UNIVERSITY OF NAIROBI

School of Engineering

OPTIMIZATION OF HANDOVER IN MOBILE SYSTEM BY USING DYNAMIC GUARD CHANNEL METHOD

BY

ABUBAKAR MOALLIM ALI IBRAHIM

F56/75367/2014

A thesis submitted in fulfilment of the requirements for the Degree of Master of Science in Electrical and Electronics Engineering, in the Department of Electrical and Information Engineering of the University of Nairobi

November, 2018

DECLARATION OF ORIGINALITY

This thesis is my original work and has not been submitted for a degree in any other university.

ABUBAKAR MOALLIM ALI IBRHAIM DATE: ………………………

This thesis has been submitted for award of the Degree of Master of Science in Electrical and Electronic Engineering with our approval as university supervisors

PROF. V.K. ODUOL DATE: ……………………...

Associate Professor,

Department of Electrical and Information Engineering

University of Nairobi, Kenya.

DR. PETER O. AKUON DATE……..……………

Lecturer,

Department of Electrical and Information Engineering

University of Nairobi, Kenya.

ACKNOWLEDGEMENTS

I wish to appreciate and thank my supervisors **PROF. V.K. ODUOL** and **DR. PETER AKUON** for their continued guidance and support during the writing of my thesis. Their expertise bore fruit the achievement and completion of the thesis.

My gratitude goes to all the academic staff of the Department of Electrical and Information Engineering of the University for their continued encouragement throughout the master's program. Also for the access granted to me for the use of all resource and facilities and for a conductive environment.

The Almightily God bless all of you as you endeavour to make the university research work more interesting and of helpful to others.

I will also give a special thanks and gratitude to all my relatives, friends and others, who directly and indirectly, lent their helping hand to the successful completion of this thesis. Their encouragement through interaction and all the strength they gave me to work through the hard days of the thesis.

This thesis therefore is a reflection and stands as proof to the love and encouragement granted.

ABSTRACT

The telecommunications industry in Somali provides communication and information services such as, delivery of voice and data over different networks, including mobile networks. GSM (Global System for Mobile Communication) has grown to become one of the most used communication system in Somalia compared to fixed lines (landlines) because of instability in the country. The mobile subscribers have rapidly increased in the last five years. On the other hand, service qualities provided by the GSM operators in Somalia have remained extremely bad, and the service quality suffering from a high drop calls rate, mostly due to lack of resources. Handover process is a very essential process in the GSM. It's one of the major key performance indicators in every GSM network, and it's linked to the quality of service which each service provider strives to attain. The failure of the handover process is regarded as the drop of quality of service which in turn dissatisfies the customers. This thesis, contributes more on improving call drop rate in general, in order to reduce handover failure rate and thus save on upgrade costs. This will be beneficial to GSM service providers to easily optimize their network performance relating to the resource management. This work is focused on Somali Telecommunication Company where data was collected, as a sample representation of the whole region. In this thesis, dynamic guard channel algorithm is presented that was developed using JAVA Software. The algorithm then prioritizes the handover calls over the new calls. All handover calls are ongoing calls and if they are dropped it causes frustrations. Matlab was used to compare real data to the simulated results by use of graphs and charts. From this thesis we were able to establish and come up with a definitive solution to the handover crisis befalling telecommunication companies. Through the Dynamic Guard Channel Algorithm we were able to propose a means of raising the standards of the region get it to adhere to the set International benchmark of communication. Furthermore, there were no additional costs on upgrades incurred and being able to modify and advance an existing system making it operate more efficiently.

TABLE OF CONTENTS

DECLARATION OF ORIGINALITY ii
ACKNOWLEDGEMENTS iii
ABSTRACT iv
TABLE OF CONTENTS v
LIST OF FIGURES x
LIST OF TABLES xii
ABBREVIATIONS xiii
CHAPTER ONE 1
INTRODUCTION 1
1.1 Background of the study 1
Figure 1.1: Somalia Mobile Subscription from 2000 to 2015 (in millions) [5] 2
1.2 Problem Statement 3
1.3 Objectives 4
1.3.1 Specific objectives 4
1.4 Justification for the Study 5
1.5 Scope of work and its limitations 5
CHAPTER TWO 7
LITERATURE REVIEW 7
2.1 Introduction 7
2.2 GSM network architecture 7
Figure 2.1: GSM Network Architecture [18] 8
2.3 GSM Key Performance Indicator 9
2.3.1 Call Setup Success Rate (CSSR) 9
2.3.2 Call Setup Failure Rate 9
2.3.3 Call Drop Rate (CDR) 10
2.3.4 SDCCH Congestion Rate 10
2.3.5 Traffic Channel Availability 11
2.3.6 Handover Success Rate 11
2.3.7 Paging Success Rate 11

2.4 Key Performance Indicators for Handovers 12

2.5 GSM Handover Initiation 12

Figure 2.2: Signal levels versus handover points [29].. 12

2.6 Types of Handover Mobile System... 13

2.6.1 Intra-BTS handover (between channels).. 13

Figure 2.3: Intra-BTS handover [31]... 13

2.6.2 Intra-BSC handover (between cells)... 13

Figure 2.4: Intra-BSC handover [31]... 14

2.6.3 Intra-MSC Handover (between BSCs) 14

Figure 2.5: Intra-MSC Handover [31].. 15

2.6.4 Inter-MSC Handover (between MSCs) 15

Figure 2.6: Inter-MSC Handover [31]... 15

2.7 Handover decision algorithms 16

2.7.1 Measurement phase.. 16

2.7.2 Resource allocation and initiation phase.. 16

2.7.3 Execution phase.. 16

2.8 Handover Protocols.. 17

2.8.1 Network Controlled Handover (NCHO)... 17

2.8.2 Mobile Assisted Handover (MAHO)... 17

2.8.3 Soft Handover (SHO) 18

2.8.4 Mobile Controlled Handover (MCHO) 18

2.9 Handover Schemes.. 18

Figure 2.7: Classification of the handover schemes.. 19

2.9.1 Non-Priority Handover Scheme 19

2.9.2 Handover Prioritization Schemes (Handover Management Schemes)....... 19

2.10 Channel Allocation Schemes... 20

2.10.1 Fixed Channel Allocation Scheme 20

2.10.2 Dynamic Channel Allocation Scheme 21

2.10.3 Hybrid Channel Allocation Scheme 22
2.10.4 Comparison between FCA and DCA... 22
Table 2.1: Comparison between DCA and FCA 22
2.11 Types of Handover Management Schemes. 23
2.11.1 Handover Queuing Prioritization Schemes 23
2.11.2 Call Admission Control.. 25
2.11.3 Guard Channel Scheme (GChS)... 25
Figure 2.8: New call and handover call 26
CHAPTER THREE 30
RESEARCH METHODOLOGY 30
3.1 Introduction.. 30
3.2 The Study Area 30
Figure 3.1: Map showing the Study Area [63].. 30
3.3 Choice of the algorithm 31
3.4 Dynamic guard channel model description 32
Figure 3.2: Channel Assignment structure with Priority for Handover Calls [48] 33
Figure 3.4: Flowchart for Handover Calls 35
3.4.1 Algorithmic Steps attained from the Flowchart 36
3.5 Data Collection 37
3.6 Materials for Case Study Simulation 37
3.6.1 Performance Indicators 38
CHAPTER FOUR 42
RESULTS AND INTERPRETATION 42
4.0 Introduction.. 42
4.1 Data Analysis and Interpretation... 42
Table 4.1: Average of the real data QoS indicators.. 42
Table 4.2: Communication Authority of Somalia (CAS) KPI Benchmark 43
4.1.1 Call Setup Success Rate 43

Figure 4.1a: Call Setup Success Rate of 14 cells with Communications Authority of Somalia target 44
Figure 4.1b: Call Setup Success Rate of 13 cells with Communications Authority of Somalia target 44
4.1.2 Call Drop Rate 45
Figure 4.2a: Call Drop Rate of 14 cells with Communications Authority of Somalia target 45
Figure 4.2b: Call Drop Rate of 13 cells with Communications Authority of Somalia target 46
4.1.3 Call Block Rate 46
Figure 4.3 a: Call Block Rate of 14 cells with Communication Authority of Somalia target 47
4.1.4 Handover Success Rate 48
Figure 4.4a: Handover Success Rate of 14 cells with Communications Authority of Somalia target 48
Figure 4.4b: Handover Success Rate of 13 cells with Communications Authority of Somalia target 48
4.1.5 New Call Blocking Probability and Handover Call Dropping Probability. 49
Table 4.3: New Call Blocking Probability and Handover Call Dropping Probability 50
4.2 Performance Analysis of the simulated results 51
Table 4.4: Average Simulated results 52
Table 4.5: Non-priority Scheme simulated values 52
4.2.1 Handover Call Dropping Probability 53
Figure 4.6: Comparison of handover dropping probability real, handover dropping Probability simulated (with guard channel) and non-priority scheme handover dropping Probability simulated 54
Figure 4.7: Handover call dropping probability 55
4.2.2 Comparing handover success rate real to simulated handover 55
Figure 4.8: Comparison between handover success rate simulated and handover success rate real 56
4.2.3 New call blocking probability 57

4.2.4 Comparing New Call Blocking Probability to handover call dropping probability ... 58
Figure 4.10: New call blocking probability vs handover-call dropping probability ... 59
4.2.5 Channel utilization ... 59
Figure 4.12: Channel utilization ... 60
CHAPTER FIVE ... 61
CONCLUSION, RECOMMENDATIONS AND FURTHER INVESTIGATIONS ... 61
5.1 Conclusion ... 61
5.2 Recommendations ... 62
5.3 Suggestions for Further Work ... 63
REFERENCES ... 65
APPENDIX A: JAVA Code for our algorithm ... 71

LIST OF FIGURES

Figure 1.1: Somalia Mobile Subscription from 2000 to 2015 (in millions) [5]............ 2
Figure 2.1: GSM Network Architecture [18].. 8
Figure 2.2: Signal levels versus handover points [29]... 12
Figure 2.3: Intra-BTS handover [31].. 13
Figure 2.4: Intra-BSC handover [31].. 14
Figure 2.5: Intra-MSC Handover [31].. 15
Figure 2.6: Inter-MSC Handover [31].. 15
Figure 2.7: Classification of the handover schemes 19
Figure 2.8: New call and handover call 26
Figure 3.1: Map showing the Study Area [63]... 30
Figure 3.2: Channel Assignment structure with Priority for Handover Calls [48] 33
Figure 3.4: Flowchart for Handover Calls 35
Figure 4.1a: Call Setup Success Rate of 14 cells with Communications Authority of Somalia target 44
Figure 4.1b: Call Setup Success Rate of 13 cells with Communications Authority of Somalia target 44
Figure 4.2a: Call Drop Rate of 14 cells with Communications Authority of Somalia target.. 45
Figure 4.2b: Call Drop Rate of 13 cells with Communications Authority of Somalia target.. 46
Figure 4.3 a: Call Block Rate of 14 cells with Communication Authority of Somalia target.. 47
Figure 4.4a: Handover Success Rate of 14 cells with Communications Authority of Somalia target 48
Figure 4.4b: Handover Success Rate of 13 cells with Communications Authority of Somalia target 48
Figure 4.6: Comparison of handover dropping probability real, handover dropping Probability simulated (with guard channel) and non-priority scheme handover dropping Probability simulated... 54
Figure 4.7: Handover call dropping probability... 55
Figure 4.8: Comparison between handover success rate simulated and handover success rate real 56

Figure 4.10: New call blocking probability vs handover-call dropping probability... 59
Figure 4.12: Channel utilization.. 60

LIST OF TABLES

Table 2.1: Comparison between DCA and FCA 22
Table 4.1: Average of the real data QoS indicators 42
Table 3.1: Simulation parameters 40
Table 4.2: Communication Authority of Somalia (CAS) KPI Benchmark 43
Table 4.3: New Call Blocking Probability and Handover Call Dropping Probability 50
Table 4.4: Average Simulated results 52
Table 4.5: Non-priority Scheme simulated values 52

ABBREVIATIONS

AuC	Authentication centre
BSC	Base station controller
BTS	Base transceiver station
BER	Bit error rate
CAC	Call admission control
CDR	Call drop rate
CGI	Cell Global Identity
CSSR	Call setup success rate
DCA	Dynamic channel allocation
DGCM	Dynamic guard channel method
EIR	Equipment identity registers
FCA	Fixed channel allocation
GChs	Guard channel scheme
GSM	Global system for mobile communication
HCA	Hybrid channel allocation scheme
HLR	Home location register
HQS	Handover queuing scheme
IMEI	International mobile equipment identity
KPI	Key performance indicator
MAHO	Mobile assisted handover
MCHO	Mobile controlled handover
MSC	Mobile switching centre

MS	Mobile station
NCHO	Network controlled handover
QOS	Quality of service
RSS	Received signal strength
SDCCH	Standalone dedicated control channel
SHO	Soft handover
TCH	Traffic channel
VLR	Visitor locations register

CHAPTER ONE
INTRODUCTION

1.1 Background of the study

Telecommunication has been a very significant communication medium in our society since decades ago where we used telephones to communicate with other people, and also is considered an important part in facilitating the combination of domestic economy and promoting business and economic development. The national Somali telecommunication collapsed, after the fall of Somalia's central government in 1991. Telecommunication has managed to develop inspite of the violence an over twenty five year's conflict devastating the country, piracy, Islamic extremist fighting groups and lack of regulation. In fact, the absence of taxes and lack of regulation has been a major innovation in the growth of the private operators.

Telecommunications industry in Somali provides communication and information services that is, delivery of voice and data over different networks, including mobile networks. Currently, there are seven network licensed operators whose networks cover the whole country. Hormuud is the largest company in Somalia commanding 41% of the total subscribers in the country. Hormuud Telecom alone grosses about $40 million a year [1].It was taken over by the largest money transfer organization in the region. This was done eventually to manage the mobile money platform developed by Hormuud [2].

GSM has grown to become the most used in Somalia compared to fixed lines (landlines) because of instability. In early 2013, the study showed the Somali mobile subscriber's rates are at 78.5% in South-Central, 73.1% in north-eastern and 56.2% in north-western [3]. Currently, more than nine in ten Somalis (say 90%) personally own a mobile phone, and are still growing. Some factors are responsible for this growth include: they have the lowest international call rates in Africa and one of the cheapest in the world [4]. Mobile subscribers have rapidly increased in the last five years. They moved from 0.6 million in 2010 to 5.18 million in 2015 as shown in Figure 1.1.

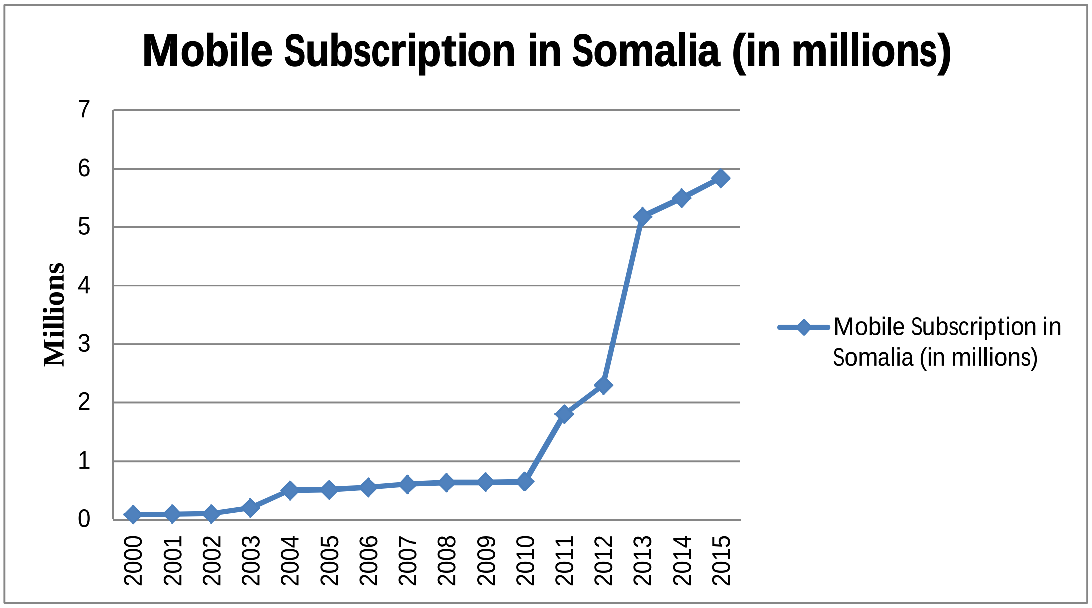

Figure 1.1: Somalia Mobile Subscription from 2000 to 2015 (in millions) [5]

GSM is a second generation, digital cellular system. The GSM network consists of building blocks called cells. Cells are geographical areas that have Base Station Transceiver (BST). The implementations of cells are to maximize the number of served customers within a given zone. GSM is characterized by mobility and limited resource. GSM works on three frequency spectrum. In order to efficiently use the frequency bands, a GSM network always use a combination of FDMA and TDMA. With this it is able to serve mobile users who are also on the move.

Conventionally, after network infrastructure deployment, a few unforeseen problems are met. We start to notice the process where a person is able to migrate from a cell area to another and still maintain the network availability. This process is called "*handover process*". As [6] described, Handover is a process that transfers or shifts an active call from one cell area to another cell area as the subscriber goes through the coverage area of a cellular wireless network. Noerp [7] elsuggested that, for a handover to be successfully processed, the new BS selected, must have an idle available channel that is open to handle the new handover call. Otherwise, the handover call will be dropped. Rather than dropping the call to the current MS, the MSC that is serving currently may decide to handover or shift this call to another available and better serving base station system or in some times to another mobile switching centre [8]. By doing this, GSM ensures a greater retention of calls thus improving the quality of service. Handover process is therefore very critical where improper handover process can result to call drops. If the rate of dropped calls increases, the subscriber becomes annoyed and dissatisfied and sooner or later he/she moves to another network. In addition to that, as [9], [10] acknowledged, new calls are usually have low sensitivity to delay than a handover calls because handover calls need continuity for voice and message transfer.

According to [7], The Guard channel offers an alternative method to achieve successful handover requests which requires preserving some channels only for handover calls. The rest of unreserved channels are therefore shared equally both new calls and handover calls. Increasing the number of guard channels will increase the calls priority of accessing the channel and at the same time decrease the call priority on accessing originated calls [8].

1.2 Problem Statement

Generally, the service qualities provided by the GSM operators in Somalia have remained extremely bad. Mainly, every user to the country's GSM networks is affected. In order to implement a better QoS, The Communication Authority of Somalia (CAS), sets a benchmark for the various key performance indicators (KPIs).Therefore, benchmarking is some sort of test on the quality of service for a network in operation [11]. The KPIs are metrics for measuring the quality of service of the GSM networks [12]. Research by Ministry of telecommunication and information on the quality of GSM KPIs in Somalia showed that GSM operators in Somalia has failed to achieve benchmark for the parameters call setup success rates (CSSR), "which is the rate of no of call attempts that have connected successfully", the call drop rate (CDR), "the rate of incomplete calls" and Handover Success Rate (HSR)," the rate of successfully completed handovers". The report also confirmed that the handover call drop rate is among the worst performing metric in the country which is above 15%, which is beyond the 2% handover failure rate benchmark set by the CAS. It comes to this high because the available Base stations and channels are inadequate to support the variation subscribers within one cell. In other words, it can be due to the increasing gap in growth between the growth subscribers' base and lack of equal growth in investment in the mobile telecom infrastructure [13].

For us to measure the performances of cellular networks, three parameters for the QoS are put into consideration: the originated call setup failure rate, the network utilization and the handover call failure rate [14]. From a subscriber's viewpoint the goal is to reduce the originated call setup failure rate and the handover failure rate, while from the operator's point of view the main aim is to maximize the utilization of the system without any hiccups. Different researchers have proposed different ideas of how to minimize the problem of handover call drops. This included: setting up of additional BTSs, increasing channels in each base station to accommodate traffic and establishing in-building coverage. Adding to that, the process that is being proposed from [15], which is indicated the minimum amount of info to be exchanged before the handover is done.

Based on the above discussion, the thesis focused on and designed a system that can make a fairness between both the service provider and user satisfaction. We presented an algorithm that would get a low handover failure rate, reduce the call setup failure rate and increase the system utilization.

1.3 Objectives

The main general objectives are to present an algorithm that guarantees the targeted QoS benchmarks in cellular system. *Case study of Somali Telecommunication Company*.

1.3.1 Specific objectives

The Specific objectives are:

i) To improve communication within the region.

ii) To increase handover success rate.

iii) To calculate the number of new call blocked and handover call dropped within a cell.

iv) To study the impact of new call blocking probability, handover call dropping probability and call set-up time on poor quality of service.

v) To design dynamic guard channel algorithm to improve the handover success rate and new call blocking rate to improve the quality of service of cellular networks.

1.4 Justification for the Study

Somali GSM operators are suffering high drop calls, and that mostly due to lack of resources. Handover process is a very essential process in the GSM. It's one of the major key performance indicators in every GSM network, and it's linked to the quality of service each service provider strives to attain. The failure of the handover process is regarded as the drop of quality of service which in turn dissatisfies the customers. This thesis, however, contributed more on improvement of call drop rate in general, reduce handover failure rate and thus saved on upgrade costs. This would be beneficial to GSM service providers to easily optimize their network faults relating to the resource management.

In addition to that, this thesis allows other researchers interested in this field to take advantage of the results presented in this thesis. It should also be noted that this algorithm is also applied to other generations like 4G. One of the motivations of this research is the new Safaricom service Guarantee Program which refunds customers for any dropped calls because of poor network or any other possible network interruptions.

1.5 Scope of work and its limitations

The thesis focused on handovers and new calls processes. It focused on Somali Telecommunication Company where data was collected, as a sample representation of the whole region.

A simulation of the Dynamic Guard Channel Algorithm was developed using JAVA Environment Software. The algorithm therefore prioritizes the handover calls over the new calls. All handover calls are ongoing calls and if they are dropped it causes frustrations. Matlab has been used to compare the real data to the simulated results by use of graphs and charts.

The limitation of the thesis is:

- This algorithm does not require any additional infrastructure but uses the available resources provided by the system.
- The algorithm minimizes the handover failure rates to an acceptable level but does not completely eliminate handover drop calls.
- Due to the constraints of time, finances and logistics, we were not able to test the algorithm on a real live GSM System.

CHAPTER TWO
LITERATURE REVIEW

2.1 Introduction

The aim of this chapter is to review the literature and current knowledge from various scholars within the area of GSM system, key quality of service performance indicators and optimization of handover mobile system discussing several channel allocation methods.

2.2 GSM network architecture

GSM is a second generation standard digital cellular system. It deploys eight time slotted subscribers for each 200 KHz radio channel and employed in the entire world. The GSM system is categorized into three key systems: Mobile Station, Network subsystem, and the Base Station Subsystem. The main GSM network elements are shown in Figure 2.1. GSM network normally known as 'cellular system' (as the entire area covered is divided into several sectors and cells, each served by its local antenna) constituted of a MS which is directly connected to the BTS through the Um interface. The GSM system identifies each cell through the CGI number allocated to every cell. Adding to other hardware, base transceiver station has the equipment known as Transceiver (TRX), which facilitates the receiving and transmitting a number of a RF (radio frequency) signals from or to the mobile device [16]. After that BTS connected to the BSC through A-bis interface. Base station controller generally controls network resource management and the process of the handover where a person is able to migrate from a cell area to another equipped in it, then the base station controller is connected to MSC [17]. Generally, we have three main interfaces, A interface between MSC and BSC, Um interface between the BTS and MS, and A-bis Interface between BSC and BTS.

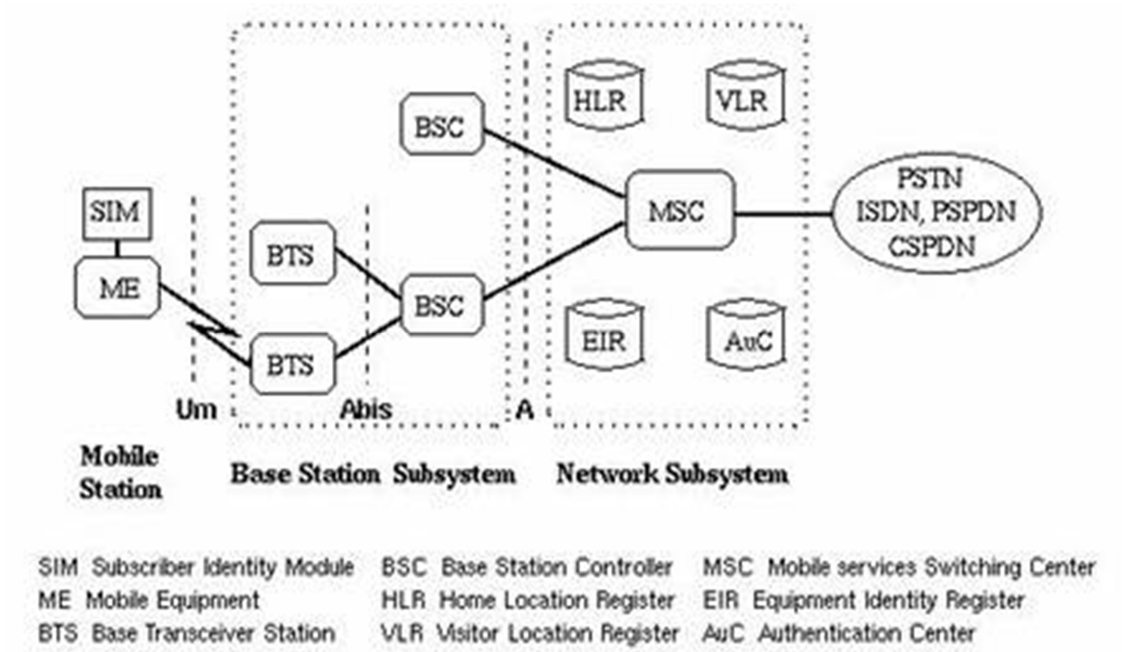

Figure 2.1: GSM Network Architecture [18]

Based on the number of subscribers and the topology, each base station can serve the range between 5-20 base transceiver stations while mobile switching centre controls about 5-15 base station controls each [19]. The network providers have at least one MSC. When a MS roams into another mobile switching centre area, the VLR connected to that mobile switching centre will require information about the MS from the HLR. The HLR stores the data permanently and temporarily on each of the MS as long as the user is with the operator's coverage area. The importance of the HLR is to manage the mobility of the mobile terminal. When users buy a SIM Card from the operator, their data is registered in HLR. The VLR mostly deals with the exchange and roaming characteristics of subscribers acquired from foreign a network which also holds the up-to-date details of its own subscribers roaming in other systems. The VLR is essential when the subscriber goes out the coverage area concealed by his local mobile switching centre, so that it assists handovers. The both registers are used for security and authentication issues. The (AuC) has the encryption and validation parameters which are kept in every SIM subscriber for encryption and verification over radio frequency [20]. It protects the service provider from various types of harmers in today's wireless network. The equipment identity register is an information record that has information of all legitimate mobile equipment on the system. It tracks the type of equipment that available at the MS. This assists in security within the network and also prevents calls from unauthorized and stolen MSs which are not approved to use the network. There are two fundamental types of services offered by GSM system which are voice services and data services.

2.3 GSM Key Performance Indicator

The performance of wireless cellular networks can be measured using different Performance metrics (KPIs). For effective radio network optimization, it is important to pre-select appropriate KPIs to focus on, and monitoring closely during network processing. The following KPIs are essential for GSM system optimization and evaluation to attain good Quality of service:

2.3.1 Call Setup Success Rate (CSSR)

CSSR is one of the main factors to consider when determining the KPIs to be used by the operators to create a measure of the performance of the networks which contain a direct impact on the client contentment due to the services offered by the network operators [21]. A higher call setup success rate is gotten when the SDCCH seizures and traffic channel (TCH) distribution are easily realised to establish a call. The model value of CSSR is One (1), meaning that the network ought to be able to accept 100% of the calls made. It is calculated as [22]:

$$\text{CSSR} = \frac{\text{Total no of accepted originated calls}}{\text{Total number of Call attempts}} * 100\% \qquad 2.1$$

2.3.2 Call Setup Failure Rate

This talk about the ratio of the number of the call attempts blocked to the sum of number of phone call attempts made. This is also known as the Blocking Probability and conveyed in percentage 100% [23]

$$\text{Call Setup Failure Rat} = \frac{\text{Total no of rejected originated calls}}{\text{Total number of Call attempts}} * 100\% \qquad 2.2$$

The [21], indicated that there are several causes for a poor CSSR. Like; TCH Failure Assignment, Hardware Problem, SDCCH Congestion, CM Service Reject and Low Signal Strength.

2.3.3 Call Drop Rate (CDR)

CDR measures the network capability to hold ongoing calls when it has been established or setup. A dropped call is a call that has been dismissed early before being released usually by either the called party or the caller [22]. The CDR is the ratio of the number of dropped calls to the total sum of call tries [23]. Call drop rate would be equivalent to or less than 3% [24].

$$\text{CDR} = \frac{\text{Number of dropped calls(calls terminated unwillingly)}}{\text{Total number of call attempts}} * 100\% \quad 2.3$$

$$\text{CDR} = 1 - \text{CallCompletionRatio} * 100\% \quad 2.4$$

This indicator is very crucial when one wants to compare different users.

2.3.4 SDCCH Congestion Rate

This Stand-Alone Dedicated Control Channel (SDCCH) measures the availability of signalling capacity to establish a call; its recommended value is 0.2% or less [25]. Its congestion happens during network overload due to high amount of location update requirements, large number of Short Message Services (SMS) message traffic or in circumstances where a lot of users are trying to establish a call at the same time without sufficient SDCCH properties to hold these requests [22]. This involves location update message, SMS, and call setup [23]. SDCCH success rate is calculated as a percentage of all Stand-Alone Dedicated Control Channel accesses received in the base station controller. More analysis and calculations [25]:

$$\text{SDCCH Congestion}(\%) = \frac{\text{Number of connect fails due to immediate assignment failures}}{\text{Mobile Originating Call attempts}} * 100\% \quad 2.5$$

2.3.5 Traffic Channel Availability

It is that channel used by MS for communication. TCH congestion is caused by luck of accessible of TCH channels [22].TCH availability is calculated as [23].

$$TrafficChannelAvailability = \frac{BusyHourTCHTraffic\ (Erlang) - AverageTCHTraffic\ (Erlang)}{Busy\ Hour\ TCH\ Traffic\ (Erlang)} \quad 2.6$$

2.3.6 Handover Success Rate

This parameter is used to measure the percentage of successful accepted handover calls by mobile subscribers during a voice call. Mathematically, HSR is the number of successfully handover calls completed divided by the sum of the handover attempts [23].

$$HSR = \frac{Number\ of\ successfully\ completed\ handovers}{Number\ of\ initiated\ handovers\ (all\ handover\ attempts)} * 100\% \quad 2.7$$

$$HandoverFailureRate = 1 - HandoverSuccessRate \quad 2.8$$

Handover attempts are when a handover signal is sent to a mobile device [26].

A few reasons why handover success rates could be poor are [27]:

I. Congestion
II. Bad Antenna Installation
III. Poor coverage
IV. Co-channel/Interference is high

2.3.7 Paging Success Rate

PSR usually [27] calculates the rate of how many successful paging tries made that have been responded, resulting from first or the second recurrent page.

$$PSR = \frac{Paging\ Response\ Time}{Paging\ Time} \quad 2.9$$

Low paging Performance could be attributed to:

I. Poor paging strategy
II. Poor coverage
III. High interference

2.4 Key Performance Indicators for Handovers

The following factors are viewed as the performance indicators for a handover call process [6]:

a) *Handover blocking probability:* The probability that a handover request is rejected.
b) *Handover probability:* when the subscriber goes out of the current cell, the on-going call needs a handover before the call drops. This indicator expresses the average number of handovers per cell.
c) *Call dropping probability:* This happens when a call drops as a result of handover failure.
d) *Interruption Duration:* The time period of handover which the mobile device is in connection with neither BS.

2.5 GSM Handover Initiation

Handover depends on signal received strength from the under usage running BS and the target BS. The Figure 2.2 illustrates a MS is moving from one BS (called BSold) to another BS (called BSnew). The RSS of BSold goes down as the MS moves away and goes up as the MS approaches to the BTSnew due to the signal propagation [28]. There are four approaches that can be used to decide whether to initiate a handover request, namely: signal relative strength with hysteresis, Relative signal strength, threshold & signal relative strength with threshold.

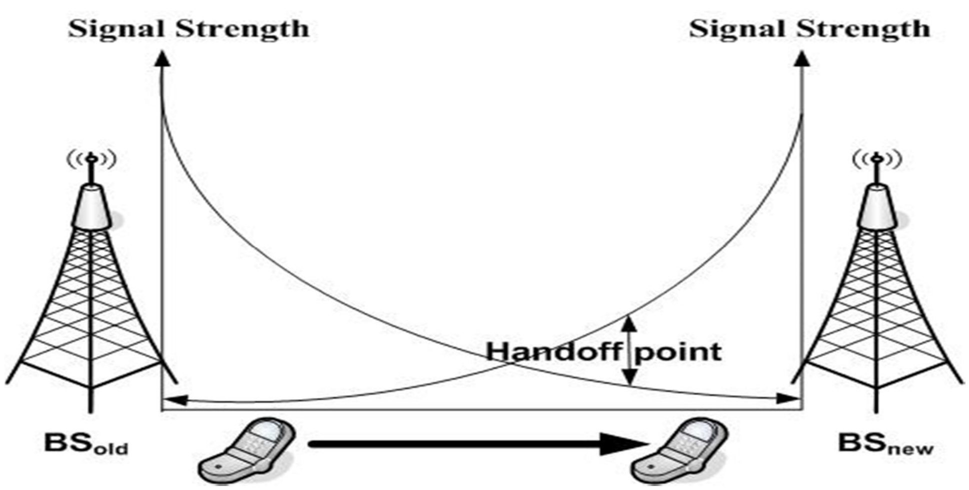

Figure 2.2: Signal levels versus handover points [29]

2.6 Types of Handover Mobile System

The word handover is largely used in Europe, where handoff is inclined to be used in North America [30]. Handoff and Handover refer to the same mechanism. There are four scenarios for handovers calls: Within the second generation system there are four categories of handoff calls that can be done for GSM networks [28]:

2.6.1 Intra-BTS handover (between channels)

In this kind of handover, handover occurs when a new channel in the same base transceiver station is allocated to the mobile station. Intra-BTS handover, occurs when a new frequency is needed because of interference, or other reasons [20]. The procedure is performed independently by the BSC, but the MSC also may also be in charge. It is important to emphasize that an Intra-BTS handover is consistently synchronized, since all transceivers TRXs of a BTS have to use the same clock [31]. Figure 2.3 illustrates Intra-BTS handover.

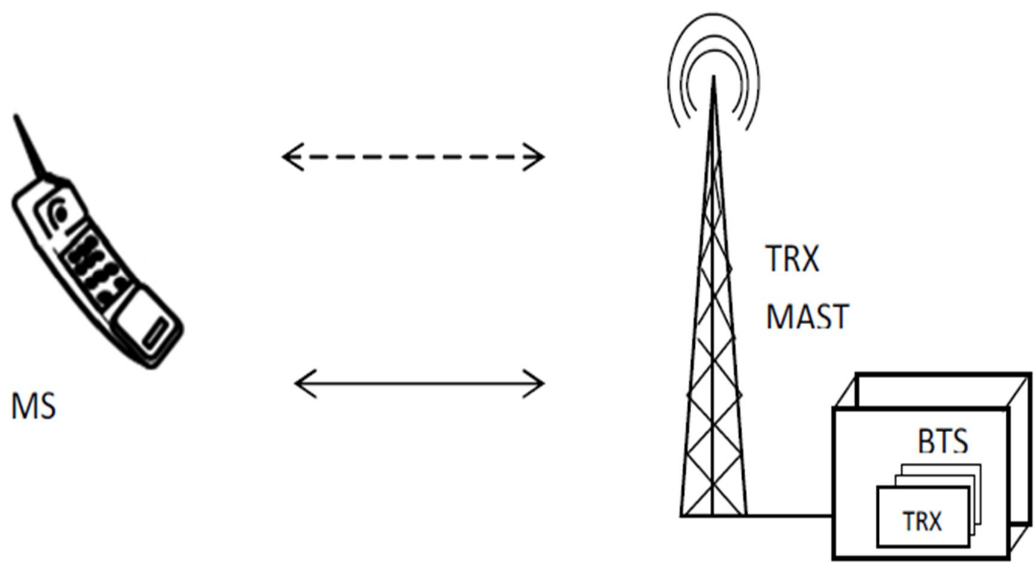

Figure 2.3: Intra-BTS handover [31]

2.6.2 Intra-BSC handover (between cells)

This kind of handover occurs when the call is relocated from one base transceiver station into another base transceiver station coverage area under the control of the same BSC, as illustrated in Figure 2.4. In this case base station controller can do a handover and it allocates a fresh channel before dismissing the discarded BTS from communicating with the handset [20]. This handover may be done independently by the base station controller without the intervention of the mobile switching centre [31]. Moreover, the MSC is alerted when the handoff has occurred. Once the new cell is located in another location then the MSC needs to achieve the location information process after the call [28]. For Intra_BSC handover, dependent on the present situations, synchronized and non-synchronized handover are achievable.

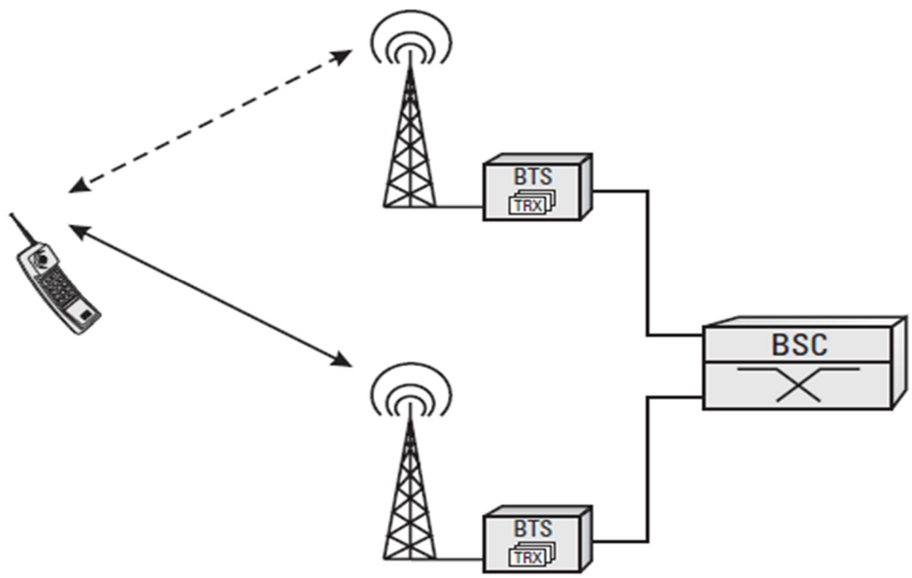

Figure 2.4: Intra-BSC handover [31]

2.6.3 Intra-MSC Handover (between BSCs)

Whenever the cell or mobile moves outside the range of the cells under the control of one base station controller, a more complicated procedure of handover may be executed, handing over the call not only from one base transceiver station to another but also from one BSC to another [31], or, in other words, In this handover, the mobile station changes the base station controller and the base transceiver station, as well, as illustrated in Figure 2.5. Unlike to the Intra-BTS Handover and the Intra-BSC Handover, this handover is controlled by the MSC. The role of the MSC is not processing the capacities of the BTS or MSC but just to achieve the handover. The MSC communicates with the new BSC for the distribution of the mandatory resources and then notifies the BSC when it's done. When the resources are allocated the MSC initiated to access the fresh channel and the call is relocated to the new BSC [28].

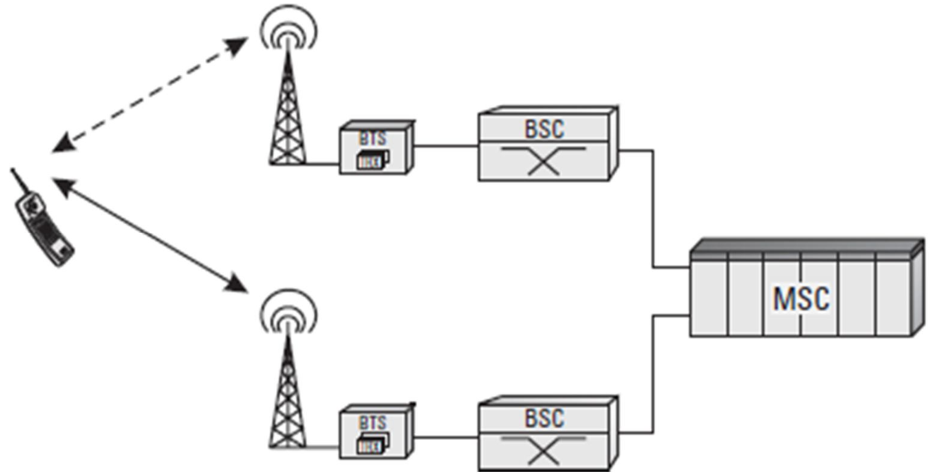

Figure 2.5: Intra-MSC Handover [31]

2.6.4 Inter-MSC Handover (between MSCs)

It occurs when there are two cells which belong to different MSC in the same system [31]. This is illustrated in Figure 2.6. For an inter-MSC handover the old (current serving) MSC is mostly referred to as the anchor the MSC and the new (target) MSC is referred to as the relay MSC [32].

Handovers are created by the MSC or the mobile. The GSM cell use TDMA to scan the 'broadcast control channel' to a limit of 16 neighbouring nodes and therefore creates a list of 6 top best cells for a handover requests. The decision is dependent on the strength of the signal [32].

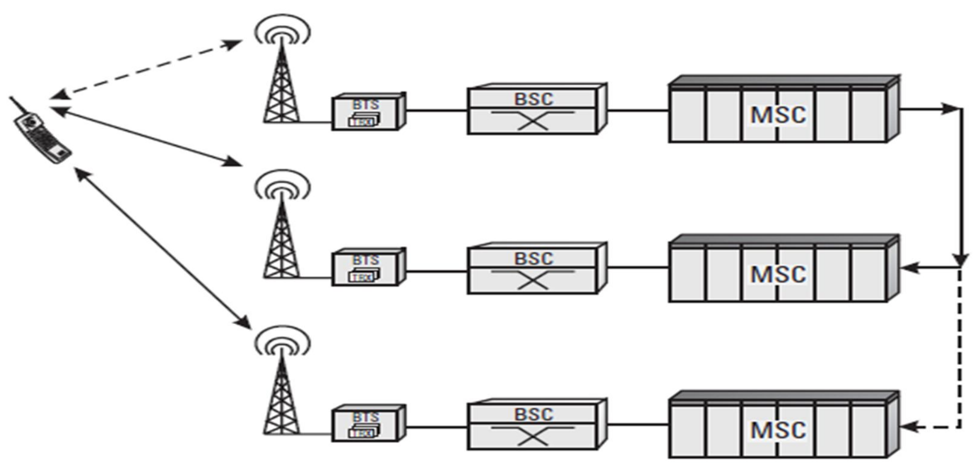

Figure 2.6: Inter-MSC Handover [31]

2.7 Handover decision algorithms

A number of handover decision algorithms are in existence and can be of use, so as to make the best and correct decision to handover the progress link. Irrespective of the nature of the principle, we could discover three successive stages that are involved in all of them [33]:

2.7.1 Measurement phase

The extent of the general quality of communication is determined moreover by the BS, the MS or the two. The measurement of data is then processed and assessed again by either one or the both of MS and BS where suitable act is taken in accordance to the outcome of the assessment i.e. execute handover or not. The parameters being measured could be: received signal levels from the adjacent cell ,received power in down-and-uplink, interference degrees in down-and-uplink, estimate of the terminal velocity, an assess of the distance between the terminal and the BTS and a BER in both down-and-uplink.

2.7.2 Resource allocation and initiation phase

This part contains the verdict whether a handover is required, irrespective of the actual presence of a fresh channel on a neighbouring BS; it is also based on network variables or the processed measurement results, with provided traffic. When the desire for the handover is determined, originated channel is nominated, doing the ideal frequency resource accessibility & network load considerably.

2.7.3 Execution phase

The moment a new channel is chosen, the handover is implemented. The three phases mentioned above are susceptible to errors. The result of measurements is gauged on, may not be true or subjected to noise, which can lead to incorrect decisions. The response to the channel reservation analysis may be affected by an incorrect transmission of, and the effecting of the handover may probably take a lot of time that the environments in the intended end point might have changed while waiting for the handover enters. Any and all achievable precautions is required in order to stay away from the above-mentioned instances that interrupt the handover process, by selecting appropriate handover protocol and handover procedure which is not an easy task to achieve.

The algorithms decisions for Handover, that implement the above-mentioned handover phases, can be classified considering the type and method of input that the system uses when taking the handover decision.

2.8 Handover Protocols

Usually the Handover request is originated either by the MS or by the BS. Handover decision protocols are categorized into four basic types as briefly described below.

2.8.1 Network Controlled Handover (NCHO)

In this technique, the MSC is accountable for the whole handover decision [34]. In network-Controlled Handover protocol [35], the MSC creates a handover decision depending on calculations of the RSS (Signal Received Strength) from the mobile station at a number of base transceiver stations. At occasion when the system creates a linkage between the current and target BTS and this reduces the time period of handover. Generally, the handover process takes 100–200 milliseconds and creates obvious impact in the conversation [36]. Statistics about the quality of the signal for all subscribers is found at MSC. The information enables resource assignment. Therefore [37], the total delay may be about 5–10s. This kind of handover is not fitting for a dynamic condition and congested customers as the result of the associated delay. This protocol is used in the first generation analogue systems like AMPS &NMT [36].

2.8.2 Mobile Assisted Handover (MAHO)

As [35] indicated, a mobile assisted handover (MAHO) procedure allocates the handover decision process. Here the Mobile Station is in charge of discovering the BS where the strength of the signal is closest to it. The Mobile Station calculates the signal strength and bit error rate values occasionally in the neighbouring base station.

Based on the obtained parameters, the MSC and BSs decides when to perform handover [38]. According to [37], there is a possibility of a delay of 1 sec. MAHO is currently being used in GSM.

2.8.3 Soft Handover (SHO)

Soft handover is creating a connection to another station before breaking it, which is, the link to the current base station (BTS) is not destroyed till the link to the target BTS is made. Soft handover utilizes the method of macroscopic diversity [39]. The same concept can be used at the MS too. When a MS is in the overlapping area of two neighbouring cells then a soft or softer handover can occur. The user will therefore have a dual concurrent connection to the UTRAN portion of the network, utilizing various air interface channels simultaneously [40]. This protocol is for CDMA.

2.8.4 Mobile Controlled Handover (MCHO)

In this protocol [35] the mobile station is totally in control of the handover process. This kind of handover has a quick response (about 0.1 s) and is appropriate for micro-cellular systems [37]. The Mobile Station measures interference levels on all channels and the strength quality of the signal from the neighbouring BSs. A handover is originated when the strength of the signal which is serving the base station becomes lesser than the signal of another BS by a definite threshold. The targeted BS is therefore requested by the MS for a channel which has the least interference.

This protocol is the highest level of handover devolution. There are several advantages of handover decentralization, where some of the handover decisions may be done fast, and the mobile switching centre doesn't need to do handover decisions for all mobile or cells, which are very complicated duty for the mobile switching centre of high volume micro-cellular schemes [41]. This protocol is used in the DECT. The handover decisions are made by the mobile station. Both inter cell and intra cell handovers are possible. The handover time is around 100 Ms [35].

2.9 Handover Schemes

In GSM system supporting handover, there is a main element in present to guarantee that there is a continuity of connections and the Quality of Service observed by subscribers. The handover process is then accomplished by the purported handover methods; the main importance is focused on them. The handover methods can be generally categorised into Prioritization and Non-Prioritized Schemes as in Figure 2.7 [42].

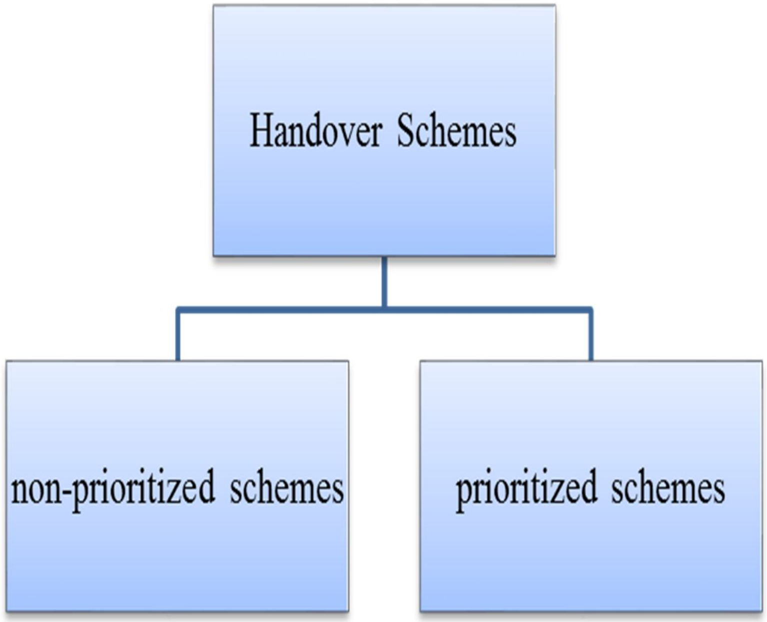

Figure 2.7: Classification of the handover schemes

2.9.1 Non-Priority Handover Scheme

For non-prioritization scheme originated calls and handover calls are to be handled using a similar approach. The moment the BS has a free channel, it is allocated according in a first arrive first serve technique irrespective of if the present call is a new call or handover call. If there aren't any free channels, then the request will be rejected instantly. One of the disadvantages of this method is that, because there isn't any precedence that accorded to the handover request calls over new calls, the handover dropping rate is quite higher than it is basically estimated [43].

From the subscriber's view point, the unintended end of a call which is on-going is more irritating than the blocking of originated call that hasn't even started [44]. Subsequently, the handover dropping and the involuntary dropping probabilities need to be reduced. To attain these requirements, various handover prioritizing methods have been suggested.

2.9.2 Handover Prioritization Schemes (Handover Management Schemes)

The mobile device usage, rapid increase necessitates the needs to meet QoS requirements of the subscribers. These requests nevertheless, allocation usually cause network congestion and call drops. The Different concepts and methods are presented to lessen the handover dropping rate. One of the used techniques is to minimize the percentage of handover failure rate which gives handover higher priority than fresh calls [28]. This prioritization method has a major effect on the handover call drop rate and originated call rejected probability. Therefore such a system works by allowing a high consumption of capacity while ensuring the QoS of handover calls is maintained [20].

Handover prioritization schemes offer a better performance at the cost of a falling in the sum of accepted load traffic and upsurge in the rejected call probability of originated calls [45]. Nonetheless, the enhancement in efficiency is normally associated with the way that each technique provides precedence to handover requests.

2.10 Channel Allocation Schemes

According to [46] channel allocation technique can be categorized into a various types along the foundation of contrasting plans, they devised for allocation of channels. There are three fundamental channel assignment methods in cellular networks.

2.10.1 Fixed Channel Allocation Scheme

In this method, a group of channels is permanently allotted to each base station in such a way that the frequency reuses (the band of frequency assigned for cellular system use can be reused with different cells) control can hardly be disrupted. Each cell only can use its channels. The FCA strategy, originated call or a handover call can only be accepted if there are free channels available in the cell; or else, the call must be dropped or rejected [42]. In this algorithm, the capacity will get wasted if all the assigned channels neither in use nor freely available at that moment bandwidth won’t be allotted [47]. The main advantage of FCA scheme is its lack of complexity, but it is not flexible to varying traffic environments. The major disadvantage of FCA scheme is high blocked calls, because of lag of flexibility in the scheme [46].

In this method, the preliminary channel allocation is essential since every cell in the network is allocated particular channels and cannot be altered during network process. To achieve further productivity and success, FCA networks usually distribute channels in a way that increases as much as possible the frequency reuse, with considering the minimum reuse distance restrictions. In FCA methods, the work of the mobile switching centre is limited and is to notify the new base station about handover requests [46].

This algorithm is useful and well functioned when the demand is not dynamic. However, in the real world the traffic is irregular and changes dynamically with time. This method lacks flexibility to satisfy the dynamic requirement of channels [48].To get a solution of this drawback Dynamic Channel Assignment (DCA) method has been proposed.

2.10.2 Dynamic Channel Allocation Scheme

This method doesn’t assign any channel to each cell [48]. In DCA method, channels are dynamically assigned to cells [46], this is in contrasts to fixed channel allocation. In DCA, The mobile switching centre manages all channels in a region of many cells. When a call comes to these cells, the BS of the cell will request for a channel to MSC. If the centre has any open channels, then it will accept this call. If not, then this call will be blocked [48]. Also, a base station does not own any specific channels and it is returned to the central chosen cell when a call is finished.

In DCA method [49], when the traffic is congested in a cell, additional bandwidth are provided to that cell, and when the traffic become slight, assigned channels are minimized. Dismissed frequencies are made available to other cells which need extra channels. This reflects reducing the call setup failure rate in these high traffic load cells. This channel readjustment process needs the cells to have loads of communication and data exchange. Consequently, a DCA approach must be applied in a way that necessitates the lowest info being exchanged among BSs to reduce the signalling overhead and complexity.

Dynamic channel allocation method offers better QOS than the FCA method, because channels are being allocated depending on bandwidth and demand requirement of the handset in the cell. In this algorithm, channels in the system will be used properly [47]. Generally, in this method the relationship between the modes of communication through channels and cells is not fixed. Differing from the FCA, the sum of the channels in every cell continuously ongoing and dynamically changing, to house traffic variations. This method provides high flexibility, because different cells can get different channels according to their requirements [48]. DCA is less effective than static channel assignment method under high traffic situations. To avoid this problem Hybrid Channel Allocation (HCA) algorithm are projected by combining both DCA and FCA methods.

2.10.3 Hybrid Channel Allocation Scheme

HCA combines both of features of DCA and FCA methods. HCA algorithm allocates other channels dynamically and some channels statically. This algorithm [47], total bandwidth are divided into two clusters. One is static and the other one is dynamic. In [49], the frequencies included in the static channels are allocated to each cell via the FCA methods. But, the dynamic settle of channels is shared by the BSs. When a mobile station needs a channel to handle a call, and the frequencies in the static set are all occupied, then a request from the dynamic set is sent [49]. This technique therefore gives fewer loads to centre than DCA and provides more flexibility than FCA [48].

2.10.4 Comparison between FCA and DCA

Table 2.1: Comparison between DCA and FCA

Evaluation Parameter	**FCA**	**DCA**
Average handover blocking rate	High	Low
Average call dropping probability	More	Less
Minimize the interference	Not good	Good
The load in the network traffic	Un balance	Balance
Resources utilization	Less	More
Channel Al location	Do not modify during processing of calls	Changing Dynamically
Complexity	Less	More
Flexibility	Less	More
Implement at ion Cost	Low	High

The Average handover blocking rate of FCA methods is high when compared with Average handover blocking rate of DCA method. In FCA methods, channels are static and don't change during process. In DCA methods, all frequencies are putted in a central pool and are allocated dynamically to fresh calls as they enter to the network, this led to change dynamically [50].

In an FCA method, the distance among cells using the similar channel is following the rule of "*minimum reuse distance*" for that network. In dynamic channel allocation, as the calls arrive the channels are assigned not fixedly and this has a higher level of randomness but contains more complex algorithms. FCA is easier and out performs than DCA when there is a heavy traffic, but FCA doesn't adapt to shifting traffic circumstances [50].

2.11 Types of Handover Management Schemes.

Basic methods of handover prioritization methods are handover queuing schemes, call admission control (CAC), and guard channels (GC). At times some of these schemes are joined together to find better results [35]. The main goal of any handover technique is on how call dropping or the unexpected dropping probability can be minimized.

2.11.1 Handover Queuing Prioritization Schemes

Handover queuing scheme (HQS) allow either the handover to be queued or both the new calls and handover requests to be queued [42]. In HQPS, this method queues the handover calls when there is no available channels in the BSC. If one of the calls are finished and the frequency is dismissed in the BSC, then it is allocated to one of the handover call in the line. This method minimizes the unexpected termination probability at the cost of the augmented call setup failure rate. As [35] stated, Queuing is performing well when handover requests come in bunches and the traffic load is light. When handover requests occur uniformly, no need for queuing. The FIFO technique is one of the most common queuing methods.

Due to an overlap of regions in between the neighbouring cells, queuing handover is possible in which the MS can connect with more than one BS [51]. In HQS method when the RSS of the BSC in the under usage cell extends to certain predefined threshold then the call is queued from service a neighbouring cell. Originated call request is allocated to the channel frequency if the queue is unoccupied and if there is any of idle channel in the BSC [20].

2.11.1.1 Static Handover Queuing Schemes

The two most common policies used in the static priority queuing, that is: non-pre-emptive or pre-emptive. A non-pre-emptive queuing discipline requires a call that starts service until it finish its service without disruption. In a pre-emptive priority queuing, if a call entering at the queue, discoveries a call of lower priority in the system, then the upcoming call pre-empts the lower importance call in the queue and starts service directly. A pre-empted call will start again the service, at the point at which its service was hanged, once there are no higher urgency calls to be left in the line queue. A pre-emptive planning strategy that resumes call service is called pre-emptive resume [42]. Chang et al [52] presented a priority method that allows finite queuing of both handover and new calls, in two separate first in first out queues. Furthermore, the writer measured the refusing of originated calls and terminating of queued calls when they go out of the handover region, before the handover call is completed.

2.11.1.2 Dynamic Handover Queuing Schemes

Dynamic priority queuing for handover calls, considers the different network elements, and they dynamically rearrange the handover requests in the line to minimize the probability of dropped calls. Dynamic reordering is necessity, since the handover requests have to be lined in the queue in a way to influence and adapt to the dynamics of the subscriber motion (that is the change of his speed movement).

Jabbari and Tekinay [53] projected the Measurement-Based Prioritization Scheme (MBPS), where a handover call request in the line with the lowest signal received strength accesses the connection, when the call is ended and the channel is released. The implementation and the study results show that the FIFO queuing method has better efficiency in terms of Quality of service & capacity utilization.

2.11.2 Call Admission Control

CAC is a method that is used to maintain the Quality of service by estimating the originated call blocking rate and handover unexpected termination probability where average channel holding time is very important term to calculate this quality of service (QoS). These measurements may be specified in cellular systems that the handover call terminated rate is lower than 5%, for voice calls [54]. The CAC method [55], is responsible to decide and view whether originating call requests are going to be accepted into the system or not. This algorithm, the arrival of originated call is assessed constantly and when they are of a higher threshold than the advanced declared threshold point then the calls are cleared and rejected regardless if a frequency is free or full to minimize the rate of handover dropped calls. There is a transaction between the Quality of Service level supposed by the subscriber and the utilization of unusual network resources.

There exists various techniques and expressions to estimate these QoS parameters where following presumptions are generally utilized: and call arrival is a Poisson process [56], the channel holding time, and the call holding time those are supposed exponentially distributed, the cell residence time. In call admission control method both the handover calls and new calls have a permission to use all the available channels, and when the originated call cannot get a free channel then the call is rejected instantly [55].

2.11.3 Guard Channel Scheme (GChS)

The Quality of service is not satisfactory if the handover dropping rate is same as originated call blocking rate. In the GCh policy, static numbers of frequencies in each cell are allocated mainly to support handover calls. The GCs improve the probability of a successful handover because they take the highest priority by assigning a static or dynamically adjustable number of channels only for handovers among the total number of frequencies in the network, while the remaining channels can be shared equally by handover and originated calls [57]. The arriving calls can be categorized into two classes, the originated new calls, and the handover calls, which are handovered from the adjacent cells, as shown in Figure 2.8.

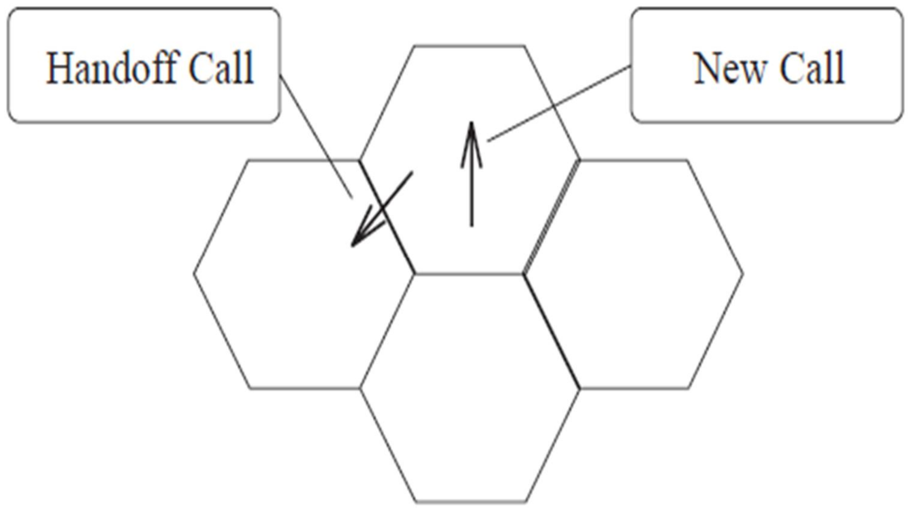

Figure 2.8: New call and handover call

A originated call is accepted if it discovers that the amount of the existing channels are less than the boundary between handover and originated calls, then a new call is accepted, or else it's rejected. On the other hand, a handover call is served if the is channel available, or else it's terminated and dropped. It is a familiar this kind of strategy that products in a lower call terminated probability. But, this enhancement is achieved at the cost of a slight increase in the originated call setup failure rate [58].

The choice of number of GChs only for handover call is basically significant factor to obtain good QoS. In different classes of traffic situations and movement factor, various numbers of GChs is wanted to be assigned. The number of GChs can't be static when the network traffic situation is varying with the time. Therefore the GChs allocation is dynamically changed by observing he traffic condition for certain time period [6].

It is worth to notice that to drop the handover call is more destructive than to block the new call, because the handover call is an existing and working call. To drop a handover call means to disconnect a communicating call. However, there is not a real method to evaluate the threshold value [48]. There are two GChs methods, static (fixed) and dynamic, as explained in this proposal.

2.11.3.1 Fixed guard channel scheme

In fixed GChs method, the real time calls have been given priority by reserving some guard channels for them [57]. In this scheme the higher priority calls such as voice and video calls have been given more priority than data. So by ignoring some resources of the lower priority classes the rejected call probability of the higher priority classes can be reduced. In this case the channel utilization falls. Because in this scheme if the number of higher priority traffic arrival rates are low; then some channels remain empty and these channels cannot be used by the lower priority traffic classes. It causes the reduced utilization of channels [59].The fixed GChs normally offers acceptable performance under nominal stationary traffic loads, but actual traffic loads are rarely fixed or have the same level as the nominal.

In static GChs method, a static number of GCh are reserved in every cell. Both handover and new calls are served if the number of occupied channels is less than C – GCh, else or, only the handover calls are served. Lastly, when there is no available or ideal channel, handover calls are definitely dropped [60].

Where

C = the total number of the available channels in that cell.

2.11.3.2 Dynamic Guard Channel Method

This method dynamically adapts the currently active channel access priority in a radio cell based on the current state information of the cell and its adjacent cells, since the channels are reserved for handover calls based on the current estimate of terminating probability of handover purposes. The handover probabilities between two neighbouring cells can be measured dynamically as the ratio of the number of handovers, from the first cell to the second cell, to the number of calls accepted in the first cell [61].

The use of GChs for handover calls needs a good cooperation between channel reuse efficiency and system performance. In reality, any channel reserved for handover calls plays a significant role in decreasing the efficiency of frequency reuse method. Even you could anxiously select a correct amount of guard frequencies, the use of static number of GChs can cause wasting the expensive channels; this is because the handover traffic depends on the how the mobile users are, which its movement and mobility are not always static. For instance, the movement of subscribers may be rising in the rural areas than in the big towns and cities because there is no traffic jam there. Addition to that, the time and location are an important factor which reflects the movement of subscribers.

To manage the reserved channels in better way, we propose allocating the number of GChs dynamically in a cell depending on the amount of calls which may arrive to this cell. So that when a new or handover call enters or evacuates a cell, the directly related base station notifies its neighbour BSs. A Base Station decides the numbers of GChs based upon the amount of active/ongoing calls in its neighbour cells as these calls are the only calls which can handover to this cell. The remaining mission is to determine how many GChs might be reserved, or in other words, as the movement and mobility of subscribers are varying, how many guard channels may be reserved to handover calls in the different cells?

With each BS aware the handover calls from each of its neighbouring cells, the DGC method works like this: When a call is admitted (either a handover call or a new call) in a chosen cell, say *x*, each of cell *x's* neighbouring BSs, say y, reserves a frequency channel, if any is a free for the call with a probability equal to the value of the handover probability from cell *x* to cell y [61]. Or we say, suppose there are *n* active/ongoing calls in the neighbour cells of cell Z. And that is fairly straight forward to reserve *n* channels in Z. Nevertheless, there is a possibility to discard the costly channels for these *n* calls may not all handover to Z. An instinctive way is to reserve the estimated amount of calls which may handover to Z. Mathematically we can calculate,

$$\sum_{x=1}^{b} n_{x} / b_{x} \qquad 2.10$$

Where

n*i*, is the amount of ongoing calls

bi, is the amount of neighbour cells of the cell x [60].

To overcome the low system performance by keeping the channels reserved for an unlimited time like the case in the fixed GChs method, each reserved channels is returned to the open channels pool in cell y by the network after a given time T [61]. The main advantage of this algorithm is its ability to work under dynamic traffic change.

CHAPTER THREE

RESEARCH METHODOLOGY

3.1 Introduction

This chapter reviews the methodology which was used in this study. The following topics will be discussed:

i. The study area
ii. Choice of the algorithm
iii. Dynamic guard channel model description
iv. Data Collection.
v. Materials for Case Study Simulation

3.2 The Study Area

Mogadishu is the capital city of Somalia. It was chosen because is a meeting point and movement point from one place to another. Having the highest population in the country, it is the second in demography ranking as the fastest growing cities in the world. It has a population of approximately 2.1 million with a growth of 6.9% [62].

Figure 3.1: Map showing the Study Area [63]

It is the main base of operation for where the major telecommunication companies are housed. Communications in terms of phones call service are much more spread compared to internet data services. Mogadishu is a location provisioned with all these services, making it a good location to conduct the research.

3.3 Choice of the algorithm

The region overall, experiences many dropped calls despite having up to seven service providers competing for the market. These calls are mainly the handover calls. Standards have been set by the Commination Authority of Somalia, but are still not being met by these companies in the region. There are several reasons why the region has a high number of dropped calls, but one of the reasons is network traffic congestion. From a basic study of their network, it has been established that the system under use can undergo further improvement without costing the companies much. This is done by completely utilizing the systems potential. To reduce the handover drop calls we can use handover prioritization schemes.

There are different handover prioritization techniques which can be used in prioritizing the handover so that handover drop rate can be reduced, such as: Handover Queuing Prioritization Schemes and call admission control (CAC). With this in mind there we came up with the most effective optimization algorithm so that reliable results are achieved. Most of these methods can guarantee lower handover call dropping probabilities but at the cost of high originated blocked calls. The reservation of channels can be static; referred to as *static guard channel,* with which drop calls can be minimized. On the other hand, this can cause wasting the expensive channels; this is because the handover traffic depends on the how the mobile users are, which is not always the case. Adding to that, Mogadishu is a densely populated city and also is a meeting point and movement point from one place to another, so this means that there is high inter-BTS handover requests taking place. On another hand, the journal article of [58] was another key one in selecting and implementing this algorithm.

Variable reservation scheme (DVR) algorithms [58].

Reservation
For each neighbor i
If a multiple of the threshold(0 is reached Then
A reservation of an eligible channel j for neighbor(i) is set in the pool
The channel j is blocked in every interfering cell with neighbor(i).

Getting in cell
If the number of free channels > 0 then
The communication is established
Reservation
Else
If it is a New Call Then
The communication is blocked
Else
If the number of reserved channels > 0 Then
The communication is established
Reservation
Else
The Handoff is dropped

Cancel a Reservation
For each neighbor(i)
If a multiple of the threshold is lost Then
A reservation is cancelled in the pool

Getting out cell

Cancel a Reservation

Based on that, we have developed a Guard channel scheme operating dynamically, so channels are reserved only when the system predicts it will need to have guard channels for some handover calls that may be dropped.

3.4 Dynamic guard channel model description

This method uses threshold to explain the present state of traffic. If channels occupied are less than the stated threshold, the load traffic is light; otherwise, the load traffic is heavy. The channels used in the cell are divided into two parts: the guard channels "for handover calls only" the shared channels" for both" as shown in the Figure 3.2.

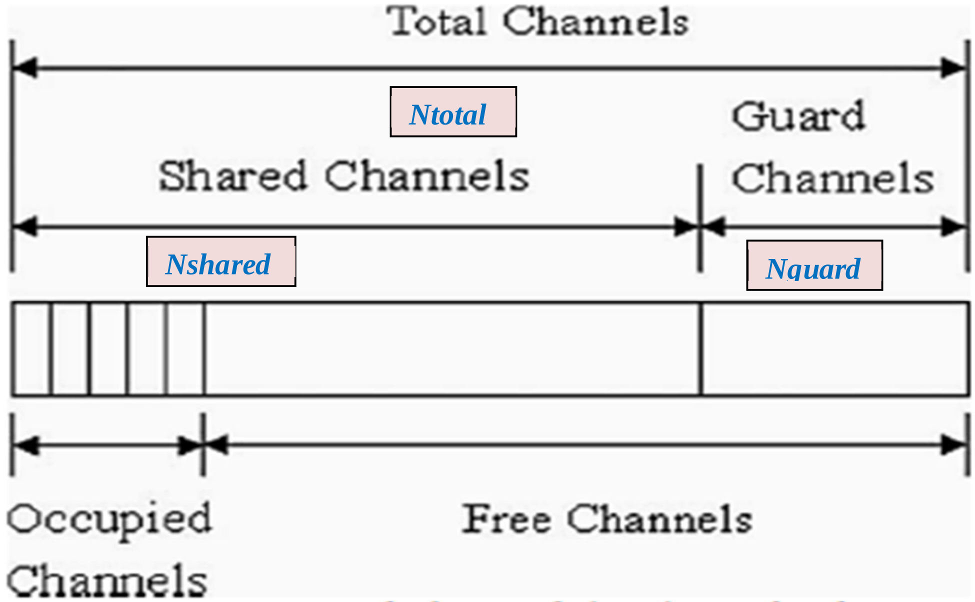

Figure 3.2: Channel Assignment structure with Priority for Handover Calls [48]

For instance, when new incoming calls can admit the idle channels limited by the selected number of remained channels, the incoming handover calls can then access all the available channels as illustrated in Figure 3.2. Moreover, Figure 3.2 shows how channels are allocated to new and handover calls and as well.

The model mathematically described as:

$$N_{total} = N_{shared} + N_{guard} \qquad (3.1)$$

$$N_{shared} = N_{total} - N_{guard} \qquad (3.2)$$

$$N_{guard} = N_{total} - N_{shared} \qquad (3.3)$$

Where

$\mathbf{N_{total}}$: The total amount of available channels.

$\mathbf{N_{shared}}$: The number of shared channels.

$\mathbf{N_{guard}}$: The number of GChs .

In summary, a handover call is accepted as if there is a free channel and rejected when there's no channel available. And new calls are rejected when availability of channels is less than shared channels.

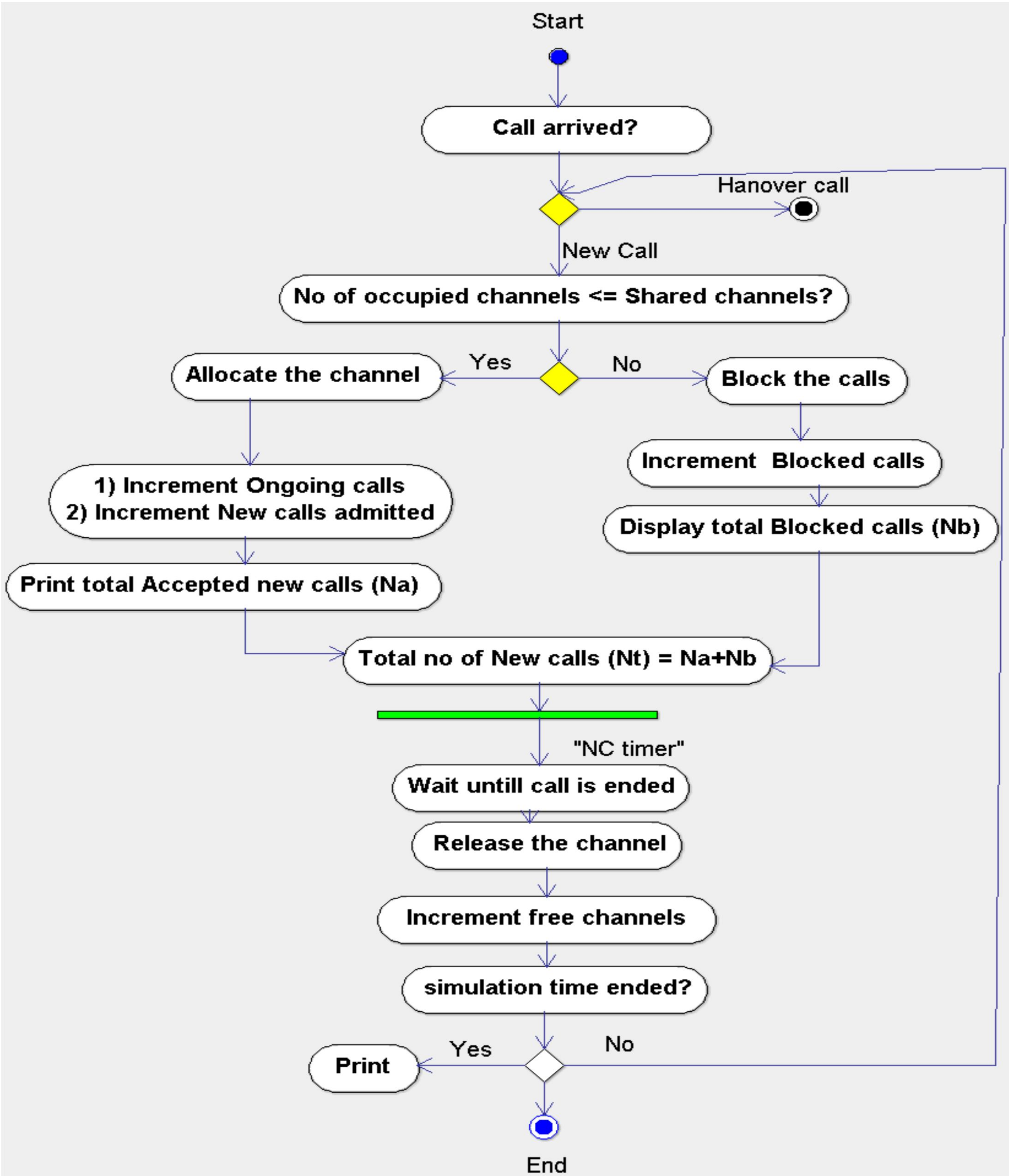

Figure 3.3: Flowchart for New Calls

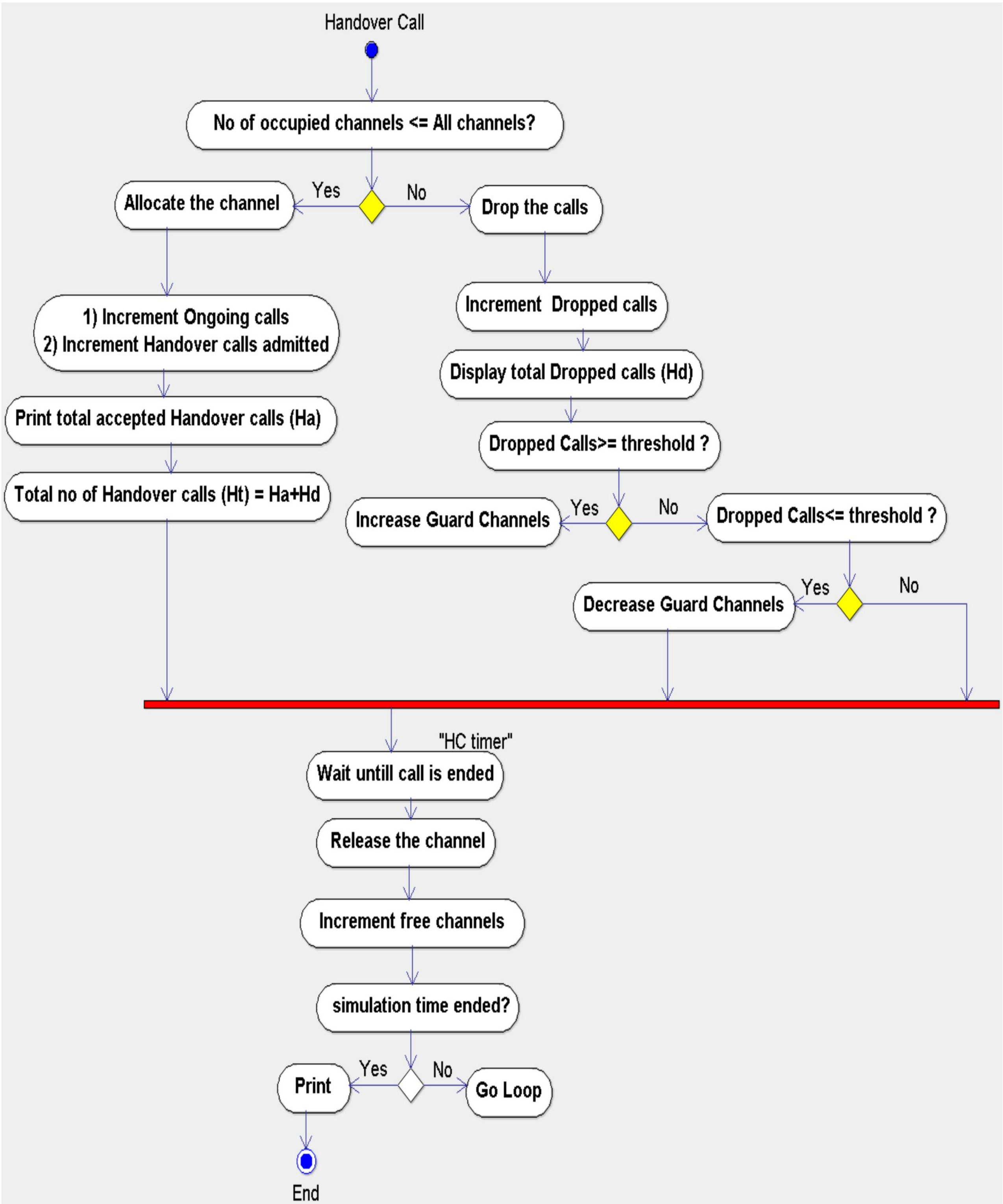

Figure 3.4: Flowchart for Handover Calls

3.4.1 Algorithmic Steps attained from the Flowchart

1. Enter number of channels and Simulation Time periods.

The program is built in a way that the user, needs to input time periods. These time periods represent the length of time the simulation will run. For example, if we enter the first time period as 2, we will expect the system to generate calls for those 2 minutes. The time periods can then be separated with comas, if we need to generate more data for an extended amount of time period. These values are compiled and the sum of all the values input will be the running time of the simulation. The next inputs are channels, and in here we can specify the number of channels we desire the system to have.

2. We initialize all the data we need, such as guard channels and channel holding time for both new calls and handover calls, and the number of calls the user willing to generate. Our system we initialized the system to generate a call in a second. "*Since we are not using the real system, and the cell can't know how many calls are ongoing in the neighboring cell in order to reserve we use upper and lower limits instead*". For example if we take the upper limit threshold = 0.07 lower limit threshold =0.06. The limits are the acceptable limits that the system should remain in. With these limits set, whenever the handover call dropping probability goes above the upper limit, the system automatically increases the number of guard channels otherwise is decreased. The maximum number of guard channels we can have is 9 and a minimum is 2.We initialized the new call holding time =3 seconds, and handover holding time= 3.2 seconds, then when a new call is made the call is programmed to run for 3 seconds for new call and 3.2 seconds for handover call, then the call is cut.

3. Here the system program generates a random number from 0, where each a random number will either represent a new call or handover call.

4. If the random number falls within the new call range, a new call will be generated in the system. That call is accepted if there are any free channels in the shared channel segment. If any is free the call will go through, else it will be blocked, will increment either blocked calls or accepted call in the cell.

5. If the call is a handover call, there will be a check if there is an empty shared channel available, if none is available, then because of priority we check if there are any guard channels that are free which the call can take. If any is available it will be accepted. If neither is available, then the call will be dropped. This will increment the dropped handover calls counter which is now used to calculate the percentage and probability that a new handover call will fail.

6. Finally we print out the calculation value from section 3.6.1“Performance indicators” calculated and evaluated

7. This cycle will proceed from number 3 above until the time period is over. .

3.5 Data Collection

The data used in this research was obtained from Somalia Telecommunication Company in Mogadishu. The data are collected from the period between 2 July, 2016 and 8 July, 2016, for 24 hours a day. The twenty-seven base stations (BSs) in the most populated regions in Mogadishu are investigated. The most important information gathered from Network Planning Operation (RNO) Section of Somalia Telecommunication Company in Mogadishu are: GCell, Traffic Volume on SDCCH (Erl), Traffic Volume on TCH (Erl), Call Drop Rate on SDCCH, TCH Seizure Success Rate, Call Drop Rate on TCH per cell (Excluding Handover), Call Drop Rate on TCH per cell (including Handover) and Handover Success Rate.

From the data we analysed the different patterns that were created, and we presented it in a various ways as charts and graphics. Through comparison to the Communication Authority of Somalia (CAS) standard of how a network should operate, we got the exact performance of the network operator. From that we extracted the information associated with handover calls and new originated calls.

3.6 Materials for Case Study Simulation

Dynamic guard channel method was implemented by using JAVA Software. We used NetBeans IDE 8.1 because it is a lightweight program requiring minimal specifications from the system it runs on. This allows it to be able to run in multiple devices while still maintaining its simplicity of use and operation of the same.

Dynamic guard channel method was implemented by using JAVA Software. We used NetBeans IDE 8.1 because it is a lightweight program requiring minimal specifications from the system it runs on. This allows it to be able to run in multiple devices while still maintaining its simplicity of use and operation of the same.

The Secondary Data was obtained in Microsoft Excel format; therefore, bringing up the need to make the data more readable visually by creating graphs and charts using Matlab. Matlab software is applied to represent the various sets of data obtained that was gathered from the region. With graphical representations we are able to easily compare the simulated results to the real data without much effort and be able to spot the obvious differences in the comparisons in data. In order to provide proof on the algorithm performance, performance indicators, in section 3.6.1 are used.

3.6.1 Performance Indicators

The performance of the presented algorithm (DGCM) is measured in terms of

Call Setup Failure Rate,

Call setup success rate,

Handover Call Dropping Probability,

New call blocking probability,

Traffic Load and Handover Success Rates.

3.6.1.1 New Call-Blocking Probability (NCBP)

Considering the utilization of cell GSM system resource, originated call is accepted instantly if the amount of an active calls in that cell, when its arrival is less than shared channels. If all the shared channels are full then the call is blocked. In a certain time of period, there will be a new calls originated in the test cell. So the NCBP is the fraction of total rejected originated calls to the whole new calls attempted.

$$NCBP = \frac{Nb}{Na + Nb} = \frac{Nb}{Nt} \qquad 3.4$$

Where Nb = total blocked (rejected) new originated calls

Na= accepted fresh calls

Nt= is the summation of Na and Nb.

3.6.1.2 Call Setup Success Rate (CSSR)

$$CSSR = \frac{Na}{Nt} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots.\ 3.5$$

The call setup success rate has direct impact on the clients' contentment due to service obtained by the network operators [21].

3.6.1.3 Call Setup Failure Rate

Similarly, HFR (handover failure rate) is calculated from equation (3.5) as:

$$HFR = 1 - HSR \ldots\ldots\ldots\ldots\ldots\ldots..3.6$$

3.6.1.4 Handover-Call Dropping Probability (HCDP)

Users are further sensitive towards handover call dropping than to call blocking at instigation, because the handover call already exists unlike an ongoing call. To drop a handover, call means to sever connections of an ongoing call. In general, the call terminating probability is a very significantly crucial connection Quality of Service parameter, since it characterizes the probability which a call is terminated as the result of the handover failure. The moment all available channels to the target cell are busy, then the call will be released and cleared out. Basically, a handover call dropping is initiated by lack of available channels in the targeted base station.

Similarly, in a certain time period, there will be a handover call experienced by the user. The dropped handover calls is denoted as". Then, the HCDP is calculated as the proportion of overall rejected handover calls to entire handover calls attempted.
Mathematically it showed that:

$$\text{Handover Call Dropping Probability}(HCDP) = \frac{Hd}{Ha + Hd} = \frac{Hd}{Ht} \ldots\ldots\ldots 3.7$$

Where Ha = accepted handover calls
"Hd= handover dropped calls
In this work what we interested is the point where the handover call is arrives to the system and the point where the handover call is ends
As [64] indicated, the minimization of drop call rate requires some methods to make handover requests at the right time and place based on the transmission situation.

3.6.1.5 Handover Success Rate

Then we calculate HSR (Handover Success Rate) which mathematically described as:

$$HSR = \left(\frac{Ha}{Ht}\right) * 100\% \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots. \quad 3.8$$

Handover Success Rate is the number of successfully completed handover calls divided by the total amount of initiated handovers [23].

3.6.1.6 Traffic Load

The traffic load is calculated from mean holding time (α) or service time [65]and the number of calls intensity (λ). But here we calculate as:

$$\text{Traffic Load} = \frac{Na + Ha}{Nt} \ldots\ldots\ldots\ldots\ldots\ldots \quad 3.9$$

Where Nt = Total number of calls processed

Table 3.1: Simulation parameters

total_available_channels	Number of channels entered by the user
number_of_periods	Number of simulation periods the user entered
reserved_channels	Number of guard channels, reserved for handover calls only
shared_channels	Channels that are shared both new and handover calls
Type_of_Call	To identify either the call is new generated call or handover call
ongoingCalls	Number of ongoing calls
totalNo_ofOriginated_calls	Number of new incoming new calls
incomingHanodverCalls	Number of new incoming handover calls
OriginatedCalls	Number of new originate calls
CompletedHandoverCalls	Number of handover calls completed
CompletedNewCalls	Number of New calls completed
no_ofOriginated_calls_accepted	Number of new calls successfully admitted
no_ofOriginated_calls_rejected	Number of new calls blocked
handover_calls_accepted	Number of handovers successfully admitted
handover_calls_dropped	Number of handover calls rejected
totalNo_ofOriginated_calls	Number of all new calls admitted and rejected
totalNo_ofHandover_calls	Number of all handovers admitted and dropped

all_attempted_calls	Total number of calls processed
call_generation_time_delay	Time taken for a call to generate
newCall_timeDelay	Time taken for a new call
Handovercall_timeDelay	time taken for a handover call
timePeriod	time taken for a simulation
Originated_call_blocking_prob	New Call Blocking Probability
handover_call_dropping_prob	Handover Call Dropping Probability
probable_increase	probability of increase of the guard channels
Successrates	Call setup success rate
handover_success_rate	The rate of successfully completed handovers
traffic_load	Channel Utilization
dynamic_guardChannel_algorithm	Dynamic Guard Channel Scheme

CHAPTER FOUR
RESULTS AND INTERPRETATION

4.0 Introduction

In the first section, we analysed the collected data. In the second section we analysed simulation results.

4.1 Data Analysis and Interpretation

From the data collected from Network Planning Operation (RNO) Section of Somalia Telecommunication Company in Mogadishu, in the period between 2nd July, 2016 and 8th July, 2016, the averages for each cell running 24 hours a day were investigated as shown in the Appendix1 not included here. The twenty-seven different base stations (BSs), in the most populated regions in Mogadishu were investigated. From appendix 1, we calculated the average of the most important and relevant indicators and presented them in Table 4.1**.**

Table 4.1: Average of the real data QoS indicators

S/N	GCELL	Call Setup Success Rate (%)	Call Drop Rate (%)	Handover Success Rate (%)	Call Block Rate (%)	Handover Failure rate (%)
1	B_BAKAARAHA_D	100	0.267	98.435	0	1.565
2	B_BAKAARAHA_E	98.321	0.506	96.286	1.679	3.714
3	B_BAKAARAHA_F	98.061	0.482	96.864	1.939	3.136
4	B_HQ BUILDING_D	99.306	0.381	96.586	0.694	3.414
5	B_HQ BUILDING_E	99.829	0.379	97.456	0.171	2.544
6	B_HQ BUILDING_F	96.452	1.247	90.808	3.548	9.192
7	B_S.CAANAHA_A	99.091	0.149	97.379	0.909	2.621
8	B_S.CAANAHA_B	98.241	1.113	96.207	1.759	3.793
9	B_S.CAANAHA_C	98.462	1.383	92.495	1.538	7.505
10	B_S.CAANAHA_D	99.227	0.767	96.033	0.773	3.967
11	B_S.CAANAHA_E	98.346	0.351	97.322	1.654	2.678

12	B_S.CAANAHA_F	99.183	0.598	95.553	0.817	4.447
13	B_QARAN_D	99.424	0.467	97.652	0.579	2.348
14	B_QARAN_E	99.246	0.397	95.888	0.754	4.112
15	B_QARAN_F	99.488	0.366	97.812	0.512	2.188
16	B_DAHABLAHA_D	97.926	1.877	94.633	2.074	5.367
17	B_DAHABLAHA_E	98.899	0.888	94.422	1.101	5.578
18	B_DAHABLAHA_F	99.161	1.148	94.766	0.839	5.234
19	B_SARIF YARAHA_D	98.362	1.764	96.733	1.638	3.267
20	B_SARIF YARAHA_E	99.356	0.373	96.544	0.644	3.456
21	B_SARIF YARAHA_F	98.239	1.414	95.242	1.761	4.758
22	B_CIRTOOGTE_D	98.008	0.905	96.403	1.992	3.597
23	B_CIRTOOGTE_E	99.431	0.685	97.017	0.569	2.983
24	B_CIRTOOGTE_F	98.402	0.868	96.806	1.598	3.194
25	B_GEED JACEYL_D	100	0.745	97.789	0	2.211
26	B_GEED JACEYL_E	99.558	0.389	96.763	0.442	3.237
27	B_GEED JACEYL_F	98.346	1.545	96.149	1.654	3.851

In order to best assess the performance of the cells, we used the Communications Authority of Somalia's KPIs standards as in Table 4.2.

Table 4.2: Communication Authority of Somalia (CAS) KPI Benchmark

Quality of Service Parameter	**Target Value**
Call Setup Success Rate	>=98%
Call Drop Rate	<=2%
Handover Success Rate	>=98%
Handover Failure Rate	<=2%
New Call Blocking Probability	<=0.02
Handover Call Dropping Probability	<=0.02

4.1.1 Call Setup Success Rate

This parameter considered one of the most important KPIs because it reflects the consumer's satisfaction and ability to retain their subscribers. Figure 4.1a and 4.1b illustrates the results of the collected real data.

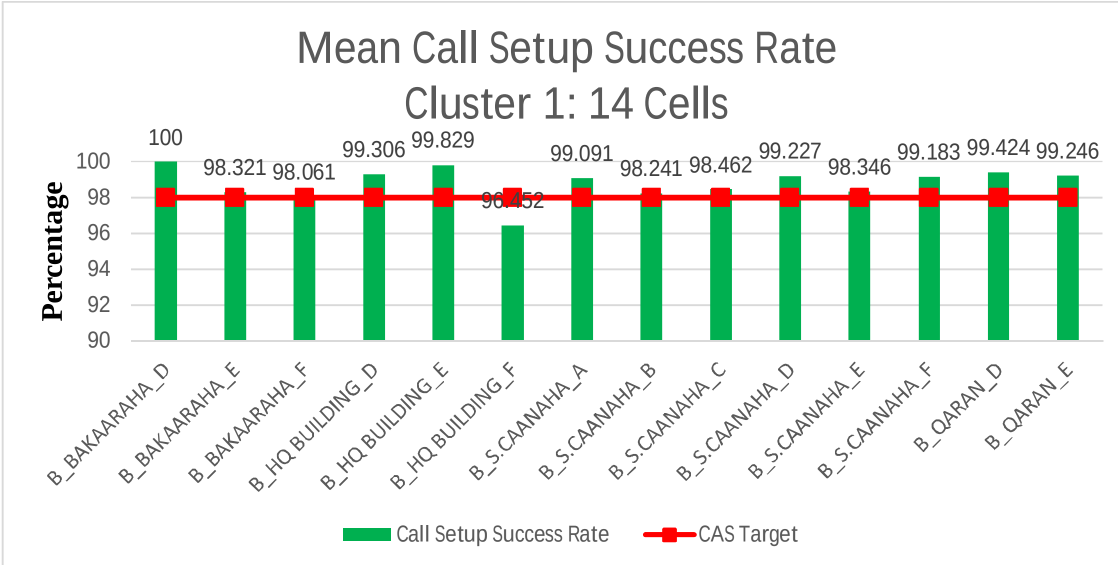

Figure 4.1a: Call Setup Success Rate of 14 cells with Communications Authority of Somalia target

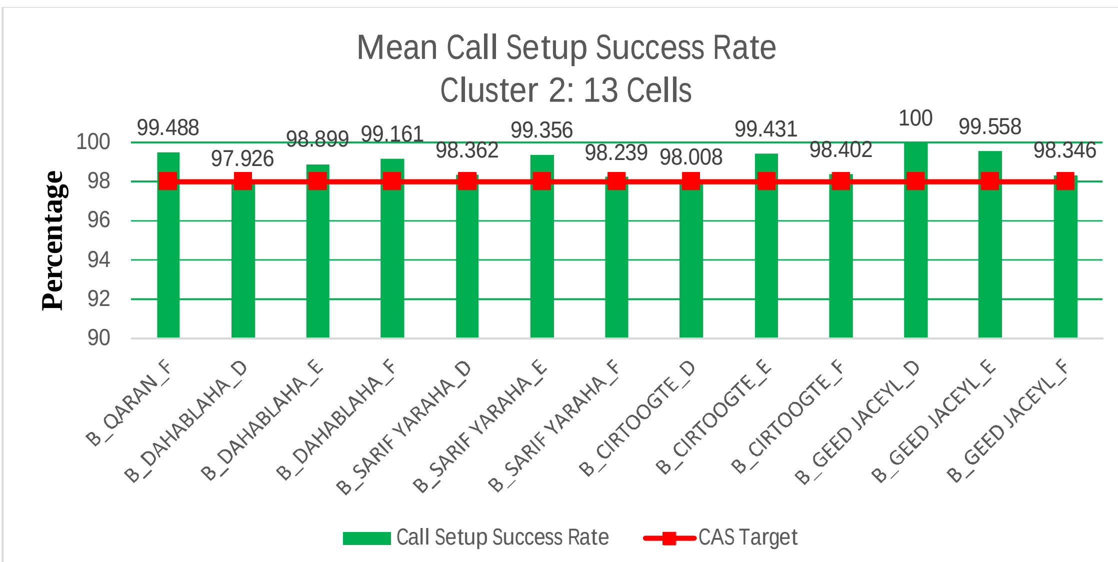

Figure 4.1b: Call Setup Success Rate of 13 cells with Communications Authority of Somalia target

The first and second group of 14 and 13 all met the Communications Authority of Somalia target of 98% except cell B_HQ BUILDING_F and B_DAHABLA_D. This means customers on a cell B_HQ BUILDING_F and cell B_DAHABLA_D try many times on particular times to establish a connection on the network compared to customers on other cells.

4.1.2 Call Drop Rate

CDR measures the network capability to hold ongoing calls when it has been established or setup. A dropped call is a call that has been dismissed early before being released usually by either the called party or the caller. Call drops may be as a result of poor network coverage. The CDR is the ratio of the number of dropped calls to the total sum of call tries. Fig. 4.2a and 4.2b provides graphical representation results of the collected real data.

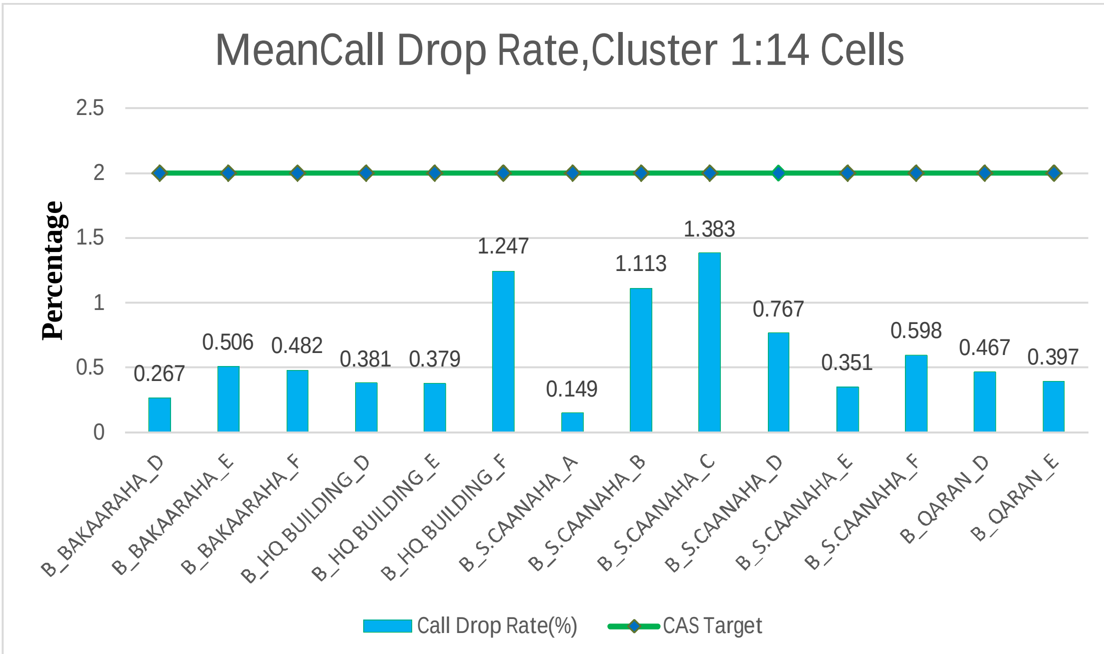

Figure 4.2a: Call Drop Rate of 14 cells with Communications Authority of Somalia target

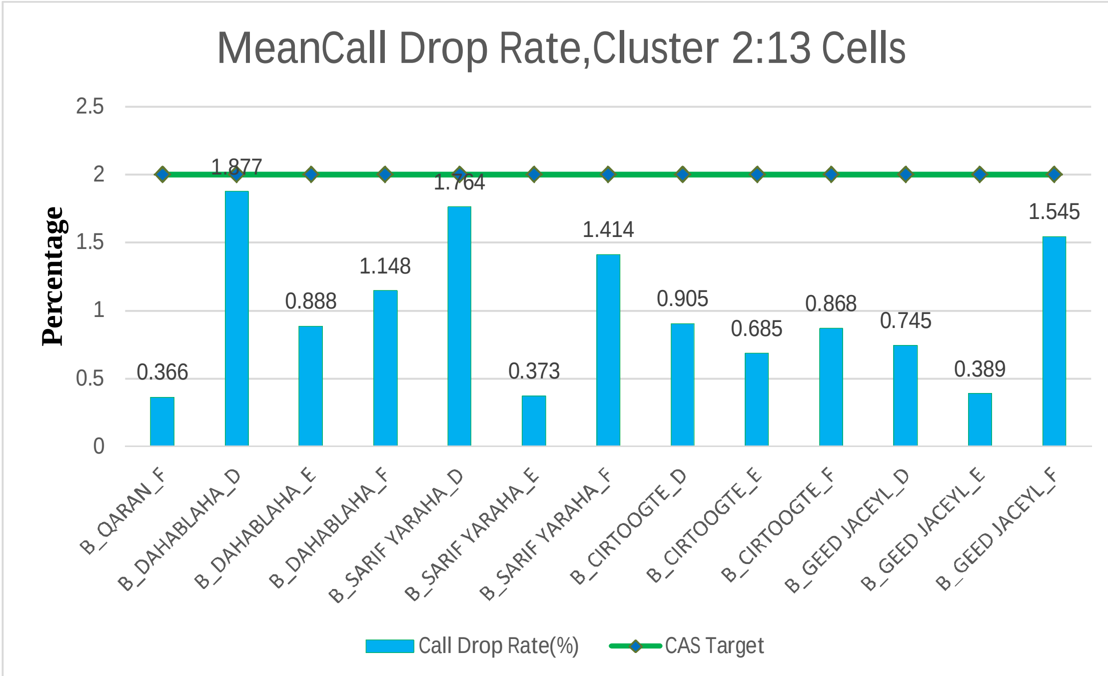

Figure 4.2b: Call Drop Rate of 13 cells with Communications Authority of Somalia target

The requirement for dropped calls from CAS is such that operators should ensure that at not more than 2% of all calls are dropped. From the figure 4.2 we can see all cells met the less than 2% target in both clusters combined.

4.1.3 Call Block Rate

This parameter measures calls that are unsuccessful because of lack of resources for connection due to congestion expressed as a percentage of total call attempts. Figure 4.3a and 4.3b illustrates the results of the collected real data.

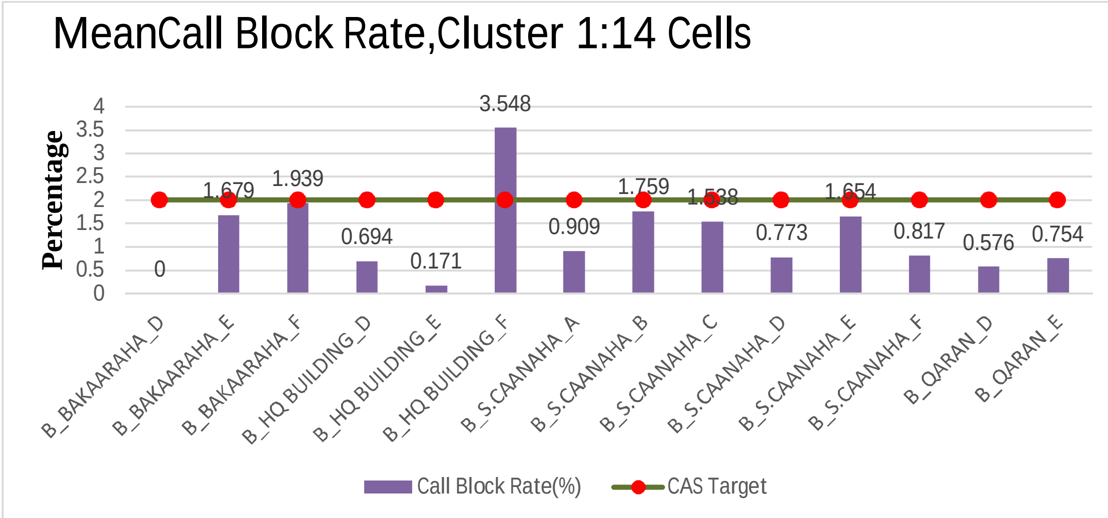

Figure 4.3 a: Call Block Rate of 14 cells with Communication Authority of Somalia target

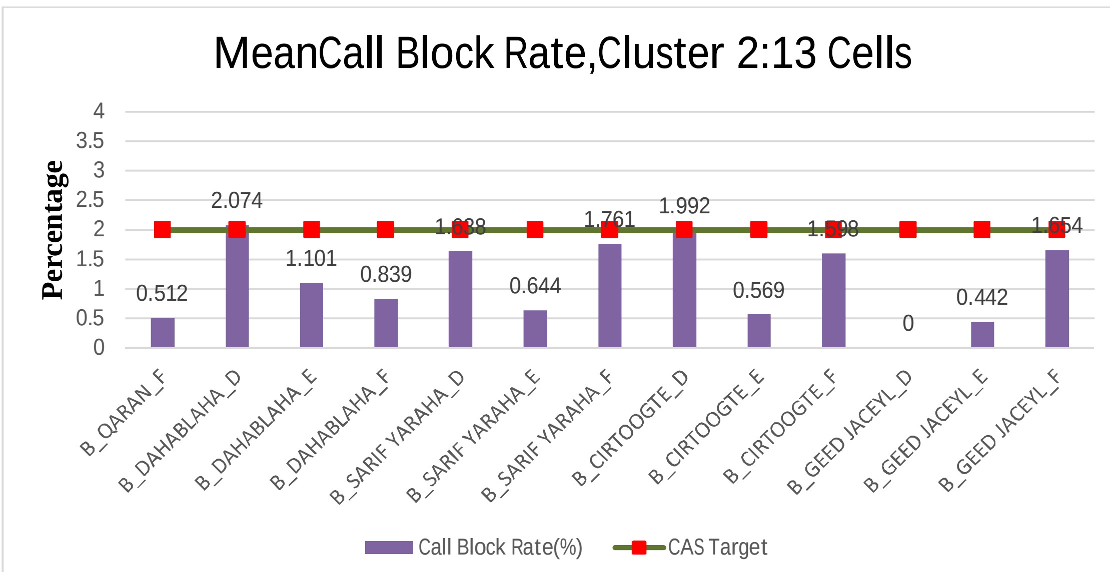

Figure 4.3b: Call Block Rate of 13 cells with Communication Authority of Somalia target

The requirement for blocked calls from Communications Authority of Somalia is such that operators should ensure that at not more than 2% of all calls are blocked.

One out of fourteen cells met the minimum target value of 2% in Cluster 1, whereas only B_DAHABLA_D was not able to achieve CAS requirements in Cluster 2. This can be as a result of TCH Failure Assignment, Hardware Problem, SDCCH Congestion, CM Service Reject and Low Signal Strength.

4.1.4 Handover Success Rate

This parameter is used to measure the percentage of successful accepted handover calls by mobile subscribers during a voice call. HSR is the number of successfully handover calls completed divided by the sum of the handover attempts. Figure 4.2a and 4.2b illustrates the results of the collected real data.

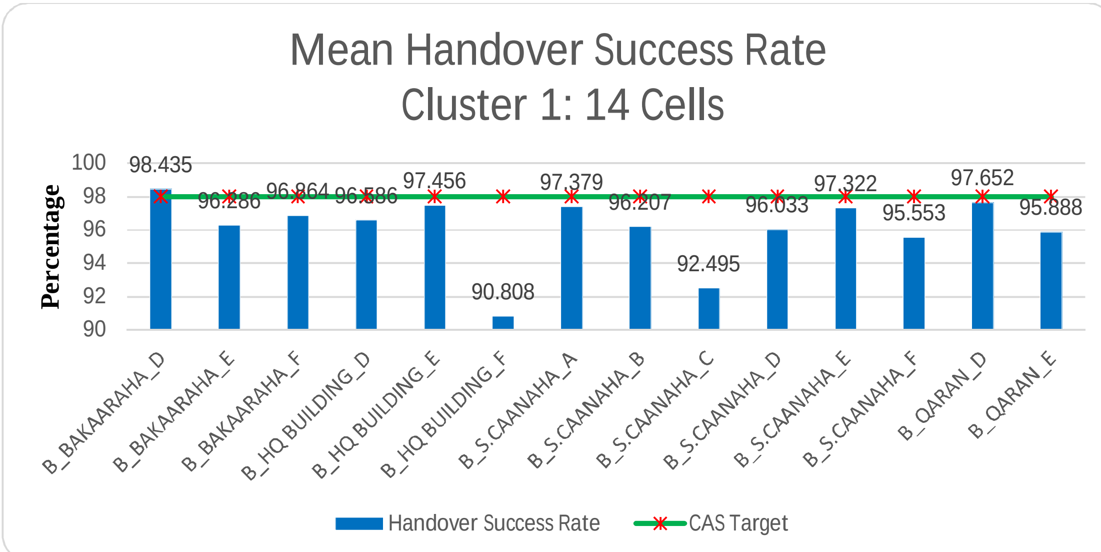

Figure 4.4a: Handover Success Rate of 14 cells with Communications Authority of Somalia target

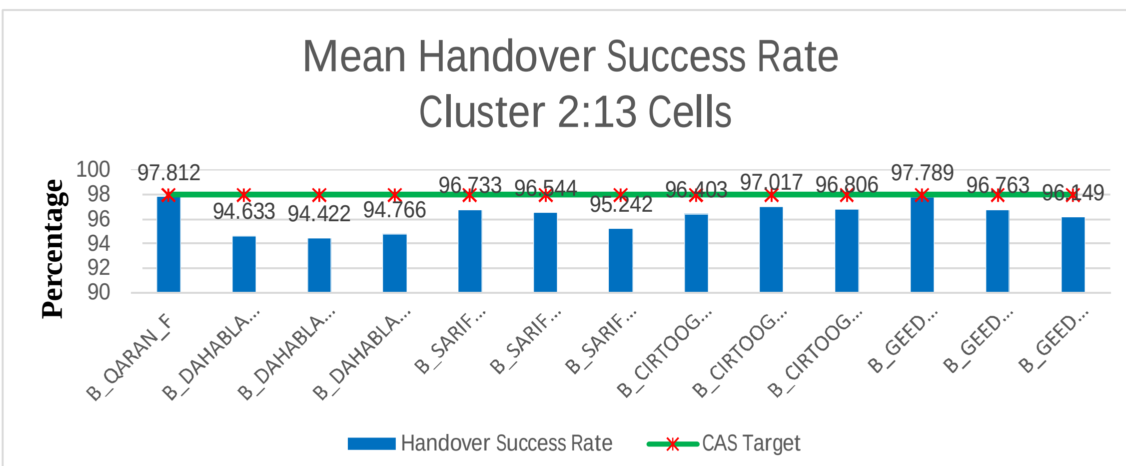

Figure 4.4b: Handover Success Rate of 13 cells with Communications Authority of Somalia target

The requirement for accepted handover calls from CAS is such that operators should ensure that at not less than 98% of all calls are admitted. For a normal running system, a handover is initiated whenever a mobile subscriber shifts from one cell area to another cell area and unless the Signal strength is below 102dB or all the channels are occupied. In the cluster 1 of 14 cells, we can see only B_ISG.BAKAARAHA_D, has achieved the CAS minimum target requirement of 98% success. In the Cluster 2 of 13 cells, no Cell is able to achieve this recommended minimum target. The figure 4.4 visually show the handover performance with the green line showing the CAS minimum target. This means that 96.3% of the cells fell below the intended target, which clearly shows how the service qualities provided by the GSM operators in Somalia remained extremely bad .and as we aware, if the rate of handover dropped calls increases, the subscriber becomes annoyed and dissatisfied and sooner or later he/she moves to another network. Therefore this system needs to be optimized.

4.1.5 New Call Blocking Probability and Handover Call Dropping Probability

Through equations (3.4) and (3.7), NCBP and HCDP are calculated from table4.1 and presented as shown in Table4.3.

Table 4.3: New Call Blocking Probability and Handover Call Dropping Probability

S/N	GCELL	Handover Call Dropping Probability	New Call Blocking Probability
1	B_BAKAARAHA_D	0.016	0
2	B_BAKAARAHA_E	0.037	0.017
3	B_BAKAARAHA_F	0.031	0.019
4	B_HQ BUILDING_D	0.034	0.007
5	B_HQ BUILDING_E	0.025	0.002
6	B_HQ BUILDING_F	0.092	0.035
7	B_S.CAANAHA_A	0.026	0.009
8	B_S.CAANAHA_B	0.038	0.018
9	B_S.CAANAHA_C	0.075	0.015
10	B_S.CAANAHA_D	0.039	0.008
11	B_S.CAANAHA_E	0.027	0.017
12	B_S.CAANAHA_F	0.044	0.008
13	B_QARAN_D	0.023	0.006
14	B_QARAN_E	0.041	0.008
15	B_QARAN_F	0.022	0.005
16	B_DAHABLAHA_D	0.054	0.021
17	B_DAHABLAHA_E	0.056	0.011
18	B_DAHABLAHA_F	0.052	0.008
19	B_SARIF YARAHA_D	0.033	0.016
20	B_SARIF YARAHA_E	0.035	0.006
21	B_SARIF YARAHA_F	0.048	0.018
22	B_CIRTOOGTE_D	0.036	0.019
23	B_CIRTOOGTE_E	0.029	0.006
24	B_CIRTOOGTE_F	0.032	0.016
25	B_GEED JACEYL_D	0.022	0
26	B_GEED JACEYL_E	0.032	0.004
27	B_GEED JACEYL_F	0.039	0.017

4.2 Performance Analysis of the simulated results

In this section, we analyzed the simulated results. In the simulated environment, we managed to create a similar system that mimics the originals working and had it generate a similar values from different tweaks. To show the efficiency of the dynamic guard channel algorithm, performance indicators such as handover success rate, handover call dropping probability, originated call blocking probability and channel utilization are generated and compared. New originated and handover calls need to be analyzed separately, because handover call dropping probability has a much greater impact on QoS than the probability of originated call blocking. Also new originated calls have low sensitivity to delay than a handover calls because handover calls need continuity for voice and message transfer [10]. Moreover, for the purpose of analyzing the performance of our algorithm, with a holding time of 3 seconds and 3.2 seconds per call for new originated calls and handover calls respectively, an additional 2 guard channels are factored in.

In Table 4.4 we see the simulation results generated on a system that was created to mimic the original system that has been deployed in Somalia. From this we can see how calls are generated, processed and all the details that come attached to it. Next we run a similar system without the use of any guard channels, and is tabulated in Table 4.5.

Table 4.4: Average Simulated results

No of Channels	Simulation Time (Min)	No of Calls	New Calls	Handover Calls	Call Setup Success Rate (%)	Handover Success Rate (%)	New Call Blocking Probability	Handover Call Dropping Probability	Channel Utilization
120	1	50	34	16	100	100	0	0	100
120	2	112	72	40	100	100	0	0	100
120	3	172	128	44	94.77	100	0.0523	0	94.767
120	4	233	176	57	81.55	100	0.1845	0	81.545
120	5	294	227	67	75.85	99.66	0.2415	0.0034	75.510
120	6	355	275	80	69.86	99.44	0.3014	0.0056	69.296
120	7	415	284	131	65.31	98.55	0.3469	0.0145	63.855
120	8	475	367	108	62.53	99.37	0.3747	0.0063	61.895
120	9	535	381	154	62.62	98.88	0.3738	0.0112	61.495
120	10	596	437	159	60.24	99.66	0.3976	0.0034	59.899
150	11	652	461	191	62.88	98.62	0.3712	0.0138	61.503
150	12	712	493	219	60.95	98.03	0.3905	0.0197	58.988
150	13	772	523	249	59.59	97.67	0.4041	0.0233	57.253
150	14	832	592	240	58.29	98.80	0.4171	0.0120	57.091
150	15	893	469	244	58.23	98.81	0.4177	0.0119	57.11

Table 4.5: Non-priority Scheme simulated values

No of Channels	Simulation Time (Min)	No of Calls	New Calls	Handover Calls	Call Setup Success Rate (%)	Handover Success Rate (%)	New Call Blocking Probability	Handover Call Dropping Probability	Channel Utilization
120	1	50	35	15	100	100	0	0	100
120	2	111	84	27	100	100	0	0	100
120	3	172	125	47	90.12	97.68	0.0988	0.0232	87.791
120	4	233	162	71	85.84	90.13	0.1416	0.0987	75.965
120	5	293	210	83	81.57	93.52	0.1843	0.0648	75.085
120	6	353	257	96	79.33	91.51	0.2067	0.0849	70.821
120	7	414	286	128	77.54	89.37	0.2246	0.1063	66.908
120	8	476	333	143	72.69	90.34	0.2731	0.0966	63.025
120	9	513	390	145	71.97	90.84	0.2803	0.0916	62.803
120	10	595	434	161	71.09	91.93	0.2891	0.0807	63.025
150	11	652	460	192	76.23	88.19	0.2377	0.1181	64.417
150	12	713	526	187	72.65	90.74	0.2735	0.0926	63.394
150	13	773	554	219	73.35	88.09	0.2665	0.1191	61.449
150	14	834	611	223	70.02	90.89	0.2998	0.0911	60.991
150	15	894	627	267	70.92	89.49	0.2908	0.1051	60.402

4.2.1 Handover Call Dropping Probability

Handover call dropping probability shows the probability that a new initiated handover call will be dropped. This value should be as small as possible an average of 0.02. The smaller the number is, the better the networks' quality of service would be, for both Handover call dropping and new call blocking probability.

Figure 4.5 shows the comparison between real handover call dropping probability and simulated handover call dropping probability. It's clear from the figure that the proposed algorithm outperforms and reduces the dropping probability of handover calls in comparison to the real data. The requirement for dropped calls from CAS is such that the operators should ensure that not more than 2% of all calls are dropped. In the real data, 96.3% of all calls was not able to achieve CAS requirements. Whereas in the simulated data, only one out of fifteen, produces an anomaly that puts it above the values recommended by the CAS. This means that we reduced this percentage from 96.3% down to 6.7%, which when projected to a larger system puts the System within acceptable range. The aim of the CAS is to ensure that the customer receives the best services by enforcing the rules and having telecommunication companies stick to certain levels of quality of service.

To prove that our algorithm efficiently utilizes the available resources, we intentionally run our system without the use any guard channels, which has been reserved only for handover calls, as shown in figure 4.6. We can observe from the figure below, the use of 15 times sampling, we found that only two time periods within the system set at 15 minutes (13.3% of calls) met the minimum target recommended by the CAS.

Based on all of that, we can truly say that the use of guard channel in a dynamic manner is a revolutionary solution to a system not utilizing it, in order to increase efficiency of the system while still being able to give a better quality of service. In comparison to the real data the rate of accepting handover calls substantially raises up by an 89.6% increase.

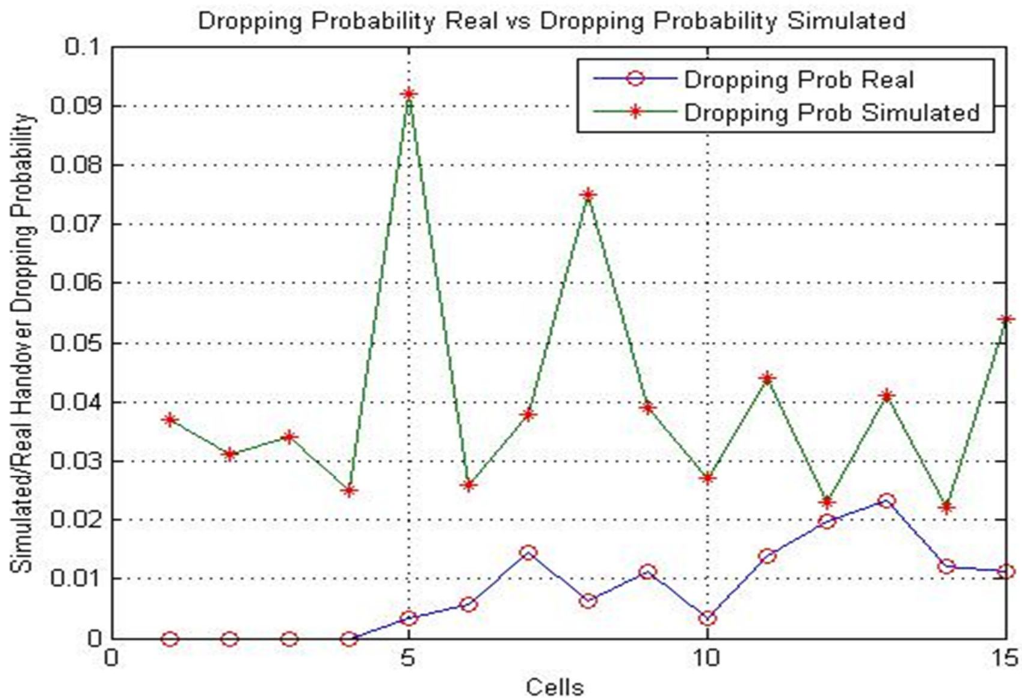

Figure 4.5: Real handover call dropping probability vs simulated handover call dropping probability

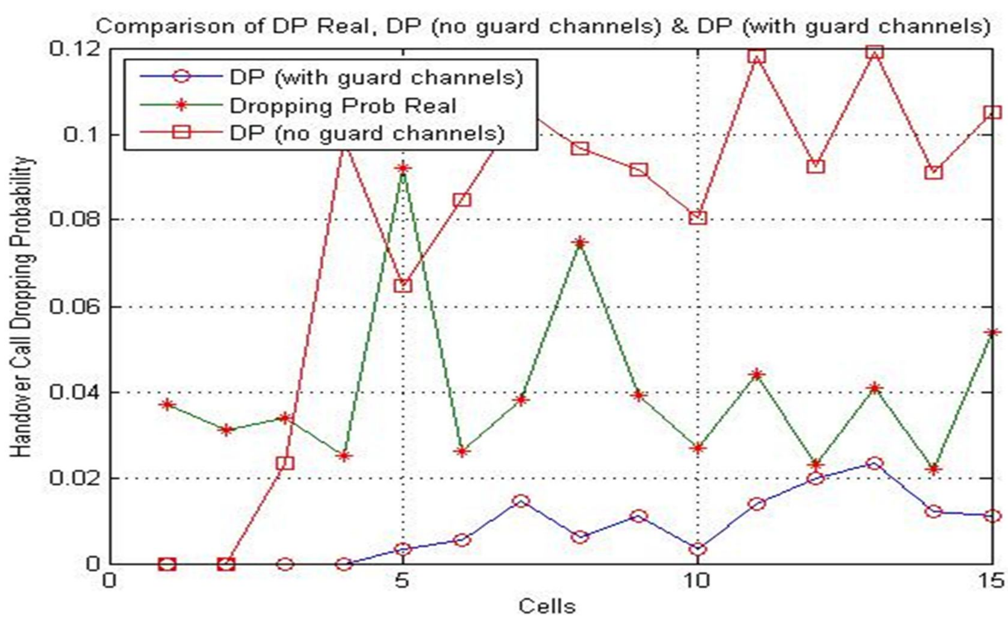

Figure 4.6: Comparison of handover dropping probability real, handover dropping Probability simulated (with guard channel) and non-priority scheme handover dropping Probability simulated

It’s clear from the figure 4.7 that the number of handover calls gradually increased with time, but handover call dropping probability is maintained between 0.0 and 0.02**.** It was found that even though handover calls are increased almost three hundred, still handover call dropping probability is at the acceptance level. It is demonstrated in Figure 4.7, that the number of handover calls dropped is much less in number even during peak times.

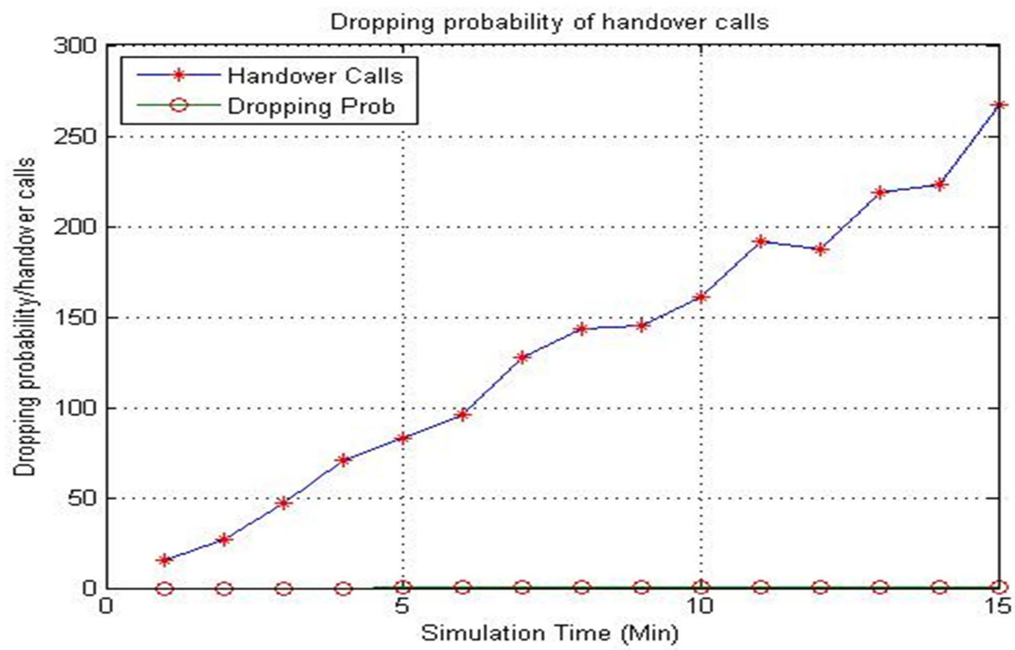

Figure 4.7: Handover call dropping probability

4.2.2 Comparing handover success rate real to simulated handover

Figure 4.8 represents the graphical comparison between the real data handover success rate and simulated data handover success rate. From the figure we can observe that the system for the real data isn’t efficient enough to keep the data within the recommended ranges (<=98%) provided by the CAS. It puts the real data at a 96.3% loss, having handover calls failing most of the time. With the idea that a handover call is more important than a new call, handover call being an ongoing call would really be upsetting if it were to be dropped. Whereas for a new call to be blocked to access the service won’t have the same effect. For the simulated data, it is noticeable that the handover calls have a higher acceptance rate than the real one. The red line represents the CAS recommended limit, this shows that the for handover calls, all the data sets are within range, except one, which represents only 6.7% of all handover calls processed, has exceeded the line. And that directly means less and less customers being dissatisfied with the service.

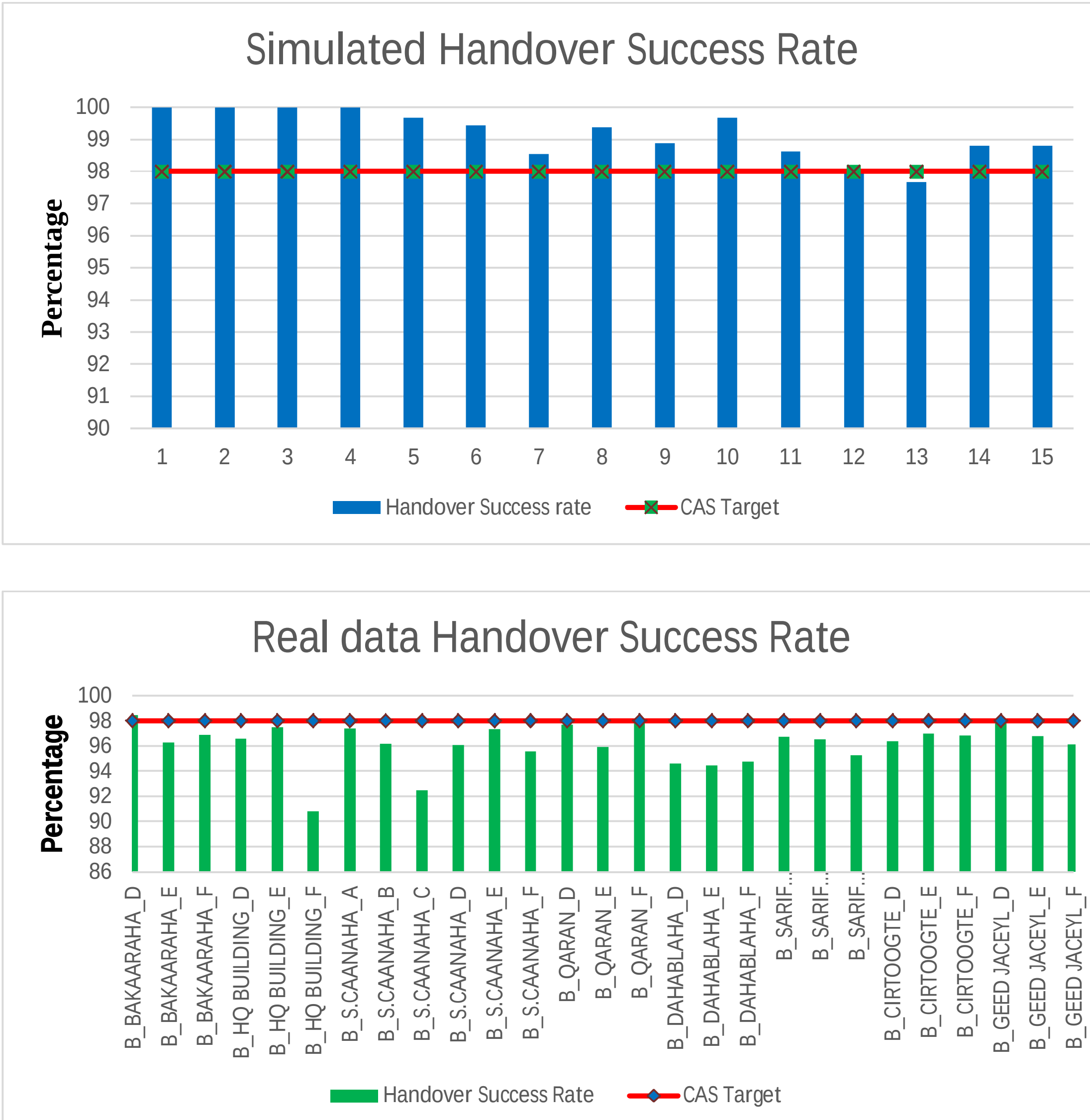

Figure 4.8: Comparison between handover success rate simulated and handover success rate real

4.2.3 New call blocking probability

The dynamic guard channel method reduces significantly, the blocking probability in comparison to the static guard channel algorithm due to the limited time reservation of the channels, this means that the channel will get wasted if all the assigned channels neither in use nor freely available at that moment bandwidth won't be allotted. The call blocking probability shows the possibilities in which a new originated phone call may be dropped before it is admitted to the system. Each base station is different from the next and this probability should be kept as low as possible. It's observed from Figure 4.9, when the number of calls increases then the chance of getting a channel by the originated calls decreases and this results in more new call blocking probability taking place. In order to maintain a blocking probability smaller than this simulated result, the arrival rate of originated calls would have to be less.

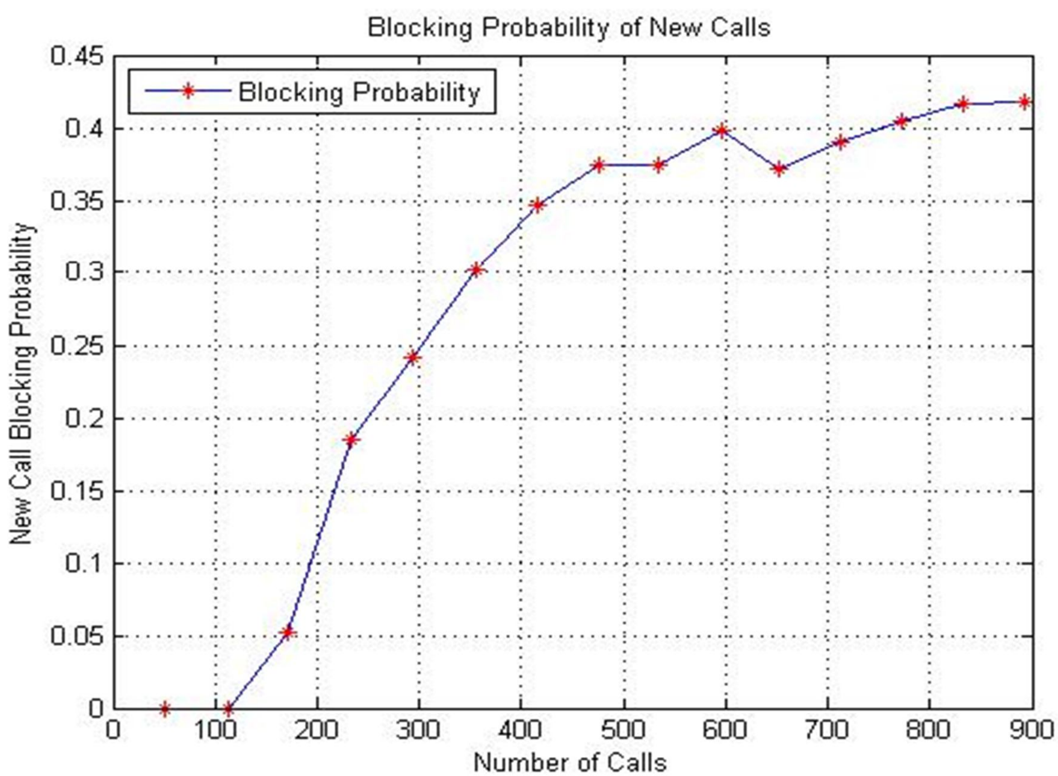

Figure 4.9: New call blocking probability

4.2.4 Comparing New Call Blocking Probability to handover call dropping probability

Figure 4.10 shows the comparison between new call blocking probability and handover call dropping probability. From the Figure, the more the arrival of calls increases, the more blocking probability occurs. This means that the chance of fresh originated calls allocating the channels successfully is reduced because a request for handover calls blocks the channels. Another factor affecting the performance of the system is what is called channel holding time. As it's proved in [56], when average channel holding time of handover call is greater than the average channel holding time of new call there is no effect in new call blocking probability. On the other hand, when the average channel holding time of new calls is greater than the average channel holding time of handover call there is a remarkable in new call blocking probability estimation. It can also be seen when new originated calls arrival rates are low compared to handover calls, handover call dropping probability is zero, because more number of channels are ready to handle handover calls.

Finally, as we mentioned and proved earlier, our algorithm has a better utilization than non-propitiation scheme (no guard channels), because it's overall performance was not satisfying both new calls and handover calls. Looking at the data from the simulation output, it is more controlled and predictable. When the counter starts, it shows there is complete utilization of the available resources first, then it presents a substantial increase as the number of calls increase, and even with so many call being generate, it is able to maintain a constant where the system stabilizes. So it's straight forward to say that; our algorithm is capable of confining the handover drop rate below the CAS target all while enhancing the new call blocking rate within acceptable level.

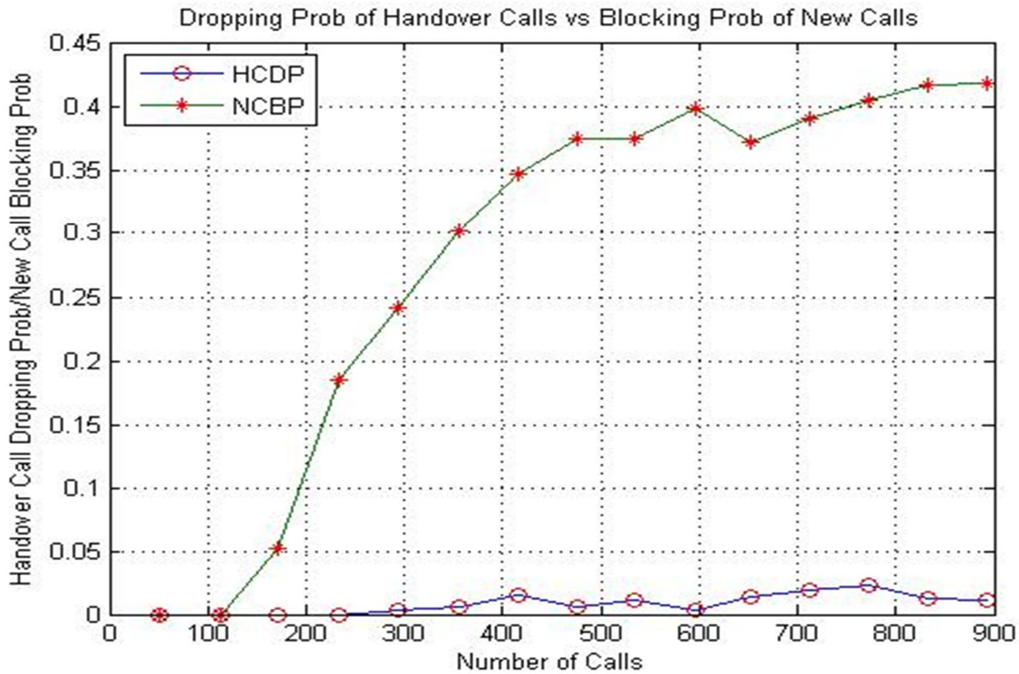

Figure 4.10: New call blocking probability vs handover-call dropping probability

4.2.5 Channel utilization

Channel Utilization is able to monitor the general and overall performance of a network and how the resources are being utilized and if there is too much traffic it affects the overall experience that a user gets. In Figure 4.11, its observable with the increment of new call admittance rate the channel utilization is increased. Generally it is also noticed that acceptance rate below fifty percent has a propensity to lessen channel utilization because of a slight drift to increase the NCBP. New call acceptance rate of any greater than 60% has satisfied channel utilization. Our algorithm shows a nearly constant value of channel utilization regardless of the variant of mobility and the time as presented in Figure 4.12, and the use of dynamic GCS can achieve a better utilization of the channels since the number of guard channels is dynamically assigned.

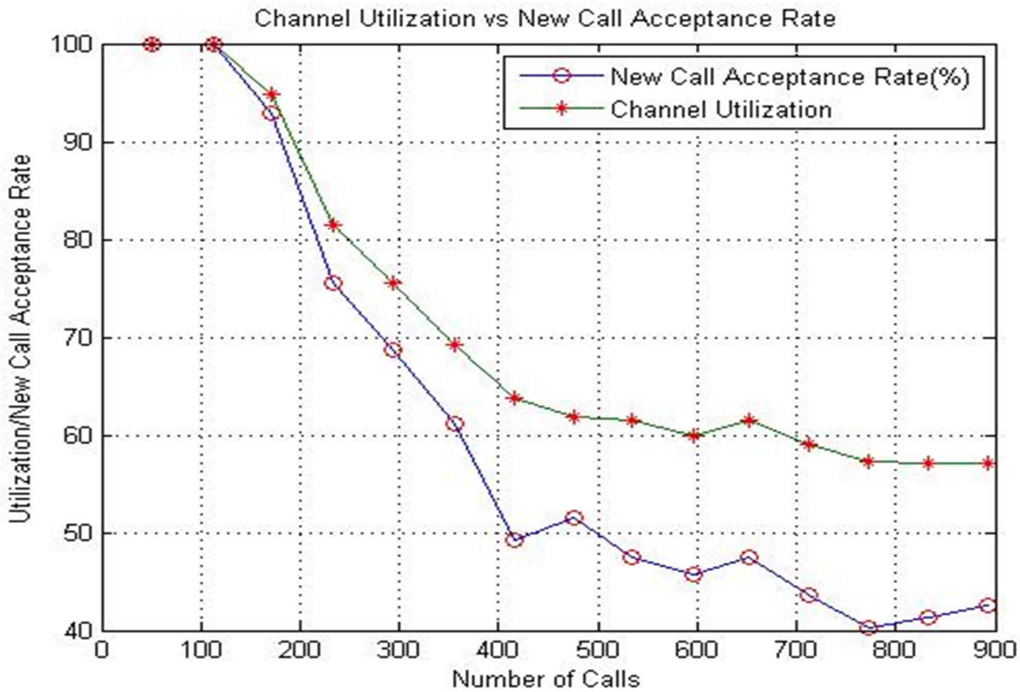

Figure 4.11: Channel utilization with new call acceptance rate

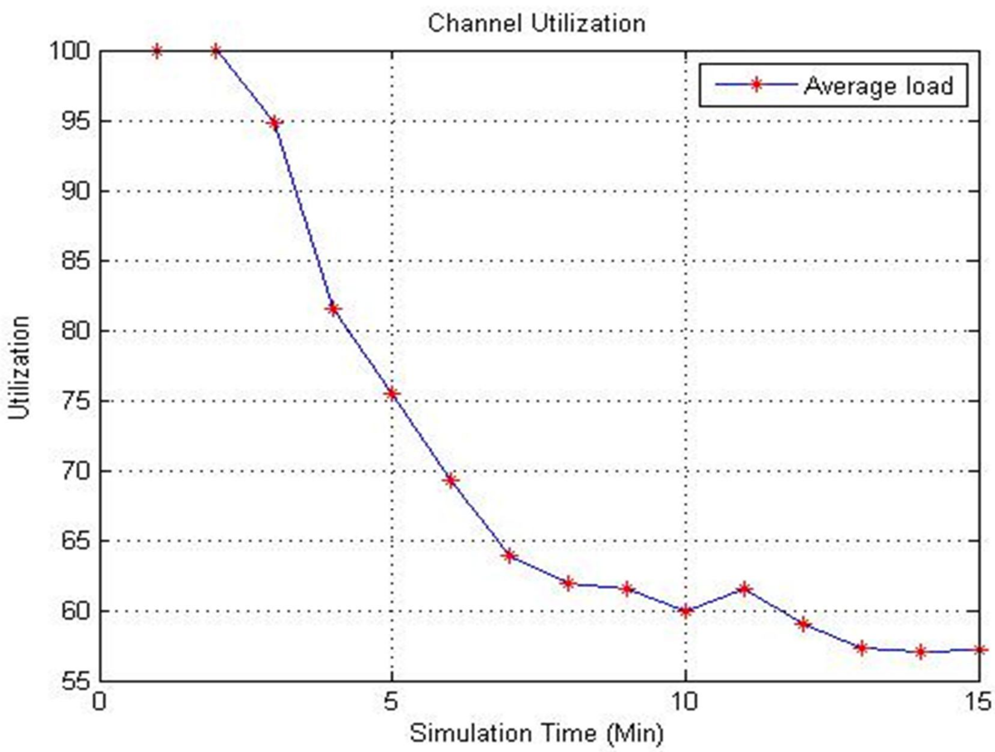

Figure 4.12: Channel utilization

CHAPTER FIVE

CONCLUSION, RECOMMENDATIONS AND FURTHER INVESTIGATIONS

5.1 Conclusion

In this thesis, a dynamic guard channel scheme has been developed and implemented. The collected data from RNO Section of Somalia Telecommunication Company in Mogadishu like GCell, Traffic Volume on SDCCH (Erl), Traffic Volume on TCH (Erl), Call Drop Rate on SDCCH, TCH Seizure Success Rate, Call Drop Rate on TCH per cell (Excluding Handover), Call Drop Rate on TCH per cell (including Handover) and handover success rate were analysed by using Microsoft Office Excel 2013 data analysis tool. In order to assess the performance of the cells, Communications Authority of Somalia's KPIs standards was carried out. The twenty-seven base stations (BSs) in the most populated regions in Mogadishu are investigated. The results shown that, most of the cells are not meeting all the KPIs indicators set by CAS. Out of the five indictors under study, Cell one only met all parameter according to the recommended targets set by CAS. For call setup success rate, two out of twenty seven cells were not managed to meet the Communications Authority of Somalia target of 98%. Which means 7.4% percent failed to achieve call setup success rate target. The research also confirmed that the handover failure rate is among the worst performing metric in the country, only one cell out of twenty seven cells, has achieved the CAS minimum target requirement of 98% success. This means that 96.3% of the cells fell below the intended target, which clearly shows how the service qualities provided by the GSM operators in Somalia remained extremely bad.

To overcome that problems Dynamic Guard Channel Algorithm was developed by using JAVA Software, NetBeans IDE 8.1. This system in spite of being accurate, we are able to tweak other factors within it and add some more functionality that from our theories with theoretically boost these numbers, while still maintaining the use of the systems infrastructure and available resources. By altering the existing software and from analyzing the patterns of existing systems we are able to come up with a mathematical and software type of solution to the telecommunications company. The algorithm prioritizes the handover calls over the new calls, by reserving some channels only for handover calls only. This scheme reduced the handover call dropping probability by remaining the new call blocking probability almost constant which provide better QoS for wireless network. The calls generated are divided into two types; handover calls, which is given the first priority and new calls. The performance of the proposed algorithm is evaluated in terms of new call blocking probability, handover call dropping probability and channel utilization and presented in graphics by using Matlab.

The results obtained from the simulation shows that, for handover success rate, only one time period within the system set at 15 minutes, produces an anomaly that puts it above the values recommended by the CAS. That means, we reduced the handover failure rate from 96.3% down to 6.7%, which when projected to a larger system puts the System within acceptable range. In comparison to the real data the rate of accepting handover calls raises up by 89.6% improvement increase. This clearly are an indication our system is effectively utilize the available resources.

Also we compared our simulated results with non-prioritization (no guard channels) scheme and we have seen our algorithm was able to deliver better performance than other schemes. Our algorithm maintained the handover call dropping probability at CAS target while constraining the new call blocking probability within acceptable level and optimizing the channel utilization. Even in the heavy traffic, the handover call dropping probability are at the acceptance level. Moreover, with this algorithm, one hundred percent channel bandwidth are utilized and wastage of bandwidth are avoided, which will allow the service providers to generate more revenue.

5.2 Recommendations

The research was tested in a system that utilizes the 2G (GSM) technology. This already provides a limitation to my research which at the moment works best in a system using the stated technology. As the world is moving forward and we now have 3G, 4G and latest 5G, the system gets more complex with new algorithms that solve some problems while introducing others. If the system can be extended to be used in these other new advanced technologies, we may be able to grow, understand and feel the effect of Dynamic Guard channel better in a more robust system.

At the time of performing this research, what was most essential was solving the basic communication system and making it achieve necessary standards. Apart from voice technology, we are moving towards data and the internet. From research we see that the telecommunication companies which are also ISP agents, use the same model as voice in their system. This put the system and customers through the same predicament, and this algorithm may be tweaked to be able to solve any occurring challenges internet users may experience when their call internet is dropped.

In the above research, data was only provided by one of the companies from Somalia, different companies have different policies and implementation of the same system, accorded the chance, data from more than one Telecommunication Company is a needed requirement in order to compare and find out what variations they may possess which will help in restructuring the algorithm.

5.3 Suggestions for Further Work

Following more advanced research on the idea, it can be built and advanced even more in order to expand its use within the telecommunication industry and future work.

Since this research investigated the quality of service parameters of one of Telecommunication Company in Somalia, but there are six other telecommunication operators in Somalia, and others would be expected. Future study should be included all of that companies and analyse the parameters of data calls. Also call drop rate may analysed further in terms of Call Drop Rate on TCH per cell (including handover), Call Drop Rate on TCH per cell (excluding handover), and call drop rate on SDCCH. Till now, this research deployed just one cell, further research may deploy several cells and run simultaneously. So that when a new or handover call enters or evacuates a cell, the directly related cell notifies its neighbour cells, and if the is admitted (either a handover call or a new call) in a chosen cell, say x, each of cell x's neighbouring BSs, say y, reserves a channel for incoming call in either direction.

Testing of the algorithm in a real system, and then monitor its performance vis-à-vis what is already being used in the system, and also support the present system in use.
While establishing the algorithm, we found out that balance is an important factor in order to improve or to reduce performance of the system. Following repetitive testing and also stress testing of different variations, there are some given limits whereby the system runs effectively and efficiently where other times it isn't. A major factor in

these equations was the traffic load, which shows how the system is responding to certain pressure points. In the future research and study, the impact of the channel holding time on the quality of service can be studied and analysed.

Studying, analysing and figuring out advantages of other forms of algorithms where we may be able to improve or merge the Guard channel algorithm with another to make a better algorithm.

REFERENCES

[1] Wikipedia, "Communications in Somalia," 28 November 2016. [Online]. Available: https://en.wikipedia.org.

[2] Mohamed Elmi , "Information and Communication Technologies and the Stabilization of a Failed State: the case of Somalia," University of Cape Town, Cape Town, South Africa, September 2015.

[3] R. Program, "Responses to Information Requests (RIR)," Immigration and Refugee Board of Canada , Ottawa, February, 2015.

[4] H. M. O. (Dudde), "Telecom: Somalia's success industry," Hogan Lovells, Mogadishu,Somalia, 2012.

[5] Statista, "Number of cellular subscriptions Somalia 2000-2016," Statista Portal, 2017. [Online]. Available: www.statista.com.

[6] A. S and M. T, "DYNAMIC CHANNEL ALLOCATION SCHEME TO HANDLE HANDOFF IN WIRELESS MOBILE NETWORK," *Computer Science & Information Technology (CS & IT),* vol. 2, p. 171–184, 2012.

[7] Y.-B. Lin, Mohan and A. Noerpel, "PCS Channel Assignmnet Strategies for Hand-off and Initial Access," *IEEE Personal Communications,* vol. 1, no. 3, pp. 47-56, October 1994.

[8] B. Madan, S. Dharmaraja and K. S. Trivedi, "Combined Guard Channel and Mobile Assisted Handoff for Cellular Networks," *IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY,* vol. 57, no. 1, pp. 502 - 510, 2008 2008.

[9] M. P. B. Bhilare and P. S. Sambare, "Seamless handoff in Next Generation Wireless System," *International Journal Of Computer Science And Technologies,* vol. 2, no. 6, pp. 2525-2530, 2011.

[10] K. K. Jatin, "Study and Analysis of Call dropping and Handover Problem in cellular system," *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET),* vol. 5, no. 6, pp. 1776-1777, June 2016.

[11] L. M. C. R. Soldani D, QoS and QoE Management in UMTS Cellular Systems, 1st ed., New York, NY: Wiley, 2006.

[12] R. N. Al, "Handoff and Drop Call Probability: A Case Study of Nigeria's Global System for Mobile Communications (GSM) Sector".

[13] M. D. S. Bhawan and J. N. Marg, "Telecom Regulatory Authority of India," New Delhi, 1997.

[14] S. Alagha, "Queuing-Based Dynamic Multi-Guard Channel Scheme for Voice/Data Integrated Cellular Wireless Networks," Gazimağusa, North Cyprus, 2013.

[15] S. Ramanath, "Impact of mobility on call block, call drops and optimal cell size in small cell networks," France, 2010.

[16] H. T, R. J. and M. J., GSM, GPRS and EDGE Performance, John Wiley & Sons Ltd, 2003.

[17] B. Haider, M. Zafrullah and M. K. Islam, "Radio Frequency Optimization & QoS Evaluation in Operational GSM Network," *Proceedings of the World Congress on Engineering and Computer Science,* vol. 1, pp. 1-6, 2009.

[18] S. John, "Overview of the Global System for Mobile Communications," 1997.

[19] G. M and B. P., "Call Admission Control in mobile cellular Networks," *Comprehensive Survey wireless communications and mobile computing,* vol. 6, no. 1, pp. 69-93, march 2006.

[20] H. Kaur and A. Kaur, "VARIOUS HANDOVER MANAGEMENT TECHNIQUES IN GSM CELLULAR SYSTEM," *International Journal For Technological Research In Engineering,* vol. 1, no. 11, pp. 1335-1338, July 2014.

[21] B.VenkataSai Sireesha, Dr.S.Varadarajan, Vivek and Naresh, "Increasing Of Call Success Rate In GSM Service Area Using RF Optimization," *International Journal of Engineering Research and Applications (IJERA),* vol. 1, no. 4, pp. 1479-1485.

[22] A. Ozovehe and A. U. Usman, "PERFORMANCE ANALYSIS OF GSM NETWORKS IN MINNA METROPOLIS OF NIGERIA," *Nigerian Journal of Technology (NIJOTECH),* vol. 34, no. 2, p. 359 – 367, April 2015.

[23] Shoewu, O. a. Edeko and F.O., "Outgoing call quality evaluation of GSM network services in Epe, Lagos State," *AMERICAN JOURNAL OF SCIENTIFIC AND INDUSTRIAL RESEARCH,* vol. 2, no. 3, pp. 409-417, 2011.

[24] I. A. Laryea, "Parameters/Thresholds for Quality of Voice and Data Services (Case of Ghana)," in *Quality of Service Development Group (QSDG) meeting* , Amsterdam , 2016.

[25] April 2012. [Online]. Available: http://www.nca.org.gh/downloads/QoS_Trends_for_April_2012.pdf.

[26] M. A. Alam, "Mobile Network Planning and KPI Improvement," Sweden, 2013.

[27] M. Ali, A. Shehzad and D. M. Akram, "Radio Access Network Audit & Optimazation in GSM(Radio Access Network Quality Improvement Techniques)," *International Journal of Engineering & Technology IJET-IJENS,* vol. 10, no. 1, pp. 55-58, February 2010.

[28] J. Khan, "Handover management in GSM cellular system," *International Journal of Computer Applications,* vol. 8, no. 12, pp. 14-24, October 2010.

[29] V. Goswami and P. K. Swain, "Analytical Modeling for Handling Poor Signal Quality Calls in Cellular Network," *International Journal of Networks and Communications,* vol. 2, no. 4, pp. 47-54, 2012.

[30] O. M. Longe, "Effect of signal strength on handover in GSM networks in Owo, Ondo State, Nigeria," in *3rd IEEE International Conference Adaptive Science and Technology (ICAST),* Abuja, Nigeria, 24-26 Nov, 2011.

[31] G. Heine, GSM Networks: Protocols, Terminology, and Implementation, 1st ed., London: Artech House,INC,, 1999.

[32] A. Freedman and Z. Hadad, "Handoff Schemes Overview and Guidelines for Handoff Procedures in 802.16," 2002.

[33] Verdone and Z. A, "Performance of received power and traffic-driven handover algorithms in urban cellular networks," *IEEE Wireless Communications,* vol. 9, no. 1, pp. 60 - 71, Feb 2002.

[34] "An Adaptive Algorithm for Call Admission Control in Wireless Networks," in *IEEE Global Telecommunications Conference,* San Antonio, TX, 2001.

[35] N. D. Tripathi, N. J. H. Reed, H. F. V. MPRG and V. Tech, "Handoff in Cellular Systems," *IEEE Personal Communications,* vol. 5, no. 6, pp. 26 - 37, December 1998.

[36] F. Y. and Z. Y., "Cellular models and handover criteria," in *Proceeding 39th IEEE Vehicular Technology Conference,* retrieved from www.citeseer.com , 2012.

[37] O. P.-E, "High Performance Handover Schemes for Modern Cellular Systems," 2012.

[38] Nasser, N., Hasswa and a. Hassanein, "Handoffs in fourth generation heterogeneous networks," *IEEE Communications Magazine,* vol. 44, no. 10, pp. 96 - 103, October 2006.

[39] S. C and B. M., "Network planning aspects of DS-CDMA with particular emphasis on soft handover," in *Proceeding 43rd IEEE Vehicular Technology Conference,* 1993.

[40] D. J. Kumawat and S. Tailor, "Soft and Softer Handover in Communication Netwoks," *International Journal on Recent and Innovation Trends in Computing and Communication,* vol. 1, no. 6, p. 558 – 562, June 2013.

[41] G. M., "Correlation model for shadow fading in mobile radio systems," *Electronics Letter,* vol. 27, no. 23, p. 2145–2146, 1991.

[42] A. Sgora and Dimitrios, "Handoff prioritization and decision schemes in wireless cellular networks: a survey," *IEEE Communications and Tutorials,* vol. 11, no. 4, pp. 57-76, December 2009.

[43] N. BARTOLINI, "Handoff and Optimal Channel Assignment in Wireless Networks," *Mobile Networks and Applications,* vol. 6, no. 6, p. 511–524, 2001.

[44] J. Diederich and M. Zitterbart, "Handoff Prioritization Schemes Using Early Blocking," *IEEE Communications Surveys and Tutorials,* vol. 7, no. 2, pp. 26-45, 2005.

[45] I. Katzela and M. Naghshineh, "Channel assignment schemes for cellular mobile telecommunication systems: a comprehensive survey," *IEEE Personal Communication,* vol. 3, no. 3, pp. 10-31, 1996.

[46] M.P. Mishra and P.C Saxena, "Survey of Channel Allocation Algorithms Research for Cellular Systems," *International Journal of Networks and Communications,* vol. 2, no. 5, pp. 75-104, October 2012.

[47] Khaja Kamaluddin and Aziza Ehmaid Omar, "Channel Assignment and Minimum Dropping Probability scheme for Handover Calls in Mobile Wireless Cellular Networks," *International Journal of Recent Trends in Electrical & Electronics Engg.,* vol. 1, no. 2, pp. 1-9, Sept 2011.

[48] S. S. Nayak, R. R. Kar and M. Garanayak, "Adaptive Shared Channel Assignment Scheme for Cellular Network to Improve the Quality of the Handoff Calls," *IOSR Journal of Computer Engineering (IOSR-JCE),* vol. 17, no. 1, pp. 61-67, February 2015.

[49] R. E. Ahmed, "A Hybrid Channel Allocation Algorithm using Hot-Spot Notification for Wireless Cellular Networks," in *Canadian Conference on Electrical and Computer Engineering*, Ottawa, Ont., 2006.

[50] S. A. Elimam and A. B. A/Nabi, "Dynamic versus Static Channel Allocation Scheme," *International Journal of Science and Research (IJSR),* vol. 4, no. 6, pp. 2338-2339, June 2015.

[51] P. Chowdhury, M. Atiquzzaman and W. Ivancic, "Handover schemes in satellite networks: state-of-the-art and future research directions," *IEEE Communications Surveys & Tutorials,* vol. 8, no. 4, pp. 2 - 14, 2006.

[52] C.-J. Chang, T.-T. Su and Y-Y, "Analysis of a Cutoff Priority Cellular Radio System with Finite Queueing and Reneging/Dropping," *IEEE/ACM Trans. Networking,* vol. 2, no. 2, p. 166–175, 1994.

[53] S., Tekinay and B. Jabbari, "A Measurement-Based Prioritization Scheme for Handovers in Mobile Cellular Networks," *IEEE J. Select.Areas Commun.,* vol. 10, no. 8, pp. 1343-1350, October 1992.

[54] Y. Fang, "Modeling and performance analysis for wireless mobile networks: a new analytical approach," *IEEE/ACM Transactions on Networking,* vol. 13, no. 5, pp. 989 - 1002, Oct 2005.

[55] A. Kumar, A. Bhushan and M. Kumar, "A Study on Minimization of 2G/3G Handover Failure," *International Journal of Advanced Research in Computer Science and Software Engineering,* vol. 2, no. 4, pp. 4-7, April 2012.

[56] M. A. Rahman, A. S. Md and M. S. I. Sohag, "The Effect of Average Channel Holding Time to Estimate the QoS Parameters of Cellular Network through Guard Channel Scheme," in *international conferrence on Electrical Engineering and Information Communication Technology (ICEEICT)*, Dhaka, 2015.

[57] v. Stojmenovic, Handbook of Wireless Networks and Mobile Computing, v. Stojmenovic, Ed., JOHN WILEY & SONS, INC., 2002.

[58] S. Boumerdassi, "An efficient reservation-based dynamic channel assignment strategy," in *First International Conference on 3G Mobile Communication Technologies*, London, 2000.

[59] R. Bhattacharjee, T. Ahmed and M. Zaman, "Priority based adaptive guard channel for multi-class traffic in wireless networks," in *International Conference on Electrical Information and Communication Technology (EICT)*, Khulna, 2013.

[60] G.-C. Chen and S.-Y. Lee, "Modeling the static and dynamic guard channel schemes for mobile transactions," in *International Conference on Parallel and Distributed Systems*, Tainan, 1998.

[61] K. Chua, B. Bensaout, W. Zhuang and S.Y.Choo, "Dynamic channel reservation (DCR) scheme for handoffs prioritization in mobile micro/picocellular networks," in *nternational Conference on Universal Personal Communications*, Florence, 1998.

[62] Amisom. [Online]. Available: http://amisom-au.org/.

[63] July 2015 . [Online]. Available: http://www.mapsofworld.com/somalia/cities/mogadishu.html.

[64] N. S. Tarkaa, J. M. Mom and C. I. Ani, "Drop Call Probability Factors in Cellular Network," *International Journal of Scientific & Engineering Research*, pp. 1-5, 2011.

[65] O. Osahenvemwen and Emagbetere, "Determination of Traffic Load and Traffic Performance Parameters in Mobile Communication in Nigeria," *Research Journal of Applied Sciences, Engineering and Technology*, vol. 4, no. 11, pp. 432-1437, May 2012.

[66] G. Liodakis and P. Stavroulakis, "A novel approach in handover initiation for microcellular systems," in *IEEE 44th Vehicular Technology Conference*, Stockholm, 1994.

APPENDIX A: JAVA Code for our algorithm

```
/*
Abubaker Programs Executing
*/
package interfacetest;

import java.awt.Font;
import java.util.ArrayList;
import java.util.Vector;
import javax.swing.JButton;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JProgressBar;
import javax.swing.JTable;
import javax.swing.JTextArea;
import javax.swing.JTextField;
import javax.swing.Timer;

/**
 *
 * @author Abubakar
 */

public class AttributesCell {

    /**
     * the below are attributes used in the class interfacetest
     */
        public static int packet, number_of_periods, timePeriod;
    public static int total_available_channels,
shared_channels[],reserved_channels[];
    public static int OriginatedCalls[],incomingHanodverCalls[];
    public static int CompletedNewCalls[],CompletedHandoverCalls[];
    public static int handover_calls_dropped[],
handover_calls_accepted[],handover_calls_accepted_copy[];
      public static int no_ofOriginated_calls_rejected[],
no_ofOriginated_calls_accepted[],
no_ofOriginated_calls_accepted_copy[];
    public static int totalNo_ofHandover_calls[],
totalNo_ofOriginated_calls[], ongoingCalls[];
    public static int maximum_number_of_guard_channels = 10,
minimum_number_of_guard_channels = 2, N = 0;
    public static int newCall_timeDelay =3000, HandoverCall_timeDelay
= 3200, call_generation_time_delay = 1000;
             /**
            * all_time stores the timer of time period simulations
            * all_handoverTime stores the timers of all handovers
            * all_NewcallTime stores the timers of all new calls
            */
    public static ArrayList<Timer> all_time = new ArrayList<Timer>();
    public static ArrayList<Timer> all_handoverTime = new
ArrayList<Timer>();
    public static ArrayList<Timer> all_NewcallTime = new
ArrayList<Timer>();

        /**
         * call_drop_rate the rate of call drops initialzed to zero
         * handover_call_dropping_prob[] the handoff failure rate
         * Originated_call_blocking_prob is the call block rate
```

```
         *  lowerLimitThreshold is the lower limit of the threshold
0.06 of handover failure rate
         * upperLimitTreshold is the upper limit threshold of the
handover failure rate

         */
     public static double handover_call_dropping_prob[];
    public static double Originated_call_blocking_prob = 0, threshold
= 0.10;
    public static double lowerLimitThreshold = 0.06,
upperLimitTreshold = 0.07;
    public static double probable_increase = 0.9;

    public static float[] current_handover_call_dropping_prob;
    public static float[] Call_setup_failure_rate,
handover_failure_rate, successrates;
    public static float[] traffic_load;
    public static float[] all_attempted_calls, all_originate_calls;

    //all_period_times[]; stores the different simulations in strings

    public static String all_period_times[];

     /**
     * These are different kinds of timers
     * timer_generation for call generation timer
     * call_timers for timing new calls
     * handoverTimers for timing handovers
     * stopProgramTime for timing simulations
     */
    public static Timer[] timer_generation, call_timers,
handoverTimers, stopProgramTime;

     /**
     * the Vector object creates the tables and
all_period_timesBasket
     */
    public static Vector<Vector> tableCells = new Vector<Vector>();
    public static Vector<Integer> all_period_timesBasket = new
Vector<Integer>();

    public static Thread t[];
//definition of the thread-the threads are smaller processes that
runs in the main process best for running different timers
    public static StringBuffer data_store[];//for storing  messages
    public static String Type_of_Call;
}




package interfacetest;
import java.awt.Color;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import static java.lang.System.exit;
import java.math.BigDecimal;
import java.util.Arrays;
import java.util.Random;
import java.util.Vector;
import java.util.concurrent.ExecutorService;
```

```
import java.util.concurrent.Executors;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.Timer;
import javax.swing.table.DefaultTableModel;

/**
 *
 * @author Abubakar
 */
public class One extends javax.swing.JFrame implements ActionListener
{

    /**
     * Creates new form One
     */
    public One() {
        initComponents();
    }

     Random random = new Random();
     private String Type_of_Call;
     private volatile boolean exit = false;
     private int sss= 0;
     private int sss2= 0;


      public void shutdown(){
          exit = true;
      }

    /**
     * This method is called from within the constructor to
initialize the form.
     * WARNING: Do NOT modify this code. The content of this method
is always
     * regenerated by the Form Editor.
     * its generated code
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jButton1 = new javax.swing.JButton();
        jScrollPane2 = new javax.swing.JScrollPane();
        background_panel = new javax.swing.JPanel();
        jPanel1 = new javax.swing.JPanel();
        simulations_textfield = new javax.swing.JTextField();
        channel_textfield = new javax.swing.JTextField();
        simulation_label = new javax.swing.JLabel();
        channel_label = new javax.swing.JLabel();
        start_simulation_button = new javax.swing.JButton();
        jScrollPane1 = new javax.swing.JScrollPane();
        display_output_textfield = new javax.swing.JTextArea();
        progress_bar = new javax.swing.JProgressBar();
        table_output_field = new javax.swing.JScrollPane();
        display_table = new javax.swing.JTable();
```

```
        jLabel1 = new javax.swing.JLabel();
        generate_label = new javax.swing.JLabel();
        simulation_time_is = new javax.swing.JLabel();
        time_running = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();

        jButton1.setText("jButton1");


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setCursor(new
java.awt.Cursor(java.awt.Cursor.DEFAULT_CURSOR));

        jScrollPane2.setBackground(new java.awt.Color(0, 255, 102));

jScrollPane2.setViewportBorder(javax.swing.BorderFactory.createBevelB
order(javax.swing.border.BevelBorder.RAISED, null,
java.awt.Color.pink, null, null));
        jScrollPane2.setCursor(new
java.awt.Cursor(java.awt.Cursor.HAND_CURSOR));

        simulations_textfield.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                simulations_textfieldActionPerformed(evt);
            }
        });

        channel_textfield.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                channel_textfieldActionPerformed(evt);
            }
        });

        simulation_label.setText("Simulation");

        channel_label.setText("Channels");

        start_simulation_button.setText("Start");
        start_simulation_button.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                start_simulation_buttonActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout jPanel1Layout = new
javax.swing.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)
            .addGroup(jPanel1Layout.createSequentialGroup()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.A
lignment.LEADING)
```

```
                    .addGroup(jPanel1Layout.createSequentialGroup()
                        .addGap(144, 144, 144)
                        .addComponent(simulation_label)
                        .addGap(18, 18, 18)
                        .addComponent(simulations_textfield,
javax.swing.GroupLayout.PREFERRED_SIZE, 70,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(41, 41, 41)
                        .addComponent(channel_label)
                        .addGap(38, 38, 38)
                        .addComponent(channel_textfield,
javax.swing.GroupLayout.PREFERRED_SIZE, 70,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(jPanel1Layout.createSequentialGroup()
                        .addGap(299, 299, 299)
                        .addComponent(start_simulation_button)))

.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );
        jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.L
EADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addContainerGap(20, Short.MAX_VALUE)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.A
lignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel1Layout.createSequentialGroup()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.A
lignment.BASELINE)
                            .addComponent(simulations_textfield,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(channel_textfield,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(simulation_label))
                        .addGap(11, 11, 11))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel1Layout.createSequentialGroup()
                        .addComponent(channel_label)
                        .addGap(18, 18, 18)))
                .addComponent(start_simulation_button))
        );

        display_output_textfield.setColumns(20);
        display_output_textfield.setRows(5);
        display_output_textfield.setCursor(new
java.awt.Cursor(java.awt.Cursor.TEXT_CURSOR));
        jScrollPane1.setViewportView(display_output_textfield);

        progress_bar.setBackground(new java.awt.Color(255, 204, 0));
        progress_bar.setToolTipText("");
```

```
        progress_bar.setValue(2);
        progress_bar.setString("0:0 mins");
        progress_bar.addChangeListener(new
javax.swing.event.ChangeListener() {
            public void stateChanged(javax.swing.event.ChangeEvent
evt) {
                progress_barStateChanged(evt);
            }
        });

        table_output_field.setBackground(new java.awt.Color(255, 255,
102));

        display_table.setAutoCreateRowSorter(true);
        display_table.setModel(new
javax.swing.table.DefaultTableModel(
            new Object [][] {

            },
            new String [] {
                "S/N", "Handoff Success Rate", "CSSR", "Call
Blocking", "Handoff Failures", "Traffic Load", "Simul Time"
            }
        ));
        table_output_field.setViewportView(display_table);

        jLabel1.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel1.setText("Simulation Output...Executing");

        generate_label.setFont(new java.awt.Font("Tahoma", 1, 11));
// NOI18N
        generate_label.setText("?");

        simulation_time_is.setText("Simulation Time is:");

        time_running.setText(">>");

        jLabel2.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel2.setText("Dynamic guad channel");

        javax.swing.GroupLayout background_panelLayout = new
javax.swing.GroupLayout(background_panel);
        background_panel.setLayout(background_panelLayout);
        background_panelLayout.setHorizontalGroup(

background_panelLayout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)
            .addGroup(background_panelLayout.createSequentialGroup()
                .addContainerGap(366, Short.MAX_VALUE)

.addGroup(background_panelLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING)

.addGroup(background_panelLayout.createSequentialGroup()
                        .addComponent(generate_label,
javax.swing.GroupLayout.PREFERRED_SIZE, 137,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(2, 2, 2)
                        .addComponent(simulation_time_is)
```

```
                        .addGap(18, 18, 18)
                        .addComponent(time_running,
javax.swing.GroupLayout.PREFERRED_SIZE, 49,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addComponent(table_output_field,
javax.swing.GroupLayout.PREFERRED_SIZE, 456,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(66, 66, 66))
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
background_panelLayout.createSequentialGroup()
                .addComponent(jPanel1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addContainerGap())
            .addGroup(background_panelLayout.createSequentialGroup()

.addGroup(background_panelLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING)

.addGroup(background_panelLayout.createSequentialGroup()
                        .addGap(26, 26, 26)

.addGroup(background_panelLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING, false)
                            .addComponent(jLabel1)
                            .addComponent(progress_bar,
javax.swing.GroupLayout.DEFAULT_SIZE, 307, Short.MAX_VALUE)

.addGroup(background_panelLayout.createSequentialGroup()
                                .addGap(10, 10, 10)
                                .addComponent(jScrollPane1,
javax.swing.GroupLayout.DEFAULT_SIZE, 297, Short.MAX_VALUE))))

.addGroup(background_panelLayout.createSequentialGroup()
                        .addContainerGap()
                        .addComponent(jLabel2,
javax.swing.GroupLayout.PREFERRED_SIZE, 219,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addGap(0, 0, Short.MAX_VALUE))
        );
        background_panelLayout.setVerticalGroup(

background_panelLayout.createParallelGroup(javax.swing.GroupLayout.Al
ignment.LEADING)
            .addGroup(background_panelLayout.createSequentialGroup()
                .addGap(7, 7, 7)
                .addComponent(jLabel2)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED
)
                .addComponent(jPanel1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(background_panelLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING)
```

```
.addGroup(background_panelLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.BASELINE)
                        .addComponent(jLabel1)
                        .addComponent(time_running))

.addGroup(background_panelLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.BASELINE)
                        .addComponent(generate_label,
javax.swing.GroupLayout.PREFERRED_SIZE, 14,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(simulation_time_is)))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(background_panelLayout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING)
                    .addComponent(table_output_field,
javax.swing.GroupLayout.PREFERRED_SIZE, 309,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 349,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(18, 18, 18)
                .addComponent(progress_bar,
javax.swing.GroupLayout.PREFERRED_SIZE, 65,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(143, Short.MAX_VALUE))
        );

        jScrollPane2.setViewportView(background_panel);

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jScrollPane2,
javax.swing.GroupLayout.DEFAULT_SIZE, 865, Short.MAX_VALUE)
                .addContainerGap())
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jScrollPane2,
javax.swing.GroupLayout.DEFAULT_SIZE, 673, Short.MAX_VALUE)
                .addContainerGap())
        );

        pack();
    }// </editor-fold>

    private void
simulations_textfieldActionPerformed(java.awt.event.ActionEvent evt)
{
```

```
        // TODO add your handling code here:

    }

    private void
channel_textfieldActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }
    private void
progress_barStateChanged(javax.swing.event.ChangeEvent evt) {
        // TODO add your handling code here:
        progress_bar.setMaximum(100);
        progress_bar.setMinimum(0);
        progress_bar.setOpaque(true);
    }

    private void
start_simulation_buttonActionPerformed(java.awt.event.ActionEvent
evt) {

              /**
       * the if construct checks if the event generated is from the
simulation
       * button and gets the input data from the
simulations_textfield and channel_textfield
       * TextField and converts them into integers and puts the time
periods into the all_period_timesBasket
       */
   if (simulations_textfield.getText().isEmpty() &&
channel_textfield.getText().isEmpty()) {
             JOptionPane.showMessageDialog(this, "Invalid Entry",
"Error", JOptionPane.ERROR_MESSAGE);
        } else {

            AttributesCell. all_period_times =
simulations_textfield.getText().split(",");
            AttributesCell. timePeriod =
Integer.parseInt(AttributesCell.all_period_times[0].trim());
            int sum_time = 0;

                for (int k = 0; k <
AttributesCell.all_period_times.length; k++) {
                    sum_time +=
Integer.parseInt(AttributesCell.all_period_times[k].trim());

AttributesCell.all_period_timesBasket.addElement(Integer.parseInt(Att
ributesCell.all_period_times[k].trim()));

                    System.out.println("Summation of time "
+Integer.parseInt(AttributesCell.all_period_times[k]));
                    System.out.println("Summation of time "
+sum_time);
                }
            time_running.setText(String.valueOf(sum_time)+" Min");
            generate_label.setText("Generating Tables >>>");
            AttributesCell.total_available_channels =
Integer.parseInt(channel_textfield.getText());

            /**
             * the below block of code instantiates the following
arrays of
```

```
              * different types used in the program
             */
            AttributesCell. t = new
Thread[AttributesCell.all_period_times.length];
            AttributesCell.data_store = new
StringBuffer[AttributesCell.all_period_times.length];
            AttributesCell.current_handover_call_dropping_prob = new
float[AttributesCell.all_period_times.length];
            AttributesCell. Call_setup_failure_rate = new
float[AttributesCell.all_period_times.length];
            AttributesCell.handover_failure_rate = new
float[AttributesCell.all_period_times.length];
            AttributesCell. traffic_load = new
float[AttributesCell.all_period_times.length];
            AttributesCell.all_attempted_calls = new
float[AttributesCell.all_period_times.length];
            AttributesCell.all_originate_calls = new
float[AttributesCell.all_period_times.length];
            AttributesCell.timer_generation = new
Timer[AttributesCell.all_period_times.length];
            AttributesCell.call_timers = new
Timer[AttributesCell.all_period_times.length];
            AttributesCell.handoverTimers = new
Timer[AttributesCell.all_period_times.length];
            AttributesCell.stopProgramTime = new
Timer[AttributesCell.all_period_times.length];
            AttributesCell.shared_channels = new
int[AttributesCell.all_period_times.length];
            AttributesCell.reserved_channels = new
int[AttributesCell.all_period_times.length];
            AttributesCell.ongoingCalls = new
int[AttributesCell.all_period_times.length];
            AttributesCell.OriginatedCalls = new
int[AttributesCell.all_period_times.length];
            AttributesCell.handover_calls_accepted = new
int[AttributesCell.all_period_times.length];
            AttributesCell.handover_calls_accepted_copy = new
int[AttributesCell.all_period_times.length];
            AttributesCell.handover_calls_dropped = new
int[AttributesCell.all_period_times.length];
            AttributesCell.totalNo_ofHandover_calls = new
int[AttributesCell.all_period_times.length];
            AttributesCell.no_ofOriginated_calls_rejected = new
int[AttributesCell.all_period_times.length];
            AttributesCell.no_ofOriginated_calls_accepted = new
int[AttributesCell.all_period_times.length];
            AttributesCell.no_ofOriginated_calls_accepted_copy = new
int[ AttributesCell.all_period_times.length];
            AttributesCell.incomingHanodverCalls = new int[
AttributesCell.all_period_times.length];
            AttributesCell.totalNo_ofOriginated_calls = new int[
AttributesCell.all_period_times.length];
            AttributesCell. handover_call_dropping_prob = new double[
AttributesCell.all_period_times.length];

            AttributesCell.successrates = new float
[AttributesCell.all_period_times.length];
            AttributesCell.CompletedHandoverCalls = new
int[AttributesCell.all_period_times.length];
            AttributesCell.CompletedNewCalls = new
int[AttributesCell.all_period_times.length];
```

```
                /**
                 * using the if control flow construct different
threads are created and run in the
                 * t thread array
                 */
                for (int k = 0; k <
AttributesCell.all_period_times.length; k++) {

                     AttributesCell.number_of_periods = k;
                     AttributesCell.data_store[k] = new
StringBuffer();
                     AttributesCell.t[k] = new Thread(new Runnable()
{
                        @Override
                        public void run() {
                            timed_simulation_starter(
AttributesCell.number_of_periods);

                        }
                    });

                     AttributesCell.t[k].start();
                     AttributesCell.t[k].isAlive();
                     System.out.println("Thread 1 -- "+ k +" has
started");
                     System.out.println("Thread 2 -- "+
Arrays.toString(AttributesCell.t) +" has started");
                     System.out.println("Check this out "+
AttributesCell.all_period_times.length);

                    try {

                        Thread.sleep(
AttributesCell.call_generation_time_delay);
                    } catch (Exception ex) {
                    }

                }
            }
        //}

    }


        /**
     * this method below dynamic_guardChannel_algorithm is invoked
when a
     * call is generated and  a timer is set of in the
timer_generation array
     * @param n is the index of the array where the timer is
     * @throws Exception
     */
    public synchronized void dynamic_guardChannel_algorithm(int n)
throws Exception {
        AttributesCell.timer_generation[n] = new Timer(
AttributesCell.call_generation_time_delay, this);
        AttributesCell.all_time.add(
AttributesCell.timer_generation[n]);
        AttributesCell.timer_generation[n].start();
```

```
    }

        /**
     * the method accepts three parameters
     * @param call_type this indicates the call type can be 1 or 2
     * @param call_no irrelevant
     * @param n  the index number indicating in what simulation the
call is taking place
     * the method runs procedures when the the call or handoff is
completed to release resources
     */
   public synchronized void all_completed_calls(int call_type, int
call_no, int n) {
        AttributesCell.ongoingCalls[n] =
AttributesCell.ongoingCalls[n] - 1;
        if (  Type_of_Call == "Handover")
        {    ++AttributesCell.CompletedHandoverCalls[n];
             AttributesCell.handover_calls_accepted_copy[n] = (
AttributesCell.
                     handover_calls_accepted_copy[n] - 1 < 0 ? 0 :
AttributesCell.handover_calls_accepted_copy[n] - 1);
            print_Output("one of Handover calls has just
completed..... Still Remaining=======" +
AttributesCell.handover_calls_accepted_copy[n], n);
             AttributesCell.totalNo_ofHandover_calls[n] =
AttributesCell.handover_calls_accepted[n] +
AttributesCell.handover_calls_dropped[n];
        } else
        {
             ++AttributesCell.CompletedNewCalls[n];
             AttributesCell.no_ofOriginated_calls_accepted_copy[n] =
( AttributesCell.no_ofOriginated_calls_accepted_copy[n] - 1 < 0 ? 0 :
AttributesCell.no_ofOriginated_calls_accepted_copy[n] - 1);
            print_Output("one of New calls has just
completed....Still Remaining ............" +
AttributesCell.no_ofOriginated_calls_accepted_copy[n], n);
             AttributesCell.totalNo_ofOriginated_calls[n] =
AttributesCell.no_ofOriginated_calls_accepted[n] +
AttributesCell.no_ofOriginated_calls_rejected[n];
        }
        print_Output("The total number of available channels: " + (
AttributesCell.total_available_channels -
AttributesCell.ongoingCalls[n]), n);
    }

    /**
     * the method accepts 2 parameters
     * @param generated_random_num Is the generated random number
     * @param n   Is the index number
     * the method computes the generated_random _num to decide if the
call type is
     * new call or handover
     */
   public synchronized void newCall_handover_generated(int
generated_random_num, int n) {

         AttributesCell.ongoingCalls[n] =
AttributesCell.handover_calls_accepted_copy[n]
                 +
AttributesCell.no_ofOriginated_calls_accepted_copy[n];
```

```
        print_Output("The total number of ongoing calls are: " +
AttributesCell.ongoingCalls[n], n);
        if (generated_random_num % 2 == 0) {

            Type_of_Call = "New Call";
        } else

        {
            Type_of_Call = "Handover";
        }


        if (Type_of_Call.equals("Handover")) {

            handoverProcess(n);
        } else {
            newCallProcess(n);
        }
    }

   /**
    * this method accepts one parameter
    * @param n  Is the Index number
    * the method increments handoff calls and checks if the guard and
shared channels
    * are available
    */
   public synchronized void handoverProcess(int n) {

        ++ AttributesCell.incomingHanodverCalls[n];
        if ( AttributesCell.ongoingCalls[n] <=
AttributesCell.total_available_channels) {
             AttributesCell.handover_calls_accepted[n] =
AttributesCell.handover_calls_accepted[n] + 1;
            ++ AttributesCell.handover_calls_accepted_copy[n];//
            print_Output("Handover Call >> " +
AttributesCell.incomingHanodverCalls[n] + " has arrived and was
accepted!", n);
             AttributesCell.N = 0;
            incrementOngoingCalls(n);
            //print_Output("The total number of handoff calls
admitted are: " +  AttributesCell.handover_calls_accepted_copy[n],
n);
            print_Output("The total number of handover calls admitted
are: " +  AttributesCell.handover_calls_accepted[n], n);
             AttributesCell.handoverTimers[n] = new Timer(
AttributesCell.HandoverCall_timeDelay,  this);
             AttributesCell.handoverTimers[n].start();
        } else {
             AttributesCell.handover_call_dropping_prob[n] =
AttributesCell.handover_call_dropping_prob[n] + 1;
             AttributesCell.handover_calls_dropped[n] =
AttributesCell.handover_calls_dropped[n] + 1;
           alarmInvoked(true, n);
            print_Output("Handover call >> " +
AttributesCell.incomingHanodverCalls[n] + " has arived and was
dropped!", n);
            print_Output("All handover calls dropped are: " +
AttributesCell.handover_calls_dropped[n], n);
        }
```

```
    }

   /**
    * this method accepts one parameter
    * @param n is the index number
      handles the new calls increments OriginatedCalls and checks if
      the shared channels are available
    */
   public synchronized void newCallProcess(int n) {
         AttributesCell.OriginatedCalls[n] =
AttributesCell.OriginatedCalls[n] + 1;
        if ( AttributesCell.ongoingCalls[n] <=
AttributesCell.shared_channels[n]) {
            ++ AttributesCell.no_ofOriginated_calls_accepted[n];
            ++ AttributesCell.no_ofOriginated_calls_accepted_copy[n];
            incrementOngoingCalls(n);
            print_Output("New originated call >>" +
AttributesCell.OriginatedCalls[n]
                    + " has arrived and was admitted!", n);
            print_Output("The total number of new calls admitted are:
"
                    +
AttributesCell.no_ofOriginated_calls_accepted[n], n);
             AttributesCell.call_timers[n] = new Timer(
AttributesCell.newCall_timeDelay, this);
             AttributesCell.call_timers[n].start();
        } else {

            ++ AttributesCell.no_ofOriginated_calls_rejected[n];
            print_Output("New call >>" +
AttributesCell.OriginatedCalls[n] + " has arrived nd was rejected!",
n);
            print_Output("All new Originated calls rejected are: " +
AttributesCell.no_ofOriginated_calls_rejected[n], n);
             AttributesCell.Originated_call_blocking_prob =
AttributesCell.Originated_call_blocking_prob + 1;
        }
    }

   /**
    * the method accepts two parameters
    * @param isDrooped
    * @param n  index number
    * if the handoff is blocked the method is invoked
    * the method compares the threshold with the standard upper limit
and lower limit threshold
    * in which decides to add the number of guard channels if its
beyond upper limit or reduce
    * guard channels if its below the lower limit
    */
   public synchronized void alarmInvoked(boolean isDrooped, int n) {

        AttributesCell.totalNo_ofHandover_calls[n] =
AttributesCell.handover_calls_accepted[n]
                +  AttributesCell.handover_calls_dropped[n];
        AttributesCell.current_handover_call_dropping_prob[n] =
(float)  AttributesCell.handover_calls_dropped[n]
                /  AttributesCell.totalNo_ofHandover_calls[n];
        Random random_num = new Random(1);
        double random_double = random_num.nextDouble();
```

```
        BigDecimal db = new
BigDecimal(Double.toString(random_double));
        db = db.setScale(1, BigDecimal.ROUND_CEILING);
        double up_random_double = db.doubleValue();
        print_Output(
AttributesCell.current_handover_call_dropping_prob[n]
                + "- current_handover_call_dropping_prob and
random_double=" + up_random_double, n);
        System.out.println(
AttributesCell.current_handover_call_dropping_prob[n]
                + "- current_handover_call_dropping_prob and lower
limitter=" + up_random_double);

        if ( AttributesCell.current_handover_call_dropping_prob[n] >=
AttributesCell.upperLimitTreshold) {

            print_Output("guard channel is increased", n);
            if ((up_random_double >= 1 -
AttributesCell.probable_increase) && (isDrooped)) {

                 AttributesCell.reserved_channels[n] = Math.min(
AttributesCell.reserved_channels[n]
                         + 1,
AttributesCell.maximum_number_of_guard_channels);
                 AttributesCell.shared_channels[n] =
AttributesCell.total_available_channels
                         - ( AttributesCell.reserved_channels[n]);
                print_Output("New Guard channels is reserved for
handover calls, and no of guard channels now is" +
AttributesCell.reserved_channels[n]
                        + " and number of shared channels is " +
AttributesCell.shared_channels[n], n);
                System.out.println("New Guard channels is reserved
for handover calls, and no of guard channels now is" +
AttributesCell.reserved_channels[n]
                        + " and number of sharedchannels is " +
AttributesCell.shared_channels[n]);
            }
        } else if (up_random_double >=
AttributesCell.probable_increase) {

             AttributesCell.reserved_channels[n] = Math.max(
AttributesCell.reserved_channels[n] - 1,

AttributesCell.minimum_number_of_guard_channels);
             AttributesCell.shared_channels[n] =
AttributesCell.total_available_channels
                     - ( AttributesCell.reserved_channels[n]);

            print_Output("New Guard channels is released, and no of
guard channels now is " +  AttributesCell.reserved_channels[n]
                    + " and number of shared channels is " +
AttributesCell.shared_channels[n], n);
            System.out.println("New Guard channels is released, and
no of guard channels now is " +  AttributesCell.reserved_channels[n]
                    + " and number of shared channels is " +
AttributesCell.shared_channels[n]);
        }
    }

   /**
```

```
     * the method accepts two parameters
     * @param msg is the message
     * @param n is the index number
     */
    public void print_Output(String msg, int n) {

         AttributesCell. data_store[n].append(msg + "\n");
         AttributesCell. data_store[n].append(n);
     }

    /**
     * method accepts one parameter
     * @param n is the index number
     * increments the number of ongoing calls  ongoingCalls[n]
     */
    public synchronized void incrementOngoingCalls(int n)
     {
         ++ AttributesCell.ongoingCalls[n];
     }

    /**
     * the method is used to generate new calls by generating
     * new random numbers between 1 and 2
     * @param limiter is used to limit the random selection of numbers
to 1 and 2
     * @return  returns the result number
     */
    public int randomNumGenerator(double limiter) {
         int random_num = 0;

         double random_nums = Math.exp(-limiter);
         double aggr = 1;
         Random random = new Random();
         while (aggr > random_nums) {
             aggr *= random.nextDouble();
             random_num++;
         }
         return random_num -1;
     }

    /**
     * the method implements the formulas used in the program
     * @param n  is the index number
     */
    public synchronized void formulas(int n) {

         AttributesCell.all_originate_calls[n] =
AttributesCell.no_ofOriginated_calls_accepted[n]
                 +  AttributesCell.no_ofOriginated_calls_rejected[n];
         AttributesCell.all_attempted_calls[n] =
AttributesCell.incomingHanodverCalls[n]
                 +  AttributesCell.OriginatedCalls[n];
         AttributesCell.Call_setup_failure_rate[n] =
AttributesCell.no_ofOriginated_calls_rejected[n]
                 /  AttributesCell.all_attempted_calls[n]*100;
         AttributesCell.handover_failure_rate[n] =
(AttributesCell.handover_calls_dropped[n]
                 /  AttributesCell.all_attempted_calls[n])*100;
          AttributesCell.traffic_load[n] = ((
AttributesCell.handover_calls_accepted[n]
```

```
                 +  AttributesCell.no_ofOriginated_calls_accepted[n])
/  AttributesCell.all_attempted_calls[n]) * 100;

                AttributesCell.successrates[n] =100 -
AttributesCell.Call_setup_failure_rate[n];

        print_Output("----------------------------------------------
", n);
        print_Output(" Total number of calls processed = " +
AttributesCell.all_attempted_calls[n], n);
        print_Output( AttributesCell.OriginatedCalls[n]
                 + " New Calls and Handover Calls = " +
AttributesCell.incomingHanodverCalls[n], n);
        print_Output( AttributesCell.CompletedNewCalls[n]
                 + "= Completed New Originated Calls and Completed
Handover Calls = " +  AttributesCell.CompletedHandoverCalls[n],
n);//Abubakar.........
        print_Output(" No of New Calls blocked = " +
AttributesCell.no_ofOriginated_calls_rejected[n], n);
        print_Output(" No of handover calls dropped = " +
AttributesCell.handover_calls_dropped[n], n);
        print_Output(" Total number of all calls accepted =" + (
AttributesCell.handover_calls_accepted[n]
                +  AttributesCell.no_ofOriginated_calls_accepted[n]),
n);
        print_Output(" Call Setup Success Rate =  " +
AttributesCell.successrates[n], n);
        print_Output(" Call Setup Failure Rate = " +
AttributesCell.Call_setup_failure_rate[n], n);
        print_Output(" New Call Blocking Probability =  " +
AttributesCell.Call_setup_failure_rate[n]/100, n);
        print_Output(" Handover Failure Rate = " +
AttributesCell.handover_failure_rate[n], n);
        print_Output(" Handover Call Dropping Probability =  " +
AttributesCell.handover_failure_rate[n]/100, n);
        print_Output(" Channel Utilization = " +
AttributesCell.traffic_load[n], n);
        print_Output("----------------------------------------------
", n);

        /**System.out.println(" all_attempted_calls =" +
AttributesCell. all_attempted_calls[n]);
        System.out.println(" no_ofOriginated_calls_rejected=" +
AttributesCell.no_ofOriginated_calls_rejected[n]);
        System.out.println(" handover_calls_dropped=" +
AttributesCell.handover_calls_dropped[n]);
        System.out.println("
handover_calls_accepted+no_ofOriginated_calls_accepted=" + (
AttributesCell.handover_calls_accepted[n] +
AttributesCell.no_ofOriginated_calls_accepted[n]));
        System.out.println(" call Setup failure rate" +
AttributesCell.Call_setup_failure_rate[n]);
        System.out.println(" Handover Failure rate: " +
AttributesCell.handover_failure_rate[n]);
        System.out.println(" channel Utilization: " +
AttributesCell.traffic_load[n]);
        System.out.println(" call setup success rate is  " +
AttributesCell.successrates[n]);*/
        System.out.println(" Time left is" +
AttributesCell.all_period_times[n]);
```

```
        tableOutput(n);

    }

      /**
      * this method below accepts one parameter which is an index
number
      * @param n  index number is used to begin specific timers in
the stopProgramTime array
      * the guard channels is initialized to two and open channels to
the rest number of channels
      */
       public synchronized void timed_simulation_starter(int n) {

         AttributesCell.timePeriod = Integer.parseInt(
AttributesCell.all_period_times[n].trim());
         AttributesCell.stopProgramTime[n] = new Timer(1000 * 60 *
AttributesCell.timePeriod, this);
         AttributesCell.stopProgramTime[n].setRepeats(false);
         System.out.println("Thread.. "+n+ " ..will run for
"+AttributesCell.timePeriod+"  minutes");
            try {

                AttributesCell. reserved_channels[n] = 2;

print_Output(">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>", n);
                print_Output("Simulating for time period " +
AttributesCell.all_period_times[n]
                        + " minutes and " +
AttributesCell.total_available_channels
                        + " Number of Channels and number of Guard
Channels is: " +  AttributesCell.reserved_channels[n], n);

print_Output(">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>", n);

                AttributesCell.shared_channels[n] =
AttributesCell.total_available_channels -
AttributesCell.reserved_channels[n];
                print_Output("the no of cahnnels entered by the user
are: " +  AttributesCell.total_available_channels, n);
                print_Output("The shared Channels are: " +
AttributesCell.shared_channels[n], n);

                AttributesCell. stopProgramTime[n].start();

                dynamic_guardChannel_algorithm(n);
            } catch (Exception ex) {
                System.out.println("timed_simulation_starter class
error" + ex);
            }
    }


        public synchronized void tableOutput(int n) {

            float handoff_success_rate=(100-
AttributesCell.handover_failure_rate[n]);
            float handoff_failure_rates =
AttributesCell.handover_failure_rate[n]/100;
            Vector<Object> gen_table = new Vector<Object>();
```

```
            gen_table.addElement(n);
            gen_table.addElement(handoff_success_rate);
            gen_table.addElement(AttributesCell.successrates[n]);
            gen_table.addElement(
AttributesCell.Call_setup_failure_rate[n]/100);
            gen_table.addElement( handoff_failure_rates);
            gen_table.addElement( AttributesCell.traffic_load[n]);

gen_table.addElement(Integer.parseInt(AttributesCell.all_period_times
[n]) * 1000);

            int lims =
Integer.parseInt(AttributesCell.all_period_times[n]);
            AttributesCell.timePeriod = Integer.parseInt(
AttributesCell.all_period_times[n].trim());
            int lims2 = AttributesCell.stopProgramTime.length;
            int lims3 = lims2 ;

            System.out.println("Stop time two: " +
AttributesCell.stopProgramTime.length);
            System.out.println("Stopps 2: " + lims3);

            AttributesCell.tableCells.addElement(gen_table);
            {

                DefaultTableModel model = (DefaultTableModel)
display_table.getModel();

                model.addRow(gen_table);

            }
            sss++;
            System.out.println(sss + " Number of SSSSSS");
            boolean is_excuting = true;
            System.out.println("One ended");
            AttributesCell.
all_period_timesBasket.removeElement(Integer.parseInt(
AttributesCell.all_period_times[n]));

AttributesCell.stopProgramTime[n].removeActionListener(this);
            AttributesCell.stopProgramTime[n].stop();
            System.out.println("The " +n+ "th thread
called"+AttributesCell.stopProgramTime[n]+ " has stopped");

            display_output_textfield.append(
AttributesCell.data_store[n] + "\n");
            System.out.println("And the buffer contains: " +
AttributesCell.data_store[n]);
            System.out.println("Process Stopped " + n +
"..........");
            System.out.println("And the calls contain: " +
AttributesCell.tableCells + "\n");
            AttributesCell.data_store[n].append(
AttributesCell.tableCells);

            try {
                progress_bar.setValue(100);
            AttributesCell.stopProgramTime[n].wait();

            } catch (InterruptedException ex) {
```

```
            Logger.getLogger(One.class.getName()).log(Level.SEVERE,
null, ex);
            }

            if ( AttributesCell.all_period_timesBasket.isEmpty())
            {
                is_excuting = false;

                AttributesCell.stopProgramTime[n].stop();
                AttributesCell.handoverTimers[n].stop();
                AttributesCell.call_timers[n].stop();
                shutdown();
                // System.exit(0);
                System.out.println("System has reached exit");
                System.out.println("\t");
                System.out.println("System has reached exit");
                System.out.println("\n");
                System.out.println("System has reached exit");
                System.out.println("\t");
                System.out.println("System has reached exit");
                progress_bar.setValue(100);
            }
            if (!is_excuting) {

                System.out.println("THese are all the tableCells " +
AttributesCell.tableCells);


            }
        }

      /**
      * @param args the command line arguments
      */
     public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel
setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay
with the default look and feel.
         * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.h
tml
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(One.class.getName()).log(java.util
.logging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(One.class.getName()).log(java.util
.logging.Level.SEVERE, null, ex);
```

```
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(One.class.getName()).log(java.util
.logging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(One.class.getName()).log(java.util
.logging.Level.SEVERE, null, ex);
        }
        //</editor-fold>
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new One().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify
    private javax.swing.JPanel background_panel;
    private javax.swing.JLabel channel_label;
    private javax.swing.JTextField channel_textfield;
    private javax.swing.JTextArea display_output_textfield;
    private javax.swing.JTable display_table;
    private javax.swing.JLabel generate_label;
    private javax.swing.JButton jButton1;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JScrollPane jScrollPane2;
    private javax.swing.JProgressBar progress_bar;
    private javax.swing.JLabel simulation_label;
    private javax.swing.JLabel simulation_time_is;
    private javax.swing.JTextField simulations_textfield;
    private javax.swing.JButton start_simulation_button;
    private javax.swing.JScrollPane table_output_field;
    private javax.swing.JLabel time_running;
    // End of variables declaration


    @Override
    public void actionPerformed(ActionEvent evt) {

         /**
         * the flow construct below checks if the event generated is
an object
         * of Timer type and checks if the timer is stopProgramTime[]
timer
         * if the timers have expired methods formulas() passing to
it
         * @param n which is the index number
         */
          if (evt.getSource() == start_simulation_button) {
              start_simulation_buttonActionPerformed(evt);
          }

         /**
```

```
         * the if construct checks if the event object is generated
by a timer in the timer_generation[]
         * and performs the body function which generates new calls
either handovers or new calls  if the thread
         * is still running
         */
          if (evt.getSource() instanceof Timer) {

            int n = 0;
            while (n <  AttributesCell.stopProgramTime.length) {
                //System.out.println("n is A " + n);
                if (evt.getSource() ==
AttributesCell.timer_generation[n]) {

                   //System.out.println("n is B " + n);//prints out
the message and variable n

                    if (
AttributesCell.stopProgramTime[n].isRunning()) {

                        int random_num =
randomNumGenerator(random.nextDouble());
                        int temp = 100 /
AttributesCell.all_period_times.length;
                        int g = (Integer.parseInt(
AttributesCell.all_period_times[n]) * 10000 * 60) /
AttributesCell.call_generation_time_delay;
                        double chunck = 0.41;
                         AttributesCell.packet += chunck;
                         //Chunk is a value for the proogress bar
which moves by multiples of 0.41 to be visible
                        int cur_value = ((temp * n) +
AttributesCell.packet);
                       System.out.println(
AttributesCell.all_period_times[n]
                                + ">>>>>>" + n + ">>>>>>>." + (temp /
g) + "" + chunck + " " + cur_value);
                         System.out.println(
AttributesCell.all_period_times.length);
                        progress_bar.setValue(cur_value);
                        newCall_handover_generated(random_num, n);
                    }
                }
                ++n;
            }
        }


      if (evt.getSource() instanceof Timer) {

            for (int n = 0; n <
AttributesCell.stopProgramTime.length; n++) {

                if (evt.getSource() ==
AttributesCell.stopProgramTime[n]) {

                    formulas(n);

                    System.out.println("BALK NABKSS : " + n);
                    System.out.println("Stop time : " +
AttributesCell.stopProgramTime.length);
```

```
                    /**
                     * the attributes initialized here are array data
structures
                     * and the value in each index is initialized  to
the following
                     */
                AttributesCell.handoverTimers[n] = null;
                AttributesCell.call_timers[n] = null;
                AttributesCell.no_ofOriginated_calls_accepted_copy[n]
= 0;
                AttributesCell.no_ofOriginated_calls_accepted[n] = 0;
                AttributesCell.handover_calls_accepted_copy[n] = 0;
                AttributesCell.handover_calls_accepted[n] = 0;
                AttributesCell.CompletedHandoverCalls[n] = 0;
                AttributesCell.CompletedNewCalls[n] = 0;
                AttributesCell.OriginatedCalls[n] = 0;
                AttributesCell.ongoingCalls[n] = 0;
                AttributesCell.Originated_call_blocking_prob = 0;
                AttributesCell.handover_call_dropping_prob[n] = 0;
                AttributesCell.reserved_channels[n] = 0;
                AttributesCell.packet=0;

                    /**
                     * setting new values to the Progress bar
                     */
                    progress_bar.setValue((100 /
AttributesCell.all_period_times.length) * (n + 1));
                    ++n;

                    /**
                     * the if flow construct is used to start the
timers in  all_period_times[]
                     * using the method  timed_simulation_starter()
                     */
                    if (n < AttributesCell.all_period_times.length) {

                        timed_simulation_starter(n);
                    }
                }
            }
        }

        /**
         * also checks if the event is generated by a timer and if so
if the timer is a call timer
         * which is started when a call is admitted calls the method
all_completed_calls() to release
         *  channel resources
         */
        if (evt.getSource() instanceof Timer) {

            for (int n = 0; n <  AttributesCell.call_timers.length;
n++) {

                if (evt.getSource() ==
AttributesCell.call_timers[n]) {

                    if (
AttributesCell.stopProgramTime[n].isRunning() &&
AttributesCell.call_timers[n] != null) {
```

```
                        all_completed_calls(2, 0, n);
                    }
        //            System.out.println("THis is the TImer call
location is : === " + n);
                }
            }
        }

        /**
         * checks if the event generated is from a timer and if the
timer is a handover timer
         * this timer is started when a handover call is admitted
calls the method all_completed_calls()
         * to release the channel resource
         */
        if (evt.getSource() instanceof Timer) {

            for (int n = 0; n <
AttributesCell.handoverTimers.length; n++) {

                if (evt.getSource() ==
AttributesCell.handoverTimers[n]) {

                    if (
AttributesCell.stopProgramTime[n].isRunning() &&
AttributesCell.handoverTimers[n] != null) {
                        all_completed_calls(1, 0, n);
                    }
                }
        //  System.out.println("THis is the TImer Handoff location is
: === " + n);
            }
        //  System.out.println("THis is the TImer Handoff location is
: === " +  AttributesCell.handoverTimers.length);
        }

    }
}

abstract class cease implements Runnable{
    private volatile boolean exit = false;

    public void stop(){

        exit = true;
        System.out.println("Threds Halt here");
        System.exit(0);
    }
}
```