Simulation of Coordinated Multipoint Using Discrete Event Systems Specification

By

S. Mohammad Etemad, B.Eng.

A thesis submitted to the Faculty of Graduate Studies and Research

in partial fulfillment of the requirements for the degree of

Master of Applied Science in Electrical and Computer Engineering

Ottawa-Carleton Institute for Electrical and Computer Engineering (OCIECE)

Department of Systems and Computer Engineering

Carleton University

Ottawa, Ontario, Canada, K1S 5B6

January 2017

Abstract

With the rise in the number of subscribers and to enhance the user experience, Long Term Evolution Advanced (LTE-A) has introduced Coordinated Multipoint (CoMP), a technology which aims to improve the user experience on the cell edge. Although CoMP improves the user experience on the cell edge, it imposes lots of overhead on the network which may result in the degradation of Quality of Service (QoS).

To be able to test technologies like CoMP before deploying them on the live network, Modeling and Simulation (M&S) software are required. Through M&S, researchers can test their proposed ideas with minimal costs. Discrete Event System Specification (DEVS) is a formal platform intended for M&S of discrete time systems.

In this thesis, we present a dynamic DEVS-based model for simulating CoMP. The model consists of different components that mimic the behavior of entities in LTE-A networks. The model can be used to model different scenarios in both homogenous and Heterogeneous Networks (HetNets).

Moreover, we present a new architecture for CoMP developed in collaboration with fellow team members at Carleton University and Ericsson Canada Inc. This new approach reduces the number of control messages in the network, hence, increases download and upload rates.

To compare the performance of the newly proposed CoMP architecture with the conventional CoMP architectures, a DEVS-based simulator was developed. The results show that the newly proposed architecture significantly reduces the number of control messages required in the network, hence, it improves the data rates.

To my wife, Faezeh, for her love and support

and

to my parents and brothers

for their inspiration and guidance

Acknowledgement

First and foremost, I would like to thank my supervisor, Dr. Gabriel Wainer, whose guidance, support, and encouragement made all this possible.

I would like to also thank my colleague, Baha Uddin Kazi, whom this research was done in collaboration with. His support and coordination during the past few years has been exceptional. I also thank the members of the Advanced Real-Time Simulation Laboratory (ARS-Lab) for their support and feedback.

Last but not least, I wish to deeply thank my wife, Faezeh, for hear unconditional love and support throughout these years.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

1G	First Generation
2.5G	Generation Between the Second and Third Generations
2G	Second Generation
3G	Third Generation
3GPP	Third Generation Partnership Project
4G	Fourth Generation
ARS-Lab	Advanced Real-Time Simulation Laboratory
BBU	Baseband Unit
BS	Base Station
CCS	Central Coordination Station
CPS	Cyber-Physical Systems
CQI	Channel Quality Indicator
CS/CB	Coordinated Scheduling and Beamforming
CSI	Channel State Information
CU	Central Unit
CoMP	Coordinated Multi Point
DCEC	Direct CSI-feedback to Elected Coordination-station
DCS	Dynamic Cell Selection
DEVS	Discrete Event System Specification
EPC	Evolved Packet Core
FDMA	Frequency Division Multiple Access

GUI	Graphical User Interface
HSPA	High Speed Packet Access
HetNet	Heterogeneous Network
ICIC	Inter-Cell Interference Coordination
ITU-R	International Telecommunication Union Radio Communication Sector
JT	Joint Transmission
LTE	Long Term Evolution
LTE-A	Long Term Evolution Advanced
M&S	Modeling and Simulation
MAC	Medium Access Control
MCC	Multipoint Cooperative Communication
MCS	Modulation and Coding Scheme
MECSYCO	Multi-agent Environment for Complex Systems Co-simulation
MIMO	Multiple-Input and Multiple-Output
NED	Network Description
NS-2	Network Simulator-2
NS-3	Network Simulator-3
OTCL	Object-oriented Tool Command Language
PCI	Physical Cell Identity
PMI	Processing Matrix Indicator
PTI	Precoding Type Indicator

QoS	Quality of Service
RAN	Radio Access Network
RI	Rank Indicator
RN	Relay Nodes
RRE	Remote Radio Equipment
RRH	Remote Radio Head
RRM	Radio Resource Management
SMS	Short Message Service
TP	Transmission Point
TPS	Transmission Point Selection
TTI	Transmission Time Interval
UE	User Equipment
WSN	Wireless Sensor Networks
WiMAX	Worldwide Interoperability for Microwave Access

Chapter 1

Introduction

The 1st Generation (1G) of mobile networks was first introduced in the 1970s. AT&T was the first company to propose a mobile phone system which involved dividing areas into cells. 1G operated using analog circuit switching with Frequency Division Multiple Access (FDMA) in the 800-900 MHz frequency. They were intended to be used for voice calls only. It was not until 1979 when Mac Donald proposed a cellular system with hexagonal cells, covering areas with a bit of overlap on the edges. This concept was a great step forward in the history of telecommunications in terms of power consumption, coverage, efficiency, and interference [1].

The 2nd Generation (2G) followed in the early 1990s and purely used digital technology. As opposed to 1G where analog signals were used, the use of digital signals in 2G had certain advantages. Because of their digital nature, digital signals could be encrypted to improve the security of the network. Furthermore, error correction techniques could be applied to the received messages to correct possible errors that have occurred

during transmission. This resulted in a more reliable and more robust connection. Also, by using digital systems, the need for a dedicated channel per line was eliminated and channels could be shared while they were not in use [2, 3]. 2G also enabled users to use services such as caller ID and Short Message Service (SMS). As demand began to rise, network providers started adding services such as faxing, paging, text messages, and voicemail. In the late 1990s an intermediary phase (2.5G) was introduced which added features such as allowing users to send and receive graphic-rich data as packets.

After 2.5G, he 3rd Generation (3G), known as the revolutionary generation of cellular networks, was introduced. 3G enabled users to work with audio, graphic, and video applications. One of the main objectives of 3G was to standardize a network protocol to be used all around the globe. Before this, the U.S., Europe, and other regions were using different standards. Supporting both packet and circuit switched data transmission, 3G could achieve higher data rates compared to its predecessors.

The Long Term Evolution (LTE) standard which followed 3G and is currently in use, was first proposed by NTT DoCoMo of Japan with its first commercial use launched in Sweden and Norway in late 2009 and U.S. and Japan in 2010. Although LTE is commonly referred to as the 4th Generation (4G), it failed to meet the International Telecommunication Union Radio Communication Sector (ITU-R) requirements of 4G. It was only after marketing pressures and Worldwide Interoperability for Microwave Access (WiMAX), Evolved High Speed Packet Access (HSPA), and LTE advancements that the ITU-R decided to label the technology as 4G [4].

Long Term Evolution Advanced (LTE-A), which is also labeled 4G, is an advancement to the LTE standard proposed by Third Generation Partnership Project (3GPP). The target

of LTE-A is to reach and surpass the requirements outlined by ITU. LTE-A is compatible with LTE equipment and uses the same frequency bands. One of the main advances made in LTE-A is the optimized use of small powered cell such as picocells and femtocells. This brings the network closer to the user which results in improved capacity and coverage. Coordinated Multi Point (CoMP) has been introduced in this technology, which enables the access points to serve a user jointly [5].

With the rapid development of mobile devices and mobile internet, the number of mobile broadband users, the demand of data rates, and the total volume of data traffic are increasing very fast [6, 7]. According to Ericsson, the number of mobile broadband subscriptions is growing globally by around 25% each year, and it is expected to reach 7.7 billion by 2021 [7]. The growth rate of mobile data traffic between the 1st quarter of 2015 and the 1st quarter of 2016 was about 60 percent. The network's data traffic is expected to reach 351 Exabytes by 2025 [8, 9]. Thus, there are two main challenges faced by service providers in cellular networks: providing services to the massive number of users and coping with the increasing demand of mobile data traffic by each user.

To be able to meet this increasing demand and deal with the abovementioned challenges, LTE-A uses frequency reuse methods. Although these techniques solve the issue of high demand, they result in signal interference on the cell's edge where the signals can be received from more than one transmission point. This issue, alongside the low received signal strength, results in a poor Quality of Service (QoS) for the cell edge users. The optimal solution for frequency reuse is shown in Figure 1. As it can be seen, a minimum of three different frequency bands are required to avoid adjacent cells from using the same frequency range.

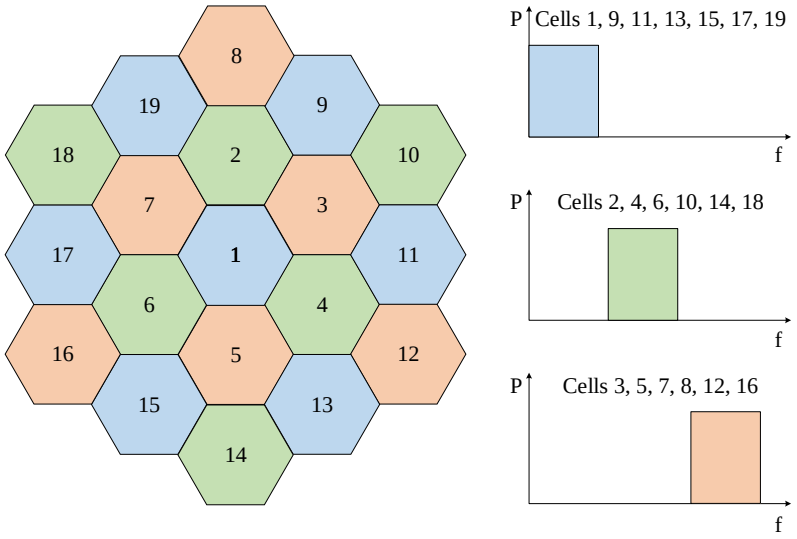


Figure 1. Frequency reuse in LTE-A

In order to deal with these issues, CoMP transmission and reception [10] is considered an effective method, especially for cell edge users [11]. In CoMP enabled systems, the Base Stations (BSs) are grouped into cooperating clusters. The BSs in each of these clusters exchange information with one another and jointly process signals to provide services to the users. A CoMP cluster can be formed based on static or dynamic clustering algorithms [12]. Furthermore, CoMP enables User Equipment (UEs), such as mobile phones, to receive signals simultaneously from more than one transmission point in a coordinated or joint-processing method [11, 13].

One of the challenges faced by CoMP-enabled systems is the need of accurate and up to date Channel-State Information (CSI) in the scheduler for adaptive transmission and appropriate Radio Resource Management (RRM) [11, 14]. In order to provide the scheduler with this information, the UEs estimate the CSI and report it to their serving BS periodically. This results in a significant increase of the feedback and signaling overheads [12, 13]. Furthermore, due to the large number of CSI feedbacks going around the network and the increasing queue sizes, the CSI feedback latency also increases.

Another type of overhead related to CoMP is called infrastructural overhead [13, 15, 16]. In this case, the networks require additional control units and low-latency links among the collaborating BSs, which might increase the network costs. This overhead mostly depends on the CoMP control architecture used and is different for the centralized and distributed control architectures [15, 16, 17]. In the distributed architecture, the cooperating cells exchange CSI over a fully meshed signaling network using X2 interfaces. This architecture increases the signaling overhead, and it is more sensitive to error patterns since these can be different for different BSs. This could be a potential reason for further performance degradation in the distributed architecture [16]. On the other hand, in the centralized architecture, a central unit is responsible for handling radio resource scheduling by processing the CSI feedback information from the UEs. This architecture also suffers from signaling overhead. Moreover, this architecture might have infrastructure overhead and also increases the CSI feedback latency [17].

With these issues in mind, in [9, 18] we proposed a CoMP control architecture named Direct CSI-feedback to Elected Coordination-station (DCEC), with the aim of reducing the signaling overhead and latency of the CSI feedback, which eventually will increase the throughput of the network. As shown in [19], the throughput of the cell can increase by as much as 20% if the latency is reduced by 5 ms.

Studying these new technologies is complex. When analytical models are not useful, Modeling and Simulation (M&S) provides good means for analysis, and different simulation engines have been used to simulate LTE networks. One promising approach to simulating such systems is the use of formal modeling approaches. Such types of development offer a high level of abstraction. One of these methods is Discrete Event

System Specification (DEVS) [20]. This formalism consists of a two-layer structure, which enables the developer to separate the simulator from the model. On the other hand, the model layer can be further categorized into sub-models to reduce the complexity of each model further. This allows the developer to test each model individually in the beginning and as a whole system towards the end of the development process. The above mentioned features of DEVS makes it a good fit for simulating complex systems such as wireless networks.

To be able to experiment with the new algorithms proposed in this research, and analyze their results, we built multiple models for simulation that permit comparing the new algorithms to other conventional CoMP architectures. The simulation models were developed in such a way that they maximize flexibility, allowing users to easily modify the structure and test any scenario in LTE-A networks.

1.1. Contributions

This thesis introduces three contributions. The main contribution was the development of a model library for CoMP, and the analysis of scenarios in which a new CoMP architecture can be beneficial. In addition, we discuss our contribution in the development of new CoMP algorithms aimed at minimizing the overhead imposed on Homogenous Networks. This contribution was done in collaboration with a team of researchers of Ericsson Canada Inc. and the ARS-Lab [21]. Finally, this same collaborative effort resulted in an extension of the proposed CoMP architecture to be applied to Heterogeneous Networks (HetNets).

The main contribution included the development of these algorithms as discrete-event models that were implemented and tested to enable the simulator to mimic the real-world behavior of cellular networks. These models were developed in accordance with DEVS.

In addition to the models, different complex message structures were developed to optimize the communication between the models while minimizing execution time. These messages align with the 3GPP standards that defines the types of messages exchanged in an LTE-A network.

1.2. Thesis Organization

The remainder of this chapter is organized as follows. In Chapter 2 we talk about some background on LTE-A networks, homogenous networks and HetNets in LTE-A, and CoMP. We also talk about the simulation of wireless networks and review some work that has been done for this matter. Finally, we present DEVS as a simulation environment for wireless networks and review the work that has been done in the field. In

Chapter 3, we define the problem we are interested in solving. In Chapter 4 we propose a new algorithm for CoMP aiming at enhancing the user experience. In Chapter 5 we present the simulation software and the models developed in detail. In Chapter 6 we demonstrate and analyze some results comparing the proposed architecture for CoMP to the other standard architectures. Finally, in Chapter 7 we conclude this thesis and talk about possible future work that can be done to extend this research.

1.3. Publications

Following is a list of publications based on this thesis:

- M. Etemad, B. U. Kazi, G. Wainer, and G. Boudreau, "Modeling coordinated multipoint (CoMP) transmission with dynamic coordination station in LTE-A mobile networks using DEVS," *IEEE International Conference on Communications (ICC)*, Paris, France, 2017, submitted.
- B. U. Kazi, M. Etemad, G. Wainer and G. Boudreau, "Signaling overhead and feedback delay reduction in heterogeneous multicell cooperative networks," *2016 International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS)*, Montreal, QC, Canada, 2016, pp. 1-8. doi: 10.1109/SPECTS.2016.7570507
- B. U. Kazi, M. Etemad, G. Wainer and G. Boudreau, "Using elected coordination stations for CSI feedback on CoMP downlink transmissions," *2016 International Symposium on Performance Evaluation of Computer and*

Telecommunication Systems (SPECTS), Montreal, QC, Canada, 2016, pp. 1-8. doi: 10.1109/SPECTS.2016.7570508

Chapter 2

Related Work

Cell edge users have always experienced degraded service due to two main factors: their distance to the cell tower and signal interference. To improve the user experience on the cell edge, LTE release 10 [22], introduced decode-and-forward relays to improve the user throughput on the edge. These specific types of Relay Nodes (RNs) are small nodes with low power consumption and do not suffer from limitations such as loop back interference between transmit and receive antennas [23]. LTE release 11 [5] introduced the concept of cooperative Multiple-Input and Multiple-Output (MIMO) also known as CoMP in which users are served by a group of BSs in a cooperative manner.

2.1. Coordinated Multipoint (CoMP) in LTE-A

As mentioned previously, CoMP is a method used for cell edge users to minimize interference and improve the user experience on the edge. Before CoMP, the 3GPP introduced MIMO in LTE to help meet the demand of high data rates [24, 25]. Although this proposal resulted in better performance, a possibility of further improvement was sought. In September 2011, 3GPP introduced a new method to cope with higher data rates demanded by users, named CoMP. CoMP transmission/reception is also known as Multipoint Cooperative Communication (MCC) and it improves the network performance by increasing the throughput on the cell edge [11]. In CoMP enabled systems, cooperating clusters contain a subset of the network BSs and are formed by grouping BSs. The BSs in

the clusters exchange information and jointly process signals by forming virtual antenna arrays distributed in space [12]. Furthermore, multiple UEs can simultaneously receive their signals from one or multiple transmission points in a coordinated or joint processing manner. This technique is an effective way of managing the inter-cell interference (ICI).

Since its release, CoMP has become one of the core features of LTE Release 11. Prior to this release, in LTE Release 8, the MIMO transmission in the cells were controlled independently from their neighbors. This would have disabled the BS to perform coordinated scheduling and resulted in independent scheduling of the resources.

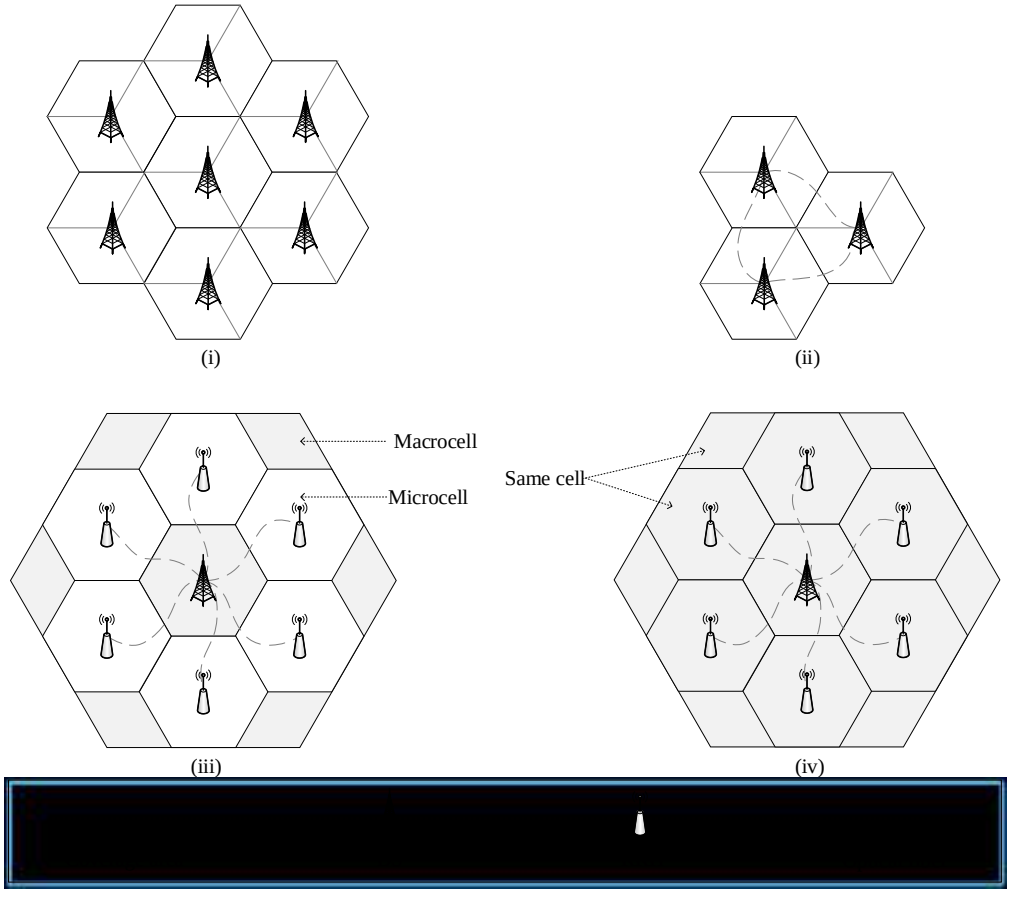


Figure 2. CoMP suggested scenarios in LTE-A release 11

Alongside the introduction of CoMP, 3GPP suggests the following four scenarios to be explored for the implementation of CoMP, which can be seen in Figure 2 [11, 26, 27]. These scenarios, shown in the figure, are as follows:

i. Coordination between cells with the same macro BS: this is a case of homogenous CoMP in which a cell is divided into 3 sectors, and a single BS is responsible for scheduling all the resources. In this case, no external connections between cells are necessary.
ii. Coordination between cells with different macro BSs: this is a case of homogenous CoMP in which more than one cell exists, and they are connected to one another. In this case, one BS controls the other BSs in the coordination area. The performance gain in this scenario depends on the number of cells involved and the connection latency.
iii. Coordination between macro and micro cells with different identities for the micro cells: this is a case of HetNet CoMP in which the high power macro cells and the low power micro cells co-exist. All the Remote Radio Heads (RRHs) in the micro cells are connected to the BS in the macro cell. Furthermore, each of the micro cells have their own Physical Cell Identity (PCI).
iv. Coordination between macro and micro cells with the same identity for the micro cells within the same macro cell: this is a case of HetNet CoMP like scenario iii, with the difference that all micro cells share the same identity of the macro cell. In this case, the support of handover among RRHs is not required.

The first two scenarios represent the cases for homogenous networks while the other two represent the cases for HetNets. Furthermore, 3GPP categorizes the CoMP techniques

into three overall categories: Coordinated Scheduling and Beamforming (CS/CB), Joint Transmission (JT), and Transmission Point Selection (TPS). CS/BS is used to reduce the interference experienced by the UE by selecting the beamforming weights, JT allows the neighboring points to transmit the desired signal rather than the interference signal, and TPS allows the UE to be scheduled by the most appropriate Transmission Point (TP) [28]. Figure 3 shows a case where (a) CS/BS is used, (b) JT is used, and (c) TPS is used.

As mentioned in the previous chapter, accurate and updated channel information is a key factor for achieving better throughput performance gain in CoMP. The CSI feedback process is a method in which a UE calculates the channel information and reports that to the BS so that the BS can perform adaptive transmission and appropriate RRM. This message is frequently sent from the UE to the BS in order to allow the BS to have the most up to date information required for scheduling. In [17] and [29], the authors state that the CSI feedback frequency can be 5 or 10 ms. The information embedded in the CSI feedback are the Rank Indicator (RI), Processing Matrix Indicator (PMI), Channel Quality Indicator (CQI), and Precoding Type Indicator (PTI). The RI indicates the number of independent data streams that can be supported by the channel in spatial multiplexing transmission. The PMI is a reference to a code word in a predefined codebook, which is used for calculating the beamforming weight. The CQI contains the quality of the target link. The PTI is a one-bit report added to the RI so that the UE can indicate the contents of the PMI and CQI reports [5, 30].

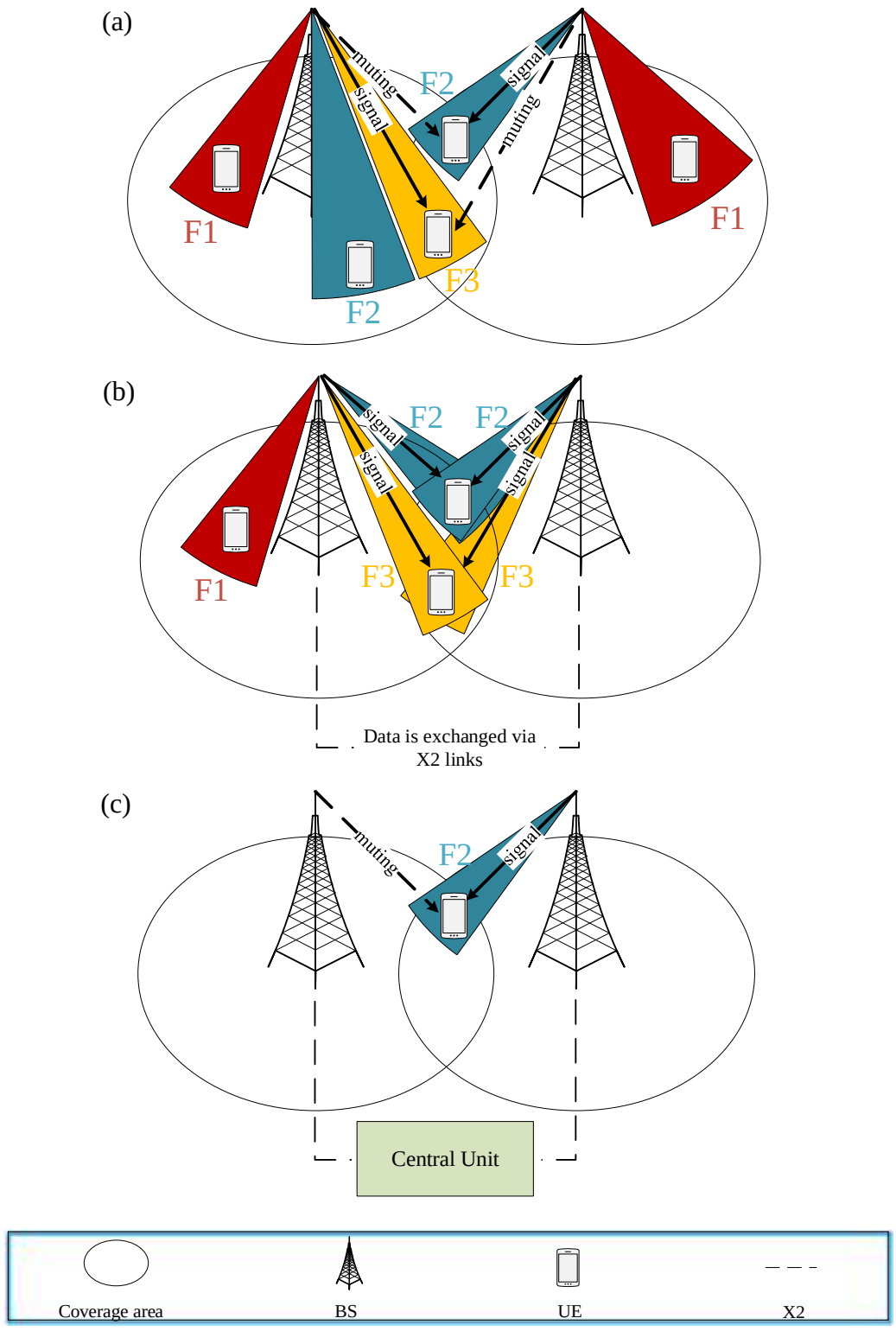


Figure 3. Categories of CoMP as outlined by 3GPP: (a) CS/BS (b) JT (c) TPS

As shown in Figure 4, there are two types of control architectures suggested by LTE release 11 for use in CoMP transmission and reception with respect to how the channel

information becomes available at different transmission points: centralized and distributed [15, 16, 17].

(a)

(b)

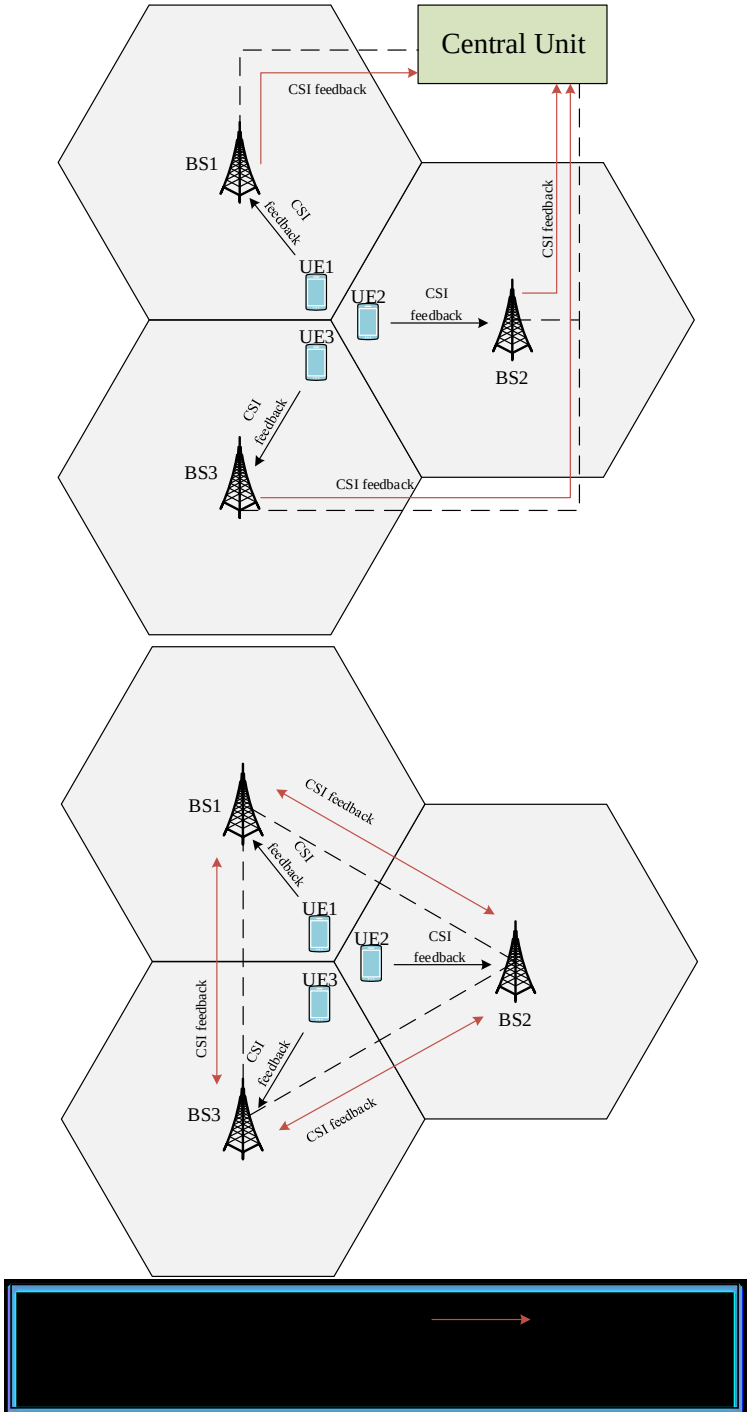


Figure 4. Standard CoMP architectures: (a) centralized (b) distributed

In the centralized architecture, a central unit is responsible for handling radio resource scheduling by centrally processing the feedback information from the cell sites. The UEs estimate the CSI related to all the cooperating BSs and feed it back to their serving BS. Their serving BS then forwards the local CSI to the Central Unit (CU). Finally, this CU calculates the global CSI and makes a decision based on the calculated information. This information is then sent back to the BSs. This framework suffers from signaling overhead and infrastructural overhead (as discussed earlier, this means that the network may require additional control units and links among the collaborating BSs), as well as an increase in the network latency. On the other hand, in the distributed architecture, the coordinated cells exchange CSI data over a star-like S1 network and a fully meshed signaling network using X2 interfaces. Prior to the download, the UEs estimate the CSI related to all the cooperating BSs and feed it back to their serving BSs. The BSs then exchange data over the backhaul and independently perform scheduling tasks based on their acquired global CSI. This architecture increases the feedback transmission, and is more sensitive to error patterns. This could potentially cause further performance degradation [16].

In addition to the abovementioned two standard approaches, different researchers have suggested new architectures for the deployment of CoMP in both homogenous networks and HetNets. In [31], the authors propose a centralized Medium Access Control (MAC) approach for CoMP joint transmission. In this approach, the authors group the BSs into clusters. Within a cluster, one of the cells is preconfigured as the head sector and the others act as proxies. They claim that this architecture is suitable for all cluster sizes. They then analyze the performance of the JT CoMP for the full buffer traffic model. Furthermore, they incorporate delay in the backhaul X2 links and conclude that a backhaul with low

latency is essential for the JT CoMP to achieve optimal performance. In [32], the authors propose a modified version of an existing algorithm for Dynamic Cell Selection (DCS) in CoMP. They extend the DCS method to a Multi-Cell scenario, which originally is limited to one chosen transmission cell. They state that their algorithm chooses the cell edge users based on their locations as opposed to the users received signal power which is a more accepted approach in proposals. They claim that this approach results in faster distortion-controlled system level simulation. Using simulations, they show that their proposed Multi-DCS method increases throughput slower than the Single-DCS but is more reliable and still works if the channel turns uncertain. In [33], the authors propose a distributed architecture for CoMP JT, which works over an IP backhaul network between BSs. They propose a control architecture in which two levels of time scales are used as a radio resource control cycle: (1) radio resources for CoMP JT are allocated every several 100s of milliseconds; (2) modulation and coding schemes for link adaptation are calculated every millisecond. They argue that by doing so they can effectively optimize radio resources in hundreds of milliseconds and calculate the Modulation and Coding Scheme (MCS) in a couple of milliseconds. They conclude that their proposed architecture optimizes the total system throughput. Furthermore, they claim that since their proposed architecture works based on IP network, the inter-cluster interference is minimized. Feng et al. [34] propose an enhanced DCS with muting method, which builds on the DCS with muting introduced by 3GPP in [35], to further improve power and frequency efficiency by using adaptive muting mode selection and flexible power allocation. They state that their technique helps eliminate downlink inter-cell interference. As a weakness for their algorithm, they state that applying this method will impose extra overhead on the uplink feedback signaling

compared to the conventional DCS with muting. They conclude that their proposed algorithm results in a 5.5% increase in cell average throughput gain and 10% in cell edge throughput gain compared to the conventional DCS schemes.

The authors in [36] propose a centralized scheme for fast inter-cell radio resource management which achieve CoMP transmission and reception. For this, they deploy a group of Remote Radio Equipment (RREs) and have them connected to a central unit via optical fiber. They point out that by doing so, since the interface between the central unit and the RREs may be designed in a single transceiver, fast transfer of the baseband digital signal is possible. They prove, via system-level simulation, that CoMP transmission in the downlink and reception in the uplink are very effective in improving cell edge throughput. Cili et al. [37] focus on improving the energy efficiency of CoMP. They combine the cell switch off scheme, which is available for LTE release 10 and beyond, with CoMP to optimize the power usage. Cell switch off is a scheme in which a cell with light traffic is switched off. To deal with the traffic in the switched off cell, the neighboring cells' transmit power is boosted to cover the area of the switched off cell. The authors propose to combine CoMP with cell switch off to be able to avoid the boost in power. In this approach, the switched off cell will be covered by CoMP between the remaining working cells. This way the users in the switched off cell will be served without the need of a power boost in any of the neighbouring cells. Tavanpour et al. [38] focus on improving the upload rate for users. They propose an algorithm which builds on CoMP and breaks down the file into smaller pieces for uploading. In this approach, the UE simultaneously uploads different pieces of the file to different BSs in the CoMP set. Once the BSs receive the piece, they send it to the serving BS. The serving BS then regenerates the file once all the pieces have

been received. They conclude through simulation that using their algorithm improves the users' data rates, however they do not study the effect of such algorithm on the backhaul.

2.2. Homogenous/Heterogeneous Networks in LTE-A

After the introduction of LTE networks which offer higher spectral efficiency, low latency, and high data rates, 3GPP has been working on further improving LTE in the framework of LTE-A. To do so, 3GPP has focused on higher order MIMO, career aggregation, and HetNets [39]. Furthermore, to cope with the evermore increasing demand of data rates, the designers decided to increase the bandwidth. This allows for the providers to offer higher data rates based on Shannon's theorem. With more spectrum in use, mobile users will spread across different frequency bands with different sizes of spectrum allocations. This requires a high spectrum flexibility [40, 41].

Wireless networks can be categorized into two major categories: Homogenous and Heterogeneous. As mentioned earlier, one of the techniques used to cope with high data demand of mobile users is adding to the spectrum that can be used by the users. However, this alone can be insufficient for addressing the high demands. A solution to this issue is densifying the network by adding more BSs, hence reducing the cell size. Although this approach does result in higher data rates and better user experience, it may not be feasible. Bringing cell towers closer together can only be pursued up to a certain degree because at a certain point, finding new sites to deploy macro cell towers becomes difficult and extremely expensive. An alternative approach, is to introduce small cells with low-power base stations called RRHs. RRHs are used for hotspots in which user demand is very high resulting in an overload on the macro BS. This results in offloading the macro cell and

improving network performance and service quality. The use of RRHs results in a heterogeneous network with micro cells within macro cells. Figure 5 shows a diagram of a (a) homogenous network vs. (b) heterogeneous network.

(a)

(b)

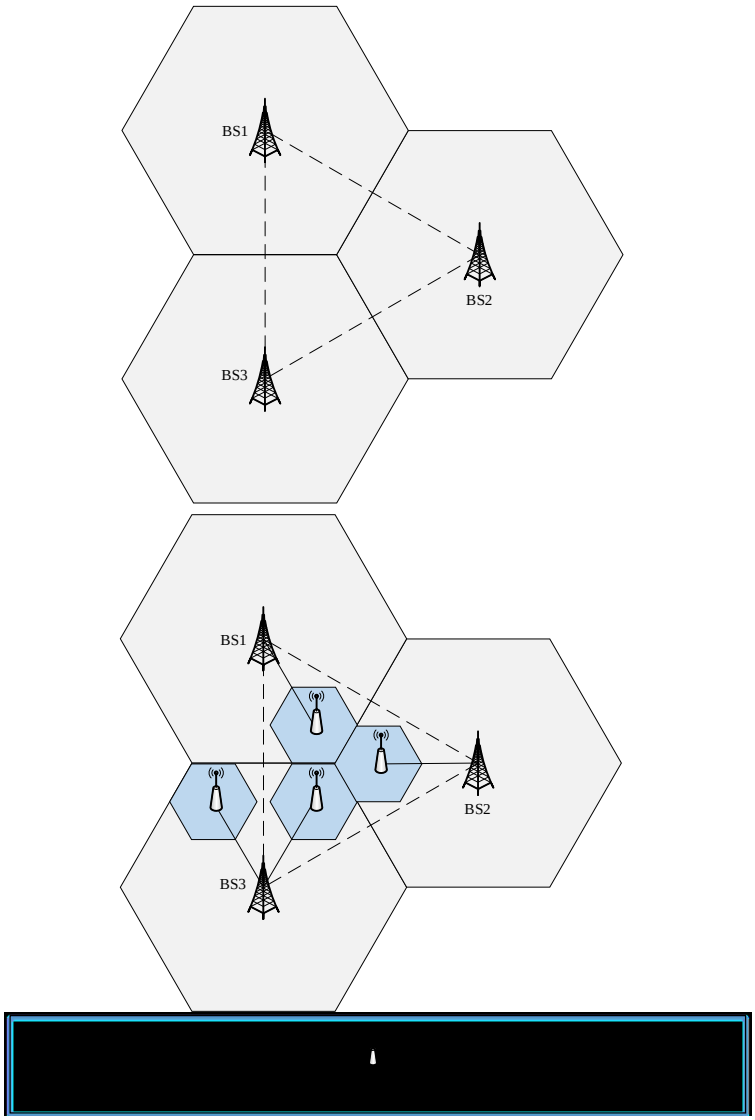


Figure 5. (a) Homogenous network (b) heterogeneous network

One of the challenges imposed by using HetNets is the interference management. In homogenous networks, each user is served by the BS with the highest signal strength and the signals from other BSs is treated as interference signals. In heterogeneous networks however, interference can result in very poor performance. To prevent this from happening,

advanced techniques for interference management should be used. Khandekar et al. in [39] outline two approaches for advanced interference management. Inter-Cell Interference Coordination (ICIC) and Slowly Adaptive Interference Management. In the first approach, the low power base station is responsible for performing control channel and data channel coordination with the macro base station. In the second approach, the resources are allocated over a timescale higher than the scheduling intervals. To do so, both central and distributed approaches can be considered [39].

2.3. Simulation of LTE Networks

To be able to evaluate the performance of LTE networks, simulations are critical. Given the cost of implementing new technologies on the network, it is impossible to test a proposed algorithm on the real network. Simulations can be used to keep the cost of testing new algorithms minimal while accurately evaluating the impacts of implementing such ideas.

Various simulators have been used to simulate wireless networks, including Network Simulator-2 (NS-2)/Network Simulator-3 (NS-3), OPNET, and OMNET++. NS-2 is a discrete event network simulator that started in 1989 as an open source simulator in which the timing of events is maintained through a scheduler. NS-2 uses two languages: a system language and a scripting language. The system language is based on C++ and the scripting language uses Object-oriented extension of the Tool Command Language (OTCL). Being an open source simulator, many models developed by researchers are freely available for non-commercial use.

As a successor to NS-2, NS-3 was released in 2008. Following the same structure, NS-3 models' behaviors are developed using C++. However, unlike NS-2, NS-3 uses C++ or python for defining network topologies and simulation parameters and no longer supports OTCL. Although this change has improved the performance of the simulator, it has eliminated backward compatibility and researchers must re-code their already developed models if they choose to use NS-3 instead of NS-2.

OPNET modeler uses C/C++ code for model development. It has a complete toolset to model protocols, devices, and technologies; simulate networks; and analyzing simulation results. It comes with pre-existing models for hundreds of protocols and vendor devices. It also has an integrated Graphical User Interface-based (GUI-based) debugging and analysis tool and an open interface for integrating external object files, libraries, and even other simulators. The OPNET modeler consists of a multi-level hierarchy: project editor, node editor, and process editor. The project editor is used to create nodes and link objects and to lay out trajectories to define node mobility. The node editor represents protocol and application functions and the flow of data to the device. Finally, the process editor is used to extend finite state machines to define protocol logic and control flow and uses C/C++ for coding the behavior of each state.

Similar to NS-2/NS-3, OMNET++ is also a public-source and component-based network simulator with GUI support. It provides a component architecture for models which are coded in C++, and then assembled into larger components using a high-level language called the Network Description (NED). This simulator is free for academic use but also has a commercial version available. One of the main disadvantages of OMNET++ is the issue of portability. Models developed by different developers may not work with

one another. Furthermore, OMNET++ has not been specifically designed for network simulation but is used for this purpose.

The approaches used to simulate wireless networks can be divided into two categories: link-level approach and system-level approach. The link-level simulators aim at simulating point-to-point physical layer technologies while the system-level simulators reflect the dynamic network behavior. Different researchers have used different tools based on their specific needs to simulate LTE networks. Ikuno et al. [42] use the LTE system-level simulator [43] on top a freely available LTE link-level simulator [44] to simulate in detail both the physical layer and the entire system. They use this simulator to evaluate the effects of a newly proposed scheduling algorithm. Mezzavilla et al. [45] use LTE-Evolved Packet Core (EPC), part of the NS-3, to implement a proposed lightweight link abstraction model for the downlink transmission. Virdis et al. [46] develop a simulator SimuLTE for the OMNeT++ simulation framework with a full protocol stack. Their simulator is used to simulate the data plane of the Radio Access Network (RAN) and the EPC. They state that their developed simulator consists of double as much code as LTE-sim [47] but also factors in more options such as mobility, IP/UDP, etc.

We decided to choose DEVS to study the newly proposed CoMP architecture. The hierarchical and discrete event nature of DEVS makes it a good fit for simulating wireless networks. Furthermore, the modular nature of DEVS allows for the developer to integrate different models with one another. This allows for model re-use with minimal change to the models. For example, if a model has been developed for crowd simulation, it can be also used as the model responsible for simulating the movement of UEs in a cellular network. Moreover, the formal specification of DEVS allows for easy model verification.

DEVS models can be executed in parallel without modifications, and be embedded in real-time platforms running in microcontrollers.

2.4. DEVS for Simulation of Cellular Networks

Introduced by Zeigler in the late 70s [48], DEVS is a formalism which allows developers to develop discrete event systems. By using a continuous time base, DEVS allows the user to achieve high timing precision. DEVS consists of a two-layer structure which enables the developer to separate the simulator from the model. Moreover, the model layer can be further categorized into sub-models to further reduce the complexity of each model. This allows the developer to test each model individually in the beginning and as a whole system towards the end of the development process. A DEVS model consists of behavioral (atomic) and structural (coupled) models. The atomic models are responsible for the behavior of the system and act as the basic building blocks of the system. On the other hand, coupled models maintain the hierarchical structure and can be made up of one or more atomic models. Models can be connected to one another via their input and output ports. When an input arrives at the input port of an atomic model, it triggers the external function and uses the time advance function to calculate the time until the next internal transition. Once the time has passed, the model will produce an output via its output function. Then the internal transition function generates the new state of the model. Figure 6 shows an informal depiction of DEVS atomic models [20]. The behavior of models is represented using the following formal notations:

$atomicDEVS = < X, S, Y, \delta_{ext}, \delta_{int}, \lambda, ta >$

$X = \{(p, v) | p \in IPorts, v \in X_p\}$, is the set of input ports and values

$Y = \{(p, v) | p \in OPorts, v \in Y_p\}$, is the set of output ports and values

S is the set of states

$\delta_{ext}: Q \times X \rightarrow S$, is the external transition function

Where Q is the total state of $M = \{(s, e) | s \in S \; and \; 0 \leq e \leq ta(s)\}$

$\delta_{int}: S \rightarrow S$, is the internal transition function

$\lambda: S \rightarrow Y$, is the output function

$ta: S \rightarrow R^{+}_{0,\infty}$, is the time advance function

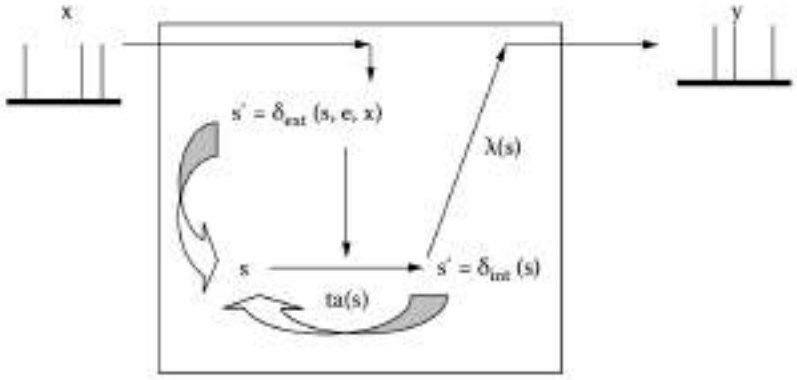


Figure 6. DEVS atomic model semantics

The hierarchical and modular nature of DEVS allows the description of multiple levels, enhances the reusability of models, and reduces the computational time by reducing the number of calculation for a given accuracy. Additionally, by using DEVS, the same model could be extended with different DEVS-based simulators, allowing for portability and interoperability at a high level of abstraction. Finally, the use of formal modeling techniques enabling automated model verification [49]. In DEVS, information is exchanged and events are triggered by the transmission of messages between models. This behavior closely resembles the behavior of wireless networks. Because of this and the previously mentioned points, DEVS can be a suitable fit for modeling and simulating wireless networks.

All DEVS models are written in C++ and consist of 6 functions: a constructor, initialization function, external function, output function, internal function, and a destructor. The input and output ports in which the model will use to communicate alongside the time which should take the model to complete processing a message is defined in the constructor. It should be noted that a model can have multiple input and output ports. The initialization function is used to initialize parameters that the model will use, for example the ID. The external function is called whenever a new message arrives. The programmer uses this function to extract any information required from the received message through the input ports as the other functions are not able to access the received message. The output function is for sending out desired information on any of the defined output ports in the constructor. The internal function is used to define internal state transitions for the model. Finally, the destructor is used to release managed and unmanaged resources. The order of execution of any DEVS model starts with the constructor followed by the initialize functions. At any given time, the model starts in a state S. If no input is received (i.e. no external events occur), the model will remain in state S until the lifetime of the current state expires. This lifetime is defined by the time advance function. The time advance can have one of these three values: 0, a real positive number, infinity. If the state has a lifetime of zero, the state is called a *transient state*, since no external events can occur during this time. If the state has a lifetime of infinity, the state is called a *passive* state which leads to the model being passivated. In this case, the model will remain passive until an external event is received. For the case that the state is assigned a real number for its lifetime, the system will remain in the state until the time advance reaches the lifetime of the state. Once this time is reached, the output function is executed and the model sends an

output on its output port(s). If an external message is received by the model before the lifetime of the state is reached, the model executes the external function followed by the output function. Therefore, it can be said that the internal function determines the next state of the model if no external events are received and the external function determines the next state if an external event is received [50] [51].

In order to connect the models to one another, the links between the models are defined in a model file. The model file also contains the name of all models and any number of arguments that can be used later in the initialization function of the models. A simple DEVS model and its corresponding Model file for a CD++ model of a scenario with one BS and one UE are shown in Figure 7 and Figure 8 respectively. Lightly shaded models are the coupled models and darker shaded models are the atomic models in the system.

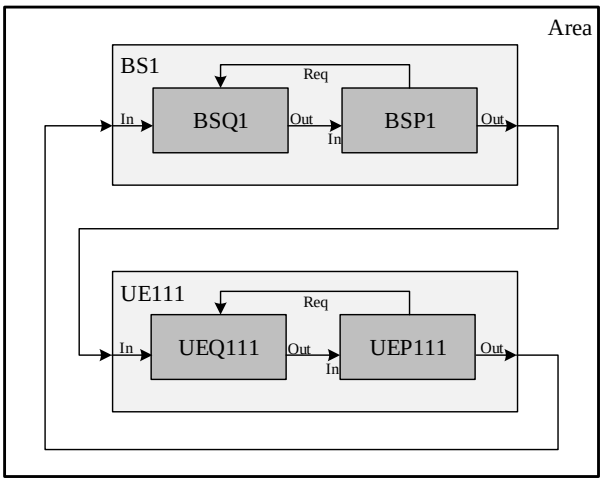


Figure 7. Simple DEVS diagram showing one UE connected to one BS

In this structure, the top components show the top-coupled models in the simulation. The links define the connection between the entities. If two entities are to communicate with one another, a link must be present between them. Each of the top-coupled models in this case contains two atomic models. Once again, just like the coupled models, links are defined for atomic models to enable them to communicate with their outside world.

Furthermore, each atomic model has a set of parameters defined. For example, the UEP111 model has an ID, a position defined by X and Y coordinates, *frequency*, which is the frequency of transmission, a *start time*, which shows when the UE will start transmitting messages in the network, and an *end time*, which if reached, the UE is forced to leave CoMP. If the *end time* is set to zero, the UE will stay in CoMP for the entire simulation.

```
[top]
components : BS1 UE111
Link : Out@BS1 In@UE111
Link : Out@UE111 In@BS1

[UE111]
components : UEQ111@globalQueue UEP111@UE
out : Out
in : In

Link : In In@UEQ1
Link : Out@UEQ111 In@UEP111
Link : Req@UEP111 Req@UEQ111
Link : Out@UEP111 Out

[UEP111]
ID : 111
currentX : 372
currentY : 1959
frequency : 900
sTime : 0
eTime : 0

[UEQ111]
ID : 111

[BS1]
components : BSQ1@globalQueue BSP1@BS
out : Out
in : In

Link : In In@BSQ1
Link : Req@BSP1 Req@BSQ1
Link : Out@BSQ1 In@BSP1
Link : Out@BSP1 Out

[BSP1]
ID : 1
currentX : 2308
currentY : 2500
frequency : 900
BSPower : 43

[BSQ1]
ID : 1
```

Figure 8. A simple Model file for a model with three BSs and three UEs

DEVS has been used for modeling and simulating wireless networks by different researchers. Using DEVS, Moallemi et al. [52] model a wireless network based on the 3GPP LTE specifications to study CoMP. Their model incorporates the properties of BS antennas, the UEs, and the area in which these two are located. They use their model to simulate the path loss and received power based on distance in both rural and urban settings. Tavanpour et al. [53] use DEVS to model and simulate standard CoMP techniques for LTE-Advanced networks. They set up an area as their top model in which they define different cells. Each cell consists of one BS and may contain multiple UEs. Furthermore, all BSs and UEs contain a queue and a processor. They define four types of messages for use between: (1) UE to UE, (2) UE to BS, (3) BS to UE, and (4) BS to BS. In another paper [38] they use DEVS to model a newly proposed algorithm for uploading large files. They develop a model for the BS, one for the UE, and one for a switch which acts as a connector for the BS and the UE. Al-Habashna et al. [54] use DEVS to model and simulate an algorithm for video download in wireless networks in which the video is segmented into pieces and cached in some members in the cell. They define a cell as the top model and place a BS, a Medium, and multiple UEs in it. Furthermore, they modify the model to present a distributed approach of their algorithm in [55].

The authors in [56] use DEVS to model IP networks in Cyber-Physical Systems (CPS). They aim at simulating CPS which are a mix of communication network models and power models, traffic models, weather models, and etc. They use the Multi-agent Environment for Complex Systems Co-simulation (MECSYCO) to create DEVS wrappers for their models and connect them to one another. Using this technique, they claim that they are able to model CPS including physical and IP models. They split the IP models into two sub

categories: spatial and structural. Using this, they are able to connect this separated models to models from other fields of expertise. Antoine-Santoni et al. [57] use DEVS to analyze Wireless Sensor Networks (WSN) performance. They define different types of messages and a number of atomic and coupled models for their purpose. Their aim is to analyze the performance of a WSN using different parameters such as energy consumption and CPU activity. They use a simulator written in pythonDEVS [58] to run their simulations.

Chapter 3

Problem Statement

One of the problems addressed in this thesis is the overhead imposed on the network by the transmission of CSI feedback messages required in CoMP. The transmission of excessive CSI feedback messages in a network may result in the two types of signaling and infrastructure overheads. Signaling overhead is a result of the UE estimating the channel coefficients for all cooperating BSs and feeding it back to the network. This is mandatory for the BSs to be able to perform their scheduling tasks. Infrastructural overhead is a result of the BSs needing to communicate with one another with very low delay. Due to the signaling overhead, the delay of the network increases. To overcome this, additional infrastructure such as extra links between BSs are necessary which results in additional network costs.

Another problem addressed in this thesis is the issue of latency. It has been shown that latency directly affects the throughput of the network. 3GPP categorizes latency into two categories: control plane latency and user plane latency [59]. Control plane latency is the time required for the UE to switch from the idle state to the active state to start receiving data. The user plane latency is the latency experienced by an application when exchanging data with a server. The metric used to characterize this delay is usually the PING delay.

Latency is inversely related to the network throughput and can cause a degradation in the network throughput. Research shows that the network throughput can improve by 20% if the latency is reduced by 5 ms. Table 1 shows the relation between delay and throughput loss [60].

Table 1. Loss of throughput with regards to network delay

Delay	5 ms	1 ms	200 µs
Throughput Loss	20%	5%	1%

The purpose of this research is to propose a new architecture for CoMP to improve the cell edge users' experience and to develop a simulator to model and simulate the conventional and proposed CoMP architectures. To do so, we have focused on proposing a new architecture for CoMP which reduces the number of control messages sent through the network, hence, reducing delay. As showed above, this will result in an improved throughput of the network. As reviewed in Chapter 2, there are two standard architecture proposed for CoMP by 3GPP. Furthermore, different researchers have proposed new architectures for the implementation of CoMP. Compared to the other architectures, our architecture not only improves the user experience by increasing throughput, but does so without the need of any additional infrastructure, therefore, eliminating additional infrastructural overhead and minimizing the costs of implementation.

To be able to evaluate the performance of the network after implementing the newly proposed CoMP architectures, a simulator was developed using the DEVS formalism. Using DEVS allows independent testing of each model, enabling the developer to pinpoint and debug possible errors easily. Other advantages of using this formalism have been outlined in Chapter 2. Alongside its advantages, DEVS proposes some challenges. Being a relatively new tool for modeling cellular networks, the community is limited. Furthermore, DEVS does not have pre built protocols to be easily used for communication. Although this may be thought as a challenge, it does on the other hand allow the developer to have more flexibility in development, and allows us to contribute also to the field of

modeling and simulation of discrete event systems (in particular using DEVS). Furthermore, having different levels in the architecture allows for a more precise and in depth study. Taking into consideration its advantages and disadvantages, we found that overall, DEVS provides a suitable framework for our intended work.

In the following chapters, we will present the newly proposed architecture for CoMP. We will also talk in detail about the simulator developed for simulating the newly proposed architecture and comparing it to the other two standard CoMP architectures. Finally, we will present some results comparing the performance of the CoMP architectures in different scenarios.

Chapter 4

An Algorithm for the Implementation of CoMP in LTE-Advanced

As previously mentioned, CoMP aims at boosting the cell edge throughput and therefore improving the user experience on the cell edge. It does so by grouping BSs into cooperating clusters. This enables the BSs to exchange information and jointly process signals and provide services to the user. Furthermore, it enables the user to receive signals simultaneously from multiple transmission points. Alongside its benefits, CoMP imposes its own challenges on the system. Currently, CoMP enabled systems suffer from feedback and signaling overhead as well as infrastructural overhead. In this chapter, we will discuss the current CoMP architectures in use and then we will propose a new architecture for CoMP that overcomes the challenges faced by the conventional CoMP architectures.

There are two types of architectures available for CoMP: centralized and distributed. In both approaches, the users calculate the CSI and send it to their serving BS. In the centralized approach, once the BS receives the CSI feedback, it sends it to the CU which is responsible for processing the messages and scheduling. Once the message is processed, the CU sends back the results to the BS which is then sent over to the UE. This approach suffers from both signaling and infrastructural overheads as it needs an additional hardware (CU) to be implemented in the network. Moreover, this architecture increases the CSI feedback latency as the message is required to travel two extra hops. On the other hand, in the distributed approach, the BSs exchange the received CSI over a fully meshed network.

Once all the BSs receive the information from all other BSs, they will independently schedule their resources accordingly. This architecture not only increases signaling overhead, but is also more sensitive to error patterns. In order to overcome these issues and enable the user to experience lower delays, we propose the DCEC algorithm to be used for CoMP enabled systems.

The control architecture of CoMP can be defined as the way the participating BSs coordinate to handle interference and scheduling to serve the UEs. In case of DL transmission, the CoMP signaling overheads are related to the inherent need of CSI at the transmit end [61]. This global CSI feedback process could be different based on the CoMP architecture. Two major challenges of the conventional CoMP architectures are the CSI feedback latency and signaling overhead. Latency is inversely related to the throughput of the network, for the coordinated schemes. Furthermore, if the feedback latency of the cooperating network is greater than the CSI feedback periodicity, then the scheduler will receive a backdated CSI. Hence, in the DCEC architecture, our goal is to reduce the CSI feedback overhead and latency to improve the cell throughput.

As mentioned in the previous sections, multiple architectures have been proposed to improve CoMP. Although these algorithms do improve the user experience in one way or another, none of them aim at reducing the number of messages communicated between the BSs and the UEs and some of them require more packets to be transmitted over the network. This can result in a waste of battery power in UEs and unnecessary occupancy of resources. Our algorithm, on the other hand, not only improves the user experience, but does so by reducing the number of CSI feedback messages that are travelling around the

network. Aside from improved user experience, this can help in improving the power and resource use efficiencies of the system.

In the DCEC architecture, one of the BSs in the CoMP cluster is dynamically elected to act as a Central Coordination Station (CCS) for the UEs. After a CCS is elected, all the UEs in the CoMP cluster with the same CCS will send the CSI feedback to this CCS only. The CCS will then calculate the global CSI information, determining the cooperation set, and will oversee scheduling. It should be noted that a cooperation set is a set of BSs within the CoMP cluster that can jointly serve the UE [10].

The main goals of this DCEC architecture are:

1. to reduce the latency of the network;
2. to reduce the feedback overhead of the network;
3. to avoid the additional infrastructure cost and;
4. to increase the cell throughput.

There will also be no increase in the error pattern in this architecture since all the participating UEs send the CSI directly to the CCS only. Furthermore, no additional hardware is necessary for this solution, so the costs for switching to such architecture will be minimal.

The main mechanisms aiming at reducing latency and overhead include network architecture optimization, shorter Transmission Time Interval (TTI), faster feedback processing, and QoS load differentiations [62]. We propose the use of an elected coordination station for CSI feedback, which addresses both abovementioned challenges (latency and overhead). This control architecture, named DCEC, dynamically uses one of the BSs in the CoMP cluster as a CCS for the UE.

The CCS is chosen based on an election algorithm, which will be described in detail later in this section. All the UEs in the CoMP cluster having the same cooperation set send the CSI feedback to the same CCS only. Therefore, this signal does not need to travel over any additional X2 or S1 links, avoiding extra latency of the CSI feedback transmission and reducing the imposed feedback overhead on the network. Figure 9 shows a simplified view of the proposed CSI feedback architecture after the CCS has been elected within the CoMP set. Although DCEC requires few more control packets to elect the CCS at the beginning, it outperforms the other two architectures as time advances. Furthermore, in DCEC the CSI feedback does not need to travel over any X2 or S1 links, which results in lower feedback latency.

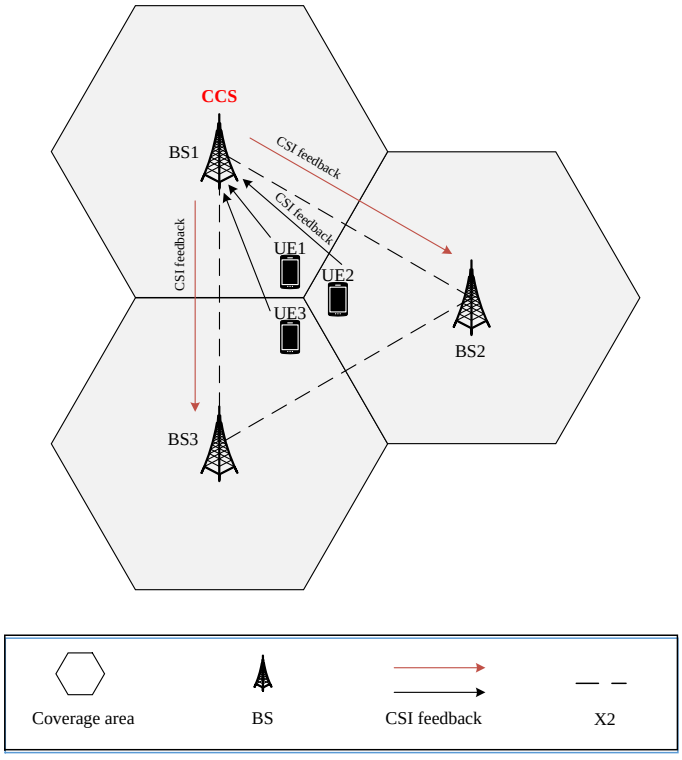


Figure 9. A simplified view of the DCEC architecture

To elect a CCS dynamically we use the following algorithm [18]:

1. A serving BS receives the CSI Feedback from a UE and calculates the CoMP cooperating set.

2. If a CoMP cooperating set contains more than one BSs, the serving BS in the CoMP set declares itself as a CCS

3. The declared CCS sends a CCS-Declaration message to other BSs in the set (containing the ID of the sender, the ID of the CCS, and the cell throughput of the CCS)

4. After receiving the message, other BSs in the set compare their throughput with the received CCS throughput.

    i) If the received CCS throughput is higher or equal than the recipient's throughput (or the current), the CCS ID will change to the received ID. The recipient then forwards the new CCS information to the BSs in the cooperation set except the sender.

    ii) If the received CCS throughput is equal to its own throughput (or the current), and the CCS ID is smaller than its own ID (or the current), the current CCS ID will become the received CCS ID. The recipient then forwards the new CCS information to the BSs in the cooperation set except the sender.

    iii) If the received CCS throughput and ID are equal to the current CCS throughput and ID, the CCS has been elected. Stop.

iv) Otherwise, the recipient BS declares itself as the new CCS and sends a CCS - Declaration message to the other BSs in the CoMP cooperation set.

5. If the cell throughput or cooperating set change, go back to step 2.

Figure 10 shows a simplified signaling procedure of the proposed scheme. We assume that BS2 and BS3 both meet the requirements of CoMP for UE1. At the beginning, the UE reports the CSI feedback to its serving BS; for instance, as shown in Figure 10, UE1 reports CSI feedback to BS1. Then, BS1 calculates the cooperation set for UE1. To do so, BS1 checks the channel quality and compares the predefined CoMP threshold (6dB as discussed in [37] and [63]) with the data received. If the cooperation set contains more than one BS, BS1 initiates the algorithm to elect the CCS by sending a CoMP request message to other BSs in the cooperation set (BS2 and BS3) with his own cell throughput. For simplicity of the explanation, we assume that in this case all three BSs are in the cooperation set. After receiving the CoMP request message, BS2 and BS3 check their own resources and compare the received throughput with their own throughput. Based on the availability of resources they send back a request grant/reject message, including the highest throughput. After receiving the responses from other BSs, BS1 decides about who the CCS is, and it advertises it to BS2, BS3, and to UE1 using a CoMP notification and CoMP command message respectively. Finally, the UE replies using the ACK message and switches to the CoMP mode. After the CoMP is established and the CCS is elected, the UE sends the CSI feedback only to the CCS, as shown previously in Figure 9.

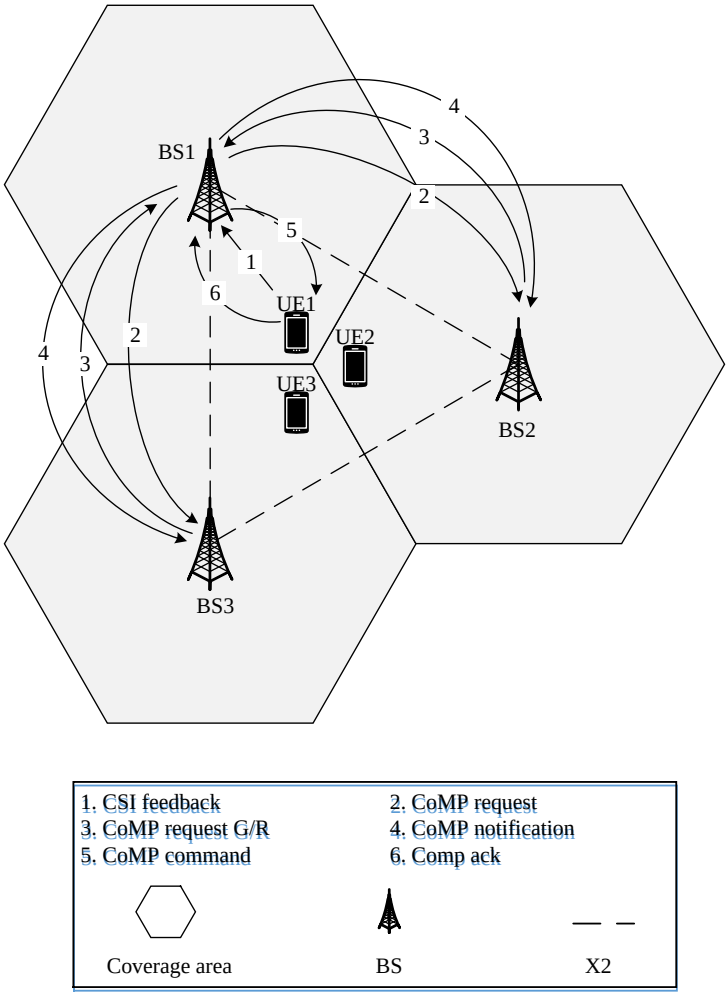


Figure 10. CCS election and CoMP establishment in DCEC

Since the DCEC architecture makes use of the existing hardware in the network, there are no infrastructural costs associated with its implementation. The only downside of using DCEC is that this architecture will require some additional messages going around the network to elect a CCS. This will not significantly affect the overall performance but can degrade the performance of the network for a short period. This period can be thought as the transient phase of this architecture. Furthermore, as mentioned above in the election algorithm, the CCS election will happen every time a change is detected in the throughput of any of the cooperation set cells. This means that, if the throughput of a cell in the set changes rapidly, the DCEC architecture can impose more overhead on the network

compared to the other two architectures. Given enough time to recover, DCEC outperforms the other two architectures. Although this is a shortcoming of this architecture, it is not normally the case since in practice, the CCS change does not occur that frequently. This is because for most UEs the maximum movement speed is 3km/h as suggested by the 3GPP release 11 for CoMP deployment [10].

4.1. DCEC Architecture in HetNets

As mentioned earlier, to densify the network to be able to provide higher data rates to users, service providers use RRHs. The use of these radio heads alongside high power BSs results in HetNets. However, the coexistence of macro and low-power cells brings technical challenges: Inter-cell Interference Coordination (ICIC, defined in 3GPP release 8 as a coordination technology used that forces UEs located at the cell edge but belonging to different cells, to use different frequencies); mobility management (a function which tracks the position of subscribers), and backhaul provisioning (setting up the connectivity of the LTE backhaul service to be consumed by end users). Since our proposed algorithm aims at improving the user experience on the cell edge, and given that more users experience interference and low signal strength in HetNets compared to homogenous networks, we decided to extend the DCEC algorithm to HetNets. To do so, minor changes were made in the algorithm to modify it to work with HetNets. Before introducing the new algorithm, we will talk about how HetNets work in terms of CSI feedback transmission and scheduling of resources.

RRHs can be thought as low power BSs with a small difference. They do not have the Baseband Unit (BBU), hence they cannot oversee scheduling resources. To be able to

schedule the resources in a small cell, RRHs send the information to a BS which they are connected to via optical fiber links. The recipient BS then processes the message and preforms the scheduling. Once the scheduling of resources is done, the BS will send back the results to the RRH which will then be sent to the UE. The signaling procedure for DCEC with the presence of RRH slightly differs from the procedure for DCEC in homogenous networks. In this case, UE1 reports the CSI feedback to its serving RRH (RRH11). RRH11 then forwards the CSI to BS1. After receiving the information from the RRH, BS1 calculates the cooperating set for UE1. To calculate the cooperating set, BS1 checks the channel quality and compares the predefined CoMP threshold (6dB as discussed in [37] and [63]) to the received data. If the cooperating set contains more than one BS, the serving BS (BS1) initiates the election algorithm to elect the CCS by sending a CoMP request message to the other BSs in the cooperating set such as BS2 or/and BS3 with his own cell throughput. After receiving the CoMP request, BS2 or/and BS3 will check their own resources and compare the received throughput with their own. Based on the availability of resources they will send back a request grant/reject message, including the highest throughput. After receiving the responses from the other BSs, the serving BS (BS1) will decide about the CCS and it will advertise it to the other BSs (BS2 and BS3) and the UEl by a CoMP notification message. Finally, the UE will reply using the ACK message and will switch to the CoMP mode. After the establishment of the CoMP and when the CCS has been elected, the UE sends the CSI feedback only to the CCS, as shown in Figure 11.

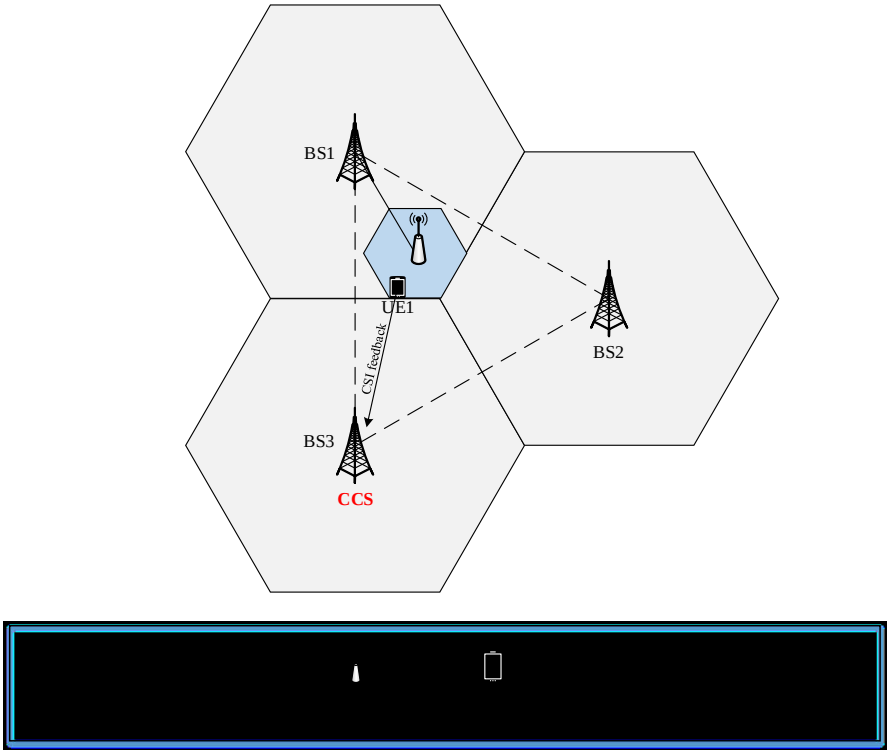


Figure 11. Simplified view of the CSI feedback process for the DCEC-HetNet architecture after CoMP has been established and CCS has been elected

In the next chapter, we will talk about the simulator developed and used for simulating the proposed architectures in detail, and we will discuss the different simulation scenarios used to study our proposed algorithms. Finally, we will present and discuss in detail the results obtained from the simulations.

Chapter 5

Simulation Software and Models

In the previous chapters, we reviewed different standard architectures for CoMP and proposed a new architecture called DCEC that overcomes the shortcomings of standard architectures. To be able to investigate the effect of the newly proposed architecture on the network further, we developed a number of models to simulate the algorithm. The models have been developed based on DEVS and take advantage of some of the properties of this specification. As mentioned in the previous chapters, DEVS is a system specification that uses a continuous time base and can thus achieve high timing precision. Other properties of DEVS that makes it useful for our purpose were discussed in detail in section 2.4.

A library of DEVS models was developed using the CD++ tool to allow for the testing of the CoMP architectures. This library uses complex messages to optimize the communication between entities while minimizing simulation run time. Four unique models were developed to allow the user to model and simulate different real world scenarios.

Multiple scenarios were considered to test the proposed algorithm. In this chapter, we will first talk about the software structure and the models developed to allow for mimicking the real-world scenarios as accurately as possible. Then we will discuss the scenarios used to test the proposed DCEC architecture. Finally, we will analyze the results and conclude the chapter.

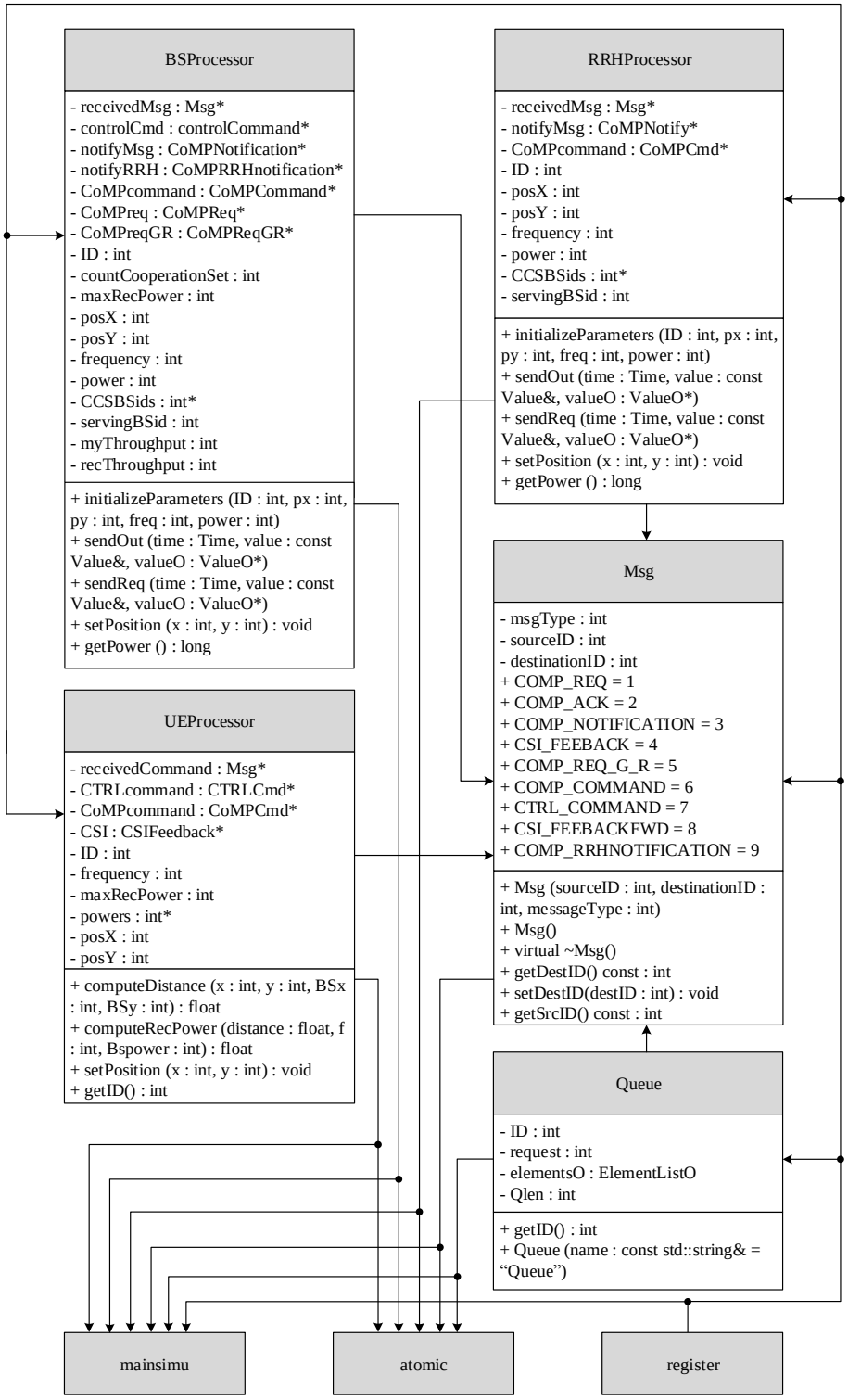


Figure 12. Simplified class diagram of the models

5.1. Software Architecture

Several models were developed to test the DCEC CoMP architecture and compare it to the other two conventional CoMP architectures. Figure 12 depicts a simplified UML class diagram of the software architecture describing models developed for this proposed architecture. In this figure, the BS class represents the *BSProcessor*, the RRH class represents the *RRHProcessor*, and the UE class represents the *UEProcessor*.

As seen in Figure 12, the models used for implementing the CoMP algorithms are: *Queue*, *Msg*, *UEProcessor*, *BSProcessor*, and *RRHProcessor*. Here we will talk about each model in detail (Note: the *Queue* model will not be discussed as it is a simple implementation of a buffer with a First In First Out (FIFO) strategy).

5.1.1. UEProcessor

This model is responsible for the behavior of the UE. To distinguish between the instances of this model, all of them are assigned unique IDs. Furthermore, the instances contain a set of XY coordinates which represent the position of the UE, a frequency, and a transmit power. The model is responsible for mimicking the behavior of a user in the real world. It processes incoming messages and generates outputs based on them.

Figure 13 shows a DEVS graph representing the basic global states of the *UEProcessor* model. Question marks ("?") stand for input messages and exclamation marks ("!") represent output messages. The states for this model are: *Idle*, *ReceivePack*, and *SendPack*.

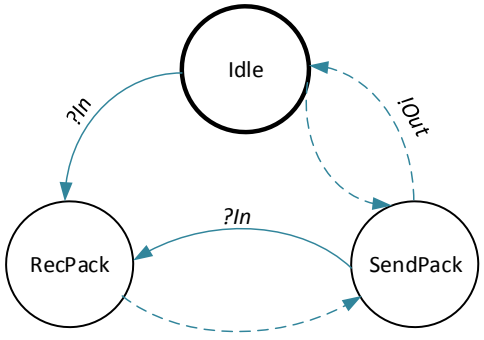


Figure 13. UEProcessor DEVS graph

The UE receives different message types form the BS and RRH based on its physical position in the network. This model is responsible for processing the received messages, extracting useful information, and producing an output based on the conditions. The UE can send a *CSI Feedback* message to a BS or RRH. The CSI feedback, as talked about in the previous chapters, includes some information which is used by the BS for scheduling purposes. The details embedded in this message will be talked about later in this chapter.

The UE can also receive two types of messages: *Control Command* and *CoMP Command*. The *Control Command* message is sent initially to the UE by all BSs and RRHs in the network. This message includes some fields (discussed later in this chapter) which allows the UE to calculate its distance from each transmission point using the *ComputeDistance* method defined in the UE. Using the distance, the UE then calculates its received power from each and every BS and RRH. This is done by using the *ComputeRevPower* method. The results of the calculations are stored in a vector and are sent to its serving BS by embedding it in the CSI feedback.

The *CoMP Command* message indicates what the UE must do in terms of joining or leaving CoMP. Once the UE receives this message, it will extract the information which includes the ID of the newly elected CCS. The UE will then update its records and send

the CSI feedback to the CCS only. The UE will keep sending the CSI feedback to the CCS until a new *CoMP Command* message is received. Depending on the command, the UE may either start sending the CSI feedback to a newly elected CCS or start sending the CSI feedback to its original serving station if CoMP no longer exists. In other words, this message is received by the UE if (a) a new CoMP session has been established or (b) the CoMP session has ended.

It should be noted that the UE has only 1 input and 1 output port which are air links. The UE is not connected to any other device using any physical links. Figure 14 shows a code snippet of some parts of the *UEProcessor* in which the UE extracts information from the *Control Command* message and computes the distance and the received power and lastly it indicates its serving BS based on the received powers.

```
if (msg.port()==In)  {
       state = SendPack;
       if (msg.valueO()->getMsgT() == controlCommand) {
              ... //extract useful information from the message
              for (int i=0; i < members.length(); i++){
                     //compute distance to the BS/RRH
                     dis = computeDistance(posx, posy, posBSx[i], posBSy[i]);

                     //received power from the BS/RRH
                     power = computeRecPower(dis, f, BSPower[i]);

                     //save received powers and their IDs in vectors
                     receivedPows.insert(receivedPows.begin()+i, power);

                     IDs.insert(CSIVectorIDs.begin()+i, i);
                     If (power > maximumRecPower){ //new maximum
                                   maximumRecPower = power;
                                   maxPowID = i;
                     } else if (power == maximumRecPower)
                                  //power is equal to the maximum
                                   maxPowID = std::min(i,maxPowID);
                      }
              }
              else if (msg.valueO()->getMsgT() == COMPCommand){
                     ...
                    // extract useful information to use later
              }
       }
}
```

Figure 14. Code snippet of part of the UEProcessor

As it can be seen in the above code snippet, a tie breaking mechanism has been incorporated in case the received powers from multiple transmission points are equal. For this, the UE simply selects the BS/RRH with the lowest ID.

5.1.2. BSProcessor

This model is responsible for the behavior of the BS. Just like the *UEProcessor*, all instances of this model also have unique IDs and XY coordinates. Furthermore, the instances contain a transmit power which represents the power of the BS and a throughput which represents the cell throughput. The model is responsible for mimicking the behavior of a BS in the real world. It processes incoming messages and generates outputs based on them. The *BSProcessor* behavior follows the following algorithm:

1. If there are no messages being processed, request a new message from the queue through the *Req* port.
2. If a new message is received, process the message, extract the required information, perform corresponding actions, and generate an output.
3. Send an acknowledgement message through the *Req* port to delete the message from the queue.

Figure 15 shows the DEVS graph representing the basic state changes of *BSProcessor*. The representation follows the same format as explained in the previous section. The states for this model are: *Idle, ReceivePack,* and *SendPack*.

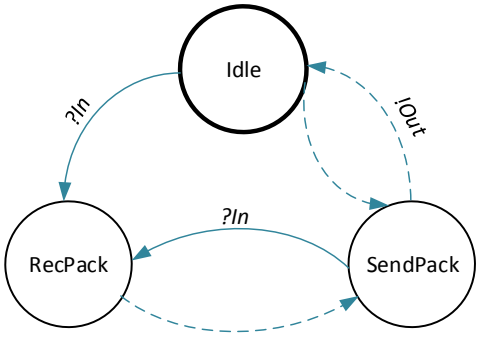


Figure 15. BSProcessor DEVS graph

The BS can receive messages from RRHs, other BSs, and UEs. Three different input and output ports which are *Air, X2,* and *Optical Fiber*. The BSs can send and receive messages to/from the UEs via air links. They can communicate with other BSs via X2 links and with RRHs via optical fiber links. The BS is assigned a random throughput once it is initialized which represents the throughput of the cell and will be used in the CCS election algorithm. This model can send a *Control Command* to the UE, which we discussed earlier, a *RRH Notification* to BSs to make them notify their RRHs of the CoMP coordination set once a CCS has been elected or if a CoMP session has ended, a *CoMP Command* to the UE which we talked about earlier, a *CoMP Request* message to other BSs to request them to join CoMP, a *CoMP Request Grant/Reject* message to the initiator of the *CoMP Request* message to grant or reject the request based on its available resources, and a *BS Notification* to other BSs to notify them of the elected CCS or to notify them if a CoMP session has ended.

This model can receive a *CSI Feedback* message through its *In* port, a *CSI Feedback Forward* its *OFin* port, a *CoMP Request,* a *CoMP Request Grant/Reject*, a *CoMP Notification*, or a *CoMP RRH Notification* through its *X2in* port.

If a *CSI Feedback* message is received from a UE, it means that the BS is the serving BS of that specific UE. To process the message and act accordingly, the BS will first extract the signal powers received by the UE and their corresponding transmission point IDs. This way, the BS will have a table containing all received signal powers by the UE and their sender IDs. Once this process is done, the BS will calculate the difference of its own signal power as received by the UE with all other transmission points. If the difference of the powers is less than the CoMP threshold (6 dB), the BS will initiate CoMP by sending a *CoMP Request* message to the members which meet the criteria. It should be noted that if an RRH meets the requirements, two cases are considered:

1. The RRH is in the same cell as the serving BS: this means that there is a direct link between the serving BS and the RRH, therefore the *CoMP Request* message is directly sent to the RRH.
2. The RRH is not in the same cell of the serving BS: this means that there are no direct links between the serving BS and the RRH, hence the message should travel through another BS which has a direct connection with the RRH. In this case the *CoMP Request* message is sent to the BS connected to the RRH.

If a *CSI Feedback Forward* message is received from an RRH, the BS will perform the same steps as if a *CSI Feedback* message has been received. If the conditions of CoMP are satisfied, the BS perform the same steps as above.

If a *CoMP Request* is received, the BS will check its resources to see if it has enough resources to join CoMP. Based on this, it will send a *CoMP Request Grant/Reject* message to the requesting BS. It should be noted that if a *CoMP Request* message is intended for an

RRH, the BS will check the RRH resources and perform the scheduling as the RRH lacks the BBU.

If a *CoMP Request Grant/Reject* is received, the BS will update its references and add the BSs which granted the request to the CoMP set. Once all grant/rejects have been received, the BS will send a *CoMP Command* to the UE to have it switch to CoMP mode.

If a *CoMP Notification* is received, the BS will update its references based on the elected CCS. The BS will know it is in CoMP and will participate in jointly serving the UE.

Multiple complex data structures have been defined for the BS to help it keep track of its CoMP sets. Each BS keeps a record of the UEs it's serving. These records include the UE IDs and the IDs of the transmission points which are jointly serving the UE. This way, the serving BS will know which BSs/RRHs should receive a notification should the CoMP conditions change. For example, if a UE leaves CoMP, its elected CCS before leaving, sends a notification to the CoMP members letting them know that they will not be jointly serving this UE anymore. Then the CCS will send a notification to the UE allowing it to switch to its serving station and sending its CSI feedback to its serving station.

To show how the election algorithm has been implemented, Figure 16 shows a snippet from parts of the code responsible for the algorithm in the *BSProcessor*. The code is responsible for the election algorithm for UEs that are connected to the BS. The same procedure happens for UEs that are connected to an RRH. Once the BS receives the *CSI Feedback Forward* message from the RRH, it will follow the exact same steps to initiate CoMP.

```
if (msg.valueO()->getMsgT() == CSIfeedback){ //received from the UE
       ... //extract useful information
       CSIfeedbackFlag[sourceUEID] = true;
       //this flag is later used to send respective outputs to other models
}

if (msg.valueO()->getMsgT() == CSIfeedbackFWD){ //received from the RRH
       ... //extract useful information
       CSIfeedbackFWDFlag[sourceUEID] = true;
}

if (msg.valueO()->getMsgT() == CoMPreq){
           //received a CoMP request from another BS
       ... //extract useful information
       CoMPrequestFlag[sourceUEID] = true;
}

if (CSIFeedbackFlag[sourceUEID] == true && countCooperationSet > 1 &&
isInCoMP[sourceUEID] == 0) {
     //the UE is not in CoMP but meets the requirements
    for (int i=1; i < members.length(); i++){
       if ((CoMPcooperatingSetIDs[i][ sourceUEID]!=this->id &&
              CoMPcooperatingSetIDs[i][ sourceUEID]!=0){
          //send CoMP request to BSs that are not me and are in the CoMP set
          requestMsg = new CoMPreq(this->id, CoMPcooperatingSetIDs[i]
             [sourceUEID], 1, MyThroughput, this->id, SourceID, CoMPmembers);
          //constructing request message and declaring itself as ccs
          sendOutput(msg.time(), X2out, NULL, requestMsg);
          //sending to other BSs over X2 links
             }
}
...
if (CoMPReqFlag[sourceUEID] == true){
       if (MyThroughput > RecThroughput) // my throughput is the highest
       ... //declare myself as CCS; send Grant/Reject message to requesting BS

       else if (MyThroughput < RecThroughput) // throughput is not the highest
              ...  // accept current CCS, update my records; Grant/Reject
                   // message to the requesting BS
       else //I have the same throughput as the currently defined CCS
              ... // compare IDs, set the BS with the lowest ID as the CCS,
                  // update tables, send the Grant/Reject message out

       CoMPReqFlag = false; //update the flag
}
```

Figure 16. Code snippet of part of the BSProcessor responsible for the election algorithm

5.1.3. RRHProcessor

This model is responsible for the behavior of the RRH and acts just like the *BSProcessor* except that it does not perform any scheduling. Since the RRH is missing the BBU, it cannot schedule resources and initiate CoMP. Therefore, the election algorithm is not available in the RRH. The RRH forwards receiving messages to its BS and updates its

information based on what the BS sends to it after processing the control messages. This model has been defined because the inputs and outputs of the RRH are different from those in the BS. The algorithm of this model follows the same steps of the *BSProcessor* algorithm and the DEVS graph representation is identical to the on in Figure 15.

The DEVS model in Figure 17 shows the overall view of how the models are connected to one another to construct the top model. It should be noted that the model can contain as many BSs, RRHs, and UEs as is required. Furthermore, the DEVS model has been defined without the definition of cells. This way, no UE is bound to a specific cell which allows for a more dynamic and realistic scenario.

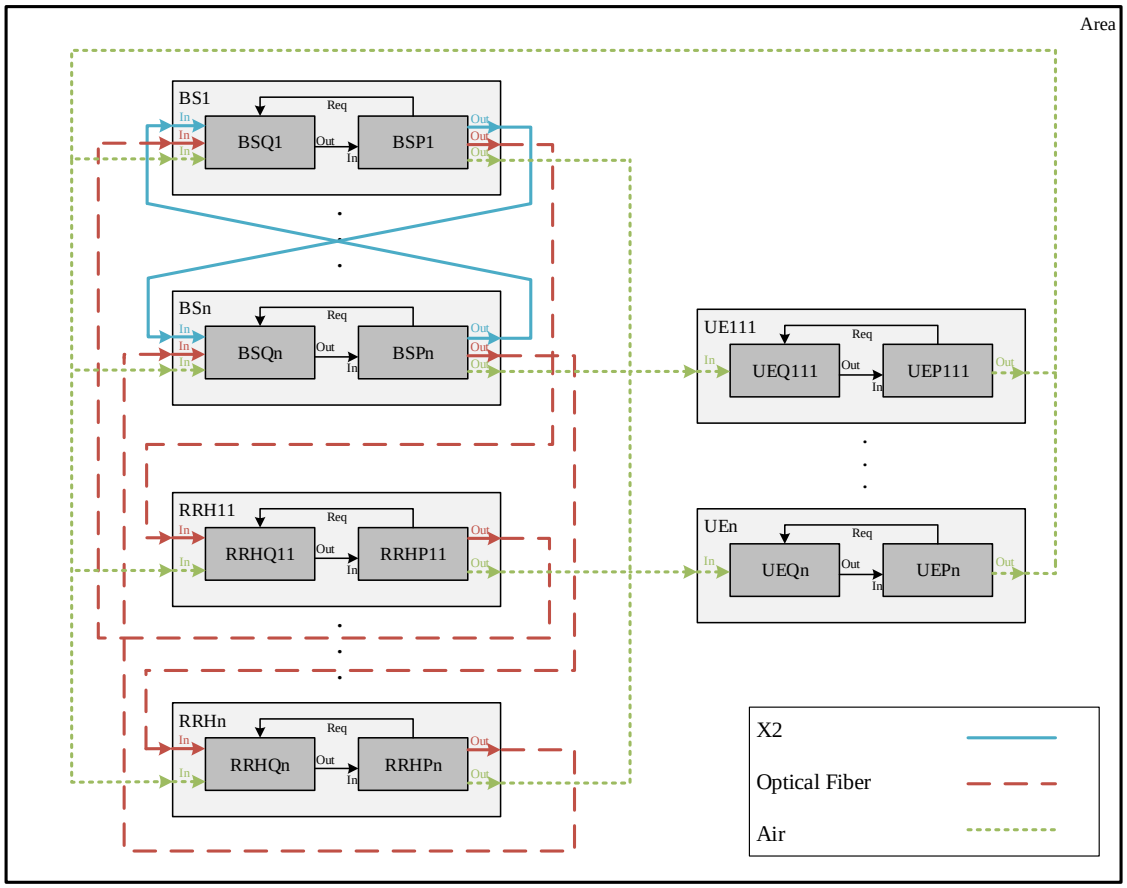


Figure 17. DEVS model: overall view of the top model

In the above figure, BSQ refers to *BSQueue*, RRHQ refers to *RRHQueue*, and UEQ refers to *UEQueue*, which are all an instance of the *Queue* class, the BSP refers to *BSProcessor*, the RRHP refers to *RRHProcessor*, and UEP refers to *UEProcessor*. Furthermore, the numbers at the end of the names indicate the ID of that model. As it can be seen, the UEs are connected to all BSs and RRHs via air links. This allows the UEs to send and receive messages to/from all BSs and RRHs based on the UE position. All BSs and RRHs are also connected to all UEs ensuring the successful delivery of the messages. These connections are set up via air links. The BSs are connected to one another to enable them to exchange messages over the backhaul. These links are X2 links. They are also connected to their own RRHs via optical fiber. As shown, not all BSs are connected to all RRHs. Finally, RRHs are also connected to their serving BS. RRHs can be connected to one and only one BS.

To differentiate between different messages going around the network, nine types of complex messages were developed. This allowed us to implement a specific method for dealing with each type of message. To ensure the correct delivery and since all messages are broadcasted over the network, a filtering mechanism has been implemented in the *Queue* class. The filter checks the destination ID embedded in the message and only accepts the message if its ID matches that field. If the message is not intended for the model, the queue will simply drop the message. For this to happen, each coupled model (BS, RRH, and UE) is assigned a unique ID. Below is a detailed description of each of the messages that are sent over the network and what they are used for:

MSG: This is the parent class for all other message classes. This way, common fields and methods can be inherited by the children. In this class, we have defined a source and

destination ID which is essential for each message. Furthermore, we have defined a message type and assigned a numeric value for each type. Four methods have been defined for this type of message. The getter methods allow for the extraction of useful information (source ID, destination ID, and message type), and the setter method allows for changing the destination of a message. This is useful when certain messages should be forwarded. For example, when the BS receives the *CoMP Notification* message intended for its RRH, it can forward the message to its RRH by simply changing the destination of it using this method.

CRTL_COMMAND: This message is used by the BSs and RRHs to send system information to UEs. This message contains a source ID, a destination ID, the X coordinate of the BS/RRH, the Y coordinate of the BS/RRH, the frequency of the source BS/RRH, and the power of the source BS/RRH. The X and Y coordinates of the BS/RRH are extracted from this message using getter methods to enable the UE to calculate its distance from the BS/RRH. After calculating the distance, the UE calculates the received power from all access points (BSs and RRHs), using the standard formula which will be discussed in the next chapter. This allows the user to establish a connection with an access point that has the highest signal strength at the UE. The UE will then choose a serving station and will send a *CSI_FEEDBACK* message to it. It should be noted that if there are more than one access point with the highest received power, the UE will choose the one with the lowest ID. Table 2 shows the fields, their defined types, and their usages in a *CTRL_COMMAND* message.

Several methods are defined for this message type. In addition to the methods inherited from the *MSG* class, this class has a constructor that takes in the source and destination IDs

along with the XY coordinates of the BS and the BSs power and frequency. Getter methods have been defined to enable the UE to extract all this information which are required for the calculation of received power at the UE.

Table 2. Fields in a CTRL_COMMAND message

#	Field	Type	Usage
1	sourceID	int	Indicates the sender of the message.
2	destinationID	int	Indicates who the message is intended for. Only the model who's ID matches this field will accept the message.
3	BSx	int	Indicates the location of the BS. This is used by the recipient (UE) to calculate the received power.
4	BSy	int	Indicates the location of the BS. This is used by the recipient (UE) to calculate the received power.
5	BSf	int	Indicates the frequency of the BS. This is used by the recipient (UE) to calculate the received power.
6	BSp	int	Indicates the power of the BS. This is used by the recipient (UE) to calculate the received power.

CSI_FEEDBACK: This message is used to transfer the CSI from the UE to the BS/RRH. It contains the channel state information generated by the UE. Alongside inheriting the fields from *MSG*, this message a vector set of all the IDs of the BSs/RRHs which the UE receives a signal from, a vector set of all the received powers of the BSs/RRHs which the UE receives a signal from, and a flag that indicates if the UE is already in CoMP or not.

The power and id sets are used by the BS to determine from which BSs/RRHs can the UE receive a signal from and if any of these BSs/RRHs have the conditions to

collaboratively serve the UE after the UE switches to the CoMP mode. To do this, the serving BS compares its own received power by the UE to other received powers in the set. If the difference is less than 6dB, the BS will initiate CoMP. The BS will constantly check the differences in power to be able to detect any change. If the differences in powers goes beyond 6dB, a leave message will be sent to the UE forcing it to switch to non-CoMP mode. The flag indicating if the UE is in CoMP or not is used by the BS to stop sending a CoMP join message to the UE if the UE is already in CoMP and stop sending a CoMP leave message to a UE who is not currently in CoMP. Table 3 shows the fields, their defined types, and their usages in a *CSI_FEEBACK* message.

Table 3. Fields in a CSI_FEEDBACK message

#	Field	Type	Usage
1	sourceID	int	Indicates the ID of the sender of the message.
2	destinationID	int	Indicates who the message is intended for. Only the model who's ID matches this field will accept the message.
3	memberIDs	vector<int>	Includes the IDs of the CoMP members after the UE joins CoMP. This is used by the CCS (BS) keep track of the CoMP set for each UE. This will be used to update the members of the CoMP in case a change occurs.
4	memberPows	vector<int>	Includes the received powers by the UE. This set is used to determine if the UE qualifies to join CoMP and if so which BSs/RRHs will be in the CoMP set. The recipient (the UE's serving station) will compare these values to the maxRecPower to determine if CoMP should exist.
5	UEinCoMP	short	This field indicates if the UE is in CoMP or not. This will be used to by the serving station to prevent sending a leave message to UEs that are not in CoMP and a join message to UEs that are already in CoMP.

A constructor and different getter methods have been defined for this class. The constructor includes the IDs of the sender and receiver, two sets containing the IDs and powers of the BSs/RRHs whose signals are received by the UE, and a field indicating if the UE is in CoMP or not. For each of these fields, a getter method has been defined to enable the recipient to extract the data.

This message is the key message in CoMP and is sent periodically every 5ms. The fields of the message are updated if a change is detected. This allows for the recipient BS to be able to correctly schedule its resources based on the most recent status of the UE.

CSI_FEEDBACKFWD: The RRH uses this message to forward the CSI received from the UE to the BS it is connected to. This message has the same fields as the *CSI_FEEDBACK* message with an additional field which is the source UE ID. This field indicates from which UE the CSI message has originated from. This way, the BS receiving the *CSI_FEEDBACKFWD* will be able to know to which UE it should send a CoMP join message if the CoMP conditions are satisfied for that UE. This class is a child of the *CSI_FEEDBACK* class and inherits all of its methods. Furthermore, an additional method is defined to enable the receiver of this message to extract the UE id from which the message originated.

COMP_REQ: This message is sent from the UE's serving BS to other BSs to request the recipient to join CoMP and jointly serve the UE or to leave CoMP if conditions have changed and the UE is not in the CoMP region anymore. The message contains a request field which determines the purpose of the message (leaving CoMP or joining CoMP). Furthermore, the message includes the throughput of the BS's cell. This will be used in the election algorithm to elect a BS as the CCS. The algorithm for this election was discussed

in Chapter 4. Furthermore, the message includes the CCS ID (its own ID to start with, as discussed earlier in Chapter 4) which is mandatory in the election algorithm. The source UE ID for which this request is intended for and the IDs of the BSs and RRHs in the CoMP are also included in the message. If the recipient BS finds an RRH ID from its network in the IDs in the message, it will send the RRH a message notifying it of the change. This is because RRHs are only connected to the BS in their cell and do not have connections to all BSs across the network. The message sent to the RRH will be of type *COMP_RRHNOTIFICATION* which is discussed later. Table 4 shows the fields, their defined types, and their usages in a *COMP_REQ* message.

Table 4. Fields in a COMP_REQ message

#	Field	Type	Usage
1	sourceID	int	Indicates the ID of the sender of the message.
2	destinationID	int	Indicates who the message is intended for. Only the model who's ID matches this field will accept the message.
3	request	short	Indicates if the message is requesting resources from other BSs for a UE joining CoMP or if it is notifying other BSs of a UE leaving CoMP.
4	throughput	int	For a UE joining CoMP, this field will indicate the throughput of the sender. This is essential in electing a CCS.
5	ccsID	int	This includes the ID of the CCS. If this message is coming from the serving station of the UE, it will include that station's ID. Like the algorithm in which the serving station initially declares itself as the CCS.
6	sourceUEID	int	This is the ID of the original UE which the CSI originated from.
7	CoMP members	Vector <int>	This field contains the IDs of the eligible CoMP members. This will be used by the elected CCS to send a notification to all serving stations in CoMP notifying them of the new CCS.

Just like other classes, a constructor and getter methods have been declared. Alongside the source and destination IDs, the constructor includes a request field, which as mentioned earlier, is indicates if the request is for the UE to join or leave CoMP. Furthermore, the constructor has a filed for CCS ID, one for the original UE ID, and a field for a vector which shoes the CoMP members. Different getter methods have also been defined to enable the recipients of this message to extract the information.

COMP_REQ_G_R: After receiving the *COMP_REQ* message, the UE uses this message to send a response to the requesting BS. This message contains all fields of the *COMP_REQ* message with one additional field indicating if the original request has been granted or rejected. This message is sent to the requesting BS by the recipient of the *COMP_REQ* after it checks its resources and makes a decision to accept or reject the request. The getter methods defined for this message are like the ones for the *COMP_REQ* class.

COMP_NOTIFICATION: This message is sent to the BSs by the serving BS and is used in two cases: when a new CoMP set has been established and when CoMP does not exist anymore. For a new CoMP set, after the election algorithm has been concluded, the serving BS sends this message to other BSs in the CoMP set to notify them of the new CCS. For leaving CoMP, the elected CCS will send this message to the other BSs in CoMP notifying them that the CoMP does not exist anymore for a particular UE. This message contains the source and the destination like all other messages. Furthermore, it includes a notify flag which indicates if the message is intended for joining or leaving CoMP, the elected CCS ID, the throughput of the elected CCS, and the source UE ID. Table 5 shows the fields, their defined types, and their usages in a *COMP_NOTIFICAION* message.

Table 5. Fields in a COMP_NOTIFICATION message

#	Field	Type	Usage
1	sourceID	int	Indicates the ID of the sender of the message.
2	destinationID	int	Indicates who the message is intended for. Only the model who's ID matches this field will accept the message.
3	notify	short	Indicates if the notification is intended for a start of a CoMP session or ending a current CoMP session.
4	ccsID	int	This field includes the ID of the elected CCS.
5	throughput	int	This field includes the throughput of the elected CCS.
6	sourceUEID	int	This is the ID of the original UE which the CSI originated from.

Once a BS receives this message, it updates its records based on the message information. By using the *getNotify()* method declared in this message, the recipient can know if the message is intended for a new UE joining CoMP or an existing UE leaving CoMP. Furthermore, using the other getter methods defined, the recipient will know for which UE is this message for.

COMP_RRHNOTIFICATION: This message was configured as a replacement for *COMP_NOTIFICATION* in HetNets. Since HetNets include RRHs that are connected only to their BS, a new message was configured to allow the message delivery to RRHs. This message includes the same fields as the *COMP_NOTIFICATION* message. It is sent from the CCS to the BS to notify it of the new change. The difference is that once a BS received this message, it will not only update its information, but will also forward this message to its RRHs which were in the CoMP set to allow them to update their information. Moreover, this message has the same constructor and getter methods as the *COMP_NOTIFICATION*.

COMP_COMMAND: This message is sent to the UE by the BS. Other than the basic source and destination fields, this message contains a command value which dictates to the UE if it has to join or leave CoMP. Once the UE receives this message, it changes its state based on this value (i.e. the UE switches to CoMP from no-CoMP or vice versa). Furthermore, this message contains the elected CCS ID to which the UE should send the CSI feedback message to after joining CoMP. If the UE is leaving CoMP, this field will contain 0 forcing the UE to calculate the received powers and find its serving BS. Table 6 shows the fields, their defined types, and their usages in a *COMP_COMMAND* message.

Table 6. Fields in a COMP_COMMAND message

#	Field	Type	Usage
1	sourceID	int	Indicates the ID of the sender of the message.
2	destinationID	int	Indicates who the message is intended for. Only the model who's ID matches this field will accept the message.
3	command	short	Indicates if the notification is intended for a joining or leaving CoMP.
4	ccsID	int	This field includes the ID of the elected CCS.

Like all other messages, this message also contains the getter methods for extracting the information.

COMP_ACK: This message is an acknowledgement message sent from the UE to the CCS indicating that the UE has received the new information and has switched to CoMP mode. This message has a simple structure: alongside the source and the destination IDs, it includes a flag set to true indicating the information has been received. This is a simple message to ensure the delivery of the *COMP_COMMAND*. Once the UE receives the *COMP_COMMAND* message, it will reply using this message to confirm the receipt.

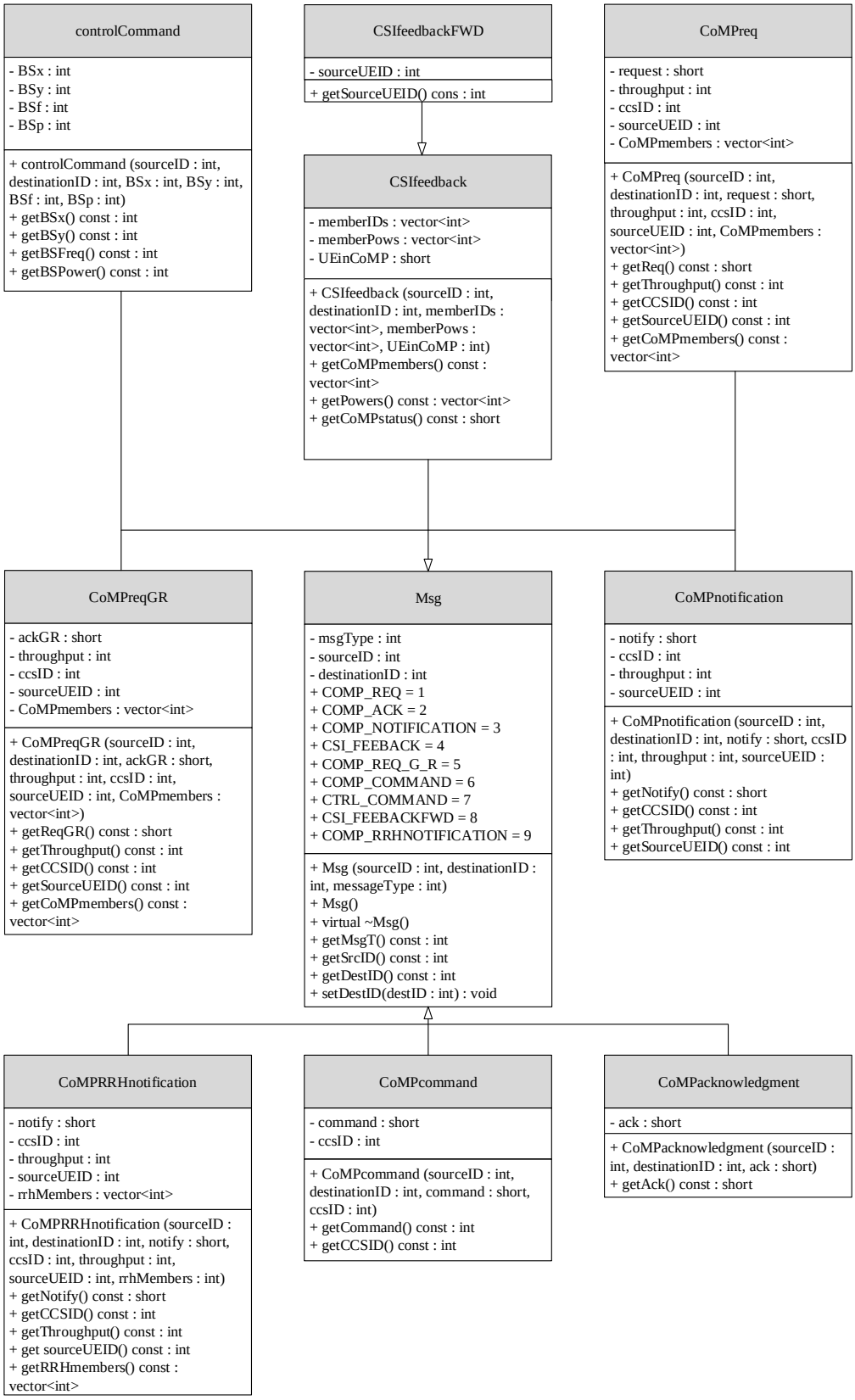


Figure 18. Message class diagram

To summarize, a class diagram was drawn for the message class. Figure 18 shows a class diagram for the message class, the relations between the classes, and the fields and methods defined for each class.

In order to be able to understand the structure in more detail, let us consider a hypothetical scenario of 1 UE, 2 RRHs, and 3 BSs. The UE is receiving signals from all stations and the CoMP cooperation set includes all 5 stations. We assume that the UE is closer to an RRH making the RRH its serving station before CoMP. Furthermore, we assume BS3 is elected as the CCS. Figure 19 shows the scenario.

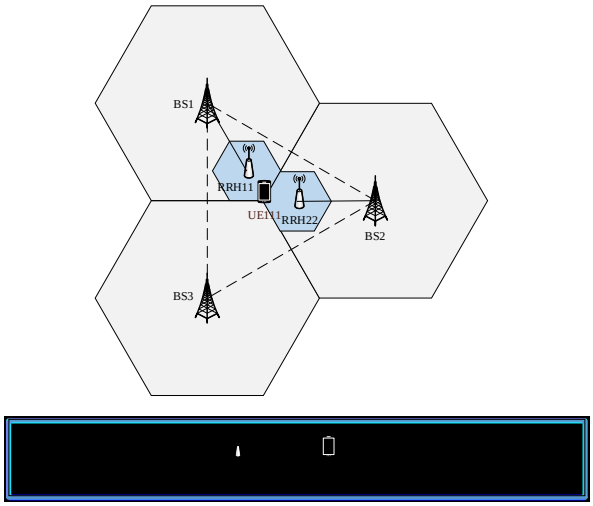


Figure 19. A scenario with 1 UE, 2 RRHs, and 3 BSs

To understand the message exchange sequence in the simulation software better, Figure 20 shows a sequence diagram that defines what happens in terms of message exchange in the simulation software.

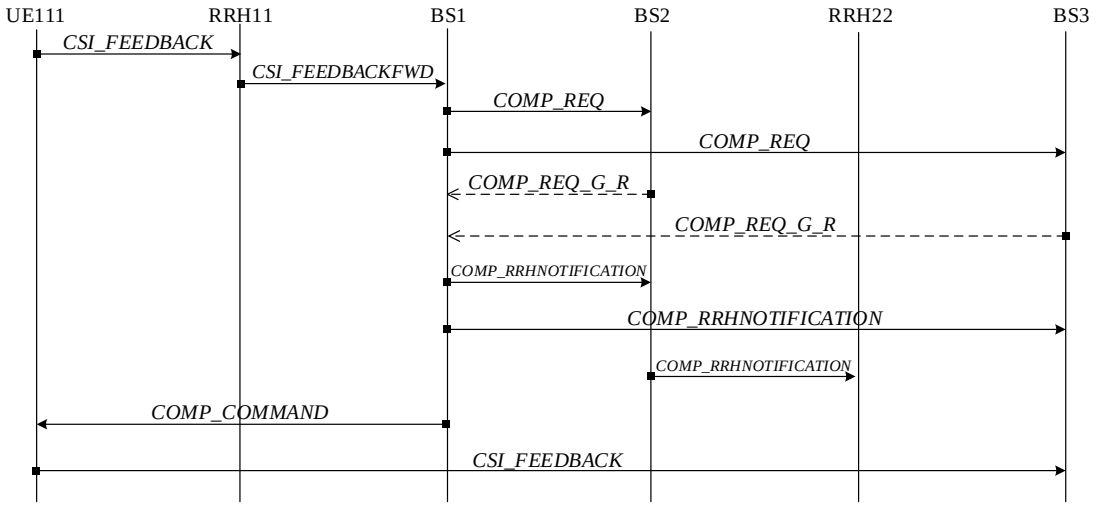


Figure 20. Sequence diagram demonstrating the messages exchanged when a UE joins CoMP

As observed in Figure 20, UE11 sends the *CSI Feedback* message to its serving RRH (RRH11). The RRH then forwards the message in the form of a *CSI Feedback Forward* message to its BS (BS1). BS1 then initiates CoMP by sending a *CoMP Request* message to BS2 and BS3. After checking their resources, BS2 and BS3 send back a *CoMP Request Grant/Reject* message to the requesting BS (BS1). At this point, having received the grants, BS1 sends a *CoMP RRH Notification* message to BS2 and BS3 and a *CoMP Command* message to UE111. The received *CoMP RRH Notification* is then forwarded to RRH22 by BS2. Once the *CoMP Command* is received by the UE, the UE switches to CoMP mode and starts sending the *CSI Feedback* message directly to the elected CCS (BS3).

In the next chapter, we will present some simulation scenarios which were used to run multiple experiments. Furthermore, we will present and analyze the results in detail.

Chapter 6

Simulation Scenarios and Results

To be able to compare the three different CoMP architectures and assess the potential of the DCEC architecture, different experiments were conducted. The simulation scenarios studied can be categorized into two general categories: homogenous scenarios and heterogeneous scenarios. Here we will present each scenario, talk about it in detail and show and analyze the results.

6.1. Homogenous Networks

To initially test the proposed architecture before modeling any real-world scenarios, a simple model was developed and tested. In this model, 3 cells, 3 BSs (one per each cell), and only 3 UEs (one per each cell) were considered. This allowed us to easily track the messages around the network and troubleshoot any possible issues. The simulation scenario is shown in Figure 21. In this scenario, the UE generates the CSI feedback based on the signal strength received from cooperating BSs and sends it to the BS every 5 ms [17] [29]. The recipient BS executes the algorithm discussed in Chapter 4 to calculate a CoMP cooperating set and to elect a CCS. The BSs are connected to one another via X2 links and the UEs can communicate with all BSs via air links.

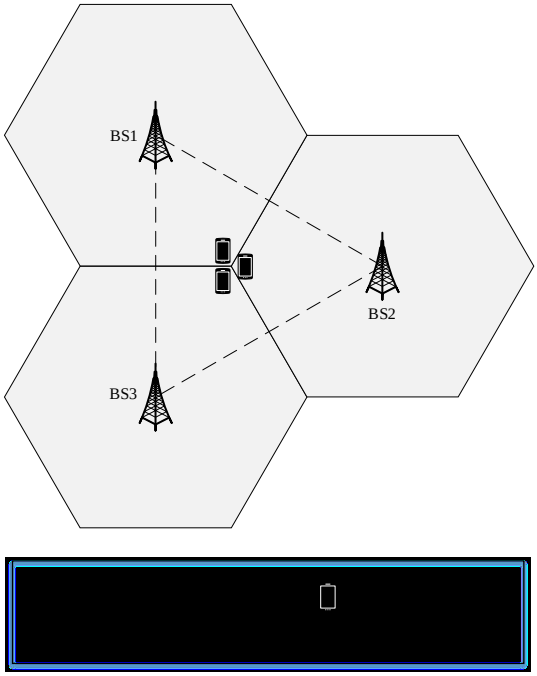


Figure 21. Simple simulation scenario used for initial testing

This scenario was tested for all three architectures (*DCEC*, *Centralized*, and *Distributed*) to make sure the developed models perform correctly. Table 7 shows the initial conditions for the simulation.

Table 7. Initial simulation assumptions for homogenous networks

Parameters	**Values**
Number of BSs	3
Number of UEs	3
Frequency	900 MHz
BS transmit power	46 dBm
Cell radius	500 m
Antenna gain	12 dBi (BS) and 0dBi (UE)
MCL	70 dB
LogF	10 dB
Cell throughput	Uniform: randomly generated
CSI feedback periodicity	5 ms
CoMP threshold	6 dB
Traffic model	Full buffer

We have chosen our cell radius and antenna gain parameters to align with the specifications outlined in LTE release 12. Furthermore, as [64] suggests, the carrier

frequency has been set to 900 MHz, the cell radius has been set to 500 m, and the antenna gain has been set to 12 dBi for the BS and 0 dBi for the UE. Based on [17], [29] the CSI feedback frequency has been set to 5 ms. In our simulations, cells are considered as macro cells in an urban area. A typical transmission power for a BS in a macro cell is normally between 43 dBm to 48 dBm and has been set to 46dBm as suggested by [65]. The received power at the UE is calculated based on the following formula [66]:

$$P_r = P_t - MAX(L_{path} - G_t - G_r, MCL)$$

in which P_r is the received signal power, P_t is the transmitted signal power of the BS, G_t is the transmitting antenna gain, G_r is the receiver antenna gain, and L_{path} is the path loss. Furthermore, in this formula, the Minimum Coupling Loss (MCL) is set to 70 dB, the BS antenna gain is set to 12 dBi, and the UE antenna gain is considered to be 0 dBi. L_{path} is the pathloss and is calculated based on the propagation model for urban areas [66]. Based on [37], we considered the CoMP threshold to be 6dB.

To be able to analyze the advantages and disadvantages of the DCEC architecture over distributed and centralized CoMP, we simulated the algorithms using the scenario above as well. To simulate the distributed CoMP architecture, we assume that the BSs are synchronized. To be able to evaluate the performance of the three architectures, the number of control packets in the network was considered as a metric. As talked about previously, the number of control packets in a network can directly affect data rates. Therefore, one way to improve data rates in a network is to reduce the total number of control packets in a network [67] [68]. Figure 22 shows the accumulative number of control packets in the network for the three different CoMP control architectures. The simulation was running for 200 ms and all three UEs were in CoMP with all 3 BSs. To be able to clearly compare

the three architectures, the figure has been zoomed in to include only 50 ms of the simulation run.

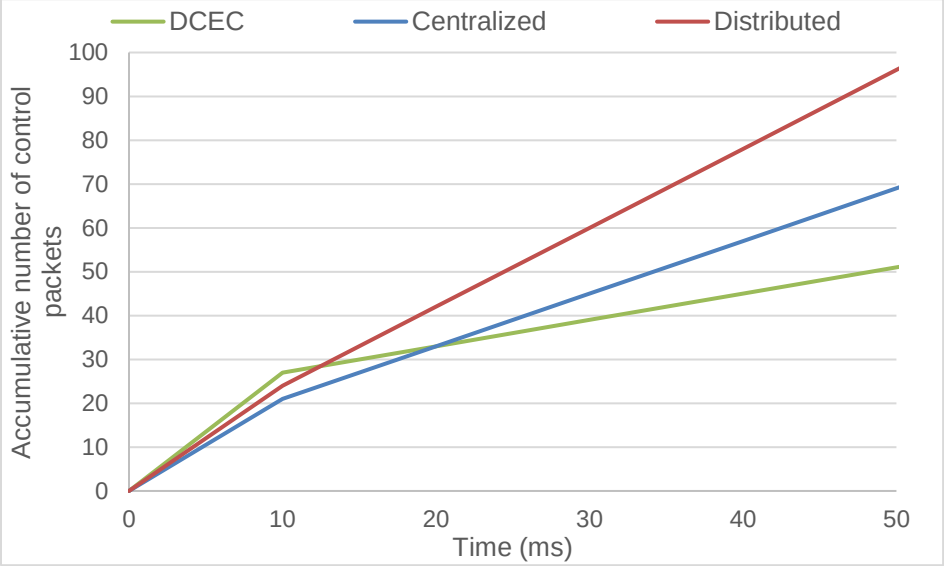


Figure 22. Comparison of DCEC, distributed, and centralized architectures in CoMP based on accumulative number of control packets over time for 3 BSs and 3 UEs

In this case, the DCEC architecture performs worse than the other two architectures in the beginning, but then after some time, the DCEC outperforms the other two architectures. This is because the DCEC architecture requires the execution of an election algorithm in the beginning to elect a CCS. This results in an increase in the number of messages travelling around the network. Once the CCS is elected, the number of control messages in the network decreases. This shows that given enough time to recover, the DCEC architecture outperforms the other two architectures.

In order to evaluate the effect of the load imposed on the network by the election algorithm in DCEC further, another simulation scenario was setup. In this scenario, 10 UEs are present in the CoMP area. Furthermore, at 80ms, 2 UEs join the CoMP set, and at 120 ms and 130 ms 3 more UEs and 1 more UE join the CoMP set respectively. Other

simulation assumptions are the same as the previous scenario and outlined in Table 7. The results of this scenario are shown in Figure 23.

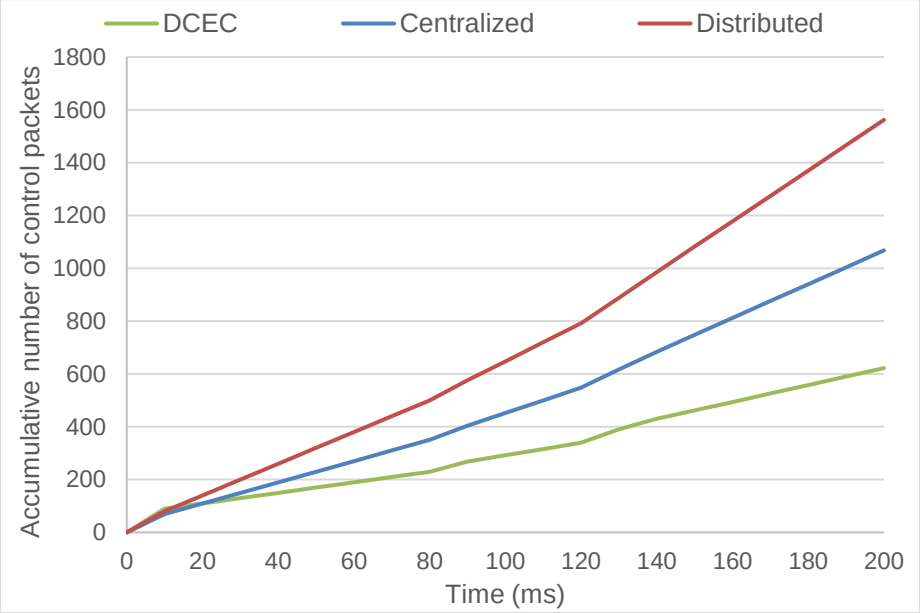


Figure 23. DCEC, distributed, and centralized architectures based on the number of control packets in the network (3 BS and multiple UE)

As it can be seen in the above figures, the DCEC architecture shows sensitivity to change. As explained earlier, the election algorithm in the DCEC must run in two cases: if a new user joins CoMP and if the throughput of a cell changes. To test the limits of the DCEC architecture, we set up an experiment in which the throughput of the cells was set to change periodically. This forces the DCEC architecture to re-run the election algorithm. For this, three simulation scenarios were set up with 10 UEs. In the first case, we assume that the cell throughput is constant, that is, the CCS does not change throughout the simulation. In the second and third cases, the cell throughput is set to change every 1s and every 100ms respectively. This allows for testing the effect of the election algorithm on the architecture. Figure 24 shows the results of these simulation scenarios. It should be

noted that the distributed and centralized approaches are not affected by the cell throughput change.

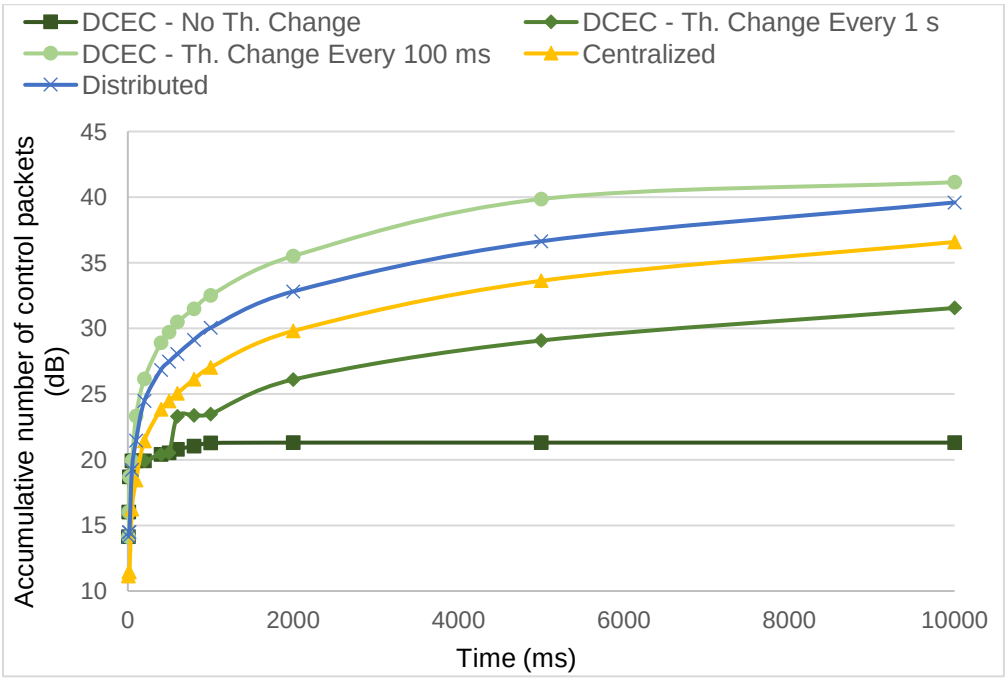


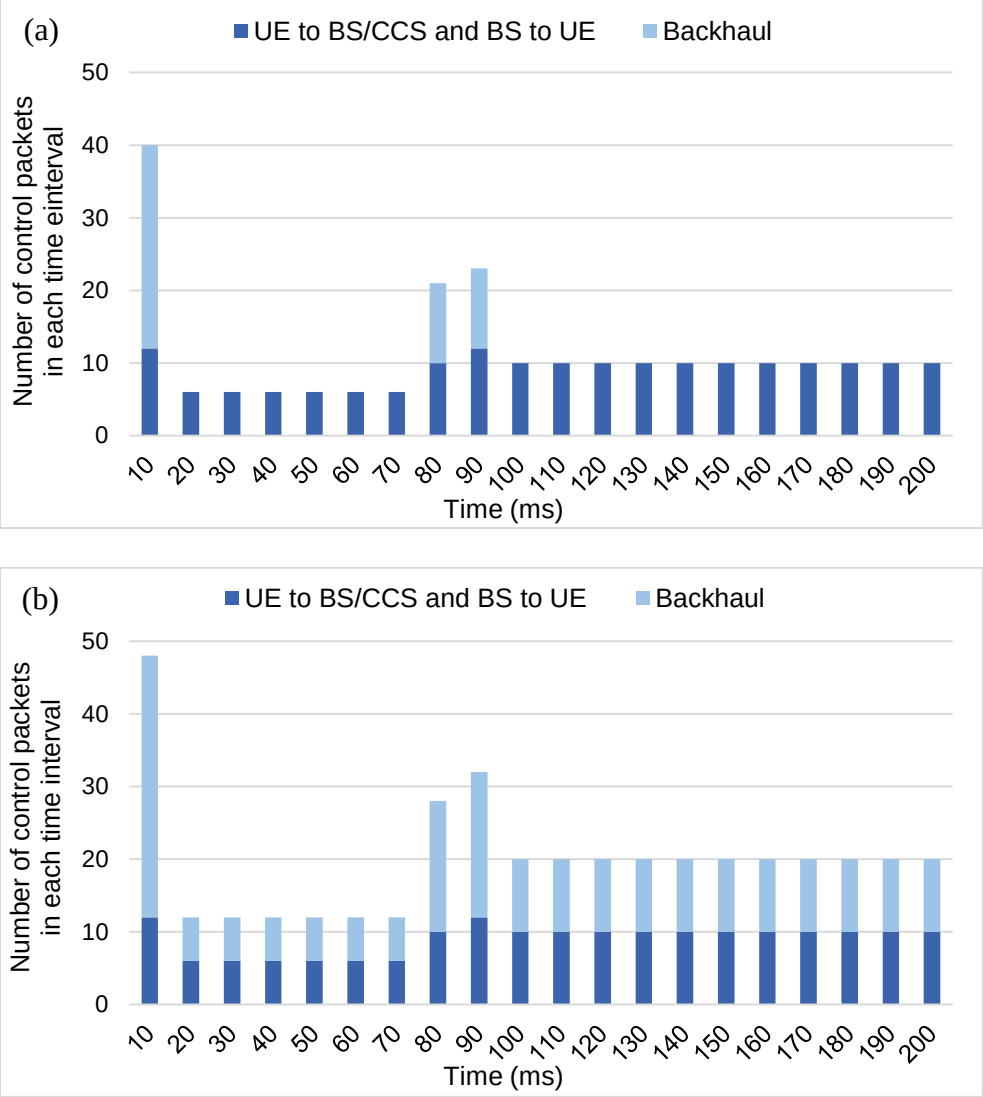
Figure 24. Accumulative control packets for DCEC without CCS change, DCEC with CCS change every 1s and 100ms, centralized, and distributed architectures

This simulation shows the fact that DCEC is sensitive to throughput change and may perform worse than the other two architectures. Although this can be thought as a shortcoming of the proposed DCEC architecture, in practice, the CCS change does not occur that frequently for most of the UEs since the maximum movement speed of a UE suggested by the 3GPP release 11 for CoMP deployment is 3km/h [69].

According to the results of the simulation, as seen in Figure 22, Figure 23, and Figure 24, the DCEC architecture has the potential to reduce the number of feedback overhead within the CoMP network compared to the other two conventional approaches.

To analyze the effect of the architectures on the backhaul, the control packets travelling around the network were categorized into two categories: BS to BS/CU and UE to BS/CCS. The first category includes messages traveling over the backhaul while the second category

contains messages travelled over air links. It should be noted that 10 UEs were present in the CoMP area. Six of which were present from the beginning of the simulation, two join CoMP at 70 ms and two more UEs join CoMP at 80 ms. Figure 25 shows the non-accumulative number of packets in 10 ms time intervals for (a) DCEC (b) centralized and (c) distributed architectures.



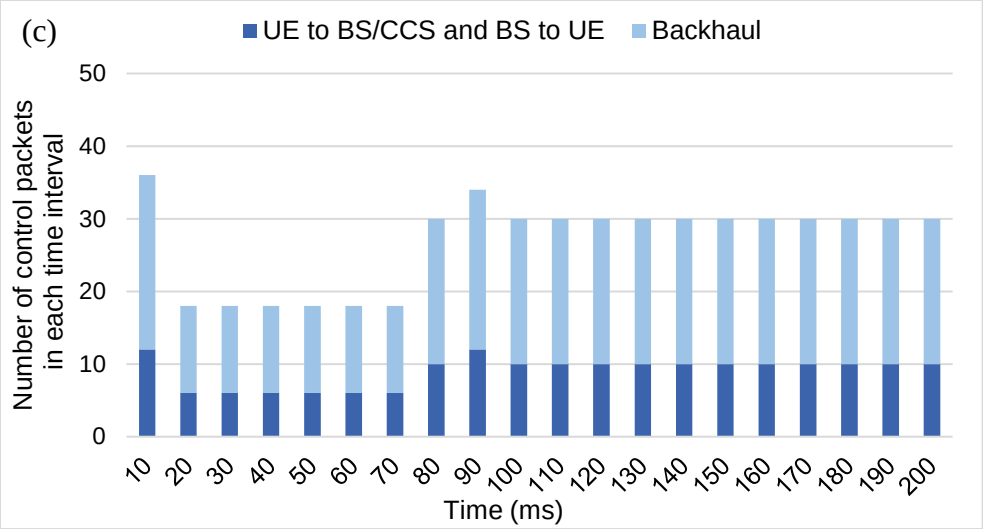


Figure 25. Number of control packets at different time intervals for (a) DCEC, (b) centralized, and (c) distributed architectures: packets over the backhaul/air links

Once again, it can be concluded from the above figure that the DCEC algorithm initially performs worse than the distributed architecture, but better that the centralized approach. Once CoMP is established, the DCEC will not need any additional messages travelling through the backhaul. This is because the UE sends the CSI feedback directly to the CCS.

To be able to analyze the effect of the number of users, multiple simulation scenarios with varying number of users (50, 100, and 200) in CoMP were simulated. All users have been placed in the CoMP region since the control architectures differ only in handling CoMP users. In this case, the position of the UEs have been picked at random from all possible CoMP positions. To omit abnormalities, 10 simulation runs were conducted for each simulation scenario and the margin of error was calculated based on a 95% confidence interval. The results were obtained and analyzed for comparison of delay and number of control messages. To mimic the real world, UEs were set to start transmitting the CSI feedback based on a Poisson distribution within a 12-hour period (from 6 AM to 6 PM)

with the peak request rate being located at 10 AM [70]. Figure 26 shows the distribution of the average request start time for 10 runs for (a) 50 UEs, (b) 100 UEs, and (c) 200 UEs.

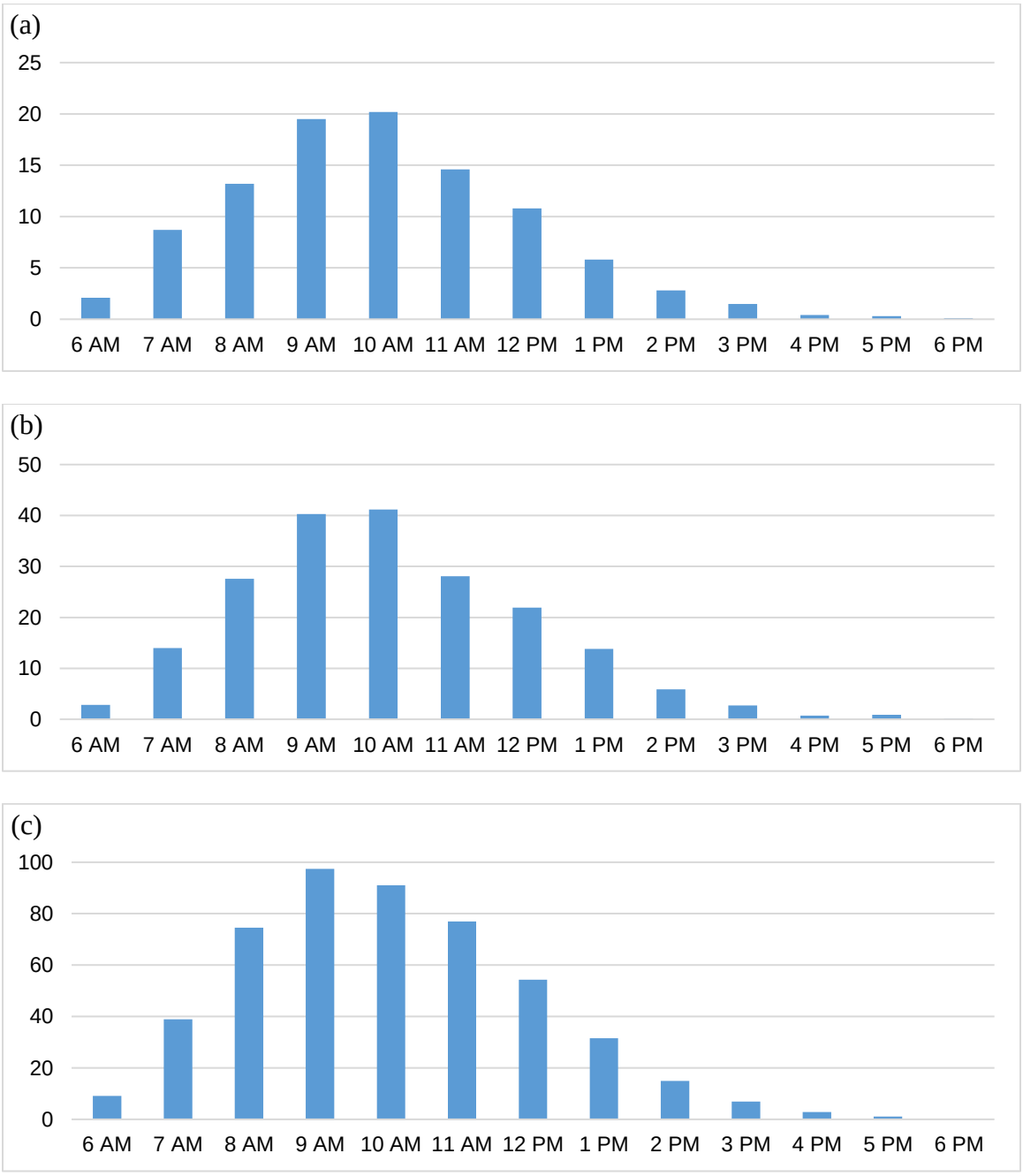


Figure 26. Average distribution of the request start time over 10 simulation runs for (a) 100, (b) 200, and (c) 500 UEs

To be able to mimic a real-world scenario, an area of 19 cells was considered. Table 8 shows the simulation assumptions and Figure 27 shows the simulation scenario.

Table 8. Simulation assumptions for a 19 cells homogenous network

Parameters	**Values**
Number of BSs	19
Number of UEs	50, 100, and 200
UE distribution	Uniform: randomly generated in the CoMP area
UE arrival	Poisson: 6 AM to 6 PM with peak at 10 AM
Frequency	900 MHz
BS transmit power	43 dBm
Cell radius	500 m
Antenna gain	12 dBi (BS) and 0dBi (UE)
MCL	70 dB
LogF	10 dB
Cell throughput	Uniform: 1 to 6
CSI feedback periodicity	5 ms
CoMP threshold	6 dB
Traffic model	Full buffer
Simulation time	12 hours

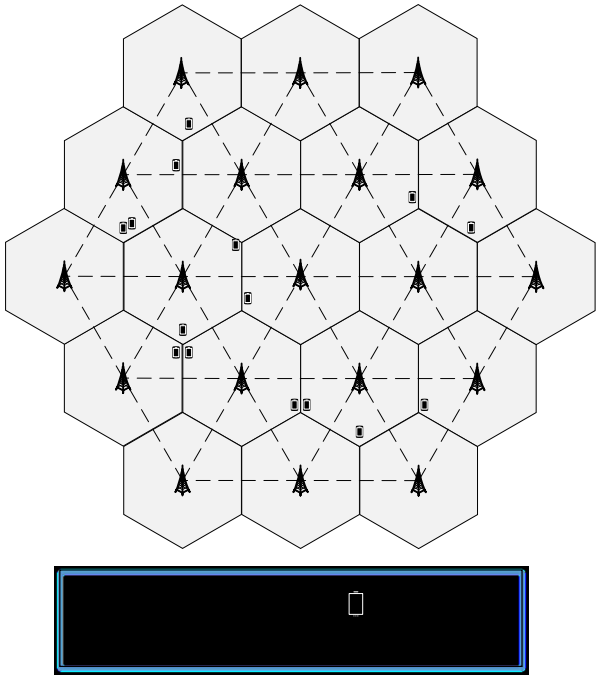


Figure 27. Sample scenario with 19 cells

Using the above scenario and as stated in Table 8, three separate experiments were conducted.

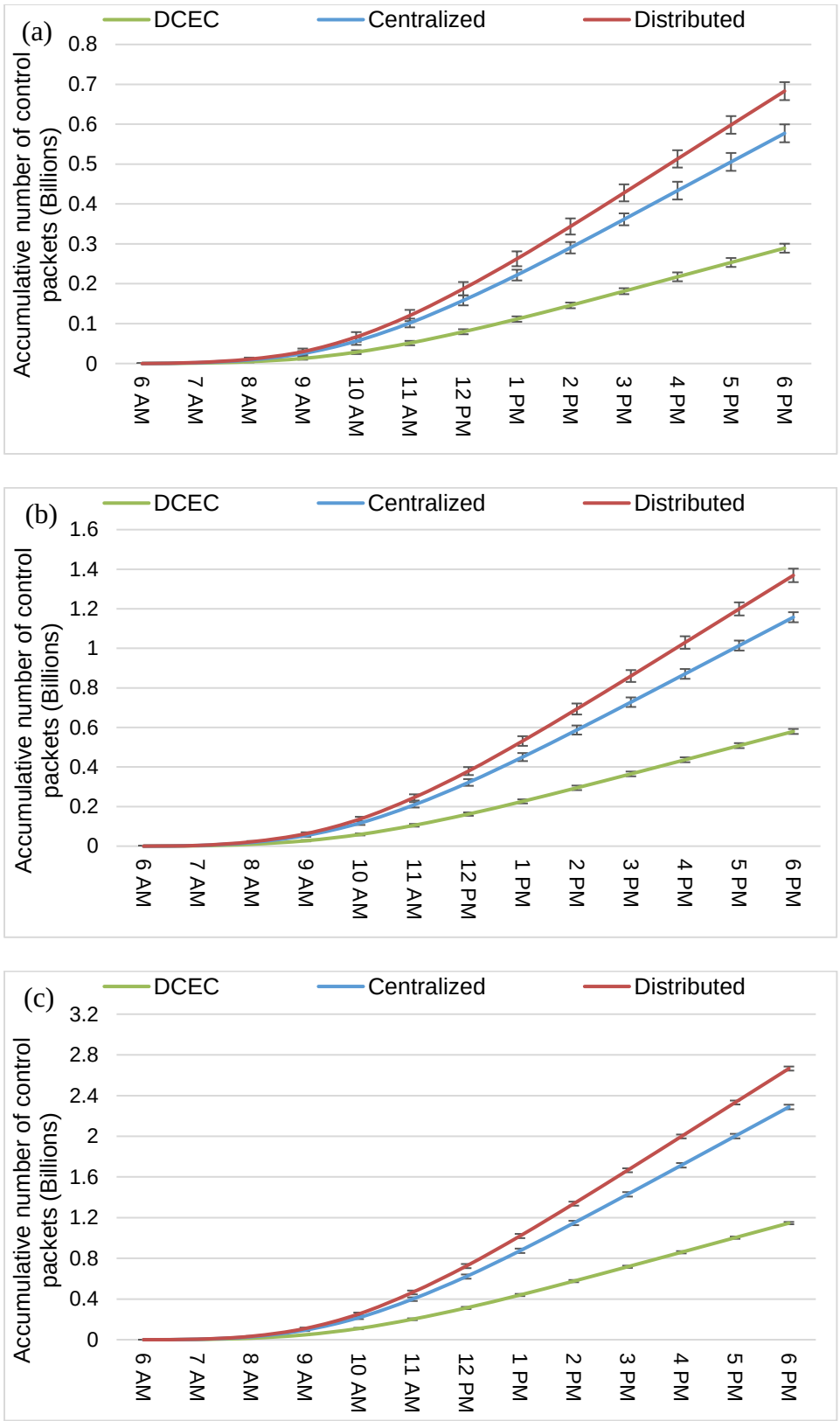


Figure 28. Accumulative number of control packets in a 12-hour period for (a) 50 UEs, (b) 100 UEs, and (c) 200 UEs

The first one with 50 UEs, second one with 100 UEs, and third one with 200UEs randomly spread across the CoMP areas. As mentioned previously, in order to ensure preciseness, 10 separate runs for each experiment were done and the results were calculated and graphed with 95% confidence interval. The results were analyzed for comparison of delay and control messages. Figure 28 shows the accumulative number of control packets over a 12-hour period for (a) 50 UEs, (b) 100 UEs, and (c) 200 UEs in all three control architectures (DCEC, Centralized, and Distributed). From the above figure, it can be concluded that DCEC requires far less control packets to be sent around the network.

In order to be able to see the effect of the election algorithm, the number of non-accumulative number of packets were also measured. By doing so, the effect of the election algorithm in the DCEC architecture can be clearly seen. Figure 29 shows the number of non-accumulative control packets in a 12-hour period for (a) 50 UEs, (b) 100 UEs, and (c) 200 UEs. As it can be seen, the DCEC algorithm performs worse in the initial hours of the day, but given enough time to recover, it outperforms the other two control architectures. This agrees with the previous results, which were obtained under different circumstances.

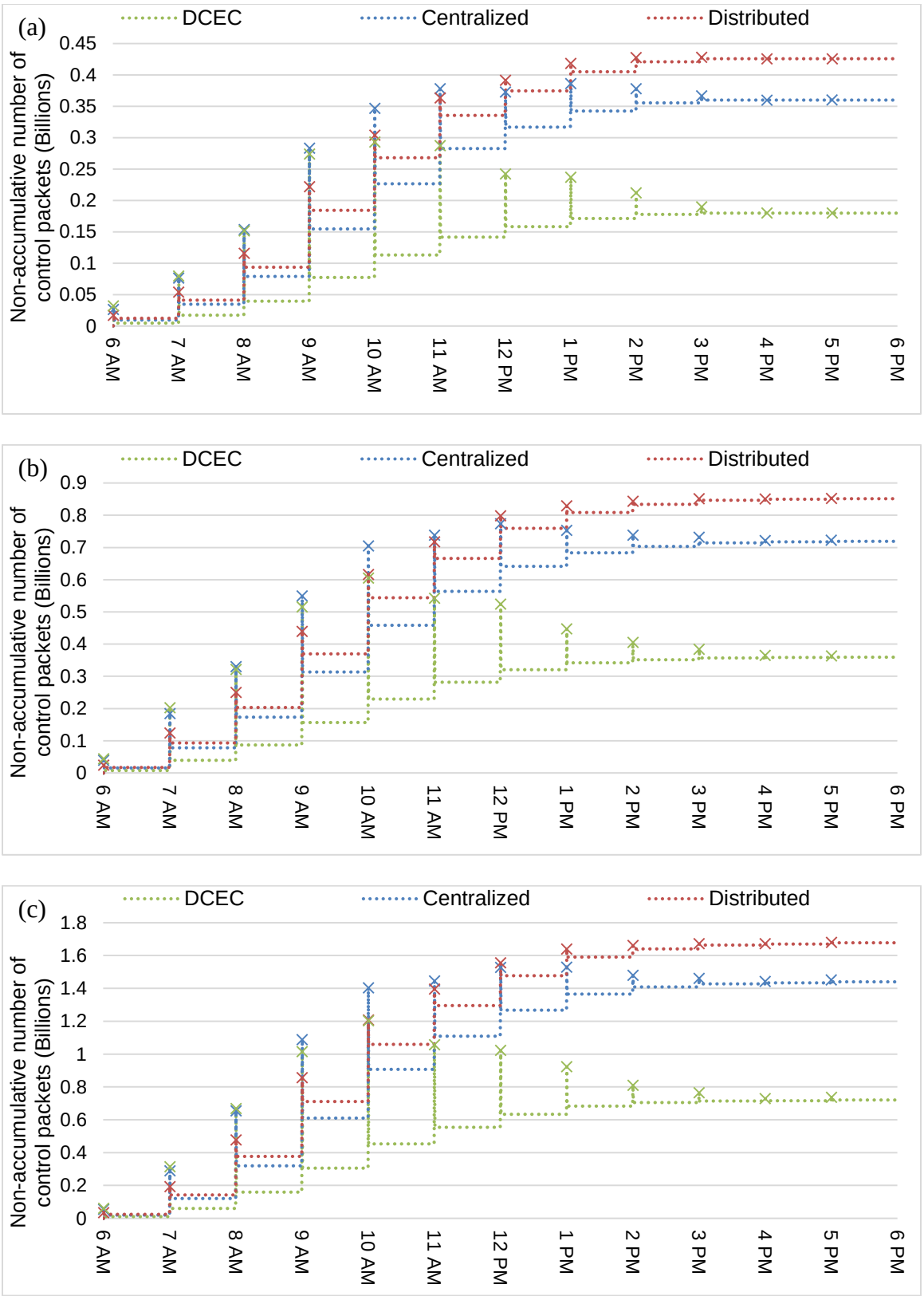
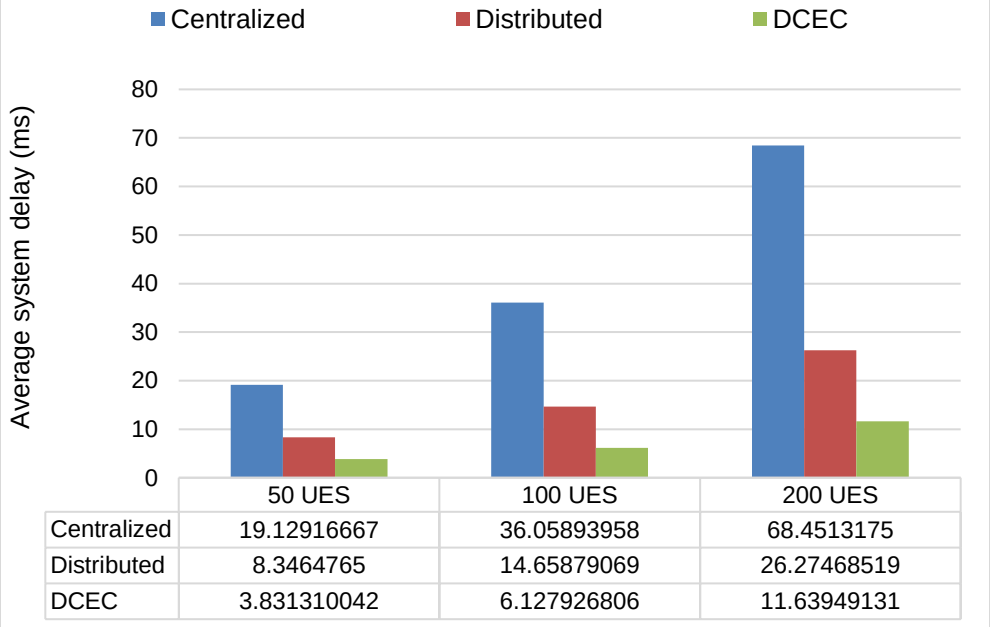


Figure 29. Non-accumulative number of control packets in a 12-hour period for (a) 50 UEs, (b) 100 UEs, and (c) 200 UEs

To analyze the effect of DCEC further, the delay was measured for every CSI feedback message sent by the UE. The delay is defined as the time it takes from when the message is sent by the UE to when the CSI feedback message is processed by the BS. To understand the results and to find the average system delay, the average of all the measurements were calculated. Figure 30 shows the average delay of the entire system for different number of UEs.

The figure shows that the DCEC approach imposes the least amount of delay on the network while the centralized approach imposes the most delay. It can be confirmed once again that the DCEC approach is less sensitive to the increase in the number of UEs in the network. This allows for the DCEC algorithm to be a very good fit for both crowded and uncrowded areas.



	50 UES	100 UES	200 UES
Centralized	19.12916667	36.05893958	68.4513175
Distributed	8.3464765	14.65879069	26.27468519
DCEC	3.831310042	6.127926806	11.63949131

Figure 30. Average system delay for 50, 100, and 200 UEs

6.2. Heterogeneous Networks

After comparing the three algorithms (DCEC, Centralized, and Distributed) for homogenous networks and concluding that DCEC has a potential to boost the data rates by decreasing the number of control messages in the network, we decided to extend the work to HetNets and study the effects of DEC on them.

Table 9. Initial simulation assumptions for heterogeneous networks

Parameters	**Values**
Number of BSs	3
Number of RRHs	3
Number of UEs	3
Frequency	900 MHz
BS transmit power	46 dBm
Cell radius	Macro: 500 m; Micro: 100 m
Antenna gain	12 dBi (BS), 5dBi (RRH), and 0dBi (UE)
MCL	70 dB
LogF	10 dB
Cell throughput	Uniform: randomly generated
CSI feedback periodicity	5 ms
CoMP threshold	6 dB
Traffic model	Full buffer

Just like the homogenous case, multiple scenarios were simulated. Table 9 shows the simulation assumptions and Figure 31 shows the simulation scenario used to run the experiments.

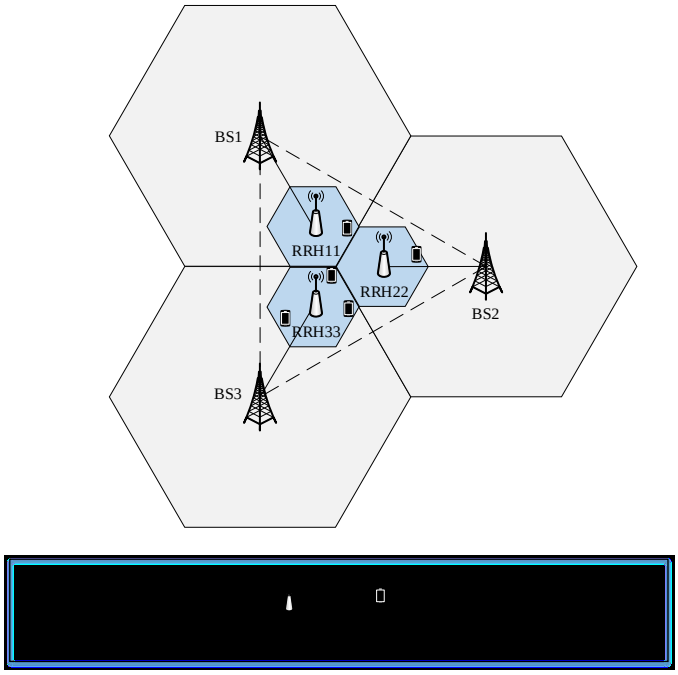


Figure 31. Simulation scenario for heterogeneous networks

To obtain some initial results, we simulated all three architectures with 50, 100, 150, and 200 UEs for 700 ms. The total number of control packets travelling around the network was measured as a metric. The results can be seen in Figure 32. To make the figure more readable, the difference in the number of packets for each architecture has been marked on the graph.

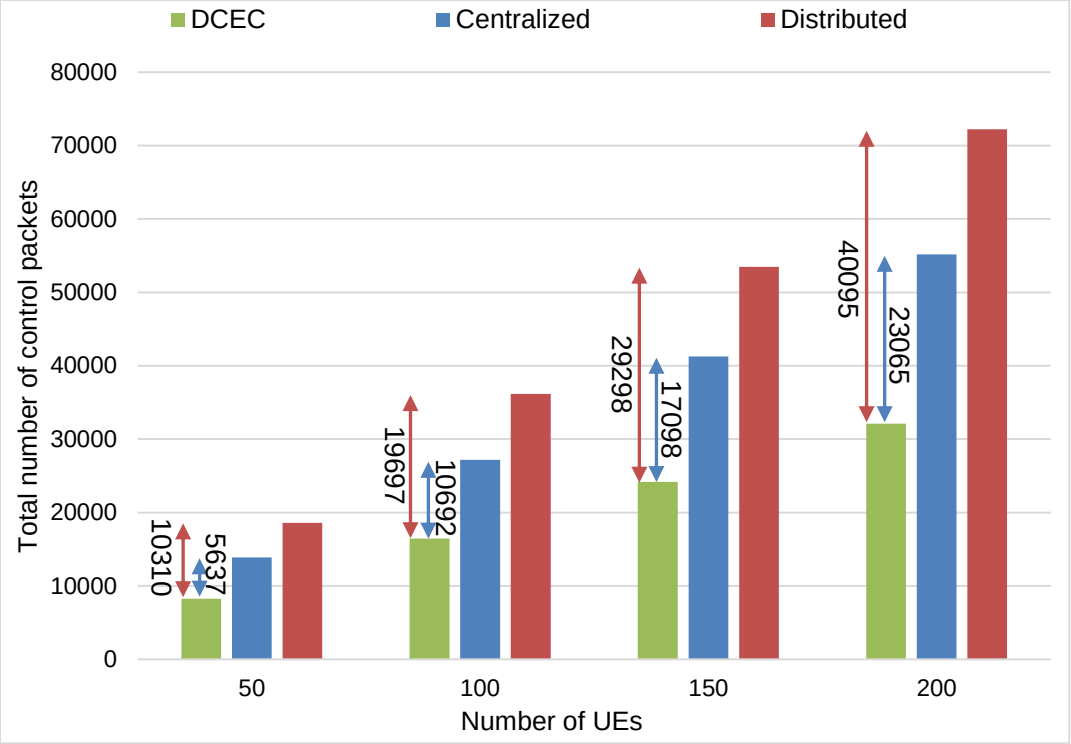


Figure 32. Total number of control packets travelling around the network

As we can see, the difference in the number of control packets increases as the number of UEs increase. This means that DCEC-HetNet is less sensitive to change in the number of users (given enough time to recover) compared to the other two architectures which makes it a good fit for use in crowded areas.

To test the effect of new UEs joining CoMP in all three architectures, a scenario was set up in which 100 UEs are present in the CoMP area with 3 BSs and 3 RRHs. In this scenario, some new UEs join CoMP at 120 ms and at 200 ms all the UEs leave CoMP. Finally, all the UEs join CoMP again at 400 ms. This scenario was set up to test the effect of the election algorithm as well as UEs leaving CoMP on the total number of control messages. Table 10 shows the simulation parameters for this scenario and Figure 33 shows the results.

Table 10. Simulation assumptions for a HetNet

Parameters	**Values**
Number of BSs	3
Number of RRHs	3
Number of UEs	100
UE distribution	Uniform: randomly generated in the CoMP area
UE arrival	All UEs in CoMP at the beginning New UEs join CoMP at 120 ms All UEs leave CoMP at 200 ms All UEs join CoMP at 400 ms
Frequency	900 MHz
BS transmit power	43 dBm
Cell radius	Macro: 500 m Micro : 100 m
Antenna gain	12 dBi (BS), 5dBi (RRH), and 0dBi (UE)
MCL	70 dB
LogF	10 dB
Cell throughput	Uniform: 1 to 6
CSI feedback periodicity	5 ms
CoMP threshold	6 dB
Traffic model	Full buffer
Simulation time	500 ms

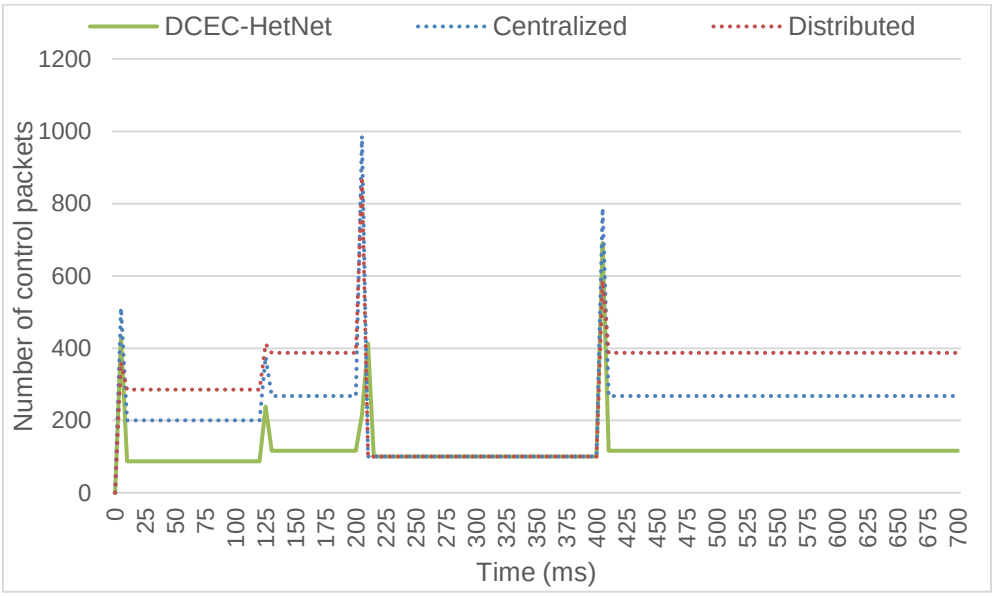


Figure 33. Non-accumulative number of control packets in a 500 ms period for 100 UEs

As we can see, when new UEs join CoMP, the DCEC-HetNet architecture outperforms the other 2 CoMP architectures. After CoMP is established and in the existence of CoMP, the DCEC-HetNet requires the least amount of control messages to travel around the network. As mentioned previously, in the initial phase, and when a CCS is being elected, DCEC-HetNet requires some additional messages to travel around the network. After the completion of this transient phase, the DCEC-HetNet performs better than the other two architectures. Furthermore, as seen in the figure above, in the time interval from 200 ms to 400 ms when no UE is in CoMP, all architectures perform the same.

To further test the architecture, another simulation scenario was tested. In this scenario, 200 UEs were present. Furthermore, several UEs join CoMP at 120 ms. The simulation was running for 200 ms. Figure 34 shows the results for the (a) DCEC, (b) centralized, and (c) distributed architectures.

With all said and by considering the results of the simulations, it can be seen that DCEC-HetNet improves the network performance by reducing the number of control packets going around the network. This reduction in CSI feedback overhead and latency can eventually result in improved throughputs and higher data rates [67] [68].

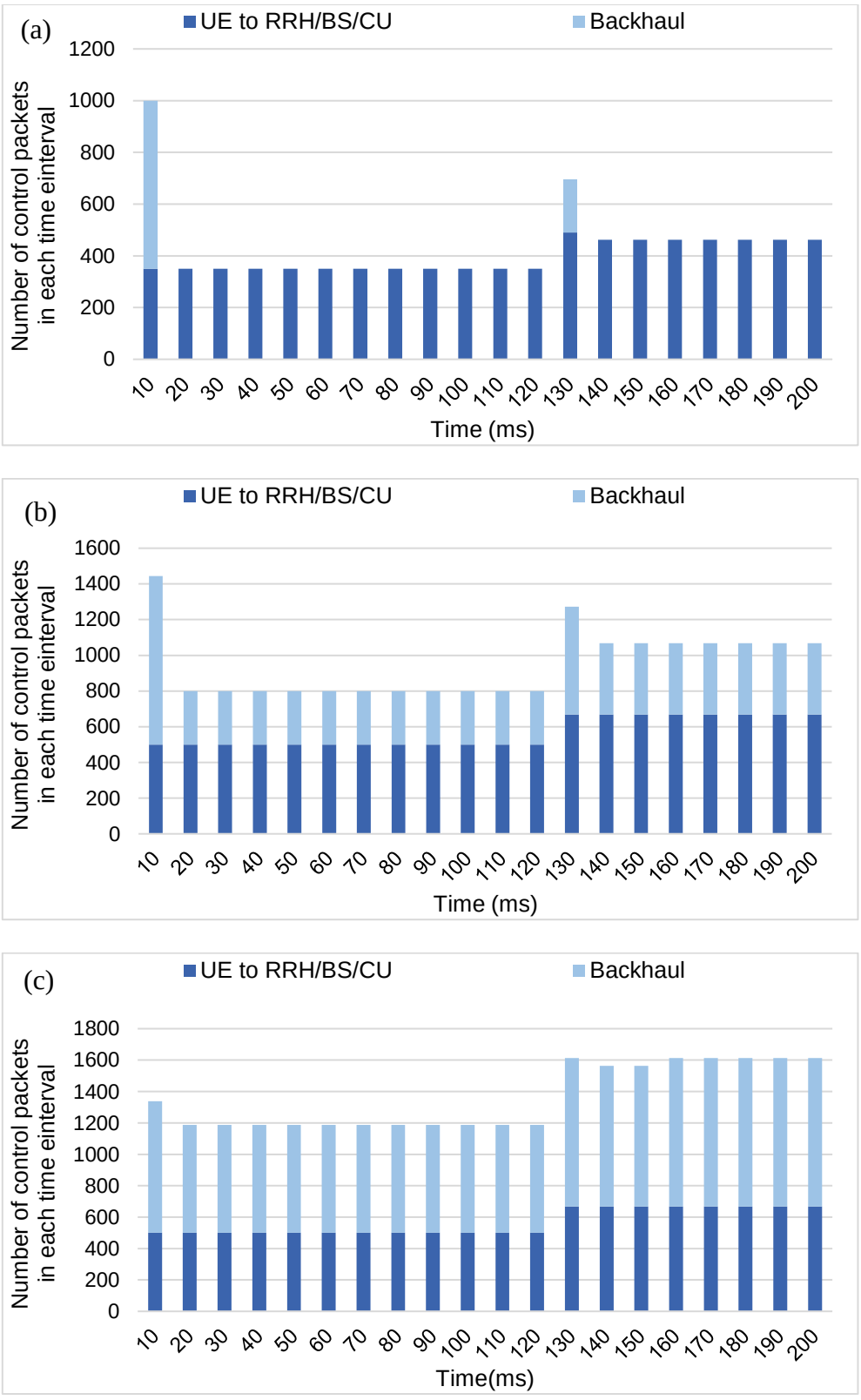


Figure 34. Number of control packets at different time intervals for (a) DCEC-HetNet, (b) centralized, and (c) distributed architectures: packets over the backhaul/air links

Chapter 7

Conclusion and Future Work

With the growing number of mobile users, the demand for service is ever more increasing. This demand has imposed new challenges on the cellular network. To cope with this demand, LTE-A has been forced to move to the idea of frequency reuse. Although this solution has solved some of the challenges, it has brought up new challenges, especially for cell edge users. The users on the edge of the cell experience low data rates due to weak signal reception and signal interference from their neighboring cell. In order to solve this issue, 3GPP has proposed the idea of CoMP allowing the BSs to jointly serve a UE which is located on the cell edge. Although this approach contributes to a better user experience, it imposes extra load on the network via the extra control messages required.

In this thesis, to solve this issue and to minimize the effect CoMP has on the network, we proposed DCEC, a CoMP control architecture aiming at reducing the number of control messages in the network. This algorithm uses an elected coordination station as the serving BS for the UEs. After the UEs join CoMP, an election algorithm will execute to choose the best serving station for that UE. Once a station is elected, the UE sends the CSI feedback message to that BS only. In this case, there is no need for extra messages to travel over the backhaul. This was done in collaboration with the ARS-Lab and Ericsson Canada.

Moreover, to evaluate the performance of the proposed algorithm in different scenarios and compare the results with the conventional two approaches (centralized and distributed), we developed a simulator based on DEVS. Several models were developed to enable the simulator mimic the real world. The simulator was developed in such a way that it is

flexible and easy to use for different scenarios with different initial conditions. Furthermore, being specific to CoMP, the simulator is unique and there is no other simulator specifically designed for simulating CoMP.

Several simulation scenarios with varying number of BSs, RRHs, and UEs, were evaluated. This allowed for careful examination of the performance of DCEC. To find the bottle neck of the proposed algorithm, we forced DCEC to run the election algorithm frequently. By doing so, we concluded that in a typical real world scenario, the DCEC architecture outperforms the other two architectures. To evaluate the performance, two metrics: the number of control messages and delay were measured. In both cases, it was concluded that the DCEC architecture outperforms the other two conventional CoMP architectures. Furthermore, it was concluded that DCEC is less sensitive to an increase in the number of users meaning that the performance of DCE is less affected by an increase in the number of UEs in CoMP.

As a possible extension to this research, DCEC can be applied to femto and pico cells inside buildings. Furthermore, D2D can be incorporated in the architecture to allow for a better performance.

References

[1] V. H. M. Donald, "Advanced mobile phone service: The cellular concept," *The Bell System Technical Journal,* vol. 58, no. 1, pp. 14-41, 1979.

[2] P. Nicopolitidis, A. S. Pomportsis, G. I. Papadimitriou and M. S. Obaidat, Wireless networks, 2003.

[3] L. Goleniewski, Telecommunications Essentials, Second Edition: The Complete Global Source, 2006.

[4] ITU, "ITU World Radiocommunication Seminar highlights future communication technologies," 6 December 2010. [Online]. Available: http://www.itu.int/net/pressoffice/press_releases/2010/48.aspx#.WES1QOYrK00. [Accessed 5 10 2016].

[5] 3GPP, "3GPP TS 36.213 version 11.0.0 Release 11," 2012.

[6] Cisco, "Cisco visual networking index: Global mobile data traffic forecast update, 2015–2020 White Paper," Cisco, 2016.

[7] Ericsson, "Ericsson mobility report," 2015.

[8] M. Peng, Y. Li, Z. Zhao and C. Wang, "System architecture and key technologies for 5G heterogeneous cloud radio access networks," *IEEE Network,* vol. 29, no. 2, pp. 6-14, 2015.

[9] B. U. Kazi, M. Etemad, G. Wainer and G. Boudreau, "Signaling overhead and feedback delay reduction in heterogeneous multicell copperative networks," in

International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS), Montreal, 2016.

[10] 3GPP, "3GPP TR 36.819 version 11.2.0," 2013.

[11] M. Ding and H. Luo, Multi-point cooperative communication systems: Theory and applications, Shanghai: Springer-Verlag Berlin Heidelberg, 2013.

[12] P. Marsch and G. P. Fettweis, Coordinated multi-point in mobile communications: From theory to practice, Cambridge University Press, 2011.

[13] B. Özbek and D. L. Ruyet, "Feedback strategies for multicell systems," *Feedback Strategies for Wireless Communication*, pp. 249-293, 2014.

[14] S. Sun, Q. Gao, Y. Peng, Y. Wang and L. Song, "Interference management through CoMP in 3GPP LTE-advanced networks," *IEEE Wireless Communications*, vol. 20, no. 1, pp. 59-66, 2013.

[15] A. Papadogiannis, H. Bang, D. Gesbert and E. Hardouin, "Efficient selective feedback design for multicell cooperative networks," *IEEE Transactions on Vehicular Technology*, vol. 60, no. 1, pp. 196-205, 2011.

[16] A. Papadogiannis, E. Hardouin and D. Gesbert, "Decentralising multicell cooperative processing: A novel robust framework," *EURASIP Journal on Wireless Communications and Networking*, vol. 2009, no. 1, 2009.

[17] I. F. Akyildiz, D. M. Gutierrez-Estevez and E. C. Reyes, "The evolution to 4G cellular systems: LTE-Advanced," *Physical Communication*, vol. 3, no. 4, pp. 217-244, 2010.

[18] B. Kazi, M. Etemad, G. Wainer and G. Boudreau, "Using elected coordination stations for CSI feedback on CoMP downlink transmissions," in *2016 International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS)*, Montreal, 2016.

[19] Cambridge Broadband Networks, "Backhauling X2," April 2011. [Online]. Available: http://cbnl.com/resources/backhauling-x2. [Accessed February 2016].

[20] G. Wainer, Discrete-event modeling and simulation: A practitioner's approach, CRC Press, 2009.

[21] B. U. Kazi, "Efficient control plain architecture for multicell cooperative LTE and beyond cellular networks".

[22] S. Parkvall and D. Astely, "The evolution of LTE towards IMT-advanced," *Journal of Communications,* vol. 4, no. 3, pp. 146-154, 2009.

[23] A. B. Saleh, S. Redana, B. Raaf, T. Riihonen, J. Hamalainen and R. Wichman, "Performance of Amplify-and-Forward and Decode-and-Forward Relays in LTE-Advanced," in *IEEE 70th Vehicular Technology Conference Fall*, Anchorage, 2009.

[24] S. Sesia, I. Toufik and M. Baker, The UMTS Long Term Evolution: From Theory to Practice, Wiley, 2011.

[25] J. Lee, J.-K. Han and J. Zhang, "MIMO technologies in 3GPP LTE and LTE-advanced," *EURASIP Journal on Wireless Communications and Networking,* vol. 2009, no. 1, pp. 1-10, 2009.

[26] J. Lee, Y. Kim, H. Lee, B. L. Ng, D. Mazzarese, J. Liu, W. Xiao and Y. Zhou, "Coordinated multipoint transmission and reception in LTE-advanced systems," *IEEE Communications Magazine*, vol. 50, no. 11, pp. 44-50, 2012.

[27] S. Sun, Q. Gao, Y. Peng, Y. Wang and L. Song, "Interference management through CoMP in 3GPP LTE-advanced networks," *IEEE Wireless Communications*, vol. 20, no. 1, pp. 59-66, 2013.

[28] D. Lee, H. Seo, B. Clerckx, E. Hardouin, D. Mazzarese, S. Nagata and K. Sayana, "Coordianted multipoint transmission and reception in LTE-Advanced: Deployment scenarios and operational challenges," *IEEE Communications Magazine*, vol. 50, no. 2, pp. 148-155, 2012.

[29] 3GPP, "3GPP TR 36.942 version 13," 2016.

[30] M. Baker, S. Sesia and I. Toufik, LTE - The UMTS Long Term Evolution: From Theory to Practice, Wiley, 2011.

[31] S. Brueck, L. Zhao, J. Giese and A. Amin, "Centralized scheduling for joint transmission coordinated multi-point in LTE-advanced," in *2010 International ITG Workshop on Smart Antennas (WSA)*, Bremen, 2010.

[32] Y. Gao, Y. Li, H. Yu and S. Gao, "Performance of dynamic CoMP cell selection in 3GPP LTE system level simulation," in *2011 IEEE 3rd International Conference on Communication Software and Networks*, Xi'an, 2011.

[33] T. Okamawari, H. Hayashi and T. Fujii, "A proposal on network control architecture for CoMP JT with IP network between eNBs," in *2012 IEEE 75th Vehicular Technology Conference (VTC Spring)*, Yokohama, 2012.

[34] M. Feng, X. She, L. Chen and Y. Kishiyama, "Enhanced dynamic cell selection with muting scheme for DL CoMP in LTE-A," in *2010 IEEE 71st Vehicular Technology Conference (VTC 2010-Spring)*, Taipei, 2010.

[35] 3GPP, "Investigation on coordinated multipoint transmission schemes in LTE-Advanced downlink," 2009.

[36] M. Sawahashi, Y. Kishiyama, A. Morimoto, D. Nishikawa and M. Tanno, "Coordinated multipoint transmission/reception techniques for LTE-advanced [Coordinated and Distributed MIMO]," *IEEE Wireless Communications,* vol. 17, no. 3, pp. 26-34, 2010.

[37] G. Cili, H. Yanikomeroglu and F. R. Yu, "Cell switch off technique combined with coordinated multi-point (CoMP) transmission for energy efficiency in beyond-LTE cellular networks," in *2012 IEEE International Conference on Communications (ICC)*, Ottawa, 2012.

[38] M. Tavanpour, M. Moallemi, G. Wainer, J. Mikhail, G. Boudreau and R. Casselman, "Shared segmented upload in mobile networks using coordinated multipoint," in *International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS 2014)*, Monterey, 2014.

[39] A. Khandekar, N. Bhushan, J. Tingfang and V. Vanghi, "LTE-advanced: Heterogeneous networks," in *2010 European Wireless Conference*, Lucca, 2010.

[40] E. Dahlman, S. Parkvall and J. Skold, 4G: LTE/LTE-Advanced for Mobile Broadband, 2 ed., Waltham: Elsevier, 2014.

[41] J. I. Agbinya, M. C. Aguayo-Torres, R. Klempous and J. Nikodem, 4G Wireless Communication Networks: Design Planning and Applications, Aalborg: River Publishers, 2013.

[42] J. C. Ikuno, M. Wrulich and M. Rupp, "System level simulation of LTE networks," in *2010 IEEE 71st Vehicular Technology Conference (VTC 2010-Spring)*, Taipei, 2010.

[43] "Vienna LTE-A Simulators," [Online]. Available: http://www.nt.tuwien.ac.at/ltesimulator/.

[44] C. Mehlfuhrer, M. Wrulich, J. C. Ikuno, D. Bosanska and M. Rupp, "Simulating the long term evolution physical layer," in *17th European Signal Processing Conference (EUSIPCO 2009)*, Glasgow, 2009.

[45] M. Mezzavilla, M. Miozzo, M. Rossi, N. Baldo and M. Zorzi, "A lightweight and accurate link abstraction model for the simulation of LTE networks in ns-3," in *15th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, Paphos, 2012.

[46] A. Virdis, G. Stea and G. Nardini, "SimuLTE - A modular system-level simulator for LTE/LTE-A networks based on OMNeT++," in *2014 International Conference on Simulation and Modeling Methodologies, Technologies and Applications (SIMULTECH)*, Vienna, 2014.

[47] "Simulating LTE cellular systems: An open-source framework," *IEEE Transactions on Vehicular Technology*, vol. 60, no. 2, pp. 498-513, 2011.

[48] B. P. Zeigler, Theory of Modeling and Simulation, New York: Wiley Interscience, 1976.

[49] U. Farooq, G. Wainer and B. Balya, "DEVS modeling of mobile wireless ad hoc networks," *Simulation Modelling Practice and Theory,* vol. 15, no. 3, pp. 285-314, 2007.

[50] G. Wainer, "CD++: a toolkit to develop DEVS models," *Journal of Software: Practice and Experience,* vol. 32, no. 13, pp. 1261-1306, 2002.

[51] B. P. Zeigler and H. S. Sarjoughian, "Introduction to devs modeling and simulation with java: Developing component-based simulation models," 2003.

[52] M. Moallemi, G. Wainer, S. Jafer, G. Boudreau and R. Casselman, "Simulation of mobile networks using discrete event system specification theory," in *16th Communications & Networking Symposium*, San Diego, 2013.

[53] M. Tavanpour, G. Wainer, G. Boudreau and R. Casselman, "DEVS-based modeling of coordinated multipoint techniques for LTE-advanced," in *16th Communications & Networking Symposium*, San Diego, 2013.

[54] "Cached and segmented video download for wireless video transmission," in *49th Annual Simulation Symposium (ANSS)*, Pasadena, 2016.

[55] A. Al-Habashna, G. Wainer, G. Boudreau and R. Casselman, "Distributed cached and segmented video download for video transmission in cellular networks," in *2016 International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS)*, Montreal, 2016.

[56] J. Vaubourg, V. Chevrier, L. Ciarletta and B. Camus, "Co-Simulation of IP Network Models in the Cyber-Physical Systems Context, using a DEVS-based Platform," in *Proceedings of the Communications and Networking Simulation Symposium (CNS'16)*, Pasadena, 2016.

[57] T. Antoine-Santoni, J.-F. Santucci, E. D. Gentili and B. Costa, "Simulation and visualization method of wireless sensor network performances," in *Proceedings of International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS'07)*, Norfolk, VA, 2007.

[58] Modelling, Simulation and Design Lab (MSDL), McGill University, "PythonPDEVS," 23 11 2016. [Online]. Available: http://msdl.cs.mcgill.ca/projects/DEVS/PythonPDEVS. [Accessed 05 12 2016].

[59] 3GPP, "3GPP TR 36.912 version 9.1.0 Release 9," 2010.

[60] Stoke, "Latency Considerations in LTE Implications to Security Gateway," 2014.

[61] B. Özbek and D. L. Ruyet, Feedback Strategies for Wireless Communication, New York: Springer, 2014.

[62] T. Blajic, D. Nogulic and M. Druzijanic, "Improvements in 3G Long Term Evolution," Ericsson, 2007.

[63] X. Zhou and X. Zhang, LTE-Advanced Air Interface Technology, CRC Press, 2012.

[64] 3GPP, "3GPP TS 36.211 version 13.0.0 Release 13," 2016.

[65] B. Mondal, E. Visotsky, T. A. Thomas, X. Wang and A. Ghosh, "Performance of downlink comp in LTE under practical constraints," in *2012 IEEE 23rd International*

Symposium on Personal, Indoor and Mobile Radio Communications-(PIMRC), Sydney, 2012.

[66] 3GPP, "3GPP TR 25.942 version 13.0.0 Release 13," 1999.

[67] B. Mielczarek and W. A. Krzymien, "Influence of CSI Feedback Delay on Capacity of Linear Multi-User MIMO Systems," in *2007 IEEE Wireless Communications and Networking Conference*, Kowloon, 2007.

[68] L. Shi, Z. Hu, T. Zhang and Z. Zeng, "Performance analysis of delayed limited feedback based on per-cell codebook in CoMP systems," in *2015 IEEE Wireless Communications and Networking Conference (WCNC)*, New Orleans, 2015.

[69] 3GPP, "3GPP TR 36.819 version 11.2.0," 2013.

[70] SalesForce, "2014 Mobile Behavior Report," 2014.