTIs); 10 SBSs, 8 MGDs and 5 UEs per SBSs.

Table 3.3: Comparison among the three algorithms

Criteria	**PF**	**DIRA**	**DMQ**
Resource allocated	Spectrum	Spectrum (Power removed)	Spectrum
Objective	Rate and Fairness	Rate and delay constraint	Delay
Network Model	two-tier	Adapted to two-tier	two-tier
Complexity (per TTI per BS)	$O(N_j)$	$O(N_j)$	$O(N_j)$

To study the fairness of DMQ, Jain's fairness index is plotted in Fig. 3.13. Since the reward function of DMQ aims to minimize UEs delay as well, it provides fairness among users. Our results show fairness values that exceed PF fairness by about 2%.

In summary, DMQ performs better than DIRA and PF, in terms of delay, throughput and fairness. However, its throughput degrades in a faster trend than DIRA and PF. As a trade-off, DMQ favors delay and fairness over throughput, which can be observed from the reward design in (3.10). Note that, the average latency and throughput performance of MGDs and UEs is close for DMQ, as well as for PF, since both algorithms have fairness in their objective. Yet, DMQ results in lower latency and higher throughput for both types of devices than the compared algorithms.

A comparison among the three algorithms is presented in Table 3.3, where N_j is the number of users per base station. The table presents the modeling assumptions as well as the complexity and drawbacks. DIRA was adopted to work on both tiers, furthermore, we revised the optimization to account for spectrum allocation only - removing the power allocation. The complexity is presented in Big-O notation, evaluated per base station per TTI.

Lastly, Fig. 3.14 presents the impact of a longer learning phase on both the delay and throughput results under the proposed DMQ scheme. It can be seen that performing more action-space exploration allows the algorithm to learn better resource allocation policy, hence the delay decreases and throughput increases at the same time. However, this also leads to the requirement of longer training time for improving performance. In the next section, we devote more attention to mission-critical services that require strict latency requirements and present a solution based on deep reinforcement learning. Indeed, the

use of deep reinforcement learning provides a capability of better generalization over the state-action space and helps in speeding up the convergence time.

3.4 Deep Reinforcement Learning for Reducing Latency in Mission-Critical Services

Although scheduling and resource allocation have been widely addressed in the literature [29, 83, 85–87], resource allocation for mission-critical applications with stringent QoS requirements brings in additional challenges that are not addressed by existing schemes. In the previous sections, reinforcement learning has been used for RRM in mission-critical applications, where it demonstrated several throughput and latency gains in addition to better fairness. However, with the increase of the state-action space, reinforcement learning poses severe convergence challenge. In particular, the method requires large number of exploration iterations to convergence to the optimal solution. Therefore, in this section, we pay more attention to the convergence problem of tabular methods and propose a deep reinforcement learning solution that provides better generalization over the state-action space with the help of a deep neural network that acts as a function approximator to the Q-values of Q-learning.

In particular, we propose a deep Q-learning-based algorithm for delay minimization of mission-critical services, namely Delay Minimizing Deep Q-learning (DMDQ). DMDQ is a deep reinforcement learning-based algorithm that uses LSTM neural network to approximate the Q-values of Q-learning. The LSTM neural network acts as a function approximator that helps in speeding up the convergence of Q-learning. Furthermore, DMDQ has the ability to find correlations in historical traffic loads in an online manner without prior initialization of Q-learning or offline training of LSTM. In particular, we utilize this feature to improve latency of mission-critical devices by letting the LSTM learn from prior traffic experiences in adjusting its network weights, which impacts future resource allocation decisions. Our results show a 30% improvement in the latency of MCDs in dense scenarios. Furthermore, DMDQ has a comparable fairness performance to Q-learning while achieving 60% speedup in convergence. However, the reduction of latency comes at an expense of about 10% throughput reduction, again in dense scenarios.

We pay attention to mission-critical services for their stringent QoS requirements. In particular, mission-critical services include safety applications in vehicles [88], medical control systems, real-time control of power systems, control of a swarm of drones for surveillance, disaster recovery and so on [89]. The typical aspect of mission-critical services is that

they serve domains that require high-reliability, low-latency and ubiquitous connectivity. 5G networks are designed with mission-critical services in mind [90], yet LTE infrastructure is expected to be in place for a number of years. Traditional LTE networks cannot provide the desired performance unless novel approaches are adopted. Recently, in parallel to 5G efforts, enhancing the performance of LTE networks is actively being researched.

Furthermore, We embrace the concept of densification and Heterogeneous Network (HetNet) environment with dense deployment of small cells [91], where Mission-Critical Device (MCD)s and traditional mobile UEs co-exist and the challenge becomes scheduling MCDs and UEs such that low-latency requirement of MCDs can be fulfilled.

3.4.1 System Model

We consider a HetNet consisting of one eNB and $M \in \mathcal{M}$ SBSs, as illustrated in Fig. 3.15. Let $J \in \mathcal{J}$ be the set of base stations. SBSs cover both mission-critical and non-critical loads, while the eNB covers non-critical loads only. All nodes adhere to the LTE standard downlink and uplink signaling as described in [64]. LTE resource grid consists of a number of RBs, where contiguous RBs can be grouped to form K RBGs. It is worth noting that all nodes obey the fixed transmit power strategy.

We consider three types of user devices: UEs and Users of eNB (UNB)s that deliver non-critical loads, and MCDs that carry critical loads. Both UEs and MCDs can be served by SBSs, while UNBs are considered to be only eNB's users. The scheduling process is performed each TTI by the base station, i.e., SBSs or the eNB. We assume that SBSs have wired connection with the backhaul. Therefore, SBSs do not perform uplink wireless transmission, and do not create cross-tier interference. The wireless channel is modeled by 3GPP pathloss model [66]: $PL_{dB} = 128.1 + 37.6 \log(d)$, where d is base station-to-user separation in km. Shadowing is modeled as a log-normal distribution with zero-mean and 8 dB variance.

Nodes adhere to two traffic models according to their profiles. Both UEs and UNBs traffic is modeled as Poisson arrivals. Meanwhile, MCDs generate traffic that complies with 3GPP TR 37.868 traffic model for MTC [68]. As such, packet arrivals of MTC devices are drawn from the distribution:

$$p(t) = \frac{t^{a-1}\,(T-t)^{b-1}}{T^{a+b-1}\,\beta(a,b)}, \quad a > 0, b > 0, \tag{3.19}$$

where $\beta(a,b)$ is the Beta function with shape parameters $a = 3$ and $b = 4$, and users are active between time $t = 0$ to $t = T$.

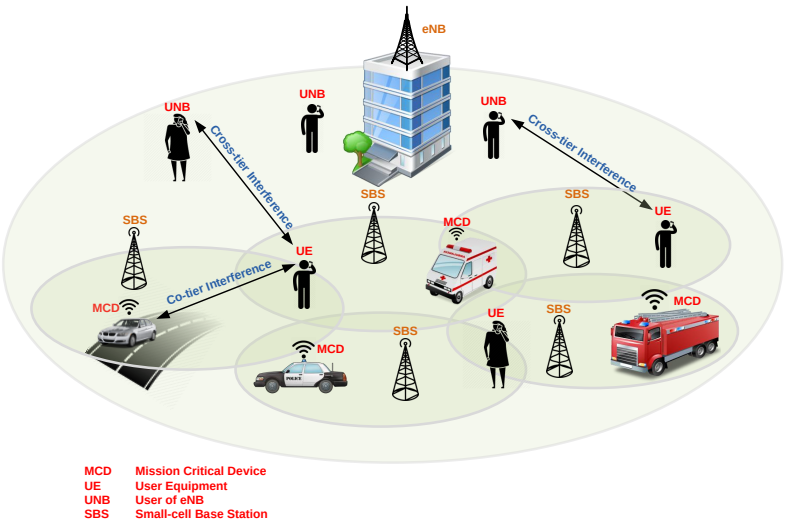

Figure 3.15: A small-cell wireless network covering mission critical and non-critical loads. SBSs cover both MCDs and UEs, while eNB covers UEs only.

3.4.2 Problem Formulation

End-to-end delay on link connecting i^{th} user and j^{th} base station (i.e., link (i, j)) can be formulated as:

$$D_{i,j} = D^{tr}_{i,j} + D^{q}_{i,j}, \tag{3.20}$$

where $D^{tr}_{i,j}$ is transmission delay on link (i, j), and $D^{q}_{i,j}$ is queuing delay on link (i, j). The transmission delay on link (i, j) can be formulated as follows:

$$D^{tr}_{i,j} = \frac{\xi_{i,j}}{C_{i,j}}, \tag{3.21}$$

where $\xi_{i,j}$ is packet size and $C_{i,j}$ is transmission rate. The transmission rate can be formulated as follows:

$$C_{i,j} = \sum_{k=1}^{K} x_{i,j,k}\, c_{i,j,k}, \tag{3.22}$$

where $x_{i,j,k}$ is an allocation indicator on link (i, k) for k^{th} RBG and $c_{i,j,k}$ is the rate on k^{th} RBG for the link between (i, j), defined as follows:

$$c_{i,j,k} = \omega_k \, \log_2 \left(1 + \frac{x_{i,j,k}\, p_{i,j,k}\, h_{i,j,k}}{\omega_k \, N_0 \, + \sum\limits_{i' \notin N_j} x_{i',j,k}\, p_{i',j,k}\, h_{i',j,k}}\right), \tag{3.23}$$

where N_j is the number of users attached to j^{th} base station, ω_k is bandwidth of k^{th} RBG and N_0 is AWGN single-sided power spectral density. $p_{i,j,k}$ is transmit power of i^{th} node on k^{th} RBG for j^{th} base station, $h_{i,j,k}$ is channel coefficient of k^{th} RBG, $h_{i',j,k}$ is the channel coefficient of i' interfering node, $x_{i',j,k}$ is the allocation indicator and $p_{i',j,k}$ is transmit power of interfering i' node. As such, we aim to minimize the delay as follows:

$$\min_{x_{i,j,k}} \quad \sum_{j=1}^{J} \sum_{i=1}^{N_j} \sum_{k=1}^{K} D_{i,j,k}. \tag{3.24}$$

To achieve this, we propose a distributed RBG allocation algorithm which is described in detail in the next section.

3.4.3 Delay Minimizing Deep Q-Learning (DMDQ) Scheme

DMDQ is a multi-agent distributed deep Q-learning algorithm that runs on the eNB and each SBS. DMDQ adopts deep reinforcement learning [92] which consists of two parts: A deep neural network and a reinforcement learning algorithm as shown in Fig. 3.16. We utilize LSTM as the deep neural network, and Q-learning as the reinforcement learning algorithm. We use an experience replay memory to store sequential Q-learning outcomes.

LSTM is a RNN that captures long-term dependencies between time steps in sequential data [93]. This is performed by storing information of long periods of time, where functions of neurons control how different information can be handled (i.e., memorized, erased, exposed). As shown in Fig. 3.16, LSTM is composed of four layers. The input layer is used to input time-series data into the network. In our model, we use Q-learning states as input to this layer, as will be explained shortly. The LSTM layer is the main layer for handling memorization, erasure, and exposure of data. Furthermore, a fully-connected layer is a multi-layer perceptron (MLP), where every neuron in one layer is connected to every neuron in another layer. Finally, the regression layer is used to compute numeric values of the Q-learning reward function value (i.e., Q-Value) for input state s, all actions $a \in \mathcal{A}$ and current network weights θ.

One drawback of reinforcement learning is its tendency to diverge when using a non-linear function approximator such as neural network [11]. To solve this, in [11], the use of an experience replay memory, in which network training is performed by sampling random experiences, is proposed. In our model, we define an experience as $e = \{s, a, r, s'\}$. This full experience can then be used in training the LSTM to refine the Q-value estimation at each iteration. In order to reduce time complexity, we perform the training every Ω TTIs, where $\Omega = 5$ in our simulations.

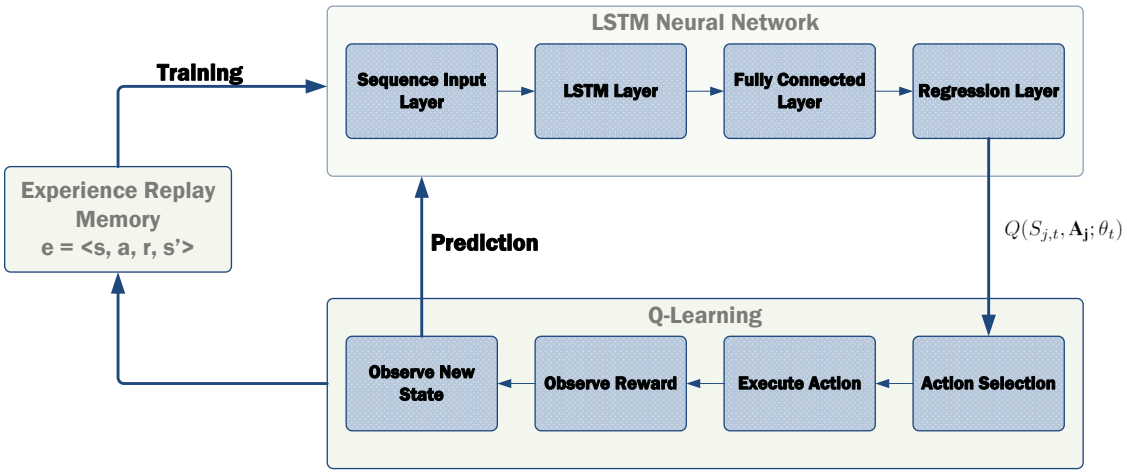

Figure 3.16: DMDQ Block Diagram

As shown in Fig. 3.16, the Q-learning part works in four steps: Action selection based on the approximated Q-values, action execution by the agent, calculating the reward, and observing the new state. The outcome of the Q-learning constitutes an experience that is stored in the experience replay memory for training purposes. In our model, the agents are represented by the SBSs and the eNB. We define the states to follow the utility function (i.e., delay) as follows:

$$s_{j,t} = \begin{cases} s_0, & D_{j,c,t} \leq D_0, \\ s_1, & \text{otherwise}, \end{cases} \tag{3.25}$$

where $s_{j,t}$ is the state of j^{th} base station at t^{th} TTI, $D_{j,c,t}$ is the average delay of critical loads (i.e., MCDs) of j^{th} base station at t^{th} TTI, and D_0 is a target delay to achieve. Consequently, state s_0 constitutes the desired agents' state. Furthermore, the reward is defined as follows:

$$r_j(s_{j,t}, a_{j,t}) = \left(\frac{-2}{\pi}\right) \tan^{-1}(D_{j,c,t}), \tag{3.26}$$

where $r_j(s_{j,t}, a_{j,t})$ is the reward of j^{th} base station at state $s_{j,t}$ and action $a_{j,t}$, and $D_{j,c,t}$ is the average critical delay computed as in (3.20) for all users of j^{th} base station at t^{th} TTI. In particular, the reward function aims at minimizing the latency of critical loads. Therefore, decisions with low packet delays are rewarded with small negative reward, while decisions with high packet delays are awarded with large negative reward (i.e., penalized). The update of the Q-value is performed using Bellman equation (2.4), whereas action selection follows ϵ-greedy strategy.

Algorithm 3.2 presents the complete steps of DMDQ scheme.

Algorithm 3.2 Delay Minimization Deep Q-Learning (DMDQ)

Initialization: Q-Table $\leftarrow 0$, α, γ, ϵ and LSTM parameters.
for TTI $t = 1$ to T **do**
  for base station $j = 1$ to J **do**
    Step 1:
    for User $i = 1$ to N_j **do**
      Receive the recent packet delay from i^{th} user of j^{th} base station.
    end for
    Step 2: Compute average packet delay $D_{j,t}$ for all MCDs of j^{th} base station.
    Step 3: Observe reward $r_j(s_{j,t}, a_{j,t})$ (3.26) and new state $s_{j,t+1}$ (3.25) due to last action execution.
    Step 4: Store current experience in the experience replay memory: $e_{j,t} = \{s_{j,t}, a_{j,t}, r_j(s_{j,t}, a_{j,t}), s_{j,t+1}\}$.
    Step 5: LSTM training: Draw experiences uniformly from the experience memory and perform LSTM training.
    Step 6: Transition to next state $s_{j,t+1}$.
    Step 7: LSTM prediction: Predict $q(s_{j,t+1}, a'_j)$ for all actions $a'_j \in \mathcal{A}_j$ at the next state $s_{j,t+1}$.
    Step 8: Select the next action $a_{j,t+1}$ based on ϵ-greedy policy.
  end for
end for

We compare DMDQ with DMQ algorithm which uses a traditional Q-learning approach without the integration of RNN. DMQ has been explained in section 3.3. As can be observed, DMDQ is different from DMQ through the incorporation of the LSTM stage, which works as a function approximator for the Q-values. We also use the Round Robin (RR) algorithm for comparison, which simply allocates all RBGs to each user in turn.

3.4.4 Performance Evaluation

Our system-level simulator is based on Matlab LTE system toolbox. Tables 3.4 and 3.5 summarize network and algorithms settings, respectively. Our simulation considers one eNB of radius 800m, and 5 SBSs of radius 50m [65]. The eNB covers 6 UNBs, each SBS covers 4 UEs and variable number of MCDs (from 4 to 10). Both DMDQ and DMQ use a learning rate $\alpha = 0.5$, a discount factor $\gamma = 0.9$, and $\epsilon = 0.8$ [83]. LSTM architecture uses four layers as shown in Fig. 3.16. For our simulation, adding more hidden units does not

Table 3.4: Network settings

Physical layer	
TTI	1 msec
Transmission bandwidth	10 MHz
Number of RBs	50 (12 subcarriers / RB)
Number of RBGs	5
eNB Tx power	40 dBm [84]
SBS Tx power	20 dBm [84]
Pathloss	3GPP pathloss model [67]
Penetration loss	20 dB
Noise figure	5 dB
Shadowing	$\sim$ LOGN(0, 8(dB))
Network	
eNB radius	800 m
SBS radius	50 m
Min distance between SBSs	30 m
Number of eNBs	1
Number of SBSs per eNB	5
Number of MCDs per SBS	4:2:10
Number of UEs per SBS	4
Number of UNBs per eNB	6
Traffic	
Beta traffic	$a = 3, b = 4$ (3.19) [68]
Poisson traffic	mean inter-arrival time = 5 ms
Packet size	Exponential (mean = 30 Bytes)

improve performance and thus 40 units has been found sufficient. LSTM input layer has a feature dimension of one, which we define as the Q-learning state. The target delay D_0 is set to 20ms.

Fig. 3.17 shows the MCDs average end-to-end delay. As observed from the figure, DMDQ outperforms DMQ, especially in dense scenarios (i.e., for MCDs from 30 to 50). On the other hand, RR incurs the highest delay. In between transmission delay and queuing delay, the latter contributes to the packet delay most significantly. It has been observed that average queuing delay follows the same trend in Fig. 3.17. At MCDs = 50, DMDQ delay becomes slightly higher than the target delay 20ms. This is reasonable since at this point, the network becomes dense in MCDs with heavy critical traffic loads. We show the

Table 3.5: DMDQ and DMQ settings

DMQ	
Learning rate (α)	0.5
Discount factor (γ)	0.9
Exploration probability (ϵ)	0.9
D_0 (Target Delay)	20 ms
DMDQ	
LSTM number of layers	4
LSTM hidden units	40
LSTM Fully connected layer units	40
LSTM initial learning rate	0.00001
LSTM batch size	5
LSTM feature dimension	1 (only state is input)
Experience replay memory size	30

Table 3.6: Average end-to-end delay of UEs

Delay [ms]				
MCDs per SBS	4	6	8	10
DMDQ	3.18	3.14	3.17	3.12
DMQ	3.15	3.09	3.12	3.14
RR	3.91	3.85	3.9	3.94

average delay of UEs in Table 3.6. Both DMDQ and DMQ achieve very close delay results, which are lower than Round Robin delay. Hence, DMDQ is able to reduce the delay of MCDs without impacting the end-to-end delay of traditional UEs.

The MCDs delay reduction comes with a trade-off of reduced throughput as shown in Fig. 3.18. This was expected since the designed reward function gives priority to reducing the latency of MCDs. However, as seen in Table 3.7, the average throughput for UEs is not impacted. In this case, RR has also comparable throughput to the proposed schemes.

We evaluate the fairness of RB allocation of DMDQ using Jain's fairness index as shown in Fig. 3.19. RR has the highest fairness since by definition it allocates resources to each of the users. Meanwhile both DMDQ and DMQ are quite fair to users with DMQ having a slightly higher fairness index.

In Fig. 3.20, we show the evolution of the accumulative reward of DMQ and DMDQ. For the sake of presentation, we plot the complement of the accumulative reward as $(1 -$

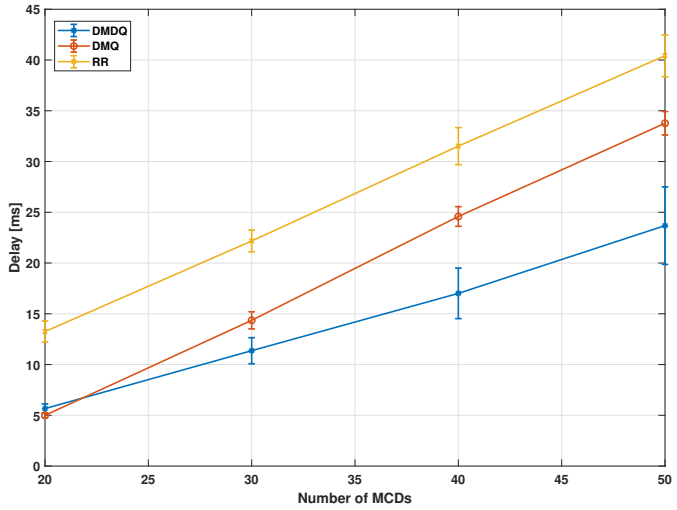

Figure 3.17: Average end-to-end delay of MCDs [ms]; number of SBS is 5, number of UEs is 20, and number of UNBs is 6.

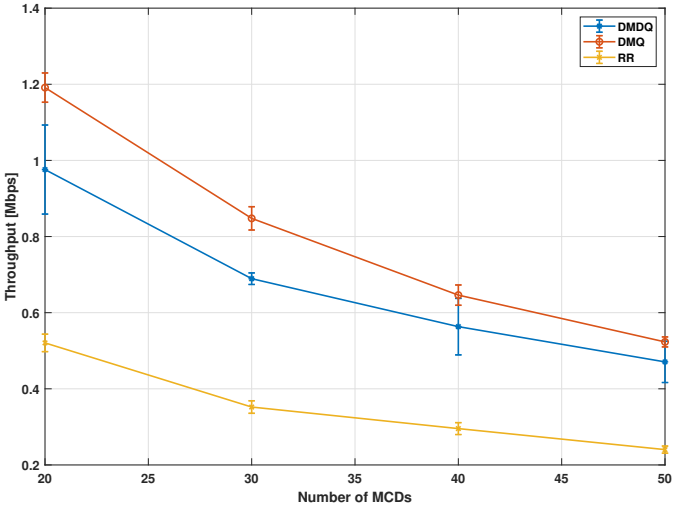

Figure 3.18: Average throughput of MCDs [Mbps]; number of SBS is 5, number of UEs is 20, and number of UNBs is 6.

Table 3.7: Throughput of UEs

Throughput [Mbps]				
MCDs per SBS	4	6	8	10
DMDQ	1.16	1.1	1.12	1.14
DMQ	1.13	1.13	1.09	1.11
RR	1.11	1.11	1.08	1.09

$\sum_{t=1}^{T} r_t/T$), where T is the subframe number (i.e., x-axis), and r_t is the average reward of all SBSs as calculated in (3.26). As can be seen in Fig. 3.20, DMDQ converges after 30 iterations while DMQ converges after 80 iterations.

In summary, the proposed DMDQ scheme reduces the latency of MDCs and it treats the users fairly while having a shorter convergence time than DQM. However, DQM offers higher throughput. Indeed, balancing throughput and latency constitutes an essential requirement in future wireless HetNets. We devote section 4.3 to multi-objective optimization, where latency and throughput are considered in a reinforcement learning setup. Furthermore, joint allocation of RRM tasks is considered as a key problem in RRM. We start by addressing joint allocation of RRM tasks in the next section, where joint spectrum allocation and user-cell association is considered with the aim to minimize network latency. Then, in section 4.3, we employ both joint allocation of RRM tasks and multi-objective optimization to improve KPIs of different network users.

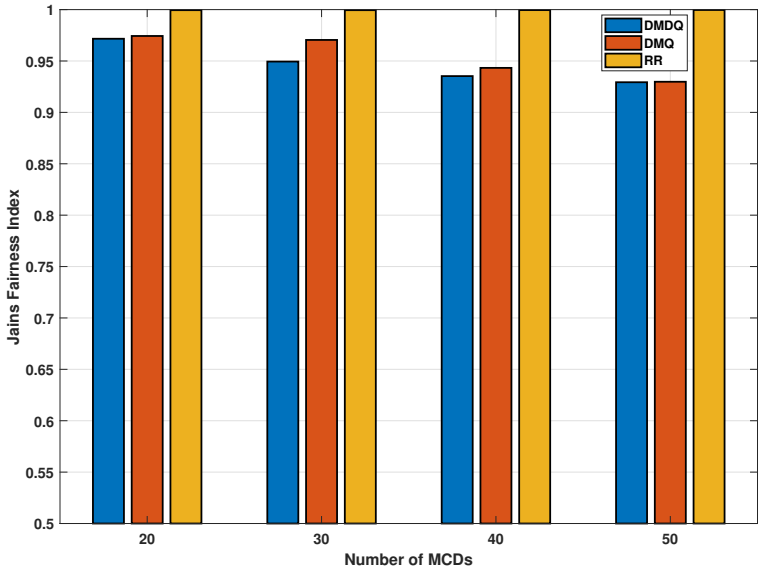

Figure 3.19: Jain's fairness index

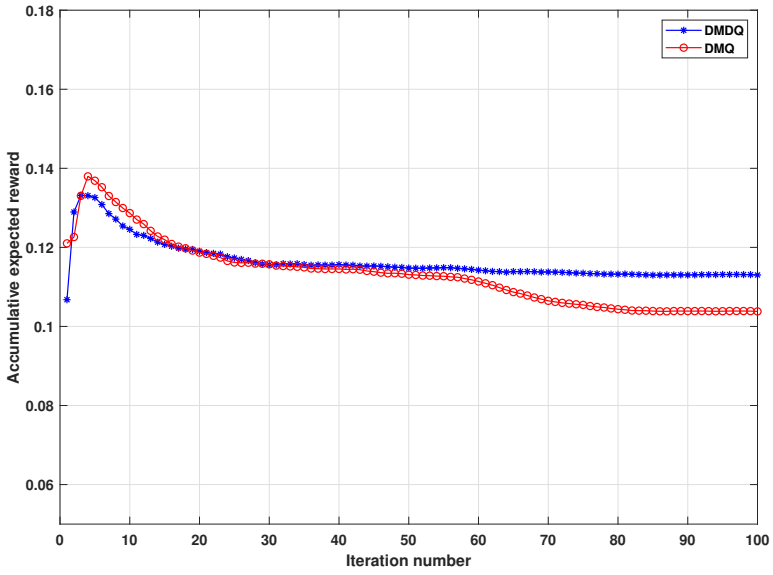

Figure 3.20: Average discounted reward for both DMDQ and DMQ; number of SBSs is 5, number of MCDs is 20, number of UEs is 20 and number of UNBs is 6.

3.5 Deep Q-Learning for Low-Latency Tactile Applications: Microgrid Communications

Tactile internet is aimed to support ultra low end-to-end latency applications, where rapid transfer of control messages is essential. Moreover, small-cell networks provide several advantages including improved coverage, latency, and throughput which position them as a promising technology for tactile applications with stringent QoS requirements. Yet, there are still challenges remaining in spectrum allocation, where Critical User Device (CUD)s need to coexist with traditional UEs. For instance, in a network of microgrids, CUDs represent microgrid devices with delay-sensitive monitoring and control data. Hence, they need efficient RB allocation algorithms. On the other hand, in a dense network, users can associate with one or the other small cell base station depending on channel quality. These challenges call for finding a balance between the quantity and quality of RB allocation. In other words, joint RB allocation and user association is needed in order to achieve low-latency in tactile communications.

The problem of resource allocation for mission-critical applications was addressed in the previous section, where deep reinforcement was employed to minimize the latency of critical users. We extend the problem in this section to consider joint allocation of RRM tasks. In particular, we propose an efficient joint RBG allocation and user-cell association algorithm, namely Delay Minimizing Deep Q-Network (DM-DQN), that aims to minimize the delay of mission-critical services. Our results show that DM-DQN achieves 41% reduction in delay of CUDs even under heavy traffic load. Moreover, DM-DQN converges faster than Q-learning.

3.5.1 System Model

We consider the uplink of a wireless small cell network composed of a set of $\mathcal{M}$ of M SBSs, a set $\mathcal{N}$ of N users, and a set $\mathcal{G}$ of G eNB (Fig. 3.21). Let $\mathcal{J}$ represent the set of base stations in the system. The set of users can be further divided to two user classes: The set $\mathcal{N}_c$ of N_c CUDs, and the set $\mathcal{N}_u$ of N_u UEs. CUDs and UEs can be served by either SBSs or eNB. We follow LTE release 12 [64] and divide the total system bandwidth into a set $\mathfrak{K}$ of K RBGs that are allocated among the CUDs and UEs each subframe (i.e., TTI).

We consider the 3GPP pathloss model [66] for the wireless channel. The pathloss (in dB) between i^{th} user and j^{th} base station on k^{th} RB (i.e., link (i, j, k)) is:

$$PL_{i,j,k} = 128.1 + 37.6 \log(d_{i,j}) \tag{3.27}$$

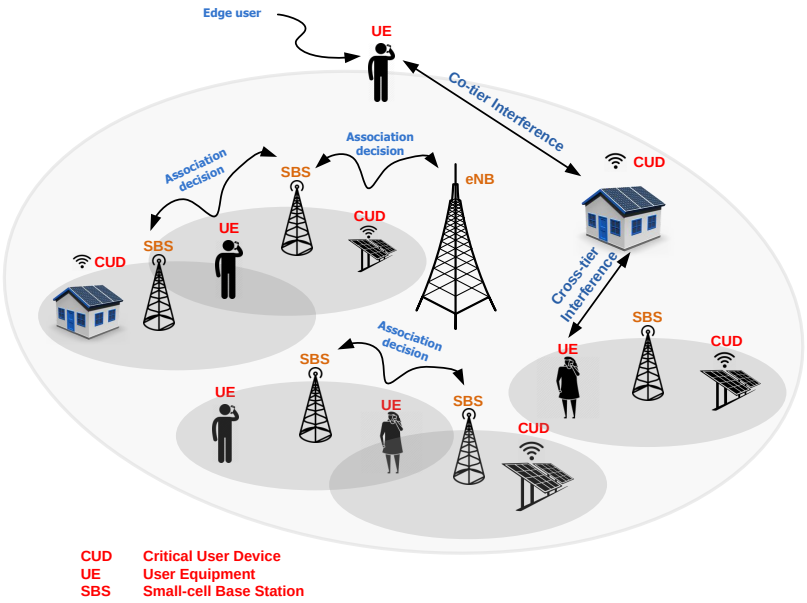

Figure 3.21: A small-cell wireless network covering CUDs and UEs.

where $d_{i,j}$ is the Euclidean distance between the i^{th} user and j^{th} base station in km. Shadowing is modeled as a log-normal distribution with zero-mean and 8 dB variance.

We consider two traffic models: Poisson packet arrivals for non-critical loads (i.e., UEs), and Beta distribution for critical loads (i.e., CUDs). Beta distribution is defined by 3GPP TR 37.868 for MTC [68] [94].

3.5.2 Problem Formulation

Our main objective is to improve the packet delay of CUDs served by the small cell network in a dense scenario. On one hand, finding the optimal RBG allocation allows for serving CUDs more rapidly, hence achieving less delay. On the other hand, channel quality, which is directly related to SINR, influences the Modulation and Coding Scheme (MCS) and Transport Block Size (TBS). Therefore, employing an efficient algorithm for finding the optimal association of users to the base station will maintain a high CQI, high MCS, high TBS, and will result in reduced latency. We seek to find a balance between RBG availability and association to a base station with better channel quality (i.e., CQI). Hence, our objective is to find an effective joint RBG allocation and user-cell association for each SBS $m \in \mathcal{M}$ on the uplink in order to minimize the average packet delay of CUDs in

the network. This problem involves finding both RBG allocation indicators x and user association indicators χ as follows:

$$\min_{\{x,\chi\}} \quad \frac{1}{N_c} \sum_{m\in\mathcal{M}} \sum_{i\in\mathcal{N}_c} \sum_{k\in\mathfrak{K}} D_{i,m,k}, \tag{3.28}$$

where $D_{i,m,k}$ is delay of CUDs on link (i,m,k). The delay on link (i,m,k) can be formulated as follows:

$$D_{i,m,k} = D^{tr}_{i,m,k} + D^{q}_{i,m,k}, \tag{3.29}$$

where $D^{tr}_{i,m,k}$ and $D^{q}_{i,m,k}$ are respectively transmission delay and queuing delay on link (i,m,k). In particular, queuing delay is the time packets experience waiting for scheduler allocation. Therefore, queuing delay is a direct result of scheduler efficiency (i.e., scheduling delay). The transmission delay on link (i,m,k) is defined as follows:

$$D^{tr}_{i,m,k} = \frac{\xi_{i,m,k}}{C_{i,m,k}}, \tag{3.30}$$

where $\xi_{i,m,k}$ is the packet size and $C_{i,m,k}$ is the transmission rate. The transmission rate is defined as follows:

$$C_{i,m,k} = \omega_k \ \log_2\Big(1 + \frac{x_{i,m,k}\ p_{i,m,k}\ h_{i,m,k}}{\omega_k N_0 + \sum\limits_{\substack{i'\neq i\\ m'\neq m}} x_{i',m',k} p_{i',m',k} h_{i',m',k} + \sum\limits_{\substack{i''\neq i\\ g\in\mathcal{G}}} x_{i'',g,k} p_{i'',g,k} h_{i'',g,k}}\Big), \tag{3.31}$$

where $x_{j,m,k}$ is the allocation indicator on link (i,m,k), ω_k is the bandwidth of k^{th} RBG, and N_0 is AWGN single-sided power spectral density. $p_{i,m,k}$, and $h_{i,m,k}$ are transmit power and channel coefficient on link (i,m,k). $p_{i',m',k}$, and $h_{i,m',k}$ are i' interferer transmit power and channel coefficient on link (i',m,k) and m' refers to the interferer SBS on m^{th} SBS. $p_{i'',g,k}$, and $h_{i'',g,k}$ are transmit power and channel coefficient of i'' user of g^{th} eNB on k^{th} RBG, respectively. As shown in Fig. 3.21, two sources of interference exist: Co-tier interference, which stems from users of neighboring SBSs (m') lying in the range of m^{th} SBS, and cross-tier interference, which stems from interference of users associated with eNB on users of m^{th} SBS. Each SBS performs link measurements to calculate SINR for its active users, hence calculating the uplink CQI. It is worth noting that all nodes comply with the equal power transmit strategy.

3.5.3 Resource Allocation using LSTM Deep Q-Network

The optimization problem in (3.28) constitutes a centralized solution which cannot be solved in feasible time. In reality, base stations are expected to operate in a distributed

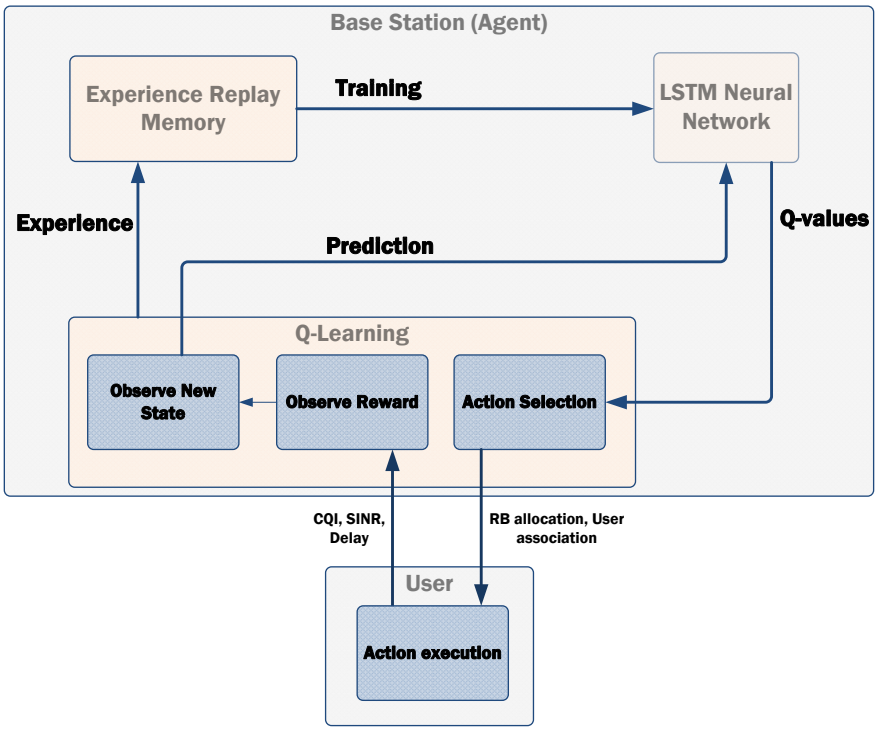

Figure 3.22: Conceptual model for delay minimization using deep Q-network.

approach. Consequently, we propose a distributed delay minimization algorithm based on DQN [11]. The proposed delay minimization using DQN , DM-DQN, algorithm offers several advantages. First, unlike tabular Q-learning, DM-DQN introduces a neural network for the estimation of the Q-value [92]. Such integration between neural network and Q-learning facilitates the estimation of Q-values of all actions at each iteration, which leads to rapid convergence. Second, the deep neural network facilitates learning long-term dependencies in input sequences. Therefore, it fits well with the periodic traffic pattern of devices that can be efficiently learned using a deep neural network.

Fig. 3.22 presents the conceptual model of DM-DQN. SBSs are the agents that can make actions defined as $a_{m,t}$ where m is the SBS index and t is the TTI index. Each agent aims to make the action that minimize its total delay. Therefore, we use two states in our model (0 and 1). State 0 represents the target state, where the achieved delay is less than a target delay D_0. Otherwise, the agent dwells in state 1. This way our algorithm targets achieving a delay lower than D_0.

The selection of two states helps in reducing the state space, hence it improves the

convergence time. The reward function defines the feedback to m^{th} agent in response to the selected action $a_{m,t}$ of the current state $s_{m,t}$ at t^{th} TTI as:

$$r_{m,t}(s_{m,t}, a_{m,t}) = \left(\frac{-2}{\pi}\right) \tan^{-1}(D_{m,t}) \tag{3.32}$$

The reward function in (3.32) rewards the agent with a less reward (i.e., closer to zero) as long as the average packet delay is small, while it penalizes the agent whenever the delay increases. This reward definition motivates base stations to associate users and allocate them RBGs as long as their average delay ($D_{m,t}$) is less than a predefined threshold (D_0).

We adopt LSTM [93] as our deep neural network as shown in Fig. 3.22. LSTM is a deep RNN that is capable of storing recent input states using its feedback connections. In traditional RNNs, a hidden node consists of a single activation function. However, in LSTM, a hidden node is a memory cell with three gates: forget, input, and output gates. These gates control the degree to which contents are memorized, erased or exposed [95]. As such, a salient feature of LSTM is its ability to learn long-term dependencies in input sequences, which makes it a promising choice for learning microgrid traffic patterns.

In DM-DQN, LSTM works as a non-linear function approximator for the Q-value function of the Q-learning algorithm. In such cases, the Q-learning algorithm tends to diverge [11]. To remedy this situation, authors in [11] introduced the experience replay memory, where the LSTM network is trained using samples drawn uniformly from past experiences. Therefore, we adopt the experience replay memory as shown in Fig. 3.22. Each experience, represented as $e = \{s, a, r, s'\}$ (i.e., current state, current action, reward, new state), is stored in memory to be used later for training the LSTM. In addition, we refrain from training the LSTM every TTI to reduce algorithm complexity. Instead, we perform it every Ω TTIs, where Ω is set to 20 in our simulations.

Algorithm 3.3 presents the DM-DQN algorithm that runs on each SBS. Note that, the eNB performs RR scheduling. In addition, we assume that users can be served by only one base station. According to our algorithm, during the training phase, a user can be selected by several base stations, hence we define a priority of association. That is the user is associated only to the base station with the lowest identification number. This decision is shared among base stations through the backhaul connection. After the training phase, each base station will learn to associate only the users that minimize its total delay.

3.5.4 Performance Evaluation

In our simulations, we use Matlab LTE system toolbox and the neural network toolbox. The network scenario comprises one eNB of radius 500m [96], and 10 SBSs each of radius

Algorithm 3.3 Delay Minimization using Deep Q-Network (DM-DQN) Algorithm

Initialization: LSTM parameters .
for TTI $t = 1$ to T **do**
  for SBS $m \in \mathcal{M}$ **do**
    Step 1 (Performed by each user):
    for User $i \in \mathcal{N}_c$ **do**
      Perform the last action assigned by the SBS (i.e., associate to base station and transmit packets on assigned RBs).
      Observe the recent packet delay.
      Transmit delay information to the associated base station.
    end for
    Step 2: Compute average packet delay, $D_{m,t}$, for all served CUDs.
    Step 3: Compute the reward, $r_{m,t}$, and the new state due last action execution.
    Step 4: Store current experience in the experience replay memory: $e_{m,t} = \{s_{m,t}, a_{m,t}, r_{m,t}, s_{m,t+1}\}$.
    Step 5: LSTM training: Perform LSTM training (every Ω TTIs) using experiences drawn uniformly from the experience replay memory.
    Step 6: Update next state $s_{m,t+1}$.
    Step 7: LSTM prediction: Predict $q(s_{m,t}, a'_m) \forall a'_m \in \mathcal{A}_m$.
    Step 8: Select the next action, $a_{m,t+1}$, using ϵ-greedy policy.
  end for
end for

80m. Number of UEs is fixed as 30, while the number of CUDs is changed between 30 to 60. Simulation settings are summarized in Table 3.8.

Fig. 3.23 presents the number of packets (in KBytes) processed by each SBS and the eNB. The results show that, DM-DQN redistributes the traffic load from eNB to SBSs. Both of the algorithms help SBSs to experience similar loads, i.e., they balance the load in a fair manner.

This redistribution of traffic helps in improving both delay and throughput. Fig. 3.24a and 3.24b show the delay of the CUDs and UEs, respectively, for DM-DQN and Q-learning. As can be seen from the figures, DM-DQN outperforms Q-learning for both CUDs and UEs in terms of delay. The delay of CUDs is reduced by 41%, whereas the delay of UEs is reduced by 80% at the dense scenario (i.e., number of CUDs is 60 and number of UEs is 30). Moreover, the delay of CUDs is below the target delay (i.e., 20 ms) for all CUD deployments. However, the delay of CUDs is close to 15ms when the number of CUDs is 60 and the delay of UEs is 10ms. This is the result of denser CUD deployment and traffic type. CUDs generate Beta traffic with higher load than the UEs. When the number of CUDs is 50, the delays for both device types are close to 10ms. Note that, neither DM-DQN nor Q-learning incurs outage. Furthermore, the latency budget for microgrid applications has a constraint of 20ms [97], hence we chose that as our target delay. Fig. 3.24a shows that DM-DQN achieves the target delay for all CUD scenarios, whereas Q-learning fails for high dense scenarios (i.e., number of CUDs $>$ 40).

Table 3.8: Simulation settings

Physical layer	
Transmission bandwidth	10 MHz
Number of RBs	50 (12 subcarriers / RB)
Number of RBGs	5 (10 RBs/RBG)
User's max transmit power	20 dBm [84]
Pathloss model	3GPP $PL_{dB} = 128.1 + 37.6 \log(d)$
Penetration loss	20 dB
Noise figure	5 dB
Shadowing	$\sim$ LOGN(0, 8(dB))
TTI	1 msec
Number of simulation runs	10
Network	
eNB radius	500 m
SBS radius	100 m
Number of eNBs	1
Number of SBSs per eNB	10
Number of CUDs per SBS	3:1:6
Number of UEs per SBS	3
Traffic	
CUDs traffic model	Beta ($a = 3, b = 4$) [68]
UEs traffic model	Poisson (mean Inter-arrival time = 5 msec)
Packet size	Exponential (mean = 40 Bytes)
Q-learning	
Learning rate (α)	0.5
Discount factor (γ)	0.9
Exploration probability (ϵ)	0.9 [83]
Target delay (D_0)	20 ms
LSTM	
Number of layers	4
Fully connected layer units	40
Hidden units	40
Initial learning rate	0.0001
Experience replay memory size	30
Training interval (Ω) (in TTIs)	20

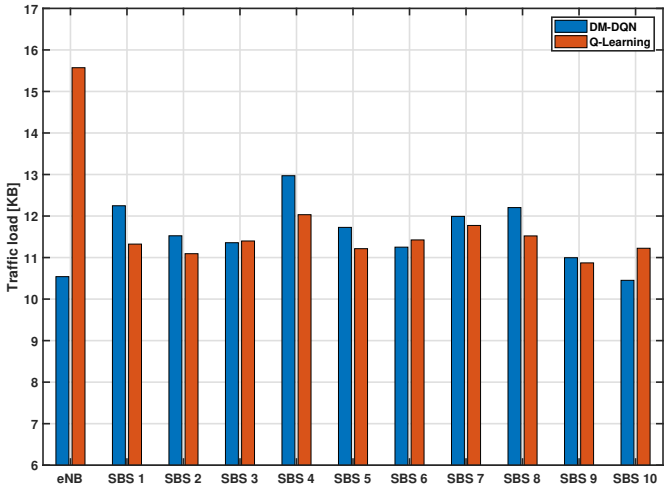

Figure 3.23: Traffic load on each BS; number of SBS is 10, number of CUDs is 50, and number of UEs is 30.

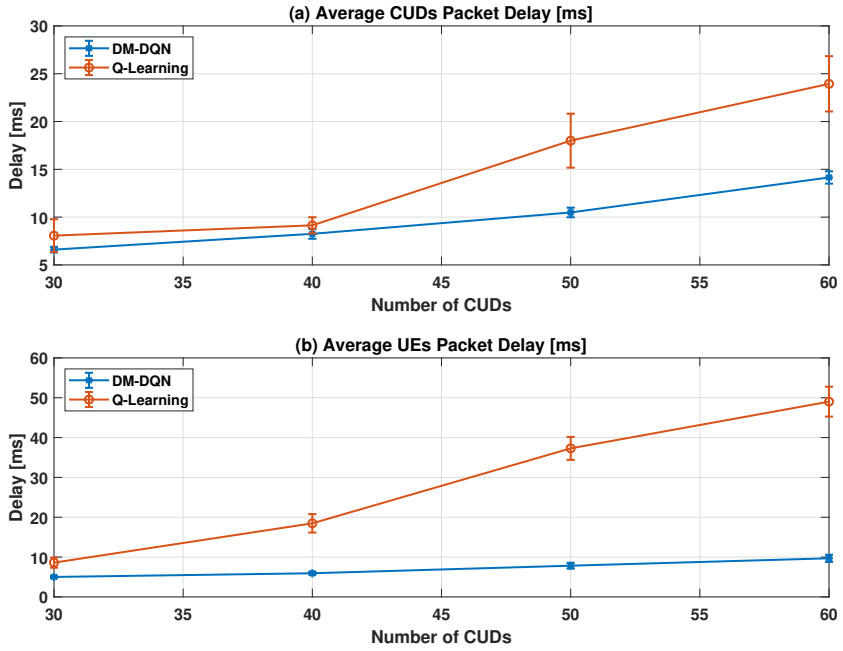

Figure 3.24: Average packets delay of (a) CUDs and (b) UEs versus number of CUDs; number of SBS is 10, and number of UEs is 30.

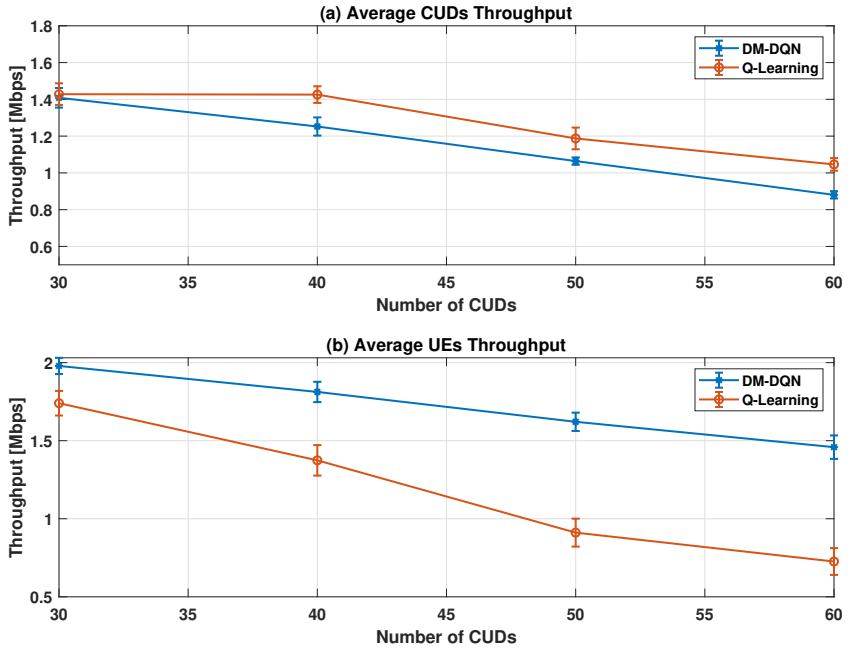

Figure 3.25: Average throughput of (a) CUDs and (b) UEs versus number of CUDs; number of SBS is 10, and number of UEs is 30.

Fig. 3.25a and 3.25b show the average throughput of both CUDs and UEs, respectively. The figures demonstrate a degradation in throughput of CUDs by 16%, while UEs experience a significant improvement when DM-DQN is employed. The proposed DM-DQN only aims to reduce the delay since for microgrid communications latency is more critical than throughput. However, for different applications it might be possible to jointly address throughput maximization and improve latency.

The convergence of the algorithms is shown in Fig. 3.26, where the accumulated reward function is plotted versus the number of iterations. Accumulated reward function is defined as $|\sum_{t=1}^{T} r_t/T|$, where $|x|$ represents the absolute value of x, T is the subframe number (i.e., x-axis) and r_t is the average reward of all SBSs at t^{th} time instant. It is worth noting that we defined the reward in (3.32) as a negative function. However, we evaluate convergence in terms of the absolute value, hence lower value indicates better reward. DM-DQN converges around 60 iterations while Q-learning has a slower convergence.

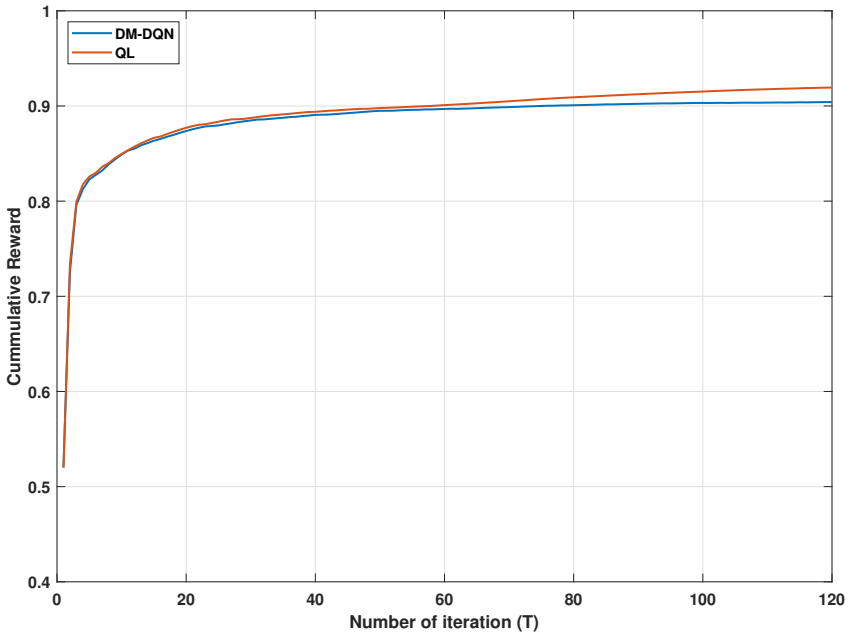

Figure 3.26: Cummulative reward of SBSs versus number of iterations computed as $|\sum_{t=1}^{T} r_t/T|$; number of SBS is 10, number of CUDs is 50, and number of UEs is 30.

3.6 Conclusion

In this chapter, we presented reinforcement and deep reinforcement learning algorithms to improve throughput, latency, and fairness of LTE networks.

First, TMQ, which is based on Q-learning, was proposed to improve throughput of tactile communications. Results showed about 130% throughput improvement, 80% reduction in delay, and 6% increase in fairness compared to the baseline algorithms.

Second, we focused on using Q-learning with the aim to improve latency of CRMs using LTE small-cell networks. As such, DMQ was proposed based on Q-learning with the aim to minimize latency of microgrid users. Furthermore, we compared DMQ to both heuristic- and optimization-based algorithms. Results show delay reduction of 66% and 33% for microgrid users compared to heuristic and optimization, respectively. Furthermore, DMQ achieves the highest fairness values among the other schemes.

Third, the previous work was extended to address the slow convergence of Q-learning by employing deep Q-learning to reduce the latency of mission-critical users. As such, we proposed DMDQ, a deep reinforcement learning-based algorithm, for RB allocation with the aim to improve network latency, where an LSTM deep neural network is used. The proposed DMDQ was compared to Round Robin and DMQ algorithms. In particular, DMDQ achieves 60% convergence speedup compared to DMQ. Furthermore, DMDQ achieves 30% delay reduction with only 9% throughput degradation compared to DMQ.

Finally, we extended the previous work further to consider joint resource allocation and user-cell association with the aim to reduce latency of microgrid users. Again, LSTM-based deep Q-Network was used for improving network latency. Results are compared to tabular Q-learning, where the proposed algorithm achieves 41% delay reduction for critical users, in addition to 40% convergence speedup compared to Q-learning.

Chapter 4

Reinforcement Learning for 5G Networks

4.1 Introduction

The rapid increase in services and use cases of wireless networks has driven the inception of LTE with a focus on improving mobile broadband connections with a flattened IP network. Such increase in mobile data traffic is expected to reach 5 zettabytes per month as of 2030 [98]. In this chapter, we focus on 5G networks and study the use of model-free reinforcement learning with 5G technologies with the aim to improve network KPIs. Similar to the previous chapter, we employ various traffic scenarios that pose diverse KPI targets, i.e., throughput, latency, and packet drop rate. In particular, uRLCC and eMBB service categories of 5G are used, where uRLLC requires high reliability and low latency, whereas eMBB requires high throughput. The coexistence of those service categories calls for an efficient RRM that balances their conflicting KPIs.

This chapter is organized as follows. Section 4.2 presents the use of reinforcement learning in a 5G network covering uRLLC and eMBB users with the aim to improve uRLLC's latency through resource block and power allocation, whereas in section 4.3, we extend the problem to address the balance between uRLLC's latency and eMBB's throughput using reinforcement learning. In section 4.4, with the use of beamforming, reinforcement learning is used to improve network's sum throughput using joint user-cell association and power allocation per beam. Furthermore, section 4.5 addresses the problem of spatial and temporal dynamicity in 5G mm-wave network employing beamforming,

where deep reinforcement learning is used to perform resource block allocation and selction of number of beams with the aim to balance uRLLC's latency and eMBB's throughput.

4.2 Reinforcement Learning-based Joint Power and Resource Allocation for uRLLC in 5G

In this section, we address the coexistency problem of uRLLC and eMBB traffic over 5G network where we seek to balance latency and reliability of uRLLC users by designing a multi-agent Q-learning algorithm for joint power and resource allocation [99]. The proposed algorithm utilizes the flexibility of the time-frequency grid introduced in the 5G standard to allocate resources to users according to their demands. In particular, our multi-agent Q-learning algorithm is adopted for each 5G NodeB (gNB) to perform joint power and resource block allocation every scheduling interval. Improving latency requires reducing both transmission and queuing delays since they constitute the main impairments against achieving the 1 msec latency requirement [100]. Furthermore, improving reliability contributes to improving transmission delay by reducing both the need for re-transmission and packet's segmentation at the Radio Link Control (RLC) layer. In addition, queuing delay is a direct outcome of scheduling delay. As such, we present a reward function crafted carefully to address reliability, transmission and queuing delays of uRLLC users. We evaluate the performance of the algorithm in the presence of Constant Bit Rate (CBR) traffic, in addition to Poisson traffic. We show that our algorithm outperforms the baseline algorithm, that is recently proposed in [101], with 4% reduction in Packet Drop Rate (PDR), while achieving lower latency for high traffic loads.

4.2.1 System Model

We consider a network consisting of a set $\mathcal{J}$ of gNBs (each gNB $j \in \mathcal{J}$ and $|\mathcal{J}|=J$) which are deployed within 500 meters inter-site distance as shown in Fig. 4.1. Each gNB covers a set $\mathcal{N}$ of users (each user $i \in \mathcal{N}$ and $|\mathcal{N}|= N$) which are stationary and deployed uniformly within the gNB coverage. $\mathcal{N}$ is composed of two types of users: a set $\mathcal{N}_r$ of uRLCC users (each uRLCC user $i_r \in \mathcal{N}_r$ and $|\mathcal{N}_r|=N_r$), and a set $\mathcal{N}_m$ of eMBB users (each eMBB user $i_m \in \mathcal{N}_m$ and $|\mathcal{N}_m|=N_m$). All users adhere to 5G realease-15 standard, where downlink communication is considered.

The 5G standard uses a flexible time-frequency grid to support variable length TTI and configurable subcarrier spacing, a.k.a numerologies. The baseline numerology is 15

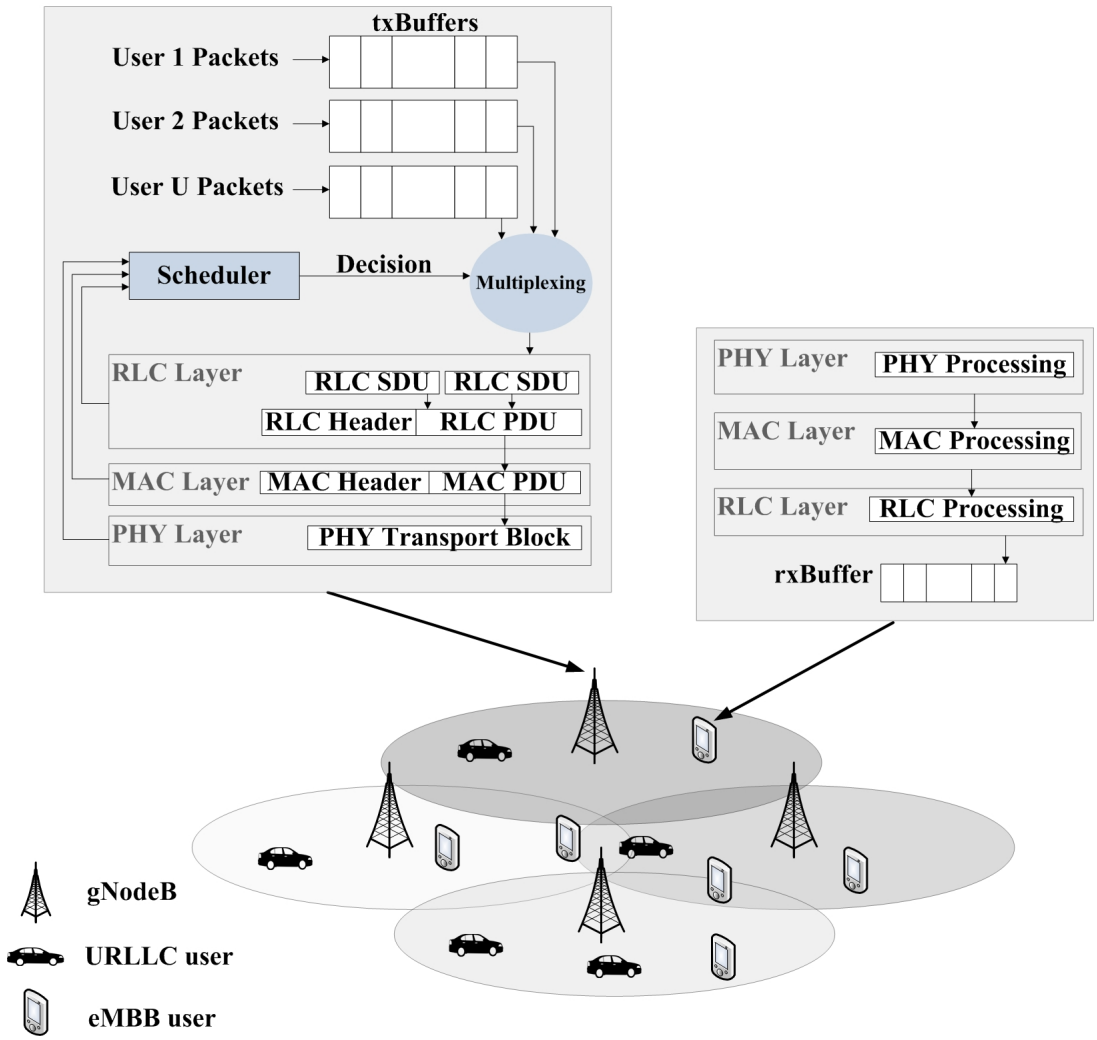

Figure 4.1: System model of joint power and resource allocation for uRLLC in 5G networks.

KHz, 12 subcarriers per resource block, and 14 symbols per subframe. Other numerologies with 2^{μ} scaling are allowed in the standard, where μ represents the numerology index. For a certain bandwidth configuration, ω MHz, let K_{RB}^{DL} be the total number of resource blocks available for downlink transmission. Consecutive resource blocks are bundled to form RBGs as defined in [102]. We define $\mathfrak{K}$ to be a set of K RBGs, where the size of a RBG is $\lceil K_{RB}^{DL}/K \rceil$ resource blocks. The RBG is considered as the unit of allocation. $p_{k,j}$ is defined as the transmission power allocated to k^{th} RBG by j^{th} gNB. In addition, scheduling resolution includes a slot, composed of 14 OFDM symbols, and mini-slots of $2, 4$, or 7 OFDM symbols. Services with strict latency requirements, such as uRLCC, can be scheduled on a short TTI (i.e., mini-slot), whereas services with high throughput requirements, such as eMBB, can be scheduled on a long TTI (i.e., slot of 14 symbols). We select the finest resolution as the scheduling interval (i.e., TTI equals 2 OFDM symbols) in order to facilitate uRLCC latency requirement close to 1 msec.

Link adaptation and Hybrid Automatic Repeat Request (HARQ) are employed using information of channel quality indicator reported on uplink control channel. For cases where first transmission is erroneously decoded, a HARQ re-transmission is triggered. In particular, HARQ transmission/re-transmission consumes D_i^{harq} round trip delay that consists of data and acknowledgement transmission times, where $i \in \mathcal{N}$. In line with [103], we assume $D_i^{harq} = 4D_{tti}$, where D_{tti} is the TTI duration which is 2 OFDM symbols (i.e., $D_{tti} = 0.143$ msec for 15 KHz subcarrier spacing). It is assumed that re-transmissions are always prioritized over new transmissions.

The traffic corresponding to each user is queued in a transmission buffer maintained at the gNB. Every TTI, the downlink scheduler is evoked by the gNB to perform RB allocation for the pending traffic in the transmission buffers. Two traffic models are considered: CBR (i.e., periodic) and Poisson arrivals with mean arrival rate λ [packets/sec]. In particular, uRLLC users' traffic is composed of a mixture of CBR and Poisson arrivals; whereas eMBB users' traffic follows Poisson arrivals.

4.2.2 Latency and Reliability

Latency experienced by a packet can be formulated as:

$$D = D^{q} + D^{tx} + D^{harq}, \tag{4.1}$$

where D^{q} is the queuing delay, D^{tx} is the transmission delay, and D^{harq} is the HARQ re-transmission delay. Transmission delay of i^{th} user attached to j^{th} gNB can be formulated

as follows:

$$D_{i,j}^{tx} = \frac{\xi_{i,j}}{\sum_{k=1}^{K} \omega_k \log_2 \left(1 + \frac{x_{k,i,j} p_{k,j} h_{k,i,j}}{\omega_k N_0 + \sum_{\substack{j' \in \mathcal{J} \\ j' \neq j}} x_{k,i,j'} p_{k,j'} h_{k,i,j'}}\right)}, \tag{4.2}$$

where $\xi_{i,j}$ is the packet size of the $(i,j)^{th}$ link, ω_k is the bandwidth of k^{th} RBG and N_0 is the AWGN single-sided power spectral density. $p_{k,j}$ is the transmit power of j^{th} gNB on k^{th} RBG, $h_{k,i,j}$ is the channel coefficient, and $x_{k,i,j}$ is the RBG allocation indicator of $(k,i,j)^{th}$ link. $p_{k,j'}$ is the transmit power of j' interfering gNB, $h_{k,i,j'}$ is the channel coefficient, and $x_{k,i,j'}$ is the allocation indicator of $(k,i,j')^{th}$ link.

The transmission delay in (4.2) incorporates the delay due to segmentation at the RLC layer which depends on the achievable rate. This is impacted by the level of interference in the network (i.e., SINR). As such, improving the quality of resources allocated to a user improves SINR which leads to a better modulation and coding scheme and higher transport block size allocation. This, in turn, reduces segmentation at the RLC layer and improves the delay. Besides, transport block size can be increased through allocation of more resource blocks, which further reduces the delay.

The queuing delay is a direct outcome of the scheduling delay incurred at the Medium Access Control (MAC) scheduler. Indeed, transmission delay and queuing delay may contribute to large delay values, if not handled carefully, and become the major roadblocks in achieving the 1 msec latency [100]. Therefore, achieving latency values close to 1 msec mandates immediate scheduling of uRLLC users and limiting HARQ re-transmission to one.

Limiting the number of HARQ re-transmissions to one as well as inter-cell interference can lead to higher PDR. In particular, interference can have significant impact on edge-users which can be solved by carefully adjusting the gNB's transmission power on different RBGs (i.e., RBG-based power allocation). As such, observing the SINR value plays an important role in determining accurate transmission power of each RBG. All in all, SINR estimation, power allocation, and RBG allocation are needed to balance the trade-off between latency and reliability of uRLLC users. In the following section, we present a joint power and RBG allocation algorithm based on Q-learning for uRLLC's latency and reliability improvement.

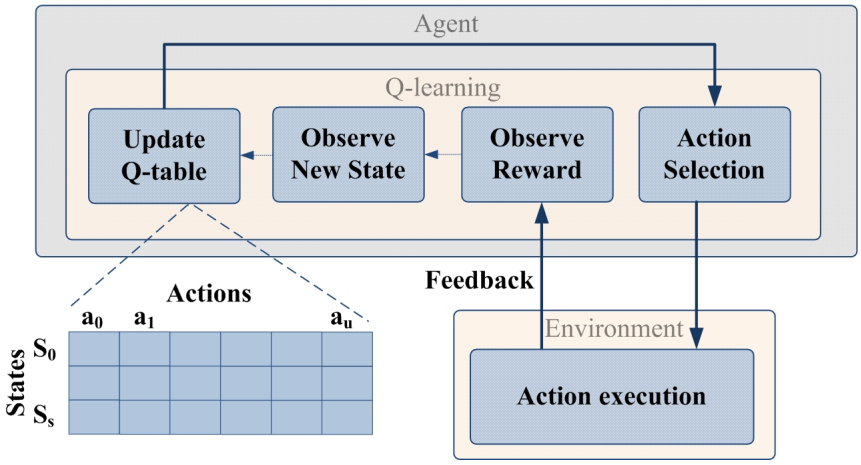

Figure 4.2: General block diagram of Q-learning.

4.2.3 Low-Latency High-Reliability for uRLLC using Q-Learning (LLHRQ)

In our decentralized learning approach, each gNB adopts the multi-agent reinforcement learning, specifically Q-learning algorithm. Fig. 4.2 presents a general block diagram of the operation of Q-learning framework. In particular, agents are the set of players aiming to select actions to maximize their reward. The agents do not share information regarding their action selection to avoid excessive overhead. However, feedback from the environment acts as a clue to each agent. Hence, states are used to observe the status of the environment. In addition, a reward function is defined to guide each agent in its decision process. In particular, each agent's objective is to find the best policy that maximizes its discounted reward over infinite time-horizon. As such, Q-learning estimates the quality of visited state-action pair using an iterative update using (2.4). The Q-values are stored in a table indexed by the states and actions. The steps of Q-learning is shown in Fig. 4.2, where the algorithm starts by selecting an action to be executed, then it observes some reward corresponding to that action. This action will lead to environment's state transition. Finally, the algorithm updates the Q-value and repeats the process.

We formulate the Q-learning algorithm to improve reliability and minimize latency of uRLLC users as follows:

- **Agents:** gNBs.
- **Actions:** The actions are defined as the joint power and RB allocations made by

each gNB for its attached users. To reduce action-space, consecutive resource blocks are bundled to form a RBG which represents 8 consecutive resource blocks as defined in [102]. We denote the number of RBGs available for each gNB with K. As such, actions of j^{th} gNB on k^{th} RBG can be defined as $\boldsymbol{a}_{k,j} = \{\boldsymbol{x}_{k,j}, p_{k,j}\}$, where $\boldsymbol{x}_{k,j} = \{x_{k,i,j} : i \in \mathcal{N}\}$ is a vector of RBG's allocation indicator for each user (i.e., $x_{k,i,j} = 1$ if k^{th} RBG is allocated to i^{th} user and 0 otherwise) and $p_{k,j}$ is the power allocated to k^{th} RBG.

- **States:** In the absence of agents' cooperation, feedback from the environment should play this role which is represented by the states. In particular, we observe the interference impacting uRLLC users by estimating the SINR value at each user. This quantifies the reliability of uRLLC users. As such, two states are designed as follows:

$$s_{k,j} = \begin{cases} s_0, & \bar{\Gamma}_{\mathcal{N}_r,k,j} \geq \Gamma_{th}, \\ s_1, & \text{otherwise}, \end{cases} \tag{4.3}$$

  where, $\bar{\Gamma}_{\mathcal{N}_r,k,j}$ represents the average SINR value of uRLLC users on k^{th} RBG. State s_0 is visited whenever average SINR of uRLLC users exceeds a certain threshold, Γ_{th}, while s_1 is visited otherwise. The value Γ_{th} is chosen to maintain high probability of decoding. Furthermore, SINR contributes to link adaptation which improves transmission delay as discussed in section 4.2.2.

- **Reward:** We formulate the reward function as follows:

$$r_{k,j} = \begin{cases} 1 - \max\limits_{i \in \mathcal{N}_r} (D^q_{i,j})^2, & \bar{\Gamma}_{k,j} \geq \Gamma_{th}, \\ -1, & \text{otherwise}, \end{cases} \tag{4.4}$$

  where $D^q_{i,j}$ represents the last packet queuing delay of i^{th} uRLLC user. The advantage of formulating the reward using (4.4) is twofold. First, the reward serves in driving each agent toward actions with better average SINR of uRLLC users (i.e., high reliability). On one hand, improving reliability should improve transmission delay since less re-transmissions will be required. On the other hand, packets can be accommodated in larger transport blocks. Second, the agent is rewarded a value that relies on the maximum queuing delay experienced by its attached uRLLC users. In particular, as the maximum delay approaches zero, the reward value approaches one. This motivates each agent to schedule uRLLC users immediately (i.e., zero queuing delay) as this will lead to the highest reward value.

4.2.4 Baseline Algorithm

We compare the proposed algorithm to a baseline algorithm that is based on traditional PF with priority given to uRLLC users. Priority-based Proportional Fairness (PPF) is proposed in [101] and here implemented with the addition of equal power allocation. PPF works as follows: on each TTI, each gNB allocates RBGs to users according to their quality of service demands. This is achieved using two-step process. First, RBGs are allocated to uRLLC users with pending data transmissions; second, the remaining RBGs are allocated to eMBB users. For each step, resources are allocated among users using PF criteria as follows:

$$i_k^* = \arg\max_{i \in \mathcal{N}} \left\{ \frac{C_{i,k}}{\overline{C}_{i,k}} \right\}, \tag{4.5}$$

where i_k^* is the user allocated k^{th} RBG, $C_{i,k}$ is the instantaneous rate of i^{th} user on k^{th} RBG, and $\overline{C}_{i,k}$ is its average delivered user throughput in the past. The maximum number of RBGs allocated to a user is deduced from the CQI value and the amount of pending data for transmission.

4.2.5 Performance Evaluation

Table 4.1 presents the network and Q-learning settings considered in our simulations. The wireless channel large-scale fading is modeled using 3GPP pathloss model [104]: $PL_{dB} = 128.1 + 37.6 \log(d)$, where d is the distance between a user and its gNodeB in km. Shadowing is modeled as a log-normal distribution with zero-mean and 8 dB variance. In addition, single-input single output system is used with transmitter and receiver antenna gains of 15 dB and noise figure of 5 dB.

Each gNodeB covers 10 uRLLC and 5 eMBB users. The traffic of uRLLC is a mixture of 20% CBR and 80% Poisson arrivals, whereas traffic of eMBB follows Poisson arrivals. Small payload size of 32 bytes is used for all users. In addition, the traffic loads per cell considered for uRLLC users are 0.5, 1, 1.5, and 2 Mbps whereas 0.5 Mbps is considered for eMBB users. The composition of traffic loads and mixture serves in assessing the performance under light and heavy network traffic conditions. Simulation results are collected for 5000 TTIs and 5 simulation runs.

The action space of LLHRQ consists of the combination of power and RBG allocations. We use $K = 13$ RBGs for a system bandwidth of 20 MHz, where the first 12 RBGs consist of 8 RBs while the last RBG contains 4 RBs. For the power, $p_{k,j} \in \{0, 1, 2, 3\}$ dBm with

Table 4.1: Network settings

Physical layer	
Max. transmission power	40 dBm [84]
Transmission power levels	$p_{k,j} \in \{0, 1, 2, 3\}$ dBm
Tx/Rx antenna gain	15 dB
Noise figure	5 dB
Penetration loss	5 dB
Network environment	3GPP Urban Macro (UMa) network
Propagation	$128.1 + 37.6 \log(d)$
Shadowing	Log-Normal with 8 dB standard deviation
Bandwidth	20 MHz bandwidth
Carrier frequency	4 GHz
numerology	15 KHz subcarrier spacing
Number of RBG	$K = 13$
TTI size	2 OFDM symbols (0.1429 msec)
MAC layer	
HARQ	Asynchronous HARQ
HARQ round trip delay	4 TTIs
HARQ processes	6
Maximum number of re-transmissions	1
Network	
Number of gNBs	5
Inter-site distance	500 meters
Number of uRLLC users	50 (10 per cell)
Number of eMBB users	25 (5 per cell)
User distribution	Stationary and uniformly distributed
Traffic	
uRLLC	20% CBR and 80% Poisson
eMBB	Poisson
Payload size (uRLLC/eMBB)	32 Byte
uRLLC load/cell	[0.5 : 0.5 : 2] Mbps
eMBB load/cell	0.5 Mbps
Q-learning	
Learning rate (α)	0.5
Discount factor (γ)	0.9
Exploration probability (ϵ)	0.05
SINR threshold (Γ_{th})	20 dB

maximum gNB's transmission power of 40 dBm. In addition, the threshold of SINR Γ_{th} is adjusted to 20 dB to maintain high probability of successful decoding at the user side.

To evaluate the performance of the proposed scheme we present latency, PDR and throughput results. Fig. 4.3 and 4.4 show empirical Complementary Cumulative Distribution Function (eCCDF) of latency of uRLLC users under various traffic loads of uRLLC users for low and high traffic scenarios, respectively. In Fig. 4.3, it can be seen that both algorithms achieve very close latency results, while in Fig. 4.4 our proposed LLHRQ outperforms PPF at the 10^{-3} percentile for 1.5 Mbps load and at the 10^{-4} percentile for 2 Mbps load. In particular, inefficient power allocation of PPF impacts both interference and link adaptation. This leads to more packet segmentation which increases transmission latency. Learning joint power and resource allocation combats that problem and improves the transmission delay. In addition, Fig. 4.5 provides more insights into the relation between reliability and latency. Indeed, better latency is achieved by allowing packets to be dropped after the first re-transmission. PPF experiences that issue due to the inefficient interference handling. Hence, in Fig. 4.5 we observe that the drop rate of PPF is increasing. Meanwhile, LLHRQ was able to achieve drop rates below 0.1% due to power allocation capabilities. In our worst case traffic load, LLHRQ outperforms PPF with 4% reduction of PDR.

Fig. 4.6 presents the throughput of eMBB users under various traffic loads of uRLLC users. The figure shows that LLHRQ outperforms the throughput of PPF under all uRLLC's traffic loads. In particular, for PPF, increasing uRLLC's traffic load significantly impacts throughput of eMBB users. However, as the traffic load increases, LLHRQ experiences a degradation in the throughput of eMBB users. Therefore, in the next section, we extend the algorithm with the aim to improve QoS requirements of uRLLC and eMBB users simultaneously. Finally, we observed a convergence of LLHRQ after 3000 TTIs.

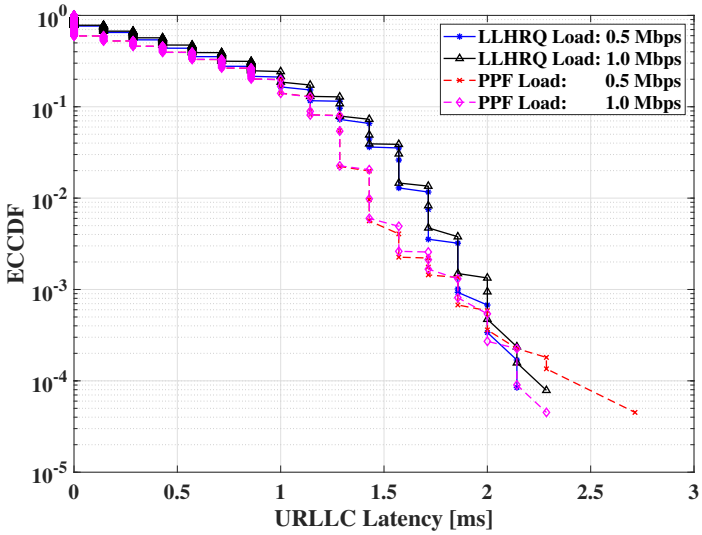

Figure 4.3: eCCDF of uRLLC's latency [msec]; eMBB load is 0.5 Mbps. uRLLC loads: 0.5 and 1 Mbps.

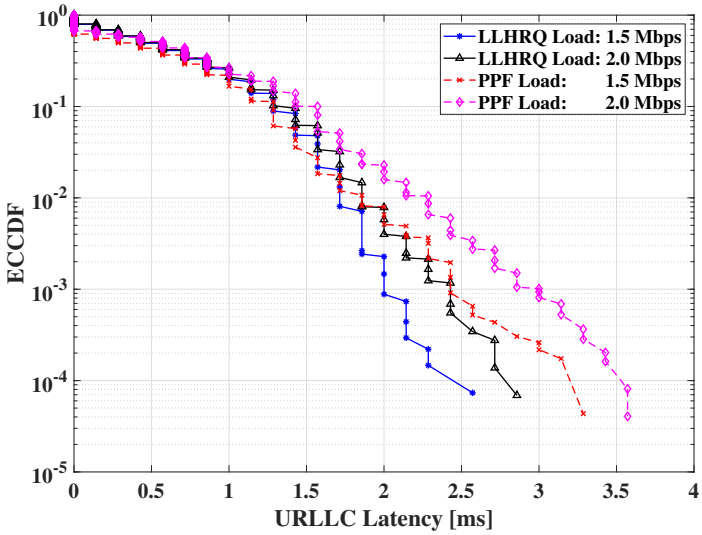

Figure 4.4: eCCDF of uRLLC's latency [msec]; eMBB load is 0.5 Mbps. uRLLC load: 1.5 and 2 Mbps.

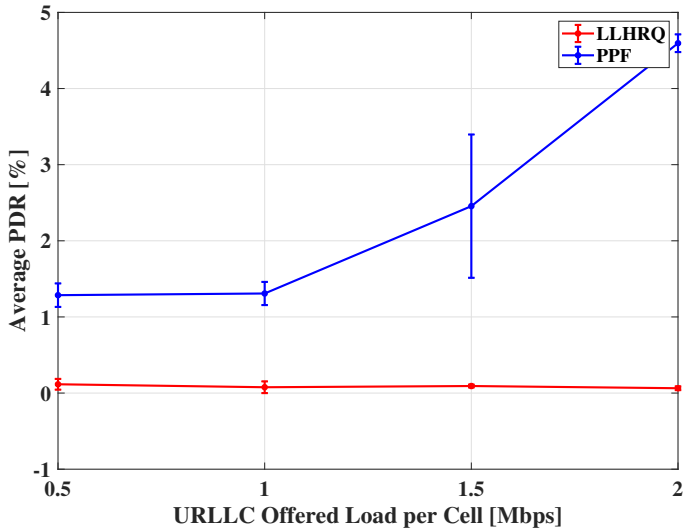

Figure 4.5: Average packet drop rate [%]; eMBB load is 0.5 Mbps.

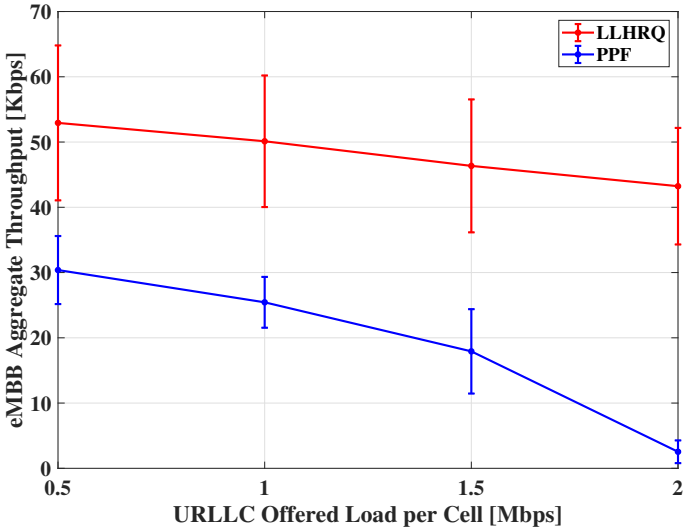

Figure 4.6: Aggregate throughput of eMBB users [Mbps] against uRLLC's traffic load.

4.3 ML-Enabled Radio Resource Allocation in 5G for uRLLC and eMBB Users

In this section, we aim to balance QoS requirements stemming from the coexistence of uRLLC and eMBB traffic over 5G network [105]. In particular, besides improving latency and reliability of uRLLC users, we aim to maintain throughput performance of eMBB users. To achieve this, we propose a multi-agent Q-learning algorithm, namely Latency-Reliability-Throughput Improvement using Q-Learning (LRT-Q), to perform joint power and resource block allocation for each gNB at every scheduling interval. The reward and state functions of LRT-Q is designed carefully to satisfy three KPIs (i.e., reliability, queuing and transmission delays of uRLLC users, and throughput of eMBB users). We evaluate the performance of LRT-Q in the presence of CBR traffic, in addition to Poisson traffic. Furthermore, we compare the performance of LRT-Q to two baseline algorithms: A PPF algorithm (with addition of equal power allocation), proposed in [101], and a Q-learning-based algorithm, namely Latency-Reliability using Q-learning (LR-Q), designed to improve KPIs of uRLLC solely. Simulation results show 29% and 21 times increase in eMBB users' throughput compared to LR-Q and PPF, respectively, at high traffic load of uRLLC (i.e., 2 Mbps). This causes less than 0.5 ms degradation in uRLLC users' latency at the 10^{-4} percentile compared to both LR-Q and PPF.

4.3.1 System Model

We follow 5G release-15 standard to verify our proposed algorithm on a set of gNBs that serve uRLLC and eMBB users. 5G standard provides a flexible resource allocation through variable length TTI. Let $\mathfrak{K}$ be a set of K RBGs, where the size of a RBG is $\lceil K_{RB}/K \rceil$ RBs. To limit the set of states in our Q-learning approach, we consider RBG as our unit of allocation in the frequency direction. Furthermore, each k^{th} RBG is allocated a transmission power, $p_{k,j}$, by j^{th} gNB. The Q-learning algorithm that is described in the following section, aims to improve the allocation of RBGs and their transmission power assignments.

According to our system model, each gNB holds a number of transmission buffers corresponding to the number of its attached users. Every TTI, downlink scheduler allocates resources to the active users (i.e., users with pending data transmissions). In particular, the scheduler performs joint power and RBG allocation while taking into account QoS demands of uRLLC and eMBB users. Traffic model of uRLLC users is composed of a

mixture of CBR and Poisson arrivals, whereas the traffic of eMBB users follows Poisson arrivals.

Capacity of a link between the i^{th} user and j^{th} gNB can be formulated as follows:

$$C_{i,j} = \sum_{k=1}^{K} \omega_k \ \log_2 \Big(1 + \frac{x_{k,i,j}\ p_{k,j}\ h_{k,i,j}}{\omega_k N_0 + \sum\limits_{\substack{j' \in \mathcal{J} \\ j' \neq j}} x_{k,i,j'}\ p_{k,j'}\ h_{k,i,j'}}\Big), \tag{4.6}$$

where ω_k is the bandwidth of k^{th} RBG and N_0 is AWGN single-sided power spectral density. $p_{k,j}$ is transmit power of j^{th} gNB on k^{th} RBG, $h_{k,i,j}$ is channel coefficient, and $x_{k,i,j}$ is RBG's allocation indicator of $(k,i,j)^{th}$ link. $p_{k,j'}$ is transmit power of j' interfering gNB, $g_{k,i,j'}$ is the channel coefficient, and $x_{k,i,j'}$ is allocation indicator of $(k,i,j')^{th}$ link with j' interfering node. Eq. (4.6) shows that interference mitigation plays a key role in enhancing throughput. As it is well-known, inefficient power allocation might impact edge-users significantly, which reduces the overall achieved throughput.

Latency of packets can be decomposed into three components as follows:

$$D = D^q + D^{tx} + D^{harq}, \tag{4.7}$$

where D^q is queuing delay, D^{tx} is transmission delay, and D^{harq} is round-trip delay of a HARQ re-transmission. During HARQ, a re-transmitted packet has higher priority than a new packet. Transmission delay of i^{th} user associated to j^{th} gNB can be calculated as follows:

$$D^{tx}_{i,j} = \frac{\xi_{i,j}}{C_{i,j}}, \tag{4.8}$$

where $\xi_{i,j}$ is the packet size and $C_{i,j}$ is the transmission rate. Eq. (4.8) shows that interference mitigation, hence optimal power allocation, plays a key role in transmission delay - besides throughput. On the other hand, transmission rate has an implication on the RLC layer. As the rate increases, less segmentation is observed. This consequently reduces the transmission delay. Furthermore, allocation of more RBGs to a user increases the size of the allocated transport block, which further decreases the transmission delay.

The queuing delay in (4.7) is identical to the scheduling delay of the MAC scheduler. As such, to achieve close to 1 ms delay for uRLLC users, the scheduler has to immediately schedule uRLLC traffic once it arrives and limit the number of HARQ re-transmissions. In particular, we assume only one HARQ re-transmission is allowed to achieve the lowest possible latency. However, limiting the number of re-transmissions can lead to higher PDR (i.e., lower reliability). Such low reliability can be more severe for edge-users. Thus, in

order to achieve high reliability while meeting the latency budget, RBG-based transmission power control is employed in our proposed algorithm.

It is worth noting that improving latency and reliability of uRLLC users is expected to impact the throughput performance of eMBB users (as in (4.6)). This calls for efficient resource allocation algorithm that balances the trade-off between KPIs of uRLLC and eMBB. In the following section, we present our proposed algorithm, based on Q-learning, for joint power and RB allocation in order to jointly optimize latency and reliability of uRLLC users as well as throughput of eMBB users.

4.3.2 Latency-Reliability-Throughput Improvement using Q-learning (LRT-Q)

The proposed algorithm is based on decentralized reinforcement learning, where each gNB acts as an agent running a Q-learning algorithm to perform resource allocation. The mathematical formulation of Q-learning relies on MDP which is defined by agents, states, actions, and reward function. The operation of Q-learning relies on interaction with the environment and learning from trial and error based rewards being given to accepted or favored actions. More specifically, an agent selects an action, executes it, and receives a reward that reflects the quality of the selected action. This process is repeated until the agent reaches a policy of action selection that maximizes its total discounted reward.

The proposed algorithm, LRT-Q, is a Q-learning algorithm with a reward function designed to improve latency and reliability of uRLLC users as well as throughput of eMBB users. In LRT-Q, actions are the joint power and RBG allocations performed by agents. To keep the size of the Q-table manageable, we group 8 consecutive RBs into a RBG, hence a RBG becomes the unit of allocation [102].

In LRT-Q, states are driven by observations from the environment which reflect the impact of actions of other agents. In particular, interference among users represent the major bottleneck against achieving better latency, reliability and throughput. As such, states are defined to capture the average SINR achieved by users attached to each gNB as follows:

$$s_{k,j} = \begin{cases} s_0, & \overline{\Gamma}_{k,j} \geq \Gamma_{th}, \\ s_1, & \text{otherwise}, \end{cases} \tag{4.9}$$

where $\overline{\Gamma}_{k,j}$ represents the average estimate of the SINR value of k^{th} RBG and defined as $\overline{\Gamma}_{k,j} = \beta\, \overline{\Gamma}^{u}_{k,j} + (1-\beta)\, \overline{\Gamma}^{m}_{k,j}$, where $\overline{\Gamma}^{u}_{k,j}$ is the average SINR of uRLLC users, $\overline{\Gamma}^{m}_{k,j}$ is the average SINR of eMBB users, and β is a factor controlling the priority given to uRLLC and

eMBB users. Γ_{th} is a threshold SINR value, which is chosen to maintain high probability of decoding. Finally, the reward function is formulated to reward actions that achieve the objectives of the proposed scheme:

$$r^u_{k,j} = \begin{cases} 1 - \max\limits_{i \in \mathcal{N}_r} (D^q_{i,j})^2, & \overline{\Gamma}^u_{k,j} \geq \Gamma_{th}, \\ -1, & \text{otherwise}, \end{cases} \tag{4.10}$$

$$r^m_{k,j} = \frac{2}{\pi} \tan^{-1}(\overline{C}^m_{k,j}), \tag{4.11}$$

$$r_{k,j} = \beta r^u_{k,j} + (1-\beta) r^m_{k,j}, \tag{4.12}$$

where $r^u_{k,j}$ is the reward of uRLLC users on k^{th} RBG, $r^m_{k,j}$ is the reward of eMBB users, and $r_{k,j}$ is the total reward of j^{th} gNB. $D^q_{i,j}$ represents the last packet queuing delay of i^{th} uRLLC user ($i \in \mathcal{N}_r$), and $\overline{C}^m_{k,j}$ is the average throughput of eMBB users. Eq. (4.12) serves in addressing the KPIs of both uRLLC and eMBB users through adjustment of parameter β. In particular, (4.10) aims at improving latency and reliability of uRLLC users where the agent is rewarded a value relative to the queuing delay as long as its reliability is meeting certain threshold (i.e., SINR threshold). Indeed, the reward value relies on the maximum queuing delay experienced by uRLLC users. This means that the algorithm will aim to improve the worst queuing delay. In addition, achieving better average SINR significantly contributes to the overall latency since better SINR leads to less packet segmentation and reduced transmission delay. Overall, (4.10) motivates the MAC scheduler to immediately allocate uRLLC users to better RBGs (i.e., hence achieving low-latency and high reliability simultaneously).

Eq. (4.11) serves in improving the throughput of eMBB users, where increased throughput leads to a reward value close to one. Using the parameter β in (4.12), we obtain the balance between the conflicting KPIs. Algorithm 4.1 presents the steps of LRT-Q algorithm performed by each agent. Furthermore, LRT-Q algorithm is compared with two baseline algorithms that are described in the following sections.

4.3.3 Baseline Algorithms: PPF and LR-Q

PPF is a PF-based scheme with priority given to uRLLC users. PPF is proposed in [101] and implemented here with the addition of equal power allocation. Simply, PPF allocates RBGs to uRLLC users with pending data transmission, then, it allocates the remaining RBGs to eMBB users.

The second baseline algorithm, LR-Q, is based on Q-learning, similar to the proposed scheme, however it only considers KPIs of uRLLC users (i.e., modeled using (4.10)).

Algorithm 4.1 LRT-Q

Initialization: Q-table $\leftarrow 0$, α, γ, and ϵ.
for TTI $t = 1$ to T **do**
    Step 1: Agent (i.e., gNB) receives uplink report (i.e., SINR) from its attached users.
    Step 2: Compute the reward using (4.10), (4.11), and (4.12).
    Step 3: Update the Q-value of the current state-action pair.
    Step 4: Observe and transit to next state as in (4.9).
    Step 5: Select the next action based on ϵ-greedy policy.
    Step 5: Repeat at **Step 1**.
end for

4.3.4 Performance Evaluation

Simulations are performed using our discrete-level simulator based on Matlab 5G toolbox. In our simulations, we consider 5 gNBs, each covering 10 uRLLC and 5 eMBB users. The traffic of uRLLC users is a mixture of 20% CBR and 80% Poisson arrivals, whereas traffic of eMBB follows Poisson arrivals only. Payload size is fixed to 32 bytes for all users. In addition, uRLLC traffic loads per cell is varied between 0.5 and 2 Mbps whereas eMBB traffic load is fixed to 0.5 Mbps. Simulation results are collected for 5000 TTIs and averaged over 10 simulation runs and presented with 95% confidence interval. In addition, we select the finest time resolution, i.e., TTI of 2 OFDM symbol, as our scheduling interval. The action space of Q-learning-based algorithms consists of the combination of power and RBG allocations. For a system bandwidth of 20 MHz, 13 RBGs are used where the first 12 RBGs contains 8 consecutive RBs while the last RBG contains 4 consecutive RBs. Maximum gNB's transmission power is set to 40 dBm [84] and power allocation, $p_{k,j}$, is drawn from the set $\{0, 1, 2, 3\}$ dBm. Finally, SINR threshold of $\Gamma_{th} = 20$ dB is used to maintain high probability of successful reception. Table 4.2 lists all the network and Q-learning settings considered in our simulations.

The performance of the proposed algorithm is evaluated in terms of KPIs of uRLLC and eMBB traffic, i.e., latency and reliability of uRLLC and throughput of eMBB. Fig. 4.7 presents the aggregate throughput of eMBB users in the presence of varying traffic loads of uRLLC users from 0.5 Mbps to 2Mbps offered load per cell. Indeed, increasing uRLLC's traffic load should impact the throughput performance of eMBB users. However, the proposed algorithm, LRT-Q, is able to maintain stability of throughput performance of eMBB users, with a slight degradation when the offered load is 2 Mbps. This shows a throughput increase of 29% compared to LR-Q and 21 times increase compared to PPF

Table 4.2: Network settings

Physical layer	
Network environment	3GPP UMa network
Carrier frequency	4 GHz
Bandwidth	$\omega = 20$ MHz
Numerology	15 KHz (numerology 0)
Number of RBG	$K = 13$
TTI size	2 OFDM symbols (0.1429 ms)
Max. transmission power	40 dBm [84]
Tx/Rx antenna gain	15 dB
Number of RBs	$K_{RB} = 100$ resource blocks
Propagation	$128.1 + 37.6 \log(d)$
Shadowing	Log-Normal (8 dB)
Noise figure	5 dB
Penetration loss	5 dB
MAC layer	
HARQ type	Asynchronous HARQ
HARQ round trip delay	4 TTIs
Number of HARQ processes	6
Maximum number of re-transmissions	1
Network model	
Number of gNBs	5
Inter-site distance	500 meter
User distribution	Stationary and uniformly distributed
Number of uRLLC users	50 (10 per cell)
Number of eMBB users	25 (5 per cell)
Traffic model	
uRLLC	20% CBR and 80% Poisson
eMBB	Poisson
Payload size	32 Byte
uRLLC load/cell	[0.5 : 0.5 : 2] Mbps
eMBB load/cell	0.5 Mbps
Q-learning	
Q-Learning rate (α)	0.5
Discount factor (γ)	0.9
Exploration probability (ϵ)	0.05
β	0.1

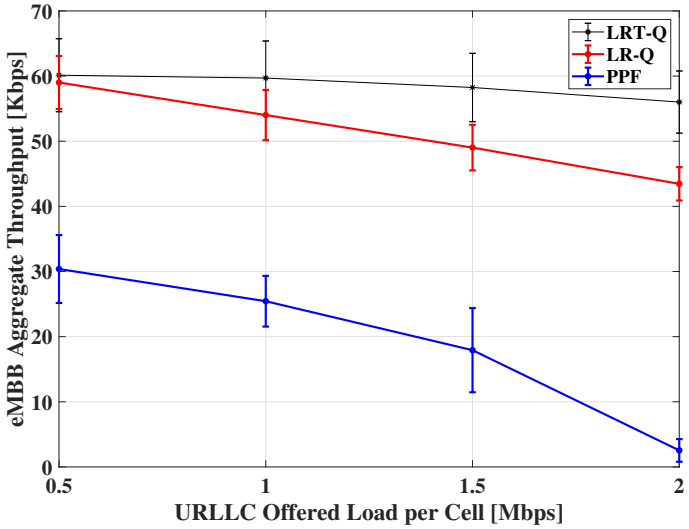

Figure 4.7: Aggregate throughput of eMBB users [Mbps] against uRLLC's traffic load.

algorithm even under the highest offered load scenario. Even when the offered load is 0.5 Mbps, the proposed algorithm has twice as much throughput than PPF.

Fig. 4.8 and Fig. 4.9 present the eCCDF of latency of uRLLC users in ms for uRLLC traffic loads [0.5, 1] Mbps and [1.5, 2] Mbps, respectively. The results are plotted in two figures in order to preserve readability. In Fig. 4.9, it can be observed that LRT-Q algorithm experiences less than 0.5 ms latency degradation at the 10^{-4} percentile compared to both LR-Q and PPF for high traffic load of uRLLC, i.e., 2 Mbps. It is worth mentioning that although PPF achieves better latency for uRLLC users compared to LRT-Q and LR-Q, its throughput is degrading faster than LRT-Q and LR-Q.

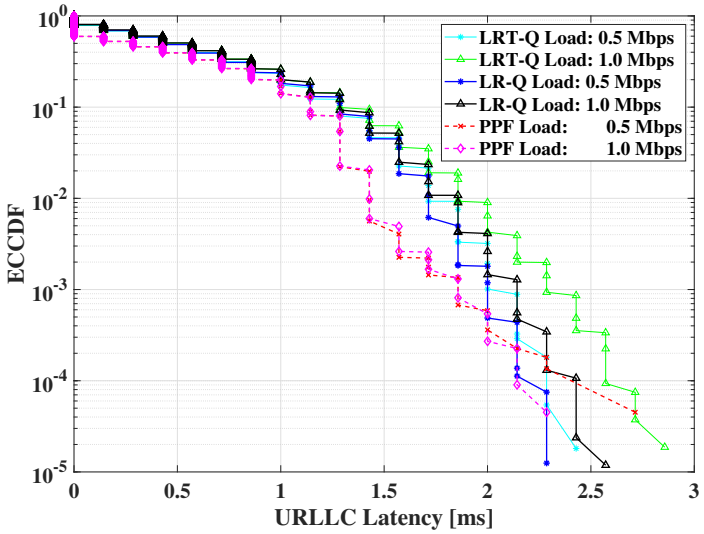

Figure 4.8: Average uRLLC latency [ms]; uRLLC load are 0.5 and 1 Mbps; eMBB load is 0.5 Mbps.

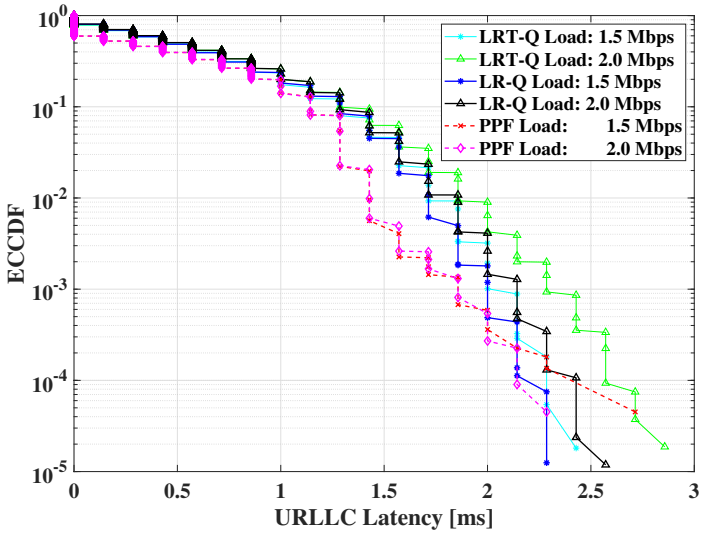

Figure 4.9: Average uRLLC latency [ms]; uRLLC load are 1.5 and 2 Mbps; eMBB load is 0.5 Mbps.

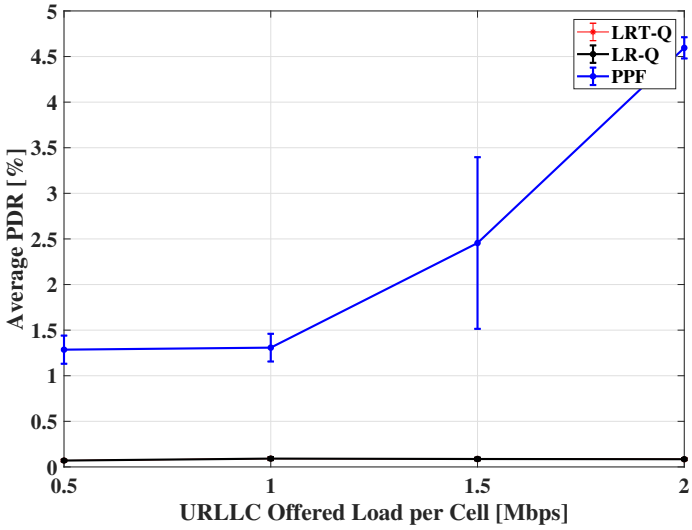

Figure 4.10: Average packet drop rate [%]; eMBB load is 0.5 Mbps.

Fig. 4.10 shows the PDR under varying traffic load of uRLLC users. Both LRT-Q and LR-Q achieve identical and very low PDR (0.06%), whereas the PDR of PPF increases rapidly with the load of uRLLC. Finally, we observed that both Q-learning algorithms converge after 3000 TTIs, i.e., 428.5 ms.

In summary, this section demonstrates the ability to combine joint allocation of RRM tasks in addition to considering multi-objective targets. However, with the advent of new 5G technologies, such as mm-wave, beamforming, and NOMA, more opportunities and challenges are also evolving. In particular, such technologies pose new interference patterns that make RRM tasks more challenging. In the next section, we present a reinforcement learning algorithm that addresses interference mitigation in a mm-wave network with beamforming and NOMA. In particular, RRM tasks are shaped in a different form, where power allocation per beam and user-beam allocation are becoming the core focus of the proposed algorithm.

4.4 Machine Learning-based Inter-Beam Inter-Cell Interference Mitigation in mm-Wave

With the explosive bandwidth demand of wireless devices, mm-wave is considered a promising solution to the spectrum scarcity problem. Mm-wave provides a large spectrum in the above-6 GHz band, i.e., Frequency Range 2 (FR-2). In contrast to sub-6 GHz band, FR-2 suffers from higher propagation losses that limit the coverage range of communication. Therefore, beamforming is used to combat mm-wave losses by reshaping the beam pattern of the antenna in the direction of the user, hence achieving better power density in the direction of propagation. On the other hand, NOMA is a promising multiple access technique for 5G and beyond 5G networks. The key idea in NOMA is to serve multiple users on the same time/frequency resources while superposing their messages in power domain, i.e., allocating different power levels to users' signals. This superposition process relies on the relative channel gains of the users such that users with better channel gains get less power levels, whereas users with bad channel gains get higher power levels. Successive Interference Cancellation (SIC) is applied at the users' side to remove inter-user interference. In particular, the user with the best channel decodes its message by successively decoding other users' messages and subtracting their effect from the received signal, whereas users with bad channel decode their respective signals directly [106].

Despite the significant spectral efficiency and capacity improvements that the aforementioned techniques bring about, several challenges hinder that performance gain. In

a downlink multi-beam scenario, the coverage of beams associated with different cells might intersect causing Inter-Beam Inter-Cell Interference (IB-ICI). Careful allocation of power to each beam, i.e., inter-beam power allocation, is essential for IB-ICI mitigation. Furthermore, the number of users covered by a single beam impacts the complexity and performance of SIC. As mentioned earlier, SIC performs successive decode and encode iterations to remove inter-user interference. Therefore, increasing number of users per beam leads to a large increase in complexity. Furthermore, the performance of SIC diminishes rapidly as the number of users increases [107]. To prevent SIC performance degradation, hence improve sum rate, it is imperative to balance the load across cells through user-cell association. In parallel to advances arising from mm-wave, beamforming and NOMA, there are significant efforts to make use of ML techniques to improve the performance of next-generation wireless networks [108].

We address IB-ICI by using reinforcement learning for joint user-cell association and inter-beam power allocation. In particular, we use a Q-learning algorithm that aims to enhance the sum rate of the network. Our results show that the proposed algorithm increases the achieved sum rate with at least 13% for the least offered traffic load with a convergence of about 286 ms. In addition, about 30% increase in sum rate is achieved under the highest traffic load simulated.

4.4.1 System Model

Notations: In the remainder of this section, bold face lower case characters denote column vectors, while non-bold characters denote scalar values. The operators $(.)^T$, $(.)^H$ and $|.|$ correspond to the transpose, the Hermitian transpose, and the absolute value, respectively. The operator $(A)^-$ under set B represents the absolute complement of A, i.e., $(A)^- = B \backslash A$.

Consider a downlink mm-wave-NOMA system with $J \in \mathcal{J}$ gNBs equipped with F transmit antennas and $N \in \mathcal{N}$ single-antenna users. Furthermore, users are partitioned into different clusters, $b \in \mathfrak{B}$, that are served using different beams such that $\mathcal{N}_b$ is the set of users covered by b^{th} beam. Let $\mathfrak{B}_j$ be the set of beams of j^{th} gNB. Henceforth, we use cluster and beam interchangeably. Indeed, different beams of different cells can have coverage intersection as shown in Fig. 4.11. Such intersection gives rise to IB-ICI. IB-ICI mitigation is essential in order to maximize network rate.

Poisson Cluster Process (PCP) is used to model users' deployment in the network, where the parent process follows a uniform distribution and the users of a cluster are uniformly deployed within a circular disk around the cluster center. Every gNB performs a clustering algorithm to group users that can be covered by a single beam. Under every beam, downlink

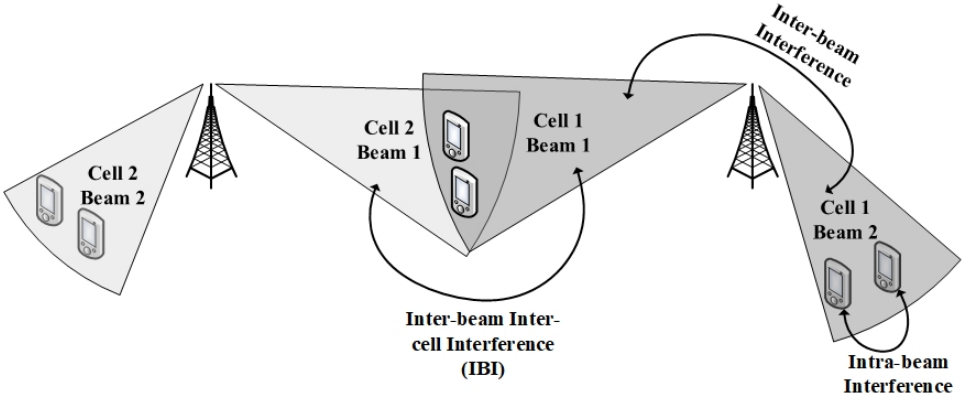

Figure 4.11: System model of mm-wave network using beamforing.

NOMA power allocation is used to multiplex users in the power domain, whereas users use SIC to demodulate their respective signals. We employ k-means clustering algorithm and the closed-form NOMA power allocation proposed in [22]. In particular, k-means is used to cluster users according to the correlation of their wireless channel properties, i.e., users with correlated channels are more likely to be located close to each other.

In mm-wave channels, the gain of the LoS path is significantly larger than the gain of the Non-LoS (NLoS) path, i.e., with around 20 dB [22], hence the mm-wave channel model can be simplified to a single-path LoS model as follows:

$$\boldsymbol{h}_{b,i,j} = \boldsymbol{v}(\theta_{b,i,j}) \frac{\varrho_{b,i,j}}{\sqrt{\zeta}(1 + d_{i,j}^{\eta})}, \tag{4.13}$$

where ζ is the number of paths, $\boldsymbol{h}_{b,i,j} \in \mathbb{C}^{\digamma \times 1}$ is the channel complex coefficient vector of i^{th} user and j^{th} gNB on b^{th} beam, i.e., link (b, i, j), $\varrho_{b,i,j} \in \mathbb{C}N(0, \sigma^2)$ is the complex gain, $d_{j,i}^{\eta}$ is the distance of link (i, j) with pathloss exponent η. In addition, $\boldsymbol{v}(\theta_{b,i,j})$ is the steering vector of the analog beamformer, which can be represented as follows:

$$\boldsymbol{v}(\theta_{b,i,j}) = [1, e^{-j2\pi \frac{\varkappa}{\lambda} \sin\left(\theta_{b,i,j}\right)}, ..., e^{-j2\pi(\digamma - 1) \frac{\varkappa}{\lambda} \sin\left(\theta_{b,i,j}\right)}]^T, \tag{4.14}$$

where $\varkappa$ is the gNB's antenna spacing, λ is the wavelength, and $\theta_{b,i,j}$ is the Angle of Departure (AoD). It is worth-mentioning that analog beamforming with 1D linear antenna array is used for its wide use.

4.4.2 Problem Analysis

In this work, we aim to improve the sum rate in mm-wave network by performing user-cell association and inter-beam power allocation. In particular, sum rate can be calculated as follows:

$$C = \omega \sum_{j \in \mathcal{J}} \sum_{b \in \mathfrak{B}_j} \sum_{i \in \mathcal{N}_k} \log_2(1 + \Gamma_{b,i,j}), \tag{4.15}$$

where ω is the bandwidth, and $\Gamma_{b,i,j}$ is the SINR of $(i, b, j)^{th}$ link, which can be expressed as

$$\Gamma_{b,i,j} = \frac{p_{b,j}\beta_{b,i,j}|\boldsymbol{h}_{b,i,j}^H \boldsymbol{w}_{b,j}|^2}{I_1 + I_2 + \sigma^2}, \tag{4.16}$$

$$I_1 = p_{b,j}|\boldsymbol{h}_{b,i,j}^H \boldsymbol{w}_{b,j}|^2 \sum_{\substack{i' \neq i \\ O(i') > O(i)}} \beta_{b,i',j}, \tag{4.17}$$

$$I_2 = \sum_{l \in \mathfrak{B}_{(j)^-}} p_l |\boldsymbol{h}_{l,i,(j)^-}^H \boldsymbol{w}_{l,(j)^-}|^2, \tag{4.18}$$

where $p_{b,j}$ denotes the power allocated to k^{th} beam of j^{th} gNB, and p_l is the power allocated to l^{th} interfering beam. $\beta_{b,i,j}$ and $\beta_{b,i',j}$ is the power allocation factor of $(b, i, j)^{th}$ and $(b, i', j)^{th}$ links respectively. $\boldsymbol{w}_{b,j}$ is the beamforming vector, and σ^2 represents the noise variance. The setup shown in Fig. 4.11 presents three types of interference: Intra-beam interference, IB-ICI, and inter-beam interference. Different beams are allocated different spectrum bands, hence inter-beam interference becomes void. With NOMA power allocation, users sharing the same time/frequency resources, are multiplexed in the power domain. This incurs intra-beam interference as expressed in (4.17). Finally, IB-ICI is expressed in (4.18). $O(i)$ denotes the decoding order of i^{th} user whereas $\mathfrak{B}_{j^-}$ denotes the set of beams that belong to the absolute complement of j under set $\mathcal{J}$, i.e., $((j)^- = \mathcal{J} \setminus j)$. Finally, $\boldsymbol{h}_{l,i,(j)^-}$ represents the channel vector between the l^{th} interfering beam from other cells in the set $(j)^-$ and i^{th} user, and $\boldsymbol{w}_{l,(j)^-}$ is the beamforming vector of l^{th} interfering beam.

4.4.3 Proposed Q-learning Algorithm

We define an online distributed multi-agent Q-learning algorithm as follows:

- **Agents:** gNBs.

- **Actions:** Each gNB decides on its user associations and inter-beam power allocation. The user-cell association is performed only for users that lie in the intersection region of two or more cells. Let $\mathcal{N}_j^{int}$ be the set of users of j^{th} gNB that lie in its intersection region with other cells. The vector of actions is defined as $\boldsymbol{a}_j = [\boldsymbol{\delta_j}, \boldsymbol{p_j}; j \in \mathcal{J}]$, where $\boldsymbol{a}_j \in \mathcal{A}_j$. The vector $\boldsymbol{\delta_j} = [\delta_{j,i}, i \in \mathcal{N}_j^{int}]$ represents a binary vector of user-cell association where each element indicates whether the gNB decides to associate the i^{th} user, $\delta_{j,i} = 1$, or not, $\delta_{j,i} = 0$. Furthermore, $\boldsymbol{p_j} = [p_{j,b}, b \in \mathfrak{B}_j]$ represents a vector that defines the power allocated to each beam of j^{th} gNB. As such the size of the action-space becomes $2^{|\mathcal{N}_j^{int}|} \times N_p^{|\mathfrak{B}_j|}$, where N_p represents the set of power values available for each beam.

- **States:** We define the states in terms of the average SINR which reflects the level of interference in the wireless environment:

$$s_j = \begin{cases} s_0, & \overline{\Gamma}_j \geq \Gamma_{th}, \\ s_1, & \text{otherwise}, \end{cases} \tag{4.19}$$

  where j^{th} gNB, i.e., agent, transits to state s_0 as long as its average SINR, $\overline{\Gamma}_j$ is greater than a threshold value, Γ_{th}, and transits to s_1 otherwise. The average SINR of j^{th} gNB is defined as follows:

$$\overline{\Gamma}_j = \frac{1}{(B \times N)} \sum_{b \in \mathfrak{B}_j} \sum_{i \in \mathcal{N}_b} \Gamma_{b,i,j}, \tag{4.20}$$

  where $\mathcal{N}_b$ is the set of users covered by b^{th} beam.

- **Reward:** We formulate the reward function based on SINR as follows:

$$r_j = \begin{cases} 1, & \overline{\Gamma}_j \geq \Gamma_{th}, \\ -1, & \text{otherwise}. \end{cases} \tag{4.21}$$

Algorithm 4.2 presents the steps performed by each gNB, whereas algorithm 4.3 presents the steps performed by each user. Furthermore, user-cell association process involves Q-learning part at gNB's side and priority list at user's side. In particular, the user maintains a priority list of the gNBs to associate with, which is computed according to SINR estimation in the last transmission interval. Afterwards, each gNB performs the Q-learning algorithm which results in an association decision for each user in the intersection region, where each user is informed about that decision. Finally, each user follows Algorithm 4.3 to combine the decisions from gNBs with its priority list and informs the selected gNB.

Algorithm 4.2 Proposed Q-learning algorithm for joint user-cell association and inter-beam power allocation (gNB)

Initialization: Q-table $\leftarrow 0$, α, γ, and ϵ.
for scheduling assignment period $t = 1$ to T **do**
  Step 1: Receive SINR estimations from attached users.
  Step 2: Perform Q-learning algorithm for joint user-cell association and inter-beam power allocation:
  Compute average SINR of users in the intersection region.
  Update reward as in (4.21).
  Update Q-value (and Q-table).
  Switch to state s' as in (4.19).
  if rand $\leq \epsilon$ **then**
    $\boldsymbol{a}_j \leftarrow$ draw an action uniformly from $\mathcal{A}_j$
  else
    $\boldsymbol{a}_j = \max_{a'_j \in \mathcal{A}_j} q(s'_j, a'_j)$
  end if
  Step 3: Downlink transmission of user-cell association decisions to each UE.
  Step 4: Wait UEs to perform final user-cell association decisions as in Algorithm 4.3.
  Step 5: Receive final user-cell association decisions from UEs.
  Step 6: Perform k-means clustering and NOMA intra-beam power allocation.
  Step 7: Perform downlink transmission, while each user performs downlink reception using SIC.
end for

4.4.4 Performance Evaluation

We use 5G Matlab Toolbox to construct a discrete event simulator. The simulator works on TTI level with 5G downlink transmission and reception. Table 4.3 presents the simulation settings. The network is composed of two gNBs with inter-gNBs distance of 150 m. Users are stationary and their positions follow a PCP with $\lambda = 7$. We consider 2 clusters, and cluster radius of 30 m. The performance of the proposed algorithm is tested under several traffic loads. The number of users in the intersection region is 2, number of power levels used is 5, number of clusters is 2, and number of states is 2. Hence, the size of the action-space becomes $2^2 \times 5^2 = 100$ and the size of the Q-table is $2^2 \times 5^2 \times 2 = 200$. In addition, we employ k-means clustering and closed-form NOMA power allocation proposed in [22] as a base for our implementation.

Algorithm 4.3 User-cell association (UE)

1: **for** scheduling assignment period $t = 1$ to T **do**
2: **Step 1:** Receive association decisions from gNBs.
3: **Step 2:** Update priority list, i.e. maintain gNBs that decided to associate and remove gNBs that decided not to associate with the UE.
4: **Step 3:** Select the gNB with the highest priority on the list to associate with and send the final decision to the selected gNB.
5: **end for**

The proposed algorithm is compared to a baseline algorithm that heuristically performs user-cell association and inter-beam power allocation. In particular, the baseline algorithm performs user-cell association by constructing a priority list of gNBs ordered according to SINR. Afterwards, users associate with the gNB with the highest priority on the list. In addition, power allocation is performed by equally dividing the total power of cell among its beams.

We present performance results of the proposed Q-learning algorithm compared with a baseline algorithm, Uniform Power Allocation (UPA), in terms of sum rate, latency, and PDR. In particular, UPA assigns equal power among beams. Fig. 4.12 presents the network sum rate versus the total offered load. The figure shows that the proposed scheme outperforms UPA in all cases with a rate increase of 13% and 33% at the lowest and highest offered loads, respectively. In addition, Fig. 4.13 presents the network sum rate versus the total number of users in the network. The figure shows that Q-learning is able to maintain a sum rate close to the total offered load (which is set to 0.5 Mbps for the presented case) when increasing the number of users in the network, whereas UPA is achieving lower sum rate. PDR is presented in Fig. 4.14, where both algorithms are achieving very comparable PDR (around $10 - 11\%$).

Furthermore, Fig. 4.15 shows the eCCDF of the average achieved latency. Latency is defined as the delay of the packet since its creation at the gNB until its delivery at the user side. This includes queuing, transmission, and propagation delays. The processing at both ends, i.e., gNB and user, includes RLC, MAC and physical layers. The figure shows that both algorithms achieve similar latency values at different offered loads. The figure also shows three main latency points: 0.1429 ms, 0.2857 ms, and 0.4286 ms, which correspond to 1, 2, and 3 TTIs respectively, where 1 TTI represents 2 OFDM symbols. In particular, queuing and re-transmission delays contribute to the total achieved delay [99]. By improving interference, i.e., SINR, re-transmission delay and total delay improve.

Finally, Fig. 4.16 shows the average cumulative reward versus the iteration number.

Table 4.3: 5G mm-wave network simulation settings

5G Physical layer configuration	
Bandwidth	20 MHz
Carrier frequency	30 GHz [109]
Subcarrier spacing	15 KHz
Subcarriers per resource block	12
TTI size	2 OFDM symbols (0.1429 msec)
Max transmission power	28 dBm
HARQ	
Type	Asynchronous HARQ
Round trip delay	4 TTIs
Number of processes	6
Maximum number of re-transmission	1
Distribution of users	
Mobility	Stationary
Distribution	PCP
PCP Average number of users	7
Number of clusters	2
Radius of cluster	30 m
Number of users	$4 - 16$
Number of gNBs	2
Inter-gNBs distance	150 m [109]
Traffic	
Distribution	Poisson
Packet size	32 Bytes
Q-learning	
Learning rate (α)	0.5
Discount factor (γ)	0.9
Exploration probability (ϵ)	0.1
Inter-beam power levels	$[0 : 2 : 8]$ dBm
SINR Threshold (Γ_{th})	20 dB
Simulation parameters	
Simulation time	4000 TTI
Number of runs	40
Confidence interval	95%

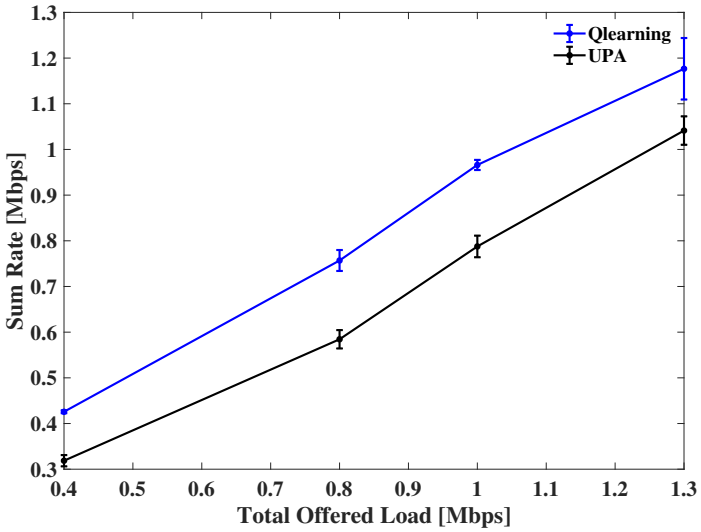

Figure 4.12: Sum rate [Mbps] versus total offered load [Mbps]. Number of users is 9.

The ϵ-greedy action selection methodology, presented on lines 9-13 in Algorithm 4.2, is applied for 2000 TTIs, whereas greedy policy is followed afterwards. The proposed algorithm converges at around 2500 TTI with a slight decrease of the reward at $500 - 600^{th}$ TTI due to the exploration policy.

In summary, the proposed algorithm demonstrates the ability to address interference challenges in mm-wave networks with beamforming by employing joint user-cell association and inter-beam power allocation. However, the mobility and traffic dynamicity give rise to spatial and temporal dynamicity in a mm-wave network with beamforming. This calls for an RRM solution that can adapt beams to different regions of users, i.e., spatial dynamicity, in addition to performing a flexible and fast RB allocation that considers traffic variations among beams, i.e., temporal dynamicity. The next section is devoted to addressing that problem using a deep reinforcement learning algorithm.

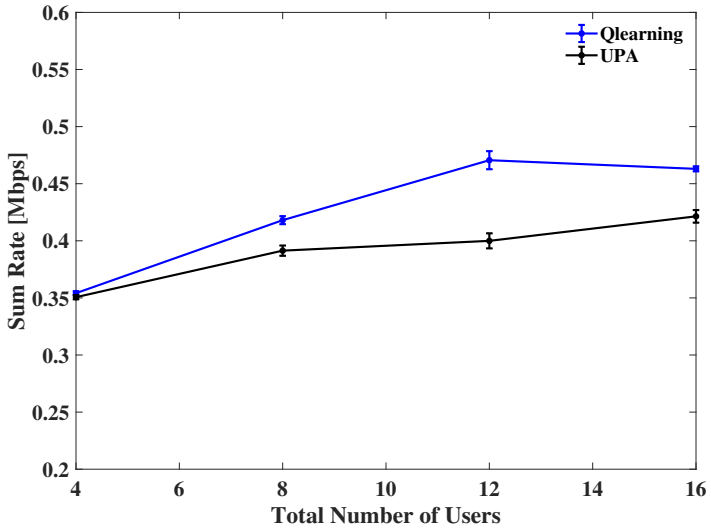

Figure 4.13: Sum rate [Mbps] versus total number of users with 0.5 Mbps total offered load.

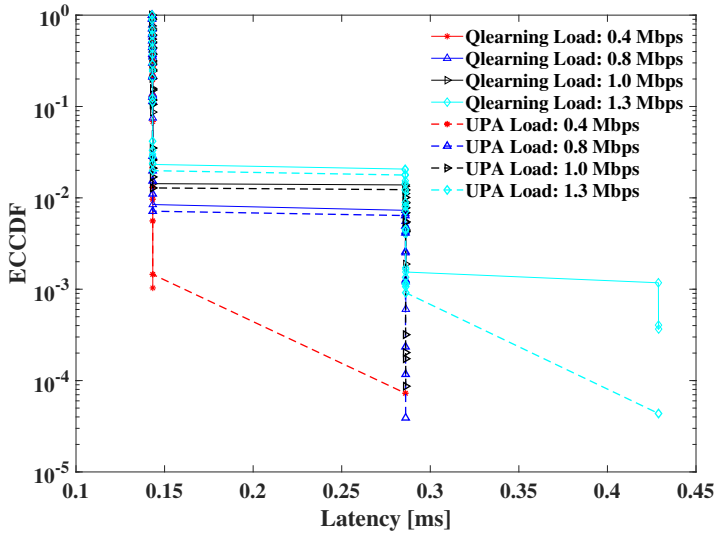

Figure 4.15: Average latency [ms] versus total offered load [Mbps].

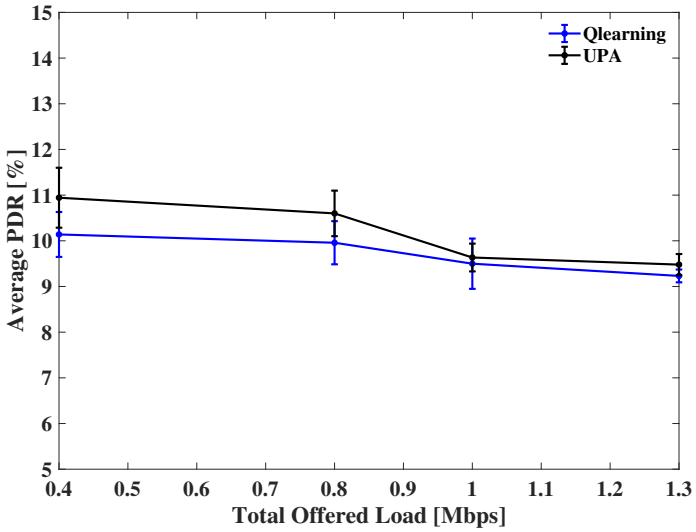

Figure 4.14: Average packet drop rate [%] versus total offered load [Mbps]. Number of users is 9.

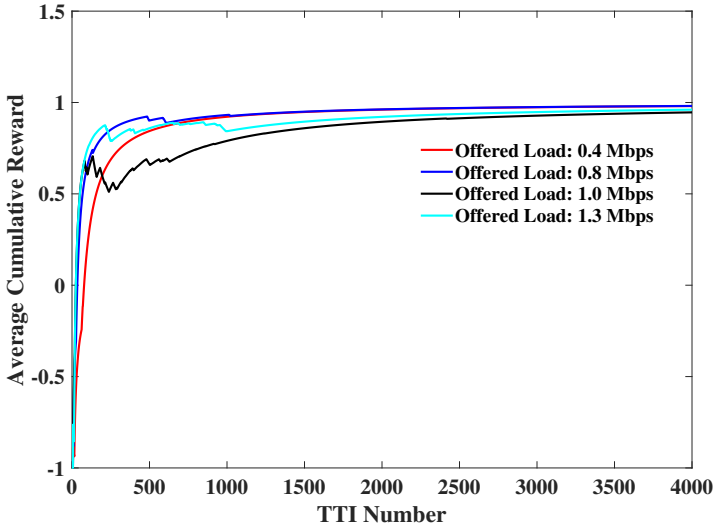

Figure 4.16: Cumulative average of Q-learning's reward versus iteration number with different total offered load.

4.5 Radio Resource and Beam Management in 5G Mm-wave Using Clustering and Deep Reinforcement Learning

With the unprecedented growth of mobile data traffic stemming from a growing use of data-hungry applications, next-generation wireless networks need to adopt a paradigm shift in the way the resources are managed. Mm-wave technology is promising large and underutilized spectrum between 30 and 300 GHz, which addresses the well-known spectrum scarcity problem of the sub-6 GHz band [110]. However, mm-wave suffers from high propagation losses that hinder its coverage range. One approach to combat such losses is to use directional communication where beamforming is used to reshape the pattern of propagation in the direction of the user.

Despite the performance gains that beamforming alongside mm-wave bring about, many challenges exist. The distribution of users and traffic can vary rapidly within a short period of time [111]. 5G standard introduced three service categories: uRLLC, eMBB, and mMTC [2]. In addition, wireless networks beyond 5G and 6G are expected to serve applications with more heterogeneity and tight QoS requirements [112]. Furthermore, an added layer of complexity arises due to mobility of users. With such network dynamicity, beam management and radio resource allocation becomes more challenging. This, first, calls for an intelligent beam management algorithm that captures QoS and mobility of users. Second, an intelligent radio resource allocation is needed to actively consider load variations across the formed beams.

We consider a heterogeneous mm-wave network that employs beamforming for serving uRLLC and eMBB users. Since users are mobile, an online clustering is sought to cluster users that can be served by a single beam. In addition, due to the fact that load per beam changes as users move among clusters, RB allocation is needed to efficiently allocate resources among users within the same beam. For this purpose, we propose a QoS-aware clustering and RB allocation technique for mm-wave networks. In particular, we propose a Density-Based Spatial Clustering of Applications with Noise (DBSCAN)-based algorithm for user clustering and managing beams, in addition to an LSTM-based deep reinforcement learning for RB allocation. We call our algorithm as Deep Q-learning with DBSCAN (DQLD). Furthermore, we compare the proposed algorithm to a baseline algorithm, namely K-means clustering with PPF (KPPF), where clustering is performed using K-means algorithm and resource allocation is performed using PPF algorithm. Simulation results reveal that DQLD outperforms KPPF in latency, reliability, and rate of uRLLC users as well as rate of eMBB users.

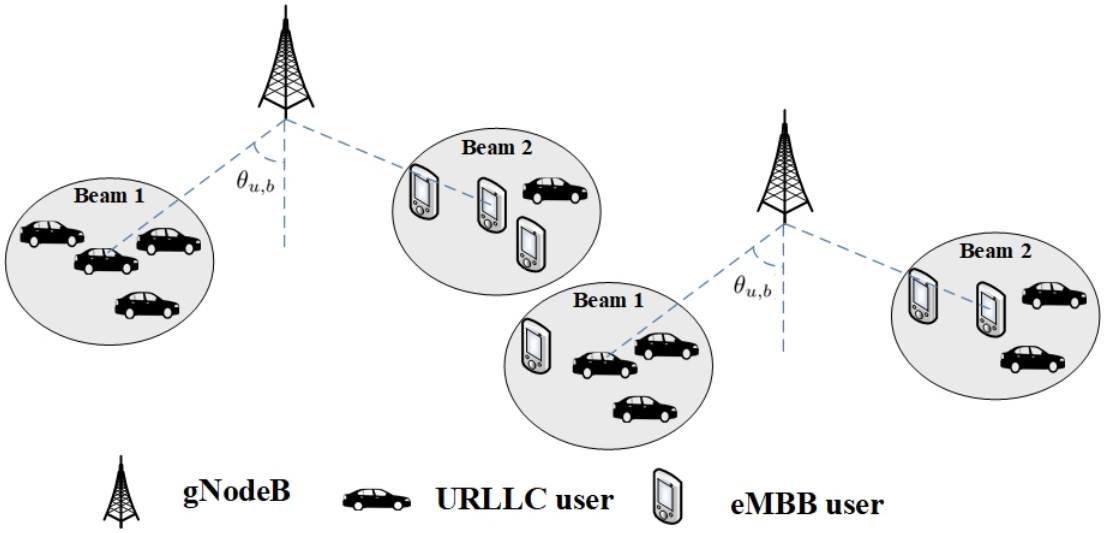

Figure 4.17: System model of 5G mm-wave network covering uRLLC and eMBB users using beamforming.

4.5.1 System Model

Consider a mm-wave network with $j \in \mathcal{J}$ of gNBs, where each gNB covers $N \in \mathcal{N}$ single-antenna users. Users are partitioned into different clusters, where each cluster is served by a single beam denoted by $b \in \mathfrak{B}$ as shown in Fig. 4.17. We consider two types of users with different QoS: uRLLC and eMBB users. In particular, uRLLC users require a low latency and high reliability communication, whereas eMBB users require high rate communication. Let $\mathcal{N}_b$ be the set of users covered by b^{th} beam and the communication between beams and their associated users follows 5G release 15 [113]. Furthermore, beams use OFDMA to allocate orthogonal resources to their users, hence intra-beam interference can be omitted. Let $k \in \mathfrak{K}$ denote a RBG and the bandwidth of a RBG is denoted by $\omega_{k,b}$. We select 2 OFDM symbols as the length of a TTI to encourage low latency for uRLLC users [94,101].

The initial positions of users follow PCP, in which heads of clusters are uniformly distributed and users within each cluster are uniformly distributed within the radius of the cluster. In addition, mobility of users follow random waypoint mobility model. The traffic of users follows Poisson distribution with λ inter-arrival time and a fixed packet size of 32 bytes. As such, users tend to leave their clusters and join new ones as time proceeds. The mm-wave channel can be modeled using a single LoS path model, where the gain of the LoS path is larger than the gain of NLoS paths [22]. As such, the channel vector,

$\boldsymbol{h}_{k,i,b} \in \mathbb{C}^{F \times 1}$, between b^{th} beam and i^{th} user on k^{th} RBG is defined as follows:

$$\boldsymbol{h}_{k,i,b} = \boldsymbol{v}(\theta_{i,b}) \frac{\varrho_{k,i,b}}{\sqrt{\zeta}(1 + d_{i,b}^{\eta})}, \tag{4.22}$$

where $\varrho_{k,i,b} \in \mathbb{C}N(0, \sigma^2)$ is the complex gain, ζ is the number of paths, $d_{i,b}$ is the Euclidean distance, and η is the pathloss exponent. Note that we removed the gNB's index to keep the formulation readable. In addition, $\boldsymbol{v}(\theta_{i,b})$ is the steering vector, which can be represented as follows:

$$\boldsymbol{v}(\theta_{i,b}) = [1, e^{-j2\pi \frac{\varkappa}{\lambda} \sin(\theta_{i,b})}, ..., e^{-j2\pi(F-1) \frac{\varkappa}{\lambda} \sin(\theta_{i,b})}]^T, \tag{4.23}$$

where $\varkappa$ is the gNB's antenna spacing, F is the number of antenna elements, λ is the wavelength, and $\theta_{i,b}$ is the AoD.

The proposed scheme aims at addressing the QoS differences among the users in the network. In particular, uRLLC users need to maintain high reliability and low latency communication links, whereas eMBB users need to achieve high rate.

4.5.2 Rate of eMBB Users

The sum rate of eMBB users per gNB is formulated as follows:

$$C = \sum_{b \in \mathfrak{B}} \sum_{i \in \mathcal{N}_b^e} \sum_{k \in \mathfrak{K}_b} x_{k,i,b} \omega_{k,b} \log_2(1 + \Gamma_{k,i,b}), \tag{4.24}$$

where $x_{k,i,b}$ is a RBG allocation indicator, $\omega_{k,b}$ is the size of RBG in Hz, $\mathcal{N}_b^e$ is the set of eMBB users that belong to b^{th} beam, and $\Gamma_{k,i,b}$ is the SINR of $(k,i,b)^{th}$ link, which can be expressed as

$$\Gamma_{k,i,b} = \frac{p_{k,b} |\boldsymbol{h}_{k,i,b}^H \boldsymbol{w}_{k,b}|^2}{\sigma^2 + \sum\limits_{b' \neq b} p_{k,b'} |\boldsymbol{h}_{k,i',b'}^H \boldsymbol{w}_{k,b'}|^2}, \tag{4.25}$$

where $p_{k,b}$ and $\boldsymbol{w}_{k,b}$ denote the power and beamforming vector of k^{th} RBG of b^{th} beam. $p_{k,b'}$ and $\boldsymbol{w}_{k,b'}$ denote the power and beamforming vector of k^{th} RBG of b'^{th} interfering beam. σ^2 represents noise variance.

4.5.3 Latency and Reliability of uRLLC Users

Latency of uRLLC users is formulated as follows:

$$D_{i,b} = D_{i,b}^{tx} + D_{i,b}^{q} + D_{i,b}^{harq}, \tag{4.26}$$

where $D_{i,b}^{tx}$ is the transmission latency, $D_{i,b}^{q}$ is the queuing latency (i.e., latency of packet pending in the transmission buffer), and $D_{i,b}^{harq}$ is the HARQ re-transmission latency of i^{th} user on b^{th} beam. In order to achieve low latency for uRLLC users, the scheduler has to immediately allocate resources to uRLLC traffic once it arrives. Furthermore, we limit the number of HARQ re-transmissions to 1 to maintain low latency.

Limiting the number of re-transmissions, however, may impact the reliability of uRLLC users. To maintain high reliability, link adaptation is performed, where users periodically report SINR measurements to the gNB in the form of CQI values. CQI indicates the quality (i.e., SINR) of the link with the associated beam. In turn, the gNB accounts for those measurements in the scheduling policy. In the following section, we show how the proposed algorithm addresses both latency and reliability of uRLLC users.

4.5.4 Deep Q-learning with DBSCAN (DQLD)

In order to maintain high QoS of uRLLC and eMBB in the midst of changing network conditions, the proposed algorithm considers an online clustering (for the purpose of beam management) and a ML-based resource allocation. Online clustering is used to cluster users that are adjacent to each other and can be covered by a single beam. In addition, the online clustering algorithm aims to find the optimal number of beams for coverage. On the other hand, for resource block allocation we use deep Q-learning. DBSCAN is used for online clustering and deep Q-learning is used for RB allocation.

An online algorithm is needed to maintain efficient coverage of mobile users. We adopt DBSCAN for user clustering and selection of number of beams due to its advantages over other clustering techniques [114]. DBSCAN does not require a predefined number of clusters. Instead, the algorithm identifies users that can belong to a cluster from sparse users and returns the number and structure of clusters. In addition, DBSCAN has low complexity and easy implementation.

Note that, frequent online clustering can lead to a challenging resource allocation problem. In particular, performing the clustering very often leads to frequent changes in the structure and number of beams. As such, resource allocation has to deal with a very dynamic environment. Furthermore, clustering might be needed only whenever the beams are not efficient enough to cover users (i.e., users have changed their positions and tend to belong to new clusters). Therefore, determining the frequency of clustering is important. We choose to perform clustering only when the average SINR of a beam drops under a predefined threshold.

Clustering returns a set of beams to cover network users. Within each beam, we perform RB allocation using an LSTM-based deep Q-learning technique, namely Deep Q-learning (DQL). The tuples of DQL is defined as follows:

- **Agents** DQL is a multi-agent distributed algorithm that is performed independently by each gNB (i.e., each gNB is a standalone agent). Each gNB performs DQL to allocate RBGs within each of its beams.

- **Actions** The actions are defined as the RBGs allocated to users per beam as

$$a_{k,b} = \{i_{k,b}\}, \tag{4.27}$$

  where $a_{k,b}$ denotes the action of k^{th} RBG of b^{th} beam, and $i_{k,b}$ is the user index.

- **States** We design the states in a way that captures the level of inter-beam interference. In particular, states are defined in terms of the CQI feedback measured at the user. Therefore, state of k^{th} RBG of b^{th} beam is defined as

$$s_{k,b} = \{\mathscr{g}_{k,b}\}, \tag{4.28}$$

  where $\mathscr{g}_{k,b}$ is the CQI of k^{th} RBG of b^{th} beam.

- **Reward** The reward function is designed to account for different users' classes (i.e., uRLLC and eMBB). In particular, uRLLC users require tight latency and reliability, whereas eMBB users require high throughput. Therefore, the reward function is defined as

$$r_{k,b} = \begin{cases} \text{sigm}(r_{k,b}^{(mbb)}\ r_{k,b}^{(llc)}), & C(i) = 1, \\ \text{sigm}(r_{k,b}^{(mbb)}), & C(i) = 2, \end{cases} \tag{4.29}$$

  where $\text{sigm}(x)$ denotes a sigmoid function defined as

$$\text{sigm}(x) = \frac{1}{1 + e^{-x}}. \tag{4.30}$$

  In (4.29), $C(i)$ represents the Quality Class Indicator (QCI) of i^{th} user, where $C(i) = 1$ denotes uRLLC users and $C(i) = 2$ denotes eMBB users. $r_{k,b}^{(mbb)}$ and $r_{k,b}^{(llc)}$ are the reward functions of eMBB and uRLLC users, respectively, which are defined as follows:

$$r_{k,b}^{(llc)} = \frac{D^{QoS}}{D_{k,b}(i)}, \tag{4.31}$$

$$r_{k,b}^{(mbb)} = \frac{\Gamma_{k,b}}{\Gamma^{QoS}}, \tag{4.32}$$

where $D_{k,b}(i)$ is the queuing latency due to allocating k^{th} RBG of b^{th} beam to i^{th} user, D^{QoS} is the QoS requirement of latency, and Γ^{QoS} is the QoS requirement of SINR. It is worth mentioning that gNB has knowledge of QCI of its radio bearers and queuing latency of its users. As such, when traffic on link (k, b) belongs to a uRLLC user, the reward constitutes a combination of reliability and queuing latency. Indeed queuing latency dominates the total latency of downlink transmission [105]. On the other hand, when traffic on link (k, b) belongs to a eMBB user, the reward constitutes reliability, which translates to higher transmission throughput (i.e., improving SINR enables allocation of higher modulation and coding scheme and higher transport block size). Finally, the sigmoid function is used to keep the reward in the interval $[0, 1]$.

Fig. 4.18 presents a conceptual diagram of the LSTM-based DQL approach. It is worth mentioning that gNB has a separate DQL entity for each beam it forms. For each beam, the DQL works as follows. The gNB computes the next state and the reward as in (4.28) and (4.29), respectively, from the CQI and SINR feedback received from its users. The experience, $\{s_t, a_t, r_{t+1}, s_{t+1}\}$, is then stored in the experience replay memory to be used later for training the LSTM neural network, where s_t, a_t, r_{t+1}, and s_{t+1} are state, action at t^{th} time step, reward, and next state at $(t+1)^{th}$ time step, respectively. Afterwards, LSTM is used to predict the Q-values of all actions of the next state (i.e., $q(s_{t+1}, \hat{a})$). Finally, the Q-values is fed to the ϵ-greedy algorithm for next action selection. The ϵ-greedy algorithm selects either a random action with probability (ϵ) or an action that follows the greedy policy with probability $(1 - \epsilon)$.

To maintain low complexity, the training of LSTM is performed every Ω TTIs. In particular, a batch of experience samples is drawn randomly from the experience replay memory. The batch is fed to the target LSTM in order to compute a sequence of reference responses. These responses constitute the labels used to train the main LSTM network. In addition, the target LSTM is initially loaded with the weights of the main network. However, the update of the target LSTM's weights is done every $\aleph$ TTIs to maintain stability [12].

4.5.5 Baseline Algorithm

For fair comparison, we use a baseline algorithm that works in a similar approach to the proposed algorithm and has been used in the literature before. In the baseline, k-means is

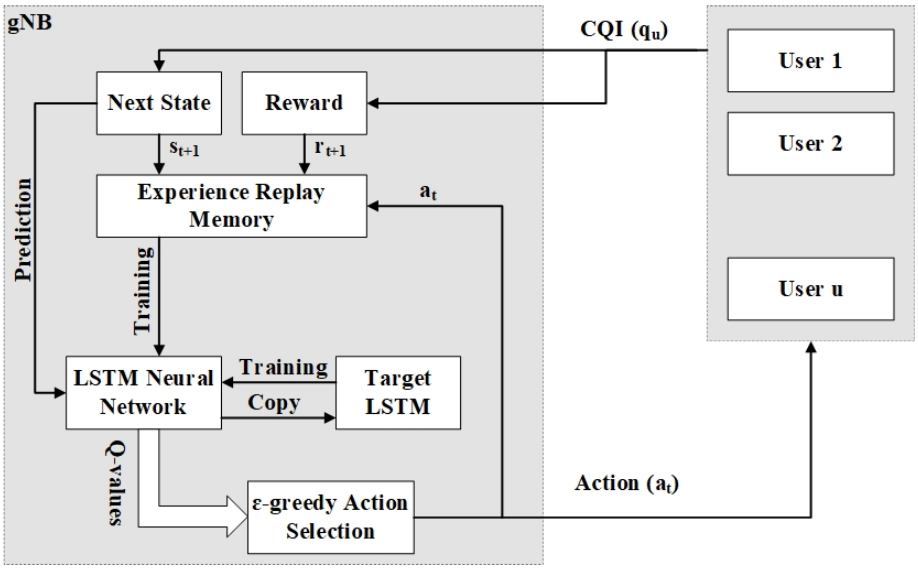

Figure 4.18: Conceptual diagram of LSTM-based deep Q-learning.

used to perform online clustering [22], whereas PPF is used for RB allocation as proposed in [101]. In [22], clustering using k-means is performed based on channel properties at the user side, i.e., users that are close in proximity are more likely to experience similar channels. Furthermore, in [22], RB allocation is performed using a hard QoS-aware criteria. In particular, RBGs are given to uRLLC users with pending data transmission first, then the remaining RBGs are allocated to eMBB users. Within each user class, RBGs are distributed according to PF criteria as

$$i^* = \arg\max \frac{C_{k,u,b}}{\overline{C}_{k,u,b}}, \tag{4.33}$$

where i^* is the selected user to be allocated k^{th} RBG.

4.5.6 Performance Evaluation

We perform simulation using a discrete event simulation based on 5G Matlab Toolbox. Table 4.4 presents network, simulator, and DQLD algorithm settings. The network is composed of two gNBs with 300m inter-gNBs distance. Initial positions of users follow a PCP with 2 clusters, where each cluster has a radius of 20m. The performance of the algorithms is tested under different traffic loads (i.e., $\{0.5, 1, 1.5, 2\}$ Mbps per gNB). The

DQLD algorithm consists of 15 states (i.e., corresponding to number of CQIs) and 24 actions (i.e., actions correspond to total number of users per gNB).

Table 4.4: Simulation settings

Physical layer	
Bandwidth	20 MHz
Carrier frequency	30 GHz [109]
Subcarrier spacing	15 KHz
Subcarriers per resource block	12
TTI size	2 OFDM symbols
Max transmission power	28 dBm
uRLLC target BLER	1%
eMBB target BLER	10%
HARQ	
Type	Asynchronous HARQ
Round trip delay	4 TTIs
Number of processes	6
Max. number of re-transmission	1
Network model	
Initial positions	PCP
Mobility	Random waypoint
number of uRLLC per cluster	2
number of eMBB per cluster	2
Number of clusters	3
Radius of cluster	20 m
Number of gNBs	2
Radius of cell	150 m
Inter-site distance	300 m [109]
Traffic	
Distribution	Poisson
Packet size	32 Bytes
Q-learning	
Learning rate (α)	0.5
Discount factor (γ)	0.9
Exploration probability (ϵ)	0.1
D^{QoS}	1 msec
Γ^{QoS}	15 dB [115]

LSTM	
Size of input layer	1
Number of hidden units	20
Size of output layer	24
Size of mini-batch	20
Size of replay memory	60
Training interval (Ω)	60
Copy Interval ($\aleph$)	120
DBSCAN	
$minPts$	5
eps	30
Simulation parameters	
Simulation time	1.5 second
Number of runs	10
Confidence interval	95%

In the following, we present the simulation results of the proposed DQLD scheme and compare it to the baseline KPPF algorithm. The performance is assessed in terms of uRLLC and eMBB QoS requirements. Fig. 4.19 and Fig. 4.20 present the eCCDF of latency of uRLLC users. The figures show the latency under increasing uRLLC traffic load. Both figures demonstrate the superiority of DQLD over KPPF despite that KPPF applies hard QoS rule for scheduling uRLLC users first. In particular, Fig. 4.19 demonstrates about 8 ms improvement at the 10^{-4} percentile and at 1 Mbps offered load. Furthermore, as offered load increases, KPPF fails to maintain a reasonable performance for uRLLC users.

In Fig. 4.20, the latency performance of KPPF degrades significantly, whereas DQLD was able to achieve much lower latency with about 350 ms difference with respect to KPPF at 2 Mbps.

The significant performance degradation of KPPF is attributed to a high Packet Loss Rate (PLR) as shown in Fig. 4.21. The figure presents the PLR of uRLLC users under different traffic loads. As seen in Fig. 4.21, DQLD demonstrates a 50% improvement in PLR compared to KPPF. Furthermore, Fig. 4.22 presents the achieved rate of uRLLC users under different uRLLC traffic load. Again, DQLD outperforms KPPF. In fact, increasing the traffic load impacts KPPF significantly, where 1 Mbps constitutes a break point for the algorithm. It is worth mentioning that the simulation is performed by increasing traffic

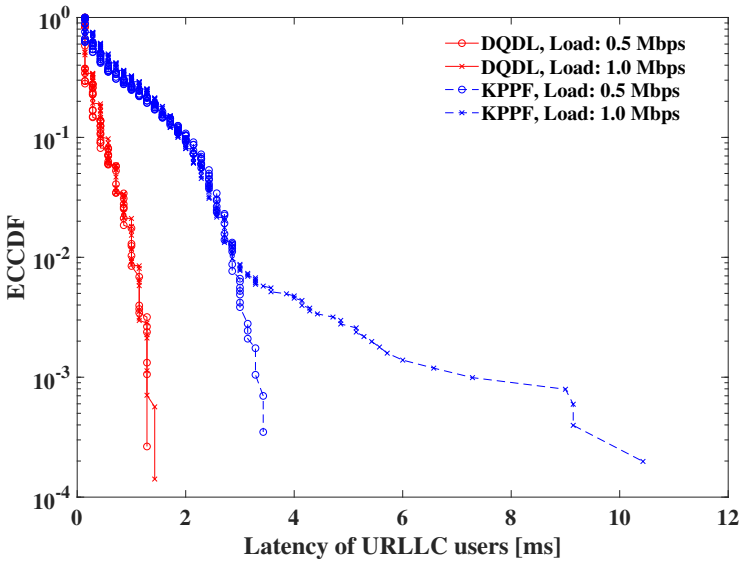

Figure 4.19: Latency of uRLLC users versus total uRLLC offered load ($[0.5, 1]$ Mbps).

loads of both uRLLC and eMBB users simultaneously. For example, 1 Mbps in Fig. 4.22 refers to uRLLC and eMBB loads (i.e., total load per gNB is 2 Mbps). As such, increasing the offered uRLLC and eMBB loads stresses both algorithms. Fig. 4.23 presents the achieved rate of eMBB users under different traffic load. Again, the same trend appears for KPPF, where 1 Mbps constitutes a break point in KPPF's performance, whereas DQLD demonstrates an ability to balance resources among users and satisfy the conflicting QoS requirements.

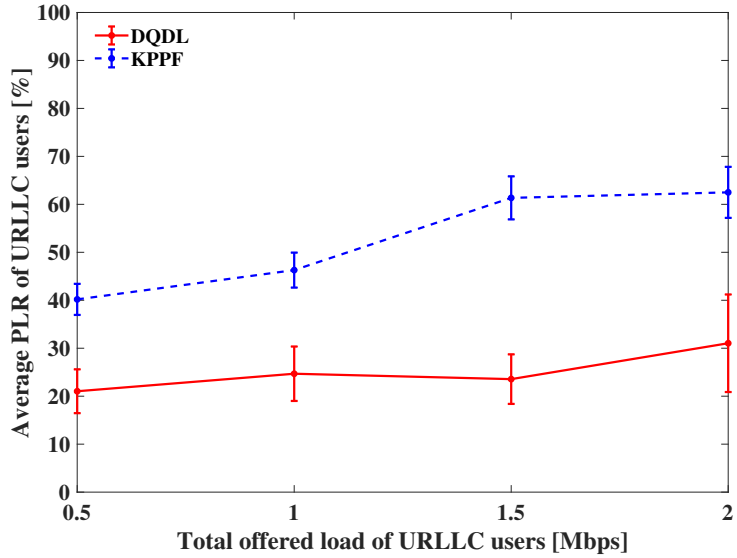

Figure 4.21: Packet loss rate of uRLLC users versus total uRLLC offered load.

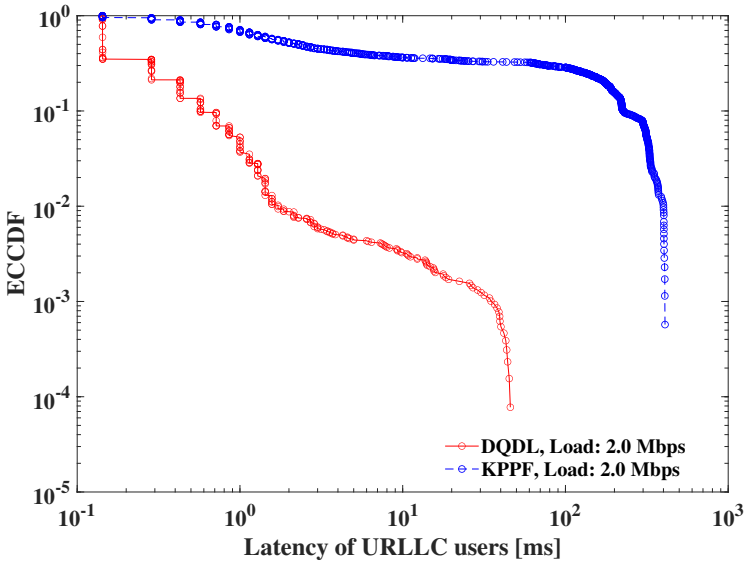

Figure 4.20: Latency of uRLLC users versus total uRLLC offered load ($[1.5, 2]$ Mbps).

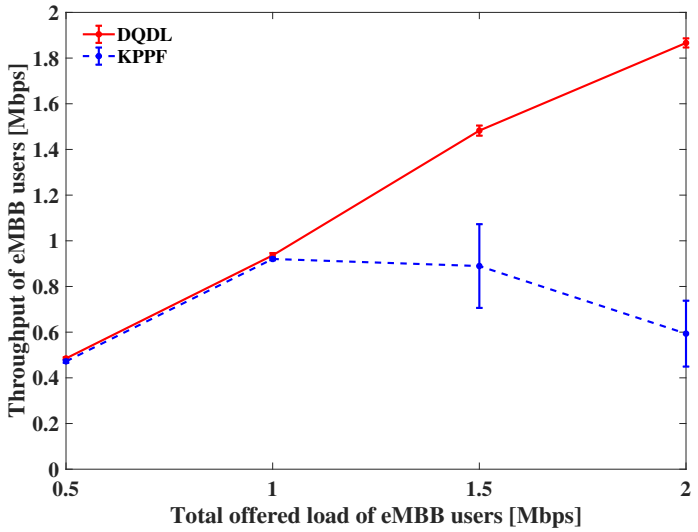

Figure 4.23: Sum rate of eMBB users versus total eMBB offered load.

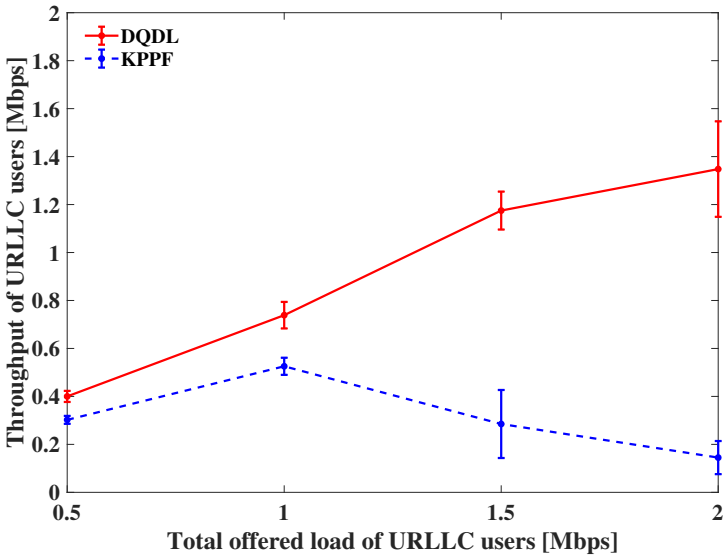

Figure 4.22: Sum rate of uRLLC users versus total uRLLC offered load.

4.6 Conclusion

In this chapter, we proposed reinforcement and deep reinforcement learning algorithms to improve latency, throughput, and reliability of uRLLC and eMBB users in 5G networks.

First, we presented LLHRQ, a Q-learning-based joint power and resource allocation technique for improving latency and reliability of uRLLC users. LLHRQ was crafted to address the main bottlenecks toward achieving high reliability and low latency. In particular, the reward and the states were formulated to improve inter-cell interference for reliability and transmission delay, on one hand, and improve the queuing delay, on the other hand. The proposed algorithm is compared to a modified PF algorithm, from the literature, that gives higher allocation priority to uRLLC over eMBB users. Simulation results show a 0.5ms latency improvement with LLHRQ, in addition to 4% improvement in packet drop rate. Furthermore, LLHRQ is able to maintain the throughput of eMBB users without any degradation.

Second, we extended the previous work to consider KPIs of both uRLLC and eMBB users simultaneously. LRT-Q algorithm based on Q-learning was proposed for joint power and RB allocation with the aim to improve both latency and reliability of uRLLC users as well as throughput of eMBB users. Results show 29% eMBB's throughput improvement with respect to LLHRQ, i.e., the Q-learning that considers uRLLC's performance solely.

Third, beamforming in mm-wave networks was employed to study the joint problem of user-cell association and inter-beam power allocation in 5G mm-wave network. Q-learning was used to improve network's sum rate by mitigating intra- and inter-beam interference. On one hand, the algorithm performs inter-beam power allocation such that it balances the interference posed by beams of adjacent cells. On the other hand, the algorithm performs user-cell association which balances users' attachments across cells, hence improving the performance of SIC. Simulation results present a performance enhancement of $13-30\%$ in network's sum-rate corresponding to the lowest and highest traffic loads, respectively.

Finally, we extended the previous work to consider the spatial and temporal dynamicity in the network by addressing clustering and RB allocation. In particular, we proposed an online clustering algorithm for identifying the number and structure of beams to cover network users, in addition to an LSTM-based deep reinforcement learning to perform resource allocation within each beam. The performance of the proposed scheme is compared to a baseline that uses k-means and PPF for clustering and RB allocation, respectively. Simulation results show 8ms latency improvement at low traffic load and about 350ms latency improvement at high traffic load. Furthermore, the proposed algorithm achieved 50% improvement in packet drop ratio.

Chapter 5

Transfer Reinforcement Learning for 5G Networks

5.1 Introduction

In this chapter, we present transfer in reinforcement learning for 5G wireless networks. In traditional reinforcement learning, a learning agent needs to collect vast number of samples of experience in order to reach an optimal result. In the previous chapters, we introduced deep reinforcement learning as a solution to speed up the convergence. However, the proposed methods were able to improve the convergence of the agent under fixed domain scenario, in which there is a certain limit of the agent's adaptability. Furthermore, with the advent of novel network architectures, such as cloud RAN and open RAN, more autonomy is needed reduce the capital expenditure (CapEx) of network setup and maintenance. Therefore, fast adaptability to network environments is needed. In this chapter, we introduce transfer in reinforcement learning, where we seek to transfer knowledge between expert and learner agents. This, on one hand, addresses the convergence problem of traditional reinforcement learning methods and, on the other hand, facilitates knowledge transfer across different domains, i.e., different state-action spaces. The latter is expected to revolutionize the wireless network by introducing agents that are able to learn from other agents.

The work presented in this chapter represents an attempt to transfer knowledge from an expert to a learner agent. This will help improve the convergence of the learner and reduce the need for large sample collection, i.e., exploration iterations. Furthermore, we

perform transfer across domains, where knowledge are transferred from a less complex to a more complex task.

5.2 Transfer Reinforcement Learning for 5G-NR mm-wave Networks

5.2.1 Introduction

Next-generation wireless networks are expected to carry heterogeneous traffic loads with QoS expectations including high capacity, low latency and enhanced reliability [2,112,116]. The recent availability of mm-wave band between 30 and 300 GHz is a promising solution for spectrum scarcity in the next-generation wireless networks, where wide bandwidth can be provided for high data rate services. Indoor environments such as schools, hospitals, and shops, as well as outdoor environments such as parks, and city centres are examples of regions of mm-wave support [117]. However, the coverage of mm-wave systems is limited due to the poor propagation characteristics of mm-wave signals and their sensitivity to blockages such as buildings and people. In order to overcome such signal degradation, mm-wave systems utilize directional communication through a large number of antennas (i.e., they use beamforming [118]). In addition, power domain NOMA provides opportunities to increase the spectral efficiency of wireless networks by superposing signals of multiple users on the same time and frequency resources while allocating different power levels to those signals [119–121]. In consequence, SIC technique is needed at the receiver side to demodulate respective users' signals [122].

Despite the capacity gains promised by integrating mm-wave, beamforming and NOMA, many technical challenges must be overcome, one of them being interference related performance degradation. In particular, intra-beam interference and inter-cell interference hinder such promised capacity gains. With NOMA, users' signals are superposed on the same time/frequency resources with different power levels. In turn, this incurs intra-beam interference which degrades the decoding performance of SIC technique (i.e., decoding performance of SIC diminishes rapidly as the number of users per beam increases [107]). Furthermore, inter-cell interference arises due to intersection among beams that belong to different cells. Consequently, balancing the number of users covered by different beams is needed to maintain high performance of SIC. In order to accomplish this, we propose a joint user-cell association and number of beams selection for sum rate maximization in a 5G mm-wave network.

Several works in the literature have addressed the problem of sum rate maximization in mm-wave networks. For example, in [22], the authors address inter-cluster and intra-cluster interference in a mm-wave network for sum rate maximization through users clustering and NOMA power allocation. With beamforming, adjacent users tend to have correlated channel characteristics. As such, the authors propose a k-means algorithm that cluster users according to their channel features. Furthermore, they derive optimal NOMA power allocation policy in a closed form. With the aid of coalitional game theory, authors in [123] propose a low complexity algorithm for users clustering in a single-cell mm-wave system with the aim to maximize sum rate of the system. An optimal power allocation within each cluster has been proposed thereafter.

The previous works consider a single-cell scenario that aims to solve the clustering and power allocation problem. This corresponds to a centralized approach. In contrast, we consider a multi-cell scenario where each cell acts as an independent agent (i.e., multi-agent scenario) that aims to mitigate interference by solving the joint user-cell association and number of beams selection. Furthermore, the previous works employ a closed-form optimization technique which is adding a prohibitive complexity in implementation. And finally, we propose three machine learning-based algorithms and analyze their performance based on the network scenario with different user deployments and under mobility. More specifically, a transfer reinforcement learning technique is proposed, where knowledge from an expert is transferred to a learner. This helps in utilizing samples of experience efficiently, hence speeding up the convergence of the learner task.

5.2.2 Transfer Reinforcement Learning

We adopt the Transfer via Inter-Task Mapping (TvITM) approach proposed in [124], where we consider transfer occurring from a single source task to a single target task. Fig. 5.1 presents a conceptual model for TvITM approach. In particular, the expert's reinforcement learning task is defined as an MDP with the four-element tuple $\{s_s, a_s, T_s, r_s\}$, where s_s is the state, a_s is the action, T_s is the transition function, and r_s is the reward function of the source (expert) task. Similarly, the learner's task is defined as an MDP with the tuple $\{s_t, a_t, T_t, r_t\}$. While the state-action space defines the domain, the transition and reward functions define the objective of the task. As such, if both source and target tasks have the same state-action space, the transfer is said to be across fixed domain, otherwise it is a transfer across different domains. TvITM works as shown in Fig. 5.1. The state-action pair of the target, (s_t, a_t), are mapped to the state-action pair of the source, (s_s, a_s), via a state and action mapping functions, ϕ_s and ϕ_a, respectively. Afterwards, the Q-value, q_s,

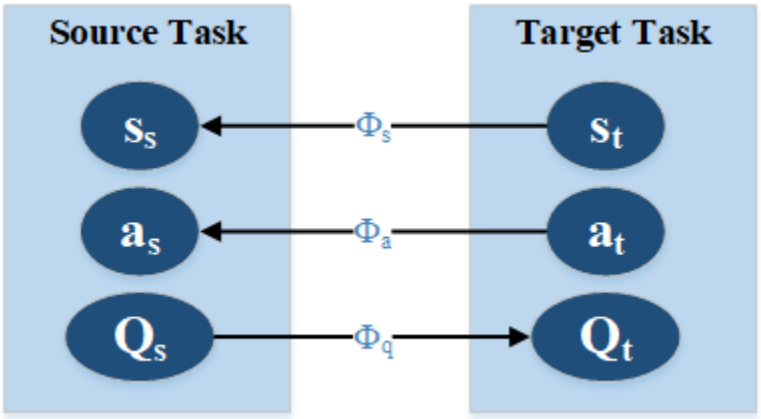

Figure 5.1: Transfer via inter-task mapping.

corresponding to (s_t, a_t) is retrieved from the Q-table of the source task and mapped to a Q-value of the target task, q_t, via a mapping function ϕ_q.

5.2.3 System Model

Notations: In the remainder of this section, bold face lower case characters denote column vectors, while non-bold characters denote scalar values. The operators $(.)^T$, $(.)^H$ and $|.|$ correspond to the transpose, the Hermitian transpose, and the absolute value, respectively.

Consider a downlink mm-wave-NOMA system with $e \in \mathfrak{E}$ and $l \in \mathfrak{L}$ expert and learner gNBs respectively as shown in Fig. 5.2. It is worth noting that expert and learning gNBs are spatially separated with no intersection zones. More specifically, Fig. 5.2a and Fig. 5.2b are used in conjunction when applying transfer reinforcement learning, whereas Fig. 5.2b is only used in case of Q-learning and Best SINR association with DBSCAN (BSDC). In addition, we consider two scenarios for users deployment. The first scenario considers stationary users, where initial positions follow PCP. In PCP, the parent process follows a uniform distribution and the users of a cluster are uniformly deployed within a circular disk around the cluster center. The second scenario considers random waypoint mobility, where initial positions of the users follow PCP distribution.

Expert gNB (Only for Transfer Reinforcement Learning): Expert gNBs are equipped with F antennas to communicate with its associated single-antenna UEs, which constitute a Multiple Input Single Output (MISO) scenario. In addition, downlink NOMA power allocation is used to multiplex messages of UEs in the power domain (i.e., allocating

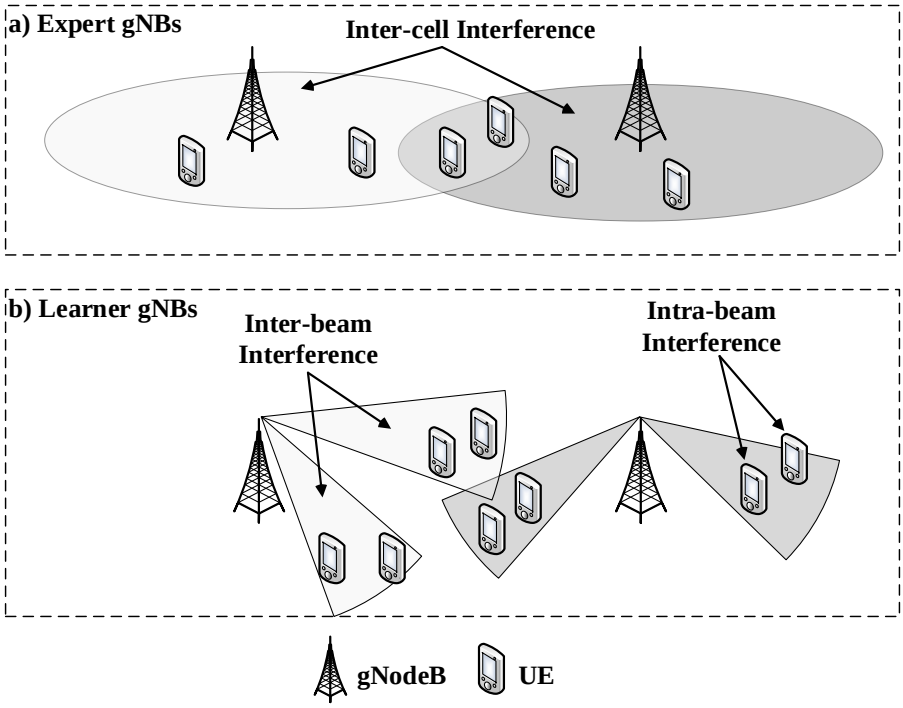

Figure 5.2: Network model of transfer learning in reinforcement learning with two expert and two learner gNBs.

different power levels to signals of different UEs). Consequently, UEs should employ SIC technique to demodulate their respective signals. The mm-wave channel between an expert gNB and its associated UE can be considered as a single-path mm-wave channel since the gain of the LoS path is significantly larger than the gain of the NLoS path [125]:

$$\boldsymbol{h}_{i,e} = \boldsymbol{v}(\theta_{i,e}) \frac{\varrho_{i,e}}{\sqrt{\zeta}(1 + d_{i,e}^{\eta})}, \tag{5.1}$$

where ζ is the number of paths, $\boldsymbol{h}_{u,e} \in \mathbb{C}^{F \times 1}$ is the channel complex coefficient vector of i^{th} UE and e^{th} learner gNB (i.e., link (i,e)), $\varrho_{i,e} \in \mathbb{C}N(0, \sigma^2)$ is the complex gain, $d_{i,e}^{\eta}$ is the distance of $(i,e)^{th}$ link with pathloss exponent η. In addition, $\boldsymbol{v}(\theta_{i,e})$ is the steering vector, which is represented as follows:

$$\boldsymbol{v}(\theta_{i,e}) = [1, e^{-j2\pi \frac{\varkappa}{\lambda} \sin(\theta_{u,e})}, ..., e^{-j2\pi(F-1) \frac{\varkappa}{\lambda} \sin(\theta_{u,e})}]^T, \tag{5.2}$$

where $\varkappa$ is the gNB's antenna spacing, λ is the wavelength, and $\theta_{u,e}$ is the AoD.

We employ Q-learning algorithm for expert gNBs for sum rate maximization. In particular, expert gNBs aim at improving the sum rate through user-cell association. Sum rate can be modeled as follows:

$$C_e = \sum_{e \in \mathfrak{E}} \sum_{i \in \mathcal{N}_e} \omega \log_2(1 + \Gamma_{u,e}), \tag{5.3}$$

where ω is the bandwidth, $\mathcal{N}_e$ is the set of UEs covered by e^{th} expert gNB, and $\Gamma_{i,e}$ is the SINR of $(i,e)^{th}$ link, which can be expressed as

$$\Gamma_{i,e} = \frac{p_e \beta_{i,e} |\boldsymbol{h}_{i,e}|^2}{\sum\limits_{\substack{e' \in \mathfrak{E} \\ e' \neq e}} p_{e'} |\boldsymbol{h}_{i,e'}|^2 + \sigma^2}, \tag{5.4}$$

where p_e denotes the power of the e^{th} expert gNB, and $\beta_{i,e}$ is the NOMA power allocation factor of $(i,e)^{th}$ link. $\boldsymbol{h}_{i,e'}$ represents the channel vector between the e' interfering expert gNB and i^{th} UE, and σ^2 represents the noise variance.

Learner gNBs: Learner gNBs are equipped with F uniform array transmit antennas for mm-wave beamforming. gNBs use a clustering algorithm to group UEs that can be covered by a single beam, forming up to $B \in \mathfrak{B}$ beams. Henceforth, we use cluster and beam interchangeably. Within each beam, downlink NOMA power allocation is used to multiplex messages of UEs in the power domain, and UEs use SIC technique at the receiver side.

Furthermore, we consider that learner gNBs use k-means clustering algorithm and closed-form NOMA power allocation as proposed in [22]. In particular, the k-means algorithm clusters UEs according to the correlation of their wireless channel properties (i.e., users with correlated channels are more likely to be located close to each other).

The mm-wave channel follows a single-path model represented as follows:

$$\boldsymbol{h}_{b,i,l} = \boldsymbol{v}(\theta_{b,i,l}) \frac{\varrho_{b,i,l}}{\sqrt{\zeta}(1 + d_{i,l}^{\eta})}, \tag{5.5}$$

where $\boldsymbol{h}_{b,i,l} \in \mathbb{C}^{\digamma \times 1}$ is the channel complex coefficient vector of i^{th} UE and l^{th} learner gNB on b^{th} beam (i.e., link (i, b, l)), $\varrho_{b,i,l} \in \mathbb{C}N(0, \sigma^2)$ is the complex gain, $d_{l,i}^{\eta}$ is the distance of $(i, l)^{th}$ link with pathloss exponent η. In addition, $\boldsymbol{v}(\theta_{b,i,l})$ is the steering vector, which is represented as follows:

$$\boldsymbol{v}(\theta_{b,i,l}) = [1, e^{-j2\pi \frac{\varkappa}{\lambda} \sin\left(\theta_{b,i,l}\right)}, ..., e^{-j2\pi(\digamma - 1) \frac{\varkappa}{\lambda} \sin\left(\theta_{b,i,l}\right)}]^T, \tag{5.6}$$

where $\varkappa$ is the gNB's antenna spacing, λ is the wavelength, and $\theta_{b,i,l}$ is the AoD.

The objective of learner gNBs is equivalent to expert gNBs' objective, which is improving the sum rate of the network. However, learner gNBs aim to accomplish this through joint user-cell association and selection of the number of beams. In particular, sum rate of learner gNBs can be calculated as follows:

$$C_l = \sum_{l \in \mathfrak{L}} \sum_{b \in \mathfrak{B}_l} \sum_{i \in \mathcal{N}_b} \omega \log_2(1 + \Gamma_{b,i,l}), \tag{5.7}$$

where $\mathfrak{B}_l$ is the set of beams formed by l^{th} gNB, and $\mathcal{N}_b$ is the set of UEs covered by b^{th} beam. $\Gamma_{b,i,l}$ is the SINR of $(i, b, l)^{th}$ link, which can be expressed as

$$\Gamma_{b,i,l} = \frac{p_{b,l} \beta_{b,i,l} |\boldsymbol{h}_{b,i,l}^H \boldsymbol{w}_{b,l}|^2}{I_1 + I_2 + \sigma^2}, \tag{5.8}$$

$$I_1 = p_{b,l} |\boldsymbol{h}_{b,i,l}^H \boldsymbol{w}_{b,l}|^2 \sum_{\substack{i' \neq i \\ O(i') > O(i)}} \beta_{b,i',l}, \tag{5.9}$$

$$I_2 = \sum_{l \in \mathfrak{L}} \sum_{\substack{b' \in \mathfrak{B}_l \\ b' \neq b}} p_{b'} |\boldsymbol{h}_{b',i,l}^H \boldsymbol{w}_{b',l}|^2, \tag{5.10}$$

where $p_{b,l}$ denotes the power allocated to b^{th} beam of l^{th} gNB, $\beta_{b,i,l}$ is the power allocation factor of $(b, i, l)^{th}$ link, and $\boldsymbol{w}_{b,l}$ is the beamforming vector. I_1 in (5.9) represents intra-beam interference caused by NOMA power allocation (i.e., UEs under the same beam

share the same time/frequency resources). In addition, I_2 in (5.10) represents inter-beam interference. $O(i)$ denotes the decoding order of i^{th} user. Finally, $\boldsymbol{h}_{b',i,j}$ represents the channel vector between the b' interfering beam and i^{th} user.

5.2.4 Transfer Reinforcement Learning

Expert (Q-Learning): Conventional Q-learning has been adopted for the expert gNBs. In particular, the state, s_e, of e^{th} expert gNB is formulated to capture the level of interference represented as follows:

$$s_e = \begin{cases} s_0, & \overline{\Gamma}_e \geq \Gamma_{th}, \\ s_1, & \text{otherwise}, \end{cases} \tag{5.11}$$

where Γ_{th} is the SINR's threshold for successful packet decoding, and $\overline{\Gamma}_e$ is the average SINR of e^{th} expert gNB due to downlink transmission to its associated users, which can be formulated as follows:

$$\overline{\Gamma}_e = \frac{1}{N_e} \sum_{i \in \mathcal{N}_e} \Gamma_{i,e}, \tag{5.12}$$

where $\Gamma_{i,e}$ is the SINR of i^{th} user associated to e^{th} expert gNB. The state s_0 is visited whenever the achieved average SINR meets the minimum threshold, and s_1 is visited otherwise. The expert's actions are the user-cell association decision which is formulated as $\boldsymbol{a}_e = [\boldsymbol{\delta}_{i,e}; i \in \mathcal{N}_e, e \in \mathcal{E}]$, where $\boldsymbol{\delta}_{i,e}$ represents a logical indicator of i^{th} UE's association to e^{th} gNB. The reward function of e^{th} gNB, r_e, is formulated using a sigmoid function as follows:

$$r_e = \frac{1}{1 + e^{-0.5(\overline{\Gamma}_e - 0.5\Gamma_{th})}}. \tag{5.13}$$

Eq. (5.13) implies that a better SINR value, $\overline{\Gamma}_e$, rewards the expert gNB with higher reward value.

Learner (Transfer Q-learning (TQL)): The learner gNB employs a TQL approach which is based on the framework of TvITM in [124]. In particular, the ultimate goal of TQL's agent is to speed up its learning process in a target task by mapping a learned value function (i.e., Q-value function) of a different but related source task. In TQL, the target task is performed by the learner gNB (i.e., joint user-cell association and selection of number of beams), whereas the source task is performed by the expert gNB (i.e., user-cell association). In addition, we assume that knowledge of expert gNB (i.e., converged Q-table) becomes available to learner gNB before the latter starts its learning process.

Therefore, the formulation of TQL is similar to conventional Q-learning with the addition of a mapping function. In particular, the mapping function is used to import a Q-value from the expert gNB's knowledge domain. Such Q-value acts as a signal to guide, and speed up, the Q-learning algorithm of the learner gNB. The following lines explain the MDP tuples of TQL.

- **Agents:** TQL is a multi-agent distributed solution. As such, learner gNBs are considered the TQL's agents. It is worth mentioning that gNBs are non-cooperative (i.e., they do not share information among themselves).

- **States:** The state, s_l, of l^{th} learner gNB is equivalent to the state of an expert gNB as

$$s_l = \begin{cases} s_0, & \overline{\Gamma}_l \geq \Gamma_{th}, \\ s_1, & \text{otherwise}, \end{cases} \tag{5.14}$$

  where $\overline{\Gamma}_l$ is the average SINR of the l^{th} learner gNB.

- **Actions:** The actions of the learner is extended to consider joint user-cell association and selection of number of beams (clusters). Indeed, the selection of the number of beams plays a key role in balancing inter-beam and intra-beam interference. On one hand, increasing the number of beams enhances the coverage of users, with less users per beam, which can improve the performance of SIC. On the other hand, more beams leads to higher inter-beam interference, which degrades the performance of decoding at the UE side. Therefore, a gNB should seek to find an optimal number of beams to cover its users. As such, joint user-cell association and selection of number of beams contribute to increased sum rate per gNB. The actions are formulated as $\boldsymbol{a}_l = [\boldsymbol{\delta}_{i,l}, B_l; i \in \mathcal{N}_l, l \in \mathfrak{L}]$, where $\boldsymbol{\delta}_{i,l}$ represents a vector of logical indicators of i^{th} UE's association to l^{th} learner gNB, and B_l is the number of beams selected by the l^{th} gNB.

- **Reward:** The learner's reward is equivalent to the expert's reward as in (5.13) (i.e., the ultimate goal of both the expert and the learner is to improve the average SINR).

- **Transfer function** The transfer function is used to map a Q-value of a source task to a corresponding Q-value of a target task as shown in Fig. 5.1. The transfer process is performed as follows. The learner gNB observes its target state-action pair (s_t, a_t) which is mapped to a source state-action pair (s_s, a_s) using the mapping functions ϕ_s and ϕ_a. With the source state-action pair, the learner gNB addresses the expert's Q-table, stored at the learner gNB, to extract a source Q-value $q_s(s_s, a_s)$. Afterwards,

the source Q-value is mapped to a target Q-value $q_t(s_t, a_t)$ via a mapping function ϕ_q. The final Q-value of the learner gNB is represented as follows:

$$q(s_t, a_t) = q_t(s_t, a_t) + q_l(s_t, a_t), \tag{5.15}$$

where $q_l(s_t, a_t)$ is the local Q-value of the learner computed through reinforcement learning as follows:

$$q_l(s_t, a_t) \leftarrow q_l(s_t, a_t) + \alpha[r_l(s_t, a_t) + \gamma \max_{a' \in \hat{\mathcal{A}}} q_l(s'_t, a') - q_l(s_t, a_t)], \tag{5.16}$$

where $r_l(s_t, a_t)$ is the instantaneous reward of the learner gNB, α is a learning rate, γ is a discount factor, and $q_l(s'_t, a')$ is the expected Q-values at next state s'_t for all actions $a' \in \hat{\mathcal{A}}$. From (5.11) and (5.14), the expert's (source's) and learner's (target's) states are equivalent ($s_s \equiv s_t$), hence the mapping function of the state is $\phi_s = 1$. In addition, we select the mapping function of the Q-value as $\phi_q = 1$.

On the other hand, the action's mapping function ϕ_a is used to map a target action to a source action. We design the action's mapping function based on inter-beam and intra-beam interference where actions of the learner gNB can be classified into three classes according to the interference level they incur: actions that cause intra-beam interference, actions that cause inter-beam interference, and actions that cause both intra- and inter-beam interference. Similarly, actions of the expert gNB can be classified into three classes: actions that cause intra-cell interference, actions that cause inter-cell interference, and actions that cause both. Table 5.1 presents the actions' mapping function along with an example on each interference case, where network is comprised of two expert gNBs covering two users and two learner gNBs equipped with up to three beam capability and covering three users. In the first example (i.e., 1^{st} row), learner (1) selects one beam to cover the three associated users, which means that learner (2) does not cover any users. As such, learner (1) incurs intra-beam interference only, which maps to a case in which an expert incurs intra-cell interference only (i.e., expert (1) covers all users). In row 2, learner (1) decides to use two beams, hence both inter- and intra-beam interference exist. This action should be mapped to an expert's action that incurs inter- and intra-cell interference. That is, expert (1) covers one out of two users. Without loss of generality, this transfer function can be extended to larger number of users and learner agents.

Table 5.1: Actions' mapping function ϕ_a. Besides an example of actions' mapping for two expert gNBs with two users and two learner gNBs with three users. Number of beams ranges from one to three.

Interference caused by actions of		**Examples**	
Learner-1	**Expert-1**	**Learner-1** $[\boldsymbol{\delta}_{i,1}, B_1]$	**Expert-1** $[\boldsymbol{\delta}_{i,1}]$
intra-beam	intra-cell	[1 1 1 1]	[1 1]
inter-beam	inter-cell	[1 1 1 3]	[1 0]
inter- and intra-beam	inter- and intra-cell	[1 1 1 2]	[1 0]

5.2.5 Reinforcement Learning

The formulation of Q-learning is similar to Q-learning of the expert gNB as discussed in section 5.2.4, however, the actions are modified to account for joint user-cell association and number of beams selection (i.e., $\boldsymbol{a}_l = [\boldsymbol{\delta}_{i,l}, B_l; i \in \mathcal{N}_l, l \in \mathfrak{L}]$).

5.2.6 Best SINR with DBSCAN (BSDC)

BSDC performs disjoint user-cell association and clustering, in which user-association is performed based on best SINR (i.e., users associate with the gNB with the best downlink SINR) and DBSCAN is used for clustering (i.e., number of beams selection). DBSCAN is a well known unsupervised learning technique and the details can be found in [114]. User clustering has been proposed before in [22] using k-means clustering algorithm. However, we select DBSCAN to perform clustering for two reasons. First, DBSCAN is proven to perform well under clustered-based users distribution. Second, while k-means requires the adjustment of the parameter k (i.e., number of clusters), DBSCAN is able to infer the number of clusters from the given users distribution.

5.2.7 Performance Evaluation: Simulation Settings

We use Matlab 5G toolbox to implement a discrete-event simulator, where physical and MAC layers specifications are considered. The simulation parameters of the network model, TQL, Q-learning, and BSDC are presented in Table 5.2. In particular, we consider a network with two expert gNBs and two learner gNBs. In case of TQL, expert gNBs perform conventional Q-learning for user-cell association, whereas learner gNBs perform TQL for

joint user-cell association and selection of number of beams. In addition, the knowledge (i.e., Q-table) at the expert gNB are transferred to the learner gNBs. In case of Q-learning and BSDC, we do not consider expert gNBs, hence learner gNBs become the only gNBs of the system model (i.e., Fig. 5.2b). All gNBs use subcarrier spacing of 15 KHz and TTI size of 2 OFDM symbols. Furthermore, link adaptation is performed in conjunction with HARQ technique, where 6 HARQ processes and a maximum of one HARQ re-transmission were used. All gNBs apply power domain NOMA, which implies that all users are allocated the entire 5G resource block grid. As such, user-cell association controls the load handled by each gNB. Finally, an entire simulation run consumes 6000 TTIs, whereas 40 runs are performed to maintain a statistically valid results with confidence interval of 95%.

5.2.8 Performance Results I: Complexity and Convergence Analysis

In this section, we provide the complexity and convergence analysis of the proposed algorithms. Complexity analysis considers both runtime and memory complexity. In particular, runtime complexity is presented in Big-O notation computed per gNB per TTI. In addition, memory complexity is presented in number of memory entries required to store information of each algorithm.

Algorithm 5.1 presents the steps of both Q-learning and TQL approaches. The runtime complexity of the proposed algorithms is presented in Table 5.3. In particular, complexity of Q-learning stems from two search-for-maximum operations. When using binary search, the complexity of Q-learning becomes $O(\log(2^N)) = O(N)$, where N represents number of users. Similarly, since state, action, and Q-value mapping functions are less complex, a search-for-maximum operation dominates the complexity of TQL. Therefore, complexity of TQL is equivalent to Q-learning (i.e., $O(N)$). As such, Q-learning-based algorithms always incur a runtime complexity in the order of a search-for-maximum operation. On the other hand, the complexity of BSDC is dominated by the DBSCAN algorithm, which is $O(\log(N^2)) = O(\log(N))$ [128] under binary search assumption. Therefore, BSDC outperforms Q-learning and TQL in runtime complexity.

On the other hand, space complexity is presented in Table 5.3. In particular, Q-learning with its tabular version requires a Q-table of size $nStates \times nActions$. Therefore, the space complexity of Q-learning becomes $O(2 \times (B \times 2^N)) = O(B \times 2^N)$, where B represents the possible values of number of beams. Similarly, TQL requires two Q-tables, one for its local Q-learning, whereas the other one is the transferred Q-table from the expert gNB. In particular, the space complexity of the local Q-table is $O(B \times 2^N)$, whereas the complexity

Table 5.2: 5G mm-wave simulation settings

Physical layer	
Bandwidth	20 MHz
Carrier frequency	30 GHz [109]
Subcarrier spacing	15 KHz
Subcarriers per resource block	12
TTI size	2 OFDM symbols
Max transmission power	28 dBm
HARQ	
Type	Asynchronous HARQ
Round trip delay	4 TTIs [126]
Number of processes	6
Max. number of re-transmission	1
Network model	
Distribution	Poisson Cluster Process
Number of users per gNB (Expert)	3
Number of clusters (Expert)	1
Number of users per cluster (Learner)	6
Number of clusters (Learner)	2
Radius of cluster	30 m
Total number of users	18
Number of expert gNBs	2
Number of learner gNBs	2
Inter-gNBs distance	150 m [109]
Traffic	
Distribution	Poisson
Packet size	32 Bytes
Q-learning and TQL	
Learning rate (α)	0.5
Discount factor (γ)	0.9
Exploration probability (ϵ)	0.05
SINR Threshold (Γ_{th})	20 dB [115, 127]
BSDC	
Minimum number of points (minpts)	1
ϵ_{BSDC}	40
Simulation parameters	
Simulation time	6000 TTI
Number of runs	40
Confidence interval	95%

Algorithm 5.1 Q-learning and TQL algorithms for user-cell association and selection of number of beams

1: **for** scheduling assignment period $t = 1$ to T **do**
2: **Step 1:** gNB receives feedback from users in the form of SINRs.
3: **Step 2:** Observe next state as in (5.11) for Q-learning or (5.14) for TQL.
4: **Step 3:** Update Q-value.
5: **Step 4:** Select action through ϵ-greedy approach

$$\boldsymbol{a} = \begin{cases} \text{random}, & (1-\epsilon), \\ \arg\max\limits_{a' \in \mathcal{A}} q(s, a'), & \epsilon. \end{cases} \tag{5.17}$$

6: **end for**

Table 5.3: Complexity comparison among the proposed algorithms

Complexity	**Q-learning**	**TQL**	**BSDC**
Runtime	$O(N)$	$O(N)$	$O(\log(N))$
Space	$O(B \times 2^N)$	$O(B \times 2^N)$	$O(N)$

of the transferred Q-table is $O(2^N)$ since expert gNB does not consider the selection of number of beams. As such, the total space complexity of TQL becomes $O(B \times 2^N)$. Finally, BSDC is dominated by the memory requirement of DBSCAN, which is $O(N)$ [128]. To sum, BSDC outperforms both Q-learning and TQL with respect to both runtime and space complexity. However, it is worth mentioning that space complexity of Q-learning can be reduced by employing deep Q-learning as proposed in [94], where deep Q-learning replaces the need for a Q-table by directly predicting Q-values using a deep neural network.

The convergence of the expert gNB is presented in Fig. 5.3. Note that in case of TQL, the expert gNB is performing user-cell association solely. In the figure, the average cumulative reward is plotted against iteration number (i.e., TTI number). The figure demonstrates the successful convergence of the expert agent within the lifetime of the simulation (i.e., 6000 TTIs). This is essential since results of expert gNB beyond convergence is transferred to the learner gNB.

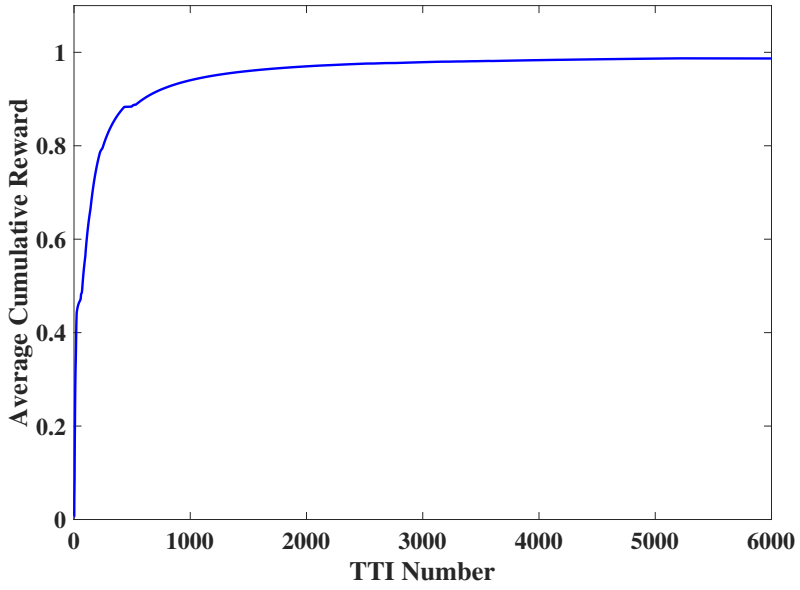

Figure 5.3: Convergence of expert gNBs represented by the average cumulative reward.

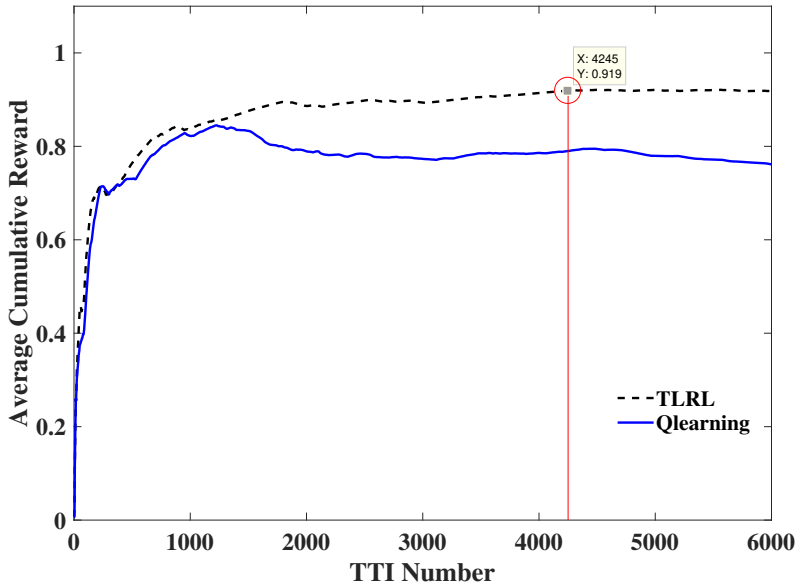

Figure 5.4: Convergence of learner gNBs represented by the average cumulative reward. Total offered load is 1.3 Mbps.

The convergence of learner gNBs for both Q-learning and TQL is plotted in Fig. 5.4. As observed from the figure, TQL outperforms Q-learning in two aspects. First, TQL converges rapidly (i.e., around 4245 TTIs), whereas Q-learning experiences more iterations with a sign of convergence toward the end of the simulation time. Although we select TTI length of 2 OFDM symbols, the convergence trend will not be impacted by the choice of other TTI configurations of 5G. However, the time for convergence will be longer and proportional to the length of TTI duration in ms. Second, TQL achieves higher cumulative reward, whereas Q-learning dwells around lower cumulative reward. This demonstrates TQL's ability to converge to a better policy for user-cell association and selection of number of beams. This constitutes an advantage for TQL compared to Q-learning. While both have the same complexity as shown in table 5.3, TQL converges faster than Q-learning. It is also wroth mentioning that the transferred Q-table from the expert can be learned in an offline setup. This means TQL can train experts offline and learners can be trained in the field with an online algorithm in a shorter time.

5.2.9 Performance Results II: Stationary Users

In this section, simulation results are provided for the proposed algorithms under stationary users scenario (i.e., no mobility). In particular, PCP is used for initial positions of users.

As presented in Fig. 5.4, TQL proved to converge to a better policy for user-cell association and selection of number of beams. This is evident from Fig. 5.5, where the sum rate of the learner gNBs is plotted against the total offered load in the network. In particular, TQL outperforms Q-learning under all traffic loads with about 23% improvement at the highest traffic load. BSDC, on the other hand, performs very closely to TQL. This was expected since BSDC uses DBSCAN clustering which performs very well under the PCP distribution model. In the next section, we perform comparison under different user distribution and mobility model to highlight the superiority of TQL compared to DBSCAN.

Besides achieving high rate, the rate of TQL is close to the total offered rate in the network, which implies high reliability as well. This is highlighted in Fig. 5.6 and in Fig. 5.7 which plots the packet loss against the total offered traffic load. The figure demonstrates that TQL outperforms the Q-learning algorithm under all traffic conditions.

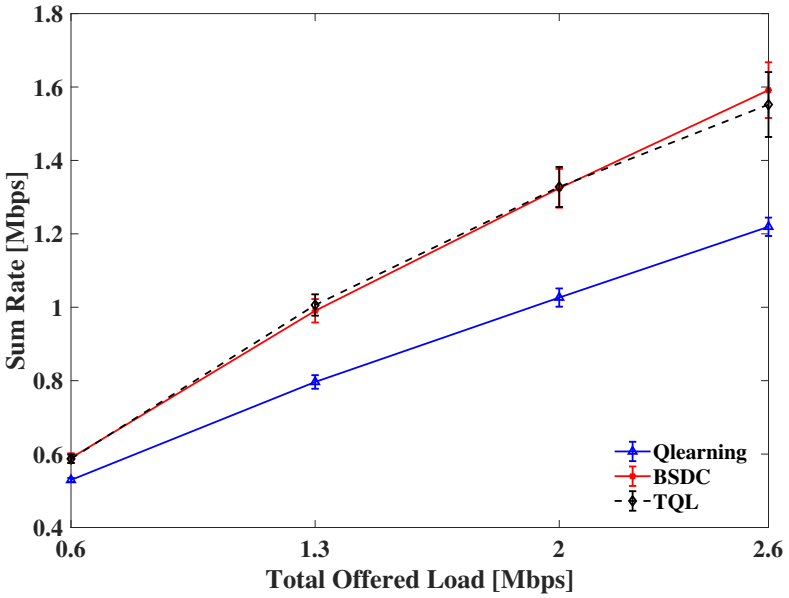

Figure 5.5: Sum rate in [Mbps] of learner gNBs against total offered network load in [Mbps] under PCP deployment of users.

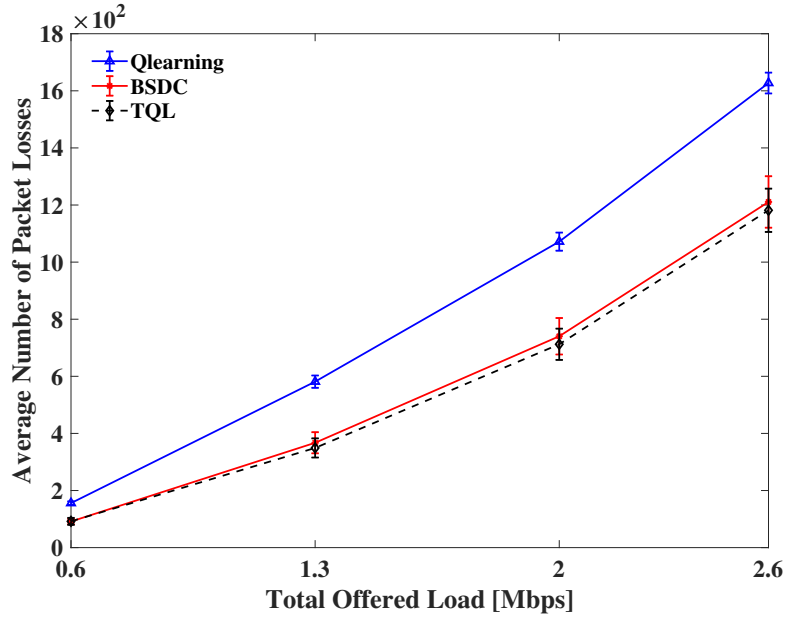

Figure 5.6: Average number of packet loss in [packets] against total offered network load in [Mbps] under PCP deployment of users.

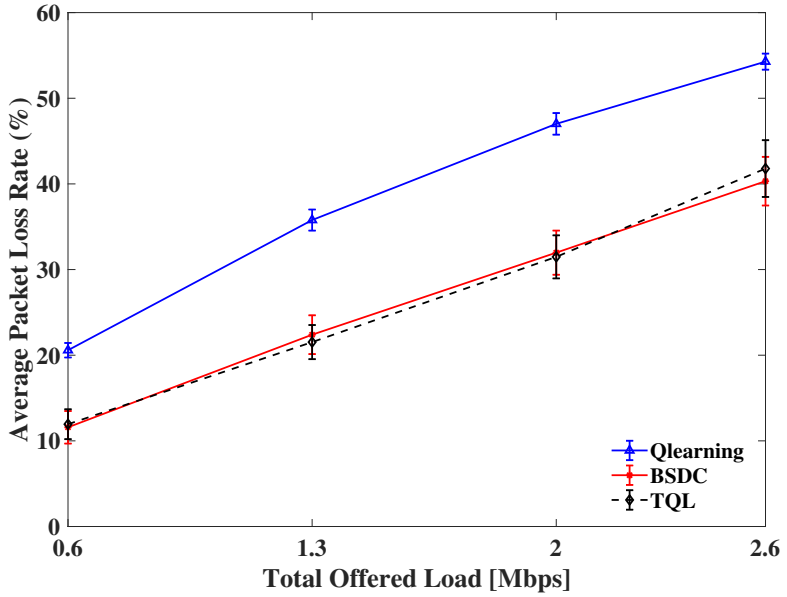

Figure 5.7: Packet loss rate in [%] against total offered network load in [Mbps] under PCP deployment of users.

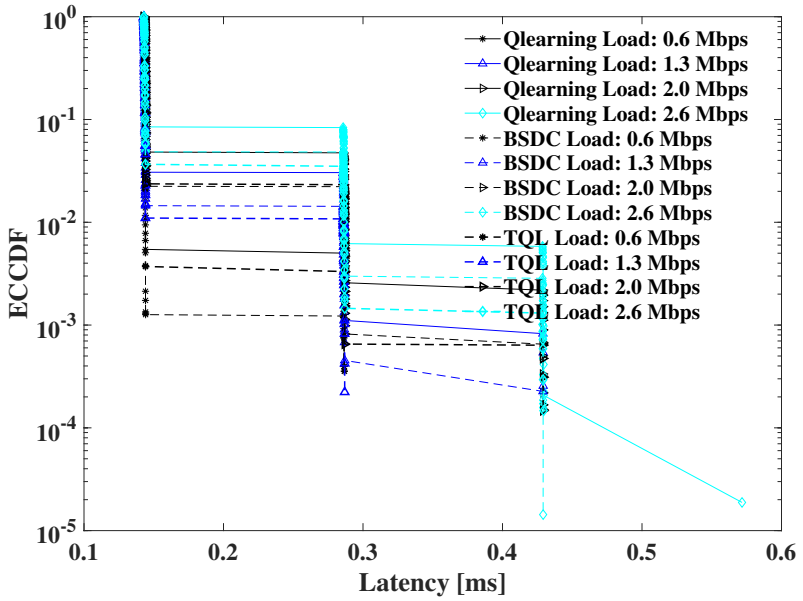

Figure 5.8: eCCDF of latency for different total offered network load under PCP deployment of users.

Finally, Fig. 5.8 presents the eCCDF of latency, where latency is defined as the end-to-end delay of successfully received packets from gNB to users. The figure shows close performance of the three algorithms. This was expected since all algorithms do not account for latency improvement (Refer to the reward function in (5.13)). Furthermore, the low latency achieved (i.e., latency below 1 msec) is due to the restriction put on the number of HARQ re-transmissions, where one re-transmission is used in our simulation [99, 101].

5.2.10 Performance Results III: Random Waypoint Mobility

The mobility of users might have a significant impact on the performance of the proposed algorithms. Mobile users tend to change their clustering behavior which might lead to different number of clusters with iterations. This mandates rapid response in terms of number of beams selection. Furthermore, due to mobility, users might change the clusters they belong to, which also impacts user-cell association. As such, enhancing performance under mobility becomes a necessary component of the learning algorithm. In this sub-section, we assess the performance of the proposed algorithms under a random waypoint mobility scenario. In particular, initial users' deployment follows PCP distribution whereas mobility of users follows random waypoint model.

Fig. 5.9, Fig. 5.10, and Fig. 5.11 present the sum rate of learner gNBs, number of packet loss, and packet loss rate in percentage versus total offered load under random waypoint mobility model, respectively. Unlike stationary case, TQL and Q-learning outperform BSDC performance in both sum rate and packet loss. In particular, TQL and Q-learning demonstrate 12% sum rate improvement over BSDC at the highest offered traffic load.

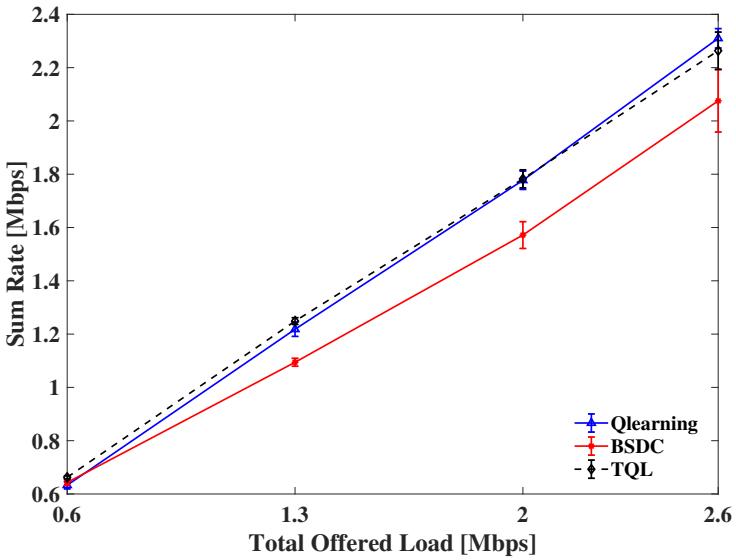

Figure 5.9: Sum rate in [Mbps] of learner gNBs against total offered network load in [Mbps] under random waypoint mobility of users.

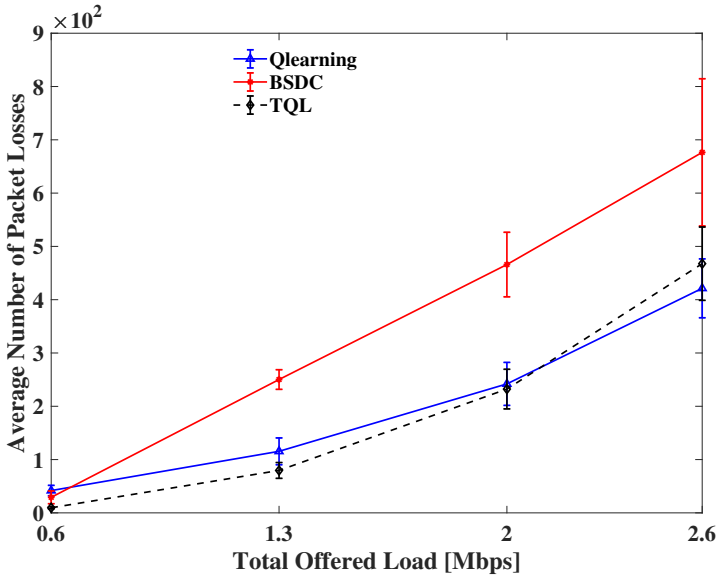

Figure 5.10: Average number of packet loss in [packets] against total offered network load in [Mbps] under random waypoint mobility of users.

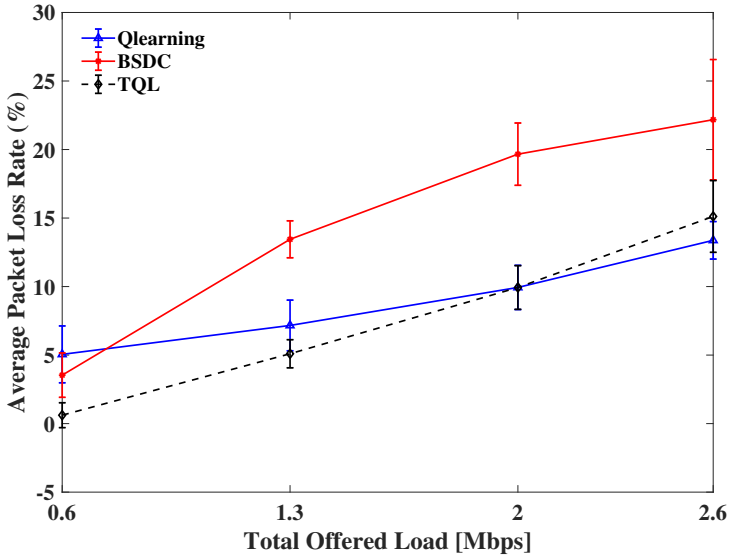

Figure 5.11: Packet loss rate in [%] against total offered network load in [Mbps] under random waypoint mobility of users.

5.3 Conclusion

In this chapter, we employed transfer in reinforcement learning in 5G mm-wave networks to the address the problem of joint user-cell association and selection of number of beams. Transfer in reinforcement learning is compared to traditional Q-learning and a heuristic based on DBSCAN under stationary and mobile network conditions. Results showed the suitability of each algorithm to specific deployment scenario. Under mobility scenario, TQL and Q-learning demonstrate 12% sum rate improvement over BSDC at the highest offered traffic load, whereas under stationary scenario, Q-learning and BSDC outperforms TQL with about $10-23\%$ at lowest and highest offered traffic loads, respectively. In addition, BSDC has lower complexity than the other techniques and TQL has faster convergence than the Q-learning based technique. Besides, TQL offers a unique advantage for offline learning of a task and transferring the knowledge to another task with online learning in the field.

Chapter 6

Conclusion

RRM constitutes a key problem in past, current, and future networks. With the advent of new technologies, network architecture, traffic demands, and network dynamicity, RRM is becoming more and more challenging. As such, a paradigm shift in the way radio resources are assigned in the network is needed. Machine learning constitutes a promising tool in providing significant improvement and agility in the network. In this thesis, we presented several uses of machine learning in improving the performance of both LTE and 5G networks. In particular, reinforcement learning, deep reinforcement learning, and transfer reinforcement learning have been adopted to address RRM in LTE and 5G networks.

We have used model-free reinforcement learning to perform several RRM techniques such spectrum allocation, power allocation, user-cell association, and beamforming in LTE and 5G networks. In particular, in LTE, we proposed TMQ, DMQ, DMDQ, and DMDQN to improve throughput, latency and convergence speed. The following points highlight the main contributions with those methods.

- Tactile communication requires high throughput communication. Hence, we proposed TMQ based on Q-learning that has a reward function aiming to balance the throughput of data intensive users and UEs. The results show that TMQ is able to achieve 130% throughput improvement, 80% reduction in delay, and 6% increase in fairness.
- Despite the large delay reduction that TMQ achieved, it failed to achieve the very tight latency requirements of tactile communication, where a 200ms delay was achieved. Hence, we proposed DMQ based on Q-learning with the aim to minimize network

delay. The algorithm was tested on a microgrid scenario, where it achieved 63% delay reduction, and 100% throughput improvement.

- To address the convergence problem of tabular Q-learning, DMDQ was proposed. DMDQ uses an LSTM deep neural network that acts as a function approximator to Q-values. Results show 60% convergence speedup compared to tabular Q-learning. Furthermore, a 30% delay reduction was achieved with 9% throughput degradation.

- Despite the importance of RB allocation, user-cell association constitutes an important decision in improving user's signal quality and in mitigating interference by associating the user to the proper base station. As such, we proposed DMDQN for joint RB allocation and user-cell association with the aim to improve delay of mission-critical services. This results in 41% delay reduction, 16% throughput improvement, and 40% convergence speedup for critical users.

In summary, the proposed algorithms for LTE networks achieved significant improvement in throughput, delay, fairness, and convergence speed. In particular, when throughput is considered as the main objective (i.e., by considering it in the reward function), the throughput improvement ranges between $100\% - 130\%$, whereas it becomes 16% when considering the delay as a primary objective. Furthermore, $30\% - 60\%$ delay reduction is achieved when considering the delay as a primary objective in the reward function. Finally, deep reinforcement learning was able to achieve $40\% - 60\%$ speedup in convergence compared to tabular Q-learning. In particular, the less speedup occurs when the algorithm involves joint optimization of multiple tasks, i.e., joint RB allocation and user-cell association.

In 5G, we proposed LLHRQ, LRT-Q, DQLD, and transfer in reinforcement learning to address the coexistence of uRLLC and eMBB users under different scenarios and tasks in a mm-wave network. The following points highlight the main contributions of those works.

- To address the coexistence problem of uRLLC and eMBB users, we first proposed LLHRQ based on Q-learning for RB and power allocation. LLHRQ has a state and reward functions crafted to achieve low latency and high reliability for uRLLC users. As such, it achieves 0.5ms latency and 4% packet drop rate improvement for uRLLC users.

- Despite the improved performance of LLHRQ with respect to uRLLC, it showed throughput degradation in eMBB's throughput. As such, we proposed LRT-Q, where

the Q-learning's state and reward were designed to balance uRLLC and eMBB performance requirements. In addition to improving uRLLC's latency, LRT-Q achieved a 29% throughput improvement for eMBB users compared to LLHRQ.

- The problem of mitigating intra- and inter-beam interference in mm-wave networks was studied. In particular, Q-learning was used to perform joint user-cell association and inter-beam power allocation with the aim to maximize network's sum throughput. Results show 20% throughput improvement.

- The temporal (i.e., traffic) and spatial (i.e., mobility) dynamicity of the previous network pose a challenge to the performance and convergence of Q-learning. Hence, we proposed DQLD which is based on deep reinforcement learning for joint RB allocation and number of beams selection with the aim to balance uRLLC and eMBB requirements. DQLD achieves about 350ms delay improvement at high traffic loads, in addition to 50% improvement in packet drops.

- Formulating a single algorithm that is able to perform well in a wide range of tasks is considered an ambitious goal. In wireless networks, an ultimate goal is to reap the benefits of ML for the sake of a fully autonomous wireless network. As such, we proposed a transfer in Q-learning algorithm (i.e., TQL), where knowledge can be transferred from an expert agent to a learner agents. In addition, we compared it to two other ML algorithms under stationary and mobile scenarios. Under mobility scenario, TQL and Q-learning demonstrate 12% sum rate improvement, whereas under stationary scenario, Q-learning outperforms TQL with about $10 - 23\%$ at lowest and highest offered traffic loads, respectively. Furthermore, TQL achieved a 29% convergence speedup compared to Q-learning.

6.1 Challenges and Open Issues

Future generations of wireless networks will have a higher level of complexity than all the preceding generations. This will bring in the need for intelligent mechanisms to orchestrate the available resources, services and users. Thus, ML-enabled methods may allow future networks to learn from their environment, adapt the changes in an automated fashion and achieve optimal performance.

ML algorithms have paved the way to significant agility in network management, yet several challenges are still open for research efforts. The open issues can be generally

classified into two main pillars: performance of ML methods and performance of wireless networks.

The relatively long convergence time of ML methods undermine their usefulness in highly dynamic wireless networks. A careful investigation of the convergence problem and the factors that influence the convergence, are needed. Novel ML techniques with faster convergence and online learning capabilities can benefit wireless networks better.

Besides convergence, the uncertainty in the wireless network calls for an on-going update of the parameters of the ML method or even the method itself. The stochastic nature of the wireless channel may require continuous adaptation. For instance, a network encompassing a large and diverse set of users will have a very dynamic operation. In particular, users who join or leave the network may have very different QoS and quality of experience requirements. Thus, there is a need to examine whether a “one size fits all" approach is feasible in real-world implementations.

In addition, scalability of ML algorithms needs to be addressed. ML algorithms can become unfeasible for moderately large data, especially in collaborative learning approaches. This calls for a scalable learning algorithm to accommodate the dense use cases of future wireless networks.

Furthermore, supervised and unsupervised learning techniques have been used for massive MIMO recently [129]. Further research is needed to investigate whether it is possible to enhance the performance of massive MIMO using reinforcement learning and deep learning.

Last but not least, ML-enabled networks also impact e-health applications. For instance, advancing outside-of-clinic operations using wearable sensors [130] requires harmonization of network resource allocation across several technologies and ML algorithms can be used for helping with harmonization. Hence, application specific use of ML needs to be further explored.

To summarize, a true gap exists at the heart of the right choice of an ML algorithm for the specific use in improving wireless system's performance. In other words, there is the question of *why* and *how* an ML method would work for the wireless domain, especially when this domain is having intact relations with a physical system that drives its performance metrics. As such, it is important to characterize the inter-dependencies between ML and wireless networks. Furthermore, optimization methods exist on the other end of the spectrum, where they have been used for decades to optimize network metrics. Therefore, a comprehensive analysis of ML methods in comparison to optimization methods is mandatory to reap the benefits of each approach.

Appendix A

Design of Reinforcement Learning

Model-free reinforcement learning involves a learner that aims to improve its performance under a certain task by mapping actions to the different situations in order to maximize a certain reward signal. This process involves a trial-and-error cycle since the learner is not told what to do. Furthermore, the actions employed by the learner might impact future situations in the form of delayed reward. Hence, the state and reward are central to the definition of reinforcement learning. They characterize the search space and the ultimate objective that the learner aims to achieve. In this appendix, we provide a number of guidelines for the design of the state and the reward functions of reinforcement learning.

A.1 Elements of Reinforcement Learning

There are four elements that constitute a reinforcement learning setup: a policy, a reward signal, a value function, and optionally a model [10].

- **Policy:** The policy represents the mapping between the states and actions, i.e., it represents the behavior of the learner under different situations. Indeed, the learner would be able to refine its behavior if it has a proper state-space. That is, a space that is well-defined to capture the essential knowledge in the environment without unnecessary situations that would increase that state-space which leads to increased convergence time.
- **Reward Function:** The reward is a signal sent from the environment to the agent. It represents the ultimate goal of the learner over the long run, i.e., the learner aims

to maximize the total cumulative reward over its time horizon. Such reward defines the good and bad behaviors of the learner under the different situations which in turn shapes the policy of the learner. The design of the reward signal, therefore, is an intricate process that relies on the application pursued. The next section provides a deep dive into the design of the reward.

- **Value Function:** While rewards indicate the immediate good and bad behaviors of the learner, the value function quantifies the good policy associated with a state on the long run. In other words, it computes the total reward of that the learner would acquire when following a certain policy at each state. As such, the value function is computed per state.

- **Model:** The model is an abstraction of the behavior of the environment. The model is usually represented by the state transition. In particular, a probability distribution is to sought to model the next state based on the current state and action pair. With that, reinforcement learning can be classified to model-based and model-free reinforcement learning. In model-based learning, a trajectory of actions can be decided by considering possible future situations, i.e., states, before they happen. On the other hand, model-free learning uses trial and error to infer a policy of action selection without inferring the model of the environment. It is worth mentioning that the rest of this appendix discusses model-free reinforcement learning only.

A.2 Reward Function Design Guidelines

In this section, we provide three main guidelines on how to design the reward function of a reinforcement learning agent. It is worth mentioning that these guidelines apply to the area of RRM in wireless networks as presented in this thesis. Furthermore, they apply to a manual design of the reward function, whereas a more advanced techniques such as reward shaping and inverse reinforcement learning are used for automatic tuning of the reward or inferring the reward function, respectively.

A.2.1 Reward Key Factors

The design of the reward function involves the identification of the key factors that will guide the learner to achieve a certain goal. In wireless networks, this goal is typically driven by the network KPIs, i.e., throughput, latency, reliability, etc. Therefore, the key factors

impacting the reward function can be deduced from the objective function that represents the wireless system. In other words, the reward resembles the objective of the optimization problem of the system. However, a normalization operation should be performed to scale the reward in a confined range to avoid instability of the learner. It is worth mentioning that many factors can be used in the reward function such as CQI, SINR, TBS, etc. In this thesis, we targeted last-mile KPIs that are representing the ultimate objective of the wireless system. Indeed, the right selection and mix of the factors is an open issue.

A.2.2 Reward and Punishment

In reinforcement learning, the learner can obtain either a reward or punishment/cost through the environment. The reward is usually represented by a positive number such as $(+1)$ and is given to the learner when it achieves its goal. On the other hand, a punishment/cost is usually represented by a negative number such as (-1). Punishment in the form of a negative reward can be an efficient technique to avoid undesired situations and behaviors. For example, the reward in (4.10) gives the leaner a positive reward when the SINR exceeds a threshold value and (-1) when otherwise. The negative value, in this case, ensures that the learner does not accept a situation in which the SINR drops under the threshold value, which ensures a minimum level of reliability.

A.2.3 Discrete versus Continuous Reward

The value of the reward function is a defining factor of the behavior of the learner. A discrete (or step-wise) reward is more appropriate for hard-objectives, i.e., goals that have well-defined boundaries. For example, a vacuum robot that is designed to collect garbage bottles can be rewarded with a $(+1)$ value if it collects a bottle and punished with (-1) if it makes an action that reaches a place with no bottles.

On the other hand, a continuous reward is more appropriate for soft-objectives such as the KPIs of the wireless network. The reward in (5.13) represents a continuous reward (i.e., Sigmoid function) which is driven by the average SINR of the learner agent. It can be generally formulated as

$$r = \frac{1}{1 + e^{(a(\overline{\Gamma} - b\Gamma_{th}))}}, \tag{A.1}$$

where a and b are the parameters of the function. Figures A.1 and A.2 plot the reward function for different values of a and b. It can be observed from the figures that the parameter a controls the steepness of the slope whereas the parameter b controls the shift

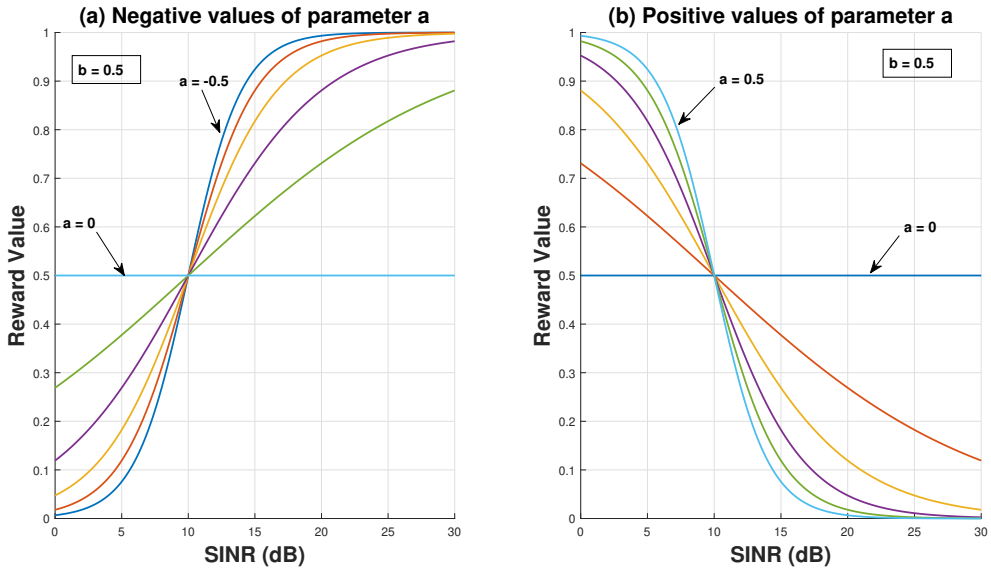

Figure A.1: Plot of the reward function defined in (5.13) for (a) $a = [-0.5 : 0.1 : 0]$ and (b) $a = [0 : 0.1 : 0.5]$. $\Gamma_{th} = 20$ dB.

on the x-axis. We select the parameter $b = 0.5$ as it rewards the learner with $r = 0.5$ when $\overline{\Gamma} = 0.5\Gamma_{th} = 10$ dB. This provides a fine granularity in the reward from $\overline{\Gamma} = 10$ to $\overline{\Gamma} = 20$, which is appropriate for users with different deployments in the environment. This is backed further with the choice of the parameter a, where a negative value is selected as it provides an increasing reward as SINR increases. It can be observed from Fig. A.1a that decreasing the value of the parameter a obtains a more steeper curve. As such, a value of $a = -0.5$ is selected to provide faster increase in the reward function as the SINR increases. This is important since it leads to better discrimination among the values of the SINR. In particular, as the SINR increases, the learner is rewarded with higher reward value, hence it improves its behavior towards an action that improves the SINR.

A.3 State Function Design Guidelines

The state function can be visualized as a snapshot of the environment in its current situation. The current state of the environment is a direct outcome of the action(s) taken by the different learner(s). For example, in a chess play, the environment state is characterised by the positions of the pieces on the chess board. The actions of the players have a direct

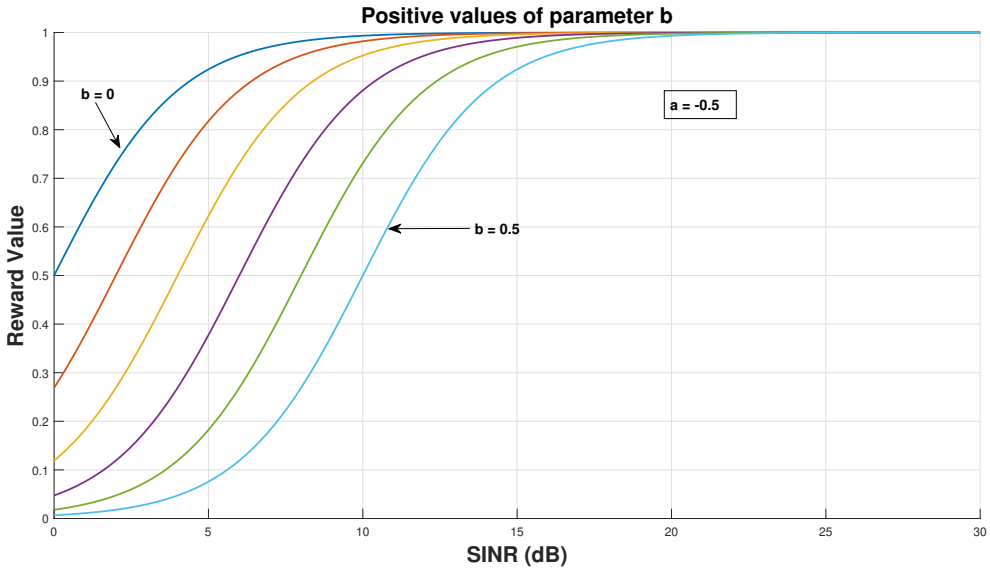

Figure A.2: Plot of the reward function defined in (5.13) for $b = [0 : 0.1 : 0.5]$. $\Gamma_{th} = 20$ dB.

impact on the next state of the board. Furthermore, the sequence of actions that starts from a certain state leads to a certain outcome of the game (i.e., checkmate for either player).

While the reward function captures the objective of the learner, the state captures the behavior that we would like to see in the environment. In wireless networks, a typical state is the one that captures the level of interference in the environment. In this case, the target state is the one that minimizes the interference level in the environment. Eq. (4.19) represents a state that captures the level of interference using the average SINR. Indeed, the guidelines presented for the reward design apply in a very similar way to the state design.

Bibliography

[1] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. The MIT Press, 1st ed., November 2016.

[2] ITU, "5G Overview," in *Setting the Scene for 5G: Opportunities and Challenges*, Geneva: ITU, 2018.

[3] S. Ali et al., "6G White Paper on Machine Learning in Wireless Communication Networks," *arXiv e-prints*, p. arXiv:2004.13875, April 2020.

[4] S. Manap, K. Dimyati, M. N. Hindia, M. S. Abu Talip, and R. Tafazolli, "Survey of Radio Resource Management in 5G Heterogeneous Networks," *IEEE Access*, vol. 8, pp. 131202–131223, June 2020.

[5] A. Ghosh, J. Zhang, J. G. Andrews, and R. Muhamed, *Fundamentals of LTE*. Upper Saddle River, NJ, USA: Prentice Hall Press, 1st ed., September 2010.

[6] D. Liu, L. Wang, Y. Chen, M. Elkashlan, K. Wong, R. Schober, and L. Hanzo, "User Association in 5G Networks: A Survey and an Outlook," *IEEE Communications Surveys Tutorials*, vol. 18, pp. 1018–1044, January 2016.

[7] A. M. Turing, "I.—Computing Machinery and Intelligence," *Mind*, vol. LIX, pp. 433–460, October 1950.

[8] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*. Pearson, 4th ed., January 2015.

[9] E. Alpaydin, *Introduction to Machine Learning*. The MIT Press, August 2014.

[10] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. The MIT Press, 2nd ed., February 2018.

[11] V. Mnih et al., "Human-Level Control Through Deep Reinforcement Learning," in *Nature Publishing Group*, pp. 529–533, February 2015.

[12] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. A. Riedmiller, "Playing Atari with Deep Reinforcement Learning," *CoRR*, vol. abs/1312.5602, pp. 1–9, January 2013.

[13] S. J. Pan and Q. Yang, "A Survey on Transfer Learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, pp. 1345–1359, October 2010.

[14] M. E. Taylor and P. Stone, "Cross-Domain Transfer for Reinforcement Learning," *International Conference on Machine Learning*, p. 879–886, June 2007.

[15] G. Joshi and G. Chowdhary, "Cross-Domain Transfer in Reinforcement Learning Using Target Apprentice," *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7525–7532, September 2018.

[16] C. Kalalas, L. Thrybom, and J. Alonso-Zarate, "Cellular Communications for Smart Grid Neighborhood Area Networks: A Survey," *IEEE Access*, vol. 4, pp. 1469–1493, April 2016.

[17] L. U. Khan, W. Saad, Z. Han, E. Hossain, and C. S. Hong, "Federated Learning for Internet of Things: Recent Advances, Taxonomy, and Open Challenges," *submitted to IEEE Communications Surveys and Tutorials*, September 2020.

[18] M. Mozaffari, W. Saad, M. Bennis, Y. Nam, and M. Debbah, "A Tutorial on UAVs for Wireless Networks: Applications, Challenges, and Open Problems," *IEEE Communications Surveys Tutorials*, vol. 21, pp. 2334–2360, March 2019.

[19] L. Liang, H. Ye, G. Yu, and G. Y. Li, "Deep-Learning-Based Wireless Resource Allocation With Application to Vehicular Networks," *Proceedings of the IEEE*, vol. 108, pp. 341–356, December 2020.

[20] K. I. Ahmed, H. Tabassum, and E. Hossain, "Deep Learning for Radio Resource Allocation in Multi-Cell Networks," *IEEE Network*, vol. 33, pp. 188–195, April 2019.

[21] D. Marasinghe, N. Jayaweera, N. Rajatheva, and M. Latva-Aho, "Hierarchical User Clustering for mmWave-NOMA Systems," in *6G Wireless Summit (6G SUMMIT)*, pp. 1–5, March 2020.

[22] J. Cui, Z. Ding, P. Fan, and N. Al-Dhahir, "Unsupervised Machine Learning-Based User Clustering in Millimeter-Wave-NOMA Systems," *IEEE Transactions on Wireless Communications*, vol. 17, pp. 7425–7440, November 2018.

[23] R. Liu, G. Yu, and G. Y. Li, "User Association for Ultra-Dense mmWave Networks With Multi-Connectivity: A Multi-Label Classification Approach," *IEEE Wireless Communications Letters*, vol. 8, pp. 1579–1582, July 2019.

[24] B. Soleimani and M. Sabbaghian, "Cluster-Based Resource Allocation and User Association in mmWave Femtocell Networks," *IEEE Transactions on Communications*, vol. 68, pp. 1746–1759, November 2020.

[25] R. Liu, M. Lee, G. Yu, and G. Y. Li, "User Association for Millimeter-Wave Networks: A Machine Learning Approach," *IEEE Transactions on Communications*, vol. 68, pp. 4162–4174, March 2020.

[26] M. K. Abdel-Aziz, S. Samarakoon, M. Bennis, and W. Saad, "Ultra-Reliable and Low-Latency Vehicular Communication: An Active Learning Approach," *IEEE Communications Letters*, vol. 24, pp. 367–370, December 2020.

[27] Q. Hu, Y. Cai, A. Liu, G. Yu, and G. Y. Li, "Low-Complexity Joint Resource Allocation and Trajectory Design for UAV-Aided Relay Networks With the Segmented Ray-Tracing Channel Model," *IEEE Transactions on Wireless Communications*, vol. 19, pp. 6179–6195, June 2020.

[28] H. Saad, A. Mohamed, and T. ElBatt, "A Cooperative Q-Learning Approach for Online Power Allocation in Femtocell Networks," in *IEEE Vehicular Technology Conference (VTC Fall)*, pp. 1–6, September 2013.

[29] M. H. M. Elsayed and A. Mohamed, "Distributed Interference Management using Q-Learning in Cognitive Femtocell Networks: New USRP-based Implementation," in *International Conference on New Technologies, Mobility and Security (NTMS)*, pp. 1–5, July 2015.

[30] Y. Luo, Z. Shi, X. Zhou, Q. Liu, and Q. Yi, "Dynamic Resource Allocations based on Q-learning for D2D Communication in Cellular Networks," in *International Computer Conference on Wavelet Actiev Media Technology and Information Processing(ICCWAMTIP)*, pp. 385–388, December 2014.

[31] X. Chen, C. Wu, Y. Zhou, and H. Zhang, "A Learning Approach for Traffic Offloading in Stochastic Heterogeneous Cellular Networks," in *IEEE International Conference on Communications (ICC)*, pp. 3347–3351, June 2015.

[32] R. Ghanavi, E. Kalantari, M. Sabbaghian, H. Yanikomeroglu, and A. Yongacoglu, "Efficient 3D Aerial Base Station Placement Considering Users Mobility by Reinforcement Learning," in *IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–6, April 2018.

[33] M. M. U. Chowdhury, W. Saad, and I. Güvenç, "Mobility Management for Cellular-Connected UAVs: A Learning-Based Approach," in *IEEE International Conference on Communications Workshops (ICC Workshops)*, pp. 1–6, July 2020.

[34] M. A. Qureshi and C. Tekin, "Fast Learning for Dynamic Resource Allocation in AI-Enabled Radio Networks," *IEEE Transactions on Cognitive Communications and Networking*, vol. 6, pp. 95–110, November 2020.

[35] Y. Zhou, F. Tang, Y. Kawamoto, and N. Kato, "Reinforcement Learning-Based Radio Resource Control in 5G Vehicular Network," *IEEE Wireless Communications Letters*, vol. 9, pp. 611–614, December 2020.

[36] Y. Wei, F. R. Yu, M. Song, and Z. Han, "User Scheduling and Resource Allocation in HetNets With Hybrid Energy Supply: An Actor-Critic Reinforcement Learning Approach," *IEEE Transactions on Wireless Communications*, vol. 17, pp. 680–692, November 2018.

[37] B. Matthiesen, A. Zappone, E. A. Jorswieck, and M. Debbah, "Deep Learning for Real-Time Energy-Efficient Power Control in Mobile Networks," in *IEEE International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pp. 1–5, July 2019.

[38] W. Ma, C. Qi, and G. Y. Li, "Machine Learning for Beam Alignment in Millimeter Wave Massive MIMO," *IEEE Wireless Communications Letters*, vol. 9, pp. 875–878, February 2020.

[39] M. Chen, W. Saad, and C. Yin, "Echo State Learning for Wireless Virtual Reality Resource Allocation in UAV-Enabled LTE-U Networks," in *IEEE International Conference on Communications (ICC)*, pp. 1–6, July 2018.

[40] Y. Wang, M. Chen, Z. Yang, T. Luo, and W. Saad, "Deep Learning for Optimal Deployment of UAVs with Visible Light Communications," *Submitted to IEEE Transactions on Wireless Communications*, pp. 1–1, July 2020.

[41] M. Chen, O. Semiari, W. Saad, X. Liu, and C. Yin, "Federated Deep Learning for Immersive Virtual Reality over Wireless Networks," in *IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, December 2019.

[42] Y. He, Z. Zhang, and Y. Zhang, "A Big Data Deep Reinforcement Learning Approach to Next Generation Green Wireless Networks," in *IEEE Global Communications Conference*, pp. 1–6, December 2017.

[43] J. Zhu, Y. Song, D. Jiang, and H. Song, "A New Deep-Q-Learning-Based Transmission Scheduling Mechanism for the Cognitive Internet of Things," *IEEE Internet of Things Journal*, pp. 1–1, August 2018.

[44] F. B. Mismar, B. L. Evans, and A. Alkhateeb, "Deep Reinforcement Learning for 5G Networks: Joint Beamforming, Power Control, and Interference Coordination," *IEEE Transactions on Communications*, vol. 68, pp. 1581–1592, June 2020.

[45] M. K. Sharma, A. Zappone, M. Debbah, and M. Assaad, "Multi-Agent Deep Reinforcement Learning based Power Control for Large Energy Harvesting Networks," in *International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOPT)*, pp. 1–7, July 2019.

[46] L. Liang, H. Ye, and G. Y. Li, "Multi-Agent Reinforcement Learning for Spectrum Sharing in Vehicular Networks," in *IEEE International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pp. 1–5, July 2019.

[47] X. Liao, J. Shi, Z. Li, L. Zhang, and B. Xia, "A Model-Driven Deep Reinforcement Learning Heuristic Algorithm for Resource Allocation in Ultra-Dense Cellular Networks," *IEEE Transactions on Vehicular Technology*, vol. 69, pp. 983–997, November 2020.

[48] Y. Zhang, C. Kang, T. Ma, Y. Teng, and D. Guo, "Power Allocation in Multi-Cell Networks Using Deep Reinforcement Learning," in *IEEE Vehicular Technology Conference (VTC-Fall)*, pp. 1–6, April 2018.

[49] F. Al-Tam, N. Correia, and J. Rodriguez, "Learn to Schedule (LEASCH): A Deep Reinforcement Learning Approach for Radio Resource Scheduling in the 5G MAC Layer," *IEEE Access*, vol. 8, pp. 108088–108101, June 2020.

[50] C. He, Y. Hu, Y. Chen, and B. Zeng, "Joint Power Allocation and Channel Assignment for NOMA With Deep Reinforcement Learning," *IEEE Journal on Selected Areas in Communications*, vol. 37, pp. 2200–2210, August 2019.

[51] S. Bhardwaj, R. R. Ginanjar, and D. Kim, "Deep Q-learning based Resource Allocation in Industrial Wireless Networks for URLLC," *IET Communications*, vol. 14, pp. 1022–1027, March 2020.

[52] A. Asheralieva, "Bayesian Reinforcement Learning-Based Coalition Formation for Distributed Resource Sharing by Device-to-Device Users in Heterogeneous Cellular Networks," *IEEE Transactions on Wireless Communications*, vol. 16, pp. 5016–5032, May 2017.

[53] H. Vaezy, M. Salehi Heydar Abad, O. Ercetin, H. Yanikomeroglu, M. J. Omidi, and M. M. Naghsh, "Beamforming for Maximal Coverage in mmWave Drones: A Reinforcement Learning Approach," *IEEE Communications Letters*, vol. 24, pp. 1033–1037, February 2020.

[54] K. Hamidouche, A. T. Z. Kasgari, W. Saad, M. Bennis, and M. Debbah, "Collaborative Artificial Intelligence (AI) for User-Cell Association in Ultra-Dense Cellular Systems," in *IEEE International Conference on Communications Workshops (ICC Workshops)*, pp. 1–6, May 2018.

[55] F. D. Calabrese, L. Wang, E. Ghadimi, G. Peters, L. Hanzo, and P. Soldati, "Learning Radio Resource Management in RANs: Framework, Opportunities, and Challenges," *IEEE Communications Magazine*, vol. 56, pp. 138–145, September 2018.

[56] Z. Yang, M. Chen, W. Saad, C. S. Hong, M. Shikh-Bahaei, H. V. Poor, and S. Cui, "Delay Minimization for Federated Learning Over Wireless Communication Networks," in *International Conference on Machine Learning*, pp. 1–7, April 2020.

[57] M. Chen, H. V. Poor, W. Saad, and S. Cui, "Convergence Time Minimization of Federated Learning over Wireless Networks," in *IEEE International Conference on Communications (ICC)*, pp. 1–6, July 2020.

[58] X. Zhang, M. Peng, S. Yan, and Y. Sun, "Deep-Reinforcement-Learning-Based Mode Selection and Resource Allocation for Cellular V2X Communications," *IEEE Internet of Things Journal*, vol. 7, pp. 6380–6391, December 2020.

[59] M. Lee, G. Yu, and G. Y. Li, "Learning to Branch: Accelerating Resource Allocation in Wireless Networks," *IEEE Transactions on Vehicular Technology*, vol. 69, pp. 958–970, November 2020.

[60] Y. Shen, Y. Shi, J. Zhang, and K. B. Letaief, "LORM: Learning to Optimize for Resource Management in Wireless Networks With Few Training Samples," *IEEE Transactions on Wireless Communications*, vol. 19, pp. 665–679, October 2020.

[61] Y. Hu, M. Chen, W. Saad, H. V. Poor, and S. Cui, "Meta-reinforcement learning for trajectory design in wireless uav networks," in *IEEE Global Communications Conference*, pp. 1–6, December 2020.

[62] I. Comşa, S. Zhang, M. E. Aydin, P. Kuonen, Y. Lu, R. Trestian, and G. Ghinea, "Towards 5G: A Reinforcement Learning-Based Scheduling Solution for Data Traffic Management," *IEEE Transactions on Network and Service Management*, vol. 15, pp. 1661–1675, December 2018.

[63] M. Elsayed and M. Erol-Kantarci, "Learning-Based Resource Allocation for Data-Intensive and Immersive Tactile Applications," in *IEEE 5G World Forum (5GWF)*, pp. 278–283, July 2018.

[64] 3GPP, "Evolved Universal Terrestrial Radio Access (E-UTRA); Medium Access Control (MAC) protocol specification," Technical Specification (TS) 136.321, 3rd Generation Partnership Project (3GPP), April 2015. Version 12.5.0.

[65] A. Saeed, E. Katranaras, M. Dianati, and M. A. Imran, "Dynamic Femtocell Resource Allocation for Managing Inter-tier Interference in Downlink of Heterogeneous Networks," *IET Communications*, vol. 10, pp. 641–650, April 2016.

[66] 3GPP, "Evolved Universal Terrestrial Radio Access (E-UTRA); Radio Frequency (RF) requirements for LTE Pico Node B," Technical Specification (TS) 36.931, 3rd Generation Partnership Project (3GPP), May 2014. Version 12.0.0.

[67] C. C. Coskun and E. Ayanoglu, "Energy- and Spectral-Efficient Resource Allocation Algorithm for Heterogeneous Networks," *IEEE Transactions on Vehicular Technology*, vol. 67, pp. 590–603, August 2018.

[68] 3GPP, "Technical Specification Group GERAN; GERAN Improvements for Machine-type Communications," Technical Specification Group GERAN 43.868, 3rd Generation Partnership Project (3GPP), November 2011. Version 0.5.0.

[69] Smith, Adam B., "U.S. Billion-dollar Weather and Climate Disasters," Tech. Rep. 36.104, National Oceanic and Atmospheric Administration, January 2014.

[70] T. Jiang, H. Wang, M. Daneshmand, and D. Wu, "Cognitive Radio-Based Smart Grid Traffic Scheduling With Binary Exponential Backoff," *IEEE Internet of Things Journal*, vol. 4, pp. 2038–2046, December 2017.

[71] L. Wu, J. Li, M. Erol-Kantarci, and B. Kantarci, "An Integrated Reconfigurable Control and Self-Organizing Communication Framework for Community Resilience Microgrids," *The Electricity Journal*, vol. 30, pp. 27 – 34, May 2017.

[72] L. Wu, T. Ortmeyer, and J. Li, "The Community Microgrid Distribution System of the Future," *The Electricity Journal*, vol. 29, pp. 16 – 21, December 2016.

[73] T. Ortmeyer, L. Wu, and J. Li, "Planning and Design Goals for Resilient Microgrids," in *IEEE Power Energy Society Innovative Smart Grid Technologies Conference (ISGT)*, pp. 1–5, September 2016.

[74] R. Hidalgo-León, C. Sanchez-Zurita, P. Jácome-Ruiz, J. Wu, and Y. Muñoz-Jadan, "Roles, Challenges, and Approaches of Droop Control Methods for Microgrids," in *IEEE PES Innovative Smart Grid Technologies Conference - Latin America (ISGT Latin America)*, pp. 1–6, September 2017.

[75] V. C. Gungor, D. Sahin, T. Kocak, S. Ergut, C. Buccella, C. Cecati, and G. P. Hancke, "A Survey on Smart Grid Potential Applications and Communication Requirements," *IEEE Transactions on Industrial Informatics*, vol. 9, pp. 28–42, February 2013.

[76] G. Pocovi, K. I. Pedersen, and P. Mogensen, "Multiplexing of Latency-Critical Communication and Mobile Broadband on a Shared Channel," in *IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–6, April 2018.

[77] H. Liao, P. Chen, and W. Chen, "An Efficient Downlink Radio Resource Allocation with Carrier Aggregation in LTE-Advanced Networks," *IEEE Transactions on Mobile Computing*, vol. 13, pp. 2229–2239, October 2014.

[78] D. López-Pérez, X. Chu, A. V. Vasilakos, and H. Claussen, "On Distributed and Coordinated Resource Allocation for Interference Mitigation in Self-Organizing LTE Networks," *IEEE/ACM Transactions on Networking*, vol. 21, pp. 1145–1158, August 2013.

[79] T. B. Sorensen and M. R. Pons, "Performance Evaluation of Proportional Fair Scheduling Algorithm with Measured Channels," in *IEEE Vehicular Technology Conference*, vol. 4, pp. 2580–2585, September 2005.

[80] J. Yang, Z. Yifan, W. Ying, and Z. Ping, "Average Rate Updating Mechanism in Proportional Fair Scheduler for HDR," in *Global Telecommunications Conference (GLOBECOM)*, vol. 6, pp. 3464–3466, November 2004.

[81] G. Miao, J. Zander, K. W. Sung, and S. Ben Slimane, "Scheduling," in *Fundamentals of Mobile Data Networks*, ch. 4, p. 65–94, Cambridge University Press, March 2016.

[82] H. Zhang, C. Jiang, N. C. Beaulieu, X. Chu, X. Wen, and M. Tao, "Resource Allocation in Spectrum-Sharing OFDMA Femtocells With Heterogeneous Services," *IEEE Transactions on Communications*, vol. 62, pp. 2366–2377, July 2014.

[83] Y. Y. Liu and S. J. Yoo, "Dynamic Resource Allocation using Reinforcement Learning for LTE-U and WiFi in the Unlicensed Spectrum," in *International Conference on Ubiquitous and Future Networks (ICUFN)*, pp. 471–475, July 2017.

[84] 3GPP, "Evolved Universal Terrestrial Radio Access (E-UTRA); Base Station (BS) Radio Transmission and Reception," Technical Specification 36.104, 3rd Generation Partnership Project (3GPP), October 2014. Version 12.5.0.

[85] Y. Su, X. Du, L. Huang, Z. Gao, and M. Guizani, "LTE-U and Wi-Fi Coexistence Algorithm Based on Q-Learning in Multi-Channel," *IEEE Access*, vol. 6, pp. 13644–13652, February 2018.

[86] H. Saad, A. Mohamed, and T. ElBatt, "Distributed Cooperative Q-Learning for Power Allocation in Cognitive Femtocell Networks," in *IEEE Vehicular Technology Conference (VTC Fall)*, pp. 1–5, September 2012.

[87] Z. Gao, B. Wen, L. Huang, C. Chen, and Z. Su, "Q-Learning-Based Power Control for LTE Enterprise Femtocell Networks," *IEEE Systems Journal*, vol. 11, pp. 2699–2707, December 2017.

[88] K. Zheng, Q. Zheng, H. Yang, L. Zhao, L. Hou, and P. Chatzimisios, "Reliable and Efficient Autonomous Driving: The Need for Heterogeneous Vehicular Networks," *IEEE Communications Magazine*, vol. 53, pp. 72–79, December 2015.

[89] K. Al-Begain and A. Ali, "Introduction to Mission Critical Systems and Its Requirements," in *Multimedia Services and Applications in Mission Critical Communication Systems.*, ch. 1, pp. 1–19, IGI Global, February 2017.

[90] Q. Zhang and F. H. P. Fitzek, "Mission Critical IoT Communication in 5G," in *Future Access Enablers for Ubiquitous and Intelligent Infrastructures*, pp. 35–41, Springer International Publishing, October 2015.

[91] M. Peng, C. Wang, J. Li, H. Xiang, and V. Lau, "Recent Advances in Underlay Heterogeneous Networks: Interference Control, Resource Allocation, and Self-Organization," *IEEE Communications Surveys Tutorials*, vol. 17, pp. 700–729, Second quarter 2015.

[92] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. A. Riedmiller, "Playing Atari with Deep Reinforcement Learning," *CoRR*, vol. abs/1312.5602, December 2013.

[93] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," in *Neural Computation*, vol. 9, pp. 1735–80, November 1997.

[94] M. Elsayed and M. Erol-Kantarci, "Deep Reinforcement Learning for Reducing Latency in Mission Critical Services," in *IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, December 2018.

[95] M. Chen, U. Challita, W. Saad, C. Yin, and M. Debbah, "Artificial Neural Networks-Based Machine Learning for Wireless Networks: A Tutorial," *IEEE Communications Surveys Tutorials*, vol. 21, pp. 3039–3071, July 2019.

[96] M. Chen, W. Saad, and C. Yin, "Optimized Uplink-Downlink Decoupling in LTE-U Networks: An Echo State Approach," in *IEEE International Conference on Communications (ICC)*, pp. 1–6, May 2016.

[97] B. Yang, K. V. Katsaros, W. K. Chai, and G. Pavlou, "Cost-Efficient Low Latency Communication Infrastructure for Synchrophasor Applications in Smart Grids," *IEEE Systems Journal*, vol. 12, pp. 948–958, May 2018.

[98] F. Tariq, M. Khandaker, K.-K. Wong, M. Imran, M. Bennis, and M. Debbah, "A Speculative Study on 6G," April 2019.

[99] M. Elsayed and M. Erol-Kantarci, "Reinforcement Learning-Based Joint Power and Resource Allocation for URLLC in 5G," in *IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, December 2019.

[100] A. A. Esswie and K. I. Pedersen, "Null Space Based Preemptive Scheduling for Joint URLLC and eMBB Traffic in 5G Networks," in *IEEE Globecom Workshops (GC Wkshps)*, pp. 1–6, December 2018.

[101] G. Pocovi, K. I. Pedersen, and P. Mogensen, "Joint Link Adaptation and Scheduling for 5G Ultra-Reliable Low-Latency Communications," *IEEE Access*, vol. 6, pp. 28912–28922, May 2018.

[102] 3GPP, "NR; Physical Layer Procedures for Data," Technical Specification 38.214, 3rd Generation Partnership Project (3GPP), July 2018. Version 15.2.0.

[103] G. Pocovi, B. Soret, K. I. Pedersen, and P. Mogensen, "MAC Layer Enhancements for Ultra-Reliable Low-Latency Communications in Cellular Networks," in *IEEE International Conference on Communications Workshops (ICC Workshops)*, pp. 1005–1010, May 2017.

[104] 3GPP, "Evolved Universal Terrestrial Radio Access (E-UTRA); Radio Frequency (RF) requirements for LTE Pico Node B," Technical Specification (TS) 36.931, 3rd Generation Partnership Project (3GPP), May 2014. Version 12.0.0.

[105] M. Elsayed and M. Erol-Kantarci, "AI-Enabled Radio Resource Allocation in 5G for URLLC and eMBB Users," in *IEEE 5G World Forum (5GWF)*, pp. 590–595, September 2019.

[106] Z. Ding, Z. Yang, P. Fan, and H. V. Poor, "On the Performance of Non-Orthogonal Multiple Access in 5G Systems with Randomly Deployed Users," *IEEE Signal Processing Letters*, vol. 21, pp. 1501–1505, December 2014.

[107] X. Zhang and M. Haenggi, "The Performance of Successive Interference Cancellation in Random Wireless Networks," *IEEE Transactions on Information Theory*, vol. 60, pp. 6368–6388, October 2014.

[108] M. Elsayed and M. Erol-Kantarci, "AI-Enabled Future Wireless Networks: Challenges, Opportunities, and Open Issues," *IEEE Vehicular Technology Magazine*, vol. 14, pp. 70–77, September 2019.

[109] W. Zhang, Y. Wei, S. Wu, W. Meng, and W. Xiang, "Joint Beam and Resource Allocation in 5G mmWave Small Cell Systems," *IEEE Transactions on Vehicular Technology*, vol. 68, pp. 10272–10277, July 2019.

[110] J. G. Andrews et al., "What Will 5G Be?," *IEEE Journal on Selected Areas in Communications*, vol. 32, pp. 1065–1082, June 2014.

[111] Y. Liu, X. Wang, G. Boudreau, A. B. Sediq, and H. Abou-zeid, "Deep Learning Based Hotspot Prediction and Beam Management for Adaptive Virtual Small Cell in 5G

Networks," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 4, pp. 83–94, January 2020.

[112] W. Saad, M. Bennis, and M. Chen, "A Vision of 6G Wireless Systems: Applications, Trends, Technologies, and Open Research Problems," *IEEE Network*, pp. 1–9, October 2019.

[113] 3GPP, "NR; Physical Channels and Modulation," Technical specification (TS) 38.211, 3rd Generation Partnership Project (3GPP), July 2018. V15.2.0.

[114] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," *Second International Conference on Knowledge Discovery and Data Mining*, p. 226–231, August 1996.

[115] N. I. B. Hamid, N. Salele, M. T. Harouna, and R. Muhammad, "Analysis of LTE Radio Parameters in Different Environments and Transmission Modes," *International Conference on Electrical Information and Communication Technology (EICT)*, pp. 1–6, March 2014.

[116] I. Comşa, G. Muntean, and R. Trestian, "An Innovative Machine-Learning-Based Scheduling Solution for Improving Live UHD Video Streaming Quality in Highly Dynamic Network Environments," *IEEE Transactions on Broadcasting*, pp. 1–13, April 2020.

[117] R. Baldemair, T. Irnich, K. Balachandran, E. Dahlman, G. Mildh, Y. Selén, S. Parkvall, M. Meyer, and A. Osseiran, "Ultra-Dense Networks in Millimeter-Wave Frequencies," *IEEE Communications Magazine*, vol. 53, pp. 202–208, January 2015.

[118] M. N. Soorki, M. J. Abdel-Rahman, A. MacKenzie, and W. Saad, "Joint Access Point Deployment and Assignment in mmWave Networks with Stochastic User Orientation," *International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, pp. 1–6, May 2017.

[119] Y. Saito, Y. Kishiyama, A. Benjebbour, T. Nakamura, A. Li, and K. Higuchi, "Non-Orthogonal Multiple Access (NOMA) for Cellular Future Radio Access," *IEEE Vehicular Technology Conference (VTC Spring)*, pp. 1–5, January 2013.

[120] T. Park, G. Lee, and W. Saad, "Message-Aware Uplink Transmit Power Level Partitioning for Non-Orthogonal Multiple Access (NOMA)," *IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, February 2018.

[121] S. M. R. Islam, M. Zeng, O. A. Dobre, and K. Kwak, "Resource Allocation for Downlink NOMA Systems: Key Techniques and Open Issues," *IEEE Wireless Communications*, vol. 25, pp. 40–47, April 2018.

[122] L. Dai, B. Wang, Y. Yuan, S. Han, C. I, and Z. Wang, "Non-orthogonal Multiple Access for 5G: Solutions, Challenges, Opportunities, and Future Research Trends," *IEEE Communications Magazine*, vol. 53, pp. 74–81, September 2015.

[123] K. Wang, J. Cui, Z. Ding, and P. Fan, "Stackelberg Game for User Clustering and Power Allocation in Millimeter Wave-NOMA Systems," *IEEE Transactions on Wireless Communications*, vol. 18, pp. 2842–2857, April 2019.

[124] M. E. Taylor, P. Stone, and Y. Liu, "Transfer Learning via Inter-Task Mappings for Temporal Difference Learning," *Journal of Machine Learning Research*, vol. 8, pp. 2125–2167, September 2007.

[125] G. Lee, Y. Sung, and J. Seo, "Randomly-Directional Beamforming in Millimeter-Wave Multiuser MISO Downlink," *IEEE Transactions on Wireless Communications*, vol. 15, pp. 1086–1100, February 2016.

[126] A. A. Esswie and K. I. Pedersen, "Multi-User Preemptive Scheduling For Critical Low Latency Communications in 5G Networks," in *IEEE Symposium on Computers and Communications (ISCC)*, pp. 00136–00141, June 2018.

[127] S. Gulia and A. Ahmad, "Physical Layer Performance Analysis of LTE Networks for Downlink," *International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, pp. 164–169, January 2019.

[128] E. Schubert, J. Sander, M. Ester, H. P. Kriegel, and X. Xu, "DBSCAN Revisited, Revisited: Why and How You Should (Still) Use DBSCAN," *ACM Transactions on Database Systems*, vol. 42, July 2017.

[129] C. Jiang, H. Zhang, Y. Ren, Z. Han, K. Chen, and L. Hanzo, "Machine Learning Paradigms for Next-Generation Wireless Networks," *IEEE Wireless Communications*, vol. 24, pp. 98–105, December 2017.

[130] S. Patel, H. Park, P. Bonato, L. Chan, and M. Rodgers, "A Review of Wearable Sensors and Systems with Application in Rehabilitation," *Journal of NeuroEngineering and Rehabilitation*, vol. 9, p. 21, April 2012.