TECHNISCHE UNIVERSITÄT
ILMENAU

Coordinating Coupled Self-Organized Network Functions in Cellular Radio Networks

A thesis submitted in fulfilment of the requirements for the degree of

Doktoringenieur (Dr.-Ing.)

by: Stephen Ssekiranda Mwanje

to the
Faculty of Computer Science and Automation
Ilmenau University of Technology

urn:nbn:de:gbv:ilm1-2015000135

TECHNISCHE UNIVERSITÄT
ILMENAU

Coordinating Coupled Self-Organized Network Functions in Cellular Radio Networks

Dissertation Zur Erlangung des akademischen Grades

Doktoringenieur (Dr.-Ing.)

von: Stephen Ssekiranda Mwanje

vorgelegt der
Fakultät für Informatik und Automatisierung
der
Technischen Universität Ilmenau

urn:nbn:de:gbv:ilm1-2015000135

Submitted on: 08.09.2014

Defended on: 08.04.2015

Reviewer 1: Prof. Dr. Ing. habil. Andreas Mitschele-Thiel
Ilmenau University of Technology

Reviewer 2: Professor Dr.-Ing. habil. Armin Zimmermann
Ilmenau University of Technology

Reviewer 3: Prof. Dr.-Ing Anja Klein
Technische Universität Darmstadt

Abstract

Owing to increase in desired user throughput and to the subsequent increase in network traffic, the number and density of cells in cellular networks have increased, especially starting with LTE. This directly translates into higher capital and operational expenses as well as increased complexity of network operation. To counter all three challenges, Self-Organized Networks (SON) have been proposed. A number of SON Functions (SFs) have been defined both from the network operator community as well as from the standardization bodies. In this respect, a SF represents a network function that can be automated e.g. Mobility Robustness Optimization (MRO) or Mobility Load balancing (MLB).

The different SFs operate on the same radio network, in many cases adjusting the same or related parameters. Conflicts are as such bound to occur during the parallel operation of such SFs and mechanisms are required to resolve or minimize the conflicts. This thesis studies the solutions through which SON functions can be coordinated in an automated and preferably distributed manner.

In the first part we evaluate the design principles of SFs that aim at easing the coordination. With the observation that the SON control loop is similar to a generic Q-learning problem, we propose designing SFs as Q-learning agents. This framework is applied to two SFs (MRO and MLB) with very positive results.

Given the designed QL based SFs, we then evaluate two SON coordination approaches that consider the SON environment as a Multi-Agent System (MAS). The first approach based on Spatial-Temporal Decoupling (STD) separates the execution of SF instances in space and time so as to minimize the conflicts among instances. The second approach applies multi-agent cooperative learning for an automated solution towards SON coordination. In this case individual SF instances learn based on utilities that aggregate their own metrics as well as the metrics of peer SF instances. The intention in this case is to ensure that the learned state-action policy functions apply actions that guarantee the best result for the active SF but also have the least effect on the peer SFs. Both coordination approaches have been evaluated with very positive results in simulations that consider the MRO - MLB conflict.

Zusammenfassung

Nutzer der Mobilfunknetze wünschen und fordern eine Steigerung des Datendurchsatzes, die zur Erhöhung der Netzlast führt. Besonders seit der Einführung von LTE erhöht sich daher die Anzahl und Dichte der Zellen in Mobilfunknetzen. Dies führt zusätzlich zur Zunahme der Investitions- und Betriebskosten, sowie einer höheren Komplexität des Nerzbetriebs. Der Einsatz selbstorganisierter Netze (SONs) wird vorgeschlagen, um diese drei Herausforderungen zu bewältigen. Einige SON-Funktionen (SF) wurden sowohl von Seiten der Netzbetreiber als auch von den Standardisierungsgremien vorgeschlagen. Eine SF repräsentiert hierbei eine Netzfunktion, die automatisiert werden kann. Ein Beispiel ist die Optimierung der Robustheit des Netzes (Mobility Robustness Optimization, MRO) oder der Lastausgleich zwischen Funkzellen (Mobility Load Balancing, MLB).

Die unterschiedlichen SON-Funktionen werden innerhalb eines Mobilfunknetzes eingesetzt, wobei sie dabei häufig gleiche oder voneinander abhängige Parameter optimieren. Zwangsläufig treten daher beim Einsatz paralleler SON-Funktionen Konflikte auf, die Mechanismen erfordern, um diese Konflikte aufzulösen oder zu minimieren. In dieser Dissertation werden Lösungen aufgezeigt und untersucht, um die Koordination der SON-Funktionen zu automatisieren und, soweit möglich, gleichmäßig zu verteilen.

Im ersten Teil werden grundsätzliche Entwürfe für SFs evaluiert, um die SON-Koordination zu vereinfachen.. Basierend auf der Beobachtung, dass die Steurung der SON-Funktion sich ähnlich dem generischen Q-Learning Problem verhält, werden die SFs als Q-Learning-Agenten entworfen. Dieser Ansatz wurde mit sehr positiven Ergebnissen auf zwei SFs (MRO und MLB) angewandt.

Die als Q-Learning-Agenten entworfenen SFs werden für zwei unterschiedliche Ansätze der SON-Koordination evaluiert. Beide Koordinierungsansätze betrachten dabei die SON-Umgebung als ein Multi-Agenten-System. Der erste Ansatz basierend auf einer räumlich-zeitlichen Entkoppelung separiert die Ausführung von SF-Instanzen sowohl räumlich als auch zeitlich, um die Konflikte zwischen den SF-Instanzen zu minimieren. Der zweite Ansatz wendet kooperatives Lernen in Multi-Agenten-Systemen als automatisierten Lösungsansatz zur SON-Koordination an. Die einzelnen SF-Instanzen lernen anhand von Utility-Werten, die sowohl die eigenen Metriken als auch die Metriken der Peer-SF-Instanzen auswerten. Die Intention dabei ist, durch die erlernte Zustands-Aktions-Strategie Aktionen auszuführen, die das beste Resultat für die aktive SF, aber auch die geringste Auswirkung auf Peer-SFs gewährleisten. In der Evaluation des MRO-MLB-Konflikts zeigten beide Koordinierungsansätze sehr gute Resultate.

To Vivienne, Shanna, and Shevonne
and to my parents Pollycap and Sarah Ssekiranda

"Man can achieve all to which he puts his mind"

My personal belief

Declaration of Authorship

I, Stephen Ssekiranda Mwanje, declare that this thesis titled, 'Coordinating Coupled Self- Organized Network Functions in Cellular Radio Networks' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Stephen S. Mwanje
Ilmenau, 14th April 2015

Acknowledgements

As I conclude my PhD work, I have the opportunity of expressing my gratitude to all the people and institutions that supported me through this work. I am above all indebted to God the almighty who has given me health and resilience to undertake this task. This PhD thesis is the result of a three-year research project carried out within the Integrated Communication Systems group as part of the international Graduate School of Mobile Communications. It was financed by the German Academic Exchange Service / Deutscher Akademischer Austauschdienst (DAAD) under the DAAD Research Grants for Doctoral Candidates and Young Academics.

First, besides the financing, this work would not have succeeded without the unabated support of my supervisor Prof. Andreas Mitschele-Thiel. His uncompromising critique and guidance was immeasurable and has enabled me to finish a rather large project in time. Secondly, I thank Prof. Armin Zimmerman and Prof. Anja Klein for accepting to be reviewers of my thesis and doing it wholeheartedly.

I would also like to thank Siegfried Klein and Dr. Edagr Kühn from Alcatel-Lucent Bell Labs, Germany for their support with the simulator libraries and for reviewing my publications throughout the three years. In the same light, I thank my research team colleagues Elke Roth-Mandutz, Nauman Zia, and Dr. Naseer Ul-Islam with whom I shared my office and also collaborated on the research work in our team. They not only provided a good working environment, but the extensive discussions we had, both technical and non-technical, were very rewarding. I would also like to thank my colleagues in the Integrated Communication Systems group and in the International Graduate School on Mobile Communications for their critical feedbacks during our research seminars and workshops. I am also deeply thankful to our administrative team including Nicole Sauer, Jürgen Schmit and Dr. Mirko Kirschkowski for the friendly assistance with administrative issues.

Most importantly, I thank my entire family - those I traveled with to Germany and those I left home in Uganda - for their patience and support throughout this time.

Summary of Contents

Contents

List of Figures

List of Tables

List of Algorithms

List of Acronyms

3GPP	Third Generation Partnership Project
ALU	Alcatel-Lucent Bell Labs, Germany
BS	Base Station
CCG	Concurrent Cooperative Games
CCO	Coverage and capacity Optimization
CIO	Cell Individual Offset
CIR	Channel Impulse Responses
CSS	Concurrent learning with Spatial Separation
E-UTRA	Evolved Universal Terrestrial Radio Access
FCC	Fuzzy and Combined Conflicts
FFT	Fast Fourier Transform
GBR	Guaranteed Bit Rate
HARQ	Hybrid Automatic Repeat Request
HO	Handover
HOAP	HO Aggregate Performance
HOM	Handover Margin
Hys	Handover Hysteresis
ICIC	Inter-Cell Interference Coordination
IIR	Infinite Impulse Response
IKR	Institute of Communication Networks and Computer Engineering at the University of Stuttgart, Germany
KPI	Key Performance Indicator
LB	Load Balancing
LTE	Long Term Evolution
MAS	Multi-Agent System
MIMO	Multiple Input Multiple Output
MLB	Mobility Load Balancing
MRC	Maximum Ratio Combining

MRO Mobility Robustness Optimization
MVC Metric Value Conflict
NH Number of HO Candidates
OFDM Orthogonal Frequency Division Multiplexing
OFDMA Orthogonal Frequency Division Multiple Access
OTP Optimum Trigger Point
PP Ping-Pong HO
PRB Physical Resource Block
PVC Parameter Value Conflict
QL Q-Learning
QLB Q-Learning based Load Balancing
QMRO Q-Learning based MRO
QoE Quality of Experience
QoS Quality of Service
RACH Random Access Channel
RLB Reactive Load Balancing
RLF Radio Link Failure
RLFE Radio Link Failure due to Early Handover
RLFL Radio Link Failure due to Late Handover
RSRP Reference Signal Received Power
SAS Single-Agent Systems
SC-FDMA Single Carrier Frequency Division Multiple Access
SF SON Function
SINR Signal to Interference and Noise Ratio
SO Self-Organization
SON Self-Organizing Networks
STD Space - Time Decoupling
STS Space - Time Scheduling
TD Temporal Difference
TSL Temporal Separation during Learning
TTT Time To Trigger
UC Use Case
UE User Equipment
UML Universal Modeling Language

Chapter 1

Introduction

Contents

Demand for mobile communication services have exceptionally grown in the last two decades; especially recently driven by the need for mobile Internet connectivity. This has necessitated networks operators to deploy more Base Stations (BSs) in order to meet users' Quality of Service (QoS) expectations at all locations and times. These deployments, characterized by high BS densities, translate into into high *Capital expenditures (CapEx)* and *Operational Expenditures (OpEx)*. Besides, continued demand for better user experience has necessitated newer access schemes and technologies. The combination of dense BS deployments and simultaneous operation of multiple *Radio Access Technologies (RAT)* has however increased the complexity of network design and operation.

In order to overcome these operational challenges and reduce the related expenditures, different players have suggested the need to introduce automation in the networks through Self-Organization (SO) of radio networks [1]. The idea behind SON is the automation of network functions in all the phases of a network's lifetime, like deployment, operation, optimization, maintenance and service recovery. Within the context of SON, a number of SFs also refereed to as Use Cases (UCs)

have been developed. This thesis studies the special problem of how to combine and coordinate multiple SFs in an environment where many are required to work concurrently. The rest of this chapter describes: 1) the problem that has been studied; 2) the scope of the study; 3) a high level introduction of the approaches that have been evaluated and 4) an aid to reading the rest of the thesis through a reading map for different audiences.

1.1 Self-Organization in Cellular Networks

In justifying the need for SO in mobile networks, the *Next Generation Mobile Networks (NGMN)* consortium, a worldwide cooperation of mobile network operators, presented the operators' SON requirements to the standardization bodies [2]. The NGNM identified UCs where SO can be applied, classifying them among self-configuration, self-optimization and self-healing. Following NGNM's publication, research on the structure and development of the UCs in all three network functional areas of configuration, optimization and healing has been undertaken in a number of projects involving both industrial and academic researchers [3][4][5].

Concurrently, in working towards inclusion of SON functionality in the Long Term Evolution (LTE) standards, the Third Generation Partnership Project (3GPP) generated an informative list of UCs which could be considered priority for development [6]. Published as part of the LTE SON standard in [6], the list included nine priority UCs. For each one of the nine UCs, the 3GPP defined concepts and requirements for their development and in some cases generated solution ideas on how they can be realized. In this context, a SF represents any network functionality that can be made self-organized. Some major examples of SFs are:

- *Mobility Robustness Optimization (MRO)* that adjusts *handover (HO)* control parameters – *Hysteresis(HO)* and *Time to Trigger (TTT)* – so as to optimize HO performance
- *Mobility Load balancing (MLB)* that adjusts the HO *Cell specific Offset (CIO)* in order to balance load distribution among neighbor cells and eventually improve user satisfaction in terms of data throughput
- *Coverage and Capacity Optimization (CCO)* that manages the trade off between increasing cell coverage and the effects that such an action would have on the cell/system capacity
- *Inter Cell Interference Coordination (ICIC)* that manages the relative interference caused by one cell to another

As shown in Fig. 1.1, Each SF is characterized by a trigger condition that initiates execution of an associated SON algorithm. The algorithm adjusts or configures a set of network parameters in order to optimize a particular network metric.

Since SFs operate on the same network adjusting the same or related radio parameters, conflicts arise during their operation. Two major conflicts types are

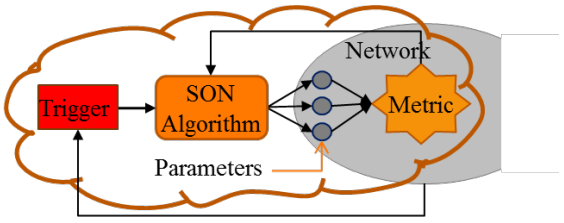


Figure 1.1: Operational Structure of a Self-Organized Network Use Case.

observed: Parameter Value Conflicts (PVCs) and Metric Value Conflicts (MVCs). PVCs arise when multiple SFs adjust the same control parameter(s) and so the SFs conflict on their desired value for those parameter(s). For MVCs, the SFs adjust different parameters but a parameter adjusted by one SF affects a metric of another SF. In either cases, the network may experience instability as different SFs act to restore desired values for either parameters or metrics. Moreover users may experience deteriorated QoS when a setting assumed to have been configured by one SFs has actually been altered by another SF. It is then necessary to devise means to either eliminate or at least minimize these conflicts.

1.2 Thesis Objectives and Scope

SON being a new concept, most work has been done on the development of SFs, with few studies, and only recently, having considered the coordination of SFs. These early studies have focused on ideas related to avoiding the conflicts or on how rules can be designed to schedule execution of SFs in a way that minimizes their cross effects. A number of questions and approaches however remain open:

1. The management of SON conflicts through the design paradigms of individual SFs can be investigated. General principles can be developed on how SFs should be designed so as to ease the management of their conflicts.

2. Opportunities do exist in the scheduling of SFs as a means of minimizing conflicts. These could be evaluated further especially given the multitude of operating conditions in which cellular radio systems are operated.

3. It is also possible to apply generic concepts of multi-agent systems to the SON environment. Considering individual SFs as agents, the coordination challenge becomes a multi agent coordination challenge. In that case, ideas that have been developed in such areas as control theory and multi-agent learning could be applied.

1.2.1 Problem Delineation

The core problem in this work is to devise solutions through which SON conflicts can be coordinated in an automated and preferably distributed manner. Part of

the problem is to implement SON solutions for the candidate SFs designed in a way that eases their coordination. The next step then is to develop coordination approaches and apply them to the developed SFs. The thesis does not address all SON conflicts but focuses on coordination of SFs which demonstrate MVCs. We review the design paradigms for SON functions to identify challenges to effective coordination, and then propose and evaluate different coordination approaches.

1.2.2 Proposed Solutions

The ideas that have been investigated towards SON coordination can be classified into two major packages:

1. **Design and development of SFs:** The ease of coordinating SFs highly depends on how SFs are designed. As such, the first work package concentrated on studying how SFs should be designed so as to simplify their coordination. A single development framework based on QL has been considered. Performance results for two SFs developed based on this framework prove the feasibility of using QL for developing SFs.

2. **SON Coordination:** With a unified development framework, generic approaches have been sought for coordination. Constrained to ensuring system stability, we require generic approaches that can easily be reapplied to any set of SFs. The two approaches considered - one based on Space-Time scheduling and another applying multi-agent cooperative learning - have been tested on the conflicts between MRO and MLB with very good results.

1.2.3 Research Plan

In line with the proposed solutions, activities were structured and followed as listed in the plan below. This is also reflected in the layout of the thesis:

1. Developing individual SFs to be used in the study, where two SFs for MRO and MLB were developed as QL agents

2. Justifying the study by investigating the nature and quantitative measures of conflicts among the SFs

3. Developing the coordination architecture constrained to ensuring that the system remains stable and that it can be scaled by adding new SFs as they are developed

4. Validating the coordination architecture by applying the suggested proposals on the MRO-MLB conflict

5. Generalizing the coordination architecture by evaluating what changes would be required for multiple SFs

1.2.4 Research Methodology

SON functions optimize parameters in different network layers and sub-layers of the OSI model, each parameter affecting one or multiple layers. It is therefore difficult to derive strictly closed-form mathematical solutions that describe system performance. The interaction among multiple SFs further complicates the mathematical modeling and theoretical estimation of system performance. Consequently, performance in this thesis is evaluated relying on system-level simulations. Simple theoretical analyses are however made where possible, especially as a means to verify the simulation results.

1.3 Contributions of the Thesis

In line with the research activities in 1.2.3, the contributions of the thesis can be categorized in three major groups: the development of SFs, the coordination architecture for SFs and the application of learning methods to general optimization problems in communication systems.

1.3.1 Development of SON Functions

A framework for developing SON functions as Q learning agents was proposed including its mapping to the major optimization problems. Three of these are implemented and validated with simulations based on an LTE system-level simulator. The following articles have been published based on this work:

- S. Mwanje, N. Zia, A. Mitschele-Thiel, "Self-Organized Handover Parameter Configuration for LTE", in Proc. ISWCS 2012, Paris, France.
- S. Mwanje, A. Mitschele-Thiel, "Q-Learning for LTE Self-Organized Mobility Load balancing", in Proc. PIMRC 2013 in London, UK.
- S. Mwanje, A. Mitschele-Thiel, "Distributed Cooperative Q-Learning for LTE for Mobility-Sensitive Handover Optimization in LTE SON", in Proc. ISCC 2014 in Madeira, Portugal.
- U. Sallakh, S. Mwanje and A. Mitschele-Thiel, "Multi-Parameter Q-Learning for Downlink Inter-Cell Interference Coordination in LTE SON", in Proc. ISCC 2014 in Portugal.

1.3.2 Coordination of SON Functions

Concepts for coordinating multiple SFs both within and across cells have been studied. The core of the solution is to allow the SFs to communicate with each other so that each SF accounts for its cross effects to the other SFs. Owing to the

need for the SFs to observe and quantify the effects of each action without any superposition of other SFs' actions on top, a space-time scheduling mechanism was developed to separate the SFs' actions in space and time. This ensures that no two SFs take actions that may concurrently affect the same environment or parameters. The following articles have been published on this work:

- S. Mwanje, A. Mitschele-Thiel, "Minimizing Handover Performance Degradation due to LTE Self-Organized Load Balancing", in Proc. VTC spring 2013, Dresden, Germany.
- N. Zia, S. Mwanje, A. Mitschele-Thiel, "A Policy Based Conflict Resolution Mechanism for MLB and MRO in LTE Self-Optimizing Networks", in Proc. ISCC 2014 in Portugal.

1.3.3 Learning in control and optimization problems

The ideas and concepts learned during this work have also been investigated in cooperation with other researchers working on other topics. The following papers have been published from these collaborations:

- E. Roth-Mandutz, S. Mwanje and A. Mitschele-Thiel, "RSS based Cell Fingerprint Patterns and Algorithms for Cell Identification in the Context of Self-organized Energy Saving", Proceeding of Mobiquitous 2014, Tokyo, Japan.
- A.H. Mahdi, Z. Ansar, S. Mwanje, O. Artemenko, A. Mitschele-Thiel, "improving performance of a Self-Organized Cognitive Engine using Q-Learning", Proceeding of WCNC 2014, Istanbul, Turkey.

1.4 Thesis Organization

The previous sections presented material that introduces the problem and the approaches that have been taken towards solving the problem. They also summarized the contributions of the thesis on different areas. This section summarizes the contents of the rest of the thesis and lays out a map indicating how its reading could be undertaken by different audiences in consideration of their backgrounds.

1.4.1 A Review of the Chapter Contents

Chapter 2 reviews the fundamental literature required to fully understand the rest of the thesis. It gives a brief review of the LTE radio system with emphasis on Radio Resource Management, the understanding of which is crucial to the discussion on the SFs. It then presents the previous work on SON highlighting the state of art on SON coordination and the design criterion that have been applied

in SON. It concludes with a discussion of Learning in Multi Agent Systems and how it could be applied to SON coordination.

In chapter 3 we describe the simulation environment and modeling of the critical radio system precesses and events. Unlike in typical research reports where the simulation scenario and parameters are presented after a discussion of the solutions, I present this before since foregoing chapters present different aspects of the solutions with their results. The chapter begins with the high level system structure followed by the cellular radio models. It then discusses the mobility and handover models highlighting the events that result from different control settings of the handover parameters. It concludes with a discussion of the snapshot-, event-based simulation model concentrating on the deployment process of the different network components and the subsequent snapshot processing during the simulation.

The proposed framework for QL as the design structure for the SFs is presented in chapter 4. The chapter begins with a review of QL, then summarizes how QL has been applied in SON and justifies how QL can be used for any generic SF. The discussion then turns to the application of this framework to the two selected SFs, starting with MRO and followed by MLB. For each SF, I highlight the design requirements and constraints as well as the possible solutions to these constraints. Finally, the expected performance results are presented, in each case comparing the observed performance to that of a reference system that does not apply any automation or SON solution.

Chapter 5 discusses the coordination problem by evaluating the performance degradation that results when the multiple SFs are operated concurrently without coordination. We discuss the expected sources or causes of the conflicts and demonstrate the degree of conflicts through results of a set of simulations. Since the objective is only to demonstrate that conflicts do happen and not to show all forms of conflicts, the chapter only focuses on the results that show that conflicts exist and not on the detailed evaluation of their nature.

The SON coordination proposals on Space-Time Scheduling procedures and Concurrent Cooperative Games are presented in chapter 6. Considering the MRO - MLB conflict in both cases, I highlight the benefits of the coordination solutions through simulation evaluations. The evaluations compare results of the two SFs when operated with the coordination solutions against results of the independent SFs as obtained in chapter 4 and also against the degradation results of chapter 5. We conclude with a discussion of the potential limitations of the methods in question and where possible how such limitations can be addressed or managed in a real deployment.

Chapter 7 presents the concluding remarks of the thesis and an outlook to what future studies on the subject should entail. It begins with a summary of the general philosophy to the considered approaches and obtained results. It then discusses potential areas in which the proposed solutions could be improved and other approaches that could be taken, not only towards SON integration but also for the general SON philosophy. In particular it highlights the concept of

Bayesian QL that applies a combination of game theory and QL to allow SFs to independently learn of the potential conflicts instead of such possible conflicts being configured into the Q-learner as is the current approaches. It also proposes reworking the approach to SON. I propose to change the focus from the 4 functional areas of network operation (planning, configuration, optimization and healing) and instead focus on the natural layering of network functions in terms of Physical resources, logical resources and quality of experience.

1.4.2 Proposed Reading Order

Fig. 1.2 presents the proposed order of reading the thesis for different audiences. For one working on SON in LTE/LTE-A, it may be adequate to go direct to the core content in chapters 4 and 6. Otherwise chapter 2 gives a quick, albeit steep introduction to the basic background on LTE and SON that would be required for the further chapters. Chapter 3 is critical for a reader interested in evaluating the soundness of the ideas and as to whether they have been tested in a realistic environment. Otherwise one can still follow the discussions on the SFs in chapter 4 without the detailed understanding of the simulation environment.

For those who need to appreciate the nature of the conflicts among the two SFs, these are discussed in chapter 5 laying the foundation for evaluating the benefit of the coordination as presented in chapter 6. Those who however are working on SON should be able to follow directly from the discussion of the SFs in chapter 4 to the discussion of the solutions to SON conflicts in chapters 6.

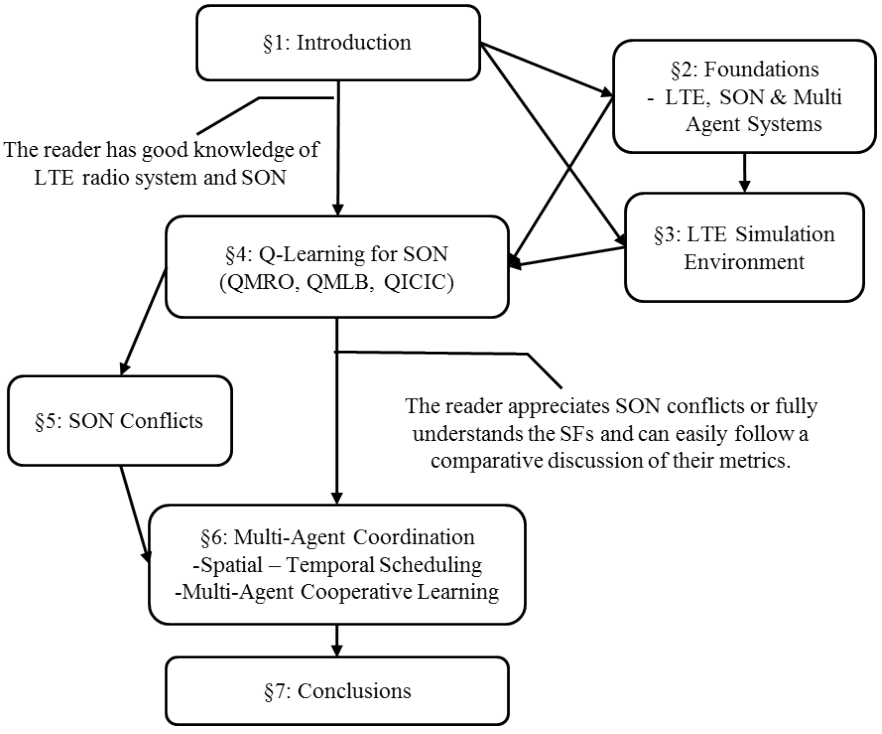


Figure 1.2: Proposed thesis reading order for different audiences.

Chapter 2

LTE, SON and Multi-Agent Systems: A Review

Contents

Studying the coordination of SFs in Cellular Systems requires the understanding of not only the structure and operation of the SFs but also of the underlying cellular radio technology. Only then can potential approaches be evaluated in consideration of the foundational constraints. This chapter reviews these foundations, setting a basis for understanding the rest of the content in the thesis. It begins with a brief discussion of the fundamental technological ideas behind the LTE radio system. These are important not only for the discussions on individual SFs, especially where the discussion references the standardized LTE features, but

also for the understanding of the reference system on which the ideas have been validated. The discussion then turns to SON and describes the critical developments to date and the SON coordination challenge. It also justifies the need to study this subject and highlights the potential benefits that would be expected. With the observation that SON exemplifies a Multi-Agent System (MAS), the last section discusses MASs focusing on their coordination and control aspects. It provides insight into existing work on MAS as undertaken in other fields and then highlights how such ideas could be applicable to the SON environment.

2.1 The LTE Radio System

LTE was developed to provide users with rich-content such as High Definition TV (HDTV) broadcast, movies on demand, and voice-over-IP (VoIP). The LTE E-UTRA was designed with the main objectives being: 1) development of an optimized packet-based access system with high data rate and low latency; 2) providing common base-band support for Time Domain Duplex (TDD) and Frequency Domain Duplex (FDD) modes; 3) reducing capital and operating expenditure; and 4) reasonable system and terminal complexity [7]. The resultant physical layer (PHY) performance requirements are as summarized in Table 2.1 while the critical radio system specifications are described in the subsequent subsections.

2.1.1 OFDM Downlink Transmission

In the downlink, Orthogonal Frequency Division Multiplexing (OFDM) was selected to efficiently meet E-UTRA throughput requirements. OFDM systems do

Table 2.1: Physical layer requirements for LTE E-UTRA.

Requirement	LTE E-UTRA Specification
Peak data rate	100 Mbps DL /50 Mbps UL
Spectral Efficiency	3-4x DL / 2-3x UL of HSxPA Rel-6
5% packet call throughput	3-4x DL / 2-3x UL of HSxPA Rel-6
Averaged user throughput	3-4x DL / 2-3x UL of HSxPA Rel-6
U-Plane Latency	less than 10 msec from UE to server
C-Plane Latency	50 – 100 msec to establish U-plane
Call setup time	50 ms
Broadcast data rate	6-8x improvement
Mobility	Optimized for low speeds (<15 Kmph) High performance at speeds up to 120 Kmph Maintain link at speeds up to 350 Kmph
Multi-antenna support	4x2, 2x2, 1x2, 1x1 DL / 1x2, 1x1UL
Bandwidth	Scalable: 1.25, 2.5, 5, 10, 20 MHz
Coverage	Full performance up to 5 km Slight degradation 5 km – 30 km Operation up to 100 km also possible

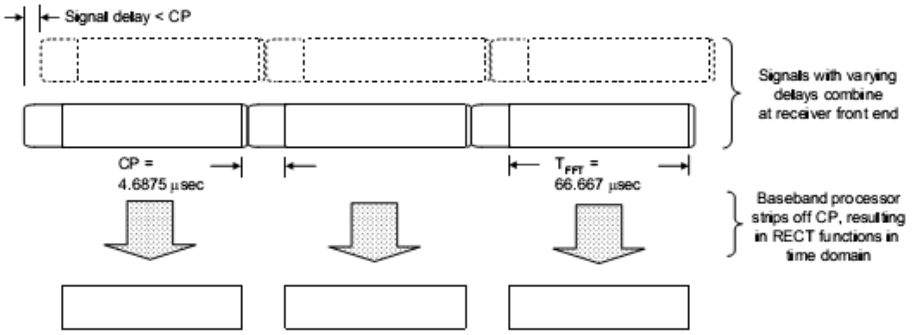


Figure 2.1: OFDM eliminates ISI via longer symbol periods and a cyclic prefix.

not rely on increased symbol rates in order to achieve higher data rates. Instead they break the available bandwidth into many narrower sub-carriers and transmit the data in parallel streams. Each subcarrier is modulated using varying levels of QAM modulation, e.g. QPSK, QAM, 64QAM or possibly higher orders depending on signal quality. Each OFDM symbol is therefore a linear combination of the instantaneous signals on each of the sub-carriers in the channel.

Consider the time-domain representation of an OFDM symbol shown in Fig. 2.1. The OFDM symbol consists of two major components: the cyclic prefix (CP), and a FFT period. The duration of the CP is determined by the highest anticipated degree of delay spread for the targeted application [8]. When transmitted signals arrive at the receiver by two paths of differing length, they are staggered in time as shown in Fig. 2.1. Consequently, within the CP, it is possible to have distortion from the preceding symbol. However, with a CP of sufficient duration, preceding symbols do not spill over into the Fast Fourier Transform (FFT) period (TFFT) so that interference is only due to time-staggered "copies" of the current symbol. Once the channel impulse response is determined (by periodic transmission of known reference signals), distortion can be corrected by applying an amplitude and phase shift on a subcarrier-by-subcarrier basis.

2.1.2 OFDMA and the LTE Generic Frame Structure

Orthogonal Frequency Division Multiple Access (OFDMA) is employed as the multiplexing scheme in the LTE downlink. In OFDMA, users are allocated a specific number of subcarriers for a predetermined amount of time. These are referred to as Physical Resource Blocks (PRBs) in the LTE specifications. PRBs thus have both a time and frequency dimension, allocation of which is handled by a scheduling function at the 3GPP base station (eNodeB).

Consider the LTE PHY layer generic frame structure of Fig. 2.2. This structure is used with FDD but alternative frame structures are defined for use with TDD. As shown in the figure, LTE frames are 10 msec in duration. They are divided into 10 subframes, each being 1.0 msec long. Each subframe is further divided into two

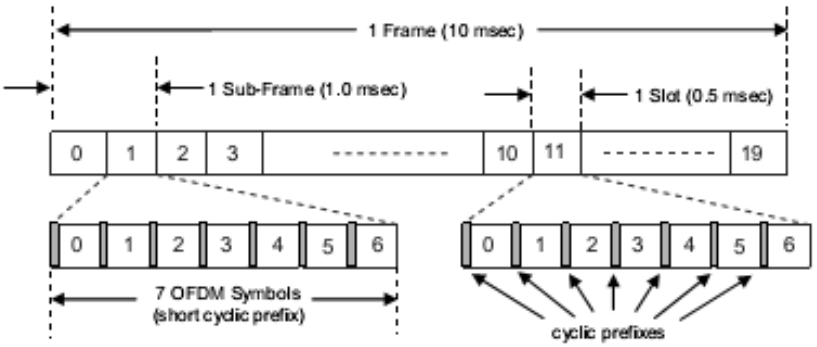


Figure 2.2: LTE Generic Frame Structure.

slots, each of 0.5 msec duration and consisting of either 6 or 7 OFDM symbols, depending on whether the normal or extended cyclic prefix is employed.

To provide great operational flexibility, E-UTRA physical layer specifications are bandwidth agnostic and designed to accommodate up to 20 MHz system bandwidth so that the total number of available subcarriers depends on the overall system transmission bandwidth as shown in Table 2.2. The smallest element of resource allocation assigned by the base station scheduler is the PRB. It is defined as consisting of 12 consecutive sub-carriers for one slot (0.5 ms) in duration. Sub-frames with one of two cyclic prefix (CP) durations may be time-domain multiplexed, with the longer designed for larger cells. The useful symbol duration is constant across all bandwidths and the 15 kHz sub-carrier spacing is large enough to avoid degradation from phase noise and Doppler effects up to 250km/h at 2.6 GHz with 64QAM modulation [9].

The transmitted downlink signal consists of N_{BW} subcarriers for a duration of N_{symb} OFDM symbols as represented by the resource grid in Fig 2.3. Each box within the grid represents a single subcarrier for one symbol period and is referred to as a resource element. Note that in Multiple Input Multiple Output (MIMO) applications, there is a resource grid for each transmitting antenna.

Table 2.2: OFDMA parameters for LTE E-UTRA.

Bandwidth (MHz)	1.25	2.5	5.0	10.0	15.0	20.0
Subcarrier bandwidth (kHz)	15					
Subcarrier duration (ms)	1.0					
PRB bandwidth (kHz)	180					
Sampling Frequency (MHz)	1.92	3.84	7.68	15.36	23.04	30.72
FFT Size	128	256	512	1024	1536	2048
Number of available PRBs	6	12	25	50	75	100
CP Length (μS) - Normal	4.69 x 6, 5.21 x1					
CP Length (μS) - Extended	16.6					

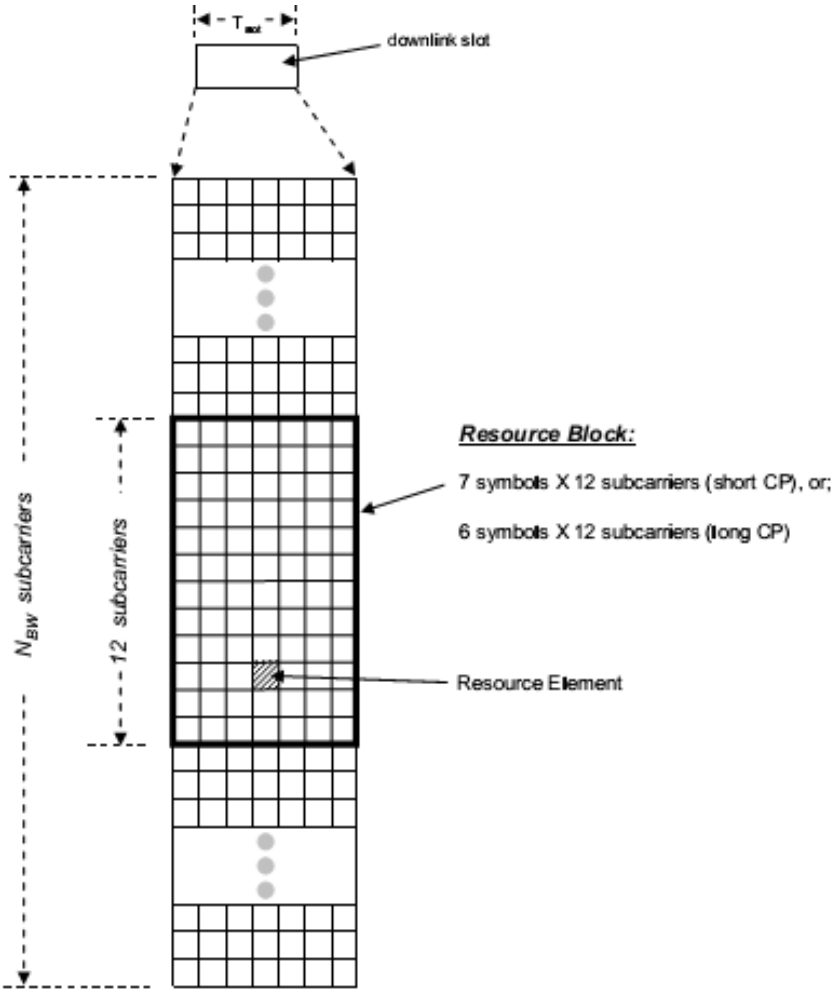


Figure 2.3: Downlink Resource Grid.

2.1.3 SC-FDMA Uplink Transmission

LTE up-link requirements differ from down-link requirements in several ways, the most important being power consumption for UE terminals. The high *Peak to Average Power Ratio (PAPR)* and related loss of efficiency associated with OFDM signaling are major concerns. Single Carrier Frequency Division Multiple Access (SC-FDMA) was selected as best suited to the LTE uplink requirements. The basic transmitter and receiver architecture (Fig. 2.4) is however very similar (nearly identical) to OFDMA, and it offers the same degree of multipath protection [8]. Importantly, because the underlying waveform is essentially single-carrier, the PAPR is lower.

In the transmitter / receiver arrangement, many of the functional blocks are common to both SC-FDMA and OFDMA. As such, there is a significant degree of functional commonality between the uplink and downlink signal chains. The functional blocks in the transmit chain are:

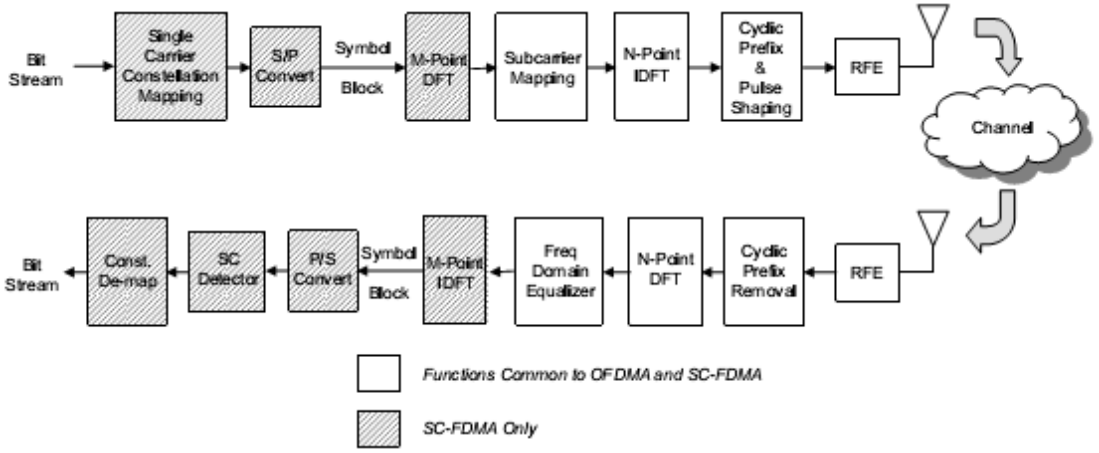


Figure 2.4: SC-FDMA and OFDMA Signal Chains [8].

Figure 2.5: SC-FDMA Subcarriers in Localized and Distributed Mode [8].

- Constellation mapper: This converts the incoming bit stream to single carrier (SC) symbols as BPSK, QPSK or 16 QAM depending on channel conditions.

- Serial/parallel (S/P) converter: This formats time-domain SC symbols into input blocks to the FFT engine.
- M-point DFT: This converts time domain SC symbol block into M discrete tones.
- Subcarrier mapping: It maps DFT output tones to specified subcarriers for transmission. SC-FDMA systems either use contiguous tones (localized) or uniformly spaced tones (distributed) as shown in Fig. 2.5.
- N-point IDFT: This converts mapped subcarriers back into the time domain for transmission.
- Cyclic prefix (CP) and pulse shaping: As in the case of OFDM, the CP is prepended to the composite SC-FDMA symbol to provide multipath immunity while pulse shaping is employed to prevent spectral regrowth.
- RF Front End (RFE): It converts digital signals to analog and then upconverts to RF for transmission.

In the receive side chain, the process is essentially reversed, although as in OFDM, SC-FDMA transmissions can also be considered linear summations of discrete subcarriers. Multipath distortion is also handled in the same manner i.e. removal of

CP, conversion to the frequency domain, then applying the channel correction on a subcarrier-by-subcarrier basis. Unlike OFDM however, the underlying SC-FDMA signal represented by the discrete subcarriers is - not surprisingly - single carrier. This is distinctly different because SC-FDMA subcarriers are not independently modulated. As a result, PAPR is lower than for OFDM transmissions [8].

2.1.4 Multi-Antenna Subsystem and MIMO

The LTE physical layer can optionally exploit multiple transceivers at both the base-station and UE in order to enhance link robustness and increase data rates. In particular, Maximum Ratio Combining (MRC) is used to enhance link reliability in challenging propagating conditions e.g. when the signal strength is low and/or multipath conditions are challenging. MIMO is a related technique that is used to increase system data rates.

With MRC, a signal is received via two (or more) separate antenna/transceiver pairs (Fig. 2.6). The antennas are physically separated, and therefore have distinct Channel Impulse Responsess (CIRs). Channel compensation is applied to each received signal within the baseband processor before being linearly combined to create a single composite received signal. The result is that the received signals add coherently but the thermal noise from each transceiver is uncorrelated. Thus, linear combination of the channel compensated signals at the baseband processor results in an increase in SNR, i.e. up to 3 dB on average for a two-channel MRC receiver in a noise-limited environment [8].

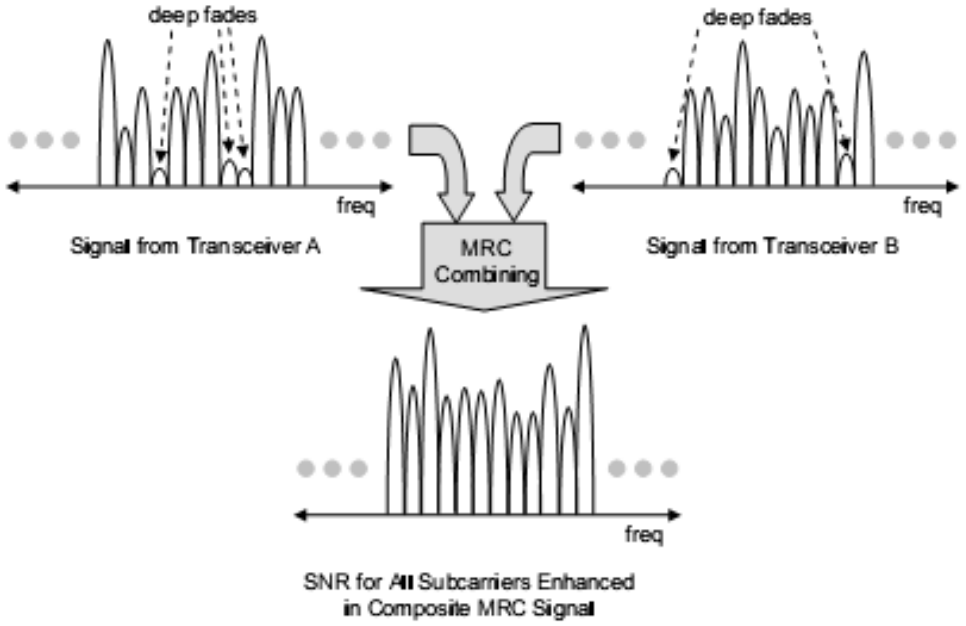


Figure 2.6: MRC under AWGN and Frequency Selective Fading.

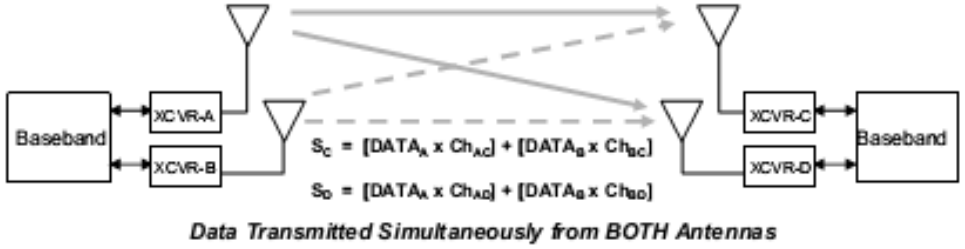


Figure 2.7: MIMO Operation and A Priori Knowledge of all CIRs [8].

MRC enhances link reliability, but does not increase the nominal system data rate. In MRC, data is transmitted by a single antenna and is processed at the receiver via two or more receivers. MRC is therefore a form of receiver diversity rather than more conventional antenna diversity. MIMO on the other hand does increase the system data rates, by using multiple antennas at both the transmitting and receiving ends. Referring to the 2 x 2 MIMO system in Fig. 2.7, there are a total of four CIRs (C_{AC}, C_{AD}, C_{BC} and C_{BD}). Once the CIRs are known, data can be transmitted from both antennas simultaneously. The linear combination of the two data streams at the two receiver antennas results in a set of two equations and two unknowns, which is resolvable into the two original data streams.

2.2 SON in LTE/LTE-A

Following the brief introduction to LTE as given in 2.1, this section reviews the work on SON especially as has been applied in LTE and its advancement (LTE-A).

Most work on SON in LTE has been on three major fronts: 1) The network operators' perspective on SON benefits; 2) standardization of SON in LTE; and 3) research on the requirements, nature and implementation of SON functions. The Next Generation Mobile Networks (NGMN) consortium, the cooperation of mobile network operators from all over the world, justified the need for SON and presented the operators' SON requirements to the standardization bodies [2]. As part of the LTE standard both in the initial release 8 and in the subsequent release 9, the 3GPP included descriptions of the required SON functionality as part of the standard.

Various research projects have been undertaken to define, prototype and test SON functionality, among them being: the Monotas project (Mobile Network Optimization Through Advanced Simulation) and the Gandalf project (Monitoring and self-tuning of RRM parameters in a multi-system network). The Monotas project examined the feasibility of decreasing the time taken to optimize mobile networks by "using simulation techniques to match network performance with parameter settings" [3]. The Gandalf project studied the application of large scale network monitoring, Radio Resource Management (RRM) rules and quality of

service evaluation towards automation of network management tasks in a multi-system environment [4].

Most SON related work has however been undertaken within the SOCRATES project (a European Union FP7 Project undertaken by a number of major research institutions and equipment vendors across Europe). The project expanded the list of SFs as published by NGNM and proceeded to develop solutions for some of the most critical ones [10]. SEMAFOUR, the currently running follow up project to SOCRATES aims to develop a "unified self-management system, which enables the network operators to holistically manage and operate their complex heterogeneous mobile networks" [11].

2.2.1 SO Functions and Development Efforts

The NGNM defined SON use cases as foreseen by operators [12]. The underlying objective was not only to maximize the users quality of experience but to do so at the least possible cost to the operators, both financially and operationally. However, even current second and third generation networks involve hundreds of parameters that can be tuned as well as hundreds of Key Performance Indicators (KPIs) that must be monitored in normal network operation. With this level of complexity the underlying objective could not be addressed with a single automation function, but it had to be broken into multiple sub objectives (called use cases) each of which could then be solved with a single function.

The definition of the use cases, including the operators' guidelines and recommendations for implementation, covered the typical areas of network operation classified under 5 themes:

- SON enablers
- Self-configuration
- Self-optimization
- Fault Management and Fault Correction
- Operation and maintenance (O&M) Related SON

In the LTE SON standard, the 3GPP selected nine use cases/SFs that are expected to be critical to network operations and highlighted them as priority for implementation. Defined in [6], these nine SFs cover the three SON paradigms of configuration, optimization and healing as summarized by Table 2.3.

As stated earlier, the biggest output related to SON has to date come from the Socrates Project. The Project described 25 SFs related to the LTE air interface, classifying each as belonging to one of the three network functional areas - configuration, optimization and healing [13],[14]. Socrates also defined the criteria for evaluating self-organization methods [15] and identified the need for integrating these methods [16] [17].

Table 2.3: Critical SON Use cases as proposed by 3GPP.

Use Case	acronym	SON area
Coverage and capacity optimization	CCO	Optimization, Healing
Reduction of energy consumption	REC	Optimization
Interference Reduction	IR	Optimization
Automated Configuration of Physical Cell Identity	PCI	Configuration
Mobility Robustness Optimization	MRO	Optimization
Mobility Load Balancing Optimization	MLB	Optimization
RACH Optimization	RACH	Optimization
Automatic Neighbor Relation Function	ANR	Configuration, Optimization
Inter Cell Interference Coordination	ICIC	Optimization

Individual SFs have been studied, albeit to varying depths and results, with the most famous being handover optimization [17][18][19][20][21], admission control optimization [22][23] and load balancing [24][25]. However, other functions have also been studied including RACH self-tuning in [26] as well as packet scheduling parameter optimization, self-optimization of home eNodeBs, cell outage management, X-map estimation, and automatic generation of initial parameters for eNodeB insertion in [5].

2.2.2 Design Paradigms for SON Functions

SON algorithms have been designed using a number of different control and optimization structures. To fully appreciate the coordination challenge, we need to understand these different structures and their implication towards SON coordination. This section reviews these structures and then highlights some of the works where they have been applied towards SON and automated management of cellular networks. We describe the four major categories of meta-heuristic, rule-based, gradient-based and machine learning algorithms. We then conclude with a qualitative comparison of the different approaches and justify our selection of reinforcement learning for the studies in this thesis.

2.2.2.1 Meta-Heuristic Algorithms

Meta-heuristic algorithms (MHA) are optimization algorithms that iteratively try to improve the solution based on some measure of quality. These algorithms can quickly search large portions of the solution space but do not guarantee to find the optimal solution. A number of SFs have been designed using meta-heuristic algorithms since they can quickly find a reasonably good solution.

The simplest MHA is Local Search (LS) which starts with one of the possible solutions and then iteratively checks the quality of the neighboring solutions. If the neighbor solution performs better than the previous solution, it is selected as the best solution. This process repeats until a certain quality of solution is achieved

or a fixed number of iterations is completed. LS has been used for Antenna tilt optimization in [27] and Handover parameter configuration in [28]. Simple LS algorithms however suffer from the local maxima problem where the optimizer gets locked in a specific solution when all its neighbor solutions are worse yet there could be a better solution away from its neighbors.

Simulated Annealing (SA) tries to solve the local maxima problem by selecting neighbor solutions as best solution with some probability. This is done even when the neighbor solutions have lesser quality of measure. In order to ensure convergence of the optimization, this probability of selecting neighbor solutions decreases with increasing number of iterations. SA approaches have also been utilized for antenna tilt optimization in [29] [30] [31] [32] [33]; for optimizing micro cell base station locations in [34] and for self-optimization of handover parameters in [35]. Performance of these techniques highly depends on the algorithm's parameter setting for example the probability of selecting a bad solution or the definition of a neighbor solution. This is critical to the extent that an appropriate definition of neighbor solutions can enhance the quality of the final solution as well as reduce the computational complexity [36].

Tabu Search (TS) is another local search algorithm which tries to overcome the problem of local maxima by maintaining a list of recently tested solutions. The solutions in this list remain tabu (prohibited) for some time to ensure that the optimizer does not remain stuck in a specific region of the solution space. This has been applied for antenna tilt optimization in [37] [38]; for generic radio coverage optimization in [39] and for the automated design of LTE Access networks in [40]

Another class of meta-heuristic algorithms are Genetic Algorithms (GA). These mimic the process of natural evolution to find the best solution for the problem. Instead of getting a neighboring solution that is a slight variation of the current solution, GA combine two solutions from a population of solutions to create a child solution that inherits the best qualities of the two parent solutions. GA have been proposed for automation of cell planning tools [41][42], for automated LTE resource allocation [43][44], for PCI planning [45] and for improving energy efficiency in LTE [46].

Another population-based approach, Particle Swarm Optimization (PSO) has also been applied in SON. In this case, a population of candidate solutions moves in the solution space in a way that their movement is influenced by their own best known solution and the global best known solution at any time. Corresponding to parallel search, this has been used for antenna tilt optimization [47] and for load balancing [48].

2.2.2.2 Rule-Based Algorithms

Some rule-based techniques have also been proposed for SFs. Relying on network and cell statistics, rules are defined to describe how cells react in case certain

events are observed. Such techniques have been applied for handover optimization [18], antenna tilt optimization [49] [50] [51], and load balancing [24] [52] [53].

2.2.2.3 Gradient-Based Algorithms

Gradient methods are algorithms that solve problems of the form described by equation 2.1. The solution is defined such that at any point during the optimization, the search direction follows the gradient of the function at that point.

$$\min_{x \in \mathbb{R}^n} \; f(x) \tag{2.1}$$

As evident, the utility function $f(x)$ whether continuous or discrete over x, must be available or defined for these methods to be used. This makes gradient methods less useful in SON where such functions are in many cases unavailable and yet synthesizing them requires operation over long periods of sub-optimal performance so as to obtain adequate sample values of the function. Nevertheless, gradient-based schemes have been exploited for antenna tilt optimization in [54] and [55].

2.2.2.4 Machine Learning

Machine learning is a scientific discipline in which a system automatically learns and improves solutions through experience. The learning methods can be mainly classified into:

1. supervised learning where known input-output pairs are supplied to the learning agent so as to learn patterns that can then be used to deduce solutions for inputs with unknown outputs;
2. unsupervised learning where the agent is supplied with un-categorized data so that the agent learns to categorize the data;
3. reinforcement learning where an agent acts in an environment in which effects of actions are unknown but where with the experience of acting, the agent learns and improves its knowledge of the goodness of solutions.

Machine learning has been applied in many network automation problems among then: interference management [56], handover management [57] and coverage and capacity optimization [58].

2.2.2.5 Benefits and Challenges of Optimization Approaches

All the approaches discussed above have been quite effective for the different automation problems where they have been applied. Meta-heuristic algorithms are the most widely used optimization approaches as they can provide solutions with significant performance improvements in short intervals of time even for very large networks. However, they can only be run in an off-line manner because of the

requirement to test different candidate solutions for the specific quality of measure. This implies that the quality of the solutions derived with Meta-heuristic algorithms will highly depend on the network and environmental models used in the off line modeling and planning tools.

Rule-based algorithms can be applied online within an operational network, although their performance gains have typically been proved using simulations. Nevertheless, most of these approaches have only considered synthetic scenarios, like uniform cell structures, static mobility profiles and deterministic or statistical load offers, for which rules are easier to derive and design. In reality, the number of possible network configurations increases exponentially with network sizes which, in case of non-uniformly structured rules, would require an extremely large rule base to fully address all configurations.

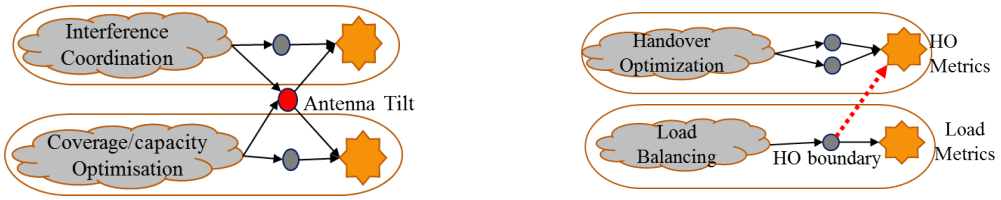
Another major problem with these approaches is that the optimization intelligence remains heavily concentrated in the design phase of the algorithms. Additionally, these algorithms are not able to learn from their experience of previous optimization steps. Instead, they rely on network optimization engineers to either fine tune their results in the operational network or to adjust the parameters of the optimization algorithms. Gradient methods which may not require fine tuning instead require the optimization functions which are also not typically known.

The above challenges are overcome with the use of machine learning algorithms which allow the system to use its gained experience to improve solutions. Additionally, using a distributed architecture for the learning mechanism, the optimization can be adjusted according to the local requirements of the individual cells. More importantly however, to coordinate multiple control agents each of which is based on a different optimization structure inherently complicates the challenge further. Consequently, a single optimization structure should as much as possible be applied for all SFs. Given the benefits of machine learning, we have thus selected it as the best control structure for designing the optimization functions. Specifically, we use Reinforcement (Q-) learning for all SFs so that the coordination problem is reduced to a multi-agent Q-learning problem.

Details on the example designs and implementations of Q-learning for SON function is the subject matter for chapter 3 but we briefly review the literature on multi agent system in section 2.4.

2.2.3 SON Conflicts - The Need for SON Coordination

Individually, each of the SFs described in 2.2.1 optimizes only a very small part of a network's operation. To achieve the desired global goals, all the SFs must be operated on the same radio network and in many cases multiple SFs must be simultaneously operated. However, because they are operating on the same network on which they adjust the same or related parameters, the SFs Conflict with one another. As shown in Fig. 2.8, two (2) major conflicts have been identified



(a) Parameter Value conflict. **(b)** Metric Value conflict.

Figure 2.8: Examples of the two Major SON conflicts.

– PVCs and MVCs [17][5], although other non distinct conflicts and interactions here classified as Fuzzy and Combined Conflicts (FCC) may also be observed.

2.2.3.1 Parameter Value conflicts (PVC)

To achieve the desired metric values for a given SF, a number of parameters have to be tuned. The same parameters may however be tuned by a different SF to achieve a different metric. In this case the two SFs are said to experience a parameter value conflict. An example of this is the possible conflict for antenna tilts by both Inter-Cell Interference Coordinations (ICICs) and Coverage and capacity Optimizations (CCOs) as shown in Fig. 2.8a.

2.2.3.2 Metric Value conflicts (MVC)

In the case of MVCs, a SF optimizes its own parameters but inadvertently affects a metric of another SF although the two SFs may be adjusting different parameters. An example MVC is when the MLB-adjusted Cell Individual Offset (CIO), affects MRO metrics as shown in Fig. 2.8b. Another example is that Interference Coordination can change transmit data rates within cells which would have effects on load conditions within the cells. This will as such affect overload related SFs like MLB or admission control optimization.

2.2.3.3 Fuzzy and Combined Conflicts (FCC)

Besides PVCs and MVCs, other relationships may exist but would not require integration activity during SON development. An example is having multiple SFs that share inputs. In that case coordination may not be necessary unless the two SFs also exhibit PVCs and/or MVCs. Similarly there could be situations where the effective conflict is an indirect combination of both PVCs and MVCs. Such a case is where two parameters that have the same effect are controlled by two SFs aimed at different metrics. Although this may be considered a MVC, the fact that the two parameters are highly related makes it exhibit a PVC. The case of MLB adjusting CIO is such a conflict. In effect, the CIO has exactly the same effect as the HO hysteresis controlled by MRO only that the CIO is specific to a single

neighbor while the Hysteresis is uniform for all neighbors. A clear boundary may therefore not be easy draw in some circumstances between PVCs and MVCs.

Although PVCs present an interesting problem that arises from toggling of a parameter's values between two SFs, their nature indicates that a single optimization function has actually been split into multiple small functions. Consequently, quick solutions can be developed for PVCs either by considering the two conflicting SFs as a single SF or by jointly optimizing them. MVCs on the other hand pose an interesting problem. Given that they do not conflict on the parameter values indicates two different objectives that can not easily be combined. Instead the conflicts represent competing objectives for which either smart solutions would be required to decouple the competition or compromises would need to be developed in ways that maximize all the competing objectives. As such, this thesis focuses more on MVCs or the fuzzy cases in which the MVCs is more prominent.

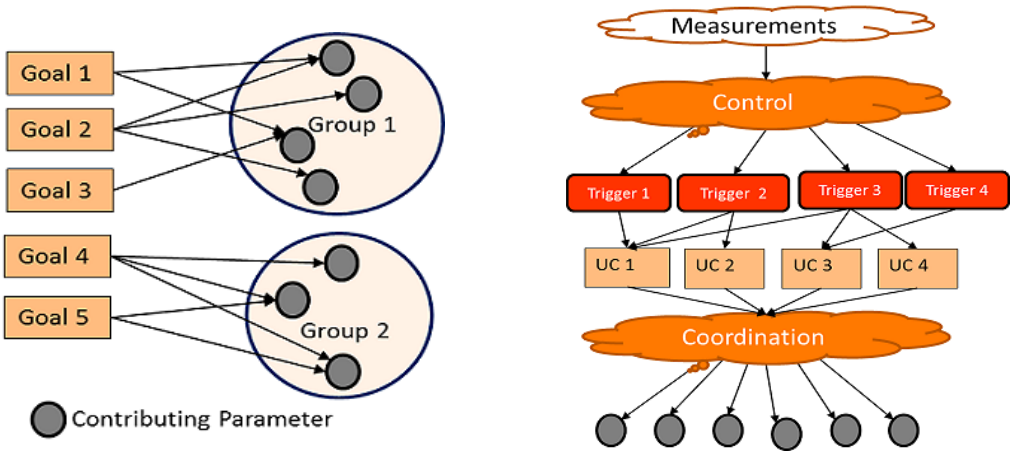
2.3 SON Coordination: The State of the Art

In demonstrating the need and potential benefits of SON integration within the SOCRATES project, some interdependent SFs were simulated together. In one such case, handover performance is compared for MRO both in the presence and in the absence of self-organized admission control [22]. Results show, as would be expected, that the admission control parameter optimization algorithm considerably improves the HO performance, since admission control blocks the would-be cell-edge users. In doing so, it reduces the amount of calls that would have been dropped prior to or during HO. There is however little negative interaction between the two algorithms [5].

In another study, interaction between a MRO algorithm and a MLB algorithm is simulated with a coordinator that controls the two base algorithms. Results show that by combining the strengths of the two algorithms, the coordinator is able to achieve better performance, that for the individual SFs is in some cases even better than any of two separately [25].

Another study evaluated the effect of HO optimization between macro and Home eNBs [5]. Simulation results do not however show any performance improvement for the scenario of a simplified, trend based, macro HO optimization algorithm, although improvements may be expected in other scenarios.

In general, although significant SON-related work exists, most has focused on individual SFs and little has considered the need to coordinate multiple SFs. There are as such multiple challenges that need to be addressed. The two major SON research projects – SOCRATES and UNIVERSELF – have however provided a sound starting point for further work on the problem. The following sections describe some of the ideas that have been investigated and highlight their limitations.



(a) Functional Parameter Groups. **(b)** Control and Coordination Planes.

Figure 2.9: State of the Art SON Control and Coordination Approaches.

2.3.1 Functional Parameter Groups

In [5] and [16] an idea was presented to classify radio parameters into groups - called functional parameters groups (Fig. 2.9a) - which are created in a way that parameters in any one group contribute to the satisfaction of the same goal(s). It was expected that parameters in one group would not be coupled with the goals of other groups, which would allow for the identification of SFs that can be developed in parallel and that would not need to be coordinated and/or simulated together.

However, it was found that a majority of the parameters fell under a single functional parameter group which would lead to impractical results, as the majority of the algorithms would need to be simulated together. An alternative attempt to reduce the complexity considered the relation between the parameters by introducing a metric interrelation weight (IW). High IWs between parameters would suggest that the associated parameter pair be jointly investigated and optimized. This has however not been demonstrated as an applicable solution to the problem.

2.3.2 Control and Coordination

The authors in [5] and [16] also suggested handling conflicts by introducing a control plane and and a coordination plane as shown in Fig. 2.9b. The control plane would compare the measurements against operator defined thresholds and decides on the activation of triggers while the coordination plane would process the parameter changes proposed by concurrent functionalities before real network parameter adjustments are executed. The authors specify the requirements for the control and coordination planes as well as the challenges including identification of possible conflicts. They also specify the functionality that may be part of the coordinator, among them autognostics, super operator policy functionality, guard functionality against extreme behavior due to SON, arbitration as well as SON

parameter execution. They however leave the details of the specification and implementation of coordination functions to future work.

Efforts to derive the control and coordination framework have been undertaken in [59] where, a flexible, operator policy-based decision coordination is developed. In [60] an experimental system for SON function coordination based on the above framework is presented. It however only considers coordination of instances of a single SF as has been demonstrated for CCO.

A case study for a coordination function has also been implemented in [25] [5] to evaluate the Integration of MROand MLB. The study implements an alignment function that prevents handover optimization from adjusting the hysteresis of a cell that has previously been often overloaded. The SON coordination combines the benefits of both algorithms to significantly improve the call drop rate and to a small extent the HO failure rate. This comes however at the cost of slight increases in number of unsatisfied customers and HO ping-pong rate.

The studies above demonstrate the benefits of a coordination function in integrating multiple SFs. These algorithms are however specific to the SFs under consideration and cannot be generalized to any other SFs. Besides only a few subfunctions of the coordinator have been implemented and not the full functionality. This therefore needs to be studied further to generate the required solutions.

2.3.3 Impact Time

In [61] an Impact-time concept towards coordination of SFs was presented. The concept defines a set of time intervals describing the pair-wise temporal interdependency of SFs. Based on these temporal relations, each SF would be allocated the appropriate time whenever scheduled by a SON Coordinator. Although the solution addresses the concurrency of operation of SFs, it does do not address the coupling of parameters and metrics as is likely to happen for many SFs.

2.3.4 Combination of multiple approaches

A coordination function of a larger management framework is presented in [62]. The solution describes a set of approaches enforced using a coordination block. One such approach has been implemented as a hierarchical structure for SFs with the SFs having different operating time scale that allow the SFs to be separated in time. Similar to the impact time concept in 2.3.3, the published solution does not (at the time of publication) protect the SFs against metric value conflicts as described in section 2.2.3.

The discussion in this section has shown that some solutions have been proposed towards SON coordination but mainly as proposals. Where such proposals have been tested, the solutions are also incomplete and so more approaches need to be evaluated. A possible promising approach is the consideration of the SONs system

as a MAS which can be managed using MAS approaches as have been developed in other scientific fields. The subsequent section briefly discusses MASs especially as applied to a learning environment.

2.4 Multi-Agent Systems

Multi-Agent Systems (MAS) in contrast to Single-Agent Systems (SAS) deal with control problems involving multiple agents acting within a shared environment but constrained by: 1) that the agents have some degree of interaction and 2) that the overall performance of the system depends on the joint behavior of all agents. Given this definition, the operation of multiple SFs in multiple cells in a network results into a real MAS. In effect though, each SF is actually a MAS, since each version of the SF in a cell is in principle an agent. The constraints characterize MAS problems such that [63]:

1. In the event that there is no interaction among the agents, the problem ceases to be a MAS since it can be decomposed into independent SAS problems each solvable by a separate agent.

2. the solution space for a MAS problem grows enormously, often exponential in the number of agents.

3. having multiple interacting agents each adapting state makes the environment non-stationary i.e. parameter and metric values as observed by any one agent continuously change in tandem with peer agents' actions. This continuous variation violates the basic assumption of numerous SAS approaches.

4. owing to the combined activity of the multiple-agents, a small change by one agent can often result in unpredictable changes in the overall behavior of the whole system.

Solutions for managing and coordinating agents in MAS problems can be classified among four major categories as described in the subsequent sections:

- SAS Decomposition
- Single Coordinator
- Peer Modeling
- Concurrent Games

2.4.1 SAS Decomposition

Where the interactions among agents are not strong, the MAS problem may be decomposed into separate SAS problems as shown in Fig 2.10. This is done with the assumption that the optimum solution can still be found despite the interactions or that the suboptimal solution is also appropriate for the application. Owing to

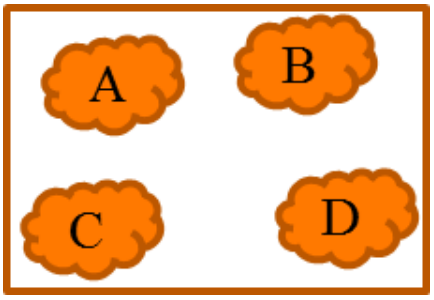


Figure 2.10: Coordination through SAS decomposition.

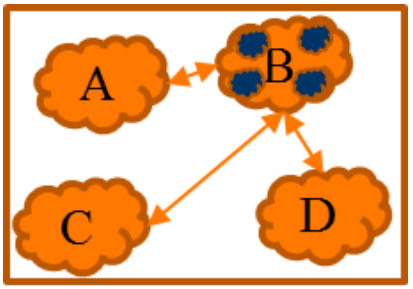


Figure 2.11: Coordination using a single coordinator / through team learning.

the simplicity of dealing with separate SAS problems, many state-of-the-art SON coordination solutions have applied this kind of approach. Among these are the solutions in 2.3.3 and 2.3.4

2.4.2 Single Coordinator /Team Learning

In this case, a single coordinator learns and decides the behavior for a team of agents. In MAS problems involving machine learning, this can also be refereed to as team learning. The single coordinator agent (e.g. agent B in Fig 2.11) makes all the decisions about which and when agents can take actions; evaluates the effects of these actions and decides the appropriate responses to the actions. To accomplish this, the coordinator requires the behavioral models of all team members (the dark bubbles in Fig 2.11) which it uses for coordination. The control and coordination solution in section 2.3.2 [5], is an example of such a coordinator. in the SON environment, this coordinator could be local to the eNB or central to the network. In learning problems, three approaches to team learning may be considered:

1. **Homogeneous Team Learning:** This tries to learn a single agent's behavior for all the agents in the team, even if the agents have different capabilities. By learning the same behavior, the search space is drastically reduced, which is particularly important for problems where the heterogeneous solution space is infeasible to explore. Additionally, for problems that do not require agent specialization for better results, homogeneous team learning would perform better with less complexity. This is applicable for single SFs if such an SF's activity in each cell is considered an agent. It would however not be applicable for different SFs even within a single cell.

2. **Heterogeneous Team Learning:** Here a single learner learns different behaviors for the different members of the team while improving the performance of the team as a whole. This allows agent specialization within the team at the expense of an increased search space. An application of this would be the case of having a single learning agent for all SFs in a cell.

3. **Hybrid Team Learning:** This combines the benefits of both homogeneous and heterogeneous team learning. It divides the team into multiple squads, with each agent belonging to only one squad and each squad then taken as an agent within the system. With all agents in a squad having the same behavior with squads having similar or differing behaviors, it applies homogeneous Team Learning within the squad and heterogeneous Team Learning among squads to achieve specialized characteristics. This would be an appropriate candidate for system-wide learning where all agents for the same SF but in different cells can be taken as one squad applying Homogeneous learning. The entire system would then be a set of squads applying heterogeneous learning among the SFs.

The advantages of team learning are that with a single coordinator, it can utilize the better understood SAS techniques with good convergence and stability characteristics [64] and that it tries to improve the performance of the entire team and not just that of a single agent. However, team learning suffers from scalability challenges on three fronts:

1. In an environment where agents are not all implemented at once, as is the case with SON, the coordinator will require to be revised and/or reimplemented each time a new agent is added. This is because, before a new SF is added, the coordinator does not have a model of that SF and thus needs to be extended with the model of that new SF.

2. The number of possible states that the team can have increases exponentially with the increasing number of agents. This can make it infeasible to maintain the state-value function or it would extremely slow-down the learning process.

3. Its centralized nature implies that the single learning agent needs to have information from all agents of the system in order to make a decision and that all the computational resources must be concentrated at a single place.

2.4.3 Peer Modeling

In peer modeling, each agent focuses on optimizing its objective but models the behavior of its peers so as to account for their actions (Fig 2.12). Using the models (the dark bubbles in Fig 2.12), the agent evaluates its actions and determines the effects that such actions would have on the peers. It then predicts how peers are also likely to behave in response to its actions. Two kinds of agents are observed - cooperative and competitive agents.

A cooperative agent ensures to select actions that not only maximize its benefits but that have the least possible negative effects on its peers. On the other hand a competitive agent focuses only on maximizing its objective with the expectation that each of the other agents is doing exactly the same for its respective objectives.

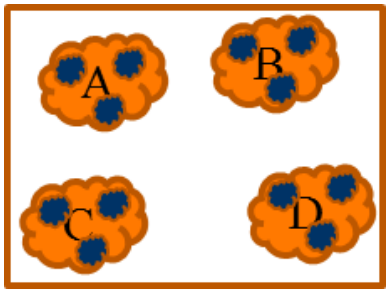


Figure 2.12: Coordination through peer modeling.

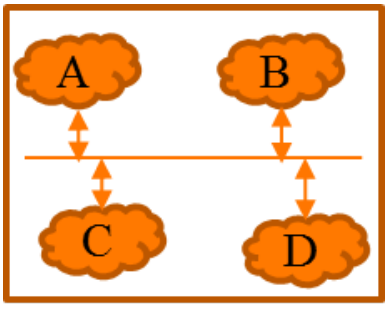


Figure 2.13: Coordination as multi-agent concurrent games.

The challenge however is that peer modeling would also suffer from scalability challenges in the SON environment since models in all SFs must be updated each time a new SF is added to the system. Besides such models are very complex owing to the complexity of the individual SON functions. As such the modeling processes would make each SF very complex. In effect each SF will be as complex as the heterogeneous team learner in Fig 2.11. In concurrent games, multiple learners try to partly solve the MAS problem, especially where some decomposition is possible and where each sub-problem can to some degree be independently solved [65]. CGs reduce the search space and the associated computational complexity of the agents by projecting the large team-wide search space onto smaller separate search spaces. In a learning environment where it is refereed to as concurrent learning, with each agent free to learn individually, heterogeneity or homogeneity becomes an emergent behavior of the system.

However, learning is more difficult in concurrent learning because of the presence of multiple learners. Each learner interacts with the environment and tries to learn the behavior that improves its performance. At the same time, other learners are also co-learning and co-adapting their behavior. Each change by any agent can make the assumptions of other learning agents obsolete and ruin their learned behavior. This makes the whole environment non-stationary and the learning agents can at best try to keep track of these changes in the environment and their optimal behavior [63]. Basing on the behavior of the agents within the CG, concurrent learning may be categorized as cooperative games, competitive games or a mixture of the two.

2.4.3.1 Cooperative Games

Fully cooperative games utilize global reward to divide the reinforcement equally among all the learning agents. All agents have the same goal to maximize the common utility. With differing utilities, a further coordination structure is required to decompose observed rewards into the different utilities.

2.4.4 Concurrent Games (CGs)

Also where no single utility exists, cooperation may also be realized through the sharing of information during the optimization process as shown in Fig. 2.13. In this case, as independent entities and without modeling peers, agents communicate observed effects to each other. In effect, after taking an action, an agent is informed of the actual effects of its action as observed by the peers instead of trying to predict the effects from the built-in internal models as in the case Peer Modeling. In this case, each coordination episode becomes a Concurrent Cooperative Game in which the agents are in competition with each other for the shared parameter or metric but where they are willing to cooperate on what the best compromise value should be.

2.4.4.1 Competitive and Mixed Games

In competitive games agents compete against each other and the reward of one agent acts as a penalty for the other agents i.e. only one wins and the others lose. In that case agents learn to avoid losing strategies and to chose the winning strategies. Such games would be inapplicable in SON where we require all SFs to win. Mixed games are the ones, which are neither fully cooperative nor fully competitive. These may be applicable to SON but the degree of competition would have to be developed.

Given its potential for a fully distributed solution that would easily scale, Concurrent Cooperative Games (CCG) have been considered to be at the core of the coordination solutions presented in this thesis. A detailed discussion of the implementation and validation is given in Chapter 6.

2.5 Summary

As preparation for the detailed discussion of the thesis contributions, this chapter has reviewed the fundamental literature that relates to the rest of the thesis. We started with a steep review of the most important concepts related to the LTE radio system focusing mainly on understanding the available radio resources, how they are allocated and they can maximized. We then discussed SON related literature to date delving into some example SFs that have been developed and discussing the generic approaches through which SFs can be designed. We also discussed the coordination problem not only to clearly map out the scope of the work in this thesis but to also appreciate why it needs to be studied. The last section discussed multi-agent systems highlighting the different generic approaches that can be employed to deal with multi-agent control problems.

Chapter 3

LTE Network Simulator

Contents

Owing to the dynamic nature of the cellular radio environment, SON coordination ideas can not be fully validated analytically; so that performance is best evaluated using system-level simulations. In this case, the system-level simulator, in contrast to a link-level simulator, does not focus on the simulation of all link-level procedures, but abstracts such procedures so as to focus on the end-end performance. Such a simulator models in as much abstract form as possible: the link-level procedures and processes; the trigger and outcome events; as well as the protocols involved in setting up, managing and tearing down data sessions. In this thesis, all

the proposed ideas and algorithms have been evaluated using studies simulating the operation of such ideas in an LTE network. The system-level LTE network simulator used is based on the C++ software libraries provided by Alcatel-Lucent Bell Labs, Germany (ALU) and the Institute of Communication Networks and Computer Engineering at the University of Stuttgart, Germany (IKR) [66]. The libraries enable simulation of the down-link radio environment of a 3GPP-compliant LTE radio network.

For the work in this thesis, the simulator was extended so as to implement the proposed Q-Learning algorithms and the coordination approaches. The basic features of the simulator especially those critical for radio network modeling of SFs under study are explained in the following sections.

3.1 Simulator Structure

This section describes the end-end organization of the simulation environment. We briefly describe the fundamental components of the simulator as provided by ALU and IKR. We then describe the extra novelty required to implement our SON solutions and their coordination. In both cases the focus is on the high-level logic of the components and not on the specific details of each of the components.

3.1.1 LTE RAT Simulator

The high-level design of the base simulator as provided by ALU can be described by the Universal Modeling Language (UML) model in Fig. 3.1. The base simulator implements a factory class whose object creates all the necessary objects for the simulation. For each simulation functionality, different versions (classes) of the function can be provided so that based on the input simulation configuration, the factory loads the required object at runtime from the object pool. For example, if we defined an omni-directional BS, a tri-sector BS and a 6-sector BS, we can select to run a simulation with any of the three. The desired BS kind is selected at the start of the simulation and will be dynamically created by the factory from the BS object pool. For the LTE RAT, the basic factory is extended so that it is able to create objects with characteristic LTE behaviors.

The factory creates a playground and a desired Radio Access Technology (RAT), in this case the LTE RAT. It then attaches the RAT to the playground. Mobile groups each of which is expected to have User Equipments (UEs) (or simply mobiles) with the same characteristics are also created and attached to the playground. The factory then creates the different kinds of UEs and Base Stations (BSs), attaching UEs to the mobile groups and BSs to the RAT. Each BS is then populated with the right transceivers e.g. 3 LTE transceivers in the case of the tri-sector LTE BSs used in the studies in this thesis. Characteristic models are also added to each of the devices as needed. For example, handover processing, mobility and traffic models are added to the UEs while resource allocation (or

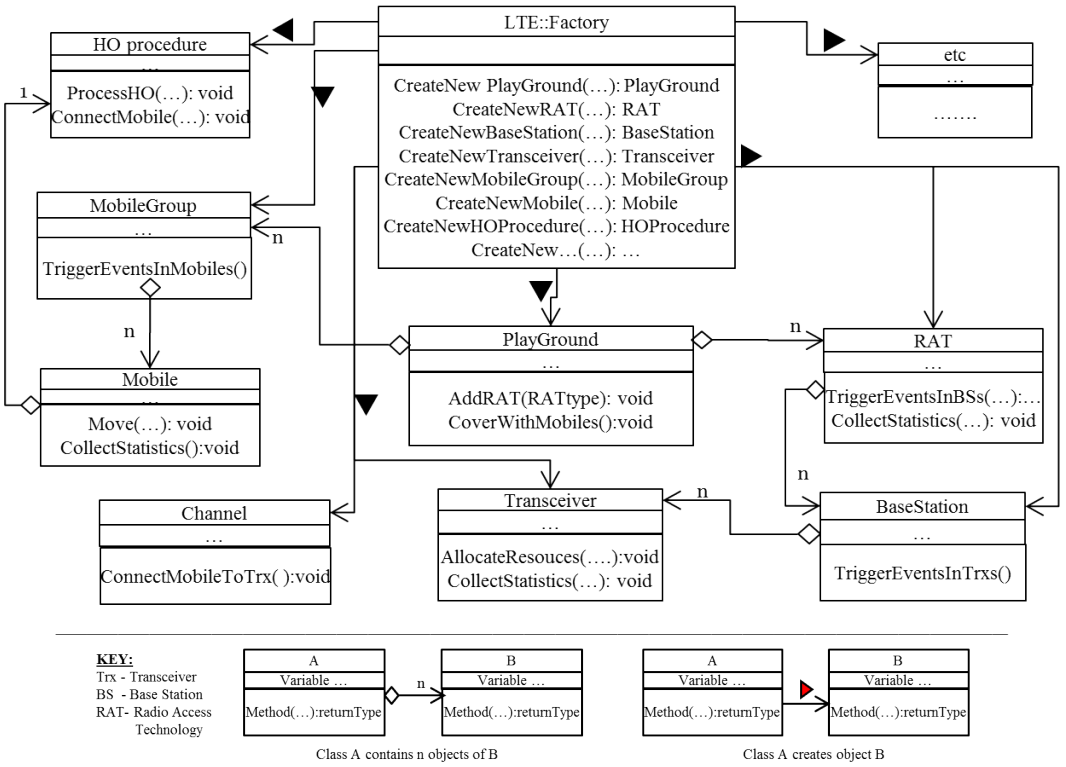


Figure 3.1: UML model of the LTE down-link system level simulator.

scheduling) is added to the transceivers. After all devices are added, channels that connect UEs to the transceivers(cells) are created with the specific characteristics of the respective RAT.

The simulator also implements general-purpose simulation tools from IKR that can be called by any object. These can be used e.g. to initialize the objects, to model probabilistic and statistical distributions or for deriving descriptive statistics on general metrics like user distribution or throughput.

3.1.2 SON Extension for the Simulator

To implement the SFs and coordination, the extended Simulator described by Fig. 3.2 is used. The extension provides for independent classes that define functionality for SFs. For example, as in the figure, separate classes are added for each of the SFs: HO optimization(MRO) and load balancing (MLB). For each considered SF, a local instance of the SF object is added to each cell to ensure that each cell is able to maintain local state of the optimization.

A separate class called *RLagent*, that defines a generic Q-learning agent is added so that each learning-based SF can instantiate its own learning agent as an object of *RLagent*. For distributed learning, such an agent would be local at each cell while for centralized learning, it is added to the RAT to simulate a situation where the agent is shared among all cells. In a real network, such a centralized agent

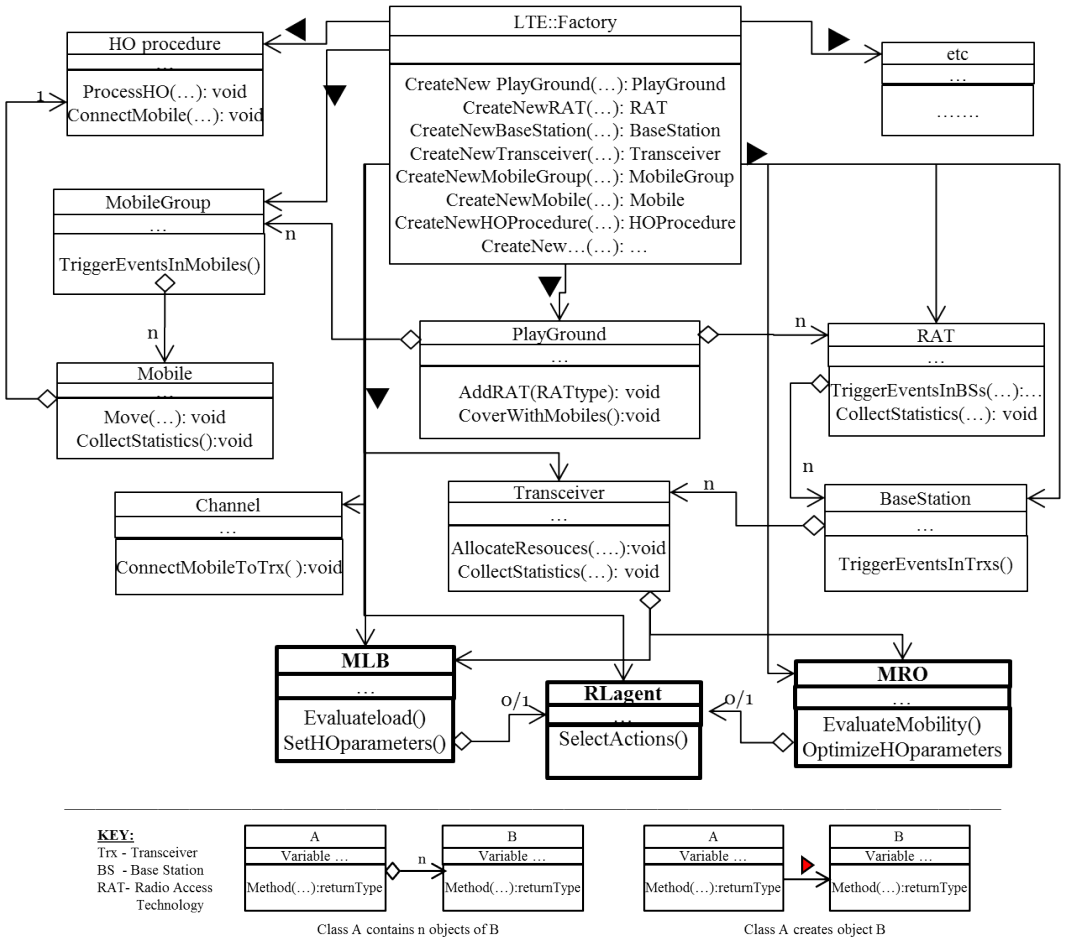


Figure 3.2: The LTE system simulator with SON extensions.

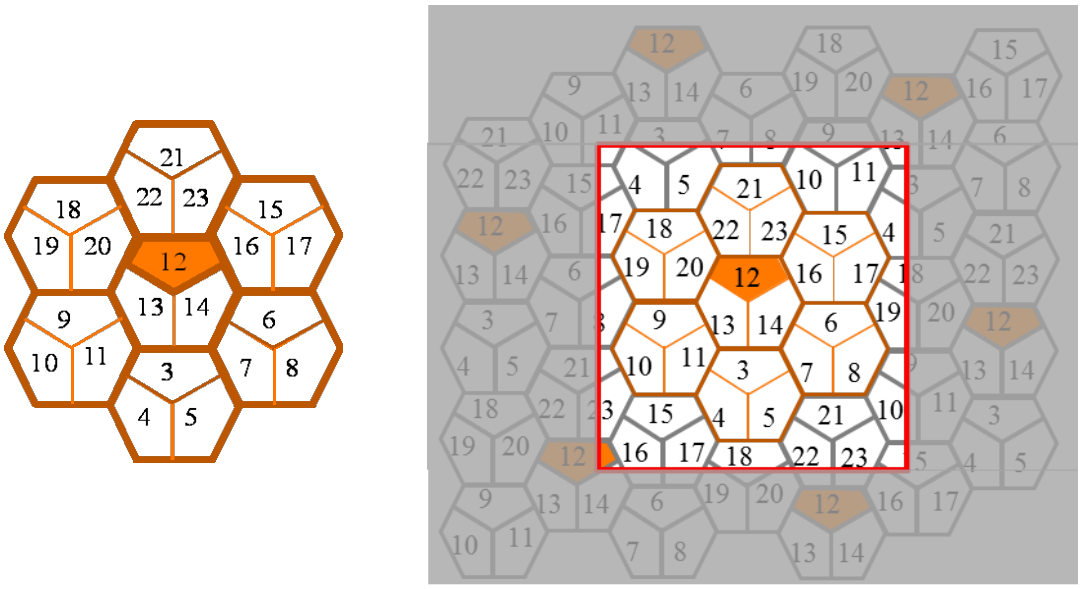
could be placed in the LTE serving Gateway (SGW) for the cells under a single SGW.

For the studies in this thesis, we consider only distributed coordination approaches that do not have a specific coordinator, be it as a central agent in the network or as separate agents in the cells. As such, no specific coordination class is added but where necessary the required logic is added to the existing classes. For example any required global properties e.g. allocation of identities or clustering of cells would be a part of the RAT while individual coordination logic between SFs and cells would be added to the transceivers and the SFs themselves.

After all objects and instances are created, the simulation is initialized to run a specific number of batches each lasting a specified batch period. The detailed processing of the simulation and the batches is described in 3.5.

3.1.3 Network Structure

The basic cellular network used for our evaluations consists of 7 LTE base stations (eNBs) as shown in Fig. 3.3a. Each eNB operates 3 cells, where the term cell refers



(a) Hexagonal grid network. **(b)** Effective network with wraparound and playground.

Figure 3.3: Simulation network structure with and without wraparound.

to the coverage area of a single radio transceiver (TRx). The eNBs are deployed in a regular hexagonal structure, such that one center eNB is surrounded by one layer of eNBs with an inter-site distance of 500 m. In the foregoing sections, reference is made to a center cell which in effect is cell 12 as highlighted in Fig. 3.3a

3.1.4 Wrap-around

A wrap-around is implemented for better interference calculation and to avoid border effects due to discontinuity of coverage. This replicates the simulation area six times around the original area such that the original area is in the center and the replicates form a layer around it as shown in Fig. 3.3b. However, movement of the UEs is restricted to the area within the playground which is big enough to encircle the desired coverage area but not so big to encircle the replicates. Effectively, the replicates are used only for calculation purposes to ensure that as the UEs gets to the edge of the playground it can still calculate interference accurately. When it reaches the playground boarder it has to turn direction to ensure that further movement is still with in the playground. Other than that, the UEs would either continuously move away from the study network or circle around in one path throughout the simulation.

3.2 Radio and Propagation Models

In wireless communications, the received power depends not only on the transmit power, but also on a number of environmental factors and device characteristics.

For useful simulation studies, accurate propagation models for the calculation of received power are required, especially as all other system performance metrics like Signal to Interference and Noise Ratio (SINR) , load, Radio link failures, etc are calculated based on the received power. The relation between transmit power P_{Tx} and received power P_{Rx} (both in dBm) is:

$$P_{Rx} = P_{Tx} - PL - L_{FS} + G_{Ant} + G_{Dir} + G_{Mob} \tag{3.1}$$

where PL in dB is the pathloss; L_{FS} in dB is the slow fading also known as shadow or large scale fading; G_{Ant} in dBi is the maximum antenna gain; while G_{Mob} also in dBi is the mobile's antenna gain and G_{Dir} in dB is the directional gain of transmitter antenna depending upon the position of the receiver with respect to the transmitting antenna. Equation 3.1 ignores fast fading specifically because the effects studied in SON are at such large time scales that the effects of fast fading are assumed to be averaged out. The calculation of the individual terms in equation 3.1 are described in the following sections

3.2.1 Signal Loses and Fading

Fading is the variation in the attenuation, experienced by a signal over a propagation medium. Typically modeled as a random process, fading may vary with time, radio frequency, or the geographical location of the transmitter and receiver. The total signal degradation experienced is a combination of the pathloss and fading.

3.2.1.1 Pathloss

Pathloss (PL) defines the degradation in power with respect to the distance between the transmitter and the receiver. For a distance of d meters between the transmitter and the receiver, 3GPP defines PL for LTE simulation studies as [67]:

$$PL = 128.1 + 37.6xlog(d) \tag{3.2}$$

with the carrier frequency assumed to be 2GHz and the Base Station height above the average rooftop assumed to be 15 meters.

3.2.1.2 Shadow Fading

Shadow fading is caused by large obstacles, e.g. buildings or hills, that obscure the line-of-sight signal between the transmitter and the receiver. The resulting amplitude variations are often modeled as a log-normal distribution with an environment-dependent standard deviation [68][69]. Shadow fading exhibits a slow variation, such that, for moving users, the successive fading values are correlated. The normalized autocorrelation between two successive values is given as [70]:

$$R(\Delta d) = e^{-\frac{\Delta d}{d_{cor}} ln2} \tag{3.3}$$

where, Δd is the distance between the two mobile positions and d_{cor} is the environment dependent decorrelation distance in meters. For example, if the shadow fading value at position D_1 is L_{FS1}. At the next position D_2, which is Δd meters away from D_1, the shadow fading value L_{FS2} is normally distributed with mean $R(\Delta d) \cdot L_{FS1}$ and variance $(1 - R(\Delta d)^2) \cdot \sigma^2$, where σ is the standard deviation.

3.2.2 Antenna Gains

All BS transceivers in the simulator utilize directional antennas. These allow transmission of signals in a particular direction with higher gain thereby reducing inter-cell interference in the other directions. The maximum antenna gain G_{Ant}, is the gain of the directional antenna relative to that of an isotropic radiator. The directional gain G_{Dir}, is the relative strength of the radiated power at a particular angle with respect to antenna bore-sight (axis of maximum gain). This depends on the vertical and horizontal location of the receiver from the transmitter antenna.

Our studies utilize the 3D antenna model of equation 3.4 defined by 3GPP [71]. The model includes both horizontal and vertical antenna radiation patterns with the gain at any position calculated as the sum of the two patterns.

$$G_{Dir} = G(\phi, \theta) = min\left\{-[G_H(\phi) + G_V(\theta)], FBR\right\} \tag{3.4}$$

$G_H(\phi)$ and $G_V(\theta)$ are respectively the horizontal and vertical antenna gains in dB while ϕ and θ are respectively the angles in the horizontal and vertical directions between the antenna's bore-sight and the UE's direction. FBR is the front to back ratio or the backward attenuation. It defines the ratio between the power of an antenna in the main lobe to the power in the back lobe. Both radiation patterns as defined in [71] are normalized to the maximum antenna gain so that the maximum value that $G_H(\phi)$ and $G_V(\theta)$ can have is 0 dB. As such, the directional gain decreases as the mobile moves away from the bore-sight.

All the UEs in the simulations use omni directional antennas. In comparison to directional antennas, these radiate with an equal gain G_{Mob} in all directions. In particular a Gain of $G_{Mob} = 0$ dBi is assumed.

3.2.3 Signal to Interference plus Noise Ratio

Each UE receives signals from its serving cell as well as from other cells that use the same frequency resources. The received power is thus the combination of the signals from the serving cell and the interfering cells. The ability of the receiver to correctly interpret the desired signal from its serving cell depends on the Signal to Interference plus Noise Ratio (SINR). The SINR is the ratio of the power in the desired signal to the sum power of all interfering signals and the local noise in the receiver.

Assume that in a network with N cells (transceivers), a UE u connected to a cell c receives power P_c watts from its serving cell and P_i watts from the i'th interfering cell. If the UE has thermal noise power P_{Noise} watts, its SINR in dB will be:

$$SINR(dB) = 10.log_{10}\left(\frac{P_c}{\sum_{\forall i \in N, i \neq c} P_i + P_{Noise}}\right) \tag{3.5}$$

The thermal noise in dBm can be calculated for a UE with a receiver noise figure of NF_{Rx} dB and a noise band-width of N_{BW} Hz as [72]

$$P_Noise(dBm) = -174 + 10.log_10(N_BW) + NF_{Rx} \tag{3.6}$$

where, 174 is the thermal noise density in dBm/Hz.

3.3 Resource Management

Each cell manages allocation of resources (the OFDM frequency-time resources) to its associated UEs. The cell determines each UE's throughput and allocates the appropriate resources based on demand and the applied scheduling policies.

3.3.1 User Throughput

The user throughput, Th, defined as the maximum transmission data rate that the user can achieve on a given channel depends on the user's SINR . The upper bound expressed in bits per second (bps) for a given bandwidth BW Hz can be calculated from the Shannon formula as:

$$Th(bps) = BW.log_2(1 + SINR) \tag{3.7}$$

This upper bound assumes availability of perfect Modulation and Coding Schemes (MCS). Using realistic MCS that are available for LTE, an approximate function can also be developed for realistic throughput calculations [73][74]. In this thesis for a given SINR S, we use the approximate function in [74] that estimates Th_{symbol} the number of bits that can carried per symbol using the realistic MCS as:

$$Th_{sym} = \begin{cases} 0; & S < 7.04 \\ -0.0001S^3 + 0.0074S^2 + 0.1397S + 0.6218; & -7.04 < S < 20.2 \\ Th_{sym}(20.2); & S > 20.2 \end{cases} \tag{3.8}$$

In LTE users are allocated resources in terms of the number of Physical Resource Blocks (PRBs). Given that each PRB has 7 symbols (as discussed in 2.1.2), the

achievable throughput per PRB Th_{PRB} is:

$$Th_{PRB}(bps) = 7xTh_{sym} \tag{3.9}$$

3.3.2 Resource Allocation and System Load

We consider that the network applies the Guaranteed Bit rate (GBR) traffic model for each user, wherein the user must always be allocated resources that ensure it achieves the specified rate. Then the number of PRBs, N_{PRB}, per scheduling interval T_s required for data transmission is the ratio of the total data to be transmitted in the interval T_s to the achievable rate per PRB i.e.

$$N_{PRB} = \frac{GBR \cdot T_s}{Th_{PRB}} \tag{3.10}$$

LTE defines different system bandwidths B_{sys} as shown in table 2.2, where B_{sys} defines the total spectrum available in the cell. The offered load ρ can be defined as the ratio of the required number of PRBs to the total number of available PRBs. Given the PRB bandwidth B_{PRB} of 180 KHz, the number of PRBs for a given system bandwidth will be constant, with the corresponding offered load given as:

$$\rho = \frac{\sum N_{PRB}}{B_{sys}/B_{PRB}} \tag{3.11}$$

Cell overload occurs when the offered load ρ exceeds a preset threshold ρ_{max} for the given cell bandwidth. Accordingly, ρ can be greater than 1, representing the case when the total required PRBs exceed the maximum PRBs available within B_{sys}.

The actual distribution of resources to users depends on the applied scheduler. In these studies, because we have assumed GBR traffic but with all UEs belonging to one class, the allocation is influenced by the order of selection and allocation. We use the simple round robin selection that continuously allocates resources to users ensuring that each achieves the desired GBR before allocating the next user. The scheduler does this either until all resources are exhausted or until all users are allocated.

3.4 Mobility and Handover Management

For accurate results, simulations require that user behavior is modeled in a way that mimics their realistic behaviors in a real environment. This section describes the modeling of user movement and the handover procedures when users move from one cell to another. In particular it describes the mobility model, the handover procedure and related parameters as well as the modeling of handover events.

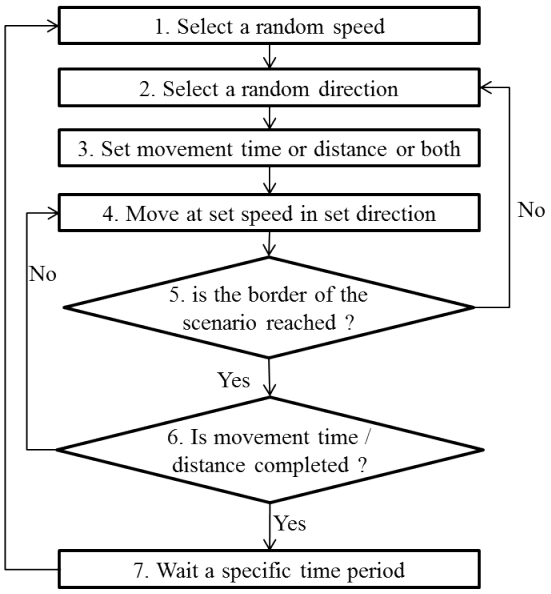


Figure 3.4: Sequence of events for the Random Walk mobility model.

3.4.1 Mobility Model

A mobility model characterizes the patterns of mobile users, describing how a user's location, velocity and acceleration change over time. In this thesis, the random walk mobility model is used. In the random walk model, a UE selects a random direction which it follows until some control event is fulfilled. When that happens, the UE stops for a specified amount of time before repeating the process. The general algorithm can be broken down into the seven steps in Fig. 3.4 [75].

In these studies, the random selection of speed is continuous even while the UE is moving. This is intended to model the fact that UEs never move at perfectly constant velocities even for very short intervals. Instead the actual velocity at any instant differs by a small variation from a generic desired velocity. Consequently, at each snapshot, the velocity is allowed to increase or reduce by some small value ($\leq 5\%$).

The selection of the direction in step 2 follows a uniform distribution over all directions. In step 3, the UE is not time limited but moves until it reaches the border of the playground. When this happens, it turns in a new randomly selected direction and continues to move as before.

3.4.2 Handover Modeling

As UEs move within the network, they need to change serving cells using the handover (HO) process. In this thesis, HOs are triggered according to the A3 condition [76]. Using A3 HOs could be initiated based on either the Reference Signal Received Power (RSRP) or the SINR in the two cells. For HO from a

serving cell s towards a target cell t the A3 event is

$$F_t + O_t^{s,t} - Hys > F_s + O_s^{s,t}. \tag{3.12}$$

F_t,F_s are respectively the RSRP or SINR in dBm of the respective mobile in t and s cells, without any offsets. O_t^s, O_s^t are the Cell Individual Offsets (CIOs) in dB in t and s respectively for HOs between the two. Hys is the hysteresis in dB which is uniform for the serving cell. The Handover Hysteresis (Hys) is specified for LTE to be 0 to 15 dB in steps of 0.5 dB [76].

If A3 is fulfilled for a critical time called Time To Trigger (TTT), the UE is said to have entered the A3 and is ready to initiate a HO. The UE initiates HO by sending a measurement report of the values F_s and F_t. However, before sending the report, F_s and F_t are filtered by a Layer 1 averaging filter (L1) and a Layer 3 Infinite Impulse Response (IIR) filter (L3). *L1* averaging which is not standardized and can be vendor-specific, is implemented as a simple moving average of the most recent values. The averaging window which must be updated every $200ms$ is selected to be 100 samples. This is required to be long enough to average fast fading yet not so long to affect the results of *L3*.

L3 filtering is standardized in the LTE Radio Resource Control (RRC) specifications [76]. For a measurement quantity F (RSRP or SINR), it is defined as:

$$F_n = (1 - 0.5^{K/4})F_{n-1} + 0.5^{K/4}M_n \tag{3.13}$$

where M_n is the latest *L1* measurement, F_n the updated *L3* measurement required for reporting and F_{n-1} the old filtered measurement result, which for F_1, $F_0 = M$. K, the filter coefficient for the quantity F, controls the length of the IIR filter used to filter the effects of fast fading. The possible K values are specified in [76] to be the values from 1 to 18 in steps of 1.

The optimum TTT for any UE also depends on the UE's mobility. The range of TTT values is also specified as the 16 values (0, 40, 64, 80, 100, 128, 160, 256, 320, 480, 512, 640, 1,024, 1,280, 2,560, and 5,120) in ms.

3.4.3 Handover Events

From the foregoing discussion, it is evident that the outcome of any HO depends on Hys, TTT and K. All three parameters have the effects of either initiating HOs early or delaying them. The possible outcomes are either a HO success, a Ping-Pong HO (PP) or a Radio Link Failure (RLF).

3.4.3.1 HO success

A HO is considered successful if a HO Confirm RRC message is delivered to the target cell, following success of four prerequisites [77]:

1. delivery of the HO measurement report to the serving cell;
2. preparation for HO between the serving and target cells over the X2;
3. delivery of HO command RRC message from the serving cell to the UE;
4. successful completion of random access procedure to the target cell

The entire process must last for less than a critical time T304 [76] otherwise the HO is considered to have failed. We model this entire process using a HO-success timer over which the SINR of both the source and target cells must be above the threshold required for all the messaging described above to be successful. If any of the SINR values falls below the threshold, a RLF occurs and the HO fails. The range of values for T304 in seconds is 0.1, 0.2, 0.5, 1, 2, 4, 8 [76], but here the most stringent value 0.1s is used to ensure that we evaluate the cases with the highest possibilities of HO failures.

3.4.3.2 Ping-Pong Handover

A Ping-Pong HO, also refereed to as a Handover oscillation, is registered where for a user u, a successful HO from a cell B to another cell A occurs in a time less than the "PP time" after another successful HO had already occurred from A to B. In our model, the PP Time is set to 5 seconds implying that any back and forth HO in less than 5 seconds is considered a PP.

3.4.3.3 Radio Link Failure

A RLF occurs if the UE SINR stays below a threshold for a duration equivalent to the critical time (T310)[76]. Two kinds of RLFs are modeled - Radio Link Failure due to Early Handover (RLFE) and Radio Link Failure due to Late Handover (RLFL). A RLFL occurs if the RLF condition is fulfilled within the serving cell and, after RLF, the UE connects to another cell different from the serving cell. A RLFE occurs when the RLF condition is fulfilled in the target cell and the UE reconnects to the serving cell after the RLF. In either cases, to model them, a RLF timer (T310) is started if any of the SINR goes below the threshold and the RLF is registered if the SINR does not improve before the time runs out. The range of values for T310 in seconds is 0, 0.05, 0.1, 0.2, 0.5, 1, 2 [76] but the median value of 0.2s is applied in these studies.

3.5 Simulation Model

The section above have described the structure of the simulator and the modeling applied to the individual processes and protocols. After all the necessary objects are created in the simulator with the respective models, the simulation is initialized to run a number of "batches". With in each batch, the simulation follows a

snapshot-based evaluation, where a snapshot of the entire network is taken after every "snapshot interval". This section describes the processes that the simulation follows, starting with the deployment of the different entities followed by a description of the processing of the batches and the individual snapshots.

3.5.1 Deployment

Each simulation starts with the deployment of cells according to the scenario explained in Section 3.1. Mobile devices or UEs are then deployed to generate user traffic and to obtain SINR measurements. The deployment of mobiles follows a random distribution that distributes the total number of users $TotalUsers$, among the cells in such a way that the average number per cell is the ratio of $TotalUsers$ to the number of cells.

For studies involving load balancing, a hot-spot-induced overload is artificially created by deploying an extra number of static users (i.e. UEs with velocity = 0 m/s) in a cell belonging to the center eNB. Specifically, they are deployed in cell 12 (in Figure 3.3a), which is hereafter referred to as the center cell. Fig. 3.5 shows an example distribution of the users after deployment.

3.5.2 Batch Processing

The simulation runs multiple batches each lasting a specified batch period and within which the snapshot are taken as shown in Fig. 3.6. At the beginning of each batch, all users are re-introduced in the network following the random deployment model in 3.5.1. The intention here is to ensure that as many of the

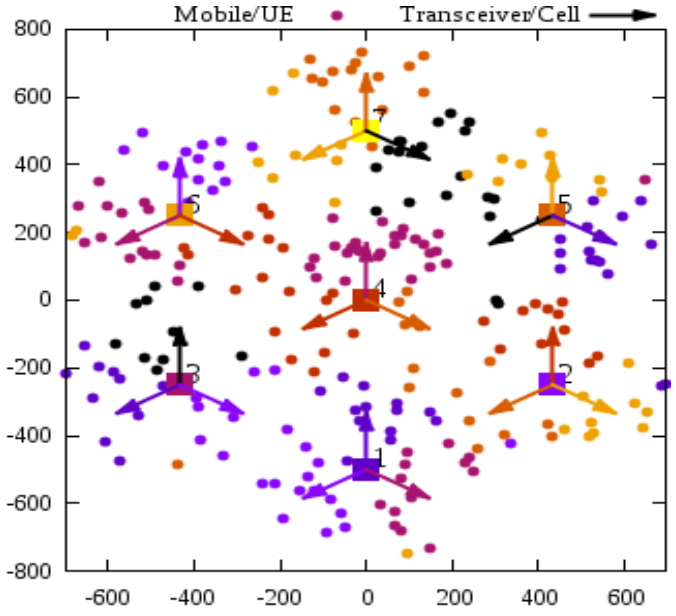


Figure 3.5: Initial Network with deployed users.

possible radio conditions have been experienced by the users as possible. With the random deployment is employed, the assumption is that each is placed at a different location in each batch and that the user follows a different path during the simulation. As such the aggregate scenario-independent simulation results will be indicative of all the possible radio conditions in the network and not specific to only a few limited scenarios.

Within each batch, we desire to evaluate all the events and procedures that each user experiences at each point in time. Such events have varying interval periods including the infinitesimal-time (in reality continuous) variations in the radio signal. This would thus require a simulator that can evaluate events in continuous time, which is not possible. As such, the simulation is periodically triggered to execute a snapshot of the network as indicated in 3.6 i.e. the simulation evaluates the state of the network at periodic points in time. A single snapshot represents the aggregated effects of all the events that would have happened in the network since the last snapshot was taken. For example based on the velocity and direction of movement, each UE is moved to the point where it would have reached if it were continuously moving. Similarly, if the UE was evaluating a HO and its success time has expired during the snapshot interval, the HO is registered at the snapshot.

The snapshot is taken every "snapshot interval" whose length is selected based on the desired accuracy of the events and scenarios being evaluated. For example a simulation that evaluates HO events must use a snapshot interval that is smaller than the shortest HO events (HO success at 60 ms) while a simulation that evaluates user throughput without concern for HO events can consider intervals as longs as 1-2 s. In these studies, the snapshot interval is selected to be 50 ms to ensure that the necessary events are accurately registered. In each snapshot, all events ranging from variation of radio signals to movement of users, allocation of resources and to collection of (quasi-) instantaneous statistics should be scheduled. The detailed processing of these events is described in the next section.

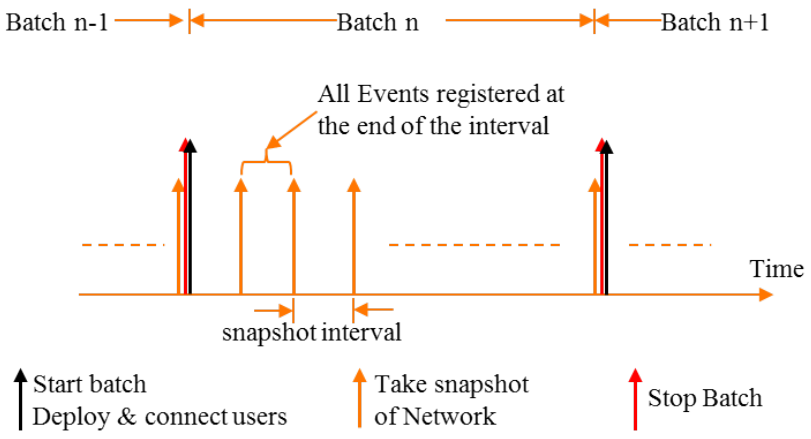


Figure 3.6: Batch and snapshot processing during the simulation.

3.5.3 Snapshot Processing

The processing of each snapshot runs though an open- event loop containing three major events. All other activities are scheduled within these three events:

1. TriggerInitial: Here the UEs are caused to move and they update their position depending on their mobility as described in section 3.4.1. UEs then measure their received power on the pilot signal from all the cells and perform HO evaluation as described in section 3.4.2. Where a HO event such as a ping pong or RLF occurs, the UE reports such an event to the serving cell and in case of RLF the UE initiates re-connection to the network. Similarly, each UE calculates its aggregate throughput over the previous 1 second period and where the UE achieves less than the expected throughput, it reports an "un-satisfied user" event to the cell.

2. TriggerSchedule: In this step, schedulers in the transceivers are triggered to distribute resources to the UEs based on the updated SINR conditions. In our studies, the round robin scheduling described in 3.3.2 is applied although even then SINR values are used to determine both the resources to be allocated to each selected user and the remaining unallocated resources.

3. TriggerEvaluate: In this part, the necessary evaluations in the UEs, cells and the entire RAT are undertaken. Among such evaluations are the interference and load conditions in the cells and network counter statistics for HO events. SON related evaluations are also done during this period. After accomplishing the resource allocation, each transceiver then evaluates the events for which SON could be triggered and where needed, the appropriate SON algorithm is triggered.

3.5.4 Simulation Parameters

As described in the preceding sections, users are deployed as described in 3.5.1, and simulation processed according to 3.5.2 and 3.5.3. The "BatchPeriod" has to be selected long enough to ensure that for the particular deployment, adequate statistics are obtained for all the necessary parameters. However to obtain deployment-independent statistics, multiple batches are executed, in each redeploying the users. Based on test studies that showed that counter statistics for HO events are stable after at most 80 batches of 200 s each, MRO related results are based on simulations of 120 batches each simulating 200s of operation. Load statistics on the other hand are stable after at most 20 batches and so MLB-only results are based on simulations of 30 batches each simulating 200 s of operation. The major simulation parameters are summarized in Table 3.1.

Table 3.1: Simulation Parameters.

Parameter	value
System bandwidth	10 MHz
Inter-site distance	500 m
Time between snapshots	50 ms
Number of users	240 mobile, 40 static
User velocity	variable, mean = 3, 10, 30, 60 or 120 Kmph
Mobility Model	random walk
Pathloss	$A + B \cdot log_{10}[max(dKm, 0.035)]$; A=128.1 and B=37.6
Shadowing	Standard deviation = 6 dB; Decorrelation distance = 50 m
eNB Tx power	46 dBm
eNB Tx antennas	1 per sector, gain 15 dBi, at height = 32 m
UE receive antennas	1 Omni, gain 2 dBi, at height = 1.5m
Data rate	512 Kbps

3.6 Summary

This chapter has discussed the simulation environment that has been used to evaluate the ideas presented in this thesis. This was selected to be presented before the ideas and proposals since foregoing chapters present different aspects of the solutions with their corresponding results. We have presented the generic structure of the LTE simulator and the extra extensions that have been made to simulate SON algorithms and their coordination. We then presented the underlying cellular radio models in the simulator and the related modeling of the radio resources. We discussed the mobility and handover models as well as the handover events that result from the different control settings of the handover parameters. The last section discussed the snapshot-, event-based simulation model in which we described the simulation structure and how the results are derived for the different SFs.

Chapter 4

Q-Learning Framework for SON Functions

Contents

A number of approaches have been applied in developing SON solutions as discussed in 2.2.2. Of these, Machine Learning (ML) and specifically Reinforcement Learning (RL) offers the best promise since for a specific environment, it allows the system to automatically learn and improve its solutions through experience in that environment. An RL agent interacts with its environment and learns the optimal behavior for particular states in the environment [78]. The learning problem can modeled as a Markov Decision Process (MDP) and solved using Dynamic Programming (DP) techniques.

An MDP is a quadruple $(X; A; P; r)$, where, X and A are respectively the sets of all possible states and actions, $P : X \cdot A \cdot X' \rightarrow [0; 1]$ is the state transition

probability function and $r : X \cdot A \cdot X' \rightarrow \mathcal{R}$ is the reward function. P represents the environmental behavioral model that defines $P(x_t; a_t; x_{t+1})$, the probability of ending in state x_{t+1} when the agent takes action a_t in state x_t. The need for this environmental behavioral model implies that DP techniques would be inapplicable in SON where such models are unavailable. A solution that does not require such models is thus required for SON.

Q-learning (QL) is an RL algorithm that does not require a model of its environment to learn the optimum actions[79]. Instead it learns the abstract model of the environment by learning the best actions for specific states of the environment. This makes QL a good candidate for on-line optimization problems as is required in SON. We have thus selected to implement our SFs as Q-learning problems. This chapter describes our proposed QL framework for SON. It begins with a review of QL highlighting how the QL approach can be mapped to SON problems especially for Self Optimization problems. It then describes the application of QL to two such problems - MRO and MLB.

4.1 Q-Learning in SON

This section discusses Q-learning with a focus on its use in cellular systems and SON. We begin with a brief review of Q-learning and its relation to generic RL followed with a discussion on the policies for selection of actions. We then discuss the possible approaches to learning in a multi-agent environment i.e. cooperative versus distributed learning. We conclude with a discussion of the application of QL in SON highlighting on how QL can be mapped to a general SON problem.

4.1.1 Q-Learning (QL)

Q-Learning (QL) is a model-free RL algorithm which, using the Temporal Difference (TD) method can solve learning problems even without models. QL promises a realistic way of implementing SON and other automation functions in cellular systems especially where the dynamic behavior of the problem can not be easily synthesized. Instead of the MDP quadruple, QL is a triple $(X; A; r)$, where as before, X is the set of all possible states, A is the set of all possible actions and $r : X \cdot A \cdot X' \rightarrow \mathcal{R}$ is the reward function. The difference is that we no longer require the environment model which for the MDP was the state transition probability.

4.1.1.1 Model free RL with QL

Assume that a learning agent (Q-learner) needs to determine the optimum action for a problem. As shown by figure 4.1, assume that at time t when the world is in state $x_t \in X$, the Q-learner chooses an action $a_t \in A$. Based on the effect of this action the Q-learner receives an immediate reward $r_t \in \mathcal{R}$ and the environment

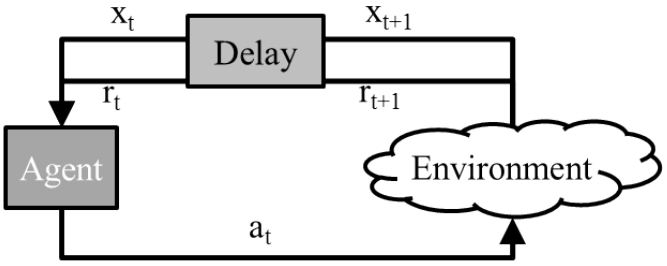


Figure 4.1: the QL process.

undergoes a transition into a new state $x_{t+1} \in X$. The objective of the learner is to choose actions that maximize the discounted cumulative rewards over time. More precisely, let γ be a specified discount factor in the range [0 1). The total discounted return (or simply return) received by the learner starting at time t is given by :

$$r(t) = r_t + \gamma \cdot r_{t+1} + \gamma^2 \cdot r_{t+2} + \gamma^3 \cdot r_{t+3} + ...\gamma^m \cdot r_{t+m} + ... \tag{4.1}$$

γ is used to balance immediate rewards against futuristic rewards such that the closer γ is to 1 the greater the weight of future reinforcements. In general the individual rewards at each time step is not deterministic, so that the value of a state x under the policy π is given by:

$$\begin{aligned} V^{\pi}(x_t) &= E[r(x_t) + \gamma \cdot r(x_{t+1}) + \gamma^2 \cdot r(x_{t+2}) + ... + \gamma^m r(x_{t+m}) + ...] \\ &= E[r_t + \gamma r_{t+1} + \gamma^2 \cdot r_{t+2} + \gamma^3 \cdot r_{t+3} + ... + \gamma^m \cdot r_{t+m} + ...] \\ &= E[r_t + \sum_{m=1}^{\inf} \gamma^m \cdot r_{t+m}] \\ &= E[r_t + \gamma \cdot \sum_{m=0}^{\inf} \gamma^m \cdot r_{t+1+m}] \end{aligned} \tag{4.2}$$

This is the Bellman Equation also known as the equation of Dynamic Programming. It can be re-written as:

$$V^{\pi}(x_t) = E[r(x_t) + \gamma \cdot V^{\pi}(x_{t+1})] \tag{4.3}$$

Equation 4.3 represents the value of the state x at time t as dictated by the policy π. The objective is to find π, the policy of selecting actions, so that the expected value of the return is maximized. The policy in this case represents a rule (or set of rules) for selecting actions. For stationary policies, those that select actions based only on the current state, we can define a value function (the Q-value) for every state action pair. For a policy π, the Q-value ($Q^{\pi}(x, a)$) is the expected total discounted return received when starting in state x and following the policy π thereafter. Since the policy selects actions in the different states, the Q-value is similar to the value function in 4.2. However, instead of the value of the state (which is the value of the best action), the Q-value considers the value of each

action in each state. Thus,

$$\begin{aligned} Q^{\pi}(x_t, a_t) =& E[r(x_t, a_t) + \gamma \cdot r(x_{t+1}, a_{t+1}) + \gamma^2 \cdot r(x_{t+2}, a_{t+2}) \\ &+ \gamma^3 \cdot r(x_{t+3}, a_{t+3}) + ... + \gamma^m \cdot r(x_{t+m}, a_{t+m}) + ...] \end{aligned} \tag{4.4}$$

The corresponding Bellman equation for the Q-value is

$$Q^{\pi}(x_t, a_t) = E[r(x_t, a_t) + \gamma \cdot Q^{\pi}(x_{t+1}, a_{t+1})] \tag{4.5}$$

For the optimal policy π^*, the Q-value of action a in state x, $Q^{\pi^*}(x, a)$ is the expected total discounted return received when action a is taken in state x and after that continuing with the optimal policy π^*, i.e.

$$Q^{\pi^*}(x_t, a_t) = E[r(x_t, a_t) + \gamma \cdot Q^{\pi^*}(x_{t+1}, a_{t+1})] \tag{4.6}$$

The optimal Q-value defines the expected rewards of taking action a in state x when following the optimal policy.

$$Q^{\pi^*}(x_t, a_t) = \max_{\pi} Q^{\pi}(x_t, a_t) = \max_{a_t} Q(x_t, a_t) \tag{4.7}$$

This can be calculated iteratively over the sequence of actions as:

$$Q^{\pi^*}(x_t, a_t) = E\left[r(x_t, a_t) + \gamma \cdot \max_{a_{t+1}}\{Q^{\pi}(x_{t+1}, a_{t+1})\}\right] \tag{4.8}$$

4.1.1.2 The QL Algorithm

The Q-learning algorithm works by maintaining an estimate of the Q function, denoted here by $\hat{Q}$, and adjusting $\hat{Q}$ values (often just called Q-values) based on actions taken and rewards received. This is done using the TD error $e(x_t, a_t)$ [78] which is the difference between the actual Q-value ($Q(x_t, a_t)$) and its current estimate ($\hat{Q}(x_t, a_t)$), i.e:

$$e(x_t, a_t) = Q(x_t, a_t) - \hat{Q}(x_t, a_t) \tag{4.9}$$

For any actions taken at time t, the algorithm updates the estimate at time t+1 by adding a small portion of the error (difference) to the current estimate as:

$$\begin{aligned} \hat{Q}_{t+1}(x_t, a_t) &= \hat{Q}_t(x_t, a_t) + \alpha_t \cdot e(x_t, a_t) \\ &= \hat{Q}_t(x_t, a_t) + \alpha_t \cdot \left\{Q(x_t, a_t) - \hat{Q}(x_t, a_t)\right\} \\ &= \hat{Q}_t(x_t, a_t) + \alpha_t \cdot \left\{r(x_t, a_t) + \gamma \cdot \max_{a_{t+1}} Q(x_{t+1}, a_{t+1}) - \hat{Q}^*(x_t, a_t)\right\} \\ &= (1 - \alpha_t)\hat{Q}_t(x_t, a_t) + \alpha_t \cdot \left\{r(x_t, a_t) + \gamma \cdot \max_{a_{t+1}} Q(x_{t+1}, a_{t+1})\right\} \end{aligned} \tag{4.10}$$

Algorithm 4.1: The Q-Learning Algorithm

initialize: Q-value estimates as $\hat{Q}(x_t, a_t) := 0 \ \forall x \in X, a \in A$

1. Observe the current state $x = x_t$
2. Select and execute an action a_t
3. Receive the immediate reward r_t
4. Observe the new state $x = x_{t+1}$
5. Update the estimate $\hat{Q}(x_t, a_t)$ according to equation 4.10
6. update time $t \leftarrow t + 1$ and current state $x \leftarrow x_{t+1}$
7. Repeat steps from 2 to 6 **until** the terminal condition is fulfilled

where α in the range $(0, 1]$ is the learning rate parameter that balances new information against previous knowledge. A value of $\alpha = 0$ implies that no learning is undertaken while $\alpha = 1$ means that only the latest information is considered with complete disregard to the old knowledge.

The step-by-step flow of the algorithm can be summarized by the procedure in Algorithm 4.1.

4.1.2 QL Process and the Exploration of Actions

It is important to note that the Q-learning method does not specify what actions the agent should take in each state as it updates its estimates. To find the optimal Q function eventually, however, the agent must try out each action in every state many times. It has been shown [80] that if each state-action pair's Q-value is updated infinitely often, then $\hat{Q}^*$ will converge to Q^* with probability 1 as long as α is reduced to 0 at a suitable rate. Multiple approaches for finding the compromise between exploration and exploitation exist, the most common being ϵ-greedy. In this thesis, we have used the ϵ-first strategy which can be considered a special form of the ϵ-greedy strategy.

4.1.2.1 ϵ-greedy strategy

In this case the policy $\pi(x) = arg\max_a \hat{Q}^*(x; a)$ is only used a part of the time in order to be able to explore the state-action space completely. At each iteration i, the agent chooses to take either a random action $a_i = random$ with a probability ϵ or the optimal action $a_i = \pi(x_i)$ with a probability $1-\epsilon$. At the beginning of the learning ϵ must be huge (near 1), but at the end of the learning, when $\hat{Q}^* \approx Q^*$, the agent should set $\epsilon = 0$ to always use the optimal policy. An appropriate method must be found for reducing ϵ. A constant ϵ has the properties that when ϵ is small, the agent does not explore wide enough at the beginning yet when ϵ is large the agents explores widely even when it has learned enough.

4.1.2.2 ϵ-first strategy

The ϵ-first is in general similar to ϵ-greedy but with a step change between exploration and exploitation. It begins by exploring all the actions without exploiting until all actions have been tested a given number of times. At that points it changes to exploitation of the learned knowledge. We apply the naïve (ϵ, δ) algorithm that explores all the actions N times before exploiting [81]. The solution gives with probability $1 - \delta$ an ϵ-optimal action for each state with

$$N = \frac{2}{\epsilon^2} ln \frac{2n}{\delta}. \tag{4.11}$$

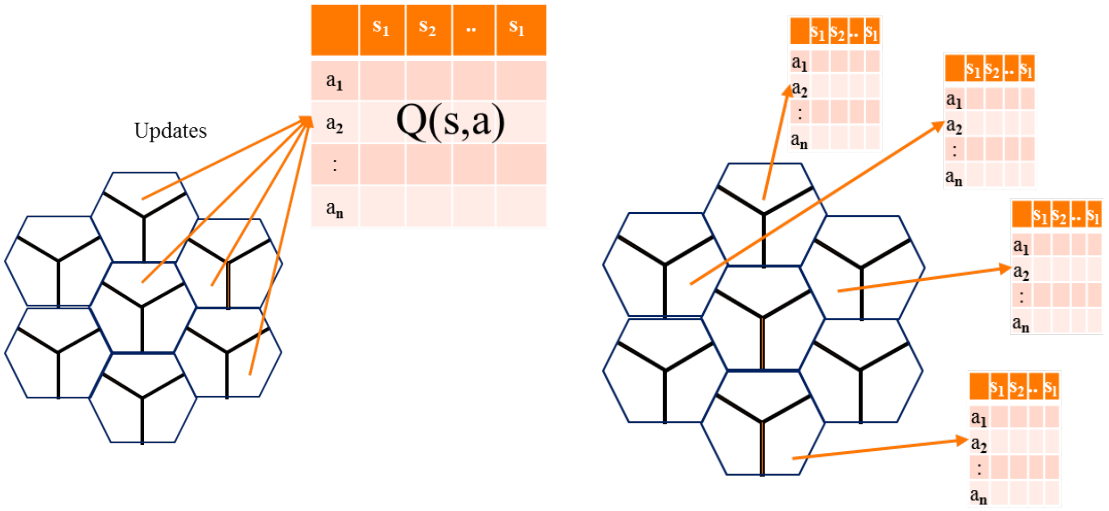
where: n is the number of actions; δ the probability of an action being non-optimal; and ϵ is the greed or desired speed of convergence to optimality. Small ϵ values give more accuracy at the cost of delayed convergence. In this work, we have selected ϵ=0.4 and optimality probability of 80%($\delta = 0.2$) which represent mid ground greed while allowing good accuracy.

4.1.3 Multi Agent Q-Learning

This thesis focuses on Distributed SON solutions where optimization decision are made for each cell individually as opposed to centralized SON that seeks solutions for an entire network as a whole. The QL algorithm as presented so far focused on a single agent learning to optimize a problem. The Distributed SON environment on the other hand is a multi agent system even for a single SF since optimization decisions are made by each cell individually. Consequently, for each SF, each cell is a learning agent and the entire system is a multi agent learning environment. In such a case, it should be decided whether to apply cooperative learning or fully-distributed learning.

4.1.3.1 Cooperative Learning

Assuming for a given multi agent QL problem that an observed state at one agent will at some time be observed by another agent, it makes sense to allow the agents to share their observations and learn a single shared policy instead of learning independent policies. The result is a Cooperative QL problem in which the agents take actions and update a single Q-table. An example of a cooperative QL MAS is the cellular network MAS in Fig.4.2a where each cell represents a QL agent. Besides the benefit of enabling cells to share experiences, Cooperative QL also speeds up the learning processes. Each cell does not necessarily need to experience each state for it to learn the optimum action to take in the state since it is has access to communal policy as learned by the other cells. The challenge here is that all agents have the same Q table (same policy), which may not be optimal for all agents in all states. Having different policies would allow agents to explore the state space differently, enforcing the small differences that are due



(a) Cooperative Learning.

(b) Fully distributed Learning.

Figure 4.2: Distributed learning strategies.

to specific conditions for each cell. Nevertheless, where the QL states are defined in a way that they are fairly consistent across different agents (cells), Cooperative QL can significantly improve the overall performance.

4.1.3.2 Fully-Distributed Learning

The alternative to cooperative QL is distributed QL in Fig.4.2b with, as expected, the reverse merits and demerits. In this case each agent in the MAS learns and updates an independent Q table, which may result in different policies in case of differences environment behaviors observed by the agents.

Implementing QL in a cellular environment even for a single SON function is equivalent to a MAS problem which needs to be treated accordingly. In problems where different cells observe similar states i.e. where states can re-occur from one cell to another, cooperative QL solutions are preferred. This is the case also in the QL problems considered in this thesis, the challenges mentioned above not withstanding.

4.1.4 Application of QL to SON functions

Reinforcement learning and specifically Q-learning has been applied in a number of works on Self-Organization or generally on automation of network functions in cellular networks. In this section we highlight of few of such publications.

4.1.4.1 Coverage and Capacity Optimization

In [82], the authors present an antenna tilt based CCO solution which proved robust to noisy feedback information from mobile users yet remains responsive to the changes in the environment. Their simulation results show that the solution converges to the global optimal settings and that the proposed solution can provide up to 20% performance improvement when compared with another already existing fuzzy-logic-based reinforcement learning approach. In [83], different learning strategies for a Fuzzy Q-Learning based solution for CCO were presented. Results show that the network is able to autonomously optimize system-wide capacity while ensuring good coverage at all points. The solutions are also able to autonomously recover a single eNB failure by adjusting the cell antenna tilt.

4.1.4.2 Interference Management

In [84] a Cooperative Q- Learning Approach for ICIC was proposed. Simulation results showed that due to improved interference management, improvements were realized in system capacity and file transfer times. A distributed Q-learning algorithm for Interference Management in a Femto cell environment was proposed in [85] where the Femto Base Stations learn and adapt their channel selection strategies. The problem was separated into two subproblems, the frist to find the channel allocation through Q-learning in a decentralized way, while the second computes the optimal power allocation. Results showed that femtocells were able to self-organize with only local information and were also able to mitigate their interference towards the macrocell network.

4.1.4.3 Load Balancing

The authors in [86] proposed a fuzzy Q-Learning algorithm to find the optimal set of fuzzy rules in a Fuzzy Logic Controller (FLC) for traffic balancing in GSM-EDGE Radio Access Networks. Undertaking load balancing through modification of Handover Margins (HOMs), their simulation results show that the optimized FLC provides a significant reduction in call blocking.

Authors in [87] proposed several methods for load balancing in an enterprise LTE femtocell scenario. Based on tuning HOMs and femtocell transmit power, they evaluated different strategies implementing FLCs and fuzzy rule-based Q-learning systems (FRLS). Their results showed that performance is better when Q-Learning is applied to transmit power instead of HOMs, since the network is more sensitive to transmit power, and therefore, better solutions can be achieved by some exploration of the algorithm. Instead, when Q-Learning is applied to HOMs, typically only suboptimal solutions are reached.

4.1.4.4 Handover management

The authors in [57] used a self-organizing map (SOM) to allow a femtocell to learn the HO performance in an indoor environment. The Femtocell notes the locations of the indoor environment from where HO requests have occurred and, based on previous experience, decides whether to permit or prohibit these HOs. By prohibiting unnecessary HOs, the algorithm could reduce the total number of HOs by up to 70% while still permitting the necessary HO requests to proceed.

4.1.5 Generalization of QL for SON functions

Section 4.1.4 has showed that a lot of work has been done towards implementing network automation functions as RL or specifically QL agents or systems of agents. This thesis generalizes this approach by considering all SON functions as QL problems some of which are presented in the subsequent sections. A typical SF is a controller that observes its environment, takes an action based on the observations and in many cases receives feedback on the suitability of the actions. This is the same structure as for a typical learning agent depicted in Fig. 4.3. As such we consider each SF to be a Q-learning agent given the appropriate elements.

Essentially, with the radio network acting as the environment for each SF's learning agent, we define the state(s) for which actions are required. These are typically related to the observations or measurements that are supposed to trigger the SFs. For example, since the MLB SF is triggered by cell overload, MLB states could be set according to the degree of overload. Then, given the states, we define the action set as the different possible actions that can be taken in such state. Owing to the need to quantify the quality of the actions, a feedback mechanism in form of a reward system for the actions is added. Then for each action taken, a reward is derived for the SF QL agent from which the agent learns the best actions over multiple interactions with the network. Table 4.1 summarizes how some of the most common SFs can be mapped to the QL framework.

The biggest benefit of the QL framework is that each SF can designed to learn based on all metrics influenced by its actions. As such each SF may be able to learn to minimize its effects on peers metrics. In table 4.1 for example, since it is known that MLB affects MRO, the QL agent for MLB can be designed to learn based on MLB's metrics (load and user dissatisfaction) as well as based on HO

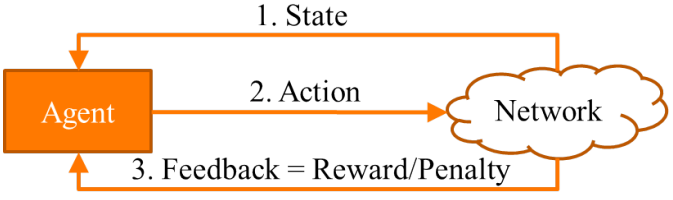


Figure 4.3: Abstract SON Controller is structurally similar to a learning agent.

Table 4.1: Mapping SFs to QL Framework.

SF	**States**	**Solution: Adjust ..**	**Action to learn**	**QL agent rewards**	**QL agent penalizes**
MRO	Mobility state e.g. average UE speed in cell	Hys, TTT	Hys-TTT tuple e.g. (2.5,0.64); ..... (dB,s)	change in no. of RLFs and ping-pong HOs	
MLB	serving & neighbor cell load, user distribution	CIO by δ	sizes of δ e.g. [δ =0.5,1.0, ..]	change in load	HO effects
CCO	serving & neighbor cells' spectral efficiency, Tx Power	Antenna Tilt by δ	sizes of δ e.g. [δ =0,1,2, ...]	change in spectral efficiency	User dissatisfaction
ICIC	serving & neighbor cells' spectral efficiency, Tx Power	Tx Power, spectrum allocation policy	change in or the absolute Tx power, spectrum allocation, ...	Mean throughput, interference level	HO and Load effects
:	:	:	:	:	:

MRO: Mobility Robustness Optimization
MLB: Mobility Load Balancing
CCO: Coverage and Capacity Optimization
ICIC: Inter Cell Interference Coordination
Hys: Hysteresis
TTT : Time to Trigger
CIO : Cell Individual Offset
RLF : Radio Link Failure

metrics (ping pongs and Radio Link failures). In the next sections, we describe the two SFs (MRO and MLB) that have been implemented using this QL framework.

4.2 QL for Mobility Robustness Optimization

One of the major activities in network operations is determining the optimum handover (HO) settings. Hys and TTT are the main control parameters and it is necessary that they are configured according to the prevailing user speeds within the cell or the network. This need to have different settings for different user speeds translates into both a large state-space and a large parameter-space which can not be effectively evaluated manually. In the context of SON, MRO seeks to mitigate this manual operation. This section discusses our proposed approach to MRO using Q-learning.

4.2.1 Mobility Robustness Optimization

For each HO, depending on the Hys-TTT tuple, hereafter called the Trigger point (TP), either a HO success, a Ping-Pong (PP), or a RLF occurs. To optimize

the relative occurrence of these HO events, MRO was proposed in the LTE SON standard [6]. The MRO objectives are [6]: 1) to minimize RLFs that are due to either too early HOs, too late HOs or HO to the wrong cell; and 2) to reduce PPs and unnecessary HOs while achieving objective 1.

4.2.1.1 State of the Art MRO solutions

Multiple studies have been done on MRO. The authors in [18] presented a rule-based HO optimization algorithm that adjusts both the control parameters and the performance targets. Evaluating one metric at a time, while tracking periods of good or bad performance, the algorithm selects the optimization action based on the absolute Hys and TTT values at that time.

The authors in [88] proposed constraining the valid parameter values to the set of Hys-TTT combinations that lie along a diagonal line in a 2-dimensional Hys-TTT grid. They then proposed an algorithm that searches through this set to determine the Optimum Trigger Point (OTP). The obtained value is then applied thereafter regardless of the velocity in the cell. Then in [89], an enhancement to [88] was presented. The enhancement applies a threshold to control the change of parameter search direction so as to improve convergence time. In both cases however, a lot of manual effort would be required if the search space is to be selected for each applicable velocity in the network.

Meanwhile, the authors in [28] removed the restriction on the parameter space in [88] and proposed a local search algorithm for the optimum point.

Then, in another attempt to improve convergence time, the authors in [19] applied soft metrics to reduce the time required to obtain adequate evaluation statistics so as to deduce appropriateness of parameter settings. Instead of relying on the typical HO metrics like ping-pongs and HO failures, the authors rely on statistics of other soft metrics that have demonstrated some correlation with the handover reliability. Examples of such soft metrics are the handover command transmission time or the number of Hybrid Automatic Repeat Request (HARQ) transmissions required to send the handover command to the UE. Based on these statistics, the designed HO algorithm adjusts the target values of the soft metrics that would guarantee the desired HO success and HO failure targets.

4.2.1.2 Proposed Approach for MRO

Virtually all the approaches as reviewed above have mainly relied on expert knowledge control loops to search through the parameter space. These approaches make two fundamental assumptions that are unlikely to hold in real systems:

1. that the mobility profile in the network is static to the extent that a single search is adequate to get the best settings. This is never the case, while

the alternative of re-initiating the parameter search each time the velocity profile changes is also impractical;

2. that, when designing search rules, the designer fully understands the underlying dependence of HO metrics on the control parameters. Besides being prone to error in case of wrong assumptions about this dependence, the required rules would be very complex even with the right dependence model.

To counter these challenges, we propose a solution that does not rely on expert knowledge or rules in selecting the OTP. Instead, it learns the OTPs as would be derived from the perfect dependence model. The solution, QMRO, is a Q-Learning approach that abstracts UE velocities into a finite set of mobility states, so as to learn the OTP for each state. As cells apply different settings (also called actions) in the different states, their observed performance for the state - action pairs are used to update a central learning agent. Subsequent to learning, the cells reference the agent to select the appropriate TP in different observed mobility states.

4.2.2 HO Performance Metrics

Increase in Hys and/or TTT delays HO triggering, subsequently reducing HOs and PPs. Over delayed however, the SINR degrades so much that a RLF occurs, specifically the RLFLs. The reverse happens when the Hys and/or TTT are made so short in a bid to trigger the HO earlier. In that case the HO is made to a cell whose signal is not consistently good that the UE re-initiates a HO back to the original cell resulting into a PP. In the extreme case, the SINR in the new cell is so poor that the UE loses its link before triggering or during the reverse HO to the source. In that case a RLFE is said to have occurred. Consequently, for HO performance, 3 metrics need to be considered i.e. 2 RLF rates and the PP rate.

4.2.2.1 Radio Link Failure Rate (F)

A RLF is assumed to occur if the SINR stays below a threshold for a duration of the critical time (*T310*)[76]. The RLF rate (denoted by F), either due to HOs being triggered too early (F_E) or due to HOs being triggered too late (F_L), is the number of RLF events per second evaluated for the cell or the network.

4.2.2.2 Ping-Pong rate (P)

A PP occurs when a user successfully handed from a cell A to another cell B is handed back from B to A in a time less than the critical time, *PP-Time*. The PP rate (P) can then be defined as the rate of occurrence of PPs per second in the cell or the network. The *PP-Time* is are not standardized and has in this work been set to be approximately equal to the longest TTT (i.e. *PP-Time*= 5 s).

4.2.2.3 HO candidates (NH)

Note should be taken that all rates above are ratios of the events to time and are not normalized to the number of active calls. This is because we require statistics that compare HO performance yet the number of calls include those call sessions for users in the cell center. In that case a cell that has many center users may seem to be performing well if its events are normalized to the number of active calls. For learning purposes however, the rates are normalized to the number of HO candidates (NH) in the cell. This is intended to ensure that all cells use comparable statistics in evaluating their actions. On the network scale where we evaluate the global performance, this condition is not necessary since in that case we consider the same number of users with the same mobility patterns.

The conclusions drawn from the network-wide performance results would be the same whether or not Number of HO Candidates (NH) is considered. This is not the same for short term (quasi instantaneous) statistics. In this case the HO candidates may not actually match with the HO events, since the two may occur in different evaluation periods. Take for example results calculated every 5 seconds and assume the HO candidate is counted the first time the UE sends a measurement report. If an initiated HO succeeds 2 seconds before the statistics are recorded and its ping-pong occurs 2 seconds after the statistics record, the two events get registered in different records yet they are related. In that case, the recorded statistics would be wrong and yet such events can easily occurs for any of the metrics. It is as such better to rely on the event counts or their rates per unit time when evaluating instantaneous statistics.

As stated, when evaluating the effects of actions during the learning phase, the metrics are normalized to the NH in the cell. In reality, the number of real HO candidates (those who are ready for HO) also depends on the two HO parameters i.e. the decision of whether a UE is due for HO or not is governed by the Hys and TTT. This creates a cyclic dependence as Hys and TTT determine NH yet we need NH to evaluate the right Hys and TTT. To avoid the cyclic dependence, we redefine a HO candidate as one who has either initiated a HO or experienced a RLF within the evaluation time interval. Consequently, NH is the sum of all users who have either initiated a HO or experienced a RLF ensuring that within the evaluation time interval, each is counted only once especially where a single user experiences multiple events.

4.2.2.4 HOAP metric

In order to effectively compare trigger points so as to select the best one, a single metric is needed and any comparison using the 3 metrics would be impractical. As such, we translate the 3 metrics P, F_E and F_L into an aggregate metric, the HOAP, as a weighted combination given by equation 4.12.

$$HOAP = w_1 P + w_2 F_E + w_3 F_L; \quad \sum w_i = 1 \tag{4.12}$$

Two questions could be raised about the HOAP - the exclusion of HO failures and the policy of selecting the weights.

Basically, one could consider including either the HO rate, the HO success rate or the HO failure rate in the HOAP evaluation. However, since minimizing PPs directly also minimizes unnecessary HOs and HO successes, HOs and HO successes are not directly included in the HOAP. On the other hand, a HO failure is just a RLF that occurs during the HO process. As such HO failures being special cases of RLFs, they are indirectly included in the HOAP evaluation as part of RLFs. Note however, that RLFs could also be due other radio related problems in the network e.g. coverage problems. We assume in this case that such do not exist or their effects are so small that they can be ignored. This is a justified assumption based on test results that showed that if we do the HO adequately early (Hys=0 dB and TTT=0 s), RLFs will be eliminated (with excessively high PPs).

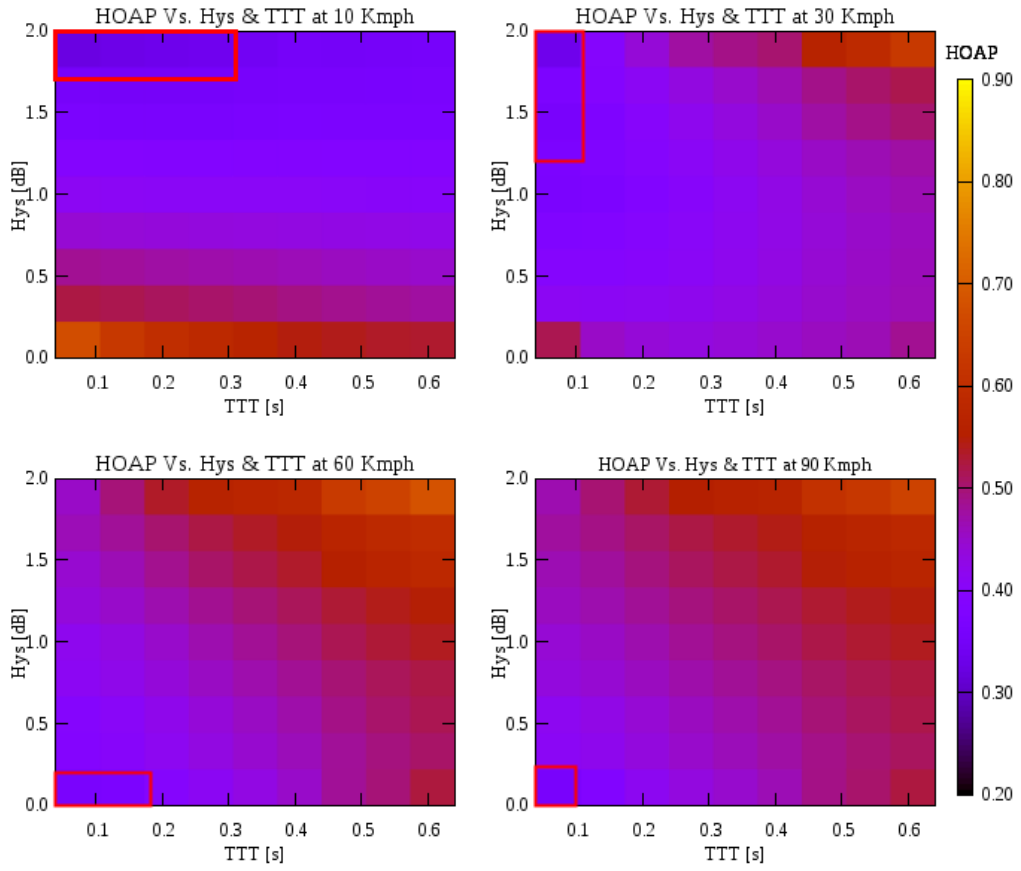
Meanwhile, although the selection of weights is subjective and could be done differently, the weights w_i are selected so as to equally balance effects of early HOs (P, F_E) against effects of late HO (F_L). That means that w_3 should be equal to the other two weights combined i.e. w_3 =0.5 if $\sum w_i = 1$. Then, since RLFs are less desirable compared to PPs, w_2 should be larger than w_1, in this case respectively selected as 0.3 and 0.2. The corresponding weight vector is $w = (0.2, 0.3, 0.5)$ and is the one used in all cases where HO performance is evaluated.

4.2.2.5 HO Control Parameters Sensitivity

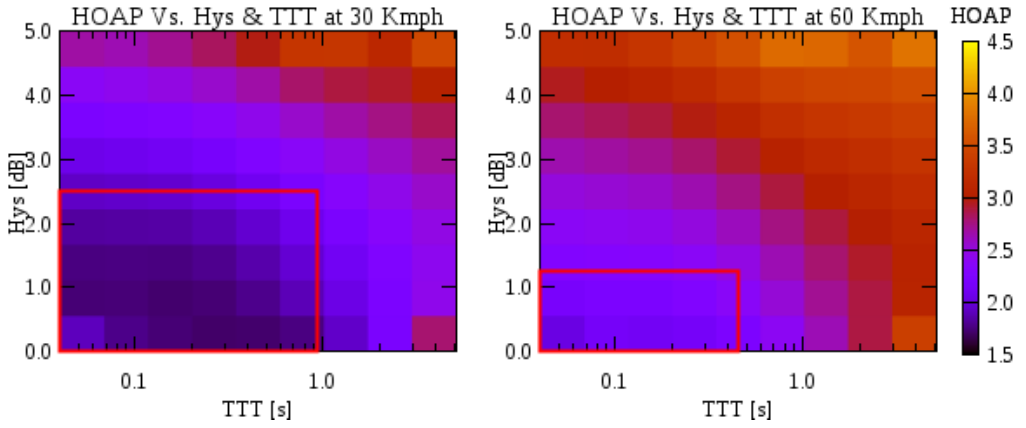
The core MRO goal is to dynamically select the optimum settings (OTP) even for a network with a dynamic mobility profile. To design an apt learning strategy, we investigate the sensitivity of the parameters to UE velocity. We do so by sweeping a selected range of the parameter space for four velocity scenarios. With UEs moving at constant velocity in each scenario, we observe that the OTP changes with velocity as shown in Fig. 4.4. Fig. 4.4a gives the linear variation of the HOAP with both Hys and TTT while Fig.4.4b describes the detailed variation with TTT using a log scale for the TTT.

From Fig. 4.4a, we observe that at all velocities, very high values of Hys are unacceptable although a combination of moderately high Hys and low TTT could be acceptable. High TTTs are only acceptable at low velocities and in combination with low to medium Hys values (Fig. 4.4a). Even then, the optimal settings at low velocity should be the medium Hys and low-to-medium TTT. In effect HOs can moderately be delayed without great penalty since the risk of RLF is low yet even the possibility of PPs is low owing to the low velocity. This is evident in the 10 Kmph environment where for most of the TTTs the performance is good at a Hys of 2dB.

As the velocity increases, the HO delay needs to reduce especially using the TTT. The HOAP is more susceptible to change in TTT, that the OTP continuously grazes the Hys axis, i.e. the performance changes with TTT but is fairly constant



(a) Linear variation of HOAP with Hys and TTT.



(b) Detailed (log scale) variation of HOAP with TTT for 30 and 60 Kmph.

Figure 4.4: Handover Control Parameter Sensitivity.

with Hys. At high velocity, even the Hys has major effect and so both parameters should be low. This is evident in the 60 and 90 Kmph environments in Fig. 4.4a where the OTPs are restricted to the lower left corners of the grid i.e. the part where TTT are within the range of 0-0.64 s. In Fig. 4.4b we observe that with in this small range, although there is major variation in the HOAP with TTT for most Hys, this variation is blurred at points near the optimum point. In that case adjacent TTTs will have practically similar performance. next The most obvious conclusion from Fig. 4.4 is that the OTPs do not lie along any one diagonal for the different velocities as was assumed in [88] and [89]. Any MRO algorithm must as such scan the entire parameter-space or at least more than half the space in order to determine the required trigger point.

4.2.3 QMRO: Q-Learning based MRO

QMRO wishes to determine the optimum *Hys-TTT* action to apply in any mobility state in a cell. Since the actions only affect the performance of the cells and do not change the environment (network) from one state to another, i.e. they do not change the UEs' mobility states, it is adequate to learn an action a in state x that maximizes the expected instantaneous reward r at time t. As derived in equation 4.10, the corresponding Q-update algorithm for instantaneous rewards is equation (4.13)

$$Q_{t+1}(x_t, a_t) = (1 - \alpha)Q_t(x_t, a_t) + \alpha[r_t(x_t, a_t)] \tag{4.13}$$

where α is the learning rate as previously defined in section 4.1.1.2. The following sections discuss the components of the QMRO algorithm.

4.2.3.1 QMRO State Space

The required HO settings in a cell depend on the mobility of the UEs in the cell as shown in the discussion on parameter sensitivity in 4.2.2.5. Consequently, the states x are defined to be the degree of mobility in the cell evaluated as the average velocity over the *SON* interval. Since the average velocity is a continuous variable, we discretize the states x into bands for which the appropriate HO settings must be learned. Table 4.2 describes how the different velocity environments are grouped into mobility states. It also shows, based on the results of the parameter sensitivity in Fig. 4.4, the estimates of the expected default settings that would be used in a manual optimization process.

We assume here that the velocities are known or at least can be estimated by the cells. A simple estimate can be obtained as the ratio of the approximate cell size to the average time that a UE stays in the cell. The velocity can also however be more accurately estimated using the UEs Doppler power spectrum as proposed in [90]. Either way, with a good estimate of the velocity, QMRO can then learn the best configuration for the given cell.

4.2.3.2 QMRO Action Space

Actions are defined as the Hys-TTT tuples that are signaled by the cells to their associated UEs in any observed mobility state. Without a SON solution, an operator configures a cell with default parameter settings obtained through trial and error, while with a local search based SON solution, a fixed set of settings similar to those shown in Table 4.2 are applied. Here, based on the observation that optimum settings depend on speed, we need to change the configuration based on the instantaneous speed in the cell.

It is evident from the parameter-sweep results in Fig. 4.4 that for all practical speeds, performance at Hys>5 dB is almost always sub optimal. We thus consider Hys values only up to 6 dB. Meanwhile differences in HOAP for some TTT settings are unresolvable especially at low TTT values. For example for most Hys values at all velocities, the performance at TTT= 0.08 s, 0.1 s or 0.16 s is practically the same. As such not all TTT values are considered, i.e. the possible TTT actions are the 11 values 0.040, 0.100, 0.128, 0.256, 0.320, 0.480, 0.512, 0.640, 1.02, 1.28, 2.56, 5.12 in s. The resulting action space (total number of actions) for each state is 143 possible combinations of the considered Hys and TTT .

4.2.3.3 QMRO Reward function

We desire to minimize RLFs without excessively increasing PPs and HOs. Since the learner is a rewards-maximizing agent, the reward $r_{x,t}$ should be the negative HOAP evaluated over the SON interval. As stated in 4.2.2.3, the individual rates are normalized to the number of HO candidates, NH as given in equation 4.14.

$$r_{x,t} = -(w_1 P + w_2 F_E + w_3 F_L)/NH; \tag{4.14}$$

Table 4.2: QMRO Mobility States and their default actions.

Average Velocity (Kmph)	State (x)	Default Hys (dB)	Default TTT (s)
0-4	0	3.0	0.0-5.2
4-8	1	2.5	0.0-2.56
8-12	2	2.0	0.0-1.25
12-17	3	2.0	0.0-1.02
17-22	4	1.5-2.0	0.0-0.64
22-28	5	1.5-2.0	0.0-0.48
28-34	6	1.5-2.0	0.0-0.256
34-41	7	1.5	0.0-0.52
41-48	8	1.0	0.0-0.52
48-56	9	0.5	0-0.48
56-65	10	0.5	0.0-0.256
65-75	11	0.0-0.5	0.0-0.16
75+	12	0.0-0.5	0.0-0.16

The weight vector as applied during the learning may need to be adjusted to enforce particular results especially given the small evaluation period (i.e. the SON interval). For example, there may be instances in which no RLFEs are observed during the SON interval which could tilt the result in favor of too many PPs. In this case this is as $w = (0.2, 0.3, 0.5)$ for the typical results and to $w = (0.4, 0.0, 0.6)$ when no RLFEs are observed.

4.2.3.4 QMRO Cooperative Learning

HOs triggering could be affected by the channel conditions that dictate the respective RSRP. However using the *A3* condition for control of the HO trigger point minimizes this dependence on the absolute RSRP values, since decisions are made based on RSRP differences among the cells. In that case, HO performance depends only on the mobility of the users and the values of the control parameters Hys and TTT.

With mobility based HO states, it is possible that a state observed in one cell reoccurs in another cell at some other time. As such, cells do not need to learn independent policies but can learn a single policy function based on the abstract mobility states. The result is a cooperative QL problem in which individual cells take actions but update a single Q-table that represents the shared learned policy.

The cooperative learning solution holds if all other critical parameters are comparable among the cells. For example the cells in the considered network are assumed to be of similar size and applying comparable transmit powers. Other than this, the RSRP profiles at the cell edges may be different resulting in differing behaviors for different cells or cell pairs. Similarly if individual cells concurrently have users with differing behavioral patterns, the solution may break down. For example, the assumption that a state in one cell will be observed in another will not hold for cell that covers a highway crossing through an office park. Such a cell concurrently has 2 groups of users - the slow-moving office users and the fast-moving highway users, each of which will requires different settings. Nevertheless, for the majority of networks that do not have such special conditions, cooperative learning as considered in these studies would be fully applicable.

4.2.3.5 QMRO Parameter Search Strategy

Given the Hys and TTT values in section 4.2.3.2, each cell has up to 143 possible actions to consider for each velocity state. Although learning is accelerated by the cooperative strategy, the need to consider each action multiple times, would still require a long time to converge to the desired solutions.

To accelerate convergence, we subgroup the 143 actions so that for any state, 3 learning regimes R1-R3 are executed as shown in Fig. 4.5. At *R1*, actions are selected from different regions of the grid in order to determine the area in which the desired action lies. From the parameter sensitivity analysis in section 4.2.2.5,

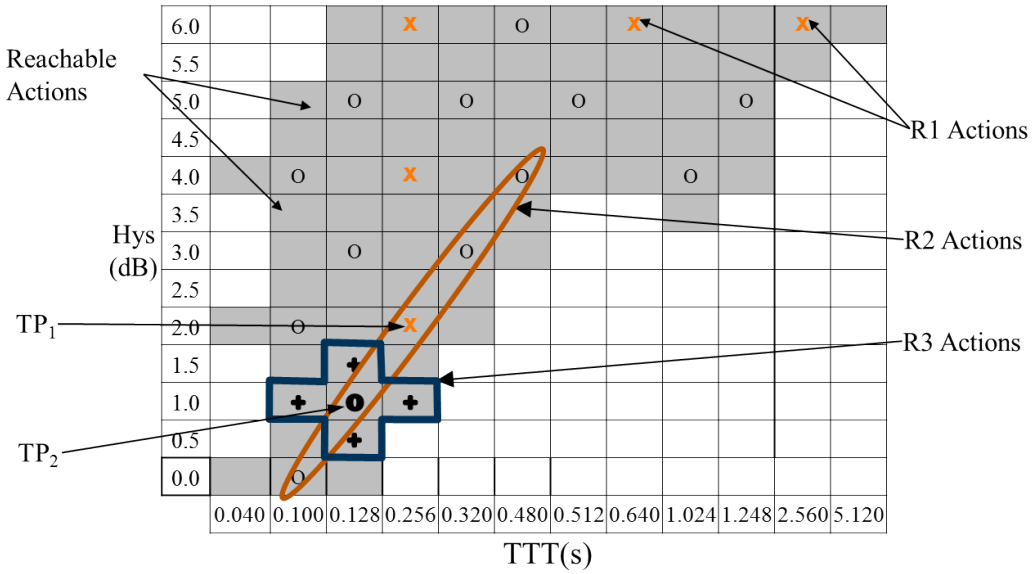


Figure 4.5: QHO action space within the three learning regimes R1, R2, R3.

combinations of low Hys and high TTT are never optimal. As such this region is excluded from the possible candidates that are denoted by "R1 actions" in Fig. 4.5. The outcome trigger point of *R1* (TP_1) specifies the region in which the optimum point lies. This outcome is thus used to define the search space for the next regime *R2*.

At *R2*, actions along the diagonal that goes through TP_1 are explored to obtain the approximate delay that is acceptable for the observed mobility state. In this case, subsequent actions differ in Hys by 1 dB to enable a large enough action space to be explored. As an example, if at *R1* TP_1 is obtained as $TP_1 = (2.0dB, 0.256s)$, at *R2* the agent explores the region marked *R2 Actions* in Fig. 4.5. The obtained TP (TP_2) is then used to define the search space for the next regime *R3*.

R3 refines the learned TP_2 by exploring points in TP_2's vicinity. It compares TP_2 with its four neighbor points to the left, right, top or below i. In Fig.4.5, '*R3 Actions*' shows the exploration region for *R3* assuming that at *R2*, $TP_2 = (5.0dB, 0.128s)$.

4.2.3.6 The QMRO Optimization Algorithm

For any setting that is applied in a cell, the cell needs to observe the performance over a SON interval at the end of which the cell determines the quality of the applied setting. Each optimization decision is a comparison of the number and type of events that result from the different parameter combinations. As such fair comparison of the trigger points requires that comparable statistics are considered for each point, i.e. that all trigger points are evaluated based on comparable numbers of events. However, the occurrence of events depends on the number of users in the cell as well as their velocities, so that within the same fixed time interval, cells may observe different statistics which may give contradicting conclusions

about the parameter settings. Consequently, instead of setting the SON interval based on a fixed time period, it is based on a minimum number of HO events that must occur following the application of any action or configuration. This minimum number is the sum of the disjoint HO related events (i.e. HOs, RLFEs and RLFLs). In this case, this number is set to 100 events although any value that ensures that adequate counts of all the necessary statistics (i.e. for NHs, PPs and RLFs) are observed would be appropriate.

Then, given the Q-learner's elements as discussed above, the optimization algorithm can be described by the procedure of *Algorithm 4.2*, i.e: For each possible state, the action set is initialized with R1 actions and the Q-table entries initialized to 0. The learning is then triggered to be executed after every SON interval. Each cell c observes its environment over the SON interval and at the end of interval t, the cell determines if an optimization is necessary. During the learning phase, c selects an action as described in Section 4.2.3.5, otherwise it selects the best action that would have been learned. It then signals that action to all its associated UEs and starts the monitoring period (interval $t+1$) to collect the necessary performance statistics. At the end of interval $t+1$, c evaluates its HOAP and derives

Algorithm 4.2: QMRO - The Q-Learning HO Algorithm

Require: UE velocities during *SON* interval, action set a
1. Initialize action set $A_{x,R1}$ for learning regime 1 in all states x
   Repeat for each SON interval t
2. **if** HO action was taken in at SON interval $t-1$ **do**
3. determine $HOAP$ and derive reward $r_{t-1}(x_{t-1}, a_{t-1})$
4. update Q-table according to 4.13
   end if
5. evaluate current mobility state x_t (table 4.2)
6. **if** learning complete for state x **do**
7. select $a_{x,t} = a_x^{opt}$, the absolute optimum value for state x
8. **else if** regime Ri exploration is incomplete **do**
9. select $a_{x,t}$ (sequentially after $a_{x,t-1}$) from $A_{x,Ri}$
10. **else do**
11. select $a_{x,t} = a_{x,Ri}^{opt}$, the optimum value for state x at Ri
12. **if** all learning regimes complete for state x **do**
13. record all regimes complete for state x
14. record $a_{x,t}$ as best action in state x
15. **else do**
16. use $a_{x,t}$ to set $A_{x,Ri+1}$
    i.e. reconfigure the action set A for regime $Ri+1$
    end if
    end if
17. Signal selected action $a_{x,t}$ to all UEs in the cell.
18. $t \leftarrow t+1$, start monitoring
    end loop

the reward r_t for the action at t. It then updates the learning agent (the Q-table) before repeating the process.

4.2.4 Simulation Results and Discussion

Simulation studies were done using the LTE down-link system simulator as described in chapter 3. Multiple simulations each evaluating a different velocity environment are considered. Each scenario considers an environment in which the velocity continuously varies as described in the next section. In all cases, the parameters introduced in chapter 3 are considered with multiple batches executed for each simulation. The most critical parameters are however restated in table 4.3.

4.2.4.1 Realistic, Dynamic, Mobility Environments

The crux of MRO is to adjust the HO parameters in line with varying mobility profiles. We evaluate the performance in the 5 different velocity scenarios shown in Table 4.4 to prove that the algorithm is applicable to any network. The 3 'normal' environments (*10, 30, 60 Kmph*) would for example represent 3 typical city districts - a city center, city edge and residential suburb. We however also consider two extreme environments (*3 & 120 Kmph*) which could respectively represent an office park and a highway.

In each mobility scenario, all UEs have independent randomly varying velocities of up 40 % below or above the mean. We implement this by allocating random velocities to the UEs at the start of the simulation and also randomly adjusting the velocities at the start of every batch as well as during the simulation. For example, each of the 240 mobile users in the *city suburb* (60 Kmph) network would have individually assigned and continuously changing velocities during the simulation as shown in Fig. 4.6 for the average velocity in each of three selected cells over a period of 10 batches.

Table 4.3: QMRO-specific Simulation Parameters.

Parameter	value
Inter-site distance	500 m
Time between snapshots	50 ms
No. of mobile users	240
User velocity	variable
Data rate	512 Kbps

Table 4.4: QMRO Velocity scenarios.

City Area	Velocity (Kmph)	
	Mean	range
Office park	3	2 - 4
City Center	10	6 -14
City Edge	30	18 - 42
City Suburb	60	36 - 84
Highway	120	72 -168

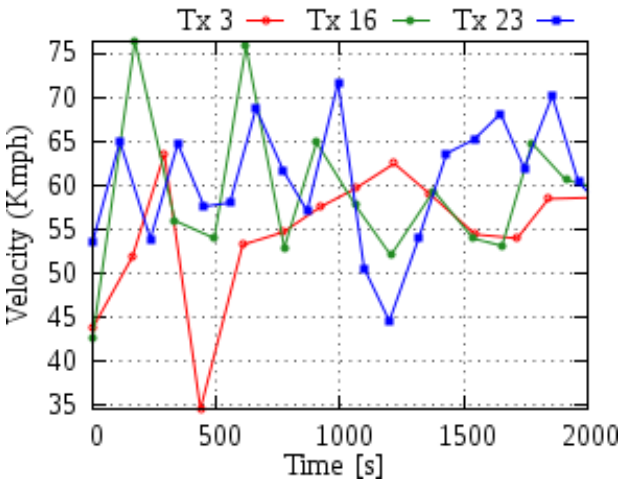


Figure 4.6: Typical average velocity variation in 3 cells in the $60Kmph$ scenario.

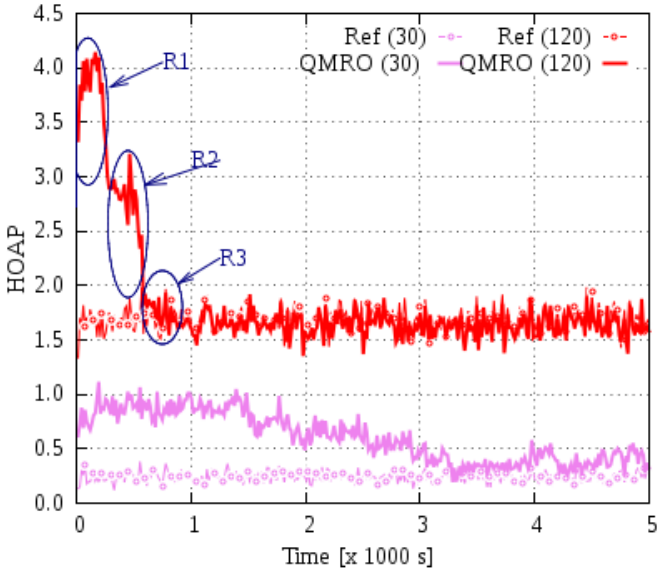
4.2.4.2 Performance in terms of HOAP

To evaluate the benefits of the QL solution, we compare its resulting performance for each velocity scenario against the reference network denoted by Ref. Ref represents the case when all cells in the network apply the best static settings as obtained from the parameters sensitivity analysis in section 4.2.2.5. The analysis of the results is done is terms of the average of the metric(s) values throughout the network although cell-specific results would demonstrate the same trends.

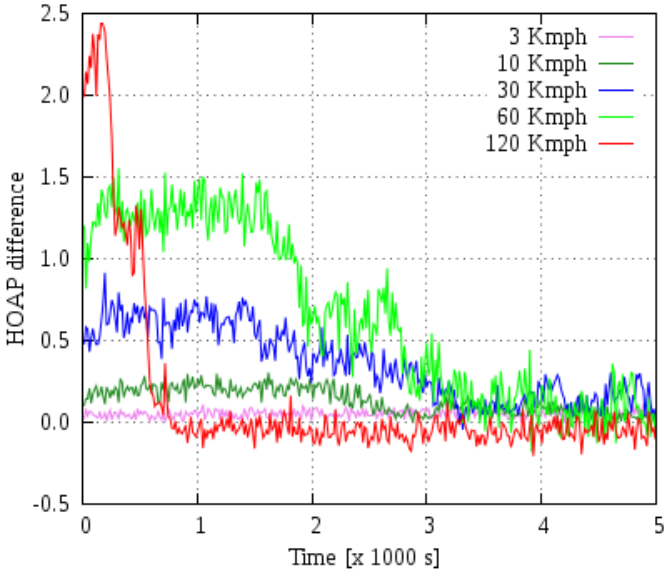
Fig. 4.7 summarizes the performance in the different velocity environments. Fig 4.7a shows the comparison of the average HOAP values for QMRO and Ref in two typical velocity cases, while Fig. 4.7b describes the differences in performance between QMRO and Ref for all the five velocity cases. We observe in both figures, that at the beginning QMRO performs poorly as it executes the first learning regime R1. The performance then improves in regimes R2 and R3 as QMRO focuses in on the OTP, to the extent that it is eventually equivalent to that of Ref.

Meanwhile, in cases where user velocities widely spread, QMRO actually performs better since it is able to set the right setting for each observed velocity range as opposed to a single setting for all velocities. This is evident for the 120 Kmph case in Fig. 4.7b where, after learning, QMRO is consistently better than Ref.

We observe that the convergence time is different for different velocities - typically smaller at higher velocities. This is mainly because the SON interval is shorter at higher velocities. Remember that the SON interval depends on a minimum number of events that need to be registered. At high velocity, each user undertakes more HOs than at low velocity. Consequently, it takes a much shorter time for a cell to reach the minimum event count i.e. a shorter SON interval. Thus, within the same time, the network evaluates more actions at higher velocity than at low velocity resulting into faster learning.



(a) HOAP comparison for two velocity scenarios.

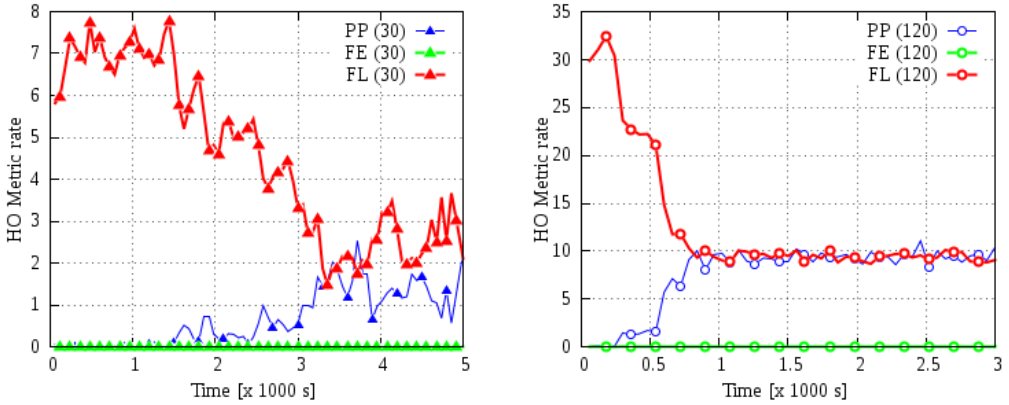


(b) HOAP gains for all velocities.

Figure 4.7: QMRO Performance:Average network-wide HOAP for QMRO in comparison to the reference network.

4.2.4.3 QMRO Learning Trend

Fig. 4.8 evaluates the time variation of the individual metrics (P, F_E and F_L) when applying QMRO in two velocity scenarios (30 and 120$Kmph$). We observe in both cases, that the agent learns to minimize RLFLs (F_L) by trading them with Ping-pongs (PPs) which have less effect on the user's quality of experience. This is all while ensuring that RLFEs (F_E) remain low. In both velocity scenarios, QMRO suffers many RLFs at the beginning as it considers settings across a large parameter space. Over time however, the agent continuously reduces F_L by trading such reduction with increase in PP. It then stops this trend as soon as it registers



(a) Performance Trend at 30 Kmph. **(b)** Performance Trend at 120 Kmph.

Figure 4.8: QMRO Performance: Variation of average rates for all 3 core metrics in 30 and 120 Kmph environment.

decreasing returns, i.e. when each extra reduction in F_L translates into an excessive increase in PPs or if it instead causes RLFEs to occur.

The foregoing results demonstrate that given a good definition of states that appropriately capture the UEs' mobility and also given adequate learning time, a QL based MRO algorithm is able to learn the appropriate Hys-TTT settings for any mobility environment.

4.3 QL for Mobility Load Balancing

Users are rarely uniformly distributed in cellular networks. This becomes a problem when a serving cell s gets overloaded at a time when free resources exist in neighbor cells. Although for SINR maximization s is the best cell to serve the associated users, it cannot efficiently do so while in overload. A solution is then required to automatically redistribute the load among cells - thus the Mobility Load Balancing(MLB) use case of SON. This section describes our QL proposal for such a self-organized MLB solution.

4.3.1 Mobility Load Balancing

To lower the serving cell's load ρ_s, MLB moves some of the edge users in cell s towards one or more neighbor cells or so called target cells. Let us denote the set of all target cells as T, any one cell in the set as t and all cells together as T-cells. Then consider the A3 HO entry condition earlier defined in equation 3.12 and reproduced in 4.15 for completeness. MLB undertakes load re-distribution by virtually shrinking s while concurrently expanding the T-cells. As proposed in the LTE SON standard [6], this can be achieved by adjusting the relative HO margins of the respective cells using the CIOs ($O_t^{s,t}$ and $O_s^{s,t}; \forall t \in T$). In principle, the two

offsets $O_t^{s,t}$ and $O_s^{s,t}$ define the applicable HO boundary between the respective two cells s, t.

$$F_t + O_t^{s,t} - Hys > F_s + O_s^{s,t}. \tag{4.15}$$

As stated the set T for which CIOs are adapted may contain all of s's neighbor cells or only a few, which in principle could also be a single neighbor. Considering all neighbors is straight forward while for the case of few of the neighbors, the set elements could be selected as those neighbor cells that fulfill a certain condition e.g. neighbor cells in very low load. Selecting a single neighbor is however not as obvious since a choice would be required as to which of the neighbors is best. Such a neighbor would be the one that achieves the highest reduction of load in s, yet without itself getting highly overloaded. This then requires s not only to have access to load information in all neighbors but to also be able to predict what that load would be after users have been transfered to any selected target. Moreover, in a highly dynamic network, such a prediction would very quickly change with the mobility of the users. In that case, a simple decision to adapt CIOs for all neighbor cells may achieve better results.

4.3.1.1 State of the Art MLB Solutions

Load Balancing (LB) is a major area of interest in SON. A LB solution that estimates required resources in t before LB action is presented in [24]. In [91], network wide LB is expressed as a single integer optimization problem to be solved centrally. While it ensures to uphold network-wide throughput, its centralized nature may lead to suboptimal solutions in some cells and may limit its speed of convergence especially due to network wide signaling and processing. Besides, simulations were only done for up to 3 Kmph, a small part of the expected velocity spectrum in real network deployments.

For a fully SO solution, a distributed automated approach is required as described here. The basic solution is a Reactive MLB algorithm (RLB) that symmetrically adjusts the CIO by a value ϕ between s and all low loaded neighbor cells. With different ϕ values for user speeds of up to 30 Kmph, we observe that the optimum ϕ depends on the offered load in cell s, ρ_s; the average load of the cells in set T, and the geographic user distribution in s. We thus apply Q-Learning (QL) for a distributed fully self organized MLB algorithm, called QLB that learns the best ϕ required for any such load conditions.

4.3.1.2 Evaluation Parameters and Performance Metrics

When a cell is overloaded, its users are unsatisfied, i.e. they are allocated fewer PRBs resulting in lower data rates than expected. Due to scheduling variations, a user is considered unsatisfied (an un-satisfaction event occurs) only if the user's total achieved data rate in a continuous 1 second period is less than the Guaranteed Bit Rate (GBR). We evaluate the overload situation in a cell or the network in

terms of the "Number of unsatisfied users" (N_{us}), which is the average number of un-satisfaction events in the cell/network per second over the evaluation period. Such an evaluation period could be 5s for the quasi-instantaneous performance in the cell or network or a batch period for the long-term results evaluating the improvements achieved by the optimization algorithm(s). Meanwhile, where actions are taken to reduce overload, we evaluate the effectiveness of the actions in terms of their change in offered load in serving cell.

As presented in chapter 3, to evaluate the effectiveness of the MLB solution, extra overload is artificially created in one cell - the so called "center cell". This is achieved by randomly placing a number of static GBR users, N_s, within the boarders of this center cell. With $N_s = 40$ users, each user (static or otherwise) who then does achieve the desired GBR rate is considered unsatisfied.

4.3.2 Static, Reactive MLB Solution: RLB

We have observed that load redistribution may be easier to managed by changing the generic boundary of a cell to all its neighbors. This section describes our proposed approach, RLB, that applies this strategy. We justify the need to use this approach and describe how such a solution can be implemented. This solution then sets the basis on which the subsequent learning based solution is built.

4.3.2.1 Structure of the Static RLB Solution:

Although it is not so obvious, the minimum MLB action is to select an edge user of cell s for HO to a cell t and then appropriately reducing the offset $O_s^{s,t}$. To avoid the user from moving right back to s, an opposite value (to $O_s^{s,t}$) is applied on $O_t^{s,t}$. This shifts the $s-t$ HO boundary with the specific cell t which however does not guarantee that overload will not quickly reoccur due to a user at a boundary to another cell.

To manage the general load in the cell, the proposed approach,RLB, adjusts the generic boundary with all low-load neighbor cells. This excludes higher load neighbors as shown in Fig. 4.9, where boundaries are adjusted for all cells except t_4 which is highly loaded. Adapting boundaries for multiple neighbors ensures that overload does not quickly re-occur in s. Cell s applies the RLB algorithm to adjust CIOs to all T-cells (the low load neighbors) by a fixed value as in equation 4.16.

$$\begin{aligned} O_t^{s,t} &= O_t^{s,t} - \phi \\ O_s^{s,t} &= O_s^{s,t} + \phi \end{aligned} \quad all\ t \in T. \tag{4.16}$$

In the simplest form, CIOs could be gradually changed, each time by a small step over multiple iterations. Although this avoids unnecessary load transfer to the T-cells, it takes too long to remove the s overload. To improve convergence speed,

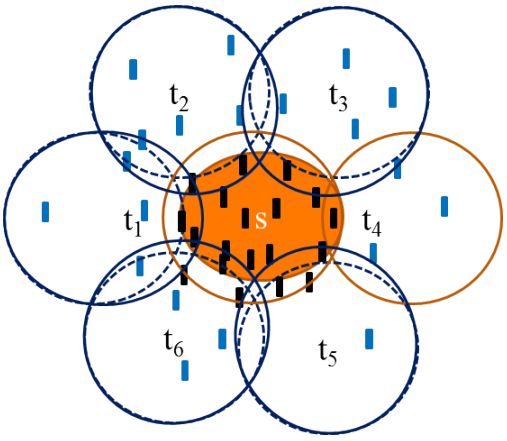


Figure 4.9: Reactive change of CIOs.

CIOs are adjusted in a single precise step ϕ that removes s's overload without overloading T-cells.

A large change of CIOs may however cause oscillations, where after load transfer from s to t, t gets overloaded and also initiates LB towards s, causing s to restart the process once again. RLBs does not explicitly control this LB induced T-cell overload, but we mitigate it using an oscillation control timer T_{oc}. Following s-t LB , T_{oc} has to expire before a LB HO can be triggered from t to s. The size of T_{oc} is set equal to the SON interval although higher values could also be applicable.

Note however that minimal T-cell overload after LB HO may be a good result, as it allows the extra load to propagate outwards from the "center" cell to outer cells over subsequent LB actions in different cells. This is especially so if combined with the timer T_{oc}. In this case the new s (original T-cell) would not move load back to the original s. Instead it moves the load to other cells in its neighborhood, which are further away from the original s.

4.3.2.2 Dependence of RLB Gains on Load

As a manual solution, the optimal ϕ could be determined by operating different step sizes and selecting the best which is used thereafter. The achieved change in ρ_s, $\Delta\rho_s$ for any ϕ applied on the $s-t$ boundaries is however dependent on ρ_s the load in s as well as ρ_n the average load in t. To investigate this dependence, we define a set of load scenarios and evaluate $\mathrm{E}[\Delta\rho_s]$, the expected change in s load for each scenario Γ and applied CIO change ϕ. Note that $\Delta\rho_s$ is not deterministic for each combination of ϕ, ρ_s and ρ_n. The dependence can as such only be expressed in terms of the expected outcome $\mathrm{E}[\Delta\rho_s]$.

Each load scenario Γ is a combination of ranges of ρ_s and ρ_n as shown in Table 4.5, e.g. Γ=3 is the tuple $[0.9 \leq \rho_s < 1.1;\ \rho_n < 0.45]$. For each ϕ in a given Γ, if $\Delta\rho_s$ is observed, $\mathrm{E}[\Delta\rho_s]$ is updated according to equation 4.17.

$$E[\Delta\rho_s] = \gamma \cdot E[\Delta\rho_s] + (1-\gamma)\Delta\rho_s. \tag{4.17}$$

Table 4.5: Cell Load Scenarios (Γ).

		Mean T-cell Load ρ_n		
		<0.45	[0.45-0.60)	0.60+
ρ_s	[0-0.9)	0	1	2
	[0.9-1.1)	3	4	5
	1.1+	6	7	8

ρ_s - serving cell load

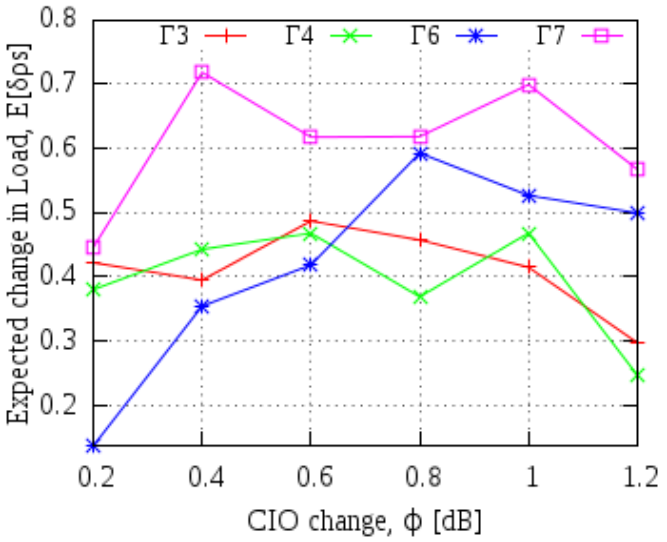


Figure 4.10: Dependence of RLB gain on cells load.

The forgetting factor γ is selected as 0.95 to ensure that any one observation does not unexpectedly skew the average.

Fig. 4.10 evaluates the dependency of $E[\Delta\rho_s]$ on Γ for different ϕ values. The ϕ values are evaluated in a network having 240 mobile users and where the 'centre cell' is loaded with 40 static users. We observe that the best ϕ, ϕ_{opt} is different for each Γ e.g. ϕ_{opt} is 0.6, 1.0 and 0.4 for $\Gamma 3$, $\Gamma 4$ and $\Gamma 7$ respectively. This justifies the need for a learning solution that learns the best ϕ_{opt} for each of the load conditions.

4.3.3 QLB: Learning the Optimum Actions

With RLB, a fixed ϕ is found that guarantees good average performance, but not the best in each load scenario. For optimal performance in each scenario, different ϕ values would be required for the different scenarios. Moreover, for any change in CIO, the resultant $\Delta\rho_s$ also depends on the user distribution within the serving cell s, i.e. more load can be offloaded from s if there are more users at the cell edge. The solution then is to apply QLB so as to learn the required CIO change for each combination of the three - the serving cell load ρ_s, the average neighborhood load

ρ_n and the user distribution in s. The user distribution denoted by uD describes how many of the users in cell s are close to the cell boarder, with the expectation that if there are more users at the edge, only a small change in the CIO would be required in order for these users to be handed over to neighbor cells.

QLB learns the action that instantaneously removes overload, constrained to ensuring that target cells are not overloaded as a result of its actions. From 4.1.1.2, the corresponding Q-update equation for the instantaneous rewards is:

$$Q_{t+1}(x_t, a_t) = (1 - \alpha)Q_t(x_t, a_t) + \alpha[r_t(x_t, a_t)]. \tag{4.18}$$

where α is the learning rate as previously defined in 4.1.1.2. The following sections discuss the structure and components of the *QLB* learning algorithm.

4.3.3.1 State-space

As we expect that the size of ϕ in equation 4.16 depends on ρ_s, ρ_n and uD, each state is constructed as the vector [ρ_s, ρ_n, uD]. We defined 27 states to be the 9 load scenarios in Table 4.5 for each of 3 uD cases. In this case, uD represents the fraction of s users at the edge, with the 3 uD cases considered being $uD < 20\%$; $uD = [20 - 35]\%$ and $uD \geq 35\%$. Note that since the idea is to change the boarder to all neighbors, uD is evaluated not specifically to any one neighbor but generically within the serving cell s.

4.3.3.2 Action-space

Actions are the possible values that ϕ can take, i.e. the amount in dBs by which the CIO should be changed. Guided by RLB results, actions are selected as the discrete ϕ values [0.2, 0.4, ... 1.0] dB.

4.3.3.3 Rewards

QLB aims to determine the action ϕ that instantaneously removes overload from the serving cell, but without overloading the target cell(s). As such the rewards, as set in equation 4.19, consider $\Delta\rho_s$ (the achieved reduction in ρ_s the serving cell load) and the extra load created in neighbor cells.

$$r = \begin{cases} \Delta\rho_s + 1 & ; \quad \Delta\rho_n = 2 \; and \; \Gamma < 3 \\ \Delta\rho_s & ; \quad \Delta\rho_n < 1 \\ \Delta\rho_s - 1 & ; \quad otherwise \end{cases} \tag{4.19}$$

Positive $\Delta\rho_s$ represents reduction in the offered ρ_s that results from users having moved to neighbor cells. Positive $\Delta\rho_s$ is thus rewarded while the reverse is penalized. Since ρ_n is expected to increase as a result of adding users at the very edge of the cells, only $\Delta\rho_n$ of more than 1 is penalized. In general, larger $\Delta\rho_s$

values receive greater reward, but are accompanied by penalties for unrestrained actions taken in LB-states with high ρ_n. This allows high load t cells to overload just enough so as to propagate the load outwards but not so much as to counter-productively cause further un-satisfaction after LB. However, special consideration is taken in cases where a large reduction in ρ_s can be achieved without overloading the target cell. In those cases, e.g. for the change from scenario 9 to scenarios 1 or 2, the reward is increased by 1.

4.3.3.4 QLB algorithm

Each load state that is observed in one cell can reappear in any of the other cells in the network. As such, cells do not need to learn independent policies but can learn a single shared policy in a cooperative learning process. QLB thus applies a single Q-table which is updated by all cells during the learning process.

During operation, each cell s observes its environment and at the end of a SON interval of $\tau = 5s$, the cell evaluates its load conditions. The cell applies the procedure in Algorithm 4.3 to either take actions that reduce overload, learn from its previous actions or both. In case of overload, s selects an action as described in Section 4.3.3.2 and signals the new HO settings that include the revised CIO to all its associated UEs. Since the CIO change is symmetric for any two cells at a given boundary, s also sends the revised CIOs to its affected neighbor cells via the LTE X2 interface. At the end of interval $\tau + 1$, s evaluates the changes in load and derives the reward. It then updates the learning agent (the Q-table) before repeating the process.

4.3.4 QLB Simulation Results

Simulation studies were undertaken using the LTE down-link system level simulator of chapter 3. The results consider two perspectives - load variation in individual cells to evaluate the dynamic performance of the solution and the number of unsatisfied users (N_{us}) evaluating the global the effect of the algorithm and its actions on user satisfaction. As earlier defined in 4.3.1.2, the N_{us} in a cell or in the network is the average number of user un-satisfaction events in the cell/network per second over the evaluation period. In this case, a user is considered unsatisfied (i.e. an un-satisfaction event occurs) only if the user's total achieved data rate in the continuous 1 second period is less than the GBR.

The load related studies consider the network at 3 Kmph while two other velocities - 10 and 30 Kmph are also considered to evaluate the N_{us}. The corresponding results are given by Figs. 4.11 and 4.12.

Algorithm 4.3: QLB - The Q-Learning MLB Algorithm

Require: T-List, the list of s neighbors, action set a

1. **if** LB action was taken in previous SON interval i **do**
2. determine $\Delta\rho_s$ **and** derive reward
3. update Q-table according to 4.18

end if

4. evaluate ρ_s
5. **if** overloaded **do**
6. determine load state $l = [\rho_s, \rho_n, uD]$
7. **if** exploration incomplete **do**
8. from set a, select $\phi = a_l^{i+1}$ i.e. in sequence of last selected value a_l^i in state l
9. **else do**
10. select $\phi = a_l^{opt}$, the optimum value learned for state l

end if

11. **for** each cell t in T-List with timer TOC expired
12. reduce ${O_t}^s$ by ϕ **and** increase ${O_s}^t$ by ϕ
13. start timer T_{OC} for t LB HO towards s

end for

end if

14. Restart SON interval timer i.e. $t \leftarrow t + 1$

4.3.4.1 Learning towards load redistribution

Fig. 4.11 shows the dynamic behavior of the load balancing solutions in terms of variations of cell load during the simulation, with the load evaluated every second for each cell. For clarity, the figure considers simulations at 3 Kmph for which the dynamic behavior is slow enough to be analyzed.

We observe that both solutions RLB and QLB are able to lower the load in cell 12 by transferring it to the neighbors cells (e.g. cell 14) and eventually to the outer cells (e.g. cell 6). After learning however, QLB achieves better response to overload compared to RLB. This can be seen for the period after 3000 s where cell 12 consistently has slightly lower load for QLB as compared to RLB. This is the direct consequence of having learned the best CIO change for each load state in the case of QLB, which is not the same for RLB. The drawback here is that in a low mobility network, the extra load added to the neighbor cells may take long to move outwards. In that case we need to evaluate performance in terms of the Number of unsatisfied users, N_{us}.

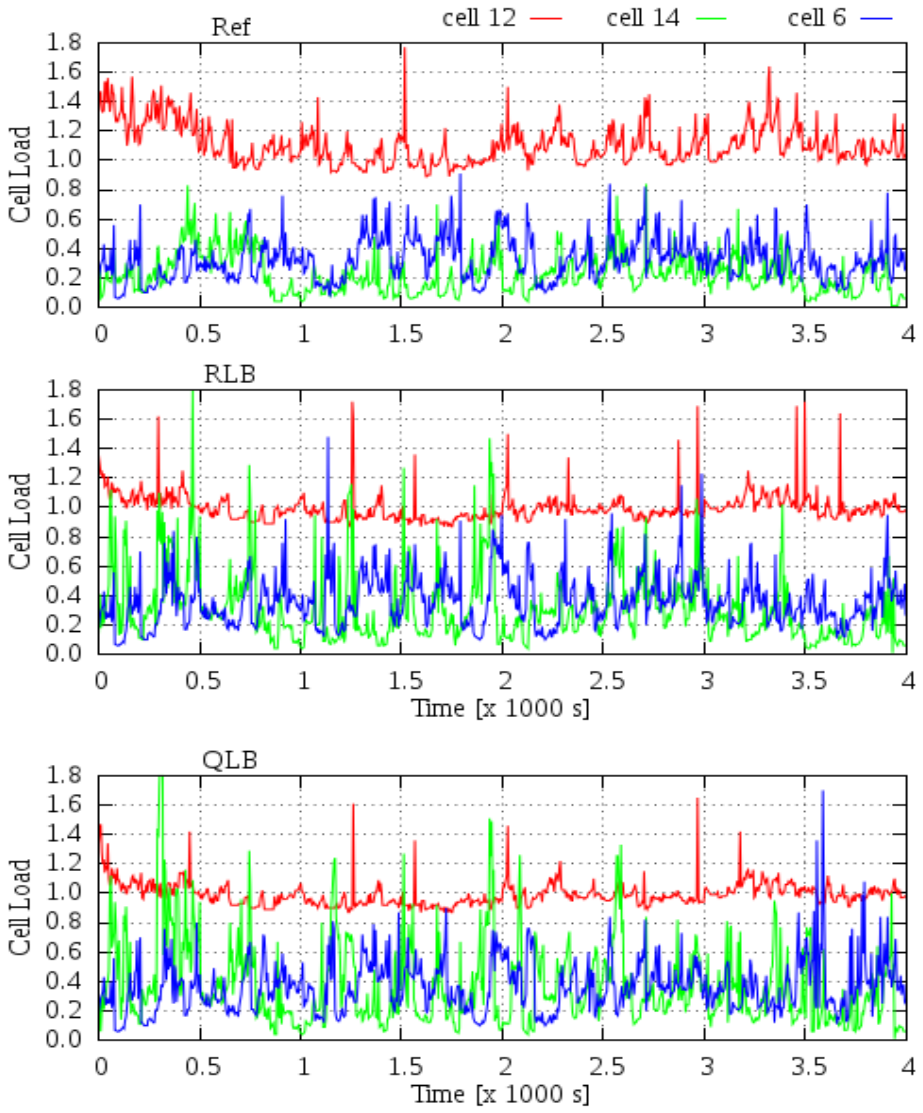
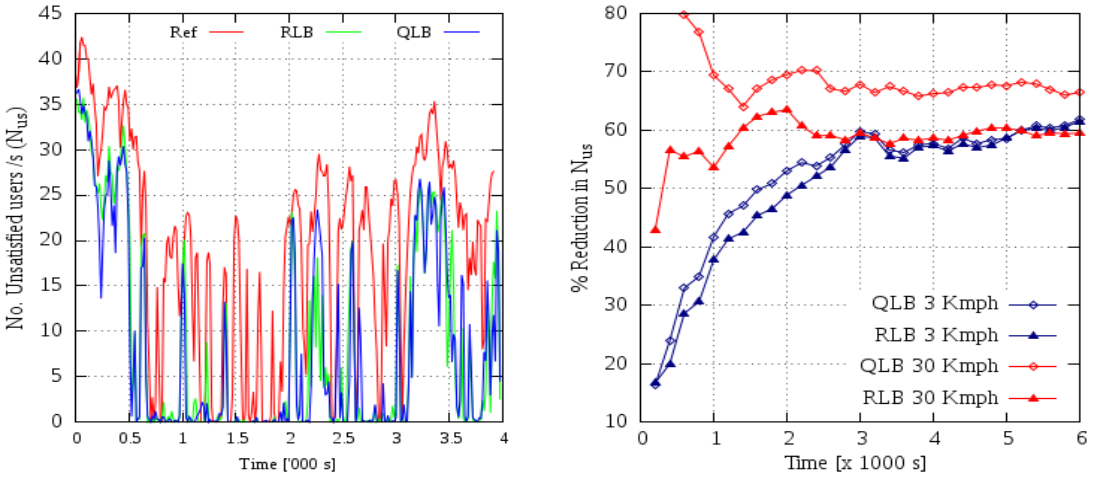


Figure 4.11: QLB and RLB load redistribution (load moved from cell 12 to neighbor cells e.g. 14 and 6).

4.3.4.2 QLB Effect on User satisfaction

Fig. 4.12 compares the performance of RLB and QLB against the reference case (Ref) in terms of the N_{us} in the network for different velocity scenarios. Subfigure 4.12a shows the quasi-instantaneous variation of network-wide N_{us} in the 3 Kmph network scenario for the same period considered in Fig. 4.11. With user satisfaction evaluated every $15s$ and each point capturing the total un-satisfaction events over the $15s$ interval, it is evident in the figure that both RLB and QLB reduce the N_{us} through the load redistribution.

Subfigure 4.12b evaluates the global benefit of solutions for two velocity scenarios across the simulation. It evaluates the gains of each of the two solution in terms of the percentage reduction in N_{us} when compared to Ref in the different batches of



(a) Variation of Nus at 3 Kmph. **(b)** QLB's reduction in Nus.

Figure 4.12: Effect of QLB on user satisfaction.

the simulation. We observe that for both mobility scenarios, both RLB and QLB improve user Quality of Experience (QoE) by reducing the N_{us}. The reduction in user dissatisfaction is comparable at low velocity since there is not much dynamism to be exploited by varying the CIO change. At higher velocity however, by adjusting CIOs for each instantaneous state that is observed, QLB achieves better user satisfaction.

The results above prove that Q-learning can easily and successfully be applied towards a dynamic autonomous solution for Mobility Load balancing.

4.4 Summary

In this chapter, we have discussed the proposed Q-Learning (QL) framework as the development structure for the SON functions. Beginning with a review of QL, the discussion highlighted how QL has so far been applied in SON and justified how it can be used for any generic SON Function. The chapter then discussed the application of this framework to two selected SFs: MRO and MLB. Critical to note here is that, although all SFs apply the same framework, special adjustments are required for each SF as dictated by its specific constraints. For example each SF required a different strategy on how to explore its action space. In general however, comparing the performance of the developed solutions to a system that does not apply any automation or SON solution, both solutions had positive results that proved that Q-Learning (QL) provides a good framework for developing SON functions.

The QL solution for MRO, QMRO, presented an algorithm that learns the best HO Hys and TTT settings which would need to be applied in any mobility state observed by a cell. With actions defined as the Hys - TTT combinations, each cell

takes an action in the network and, based on the observed quality (performance) of the action, the cell updates a cooperative learning agent (single Q-table) which learns the right actions. Performance was evaluated in a realistic network with UEs having distinct and dynamically varying velocities. Considering diverse velocity environments, results showed that QMRO is able to learn the required parameters in any environment.

QLB, the QL solution for MLB presented a learning algorithm built on top of a reactive solution, RLB. RLB responds to overload in a cell by adjusting the generic HO boundary between the overloaded cell and all its low load neighbors. Using the CIO for adjusting the HO boundary, it was observed that the required change in CIO depends on the load state that is characterized by the load in both the serving and neighbor cells as well as the user distribution in the serving cell. Based on this observation, the QL solution then evaluates different CIO changes and learns the best for each load state. QLB performance was evaluated in different velocity environments in each case comparing it to a reference network, Ref, that does not apply any load redistribution as well as to the same network when applying RLB, the reactive manually optimized solution. It was observed that, compared to the Ref and RLB, QLB achieves better distribution of load among cells. In particular QLB achieves better results in a more dynamic environment e.g. where users move at higher velocities. The result of the load re-distribution is that the number of users who would otherwise be unsatisfied, owing to low data rates induced by the overload, is reduced.

In both SFs considered here, results showed that the QL solutions are able to achieve their objectives. The learning strategy in both cases considered the naive approach of trying all the available actions a number of times and thereafter indefinitely applying the best action. Using such a simple exploration strategy, convergence - measured in terms of the time required to learn the optimum solutions - would depend on how often the states re-occur in the cells. This was however improved by the combination of distributed exploration and centralized cooperative learning. In practice some variant of this approach would be required since in a large network, cells operate in diverse environments and would require specific handling. Moreover human intelligence could improve the performance especially during the learning phase, e.g. by limiting the applicable parameter space to a range known to have good performance. It may also be worth investigating methods through which states and actions can be automatically derived. This would reduce the design time and possible the human subjectivity that is typically included in the solutions. However, the combination of these results with the state of the art solutions presented in section 4.1.4 confirms that QL provides a good approach to developing SON solutions especially where the dependence of metrics to the control parameters is not perfectly known.

Chapter 5

Characterizing Conflicts among SON Functions

Contents

In Chapter 4, we presented Q-learning as an appropriate approach for developing SON functions with the achieved results re-affirming the hypothesis. Acting independently within their environment, each of the SON functions fulfilled its objectives, i.e. QMRO optimized the occurrence of HO events (RLFs against PPs) and, QLB minimized user dissatisfaction resulting from cell overload. When combined however, SON functions affect each others' metrics and may cause system instability. This chapter explores the extent of these conflicts. Specifically, it evaluates the different cross conflicts between MRO and MLB.

5.1 Scenario, Parameters and Metrics

The MLB-MRO interaction exhibits a special form of conflict since it has characteristics of both PVCs and MVCs. The PVC results from the fact that both SFs adjust the relative HO margin between cells albeit to different extents. As such they will conflict on the desired optimal level of margin for any two neighbor cells. On the other hand, considering the specific parameters as separate (in this case Hysteresis for MRO and CIO for MLB), the SFs exhibit an MVC conflict since

Table 5.1: Optimum HO settings in three mobility scenarios.

Average Velocity (Kmph)	Optimum Hys (dB)	Optimum TTT (s)
3	1.5	1.0
10	1.0	0.64
30	1.0	0.48

any change in one parameter by the respective SF affects the performance of the other SF. The following sections characterize the effects of these conflicts.

We begin in section 5.2with an evaluation of the effect of a generic MRO on load and user satisfaction. without considering any learning, we apply different HO settings and evaluate the respective degree of user satisfaction. Then in section 5.3, we evaluate the effect MLB on HO performance by evaluating the degradation in HO metrics that results from the application of the learning solution (QLB) earlier presented in 4.3. Finally, in section 5.4, we evaluate the cross effects that arise when the two learning solutions are concurrently executed.

In all cases, we consider the network described in 3.1.3 with mobile users randomly placed anywhere in the network. The users move at individually varying velocities but with an average velocity of either 3,10, 30 or 60 Kmph. The specific velocity considered in each case is selected so as to ensure that results are clear and easy to understand. The default HO settings,which are applied in cases where no MRO solution is considered, are set as described by table 5.1. These are the best HO settings for the respective velocities as derived from the HO parameter sensitivity results in 4.2.2.5. To simulate overload, 40 static users are randomly placed in a cell in the center of the network to create a localized load hotspot of approximately 50 m in radius. Simulations are repeated over multiple batches, each batch simulating 200 seconds of operation. In each batch users are placed in the network using the same approach i.e. mobile users randomly placed anywhere in the network and a static user hotspot placed in the center of the network. Different numbers of batches are used in the 3 studies owing to the nature of the solutions - e.g. considering that QMRO requires more learning time compared to QLB. Section 5.2 where no learning is considered applies very few batches while section 5.3 that has only QLB has more batches although still less than section 5.4 in which MRO learning is involved.

In all cases performance is evaluated using the metrics introduced in chapter 4. HO performance is quantified using the HOAP which is the weighted combination of the HO events as defined in 4.2.2.4. Meanwhile, MLB performance is evaluated in a cell or in the network for given evaluation period in terms of the N_{us}. This as earlier defined in 4.3.1.2, is the average number of user un-satisfaction events in the cell/network over that period. In this case, a user is considered unsatisfied (i.e. an un-satisfaction event occurs) only if the user's total achieved data rate in the continuous 1 second period is less than the GBR. The network is configured with HO settings as [0.5dB, 0.1s], which were obtained as the best fixed settings for an environment in which move at 60 Kmph. All users are connected to the

network throughout the simulation, and in case of a RLF, a user is reconnected to the best available cell as expected for LTE.

5.2 MRO Effect on Load and User Satisfaction

MRO may not cause load imbalance but it may result in increased user dissatisfaction. MRO implements actions aimed at minimizing RLFs and PPs, essentially by delaying HOs as much as may be necessary. This is done without concern to the load that any such users may cause in the concerned cells. As such, by delaying or advancing HOs, MRO may increase load imbalance subsequently increasing N_{us}.

To study this effect, we consider a network having a hotspot as described above and having 240 mobile users moving at different velocities but with the overall average of 10Kmph. We apply different HO settings to this network and evaluate the resulting performance in both HO and user satisfaction. Fig. 5.1 evaluates how the HOAP and N_{us} change with HO settings. The result evaluates, for the different HO settings in the network, the average HOAP and N_{us} over a period of 10 batches each of 200 seconds.

We observe in the figure that the more HOs are delayed to minimize RLFLs, the more the N_{us} increases. In effect the optimum HO settings -those that minimize the HOAP- can result in a higher N_{us}.

This result suggests that if MRO learns the required actions with consideration of their likely effect on the N_{us}, a different outcome could be observed. The solution would ensure to minimize N_{us} even as RLFLs are minimized, i.e. the HO delay would have to be reduced in the case of an overloaded cell.

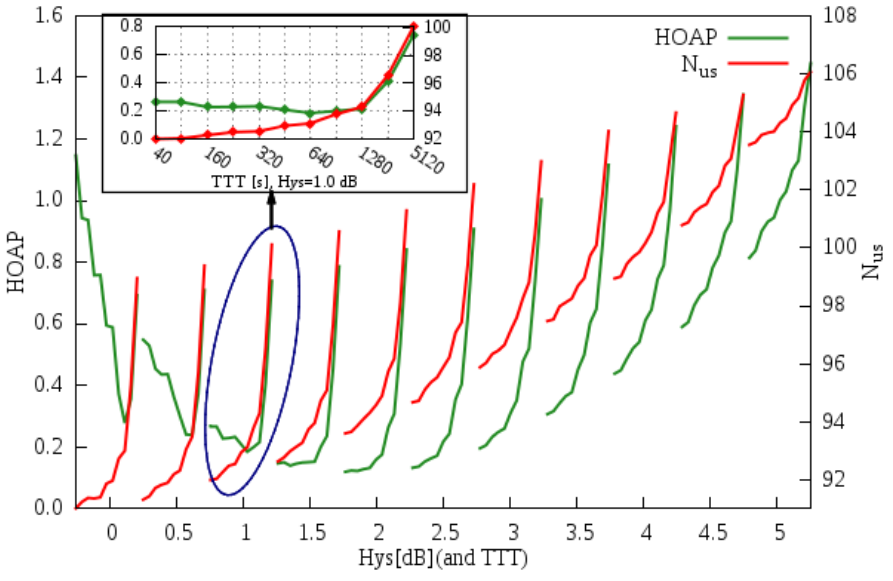


Figure 5.1: Variation of network-wide HOAP and N_{us} for different HO settings.

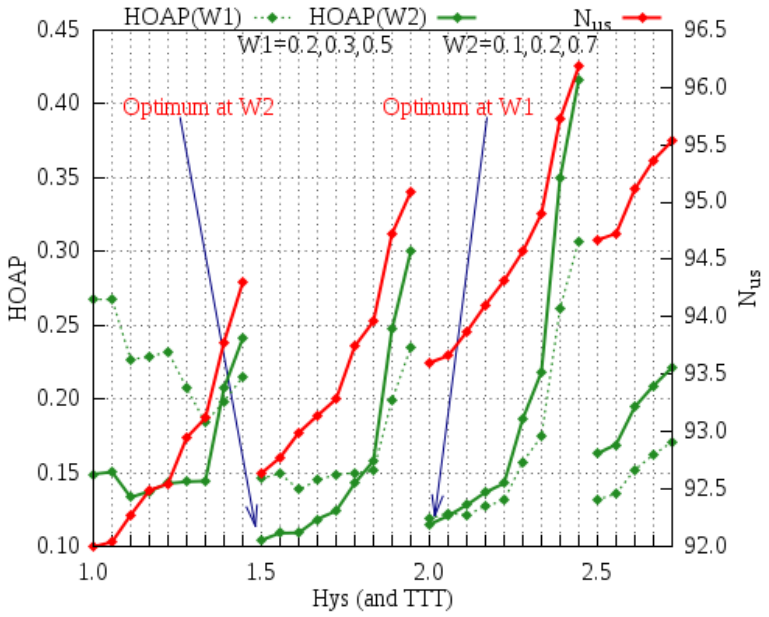


Figure 5.2: Possible reduction in N_{us} based on MRO weight vectors.

Consider the furthermost users at the very edge of a cell. These users could be in the radio sense nearer to the neighbor cell (target cell) than to their current serving cell. They may as such cause less load if they were in the target cell than when in their current serving cell. Then, by initiating HOs earlier in a bid to minimize RLFLs, MRO reduces the load in the serving cell with only a small load increase in the target cell, which may reduce N_{us} in the network. Consequently, MRO can actually minimize N_{us} by just increasing the degree of un-desirability of RLFLs. This is possible by increasing the weight given to RLFL in comparison to the other metrics i.e. PPs and RLFEs as showed in Fig. 5.2.

Fig. 5.2 compares the variation of the HOAP and N_{us} when using two different MRO weight vectors - the default vector W1=[0.2, 0.3, 0.5] as used in Fig. 5.1 and a revised weight vector W2=[0.1, 0.2, 0.7]. We observe that in case of W2, the optimum point would be selected as [1.5dB, 0.4s] for which N_{us} is lower than that at the optimum point of [2.0dB, 0.4s] at W1. This shows that by giving more focus on minimizing early failures and PPs, MRO may inadvertently increase user un-satisfaction that could have been avoided.

5.3 MLB Effects on HO performance

MLB redistributes load among cells by delaying or advancing HOs. By doing so, MLB affects HO metrics since users get connected to cells where, as per their mobility states, they should not be connected to.

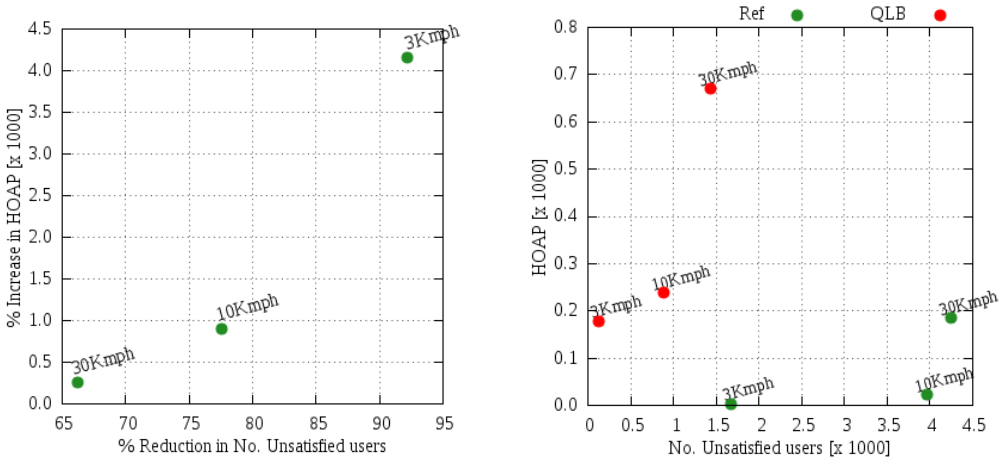
To evaluate the MLB effects we consider a network with the best HO settings as derived from the HO parameter sensitivity results in section 4.2.2.5. In particular we evaluate three velocity scenarios (3, 10 and 30 Kmph) with optimum HO

settings as given in table 5.1. In all cases, we apply the load hotspot as described 5.1 to artificially introduce overload and to allow the load balancing SF to react on the load. We consider the Q-learning LB solution presented in section 4.3. Then, for each velocity, we compare the system applying QLB against the reference network Ref, that does not apply any MLB solution.

We evaluate the results in terms of the related gain and cost of applying QLB. The gain is the percentage reduction in N_{us} while the cost is the percentage increase in the HOAP. This is important since we need to evaluate not the absolute values of N_{us} and HOAP, which are dependent on the scenario, but the relative values which communicate the underlying benefit and costs regardless of the considered scenario.

Figures 5.3 and 5.4 compare the steady state gains and costs of QLB for the different velocity environments over a period of one batch. Evaluating the steady state performance after learning is complete, the Fig. 5.3 evaluates the effect on the weighted HO metric (HOAP). The subfigure 5.3a compares the gains and cost for the different velocity environments while 5.3b highlights the absolute values of the HOAP and N_{us}. Meanwhile Fig. 5.4 evaluates the effects on the individual HO metrics i.e. the PP, RLFE and RLFL rates.

As would be expected, we observe in all three velocity cases in Fig. 5.3a that, QLB achieves its objective of reducing the N_{us}, e.g. approximately 67 % reduction at 30 Kmph and 93 % reduction at 3 Kmph. This however comes at the cost of increased HOAP, where we observe that the cost of applying QLB is more than 100 % increase in HOAP for all velocities. The relative cost (measured by the percentage change) is higher for lower velocities mainly because the HOAP (and its constituent HO) events are lower at lower velocities as seen in Fig. 5.3b. As such small changes in the number of HO have major effects on the resulting performance when compared to higher velocities. Meanwhile, the particular results in Fig. 5.3 show a direct correlation between the achieved gain and the resulting



(a) MLB gains and costs. **(b)** Effect of MLB on HOAP.

Figure 5.3: Effect of MLB on HO Performance.

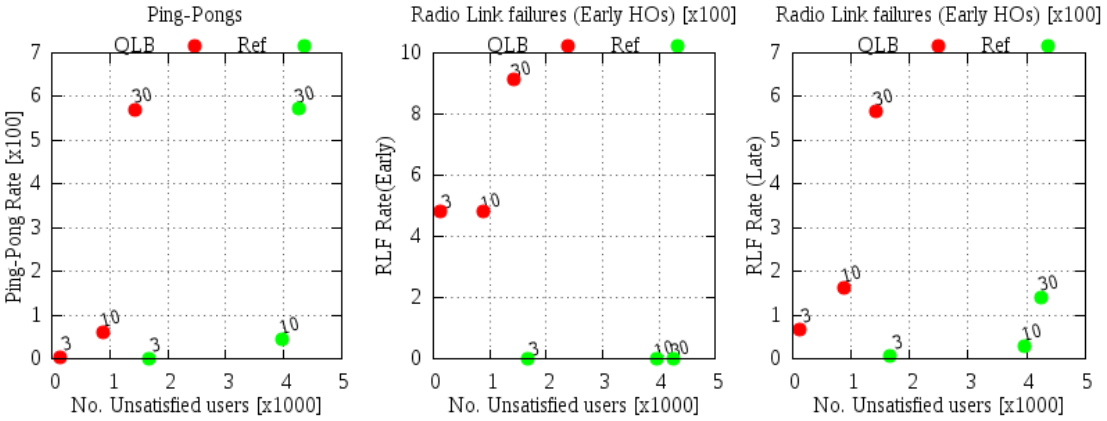


Figure 5.4: Effect of MLB on individual HO metrics.

effect on HOAP. This however is not always the case since the effect depends on the changes in all the three HO metrics shown in Fig.5.4.

We observe from the detailed charts in Fig.5.4 that the major effect comes from the change in the number of RLFEs. In all cases, the increase in the RLFEs rate is higher than that in the other two metrics. This is the direct consequence of triggering HOs too early. In effect a UE is handed over to a cell in which the signal not only fluctuates a lot but is not consistently able to sustain the session. As such, shortly after being handed over to the new cell, the UE's SINR degrades resulting in a link failure. Since the previous serving cell still had a good signal for this UE, the UE reconnects to that original cell indicating that the HO was triggered too early.

Fig. 5.4 also gives an explanation for the excessive relative increase in HOAP at low velocities. We observe that without QLB, all three types of HO events are practically inexistent at low velocities yet their number becomes noticeable on applying QLB.

The above results suggest that, if the QLB solution learns with consideration of the effects of its actions on HO performance, a better outcome could be observed. This may however be at the cost of reduced N_{us} gains. The challenge then becomes how to determine the required compromise between minimizing the N_{us} and minimizing the subsequent increase in RLFEs and HOAPs.

5.4 Cross Effects of Combined MLB and MRO

Sections 5.2 and 5.3 have considered only one dynamic agent to be active at a time i.e. MRO and MLB respectively. Given the observed effects, a much more adverse result should be expected when the two SFs act concurrently in the same network. This section evaluates these effects. First, we consider a reference network (Ref) that does not apply any SON solution. We then consider the same network when both QMRO and QLB are concurrently active which we denote with

QLH. Meanwhile, for reference and completeness, we compare the results above to the cases with the individual SFs - respectively denoted with QMRO) and QLB.

In each case, the network considers mobile users moving with dynamic continuously varying velocities, which enables us to evaluate the effect of MRO in the network. With mobile users moving at an average velocity of 60Kmph (i.e. individually between 36 and 84 Kmph), we set the HO settings in the scenarios without QMRO to [0.5dB, 0.1s]. These are obtained from the HO parameter sensitivity results in 4.2.2.5 as the best fixed settings for an environment in which users move at an average of 60 Kmph. Meanwhile, as described in 5.1, we consider for all scenarios that overload occurs in one cell in the network where the load hotspot of static users is placed at the beginning of the simulation.

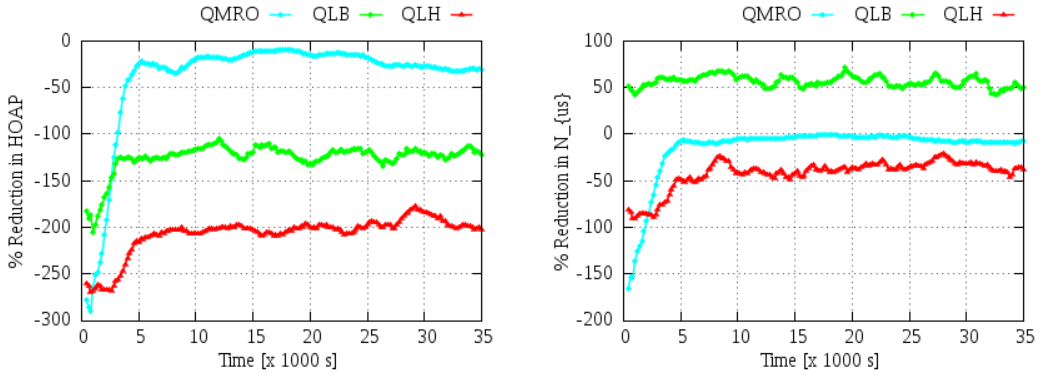
The different solution require different learning periods but simulation trace files have shown that the longest learning time is approximately 16000 s. Nevertheless, to give all solutions ample learning time and allow a long steady state period, the simulation is repeated for 150 batches each batch simulating 200 s seconds of operation. The corresponding HO and LB performance for all the 4 scenarios are given by Fig. 5.5. Figs. 5.5a and 5.5b present the transient performance throughout the simulation while Fig. 5.5c compares the steady state performance after learning has been completed. The results track the HOAP and N_{us} variation by recording the respective values for each batch of the simulation. For clarity however, since we wish to evaluate the effect of the QL solutions on the network, the transient charts in Figs. 5.5a and 5.5b present results:

1. as percentage differences between the reference network values and respective values for the QL solutions (% reductions inHOAP and N_{us})
2. as Simple Moving Averages (SMAs) of the resulting differences in 1 above, with each point indicating the SMA of the previous 10 values (10 batches) of the simulation up to that instant.

For example, if at time $t = \tau$ the N_{us} for Ref and QLB are respectively $N^t_{us,ref}$ and $N^t_{us,QLB}$, the transient N_{us} chart will record $\mathop{\mathbb{E}}_{t=\tau-9}^{\tau}[100 \cdot (N^t_{us,ref} - N^t_{us,QLB})/N^t_{us,ref}]$. Meanwhile, Fig. 5.5c compares the steady state performance of the solutions by considering the averages of the very last 20 batches for each solution. We observe that in both dimensions (HOAP and N_{us}), all solutions perform poorly at the beginning as they test different configurations. They then improve over time but achieve different steady state performances as described in the following subsections.

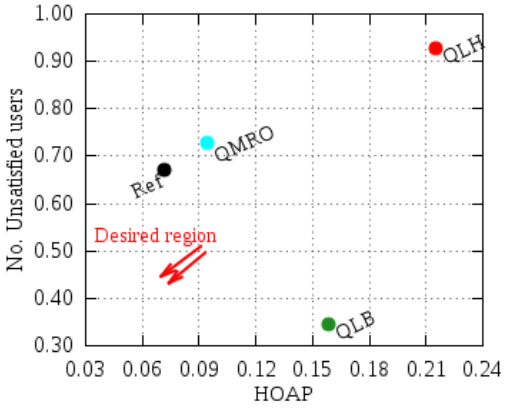
5.4.1 Ref Performance

The performance in the Reference network (Ref) in Fig. 5.5c sets the starting point that requires to be improved by the QL functions. This network applies static settings that remain unchanged throughout the simulation. This network



(a) Transient HOAP performance.

(b) Transient N_{us} performance.



(c) Steady State Performance.

Figure 5.5: Performance of uncoordinated QL functions for 50 m hot-spot in a 60 Km/hr environment.

has an inherent N_{us} that results from the load imposed on the network by the specific user deployment. Similarly, the HOAP basis is the expected performance when configured with the reference HO settings of [0.5dB, 0.1s].

5.4.2 QMRO Performance

As initially explained in 4.2, objective of QMRO is to determine the best HO settings for any mobility state. Thus with the assumption that Rfe settings are optimal as determined in 4.2.2.5, QMRO needs to learn HO settings that achieve the same or even better performance as Ref. We observe in Fig. 5.5a that QMRO endeavors to achieve this objective. It starts out with poor performance as it spans the entire parameter space but improves its performance as it refines the search towards the area that minimizes HOAP. Eventually, it achieves a steady state performance that is close in performance to Ref as shown in Fig. 5.5c.

The failure to achieve exactly the same performance is due to the fact that not all cells contain a single velocity group as was intended. QMRO was designed based on the assumption that all users within a cell have close to the same velocity so

that when a configuration is selected, it would be adequate for all. In the current configuration however, although most cells have only fast moving users at or near 60Kmph, some cells in middle of the network have both fast moving users and slow moving users that cause the hotspot. As such a configuration selected based on fast moving users will cause some poor performance for slow moving users.

We observe in this case however that QMRO has only a small effect on N_{us} (Figs. 5.5b) specifically because no load redistribution has been undertaken. Effectively, QMRO tries to maintain all users in the receptive cells where they are connected in accordance with the aim of maximizing their SINR.

5.4.3 QLB Performance

QLB undertakes action to minimize user dissatisfaction resulting from overload in some cells. We observe in Fig. 5.5b that QLB is able to re-distribute the load in a way that the number of unsatisfied users is reduced - in some batches by more than 50 %. This however comes at the cost of increased HOAP across the network (Fig. 5.5a). Consistently as N_{us} reduces, the HOAP increases. This results in a significantly different steady state performance for QLB as compared to Ref in Fig. 5.5c.

5.4.4 QLH: Uncoordinated operation of QMRO and QLB

The two subsections above have showed that each SF is able to achieve its objective when acting alone in the network even though this may affect the other SF. When combined, a different result is observed, as denoted by QLH in the figures. Without coordination the performance of both SFs degrades. Effectively, during learning, each SFs observes effects that are not strictly due to its actions but that are also due to the peer's actions. As such, the action perceived as optimum for any state is actually worse than what that SF would select when acting alone. In the end it results in the worse performance for both SFs.

In effect the actions selected as optimum instead cause more RLFs in case of QMRO. And for QLB that the actions move users to more loaded cells instead of moving them to less load cells. This is the direct effect of each of the SFs learning based on rewards computed based on events that are not only due to the actions of that SF but that are also due to the actions of the peer SF. Fig. 5.5c gives a simple evaluation of the degree of degradation. While the desire is to have a solution that achieves the standalone performances of QMRO and QLB (i.e. one that follows the directions of the arrows), QLH instead registers worse performance for both SFs. This justifies the need for coordination among MRO and MLB if the respective solutions are to act concurrently within the same network.

5.5 Summary

In this chapter we endeavored to quantify the coordination problem by evaluating the performance degradation that results when multiple SFs are concurrently operated without coordination. Considering the MRO-MLB conflicts, we observed that owing to their nature i.e. being a combination of parameter value conflicts and metric value conflicts, MRO and MLB affect each others performance. We have demonstrated the effects of the SFs on other SF's metrics when acting individually, and then quantified the net conflicts when the two act concurrently. We observed that:

1. that if MRO learns optimum actions with more focus on reducing RLFEs, it could reduce N_{us} in case of an overloaded cell.
2. that by focussing on reducing N_{us}, QLB leads to degradation of HO performance mainly because it significantly increases the number of RLFEs. Consequently, if QLB learns with consideration of the effects of its actions on HO performance, the degree of degradation could be reduced.
3. that by acting concurrently, the SFs degrade each other's performance so much that their combined operation could have worse results that the individual SFs or the manually optimized network.

These observations suggest that the objective of SON may be lost if the parallel operation of multiple SFs leads to worse performance. This justifies the need for solutions to manage conflicts among SFs. Two such solution categories (based on scheduling and cooperative games) will be the subject of the next chapter.

Chapter 6

Multi-Agent SON Coordination

Contents

Consider a system in which multiple SFs are concurrently active. The individual SON solutions assume that each agent acts separately within the environment i.e. that each agents observes effects that are due to only its actions and no actions of other actors. In reality however, the different agents act and observe the same environment – either within a single cell or among neighbor cells. Without coordination, each agent observes effects that are due to both its actions and the peers' actions. Coordination then provides a mechanism of accounting for each SF's effects on the peers or of separating such effects.

The entire SON system can be considered as a MAS in which the agents are the individual SFs within each cell. Then, the generic approaches as discussed in section 2.4 would be the candidate alternatives to coordinate the multiple agents. In particular we consider some form of SAS decomposition achieved through scheduling of SFs in space and time as well as a MAS concurrent games approach. This chapter discusses these approaches and compares their relative performance gains. We

begin with a discussion of the three candidate approaches with in Space - Time Decoupling (STD) followed with a discussion of CCG. We end with a general discussion of their performance limits and limitations.

As was the case with the individual SFs in chapter 4 and conflicts among SFs in chapter 5, we evaluate the performance of the coordination solutions using the LTE system simulator described in chapter 3. For fair comparison we ensure to use the same simulation settings. However, parameters that are of specific concern to this chapter are described as follows:

The network assumes mobile users randomly placed anywhere in the coverage area and moving at an average velocity of 60 Kmph (i.e. individually between 36 and 84 Kmph). This network is configured with HO settings as [0.5dB, 0.1s], the best fixed settings for an environment in which UEs move at 60 Kmph. To simulate overload, a hotspot of approximately 100 m in radius is placed in a cell in the center of the network. Simulations are repeated over multiple batches, specifically 175 batches, each simulating 200 seconds of operation. In each batch users are placed in the network using the same approach, i.e. mobile users randomly placed anywhere in the network and a static user hotspot placed in the center of the network. All users are connected to the network throughout the simulation, and in case of a RLF, a user is reconnected to the best cell as expected for LTE.

In all the evaluations, the MRO weight vector of (0.2, 0.3, 0.5) is applied. The respective metrics have been selected first to balance opposing metrics (i.e. RLFLs against the combination of RLFEs and PPs), and then owing to the more undesirable effects of RLFs compared to Ping-pongs, RLFEs are weighted higher than PPs. Other critical parameters are summarized in table 6.1.

Table 6.1: Simulation Parameters.

Parameter	value
System bandwidth	10 MHz
Inter-site distance	500 m
Time between snapshots	50 ms
Number of users	420
Mobile users' velocity	variable, mean 60 Kmph
Mobility Model	random walk
Pathloss	128.1 + 37.6 $log_{10}[max(dKm, 0.035)]$
Shadowing standard deviation,	50m
Decorrelation distance	50m
eNB Tx power	46 dBm
eNB Tx antennas	1 per sector at height = 32 m
UE receive antennas	1 Omni at height = 1.5m
eNB max. antenna gain	15 dBi
UE max. antenna gain	2 dBi
Data rate	512 Kbps

For each of the solutions that have been considered, performance is evaluated from two dimensions:

1. the temporal variation of the metrics throughout the simulation. Considering gains as the reduction in HOAP and N_{us} for each batch, the temporal variation tracks the Simple Moving Average (SMA) of the gains in each of the two metrics using an SMA window of 10 values (i.e. 10 batches).
2. the steady state performance when all SFs are considered to have completed learning. This considers the average of the very last 20 batches for each of the two metrics, plotting them on a 2-dimensional grid.

6.1 Spatial-Temporal Decoupling

As discussed in section 2.4, the simplest coordination solutions typically consider some form of SAS decomposition. This section discusses our SAS decomposition approach which is based on STD. To minimize the conflicts among the SFs, STD procedures separate the execution of the different SFs or SF instances in space and time, i.e. the different SF instances are executed at different points in either time, space or both. The three approaches that have been evaluated are:

1. Temporal Separation during Learning (TSL)
2. Concurrent learning with Spatial Separation (CSS)
3. Space - Time Scheduling (STS)

The following sections describe the concepts in each of the three cases where their relative performance of is evaluated using the two QL SFs (QMRO and QLB) as described in chapter 4.

6.1.1 Temporal Separation during Learning (TSL)

If two conflicting SFs concurrently take actions in the same cell, they cannot fully differentiate the effects of their actions on the cell, i.e. the learning agents get "confused". For this reason a number of studies have proposed separation of the SFs such that at any time only one is allowed to act. However, any SFs is likely to under perform during the time when it is deactivated. For example consider that MLB is deactivated so as to wait for completion of an MRO episode. Typically, MRO requires time for at least 1000 HOs in order to obtain adequate statistics on HO metrics. With MLB deactivated during this period, even if free resources exist in the neighbor cells, any affected cell will not take any action even if it experiences excessive overload. This is even worse if the SF is deactivated not only in one cell but throughout the whole network. To counter this, we have proposed that for learning agents, the time based separation of execution should be undertaken only during the learning period, thus TSL.

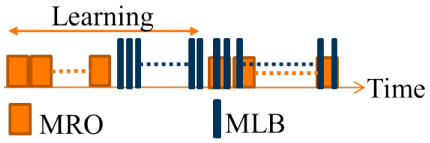


Figure 6.1: Time Separation during Learning between QMRO & QLB.

6.1.1.1 Operation of TSL

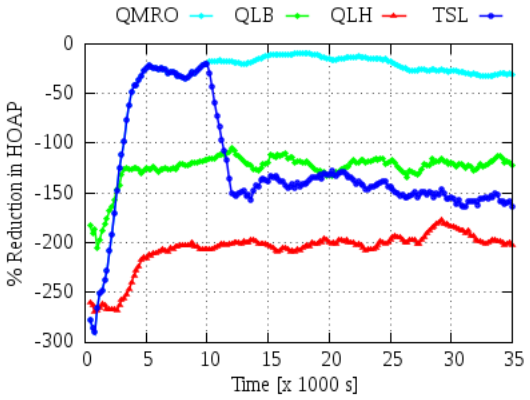
The idea behind TSL is that different SFs are scheduled to operate at different time during their learning period. All SFs are then fully activated after the learning is completed. Effectively, only 1 SF learns at a time. Prioritizing the SF that should learn first before the other is assumed to be an operator policy. In general however, this should consider which SF is likely to be more affected by the other. For MLB and MRO, since MLB adjusts the HO boundary between cells, it should do so in an environment of optimum HO performance. As such, QMRO is selected to learn first so that QLB learns only after HO triggering is assumed optimized as shown in (Fig. 6.1).

Given the cooperative learning among cells, each cell executes the MRO algorithm at the start of the optimization and updates the shared QMRO-policy function (the QMRO Q-table). This continues until QMRO learning is completed (i.e. until the Q-table is fully updated for observed mobility states). When that happens, QLB learning is then activated. In that case the cells take actions and update the shared QLB-policy function (the QLB Q-table). When the QLB Q-table is fully updated, both SFs are let to freely run concurrently. The assumption here is that each SF would have had a chance to learn alone on the network without interference from the other.

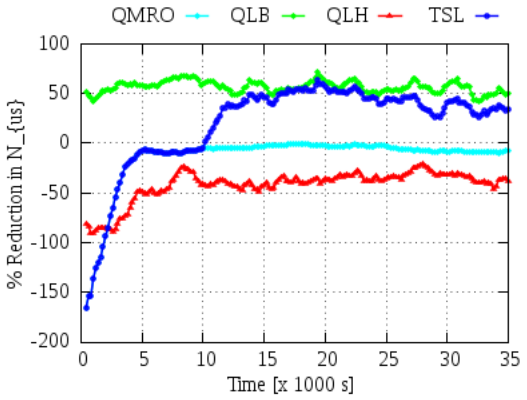
6.1.1.2 TSL Performance Results and Discussion

Fig. 6.2 compares the performance of the two SFs for: 1) the independent SFs when each is acting alone in the network, 2) the combined but uncoordinated operation and 3) the coordinated operation using TSL. Specifically Figs. 6.2a and 6.2b show the performance throughout the simulation using the SMAs of the metrics while Fig 6.2c evaluates the stead state results, which consider the averages over the last 20 batches.

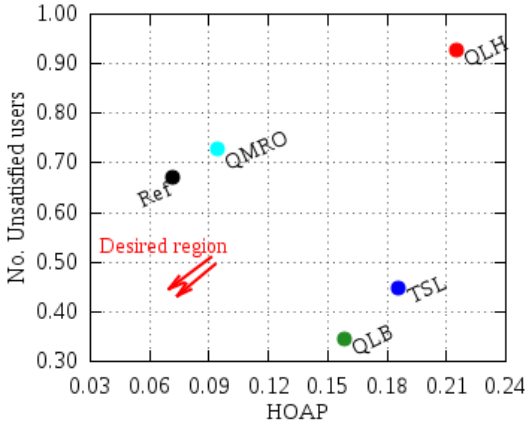
We observe from Figs. 6.2a and 6.2b that at the start performance of TSL matches that of QMRO since QMRO is the only active SF. However, when QLB is activated, TSL's performance begins to follow QLB and the two moving average curves move towards the QLB curves. The end result is that there is major degradation in HO performance to the extent that the HO steady state result is far worse than both QMRO and Ref, although it is better than the uncoordinated case represented by QLH.



(a) Transient HOAP performance.



(b) Transient N_{us} performance.



(c) Steady State Performance.

Figure 6.2: TSL Performance for 50 m hotspot in a 60 Kmph environment.

The challenge with TSL is that the last SF to learn changes the environment for the first SF(s). For example QMRO learns in an environment where its the only actor, yet after it has finished learning, QLBs joins that environment. The initial policy learned by QMRO is thus no longer applicable in this new environment, implying that it may be better for the SFs to learn concurrently. Moreover this is made worse by the fact that execution is undertaken concurrently in all cells. This implies that when one SF evaluates the effects of an action, its environment would have been affected by that action, the action(s) of the other SF(s) with in the same cell as well as the actions of other SFs in neighbor cells. The alternative approach then is to consider separating SFs in space as described in the next section.

6.1.2 Concurrent learning with Spatial Separation (CSS)

Among neighbor cells, actions of a SF in one cell A can affect metrics of a similar or different SF in another cell B. In general, consider two SFs SF 1 and SF 2 active in the two cells A and B. SF 1 that is active in A can have effects on SF 1 in B and on SF 2 in both A and B. For example MLB in cell A affects MLB in the

neighbor cell B as well as MRO metrics (HO performance metrics) in both cells A and B. In that case if either SF 1 or SF 2 is active in B and it takes an action from which it is supposed to learn, the observations made on its metrics will have been affected by both SF 1 in A and the active SF in B. The SFs then make the wrong deductions and learn the wrong functions. In that case, execution in the two cells A and B should be separated so as to minimize these effects. We consider the case of CSSs, where one cell out of a set of neighbors is active but with all SFs concurrently running within this active cell.

It should be noted that there could be two SFs that do not conflict across cells, which could without negative effects concurrently run in neighbor cells. This however requires that such knowledge is available which is generally not the case. As such the discussion here considers the generic case in which all SFs are assumed to have effect across cells however small such effects may be.

The minimum requirement for CSS is that actions should only be taken concurrently in two cells, if there exists at least 1 other cell in between the two. This is based on the assumption that besides the effects on self actions within a cell, effects from other cells would only be due to action of the first tier neighbors. There could as well be effects from further neighbor tiers but this seems to be an overly cautious assumption without a justified cause. Given the assumption above, the result is a reuse-3 concurrency structure among the cells as shown in Fig. 6.3. Essentially only 1 out of every 3 cells will be active in any one time interval. The set of active cells, referred to here as a cluster, is then rotated over time. Each cluster of cells is allocated a time-frame within a single multi-frame as shown in Fig 6.4. Each cell then schedules SFs within the frame according to its local requirements.

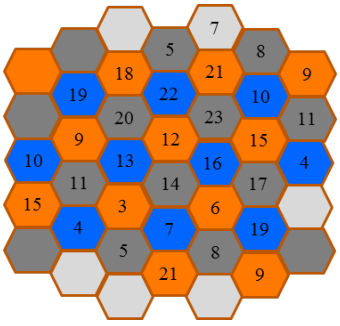


Figure 6.3: Cell clustering for spacial scheduling.

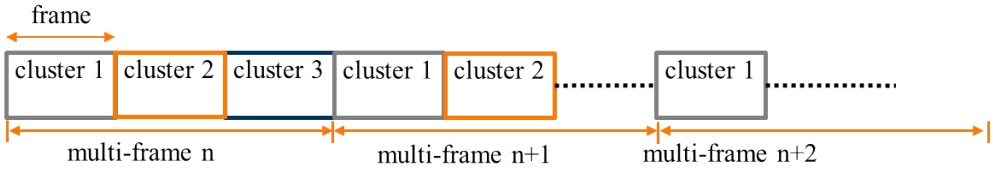


Figure 6.4: CSS Multi-frame.

Algorithm 6.1: Clustering cells for CSS

Require: *Seed cell* a; *sets* $X_i, i = 1, 2, 3$ *for the 3 colors*

1. add seed a to X_1
2. Select c = any neighbor of a i.e. any cell with ***exactly***1 colored neighbor
3. add c to X_2
4. **Repeat while not all cells colored / allocated**
5. **for** each uncolored cell c with ***exactly*** 2 colored neighbors; **do**
6. given neighbors' colors as $X_{i=j}, X_{i=k}$
7. Allocate c to third color $X_{i=l}$

done

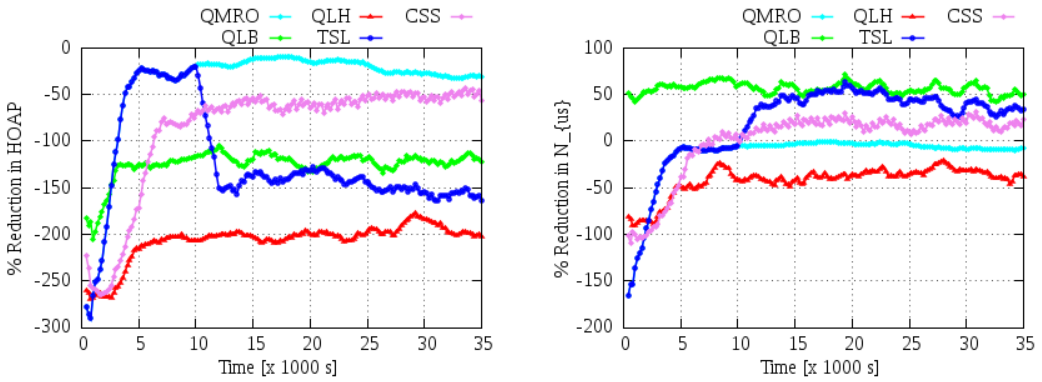
6.1.2.1 Clustering Cells for CSS

The clustering structure in Fig. 6.3 requires that cells are pre-associated with the clusters. The following procedure describes a clustering scheme that allows one of any 3 neighbor cells to act. It assumes that the clustering is executed centrally as part of the network configuration. Essentially each time, a new eNB is added, the clustering scheme of *Algorithm 6.1* is executed to allocate cluster indices (or colors) to the cells. The algorithm requires a seed which can be selected randomly or by the operator. The seed is allocated to the first cluster index and the subsequent cells colored depending on their neighbors' colors. In the uniform hexagonal grid as here considered, the clustering scheme achieves the same cell distribution of Fig. 6.3 among the cells regardless of the applied seed.

6.1.2.2 CSS Performance Results and Discussion

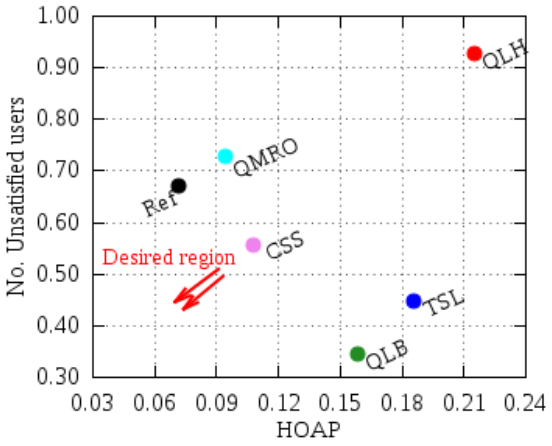
Fig. 6.5 compares the performance in the two SFs when coordinated using CSS with the performance of all the previous solutions i.e. 1) the independent SFs when each is acting alone in the network, 2) the combined but uncoordinated operation and the coordination using TSL. As before Figs. 6.5a and 6.5b evaluate the entire simulation including the transient state while Figs. 6.5c evaluates the steady state performance using the last 20 batches.

We observe from Figs. 6.5a and 6.5b that unlike with TSL, CSS performance does not follow that of any of the SFs because in this case both SFs are concurrently active. Instead, CSS seeks a compromise performance between the independent solutions and the uncoordinated environment. However, the most visible effect of the space separation is that both solutions take longer to converge. This is the direct effect of having only a few of the cells learning and contributing to the shared Q-function. In the end however, CSS achieves a better compromise but is also unable to achieve equivalent performance to the standalone SFs. This is owing to the fact that within an active cell, both SFs are still operational and as



(a) Transient HOAP performance.

(b) Transient N_{us} performance.



(c) Steady State Performance.

Figure 6.5: CSS Performance for 50 m hotspot in a 60 Kmph environment.

such will affect each other's observations. The next alternative then is to fully separate the SFs both in space and in time.

6.1.3 Space - Time Scheduling (STS)

Sections 6.1.1 and 6.1.2 have considered separation of the SFs in one of the two dimensions - space and time. They each make small improvement but are unable to achieve results that are comparable to those achieves by the independent SFs. The challenge in both cases is that SFs are unable to independently observe their metrics yet this is very important during the learning period. In the case of TSL, each observation is affected by action in the neighboring cell, while for CSS the observation is affected by the other SF within the same cell. To counter these degradations, we combine time separation and special scheduling, thus STS, to ensure that during any observation period, each cell is affected by the action of only one SF.

6.1.3.1 Operation of STS

As earlier proposed, we apply separation only during the learning and reactivate all SFs in all cells after learning. This ensures that SFs learn functions that are only due to their actions and that after learning SFs do not under perform as a result of having been deactivated to wait for other SFs. We apply CSS among the cells and apply TSL among the SFs in each cell.

As was done in CSS, cells are clustered as in Fig. 6.3 using *algorithm 6.1*. Each cluster is allocated a time frame as shown in Fig. 6.6, such that each cell has 1 out of the 3 frames of the multi-frame within which to take actions. For example, in Fig. 6.6, cluster 1 cells a and k always take action in frame 1. Within each frame, the cells apply time division scheduling to allocate time to each of the active SFs.

It is imaginable that all cells in a cluster will have the same SF acting. We however, do not consider this. Instead, we assume the generic case that cells could be having different optimization needs so should schedule SFs accordingly. As such, each cell will determine how to schedule SFs within its allocated time-slot based on those optimization needs but also based on the timing requirements of the different functions. For example in Frame 1 in Fig. 6.6, cells a and k respectively schedule MRO and Coverage-Capacity Optimization (CCO). Contrarily, cell k schedules multiple SFs (e.g. twice MLB and once Inter-Cell Interference Coordination (ICIC)) at a time when cell a schedules CCO.

For the MRO-MLB conflicts, we assume as before that MRO learns first to ensure that MLB adjusts HO boundaries within an environment of optimum HO settings.

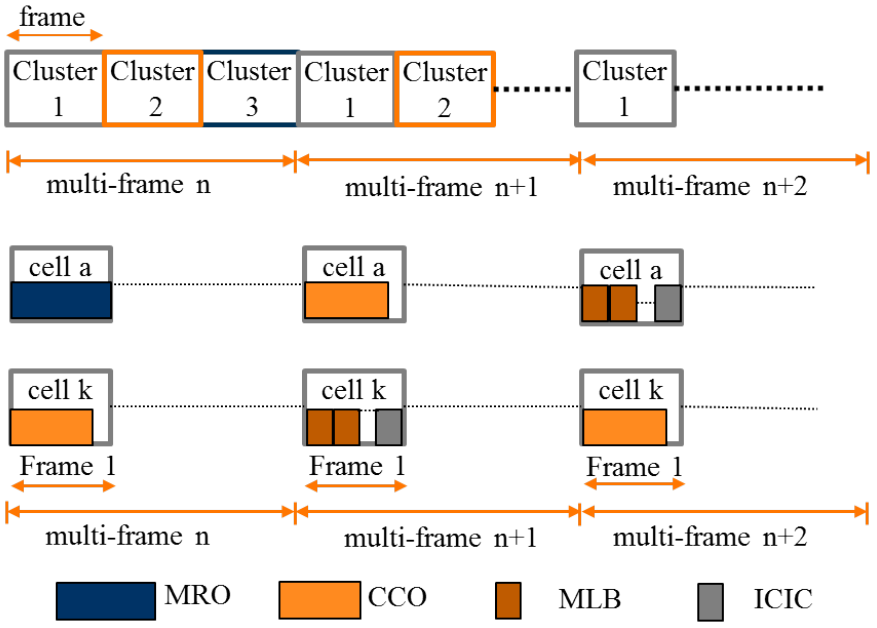
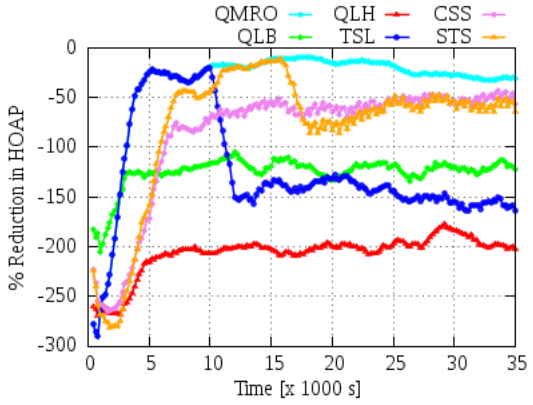


Figure 6.6: STS Frames in CSS Multi-frame.

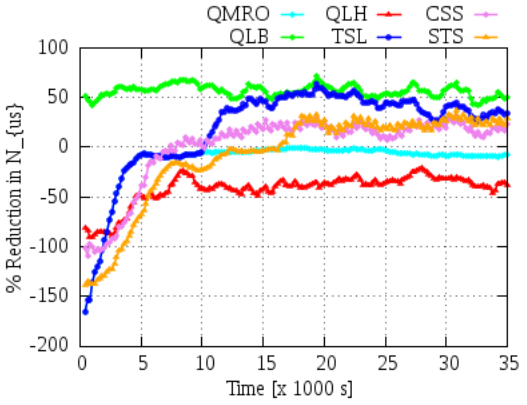
6.1.3.2 STS Performance Results and Discussion

As was the case with TSL and CSS, we evaluate STS's performance over the entire simulation using SMAs (Figs. 6.7a and 6.7b) and evaluate the steady state performance as averages of the last 20 batches(Fig. 6.7c). Again we compare STS's performance to: 1) the independent SFs, 2) the combined but uncoordinated operation and 3) the previous coordination solutions (i.e. TSL and CSS).

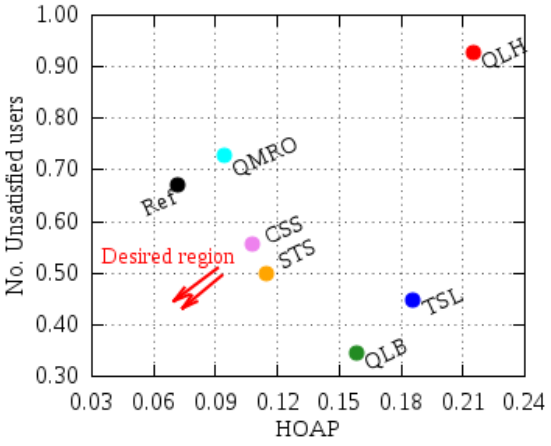
By separating both in space and time, STS enables SFs to learn the strictly independent behaviors that are free of peers' actions. This results in better compromise compared to TSL and CSS as seen in Fig. 6.7c. In general, because each SF is given a chance to learn its independent policy function, STS achieves good performance even when multiple SFs act concurrently in the network. This can be seen better in a lower velocity environment as shown in Fig. 6.8 for a network with mobile users moving at 30Kmph. The lower velocity translates into a smaller effect of load balancing on HO performance (see QLH in Fig. 6.8 compared to that in Fig. 6.7c). STS then allows the SFs to learn better solutions that allow the effect to be reduced further, resulting in a better compromise compared to CSS. However, because during operation each SF's actions are superimposed onto



(a) Transient HOAP performance.



(b) Transient N_{us} performance.



(c) Steady State Performance.

Figure 6.7: STS Performance for 50 m hotspot in a 60 Kmph environment.

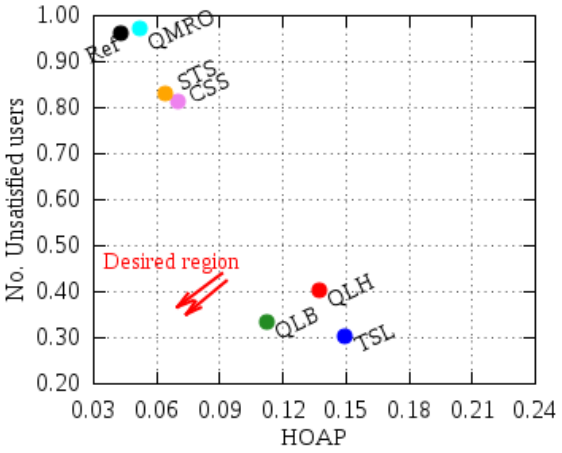


Figure 6.8: Steady State STS Performance for 50 m hotspot in a 60 Kmph environment.

peers' actions, the performance gains will still not be equivalent to those achieved when the SFs act independently in the network.

6.2 Multi-Agent Concurrent Cooperative Games

The scheduling of SFs assumes that conflicts occur during the execution of actions, i.e. that if two SFs act within the same environment but at different times, that the SFs would not have any effect on one another. This may however not always be the case since there would be SFs that conflict e.g. on the metric values even though they are executed at different times.

As described in chapter 5, even when acting individually, SFs can affect each others' metrics. Consider for example that MLB changes the CIO in a cell. Even if that action is taken independent of any MRO action at that time, its effect on HO performance will be observed even at later points in time. This then requires that such an action should have been taken with concern for HO metrics. In this case Spatial-Temporal Scheduling may be inadequate to the task. Each SF needs to account for its effects on all other SFs even if the are not competing at the same space-time coordinate. This is what CCG aim to accomplish.

Moreover, CCG may also allow operator to decide how much compromise to be undertaken between the competing SFs /metrics. STS's scheduling does not offer the operator any control on the actions of the SFs in accordance with the desired level of compromise. This can to some degree be implemented through CCG.

6.2.1 Concurrent Cooperative Games in SON

Consider a network with a set of cells, C, such that when a SF i in cell j (SF_{ji}) takes an action, it affects its peer $SF_{jl}, l \neq i$ and SF_{ci}. $SF_{jl}, l \neq i$ are the other

SFs in cell j while SF_{ci} are SFs in the other cells $c \in C$, $c \neq j$. Consider also that SF_{ji} acts alone in the respective environment.

With STS, SF_{ji} takes an action and at the end of the monitoring period, it evaluates the effect of the action on its own metrics without concern on the effect of the action on other SFs. With CCG however, SF_{ji} wishes to learn the action that: 1) has the best performance considering SF_{ji}'s metrics and 2) also has the least possible effects on the other SFs. To achieve this SF_{ji} must be able to determine any such effects of its actions on the peers.

To learn the effects, after selecting an action to execute in the environment, SF_{ji} communicates that action to all its peers who at the end of the monitoring period report back their observed effects. SF_{ji} then aggregates these effects in evaluating the quality of its actions. An example of this procedure is described by Fig. 6.9 for the generic CCG between two SFs *SF1* and *SF2*.

When *SF1* takes an action it informs *SF2* of the action taken, prompting *SF2* to monitor and evaluate its metrics. Then at the end of the observation period *SF2* informs *SF1* of the effect of the action on *SF2*'s metrics, which *SF1* considers in evaluating the quality of the action. In principle *SF1* and *SF2* could each be an instance of the respective SF or the full SF with all its multiple instances e.g. with one instance in each cell. In either cases, each SF instance that receives the message from the active SF must report back its observed effect, so that the penalty is derived from the aggregation of the reports. Consider two cells each with an instance of MLB and an instance of MRO as shown in Fig. 6.10. For a MLB action in cell A, we expect to receive responses from MRO in both cells A and B as well as MLB in cell B.

Meanwhile, to achieve effective coordination with CCG, two critical issues must be addressed: 1) how to differentiate actions from multiple concurrent SFs and 2) how to structure and aggregate information from multiple affected peers. These concerns are addressed in the foregoing sections.

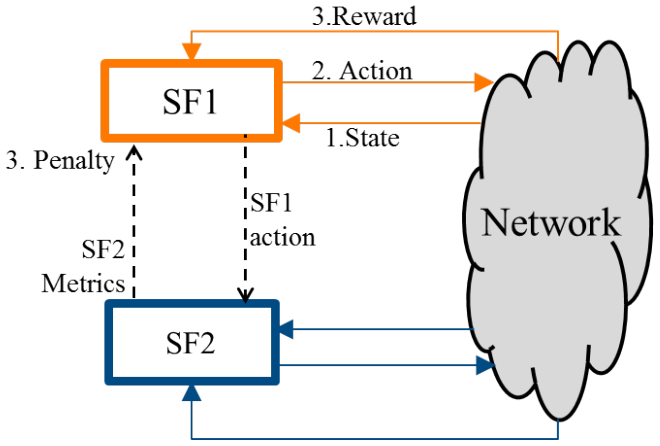


Figure 6.9: CCG principle.

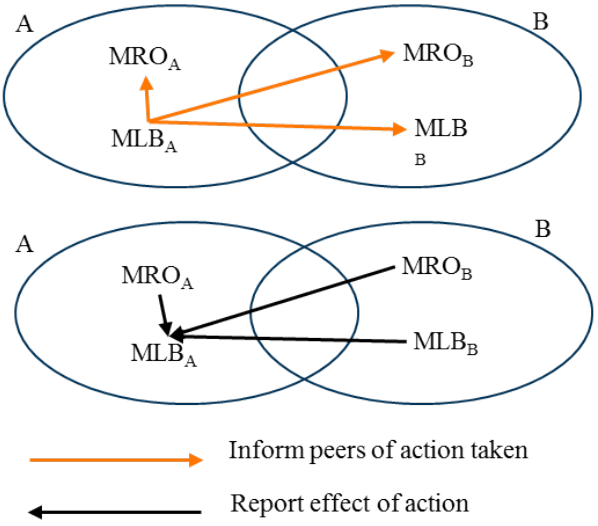


Figure 6.10: Message exchanges in 2 cells.

6.2.2 Managing Concurrency of Actions

Consider that SF_{ji} has taken an action and expects a report from a peer SF_{cl}. We require that during the observation interval SF_{cl}'s metrics are not affected by any other agent except SF_{ji}, otherwise SF_{cl}'s report would be misleading. It is appropriate to assume that each SF affects only SFs within its cell and the first tier neighbor cells and not any other cells further out. This is a justified assumption except for a few radio propagation based SFs like interference management where the signal may propagate beyond the first tier neighbor cells. Even then, compared to the effect in the first tier neighbors, the effects in the second and higher tier neighbors is so small that it can be neglected. Given this assumption and the requirement that SF_{cl} is only affected by SF_{ji} (at least during the learning period), it follows that, SFs should only be scheduled in a way that no two SFs concurrently affect the same environment during the learning.

6.2.2.1 Spatial-Temporal Scheduling

To address the need to separate actions among SFs, consider the subnetwork represented by Fig. 6.11a. We wish to ensure that an SF executed in cell A is able to learn the effects of its actions on: 1) all SFs in cell A and 2) all SFs in the neighbor cells. To achieve this none of A's neighbor cells should execute any SON activity and none of the others SFs in A should also run, otherwise this would negate requirement 1. Moreover, to fulfill requirement 2, none of A's tier-2 neighbor cells (e.g. C) should execute any SON activity since such activity would affect A's tier one neighbors' SFs (SFs e.g. in B). In effect the nearest SON activity should be within A's tier-3 neighbors (e.g. D) since such activity would only affect tier-2 neighbors which are outside A's reporting area.

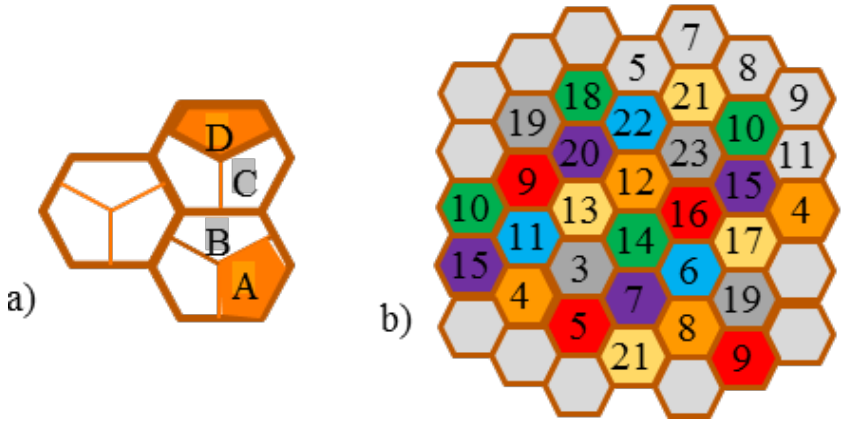


Figure 6.11: CSS: Spacial scheduling in cell.

Correspondingly, spatial scheduling for GGC should ensure that there are at least 2 inactive cells between any two active cells as shown in Fig. 6.11a. This requires a reuse-7 structure, which in a network with a uniform hexagonal grid is equivalent to Fig. 6.11b. In Fig. 6.11b, concurrent activity is only allowed among cells with the same color, which are the cells that are exactly 2 cells apart. In this case, each cell is able to measure its effects on the SFs in neighbor cells without influence from any action in the other concurrently active cells. For example when cell 12 takes action, it can measure independent reports from cells 1,14,16,23,22 and 20, none of which would have been affected by ongoing activity in cells 4 and 8.

6.2.2.2 Spatial Scheduling Algorithm

To achieve the reuse-7 scheduling, we have designed the clustering scheme of *Algorithm 6.2* for spatial scheduling. Assumed to be executed centrally as part of the network configuration (each time a new eNB is added), the scheme selects only one of any 7 neighbor cells to act. The set of all cells that will be concurrently active is referred to as a Cluster List, identified by a Cluster List index (CLI), while each cell that is allocated to an index is refereed to as having been colored. The algorithm allows the operator to select a seed (the starting cell) which is allocated to the first CLI. Then, beginning with the first ring around the seed and proceeding outwards, the cells are allocated to CLIs depending on the CLIs allocated to their neighbors.

To allocate the first cell in the first ring, the CLI is first incremented to 2. The algorithm then randomly selects the first cell in the first ring and allocates it to that CLI (i.e. CLI 2). Subsequent cells in the first ring are also selected randomly and allocated to the subsequent index up to the seventh index.

In the outer rings, any cell that has at least 2 colored neighbors is selected for coloring. The cell is supposed to have a color that is different from those within

Algorithm 6.2: Clustering for Spatial Scheduling in CCG

Require: *Seed cell a; sets* $X_i, i = 1, 2, ..7$; *for each cell, Lists* $T1$, $T2$ *for Tier1, Tier2 Neighbors CLIs respectively*

1. Add seed a to X_1
2. Add i=1 to $T1_j$ for each of seed's neighbor j
3. **Repeat for** each i's neighbor j - first ring around seed
4. i← i+1
5. Add j to X_i
6. Add i to $T1_k$ for each j's neighbor $k, k{\neq}i$
7. Add $T1_j$ to $T2_k$ for each j's neighbor $k, k{\neq}i$

end loop

8. Select any second ring cell m
9. i = first CLI that is not used among m's neighbors
10. Add m to X_i
11. Add i to $T1_j$ for each m's neighbor j
12. Add $T1_m$ to $T2_j$ for each m's neighbor j
13. **Repeat while** not all cells allocated
14. **for** each cell m not allocated **do**
15. i = first CLI that is not used among m's neighbors
16. Add m to X_i
17. Add i to $T1_j$ for each m's neighbor j
18. Add $T1_m$ to $T2_j$ for each m's neighbor j

done

end loop

its first two neighbor tiers. A list of forbidden indices (colors) is created for the cell and populated with the indices of the cell's first two tiers of neighbors. The cell is then randomly allocated to any index that is not within its forbidden-index list. This continues until all cells have been colored.

The output of the Clustering scheme is the cell distribution of Fig. 6.3b for a network with a uniform hexagonal grid. This is always the same for such a network regardless of the applied seed.

6.2.3 Type and Aggregation of Reported Information

The complexity of the CCG algorithm can be controlled by the structure of the communication especially the response sent to the active SF, SF_{ji}. Even the way the information in the reports from the peers is aggregated will highly depend on how that information is structured.

6.2.3.1 Type of Shared Information

The return message sent by the peers to the active SF, SF_{ji}, could contain either the metrics of the peer SFs directly or a penalty value based on some function that measures the effects. The structures of the two approaches significantly differ and so do their relative merits:

1. **Communicating the metrics:** Here, following an action from SF_{ji}, each affected SF communicates its observed metrics to SF_{ji}. So to derive the quality of the action, SF_{ji} requires, for each peer, a local model of the goodness of the peers' metrics i.e. a model that describes which metric values are good or otherwise. Thus, using these local models on the received metric values, SF_{ji} derives the associated quality of the action it took. For the example in Fig. 6.10, both MRO instances would directly report the PP and RLF rates while MLB in cell B would report the unsatisfied user rate observed over the SON interval. SF_{ji} would then have to use these rates to evaluate the aggregate effect on the peers. Note should be taken that the structure is the same if, instead of reporting the metric values, the peers reported changes in metric values, i.e. between the start and end of the SON interval. SF_{ji} would still require models that interpret how much change is good, acceptable or otherwise.

   One challenge here is that need for local models in SF_{ji} for each of the peers makes SF_{ji} very complex, in principle almost as complex as the entire set of SON functions. It would require a lot of expert knowledge to be added to each SF as a result of the need to include the peer models. The biggest concern however would be that this structure is not easily scalable. Since each SF requires models of all peers, each time a new SF is added, its model must be included in all existing SFs. This makes the integration of new SFs very complex and the entire solution undesirable. Moreover, ensuring that the interpretation of the metrics is consistent among all SFs would be challenging too. This is especially the case where SFs are implemented by different vendors or teams.

2. **Communicating the quality of effects:** In this case, instead of communicating the metrics, the affected SFs only report a metrics quality indicator. A generic report structure that allocates credit for observed effects would as such be required. An example of such a generic structure for grading the appropriateness of an action on the metrics can be a scale of 5 values between -2 and +2 that respectively corresponds to the grades 'very bad', 'bad', 'neutral', 'good' and 'very good'. Such a structure allows each SF to define its internal meanings of the 5 grades offering the aggregating SF a single scale on which to compare effects across all peers. Using such a reporting structure, after the monitoring period, peers signal these grade reports to the offending SF who then aggregates the grades to determine how good or bad the action was to the peers.

Reporting effects based on a metrics quality indicator eases the integration of SFs in an environment where SFs are gradually added to the network. This is because

no extra refinement is required for the existing SFs in order to be able to account for effects on the new SFs and vice versa. This therefore is the preferred option and is the considered form of reporting among SFs in the foregoing studies.

6.2.3.2 Aggregating Peer Information

Besides designing a good reporting structure, an appropriate objective function for aggregating the quality indicators from the multiple affected SFs must be also designed. Learning the actions with the least effects on peers, highly depends on how those quality indicators from the multiple SFs are combined during SF_{ji}'s learning. In general this would be operator policy, typically as a weighted multi objective optimization function, for which the operator sets the weight vector.

In our studies, to reduce complexity and evaluate the benefit of the proposed solution, we have only considered coordination of SFs that are within the same cell. For example considering the CCG in Fig. 6.10, it means that if MRO takes an action in cell A, it only receives feedback from MLB within the same cell A, and does not consider effects on MRO and MLB in the neighbor cell although in principle the neighbor cell is also affected.

6.2.4 CCG Algorithm

The reuse-7 spatial scheduling combined with time scheduling and the reporting structure in 6.2.3.1 enable SFs to independently act in a given area and measure their effects on all peers within that area. To ensure concurrent separation in time, the time division concept represented by Fig. 6.12 is applied. This is similar to the STS frame structure in Fig. 6.6 except that the composition of clusters changes

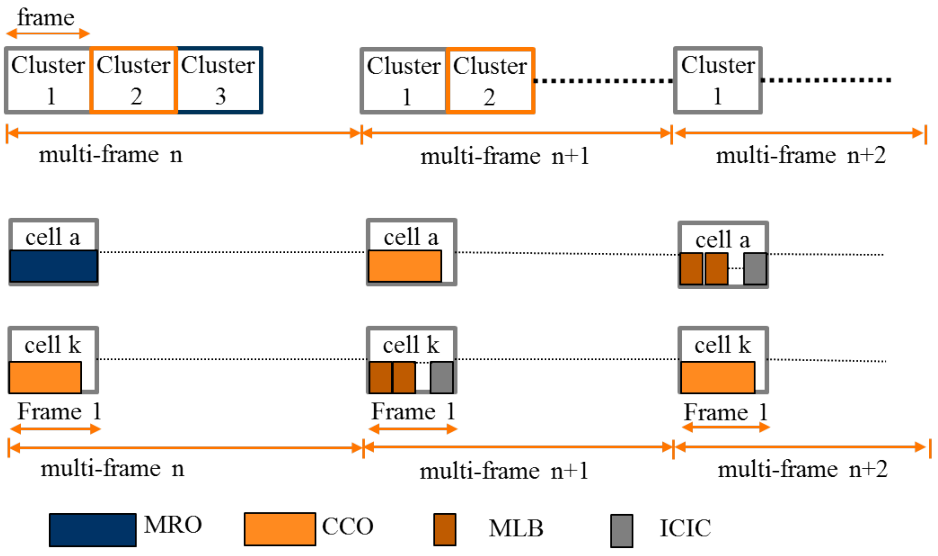


Figure 6.12: Reuse-7 STS Frames in a Multi-frame.

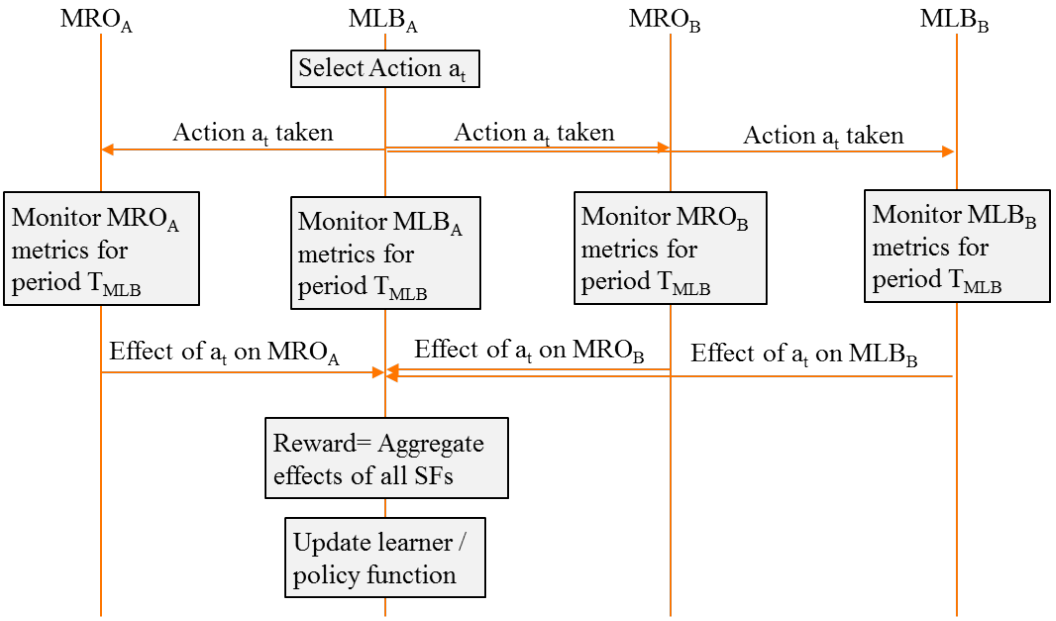


Figure 6.13: CCG activity flow.

according to *Algorithm 6.2*. In this case, each cell cluster is allocated to one of seven frames of the multi-frame in which to take actions. Within each frame, the cells apply time division scheduling to allocate time to each of the active SFs. As before, we assume a generic case in which cells in any cluster do not all have the same active SF at any given time. Instead we assume that each cell schedules SFs according to its own requirements.

Considering the two cells A and B each with an instance of MLB and an instance of MRO as originally introduced in Fig. 6.10, the operation of CCG may be described by the Time line chart in 6.13. Consider that the MLB instance in A (MLB_A) is to take an action and learn from that action. After selecting the action, MLB_A informs the other SF instances that it has taken an action, which triggers the SFs to monitor their respective metrics. At the end of the observation period (equivalent to the designated SON interval for MLB), each of the SF instances reports the effect of the MLB action using the reporting structure in 6.2.3.1. Then, to obtain the reward to be applied during learning, MLB_A aggregates the effects based on the function designed according to 6.2.3.2. This reward is then used in updating the Q-learning policy function.

6.2.5 Simulation studies and Performance Results

As in the STS studies, to study the performance of CCG, we consider the two SFs that were developed in chapter 4 i.e. QMRO and QLB. The studies here consider only the case in which the SFs account for effects on co-located peers. This means that each MLB instances only considers the effects on MRO within the same cell and vise versa. This may definitely not be sufficient to ensure that all effects are accounted for. However, accounting for all effects requires a detailed study of how

metric effects from all affected SFs should be combined. Such a study could not fit in the scheduled time for this project and has as such been left to future research.

Meanwhile, to account for the cross effects, the reward functions for both MLB and MRO as presented in chapter 4 are revised. The revised reward functions are described below followed by the observed results.

6.2.5.1 QMRO Reward Function

Besides minimizing RLFs and PPs, QMRO now intends to also minimize the occurrence of unsatisfied users by reducing the load in an overloaded cell. As such the reward function should also consider the load reduction that results from any action that is selected. Since the number of unsatisfied users and the load are on different scales compared to RLFs and PPs, a simple addition of N_{us} or load to the original reward function may be counterproductive. A simple alternative that has been selected is to scale the reward by a small bias value, called the load bonus, that depends on whether the cell load was significantly reduced or not. The load bonus makes the reward appear slightly better for actions that reduce load in an overloaded cell. The revised reward $r_{x,t}$ is thus given as in equation 6.1.

$$\begin{aligned} r_{x,t} &= -\,Loadbonus * (w_1 P + w_2 F_E + w_3 F_L) \\ and \qquad Loadbonus &= \begin{cases} 0.9 & ; \quad significant\ load\ change \\ 1 & ; \quad otherwise \end{cases} \end{aligned} \tag{6.1}$$

The condition of "significant load change" is fulfilled: 1) if the concerned cell was overloaded at the time the action was taken and 2) if the reduction in load (difference between the previous and current load) is more than 20 %. The previous and current load are respectively the offered load in the cell at the time when the action is taken and that at the end of the evaluation period. In both cases, the load is evaluated as the average over the SON interval. The alternative of taking the instantaneous load value at the end of the SON interval would not necessarily reflect the effect of the MRO action in all cases, since the load may significantly vary e.g. owing to some sudden movement of one or more users.

Note should be taken that the parameter values applied here are subjective having been obtained through informed trial and error. The principles behind them are however straight forward, i.e:

1. in order to produce the result that actions which lead to load reduction have better quality, the *loadbonus* for such actions has to be less than the default *loadbonus* applied to all actions

2. when rewarding load reduction, the reward should not be so high that the intended focus of the SF is lost i.e the MRO objective of minimizing RLFs and PPs should still be paramount.

Following these principles ensures that QMRO continues to focus on optimizing MRO but with an eye to reducing load in case of an overloaded cell.

6.2.5.2 QLB Reward Function

The initial objective of QLB was to select the action ϕ that instantaneously removes overload from the serving cell, but without overloading the target cell(s). In order to account for effects on MRO, QLB now requires to also include a penalty to penalize actions that cause excessive degradation of HO performance. Thus, for each of the load scenarios Γ as defined in 4.3.2.2 (Table 4.5), the reward for an action taken in scenario Γ will be as given in equation 6.2

$$r = \begin{cases} \Delta\rho_s + 1 - HOpenalty & ; \quad \Delta\rho_n = 2 \ and \ \Gamma < 3 \\ \Delta\rho_s - HOpenalty & ; \quad \Delta\rho_n < 1 \\ \Delta\rho_s - 1 - HOpenalty & ; \quad otherwise \end{cases} \tag{6.2}$$

$\Delta\rho_s$ is the achieved reduction in serving cell offered load ρ_s, while $\Delta\rho_n$ is the change in average target cells offered load. The HO penalty (*HOpenalty*) is derived based on some estimated cost (called the *HOcost*) of taking the respective LB decision. Consider that for an action taken, the respective numbers of HO, PPs, RLFEs and RLFLs are H, P, F_E and F_L. The *HOcost* and the respective penalty for the action are given as

$$HOcost = \frac{P + 2 * FE}{H + F_E + F_L} \tag{6.3}$$

$$and \quad HOpenalty = \begin{cases} 0.0 & ; \quad HOcost < 0.2 \\ 0.5 & ; \quad 0.2 \geq HOcost < 0.5 \\ 1.0 & ; \quad 0.5 \geq HOcost < 0.9 \\ 1.5 & ; \quad 0.9 \geq HOcost < 1.4 \\ 2.0 & ; \quad otherwise \end{cases} \tag{6.4}$$

In the *HOcost*, F_Es are given a higher weight compared to PPs since F_Es are less desirable than PPs. Meanwhile the range of the *HOcost* values that are considered for the different *HOpenalty* values are selected subjectively but with the intention that actions that lead to more undesirable HO events should be penalized more. The penalty then lowers the effective reward awarded to an action that severely affects HO performance. In doing so, QLB learns not only to remove overload but to minimize the resulting effects on HO performance while removing the overload.

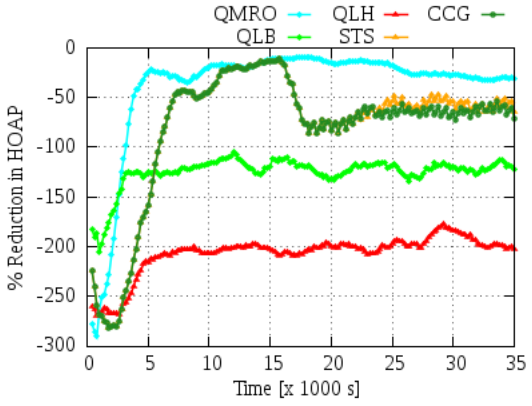
6.2.5.3 Performance Results

Considering the MRO-MLB conflicts and applying the reward function above, simulation studies were undertaken with the corresponding results shown in Fig 6.14. The figures compare the performance of CCG against that of the independent SFs as well as that of the combined but uncoordinated operation represented by QLH.

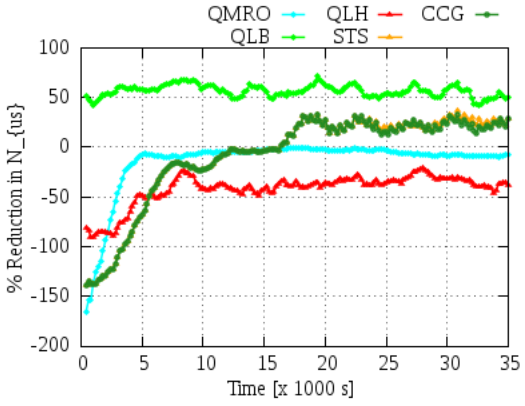
For a comparative discussion, we also include the STS results. In particular Figs. 6.14a and 6.14b evaluate the transient state in terms of the SMAs of the HOAP and N_{us}, while Fig. 6.14c evaluates the steady state performance as averages the last 20 batches.

Comparing CCG with the uncoordinated cases, we observe that CCG achieves good compromise between the two conflicting SFs. However, CCG may not always perform as good as STS. During the learning phase, the performance of CCG and STS are exactly the same since the same SFs are active and are exploring in the same way. After learning is completed however, the performance diverges since the learned functions are different.

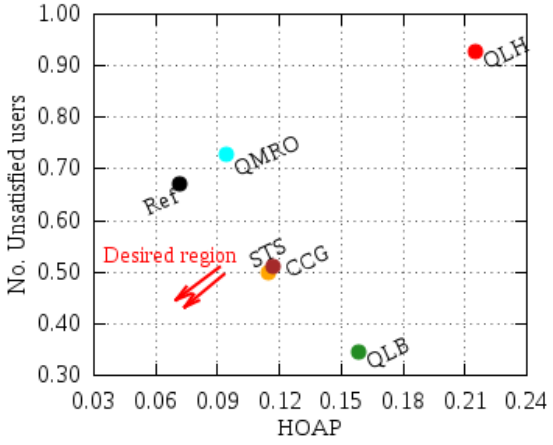
In the case of CCG, each SF is not only given a chance to learn its independent policy function, but is also required to learn the solutions that have minimal effect on the peers. This cooperative competition minimizes trigger oscillations in the sense that one SFs does not have to be triggered as a result of action of a peer since the peer will have ensured that its actions cause the least possible negative effect on all other SFs. Herein however lies also the challenge. Mixing metrics from different network perspectives implies comparing dissimilar items. For example



(a) Transient HOAP performance.



(b) Transient N_{us} performance.



(c) Steady State performance.

Figure 6.14: Performance of CCG based SON coordination.

comparing PPs to unsatisfied users whose dissatisfaction is caused by low data rates may be similar to comparing apples to mangoes.

Conversely, using a goodness scale e.g. in the range [-2,+2] as proposed in 6.2.3.2, is also not obvious. A goodness scale can be created for the cell load since the target is clear, i.e. to maintain cell load below 100 %. In that case positive values can be assigned to cell load below 100% with the negative values assigned otherwise and 0 assigned to cell load $\approx$ 100%. For HO however, the objective is to minimize the events and there is no specific target value since such a target would vary with degree of mobility. In that case the goodness scale can not be used. Consequently mixing load targets with HO events minimization in a single optimization may not always give positive results.

In general, by allowing QLB to consider the effects of its actions on MRO during learning, QLB ensures to select actions that have minimal negative impact on HO metrics. In the same way MRO also learns to select actions that will reduce the load in case such an action is taken in an overloaded cell. Further work can however be done to define the aggregation functions that would guarantee good performance.

6.3 Discussion Point: Limits and Constraints

The coordination solutions both for STD and CCG as presented here do not address all concerns. There are as such constraints that remain - partly due to the structure of the solutions but also due to the nature of the problem. The subsections below highlight these limits and constraints and where possible suggest appropriate candidate solutions.

6.3.1 Scenario Dependent Performance

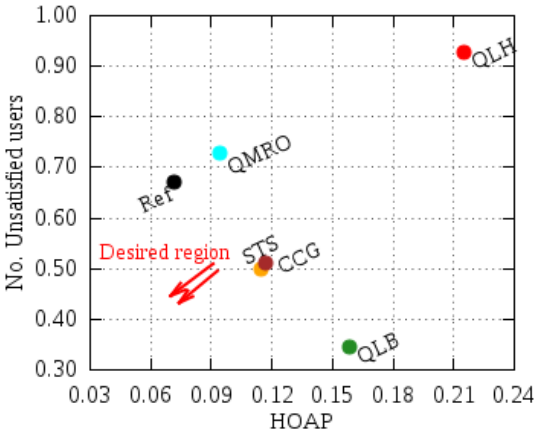
The results presented in both solutions categories considered one typical operational environment. In general multiple variations of this scenario will occur in any network with significant variations mainly expected in user numbers, velocity profiles and distribution; hotspot locations and sizes, as well as in physical or propagation characteristics like Shadowing. It is thus necessary to ensure that the proposed coordination solutions will at the least not degrade the performance in case of such scenario variations.

For the scenario and the two SFs considered in this thesis, variations that may be of concern are: 1) the hotspot size which could be smaller or larger, 2) the average velocity which could be higher or lower or 3) the number of users. The number of users may have a small impact on HO performance, where it only changes the rate of occurrence of events without altering the relative comparison of the events. User count may however have more effect on MLB since it directly affects the offered load. Meanwhile, hotspot size and velocity are expected to have major effects both

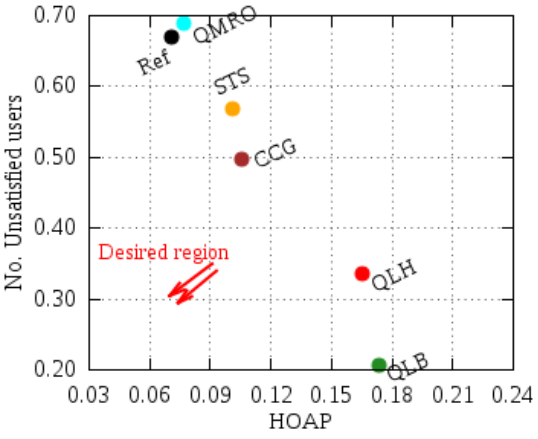
on load and HO metrics. To evaluate these dependences, we re-execute the STS and CCG simulations in two variations of the initial scenario.

The results, shown in Fig 6.15, compare the steady-state performance for the coordination solutions against the reference network, the independent SFs, as well as the combined SFs without coordination (QLH). The figure shows the performance of the initial scenario (i.e. 50 m hotspot and 60 Kmph velocity) in Fig 6.15a. It then respectively shows a smaller-hotspot scenario in Figs 6.15b (i.e. 20 m hotspot, 60 Kmph velocity) and a lower-velocity scenario (i.e. 50 m hotspot, 30Kmph velocity) in Fig 6.15c.

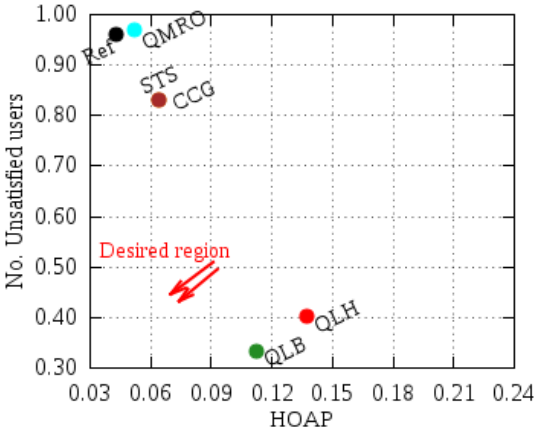
We observe that in case of a small hotspot (in this case 20 m radius in Fig. 6.15b), the degradation in the LB metric that results from combined un-coordinated operation (QLH) may be smaller owing to reduced load. This is however not the case for MRO metrics where degradation may still be significant. Coordination, using STS and CCG, still achieves good compromise by giving up some MLB gains in order to reduce MRO effects. In this case, CCG achieves a better compromise



(a) 50 m hotspot, 60 Kmph.



(b) 20 m hotspot, 60 Kmph.



(c) 50 m hotspot, 30 Kmph.

Figure 6.15: Performance of SON coordination solutions in various network hotspot and velocity scenarios.

compared to STS. The result is not the same in case of a lower velocity where both solutions achieve the exact same performance. The general observation however is that STS and CCG will always give good compromise results but their comparative performance may differ depending on scenario.

Meanwhile the inability of CCG to ever be better than STS may be an indication that STS is actually on the Pareto curve of the competition between MRO and MLB. If so, it would be that no better performance can be expected regardless of the design used for CCG. This however is not conclusive and may need to be studied further.

6.3.2 Design Challenges

The values of many of the parameters introduced in the coordination solutions have been determined heuristically through trial and error. For other parameters, compromises have been taken in order to expedite results. All such parameters require detailed studies but two are absolutely critical - the sizing of the frames and multi-frames as well as the design of the objective functions.

6.3.2.1 Sizing the Frames

All the coordination solutions require a frame period (and a related multi-frame period) within which an SF is scheduled to execute. Caution is required to ensure that SFs are allocated the appropriate lengths of time in order to obtain conclusive results. This requires that frame periods be designed from the inside outwards, first by establishing the SF requiring the longest interval so that the frame length is greater or equal to this interval. Shorter SFs would then either use part of the frame with some time within the frame remaining unused or such SFs would be cascaded one after another if their combined time can fit within the frame (e.g. cell k in multi frame n+1 in Fig. 6.12 cascades multiple SFs within the allocated frame).

6.3.2.2 Generic Aggregate Objective Functions

As previously stated, the presented implementation of CCG considers only local effects within a single cell, implying that better performance may be achieved if the effects across neighbor cells are also considered. For this, an appropriate objective function must be designed that ensures that the different effects in different cells will be combined in a way that enables the SF to minimize the conflicts. The challenge with such a function is design of the weights that appropriately define the relative importance of the different effects i.e. on the active SF, on the local cell and on neighbor cells

6.3.3 Performance Limits

Besides design challenges, the separation of actions thorough scheduling, i.e. deactivation of SFs at some space-time points, places limits on the performance of the solutions. These periods of inactivity have major effects on the instantaneous performance of the solutions and their convergence periods.

6.3.3.1 Sub-optimality during Learning

As SFs use only short portions of the space-time resources during learning, there will be long periods of suboptimal performance where SFs rely on stale configurations while waiting for their execution periods. For example consider CCG in a network with 7 clusters. MLB may be required to run every minute, but if the cluster frame period is say 5 minutes as may be dictated by say MRO, MLB may have to wait for 30 minutes before reacting, which compromises the intended benefits of dynamic optimization. This however is the cost of ensuring accuracy of measurements taken in each cell following any SON action. Incidentally this is required in both learning and rule based algorithms since they all require feedback on which to base their next actions. Learning methods however are able to handle this better since after a good amount of learning, they may cease to require the feedback.

6.3.3.2 Time to Convergence

The biggest challenge with the proposed space and time separation schemes is that solutions would take long to converge since the SFs are only allocated small portions of the space-time resources. As earlier stated however, this is the cost for ensuring that accurate policy functions are learned. A compromise would be to consider a double (or multiple) TDM multi-frame structure where very long interval SFs are considered quasi-static by the short interval SFs. In that case, the two multi-frames can be executed concurrently either as a generic multi-multi-frame structure or as a local decision made by each cell depending on the local requirements.

6.4 Summary

In this chapter, we have presented the proposed SON coordination solutions based on Space-Time Decoupling procedures and Concurrent Cooperative Games. STD solutions attempt to minimize negative cross effects among SON functions by separating the execution of the SFs in space and time. On the other hand CCGs assume an environment in which all affected SFs are able to measure the effect of each action taken and that the SFs are able to communicate these effects to each other so that the offending SF is able to learn to minimize these effects. We

have evaluated the performance of the solutions using the developed Q-learning SFs that were presented in chapter 4.

Our results have shown that separating the SFs in one dimension (i.e. either space or time) still leads to some degree of degradation in the metrics of the SFs. However, combining spatial and temporal separation ensures that SFs are able to learn the effects of their actions without interference from other actors. Each SF is able to act and measure the effect of its action in its cell without another SF (either in the same cell or in a neighbor cell) superimposing its action on top. The result is that SFs end up with more accurate Q-tables and thus achieve better compromises amidst each other.

With the success of STS, we applied CCG on top to ensure that besides measuring individual effects accurately, SFs can attempt to also consider their effects on the peer SFs. This is especially important given the inclusion of temporal separation, i.e. some SFs learning after others. We require that the SFs that learn later causes as little degradation in the metrics of the earlier SFs as possible. With simulations only considering effects among SFs in the same cell, our results when compared to STS registered improvements in some scenarios but not all. The results however demonstrated that the solution is promising. We observed that extra improvements could possibly be achieved if reports of the effects of the actions across cell boundaries are considered. We observed that the major challenge in undertaking these improvements would be determining how best the reports from the different SFs in different cells should be aggregated so as to ensure that the final policy function will always give positive results.

Besides the limitations of the CCG solution, the chapter ends with a discussion of the potential limits to the performance gains and the dependency of the absolute gains on the specific scenario in the network. It also gave a glimpse into what it would require to generalize these solutions especially CCG to an environment of multiple SFs. These are open questions that could be undertaken by future research work. The next chapter discusses other such open questions and hints at other possible approaches besides giving a high level summary of the thesis.

Chapter 7

Conclusion and Future Work

Contents

This chapter attempts to enclose all the work that has been undertaken in this thesis into a single enclosing box. This however is no mean task given the scope of the content in the different chapters, i.e. from radio models to system-like UML models and from LTE technology specifications to protocols and process flows. Consequently, the chapter focuses more on the high-level conceptual discussion that surrounds the contents and contributions of the thesis.

7.1 Contributions

The thesis has studied the problem of coordinating SFs with special focus on SFs that affect each others' metrics. The ideas have been evaluated using network/system level simulations of an LTE network. However even though LTE was considered, it is expected that the ideas presented can be easily extended to other systems, e.g. LTE-Advanced. Building on to SON as proposed by earlier research activity, the thesis set out to answer two questions: 1) that how should SFs be designed and implemented in order to ensure that their coordination is simplified and 2) how then should SFs be coordination across space (in different cells) and across time (within one or multiple cells).

To ensure an appropriate grounding for a reader to be able to follow the discussions, the thesis included a steep but complete guide to the fundamental background

material. This related to LTE, its simulation environment and the state of research on SON in LTE, on which the thesis contributions were the overlaid.

The major contributions of this thesis are in 2 areas:

1. A framework for developing SON functions as Q-Learning agents: Q-Learning (QL) was proposed and justified as an appropriate framework for developing SON functions. The framework is based on the generalization of the QL algorithm to ensure that any SF can be mapped to the algorithm. Then, two SFs were developed based on this framework and their performance evaluated using the LTE simulation environment. The achieved positive performance results further justified the proposed framework allowing the 2 SFs to be used in studying the coordination problem.

2. Multi Agent Systems approaches to coordinating SFs: Considering each instance of each SF as an agent that interacts with its environment, it was observed that the multi-SF SON environment is a multi agent system. In fact it could be considered a multi-multi-agent system in which each SF is a MAS whose instances are the agents. First, we demonstrated that conflicts do happen and we highlighted the different kinds of metric value conflicts between the two SFs. We then presented two coordination approaches and compared the resulting performance to that of the individual SFs and that of the combined but uncoordinated operation of the SFs. The first approach was based on multi agent scheduling with the SFs and their instances as agents, while the second evaluated concurrent cooperative games among the SF instances.

7.2 Future Work

A number of extensions can be undertaken from this work. We highlight here a few that we consider to be absolutely critical.

7.2.1 Improvements to the Q-learning Solutions

The QMRO solution is able to determine the right parameter settings for a cell in which all users have velocities within the same range. In other words, the solution works if the mobility profiles of all users in a cell are comparable e.g. if the cell is within a city district, or if the cell covers a highway where all users are either always moving fast or if they slow down, their velocities are still comparable to each other. The solution however breaks down if a single cell has multiple velocity groups at the same time. Such can be the case when: 1) on a highway one side moves fast but the other side has slowed down owing to some traffic condition; or 2) the highway moves through an office district so that the cell concurrently has fast moving users (e.g at $\geq 60Kmph$) and slow moving users (at say $\leq 5Kmph$). This failure may to some extent explain the inability of QMRO to achieve exactly the same performance as the Reference network in simulation studies in Chapters

5 and 6. In the simulated scenario some cells in the network have static hot-spot users concurrent with other fast moving users. The configuration learned in such cells are thus misleading to the other cells where only one velocity group exists. It is therefore necessary to devise a solution for such special scenarios. Such a solution may involve redefining the states in a way that we can capture the fact that two velocity groups exist. This would then require that the resulting HO events are appropriately accorded to the respective velocity groups.

7.2.2 Q-learning for Combined Optimization

In this thesis, MRO and MLB have been studied as separate functions owing to their different objectives. Since both SFs however focus on adjusting the logical handover boundary, they would be good candidates to be studied as one. We expect that a QL solution can be designed to concurrently optimize MRO and minimize the cell load in a dynamic format. The major challenge to such a solution may lie in the different optimization intervals expected for the two SFs. A study can however be executed that determines a compromise interval that would ensure that both are managed at once. Our expectation is that success of such a combined optimization would imply reduced need to coordination. This however does not negate the results achieved in the thesis since the ideas are generic and can be used for other conflicts including the conflict between the combined MRO-MLB solution and other SFs like self-organized admission control or interference management.

7.2.3 Autonomic Optimization Objectives

As discussed under the performance limits for the coordination solutions, methods of combing multiple objectives still remain an open problem. The success of CCG between any SFs will depend on how the measured effects of a single action on multiple metrics can be combined. Expert knowledge will be required at the beginning to design such objective functions but automated solutions should be sought to find ways in which high level definitions of the objectives can be decomposed into the objective functions to used by the SFs in evaluating the actions.

7.2.4 Bayesian Q-learning for SON Coordination

An area of direct improvement to the coordination of QL solutions is the combination of Bayesian game theory and QL to ensure that SFs learn not only over the state space of a single SON problem but over the state space of the combined set of SON problems. Such an approach would combine basic game theory, Bayesian theory and Q-learning in a single solution.

Game theory can be defined as the study of rational decision-making in situations of conflict and/or cooperation". It generally deals with games of complete information i.e. players have common knowledge of the structure of the game and the

payoffs. In such games, each information set consists of a single decision node, i.e. the players in the game learn strategies of how to win, where a pure strategy s_i for player i consists of a choice for each of player i's information sets.

In case of imperfect information where the players do not know some of the parameters of the game, Bayesian theory is applied, thus such games are called Bayesian Games [92]. In this case, the unknown information for each player is bunched up into a 'type' X for the player for which a probability distribution P(X) is known. The type X represents the states of nature for each player and the strategies can then be determined using Bayesian theory by maximizing utilities over all types.

There are two instances however where Bayesian games also break down:

1. the probability distribution P(X) over the possible states of nature X is unknown, even though the consequence of an action a is still a deterministic function C(a, X) of the action a and the state X;
2. both P(X) and C(a, X) are not known

In such a case a learning solution can be employed to act in the environment and learn the distribution over the type and/or the utility from which it derives the strategies. Such an approach would also be applicable for the learning towards the coordination of SON solutions.

In principle, SON coordination would be equivalent to learning over a number of Bayesian games. Each SF is a player competing with the peers. The SF/player has a set of internal/known states for which strategies need to be learned. Each of the known states however represents an extensive state of nature that includes all the unknown states of the peer SFs. The candidate SF can thus learn the distribution over the states of nature concurrently with the appropriate strategies for those states. The solution would be characterized by:

- A list of players $i = 1 \cdots n.$ which are the QL agents
- A finite set of types Θ_i for each player i, which are equivalent to the states as used in QL. These capture all information about the state of nature in which the player finds itself including all information about other players that he may not know.
- A finite set of actions A_i for each player i.
- A utility function u_i for each player i, where $u_i{:}A \,\mathrm{x}\, \Theta \rightarrow R$, i.e. for a player i, u_i maps a combination of the player's actions and types to its payoff
- A distribution on types, $P(\theta), \forall \theta \in \Theta$ which is to be learned.

Note should be taken however that it may not be easy to derive a priori all the possible types of nature or that it may not be desirable to do so owing to the size of the state space. In that case, it would require that the agents learn long enough to ensure that each decision taken maximizes over all the known and unknown states of nature. This is a promising approach that can be evaluated for a set of coupled SON functions.

7.2.5 Work-flow Decomposition Versus Layering in SON

SON functions have been proposed and designed following the traditional 'work flow' model of network operations and management. They are grouped into four categories i.e. planning, configuration, optimization and healing. The challenge is that this grouping breaks up related functionality into different SFs. Correspondingly, parameters are then required to be adjusted by different SFs for different and in many cases competing objectives. We propose that the structuring of SON should in general be revisited to minimize the duplication and the need for coordination. One strategy is to break up SON into the natural layering of network functions in three areas of reducing priority: 1) optimization of physical network resources, 2) optimization of logical network resources and 3) the optimization of user quality of experience.

The allocation of priorities is based on the expectation that lower priority functions can never be achieved without the higher priority functions. Area 1 would focus on the availability of system resources and would include problems such as CCO, Interference avoidance and minimization of energy consumption, all optimized as one SON functional block. Area 2 focuses on ensuring availability and robustness of user connectivity and should be done after the first. This is because there is no need to undertake logical optimization if and when physical resources are unavailable. This level would cover SFs as MRO, admission control and Random Access Channel (RACH) optimization. The third level then focuses on user quality optimizing the user throughput, fairness of scheduling, and minimization of latency and jitter.

In the same sense we propose that the optimization goals should not be designed based on low level KPIs without clear higher-level abstract objectives. Generic higher-level objectives to which the lower level KPIs contribute should be set for each priority area. In that case optimization is done based on the higher level objectives which requires that concurrent adjustments are done on all parameters that influence lower level KPIs needed to achieve the desired higher level objective.

Bibliography

[1] L. Schmelz, J. V. D. Berg, R. Litjens, A. M. Amirijoo, O. Linnell, C. Blondia, T. Kürner, N. Scully, and J. Oszmianski, "Self-configuration, -optimisation and -healing in wireless networks," in *WWRF*, December 2008, pp. 4477–4479. [Online]. Available: http://link.aip.org/link/?RSI/72/4477/1

[2] N. G. M. Networks, "Use cases related to self organising network, overall description," May 2007. [Online]. Available: http://www.ngmn.org/technology.html

[3] Monotas, "Mobile network optimisation through advanced simulation." [Online]. Available: http://www.macltd.com/monotas/index.php

[4] Gandalf, "Monitoring and self-tuning of rrm parameters in a multi-system network." [Online]. Available: http://www.celtic-initiative.org/Projects/Celtic-projects/Call2/GANDALF/gandalf-default.asp

[5] SOCRATES, "Deliverable d5.9: Final report on self-organisation and its implications in wireless access networks," EU STREP SOCRATES (INFSO-ICT-216284), Tech. Rep., December 2010.

[6] 3GPP, "Evolved Universal Terrestrial Radio Access Network (E-UTRAN); Self-configuring and Self-Optimizing Network (SON) use cases and solutions," 3GPP, Technical Report 36.902 version 9.3.1 Release 9, May 2011. [Online]. Available: http://www.3gpp.org

[7] ——, "Requirements for evolved utra (e-utra) and evolved utran (e-utran)," TR 25.913 V7.3.0, Tech. Rep., 2009. [Online]. Available: http://www.3gpp.org

[8] W. M. Jim Zyren, "Overview of the 3gpp long term evolution physical layer," *Freescale semiconductor*, 2001. [Online]. Available: http://link.aip.org/link/?RSI/72/4477/1

[9] Motorola, "Long term evolution (lte): Overview of lte air-interface technical white paper," Motorola, Tech. Rep., 2001. [Online]. Available: http://link.aip.org/link/?RSI/72/4477/1

[10] SOCRATES, "Self-optimisation and self-configuration in wireless networks." [Online]. Available: http://www.fp7-socrates.org/

[11] SEMAFOUR, "Self-management system for heterogeneous radio access networks." [Online]. Available: http://fp7-semafour.eu/

[12] N. G. M. Networks, "NGMN Recommendation on SON and OandM Requirements," 2008. [Online]. Available: http://www.ngmn.org/technology.html

[13] SOCRATES, "Deliverable d2.1: Use cases for self-organising networks, eu strep socrates," EU STREP SOCRATES (INFSO-ICT-216284), Tech. Rep., March 2008. [Online]. Available: http://www.fp7-socrates.org/?q=node/10

[14] ——, "Deliverable d2.2: Requirements for self-organising networks, eu strep socrates," EU STREP SOCRATES (INFSO-ICT-216284), Tech. Rep., June 2008. [Online]. Available: http://www.fp7-socrates.org/?q=node/10

[15] ——, "Deliverable d2.3: Assessment criteria for self-organising networks, eu strep socrates," EU STREP SOCRATES (INFSO-ICT-216284), Tech. Rep., June 2008. [Online]. Available: http://www.fp7-socrates.org/?q=node/10

[16] ——, "Deliverable d2.5: Review of use cases and framework," EU STREP SOCRATES (INFSO-ICT-216284), Tech. Rep., March 2009. [Online]. Available: http://www.fp7-socrates.org/?q=node/10

[17] ——, "Deliverable d2.6: Review of use cases and framework ii," EU STREP SOCRATES (INFSO-ICT-216284), Tech. Rep., December 2009. [Online]. Available: http://www.fp7-socrates.org/?q=node/10

[18] T. Jansen, I. Balany, I. Moermanz, and T. Kürner, "Handover parameter optimization in lte self-organizing networks," in *COST2100 TD(10)10068, Athens, Greece,*, 2010, p. February.

[19] G. Hui and P. Legg, "Soft metric assisted mobility robustness optimization in lte networks," in *International Symposium on Wireless Communication Systems (ISWCS 2012), Paris France*, August 2012, pp. 1–5.

[20] J. Alonso-Rubio, "Self-optimization for handover oscillation control in lte," in *IEEE/IFIP Network Operations and Management Symposium, Osaka, Japan,*, 2010.

[21] I. Balan, T. Jansen, B. Sas, I. Moerman, and T. Kürner, "Enhanced weighted performance based handover optimization in LTE," in *Proceedings of FNMS, Warsaw, Poland,*, 2011.

[22] B. Sas, K. Spaey, I. Baran, K. Zetterberg, and R. Litjens, "Self-optimisation of admission control and handover," in *Proceedings of IEEE 73rd Vehicular Technology Conference (VTC-Spring), Budapest, Hungary,*, May 2011.

[23] K. Spaey, B. Sas, and C. Blondia, "Self-optimising call admission control for lte downlink," in *Joint COST 2100 / SOCRATES workshop*, February 2010.

[24] A. Lobinger, S. Stefanski, T. Jansen, and I. Balan, "Load balancing in downlink lte self-optimizing networks," in *VTC 2010-Spring, Taipei, Taiwan*, May 2010.

[25] L. A. S. S., J. T., and B. I., "Coordinating handover parameter optimization and load balancing in lte self-optimizing networks," in *IEEE 73rd Vehicular Technology Conference (VTC-Spring), Budapest, Hungary*, 2011.

[26] M. Amirijoo, P. Frenger, F. Gunnarsson, J. Moe, and K. Zetterberg, "Towards RACH Self Tuning in LTE," in *IEEE Vehicular Technology Conference, Spring*, 2009.

[27] I. Siomina, "P-cpich power and antenna tilt optimization in umts net-works," in *In Telecommunications, 2005. advanced industrial conference on telecommunications/service assurance with partial and intermittent resources conference /e-learning on telecommunications workshop, aict/sapir/elete 2005. proceedings*, july 2005, pp. 268 – 273.

[28] S. Mwanje, N. Zia, and A. Mitschele-Thiel, "Self organised handover parameter configuration for lte," in *International Symposium on Wireless Communication Systems (ISWCS 2012), Paris France*, August 2012, pp. 26–30.

[29] I. Siomina, P. Varbrand, and D. Yuan, "Automated optimization of service coverage and base station antenna con
guration in umts net-works," in *Wireless Communications, IEEE,*, 2006, pp. 16 –25.

[30] U. Turke and M. Koonert, "Advanced site configuration techniques for automatic umts radio network design," in *In Vehicular Technology Conference, VTC-Spring. 2005 IEEE 61st, Vol. 3*, june 2005, pp. 1960 – 1964.

[31] I. Siomina and D. Yuan, "Enhancing hsdpa performance via automated and large-scale optimization of radio base station antenna configuration," in *In Vehicular Technology Conference, 2008. VTC Spring 2008. IEEE*, may 2008, pp. 2061 – 2065.

[32] A. Temesvary, "Self-configuration of antenna tilt and power for plug and play deployed cellular networks," in *In Wireless Communications and Networking Conference, 2009. WCNC 2009. IEEE*, april 2009, pp. 1–6.

[33] S. Allen, S. Hurley, and R. Whitaker, "Automated decision technology for network design in cellular communication systems," in *In System Sciences, 2002. HICSS. Proceedings of the 35th Annual Hawaii International Conference on*, jan 2002, p. 8.

[34] J. M. H.R. Anderson, "Optimizing microcell base station locations using simulated annealing techniques," in *Vehicular Technology Conference, 1988, IEEE 38th In proceeding of:Vehicular Technology Conference, 1994 IEEE 44th*, 07 1994.

[35] V. Capdevielle, A. F. A., and A. Fakhreddine, "Self-optimization of handover parameters in lte networks," in *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt), 2013 11th International Symposium on*, May 2013, pp. 133 – 139.

[36] M. Garcia-Lozano, S. Ruiz, , and J. Olmos, "Umts optimum cell load balancing for inhomogeneous traffic patterns," in *In Vehicular Technology Conference, 2004. VTC2004-Fall. 2004 IEEE 60th, Vol. 2*, sept 2004, pp. 909 – 913.

[37] I. Siomina, P. Varbrand, and D. Yuan, "An effective optimization algorithm for con
guring radio base station antennas in umts networks," in *In Vehicular Technology Conference, 2006. VTC-2006 Fall. 2006 IEEE 64th*, sept 2006, pp. 1–5.

[38] M. N. ul Islam, R. Abou-Jaoude, C. Hartmann, , and A. Mitschele-Thiel, "Self-optimization of antenna tilt and pilot power for dedicated channels," in *In Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt), 2010 Proceedings of the 8th International Symposium on*, june 2010, pp. 196 – 203.

[39] E. Amaldi, A. Capone, , and F. Malucelli, "Radio planning and coverage optimization of 3G cellular networks," in *Wireless Networks*, 2008, pp. 435–447.

[40] F. Gordejuela-Sanchez and J. Zhang, "Lte access network planning and optimization: A service-oriented and technology-specific perspective," in *Global Telecommunications Conference, 2009. GLOBECOM 2009. IEEE*, Dec 2009, pp. 1 – 5.

[41] L. Du, J. Bigham, L. Cuthbert, C. Parini, , and P. Nahi, "Using dynamic sector antenna tilting control for load balancing in cellular mo-bile communications," in *In International Conference on Telecommunications, ICT2002, Beijing, Citeseer*, 2002, pp. 344– 348.

[42] S. Jamaa, Z. Altman, J. Picard, , and B. Fourestie, "Combined coverage and capacity optimisation for umts networks," in *In Telecommunica-tions Network Strategy and Planning Symposium. NETWORKS 2004, 11th Internationa*, june 2004, pp. 175 – 178.

[43] D. Z. C. Xu Yang/Yapeng Wang, "Resource allocation in lte ofdma systems using genetic algorithm and semi-smart antennas," in *in proceedings of Wireless Communications and Networking Conference (WCNC), 2010 IEEE*, April 2010, pp. 1 – 6.

[44] M. E. Aydin, R. Kwan, W. Ding, and J. Wu, "A genetic algorithm approach for multiuser scheduling on the lte downlink," in *Proceedings of the World Congress on Engineering 2012, Vol II*, July 2012.

[45] H. Sun, N. Li, Y. Chen, J. Dong, N. Liu, Y. Han, and W. Liu, "A method of pci planning in lte based on genetic algorithm," in *Progress in Electromagnetics Research Symposium Proceedings*, August 2012, pp. 1575 – 1578.

[46] K. Lin, "Improving energy efficiency of lte networks by applying genetic algorithm (ga)," in *Dependable, Autonomic and Secure Computing (DASC), 2011 IEEE Ninth International Conference on*, Dec 2011, pp. 593 – 59.

[47] H. Elkamchouchi, H. Elragal, , and M. Makar, "Cellular radio network planning using particle swarm optimization," in *In Radio Science Conference, 2007. NRSC 2007. National*, 2007, pp. 1–8.

[48] Z. Altman, S. Sallem, R. Nasri, B. Sayrac, and M. Clerc, "Particle Swarm Optimization for Mobility Load Balancing SON in LTE Networks," *arXiv preprint arXiv:1401.6621*, 2014. [Online]. Available: http://arxiv.org/abs/1401.6621

[49] A. Gerdenitsch, S. Jakl, Y. Chong, , and M. Toeltsch, "A rule-based algorithm for common pilot channel and antenna tilt optimization in umts fdd networks," in *ETRI journal*, 2004, pp. 437–442.

[50] J. Wu, J. Bigham, P. Jiang, and J. Neophytou, "Tilting and beam-shaping for traffc load balancing in wcdma network," in *In Wireless Technology, 2006. The 9th European Conference on*, sept 2006, pp. 63–66.

[51] M. Pettersen, L. Braten, , and A. Spilling, "Automatic antenna tilt control for capacity enhancement in umts fdd," in *In Vehicular Technology Conference, . VTC2004-Fall. 2004 IEEE 60th, Vol. 1, . 2004*, sept 2004, pp. 280 –284.

[52] N. Zia and A. Mitschele-Thiel, "Performance evaluation of intra-lte mobility load balancing for hotspot clustering," in *30th Meeting of Wireless World Research Forum (WWRF30)*, April 2013.

[53] ——, "Self-organized neighborhood mobility load balancing for lte networks," in *6th IFIP/IEEE Wireless Days Conference (WD'13)*, November 2013.

[54] G. Calcev and M. Dillon, "Antenna tilt control in cdma networks," in *In Proceedings of the 2nd annual international workshop on Wireless internet*, 2006, p. 25.

[55] H. Eckhardt, S. Klein, and M. Gruber, "Vertical antenna tilt opti-mization for lte base stations," in *In Vehicular Technology Conference (VTC Spring), 2011 IEEE 73rd*, may 2011, pp. 1–5.

[56] S. Deb and P. Monogioudis, "Learning based uplink interference management in 4g lte cellular systems." [Online]. Available: http://arxiv.org/abs/1309.2543

[57] N. Sinclair, D. Harle, I. A. Glover, J. Irvine, and R. C. Atkinson, "An advanced som algorithm applied to handover management within lte," *IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, VOL. 62, NO. 5*, pp. 1883–1894, JUNE 2013.

[58] J. Li, J. Zeng, X. Su, W. Luo, and J. Wang, "Self-optimization of coverage and capacity in lte networks based on central control and decentralized fuzzy q-learning," *International Journal of Distributed Sensor Networks, Volume 2012*, 2012.

[59] T. Jansen, M. Amirijoo, U. Türke, L. Jorguseski, K. Zetterberg, R. Nascimento, L. C. Schmelz, J. Turk, and I. Balan, "Embedding multiple self-organisation functionalities in future radio access networks," in *69th Vehicular Technology Conference, VTC-Spring 2009, Barcelona, Spain*, 2009.

[60] T. Bandh, H. Sanneck, and R. Romeikat, "An experimental system for SON coordination," in *IWSON IEEE 73rd Vehicular Technology Conference, Spring*, 2011.

[61] T. Bandh and L. C. Schmelz, "Impact-time Concept for SON-Function Coordination," in *International Symposium on Wireless Communication Systems (ISWCS 2012), Paris France*, August 2012, pp. 16–20.

[62] K. Tsagkaris, N. Koutsouris, P. Demestichas, R. Combes, and Z. Altman, "SON Coordination in a Unified Management Framework," in *77th Vehicular Technology Conference, VTC-Spring, Dresden, Germany*, 2013.

[63] L. Panait and S. Luke, "Cooperative multi-agent learning: The state of the art," in *Autonomous Agents and Multi-Agent Systems*, 2005, pp. 387–434.

[64] L. Busoniu, R. Babuska, and B. D. Schutter, "A comprehensive survey of multi agent reinforcement learning," in *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 2008, pp. 156–172.

[65] T. Jansen and R. Wiegand, "Exploring the explorative advantage of the cooperative coevolutionary (1+ 1) ea," in *in Genetic and Evolutionary Computation IGECCO*, 2003, pp. 197–197.

[66] Institute of Communication Networks and Computer Engineering (IKR), "Ikr simulation and emulation library," Universität Stuttgart, Tech. Rep., Sept 2013, v.4.0. [Online]. Available: http://www.ikr.uni-stuttgart.de/en/Content/IKRSimLib/

[67] 3GPP, "Evolved Universal Terrestrial Radio Access Network (E-UTRAN); Further advancements for E-UTRA physical layer aspects (Release 9)," 3GPP, Technical Report 36.814 version 9.0.0, March 2010.

[68] T. Rappaport, *Wireless communications: principles and practice*. Pren-tice Hall, 2002.

[69] E. SMG, "Universal mobile telecommunications system (umts); selection procedures for the choice of radio transmission technologies of the umts," *ETSI Document TR*, vol. 101, p. 112, 1997.

[70] M. Gudmundson, "Correlation model for shadow fading in mobile radio systems," *Electronics letters*, vol. 27, no. 23, pp. 2145–2146, 1991.

[71] 3GPP, "Physical layer aspect for evolved universal terrestrial radio access (utran)," 3GPP TR 25.814, Tech. Rep., October 2006. [Online]. Available: http://www.3gpp.org

[72] ——, "Selection procedures for the choice of radio transmission technologies of umts (release 1999)," 3GPP TR 30.03, Tech. Rep., April 1998.

[73] ——, "Radio Access Network;Evolved Universal Terrestrial Radio Access (E-UTRA);Radio Frequency (RF) system scenarios (Release 8)," 3GPP TR 36.942, Tech. Rep., October 2010.

[74] A. M. Rao, A. Weber, S. Gollamudi, , and R. Soni, "Lte and hspa+: Revolutionary and evolutionary solutions for global mobile broadband," *Bell Labs Technical Journal*, vol. 13, no. 4, p. 7–34, 2009.

[75] T. Camp, J. Boleng, and V. Davies, "A survey of mobility models for ad hoc network research," *Wireless Communications and Mobile Computing: Special issue on Mobile AdHoc Networking: Research, Trends and Applications*, vol. 2, no. 5, pp. 483–502, 2002.

[76] 3GPP, "E-UTRA Radio Resource Control (RRC) Protocol specification (Release 8)," 3GPP, Technical Report 36.331 version 8.16.0, December 2011.

[77] K. D. et al., "Handover within 3gpp lte: Design principles and performance," in *in Proceedings of IEEE VT-cell 2009-Fall*, September 2009.

[78] R. Sutton, *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[79] C. Watkins, "Learning from delayed rewards," PhD Thesis, Kings College, Cambridge, England, May 1989.

[80] C. J. Watkins and P. Dayan, "Q,-learning," *MachineLearning*, vol. 8, pp. 279–292, 1992.

[81] E. Even-Dar, S. Mannor, and Y. Mansour, "Pac bounds for multi-armed bandit and markov decision processes," in *in Proceedings of 15th Conference on Computational Learning Theory*, July 2002, pp. 258–259.

[82] R. Razavi, S. Klein, and H. Claussen, "Self-optimization of capacity and coverage in lte networks using a fuzzy reinforcement learning approach," in *Personal Indoor and Mobile Radio Communications (PIMRC), 2010 IEEE 21st International Symposium on*. IEEE, 2010, pp. 1865–1870.

[83] M. N. ul Islam and A. Mitschele-Thiel, "Reinforcement learning strategies for self-organized coverage and capacity optimization," in *Wireless Communications and Networking Conference (WCNC), 2012 IEEE*. IEEE, 2012, pp. 2818–2823.

[84] M. Dirani and Z. Altman, "A cooperative reinforcement learning approach for inter-cell interference coordination in ofdma cellular networks," in *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt), 2010 Proceedings of the 8th International Symposium on*. IEEE, 2010, pp. 170–176.

[85] M. Bennis and D. Niyato, "A q-learning based approach to interference avoidance in self-organized femtocell networks," in *GLOBECOM Workshops (GC Wkshps), 2010 IEEE*. IEEE, 2010, pp. 706–710.

[86] P. Munoz, R. Barco, I. de la Bandera, M. Toril, and S. Luna-Rami?rez, "Optimization of a fuzzy logic controller for handover-based load balancing," in *Vehicular technology conference (VTC Spring), 2011 IEEE 73rd.* IEEE, 2011, pp. 1–5.

[87] P. Munoz, R. Barco, J. M. Ruiz-Aviles, I. de la Bandera, and A. Aguilar, "Fuzzy rule-based reinforcement learning for load balancing techniques in enterprise lte femtocells," *Vehicular Technology, IEEE Transactions on*, vol. 62, no. 5, pp. 1962–1973, 2013.

[88] T. Jansen, I. Balan, S. Stefanski, I. Moerman, and T. Kurner, "Weighted performance based handover parameter optimization in lte," in *Vehicular Technology Conference (VTC Spring), 2011 IEEE 73rd.* IEEE, 2011, pp. 1–5.

[89] I. Bălan, B. Sas, T. Jansen, I. Moerman, K. Spaey, and P. Demeester, "An enhanced weighted performance-based handover parameter optimization algorithm for lte networks," *EURASIP Journal on Wireless Communications and Networking*, p. 98, 2011.

[90] J. Lorca and A. Sierra, "A Simple Speed Estimation Algorithm for Mobility-Aware SON RRM Strategies in LTE ," in *6th IFIP/IEEE Wireless Days Conference (WD'13)*. Valencia, Spain: Wireless Days 2013, 2013.

[91] H. Wang, L. Ding, P. Wu, Z. Pan, N. Liu, and X. You, "Dynamic load balancing and throughput optimization in 3gpp lte networks," in *Proceedings of the 6th International Wireless Communications and Mobile Computing Conference*. ACM, 2010, pp. 939–943.

[92] D. Fudenberg, *The theory of learning in games*, 1998, vol. 2.

Erklärung

Ich versichere, dass ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus anderen Quellen direkt oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet.

Weitere Personen waren an der inhaltlich-materiellen Erstellung der vorliegen- den Arbeit nicht beteiligt. Insbesondere habe ich hierfür nicht die entgeltliche Hilfe von Vermittlungs- bzw. Beratungsdiensten (Promotionsberater oder anderer Personen) in Anspruch genommen. Niemand hat von mir unmittelbar oder mittelbar geldwerte Leistungen für Arbeiten erhalten, die im Zusammenhang mit dem Inhalte der vorgelegten Dissertation stehen.

Die Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder ähnlicher Form einer Prüfungsbehörde vorgelegt. Ich bin darauf hingewiesen worden, dass die Unrichtigkeit der vorstehenden Erklärung als Täuschungsversuch bewertet wird und gemäß §7 Abs. 10 der Promotionsordnung den Abbruch des Promotionsverfahrens zu Folge hat.

Ilmenau, den 08. September 2014 Stephen Ssekiranda Mwanje