Distributed interference management in dense subnetworks

Jeppe Bro Kristensen

Wireless Communication Systems, WCS-1023, 2022-05

Master's Project

Electronics and IT
Aalborg University
http://www.aau.dk

AALBORG UNIVERSITY
STUDENT REPORT

Title:
Distributed interference management in dense subnetworks

Theme:
Interference management

Project Period:
Spring semester 2022

Project Group:
WCS-1023

Participant(s):
Jeppe Bro Kristensen

Supervisor(s):
Gilberto Berardinelli
Ramoni Ojekunle Adeogun

Copies: 1

Page Numbers: 70

Date of Completion:
May 24, 2022

Abstract:

Interference is an inherent challenge in multi-agent radio communication. In-X subnetworks are envisioned to handle extreme demands of the 6th generation of radio access technology. Reinforcement learning is then applied to the task of interference management in In-X subnetworks. The purpose is to investigate whether reinforcement learning is a good candidate for the interference management task.

A Monte Carlo simulation is made based on a possible use case for In-X subnetworks. The simulation implements a dense subnetwork deployment, radio propagation models and channel allocation with Q-learning.

The results suggest that it is possible to use Q-learning for resource allocation. The performance is constrained by the state quantization thresholds and non uniform action distributions.

Contents

Preface

This thesis is the result of the 10th semester at the Wireless Communication Systems program at Aalborg University. The purpose is to investigate the applicability of reinforcement learning for the task of interference management.

Aalborg University, May 24, 2022

Jeppe Bro Kristensen
jbkr15@student.aau.dk

Abbreviations

5G Fifth Generation

6G Sixth Generation

ABSF Almost Blank Sub Frame

ADAS Advance Driver Assistance Systems

AP Access Point

AWGN Additive White Gaussian Noise

BAN Body Area Network

BS Base Station

CC Component Carrier

CDF Cumulative Distribution Function

CERT Concurrent Experience Replay Trajectories

CGC Centralized Graph Coloring

CNN Convolutional Neural Network

DCA Dynamic Channel Allocation

DL Down Link

DNN Deep Neural Network

eMMB enhanced Mobile Broad Band

FCA Fixed Channel Allocation

FNN Feed-forward Neural Network

G-ACCS Generalized Autonomous Component Carrier Selection

GRACE Game-based Ressource Allocation in a Cognitive Radio Environment

GRU Gated Recurrent Unit network

HetNet Heterogeneous Network

IID Independent Identically Distributed

IoT Internet of Things

LBT Listen Before Talk

LOS Line Of Sight

MAQL Multi-Agent Q-learning

MDP Markov Decision Process

MIMO Multiple-Input and Multiple-output

minSINR minimum SINR

ML Machine Learning

mMTC massive Machine Type Communications

MTC Machine Type Communication

MUE Macro User Equipment

NNCA Nearest Neighbor Conflict Avoidance

PC Power Control

PSD Power Spectral Density

QL Q-learning

RL Reinforcement Learning

SINR Signal to Interference plus Noise Ratio

SOCCER Self Organizing Coalitions for Conflict Evaluation and Resolution

TD Temporal Difference

UE User Equipment

URLLC Ultra Reliable Low Latency Communication

Summary of notation

$\doteq$	Equality relation that is true by definition
$\sim$	Is distributed as
$\mathcal{U}$	The uniform distribution
Pr{}	Probability
$\mid$	On the condition
$\lvert \cdot \rvert$	The size of the set $\cdot$
$\lVert \cdot \rVert$	The euclidean distance of vector $\cdot$
$\max_x f(x)$	The maximization of function f(x)
ϵ	The probability of taking a random action
α	The learning rate
γ	The discounting rate
s	Current state
s'	Next state
r	A reward
a	An action
$\subset$	The subset of
$\in$	Is an element of
t	Discrete time step
T	The total amount of discrete time step
A_t	Action at time t
S_t	State at time t
R_t	Reward at time t
G_t	The return after time t
p(s',r | s,a)	Probability of transitioning to state s' with reward r by taking action a in state s
$q_*(s,a)$	Value of taking action a in state s under optimal policy
Q_t	State-action value function estimate
$\mathcal{R}$	Deployment area
p_n	Center position of subnetwork n
$\mathcal{A}$	Subnetwork coverage area
$R_{\text{subnetwork}}$	The radius of the subnetwork area
Δ_n	Position of device n within subnetwork area
$s_{n'}$	The absolute position of device n in cartesian coordinates
K	The channel allocation set
k_n	Channel allocation of subnetwork n
m	The total amount of channels available for allocation
$Y_n^{m_i}$	The total received power at subnetwork n on channel m_i
X_n'	The transmit power of device n'

B	The bandwidth of the used channel
w_n	The noise power spectral density
$d_{n,n'}$	Distance between subnetwork n and device n'
$G_{n,n'}$	The channel gain between subnetwork n and device n'
$G_{pl}(d_{n,n'})$	The path loss gain as a function of distance
$h_{n,n'}$	The small scale fading gain between subnetwork n and device n'
$G_s(p_n, s_{n'})$	The shadowing gain as a function of position p_n and s'_n
$\delta[.]$	The Kronecker delta
m_i	The channel for which a value holds true
u_i	The power level to use
$I_n^{m_i}$	The total interference power at subnetwork n on channel m_i
$\text{SINR}_n^{m_i}$	The SINR value at subnetwork n on channel m_i
C_t	The capacity at time t
O	The target capacity for the binary reward function
$\mathbb{S}$	The set of states
s_{th}	The state quantization threshold value
$\mathbb{A}$	The set of actions
c_n	Channel available for allocation
p_n	Power level available for allocation
u	The total amount of power levels available for allocation
z	The total amount of channels to group together
$\epsilon(t)$	ϵ decay as a function of t
ϵ_{max}	The maximum value for the ϵ decay function
ϵ_{min}	The minimum value for the ϵ decay function
ϵ_{steps}	The amount of time steps to use to go from ϵ_{max} to ϵ_{min}
$LS_{n,n'}$	The large scale fading between subnetwork n and device n'
d_0	The reference distance for the large scale fading model
η	The path loss exponent
$cov(f(p_n), f(s_{n'}))$	The covariance between point p_n and $s_{n'}$
σ_s	The standard deviation for the shadow map
δ_d	The decorrelation distance for the shadow map
σ_r	The Rayleigh distribution scale parameter
NF	Noise figure

Chapter 1

Introduction

Electronic technology is in a constant state of evolution. Technological breakthroughs are sought in every avenue whether the goal is to make devices smaller, cheaper, less power consuming or something else there is a constant drive to achieve better performance. The push for better performance also affects the world of wireless communication. Indicative of this is the different generations of cellular networks that enable mobile telephony. The Fifth Generation (5G) of cellular technology is currently being adapted throughout the world. It promises higher data rates and stability through the enhanced Mobile Broad Band (eMMB) use case well know from our personal devices. The eMMB use case is not the only advancement made by the fifth generation. Progress is also made in the field of Machine Type Communication (MTC). MTC is characterized by the lack of a human initiator of the communication [11]. It is machines communicating with each other or a server. MTC is an enabler of the Internet of Things (IoT), smart -houses, -cities, -metering etc. Common for these use cases is small data packets but high machine density. The fifth generation of cellular technology addresses this type of communication through the massive Machine Type Communications (mMTC) core service. Another type of MTC addressed by 5G is the Ultra Reliable Low Latency Communication (URLLC). This core service aims to support industrial automation, self driving vehicles and remote surgery. Common for these applications is the use of control loops that require high reliability and low latency to guarantee stability [26].

With adaption of 5G well underway some research focus has shifted towards the Sixth Generation (6G) radio access technology[30]. No unequivocal vision for 6G has been reached by different working groups. Common for most though is the vision to seamless interconnect the digital-, physical- and human world. This vision poses a huge demand on the wireless technology that shall implement it. Targets that have been identified includes throughput exceeding 1 Tbps, reliability

of one in a billion bit errors and latency of no more than 0.1 ms[19]. In [7] these requirements where dubbed "extreme" and the need to move intelligence and decision making to the edge of the infrastructure was identified.

1.1 In-X subnetworks

The extreme demands envisioned in 6G has led to the proposal of In-X Subnetworks [6]. In-X subnetworks are to be deployed where the extreme demands arises. The subnetworks are necessary because the extreme demands requires a lot of radio resources. If the extreme demands where to be handled at a macro- or microcell level the base station would simply run out of resources.

The X part of In-X is substituted for the relevant use cases. Some examples are In-Robot, In-Vehicle, In-Body and In-House [7]. These use cases stipulate that the deployment of In-X subnetworks will happen in an uncoordinated fashion. It is the use cases that outlines where and when to use In-X subnetworks. Requirements on where somebody with an In-body subnetwork can go or which vehicle they can use would be detrimental to the applicability of In-X subnetworks.

The uncoordinated deployment approach is in stark contrast to the usual coordinated deployment scheme used in cellular networks.This departure from coordination exacerbate the problem of interference. Interference in this case is when two or more transmitters try to use the same radio resource. A relatable comparison is when two or more people speak at the same time in a conversation. It becomes more difficult for the listener to decipher what each individual says. In the same way it becomes difficult for a radio receiver to decipher a message if multiple transmissions are received at the same time.

The task of In-X subnetworks is to support extreme demands. The support of extreme demands is however not the only characterizing feature of In-X subnetworks. In [7] four characterizing features are defined. The four characterizing features of In-X subnetworks are:

- Support for extreme requirements
- Limited up- and downlink transmission power for a limited coverage area
- Hierarchical structure with tree or star topology controlled by an Access Point (AP)
- No mobility across subnetworks. Devices can only connect to one subnetwork for the operation time.

Use case	**In-robot, In-production module**	**In-vehicle**	**In-body**	**In-house**
Example of applications	Motion control, force/torque control, position/proximity control	Engine control, electric power steering, ABS, electric park brakes, suspensions, ADAS sensors	Heartbeat control, vital signs monitoring, insulin pumping, muscle haptic control	Entertainment, gaming training, education healthcare(robotic-aided surgery)
Number of devices	~20 (motion/force control) ~20-40 (mobile robots)	~50-100	<20	~10
Max range	~5 m	~10 m	~2 m	~10 m
Data rate per link	<10 Mbps	<10 Mbps (control) <10 Gbps (ADAS sensors)	<20 Mbps	~7 Gbps
Traffic type	Periodic, event-based	Periodic, event-based uncompressed video streaming	Periodic, event-based	Event-based, compressed video streaming
Min latency	~100 μs	~54 μs	~20 ms	~5 ms (VR) ~2 ms (healthcare)
Communication service availability	99.999 9 % to 99.999 999 %	99.999 9 % to 99.999 999 %	99.999 999 9 %	99.999 9 % (VR) 99.999 999 % (healthcare)
Max subnetwork density	~40 000 km/m^2	~150 / lane-km (car) ~15 /aircraft (~80 m long)	~2/m^2	~1 / room
Life-critical	No	Yes	Yes	No (entertainment, gaming, training education) Yes (healthcare)
Criticality of power consumption	Low	Low	High	Low/medium

Table 1.1: Table of key parameters for 6G In-X subnetworks. [7]

All four characteristics must be fulfilled to characterize an In-X subnetwork. The purpose of the In-X subnetworks can broadly be categorized into two categories. One is a low data rate, low latency, and high reliability. The other is high data rate with more relaxed latency and reliability requirements although compared to other technologies the latency and error rate is still low. The first category encompasses the use cases that require a closed loop control function for rapid processes. Such processes could be fast moving robots or ignition timing in an internal combustion engine. The second category involves application where data intense information is to be conveyed. Data intense information could be a video link used in a life critical application such as remote surgery or Advance Driver Assistance Systems (ADAS) in a vehicle [7]. A summary of identified use cases and their requirements are given in table 1.1

Combining some of the identified use cases can lead to one subnetwork contained within another i.e. a person with an in-body subnetwork who uses a car with it's own in-vehicle subnetwork. The uncoordinated deployment necessitates some considerations as to how to handle interference in areas dense with subnetworks.

1.2 Machine learning in wireless technology

Advances in electronic technology has led to a whole new era of connectivity. Whether it is computer, network or wireless technology they all play an instrumental role in the technological landscape today. The high level of connectivity and interaction has brought about massive amounts of data and computational power [15]. The large amount of data is incomprehensible for the individual who tries to analyze and process it [5]. It does however beg the question: *is it possible to use computational resources to learn from a data set* this question is the epitome of machine learning. Machine learning encompasses computer algorithms whose objective it is to infer knowledge about the given task. How the algorithms do it depends on the task at hand which can generally be split in to three categories. The three categories are: supervised learning, unsupervised learning ,and reinforcement learning [25].

Supervised learning relies on a knowledgeable agent to feed it labelled training data. The task of the supervised learning agent is to find a mapping between input data and the desired output which is given by the data labels. The learnt mapping can be applied to other data which was not in the training data set. Correct mapping for other data is based on the assumption that the training data set generalized the task that the learning agent is supposed to learn. This type of machine learning is used in classification and regression problems mapping input to labels or values respectively [25]. Examples of areas where supervised learning can be applied in wireless communication includes signal detection, channel encoding and decoding, channel estimation and prediction, and link adaption.

Unsupervised learning uses unlabelled data as input and finds clusters in the data set. The clusters are groupings of data points that are more similar to each other than the other data points. The learning agent looks for some kind of structure or other commonalities that tie data points together. This type of machine learning is used for tasks such as density function estimation [25].

Reinforcement learning is fundamentally quite different from the other two kinds of machine learning. The objective of the learning agent is to interact with its environment in order to learn how to solve a task. It does not need a large set of input data, it creates its own data through interaction. The learning agent interacts with its environment through its states and actions. A state is a summary of relevant information about the environment. The actions represents acts taken by the learning agent that influences the environment. As the action interact with the environment it leads to changes in the state. Each action is associated with a reward and the learning agent tries to maximize the cumulative reward. To max-

imize the cumulative reward it forms a policy. The policy is a mapping between the states and the probability of taking each action. To learn the policy some degree of exploration is necessary for the agent to learn all the rewards associated with the state and action space [28]. To use reinforcement learning careful considerations must be made in how to define states, actions and rewards. Unintended or detrimental behavior might ensue from a bad design. Reinforcement learning is an obvious choice to use in problems that require decision making based on environmental states. Robot trajectory planning, autonomous driving, game playing and interference management is obvious candidates for reinforcement learning solutions.

1.3 Scope of project

In the previous sections the concepts of 6G In-X subnetworks, interference challenges and machine learning was summarized. The scope of this project is to combine the task of interference management in 6G In-X subnetworks with the concept of reinforcement learning. The concept is to use reinforcement learning to learn how to allocate channels and transmit power levels in In-X subnetworks. The performance of the reinforcement learning method will be characterized by the achieved Signal to Interference plus Noise Ratio (SINR) and capacity values. A performance comparison is made between reinforcement learning and other methods such as a Centralized Graph Coloring (CGC) algorithm and a greedy selection algorithm.

Chapter 2

Interference management in dense subnetworks

In this chapter, the problem of interference is presented, along with common approaches for dealing with it. In particular, the most suited interference management approaches for subnetworks will be analyzed.

2.1 Deployment of In-X subnetworks

In this section an example is given on how a possible deployment of an In-X subnetwork could be.

To describe the concept of interference it is worthwhile to consider how In-X subnetworks are envisioned to be deployed. One of the prerequisites of an interference event is the close proximity of multiple radios. For interference to cause problems the deployment of In-X subnetworks must lead to multiple radios in close proximity of each other. In section 1.1 different use cases were identified. Even though each use case is different the problem of interference is the same for all. The level of interference is governed by the effective deployment density. The use cases with the highest probable density includes in-vehicle and in-robot subnetworks. The in-vehicle happens in close traffic and in-robot when a large amount of robots is deployed in the same area.

A possible scenario where In-robot subnetworks can be employed is in autonomous robots in a factory setting. Each robot uses its own subnetwork to communicate between a controller, actuators and sensors. The position of each robot in the factory depends on the given task for the robot. A very common task for an autonomous robot in a factory setting is to transport components or modules

between factory units. Many robots may operate at the same time and each one has its own distinct task. This leads to an environment where the robots would come very close to each other from time to time.

An environment where subnetworks may come in close contact with each other is the case of an uncoordinated radio deployment. An uncoordinated radio deployment is characterized by the lack of regard to radio performance in the placement of radio units. The placement of radio units is driven by other requirements such as the ability to move freely in space. Every use case summarized in section 1.1 has elements of an uncoordinated deployment. The in-Robot use case requires mobility so robots may come in close contact with each other. In-car subnetworks come in close contact in traffic. The in-body use case is even worse because an in-body subnetwork may be fully contained within other subnetworks. This situation arises when a person with an in-body subnetwork is a passenger in a vehicle with a subnetwork or is an occupant of a house with a subnetwork. The in-house use case is also uncoordinated as subnetworks will be placed by the end user with no regard to other subnetworks.

A coordinated deployment is one where radio performance is taken into consideration when placing some or all radio units. An example of a coordinated deployment is cellular networks. In a cellular network the Base Stations (BS) are placed in a hexagonal grid to tessellate the coverage area. In the context of in-X subnetworks it is possible to make a coordinated deployment in the in-production module use case. Production modules are static which enables the possibility of coordinating the deployment.

A comparison of a coordinated and uncoordinated deployment is shown in figure 2.1. Notice how some devices are part of multiple Access Points (APs) coverage areas in the uncoordinated case. This overlap in coverage area exacerbates the interference.

2.2 The problem of interference

The concept of interference was shortly presented in section 1.1. It was defined as two or more radios trying to access the same resources shared by multiple devices. The resources in the interference case can be subdivided into three domains: space, time and frequency. An overlap must occur in all domains for an interference event to happen. Interference deteriorates the receivers ability to discern the received message.

In the context of cellular networks two types of interference exist: inter-cell

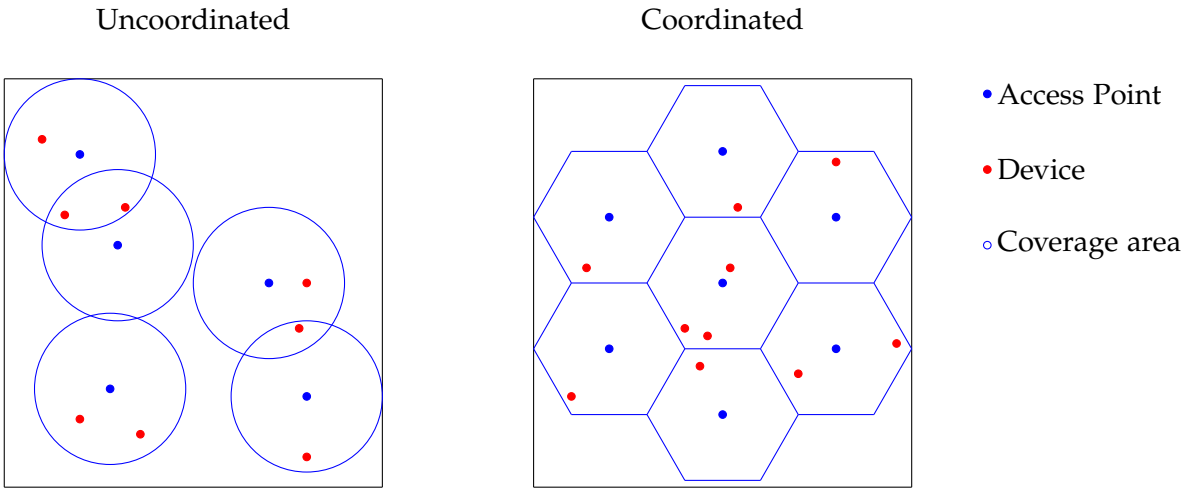

Figure 2.1: A comparison of an uncoordinated and a coordinated radio deployment. Notice the overlap in coverage area in the uncoordinated case.

and intra-cell. The distinction between the two is the relation between the interfering radios. In intra-cell interference the interfering radios belong to the same cell. An example is two User Equipments (UEs) transmitting at the same time in the uplink. In intra-cell interference the interfering radios belong to different cells. The worst case scenario in intra-cell interference is when UEs are located along a cell edge. Assuming that the cell radius is equal for both cells and the fading is roughly equal then the downlink signal from each cell is received with almost equal strength. In the uplink a similar problem occurs. The signal from the UE is received with roughly equal strength at both BS. In figure 2.2 the geometry of a UE on a cell edge is shown.

A fourth possible radio resource domain is spreading codes as used in spread spectrum access. The use of spread spectrum techniques requires Power Control (PC) to mitigate the near-far problem [22]. PC is a method where a control loop is used to vary transmission power. The control loop guarantees a link quality with the lowest possible transmission power which in turn corresponds to the lowest possible interference power. PC methods exist on a inter-cell level. Applying PC with coordination between cells to mitigate intra-cell interference is a much more complex task. As PC in spread spectrum requires coordination between cells it is considered a poor choice for networks using uncoordinated deployment. Spread spectrum techniques are thus not considered a viable option for In-X subnetworks. Another issue with PC is the control loop sample time. When the communication has to operate with very low latencies the sample time must be very small. This introduces a large overhead in communication which is not desired.

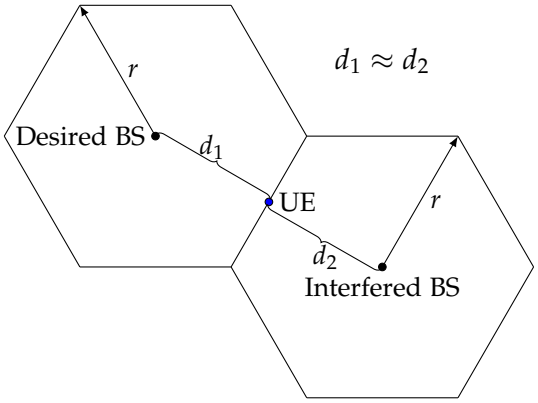

Figure 2.2: The cell geometry of a worst case interference scenario with an UE on a cell edge.

2.3 Interference management techniques

In this section the different radio resources and interference mitigation methods within each domain is presented.

2.3.1 Spatial interference management

Propagation of radio waves have naturally occurring attenuation with distance due to path loss [22]. This phenomenon introduces the idea of separating radios in space with a distance such that the interference power between the radios becomes negligible [22]. Separation is a very simple solution to the problem of interference but it is often not a viable solution. To use space separation rules must be applied on the placement of radios that is it necessitates a coordinated deployment. An example of such deployment is cellular network planning where BS are placed in hexagonal grids to tessellate the coverage area. Many applications dictate requirements in opposition to intelligent placement such as the need for mobility or lack of a placing agent.

Another spatial interference management method is PC as described in section 2.2. By reducing the transmission power the perceived interference power level for others are also reduced [14]. PC is a spatial method because the lower power reduces the coverage area due to path loss.To have a positive effect for every radio in range of each other everyone must use PC.

Directional antennas are also a case of a spatial interference management technique. The directional antenna differs from the omnidirectional antenna in that it has a peak in directivity i.e. it transmits and receives more power in a certain di-

rection than others [14]. This solution is appropriate when it is possible to aim the antennas at each other. An example of where directional antennas are used is when sectoring a cell. Here the directional antennas replace an omnidirectional one. The cells are divided into sectors of typically 120 or 60 degrees. In mobile applications the directional antenna must be mounted on a servo gimbal to aim. A servo gimbal is both expensive and bulky so their use is limited. Another issue with aiming is the necessity of tracking the recipients position. An application where the price and bulkiness of an aimed antenna is tolerated is in satellite communication.

Beamforming is similar to directional antennas in that it achieves directionality. The method to achieve directionality is different though. A directional antenna achieves its directionality through its physical design. In beamforming the directionality is achieved with an antenna array and signal processing [22]. An antenna array is a collection of antennas in close physical proximity. The radiation pattern of each antenna element augment each other to form a combined radiation pattern. By adjusting phase and amplitude of the signal for each antenna element in the array it is possible alter the combined radiation pattern and form a steerable beam. As no mechanical movement of the antenna array is necessary to aim beamforming is more rapidly adaptable. Similarly to directional antennas it is also necessary to either track the recipients position or derive it based on the recipients transmissions.

Common for both directional antennas and beamforming is that they will only work as a solution to interference if the interfering radios are not in the same directions as the desired link. Another drawback is the size of directional antennas and antenna arrays. They are generally bulkier than non directional antennas. This is a limitation when the application requires a small form factor.

2.3.2 Time domain interference management

One of the prerequisites of an interference event to happen is for two or more transmission to take place at the same time. An opportune possibility is then to separate all transmissions in time. To separate in time each radio must have knowledge about when other radios transmits. A regulatory solution to this problem is the Listen Before Talk (LBT) procedure where each radio listens whether any other radios are transmitting before starting its own transmission [22]. This approach is used in unlicensed bands.

Another way to coordinate transmissions in the time domain is to use a central coordinator that keeps track of every radio in an area and their wish to transmit.

This case is often encountered in cellular networks where the central coordinator is the scheduler of the BS. The scheduler allocates unique time slots for each device. This solution requires strict time synchronization between BS and the device. The time synchronization is necessary because if a device were to drift out of time alignment it might end up transmitting out of turn and thereby interfere the communication of another radio [14]. The centralized coordination is often used in licensed bands as regulations for the unlicensed bands may prohibit its use.

Heterogeneous Network (HetNet) is the coexistence between cellular networks and femtocells. Femtocells are short range devices using the same resources as the cellular network it exist within. Femtocells are deployed based on demand and with no regard for the cellular network. To coexist in time domain Almost Blank Sub Frames (ABSFs) are used. ABSF is sub frames where only reference subcarriers are transmitted. Macro User Equipments (MUEs) in the vicinity of the femtocell are then scheduled to transmit in the blank subframe and will not interfere the femtocell [20].

2.3.3 Frequency interference management

Frequency interference management concerns the frequency at which a transmission takes place. Transmissions on different frequencies will not interfere with each other. A simple solution to interference could be to assign a unique frequency to each radio pair that wish to communicate [14]. Unfortunately there is a scarcity of available frequencies. The scarcity is caused partly by technological limitations and partly by the properties of different frequencies. Properties such as propagation through walls is only observed for a limited set of frequencies. Only a finite amount of frequencies is available and the usage of large parts of it are regulated. Therefore one has to buy a license to be the sole user of a frequency block or operate in a frequency block that is free to use [22]. The free to use blocks however suffers from congestion and it is unlikely to be interference free. Especially the blocks below 10 GHz see heavy usage.

2.4 Interference management in In-X subnetworks

In this section the different interference domains are analyzed for their eligibility for In-X subnetworks.

With three possible domains to combat interference in some considerations can be made as to which is suitable for In-X subnetworks.

The spatial domain has multiple drawbacks when it comes to In-X subnetworks. The concept of separation distance is impossible in an uncoordinated deployment. PC is a possibility and may be beneficial with respect to power consumption. It depends on how much power is necessary to transmit feedback on the received Signal to Interference plus Noise Ratio (SINR) versus how much power is saved with respect to the transmissions. The eligibility of directional antennas and beamforming depends on the form factor of the antennas. The size of the antennas in turn depends on the frequency used. The higher the frequency the lower the wave length and smaller antennas can be used. No decision has been made on what frequency area In-X subnetworks should operate in. Therefore it is difficult to assess the eligibility of directionality methods with respect to antenna size. Even if the antenna form factor is not a problem the directionality methods still suffer from the case of an interferer being in between entities in a desired link. Another consideration is the necessity to track mutual positions that introduce overhead to the communication. This overhead is detrimental because it requires even greater radio performance to support it. The directionality methods are therefore not considered a viable option for In-X subnetworks.

The time domain separation is in direct opposition to the low latency requirement. Postponing a transmission would require even faster pre- and post processing of the signal to meet the latency requirement. The processing time without scheduling delay is already a difficult requirement to meet so time separation is considered unfeasible for In-X subnetworks.

The frequency domain is much more promising than the other two domains. Even if frequencies are a limited resource it still ought to be possible to operate in a frequency band that can be subdivided into a number of channels. Frequency interference management is thus the primary method to use in In-X subnetworks.

Even though the three domains have been presented separately there is nothing preventing a combination of multiple domains to be used. PC can be combined with frequency allocation and is a viable option in In-X subnetworks.

2.4.1 Interference management in the frequency domain

Two types of interference management is employed in the frequency domain. Those based on heuristic methods and those based on Machine Learning (ML) methods. To discuss the different methods some key concept should be explained. The first concept is whether the interference management is coordinated or un-

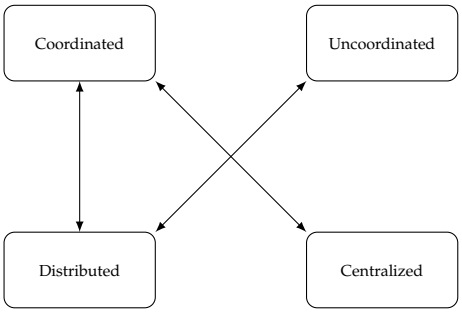

Figure 2.3: The relations between (un)-coordinated and centralized and distributed interference management.

coordinated. In the coordinated case relevant information is exchanged between radio entities. In the uncoordinated case no information is shared. Another concept is whether the interference management is centralized or distributed. In a centralized setup a central agent handles every channel allocation decision for every radio entity. In a distributed scheme each radio is responsible for its own channel allocation. The two concepts combine into three possible classifications of interference management methods. The centralized method is always coordinated. The distributed method can be either coordinated or uncoordinated. In figure 2.3 the relations of the concepts are summarized.

Two groups of interference mangament methods exist: those based on heuristics and those based on ML.

Heuristic channel allocation methods

As In-X subnetworks are a relatively new area of research, inspiration for how to do channel allocation must be sought elsewhere. Two research areas that touch upon the problems of interference, channel allocation, and uncoordinated deployment are cognitive radio and HetNets.

Cognitive radio has come about due to the scarcity of frequency resources. Cognitive radio seeks to utilize licensed bands when they are unused [10]. The users of the licensed band is divided into two groups: the primary users which holds the license, and the secondary users. The cognitive part of the secondary users relates to how they sense the presence of the primary users and adapt their own communication to not interfere the primary users.

HetNets were introduced in subsection 2.3.2 as femtocells deployed in an uncoordinated fashion in a cellular network. HetNets are relatable in the sense that In-X subnetworks are also deployed in an uncoordinated fashion so the problem

of interference is similar between the two.

Many methods have been proposed on how to handle the channel allocation task. Some of the first methods for cellular networks were Fixed Channel Allocation (FCA) where a centralized agent made an allocation based on maximizing the frequency reuse distance [18]. FCA evolved into Dynamic Channel Allocation (DCA) where a centralized agent can redistribute frequency resources based on demand [18].

Two distributed and uncoordinated frequency allocation methods have been proposed in [8] and [12]. The first method is based on balancing resources with game theory and using cognitive radio to sense the environment. Game theory is a mathematical way to describe decision making among multiple agents. It encompasses complex interactions between agents based on each agent's own decisions. Such decision processes are often encountered in games where each player must chose a strategy to beat the other players. When game theory is applied in ressource allocation it is often with purpose to balance resource use among agents. The method in the paper is called Game-based Ressource Allocation in a Cognitive Radio Environment (GRACE).

The second method proposes to use local knowledge to identify a pool of low interference resources and make them available to the scheduler. The paper uses game theory to prove the stability of allocation with selfish agents. It also compares performance of the distributed approach and an optimal centralized solution.

Two distributed and coordinated methods have been proposed in [31] and [13]. The first one proposes the Generalized Autonomous Component Carrier Selection (G-ACCS) scheme. The scheme is based on cognitive sensing of the environment and explicit communication between femtocells about which Component Carriers (CCs) they use and at which power level. This information is used to make the CC allocation for each cell.

The second method is based on coalitional game theory. It is called Self Organizing Coalitions for Conflict Evaluation and Resolution (SOCCER). A coalition is a group of cells that agree on a certain policy on how to share CCs among them. This method differs from G-ACCS in that it reduces overhead by limiting coordination to those femtocells that are in a coalition.

In [2] and [1] some heuristics based channel allocation methods were applied to in-X subnetworks. In [2] three distributed methods were tried. The three methods are: ϵ-greedy channel selection, minimum SINR (minSINR) guarantee, and

Nearest Neighbor Conflict Avoidance (NNCA). The ϵ-greedy algorithm selects the channel with the lowest interference power with probability ϵ. with the remaining probability $1 - \epsilon$ it selects the channel at random. The minSINR selects the worst channel that will satisfy a minimum SINR requirement. If no channel is able to satisfy the SINR requirement the channel with the least interference power is selected. The NNCA method relies on the controllers ability to discern which channels are in use by $m - 1$ nearest neighbor subnetworks where m is the amount of channel available. This method then selects the remaining channel not in use by the nearest neighbor. The NNCA method only rely on local measurements to infer the channels in use by neighboring subnetworks. Every method is then distributed and uncoordinated. In [1] the ϵ-greedy and NNCA method is also used. The only difference is that the ϵ is fixed at one which leads to a purely greedy selection. The ϵ-greedy method may lead to channel changes relatively often. Many channel changes requires communication overhead to signal the devices about the channel change. The minSINR may also lead to rapid channel changes. Another issue is the choice of the worst channel. This choice assumes that the channel performance will remain constant for the next transmission. In an uncoordinated method no guarantee of this can be made. This leads to a higher probability of packet loss. The NNCA adds complexity to the controller to infer who the nearest neighbors are. This complexity may be detrimental in terms of cost or it may migrate to the overall system design and make it unfeasible.

Machine Learning Based Channel Allocation Methods

As ML has gained ground it has become relevant to consider ML as a possible method to use for channel allocation tasks.

One approach to applying ML to channel allocation is to use supervised learning to learn how to map between channel states and channel selection. In [3] a Deep Neural Network (DNN) is used to learn a map based on a graph coloring algorithm. The DNN performs well even in propagation environments different from those used for training. The DNN method does not perform better than heuristic methods.

In [16] a similar approach is used where a convex optimization algorithm does the assignment but at high computational load. To optimize the computational load different ML methods were tried. The ML methods used were Convolutional Neural Networks (CNNs), Feed-forward Neural Networks (FNNs), random forest and Gated Recurrent Unit networks (GRUs). The ML methods reduce computational complexity with only a small loss in performance compared to the convex optimization solution.

Another possible approach is to use reinforcement learning. This method learns how to allocate channels through interactions with the environment which it belongs to. In [4] reinforcement learning was used for channel assignment in a Body Area Network (BAN) context. The error rate is improved by 30% compared to a fixed channel assignment. This master thesis focuses on applying reinforcement learning to the channel allocation task for in-X subnetworks. A thorough explanation of how to apply reinforcement learning in In-X Subnetworks will be given in chapter 4.

2.5 Summary

In the preceding chapter the concept of interference was presented. It was defined as overlaps in transmissions of two or more radios. Two different types of radio deployment were defined: coordinated and uncoordinated. Three domains of resources were described. How they contribute to interference and how they can be used to mitigate interference was explained. The frequency domain was identified as the most applicable for In-X subnetworks. Different state of the art methods for channel allocation were briefly summarized both methods based on heuristics and those based on ML.

Chapter 3

Reinforcement learning

This chapter presents the concept of reinforcement learning and the mathematical framework to analyze it. This chapter is largely inspired by [28].

In section 1.2 the basic concept of reinforcement learning was introduced. Reinforcement learning consists of a learning agent that learns to perform a task through interaction with an environment. Some examples of problems where reinforcement learning could be applied is to solve a maze or play a game. For both problems there is a clear defined goal i.e. solve the maze or win the game. As the learning agent does not have any prior knowledge about the given problem it resorts to a trial and error approach. For each trial the learning agent formalizes what it has learned from the trial. It is advantageous to do as many different trials as possible and learn from them in the beginning. The learning agent is said to be in an exploration phase. As the trials progress the learning agent becomes wiser. It can begin to use what it has learned to affect which trials to try next. The learning agent is now progressing to an exploitation state. The formal definition in the mathematical sense of reinforcement learning is made with Markov Decision Processes (MDPs).

3.1 Markov decision processes

In this section Markov Decision Processes is presented as a framework to use for the mathematical analysis of reinforcement learning.

To thoroughly understand reinforcement learning a more formal definition of the learning agent and the environment is necessary. The learning agent is an entity with ability to perform actions and learn from them. The actions influence the environment. The environment consists of two parts: the states, and the rewards. The states summarise everything of importance for the learning agent in the en-

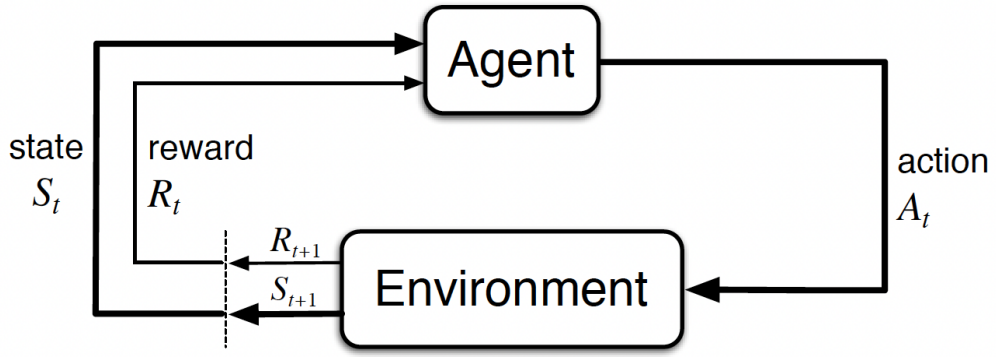

Figure 3.1: The agent and environment interaction model[28].

vironment. The reward is a numerical value that expresses how well the learning agent is performing with respect to the goal. The agent environment interaction is described in figure 3.1. The interaction happens at discrete time steps $t = 1, 2, 3...$ At time step t the learning agent receives the state S_t based on the state it chooses the action A_t. At the next time step $t + 1$ the learning agent receives a new state S_{t+1} and a reward R_{t+1}. The new state and the reward values are influenced by the chosen action. Other factors that influence the states and rewards might be beyond the control of the learning agent.

If the sets of all states, actions and rewards have a finite number of elements the MDP is said to be finite. In this case it is possible to describe S_t and R_t as random variables with discrete probability distributions [24]. The Markov property states that the probability of each possible S_t and R_t depends only on the immediately preceding state and action S_{t-1}, A_{t-1}. For the remainder of this chapter it will be assumed that the Markov property holds. The probability of a certain state $S_t = s'$ and reward $R_t = r$ happening at time step t is then defined as

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

Where the $|$ denotes conditional probability. An interesting consequence of the Markov property is the requirement it poses on the states. The states must succinctly summarize every past agent-environment interaction.

The description of MDPs has been made at a high level of abstraction. As MDPs are very versatile and can be applied to many different problems the high level of abstraction is necessary. The choice of how to define the interface between learning agent and environment i.e. the states, actions and rewards are of crucial importance to the succes of the learning agent. How to design these parts depends on the goal of the reinforcement learning that is the task the designer seeks to

solve.

3.1.1 Goals and rewards

Recall that the reward is a value from the environment at each time step and it is indicative of how well the learning agent is performing with respect to the goal. The learning agent seeks to maximize the cumulative reward over time. The rewards have to formalize what is beneficial and what is detrimental to do. In the previous used example of solving a maze, the reward could be -1 for every time step that the maze is not solved. This reward encourages the learning agent to solve the maze as quickly as possible. If the goal is to win a game the reward could be +1 when the game is won or -1 if it is lost and 0 for every time step where it is neither won or lost. It is important in this case to reward only the final time step. If sub goals were rewarded as well it might result in behaviour where the learning agent becomes more focused on achieving sub goals rather than to actually win. The rewards are the designers way of telling the learning agent what to achieve but not how to achieve it.

3.1.2 Returns

In the previous subsection it was stated that the learning agent seeks to maximize the cumulative reward. What to maximize is not immediately clear from this definition. At time step t the parameter to maximize is the expected return. The return is a function of the reward sequence $R_{t+1}, R_{t+2}, R_{t+3}, ...$ in its simplest form it is just the sum of the reward sequence

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

Here G_t is the return at time step t and R_{t+1} is the reward at time step $t+1$ and so on and T is a final time step. The return is well defined for episodic tasks. That is tasks that have a terminal state. Such tasks could be playing a game or solving a maze. Other task might not have a terminal state but will run forever. These are continuing tasks. A problem then arises with the definition of the return being the cumulative future reward. No guarantee can be made that a infinit sum will converge for any reward formulation. The cumulative reward might even approach infinity which makes it impossible to maximize. For continuing tasks the return formulation must contain discounting. Discounting introduces a factor to the rewards in the sum. The introduced factor is called the discount rate and it is denoted by γ. The discount rate must fall within the interval $0 \leq \gamma \leq 1$. In equation (3.3) the return with discounting is shown.

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

The choice of the discount rate influences the learning agents outlook with respect to time. For $\gamma = 0$ the learning agent is only concerned with the immediate next reward. When the discount rate approaches one the learning agent becomes more far sighted and put greater emphasis on future rewards.

3.2 Q-learning

Q-learning (QL) is part of a class of reinforcement learning methods called Temporal Difference (TD) [32]. The characterizing features of TD compared to other reinforcement learning methods is its ability to bootstrap. Bootstrapping is to base new estimates on old estimates as time progresses. The quantity to estimate is the state-action value function. The state-action value function describes how beneficial it is to chose a certain action when in the given state. The task of the QL learning agent is then to find an optimal policy based on the state-action values. The policy is what constitutes the learning agent that is it defines which actions to take based on the state-action values.

TD methods can be split into two problems: how to estimate the state-value function, and how to find an optimal policy. The estimation of the value functions with TD methods is described in the update equation (3.4). The update equation is based on the Bellman equation [28].

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t)] \tag{3.4}$$

$Q(S_t, A_t)$ is the estimate of the value function for state S_t and action A_t, and α is a parameter on the interval (0,1] called the learning rate. The learning rate describes how much is learnt at each time step. The bootstrapping is evident in that the value estimate $Q(S_t, A_t)$ is based on the future value estimate $Q(S_{t+1}, a)$.

The characteristic that sets QL apart from other TD methods is how it finds an optimal policy. QL is an off-policy TD algorithm which means that it learns the optimal policy even if it does not follow the optimal policy. The key difference is the maximum selection of the next state-action value with respect to the actions. The maximum selection is what makes QL an off-policy algorithm. In the ordinary case what is learnt is the value of the chosen action but in QL it is the maximum value of all possible actions.

The off-policy aspect of QL enables a very simple action selection. A ϵ-greedy algorithm is used to select actions based on the state-action value estimates. With probability ϵ the algorithm choses a random action and with probability 1-ϵ the

algorithm chooses the action with the greatest Q-value. The ϵ parameter is at the heart of the exploration and exploitation dilemma. This dilemma encompasses the need for experience from all states and actions to learn the optimal policy with the desire to perform optimally at each time step. It is proofed that the estimated state-action values will converge to the optimal state-action values q_* if every state and actions keep being visited [28].

The application of QL is described in the algorithmic description 1.

Algorithm 1 Q-learning algorithm

1: $S_0 \leftarrow$ Initial state
2: $\underline{\underline{Q_0}} \leftarrow \underline{\underline{0}}$
3: $T \leftarrow$ Number of discrete time steps to use
4: **for** t < T; t++ **do**
5: Choose A_t form Q_t with S_t and ϵ-greedy policy
6: Take action A_t and observe R_{t+1}, S_{t+1}
7: Update Q_t with update equation
8: **end for**

3.3 Summary

In this chapter the key concepts of reinforcement learning was presentend. At the core of reinforcement learning is the learning agent which is capable of learning through interaction with an environment. The environment consists of states and rewards. The states are the information presented to the learning agent that are relevant for the learning agent to take an action. The actions are decisions that affects the environment. The rewards are values that describe how beneficial the actions were in completing the task of the learning agent. The rewards arise as a direct consequence of the actions. Based on the rewards the learning agent formulates a policy. The policy dictates which actions to take in which states. To form the policy the learning agent uses value functions. The value functions describes how beneficial a state or a state action pair is in the terms of expected reward.

QL is reinforcement learning method of the temporal difference type. It uses bootstrapping where estimates are based on other estimates. QL is an off-policy method. It achieves the off-policy by using a maximum selection over actions in its update equation. A simple ϵ-greedy algorithm can be used to select actions. The choice of the ϵ parameter is difficult due to the exploration and exploitation dilemma. It is possible to proof that QL achieves the optimal state-action values if every state and actions keeps being visited.

Chapter 4

Interference management with Q-learning

In this chapter a system model of the proposed solution on how to apply Q-learning (QL) to the interference management problem is presented. Comparison metrics are derived and a thorough explanation of how to apply QL is given.

4.1 System model

In section 2.1 the In-robot use case was introduced. This use case is chosen as the basis for the system model. The task of an in-robot subnetwork is to facilitate communication between a controller, sensors, and actuators. The controller is in this case equivalent to the Access Point (AP) the sensors and actuators are collectively called devices. The AP makes the channel allocation based on uplink performance. Each robot contains a single subnetwork. The total number of subnetworks that coexist in the system is denoted as $n_{\text{subnetworks}}$. The subnetworks exist in a rectangular area denoted

$$\mathcal{R} = [0, x] \times [0, y] \subset \mathbb{R}^2$$

The position of a subnetwork n in the area is distributed uniformly

$$p_n \sim \mathcal{U}(\mathcal{R})$$

Where $\mathcal{U}$ is the uniform distribution. The shape of each subnetwork is assumed identical and circular with the AP at the center. The position of one subnetwork is dependent on the position of every other subnetwork. This distinction is made to avoid physical overlaps of the robots which are not possible.

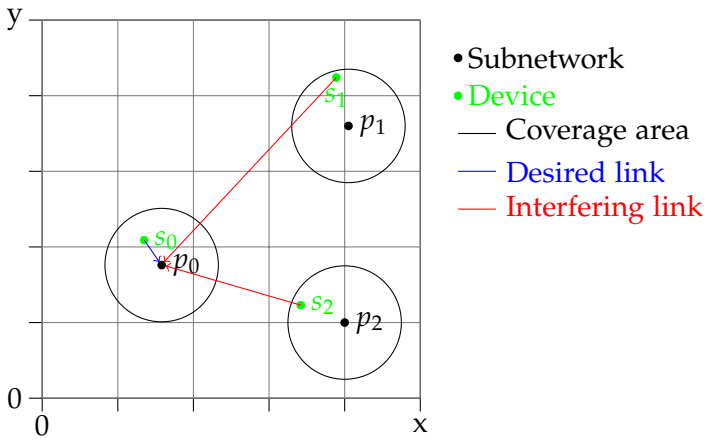

Figure 4.1: A subnetwork deployment with three subnetworks and one device each and illustrations of desired and interfering links.

The number of devices per subnetwork is denoted n_{devices}. The devices are within the subnetwork area and randomly distributed. The subnetwork area is given as

$$\mathcal{A} = \{\Delta \in \mathbb{R}^2 : 0 \leq |\Delta| \leq R_{\text{subnetwork}}\} \cap \mathbb{R}^2$$

Where $R_{\text{subnetwork}}$ is the subnetwork radius. The position of the device n in the subnetwork area is given as

$$\Delta_{n'} \sim \mathcal{U}(\mathcal{A})$$

The absolute position of the sensor n in the area $\mathcal{R}$ then becomes

$$s_{n'} = p_n + \Delta_{n'}$$

In figure 4.1 an example of an deployment is shown with three subnetworks each with one device and illustration of desired and interfering links.

The main task of the system is to make a channel allocation. The channel allocation assigns one or more channels to each subnetwork. The allocation is

$$K = [k_0, k_1, \ldots, k_{n_{\text{subnetworks}}}] | k_n \in \{1, 2, \ldots, m\}$$

Where m is the total amount of channels available. To keep the notation intuitive it does not account for multiple channels in the allocation even though it is possible. The channels in this context is a set of non overlapping partitions of the total available bandwidth in to equal parts. The bandwidth of each part is denoted

the channel bandwidth B.

The total received power at a controller is then given as

$$Y_n^{m_i} = \sum_{n'} X_{n'} \cdot G_{n,n'} \cdot \delta[k_n = k_{n'}] + B \cdot w_n$$

$Y_n^{m_i}$ is the received power at subnetwork n on channel m_i, $X_{n'}$ is the transmit power of device n', $G_{n,n'}$ is the channel gain, δ is the Kronecker delta and w_n is the receiver noise Power Spectral Density (PSD).

The channel gain is defined as

$$G_{n,n'} = G_{pl}(d_{n,n'}) \cdot |h_{n,n'}|^2 \cdot G_s(p_n, s_{n'}) \tag{4.1}$$

Where G_{pl} is the path loss gain and it is a function of distance between AP and the device. $h_{n,n'}$ is the small scale fading and G_s is shadowing gain and it is a function of AP and device position. The distance is calculated as

$$d_{n,n'} = \|p_n - s_{n'}\|$$

Where the $\|.\|$ is the euclidean norm operator. The interference power at a controller is then

$$I_n^{m_i} = \sum_{n' \neq n} X_{n'} \cdot G_{n,n'} \cdot \delta[m_i = k_{n'}]$$

A new index is added to the notation m_i which indexes which channel the interference value is true for. The Signal to Interference plus Noise Ratio (SINR) is

$$\text{SINR}_n^{m_i} = \frac{X_n \cdot G_{n,n}}{I_n^{m_i} + B \cdot w_n} \tag{4.2}$$

The addition of m_i is made because in the channel allocation task it is of interest to know the SINR for every channel not just the one in use. Note that the SINR values for the channels not in use assume that the desired link power will be equal across all channels.

4.2 Channel allocation problems

Interference is an inherent problem in multi-agent radio systems. To mitigate interference in the frequency domain a channel allocation must be made. To succinctly define an optimization problem some performance metric must be defined. An obvious choice to characterize the level of interference is to use the interference

power. The interference power in itself does no tell anything about how much information is exchanged. A metric closer related to communication performance is the SINR level. The SINR contains information about the interference power but it takes the ratio between the desired link power and the interference power plus the Additive White Gaussian Noise (AWGN). The SINR metric manages to capture dynamics which the interference power does not. For instance if the interference power is very high but so are the desired link power the communication performance might still be good. The opposite is also true if both interference power and link power are very low the communication performance might be poor. It all depends on the ratio not absolute power. Another metric closely related to the SINR is the capacity. The capacity is the theoretical upper bound of information that can be transmitted. The Shannon-Hartley theorem describes the relation between the capacity and the SINR

$$C = B \cdot \log_2(1 + \text{SINR})$$

Where C is the capacity and B is the channel bandwidth. The Shannon-Hartley theorem assumes that the interference is noise like. This assumption might not hold true but the capacity is still useful as a comparison metric. The theorem is applied consistently so any bias introduced will be equal for all results.

With the performance metrics defined a formulation of the resource allocation task can be given. Three separate but similar problems will be considered. Common for all problems is the objective of maximizing the sum capacity.

Problem I

The first problem is to only allocate a single channel.

$$K = \max_{\mathcal{C}} \sum_{n=1}^{n_{\text{subnetwork}}} B \cdot \log 2(1 + \text{SINR}_n^{m_i}) \tag{4.3}$$

Where $\mathcal{C} = \{m_i \in \{1, m\}\}_{i=1}^{n_{\text{subnetwork}}}$

Problem II

For the second problem the transmit power level is added to the resource allocation task

$$K = \max_{\mathcal{C},\mathcal{K}} \sum_{n=1}^{n_{\text{subnetwork}}} B \cdot \log 2(1 + \text{SINR}_n^{m_i}) \tag{4.4}$$

Where $\mathcal{C} = \{m_i \in \{1, m\}\}_{i=1}^{n_{\text{subnetwork}}}$ and $\mathcal{K} = \{u_i \in \{1, u\}\}_{i=1}^{n_{\text{subnetwork}}}$ Where u_i is the power level to use and u is the amount of power levels available for allocation.

Problem III

The third problem uses channel aggregation for the channel allocation. Channel aggregation is to use multiple channels simultaneously. Using more channels effectively increase the bandwidth which in turn increase capacity. The optimization formulation is the same as for problem I with the only difference being that multiple channels can be allocated. Then m_i becomes a set where $z = |m_i|$ and z is a design parameter to chose.

4.3 Reinforcement learning in the optimization task

In chapter 3 reinforcement learning was presented as a method to learn how to solve a task through experience. This makes reinforcement learning a possible candidate for the channel allocation task. The QL method will then be applied to the channel allocation task. In chapter 3 reinforcement learning was described in a theoretical and abstract way. Some adaptation of concepts must be made before reinforcement learning is applicable to the channel allocation task. This entails the definition of the: state space, action space, reward function, and policy.

4.3.1 State space representation

The states are the signals to the learning agent about the environment which the learning agent uses to make decisions. The state signal must then contain the relevant and available information about the environment. The state signal is chosen to be the estimated SINR. It is assumed that the controller is able to measure the interference level of each channel. The controller also knows the received power of the desired link. It is then able to estimate the SINR of each channel under the assumption that the received power of the desired link is equal for all channels. This choice of state signal relies only on local knowledge at the controller. Therefore it satisfies the uncoordinated requirement. Other information that could be beneficial to include in the states are the positions of every other subnetwork or their channel allocation. Including such information would void the uncoordinated requirement though.

The Q-learning method is inherently discrete as, the states and actions are stored as discrete values. The SINR calculation contains multiple continuous random variables so a quantization of the SINR values is necessary. The quantization introduces two design parameters how many quantization thresholds to use and their value. The quantized threshold values are in the radix of the number of quantization thresholds plus one. The quantization function for a single quantization threshold is shown in equation (4.5).

$$S = \begin{cases} 0 & \text{SINR} < s_{th} \\ 1 & \text{SINR} \geq s_{th} \end{cases} \tag{4.5}$$

The state signal is then the collection of quantized SINR values for one for each channel used. The size of the state space then becomes

$$|\mathrm{S}| = (n_{\text{thresholds}} + 1)^m \tag{4.6}$$

Where $n_{\text{thresholds}}$ is the number of thresholds and m is the number of available channels. The size of a state space with one quantization threshold and four channels is then 16. The state space contains every possible permutation of state values from 0000 to 1111.

4.3.2 Action space representation

The action space contains all the possible actions the learning agent can take. In section 4.2 three different problems were presented. Each problem has its own unique action space.

Problem I

For problem I the actions are trivial they are simply the available channels

$$\mathbb{A} = \{1, \ldots, m\}$$

Problem II

In problem II the action space becomes the permutation of the available channels and power levels. The amount of actions is the product of the amount of power levels and channels.

$$n_{\text{actions}} = m \cdot u$$

Where u denotes the total amount of power levels available for allocation. The action set then becomes

$$\mathbb{A} = \{\{c_1, p_1\}, \ldots, \{c_m, p_u\}\}$$

Where c_n denotes the channel with index n that is available for allocation. p_n denotes the power level with index n that is available for allocation. With two power levels and four channels the amount of actions become eight. The amount and value of the power levels is a design choice. The addition of transmit power

level to the action space is made to enable the learning agent to limit its interference power.

Problem III

Problem III introduces the concept of channel aggregation. The aggregation adds a new design parameter which is how many channels to use at the same time. Choosing channels from a pool is an unordered sampling without replacement. The action space consist of the permutations from choosing one channel and up to z channels which is the maximum amount of channels to use simultaneously. The total amount of actions then become

$$n_{\text{actions}} = \sum_{j=1}^{z} \frac{m!}{j!(m-j)!} \tag{4.7}$$

Where z is the maximum number of channels to use simultaneously. Given four channels and a maximum of two channels used simultaneously the amount of actions becomes 10. The action set then contains the trivial action $(1), \ldots, (4)$ and the permutations $(1, 2)$ through $(3, 4)$.

$$\mathbb{A} = \{\{c_1\}, \ldots \{c_4\}, \{c_1, c_2\}, \ldots \{c_3, c_4\}\}$$

4.3.3 Reward function

The rewards inform the learning agent how beneficial the last action was with respect to solving the given task. It is essentially the designers possibility to define the task with respect to the learning agent. In section 4.2 the goal to maximize the cumulative capacity is stated. The reward function shall then motivate the learning agent to maximize the capacity. The most simple approach is then to equate the rewards with the capacity.

$$R_t = C_t$$

This type of reward is always positive. The learning agent then learns which actions gives the highest rewards. This reward function will be called capacity reward.

A different approach is to try to guarantee that every subnetwork achieves a target capacity. The pure capacity reward function motivates a greedy choice of actions. If the reward function is only concerned with good enough instead of best it might chose an action which is not the lowest interfered channel. This leaves a good channel open for other subnetworks to use. When each subnetwork has not

$$\begin{array}{c} \\ S_0 \\ \vdots \\ S_j \\ \vdots \\ S_{15} \end{array} \begin{array}{c} \begin{array}{cccc} A_0 & A_1 & A_2 & A_3 \end{array} \\ \begin{bmatrix} Q(0,0) & Q(0,1) & Q(0,2) & Q(0,3) \\ \vdots & \vdots & \vdots & \vdots \\ Q(j,0) & Q(j,1) & Q(j,2) & Q(j,3) \\ \vdots & \vdots & \vdots & \vdots \\ Q(15,0) & Q(15,1) & Q(15,2) & Q(15,3) \end{bmatrix} \end{array}$$

Figure 4.2: Q-table for a scenario with one quantization threshold, four channels, and problem I actions.

chosen greedily it is more probable that a reasonable channel is available for the last subnetwork to chose. This leads to a more balanced allocation. The reward function is then based on a comparison between the achieved capacity and a target capacity. The reward function then becomes

$$R_t(C_t, O) = \begin{cases} 1 & C_t \geq O \\ 0 & C_t < O \end{cases}$$

Here O denotes the target capacity. As this reward function values only assumes the binary values it will be refereed to as binary reward function.

4.3.4 Policy

A unique feature of the Q-learning method is the off policy aspect of learning an optimal policy while following a different policy. This is made possible by the selection of the best action in the update equation rather than the chosen action. Q-learning uses state-action values. As the states and actions have a finite size the Q-table will also have finite dimmensions. In practical applications this often leads to the Q-values being stored in a table with dimensions $n_{\text{states}} \times n_{\text{actions}}$. The dimensions of the Q-table with one quantization threshold, four channels, and problem I actions is 16×4. In figure 4.2 an example of this Q-table is shown.

In section 3.2 the ϵ-greedy algorithm was presented as a possible policy to use for Q-learning. The ϵ-greedy selects a random action with probability ϵ and the action with the greatest Q-value with probability $1 - \epsilon$. The exploration vs exploitation dilemma governs the choice of ϵ. Instead of choosing a fixed value for ϵ a function can be used instead [28]. By using a function which starts at a high level and decays to a lower level as time progresses, a gradual shift can

be made from favoring exploration to favoring exploitation. Such a function is called an ϵ decay function. Functions suitable for ϵ decay are functions that are bounded within the time steps the Q-learning runs for. The probability cannot exceed one or go below zero. Function prototypes that satisfies these constraints are power functions with base less than one or negative exponent. A straight line function is also applicable given that the number of time steps are known before the learning process starts. This is generally an undesired requirement. Alternatively the straight line function can be bounded by applying a maximum selection

$$\epsilon(t) = \max(\epsilon_{\text{min}}, \epsilon_{\text{max}} - t \cdot (\epsilon_{\text{max}} - \epsilon_{\text{min}})/\epsilon_{\text{steps}})$$

Where ϵ_{min} and ϵ_{max} are the minimum and maximum bounds respectively, ϵ_{steps} is the number of steps to use to go from the maximum bound to the minimum bound, and t is the time step. The choice of the ϵ bounds and number of time steps to use is a design parameter. The proof of optimality states that Q-learning will reach an optimum policy on the condition that every state and action still receives visits [28]. To satisfy this requirement the lower bound ϵ_{min} must be greater than zero.

4.3.5 Training framework

Another consideration is the amount of learning agents to use and how they relate to the environment. In section 2.4.1 the concepts of centralized, distributed, coordinated, and uncoordinated were presented in the context of interference management. The extreme requirements of In-X subnetworks necessitates a distributed and uncoordinated approach as there is no room for the overhead necessary to centralize and coordinate efforts. This implies that each subnetwork gets its own learning agent leading to a Multi-Agent Q-learning (MAQL) scenario. In MAQL for distributed resource allocation two common methods are used viz: centralized training with distributed execution, and distributed training and execution. In centralized training also called parameter sharing a single learning agent is trained with independent local measurements from each subnetwork. When the training is complete the learned parameters are copied from the centralized agent to the distributed agents for the execution phase. This is illustraited in figure 4.3. The distributed training simultaneously train a learning agent for each subnetwork. The distributed training has shown to exhibit slower convergence in the literature [36]. The centralized learning agent is then chosen for implementation. The training procedure in algorithmic form is shown in algorithm 2

in table 4.1 the state and action space is summarized together with the reward functions.

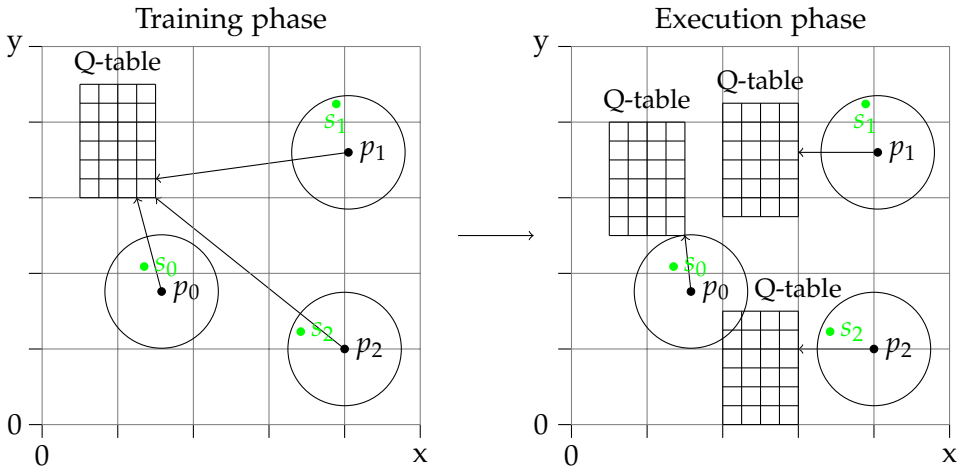

Figure 4.3: The Q-table and subnetwork relation in training and execution phase.

Algorithm 2 Training procedure

$\underline{S_0} \leftarrow$ Initial state
$\overline{\overline{Q_0}} \leftarrow \overline{\overline{0}}$
$D \leftarrow$ Number of snapshots to use
$T \leftarrow$ Number of discrete time steps to use

for i < D; i++ **do**
  $\{p_0, \ldots, p_{n_{\text{subnetworks}}}\}, \{s_0, \ldots, s_{n_{\text{subnetworks}}}\} \leftarrow$ Generate deployment
  $G_{n,n'} \leftarrow$ Generate channel gain
  for t < T; t++ **do**
    if $\epsilon(t) > \mathcal{U}_{[0,1]}$ **then**
      $A_t \leftarrow \mathcal{U}_{[1,m]}$
    else
      $A_t \leftarrow \arg\max_a Q(S_t, a)$
    end if
    Take action A_t
    Calculate SINR and capacity
    Update Q_t with update equation and R_{t+1}, S_{t+1}
  end for
end for

Table 4.1: Summary of QL design parameters. Note that problem III action space is based on the assumption $z = 2$.

Problem	I	II	II
State space	Based on estimated SINR with n-quantization thresholds		
Action space	$\{1, \ldots, m\}$	$\{\{c_0, p_0\}, \ldots, \{c_m, p_u\}\}$	$\{\{c_1\}, \ldots, \{c_m\}, \{c_1, c_2\}, \ldots, \{c_{m-1}, c_m\}\}$
Rewards	Capacity or binary based on achieved capacity and target capacity		

4.4 Summary

The system model describes how a number of subnetworks are distributed in an area. Each subnetwork contains devices that communicate with an AP at the center of the subnetwork. The communication links are affected by channel effects. It is the task of the AP to allocate the channels to use for the communication. To do the channel allocation Q-learning is used. The states are the quantized SINR value for each channel combined into a single state value. The actions are governed by three distinct problems to solve. The problems are I) channel selection II) channel and transmit power level selection and III) channel selection with aggregation. Two types of reward functions are to be considered: capacity reward and binary reward. The ϵ-greedy algorithm is to be used as the policy. A suitable decay function must be used to balance exploration and exploitation. If a guarantee on optimality is to be made the ϵ must not decay all the way to zero.

Chapter 5

Performance evaluation

In this chapter the details on the simulator implementation are presented. This includes channel models and choice of simulation parameters. This leads to a presentation of the simulation results divided into sections addressing each problem. In the end a discussion on the results is given.

5.1 Simulation overview

The simulator is an implementation of the system model described in section 4.1. The system model is implemented with an object oriented programming approach as a class in python. It uses a Monte Carlo method to generate snapshots. A snapshot is a deployment of numerous subnetworks. The simulator implements a centralized Q-learning (QL) based learning agent. The learning agent is trained for a number of time steps for each snapshot. The simulator also implements some heuristics based benchmarks algorithm for the sake of performance comparisons.

5.2 Channel models

Before a complete implementation of the system model is possible specific models must be chosen for the channel gain. In equation (4.1) the channel gain was identified as containing three parts. Path loss and shadowing are often grouped together into the term large scale fading which leaves the small scale fading.

5.2.1 Large scale fading model

The large scale fading is modelled with a close in free space reference distance model for industrial settings as proposed in [17]. The formulation is as follows

$$LS_{n,n'} = 20\log_{10}\left(\frac{4\pi f d_0}{c}\right) + 10\eta\log_{10}\left(\frac{d_{n,n'}}{d_0}\right) + G_s(p_n, s_{n'})$$

Where f is the channel center frequency, d_0 is the reference distance = 1 m, c is the speed of light = $3 \cdot 10^8$ m/s, η is the path loss exponent, and $G_s(p_n, s_{n'})$ is the shadow contribution. The shadow contribution is based on a correlated shadow map model.

The chosen shadow map model is a stationary, isotropic random gaussian field with zero mean and exponential decaying spatial correlation. To apply the model the deployment area must be quantized into sections of a reasonable size. The chosen size is 0.1 m. The covariance between two grid points in the quantized area map which is characterized by the following equation [21]

$$cov(f(p_n), f(s_{n'})) = \sigma_s^2 \exp\left(-\frac{d_{n,n'}}{\delta_d}\right) \tag{5.1}$$

Where $f(p_n)$ is the gaussian random field at position p_n. σ_s is the shadow map standard deviation, δ_d is the decorrelation distance and $d_{n,n'}$ is the distance between the subnetwork and device.

To evaluate the shadowing loss in the shadow map the following equation is used [21]

$$G_s(p_n, s_{n'}) = \frac{1 - \exp\left(-\frac{d_{n,n'}}{\delta_d}\right)}{\sqrt{2}\sqrt{1 + \exp\left(-\frac{d_{n,n'}}{\delta_d}\right)}}(f(p_n) + f(s_{n'})) \tag{5.2}$$

$G_s(p_n, s_{n'})$ is the shadow loss between p_n and $s_{n'}$, $d_{n,n'}$ is the euclidean distance between p_n and $s_{n'}$, δ is the decorrelation distance, and $f(\cdot)$ is the shadow map at the given position.

5.2.2 Small scale fading model

The small scale fading is modeled as an Independent Identically Distributed (IID) Rayleigh random variable. If Multiple-Input and Multiple-output (MIMO) techniques are used the contribution for each channel is assumed uncorrelated.

$$h_{n,n'} \sim \text{Rayleigh}(\sigma_r)$$

Where σ_r is the distribution scale parameter. The Rayleigh distribution is suitable when no Line Of Sight (LOS) exist between the transmitter and the receiver [27].

5.2.3 Noise model

The Additive White Gaussian Noise (AWGN) power is calculated with the following equation

$$w_n \cdot B = 10^{(-174+\mathrm{NF}+\log_{10}(B))/10}$$

Where NF is the noise figure of the receiver and B is the bandwidth of the used channel.

5.3 Simulation procedure

With the models defined it is possible to focus on how to apply them in a simulation. Two distinct phases characterize each simulation. The two phases are: a training phase, and a test or execution phase. In the training phase the centralized Q-table is initialized to all zeros. Then a number of snapshots are generated. A snapshot encompasses a single deployment of the subnetworks and a shadow map. The number of snapshots to use for the training phase is a design parameter. For each snapshot the Q-learning makes a channel allocation and learns from its experience. The ϵ decay function decreases for each snapshot and the Q-learning should approach the optimal policy. When the chosen number of snapshots have elapsed the simulation transitions to the test phase. The test phase also consists of a number of snapshots but the Q-learning is disabled. Instead the Q-table is only used greedily to make the channel allocation.

Each snapshot is subdivided into a number of time steps. The time steps relate to the Q-learning procedure as described in section 3.2. To avoid that every subnetwork tries to update the centralized Q-table at the same time each subnetwork is given a certain time step where they can update the Q-table. The update time step for each subnetwork is found at random and it is constant over all time steps for each snapshots.

5.4 Result types

The two simulation phases lead to two different kinds of results. In the training phase it is relevant to verify whether any thing is actually learned. To do this the reward values associated with Q-table updates over every snapshot is saved. The reward values for each subnetwork is then averaged to produce a single value for each snapshot. The reward values for each snapshot is then averaged with a moving average filter with a rectangular window of size 50 and plotted. If anything has been learned the reward values should grow as a function of snapshots.

The results of the testing phase are Cumulative Distribution Function (CDF) plots of the achieved Signal to Interference plus Noise Ratio (SINR) and capacity. As no update of the Q-table takes place the result data is taken after the warmup time. The warmup time covers the time duration it takes for each subnetwork to stabilize in a channel allocation. When a new deployment is made a new channel allocation is necessary. The result data is taken for each subnetwork at the last time step of each snapshot.

5.5 Benchmark algorithms

To evaluate the performance of Q-learning three benchmark algorithms are used. The three benchmark algorithms are:

- Random allocation
- Greedy allocation
- Centralized Graph Coloring allocation

5.5.1 Random allocation

The random allocation algorithm allocate channels based on samples of a uniformly distributed random variable

$$K \sim \mathcal{U}\{0, m\}$$

The distribution for each subnetwork is IID. When the distributions are independent there is no way for the random variable to account for the channel allocation of other subnetworks. It is therefore distributed and uncoordinated.

5.5.2 Greedy allocation

The greedy allocation is based on the calculated SINR values of each channel for the previous time step. The channel allocation choses the channel with the greatest SINR value.

$$k_n = \max_{m_i}(\text{SINR}_n^{m_i})$$

The algorithm always choses greedily for each subnetwork. It does not take the channel allocation of other subnetworks in to consideration. It is then distributed and uncoordinated.

5.5.3 Centralized Graph Coloring allocation

A third possible allocation algorithm is Centralized Graph Coloring (CGC) based. The CGC algorithm breaks the distributed requirement of in-X subnetworks. The CGC algorithm then enables the possibility of comparing distributed and centralized channel allocation. The CGC algorithm draws a conflict graph where each subnetwork is a vertex. Each vertex is connected to every other vertex with an edge, except for the vertex with the least mutual link power. A graph coloring algorithm is then used to color each vertex with a unique color for each vertex that share an edge [34]. In this case the colors represent channels. The maximum available colors are the number of available channels. If it is not possible to color the conflict graph with the available colors the conflict graph is redrawn but now with the two weakest link powers unconnected. The coloring is then attempted again. These iterations continue until a coloring of the conflict graph is achieved.

5.6 Choice of simulation parameters

A large slew of parameters have been implemented to make the simulations realistic. The simulation parameters can be subdivided into two groups: those relating to the subnetwork environment, and those relating to the Q-learning. Those two groups can be further subdivided into: those that remain constant for all results, and those that change between simulations. Those that remain constant will be presented now and those that change will be accounted for in the presentation of the simulations results.

The constant parameters for the subnetwork environment is summarized in table 5.1. The choice of the first six parameters from the top is based on an initial sensitivity analysis. These parameters affect the subnetwork density and the number of available channels which in turn affects the SINR and capacity. The exact values of SINR and capacity is not important with regard to the results. What is important is the comparison to the benchmark algorithms. The remaining parameters relates to the radio channel. These parameters must be based on real observed statistics to be applicable. The path loss exponent is reported to be in between 2.40 and 2.94 in [9]. The shadowing standard deviation is reported in [3] as 5.7 dB. The decorrelation distance describes the distance necessary for the shadowing to become uncorrelated. This in turn describe the size of the objects that lead to the shadowing. In industrial settings these objects might be large machinery. In [3] a deecorrelation distance of 10 m was used. The receiver noise figure is also reported in [3] to be 10 dB.

Parameter name	Value	unit
Factory area [$\mathcal{R}$]	[40, 40]	m
Number of subnetworks [$n_{\text{subnetworks}}$]	30	.
Number of channels [m]	4	.
Number of receive antennas	2	.
Minimum subnetwork separation distance	3	m
Bandwidth [B]	5.00	MHz
Channel group size [z]	2	.
Path loss exponent [η]	2.55	.
Shadowing standard deviation [σ_s]	5.7	dB
Decorrelation distance [δ]	10	m
Noise figure [NF]	10	dB
Rayleigh scale [σ_r]	1	.

Table 5.1: The constant parameters relating to the subnetwork environment.

Parameter name	Value
Discount rate [γ]	0.9
Learning rate [α]	0.8
ϵ lower bound [ϵ_{min}]	0
ϵ upper bound [ϵ_{max}]	1
snapshots from upper to lower bound [ϵ_{steps}]	4000
Number of time steps	60
Number of snapshots	6000
Max update delay	30

Table 5.2: The constant parameters for the Q-learning.

The constant parameters for the Q-learning is shown in table 5.2. These values are chosen based on the initial sensitivity analysis. These parameters show rapid learning and saturation of the reward results indicating no more learning is possible. The ratio between number of time steps and max update delay indicates that each subnetwork chooses action twice for one snapshot and makes two updates to the Q-table.

5.7 Simulation results

The presentation of the simulation results is divided into five parts. First the choice of the parameters to analyze is presented. Then some results of the training phase are presented to stipulate that learning happens. The last three parts are based on each problem as introduced in subsection 4.3.2. For each problem the relevant

parameters to analyze are presented along with the actual results and observations.

5.7.1 Initial simulation parameters

The four parameters to analyze are: reward function, number of state quantization thresholds, values of the state quantization thresholds, and target capacities. Some initial analysis must be made to derive which values to use. Calculating the expected SINR and capacity values is very difficult as the calculations contain multiple dependent random variables and the expected values depend on the values not yet chosen. Another approach is to use data derived with the random allocation of the benchmark algorithms. The random allocation does not depend on any of the unknown parameters. It does however operate in the same environment as Q-learning. The random allocation can then be used as a baseline for the choice of the unknown parameters. The analysis with the random allocation can be split into three parts

1. Run a simulation with only random allocation and collect SINR and capacity values
2. Generate CDF plots of the collected data
3. Choose quantization threshold and target capacity values based on the CDF percentiles

This procedure does not specify how many state quantization thresholds to use. The parameter influences the number of states as expressed in equation (4.6). This equation shows that for one state quantization threshold the amount of states is 16. If more state quantization thresholds were to be used the amount of states grow to the power of four. So for two thresholds 64 states results and for three it is 256. The large growth in states is detrimental to the Q-learning training time. Many states lead to a large Q-table. The training must then run for a longer time to visit every state-action combination. A single state quantization threshold is chosen as an initial value to keep training time low. The value of the threshold is chosen to be the 50% percentile of the SINR CDF the value is 12.71 dB. This choice assumes that the possible SINR values for the two types of allocation are similar. The target capacity is chosen based on the capacity CDF of random allocation. As the purpose of the target capacity is to guarantee a minimum performance the value is chosen at the low end of the capacity CDF. The chosen percentile is 10% which corresponds to a capacity of 10 Mbps. The reward functions serve two different purposes so both should be tried.

	Value
Tx power	-10 [dBm]
Threshold	12.71 [dB]
Reward type	Capacity
Aggregation	No

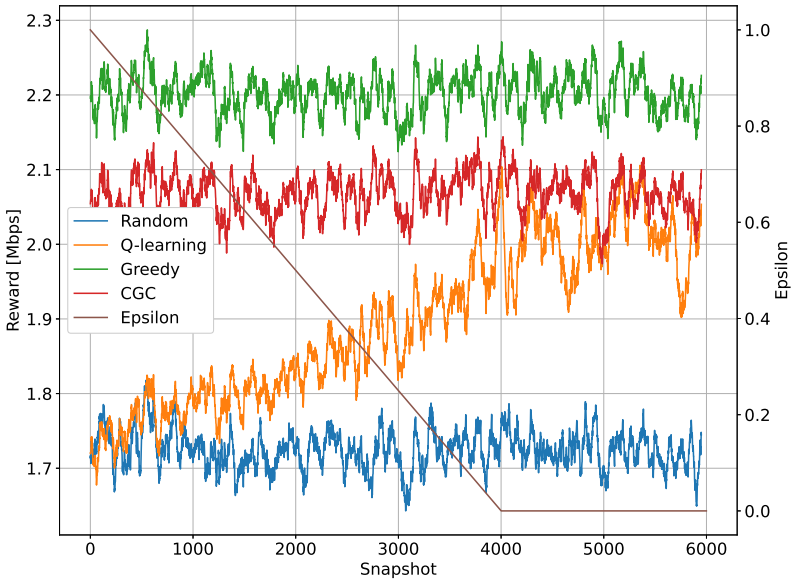

Figure 5.1: The training result for problem one with capacity reward and the initial choice of parameters.

5.7.2 Initial training results

In figure 5.1 and 5.2 the training results for the two reward functions are shown. Both figures show signs of learning because the Q-learning traces increase over snapshots. Both traces also show signs of saturation when ϵ reaches 0. The saturation shows that no improvement will happen with an increase of snapshots.The two traces differ with respect to the benchmark algorithms. The capacity reward reaches similar performance to the CGC algorithm. The binary reward only slightly improves compared to random allocation. It is worth noting that the CGC and greedy algorithms seek to maximize SINR which in turn also maximizes the capacity. The goal of the binary reward function is that every subnetwork should reach a target and not the largest cumulative capacity. The two reward functions have different units and are not directly comparable. The difference between the two rewards with respect to the benchmark is then a matter of scale. To better assess the performance of the different reward functions the test phase results should be used. From this point on only test results will be used. The training results for every simulation is shown in appendix A, when it contains relevant information it will be referenced.

	Value
Tx power	-10 [dBm]
Threshold	12.71 [dB]
Reward type	Binary
Target capacity	10 [Mbps]
Aggregation	No

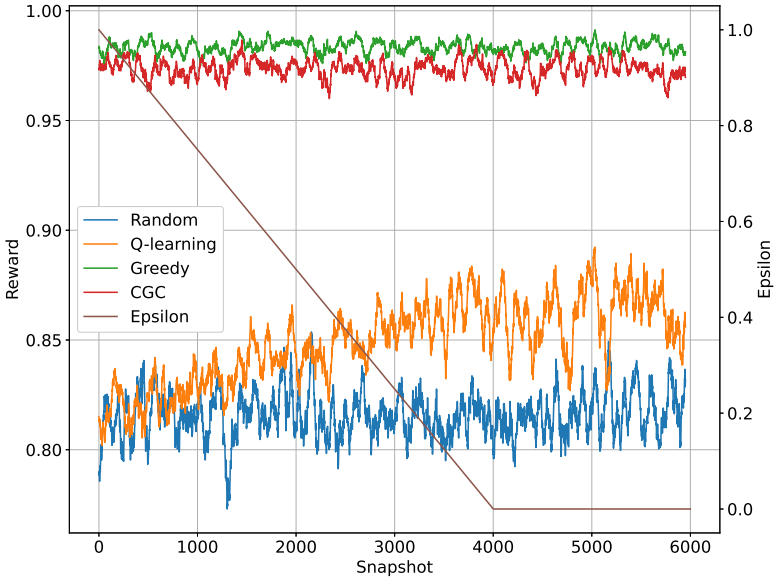

Figure 5.2: The training result for problem one with binary reward and the initial choice of parameters.

5.7.3 Test phase results for problem I

In the test phase results it is possible to compare the two reward functions. As the binary reward function depends on the chosen target capacity multiple values can be compared. This enables the possibility to verify that the target capacity works as intended. The chosen target capacities are based on the capacity CDF under random allocation. The chosen percentiles are 8%, 10% and 25% the corresponding capacity values are 8, 10, and 15 Mbps. The comparison results are shown in figure 5.3. The expected trace shape for the binary reward is low and flat from zero until the target capacity is reached. When the target capacity is reached a large growth is expected. This is however not what is observed. Every reward function type seems to perform very similarly. In figure 5.4 a zoomed view is given of the low end of the results. In this figure it is clear that no major change happens when the target capacities are reached. The binary reward does not work as intended. The two best performing reward functions are the capacity and binary with target capacity of 15 Mbps. The capacity reward function has been selected, for further use, as it has been observed that the simulation run time is lower with the capacity reward function.

The binary reward function turns out to be robust to change in target capacity. From a system perspective it can be beneficial to choose the binary reward to minimize overhead in Reinforcement Learning (RL) feedback. This happens when

	Value
Tx power	-10 [dBm]
Threshold	12.71 [dB]
Reward type	Capacity and binary
Target capacities	8, 10, 15 [Mbps]
Aggregation	No

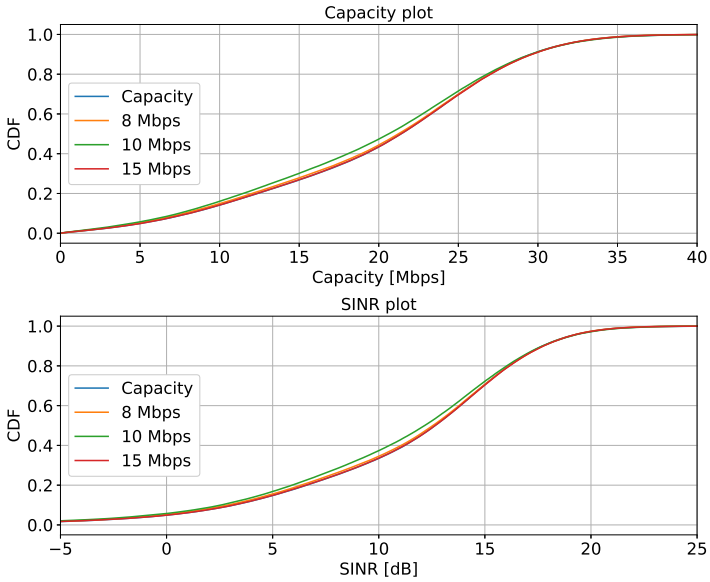

Figure 5.3: The reward function comparison results in the test phase.

	Value
Tx power	-10 [dBm]
Threshold	12.71 [dB]
Reward type	Capacity and binary
Target capacities	8, 10, 15 [Mbps]
Aggregation	No

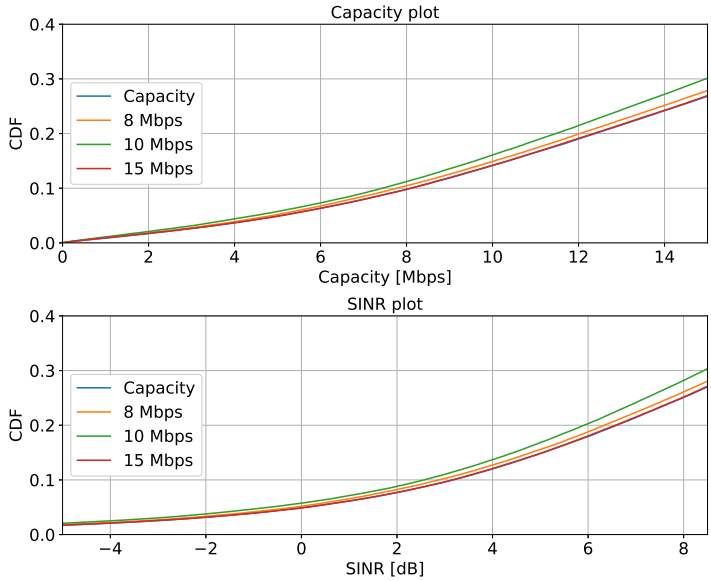

Figure 5.4: The reward function comparison results focused on the low end in the test phase.

the Q-table is placed at the device and the Access Point (AP) sends feedback on the reward. It can also happen when the Q-table is placed at the AP and relies of feedback on the reward for the Down Link (DL). As the binary reward is a single bit it is much smaller than the capacity reward which is a real value typically represented as a double data type. Minimizing the feedback data size improves reliability and latency. The feedback overhead is not implemented in the simulation. The consideration on feedback data size should not govern the choice of reward function then. The consideration is made to exemplify what could govern the choice in a real implementation.

The state quantization threshold value is another avenue to explore. By shifting the threshold value from the initial choice of 50% of the random allocation CDF towards lower SINR values it might be possible to boost low end performance. By shifting the quantization threshold value more emphasis is put on the channels that perform very poorly. Four threshold values are chosen based on the random allocation SINR CDF. The four values are the 50%, 10%, 5%, and 1% which corresponds to 12.71 dB, 4.60 dB, 1.37 dB, and -6.73 dB. The different threshold value comparison is shown in figure 5.5. The desired effect of the lower threshold value leading to better performance is observed for the 4.60 dB trace compared to the 12.71 dB trace. The results also show that there is a limit to how low the threshold value can go. The two lowest values 1.37 dB and -6.73 dB performs very poorly. The bad performance is also evident in the training data where the Q-learning traces declines when $\epsilon = 0$ this is shown in figure A.2 in the appendix.

The reason why the two lowest threshold values performs poorly could be due to an over representation of only a few states. This happens when the threshold value becomes so low that the channel SINR value is almost always quantized as performing well. To assess whether this happens in this case the state histogram for each threshold has been plotted in figure 5.6. The state histograms show that the zero state is the most prevalent for the 50% threshold value. The zero state is the one where every channel performs poorly. For the 10% threshold value it is the opposite state that is most prevalent. It is the state where every channel performs well. For the two lowest threshold values only very few states are visited. Especially for the 1% threshold. Almost no other state than the best is visited. Limiting the amount of states also limits the possible actions. In the test phase the action is always chosen greedily from the Q-table. If the state is always the same so is the chosen action. If every subnetwork choses the same action the corresponding channel becomes heavily interfered.

The knowledge about state distribution leads to a new opportunity for how to choose the threshold values. Heavily skewed state distribution leads to poor per-

	Value
Tx power	-10 [dBm]
Threshold	12.71, 4.60, 1.37, -6.73 [dB]
Reward type	Capacity
Aggregation	No

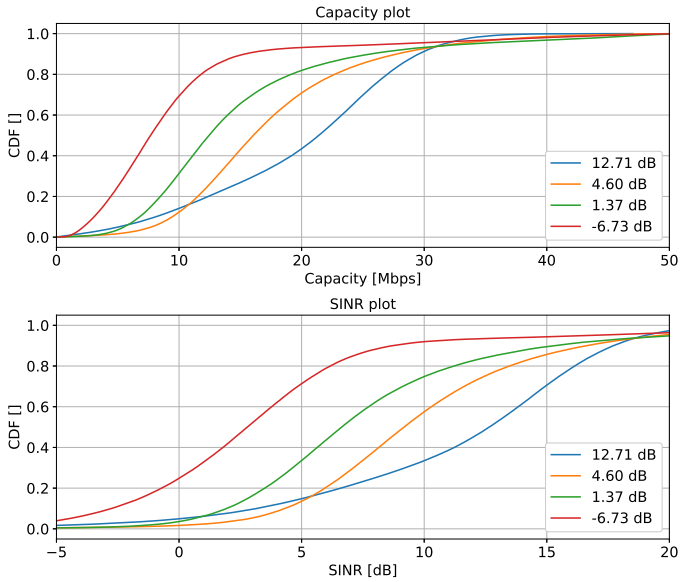

Figure 5.5: The quantization threshold values comparison in the test phase.

	Value
Tx power	-10 [dBm]
Threshold	12.71, 4.60, 1.37, -6.73 [dB]
Reward type	Capacity
Aggregation	No

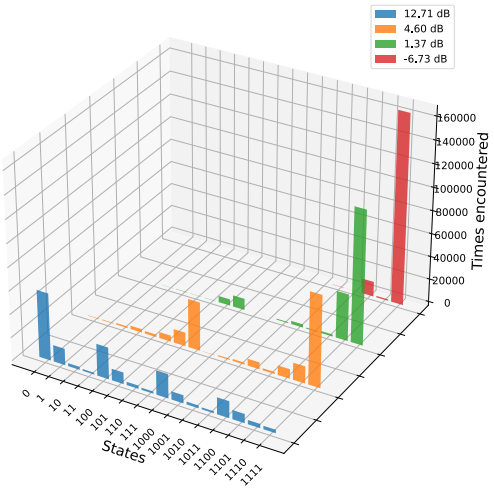

Figure 5.6: The state histograms for each quantization threshold.

	Value
Tx power	-10 [dBm]
Threshold	12.71, 8.66, 4.60 [dB]
Reward type	Capacity
Aggregation	No

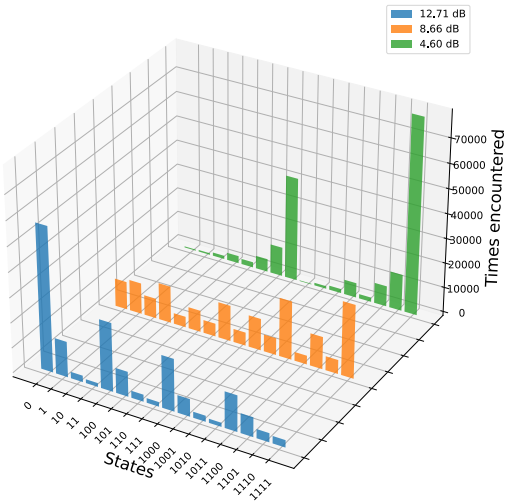

Figure 5.7: The state histogram of the 8.66 dB threshold compared to 12.71 dB and 4.60 dB.

formance. A more uniform state distribution leads to action diversity. A theoretical state distribution can be calculated under the assumption that every channel is independent. This assumption does not hold true. The performance of one channel is heavily dependent on the other channels because of the channel allocation.The theoretical state distribution can serve as starting point to understand the significance of the state quantization threshold value. The theoretical probability of each state is

$$\Pr(s) = p^i(1-p)^{m-i}$$

Where $\Pr(s)$ is the probability of the state s, p is the probability that the channel performs well, i is the number of channels that performs well, and m is the number of channels to use. The theoretical uniformly state distribution occurs when $p = 0.5$. It is difficult to calculate which threshold value that corresponds to this probability. Previously the random allocation was used to derive the threshold value. The values from the random allocation cannot be directly applied to the RL allocation. Instead a threshold value in between the 12.71 dB and 4.60 dB was tried. The median between the two is 8.66 dB. In figure 5.7 the state distribution with threshold value 8.66 dB is compared to the 12.71 dB and 4.60 dB thresholds. The 8.66 dB threshold value leads to a more uniform distribution.

The 4.60 dB threshold showed good performance in the low end and the 8.66 dB threshold showed good state distribution. To gauge the overall performance

	Value
Tx power	-10 [dBm]
Threshold	8.66, 4.60 [dB]
Reward type	Capacity
Aggregation	No

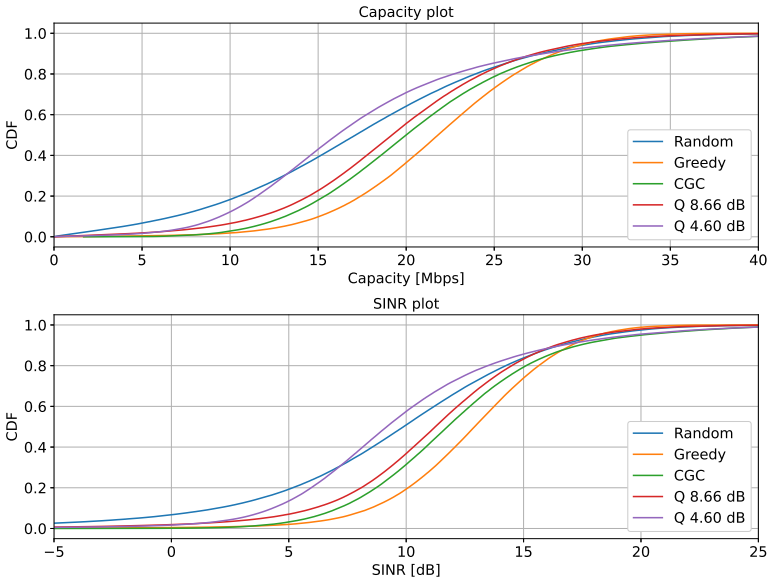

Figure 5.8: The two threshold values compared to the benchmark algorithms.

of these threshold they should be compared to the benchmark algorithms. This is done in figure 5.8. The 8.66 dB threshold value performs very similarly to the CGC algorithm but slightly worse. It also performs better than the random allocation from 0 to 26 Mbps for the capacity and -5 to 16 dB for the SINR. It is difficult to assess performance in the low end from the given figure. Figure 5.9 contains the same data but focused on the low end and in log scale. The 4.66 dB threshold is better than the 8.66 dB threshold from 0 to 6 Mbps for the capacity and from -5 to 1 dB for the SINR. Both threshold values out perform the random allocation. The best performance is observed for the CGC allocation from 0 to 8 Mbps with respect to capacity and -5 to 3 dB with respect to the SINR.

5.7.4 Results of problem II

Problem II introduces the transmit power level to the action space. To compare any possible change in performance from this addition the parameters used for problem I is reused. Specifically the 8.66 dB state quantization threshold value and the capacity reward function. Three power levels will be used for the simulation: 10, 0, and -10 dBm. The values are chosen based on the short coverage range necessary for an In-X subnetwork but still with a large enough difference to give significant changes in the results. The test phase results of problem II is shown in figure 5.10 here the variable power selection is compared to the fixed power of -10 dBm for problem I. The performance of variable power compared to fixed

	Value
Tx power	-10 [dBm]
Threshold	8.66, 4.60 [dB]
Reward type	Capacity
Aggregation	No

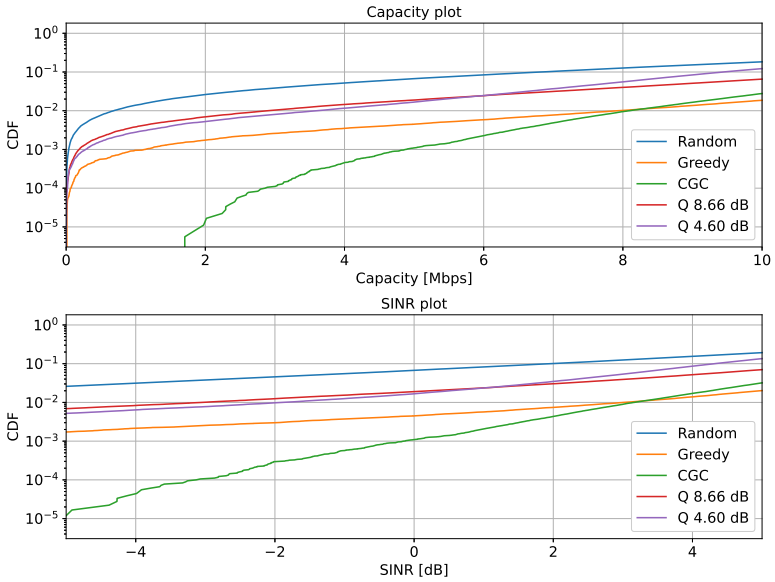

Figure 5.9: The two threshold values compared to the benchmark algorithms focused on low end in log scale.

is very similar. It is difficult to discern the difference. A possible explanation could be that the low power levels are not used. In 5.11 the action histogram for problem II in the test phase is shown. The histogram shows that the low power actions are used. Another observation to make is that two actions are unused and the zero channel with power 10 dBm is very frequently used. In figure 5.10 the greedy algorithm performs well with a fixed power level. The action histogram for the greedy algorithm is shown in figure 5.12. The greedy algorithm has a uniform action distribution. This is very different from the Q-learning. The chosen action depends on the state. The state quantization threshold value was chosen to uniformly distribute the states. In states where two or more channels perform well the is a randomness to the action selection. This randomness is impossible to account for through design parameters. The fact that the reduced transmit power level leads to similar performance as to more powerful levels can be utilized in use cases where power consumption is a critical parameter such as implantable in-body subnetworks.

5.7.5 Results of problem III

In problem III the channel aggregation is added to the action space. The group size is chosen to be two. That is a maximum of two channels can be used simultaneously. The addition of aggregation to the action space might change the state distribution. As the states depend on the chosen actions it is not possible to reuse

	Value
Tx power	10, 0, -10 [dBm]
Threshold	8.66 [dB]
Reward type	Capacity
Aggregation	No

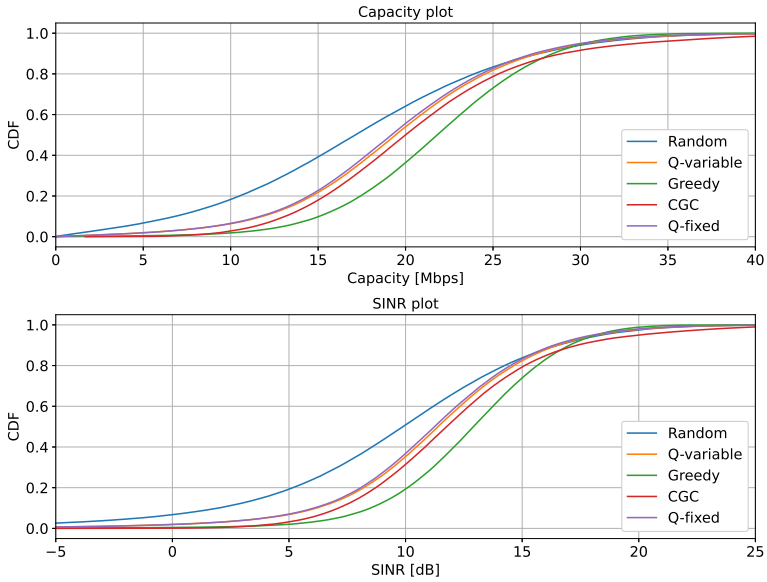

Figure 5.10: Test phase result of problem II compared to fixed power.

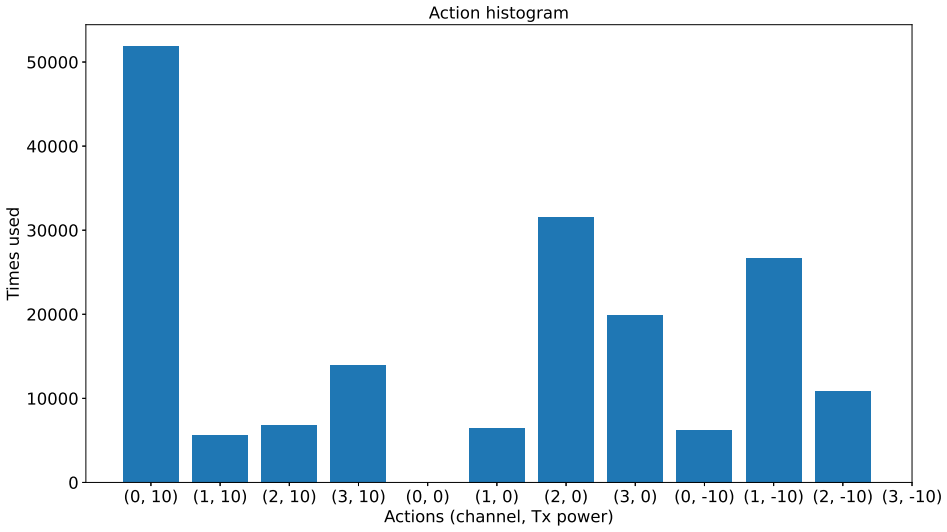

Figure 5.11: Test phase action histogram for problem II.

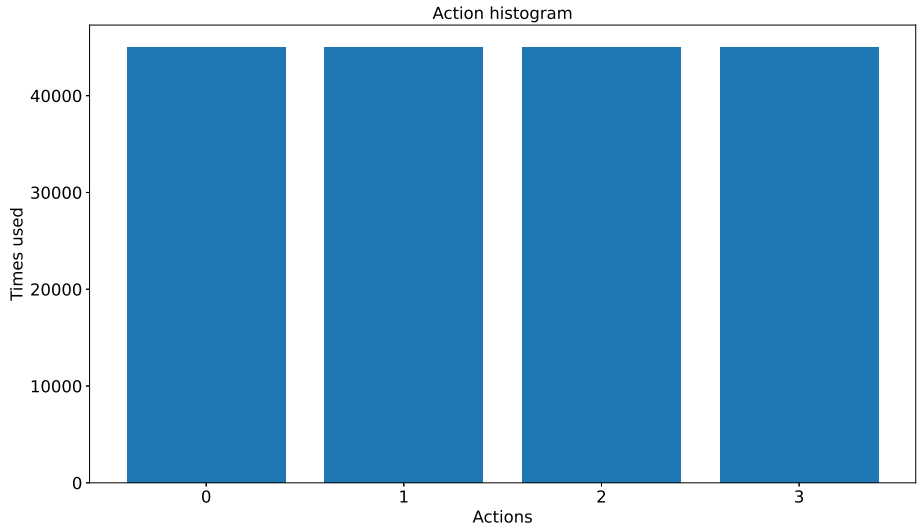

Figure 5.12: Test phase action histogram for the greedy algorithm.

the threshold values of previous results. In figure 5.13 the different state quantization threshold values of problem I are tried. The results show that the 4.60 dB threshold value performs best. In the very low end lowest threshold value performs the best. Two threshold values are then chosen to compare to the benchmark algorithms. The values are 4.60 and -6.73 dB. The comparison is made in figure 5.14. The CGC algorithm is not easily adaptable to include aggregation and it has been left out. The greedy algorithm is adapted to choose the two channels with the greatest SINR value. The random algorithm is also adapted and chooses channel pairs at random. The results only contain capacity data. Combining multiple link capacities is simply the sum of the capacities for each link. The same is not true for SINR values. The results show that the 4.60 dB threshold value performs slightly better than random allocation. The -6.73 dB threshold value is best in the low end but only up to 1 Mbps. These results correlate with the results given in subsection 5.7.3 with regard to changes in state quantization threshold values and in comparison to the benchmark algorithms. The addition of channel aggregation has then not fundamentally changed the possible results and the leaning agents' task must be similar.

5.8 Discussion

A sensitivity analysis is made for three parameters: state quantization threshold values, binary reward target capacities, and reward functions. Initial choices of

	Value
Tx power	-10 [dBm]
Threshold	12.71, 8.66, 4.60, 1.37, -6.73 [dB]
Reward type	Capacity
Aggregation	Yes
Group size	2

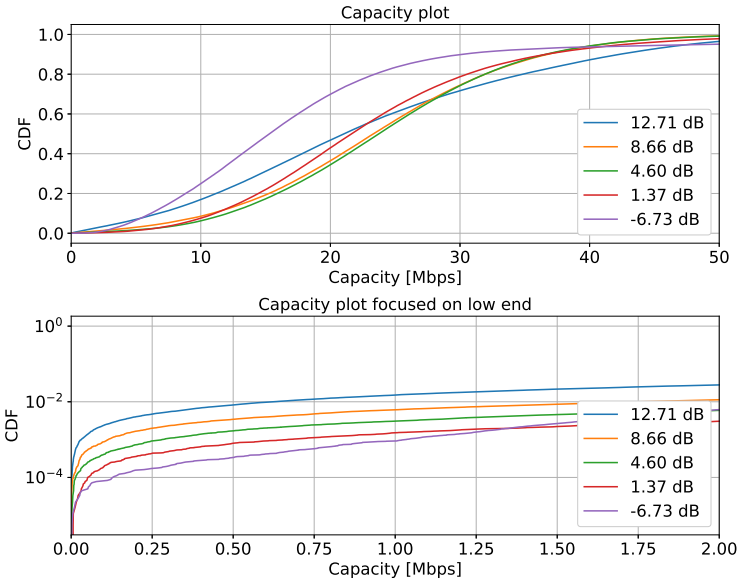

Figure 5.13: Test phase comparison of different quantization threshold values for problem III

	Value
Tx power	-10 [dBm]
Threshold	4.60, -6.73 [dB]
Reward type	Capacity
Aggregation	Yes
Group size	2

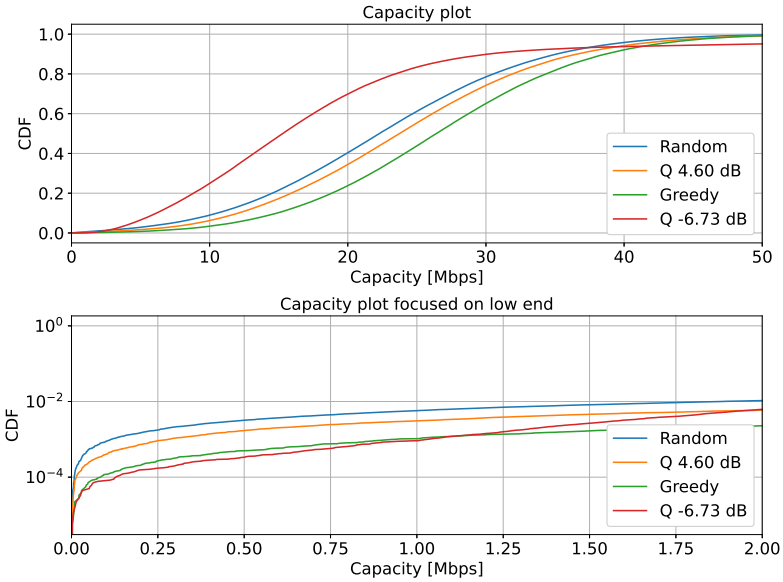

Figure 5.14: Test phase comparison of the Q-learning and benchmark algorithms for problem III

threshold values and target capacities are based on the performance of the random allocation algorithm. The initial parameter choices are made under the assumption that the random allocation performs similarly to Q-learning. Training results with the initial parameter choices show that learning does take place due to the growing rewards with respect to snapshots.

The sensitivity analysis of the reward functions and target capacities were combined into one. The analysis shows that the performance is insensitive to changes in reward function and target capacities. This shows that the learning agent is not able to shape its performance through its actions. The binary reward function only rewards those actions that have led to an exceedance of the target capacity. These actions are the same that score high with the capacity reward function. This leads to similar behaviour when the Q-table is used greedily for action selection. The expectation that the binary reward function should be able to limit results below the target capacity is flawed. The expectation is based on the assumption that previous experience can be generalized for each deployment. That is a certain action will always exceed target capacity. Due to changes in environment or the actions of other learning agents no such guarantee can be made.

In [35] it was recognized that using multiple learning agents in the same environment lead to a non-stationary environment from each learning agents' local perspective. The environment becomes non-stationary when a learning agent cannot infer whether the state transition and reward happens due to its own action or the actions of other learning agents. Some provisions are made to address this issue by randomizing at which time step each subnetwork makes updates to the Q-table. The chosen randomizing method does not guarantee that multiple subnetworks does not make concurrent updates.

The analysis of the state quantization threshold values show that the performance can be modified by changing these parameters. There is a limit though. If the state distributions become too skewed in either direction the performance suffer due to the lack of diversity in action selection. The amount and values of the state quantization thresholds should then be chosen to avoid a heavily skewed state distribution. A strategy to improve low end performance could be to add multiple thresholds with low values to motivate state and action diversity in the low end. The problem of state space size and training time does limit what is feasible.

The inclusion of transmit power offers no discernible difference in performance. In section 2.3 Power Control (PC) was presented as a method that depends on every subnetwork using it. No cooperation happens between subnetworks which

enables a collective decision to reduce transmit power. The reduced power actions do see usage though. A possible benefit is then that the reduced power level can be used in application where power consumption is critical without loss in performance.The action distribution shows that the action selection is not uniform. The greedy benchmark algorithm shows that a uniform distribution is desired. The action distribution for QL in a purely exploitation state depends on the learned experience. In a problem I scenario a single channel must be chosen. In states where two or more channels are good the chosen channel depends on which of the two channels has performed best previously. There is some degree of randomness to which channel is the better. Past shadow maps, deployments, small scale fading etc coupled with the problem of non-stationarity leads to a non uniform action distribution. For instance in the used state space for the simulation 16 different state values are possible. The state values can be represented with four digits of binary value where each digit represents a channel. The first state is then 0000 where every channel is bad. The states with values 3, 5, 7, 9, 11, 13, and 15 have the right most channel performing well. Then that exact channel may be used up to seven times instead of four times if everything was balanced. There is no modifiable simulation parameter that directly affects the action distribution.

The addition of channel aggregation does not fundamentally change the performance of Q-learning compared to the benchmark algorithms. The extra channel does lead to an increase in capacity but otherwise the performance is similar to single channel selection.

5.9 Summary

In this chapter the details on the simulator is presented. This includes chosen channel models and a genereal characterization of its functionality. Choices of the simulation parameters are argued for and initial results show that the learning agent is able to learn. Results are presented for each defined problem. It is observed that the capacity and binary reward with different target capacities performs similarly. It turns out that Multi-Agent Q-learning (MAQL) suffers from problems of non-stationary environment. The choice of state quantization threshold value is able to shape the result within some limits. Too large or small threshold value leads to very few states being visited. This in turn leads to poor action diversity. The addition of transmit power level to the action space offers no discernible difference in results. It is apparent that a uniform state distribution does not lead to a uniform action distribution. Results with reduced transmit power level do show that a reduced transmit power level can be used for power consumption critical applications. Adding channel aggregation to the channel allocation task leads to a

boost in achieved capacity but the performance is otherwise similar to the single channel selection.

Chapter 6

Conclusion and future work

This master thesis has studied interference management for dense subnetworks. The motivation to treat this subject is In-X subnetworks. In-X subnetworks has been proposed as a solution to handle the extreme demands of the Sixth Generation (6G) of radio access technology. Such demands could be data rates in excess of 5 Gbps and latencies less than 100 μs. These requirements demand large amounts of radio resources such as frequency. If every User Equipment (UE) with extreme demands should be handled at a macro level there simply would not be enough resources for everyone to have their demands met. The In-X subnetworks are to be deployed in an uncoordinated fashion where the extreme demands arises. The In-X subnetworks are to operate at a low power level for a limited coverage area. The uncoordinated deployment enables the issue of interference.

The prerequisites for an interference event to happen are overlaps in resource use for multiple radios. Three domains were identified where an overlap must happen between two or more radios for interference to take place. The three domains are: spatial, time and frequency. Each domain also offers techniques to mitigate interference. The frequency domain was identified as the most suitable for interference management in In-X subnetworks. Interference management in the frequency domain entails channel allocation. A distributed and uncoordinated channel allocation is appropriate for In-X subnetworks due to the extreme demands. Distributed and uncoordinated channel allocation method encompasses those based on heuristics and those based on Machine Learning (ML) methods. A Reinforcement Learning (RL) based method is chosen for further analysis in the In-X subnetworks context.

The RL method was described and Markov Decision Processes (MDPs) serve as the framework for the mathematical analysis of RL. The MDP contains a learning agent that interacts with an environment. This interactions happens through

three signals: states, actions, and rewards. The design of these three parameters define the goal of the learning agent. The learning agent uses value functions to summarise the benefit of different choices. The value function is learned through the interaction. If the learning agent initially explore all possible choices it will gather experience. Gradually the focus can be shifted from exploring every possible choice to focus on the choice with the greatest value. This is the exploration - exploitation trade-off. The trade-off is governed by the policy. Q-learning is a type of Temporal Difference (TD) learning. Q-learning is based on estimates of the state-action value function. The state-action values are stored in a Q-table. By using a ϵ-greedy policy the focus of the learning agent is gradually shifted from exploration to exploitation.

A system model of an In-X subnetwork deployment was given. The system model is based on an identified use case of In-X subnetworks. Performance metrics are derived and based on those, the channel allocation goal is defined. With a clear defined goal the design of the state, action and reward signals can be made to be used with the Q-learning method. Three separate problems are identified to be solved.

A detailed description of the implemented simulator is presented. A characterization of the simulation results is given. A number of benchmark algorithms are introduced to compare to Q-learning. Reasons and citations are argued for the choice of simulation parameters.

The simulation results show that learning does take place. The simulator is insensitive to variations in target capacity and reward function. The state quantization threshold values do affect the the performance but a too skewed state distribution should be avoided. Reducing the transmit power shows that a reduction can be applied which benefits power consumption critical applications. Common for all problems is the issue of the non-stationary environment and skewed action distributions.

6.1 Future work

Several avenues lends themselves to further exploration. A simple modification to the current code base can be made to address the non-stationarity issue. By changing the random update time step for each subnetwork to avoid concurrent updates. Another approach is to keep concurrent updates but modifying the Q-learning algorithm to account for the non-stationarity problem. In [23] Concurrent Experience Replay Trajectories (CERT) is proposed to address the non-stationarity issue. In [33] CERT is used for spectrum access in a vehicular network setting. The

achieved performance is similar to centralized "brute-force" method [33].

The results show that state quantization threshold values affect performance but suffers from quickly growing state space with the addition of more thresholds. To avoid this limitation deep Q-learning has been proposed [29]. In deep Q-learning the Q-table is replaced with a Deep Neural Network (DNN). The DNN can be designed with the number of input nodes equal to the number of channels to use. Each input node can take a continous input and no quantization is necesasrcy. The output layer has a node for each action. The output is then the Q-value for each action. The action selection is then still governed by the ϵ-greedy policy.

Bibliography

[1] Ramoni Adeogun, Gilberto Berardinelli, and Preben E. Mogensen. "Enhanced Interference Management for 6G in-X Subnetworks". In: *IEEE Access* 10 (2022), pp. 45784–45798. DOI: 10.1109/ACCESS.2022.3170694.

[2] Ramoni Adeogun et al. "Distributed Dynamic Channel Allocation in 6G in-X Subnetworks for Industrial Automation". English. In: *2020 IEEE Globecom Workshops, GC Wkshps 2020 - Proceedings*. GLOBECOM 2020 - 2020 IEEE Global Communications Conference ; Conference date: 07-12-2020 Through 11-12-2020. United States: IEEE, 2020. ISBN: 978-1-7281-7308-5. DOI: 10.1109/GCWkshps50303.2020.9367532.

[3] Ramoni Ojekunle Adeogun, Gilberto Berardinelli, and Preben E. Mogensen. "Learning to Dynamically Allocate Radio Resources in Mobile 6G in-X Subnetworks". English. In: *2021 IEEE 32nd Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*. I E E E International Symposium Personal, Indoor and Mobile Radio Communications. 2021 IEEE 32nd Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC) ; Conference date: 13-09-2021 Through 16-09-2021. United States: IEEE, Sept. 2021, pp. 959–965. ISBN: 978-1-7281-7587-4. DOI: 10.1109/PIMRC50174.2021.9569345.

[4] Tauseef Ahmed, Faisal Ahmed, and Yannick Le Moullec. "Optimization of channel allocation in wireless body area networks by means of reinforcement learning". In: *2016 IEEE Asia Pacific Conference on Wireless and Mobile (APWiMob)*. 2016, pp. 120–123. DOI: 10.1109/APWiMob.2016.7811445.

[5] *Beautiful data : the stories behind elegant data solutions.* eng. Beijing, 2009.

[6] Gilberto Berardinelli, Preben Mogensen, and Ramoni O. Adeogun. "6G subnetworks for Life-Critical Communication". In: *2020 2nd 6G Wireless Summit (6G SUMMIT)*. 2020, pp. 1–5. DOI: 10.1109/6GSUMMIT49458.2020.9083877.

[7] Gilberto Berardinelli et al. "Extreme Communication in 6G: Vision and Challenges for 'in-X' Subnetworks". English. In: *IEEE Open Journal of the Communications Society* 2 (2021), pp. 2516–2535. ISSN: 2644-125X. DOI: 10.1109/OJCOMS.2021.3121530.

[8] Gustavo W. O. da Costa et al. "A Scalable Spectrum-Sharing Mechanism for Local Area Network Deployment". In: *IEEE Transactions on Vehicular Technology* 59.4 (2010), pp. 1630–1645. DOI: 10.1109/TVT.2009.2039361.

[9] Ramona Croonenbroeck et al. "Measurements for the development of an enhanced model for wireless channels in industrial environments". In: *2017 IEEE 13th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. 2017, pp. 1–8. DOI: 10.1109/WiMOB.2017.8115764.

[10] Natasha Devroye, Mai Vu, and Vahid Tarokh. "Cognitive radio networks". In: *IEEE Signal Processing Magazine* 25.6 (2008), pp. 12–23. DOI: 10.1109/MSP.2008.929286.

[11] Eryk Dutkiewicz et al. "Massive Machine-Type Communications". In: *IEEE Network* 31.6 (2017), pp. 6–7. DOI: 10.1109/MNET.2017.8120237.

[12] Jan Ellenbeck, Christian Hartmann, and Lars Berlemann. "Decentralized inter-cell interference coordination by autonomous spectral reuse decisions". In: *2008 14th European Wireless Conference*. 2008, pp. 1–7. DOI: 10.1109/EW.2008.4623915.

[13] Luis G. U. Garcia et al. "Self-Organizing Coalitions for Conflict Evaluation and Resolution in Femtocells". In: *2010 IEEE Global Telecommunications Conference GLOBECOM 2010*. 2010, pp. 1–6. DOI: 10.1109/GLOCOM.2010.5683438.

[14] Andrea Goldsmith. *Wireless Communications*. eng. Cambridge: Cambridge University Press, 2005. ISBN: 0521837162.

[15] Martin Hilbert and Priscila Lopez. "The World's Technological Capacity to Store, Communicate, and Compute Information". eng. In: *Science (American Association for the Advancement of Science)* 332.6025 (2011), pp. 60–65. ISSN: 0036-8075.

[16] Guangyu Jia et al. "Channel Assignment in Uplink Wireless Communication Using Machine Learning Approach". In: *IEEE Communications Letters* 24.4 (2020), pp. 787–791. DOI: 10.1109/LCOMM.2020.2968902.

[17] Johan Karedal et al. "A Measurement-Based Statistical Model for Industrial Ultra-Wideband Channels". In: *IEEE Transactions on Wireless Communications* 6.8 (2007), pp. 3028–3037. DOI: 10.1109/TWC.2007.051050.

[18] I. Katzela and M. Naghshineh. "Channel assignment schemes for cellular mobile telecommunication systems: a comprehensive survey". In: *IEEE Personal Communications* 3.3 (1996), pp. 10–31. DOI: 10.1109/98.511762.

[19] M. Latva-aho and K. Leppänen. *Key Drivers and Research Challenges for 6G Ubiquitous Wireless Intelligence*. Tech. rep. University of Oulu, Finland, Sept. 2019. URL: http://jultika.oulu.fi/files/isbn9789526223544.pdf.

[20] David Lopez-Perez et al. “Enhanced intercell interference coordination challenges in heterogeneous networks”. In: *IEEE Wireless Communications* 18.3 (2011), pp. 22–30. DOI: 10.1109/MWC.2011.5876497.

[21] Shani Lu, John May, and Russell J. Haines. “Effects of Correlated Shadowing Modeling on Performance Evaluation of Wireless Sensor Networks”. In: *2015 IEEE 82nd Vehicular Technology Conference (VTC2015-Fall)*. 2015, pp. 1–5. DOI: 10.1109/VTCFall.2015.7390909.

[22] Andreas F. Molisch. *Wireless Communication*. Second. John Wiley & Sons Ltd, 2011. ISBN: 978-0-470-74187-0.

[23] Shayegan Omidshafiei et al. “Deep Decentralized Multi-task Multi-Agent Reinforcement Learning under Partial Observability”. eng. In: (2017).

[24] Christos H Papadimitriou and John N Tsitsiklis. “The Complexity of Markov Decision Processes”. eng. In: *Mathematics of operations research* 12.3 (1987), pp. 441–450. ISSN: 0364-765X.

[25] *Pattern recognition and machine learning*. eng. New York, N.Y, 2006.

[26] *Service requirements for cyber-physical control applications invertical domains*. Tech. rep. TS22.104. 3GPP, Sept. 2020.

[27] Marvin K Simon and Mohamed-Slim Alouini. *Digital communication over fading channels, second edition*. eng. 2nd ed. Wiley, 2005. ISBN: 0471715239.

[28] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, 2015.

[29] Fuxiao Tan, Pengfei Yan, and Xinping Guan. “Deep Reinforcement Learning: From Q-Learning to Deep Q-Learning”. eng. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol. 10637. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2017, pp. 475–483. ISBN: 9783319700922.

[30] Mikko A. Uusitalo et al. “6G Vision, Value, Use Cases and Technologies From European 6G Flagship Project Hexa-X”. In: *IEEE Access* 9 (2021), pp. 160004–160020. DOI: 10.1109/ACCESS.2021.3130030.

[31] Luis G. Uzeda Garcia et al. “Autonomous Component Carrier Selection for 4G Femtocells — A Fresh Look at an Old Problem”. In: *IEEE Journal on Selected Areas in Communications* 30.3 (2012), pp. 525–537. DOI: 10.1109/JSAC.2012.120403.

[32] Christopher J. C. H. Watkins and Peter Dayan. “Q-learning”. eng. In: *Machine learning* 8.3-4 (1992), pp. 279–292. ISSN: 0885-6125.

[33] Ping Xiang et al. “Multi-Agent RL Enables Decentralized Spectrum Access in Vehicular Networks”. eng. In: *IEEE transactions on vehicular technology* 70.10 (2021), pp. 10750–10762. ISSN: 0018-9545.

[34] Jue Xue. "Graph coloringGraph Coloring". In: *Encyclopedia of Optimization*. Ed. by Christodoulos A. Floudas and Panos M. Pardalos. Boston, MA: Springer US, 2009, pp. 1444–1448. ISBN: 978-0-387-74759-0. DOI: 10.1007/978-0-387-74759-0_253. URL: https://doi.org/10.1007/978-0-387-74759-0_253.

[35] Yaodong Yang and Jun Wang. "An Overview of Multi-Agent Reinforcement Learning from Game Theoretical Perspective". eng. In: (2020).

[36] Kaiqing Zhang, Zhuoran Yang, and Tamer Başar. "Multi-Agent Reinforcement Learning: A Selective Overview of Theories and Algorithms". eng. In: *Handbook of Reinforcement Learning and Control*. Studies in Systems, Decision and Control. Cham: Springer International Publishing, 2021, pp. 321–384. ISBN: 3030609898.

Appendix A

Training results

	Value
Tx power	-10 [dBm]
Threshold	12.71 [dB]
Reward type	Binary
Target capacity	8, 15 [Mbps]
Aggregation	No

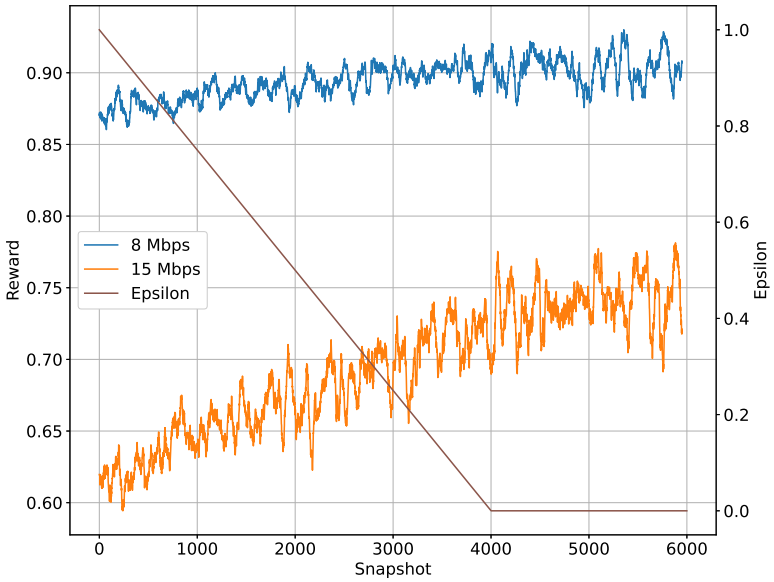

Figure A.1: The target capacities comparison in the training phase.

	Value
Tx power	-10 [dBm]
Threshold	12.71, 4.60, 1.37, -6.73 [dB]
Reward type	Capacity
Aggregation	No

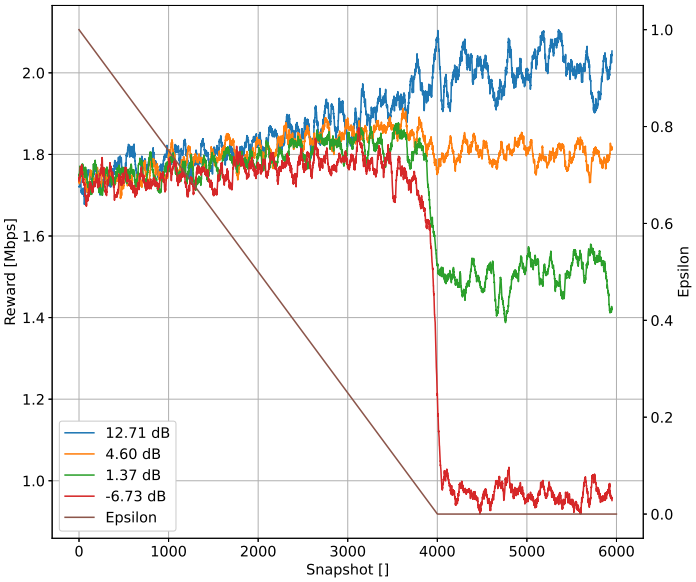

Figure A.2: The state quantization threshold values comparison in the training phase.

	Value
Tx power	-10 [dBm]
Threshold	8.66, 4.60 [dB]
Reward type	Capacity
Aggregation	No

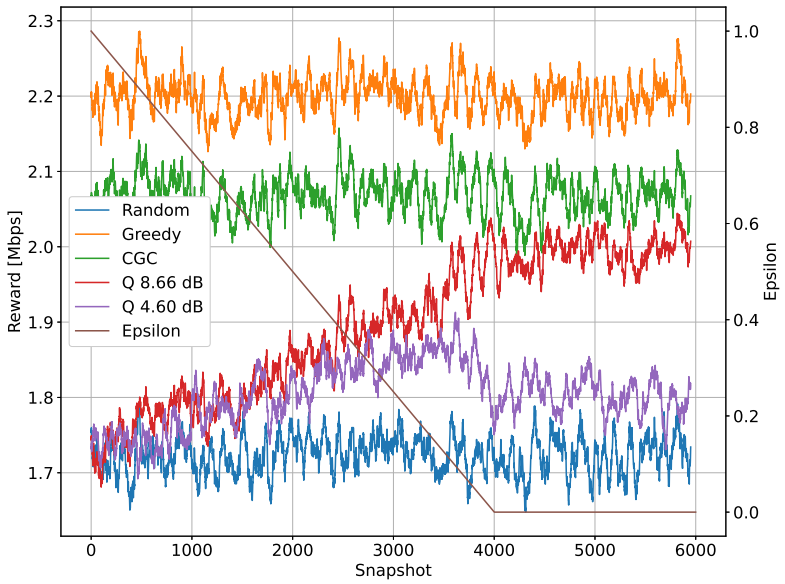

Figure A.3: The two threshold values compared to benchmarks in the training phase.

	Value
Tx power	10, 0, -10 [dBm]
Threshold	8.66 [dB]
Reward type	Capacity
Aggregation	No

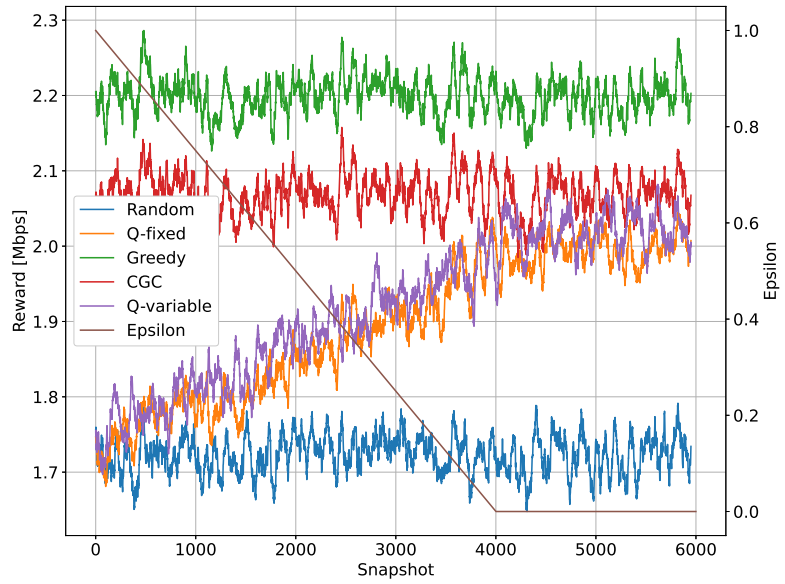

Figure A.4: Comparison of fixed and variable power Q-learning and benchmark in the training phase.

	Value
Tx power	-10 [dBm]
Threshold	12.71, 8.66, 4.60, 1.37, -6.73 [dB]
Reward type	Capacity
Aggregation	Yes
Group size	2

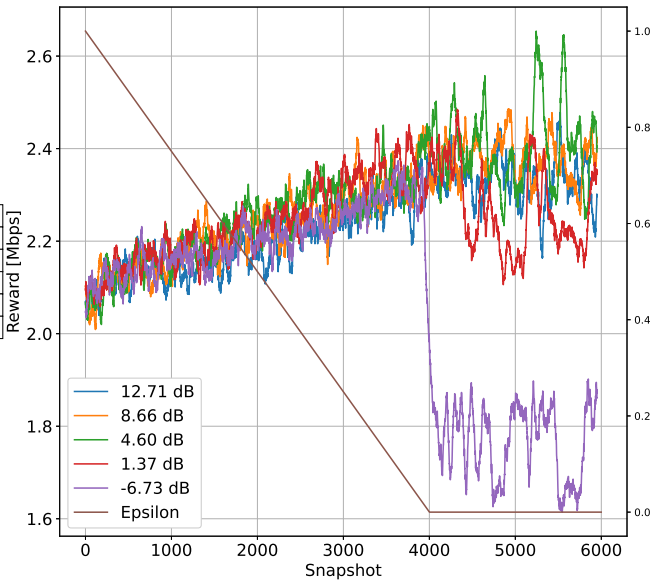

Figure A.5: The state quantization threshold values comparison for problem III in the training phase.

	Value
Tx power	-10 [dBm]
Threshold	4.60, -6.73 [dB]
Reward type	Capacity
Aggregation	Yes
Group size	2

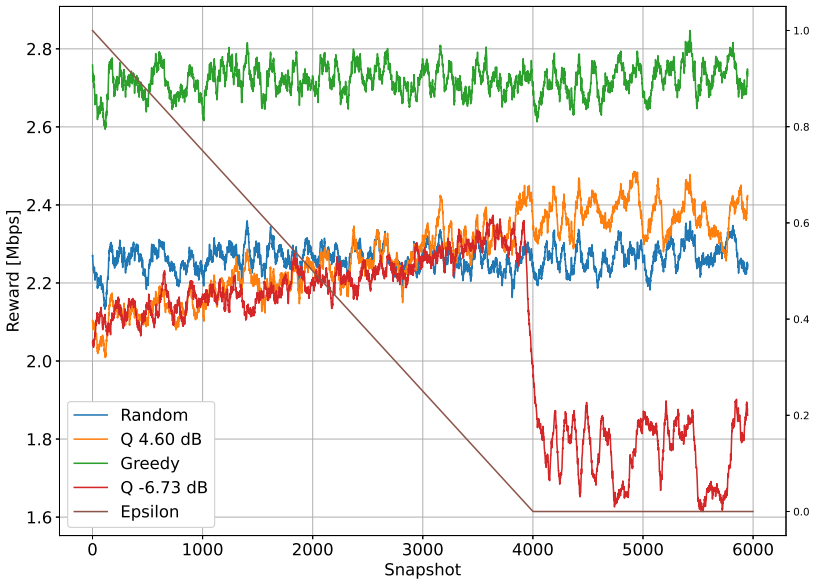

Figure A.6: The two state quantization threshold values compared with the benchmark algorithms in the training phase for problem III.