UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

SIMULACIÓN Y ANÁLISIS DE SCHEDULERS LTE PARA ESTIMACIÓN DEL MOS DEL SERVICIO VOIP

TESIS PARA OPTAR AL GRADO DE MAGISTER EN INGENIERÍA EN REDES DE COMUNICACIONES

JOHANNA RAFAELA ORTEGA BRIONES

PROFESOR GUÍA:
ALBERTO CASTRO ROJAS

MIEMBROS DE LA COMISIÓN:
CÉSAR AUGUSTO AZURDIA MEZA
JUAN EDUARDO PÉREZ RETAMALES

SANTIAGO DE CHILE
2016

RESUMEN DE LA TESIS PARA OPTAR AL GRADO DE MAGISTER EN REDES DE COMUNICACIONES
POR: Johanna Rafaela Ortega Briones
FECHA: 22/11/2016
PROFESOR GUIA: Alberto Castro Rojas

SIMULACIÓN Y ANÁLISIS DE SCHEDULERS LTE PARA ESTIMACIÓN DEL MOS DEL SERVICIO VOIP

En la actualidad las necesidades de los usuarios obligan a que la red LTE adopte nuevos procedimientos para incrementar el desempeño del sistema. Para esto, los mecanismos de "Scheduler" juegan un papel importante ya que son los responsables de distribuir los recursos de radio entre los diferentes móviles tomando en cuenta entre otras circunstancias las condiciones del canal y los requerimientos de calidad de servicio.

Existen varios estudios comparativos de los distintos métodos de *Radio Resource Management* (RRM) que analizan el desempeño de los distintos parámetros de los *schedulers*. En cambio, en este trabajo se tiene como objetivo principal el cuantificar los *schedulers* con un valor escalar que califique directamente la calidad del audio de una conversación con el desempeño de los *schedulers* en el sistema LTE.

Este trabajo tiene como finalidad obtener el valor estimado del *Mean Opinion Score* (MOS) de los schedulers: RR, PF, FD-MT, TD-MT, TTA, FD-TBFQ, TD-TBFQ, PSS y CQA. Estos schedulers son simulados en NS-3 y mediante el software estadístico "R" se revisan los datos para presentar un análisis estadístico y de esta manera generar modelos paramétricos y no paramétricos que intenten capturar la tendencia de los nueve schedulers simulados.

En cada Scheduler se obtienen dos valores de MOS. El primero en base a cálculos y modelos estadísticos desde la capa PHY hasta la capa IP, mientras el segundo se obtiene de la información proporcionada directamente por el Flow Monitor a nivel de aplicación.

Los resultados obtenidos indican que los valores del MOS mediante el primer método es considerablemente más bajo comparado con el MOS obtenido del Flow Monitor. El primero se interpreta como el MOS del sistema, mientras el segundo se categoriza como el MOS de los terminales, los cuales por defecto poseen un búfer que proporciona mayor calidad de servicio, pero con mayor variación en el tiempo, comparados al MOS del sistema.

Agradecimientos

Agradezco a Dios por brindarme la oportunidad y a todas las personas que de una u otra forma me permitieron culminar este logro.

A la Secretaría Nacional de Educación Superior, Ciencia, Tecnología e Innovación SENESCYT en representación del Estado Ecuatoriano por confiar en mis capacidades y otorgarme la Beca. A Miguel Vivanco y Emilia Albornoz por su gestión y apoyo en esa institución, también a Don Adolfo Tapia por su ayuda y diligencia en los trámites bancarios.

Agradezco a todos mis compañeros de maestría, por la amistad y el cariño brindado dentro y fuera del aula. Debo también agradecer al profesor Claudio Estévez por estar siempre pendiente buscando el bienestar personal y académico. A Lily muchas gracias por sus consejos y por la ayuda proporcionada en todos los trámites.

Al profesor Alberto Castro muchas gracias por el desafió planteado y por su guía para culminarlo, siempre comprendiendo y respetando mis tiempos y las etapas que estaba cursando.

Gracias mami, Vale y negra por estar pendientes a pesar de la distancia y a su modo apoyar para que culmine este proyecto.

A mi Javi mil gracias por inspirarme y permitirme probar a mi misma que si se puede y que todo esfuerzo tiene su recompensa.

Finalizo agradeciendo a Edu. Gracias por creer en mi, por tu apoyo, impulso, respeto y amor. Por emprender conmigo en esta travesía y siempre confiar en que “lo vamos a lograr”.

Tabla de Contenido

Índice de tablas

Índice de figuras

Acrónimos

AIC Akaike Information Criterion.

AMC Adaptive Modulation and Coding.

ARQ Automatic Repeat Request.

BET Blind Equal Throughput.

BLER Block Error Ratio.

CoIta Carrier over Interference to average.

COXPH Cox Proportional Hazards Regression.

CQA Channel and QoS Aware.

CQI Channel Quality Indication.

CSI Channel State Information.

DCI Downlink Control Information.

DFT Discrete Fourier Transform.

DRX Discontinuous Reception.

EARFCN EUTRA-Absolute Radio-Frequency Channel Number.

eNB evolved NodeB.

EPC Evolved Packet Core.

ETSI European Telecommunications Standards Institute.

E-UTRAN Evolved - Terrestrial Radio Access Network.

EXP/PF Exponential Rule/Proportional Fair.

EXP-RULE Exponential Rule.

FIFO First In First Out.

GAM Generalized Additive Models.

GBR Guaranteed Bit Rate.

GLM Generalized Linear Models.

HARQ Hybrid ARQ.

HOL Head-of-line.

HSS Home Subscriber Server.

IDFT Inverse Discrete Fourier Transform.

IMSI International Mobile Subscriber Identity.

ITU-T International Telecommunication Union-Telecommunications.

KPI Key Performance Indicator.

LENA LTE-EPC Network Simulator.

LOCFIT Local Regression.

LOESS Locally Weighted Scatterplot Smoothing.

LOG-RULE Logarithmic Rule.

LTE Long Term Evolution.

MAC Medium Access Control.

MARS Multivariate Adaptive Regression Splines.

MBR Maximum Bit Rate.

MCS Modulation and Coding Scheme.

MIMO Multiple-Input Multiple-Output.

M-LWDF Maximum-Largest Weighted Delay First.

MME Mobility Management Entity.

MOS Mean Opinion Score.

MPPDV Mean Packet to Packet Delay Variation.

MSIN Mobile Subscription Identification Number.

MT Maximum Throughput.

NAS Non-Access Stratum.

NRT No Real Time.

OCS Online Charging System.

OFCS Offline Charging System.

OFDM Orthogonal Frequency-Division Multiplexing.

OFDMA Orthogonal Frequency-Division Multiple Access.

PAPR Peak-to-Average Power Ratio.

PCC Policy and Charging Control.

PCEF Policy and Charging Enforcement Function.

PCRF Policy and Charging Rules Function.

PDCCH Physical Downlink Control Channel.

PDCP Packet Data Convergence Protocol.

PDN-GW Packet Data Network Gateway.

PDSCH Physical Downlink Shared Channel.

PESQ Perceptual Evaluation of Speech Quality.

PF Proportional Fair.

PFsch Proportional Fair Scheduled.

PHY Physical Layer.

PLMN Public Land Mobile Network.

POLYMARS Multivariate Adaptive Polynomial Spline Regression.

PPBP Poisson Pareto Burst Process.

PPR Projection Pursuit Regression.

PRB Physical Resource Block.

PSS Priority Set Scheduler.

PUCCH Physical Uplink Control Channel.

PUSCH Physical Uplink Shared Channel.

QoS Quality of Service.

RAN Radio Access Network.

RB Radio Bearer.

RBG Resource Block Group.

RLC Radio Link Control.

RNTI Radio Network Temporary Identifier.

RR Round Robin.

RRC Radio Resource Control.

RRM Radio Resource Management.

RSRP Reference Signal Received Power.

RSS Residual Sum of Squares.

RTP Real-time Transport Protocol.

SC-FDMA Single-Carrier Frequency-Division Multiple Access.

S-GW Serving Gateway.

SINR Signal-to-Interference-plus-Noise Ratio.

SM Smoothing Methods.

SON Self Organized Networks.

SPR Subscription Profile Repository.

SRB Signalling Radio Bearer.

TBFQ Token Bank Fair Queue.

TBR Target Bit Rate.

TIA Telecommunications Industry Association.

TTA Throughput to Average.

TTI Transmission Time Interval.

Capítulo 1

Introducción

Este capítulo inicia con la motivación para elaborar este trabajo de tesis, continúa indicando el objetivo general y cuáles son los objetivos específicos. Finaliza detallando brevemente la estructura que se utilizará en este trabajo.

1.1. Motivación

En la actualidad las necesidades de los usuarios obligan a que la red LTE adopte nuevos procedimientos para incrementar el desempeño del sistema. Para esto, los mecanismos de *"Schedulers"* juegan un papel importante porque son los responsables de distribuir los recursos de radio entre las diferentes estaciones tomando en cuenta entre otras circunstancias las condiciones del canal y los requerimientos de Calidad de Servicio (QoS). Existen varios estudios comparativos de los algoritmos de *schedulers* y las consecuencias dentro de los sistemas. Sin embargo, no existe un estudio donde se relacione o cuantifique los *schedulers* con un valor escalar que califique directamente la calidad de la conversación en el sistema, siendo ésta la principal motivación para desarrollar este trabajo de tesis, que permita relacionar a los *schedulers* con un valor estimado del MOS (*Mean Opinion Score*).

1.2. Objetivo General

- Simular y analizar los *scheduler* para obtener un valor estimado del MOS.

1.3. Objetivos Específicos

Dentro de los objetivos específicos se encuentran:

- Analizar el Estado del Arte de los *schedulers* y la función E-Model de estimación del MOS.
- Agrupar los algoritmos de *schedulers* según sus características.
- Simular los *schedulers* en una red VoLTE utilizando la herramienta NS-3.
- Realizar varios modelos estadísticos para los niveles PHY-MAC (*Scheduler*) y MAC–IP mediante la herramienta R, y utilizarlos en una función E-Model para estimar el MOS de los *scheduler*.
- Analizar los resultados obtenidos de las distintas simulaciones
- Conclusiones y Trabajos Futuros.

1.4. Estructura del documento

La estructura utilizada en este documento para exponer el trabajo realizado es la siguiente:

- **Capítulo 1. Introducción:** Corresponde a la descripción del tema y objetivos del trabajo.
- **Capítulo 2. Análisis del Estado del Arte:** Corresponde a la revisión general del sistema LTE. En este capítulo se explican los conceptos necesarios para la comprensión y contextualización de la red LTE. Se explica los elementos y el funcionamiento de los *schedulers* y la medición del MOS.
- **Capítulo 3. Algoritmos de *Scheduling*:** Según el estado del arte se agrupa los algoritmos de *schedulers*.
- **Capítulo 4. Simulación de los *schedulers* en NS-3:** Utilizando la herramienta NS-3, se simula algunos *schedulers* en un red VoLTE con todas sus características.
- **Capítulo 5. Análisis de Datos y Modelos:** Se realiza el análisis de datos y la obtención de varios modelos mediante el uso de software estadístico "R", con el fin de conseguir un MOS estimado de los *schedulers* que se han simulado.
- **Capítulo 6. Análisis de Resultados:** Se presenta los resultados de los modelos de las variables de interés y mediante un comparativo estadístico se elige el mejor el modelo, finalizando con una comparación entre *schedulers*.
- **Capítulo 7. Conclusiones y Trabajos Futuros:** Se enumeran las conclusiones del trabajo de tesis realizado y algunas propuestas de trabajos futuros.

Capítulo 2

Análisis del Estado del Arte

En este capítulo se presenta el marco teórico de los temas que se abordará en la tesis. Inicia con una descripción de la red LTE con sus elementos esenciales indicando la arquitectura de *Evolved Packet Core* (EPC) y de *Evolved UTRAN* (E-UTRAN); y resumiendo brevemente las capas del *stack* de protocolos. A continuación, se enfoca en el *scheduling*, los procedimientos de *Radio Resource Management* (RRM) encargado del *scheduling* y el proceso de *scheduling* como tal. Finaliza con una breve explicación del parámetro MOS y la herramienta utilizada para su cálculo.

2.1. Descripción de la red LTE

Las redes LTE (*Long Term Evolution*) fueron planeadas con requisitos muy ambiciosos sobre las características de las redes 3G las cuales fueron diseñadas principalmente para servicios de voz clásicos.

LTE tienen como objetivo duplicar la eficiencia espectral e incrementar la cobertura de la red en términos de *bit rates* para usuarios de terminales de última generación.

El sistema de LTE se basa en una arquitectura plana, conocida como "*Service Architecture Evolution*", con respecto a los sistemas 3G, tal como se indica en la Fig. 2.1 [1]. Esto garantiza un soporte de movilidad sin desconexión y una entrega de alta velocidad para datos y señalización [2].

La sección de *core* se denomina *Evolved Packet Core* (EPC) y la versión mejorada del acceso a la red es *Evolved UTRAN* (E-UTRAN) [3].

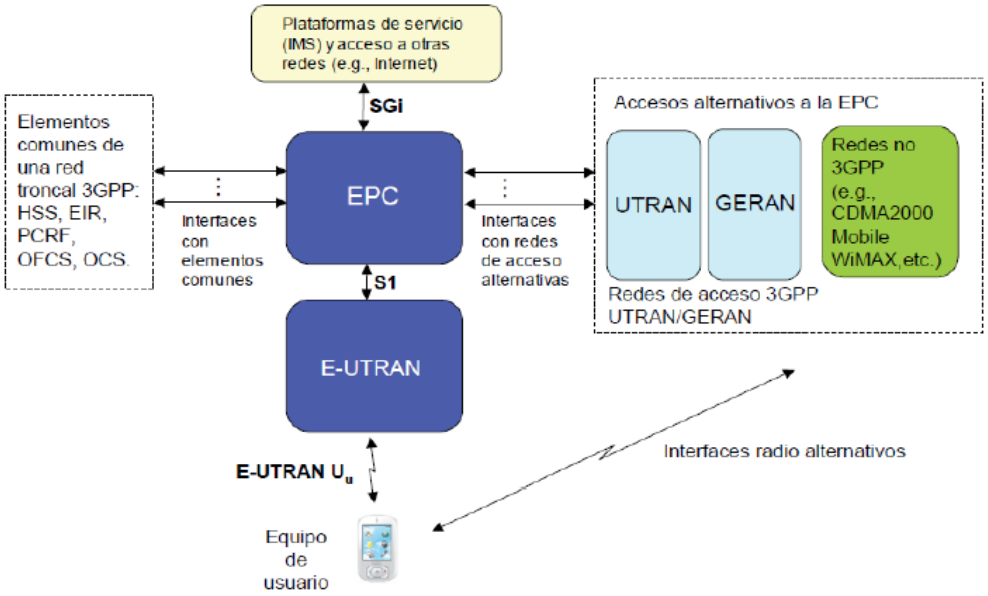

Figura 2.1: Arquitectura Básica de LTE.

2.1.1. Arquitectura de EPC

El EPC es la parte central de la red como se observa en la Fig. 2.2 [1]. Creado para proporcionar servicios de conectividad IP, permitiendo procedimientos de *handover* dentro y entre los tantos tipos de acceso que puede haber a través de redes que pertenezcan y no a las especificaciones 3GPP [4].

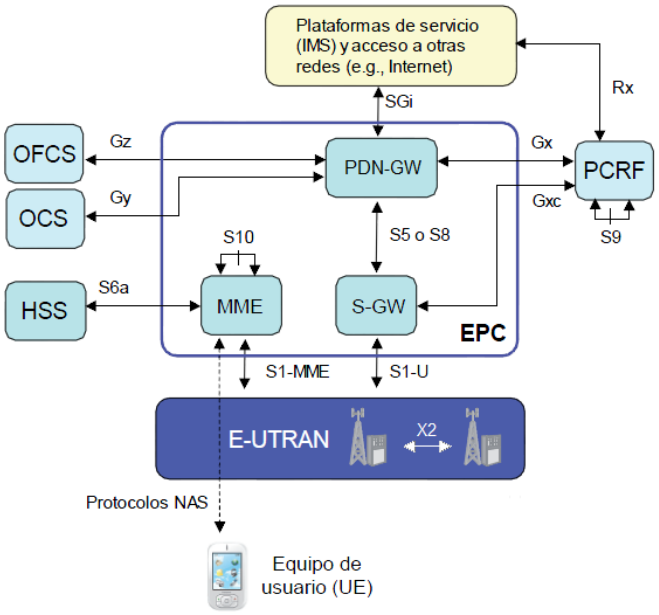

Figura 2.2: Arquitectura de EPC.

Los principales elementos funcionales son [5]:

- *Mobility Management Entity* (MME): es competente con los procedimientos de movilidad del usuario. Contempla la seguridad y señalización del NAS, selección del S-GW y PDN-GW, procedimientos de *tracking* y *paging* de los usuarios (UEs) cuando se establece la conexión, *handover* inter-LTE y autenticación con el HSS (*Home Subscriber Server*).

- *Serving Gateway* (S-GW): El propósito principal del S-GW es enrutar los paquetes de datos de usuario entre nodos LTE y entre redes LTE y otras tecnologías 3GPP. Además de realizar el *handover* inter eNB, almacenamiento de *buffering*, monitoreo para tecnología VoIP y otros servicios de paquetes.

- *Packet Data Network Gateway* (PDN-GW): Proporciona conectividad entre los UEs y las redes de datos externas. Entre sus funciones está: filtrar paquetes, monitoreo para tecnología VoIP y otros servicios de paquetes, asignación de direcciones IP, etiquetación de paquetes a nivel de transporte, control de volumen y tipo de tráfico.

- *Home Suscripción Server* (HSS): El servidor de suscripción es la base de datos principal del sistema. Abarca tanto información relativa a la subscripción del usuario como información necesaria para la propia operatividad de la red.

  En el HSS, se almacena una clave para ser utilizada junto con un *International Mobile Subscriber Identity* (IMSI) en la autenticación de suscriptores, y el perfil de QoS que va a utilizar el usuario. Por lo tanto, cuando los usuarios intentan acceder, el MME en nombre del HSS, niega a los usuarios no registrados, pero permite a aquellos con una IMSI registrada y válida al entregar información de autenticación y el perfil de QoS al MME. Un SPR trabaja con un PCRF para aplicar una política a un suscriptor [6].

- *Policy and Charging Resource Function* (PCRF): La función de política y carga de recursos (PCRF) es el elemento de red que se encarga de la política y control de carga o *Policy and Charging Control* (PCC). Toma decisiones sobre cómo manejar el servicio en términos de calidad de servicio, proporciona información a la PDN-GW, y controla las funcionalidades basadas en el flujo de carga en la función política de control de ejecución (PCEF), que reside en el PDN-GW [7]. PCRF es un servidor que normalmente se encuentra con otros elementos de la red troncal en los centros de conmutación de los operadores.

- *Offline Charging System* (OFCS y *Online Charging System* (OCS): Finalmente, las entidades OFCS y OCS constituyen el núcleo del sistema de tarifación de la red. Ambas entidades interactúan directamente con el puente PDN-GW mediante la interfaz Gz, en el caso de OFCS, y Gy, en el caso de OCS. El marco de tarifación soportado es un marco flexible que permite desplegar modelos de tarifación en base a diferentes parámetros tales como tiempo de uso, volumen de datos, eventos, etc.

En la Tabla 2.1 [1], se muestra en resumen las interfaces que lo comunicación con otras entidades.

INTERFAZ	DESCRIPCIÓN
S1-MME	MME / E-UTRAN (eNB)
S1-U	S-GW / E-UTRAN (eNB)
SGi	PDN-GW / Redes externas (IMS)
S6a	MME / HSS
S5	Dispositivos en la misma red PDN-GW S-GW
S8	Dispositivos en diferente red PDN-GW S-GW
S11	MME / S-GW
S10	MME
Señalización NAS	UE / MME
X2	eNB / eNB
Rx	PCRF Plataforma de servicios (IMS)
S9	PCRF
Gx/Gxc	PDN-GW / S-GW PCRF
Gz/Gy	PDN-GW OFCS/OCS

Tabla 2.1: Interfaces de EPC.

La red de acceso LTE puede tener solo dos nodos: el UE que es el terminal de usuario y el eNB que es la estación base. Los nodos eNB se conectan directamente entre ellos y con la MME. A diferencia de otras arquitecturas de red móvil, el eNB es el único dispositivo encargado de realizar tanto la gestión de recursos de radio y los procedimientos de control en la interfaz de radio [2].

2.1.2. Arquitectura E-UTRAN

La parte central de la arquitectura E-UTRAN se basa en eNB. Los eNBs proporcionan la conectividad entre los UE y la red troncal EPC, tal como se observa en la Fig. 2.3 [1].

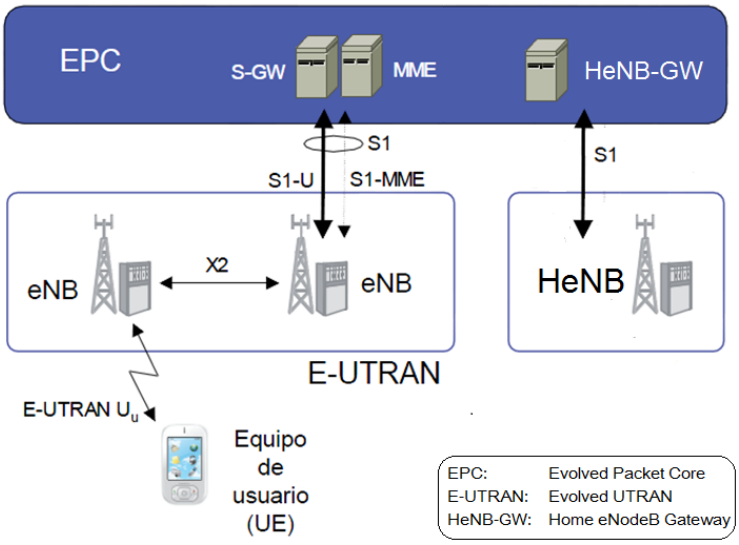

Figura 2.3: Arquitectura de E-UTRAN.

Stack de Protocolos de E-UTRAN

La separación entre plano de control y plano de usuario es una característica importante en el *stack* de protocolos asociados a las interfaces de la red LTE. Esta particular separación permite dimensionar de forma independiente los recursos de transmisión necesarios para el soporte de la señalización del sistema y para el envío de tráfico de los usuarios [1].

La arquitectura de E-UTRAN se divide en plano de usuario y de control, como se observa en la Fig. 2.4 [7], por lo que los eNBs proporcionan a la E-UTRAN los protocolos necesarios para cada plano. Estos planos soportan la misma capa física (PHY), la capa de enlace con control de acceso al medio (MAC), la capa de Control de Enlace de Radio (RLC) y Protocolo de Convergencia de Datos por Paquetes (PDCP). Pero para el plano de control se incluye el protocolo de Control de Recursos de Radio (RRC) y los protocolos NAS (*Non-Access Stratum*) [7].

- Capa PHY: Es la capa encargada de realizar la transmisión a través del canal de radio. Alberga funciones de codificación de canal, modulación, procesamiento asociado a las técnicas de transmisión/recepción de múltiples antenas (MIMO) y mapeo de la señal a los recursos físicos de frecuencia-tiempo apropiados. En el enlace ascendente, la capa física se basa en un esquema SC-FDMA. En el enlace descendente, el esquema de transmisión es OFDMA [1].

  A continuación se define brevemente los métodos de acceso que utiliza LTE para los enlaces ascendentes y descendentes:

  - OFDMA: La idea básica de OFDMA es asignar subportadoras a los usuarios en función de su tasa de bits. Con este método se facilita manejar al mismo tiempo las altas y baja tasas de usuarios en un único sistema, pero aun así, es difícil ejecutarla de manera eficiente con tráfico muy variable [8]. OFDMA es adecuado para los sistemas de datos de alta velocidad que operan en entornos de múltiples trayectorias debido a su capacidad para retrasar la propagación, eso se logra con el prefijo cíclico utilizado para mantener la ortogonalidad entre las subportadoras y operar en canales de múltiples trayectos.
  - SC-FDMA: Se puede considerar como una versión precodificada de OFDMA, mediante la transformada discreta de Fourier (DFT). Los símbolos en el dominio del tiempo modulados mediante un esquema M-QAM o M-PSK, pasan por un bloque DFT en el transmisor que convierte los símbolos al dominio de la frecuencia, esparciéndolos en todas las subportadoras que ocupan el ancho de banda destinado al usuario. Esta operación, evidentemente, deberá ser deshecha en el receptor, por lo que también se añade un bloque IDFT que no existía en OFDM. Haciendo esta operación de precodificación se logra un importante objetivo que es la reducción de *Peak-to-Average-Power Ratio* (PAPR) con respecto a OFDMA, debido a la naturaleza de portadora única de la señal resultante que es introducida al equipo antes de la transmisión. Debido a esta virtud, SC-FDMA es la técnica elegida en el enlace ascendente de la capa física de interfaz radio en LTE [9].

  Cabe recordar que en OFDMA cada sub-portadora sólo lleva la información relacionada con un símbolo específico y en SC-FDMA, cada sub-portadora contiene información de todos los símbolos transmitidos [8].

- Capa MAC:

  Es la capa encargada de controlar el acceso al canal de radio. Para ello, la capa MAC soporta

funciones de *scheduling* dinámico entre los equipos de usuario atendiendo las prioridades y multiplexación/de-multiplexación lógica de los paquetes de RLC (*Radio Link Control*) de los diferentes servicios portadores de radio en los canales de transporte ofrecidos por la capa física, además realiza un control de errores a través de HARQ (*Hybrid ARQ*). Los servicios de transferencia que la capa MAC ofrece a la capa RLC se denominan canales lógicos [1].

- Capa RLC:

  La capa RLC permite enviar de forma confiable los paquetes PDCP al eNB y al equipo de usuario. Soporta funciones de corrección de errores mediante mecanismos ARQ (*Automatic Repeat ReQuest*), concatenación, segmentación y re-ensamblado, entrega paquetes PDCP ordenados a capas superiores (excepto durante el mecanismo de *handover*), detección de duplicados y detección/recuperación de errores en el protocolo [1].

- Capa PDCP

  El Protocolo de convergencia de paquetes de datos, es la capa superior en el stack de protocolos encargada de proporcionar el punto de acceso al servicio radio portador (*Radio Bearer*, RB). Es decir, los paquetes de tráfico IP de usuario se entregan y se reciben a través del servicio de transferencia proporcionado por la capa PDCP. Las funciones principales de esta capa son la compresión de cabeceras de los paquetes IP y el cifrado de la información para garantizar su confidencialidad e integridad. La cabecera añadida por la capa PDCP básicamente contiene un número de secuencia que identifica al paquete IP enviado y permite realizar una entrega ordenada de los paquetes IP en el extremo receptor así como detectar posibles duplicados de los paquetes IP [1].

- Capa RRC:

  El protocolo de Control de Recursos de Radio permite establecer una conexión de control entre el eNB y el equipo de usuario a través del cual se llevan a cabo un importante número de funciones relacionadas con la gestión de la operatividad de la interfaz de radio. Entre dichas funciones de la capa RRC destacan los mecanismos de señalización para la creación, mantenimiento, modificación y publicación de la conexión RRC, funciones de movilidad como señalización de *handover*, funciones de gestión de QoS. El servicio de transferencia que ofrece la capa PDCP para el envío de los mensajes de señalización del protocolo RRC se denomina servicio portador de señalización (*Signalling Radio Bearer*, SRB) [1].

- Capa NAS:

  Las principales funciones de los protocolos NAS son: autenticación, autorización, gestión de movilidad de los terminales que no tienen una conexión RRC establecida y gestión de los servicios portadores de la red EPC. [1].

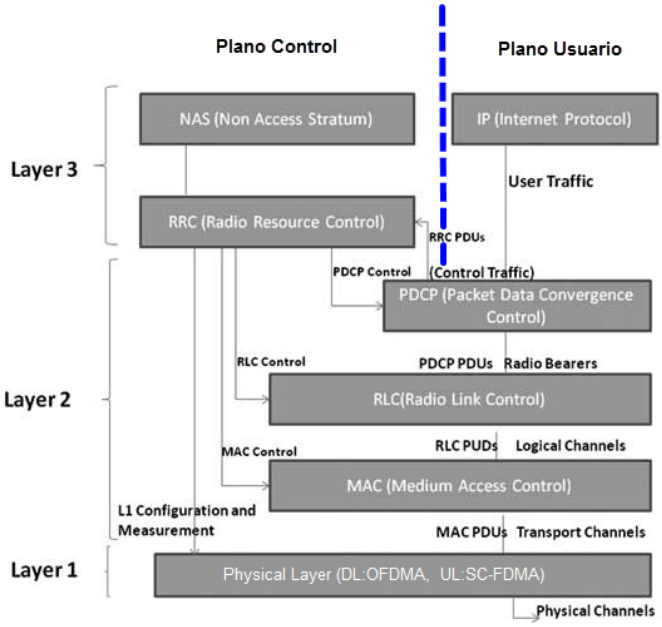

Figura 2.4: *Stack* de Protocolos E-UTRAN.

2.2. *Packet Scheduling*

La asignación de los recursos de radio a los diferentes usuarios es responsabilidad del mecanismo de *scheduling* de paquetes que funciona en la estación de base. En el caso de las redes LTE, un *scheduler* multi-usuario en el eNB asigna un subconjunto de subportadoras a los usuarios individuales los cuales permiten compartir el ancho de banda flexible en el sistema [10].

La red de acceso de radio LTE (RAN) se compone sólo de eNBs donde se realizan todas las funciones de RRM (Gestión de Recursos de Radio). Una de las funciones más significativas de RRM es el *scheduling* de paquetes que permite seleccionar inteligentemente a los UEs y transmitir sus paquetes para que los recursos de la interfaz aire se utilicen con eficacia y garantizando los requerimientos de calidad de servicio (QoS) de los usuarios [11].

En DL, la unidad más pequeña de tiempo/frecuencia de recursos de radio que se puede asignar a un usuario para la transmisión de datos se denomina bloque de recurso físico o PRB (*Physical Resource Block*). Los recursos de radio se definen en el dominio de tiempo/frecuencia. En el dominio del tiempo, la asignación de recursos en el enlace descendente LTE se lleva a cabo en cada TTI (*Transmission Time Interval*). El tiempo se divide en tramas y cada trama LTE está hecha de 10 TTI consecutivos. Cada TTI (1 ms) se compone de dos*timeslot* de duración de 0,5 ms. Cada time slot consta de 7 símbolos OFDM (con prefijo cíclico corto). En el dominio de la frecuencia, el total del ancho de banda disponible se divide en sub-canales, cada sub-canal corresponde a 12 subportadoras consecutivas y equidistantes con una separación de subportadora de 15 kHz (es decir, cada sub-canal es de $12 \times 15 = 180$kHz). Un RB extiende por más de un time slot de 0,5 ms en el dominio del tiempo y un sub-canal de 12 subportadoras en el dominio de la frecuencia. Dos RBs consecutivos (en el dominio del tiempo) se asignan a un UE para transmisión de datos en un

TTI [11].

El proceso de *scheduling* de paquetes está al comienzo de cada intervalo de tiempo de transmisión (TTI), durando dos *time slots* [12].

La QoS del sistema LTE está influenciado por varios factores, incluyendo las condiciones del canal. El eNB hace uso del *Channel State Information* (CSI) para obtener la información sobre las condiciones de los canales. Los usuarios reportan el SINR de acuerdo a las condiciones instantáneas del canal de enlace descendente para el eNB en cada TTI. La prioridad del *scheduling* de cada flujo se determinada por el *scheduling* de paquetes en el eNB, sobre la base de ciertos criterios de *scheduling* influenciados por varios factores como retrasos de paquetes HOL, condiciones del canal, estado del *buffer*, tipos de tráfico, y así sucesivamente [12].

2.2.1. *Radio Resource Management* (RRM)

Además de la distribución de recursos, LTE hace uso extensivo de los procedimientos RRM tales como: AMC (*Adaptive Modulation and Coding*), presentación de informes CQI (*Channel Quality Indicator*), canales físicos y HARQ [2].

- CQI *Reporting*:

  El procedimiento de la presentación de informes CQI es una característica fundamental de las redes LTE, ya que permite la estimación de la calidad del canal DL en el eNB. Cada CQI se calcula como una medida cuantificada y escalada de la experiencia de la interferencia de SINR. Sin embargo, el principal problema relacionado con el método de información CQI es encontrar un buen intercambio entre una estimación precisa de calidad del canal y la sobrecarga de señalización reducida.

- Adaptive Modulation and Coding (AMC):

  El procedimiento de presentación de informes CQI está estrictamente relacionada con el módulo AMC, que selecciona el esquema adecuado de modulación y codificación (MCS) tratando de maximizar el rendimiento apoyado con una determinada tasa de error de bloque (BLER). De este modo, un usuario que experimenta una SINR más alta será atendido con una tasa de bits más altas, mientras que un usuario de borde o en general un usuario que experimenta malas condiciones de canal mantendrán conexiones activas, pero a bajo rendimiento.

- Canales Físicos:

  Los datos del enlace DL se transmiten por el eNB través del canal PDSCH (*Physical Downlink Shared Channel*), como su nombre indica, se comparte entre todos los usuarios de la celda, en general no se realiza ninguna reserva de recursos en LTE. La transmisión del *payload* de PDSCH solo se permite en una determinada parte del espectro y en cierto tiempo entre la información de datos y la señalización del enlace descendente que son multiplexados en el tiempo dentro de la sub-trama.

  El canal PDCCH (*Physical Downlink Control Channel*) asigna los recursos DL y de soporte para el enlace descendente, además de los permisos de los enlace ascendente, incluidos el MCS usado. PDCCH, en particular, lleva un mensaje conocido como DCI (*Downlink Control*

Information), que transmite varias piezas de información dependiendo de la configuración específica del sistema.

En la dirección UL, se definen dos canales físicos: PUCCH (*Physical Uplink Control Channel*) y PUSCH (*Physical Uplink Shared Channel*). Debido a las limitaciones de la portadora simple por SC-FDMA, no se permite la transmisión simultánea en ambos canales. Cuando no se prevé la transmisión de datos de UL en un determinado TTI, PUCCH se utiliza para transmitir señalización (ejemplo: ACK / NACK relacionada con transmisiones de DL, CQI de enlace descendente, y peticiones de transmisión de UL). Por otro lado, PUSCH lleva las señales de control de UL cuando el UE ha sido programado para la transmisión de datos. En este caso los datos y los diferentes campos de control, tales como ACK / NACK y de CQI, son multiplexados en el tiempo en el *payload* del PUSCH.

- HARQ:

  El procedimiento de retransmisión en una capa MAC, se basa en el uso del algoritmo *stop and wait*. Este procedimiento se lleva a cabo simplemente por el eNB y el UE a través del intercambio de mensajes ACK / NACK. Un NACK se envía a través de PUCCH cuando un paquete transmitido por el enB no es decodificado con éxito en el UE. En este caso, el eNB realizará una retransmisión, enviando la misma copia del paquete perdido. Entonces, el UE tratará de decodificar el paquete mediante la combinación de la retransmisión de la versión original, y enviará el mensaje ACK al eNB a decodificar un éxito.

2.2.2. Proceso de *Scheduling*

Los principales módulos de RRM interactúan con el *scheduling* de paquetes del enlace descendente. Todo el proceso se puede dividir en una secuencia de operaciones que son repetidas en general cada TTI [13]:

1. Cada UE decodifica las señales de referencia, calcula el CQI y todo lo envía de vuelta al eNB.

2. El eNB utiliza la información de CQI para las tomar las decisiones de asignación y llena el mapa de asignación de RB.

3. El módulo AMC selecciona el mejor MCS que se debe utilizar para la transmisión de datos de los usuarios.

4. La información de estos usuarios sobre la asignación del RB y el MCS seleccionado se envían a los UEs sobre el PDCCH.

5. Cada UE lee la carga útil de PDCCH y en caso de que se ha programado, accede a la carga útil mediante el adecuado PDSCH.

El proceso se puede representar con la Fig. 2.5 [2].

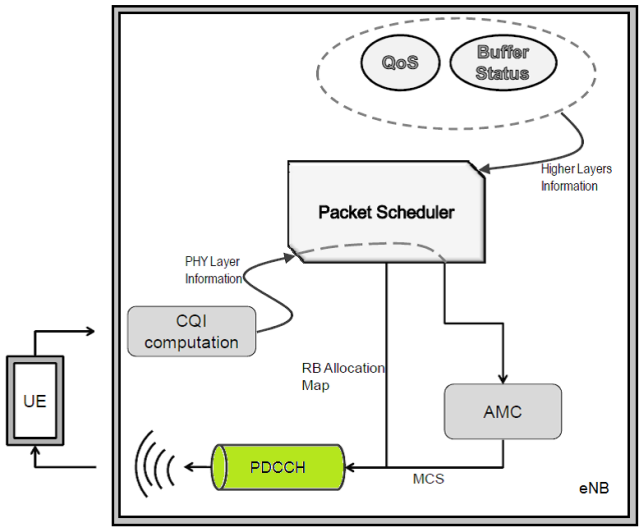

Figura 2.5: Modelo Simplicado de *Scheduling*.

2.3. MOS

El MOS o *Mean Opion Score* es una prueba que se ha utilizado durante décadas en las redes de telefonía para obtener la opinión del usuario humano de la calidad de la red [14]. El valor del MOS puede tener diferentes interpretaciones tal como se describe en la Tabla 2.2 [15].

MOS	CALIDAD	DESCRIPCIÓN
5	Excelente	Imperceptible nivel de distorsión
4	Buena	Perceptible pero no molesto
3	Media	Algo molesto
2	Pobre	Molesto
1	Mala	Muy Molesto

Tabla 2.2: Mean Opinion Score.

Existen métodos subjetivos y objetivos para medir el MOS [16]:

- Panel de auditoría: los humanos escuchan un conjunto de muestras pregrabadas y asignan una puntuación a cada muestra. Esta puntuación puede estar en el rango del uno a cinco. Hay que tener en cuenta que el valor presentado por cada auditor puede ser diferente, porque se trata de un valor subjetivo. De hecho, el MOS depende de las condiciones de la prueba y de los participantes [15].
- Técnicas intrusivas: Se envía un archivo de audio pregrabado a través de la red utilizando el protocolo *Real-time Transport Protocol* (RTP) con algún códec de voz. El archivo de audio recibido se compara con el original y se da una estimación del MOS, por ejemplo la evaluación de la calidad vocal por percepción (PESQ) [15].

- Técnicas no intrusivas: Estas técnicas se utilizan en la monitorización de la calidad de las llamadas VoIP debido al inconveniente de las pruebas subjetivas y al desperdicio de recursos consumidos por técnicas intrusivas. Las técnicas no intrusivas son capaces de determinar la calidad de las llamadas VoIP basadas en métodos como el modelo-E o E-Model cuya salida proporciona un " Factor de calificación" o "R", que puede ser relacionado con el MOS.

2.3.1. E-MODEL

E-Model es modelo computacional descrito en la recomendación de la ITU-T para calcular el factor "R" asociado a cada flujo para obtener el MOS.

Este modelo también ha sido adoptado por la ETSI y por la TIA; y es el modelo de opinión más ampliamente difundido. El resultado primario del modelo es una cuantificación escalar de la calidad de audio que se estima percibirá un usuario. Una característica fundamental de este modelo es la utilización de factores de degradación de la transmisión que reflejen los efectos de los modernos dispositivos de procesamiento de señales. El E-model se calcula en base a varios parámetros medibles de la red, un parámetro R que puede relacionarse con el MOS de acuerdo como lo indica la Fig. 2.6 [16].

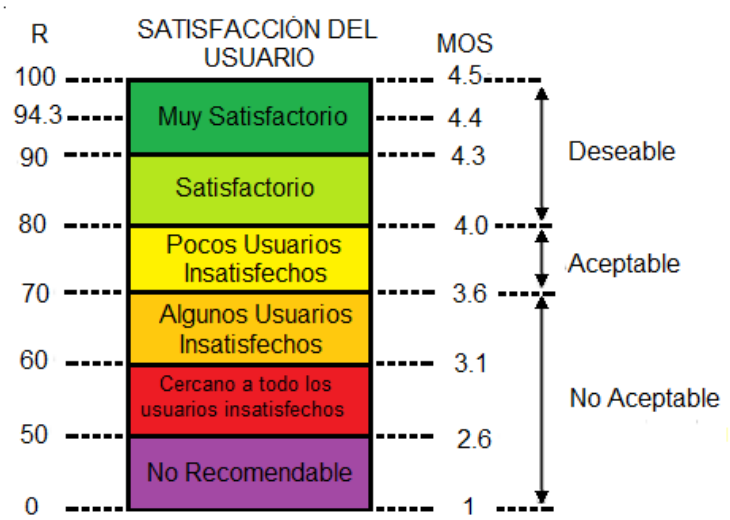

Figura 2.6: Relación R a MOS.

Los detalles del modelo fueron extraídos de la documentación oficial de la ITU-T para el E-model [17], mientras que la conversión a MOS y los parámetros específicos considerados como valores por defecto fueron tomados del trabajo de tesis doctoral presentado por Cristian Olariu en *Waterford Institute of Technology* [18].

Capítulo 3

Algoritmos de *Scheduling*

Los algoritmos de *scheduling* tienen como objetivo maximizar el rendimiento, satisfacer altas exigencias de los usuarios de servicios en tiempo real y mantener un buen equilibrio entre la asignación y el uso de recursos entre los flujos de tráfico [12].

Los algoritmos de *scheduling* se diferencian en parámetros de entrada, objetivos y metas de servicio. Para simplificarlos se ha clasificado en cuatro grupos de estrategias [13]:

1. Estrategias de Canales Inconscientes
2. Estrategias de Canales Conscientes sin soportar QoS
3. Estrategias de Canales Consientes con QoS
4. Estrategias de Canales Consciente de la energía

3.1. Estrategias de Canales Inconscientes

La estrategia de los canales inconscientes se basa en el supuesto que los medios de transmisión son invariantes en el tiempo y libre de errores. Aunque su aplicación directa en LTE no es realista, se utilizan típicamente en forma conjunta con los enfoques de canales consciente para mejorar el rendimiento del sistema [13].

First In First Out (FIFO)

Es el caso más simple de la política de asignación inconsciente del canal que sirve a los usuarios de acuerdo con el orden de las solicitudes de recursos, exactamente como una cola *First In First Out* (FIFO). El comportamiento de FIFO en LTE se puede expresar como la métrica del i-ésimo usuario en el k-ésimo RB, de tal manera [13]:

$$m_{(\mathrm{i},k)}^{FIFO} = t - T_{\mathrm{i}}, \tag{3.1}$$

donde:

$m_{(\mathrm{i},k)}$	métrica del i-ésimo usuario en el k-ésimo *Resource Block*
t	Es el tiempo actual
T_{i}	Es el instante de tiempo cuando la solicitud fue emitida por el i-ésimo usuario

Aunque esta técnica es muy simple, es ineficiente e injusta.

Round Robin (RR)

Este esquema de *scheduling* permite a los usuarios utilizar los recursos comunes en diferentes turnos, pero uno a la vez. Al comienzo, los usuarios son seleccionados aleatoriamente de una cola y los nuevos terminales se colocan al final de la cola. El primer usuario tiene todos los recursos disponibles por la entidad del *scheduling* para luego colocarse en la parte de atrás de la cola. El esquema de *Round Robin* se puede tomar como un *scheduler* justo, ya que la misma cantidad de tiempo o de recursos se entrega a cada usuario específico. La injusticia de esta combinación surge debido a que ofrecen la misma calidad de servicio a todos los usuarios del enlace [19].

La métrica de *Round Robin* (RR) métrica es similar a la definida para FIFO con la diferencia de que, en este caso, Ti se refiere al último momento en que se sirve al usuario [13].

$$m_{(\mathrm{i},k)}^{RR} = t - T_{\mathrm{i}}, \tag{3.2}$$

donde:

$m_{(\mathrm{i},k)}$	métrica del i-ésimo usuario en el k-ésimo *Resource Block*
t	Es el tiempo actual
T_{i}	Es el instante de tiempo que se atiende al i-ésimo usuario

Blind Equal Throughput (BET)

El rendimiento imparcial se puede lograr con *Blind Equal Throughput* (BET) que almacena previamente el rendimiento medio alcanzado por cada usuario y lo utiliza como métrica. En este caso la métrica (para el i-ésimo usuario) se calcula como [13]:

$$m_{(\mathrm{i},k)}^{BET} = \frac{1}{R_{\mathrm{i}}(t-1)}, \tag{3.3}$$

donde:

$m_{(\mathrm{i},k)}$	métrica del i-ésimo usuario en el k-ésimo *Resource Block*
$R_\mathrm{i}(t-1) = \beta * R_\mathrm{i}(t-1) + (1-\beta) r_\mathrm{i}(t)$	Denota la tasa de datos promedio de usuario a través de una ventana de tiempo y es una variable que va desde 0 a 1.
r_i	Significa la velocidad de datos alcanzable de acuerdo con la calidad del canal instantánea del usuario i en el instante $t - thTTI$.

Gracias a sus propiedades, esta métrica es ampliamente utilizada en la mayor parte del estado del arte de los schedulers ya que es fácil observar a cada TTI. Adicional, BET asigna los recursos a los flujos que se han atendido con menor rendimiento promedio en el pasado. En virtud de esta política de asignación, el usuario experimenta rendimiento más bajo. En la práctica el recurso será servido mientras no alcance el mismo rendimiento de otros usuarios en la celda. De esta forma, los usuarios con malas condiciones de canal se asignan con mayor frecuencia que otros, con la consiguiente mejora a la equidad [13].

3.2. Canal Consciente sin QoS

Debido a que los *feedbacks* de CQI se envían periódicamente (desde los UEs a la eNB) utilizando mensajes especiales de control, el *scheduler* puede estimar la calidad del canal percibido por cada UE. Por lo tanto, puede predecir el máximo rendimiento alcanzable.

El $\mathrm{d}^\mathrm{i}(t)$ y $\mathrm{d}^\mathrm{i}_k(t)$ son los rendimientos esperados alcanzables para el i-ésimo usuario en el t-ésimo TTI sobre todo el ancho de banda y sobre el k-ésimo RB, respectivamente. Los valores mencionados se pueden calcular utilizando el módulo AMC o simplemente estimando, considerando la expresión bien conocida de Shannon para la capacidad del canal, como:

$$\mathrm{d}^\mathrm{i}_k(t) = \log\left[1 + SINR^\mathrm{i}_k(t)\right]. \tag{3.4}$$

Esta definición da una explicación numérica de la relevancia de la conciencia del canal en contextos inalámbricos [13].

Maximum Throughput (MT)

La estrategia conocida como Máximo Rendimiento (MT) tiene como objetivo maximizar el rendimiento general mediante la asignación de cada RB al usuario que puede alcanzar el máximo rendimiento en el actual TTI. Su métrica puede ser simplemente expresada como:

$$m^{MT}_{(\mathrm{i},k)} = \mathrm{d}^\mathrm{i}_k(t), \tag{3.5}$$

donde:

$m_{(\mathrm{i},k)}$	métrica del i-ésimo usuario en el k-ésimo *Resource Block*
$\mathrm{d}_k^{\mathrm{i}}(t)$	Se puede calcular con la expresión de Shannon
TTI	Mínimo periodo de tiempo corresponde a 1 ms

MT es capaz de maximizar el rendimiento de la celda, pero por otro lado, se realiza un injusto intercambio de recursos ya que los usuarios con malas condiciones de canal (por ejemplo, los usuarios con celda de borde o cell-edge) conseguirán solo un bajo porcentaje de los recursos disponibles (o, en caso extremo pueden sufrir de falta de recursos) [13].

Proportional Fair Scheduler (PF)

El algoritmo PF puede mantener efectivamente un buen balance entre la equidad de la cantidad de usuarios y el rendimiento total introducido en la redes de datos de alta velocidad. PF pertenece a una clase de estrategias de *scheduling* conscientes del canal, es adecuado para la programación de tráficos en tiempo no real. Tiene en cuenta tanto la experiencia de los estados de canal y la velocidad de datos anteriores cuando asigna recursos de radio. PF tiene como objetivo obtener rendimiento satisfactorio al mismo tiempo que garantizar la equidad entre los flujos. Sin embargo, PF no es una buena opción para la programación de tráfico en tiempo real debido a su característica de inconscientes de calidad de servicio. Un usuario se selecciona basándose en la siguiente ecuación:

$$m_{(\mathrm{i},k)}^{PF} = \frac{\mathrm{d}_{\mathrm{i}}^{k}(t)}{(R_{\mathrm{i}}(t-1))}, \tag{3.6}$$

donde:

$m_{(\mathrm{i},k)}$	métrica del i-ésimo usuario en el k-ésimo *Resource Block*
$R_{\mathrm{i}}(t-1) = \beta * R_{\mathrm{i}}(t-1) + (1-\beta) r_{\mathrm{i}}(t)$	Denota la tasa de datos promedio de usuario a través de una ventana de tiempo y es una variable que va desde 0 a 1.

La filosofía de la ecuación de PF es maximizar el equilibrio entre el rendimiento del sistema y tener equidad entre todos los usuarios. Por otra parte, el esquema *Proportional Fair* es un buen esquema para tráficos de tiempo no real, ya que puede alcanzar sustancialmente mucho mayor rendimiento del sistema que otros esquemas de programación de paquetes representativos como el sistema *Round Robin* [19].

***Throughput to Average* (TTA)**

TTA es un *scheduler channel-aware/QoS-unaware* [20]. Este algoritmo se puede considerar como intermedio entre MT y PF. Su métrica se expresa como [13] :

$$m_{(\mathrm{i},k)}^{TTA} = \frac{\mathrm{d}_k^{\mathrm{i}}(t)}{\mathrm{d}^{\mathrm{i}}(t)}, \tag{3.7}$$

donde:

$m_{(\mathrm{i},k)}$	métrica del i-ésimo usuario en el k-ésimo *Resource Block*
$\mathrm{d}_k^{\mathrm{i}}(t)$	Se puede calcular con la expresión de Shannon

El rendimiento alcanzable en el actual TTI se utiliza como factor de normalización del rendimiento alcanzable en el RB considerado. La ventaja de la asignación de un específico RB garantiza que los mejores RBs sean asignados a cada usuario. TTA permite un alto nivel de equidad en una ventana temporal de un solo TTI. De hecho, su métrica es fácil de ver, que cuanto mayor sea el rendimiento general esperado de un usuario, menor será la métrica de un solo bloque de recursos. Esto significa que un *scheduler* explota la consciencia del canal para garantizar un mínimo nivel de servicio a cada usuario [13].

3.3. Canal Consciente con QoS

La diferenciación de QoS se maneja mediante la asociación de un conjunto de parámetros de QoS para cada flujo. Conociendo los valores de dichos parámetros, el *scheduler* puede tratar a los datos para garantizar el mínimo desempeño, ya sea garantizar la tasa de datos o el retraso en la entrega. Es importante tener en cuenta que la consciencia de QoS no significa necesariamente provisión de QoS, ya que la toma de decisión consiste en la asignación de los requerimientos de cada flujo, sin garantizar necesariamente la reunión de tales requisitos, ya que podría ser inviable si los procedimientos para el control de admisión no se implementan [13].

3.3.1. *Schedulers* para Garantizar Data-Rate

Se propone una solución genérica consciente de QoS para los flujos que requieren velocidad de datos garantizada.

***Priority Set Scheduler* (PSS)**

PSS combina ambos dominios: tiempo y frecuencia en un solo *scheduler*. Controla la equidad entre los UEs mediante la definición de una específica velocidad de bits de destino (TBR) [21]. En

la parte TD del *scheduler*, PSS selecciona a los usuarios con la más alta prioridad. En particular, los usuarios con flujos por debajo de su tasa de bits de destino (es decir, aquellos que necesitan que se asignen con urgencia para cumplir con los requisitos de calidad de servicio) formando un conjunto de alta prioridad. El resto de los usuarios, en cambio, forman un conjunto de menor prioridad. Los usuarios que pertenecen a los primeros y segundos *sets* se gestionan mediante el uso de algoritmos BET y PF, respectivamente [13]. PSS seleccionar los UE con la mayor métrica en los dos conjuntos y transmite esos UEs a la parte FD del *scheduler*. En esta parte, el *scheduler* ubica a los RBG de los UE con métrica más larga. Para esto considera: *Proportional fair scheduled* (PFsch) y *Carrier over interference to average* (CoIta). Ambos sirven para desacoplar la métrica FD desde el *scheduler* TD [21]. Recordando la métrica para PF se calcula así:

$$m_{(\mathrm{i},k)}^{PF} = \frac{\mathrm{d}_{\mathrm{i}}^{k}(t)}{(R_{\mathrm{i}}(t-1))}, \tag{3.8}$$

donde:

$m_{(\mathrm{i},k)}$	métrica del i-ésimo usuario en el k-ésimo *Resource Block*
$R_{\mathrm{i}}(t-1) = \beta * R_{\mathrm{i}}(t-1) + (1-\beta) r_{\mathrm{i}}(t)$	Denota la tasa de datos promedio de usuario a través de una ventana de tiempo y es una variable que va desde 0 a 1.

Para BET la métrica es calculada mediante:

$$m_{(\mathrm{i},k)}^{BET} = \frac{1}{R_{\mathrm{i}}(t-1)}, \tag{3.9}$$

donde:

$m_{(\mathrm{i},k)}$	métrica del i-ésimo usuario en el k-ésimo *Resource Block*
$R_{\mathrm{i}}(t-1) = \beta * R_{\mathrm{i}}(t-1) + (1-\beta) r_{\mathrm{i}}(t)$	Denota la tasa de datos promedio de usuario a través de una ventana de tiempo y es una variable que va desde 0 a 1.
r_{i}	Significa la velocidad de datos alcanzable de acuerdo con la calidad del canal instantánea del usuario i en el instante $t - thTTI$.

Token Bank Fair Queue (TBFQ)

TBFQ es un algoritmo consciente del canal derivado del mecanismos *leaky-bucket*. En este algoritmo el flujo del tráfico de un UE i = 1, . . . ,N es caracterizado por seguir los siguientes parámetros:

t_{i}	*packet arrival rate*[byte/seg]
r_{i}	*token generation rate* [byte/seg]
p_{i}	*token pool size* [byte/seg]
E_{i}	contador que registra el número de tokens prestados desde o dado el banco de tokens por el flujo i; Ei puede ser menor que cero.

Cada k bytes de datos consume k tokens. TBFQ mantiene un banco de tokens compartido (B) con fin de equilibrar el tráfico entre los diferentes flujos. Si la tasa de generación de tokens r_{i} es más grande que la tasa de paquete de llegada t_{i}, entonces los tokens desbordados del *pool* de tokens se añaden al banco y E_{i} se incrementa en la misma cantidad. De lo contrario, el flujo i necesita retirar tokens de banco basado en la prioridad métrica

$$\frac{E_{\mathrm{i}}}{r_{\mathrm{i}}},$$

y luego se disminuye E_{i}. Obviamente, el UE que contribuye más al banco de tokens tiene una prioridad más alta para pedir prestado tokens; Por otro lado, el UE que toma más tokens desde el banco tiene una prioridad más baja para seguir a retirando tokens. Por lo tanto, en caso de que varios UEs tengan la misma tasa de generación de tokens, tasa de tráfico y el tamaño de pool de tokens, el UE que sufre la interferencia más alta tiene más oportunidad de pedir prestado tokens del banco. Además, TBFQ puede vigilar el tráfico mediante el establecimiento de la tasa de generación de tokens para limitar el rendimiento [21].

Channel and QoS Aware (CQA) [22]

Este algoritmo considera los siguientes parámetros de QoS: HOL y GBR. El *scheduler* CQA realiza su planificación de acuerdo a diferentes criterios en TD y FD con el fin de lograr una alta eficiencia espectral, al mismo tiempo que cuida en satisfacer los requisitos del retardo de tráfico.

El algoritmo CQA se ejecuta cada TTI que es igual a 1 ms. El TTI es el recurso más pequeño en el dominio del tiempo. En el domino de la frecuencia el recurso más pequeño es el bloque de recursos (RB), que forma grupos RB (RBGs).

En TD, en cada TTI, el *scheduler* CQA selecciona de todos los usuario $j = 1,...,N$ que todavía no han alcanzado la tasa máxima de bits (MBR) y los agrupa por retraso de HOL calculando las métricas TD de la siguiente forma:

$$m_{(j)}^{CQA} = \lceil \frac{\mathrm{d}_{HOL}^{j}(t)}{g} \rceil, \tag{3.10}$$

donde:

$\mathrm{d}^j HOL(t)$	Es el valor actual del retardo de paquetes HOL (Head Of Line) del flujo j
g	es el parámetro de agrupación que determina la granularidad de los grupos, es decir, el número de los flujos que se considerará en la iteración de la parte FD del *scheduling*

La agrupación es usada para seleccionar los flujos más urgentes, es decir, aquellos con el valor del *delay* de HOL más alto, y forzar al mecanismo de *scheduling* a considerar dichos flujos en la siguiente iteración de la parte FD del *scheduling*.

En la parte FD, para cada RBG $k = 1, ..., K$, el *scheduler* CQA asigna el actual RBG al usuario j que tiene un máximo valor de la métrica FD la cual se define de la siguiente forma:

$$m_{(k,j)}^{CQA} = \mathrm{d}^j HOL(t) \cdot m_{GBR}^j(t) \cdot m_{k,j}^{ca}(t), \tag{3.11}$$

donde:

$m_{GBR}^j(t) = \frac{GBR^j}{\overline{R^j(t)}} = \frac{GBR^j}{(1-\alpha)\cdot R^j(t-1)+\alpha\cdot r^j(t)}$	GBR^j es bit rate especificado en el EPS bearer del flujo j $\overline{R^j(t)}$ el es promedio del throughput pasado que es calculado con el movimiento promedio $r^j(t)$ es el throughput alcanzado al tiempo t y; α es el coeficiente que va desde $0 \leq \alpha \leq 1$.

El propósito de las métrica de d_{HOL} y m_{GBR} es proporcionar al flujo el mismo nivel de QoS referente al *delay* y al GBR mediante la priorización del flujo que tienen alto *delay* HOL y el flujo cuya porción del flujo de GBR^j al $\overline{R^j(t)}$ es grande.

3.3.2. *Schedulers* para Garantizar los requerimientos de retardo

Las estrategias que tienen por objetivo garantizar un retardo acotado son los más representativos dentro de la categoría de los esquemas consciente de calidad de servicio, ya que Uno de los principales requisitos para la QoS de paquetes limitados es que se entreguen dentro de un determinado plazo. Por supuesto, se utilizan para aplicaciones en tiempo real, por ejemplo como para la transmisión de vídeo y flujos de VoIP.

Maximum-Largest Weighted Delay First (M-LWDF) [11]

M-LWDF fue propuesto para soportar múltiples usuarios de datos en tiempo real con diferentes requisitos de calidad de servicio en el sistema CDMA-HDR. Un usuario está programado sobre la

base de las siguientes mediciones de prioridad, m.

$$m_{(\mathrm{i},k)}^{M-LDWF} = a_{\mathrm{i}} W_{\mathrm{i}}(t) \frac{r^{\mathrm{i}}(t)}{R^{\mathrm{i}}(t-1)}, \tag{3.12}$$

donde:

$W_{\mathrm{i}}(t)$	Es el retardo de paquetes HOL (Head Of Line) del usuario i al tiempo t.
$a_{\mathrm{i}} = -\frac{\log \delta_{\mathrm{i}}}{\tau_{\mathrm{i}}}$	τ_{i} es el umbral retraso del usuario i. Denota la máxima probabilidad que el retardo del paquete HOL del usuario i supera al umbral de retraso de usuario i.
$r^{\mathrm{i}}(t)$	Significa la velocidad de datos alcanzable de acuerdo con la calidad del canal instantánea del usuario i en el instante t-th TTI.
$R_{\mathrm{i}}(t-1) = \beta * R_{\mathrm{i}}(t-1) + (1-\beta) r_{\mathrm{i}}(t)$	Denota la tasa de datos promedio de usuario a través de una ventana de tiempo, y es una variable que va desde 0 a 1.

El retardo de paquetes HOL es la diferencia de tiempo entre el momento actual y el momento en que el paquete llegó. El esquema de M-LWDF considera retardo de paquetes HOL junto con las propiedades de PF mientras que prioriza los usuarios priorizar.

***Exponential Rule/Proportional Fair* (EXP/PF) [11]**

EXP / PF fue diseñado para soportar simultáneamente servicios de Tiempo Real con diferentes requisitos de calidad de servicio y servicios de datos NRT (No Real Time) en un sistema de modulación - codificación Adaptiva y Multiplexación por División de Tiempo (AMC/TDM). Es una combinación de dos algoritmos de *scheduling* denominados regla EXP y regla PF. Las propiedades de EXP garantiza las restricciones de retardo de los servicios RT y las propiedades PF maximizan el rendimiento del sistema. Para cada usuario, la métrica de *scheduling* depende del tipo de servicio del usuario. La métrica de prioridad m se conoce como:

$$m_{(\mathrm{i},k)}^{EXP/PF} = \mathrm{e}^{\frac{a_{\mathrm{i}} W_{\mathrm{i}}(t) - \chi}{1+\sqrt{\chi}}} \frac{r_{\mathrm{i}}(t)}{R_{\mathrm{i}}(t)} \qquad \mathrm{i}\ \varepsilon\ RT, \tag{3.13}$$

donde:

$m_{(\mathrm{i},k)}$	métrica del i-ésimo usuario en el k-ésimo *Resource Block*
$m_{(\mathrm{i},k)}$	métrica del i-ésimo usuario en el k-ésimo *Resource Block*
$\chi = \frac{1}{N_{RT}} \sum_{N_{RT}}^{\mathrm{i}=1} a_{\mathrm{i}} W_{\mathrm{i}}(t)$	Es el retardo de paquetes HOL (Head Of Line) del usuario i al tiempo t.
$w = \begin{cases} w(t-1) - \varepsilon & W_{max} > \tau_{max} \\ w(t-1) + \frac{\varepsilon}{\mu} & W_{max} < \tau_{max} \end{cases}$	ε y μ son constantes. W_{max} Denota el máximo retardo del paquete HOL de todos los usuarios del servicio RT.

Este algoritmo ofrece una mayor prioridad a los usuarios de los servicios de tiempo real si sus retrasos HOL están acercándose al plazo final.

Logarithmic Rule (LOG-RULE) [13]

$$m_{(\mathrm{i},k)}^{LOGrule} = b_{\mathrm{i}} \log(c + a_{\mathrm{i}} W_{\mathrm{i}}(t)) \Gamma_k^{\mathrm{i}}, \tag{3.14}$$

donde:

$m_{(\mathrm{i},k)}$	métrica del i-ésimo usuario en el k-ésimo *Resource Block*
$a_{\mathrm{i}}, b_{\mathrm{i}}, c$	Son parámetros ajustables.
Γ_k^{i}	Representa la eficiencia espectral para el usuario i -ésimo en el sub-canal-k.

Exponential Rule (EXP-RULE) [13]

La regla EXP se puede considerar como una mejora de la mencionada EXP / PF. Su métrica es similar a la de la regla LOG:

$$m_{(\mathrm{i},k)}^{EXPrule} = b_{\mathrm{i}} \mathrm{e}^{\left(\frac{a_{\mathrm{i}} W_{\mathrm{i}}(t)}{c + \sqrt{\chi}}\right)} \Gamma_k^{\mathrm{i}}, \tag{3.15}$$

donde:

$m_{(\mathrm{i},k)}$	métrica del i-ésimo usuario en el k-ésimo *Resource Block*
$a_{\mathrm{i}}, b_{\mathrm{i}}, c$	Son parámetros ajustables
$\chi = \frac{1}{N_{RT}} \sum_{N_{RT}}^{\mathrm{i}=1} a_{\mathrm{i}} W_{\mathrm{i}}(t)$	Es el retardo de paquetes HOL (Head Of Line) del usuario i al tiempo t.
Γ_k^{i}	Representa la eficiencia espectral para el usuario i -ésimo en el sub-canal- k.

3.4. Canales Consciente de Energía

Las soluciones de ahorro de energía se pueden aplicar tanto a eNB y UE. Por lo que la preocupación de los dispositivos del usuario final sobre los consumos de energía se pueden limitar a través de procedimientos de asignación DRX, que es en la actualidad la única estrategia de asignación de poder cumplir con este objetivo [13].

Capítulo 4

Simulación de los *schedulers* en NS-3

Este capítulo se enfoca en simular los schedulers. Se utiliza como base el simulador del servicio de voz sobre LTE (VoLTE) descrito en el "Reporte de práctica Profesional del Simulador para el servicio VoLTE: Guia de uso" [23]. El simulador se basa en NS-3 por lo que utiliza el conjunto de herramientas del módulo LENA (*LTE-EPC Network Simulator*).

4.1. NS-3

Esta herramienta es un simulador de eventos discretos orientado al estudio de redes en ambientes de investigación y educación. NS-3 es un software libre bajo licencia de GNU GPLv2 y se encuentra disponible de manera pública para su uso, investigación y desarrollo. En particular para este trabajo, la versión *all-in-one* de NS-3 incluye el *framework* LENA que permite evaluar el diseño y desempeño de los elementos clave de una red LTE [23].

4.1.1. LENA

LENA es un producto *open source* orientado a la simulación de redes LTE/EPC que permite el diseño, prueba de algoritmos y soluciones para Redes Auto Organizadas (SON). Otras aplicaciones para LENA incluyen el diseño y evaluación del rendimiento de schedulers para DL y UL, algoritmos de manejo de recursos de red (*Radio Resource Management*), soluciones de coordinación de interferencia inter-celda, balance de carga y administración de movilidad (*Mobility Management*), soluciones para redes heterogéneas (HetNets), provisión de QoE End-to-end, soluciones para redes Multi-RAT y sistemas cognitivos LTE. LENA está basado en el popular simulador de redes NS-3 para sistemas de internet [24].

El modelo de simulación de LENA tiene dos componentes principales, como se indica la Fig. 4.1 [25]:

- Modelo LTE: Este modelo incluye el stack de protocolo de radio LTE (RRC, PDCP, RLC,

MAC, PHY). Estas entidades residen completamente dentro de la UE y los nodos eNB.

- Modelo EPC: Este modelo incluye las interfaces de red central, protocolos y entidades. Estas entidades y protocolos residen dentro de los nodos S-GW, PDN-GW y MME, y parcialmente dentro de los nodos eNB [25]. Proporciona medios para la simulación de la conectividad IP de extremo a extremo sobre el modelo de LTE. En particular, es compatible para la interconexión de múltiples UE a la Internet, a través de una red de acceso de radio de múltiples eNBs conectados a un único nodo de S-GW/PDN-GW [2].

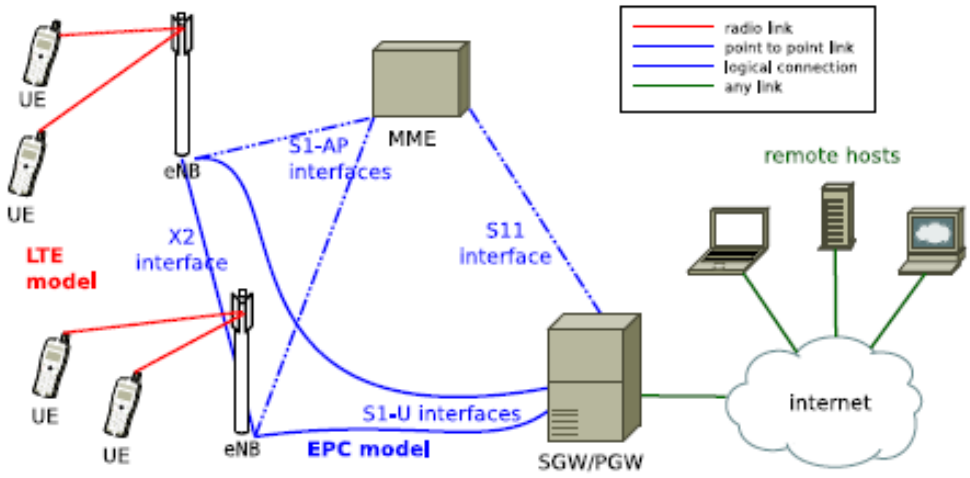

Figura 4.1: Arquitectura General del Modelo de Simulación LTE-EPC.

4.2. Schedulers LTE-MAC en NS-3

4.2.1. Arquitectura de Scheduler LTE-MAC en NS-3

La implementación de los *scheduler* de LTE en NS-3 sigue la especificación de la interfaz de los schedulers de LTE-MAC definida por el Foro de *Small Cell* [21].

La arquitectura de *scheduler* se divide en módulos, tal como lo indica el diagrama de la Fig. 4.2 [21]. El módulo *subframe* desencadena el *scheduler* MAC al comienzo de cada intervalo de tiempo de transmisión (TTI), que es la unidad más pequeña en la asignación de tiempo de longitud igual a 1 ms. El *scheduler* de LTE toma la decisión de planificación basada en la información proporcionada por el módulo de control, así como por el indicador de calidad de canal (CQI) y el estado del búfer del enlace de radio (RLC). El resultado de la decisión de *scheduling* que regresa al módulo *subframe* consiste en la asignación de recursos entre los UE y los bloques de recursos (RB).

El *scheduler* LTE es invocado en cada TTI para asignar los bloques de recursos a los UE siguiendo las prioridades de las métricas [21].

En el eNB, hay un módulo de codificación de canal y modulación adaptativa que asimila el indicador de la calidad del canal (CQI) reportado desde los UE para estimar la calidad del canal

inalámbrico [21].

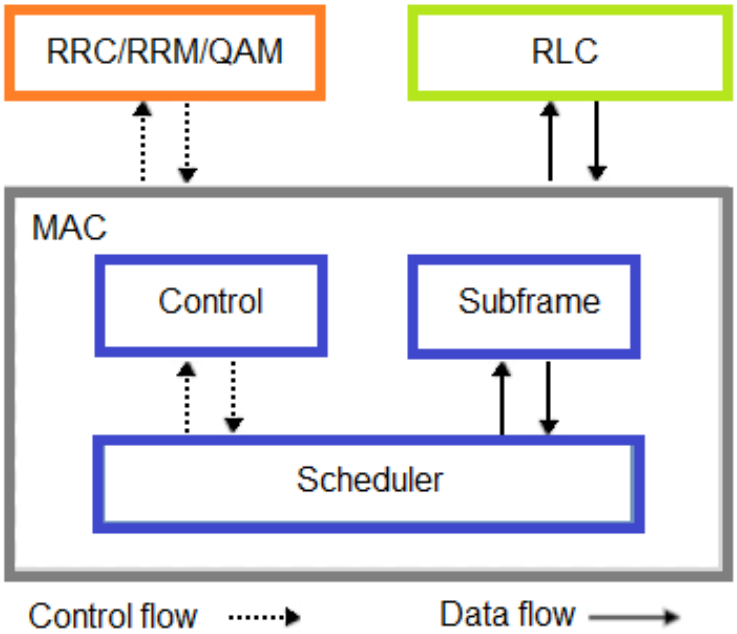

Figura 4.2: Arquitectura de un *Scheduler* LTE-MAC.

4.2.2. *Schedulers* LTE-MAC Simulados en NS-3

Este trabajo se enfoca en simular los siguientes algoritmos de *scheduling* para el *downlink* LTE:

- Round Robin (RR)
- Proportional Fair (PF)
- Maximum Throughput (MT)
- Throughput to Average (TTA)
- Token Bank Fair Queue (TBFQ)
- Priority Set (PSS)
- Channel and QoS Aware (CQA)

La mayoría de estos algoritmos de *scheduling* se enfocan en la asignación de recursos de radio siguiendo el dominio del tiempo (TD) o el enfoque de dominio de frecuencia (FD).

En el enfoque TD, el *scheduling* de LTE asigna todos los recursos de radio en el actual intervalo de tiempo de transmisión (TTI) para un UE. Por el contrario, en el enfoque FD, el *scheduling* LTE asigna los recursos a lo largo de ambas escalas, la frecuencia como del tiempo [21].

Siguiendo esta clasificación, se implementa RR y PF que pertenecen a la categoría TD, sin embargo para algunos *schedulers* existe una versión TD y FD, por ejemplo para los algoritmos MT y TBFQ. Para el algoritmo TTA, que por definición sigue el enfoque FD, se implementa sólo la versión FD. Por último, se presenta los algoritmos PSS y CQA que asignan los recursos teniendo

en cuenta los dominios de tiempo y frecuencia en forma conjunta, por tal razón no encaja en la categorización TD/FD [21].

4.3. Características del Simulador

4.3.1. Escenarios del Simulador

El simulador para el servicio de voz sobre LTE (VoLTE) considera la aproximación basada en las simulaciones sugeridas por Chris Jhonson en sus libro "Long Term Evolution in Bullets" para la planificación de redes LTE. El diagrama se indica en la Fig. 4.3 [23].

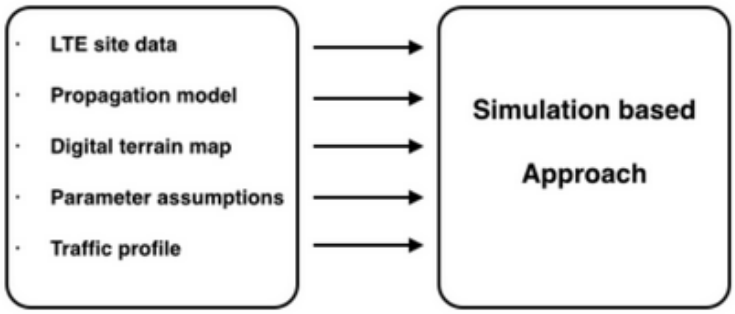

Figura 4.3: Entradas para la Planificación de Redes mediante la Simulación.

4.3.2. Consideraciones del Simulador

LENA y NS-3 en general por ser un software tiene ciertas consideraciones que se toman en cuenta al realizar la simulación. Las más relevantes se detallan a continuación.

- Banda de operación: 700 [MHz], 2600 [MHz].
  - Valores EARFCN Banda 7 (2600 [MHz]):
    - DLEARFCN = 3300 (2675 [MHz])
    - ULEARFCN = 21300 (2555 [MHz])
  - Valores EARFCN Banda 14 (700 [MHz]):
    - DLEARFCN = 5280 (758 [MHz])
    - ULEARFCN = 23280 (788 [MHz])
- Ancho de banda disponible: 10 [MHz], 20 [MHz].
- Número de RB's disponibles: 50, 100
- *Attachment* automático de los UE a los eNBs.
- Nodos trisectoriales.
- Interfaz X2 habilitada entre eNBs.
- EPS Bearer dedicado para tráfico VoIP con QCI = 1 (*Conversational Voice*).

- Condición base de tráfico en la celdas utilizando un UE dedicado con *throughput* representativo de la condición inicial.
- NO se incluye implementación de compresión de *headers* por limitaciones de LENA.
- Posicionamiento de eNB en coordenadas (x,y,z)
- El número de terminales por celda se ubicarán aleatoriamente en torno a su respectivo eNB dentro de un radio de 250 [m].
- La Movilidad es *"Random Walk"*
- La asignación de direcciones IP se realizará de manera automática teniendo las siguientes consideraciones:
  - Red externa punto a punto entre PDN-GW y host remoto (Servidor Web), con dirección base 1.x.x.x. o PDN-GW: 1.0.0.1
  - Servidor Web: 1.0.0.2
  - Red interna que comprende UE's, con dirección IP base 7.x.x.x.
- El Servicio VoIP utiliza la referencia del modelo presentado por Sanjay Ramroop basado en aplicaciones ON-OFF que permiten la generación de ráfagas de duración variable con paquetes de tamaño constante entre pares de UE.
- La condición de Tráfico que se incluyó en cada UE es un representativo del tráfico base en cada celda. Para esto se implementaron nodos dedicados con una aplicación de generación de tráfico según ráfagas que siguen una distribución de Pareto, logrando así simular el tráfico característico de Internet y permitiendo definir la tasa de transferencia máxima para el DL y el UL de manera independiente.

4.4. Implementación en NS-3

En general para la implementación del simulador se utilizaron módulos existentes en el amplio repositorio de NS-3 y que están incluidas en el paquete NS-3 All-in-one.

Para el caso de la implementación del generador de tráfico base se utilizó el módulo PPBP desarrollado por Doreid Ammar (Université Lyon) [23].

4.4.1. Parámetros y Valores de Simulación

Los parámetros básicos de las simulaciones para los diferentes *schedulers* son los mismos y están previamente definidos en los siguientes archivos:

- input-defaults.txt
- node_coordinates.txt
- node_num_ue.txt.

En el primer archivo se definen los parámetros que el simulador toma por defecto en cada ejecución para el eNB y UEs [26].

El archivo node_coordinates.txt se encuentra las coordenadas en x, y, z de las 3 celdas por cada uno de los 3 nodos simulados, medidas en metros con las bandas de operación (EARFCN) cy on su respectivo ancho de banda disponible medido en número de RBs [23].

Finalmente en el archivo node_num_ue.txt se define un conjunto de filas con un solo valor entero correspondiente a la cantidad de UEs presentes inicialmente en el entorno de la celda correspondiente. En este caso el número de fila corresponde al número de UE por celda [23].

En general, los parámetros más relevantes con los que se simulará se resumen en la siguiente tabla:

PARÁMETROS	VALORES
Número de Nodos	3
Celdas por Nodos	3
Número de UEs Base	20
Número de UEs por celda	40
Tiempo de Simulación	5 segundos
Tasa de tráfico base DL	1 Mbps
Tasa de tráfico base UL	500 Kbps
Tipo de Tráfico	VoIP (UDP)

Tabla 4.1: Valores de los Parámetros de Simulación.

Los parámetros mencionados en la Tabla 4.1, fueron elegidos luego de analizar la duración de la simulación, recordando que el tiempo de simulación difiere considerablemente del tiempo real.

El resto de parámetros y valores que se configuraron se pueden encontrar en los archivos indicados en el Anexo C.

4.5. Ejecución de la Simulación

Definido los parámetros y valores de las configuraciones básicas en los archivos previamente indicados, se procede a ejecutar la simulación, considerando que para cada *scheduler* varía el comando, de tal manera:

```
./waf --command-template="%s --ns3::ConfigStore::Filename=input-defaults.txt
--ns3::ConfigStore::Mode=Load --ns3::ConfigStore::FileFormat=RawText --simTime=segundos
--tasaTraficoBaseDl=tasaDl --tasaTraficoBaseUl=tasaUl --numUeBase=numBase"
--run scratch/Nombre del Scheduler
```

4.6. Resultados de la Simulación en NS-3

La simulación presenta sus resultados en dos tipos de archivos. Los archivos de las capas PHY, MAC, RLC y PDCP son TXT, los cuales se generan por defecto y detallan los KPI de las capas ha-

bilitadas. Por otro lado, para la parte IP se habilita un archivo XML denominado "Flowmon_VoIP" que recolecta los datos del monitor de flujo IP agregado al simulador.

- Archivos TXT [26].

  Este trabajo se enfoca en la parte DL de los archivos .txt. A continuación se detalla los KPIs que presentan este tipo de archivos.

  En la capa física, se tiene tres archivos distintos. El detalle de cada uno se presenta a continuación:

  - DlRsrpSinrStats.txt
    - Tiempo de simulación en el cual la asignación fue indicada por el *scheduler* [segundos].
    - ID de celda.
    - ID único de UE (IMSI).
    - RSRP [lineal]
    - Promedio sobre todos los RB's para el SINR en el downlink [lineal].
  - DlTxPhyStats.txt
    - Tiempo de simulación [milisegundos].
    - ID de celda
    - ID único de UE (IMSI)
    - RNTI.
    - Capa de transmisión
    - MCS.
    - Tamaño del TB [bytes].
    - Redundancy version
    - New Data Indicator flag
  - DlRxPhyStats.txt
    - Tiempo de simulación [milisegundos].
    - ID de celda
    - ID único de UE (IMSI)
    - RNTI.
    - Modo de transmisión
    - Capa de transmisión
    - MCS.
    - Tamaño del TB [bytes].
    - Redundancy version
    - New Data Indicator flag
    - Correctitud en la recepción del TB.

  La capa MAC en el DL genera el archivo

  - DlMacStats.txt
    - Tiempo de simulación en el cual la asignación fue indicada por el *scheduler* [segundos].

    - ID de celda.
    - ID único de UE (IMSI).
    - Número de frame.
    - Número de subframe.
    - Cell-specific ID para el UE (RNTI).
    - MCS del TB 1.
    - Tamaño del TB 1 [bytes].
    - MCS del TB 2 (0 si no está presente).
    - Tamaño del TB 2 (0 si no está presente).

La capa RLC presenta dos archivos, pero para nuestro interés solo se detallará para parte de DL

- DlRlcStats.txt
    - Tiempo de simulación en el cual la asignación fue indicada por el *scheduler* [segundos].
    - Tiempo de término del intervalo de medición desde el inicio de la simulación [segundos]
    - ID de celda
    - ID único de UE (IMSI).
    - Cell-specific ID para el UE (RNTI)
    - ID de canal lógico
    - Número de PDUs RLC transmitidos.
    - Total de bytes transmitidos.
    - Número de PDUs RLC recibidos.
    - Total de bytes recibidos.
    - Delay promedio de los PDU RLC [segundos].
    - Desviación estándar del delay de los PDU RLC.
    - Valor mínimo de delay para los PDU RLC [segundos].
    - Valor máximo de delay para los PDU RLC [segundos].
    - Tamaño promedio de PDU RLC [bytes].
    - Desviación estándar del tamaño de PDU RLC.
    - Tamaño mínimo de PDU RLC [bytes].
    - Tamaño máximo de PDU RLC [bytes].

La capa PDCP al igual que la anterior genera dos archivos, pero nuevamente solo se revisará el archivo DL.

- DlPdcpStats.txt
    - Tiempo de inicio del intervalo de medición desde el inicio de la simulación [segundos].
    - Tiempo de término del intervalo de medición desde el inicio de la simulación [segundos].
    - ID de celda.

    - ID único de UE (IMSI).
    - Cell-specific ID para el UE (RNTI).
    - ID de canal lógico.
    - Número de PDUs PDCP transmitidos.
    - Total de bytes transmitidos.
    - Número de PDUs PDCP recibidos.
    - Total de bytes recibidos.
    - Delay promedio de los PDU PDCP [segundos].
    - Desviación estándar del delay de los PDU PDCP.
    - Valor mínimo de delay para los PDU PDCP [segundos].
    - Valor máximo de delay para los PDU PDCP [segundos].
    - Tamaño promedio de PDU PDCP [bytes].
    - Desviación estándar del tamaño de PDU PDCP.
    - Tamaño mínimo de PDU PDCP [bytes].
    - Tamaño máximo de PDU PDCP [bytes].

- Archivo Flowmon_VOIP [27]

  En general el módulo *Flow Monitor* permite medir el rendimiento de los protocolos IP de red. El módulo utiliza sondas instaladas en los nodos de red, para realizar un seguimiento de los paquetes intercambiados por los nodos y de esta manera medir los parámetros.

  En este caso los paquetes IP se dividen en protocolos, origen (IP, puerto), destino (IP, puerto).

  Las estadísticas que se recogen para cada flujo se las archiva en formato XML.

  Los datos recogidos para cada flujo son:

  - timeFirstTxPacket: Tiempo cuando el primer paquete en el flujo fue transmitido.
  - timeLastTxPacket: Tiempo cuando el último paquete en el flujo fue trasmitido.
  - timeFirstRxPacket: Tiempo cuando el primer paquete en el flujo fue recibido por el nodo final.
  - timeLastRxPacket: Tiempo cuando el último paquete en el flujo fue recibido.
  - delaySum: Suma de todos los retardos end-to-end para todos los paquetes recibidos por el flujo.
  - jitterSum: Suma de todos los valores de los retrasos de jitter end-to-end (*delay variation*) para todos los paquetes recibidos. Definidos en el RFC 3393;
  - txBytes, txPackets: Número total de bytes/paquetes transmitidos por el flujo;
  - rxBytes, rxPackets: Número total de bytes/paquetes recibos por el flujo.
  - lostPackets: Número total de paquetes que se asumen como perdidos (no reportados sobre los 10 segundos) timesForwarded: el número de veces que un paquete ha sido reportado como reenviado.

Capítulo 5

Análisis de Datos y Modelos

Este capítulo recoge la información que emite el simulador, analiza todos los parámetros de las capas PHY, MAC, RLC, PDCP e IP y se elige las variables de interés, finalizando con varios modelos multivariables para determinar el MOS estimado de cada uno de los *Schedulers* simulados.

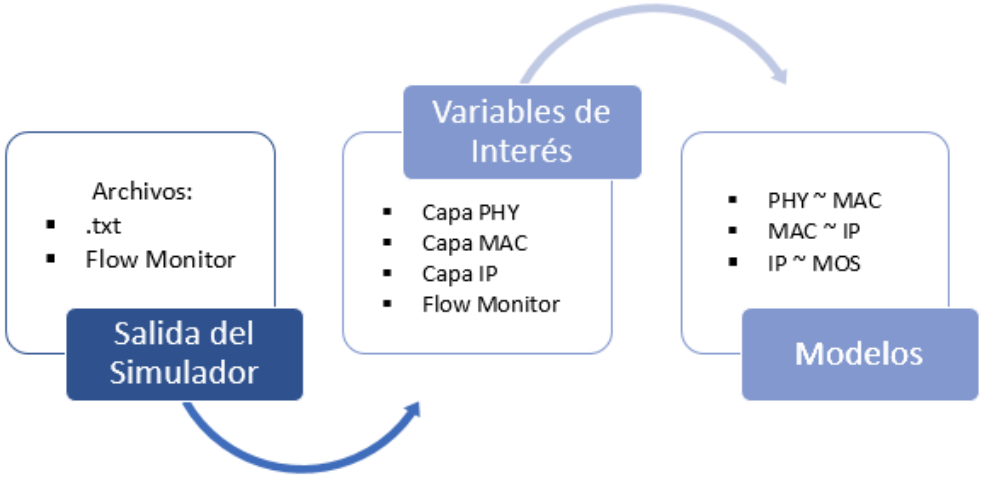

Figura 5.1: Flujo de Análisis de los Datos y Modelación del MOS.

5.1. Descripción de variables

El análisis de los datos se logra con *scripts* en "R" que permiten presentar la información de cada *scheduler* con sus variables de interés. De la información que presenta el simulador solo se analiza los atributos que se detallan a continuación:

CAPA	VARIABLE	DESCRIPCIÓN
PHY	RSRP	Señal de referencia de potencia recibida. Medido en dBm
	SINR	Relación señal sobre la interferencia más el ruido. Medida en dB.
MAC	MCS	Esquema de codificación y modulación.
IP	PDU_Tx	Unidad de datos de protocolo de Tx en la capa IP para DL
	Bytes_Tx	Número de Bytes de Tx en la capa IP para DL
	PDU_Rx	Unidad de datos de protocolo de Rx en la capa IP para DL
	Bytes_Rx	Número de Bytes de Rx en la capa IP para DL
	Delay	Retraso en la capa IP en DL
	PDU_Size	Tamaño de la PDU en la capa IP en DL

Tabla 5.1: Descripción de Variables del Simulador.

5.2. Construcción de datos

La información de las 2 primeras capas se recopila directo de los archivos.txt. Sin embargo, para obtener la información de la capa IP se recurre a una relación que se obtiene a partir del archivo "Flowmon_VoIP.xml".

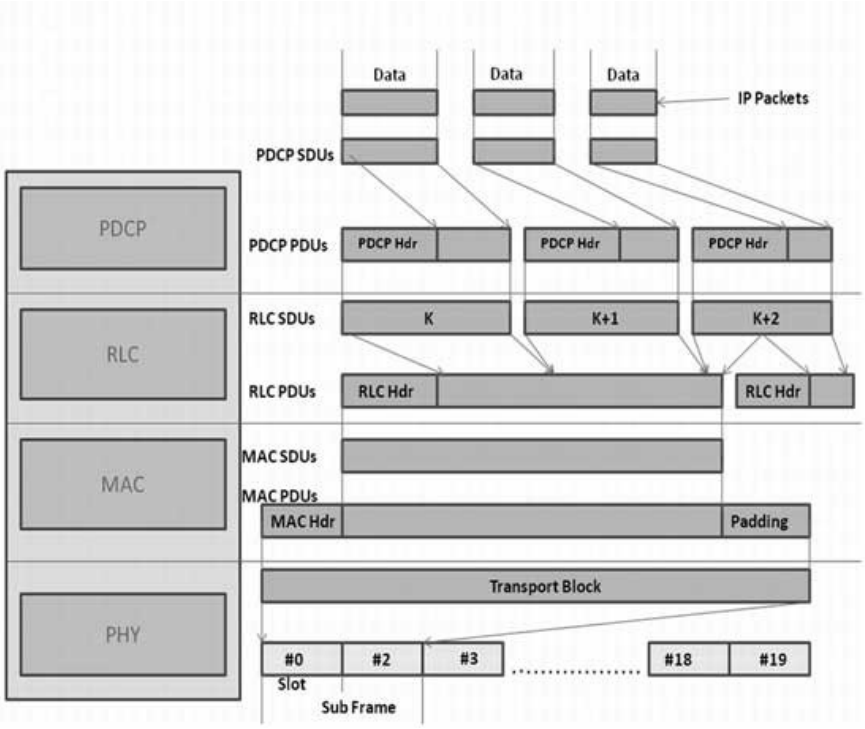

Figura 5.2: Fujo de Datos entre capas LTE.

Los datos que se rescatan del archivo .xml se presentan por flujo no por algún parámetro identificador como celda o IMSI, para solucionar este problema se usa el flujo de datos del plano usuario de LTE, como se indica en la Fig. 5.2 [7].

En este plano, la PDU entre capas PDCP e IP solo añade el header, ya que el modo de retransmisión de RLC es UM (*Unacknowledged mode*), el cual soporta la segmentación y reensamblaje en la secuencia de la entrega, pero no implementa mecanismo de retransmisión [28], adicionando que por características del simulador no se añade señalización.

El factor de relación encontrado se aplica a la PDU de la capa PDCP y de esta manera se obtiene los datos IP. Para cada uno de los *schedulers* se analiza el tamaño de la PDU solo la parte UDP (sin contar el tráfico que pertenece a la parte http). El factor presentado toma la relación entre los paquetes transmitidos, los paquetes recibidos y los paquetes perdidos, tal como se indica en la ecuación 5.1.

$$\frac{Paquetes \quad transmitidos}{paquetes \quad recibidos + paquetes \quad perdidos} \tag{5.1}$$

El factor en cada *scheduler* se indica en la siguiente Tabla:

SCHEDULER	FACTOR
RR	1.042
PF	1.041
FD-MT	1.051
TD-MT	1.054
TTA	1.106
FD-TBFQ	1.387
TD-TBFQ	1.373
PSS	1.043
CQA	1.049

Tabla 5.2: Factor Para Capa IP.

Por simetría del tráfico de VoIP emulado se utiliza el factor del sistema tanto para la parte Tx y Rx del análisis.

Al aplicar el factor presentado se obtiene la información de la capa IP.

La agrupación de los datos será utilizando IMSI en cada una de las capas. IMSI es un ID único que identifica globalmente a un suscriptor móvil. Se compone de dos partes: PLMN y MSIN [6].

5.3. Variables de Interés

Los datos previamente descritos facilitan la construcción de las variables de interés que permite realizar los modelos en cada una de las capas para cada *Scheduler*.

En la capa PHY las variables de interés son: “RSRP” y “SINR” las cuales se toman directamente de los archivos TXT. En la capa MAC el interés está en el valor de “MCS” y al igual que en la capa

anterior sus valores se rescatan de los archivos TXT. El motivo para tomar estos KPI, responde al interés de relacionar los parámetros principales de la parte PHY y MAC.

Finalmente para la parte IP, las variables de interés son: "DELAY", "JITTER" y "PACKET LOSS". Estas variables son la base para el cálculo del valor MOS con la función E- MODEL [29], por tal motivo son buscadas.

Las variables de interés de la parte IP se las obtiene de dos fuentes. La primera se rescata de manera directa del archivo "FLOW MONITOR.XML". La segunda opción es mediante el cálculo de las variables de los archivos TXT de la capa IP, recordando que estos datos provienen de aplicar el factor previamente obtenido en la Tabla 5.2 a la capa PDCP. El valor "DELAY" se lo toma directo de los archivos TXT. El "JITTER" se lo obtiene utilizado MPPDV o *Mean Packet to Packet Delay Variation* que es la base para el cálculo del jitter en aplicaciones RTP [15], y finalmente el valor perteneciente a "PACKET LOSS" es el porcentaje de la diferencia de paquetes transmitidos sobre los recibidos, tal como lo realiza *Flow Monitor*.

5.4. Estadística Descriptiva de los *Schedulers*

En cada uno de los *schedulers* se presenta la estadística de los valores resultantes de la simulación. Las variables delay, jitter y packet loss tienen dos fuentes de cálculo las mismas que también se presentan y dependiendo del *schedulers* varían sus valores.

A continuación se presenta un histograma de la variable de la capa MAC, el resto de los gráficos se puede encontrar en el Anexo A.

5.4.1. RR

FUENTE	VARIABLE	MEDIA	ERROR STD	25 %	75 %	N
Archivos.txt	rsrp[dBm]	1.27e-09	2.73e-09	1.33e-10	1.04e-09	1944000
	sinr[dB]	1.19e+06	4.07e+06	3.93e+01	5.86e+05	1944000
	mcs	19.02	10.82	8	28	49371
PDCP	delay[ms]	32.62	52.43	16.29	22.72	2891
	jitter[ms]	14.19	33.14	0.61	7.59	2891
	packet loss[%]	7.22	19.39	0.00	7.69	2891
Flow Monitor	delay[ms]	36.23	52.08	16.40	24.31	366
	jitter[ms]	7.77	14.90	0.78	4.58	366
	packet loss[%]	10.93	23.64	0.00	2.13	366

Tabla 5.3: Estadísticos Descriptivos RR.

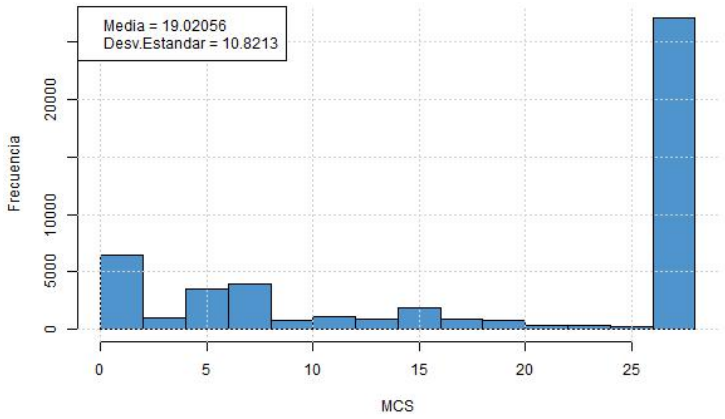

Figura 5.3: MCS - Scheduler RR.

5.4.2. PF

FUENTE	VARIABLE	MEDIA	ERROR STD	25 %	75 %	N
Archivos.txt	rsrp[dBm]	1.27e-09	2.73e-09	1.33e-10	1.04e-09	1944000
	sinr[dB]	8.85e+05	3.63e+06	1.13e+01	2.88e+05	1944000
	mcs	19.20	10.48	8	28	31698
PDCP	delay[ms]	33.62	51.43	16.54	23.26	2904
	jitter[ms]	14.38	32.41	0.75	7.95	2904
	packet loss[%]	6.73	17.26	0.00	7.69	2904
Flow Monitor	delay[ms]	35.24	48.93	16.93	23.22	366
	jitter[ms]	7.97	16.09	1.26	5.58	366
	packet loss[%]	10.07	22.87	0.00	1.01	366

Tabla 5.4: Estadísticos Descriptivos PF.

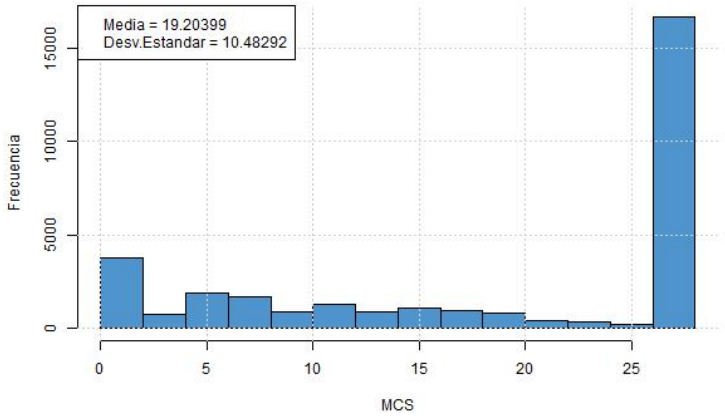

Figura 5.4: MCS - Scheduler PF.

5.4.3. FD-MT

FUENTE	VARIABLE	MEDIA	ERROR STD	25%	75%	N
Archivos.txt	rsrp[dBm]	1.27e-09	2.73e-09	1.33e-10	1.04e-09	1944000
	sinr[dB]	8.73e+05	3.61e+06	1.10e+01	2.77e+05	1944000
	mcs	20.08	10.23	10	28	31851
PDCP	delay[ms]	41.77	86.44	16.66	24.97	2916
	jitter[ms]	23.82	56.11	1.36	17.88	2916
	packet loss[%]	20.21	66.03	0.00	11.45	2916
Flow Monitor	delay[ms]	76.01	136.35	17.30	70.73	366
	jitter[ms]	14.50	24.92	1.54	20.08	366
	packet loss[%]	11.12	23.66	0.00	1.32	366

Tabla 5.5: Estadísticos Descriptivos FD-MT.

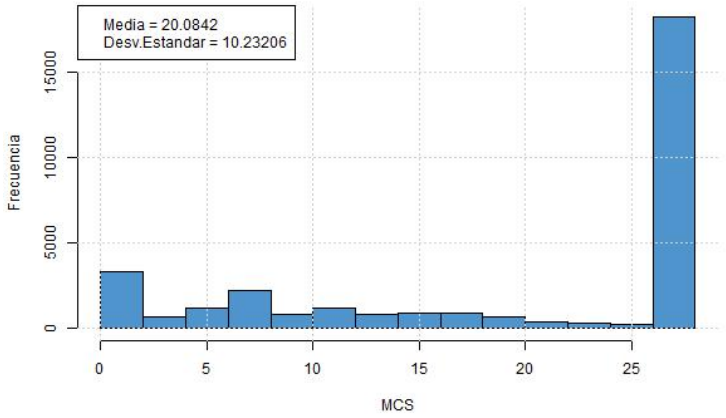

Figura 5.5: MCS - Scheduler FD-MT.

5.4.4. TD-MT

FUENTE	VARIABLE	MEDIA	ERROR STD	25%	75%	N
Archivos.txt	rsrp[dBm]	1.27e-09	2.73e-09	1.33e-10	1.04e-09	1944000
	sinr[dB]	8.81e+05	3.63e+06	1.13e+01	2.85e+05	1944000
	mcs	20.36	10.08	10	28	31772
PDCP	delay[ms]	38.05	74.78	16.73	25.06	2882
	jitter[ms]	20.65	48.77	1.46	13.78	2882
	packet loss[%]	19.92	61.94	0.00	12.50	2882
Flow Monitor	delay[ms]	75.45	139.79	17.48	70.19	366
	jitter[ms]	14.35	27.03	1.76	19.36	366
	packet loss[%]	11.68	24.85	0.00	1.32	366

Tabla 5.6: Estadísticos Descriptivos TD-MT.

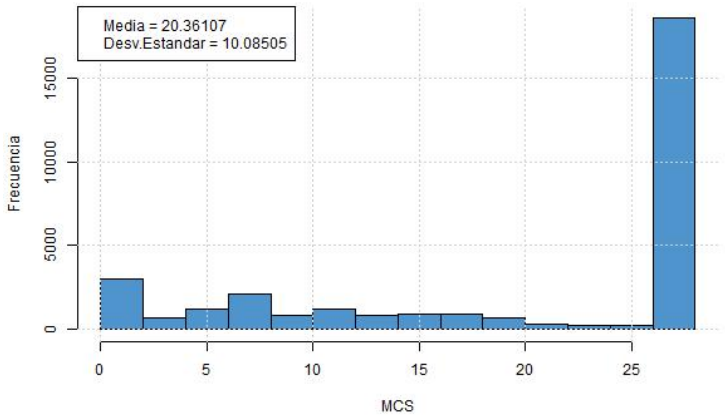

Figura 5.6: MCS - Scheduler TD-MT.

5.4.5. TTA

FUENTE	VARIABLE	MEDIA	ERROR STD	25 %	75 %	N
Archivos.txt	rsrp[dBm]	1.27e-09	2.73e-09	1.33e-10	1.04e-09	1944000
	sinr[dB]	8.93e+05	3.65e+06	1.16e+01	2.94e+05	1944000
	mcs	16.65	11.48	6	28	31585
PDCP	delay[ms]	37.43	145.44	0.00	22.81	2891
	jitter[ms]	25.19	94.18	1.19	11.66	2891
	packet loss[%]	37.59	92.90	0.00	100	2891
Flow Monitor	delay[ms]	104.44	225.23	16.51	38.59	366
	jitter[ms]	18.58	64.65	0.88	9.27	366
	packet loss[%]	22.54	32.57	0.00	49.15	366

Tabla 5.7: Estadísticos Descriptivos TTA.

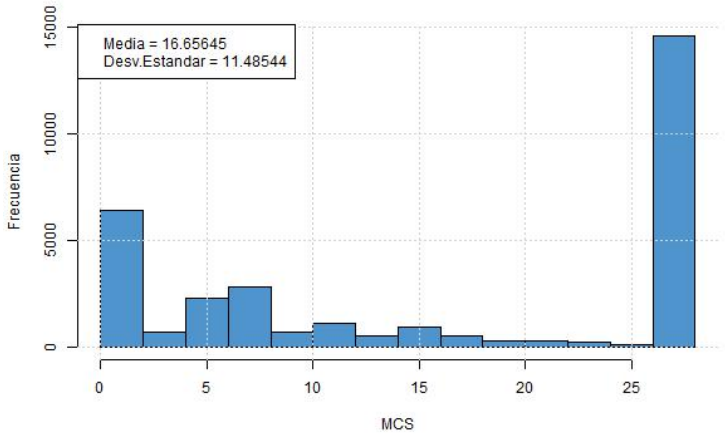

Figura 5.7: MCS - Scheduler TTA.

5.4.6. FD-TBFQ

FUENTE	VARIABLE	MEDIA	ERROR STD	25%	75%	N
Archivos.txt	rsrp[dBm]	1.27e-09	2.73e-09	1.33e-10	1.04e-09	1944000
	sinr[dB]	2.67e+06	5.76e+06	2.81e+05	2.19e+06	1944000
	mcs	27.98	0.61	28	28	41088
PDCP	delay[ms]	27.45	96.35	0.00	0.00	2846
	jitter[ms]	19.36	59.98	0.00	2.42	2846
	packet loss[%]	86.51	31.95	100	100	2846
Flow Monitor	delay[ms]	43.65	72.53	0.00	70.31	366
	jitter[ms]	20.93	33.91	0.00	29.75	366
	packet loss[%]	54.96	27.53	33.40	76.70	366

Tabla 5.8: Estadísticos Descriptivos FD-TBFQ.

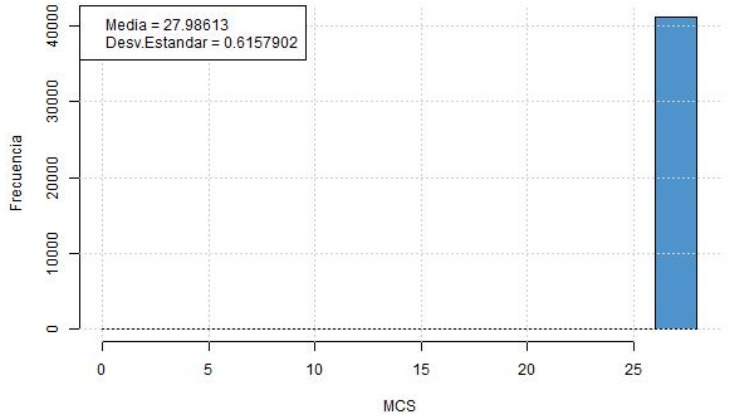

Figura 5.8: MCS - Scheduler FD-TBFQ.

5.4.7. TD-TBFQ

FUENTE	VARIABLE	MEDIA	ERROR STD	25%	75%	N
Archivos.txt	rsrp[dBm]	1.27e-09	2.73e-09	1.33e-10	1.04e-09	1944000
	sinr[dB]	2.67e+06	5.77e+06	2.80e+05	2.17e+06	1944000
	mcs	27.93	1.26	28	28	42678
PDCP	delay[ms]	2.91	46.69	0.00	0.00	2845
	jitter[ms]	2.86	32.94	0.00	0.00	2845
	packet loss[%]	95.77	19.69	100.00	100.00	2845
Flow Monitor	delay[ms]	5.18	63.34	0.00	0.00	366
	jitter[ms]	0.66	6.95	0.00	0.00	366
	packet loss[%]	68.61	25.52	54.54	88.49	366

Tabla 5.9: Estadísticos Descriptivos TD-TBFQ.

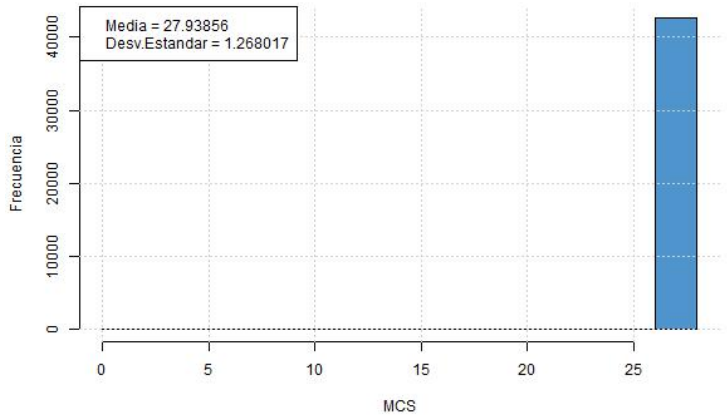

Figura 5.9: MCS - Scheduler TD-TBFQ.

5.4.8. PSS

FUENTE	VARIABLE	MEDIA	ERROR STD	25 %	75 %	N
Archivos.txt	rsrp[dBm]	1.27e-09	2.73e-09	1.33e-10	1.04e-09	1944000
	sinr[dB]	8.91e+05	3.65e+06	1.16e+01	2.92e+05	1944000
	mcs	19.56	10.35	10	28	31676
PDCP	delay[ms]	36.46	74.12	16.70	23.45	2846
	jitter[ms]	16.83	47.74	0.80	7.96	2846
	packet loss[%]	7.66	18.32	0.00	8.33	2846
Flow Monitor	delay[ms]	45.01	96.15	17.01	25.54	366
	jitter[ms]	9.87	18.17	1.35	6.66	366
	packet loss[%]	10.70	24.06	0.00	1.11	366

Tabla 5.10: Estadísticos Descriptivos PSS.

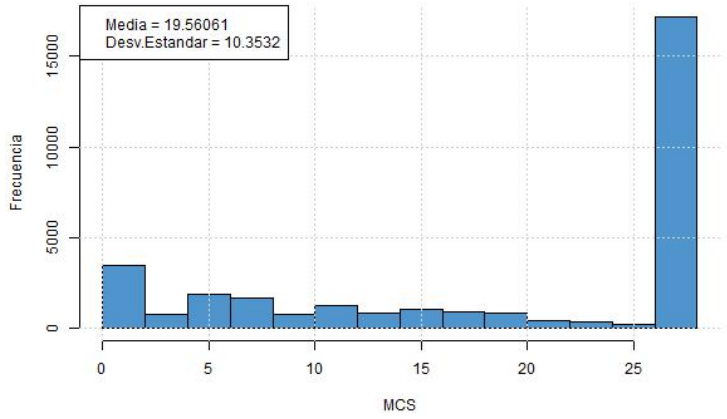

Figura 5.10: MCS - Scheduler PSS.

5.4.9. CQA

FUENTE	VARIABLE	MEDIA	ERROR STD	25 %	75 %	N
Archivos.txt	rsrp[dBm]	1.27e-09	2.73e-09	1.33e-10	1.04e-09	1944000
	sinr[dB]	1.31e+06	4.18e+06	1.02e+02	7.61e+05	1944000
	mcs	21.18	10.26	12	28	41471
PDCP	delay[ms]	38.66	77.45	16.54	28.31	2832
	jitter[ms]	19.44	49.48	1.31	11.86	2832
	packet loss[%]	12.44	19.39	0.00	9.09	2832
Flow Monitor	delay[ms]	52.22	86.05	17.36	41.10	366
	jitter[ms]	20.94	54.77	1.32	21.60	366
	packet loss[%]	11.77	24.01	0.00	3.22	366

Tabla 5.11: Estadísticos Descriptivos CQA.

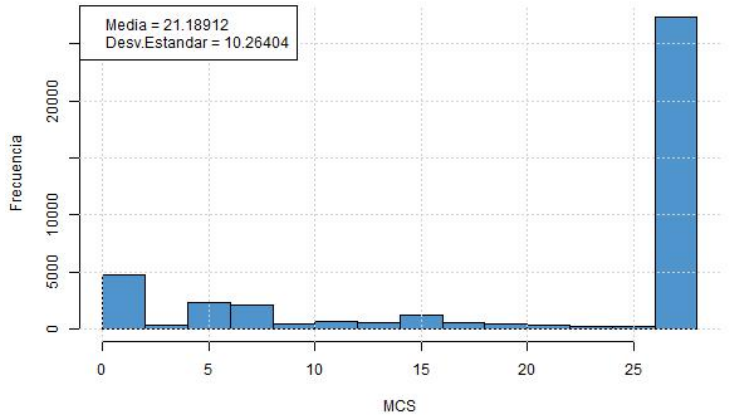

Figura 5.11: MCS - Scheduler CQA.

Los *scripts* que toman la información y presentan el análisis estadístico de cada *scheduler* se puede encontrar en el Anexo B.

5.5. Modelos

El objetivo de esta sección es buscar modelos en base a sus variables de interés previamente definidas en cada capa. Las variables del mejor modelo ingresarán al *script* E-MOS para determinar el MOS del *scheduler*.

El esquema para realizar los modelos es el indicado en la Fig. 5.12, recordando que todas las variables de interés se obtienen o construye en base a las salidas de la simulación de cada *scheduler*.

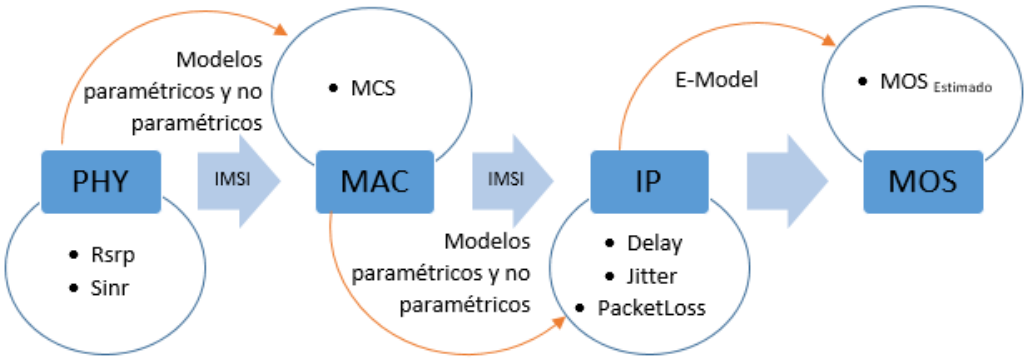

Figura 5.12: Esquema de Construcción de los Modelos de Estimación del MOS.

En los 9 *schedulers* simulados se trabaja de manera parecida. Se buscan modelos paramétricos y no paramétricos para la capa PHY-MAC y la capa MAC-IP, los resultados se ingresan a la función E-MODEL.

Los modelos paramétricos parten de una función de distribución conocida, en este caso el trabajo se reduce en estimar los parámetros que mejor ajusten las observaciones de la muestra. Sin embargo, hay situaciones en que, por el escaso número de observaciones o por el nivel de medición de las variables, no es correcto o no es posible hacer supuestos sobre las distribuciones muestrales, en estos casos se usan los métodos "no paramétricos" o de distribución libre pues no se encuentran sujetos a ninguna forma funcional [30].

En ambos casos es necesario estimar los parámetros de los que depende la forma funcional elegida. En el caso de los modelos paramétricos la elección de dicha forma funcional se establece *a priori*, por lo que una elección inadecuada producirá un modelo que no ajuste a los datos. Por su parte los modelos no paramétricos emplean formas funcionales flexibles que aproximen la función objetivo [30].

Dentro de los métodos paramétricos encontramos a los Modelos Lineales Generalizados o GLM [31], los cuales son una extensión de los Modelos Lineales que permiten utilizar distribuciones no normales de los errores (binomiales, poisson, gamma, etc) y varianzas no constantes [32]. En esta tesis se utilizan las siguientes distribuciones:

- Gausiana
- Poisson

Para el análisis de estos modelos se construye una tabla que presenta los coeficientes, el error estándar, el p-value y el valor de AIC.

Los métodos no paramétricos [33], [34] disponibles en *R* pueden ser funciones o paquetes [35], [36], de estos se elige los siguientes para ejecutarlos:

- Regresión Polinomial Local Ponderada (LOESS) [37].
- Regresión Polinomial Local (LOCFIT) [38].
- Regresión de Projection Pursuit (PPR) [39].

- Modelos Adaptivos Generalizados (GAM) [40].
- Regresión Polinomial Local con ancho de banda constante calculado según el criterio de validación cruzada (SM) [41], [42].
- Regresión Multivariada Adaptativa utilizando Splines (MARS) [43].
- Regresión Multivariada Adaptativa Polinomial utilizando Splines (POLYMARS) [44].
- Regresión de Cox o modelo de los riesgos proporcionales (COXPH) [45].

Los métodos de cálculo utilizados en "LOCFIT" desarrollan las ideas utilizadas en "LOESS" (Cleveland y Grosse, 1991). La regresión local es directamente calculada con un pequeño número de puntos y el ajuste en estos puntos se interpolan suavemente para obtener el ajuste en los puntos restantes. "LOCFIT" difiere de "LOESS" en la forma de selección de puntos para su ajuste directo. Mientras LOESS basa su elección en la densidad de puntos de datos, LOCFIT utiliza los anchos de banda en los puntos de ajustados, esto permite a LOCFIT adaptarse a diferentes esquemas de ancho de banda: fijo, vecino más cercano y de adaptación a nivel local [46].

La regresión por *Projection Pursuit* (PPR) es una extensión de los modelos aditivos y fue desarrollado por Jerome H. Friedman y Werner Stuetzle. Este modelo se adapta a los modelos aditivos porque el primero proyecta la matriz de datos de variables explicativas en la dirección óptima antes de aplicar funciones de suavizado de estas variables explicativas. A diferencia de los métodos de promedios locales (como k-vecinos más cercanos), PPR puede ignorar variables con bajo poder explicativo [47].

Por su parte los modelos "GAM" son una extensión de los Modelos Lineales Generalizados (GLM) que combinan la precisión estadística típica de una variable explicativa unidimensional con la flexibilidad de los modelos semi-paramétricos de variables explicativas multidimensionales, por lo que en estos modelos no está presente la maldición de la dimensionalidad [48].

La función "sm.regression" crea una estimación de la regresión no paramétrica de los datos, que constan de una sola variable respuesta y una o dos covariables. En dos dimensiones se produce una perspectiva, una imagen (*image*), un contorno (*slice*) o una parcela rgl de la superficie de regresión estimada [36].

"MARS" es un método de regresión no paramétrica que no hace ninguna suposición sobre la relación entre las variables respuesta e independientes así como los supuestos que requieren Regresión Lineal Múltiple. "MARS" construye esta relación basándose en una serie de coeficientes asociados a las funciones base que son totalmente determinadas a partir de la regresión de los datos. La idea del algoritmo de "MARS" opera como múltiples partes de regresiones lineales [49].

Por su lado "POLYMARS", es un procedimiento de regresión adaptativa con muchos pedazos de splines lineales para modelar la respuesta. Su algoritmo empleado tiene muchas similitudes con el algoritmo MARS, sin embargo muestra algunas mejoras [44].

El análisis de supervivencia se centra en la distribución de los tiempos de supervivencia. Aunque existen métodos bien conocidos para estimar las distribuciones incondicionales de supervivencia, el modelo de supervivencia más interesante examina la relación entre la supervivencia y uno o más predictores, usualmente denominados covariables en la literatura de análisis de supervivencia. El utilizado en este trabajo es el modelo de "regresión Cox" de riesgos proporcionales, un método

ampliamente aplicable y el método de análisis de supervivencia más utilizado [45].

En los modelos no paramétricos se imprime una tabla con el valor RSS.

Aunque el AIC es un índice muy útil para comparar modelos, debido a la naturaleza de los métodos, el cálculo del AIC no se realiza de la misma forma, por lo tanto la comparación será gracias al test de Kolmogorov-Smirnov que se presenta en el siguiente capítulo.

5.5.1. RR

Modelos PHY - MAC

En esta etapa el modelo buscado es en base a las variables de interés: MCS, SINR y RSRP. La idea es saber cuánto afectan las variables de interés de la capa PHY a la capa MAC, por lo tanto el modelo base es MCS $\sim$ RSRP+SINR.

Los valores de las distribuciones de los modelos paramétricos son los siguientes:

MODELO	COEFICIENTES	ERROR STD	P-VALUE	AIC
Gausiano	2.123e+01	3.281e-01	<2e-16	2296.3
	2.397e+08	2.927e+08	0.413	
	3.580e-07	2.867e-07	0.213	
Poisson	3.058e+00	1.243e-02	<2e-16	2425.2
	1.107e+07	1.008e+07	0.272	
	1.223e-08	9.688e-09	0.207	

Tabla 5.12: Modelos Paramétricos PHY-MAC - Scheduler RR.

A continuación se presenta los valores RSS obtenidos de los métodos no paramétricos.

MODELO	RSS
loess	697.47
ppr	22.72
locfit	24.42
gam	31.46
sm	29.02
polymars	23.69
mars	26.96
coxph	0.8

Tabla 5.13: Modelos No Paramétricos PHY-MAC - Scheduler RR.

La Tabla de Residuos de todos los modelos aplicados para la parte PHY-MAC es la siguiente:

MODELO	RESIDUO	MEDIA	VARIANZA
gaussian	0.00	0.00	31.54
poisson	-24.82	-0.06	1.77
loess	3725.96	10.23	594.32
ppr	0.00	0.00	22.78
locfit	1.64	0.004	24.49
gam	0.00	0.00	31.54
sm	45.86	0.13	29.09
polymars	0.00	0.00	23.76
mars	0.00	0.00	27.03
coxph	0.00	0.00	0.79

Tabla 5.14: Residuos PHY-MAC - Scheduler RR.

Modelos MAC - IP

En este caso se persigue modelos separados de DELAY, JITTER y PACKET LOSS en función de la capa MAC y de la variable que se obtiene al aplicar el mejor modelo obtenido en la sección PHY-MAC, por lo tanto se buscan: DELAY $\sim$ MCS + MEJOR MODELO PHY-MAC, JITTER $\sim$ MCS + MEJOR MODELO PHY-MAC y finalmente PACKET LOSS $\sim$ MCS + MEJOR MODELO PHY-MAC.

VARIABLE	MODELO	COEFICIENTES	ERROR STD	P-VALUE	AIC
DELAY	Gausiano	45.537	3.919	<2e-16	31092
		-0.586	0.172	0.000	
		4.563	4.113	0.267	
JITTER	Gausiano	20.611	2.479	<2e-16	28445
		-0.291	0.109	0.007	
		1.989	2.602	0.444	
PACKET LOSS	Gausiano	12.315	1.449	<2e-16	25341
		-0.231	0.063	0.000	
		0.723	1.521	0.634	

Tabla 5.15: Modelos Paramétricos MAC-IP - Scheduler RR.

MODELO	RSS		
	DELAY	JITTER	PACKET LOSS
loess	2704.98	1087.82	372.81
ppr	2631.68	1073.21	370.02
locfit	2672.29	1083.42	371.84
gam	2735.9	1094.9	374.15
sm	2721.65	1092.5	-36525.47
polymars	2603.55	1077.88	371.14
mars	2730	1094.03	374.01
coxph	1.05	1.02	8443957.56

Tabla 5.16: Modelos No Paramétricos MAC-IP - Scheduler RR.

MODELO	DELAY			JITTER			PACKET LOSS		
	RESIDUO	MEDIA	VARIANZA	RESIDUO	MEDIA	VARIANZA	RESIDUO	MEDIA	VARIANZA
gaussian	0.00	0.00	2736.85	0.00	0.00	1095.28	0.00	0.00	374.28
loess	-1360.78	-0.47	2705.69	913.73	0.32	1088.09	-1128.18	-0.39	372.79
ppr	0.00	0.00	2632.59	0.00	0.00	1073.58	0.00	0.00	370.14
locfit	-527.85	-0.18	2673.19	90.62	0.03	1083.79	10.5	0.00	371.97
gam	0.00	0.00	2736.85	0.00	0.00	1095.28	0.00	0.00	374.28
sm	-1272.41	-0.44	2722.39	-769.91	-0.27	1092.81	42591.06	14.73	48.56
polymars	0.00	0.00	2604.45	0.00	0.00	1078.25	0.00	0.00	371.26
mars	0.00	0.00	2730.94	0.00	0.00	1094.41	0.00	0.00	374.14
coxph	0.00	0.00	1.05	0.00	0.00	1.02	0.00	0.00	0.92

Tabla 5.17: Residuos MAC-IP - Scheduler RR.

5.5.2. PF

El esquema seguido para este *scheduler* es el mismo que se realizó para el *scheduler* anterior y el que se repetirá para todos. Se calcula el mejor modelo PHY-MAC y este sirve como variable de ingreso para los modelos MAC-IP.

Modelos PHY - MAC

A continuación se describen los valores obtenidos aplicando métodos paramétricos y no paramétricos finalizando con una tabla de residuos.

MODELO	COEFICIENTES	ERROR STD	P-VALUE	AIC
Gausiano	1.911e+01	3.890e-01	<2e-16	2418.6
	6.133e+08	3.014e+08	0.0426	
	1.537e-07	3.737e-07	0.6811	
Poisson	2.956e+00	1.305e-02	<2e-16	2722.2
	2.704e+07	8.723e+06	0.00194	
	3.939e-09	1.065e-08	0.71136	

Tabla 5.18: Modelos Paramétricos PHY-MAC - Scheduler PF.

A continuación se describe los Modelos No Paramétricos para la parte PHY-MAC.

MODELO	RSS
loess	414.37
ppr	23.33
locfit	37.04
gam	44.02
sm	41.47
polymars	38.14
mars	39.34
coxph	0.8

Tabla 5.19: Modelos No Paramétricos PHY-MAC - Scheduler PF.

La Tabla de Residuos queda así:

MODELO	RESIDUO	MEDIA	VARIANZA
gaussian	0.00	0.00	44.14
poisson	-40.2	-0.11	2.72
loess	3024.81	8.31	346.27
ppr	0.00	0.00	23.39
locfit	-1.8	0.00	37.14
gam	0.00	0.00	44.14
sm	47.99	0.13	41.57
polymars	0.00	0.00	38.25
mars	0.00	0.00	39.45
coxph	0.00	0.00	0.8

Tabla 5.20: Residuos PHY-MAC - Scheduler PF.

Modelos MAC - IP

Los valores del método utilizados para las tres variables: DELAY, JITTER, PACKET LOSS son los siguientes:

VARIABLE	MODELO	COEFICIENTES	ERROR STD	P-VALUE	AIC
DELAY	Gausiano	44.5550	2.9823	<2e-16	31119
		-0.5418	0.1401	0.000113	
		1.9281	4.0660	0.635398	
JITTER	Gausiano	18.88891	1.88157	<2e-16	28443
		-0.22328	0.08841	0.0116	
		3.32479	2.56528	0.1951	
PACKET LOSS	Gausiano	13.54528	0.99442	<2e-16	24740
		-0.33754	0.04672	6.42e-13	
		1.01759	1.35576	0.453	

Tabla 5.21: Modelos Paramétricos MAC-IP - Scheduler PF.

MODELO	RSS		
	DELAY	JITTER	PACKET LOSS
loess	2588.99	1039.33	291.1
ppr	2549.79	1019.83	284.37
locfit	2580.43	1036.68	289.68
gam	2631.03	1047.29	292.53
sm	2620.65	1044.69	-19943.92
polymars	2566.32	1031.09	288.99
mars	2631.23	1047.89	292.58
coxph	1.04	1.01	8511775.04

Tabla 5.22: Modelos No Paramétricos MAC-IP - Scheduler PF.

MODELO	DELAY			JITTER			PACKET LOSS		
	RESIDUO	MEDIA	VARIANZA	RESIDUO	MEDIA	VARIANZA	RESIDUO	MEDIA	VARIANZA
gaussian	0.00	0.00	2631.94	0.00	0.00	1047.65	0.00	0.00	292.63
loess	-1600.35	-0.55	2589.58	594.05	0.2	1039.65	782.63	0.27	291.12
ppr	0.00	0.00	2550.67	0.00	0.00	1020.18	0.00	0.00	284.48
locfit	611.83	0.21	2581.28	780.72	0.27	1036.96	701.33	0.24	289.72
gam	0.00	0.00	2631.94	0.00	0.00	1047.6	0.00	0.00	292.63
sm	-1137.62	-0.39	2621.4	-885.29	-0.3	1044.95	39008.91	13.43	83.21
polymars	0.00	0.00	2567.2	0.00	0.00	1031.45	0.00	0.00	289.09
mars	0.00	0.00	2632.14	0.00	0.00	1048.25	0.00	0.00	292.68
coxph	0.00	0.00	1.04	0.00	0.00	1.01	0.00	0.00	0.9

Tabla 5.23: Residuos MAC-IP - Scheduler PF.

5.5.3. FD-MT

Modelos PHY - MAC

A continuación se detalla los modelos paramétricos para PHY-MAC en el *scheduler* FD-MT.

MODELO	COEFICIENTES	ERROR STD	P-VALUE	AIC
Gausiano	2.028e+01	3.620e-01	<2e-16	2343.8
	4.400e+08	2.743e+08	0.11	
	1.753e-07	3.429e-07	0.61	
Poisson	3.013e+00	1.279e-02	<2e-16	2504.9
	1.927e+07	8.683e+06	0.0264	
	5.501e-09	1.069e-08	0.6067	

Tabla 5.24: Modelos Paramétricos PHY-MAC - Scheduler FD-MT.

El resumen de los modelos No Paramétricos para este *scheduler* es el siguiente:

MODELO	RSS
loess	781.51
ppr	31.45
locfit	34.46
gam	37.81
sm	36.63
polymars	34.52
mars	35.94
coxph	0.74

Tabla 5.25: Modelos No Paramétricos PHY-MAC - Scheduler FD-MT.

A continuación la Tabla de Residuos de FD-MT para los modelos aplicados en PHY-MAC.

MODELO	RESIDUO	MEDIA	VARIANZA
gaussian	0.00	0.00	37.92
poisson	-29.12	-0.08	2.1
loess	4175.75	11.57	649.51
ppr	0.00	0.00	31.54
locfit	8.76	0.02	34.56
gam	0.00	0.00	37.92
sm	29.2	0.08	36.73
polymars	0.00	0.00	34.62
mars	0.00	0.00	36.04
coxph	0.00	0.00	0.74

Tabla 5.26: Residuos PHY-MAC - Scheduler FD-MT.

Modelos MAC - IP

En este caso se tiene 3 variables de interés por lo tanto se realiza 3 modelos diferentes.

VARIABLE	MODELO	COEFICIENTES	ERROR STD	P-VALUE	AIC
DELAY	Gausiano	49.08781	5.48759	<2e-16	34135
		-0.33841	0.24971	0.175	
		-0.03738	9.50503	0.997	
JITTER	Gausiano	31.7980	3.5613	<2e-16	31625
		-0.3759	0.1621	0.0204	
		2.5534	6.1684	0.6789	
PACKET LOSS	Gausiano	28.4151	4.1767	1.24e-11	32550
		-0.4088	0.1901	0.0316	
		6.6523	7.2345	0.3579	

Tabla 5.27: Modelos Paramétricos MAC-IP - Scheduler FD-MT.

MODELO	RSS		
	DELAY	JITTER	PACKET LOSS
loess	7387.89	3119.32	4317.85
ppr	7244	3106.99	4276.07
locfit	7358.67	3118.98	4316.15
gam	7493.84	3156.08	4341.26
sm	7462.08	3141.2	-3648164.89
polymars	7318.06	3125.04	4227.76
mars	7422.14	3130.67	4316.79
coxph	1.00	1.00	8432907.53

Tabla 5.28: Modelos No Paramétricos MAC-IP - Scheduler FD-MT.

MODELO	DELAY			JITTER			PACKET LOSS		
	RESIDUO	MEDIA	VARIANZA	RESIDUO	MEDIA	VARIANZA	RESIDUO	MEDIA	VARIANZA
gaussian	0.00	0.00	7496.42	0.00	0.00	3157.17	0.00	0.00	4342.76
loess	-1565.55	-0.54	7390.15	-754.89	-0.26	3120.33	-1993.71	-0.69	4318.87
ppr	0.00	0.00	7246.5	0.00	0.00	3108.06	0.00	0.00	4277.54
locfit	1008.97	0.35	7361.09	948.03	0.33	3119.95	-780.67	-0.27	4317.57
gam	0.00	0.00	7496.42	0.00	0.00	3157.17	0.00	0.00	4342.76
sm	-2102.88	-0.72	7464.12	-1771.31	-0.61	3141.91	2266.91	0.78	94.5
polymars	0.00	0.00	7320.58	0.00	0.00	3126.11	0.00	0.00	4229.22
mars	0.00	0.00	7424.7	0.00	0.00	3131.75	0.00	0.00	4318.28
coxph	0.00	0.00	1.00	0.00	0.00	1.00	0.00	0.00	0.93

Tabla 5.29: Residuos MAC-IP - Scheduler FDMT.

5.5.4. TD-MT

Se sigue el mismo esquema de los *Schedulers* anteriores.

Modelos PHY - MAC

MODELO	COEFICIENTES	ERROR STD	P-VALUE	AIC
gaussiano	2.038e+01	3.628e-01	<2e-16	2345.3
	4.128e+08	2.786e+08	0.139	
	2.186e-07	3.459e-07	0.528	
Poisson	3.018e+00	1.276e-02	1.276e-02	2498
	1.817e+07	8.803e+06	0.0391	
	7.070e-09	1.074e-08	0.5105	

Tabla 5.30: Modelos Paramétricos PHY-MAC - Scheduler TD-MT.

A continuación se describe los Modelos No Paramétricos

MODELO	RSS
loess	709.8
ppr	19.93
locfit	35.48
gam	37.97
sm	36.68
polymars	35.56
mars	36.23
coxph	0.74

Tabla 5.31: Modelos No Paramétricos PHY-MAC - Scheduler TD-MT.

La Tabla de Residuos de la parte PHY-MAC.

MODELO	RESIDUO	MEDIA	VARIANZA
gaussian	0.00	0.00	38.07
poisson	-28.48	-0.08	2.07
loess	3958.08	10.96	591.22
ppr	0.00	0.00	19.98
locfit	-3.87	-0.01	35.58
gam	0.00	0.00	38.07
sm	29.85	0.08	36.78
polymars	0.00	0.00	35.66
mars	0.00	0.00	36.33
coxph	0.00	0.00	0.74

Tabla 5.32: Residuos PHY-MAC - Scheduler TD-MT.

Modelos MAC - IP

Los Modelos paramétricos y no paramétricos de las tres variable: delay, jitter y pl.

VARIABLE	MODELO	COEFICIENTES	ERROR STD	P-VALUE	AIC
DELAY	Gausiano	47.9457	4.8490	<2e-16	32901
		-0.4583	0.2193	0.0367	
		-4.9084	8.5386	0.5654	
JITTER	Gausiano	30.8711	3.1601	<2e-16	30444
		-0.4799	0.1429	0.000794	
		-2.0079	5.5645	0.718245	
PACKET LOSS	Gausiano	25.8178	4.0028	1.31e-10	31800
		-0.2964	0.1810	0.102	
		-9.7206	7.0484	0.168	

Tabla 5.33: Modelos Paramétricos MAC-IP - Scheduler TD-MT.

MODELO	RSS		
	DELAY	JITTER	PACKET LOSS
loess	5585.77	2369.47	3783.57
ppr	5542.37	2349.37	3732.35
locfit	5575.19	2367.38	3796.51
gam	5602.24	2379.29	3817.49
sm	5488.38	2371.48	-3371507.14
polymars	5574.91	2367.8	3738.95
mars	5573.43	2375.86	3791.41
coxph	1.00	0.99	8238704.76

Tabla 5.34: Modelos No Paramétricos MAC-IP - Scheduler TD-MT.

MODELO	DELAY			JITTER			PACKET LOSS		
	RESIDUO	MEDIA	VARIANZA	RESIDUO	MEDIA	VARIANZA	RESIDUO	MEDIA	VARIANZA
gaussian	0.00	0.00	5604.2	0.00	0.00	2380.12	0.00	0.00	3818.82
loess	3176.87	1.11	5586.49	2665.97	0.93	2369.43	-3113.29	-1.09	3783.71
ppr	0.00	0.00	5544.31	0.00	0.00	2350.19	0.00	0.00	3733.65
locfit	-222.04	-0.08	5577.13	-35.28	-0.01	2368.21	-954.75	-0.33	3797.72
gam	0.00	0.00	5604.2	0.00	0.00	2380.12	0.00	0.00	3818.82
sm	-19.59	-0.01	5490.29	-1457.91	-0.51	2372.05	3621.56	1.26	68.08
polymars	0.00	0.00	5576.86	0.00	0.00	2368.62	0.00	0.00	3740.26
mars	0.00	0.00	5575.37	0.00	0.00	2376.68	0.00	0.00	3792.73
coxph	0.00	0.00	1.00	0.00	0.00	0.99	0.00	0.00	0.93

Tabla 5.35: Residuos MAC-IP - Scheduler TD-MT.

5.5.5. TTA

Modelos PHY - MAC

Modelos buscados para la parte PHY-MAC del *scheduler* TTA.

MODELO	COEFICIENTES	ERROR STD	P-VALUE	AIC
Gausiano	1.853e+01	4.162e-01	<2e-16	2124
	1.020e+09	4.078e+08	0.0128	
	-1.794e-07	4.560e-07	0.6943	
Poisson	2.927e+00	1.419e-02	<2e-16	2371.3
	4.558e+07	1.217e+07	0.00018	
	-1.171e-08	1.346e-08	0.384412	

Tabla 5.36: Modelos Paramétricos PHY-MAC - Scheduler TTA.

MODELO	RSS
loess	644.01
ppr	24.92
locfit	35.66
gam	43.58
sm	40.34
polymars	35.01
mars	37.67
coxph	0.83

Tabla 5.37: Modelos No Paramétricos PHY-MAC - Scheduler TTA.

La Tabla de Residuos de los métodos paramétricos y no paramétricos aplicados es la siguiente:

MODELO	RESIDUO	MEDIA	VARIANZA
gaussian	0.00	0.00	43.72
poisson	-34.41	-0.11	2.66
loess	3410.47	10.66	532.09
ppr	0.00	0.00	25
locfit	7.52	0.02	35.77
gam	0.00	0.00	43.72
sm	49.26	0.15	40.44
polymars	0.00	0.00	35.12
mars	0.00	0.00	37.79
coxph	0.00	0.00	0.83

Tabla 5.38: Residuos PHY-MAC - Scheduler TTA.

Modelos MAC - IP

VARIABLE	MODELO	COEFICIENTES	ERROR STD	P-VALUE	AIC
DELAY	Gausiano	39.2333	9.0092	1.38e-05	33177
		0.1460	0.4313	0.735	
		11.2329	11.5280	0.330	
JITTER	Gausiano	22.8210	5.8206	9.06e-05	30931
		0.2024	0.2787	0.468	
		3.9166	7.4479	0.599	
PACKET LOSS	Gausiano	36.6138	5.6117	8.19e-11	30743
		-0.3462	0.2687	0.198	
		6.7731	7.1807	0.346	

Tabla 5.39: Modelos Paramétricos MAC-IP - Scheduler TTA.

MODELO	RSS		
	DELAY	JITTER	PACKET LOSS
loess	23478.25	9800.72	9104.43
ppr	23318.2	9751.43	8908.11
locfit	23486.79	9818.58	9117.12
gam	23581.89	9843.25	9149.5
sm	23567.71	9819.23	-19657300.15
polymars	23591.9	9846.44	8972.71
mars	23591.9	9846.44	9123.04
coxph	1.00	1.00	6562142.88

Tabla 5.40: Modelos No Paramétricos MAC-IP - Scheduler TTA.

MODELO	DELAY			JITTER			PACKET LOSS		
	RESIDUO	MEDIA	VARIANZA	RESIDUO	MEDIA	VARIANZA	RESIDUO	MEDIA	VARIANZA
gaussian	0.00	0.00	23591.07	0.00	0.00	9847.08	0.00	0.00	9153.07
loess	-5820.59	-2.26	23482.26	-2129.62	-0.83	9803.85	-5223.74	-2.03	9103.84
ppr	0.00	0.00	23327.28	0.00	0.00	9755.23	0.00	0.00	8911.57
locfit	861.59	0.34	23495.82	736.5	0.29	9822.32	-1971.46	-0.77	9120.08
gam	0.00	0.00	23591.07	0.00	0.00	9847.08	0.00	0.00	9153.07
sm	-982.44	-0.38	23576.74	496.75	0.19	9823.02	-27002.5	-10.51	96.74
polymars	0.00	0.00	23601.08	0.00	0.00	9850.28	0.00	0.00	8976.2
mars	0.00	0.00	23601.08	0.00	0.00	9850.28	0.00	0.00	9126.59
coxph	0.00	0.00	1.00	0.00	0.00	1.00	0.00	0.00	0.90

Tabla 5.41: Residuos MAC-IP - Scheduler TTA.

5.5.6. FD-TBFQ

Esta sección presenta los valores de los modelos para PHY-MAC y MAC-IP para el *scheduler* FD-TBFQ.

Modelos PHY - MAC

MODELO	COEFICIENTES	ERROR STD	P-VALUE	AIC
Gausiano	2.800e+01	1.238e-15	<2e-16	-21764
	1.621e-05	4.916e-0	0.974	
	-7.616e-21	2.335e-19	0.974	
Poisson	3.332e+00	1.106e-02	<2e-16	1879.7
	1.072e-05	4.391e+09	1	
	-5.038e-21	2.086e-06	1	

Tabla 5.42: Modelos Paramétricos PHY-MAC - Scheduler FD-TBFQ.

MODELO	RSS
loess	3.496368e-19
ppr	0
locfit	0
gam	4.531635e-28
sm	5.590972e-22
polymars	0
mars	4.531287e-28
coxph	5.04871e-29

Tabla 5.43: Modelos No Paramétricos PHY-MAC - Scheduler FD-TBFQ.

La Tabla de Residuos de este *scheduler* es la siguiente:

MODELO	RESIDUO	MEDIA	VARIANZA
gaussian	0.00	0.00	0.00
poisson	5.913039e-06	1.633436e-08	9.281689e-16
loess	-2.053741e-08	-5.673318e-11	3.473777e-19
ppr	0.00	0.00	0.00
locfit	0.00	0.00	0.00
gam	-4.085621e-13	-1.128625e-15	4.531415e-28
sm	-9.590408e-10	-2.649284e-12	5.536078e-22
polymars	0.00	0.00	0.00
mars	7.695178e-12	2.12574e-14	1.255204e-30
coxph	-2.732925e-12	-7.549517e-15	0.00

Tabla 5.44: Residuos PHY-MAC - Scheduler FD-TBFQ.

Modelos MAC - IP

VARIABLE	MODELO	COEFICIENTES	ERROR STD	P-VALUE	AIC
DELAY	Gausiano	3.145e+10 NA -1.123e+09	8.983e+10 NA 3.208e+09	0.726 NA 0.726	33961
JITTER	Gausiano	1.822e+11 NA -6.506e+09	5.582e+10 NA 1.993e+09	0.00111 NA 0.00111	31263
PACKET LOSS	Gausiano	-4.218e+10 NA 1.507e+09	2.977e+10 NA 1.063e+09	0.157 NA 0.157	27700

Tabla 5.45: Modelos Paramétricos MAC-IP - Scheduler FD-TBFQ.

MODELO	RSS		
	DELAY	JITTER	PACKET LOSS
locfit	9237.05	3565.61	1012.31
gam	9281.15	3597.33	1020.58
coxph	0.83	0.9	7771083.5

Tabla 5.46: Modelos No Paramétricos MAC-IP - Scheduler FD-TBFQ.

MODELO	DELAY			JITTER			PACKET LOSS		
	RESIDUO	MEDIA	VARIANZA	RESIDUO	MEDIA	VARIANZA	RESIDUO	MEDIA	VARIANZA
gaussian	1.54	0	9284.01	9.02	0	3585.14	-2.09	0	1020.22
locfit	1907.24	0.67	9239.85	1102	0.39	3566.71	-913.19	-0.32	1012.56
gam	0	0	9284.4	0	0	3598.58	0	0	1020.94
coxph	0	0	0.83	0	0	0.9	0	0	0.28

Tabla 5.47: Residuos MAC-IP - Scheduler FD-TBFQ.

5.5.7. TD-TBFQ

A continuación se detalla los valores de los modelos paramétricos y no paramétricos para este *scheduler*.

Modelos PHY - MAC

MODELO	COEFICIENTES	ERROR STD	P-VALUE	AIC
Gausiano	2.775e+01	1.432e-01	<2e-16	1768.3
	-4.070e+09	1.401e+10	0.772	
	1.947e-06	6.652e-06	0.770	
Poisson	3.323e+00	1.090e-02	<2e-16	2114.3
	-1.459e+08	1.067e+09	0.891	
	6.981e-08	5.068e-07	0.890	

Tabla 5.48: Modelos Paramétricos PHY-MAC - Scheduler TD-TBFQ.

MODELO	RSS
ppr	6.12
locfit	6.05
gam	6.17
sm	6.15
polymars	6.17
mars	6.17
coxph	0.01

Tabla 5.49: Modelos No Paramétricos PHY-MAC - Scheduler TD-TBFQ.

MODELO	RESIDUO	MEDIA	VARIANZA
gaussian	0.00	0.00	6.18
poisson	-6.57	-0.02	0.44
ppr	0.00	0.00	6.13
locfit	-4.57	-0.01	6.06
gam	0.00	0.00	6.18
sm	1.44	0.00	6.16
polymars	0.00	0.00	6.19
mars	0.00	0.00	6.19
coxph	0.00	0.00	0.01

Tabla 5.50: Residuos PHY-MAC - Scheduler TD-TBFQ.

Modelos MAC - IP

VARIABLE	MODELO	COEFICIENTES	ERROR STD	P-VALUE	AIC
DELAY	Gausiano	-0.452174	1.451566	0.755	6385.3
		0.001080	0.005627	0.848	
		0.016521	0.052732	0.754	
JITTER	Gausiano	-18.39490	43.56489	0.673	24427
		0.02299	0.16889	0.892	
		0.69389	1.58261	0.661	
PACKET LOSS	Gausiano	1.002e+02	5.263e-01	<2e-16	1004
		-3.425e-04	2.040e-03	0.867	
		-7.657e-03	1.912e-02	0.689	

Tabla 5.51: Modelos Paramétricos MAC-IP - Scheduler TD-TBFQ.

MODELO	RSS		
	DELAY	JITTER	PACKET LOSS
ppr	0.64	581.96	0.08
locfit	0.64	583.83	0.08
gam	0.65	584.08	0.09
polymars	0.65	584.13	0.08
mars	0.64	584.13	0.08
coxph	0.09	0.52	6655003.26

Tabla 5.52: Modelos No Paramétricos MAC-IP - Scheduler TD-TBFQ.

MODELO	DELAY			JITTER			PACKET LOSS		
	RESIDUO	MEDIA	VARIANZA	RESIDUO	MEDIA	VARIANZA	RESIDUO	MEDIA	VARIANZA
gaussian	0.00	0.00	0.65	0.00	0.00	584.3	0.00	0.00	0.09
ppr	0.00	0.00	0.64	0.00	0.00	582.18	0.00	0.00	0.08
locfit	-23.06	-0.01	0.64	-25.39	-0.01	584.05	7.58	0.00	0.08
gam	0.00	0.00	0.65	0.00	0.00	584.3	0.00	0.00	0.09
polymars	0.00	0.00	0.65	0.00	0.00	584.36	0.00	0.00	0.08
mars	0.00	0.00	0.64	0.00	0.00	584.36	0.00	0.00	0.08
coxph	0.00	0.00	0.09	0.00	0.00	0.52	0.00	0.00	0.00

Tabla 5.53: Residuos MAC-IP - Scheduler TD-TBFQ.

5.5.8. PSS

Siguiendo el esquema anterior, primero se busca los modelos paramétricos y no paramétricos para la parte PHY-MAC para seguir con los mismos métodos en la sección MAC-IP.

Modelos PHY - MAC

MODELO	COEFICIENTES	ERROR STD	P-VALUE	AIC
Gausiano	1.949e+01	3.823e-01	<2e-16	2390.8
	5.015e+08	2.994e+08	0.0948	
	2.457e-07	3.699e-07	0.5070	
Poisson	2.975e+00	1.298e-02	<2e-16	2667.3
	2.242e+07	8.917e+06	0.0119	
	7.545e-09	1.081e-08	0.4854	

Tabla 5.54: Modelos Paramétricos PHY-MAC - Scheduler PSS.

MODELO	RSS
loess	706.42
ppr	25.54
locfit	35.88
gam	42.29
sm	39.79
polymars	36.87
mars	37.8
coxph	0.8

Tabla 5.55: Modelos No Paramétricos PHY-MAC - Scheduler PSS.

MODELO	RESIDUO	MEDIA	VARIANZA
gaussian	0.00	0.00	42.41
poisson	-37.97	-0.1	2.6
loess	4005.82	11.07	585.59
ppr	0.00	0.00	25.61
locfit	-2.93	-0.01	35.98
gam	0.00	0.00	42.41
sm	47.06	0.13	39.89
polymars	0.00	0.00	36.97
mars	0.00	0.00	37.9
coxph	0.00	0.00	0.8

Tabla 5.56: Residuos PHY-MAC - Scheduler PSS.

Modelos MAC - IP

VARIABLE	MODELO	COEFICIENTES	ERROR STD	P-VALUE	AIC
DELAY	Gausiano	48.7751	4.5636	<2e-16	32583
		-0.5981	0.2112	0.00465	
		5.4793	6.3553	0.38867	
JITTER	Gausiano	19.0116	2.9435	1.24e-10	30087
		-0.1058	0.1362	0.437	
		0.7114	4.0991	0.862	
PACKET LOSS	Gausiano	13.57197	1.12402	<2e-16	24608
		-0.28687	0.05201	3.78e-08	
		-1.39570	1.56533	0.373	

Tabla 5.57: Modelos Paramétricos MAC-IP - Scheduler PSS.

MODELO	RSS		
	DELAY	JITTER	PACKET LOSS
loess	5423.74	2269.14	331.34
ppr	5378.16	2255.85	327.71
locfit	5416.18	2266.91	328.92
gam	5475.9	2278.06	332.19
sm	5468.72	2276.05	-23000.98
polymars	5382.5	2278.56	327.04
mars	5449.59	2278.56	331.75
coxph	1.04	1.00	8173855.84

Tabla 5.58: Modelos No Paramétricos MAC-IP - Scheduler PSS.

MODELO	DELAY			JITTER			PACKET LOSS		
	RESIDUO	MEDIA	VARIANZA	RESIDUO	MEDIA	VARIANZA	RESIDUO	MEDIA	VARIANZA
gaussian	0.00	0.00	5477.83	0.00	0.00	2278.86	0.00	0.00	332.31
loess	-2171.44	-0.76	5425.06	-938.33	-0.33	2269.83	-457.46	-0.16	331.43
ppr	0.00	0.00	5380.05	0.00	0.00	2256.64	0.00	0.00	327.82
locfit	70.32	0.02	5418.09	190.57	0.07	2267.7	532.93	0.19	329
gam	0.00	0.00	5477.83	0.00	0.00	2278.86	0.00	0.00	332.31
sm	-607.11	-0.44	5470.6	-267.52	-0.09	2276.84	36963.22	12.99	70.97
polymars	0.00	0.00	5384.4	0.00	0.00	2279.36	0.00	0.00	327.15
mars	0.00	0.00	5451.51	0.00	0.00	2279.36	0.00	0.00	331.87
coxph	0.00	0.00	1.04	0.00	0.00	1.00	0.00	0.00	0.93

Tabla 5.59: Residuos MAC-IP - Scheduler PSS.

5.5.9. CQA

Al igual que los demás *schedulers* se busca modelos para la parte PHY-MAC y MAC-IP.

Modelos PHY - MAC

Los valores de los modelos paramétricos para PHY-MAC de este *scheduler* se presentan a continuación.

MODELO	COEFICIENTES	ERROR STD	P-VALUE	AIC
Gausiano	2.303e+01	2.927e-01	<2e-16	2198.2
	2.898e+08	2.804e+08	0.302	
	1.181e-07	2.477e-07	0.634	
Poisson	3.138e+00	1.205e-02	<2e-16	2278
	1.179e+07	1.096e+07	0.282	
	3.890e-09	9.614e-09	0.686	

Tabla 5.60: Modelos Paramétricos PHY-MAC - Scheduler CQA.

Los valores modelos no paramétricos elegidos para la parte PHY-MAC es la siguiente:

MODELO	RSS
loess	133.39
ppr	16.57
locfit	20.68
gam	24.84
sm	23.79
polymars	20.32
mars	22.58
coxph	0.74

Tabla 5.61: Modelos No Paramétricos PHY-MAC - Scheduler CQA.

La Tabla de Residuos para PHY-MAC.

MODELO	RESIDUO	MEDIA	VARIANZA
gaussian	0.00	0.00	24.91
gamma	-17.62	-0.05	1.32
loess	1454.72	4.02	117.57
ppr	0.00	0.00	16.62
locfit	17.69	0.05	20.74
gam	0.00	0.00	24.91
sm	31.81	0.09	23.85
polymars	0.00	0.00	20.37
mars	0.00	0.00	22.64
coxph	0.00	0.00	0.74

Tabla 5.62: Residuos PHY-MAC - Scheduler CQA.

Modelos MAC - IP

Los modelos de las tres variables de interés: delay, jitter y packet loss se detallan a continuación:

VARIABLE	MODELO	COEFICIENTES	ERROR STD	P-VALUE	AIC
DELAY	Gausiano	75.5572	7.0597	<2e-16	32648
		-1.5594	0.2921	1.01e-07	
		-14.1066	7.4771	0.0593	
JITTER	Gausiano	37.6115	4.5215	<2e-16	30125
		-0.7679	0.1871	4.17e-05	
		-6.5386	4.7888	0.172	
PACKET LOSS	Gausiano	44.2233	2.4383	<2e-16	26627
		-1.3436	0.1009	<2e-16	
		-0.2563	2.5825	0.921	

Tabla 5.63: Modelos Paramétricos MAC-IP - Scheduler CQA.

MODELO	RSS		
	DELAY	JITTER	PACKET LOSS
loess	5910.18	2424.38	705.56
ppr	5888.54	2409.81	704.1
locfit	5902.45	2420.48	705.82
gam	5929.27	2432.17	707.32
sm	5917.87	2428.54	-67390.5
polymars	5935.16	2433.41	707.37
mars	5924.54	2432.34	706.57
coxph	1.03	1.03	8084685.52

Tabla 5.64: Modelos No Paramétricos MAC-IP - Scheduler CQA.

MODELO	DELAY			JITTER			PACKET LOSS		
	RESIDUO	MEDIA	VARIANZA	RESIDUO	MEDIA	VARIANZA	RESIDUO	MEDIA	VARIANZA
gaussian	0.00	0.00	5931.36	0.00	0.00	2433.03	0.00	0.00	707.57
poisson	3616.32	-1.28	54.15	-4955.8	-1.75	47.61	-4439.36	-1.57	29.89
loess	-1033.59	-0.36	5912.13	-652.32	-0.23	2425.18	-1256.44	-0.44	705.61
ppr	0.00	0.00	5890.62	0.00	0.00	2410.67	0.00	0.00	704.34
locfit	1183.73	0.42	5904.36	967.45	0.34	2421.21	-159.69	-0.06	706.06
gam	0.00	0.00	5931.36	0.00	0.00	2433.03	0.00	0.00	707.57
sm	1320.34	0.47	5919.74	794.86	0.28	2429.32	32789.55	11.58	136.48
polymars	0.00	0.00	5937.25	0.00	0.00	2434.27	0.00	0.00	707.62
mars	0.00	0.00	5926.63	0.00	0.00	2433.2	0.00	0.00	706.82
coxph	0.00	0.00	1.03	0.00	0.00	1.03	0.00	0.00	0.96

Tabla 5.65: Residuos MAC-IP - Scheduler CQA.

Todos los modelos aplicados pierden información, sin embargo la idea es buscar el modelo que mejor se ajuste los datos obtenidos.

Capítulo 6

Análisis de Resultados

6.1. Resultados

Esta sección presenta el resumen de aplicar el Test de Kolmogorov-Smirnov [50]. La decisión de no hacerlo mediante criterios de p-value o el AIC, corresponde a la forma de construcción de los mismos.

El test de Kolmogorov-Smirnov permite comparar dos conjuntos de datos, en este caso se hace con la variable de interés y los diversos modelos calculados para las capas. En teoría este contraste, compara la función de distribución (probabilidad acumulada) teórica con la observada, y calcula un valor de diferencia, representado como D que corresponde a la diferencia máxima en valor absoluto entre la distribución observada y la distribución teórica, proporcionando un valor de probabilidad P.

Cada scheduler se divide en tres secciones: PHY-MAC, MAC-IP e IP-MOS; y en todas se realiza el test. En cada etapa, se observa la significancia del test y se elige el modelo con menor valor estadístico D, partiendo que el nivel de significación del contraste para todos los modelos por *default* es $\alpha = 0.05$.

En el caso de la parte PHY-MAC se busca obtener un modelo de la forma mcs $\sim$ rsrp+sinr. Al obtenerlo, este modelo sirve como variable de entrada a la sección MAC-IP, de esta manera se obtiene los modelos más adecuados para nuestras variables de interés: delay, jitter y packet loss.

Los mejores modelos de las variables de interés a su vez sirven como parámetros *input* en la función E-MODEL.

Finalmente se realiza un test para comparar los valores obtenidos de aplicar E-MODEL con los parámetros proporcionados por el Flow Monitor y los valores resultantes luego de aplicar la cadena de los mejores modelos en cada etapa.

El diagrama de la Fig. 6.1 muestra el flujo de análisis de datos recién descrito.

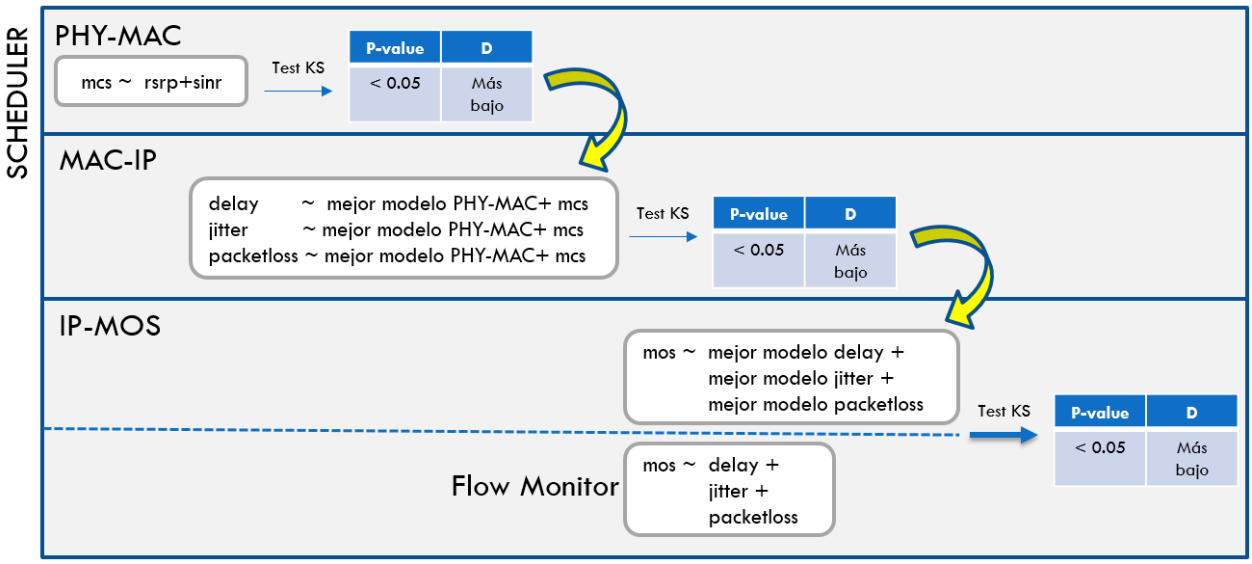

Figura 6.1: Diagrama de Flujo de Análisis de Resultados.

6.1.1. RR

- Modelos MAC-PHY

  Para este caso el mejor modelo, es el obtenido por la regresión de Cox. Este modelo se ingresa como "mejor modelo"para la siguiente etapa.

MODELO	P-VALUE	D
gausiano	0.000	0.4258242
poisson	0.000	0.4368132
loess	0.000	0.3324176
ppr	0.000	0.3241758
locfit	0.000	0.3269231
gam	0.000	0.4258242
sm	0.000	0.3296703
polymars	0.000	0.3269231
mars	0.000	0.3626374
coxph	2.997359e-09	0.2362637

Tabla 6.1: Valores del test KS MAC-PHY - Scheduler RR.

- Modelos IP-MAC

MODELO	DELAY		JITTER		PACKET LOSS	
	P-VALUE	D	P-VALUE	D	P-VALUE	D
gausian	0.000	0.8284331	0.000	0.7817364	0.000	0.7104808
loess	0.000	0.8007610	0.000	0.7644414	0.000	0.7080595
ppr	0.000	0.6502940	0.000	0.7229332	0.000	0.7080595
locfit	0.000	0.8443445	0.000	0.7862331	0.000	0.7108267
gam	0.000	0.8284331	0.000	0.7817364	0.000	0.7104808
sm	0.000	0.8398478	0.000	0.7865790	0.000	0.7094431
polymars	0.000	0.7471463	0.000	0.7350398	0.000	0.7101349
mars	0.000	0.8695953	0.000	0.8031823	0.000	0.7142857
coxph	0.000	0.2113456	0.000	0.2621930	0.000	1.0000000

Tabla 6.2: Valores del test KS IP-MAC - Scheduler RR.

Para las variables delay, jitter el modelo que mejor se ajusta es el proporcionado por la regresión de Cox. Por otro lado, para la variable packet-loss los modelos de LOESS y PPR se aproximan más a sus datos, por lo tanto se elige al segundo como mejor modelo de esta variable.

- Función MOS-IP

  El contraste de los valores proporcionado por el test K-S indica que existe gran diferencia entre los valores de las variables de interés proporcionados por los modelos y por los datos de Flow Monitor.

	DELAY		JITTER		PACKET LOSS	
	P-VALUE	D	P-VALUE	D	P-VALUE	D
Modelo Flow Monitor	<2.2e-16	0.9699	<2.2e-16	0.9042	<2.2e-16	0.776

Tabla 6.3: Valores del test KS MOS-IP - Scheduler RR.

Los gráficos correspondientes al MOS con los valores proporcionados por los "mejores modelos" y por los datos obtenidos por el Flow Monitor se observa claramente que el MOS de los modelos tiene una media de 2.6, es decir entre un nivel pobre-aceptable. Por su parte, el MOS para el mismo scheduler con los datos del Flow Monitor está alrededor de los 3.6 (entre un nivel aceptable y bueno).

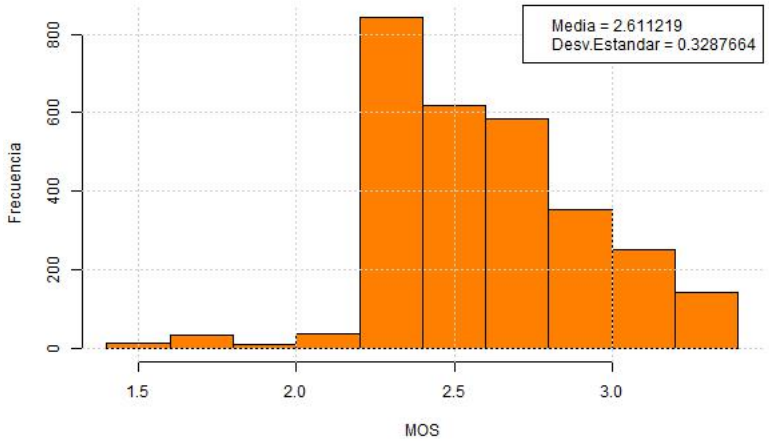

Figura 6.2: MOS (Modelo) - Scheduler RR.

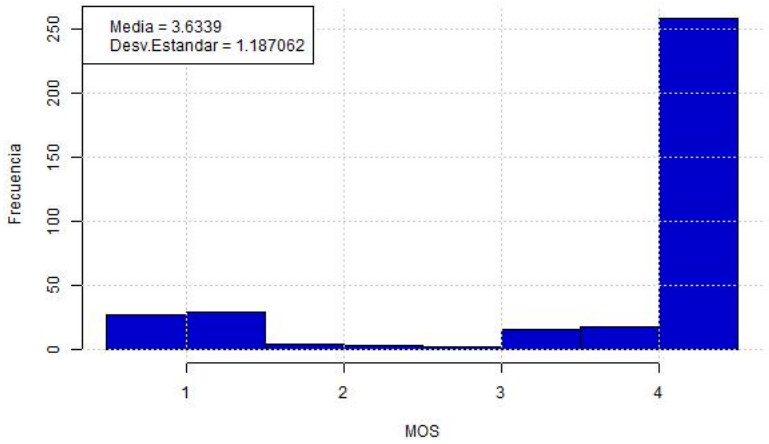

Figura 6.3: MOS (FlowMonitor) - Scheduler RR.

6.1.2. PF

- Modelos MAC-PHY

  El modelo proporcionado por la regresión de Cox es el que mejor se adecúa para los datos siguiendo la forma mcs $\sim$ rsrp+sinr

MODELO	P-VALUE	D
gausiano	0.000	0.4203297
poisson	0.000	0.4203297
loess	0.000	0.3543956
ppr	7.085e-10	0.2445055
locfit	4.374e-14	0.2939560
gam	0.000	0.4203297
sm	0.000	0.3653846
polymars	0.000	0.3269231
mars	0.000	0.3296703
coxph	7.627e-09	0.2307692

Tabla 6.4: Valores del test KS MAC-PHY - Scheduler PF.

- Modelos IP-MAC

  La regresión de Cox es el mejor modelos para las variables delay y jitter siguiendo la forma delay $\sim$ mcs+mejor modelo phy-mac y jitter $\sim$ mcs+mejor modelo phy-mac, respectivamente. Para el caso packet loss $\sim$ mcs+mejor modelo la regresión por *projection pursuit* se adecua a los datos obtenidos.

MODELO	DELAY		JITTER		PACKET LOSS	
	P-VALUE	D	P-VALUE	D	P-VALUE	D
gausian	0.000	0.8374656	0.000	0.7689394	0.000	0.7031680
loess	0.000	0.8388430	0.000	0.7703168	0.000	0.7045455
ppr	0.000	0.7338154	0.000	0.6911157	0.000	0.6962810
locfit	0.000	0.8426309	0.000	0.7820248	0.000	0.7073003
gam	0.000	0.8374656	0.000	0.7689394	0.000	0.7031680
sm	0.000	0.8529614	0.000	0.7754821	0.000	0.7045455
polymars	0.000	0.7706612	0.000	0.7858127	0.000	0.7069559
mars	0.000	0.8522727	0.000	0.7933884	0.000	0.7090220
coxph	0.000	0.1814738	0.000	0.2393251	0.000	1.0000000

Tabla 6.5: Valores del test KS IP-MAC - Scheduler PF.

- Función MOS-IP

  La diferencia entre los valores del MOS obtenido por los mejores modelos y los datos provenientes del Flow Monitor es relativamente alta, sobre todo en lo referente a las variables delay e jitter.

	DELAY		JITTER		PACKET LOSS	
	P-VALUE	D	P-VALUE	D	P-VALUE	D
Modelo Flow Monitor	<2.2e-16	0.9699	<2.2e-16	0.9436	<2.2e-16	0.7906

Tabla 6.6: Valores del test KS MOS-IP - Scheduler PF.

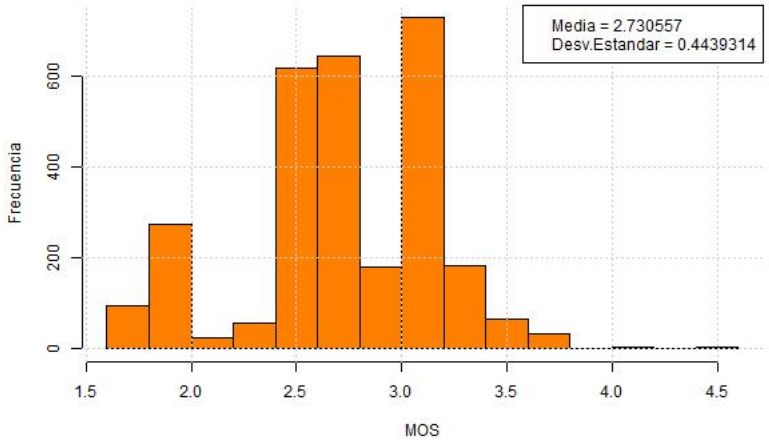

Figura 6.4: MOS (Modelo) - Scheduler PF.

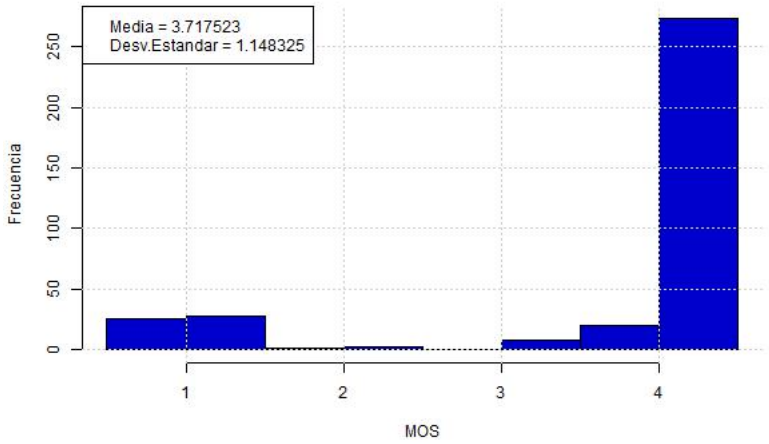

Figura 6.5: MOS (FlowMonitor) - Scheduler PF.

El MOS de este scheduler según los datos proporcionados por los mejores modelos se encuentra alrededor de 2.7 (pobre-aceptable). El MOS con los datos obtenidos por el Flow Monitor supera un poco el valor previo. En este caso su media está alrededor de 3.7 tal como lo indica la Fig. 6.5

6.1.3. FD-MT

- Modelos MAC-PHY

  Para esta sección el modelo que mejor se adecua a la forma mcs $\sim$ rsrp+sinr es el propuesto por la regresión de Cox.

MODELO	P-VALUE	D
gausiano	0.000	0.4653740
poisson	0.000	0.4653740
loess	0.000	0.3933518
ppr	0.000	0.3407202
locfit	0.000	0.3434903
gam	0.000	0.4653740
sm	0.000	0.3795014
polymars	0.000	0.4044321
mars	0.000	0.3795014
coxph	6.090684e-13	0.2825485

Tabla 6.7: Valores del test KS MAC-PHY - Scheduler FD-MT.

- Modelos IP-MAC

MODELO	DELAY		JITTER		PACKET LOSS	
	P-VALUE	D	P-VALUE	D	P-VALUE	D
gausian	0.000	0.8376981	0.000	0.7536182	0.000	0.7587870
loess	0.000	0.7815300	0.000	0.7284631	0.000	0.7446589
ppr	0.000	0.7325982	0.000	0.7253618	0.000	0.6709166
locfit	0.000	0.8252929	0.000	0.7498277	0.000	0.7939352
gam	0.000	0.8376981	0.000	0.7536182	0.000	0.7587870
sm	0.000	0.8328739	0.000	0.7587870	0.000	0.7584425
polymars	0.000	0.8259821	0.000	0.7515507	0.000	0.6943487
mars	0.000	0.8380427	0.000	0.7698139	0.000	0.7953136
coxph	0.000	0.1247416	0.000	0.1833218	0.000	1.0000000

Tabla 6.8: Valores del test KS IP-MAC - Scheduler FD-MT.

Para las variables delay y jitter, el mejor modelo lo proporciona la regresión de Cox. Para la variable packet loss de la forma $\sim$ mcs+mejor modelo la regresión por *Projection Pursuit* es la mejor.

- Función MOS-IP

  Los valores obtenidos del MOS estimado mediante los mejores modelos y los datos provenientes del Flow Monitor son relativamente altos, sobre todo en las variables delay e jitter.

	DELAY		JITTER		PACKET LOSS	
	P-VALUE	D	P-VALUE	D	P-VALUE	D
Modelo Flow Monitor	<2.2e-16	0.9617	<2.2e-16	0.9525	<2.2e-16	0.7889

Tabla 6.9: Valores del test KS MOS-IP - Scheduler FD-MT.

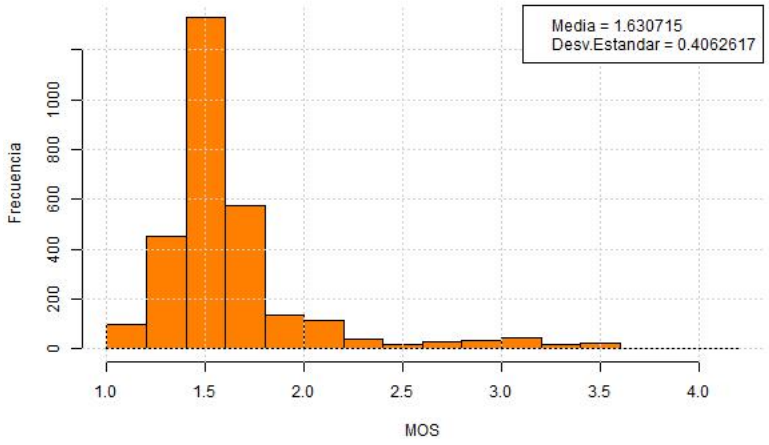

Figura 6.6: MOS (Modelo) - Scheduler FD-MT.

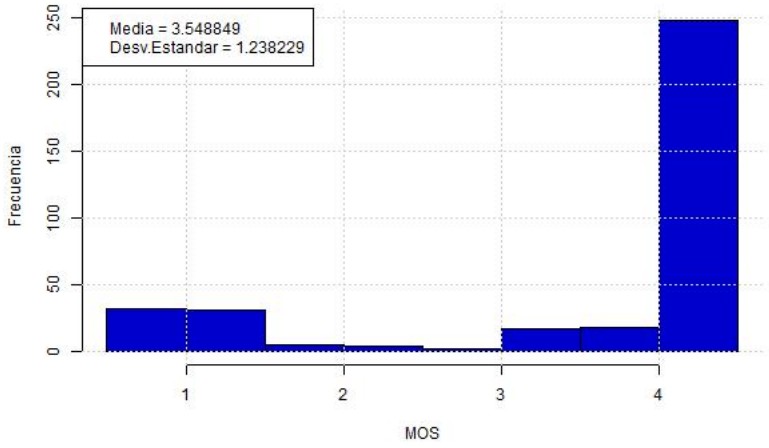

Figura 6.7: MOS (Flow Monitor) - Scheduler FD-MT.

Existe una notable diferencia entre el MOS con los datos proporcionados por los mejores modelos y los obtenidos por el Flow Monitor. El MOS de los "mejores modelos" es de alrededor de 1.6 (malo-pobre), por otro lado, el MOS con los datos obtenidos por el Flow Monitor supera considerablemente el valor previo. En este caso su media está alrededor de 3.5 niveles aceptable y bueno.

6.1.4. TD-MT

- Modelos MAC-PHY

  El modelo que se adecua a la forma mcs $\sim$ rsrp+sinr en este scheduler es el proporcionado por PPR (Regresión de Projection Pursuit).

MODELO	P-VALUE	D
gausiano	0.000	0.4432133
poisson	0.000	0.4432133
loess	0.000	0.4127424
ppr	1.068812e-12	0.2797784
locfit	0.000	0.3490305
gam	0.000	0.4432133
sm	0.000	0.3795014
polymars	0.000	0.3961219
mars	0.000	0.3850416
coxph	1.945111e-13	0.2880886

Tabla 6.10: Valores del test KS MAC-PHY - Scheduler TD-MT.

- Modelos IP-MAC

  Para las variables delay y jitter siguiendo la forma delay $\sim$ mcs+mejor modelo phy-mac y jitter $\sim$ mcs+mejor modelo phy-mac, respectivamente, el mejor modelo es el presentado por la regresión de Cox. Para la variable packet loss $\sim$ mcs+mejor, el modelo PPR es que presenta menor diferencia.

MODELO	DELAY		JITTER		PACKET LOSS	
	P-VALUE	D	P-VALUE	D	P-VALUE	D
gausian	0.000	0.8347280	0.000	0.7709205	0.000	0.7887029
loess	0.000	0.7255927	0.000	0.6555091	0.000	0.7444212
ppr	0.000	0.7238494	0.000	0.7207113	0.000	0.6541144
locfit	0.000	0.8347280	0.000	0.7716179	0.000	0.7723152
gam	0.000	0.8347280	0.000	0.7709205	0.000	0.7887029
sm	0.000	0.8029986	0.000	0.7744073	0.000	0.7890516
polymars	0.000	0.8413529	0.000	0.7702232	0.000	0.7576709
mars	0.000	0.8417015	0.000	0.7810321	0.000	0.7911437
coxph	0.000	0.1230823	0.000	0.1708508	0.000	1.0000000

Tabla 6.11: Valores del test KS IP-MAC - Scheduler TD-MT.

- Función MOS-IP

  El test K-S indica que existe gran diferencia entre los valores de las variables de interés proporcionados por los modelos y por los datos de flow monitor, en especial en las variables delay y jitter.

	DELAY		JITTER		PACKET LOSS	
	P-VALUE	D	P-VALUE	D	P-VALUE	D
Modelo Flow Monitor	<2.2e-16	0.959	<2.2e-16	0.9536	<2.2e-16	0.7887

Tabla 6.12: Valores del test KS MOS-IP - Scheduler TD-MT.

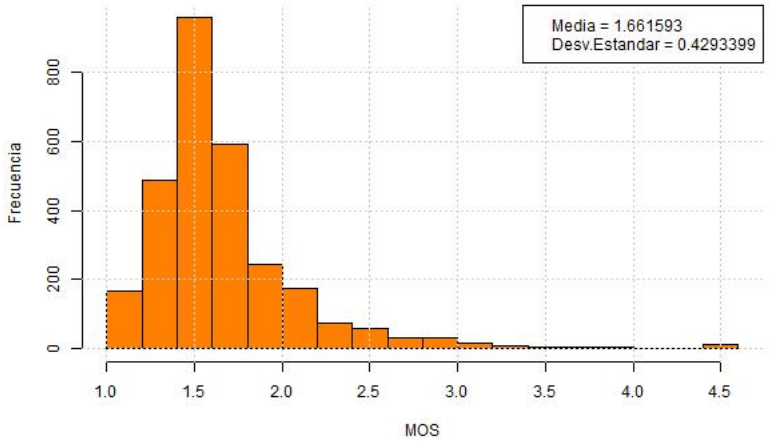

Figura 6.8: MOS (Modelo) - Scheduler TD-MT.

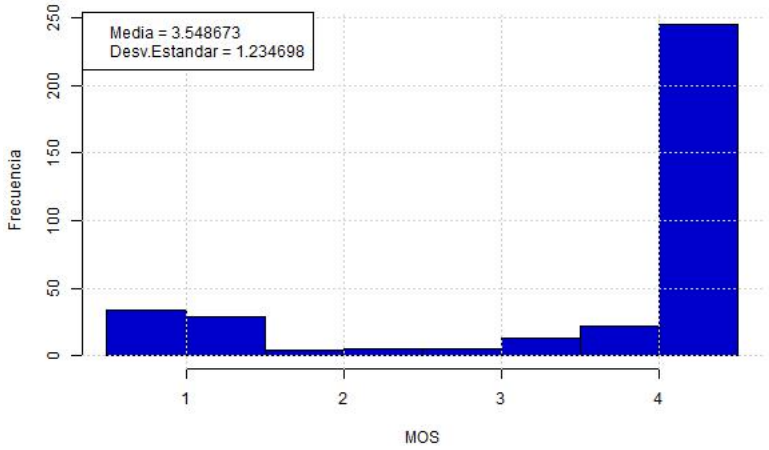

Figura 6.9: MOS (Flow Monitor) - Scheduler TD-MT.

La diferencia entre los valores del MOS es considerable. El MOS obtenido con los datos de los “mejores modelos” es de alrededor de 1.6 considerándolo malo-pobre. El MOS con los datos obtenidos por el Flow Monitor está alrededor de 2.5 entre el nivel pobre-aceptable.

6.1.5. TTA

- Modelos MAC-PHY
  La regresión de Cox es el mejor modelo que representa a los datos de esta sección.

MODELO	P-VALUE	D
gausiano	0.000	0.396875
poisson	0.000	0.396875
loess	0.000	0.375000
ppr	7.398875e-08	0.231250
locfit	3.129500e-10	0.265625
gam	0.000	0.396875
sm	1.110223e-16	0.343750
polymars	1.833997e-10	0.268750
mars	8.943017e-10	0.259375
coxph	6.911024e-07	0.215625

Tabla 6.13: Valores del test KS MAC-PHY - Scheduler TTA.

- Modelos IP-MAC

  Los valores de las variables delay y jitter siguen el modelo de la regresión de Cox; y el modelo de la variable packet-loss para esta sección es PPR.

MODELO	DELAY		JITTER		PACKET LOSS	
	P-VALUE	D	P-VALUE	D	P-VALUE	D
gausian	0.000	0.8536965	0.000	0.8315175	0.000	0.7210117
loess	0.000	0.8673152	0.000	0.8233463	0.000	0.7178988
ppr	0.000	0.7400778	0.000	0.7408560	0.000	0.5797665
locfit	0.000	0.8782101	0.000	0.8369650	0.000	0.7206226
gam	0.000	0.8536965	0.000	0.8315175	0.000	0.7210117
sm	0.000	0.8439689	0.000	0.8073930	0.000	0.7295720
polymars	0.000	0.8929961	0.000	0.8509728	0.000	0.6299611
mars	0.000	0.8929961	0.000	0.8509728	0.000	0.6836576
coxph	0.000	0.1731518	0.000	0.1972763	0.000	0.9996109

Tabla 6.14: Valores del test KS IP-MAC - Scheduler TTA.

- Función MOS-IP

  El valor obtenido por el test K-S prevé que la diferencia entre los valores de la variable delay proporcionados por los modelos y por los datos de flow monitor es significativa, así mismo para la variable jitter. Aunque para la variable packet loss existe un valor estadístico D considerable, no se compara con los valores de sus otras variables de interés.

	DELAY		JITTER		PACKET LOSS	
	P-VALUE	D	P-VALUE	D	P-VALUE	D
Modelo Flow Monitor	<2.2e-16	0.8443	<2.2e-16	0.8279	<2.2e-16	0.5934

Tabla 6.15: Valores del test KS MOS-IP - Scheduler TTA.

Las figuras 6.10 y 6.11 representan el MOS de los “mejores modelos” y el obtenido por el

Flow Monitor. El primero tiene una media de 1.4 (malo-pobre), en el segundo, el valor de la media se duplica a comparación del primero, cayendo en el nivel de aceptable.

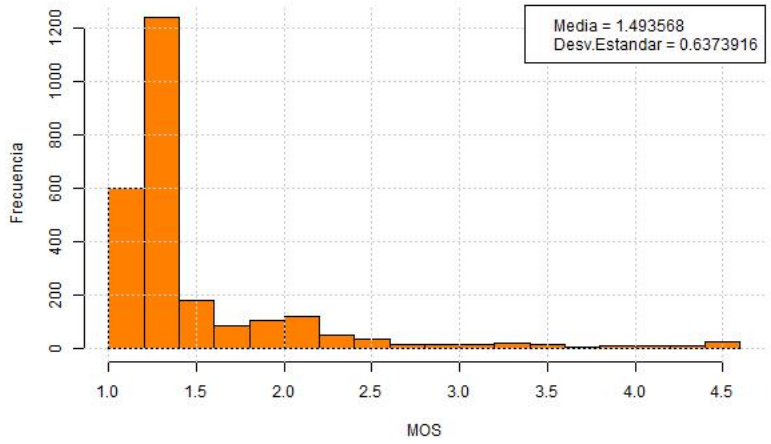

Figura 6.10: MOS (Modelo) - Scheduler TTA.

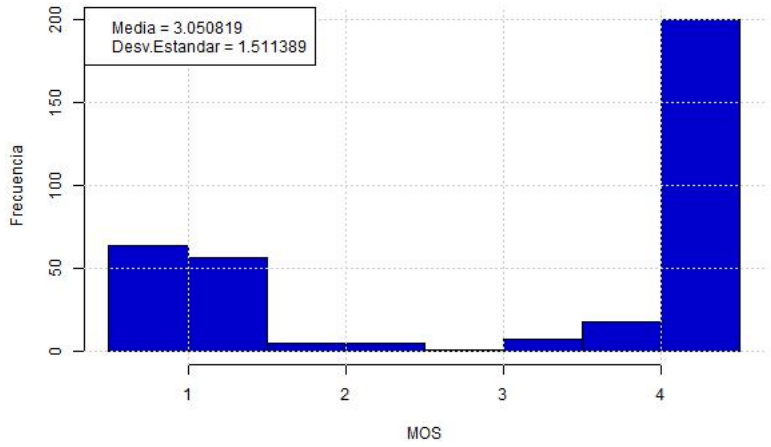

Figura 6.11: MOS (Flow Monitor) - Scheduler TTA.

6.1.6. FD-TBFQ

- Modelos MAC-PHY

  Este scheduler presenta algunos modelos que no son significativos, luego de su análisis el modelo que tiene menor diferencia es loess (Regresión Polinomial Local Ponderada).

MODELO	P-VALUE	D
gausiano	0.000	1.0000
poisson	0.000	1.0000
loess	0.000	0.6629
ppr	1.000	0.0000
locfit	1.000	0.0000
gam	1.000	0.0055
polymars	1.000	0.0000
mars	0.000	0.9972
coxph	0.000	1.0000

Tabla 6.16: Valores del test KS MAC-PHY - Scheduler FD-TBFQ.

- Modelos IP-MAC

  Aunque para este scheduler se probó pocos modelos, la tendencia sigue para las variables delay, jitter y el modelo que mejor se ajusta con la regresión de Cox. Por otro lado, la variable packet-loss sigue el modelo proporcionado por locfit, se lo puede comprobar con el menor valor del estadístico D.

MODELO	DELAY		JITTER		PACKET LOSS	
	P-VALUE	D	P-VALUE	D	P-VALUE	D
gausian	0.000	0.8179902	0.000	0.7761771	0.000	0.8215039
locfit	0.000	0.8169361	0.000	0.7758257	0.000	0.8211525
gam	0.000	0.8179902	0.000	0.7793394	0.000	0.8215039
coxph	0.000	0.8155306	0.000	0.7231202	0.000	1.0000000

Tabla 6.17: Valores del test KS IP-MAC - Scheduler FD-TBFQ.

- Función MOS-IP

  Para este scheduler el contraste de los valores proporcionado por el test K-S es significativo para sus tres variables de interés. Confirmando que los datos proporcionados por los modelos difieren de los datos de obtenidos por el Flow Monitor.

	DELAY		JITTER		PACKET LOSS	
	P-VALUE	D	P-VALUE	D	P-VALUE	D
Modelo Flow Monitor	2.2e-16	0.709	<2.2e-16	0.709	1.138e-08	0.7869

Tabla 6.18: Valores del test KS MOS-IP - Scheduler FD-TBFQ.

Los gráficos correspondientes al MOS con los valores proporcionados por los "mejores modelos" y por los datos obtenidos por el "Flow Monitor" son similares. Ambos se encuentran en un nivel malo.

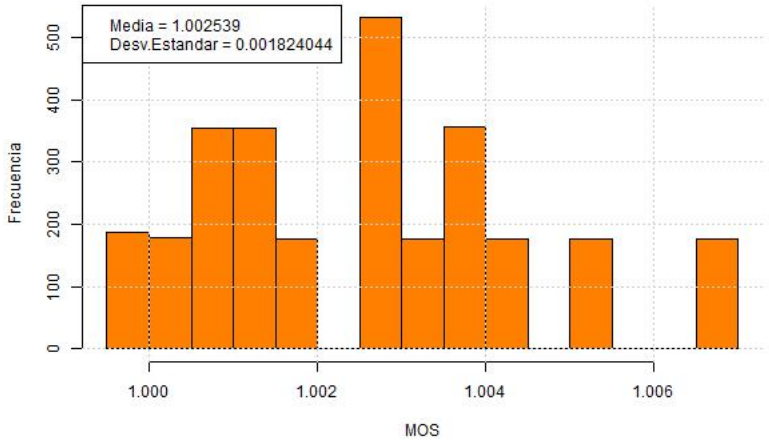

Figura 6.12: MOS (Modelo) - Scheduler FD-TBFQ.

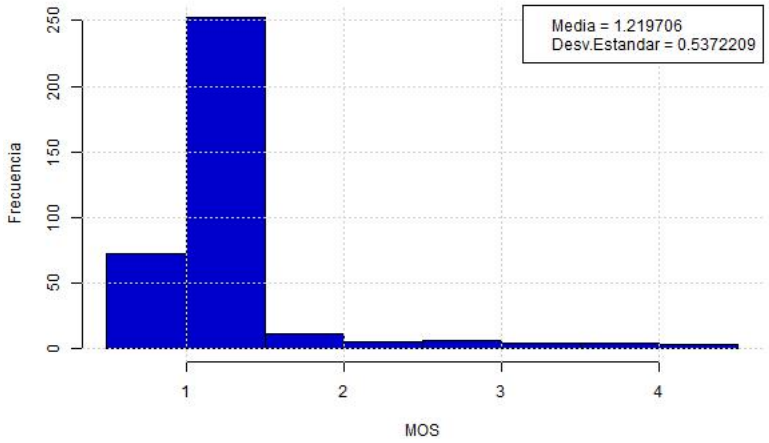

Figura 6.13: MOS (Flow Monitor) - Scheduler FD-TBFQ.

6.1.7. TD TBFQ

- Modelos MAC-PHY

  El mejor modelo para esta sección es el obtenido por locfit (Regresión Polinomial Local). Este modelo se ingresa como “mejor modelo” para la sección IP-MAC.

MODELO	P-VALUE	D
gausiano	0.000	0.9629630
poisson	0.000	0.9629630
loess	0.000	0.8994709
ppr	0.000	0.8227513
locfit	0.000	0.6111111
gam	0.000	0.9629630
sm	0.000	0.9126984
polymars	0.000	0.9920635
mars	0.000	0.9920635
coxph	2.997359e-09	0.9629630

Tabla 6.19: Valores del test KS MAC-PHY - Scheduler TD-TBFQ.

- Modelos IP-MAC

  La variable delay se ajusta al modelo MARS, la jitter sigue al modelo PPR y finalmente la variable packet-loss se aproxima al modelo proporcionado por POLYMARS.

MODELO	DELAY		JITTER		PACKET LOSS	
	P-VALUE	D	P-VALUE	D	P-VALUE	D
gausian	0.000	0.9917044	0.000	0.9506033	0.000	0.9917044
ppr	0.000	0.8269231	0.000	0.8167421	0.000	0.8076923
locfit	0.000	0.6998492	0.000	0.9472097	0.000	0.7088989
gam	0.000	0.9917044	0.000	0.9506033	0.000	0.9917044
polymars	0.000	0.9973605	0.000	0.9513575	0.000	0.6398944
mars	0.000	0.6745852	0.000	0.9513575	0.000	0.6866516
coxph	0.000	0.9917044	0.000	0.9506033	0.000	1.0000000

Tabla 6.20: Valores del test KS IP-MAC - Scheduler TD-TBFQ.

- Función MOS-IP

  El test K-S indica que la diferencia de los valores de las variables de interés proporcionados por los modelos y por los datos de flow monitor es considerable, sobre todo en la variables jitter y packet loss, a diferencia de la mayoría de schedulers.

	DELAY		JITTER		PACKET LOSS	
	P-VALUE	D	P-VALUE	D	P-VALUE	D
Modelo Flow Monitor	<2.2e-16	0.6335	<2.2e-16	0.8224	<2.2e-16	0.8279

Tabla 6.21: Valores del test KS MOS-IP - Scheduler TD-TBFQ.

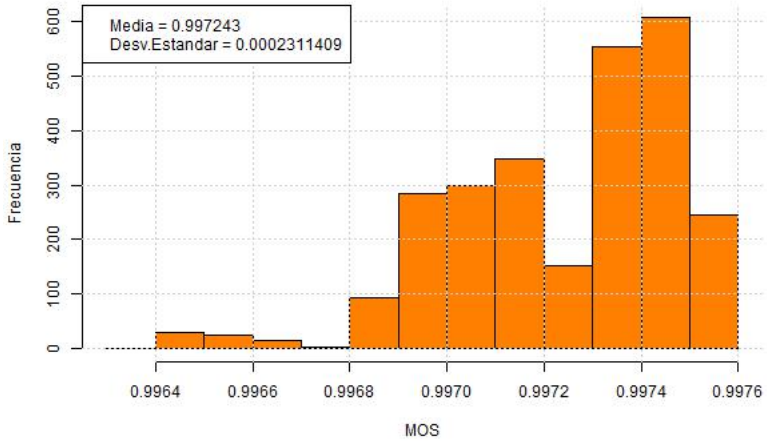

Figura 6.14: MOS (Modelo) - Scheduler TD-TBFQ.

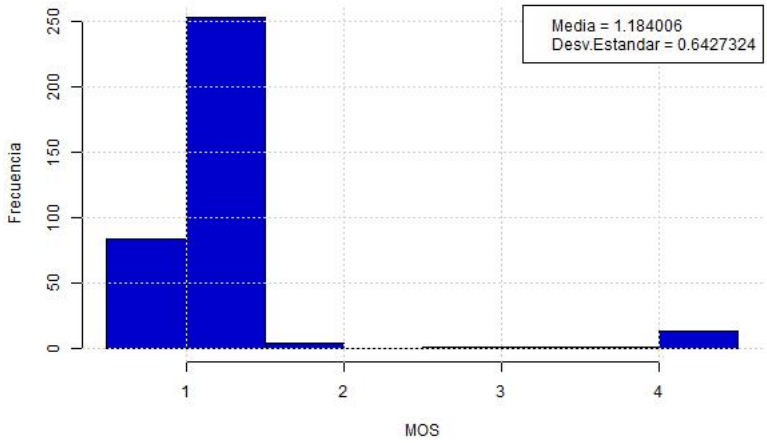

Figura 6.15: MOS (Flow Monitor) - Scheduler TD-TBFQ.

El MOS de TD-TBFQ según los datos proporcionados por los mejores modelos se encuentra alrededor de 0.9 (malo-pobre), valor muy parecido al MOS con los datos obtenidos por el Flow Monitor que es igual a 1.1.

6.1.8. PSS

- Modelos MAC-PHY

  Existen dos modelos que pasan el test de significancia y presentan un valor bajo de diferencia. Aunque el valor utilizado como en la mayoría de los schedulers es la regresión de Cox.

MODELO	P-VALUE	D
gausiano	0.000	0.4033149
poisson	0.000	0.4116022
loess	0.000	0.3646409
ppr	3.831696e-10	0.2486188
locfit	3.330669e-15	0.3066298
gam	0.000	0.4033149
sm	0.000	0.3397790
polymars	0.000	0.3453039
mars	0.000	0.3425414
coxph	1.087021e-08	0.2292818

Tabla 6.22: Valores del test KS MAC-PHY - Scheduler PSS.

- Modelos IP-MAC

  Las variables delay y jitter siguen al modelo proporcionado por la regresión de Cox; mientras que la variable packet-loss presenta un valor estadístico de D igual para los modelos de PPR y POLYMARS, tomando el primero como mejor modelo de esta variable.

MODELO	DELAY		JITTER		PACKET LOSS	
	P-VALUE	D	P-VALUE	D	P-VALUE	D
gausian	0.000	0.8359100	0.000	0.8176388	0.000	0.6746311
loess	0.000	0.8190443	0.000	0.8074491	0.000	0.6711174
ppr	0.000	0.7245257	0.000	0.7803935	0.000	0.6707660
locfit	0.000	0.8313422	0.000	0.8134223	0.000	0.6732256
gam	0.000	0.8359100	0.000	0.8176388	0.000	0.6746311
sm	0.000	0.8348559	0.000	0.8053408	0.000	0.6732256
polymars	0.000	0.7810963	0.000	0.8274772	0.000	0.6707660
mars	0.000	0.8359100	0.000	0.8274772	0.000	0.6746311
coxph	0.000	0.1588194	0.000	0.2329585	0.000	1.0000000

Tabla 6.23: Valores del test KS IP-MAC - Scheduler PSS.

- Función MOS-IP

  El test K-S indica que existe gran diferencia entre los valores de las variables de interés proporcionados por los modelos y por los datos de flow monitor, la mayor diferencia se encuentra en la variable delay.

	DELAY		JITTER		PACKET LOSS	
	P-VALUE	D	P-VALUE	D	P-VALUE	D
Modelo Flow Monitor	2.2e-16	0.9645	<2.2e-16	0.8087	<2.2e-16	0.8155

Tabla 6.24: Valores del test KS MOS-IP - Scheduler PSS.

Este scheduler presenta una diferencia entre el MOS con los datos proporcionados por los mejores modelos y el del Flow Monitor. El MOS con los datos obtenidos por los mejores

modelos tiene una media de 2.5, mientras el obtenido mediante Flow Monitor supera al anterior con un valor de 3.6 tal como lo indica la Fig. 6.17

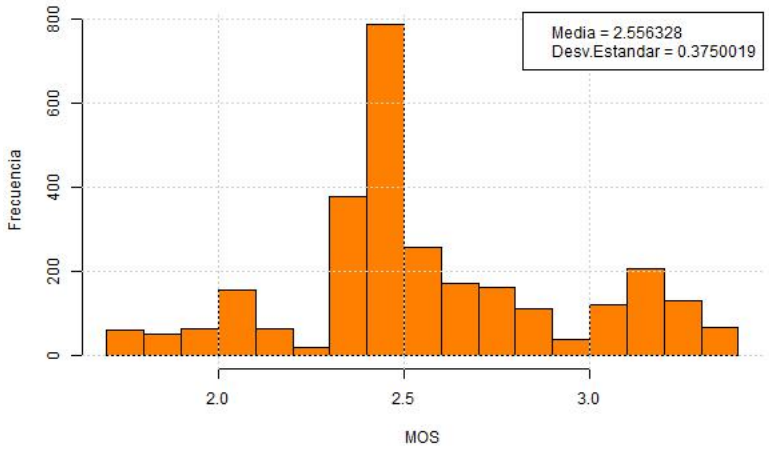

Figura 6.16: MOS (Modelo) - Scheduler PSS.

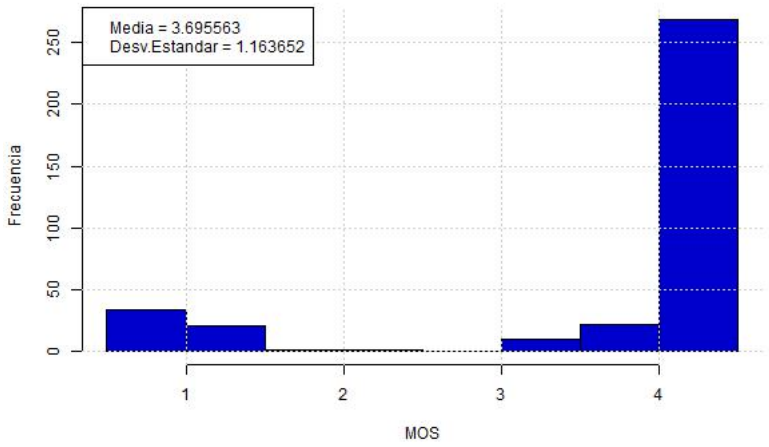

Figura 6.17: MOS (Flow Monitor) - Scheduler PSS.

6.1.9. CQA

- Modelos MAC-PHY

  El mejor modelo para esta sección es el obtenido por la regresión de Cox. Este modelo se ingresa como “mejor modelo” para la siguiente etapa.

MODELO	P-VALUE	D
gausiano	0.000	0.4944751
poisson	0.000	0.4972376
loess	0.000	0.4143646
ppr	0.000	0.3756906
locfit	0.000	0.3342541
gam	0.000	0.4944751
sm	0.000	0.4116022
polymars	0.000	0.4613260
mars	0.000	0.3618785
coxph	2.0135e-12	0.2762431

Tabla 6.25: Valores del test KS MAC-PHY - Scheduler CQA.

- Modelos IP-MAC

  Siguiendo la tendencia las variables delay y jitter se ajustan mejor al modelo proporcionado por la regresión de Cox. Sin embargo, para la variable packet-loss el modelo de loess se aproximan más a sus valores.

MODELO	DELAY		JITTER		PACKET LOSS	
	P-VALUE	D	P-VALUE	D	P-VALUE	D
gausian	0.000	0.7757768	0.000	0.7962571	0.000	0.6151130
loess	0.000	0.7203390	0.000	0.7694209	0.000	0.6002825
ppr	0.000	0.6864407	0.000	0.6878531	0.000	0.6016949
locfit	0.000	0.7722458	0.000	0.7831921	0.000	0.6133475
gam	0.000	0.7757768	0.000	0.7962571	0.000	0.6151130
sm	0.000	0.7665960	0.000	0.7842514	0.000	0.6087571
polymars	0.000	0.7906073	0.000	0.7994350	0.000	0.6151130
mars	0.000	0.7980226	0.000	0.7976695	0.000	0.6140537
coxph	0.000	0.1581921	0.000	0.1762006	0.000	1.0000000

Tabla 6.26: Valores del test KS IP-MAC - Scheduler CQA.

- Función MOS-IP

  El contraste de los valores proporcionado por el test K-S nos indica que existe gran diferencia entre los valores de las variables de interés proporcionados por los modelos y por los datos de flow monitor, de sobremanera en las variables delay y jitter.

	DELAY		JITTER		PACKET LOSS	
	P-VALUE	D	P-VALUE	D	P-VALUE	D
Modelo Flow Monitor	2.2e-16	0.9645	<2.2e-16	0.9262	<2.2e-16	0.7402

Tabla 6.27: Valores del test KS MOS-IP - Scheduler CQA.

Las representaciones gráficas del MOS de este scheduler se observa en las Fig. 6.18 y 6.19. Los datos proporcionados por los mejores modelos se encuentran alrededor de 2.1 (pobre-

aceptable), por otro lado, el MOS con los datos obtenidos por el Flow Monitor tiene una media de 3.5 (aceptable - bueno).

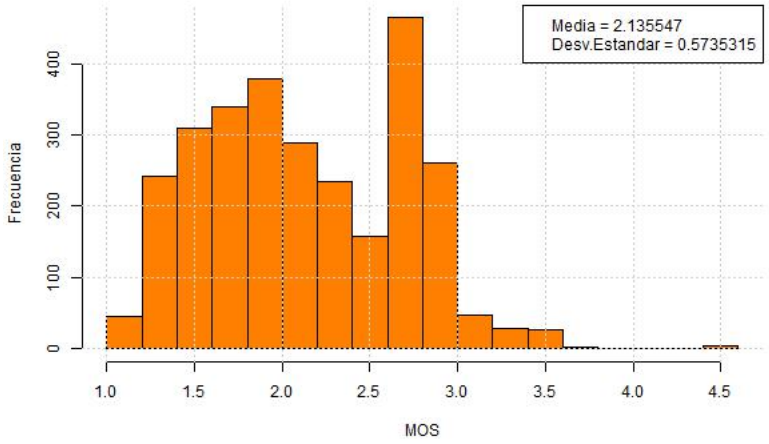

Figura 6.18: MOS (Modelo) - Scheduler CQA.

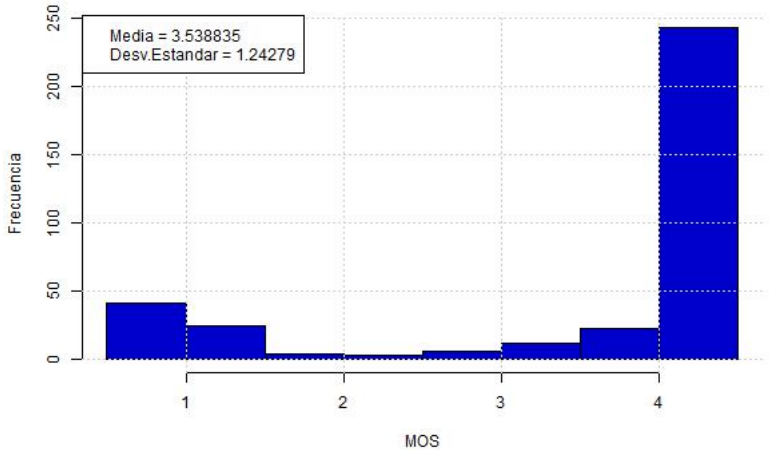

Figura 6.19: MOS (Flow Monitor) - Scheduler CQA.

6.2. Comparación de MOS estimado

A continuación se presenta una tabla resumen del valor del MOS obtenido por los mejor modelos y por Flow Monitor a nivel de aplicación.

SCHEDULER	MOS		
	MODELO	FLOW MONITOR	ERROR %
RR	2.61	3.63	28.09
PF	2.73	3.71	26.41
FD-MT	1.63	3.54	53.95
TD-MT	1.66	3.54	53.10
TTA	1.49	3.05	51.14
FD-TBFQ	1.00	1.21	17.35
TD-TBFQ	0.99	1.18	16.10
PSS	2.55	3.64	29.94
CQA	2.13	3.53	39.66

Tabla 6.28: Resumen valores de MOS.

De acuerdo a los parámetros de ingreso cada scheduler exhibe un valor de MOS.

Analizando los datos presentados, existe una diferencia entre los MOS de los "mejores modelos" y el de "Flow Monitor". El MOS obtenido por los mejores modelos se le interpreta como la calidad que existe en la red interna LTE, este MOS oscila entre el 0, 1 y 2; es decir según la escala acerca de la opinión de calidad de audio estaría entre los niveles pobre y aceptable. Por otro lado, el MOS producido por el Flow Monitor supera los valores anteriores y en ciertos casos llega a un nivel bueno, esta calidad se lo atribuye a la red LTE extremo-extremo incluida en los UE (terminales), los cuales presentan un búfer que permite mejorar la calidad percibida por los usuarios, **pero de mayor variación estándar**.

El scheduler que presenta el mejor MOS tanto en sistema como en los terminales es PF. Tal como dice la literatura es un scheduler que mantiene un balance entre la cantidad de usuarios y el rendimiento de la red, sin embargo es un scheduler no adecuado para redes de VoIP sobre LTE.

Por otro lado, el scheduler PSS es consciente de QoS para flujos que requieren garantizar la velocidad de los datos, como es el caso de VoIP. El MOS obtenido para este scheduler se encuentra entre los niveles aceptable y bueno.

Capítulo 7

Conclusiones y Trabajos Futuros

7.1. Conclusiones

- La calidad de servicio en una red LTE está ligada a la asignación de recursos que el Scheduling de eNB pueda proporcionar a los UE.
- Se pueden aplicar diferentes estrategias de *scheduler* a la red, dependiendo del servicio que se quiera ofrecer, en este caso será VoIP sobre una red LTE.
- El MOS, es un parámetro subjetivo de referencia de la calidad del audio, sin embargo, con la función E-MODEL se puede estimar cuantitativamente el MOS, con los parámetros de delay, jitter y packet loss. En este trabajo se obtiene el MOS estimado del sistema LTE a través de los mejores modelos por nivel de la red LTE. Por otro lado, también se obtuvo el MOS desde el Flow Monitor, en el nivel de aplicación IP del simulador NS-3.
- Se simulan 9 scheduler utilizando scripts en NS-3. Estos schedulers trabajan en el nivel MAC con el fin de proporcionar calidad de servicio a la red LTE.
- Se realiza un ajuste de tráfico entre los niveles PHY y PDCP con la información de la capa IP del Flow Monitor.
- Se agrupan los datos por UE con el identificador IMSI. Este parámetro distintivo de los UE en LTE permite enlazar los datos de tráfico desde la capa PHY hasta el nivel IP.
- La estadística descriptiva de los schedulers muestra que la cantidad de datos en cada una de las variables de interés no es constante, y utilizando funciones estadísticas (de "R"), se agrupan los datos con la misma cantidad de observaciones.
- En general el MOS obtenido por los mejores modelos, siempre es más bajo que el obtenido por Flow Monitor. Esta diferencia se obtiene porque en el primero se está sub-estimando el MOS, mientras que en el segundo, el MOS es mejor debido a que los terminales mantienen un búfer mayor que en comparación a los eNB.
- Entre los schedulers, el de mejor MOS tanto a nivel de aplicación como a nivel de red es el Proportional Fair (PF), sin embargo PF es un scheduler que no es consciente de la calidad de servicio. A nivel de aplicación el segundo mejor scheduler es Priority Set Scheduler (PSS) y lo sigue Round Robin (RR). Cabe recordar que PSS es una combinación de PF y FBET; además este scheduler puede garantizar un data-rate predeterminado, mientras que RR pertenece

a la familia de schedulers con una estrategia de canal inconsciente.

- El scheduler PSS tiene un valor de estimación del MOS aceptable (3.64) a nivel de aplicación, sin embargo a nivel del sistema el MOS estimado baja a 2.55, un nivel no aceptable [16].
- Los valores de error indican la diferencia que existen entre el MOS del sistema y el MOS de aplicación, por lo tanto las estrategias MT con sus variantes TD y FD presentan mayor error muestral, a diferencia de las estrategias TBFQ las cuales presenta un error menor al 18 % pero su valor de MOS estimado en ambos casos (sistema y aplicación) es bajo de un valor cercano a uno.
- Con los valores del MOS para cada Scheduler a nivel de sistema como de aplicación se cumple el objetivo planteado en esta tesis. Se verifica que el MOS a nivel de aplicación es aceptable a comparación del MOS de la red. Sin embargo para mejorar la calidad del audio de las aplicaciones VoIP existen más elementos a revisar, como por ejemplo los códecs de audio, optimización de la RAN, y en general los métodos RRM.

7.2. Trabajos Futuros

Entre las varias opciones de continuidad de este trabajo, se proponen algunos ejemplos de trabajos futuros:

- Modelar el MOS de los schedulers LTE que garantizan los requerimientos de retardo como M-LWDF, EXP/PF, LOG-RULE, EXP-RULE y los schedulers conscientes de energía.
- Analizar y estimar un MOS para servicio de streaming y multicast.
- Utilizar otras técnicas de machine learning para la modelación del MOS, por ejemplo: series de tiempo, clustering, técnicas adaptivas (por ejemplo redes neuronales).
- Realizar un prototipo de scheduler de los eNB con los métodos utilizados en este trabajo.

Bibliografía

[1] R. A. COMES and F. VODAFONE, *LTE: Nuevas tendencias en comunicaciones móviles*. Fundación Vodafone, 2010.

[2] A. KHREIS and A. E. SAMHAT, "Scheduling in ofdma based systems," Master's thesis, Lebanese University, 2014.

[3] A. EHIJO and J. GONZÁLEZ, "Introducción a volte," Universidad de Chile, Santiago, Chile, Tech. Rep., 2013.

[4] I. F. AKYILDIZ, D. M. GUTIERREZ-ESTEVEZ, and E. C. REYES, "The evolution to 4g cellular systems: Lte-advanced," *Physical Communication*, vol. 3, no. 4, pp. 217–244, 2010.

[5] A. CASTRO, "Redes lte," Universidad de Chile, Santiago, Chile, Apunte Curso Modelamiento y Simulación de Redes de Comunicaciones, Semestre Primavera 2015.

[6] M. DO, "Lte: User identifiers - imsi and guti," http://www.netmanias.com/en/post/blog/5929/guti-imsi-lte/lte-user-identifiers-imsi-and-guti, 2013.

[7] T. POINTS, "Lte quick guide," http://www.tutorialspoint.com/lte/lte_quick_guide.htm, 1999.

[8] M. L. BELHOUCHET and M. H. EBDELLI, "Lte technology performance evaluation," in *ITU/BDT Arab regional workshop on 4G wireless systems*, 2010.

[9] F. J. VALERA SANCHEZ, "Ofdma y sc-fdma en la interfaz radio de lte," Master's thesis, Universidad de Sevilla, 2012.

[10] N. SHOJAEDIN, M. GHADERI, and A. SRIDHARAN, "Tcp-aware scheduling in lte networks," in *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2014 IEEE 15th International Symposium on a*. IEEE, 2014, pp. 1–9.

[11] F. AFROZ, K. SANDRASEGARAN, and P. GHOSAL, "Performance analysis of pf, m-lwdf and exp/pf packet scheduling algorithms in 3gpp lte downlink," in *Telecommunication Networks and Applications Conference (ATNAC), 2014 Australasian*. IEEE, 2014, pp. 87–92.

[12] B. LIU, H. TIAN, and L. XU, "An efficient downlink packet scheduling algorithm for real time traffics in lte systems," in *Consumer Communications and Networking conference (CCNC), 2013 IEEE*. IEEE, 2013, pp. 364–369.

[13] F. CAPOZZI, G. PIRO, L. A. GRIECO, G. BOGGIA, and P. CAMARDA, "Downlink packet scheduling in lte cellular networks: Key design issues and a survey," *Communications Surveys & Tutorials, IEEE*, vol. 15, no. 2, pp. 678–700, 2013.

[14] WIKIPEDIA, "Mean opinion score," https://en.wikipedia.org/wiki/Mean_opinion_score, 2016.

[15] A. F. RIBADENEIRA, "An analysis of the mos under conditions of delay, jitter and packet loss and an analysis of the impact of introducing piggybacking and reed solomon fec for voip," *Georgia State University ScholarWorks*, 2007.

[16] J. JOSKOWICZ and R. SOTELO, "Medida de la calidad de voz en redes ip," *Memoria de Trabajos de Difusión Científica y Técnica*, no. 5, pp. 12–23, 2007.

[17] R. G. ITU-T and G. Recommendation, "107: The e-model: a computational model for use in transmission planning," *Recommendation ITU-T*, 2011.

[18] C. OLARIU, "Quality of service support for voice over ip in wireless access networks," Ph.D. dissertation, Waterford Institute of Technology, 2013.

[19] R. MUGISHA and N. VENTURA, "Packet scheduling for voip over lte-a," in *AFRICON, 2013*. IEEE, 2013, pp. 1–6.

[20] D. ZHOU, "Gsoc lte mac scheduler: Final review," http://mailman.isi.edu/pipermail/ns-developers/2012-August/010595.html, 2012.

[21] D. ZHOU, N. BALDO, and M. MIOZZO, "Implementation and validation of lte downlink schedulers for ns-3," in *Proceedings of the 6th International ICST Conference on Simulation Tools and Techniques*. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2013, pp. 211–218.

[22] B. BOJOVIC and N. BALDO, "A new channel and qos aware scheduler to enhance the capacity of voice over lte systems," in *Systems, Signals & Devices (SSD), 2014 11th International Multi-Conference on*. IEEE, 2014, pp. 1–6.

[23] J. CHAVEZ, "Reporte final de práctica profesional del simulador para el servicio de voz sobre lte (volte):guia de uso," Universidad de Chile, Santiago, Chile, Tech. Rep., 2015.

[24] C. T. D. T. D. C. (CTTC), "Lena," http://networks.cttc.es/mobile-networks/software-tools/lena/, 2016.

[25] C. T. D. T. D. C. (CTTC), "The lena ns-3 lte module documentation release v8," *CTTC*, January 21, 2014.

[26] NS-3, *NS-3 User Documentation*, https://www.nsnam.org/docs/models/html/lte-user.html.

[27] NS-3, *Flow Monitor*, https://www.nsnam.org/docs/models/html/flow-monitor.html.

[28] E. DAHLMAN, S. PARKVALL, and J. SKOLD, *4G: LTE/LTE-advanced for mobile broad-*

band. Academic press, 2013.

[29] I.-T. (STUDYGROUPS), "E-model tutorial," https://www.itu.int/ITU-T/studygroups/com12/emodelv1/tut.htm, 2008.

[30] M. BONILLA, I. OLMEDA, and R. PUERTAS, "Modelos paramétricos y no paramétricos en problemas de credit scoring," *Spanish Journal of Finance and Accounting/Revista Española de Financiación y Contabilidad*, vol. 32, no. 118, pp. 833–869, 2003.

[31] D. FERRARI, "Introduction to regression in r. part ii:multivariate linear regression," 2009.

[32] L. CAYUELA, "Modelos lineales generalizados (glm)," *Materiales de un curso del R del IREC*, 2009.

[33] J. FOX and S. WEISBERG, "Nonparametric regression in r," in *An R companion to applied regression*. Sage, 2010.

[34] P. DELICADO, "Curso de modelos no paramétricos," *Departament d'Estadística i Investigació Operativa, Universitad Politécnica de Catalunya*, 2008.

[35] J. FOX, "Nonparametric-regression resources in r," http://socserv.socsci.mcmaster.ca/jfox/Courses/Oxford-2005/R-nonparametric-regression.html, 2005.

[36] N. BOUKICHOU, "Regresión no paramétrica," Master's thesis, Universidad de Granada, 2009.

[37] R. DOCUMENTARION, "Local polynomial regression fitting," https://stat.ethz.ch/R-manual/R-devel/library/stats/html/loess.html.

[38] C. LOADER, "Package 'locfit'," https://cran.r-project.org/web/packages/locfit/locfit.pdf.

[39] R. DOCUMENTARION, "Projection pursuit regression," https://stat.ethz.ch/R-manual/R-devel/library/stats/html/ppr.html.

[40] T. HASTIE, "Package 'gam'," https://cran.r-project.org/web/packages/gam/gam.pdf.

[41] A. BOWMAN and A. AZZALINI, "Package 'sm'," https://cran.r-project.org/web/packages/sm/sm.pdf.

[42] M. I. GODOY, "Métodos estadísticos i:regresión no paramétrica," 2016.

[43] T. HASTIE, "Package 'mda'," https://cran.r-project.org/web/packages/mda/mda.pdf.

[44] C. KOOPERBERG, "Package 'polspline'," https://cran.r-project.org/web/packages/polspline/polspline.pdf.

[45] J. FOX, "Cox proportional-hazards regression for survival data," in *An R and S-PLUS Companion to Applied Regression*, 2008.

[46] C. LOADER, "Locfit: An introduction," 1997.

[47] WIKIPEDIA, “Projection pursuit regression,” https://en.wikipedia.org/wiki/Projection_pursuit_regression, 2016.

[48] F. MALLO FERNÁNDEZ, “Modelos multivariantes internos de riesgo de crédito, acordes con basilea, ii,” 2011.

[49] K. LATíNEZ, “Comparación de los métodos de regresión multivariada adaptativa utilizando splines (mars) y redes neuronales artificiales backpropagation (rnab) para el pronóstico de lluvias y temperaturas en la cuenca del río mantaro (pregrado), universidad nacional agraria la molina, perú.” 2009.

[50] R. DOCUMENTARION, “Kolmogorov-smirnov tests,” https://stat.ethz.ch/R-manual/R-devel/library/stats/html/ks.test.html.

Anexos

Anexo A

Gráficos y Tablas

- Scheduler RR

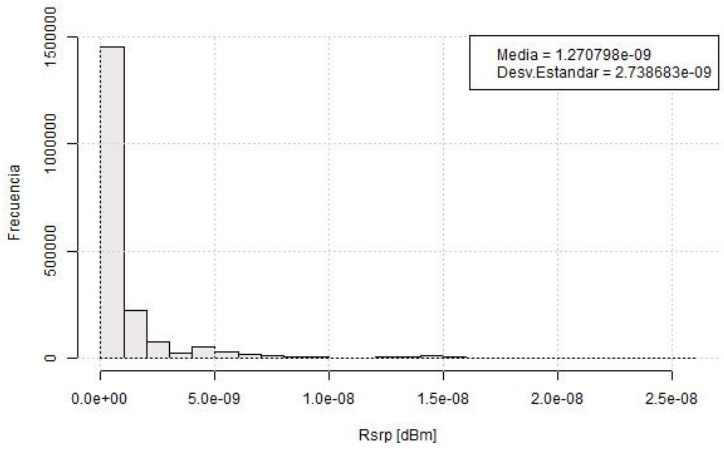

Figura A.1: RSRP - Scheduler RR

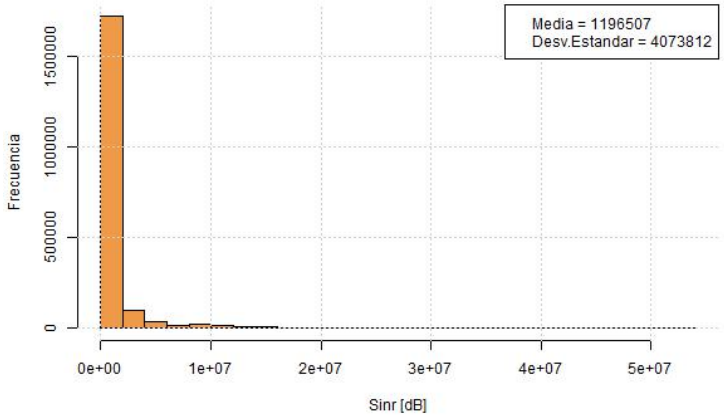

Figura A.2: SINR - Scheduler RR

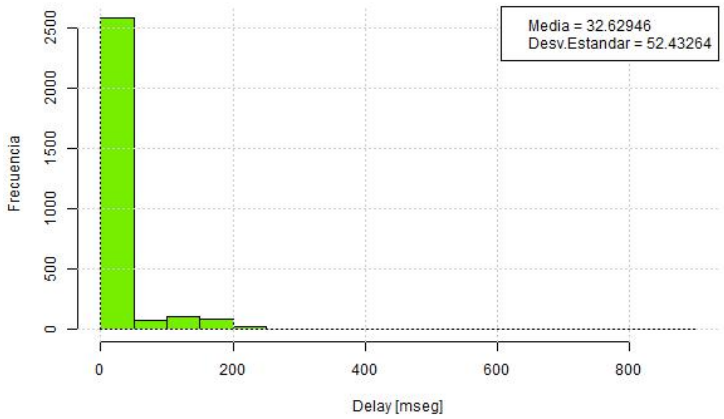

Figura A.3: DELAY (PDCP) - Scheduler RR

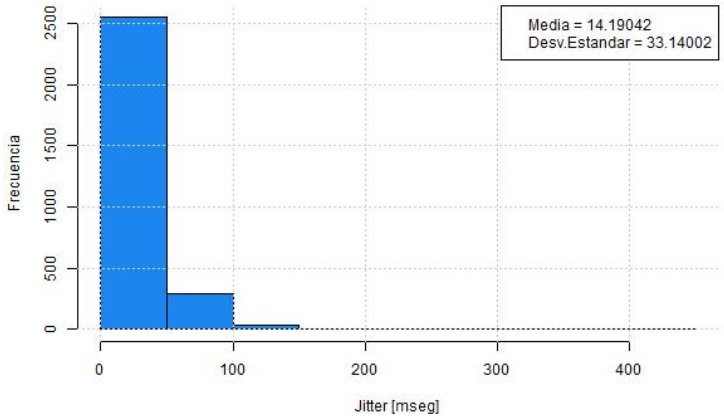

Figura A.4: JITTER (PDCP) - Scheduler RR

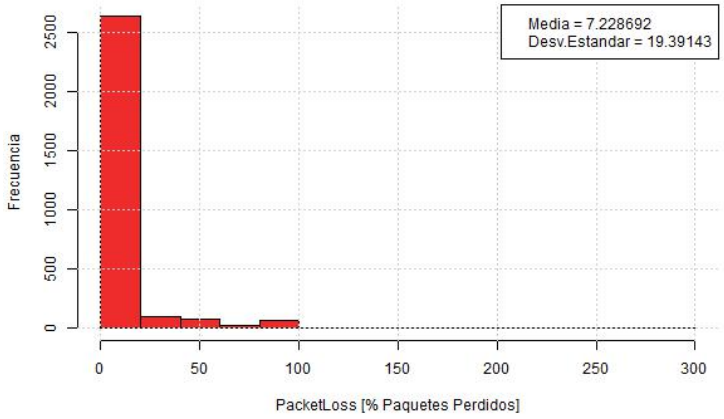

Figura A.5: PACKET LOSS (PDCP) - Scheduler RR

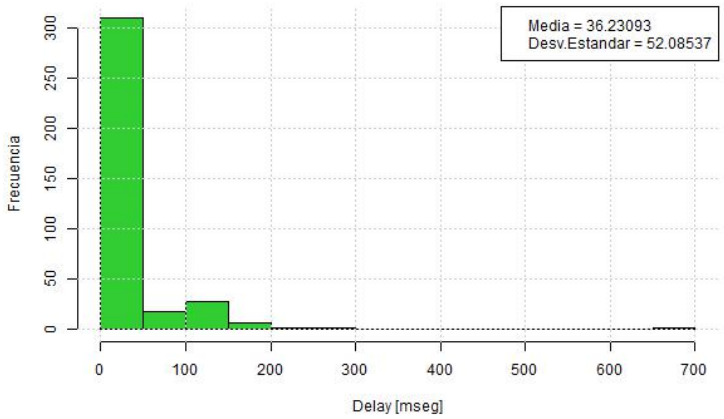

Figura A.6: DELAY (FLOW MONITOR) - Scheduler RR

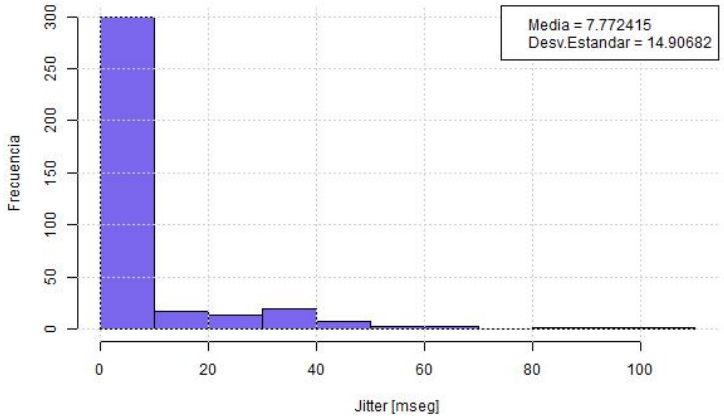

Figura A.7: JITTER (FLOW MONITOR) - Scheduler RR

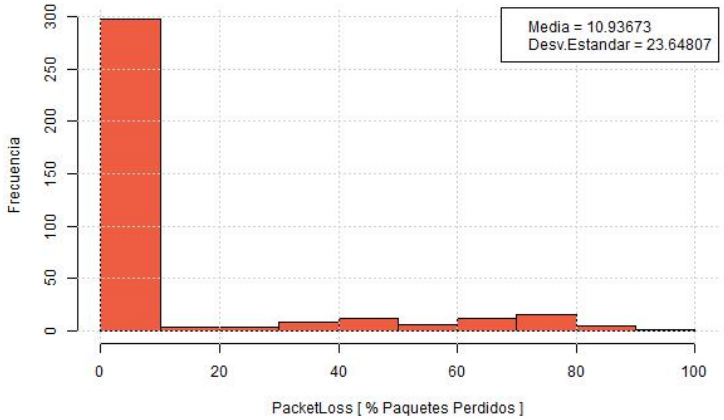

Figura A.8: PACKET LOSS (FLOW MONITOR) - Scheduler RR

- Scheduler PF

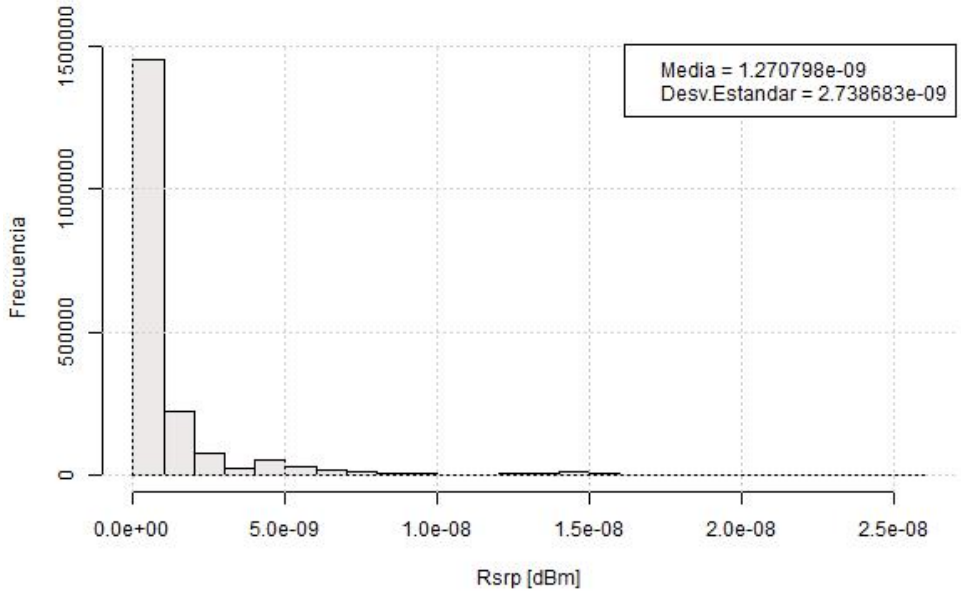

Figura A.9: RSRP - Scheduler PF

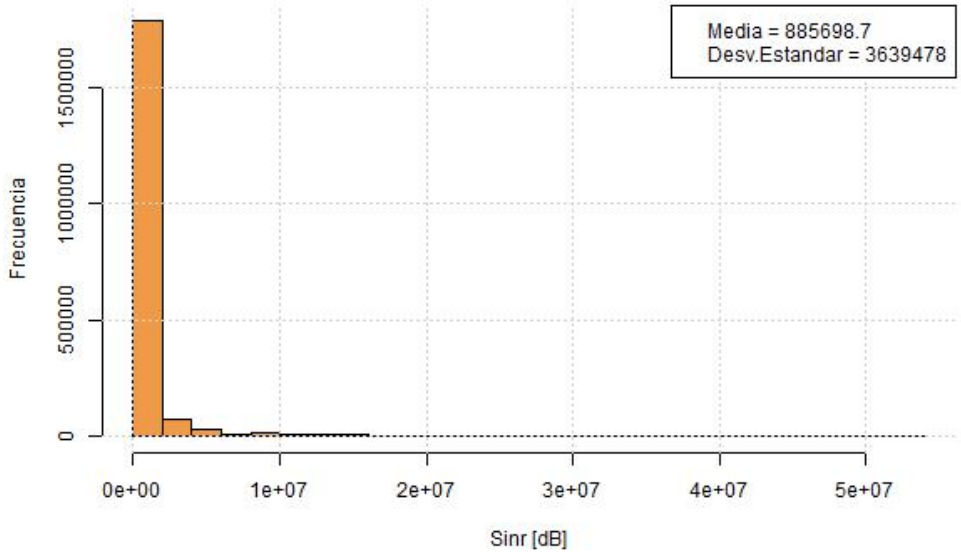

Figura A.10: SINR - Scheduler PF

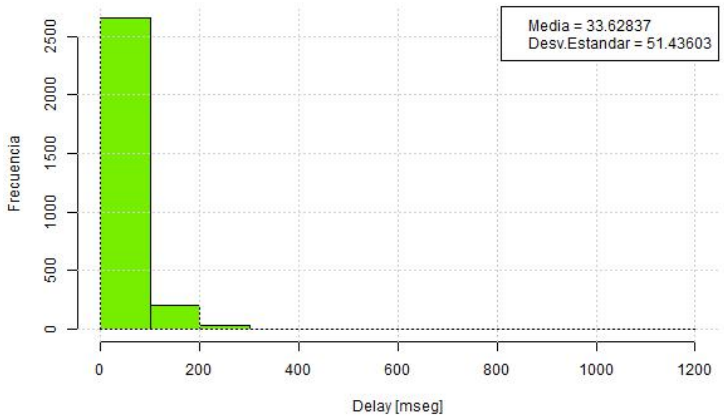

Figura A.11: DELAY (PDCP) - Scheduler PF

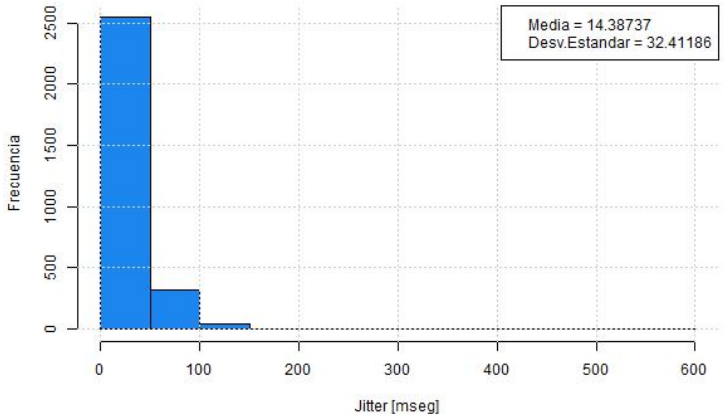

Figura A.12: JITTER (PDCP) - Scheduler PF

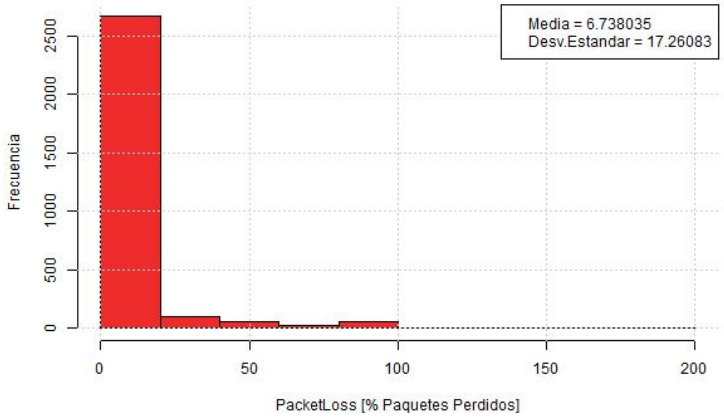

Figura A.13: PACKET LOSS (PDCP) - Scheduler PF

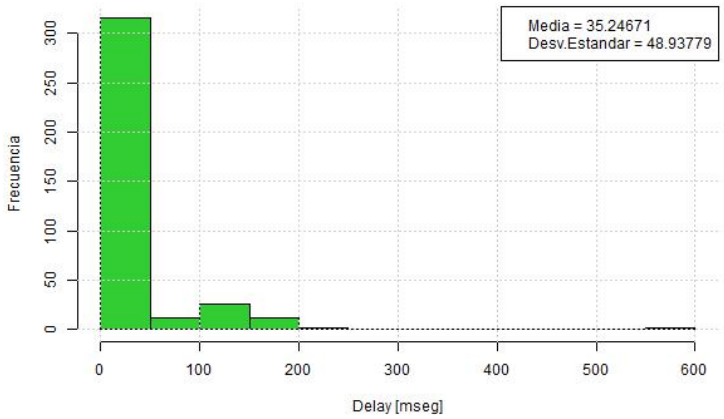

Figura A.14: DELAY (FLOW MONITOR) - Scheduler PF

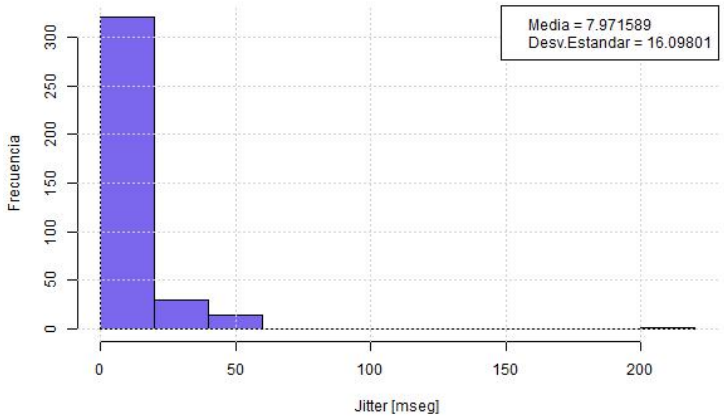

Figura A.15: JITTER (FLOW MONITOR) - Scheduler PF

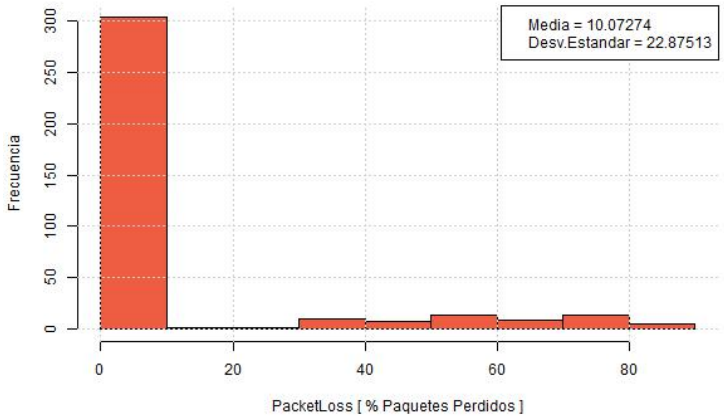

Figura A.16: PACKET LOSS (FLOW MONITOR) - Scheduler PF

- Scheduler FD-MT

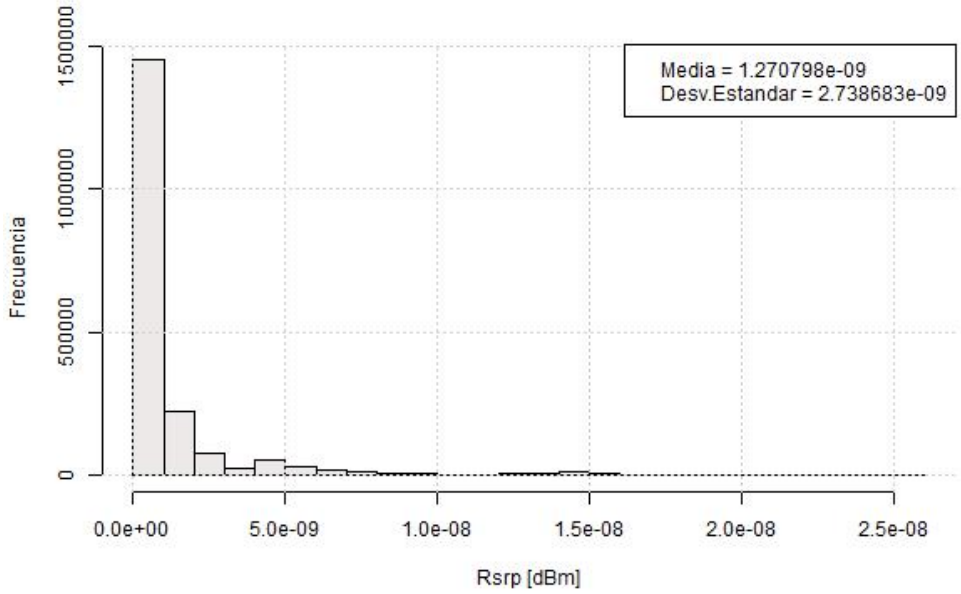

Figura A.17: RSRP - Scheduler FD-MT

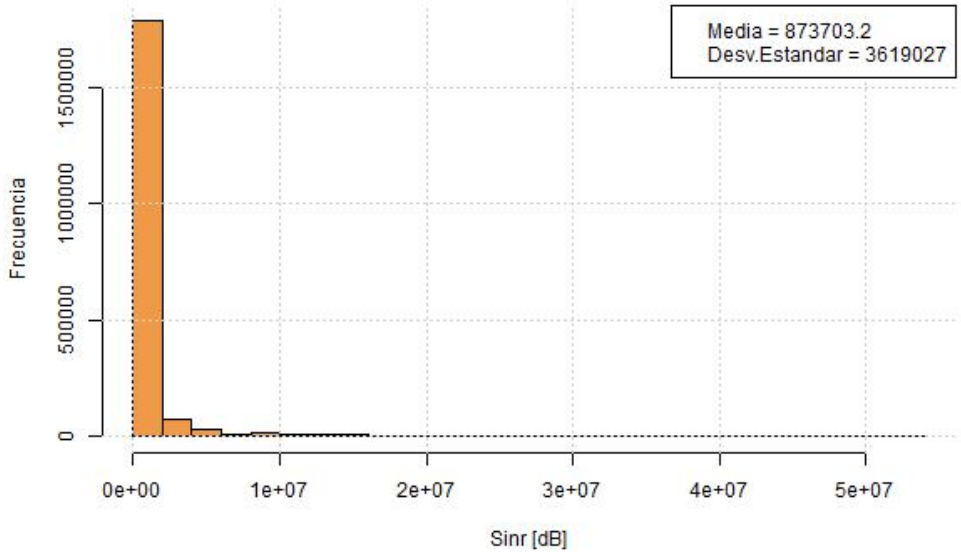

Figura A.18: SINR - Scheduler FD-MT

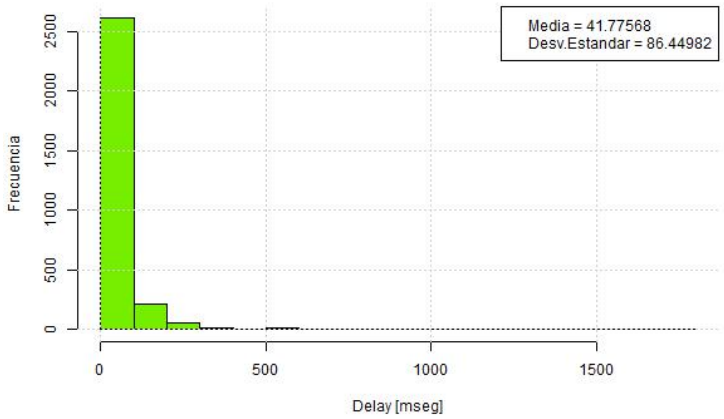

Figura A.19: DELAY (PDCP) - Scheduler FD-MT

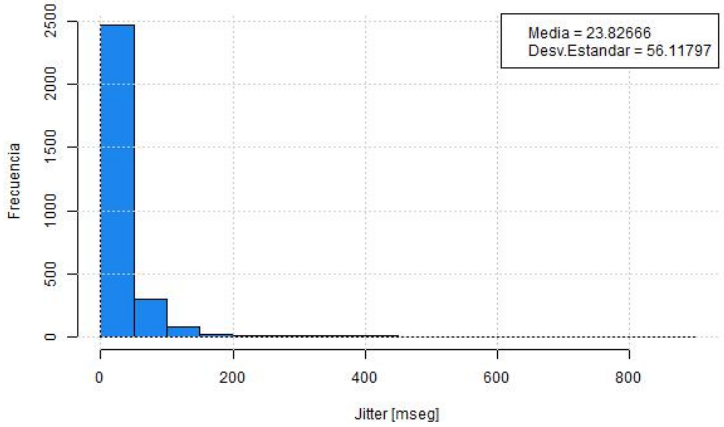

Figura A.20: JITTER (PDCP) - Scheduler FD-MT

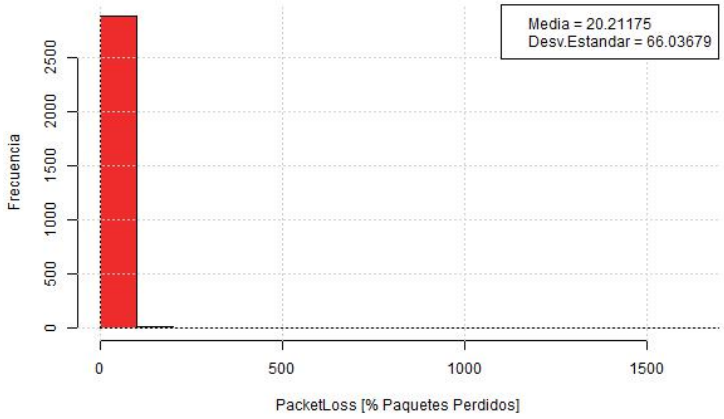

Figura A.21: PACKET LOSS (PDCP) - Scheduler FD-MT

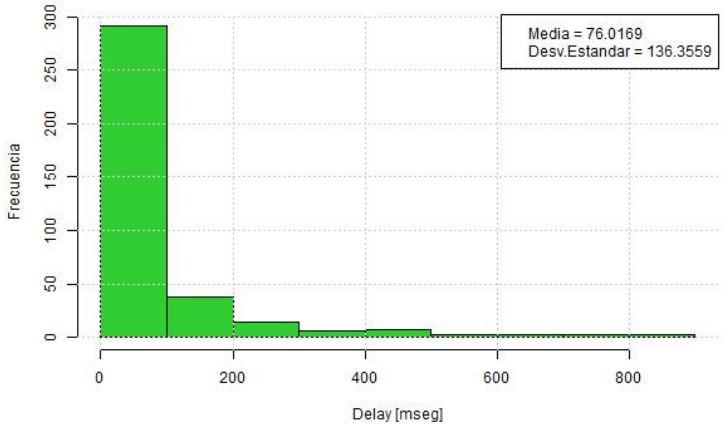

Figura A.22: DELAY (FLOW MONITOR) - Scheduler FD-MT

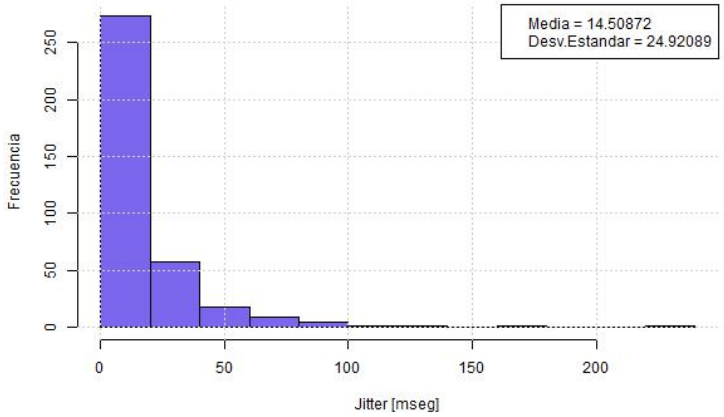

Figura A.23: JITTER (FLOW MONITOR) - Scheduler FD-MT

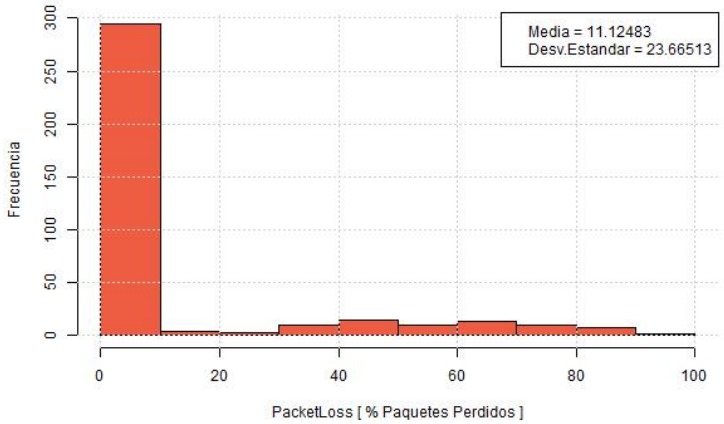

Figura A.24: PACKET LOSS (FLOW MONITOR) - Scheduler FD-MT

- Scheduler TD-MT

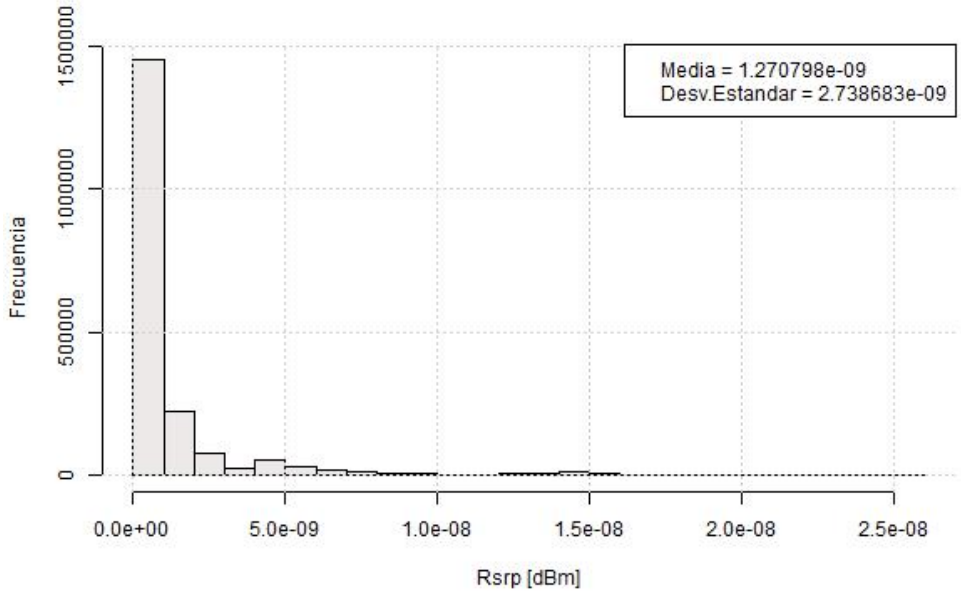

Figura A.25: RSRP - Scheduler TD-MT

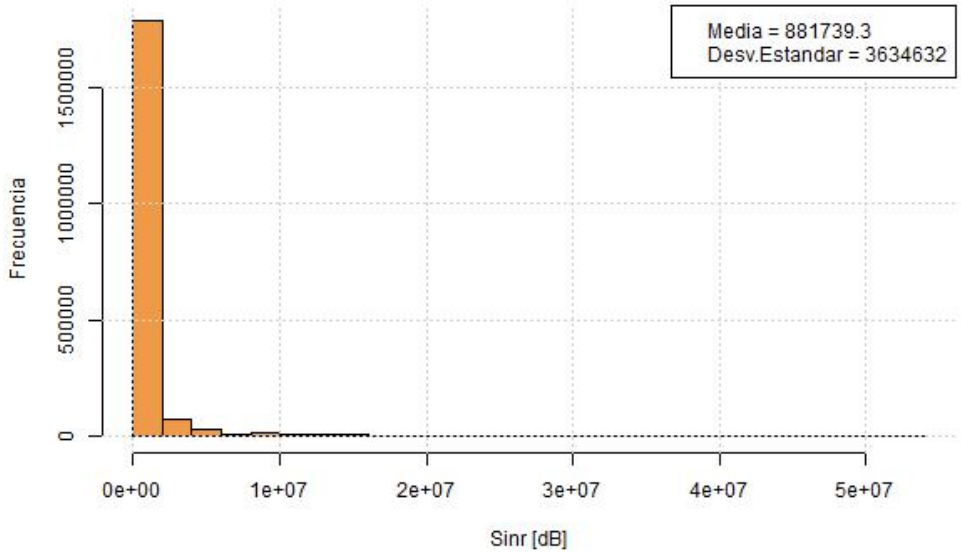

Figura A.26: SINR - Scheduler TD-MT

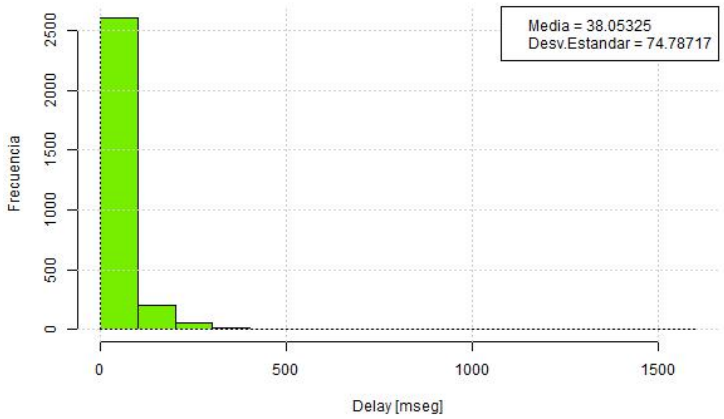

Figura A.27: DELAY (PDCP) - Scheduler TD-MT

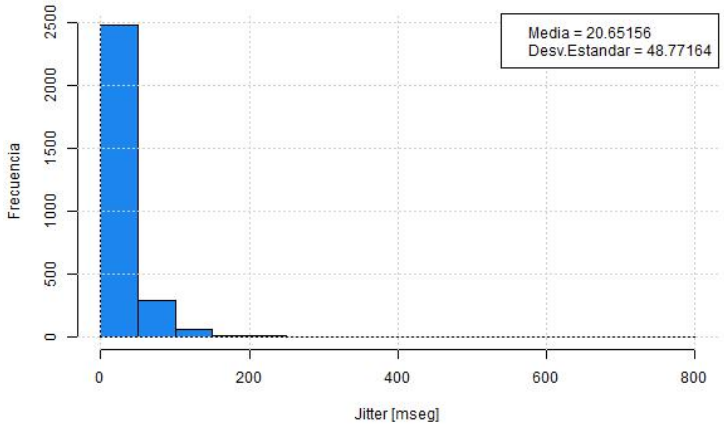

Figura A.28: JITTER (PDCP) - Scheduler TD-MT

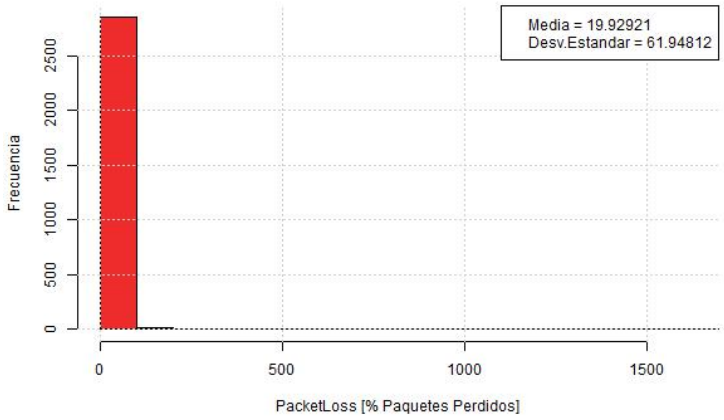

Figura A.29: PACKET LOSS (PDCP) - Scheduler TD-MT

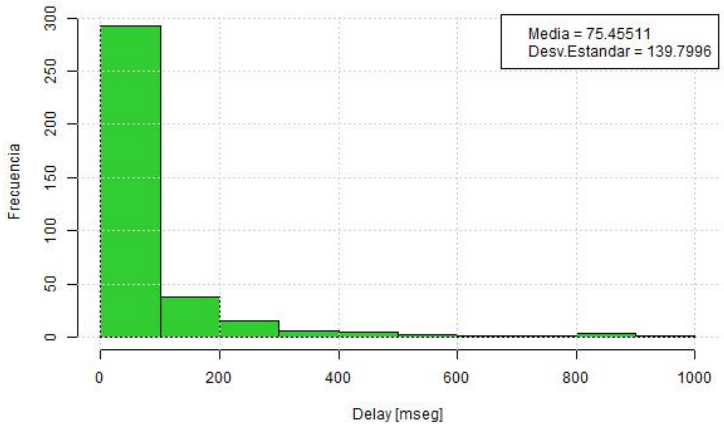

Figura A.30: DELAY (FLOW MONITOR) - Scheduler TD-MT

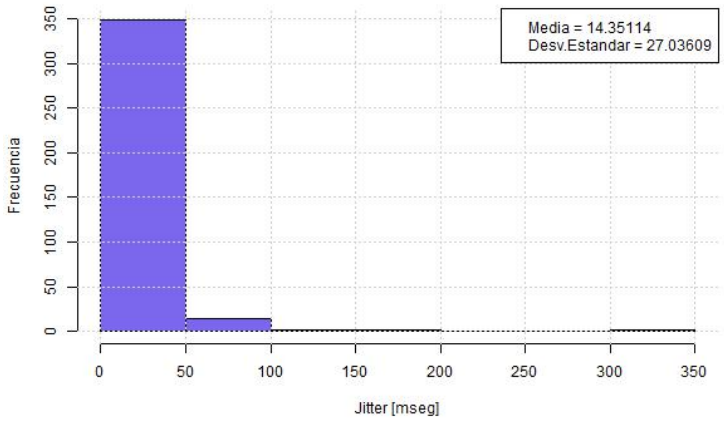

Figura A.31: JITTER (FLOW MONITOR) - Scheduler TD-MT

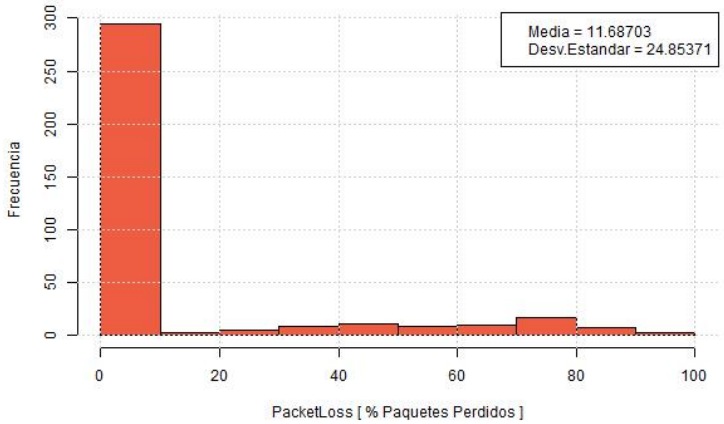

Figura A.32: PACKET LOSS (FLOW MONITOR) - Scheduler TD-MT

- Scheduler TTA

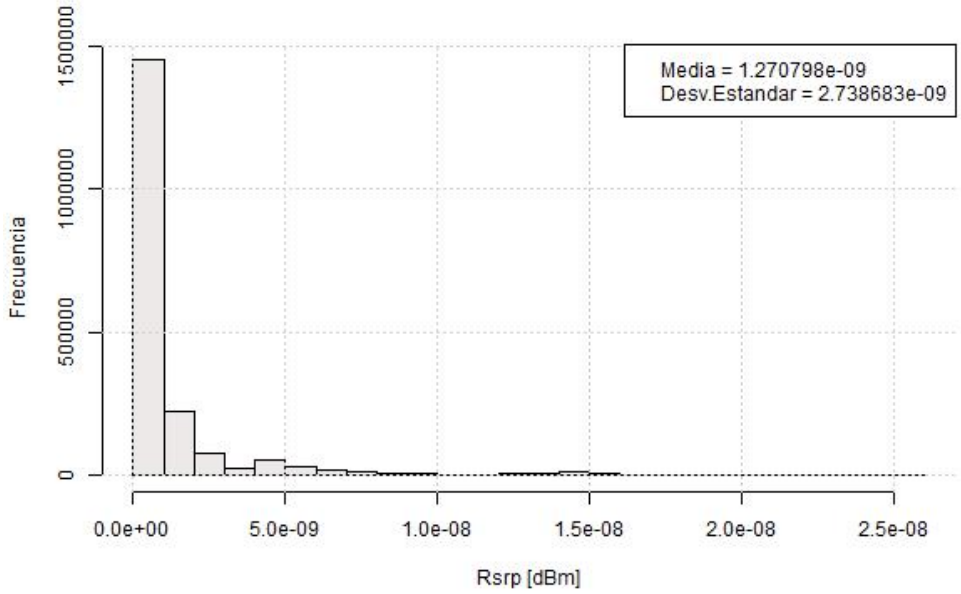

Figura A.33: RSRP - Scheduler TTA

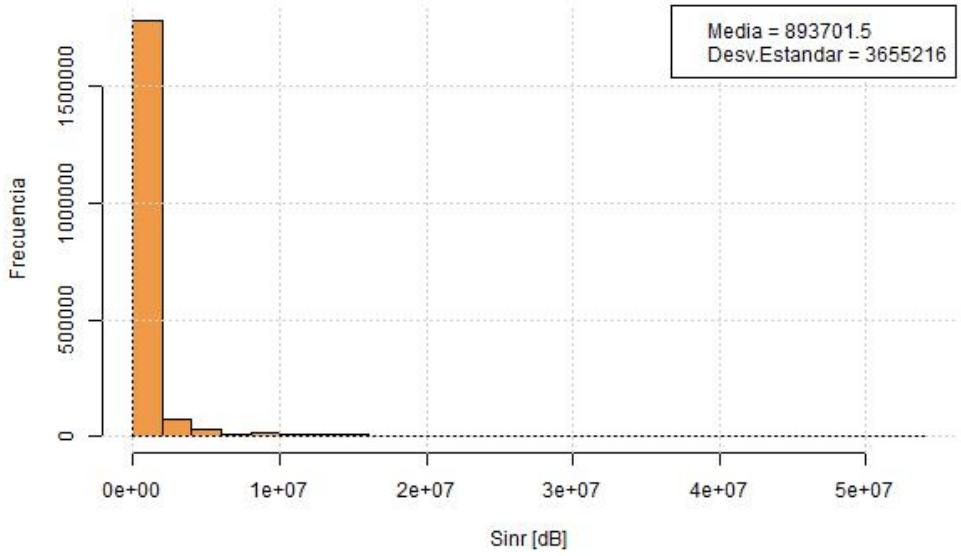

Figura A.34: SINR - Scheduler TTA

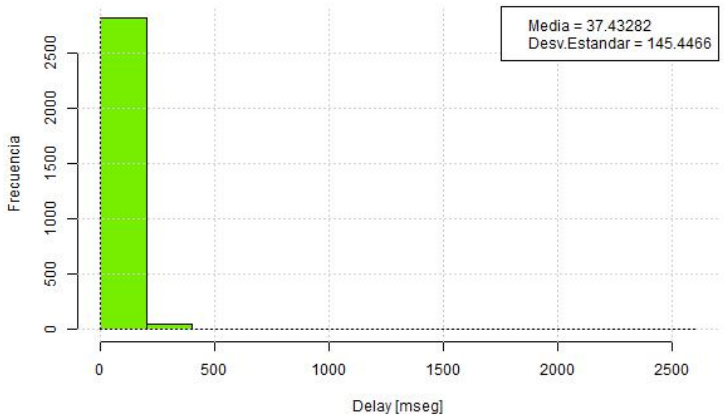

Figura A.35: DELAY (PDCP) - Scheduler TTA

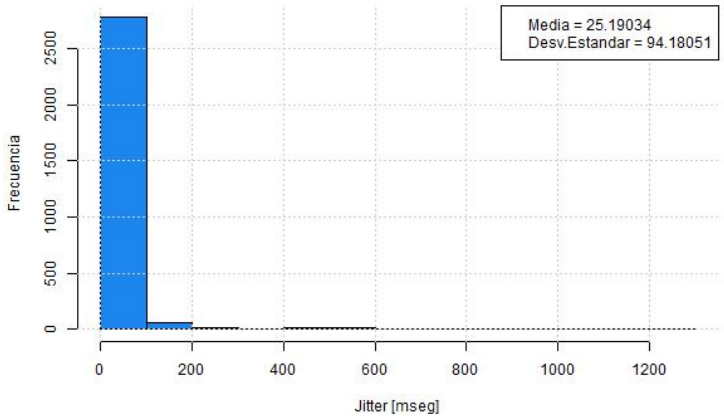

Figura A.36: JITTER (PDCP) - Scheduler TTA

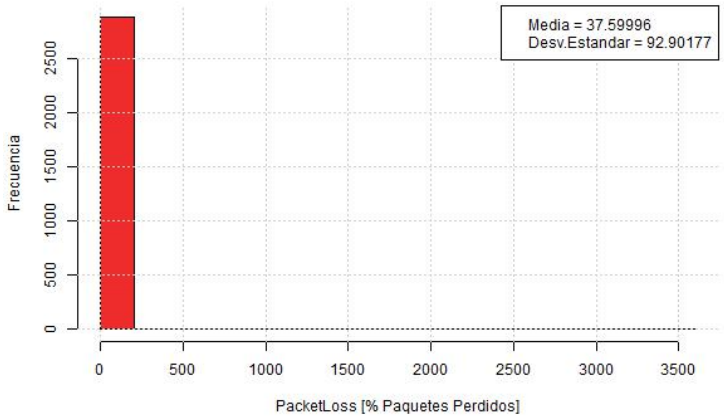

Figura A.37: PACKET LOSS (PDCP) - Scheduler TTA

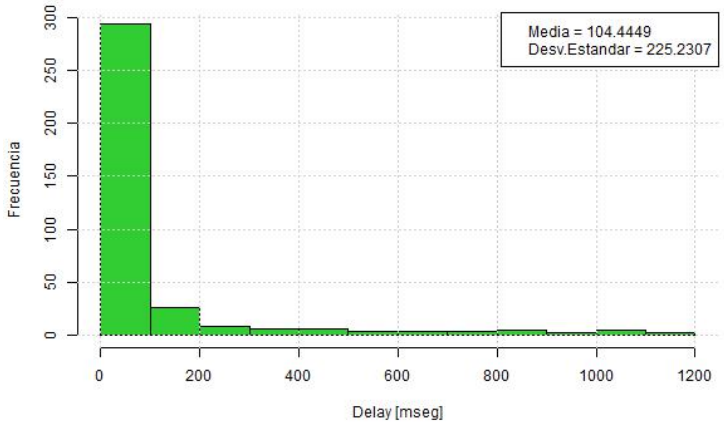

Figura A.38: DELAY (FLOW MONITOR) - Scheduler TTA

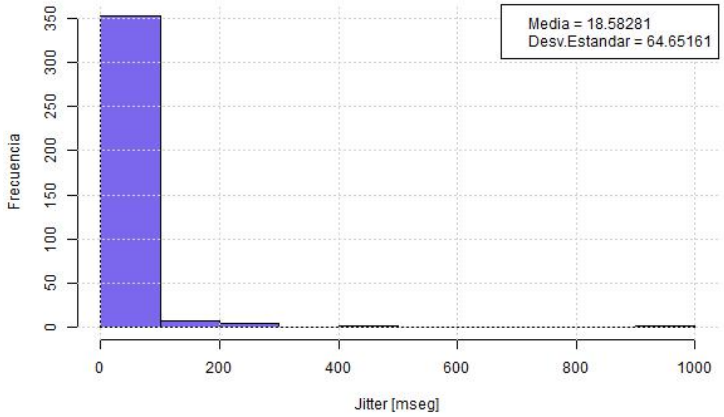

Figura A.39: JITTER (FLOW MONITOR) - Scheduler TTA

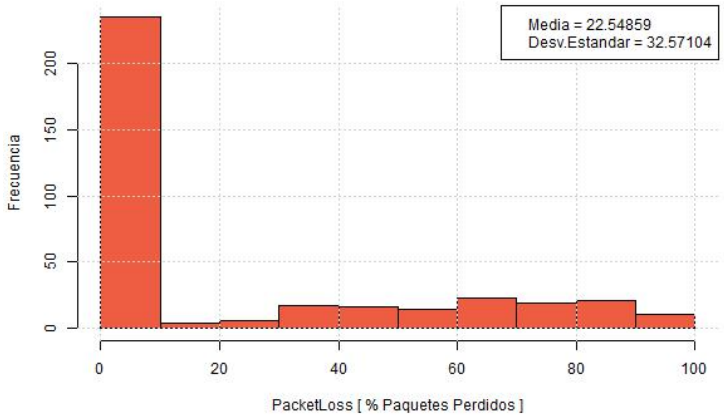

Figura A.40: PACKET LOSS (FLOW MONITOR) - Scheduler TTA

- Scheduler FD-TBFQ

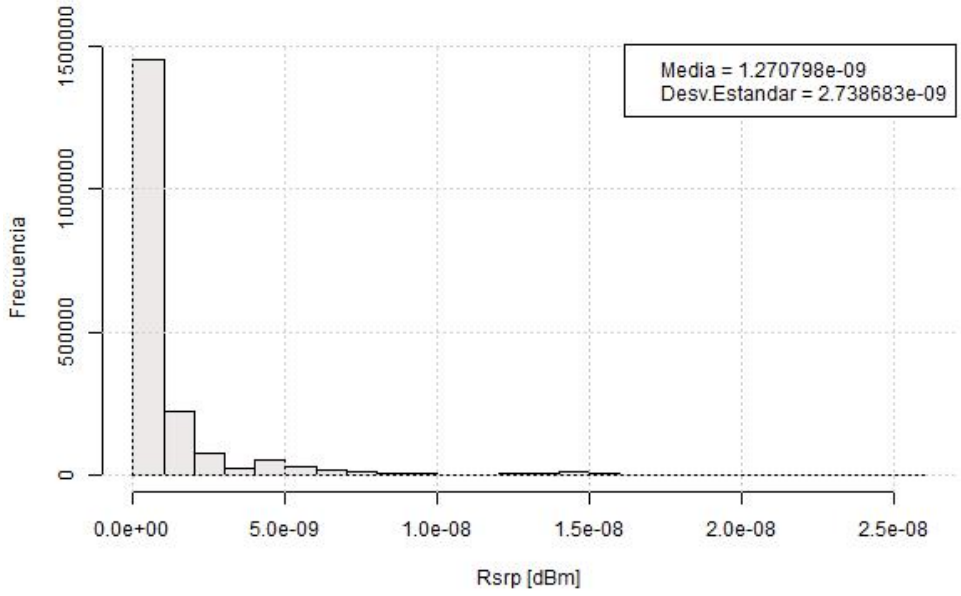

Figura A.41: RSRP - Scheduler FD-TBFQ

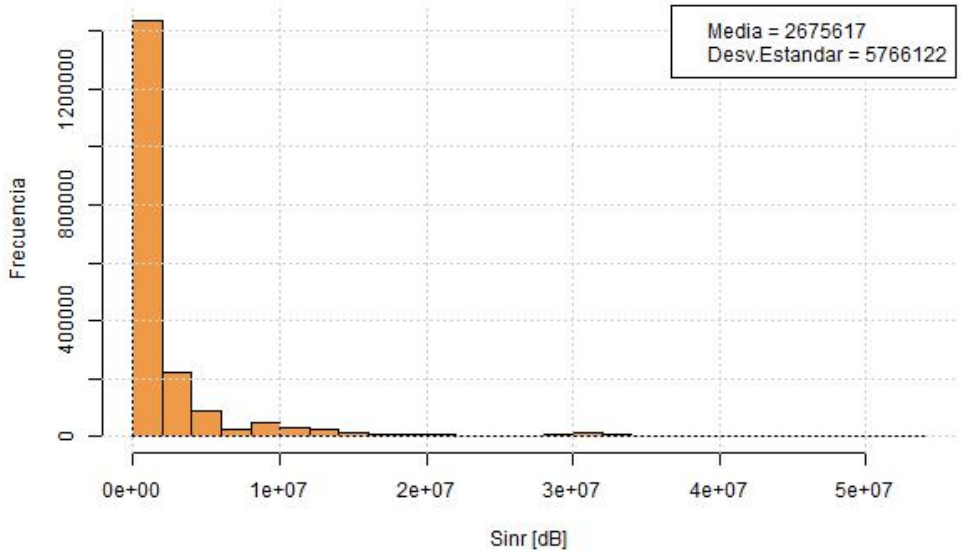

Figura A.42: SINR - Scheduler FD-TBFQ

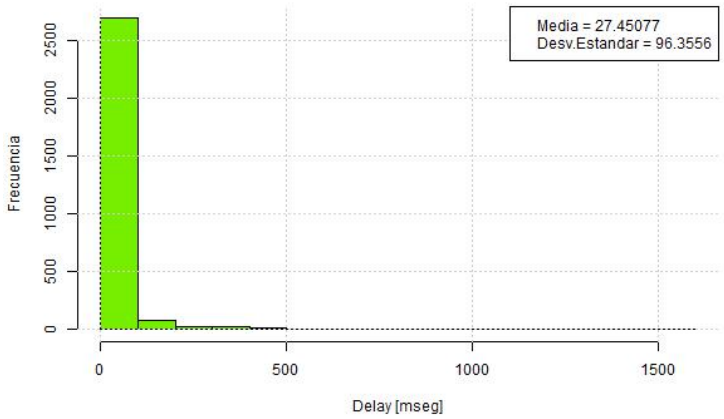

Figura A.43: DELAY (PDCP) - Scheduler FD-TBFQ

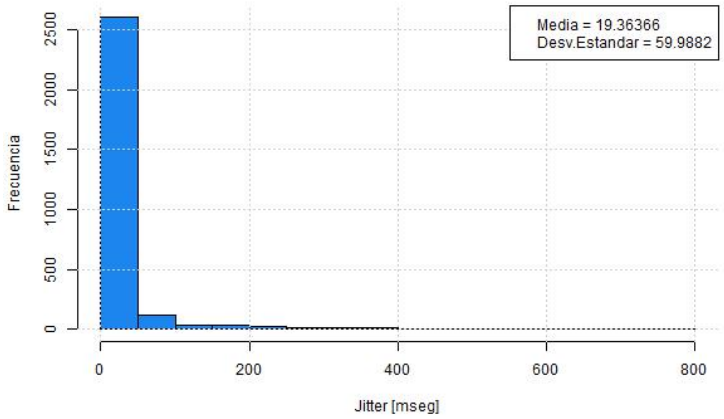

Figura A.44: JITTER (PDCP) - Scheduler FD-TBFQ

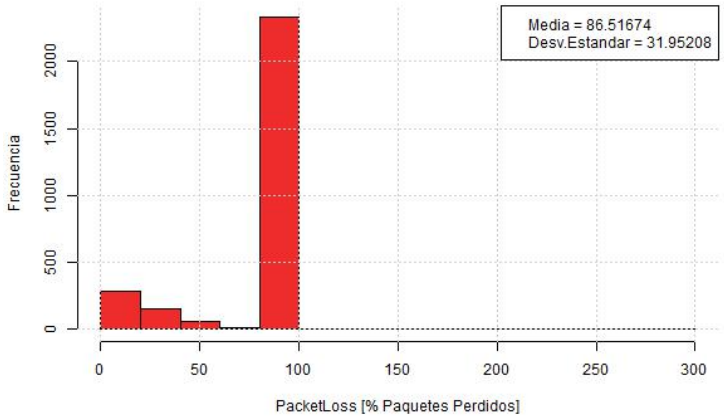

Figura A.45: PACKET LOSS (PDCP) - Scheduler FD-TBFQ

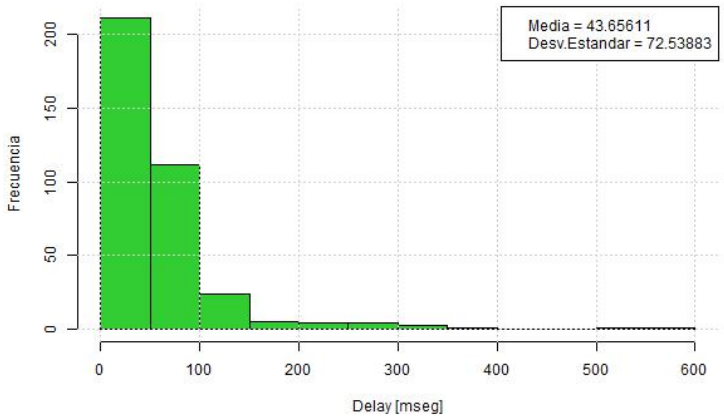

Figura A.46: DELAY (FLOW MONITOR) - Scheduler FD-TBFQ

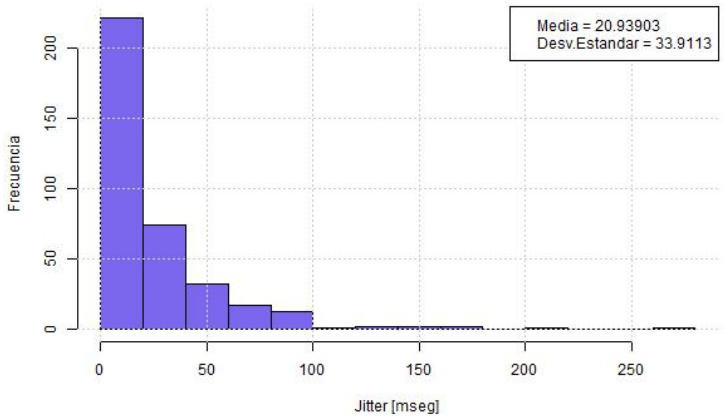

Figura A.47: JITTER (FLOW MONITOR) - Scheduler FD-TBFQ

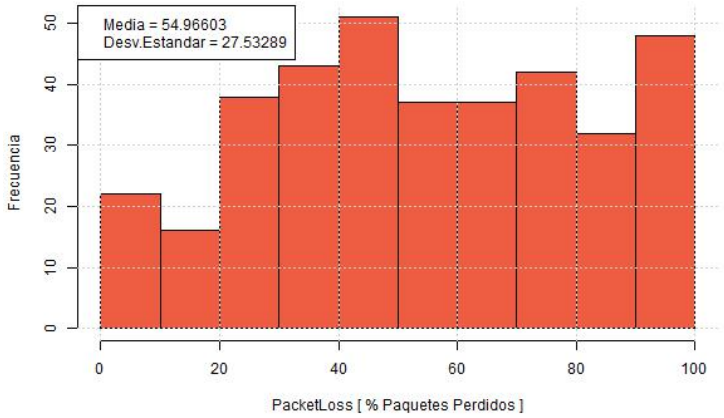

Figura A.48: PACKET LOSS (FLOW MONITOR) - Scheduler FD-TBFQ

- Scheduler TD-TBFQ

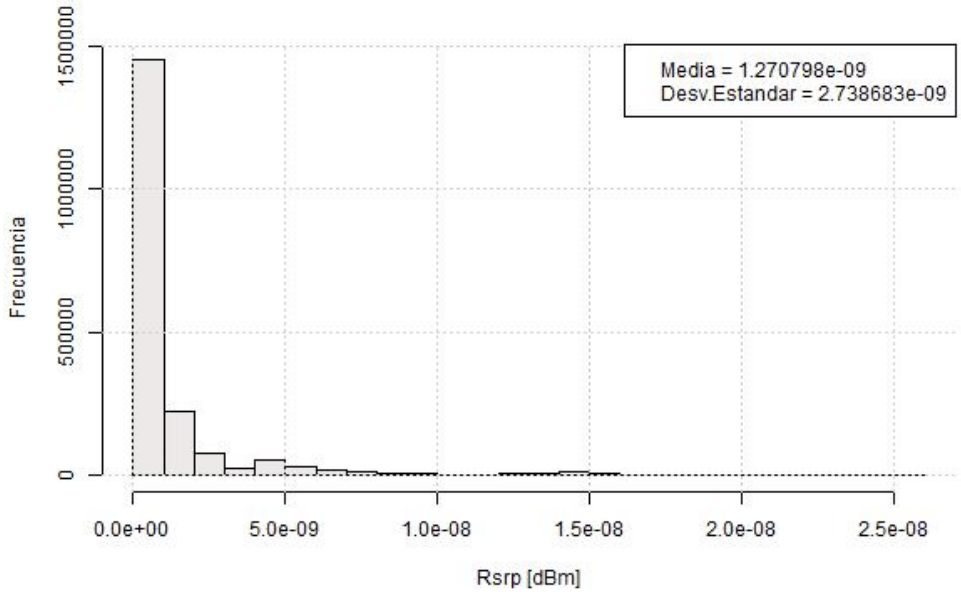

Figura A.49: RSRP - Scheduler TD-TBFQ

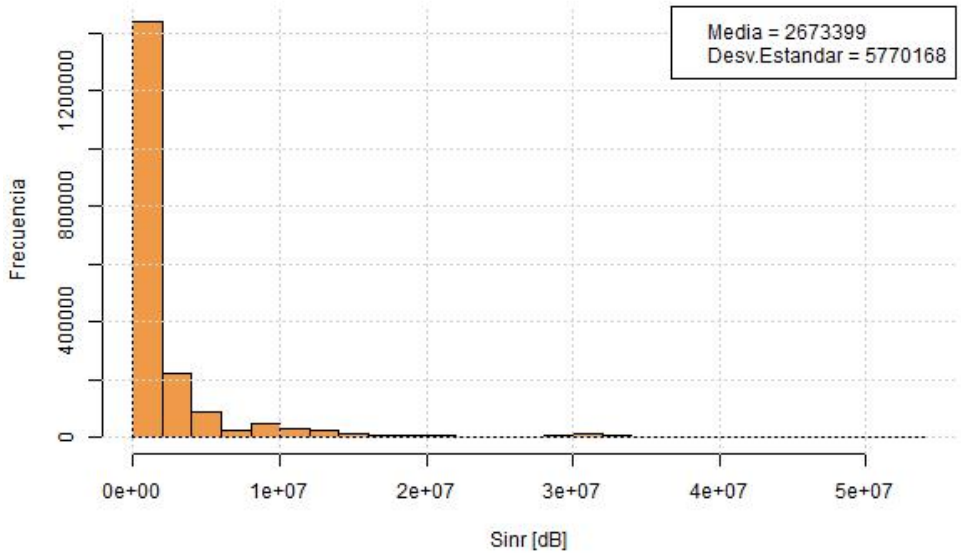

Figura A.50: SINR - Scheduler TD-TBFQ

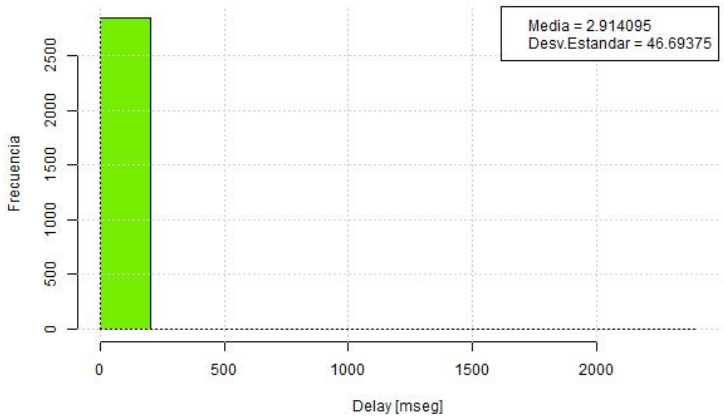

Figura A.51: DELAY (PDCP) - Scheduler TD-TBFQ

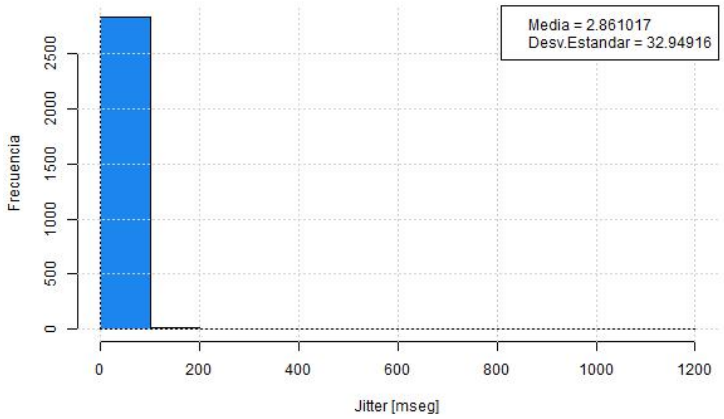

Figura A.52: JITTER (PDCP) - Scheduler TD-TBFQ

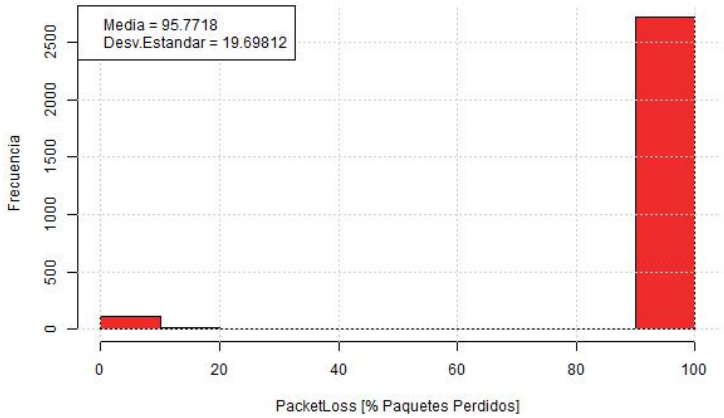

Figura A.53: PACKET LOSS (PDCP) - Scheduler TD-TBFQ

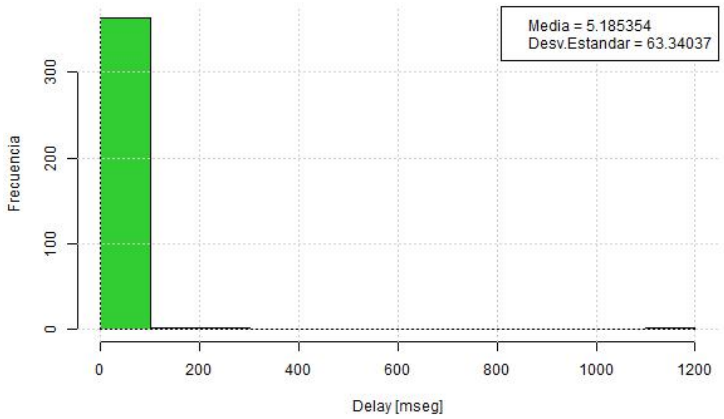

Figura A.54: DELAY (FLOW MONITOR) - Scheduler TD-TBFQ

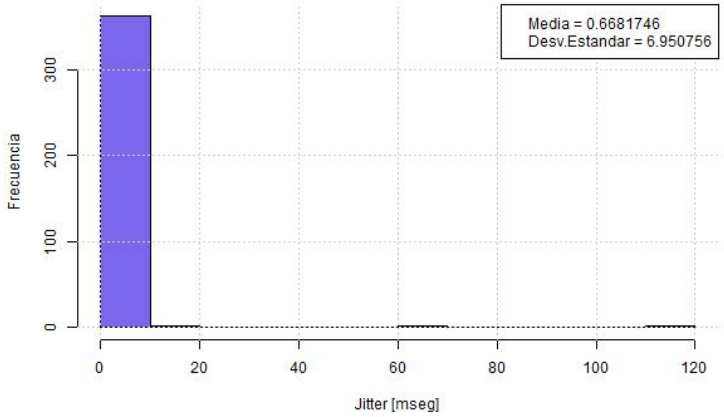

Figura A.55: JITTER (FLOW MONITOR) - Scheduler TD-TBFQ

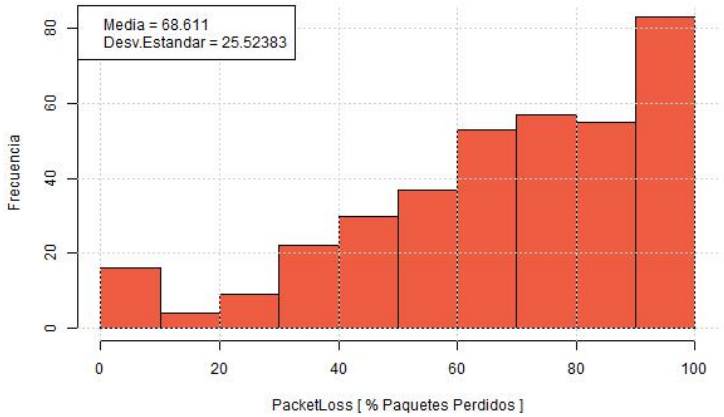

Figura A.56: PACKET LOSS (FLOW MONITOR) - Scheduler TD-TBFQ

- Scheduler PSS

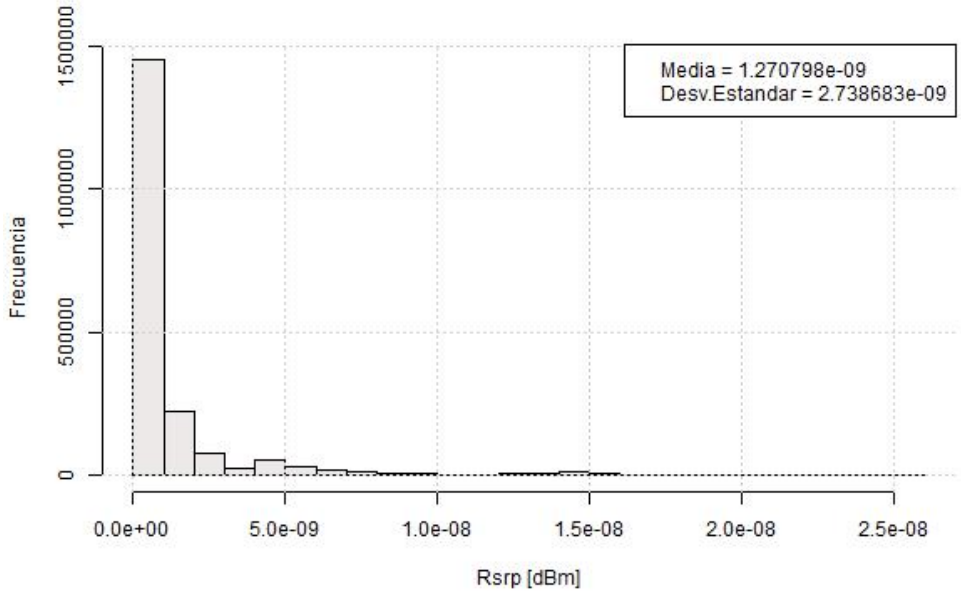

Figura A.57: RSRP - Scheduler PSS

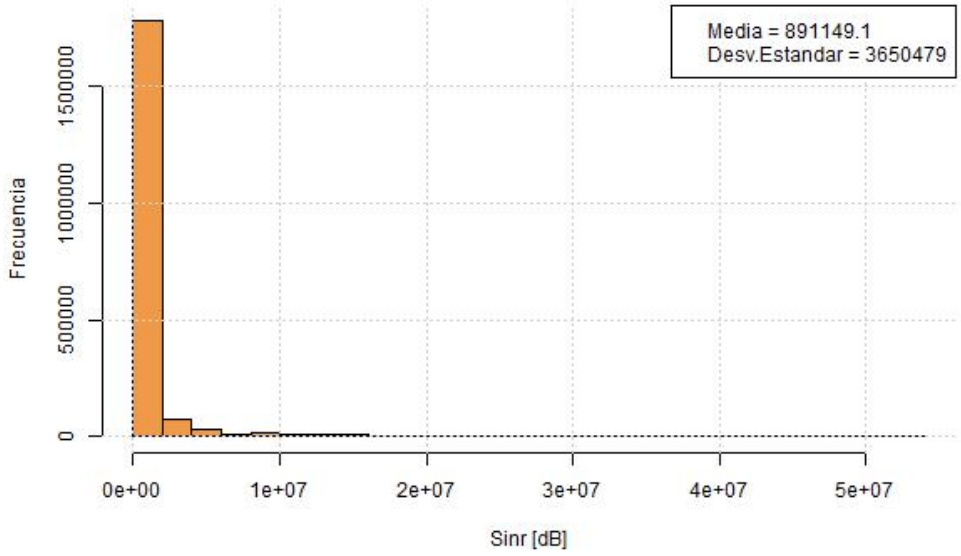

Figura A.58: SINR - Scheduler PSS

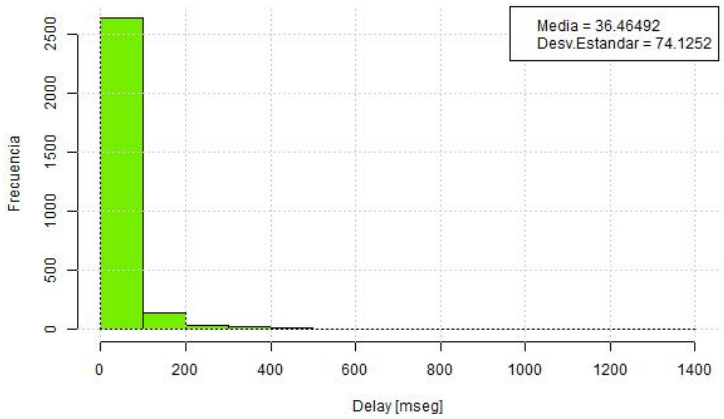

Figura A.59: DELAY (PDCP) - Scheduler PSS

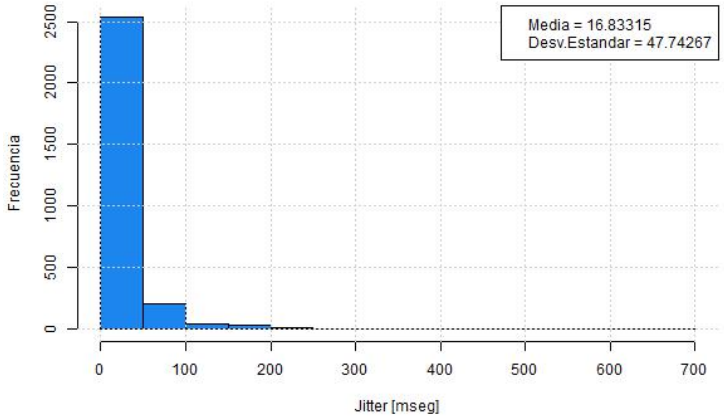

Figura A.60: JITTER (PDCP) - Scheduler PSS

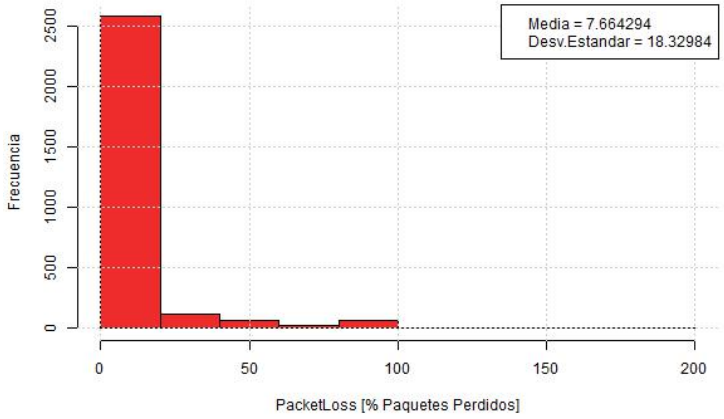

Figura A.61: PACKET LOSS (PDCP) - Scheduler PSS

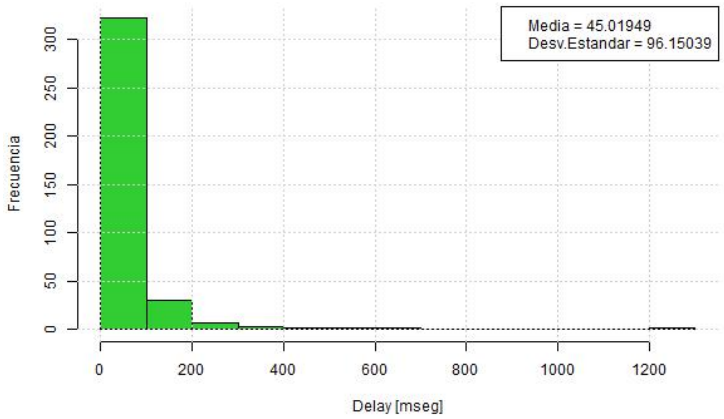

Figura A.62: DELAY (FLOW MONITOR) - Scheduler PSS

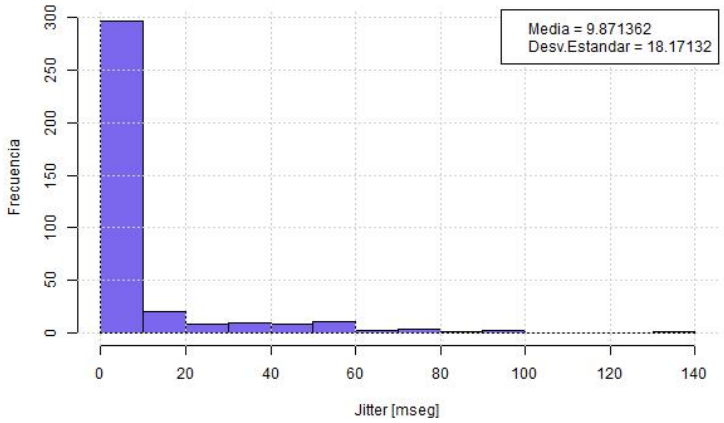

Figura A.63: JITTER (FLOW MONITOR) - Scheduler PSS

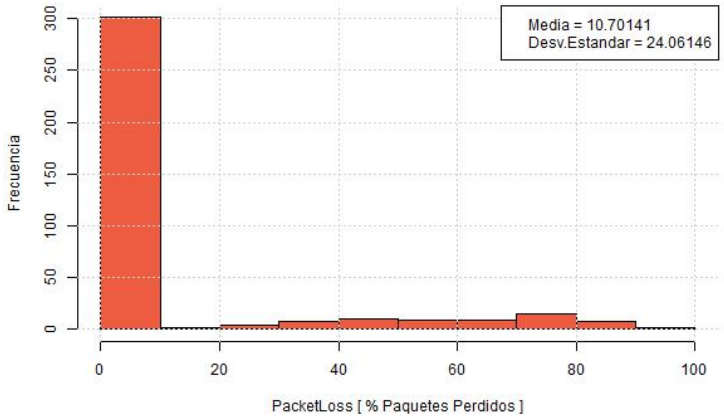

Figura A.64: PACKET LOSS (FLOW MONITOR) - Scheduler PSS

- Scheduler CQA

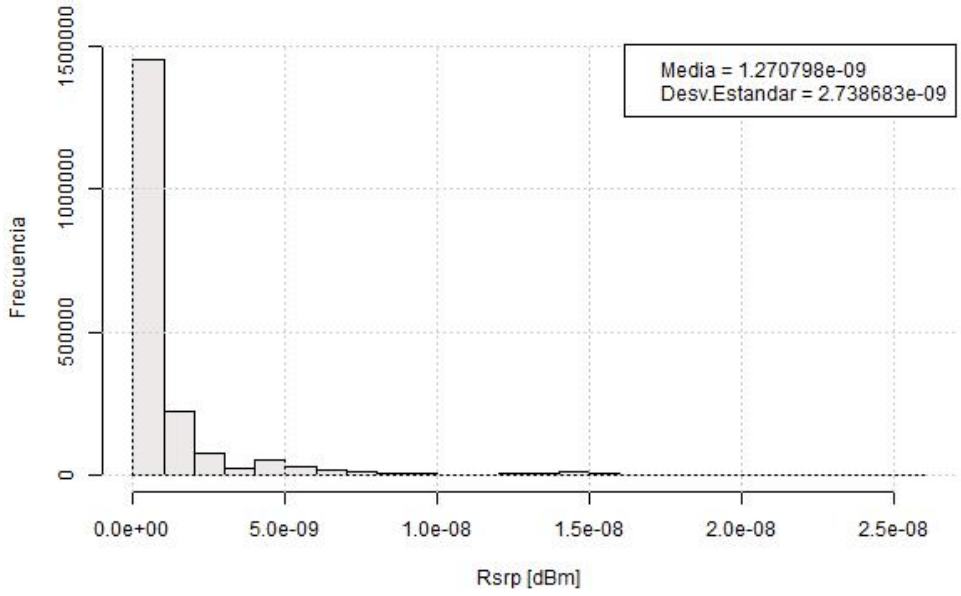

Figura A.65: RSRP - Scheduler CQA

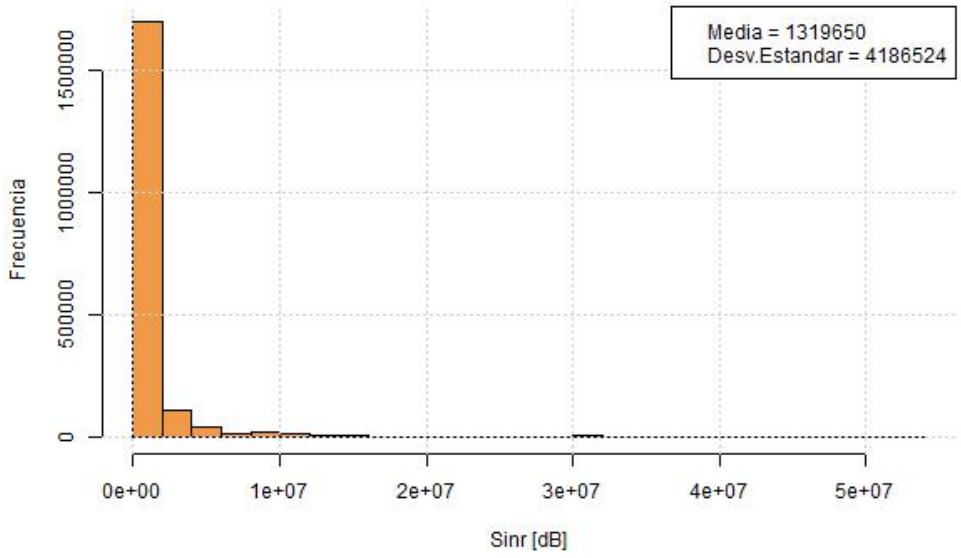

Figura A.66: SINR - Scheduler CQA

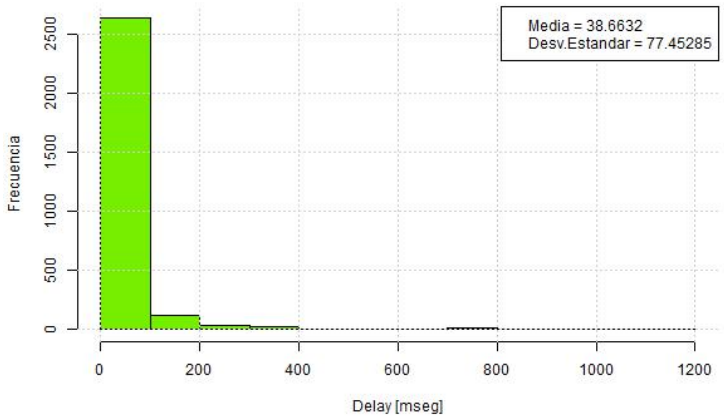

Figura A.67: DELAY (PDCP) - Scheduler CQA

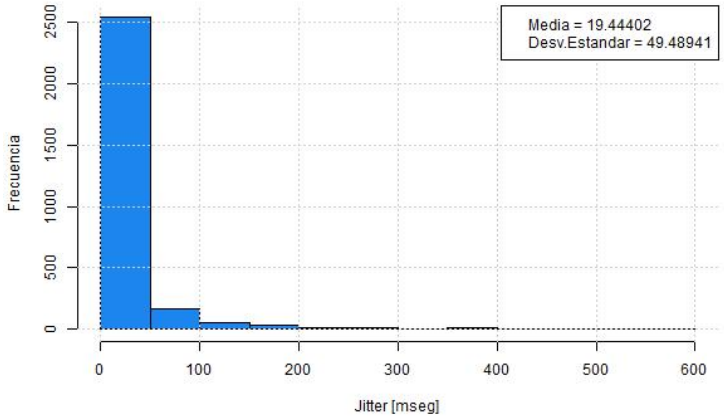

Figura A.68: JITTER (PDCP) - Scheduler CQA

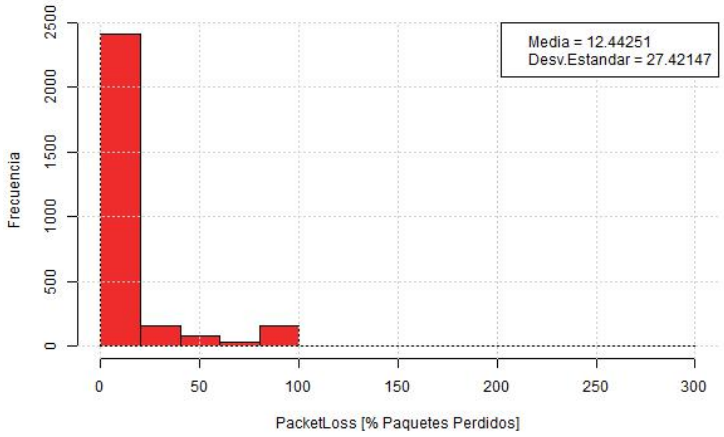

Figura A.69: PACKET LOSS (PDCP) - Scheduler CQA

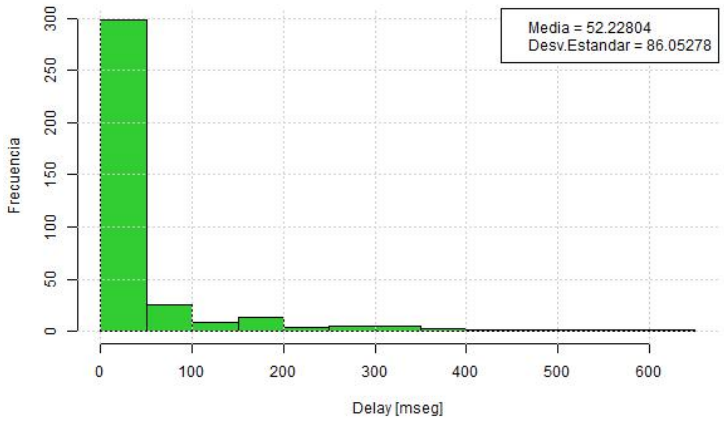

Figura A.70: DELAY (FLOW MONITOR) - Scheduler CQA

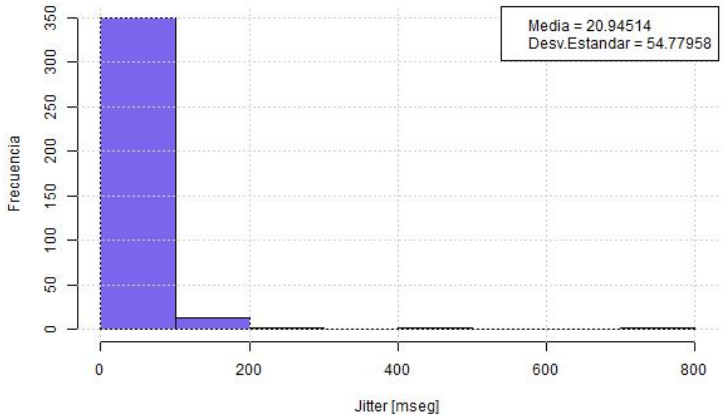

Figura A.71: JITTER (FLOW MONITOR) - Scheduler CQA

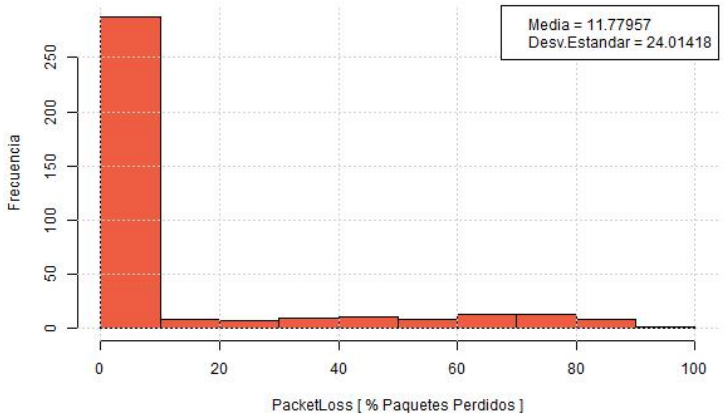

Figura A.72: PACKET LOSS (FLOW MONITOR) - Scheduler CQA

Anexo B

Scripts en R

- Script que importa, modifica y construye los datos

```
######
# Script de extraccion y analisis de datos generados por
# simulador NS-3 para VoLTE
# Preparado por:  Joaquin Chavez B.
# Fecha:          30.01.2015
# Modificado por: Johanna Ortega B.
# Fecha:          03.11.2016
                          #
######

rm(list=ls()) #Se elimina los datos anteriores

# CARGA DE LIBRERIAS ##########################################
# Importacion de paquetes necesarios para la ejecucion del codigo.
#install.packages("Rcmdr")
library(Rcmdr)
require (Rcmdr)

require(XML)

library(Hmisc)
require (Hmisc)

# IMPORTACION ARCHIVO FLOW MONITOR ################################
# Importar datos xml
Flowmonitor <- xmlParse("F:/Flowmon_VoIP.xml")
# Generacion de base de datos a partir de informacion almacenada en archivo XML de
    Flowmonitor.
flowmon_root = xmlRoot(Flowmonitor)
xmlSize(flowmon_root)
xmlSApply(flowmon_root, xmlName)
flowmon_flowstats = xmlChildren(flowmon_root)$FlowStats
```

```
flowmon_ipv4 = xmlChildren(flowmon_root)$Ipv4FlowClassifier
flowmon_flowprobes = xmlChildren(flowmon_root)$FlowProbes
# Rescatar todos los flujos con su respectivo ID.
FlowStats <- as.data.frame(xmlSApply(flowmon_flowstats, xmlGetAttr, "flowId"),
    stringsAsFactors = FALSE)
colnames(FlowStats)[1] <- "flowId"
# Rescatar tiempo de inicio y de termino de transmision de paquetes para cada flujo.
FlowStats$timeFirstTxPacket <- xmlSApply(flowmon_flowstats, xmlGetAttr, "
    timeFirstTxPacket")
FlowStats$timeFirstTxPacket = as.numeric(sub('ns$', '', FlowStats$timeFirstTxPacket))
FlowStats$timeFirstTxPacket = FlowStats$timeFirstTxPacket/10^9 # Tiempo en [seg]
FlowStats$timeLastTxPacket <- xmlSApply(flowmon_flowstats, xmlGetAttr, "
    timeLastTxPacket")
FlowStats$timeLastTxPacket = as.numeric(sub('ns$', '', FlowStats$timeLastTxPacket))
FlowStats$timeLastTxPacket = FlowStats$timeLastTxPacket/10^9 # Tiempo en [seg]
FlowStats$delaySum <- xmlSApply(flowmon_flowstats, xmlGetAttr, "delaySum")
FlowStats$delaySum = as.numeric(sub('ns$', '', FlowStats$delaySum))
# Suma de Delay y Jitter en [ms] por flujo.
FlowStats$delaySum <- xmlSApply(flowmon_flowstats, xmlGetAttr, "delaySum")
FlowStats$delaySum = as.numeric(sub('ns$', '', FlowStats$delaySum))
FlowStats$delaySum = (FlowStats$delaySum)/10^6 # suma de delay en [ms].
FlowStats$jitterSum <- xmlSApply(flowmon_flowstats, xmlGetAttr, "jitterSum")
FlowStats$jitterSum = as.numeric(sub('ns$', '', FlowStats$jitterSum))
FlowStats$jitterSum = (FlowStats$jitterSum)/10^6 # suma de jitter en [ms].
# Numero de packetes enviados, recibidos y perdidos por flujo. Bytes enviados,
    recibidos,
FlowStats$txPackets <- as.numeric(xmlSApply(flowmon_flowstats, xmlGetAttr, "txPackets"
    ))
FlowStats$txBytes <- as.numeric(xmlSApply(flowmon_flowstats, xmlGetAttr, "txBytes"))
FlowStats$rxPackets <- as.numeric(xmlSApply(flowmon_flowstats, xmlGetAttr, "rxPackets"
    ))
FlowStats$rxBytes <- as.numeric(xmlSApply(flowmon_flowstats, xmlGetAttr, "rxBytes"))
FlowStats$lostPackets <- as.numeric(xmlSApply(flowmon_flowstats, xmlGetAttr, "
    lostPackets"))
FlowStats$packetLossPerc <- FlowStats$lostPackets/FlowStats$txPackets*100

# Delay promedio
FlowStats$delayAvg = FlowStats$delaySum/FlowStats$rxPackets
for(i in 1:length(FlowStats$delayAvg))
{
  if(is.nan(FlowStats$delayAvg[i])) {
    FlowStats$delayAvg[i]=0
  }
}

# Jitter promedio
FlowStats$jitterAvg = FlowStats$jitterSum/(FlowStats$rxPackets - 1)
for(i in 1:length(FlowStats$jitterAvg))
{
  if(is.nan(FlowStats$jitterAvg[i])) {
```

```
    FlowStats$jitterAvg[i]=0
    }
}
# Rescatar informacion de origen, destino y protocolo de los flujos.
Ipv4 <- as.data.frame(as.numeric(xmlSApply(flowmon_ipv4, xmlGetAttr, "flowId")),
    stringsAsFactors = FALSE,
                      row.names = NULL)
colnames(Ipv4)[1] <- "flowId"
Ipv4$sourceAddress <- xmlSApply(flowmon_ipv4, xmlGetAttr, "sourceAddress")
Ipv4$sourcePort <- xmlSApply(flowmon_ipv4, xmlGetAttr, "sourcePort")
Ipv4$protocol <- xmlSApply(flowmon_ipv4, xmlGetAttr, "protocol")
Ipv4$destinationAddress <- xmlSApply(flowmon_ipv4, xmlGetAttr, "destinationAddress")
Ipv4$destinationPort <- xmlSApply(flowmon_ipv4, xmlGetAttr, "destinationPort")
Ipv4 <- Ipv4[order(Ipv4$flowId),]
row.names(Ipv4) <- NULL
# Generar un unico data frame con toda la informacion.
Flow_Monitor_Stats <-merge(Ipv4, FlowStats, by="flowId")

# GUARDAR INFORMACION DE FLOW MONITOR ############################
write.csv(Flow_Monitor_Stats,file = "F:/Flow_Monitor_Stats_<scheduler>.txt", row.names
    =TRUE)

# ESTADISTICAS DESCRIPTIVA FLOW MONITOR ##########################
summary (Flow_Monitor_Stats)
numSummary(Flow_Monitor_Stats$delayAvg)
numSummary(Flow_Monitor_Stats$jitterAvg)
numSummary(Flow_Monitor_Stats$packetLossPerc)

# GRAFICOS FLOW MONITOR ##########################################
# Histograma Delay
path <- file.path("F:/<scheduler>/Estadistica Descriptiva", paste("DELAY(FM)_<
    scheduler>", ".jpeg", sep = ""))
jpeg(file=path, width = 600, height = 400)
hist(Flow_Monitor_Stats$delayAvg, #decreasing = FALSE),
     col = "limegreen",
     main = " ",
     xlab = "Delay [mseg]",
     ylab = "Frecuencia")
grid()
legend("topright",
       #bty="n",
       c(paste("Media =", as.name(mean(Flow_Monitor_Stats$delayAvg))),
         paste("Desv.Estandar =", as.name(sd(Flow_Monitor_Stats$delayAvg)))
       )
)
dev.off()
# Histograma Jitter
path <- file.path("F:/<scheduler>/Estadistica Descriptiva", paste("JITTER(FM)_<
    scheduler>", ".jpeg", sep = ""))
jpeg(file=path, width = 600, height = 400)
```

```
hist(Flow_Monitor_Stats$jitterAvg, #decreasing = FALSE),
     col = "slateblue2",
     main = " ",
     xlab = "Jitter [mseg]",
     ylab = "Frecuencia")
grid()
legend("topright",
       c(paste("Media =", as.name(mean(Flow_Monitor_Stats$jitterAvg))),
         paste("Desv.Estandar =", as.name(sd(Flow_Monitor_Stats$jitterAvg)))
       )
)
dev.off()

# Histograma PacketLoss
path <- file.path("F:/<scheduler>/Estadistica Descriptiva", paste("PL(FM)_<scheduler>"
    , ".jpeg", sep = ""))
jpeg(file=path, width = 600, height = 400)
hist(Flow_Monitor_Stats$packetLossPerc, #decreasing = FALSE),
     col = "tomato2",
     main = " ",
     xlab = "PacketLoss [ % Paquetes Perdidos ]",
     ylab = "Frecuencia")
grid()
legend("topright",
       c(paste("Media =", as.name(mean(Flow_Monitor_Stats$packetLossPerc))),
         paste("Desv.Estandar =", as.name(sd(Flow_Monitor_Stats$packetLossPerc)))
       )
)
dev.off()

# IMPORTACION ARCHIVOS .TXT ########################################
DlRsrpSinrStats <- read.delim("F:/<scheduler>/DlRsrpSinrStats.txt",
                              row.names=NULL, strip.white=TRUE)

DlMacStats <- read.delim("F:/<scheduler>/DlMacStats.txt",
                         row.names=NULL, strip.white=TRUE)

DlIpStats <- read.table("F:/<scheduler>/DlIpStats.txt",
                        header=TRUE, sep="", na.strings="NA", dec=".", strip.white=
    TRUE)

# MODIFICACION DATOS .TXT ##########################################
DlIpStats$X <- NULL
#Calculo de IP - DELAY
DlIpStats$delay <- DlIpStats$delay*10^3
#Calculo de IP - JITTER
DlIpStats$jitter <- c(0)
for(i in 2:length(DlIpStats$jitter))
{
  DlIpStats$jitter[i]= (abs(DlIpStats$delay[i]-DlIpStats$delay[i-1]))/2
```

```
}
#Calculo de IP - PL
DlIpStats$start <- as.numeric(DlIpStats$start)
DlIpStats$PL <- abs((DlIpStats$nRxPDUs- DlIpStats$nTxPDUs)/DlIpStats$nTxPDUs)*100

# DATOS IMSI ##########################################################
DlRsrpSinrStatsxIMSI <-aggregate(DlRsrpSinrStats,
                                 by=list(DlRsrpSinrStats$IMSI),
                                 FUN = mean)
DlRsrpSinrStatsxIMSI[,1] <- NULL

DlMacStatsxIMSI <-aggregate(DlMacStats,
                                by=list(DlMacStats$IMSI),
                                FUN = mean)
DlMacStatsxIMSI[,1] <- NULL
DlMacStatsxIMSI$mcsTb1 <- round(DlMacStatsxIMSI$mcsTb1)

#DatosMAcxIMSIxMCS <- aggregate(DlMacStats,
#                                 by=list(DlMacStats$IMSI, DlMacStats$mcsTb1),
#                                 FUN = mean)

# GUARDAR INFORMACION PHY_MAC #######################################
Datos_PHY_MAC <- merge(DlRsrpSinrStatsxIMSI, DlMacStatsxIMSI, by ="IMSI")
write.csv(Datos_PHY_MAC,file = "F:/<scheduler>/Datos_PHY_MAC.txt", row.names=TRUE)
length(Datos_PHY_MAC$IMSI)

# GUARDAR INFORMACION MAC_IP ########################################
Datos_MAC_IP <- merge.data.frame (Datos_PHY_MAC, DlIpStats, by.name = "IMSI")
write.csv(Datos_MAC_IP,file = "F:/<scheduler>/Datos_MAC_IP.txt", row.names=TRUE)
length(Datos_MAC_IP$IMSI)

# ESTADISTICA DESCRIPITIVA ARCHIVOS .TXT ############################
summary (DlRsrpSinrStats)
numSummary(DlRsrpSinrStats)
Estadistica_PHY <- summary(DlRsrpSinrStats)
write.csv(Estadistica_PHY,file = "F:/<scheduler>/Estadistica Descriptiva/Estadistica_
    PHY.txt", row.names=TRUE)
Estadistica_PHY <- latex(Estadistica_PHY, file= "F:/<scheduler>/Estadistica
    Descriptiva/Estadistica_PHY.tex")

summary(DlMacStats)
numSummary(DlMacStats$mcsTb1)
Estadistica_MAC <- summary(DlMacStats)
write.csv(Estadistica_MAC,file = "F:/<scheduler>/Estadistica Descriptiva/Estadistica_
    MAC.txt", row.names=TRUE)
Estadistica_MAC <- latex(Estadistica_MAC, file= "F:/<scheduler>/Estadistica
    Descriptiva/Estadistica_MAC.tex")

summary(DlIpStats)
numSummary(DlIpStats)
```

```
Estadistica_IP <- summary(DlIpStats)
write.csv(Estadistica_IP,file = "F:/<scheduler>/Estadistica Descriptiva/Estadistica_IP
    .txt", row.names=TRUE)
Estadistica_IP <- latex(Estadistica_IP, file= "F:/<scheduler>/Estadistica Descriptiva/
    Estaditica_IP.tex")

# GRAFICOS PHY ######################################################
# Histograma RSRP.
path <- file.path("F:/<scheduler>/Estadistica Descriptiva", paste("RSRP_<scheduler>",
    ".jpeg", sep = ""))
jpeg(file=path, width = 600, height = 400)
hist(DlRsrpSinrStats$rsrp, #decreasing = FALSE),
     col = "snow2",
     main = " ",
     xlab = "Rsrp [dBm]",
     ylab = "Frecuencia")
grid()
legend("topright",
       #bty="n",
       c(paste("Media =", as.name(mean(DlRsrpSinrStats$rsrp))),
         paste("Desv.Estandar =", as.name(sd(DlRsrpSinrStats$rsrp)))
       )
)
dev.off()

# Histograma SINR.
path <- file.path("F:/<scheduler>/Estadistica Descriptiva", paste("SINR_<scheduler>",
    ".jpeg", sep = ""))
jpeg(file=path, width = 600, height = 400)
hist(DlRsrpSinrStats$sinr, #decreasing = FALSE),
     col = "tan2",
     main = " ",
     xlab = "Sinr [dB]",
     ylab = "Frecuencia")
grid()
legend("topright",
       #bty="n",
       c(paste("Media =", as.name(mean(DlRsrpSinrStats$sinr))),
         paste("Desv.Estandar =", as.name(sd(DlRsrpSinrStats$sinr)))
       )
)
dev.off()

# GRAFICOS MAC ######################################################
# Histograma MCS.
path <- file.path("F:/<scheduler>/Estadistica Descriptiva", paste("MCS_<scheduler>", "
    .jpeg", sep = ""))
jpeg(file=path, width = 600, height = 400)
hist(DlMacStats$mcsTb1, #decreasing = FALSE),
     col = "steelblue3",
```

```
     main = " ",
     xlab = "MCS",
     ylab = "Frecuencia")
grid()
legend("topleft",
       c(paste("Media =", as.name(mean(DlMacStats$mcsTb1))),
         paste("Desv.Estandar =", as.name(sd(DlMacStats$mcsTb1))))
       )
dev.off()

# GRAFICOS IP ##################################################
# Histograma Delay.
path <- file.path("F:/<scheduler>/Estadistica Descriptiva", paste("DELAY(PDCP)_<
    scheduler>", ".jpeg", sep = ""))
jpeg(file=path, width = 600, height = 400)
hist(DlIpStats$delay, #decreasing = FALSE),
     col = "chartreuse2",
     main = " ",
     xlab = "Delay [mseg]",
     ylab = "Frecuencia")
    grid()
legend("topright",
   c(paste("Media =", as.name(mean(DlIpStats$delay))),
         paste("Desv.Estandar =", as.name(sd(DlIpStats$delay))))
)
dev.off()

# Histograma Jitter
path <- file.path("F:/<scheduler>/Estadistica Descriptiva", paste("JITTER(PDCP)_<
    scheduler>", ".jpeg", sep = ""))
jpeg(file=path, width = 600, height = 400)
hist(DlIpStats$jitter, #decreasing = FALSE),
     col = "dodgerblue2",
     main = " ",
     xlab = "Jitter [mseg]",
     ylab = "Frecuencia")
grid()
legend("topright",
       #bty="n",
       c(paste("Media =", as.name(mean(DlIpStats$jitter))),
         paste("Desv.Estandar =", as.name(sd(DlIpStats$jitter)))
       )
)
dev.off()

# Histograma PacketLoss
path <- file.path("F:/<scheduler>/Estadistica Descriptiva", paste("PL(PDCP)_<scheduler
    >", ".jpeg", sep = ""))
jpeg(file=path, width = 600, height = 400)
hist(DlIpStats$PL, #decreasing = FALSE),
```

```
     col = "firebrick2",
     main = " ",
     xlab = "PacketLoss [% Paquetes Perdidos]",
     ylab = "Frecuencia",
     xlim = c(min(DlIpStats$PL),max(DlIpStats$PL)))
grid()
legend("topright",
       c(paste("Media =", as.name(mean(DlIpStats$PL))),
         paste("Desv.Estandar =", as.name(sd(DlIpStats$PL)))
       )
)
dev.off()
```

- Script que analiza los datos y realiza los modelos

```
######
# Script de analisis datos y generacion de modelos
# Preparado por:  Johanna Ortega B.
# Fecha:          03.11.2016
######

# CARGA DE LIBRERIAS ########################################
library(locfit)
require (locfit)
# MODELOS ###################################################
# MAC ~ PHY #################################################
# Preparacion de datos ######################################
Datos_PHY_MAC <- read.table("F:/<scheduler>/Datos_PHY_MAC.txt",
                            header=TRUE, sep=",", na.strings="NA", dec=".", strip.
    white=TRUE)
mac.phy <- cbind(Datos_PHY_MAC$mcsTb1,
                 Datos_PHY_MAC$rsrp,
                 Datos_PHY_MAC$sinr)
mac.phy <- as.data.frame(mac.phy)
head(mac.phy)
colnames(mac.phy)[1] <- "mcs"
colnames(mac.phy)[2] <- "rsrp"
colnames(mac.phy)[3] <- "sinr"
head(mac.phy)

# MODELOS LINEALES ##########################################
#Gaussian###################################################
model.mac.phy.g <- glm (mcs ~ rsrp+sinr,
                        family = gaussian( ), data = mac.phy)
step(model.mac.phy.g, direction = "backward")
model.mac.phy.g
summary(model.mac.phy.g)
fit.mac.phy.g <- fitted(model.mac.phy.g)
residual.model.mac.phy.g <- resid(model.mac.phy.g)

modelo_g.mac.phy <- cbind("g", round(sum(residual.model.mac.phy.g), digits = 2),
                          round(mean(residual.model.mac.phy.g), digits = 2),
                          round(var(residual.model.mac.phy.g), digits =2))

#Poisson####################################################
model.mac.phy.p <- glm (mcs ~ rsrp+sinr,
                        family = poisson( ), data = mac.phy)
step(model.mac.phy.p, direction = "backward")
model.mac.phy.p
summary(model.mac.phy.p)
fit.mac.phy.p <- fitted(model.mac.phy.p)
residual.model.mac.phy.p <- resid(model.mac.phy.p)
```

```
modelo_p.mac.phy <- cbind("p", round(sum(residual.model.mac.phy.p), digits = 2),
                          round(mean(residual.model.mac.phy.p), digits = 2),
                          round(var(residual.model.mac.phy.p), digits =2))

# MODELOS NO PARAMETRICOS ###################################
#loess - Nearest-neighbour local-polynomial regression########
model.mac.phy.loess = loess(mcs ~ rsrp+sinr, data = mac.phy)
summary(model.mac.phy.loess)
fit.mac.phy.loess <- fitted(model.mac.phy.loess)
RSS.mac.phy.loess<-mean((fit.mac.phy.loess-mac.phy$mcs)^2)
RSS.mac.phy.loess
aic(model.mac.phy.loess)
residual.model.mac.phy.loess <- resid(model.mac.phy.loess)

modelo_loess.mac.phy <- cbind("loess", round(sum(residual.model.mac.phy.loess), digits
    = 2),
                              round(mean(residual.model.mac.phy.loess), digits = 2),
                              round(var(residual.model.mac.phy.loess), digits =2))

#ppr - Projection-Pursuit Regression#######################
model.mac.phy.ppr <- ppr(mac.phy$mcs ~ mac.phy$rsrp+mac.phy$sinr, data = mac.phy,
    nterms =2868)
model.mac.phy.ppr
summary(model.mac.phy.ppr)
fit.mac.phy.ppr <- fitted(model.mac.phy.ppr)
RSS.mac.phy.ppr<-mean((fit.mac.phy.ppr-mac.phy$mcs)^2)
RSS.mac.phy.ppr
aic(model.mac.phy.ppr)
residual.model.mac.phy.ppr <- resid(model.mac.phy.ppr)

modelo_ppr.mac.phy <- cbind("ppr", round(sum(residual.model.mac.phy.ppr), digits = 2),
                            round(mean(residual.model.mac.phy.ppr), digits = 2),
                            round(var(residual.model.mac.phy.ppr), digits =2))

#locfit - Local Polynomial Regression#######################
model.mac.phy.locfit<-locfit( mcs ~ lp (rsrp+sinr), data = mac.phy,
                              scale = TRUE)
model.mac.phy.locfit
summary(model.mac.phy.locfit)
fit.mac.phy.locfit <- fitted(model.mac.phy.locfit)
RSS.mac.phy.locfit<-mean((fit.mac.phy.locfit-mac.phy$mcs)^2)
RSS.mac.phy.locfit
aic(model.mac.phy.locfit)
residual.model.mac.phy.locfit <- resid(model.mac.phy.locfit)

modelo_locfit.mac.phy <- cbind("locfit", round(sum(residual.model.mac.phy.locfit),
    digits = 2),
                               round(mean(residual.model.mac.phy.locfit), digits = 2),
                               round(var(residual.model.mac.phy.locfit), digits =2))
```

```
#gam - Additive Nonparametric Regression#####################
library(gam)
require(gam)
model.mac.phy.gam = gam(mcs ~ rsrp+sinr, data = mac.phy )
model.mac.phy.gam
summary(model.mac.phy.gam)
fit.mac.phy.gam <- model.mac.phy.gam$fitted.values
RSS.mac.phy.gam<-mean((fit.mac.phy.gam-mac.phy$mcs)^2)
RSS.mac.phy.gam
aic(model.mac.phy.gam)
residual.model.mac.phy.gam <- resid(model.mac.phy.gam)

modelo_gam.mac.phy <- cbind("gam", round(sum(residual.model.mac.phy.gam), digits = 2),
                            round(mean(residual.model.mac.phy.gam), digits = 2),
                            round(var(residual.model.mac.phy.gam), digits =2))

#sm.regression - Local linear Regression#####################
library(sm)
require (sm)
macphymatrizbivarianteCE<-mac.phy[,2:3]
head(mac.phy)
model.mac.phy.sm<-sm.regression(macphymatrizbivarianteCE, mac.phy$mcs, poly.index=1,
    method="cv",
                                eval.grid=FALSE, eval.points=macphymatrizbivarianteCE)
model.mac.phy.sm
summary(model.mac.phy.sm)
fit.mac.phy.sm<-model.mac.phy.sm$model.y
RSS.mac.phy.sm<-mean((fit.mac.phy.sm-mac.phy$mcs)^2)
RSS.mac.phy.sm
aic(model.mac.phy.sm)
residual.model.mac.phy.sm <- mac.phy$mcs-fit.mac.phy.sm

modelo_sm.mac.phy <- cbind("sm", round(sum(residual.model.mac.phy.sm), digits = 2),
                           round(mean(residual.model.mac.phy.sm), digits = 2),
                           round(var(residual.model.mac.phy.sm), digits =2))

#polymars - Multivariate Additive Regression Models###########
library(polspline)
require(polspline)
model.mac.phy.polymars =polymars(mac.phy$mcs, mac.phy$rsrp+mac.phy$sinr)
summary(model.mac.phy.polymars)
fit.mac.phy.polymars <-(mac.phy$mcs-model.mac.phy.polymars$residuals)
RSS.mac.phy.polymars <-mean((fit.mac.phy.polymars-mac.phy$mcs)^2)
RSS.mac.phy.polymars
aic(model.mac.phy.polymars)
residual.model.mac.phy.polymars <- resid(model.mac.phy.polymars)

modelo_polymars.mac.phy <- cbind("polymars", round(sum(residual.model.mac.phy.polymars
    ), digits = 2),
                                 round(mean(residual.model.mac.phy.polymars), digits =
```

```
    2),
                                round(var(residual.model.mac.phy.polymars), digits
   =2))

#mars- Multivariate Adaptive Regression Splines###############
library(mda)
require (mda)
model.mac.phy.mars = mars(mac.phy[,2:3], mac.phy$mcs)
model.mac.phy.mars
summary(model.mac.phy.mars)
fit.mac.phy.mars <- fitted(model.mac.phy.mars)
RSS.mac.phy.mars <-mean((fit.mac.phy.mars-mac.phy$mcs)^2)
RSS.mac.phy.mars
aic(model.mac.phy.mars)
residual.model.mac.phy.mars <- resid(model.mac.phy.mars)

modelo_mars.mac.phy <- cbind("mars", round(sum(residual.model.mac.phy.mars), digits =
    2),
                             round(mean(residual.model.mac.phy.mars), digits = 2),
                             round(var(residual.model.mac.phy.mars), digits =2))

#cox - Cox survival-Regression Models#########################
library(survival)
require(survival)
model.mac.phy.cox <- coxph(Surv(mac.phy$mcs)~mac.phy$rsrp+mac.phy$sinr)
summary(model.mac.phy.cox)
fit.mac.phy.cox <-(mac.phy$mcs-model.mac.phy.cox$residuals)
RSS.mac.phy.cox <-mean((fit.mac.phy.cox-mac.phy$mcs)^2)
RSS.mac.phy.cox
AIC(model.mac.phy.cox)
residual.model.mac.phy.cox <- resid(model.mac.phy.cox)

modelo_cox.mac.phy <- cbind("cox", round(sum(residual.model.mac.phy.cox), digits = 2),
                            round(mean(residual.model.mac.phy.cox), digits = 2),
                            round(var(residual.model.mac.phy.cox), digits =2))

## MODELOS NO PARAMETRICOS MAC-PHY ############################
NoParametricos_MAC_PHY <- rbind(cbind ("loess",round(RSS.mac.phy.loess, digits = 2)),
                                cbind ("ppr",round(RSS.mac.phy.ppr, digits = 2)),
                                cbind ("locfit",round(RSS.mac.phy.locfit, digits = 2))
    ,
                                cbind ("gam",round(RSS.mac.phy.gam, digits = 2)),
                                cbind ("sm",round(RSS.mac.phy.sm, digits = 2)),
                                cbind ("polymars",round(RSS.mac.phy.polymars, digits =
    2)),
                                cbind ("mars",round(RSS.mac.phy.mars, digits = 2)),
                                cbind ("cox",round(RSS.mac.phy.cox, digits = 2)))

colnames(NoParametricos_MAC_PHY)[1] <-"Modelo"
colnames(NoParametricos_MAC_PHY)[2] <-"RSS"
```

```
colnames(NoParametricos_MAC_PHY)[1] <-"Modelo"
NoParametricos_MAC_PHY

## TABLA DE RESIDUOS MAC-PHY ################################
Residuos_MAC_PHY <- rbind.data.frame(modelo_g.mac.phy,
                                     modelo_p.mac.phy,
                                     modelo_loess.mac.phy,
                                     modelo_ppr.mac.phy,
                                     modelo_locfit.mac.phy,
                                     modelo_gam.mac.phy,
                                     modelo_sm.mac.phy,
                                     modelo_polymars.mac.phy,
                                     modelo_mars.mac.phy,
                                     modelo_cox.mac.phy)
colnames(Residuos_MAC_PHY)[1] <-"Modelo"
colnames(Residuos_MAC_PHY)[2] <-"Residuos"
colnames(Residuos_MAC_PHY)[3] <-"Media"
colnames(Residuos_MAC_PHY)[4] <-"Varianza"

Residuos_MAC_PHY

## TABLA DE KS MAC-PHY ######################################
KS_MAC_PHY <- rbind.data.frame(ks.test(mac.phy$mcs,fit.mac.phy.g),
                               ks.test(mac.phy$mcs,fit.mac.phy.p),
                               ks.test(mac.phy$mcs,fit.mac.phy.loess),
                               ks.test(mac.phy$mcs,fit.mac.phy.ppr),
                               ks.test(mac.phy$mcs,fit.mac.phy.locfit),
                               ks.test(mac.phy$mcs,fit.mac.phy.gam),
                               ks.test(mac.phy$mcs,fit.mac.phy.sm),
                               ks.test(mac.phy$mcs,fit.mac.phy.polymars),
                               ks.test(mac.phy$mcs,fit.mac.phy.mars),
                               ks.test(mac.phy$mcs,fit.mac.phy.cox))
colnames(KS_MAC_PHY)[1] <-"D"
colnames(KS_MAC_PHY)[2] <-"P-VALUE"
KS_MAC_PHY$alternative <- NULL
KS_MAC_PHY$method <- NULL

KS_MAC_PHY

## IP ~ MAC #################################################
# Preparacion de datos #####################################
Datos_MAC_IP <- read.table("F:/<scheduler>/Datos_MAC_IP.txt",
                           header=TRUE, sep=",", na.strings="NA", dec=".", strip.white
    =TRUE)

#elijo el mejor modelo mac.phy
model.frame.default(model.mac.phy.cox)

#hago una prediccion con el modelo, pero con los datos MAC IP
mejormodelo <- predict(model.mac.phy.cox, newdata = Datos_MAC_IP)
```

```
ip.mac <- cbind(Datos_MAC_IP$mcsTb1,
                Datos_MAC_IP$delay,
                Datos_MAC_IP$jitter,
                Datos_MAC_IP$PL,
                mejormodelo)

ip.mac <- as.data.frame(ip.mac)
head(ip.mac)
colnames(ip.mac)[1] <- "mcs"
colnames(ip.mac)[2] <- "delay"
colnames(ip.mac)[3] <- "jitter"
colnames(ip.mac)[4] <- "pl"
colnames(ip.mac)[5] <- "model_phy_mac"
head(ip.mac)

# MODELOS LINEALES ##########################################
#DELAY#######################################################
#Gaussian####################################################
model.ip.mac.g1 <- glm (delay ~ mcs+model_phy_mac,
                        family = gaussian( ), data = ip.mac)
step(model.ip.mac.g1, direction = "backward")
summary(model.ip.mac.g1)
fit.ip.mac.g1 <- fitted(model.ip.mac.g1)
residual.model.ip.mac.g1 <- resid(model.ip.mac.g1)

modelo_g1.ip.mac <- cbind("gaussian_delay", round(sum(residual.model.ip.mac.g1),
    digits = 2),
                          round(mean(residual.model.ip.mac.g1), digits = 2),
                          round(var(residual.model.ip.mac.g1), digits =2))

#JITTER######################################################
#Gaussian####################################################
model.ip.mac.g2 <- glm (jitter ~ mcs+model_phy_mac,
                        family = gaussian( ), data = ip.mac)
step(model.ip.mac.g2, direction = "backward")
summary(model.ip.mac.g2)
fit.ip.mac.g2 <- fitted(model.ip.mac.g2)
residual.model.ip.mac.g2 <- resid(model.ip.mac.g2)
modelo_g2.ip.mac <- cbind("gaussian_jitter", round(sum(residual.model.ip.mac.g2),
    digits = 2),
                          round(mean(residual.model.ip.mac.g2), digits = 2),
                          round(var(residual.model.ip.mac.g2), digits = 2))

#PACKETLOSS##################################################
#Gaussian####################################################
model.ip.mac.g3 <- glm (pl ~ mcs+model_phy_mac,
                        family = gaussian( ), data = ip.mac)
step(model.ip.mac.g3, direction = "backward")
summary(model.ip.mac.g3)
```

```
fit.ip.mac.g3 <- fitted(model.ip.mac.g3)
residual.model.ip.mac.g3 <- resid(model.ip.mac.g3)
modelo_g3.ip.mac <- cbind("gaussian_pl", round(sum(residual.model.ip.mac.g3),digits =
    2),
                          round(mean(residual.model.ip.mac.g3), digits = 2),
                          round(var(residual.model.ip.mac.g3), digits = 2))

# MODELOS NO PARAMETRICOS ###################################
#loess - Nearest-neighbour Local-Polynomial Regression########
model.ip.mac.loess1 = loess(ip.mac$delay ~ ip.mac$mcs+ ip.mac$model_phy_mac, data = ip
    .mac)
summary(model.ip.mac.loess1)
fit.ip.mac.loess1 <- fitted(model.ip.mac.loess1)
RSS.ip.mac.loess1<-mean((fit.ip.mac.loess1-ip.mac$delay)^2)
RSS.ip.mac.loess1
aic(model.ip.mac.loess1)
residual.model.ip.mac.loess1 <- resid(model.ip.mac.loess1)

modelo_loess1.ip.mac <- cbind("loess_delay", round(sum(residual.model.ip.mac.loess1),
    digits = 2),
                              round(mean(residual.model.ip.mac.loess1), digits = 2),
                              round(var(residual.model.ip.mac.loess1), digits =2))

model.ip.mac.loess2 = loess(ip.mac$jitter ~ ip.mac$mcs+ ip.mac$model_phy_mac, data =
    ip.mac)
summary(model.ip.mac.loess2)
fit.ip.mac.loess2 <- fitted(model.ip.mac.loess2)
RSS.ip.mac.loess2<-mean((fit.ip.mac.loess2-ip.mac$jitter)^2)
RSS.ip.mac.loess2
aic(model.ip.mac.loess2)
residual.model.ip.mac.loess2 <- resid(model.ip.mac.loess2)

modelo_loess2.ip.mac <- cbind("loess_jitter", round(sum(residual.model.ip.mac.loess2),
    digits = 2),
                              round(mean(residual.model.ip.mac.loess2), digits = 2),
                              round(var(residual.model.ip.mac.loess2), digits = 2))

model.ip.mac.loess3 = loess(ip.mac$pl ~ ip.mac$mcs+ ip.mac$model_phy_mac, data = ip.
    mac)
summary(model.ip.mac.loess3)
fit.ip.mac.loess3 <- fitted(model.ip.mac.loess3)
RSS.ip.mac.loess3<-mean((fit.ip.mac.loess3-ip.mac$pl)^2)
RSS.ip.mac.loess3
aic(model.ip.mac.loess3)
residual.model.ip.mac.loess3 <- resid(model.ip.mac.loess3)

modelo_loess3.ip.mac <- cbind("loess_pl", round(sum(residual.model.ip.mac.loess3),
    digits = 2),
                              round(mean(residual.model.ip.mac.loess3), digits = 2),
                              round(var(residual.model.ip.mac.loess3), digits = 2))
```

```
#ppr - Projection-Pursuit Regression#########################
model.ip.mac.ppr1 <- ppr(ip.mac$delay ~ ip.mac$mcs+ip.mac$model_phy_mac, data = ip.mac
    , nterms =2868)
model.ip.mac.ppr1
summary(model.ip.mac.ppr1)
fit.ip.mac.ppr1 <- fitted(model.ip.mac.ppr1)
RSS.ip.mac.ppr1<-mean((fit.ip.mac.ppr1-ip.mac$delay)^2)
RSS.ip.mac.ppr1
aic(model.ip.mac.ppr1)
residual.model.ip.mac.ppr1 <- resid(model.ip.mac.ppr1)

modelo_ppr1.ip.mac <- cbind("ppr_delay", round(sum(residual.model.ip.mac.ppr1), digits
     = 2),
                            round(mean(residual.model.ip.mac.ppr1), digits = 2),
                            round(var(residual.model.ip.mac.ppr1), digits =2))

model.ip.mac.ppr2 <- ppr(ip.mac$jitter ~ ip.mac$mcs+ip.mac$model_phy_mac, data = ip.
    mac, nterms =2868)
model.ip.mac.ppr2
summary(model.ip.mac.ppr2)
fit.ip.mac.ppr2 <- fitted(model.ip.mac.ppr2)
RSS.ip.mac.ppr2<-mean((fit.ip.mac.ppr2-ip.mac$jitter)^2)
RSS.ip.mac.ppr2
aic(model.ip.mac.ppr2)
residual.model.ip.mac.ppr2 <- resid(model.ip.mac.ppr2)

modelo_ppr2.ip.mac <- cbind("ppr_jitter", round(sum(residual.model.ip.mac.ppr2),
    digits = 2),
                            round(mean(residual.model.ip.mac.ppr2), digits = 2),
                            round(var(residual.model.ip.mac.ppr2), digits =2))

model.ip.mac.ppr3 <- ppr(ip.mac$pl ~ ip.mac$mcs+ip.mac$model_phy_mac, data = ip.mac,
    nterms =2868)
model.ip.mac.ppr3
summary(model.ip.mac.ppr3)
fit.ip.mac.ppr3 <- fitted(model.ip.mac.ppr3)
RSS.ip.mac.ppr3<-mean((fit.ip.mac.ppr3-ip.mac$pl)^2)
RSS.ip.mac.ppr3
aic(model.ip.mac.ppr3)
residual.model.ip.mac.ppr3 <- resid(model.ip.mac.ppr3)

modelo_ppr3.ip.mac <- cbind("ppr_pl", round(sum(residual.model.ip.mac.ppr3), digits =
    2),
                            round(mean(residual.model.ip.mac.ppr3), digits = 2),
                            round(var(residual.model.ip.mac.ppr3), digits =2))

#locfit - Local Polynomial Regression#########################
model.ip.mac.locfit1<-locfit(ip.mac$delay ~ lp (ip.mac$mcs+ip.mac$model_phy_mac),
                             scale = TRUE)
```

```
model.ip.mac.locfit1
summary(model.ip.mac.locfit1)
fit.ip.mac.locfit1 <- fitted(model.ip.mac.locfit1)
RSS.ip.mac.locfit1<-mean((fit.ip.mac.locfit1-ip.mac$delay)^2)
RSS.ip.mac.locfit1
aic(model.ip.mac.locfit1)
residual.model.ip.mac.locfit1 <- resid(model.ip.mac.locfit1)

modelo_locfit1.ip.mac <- cbind("locfit_delay", round(sum(residual.model.ip.mac.locfit1
    ), digits = 2),
                               round(mean(residual.model.ip.mac.locfit1), digits = 2),
                               round(var(residual.model.ip.mac.locfit1), digits =2))

model.ip.mac.locfit2<-locfit(ip.mac$jitter ~ lp (ip.mac$mcs+ip.mac$model_phy_mac),
                             scale = TRUE)
model.ip.mac.locfit2
summary(model.ip.mac.locfit2)
fit.ip.mac.locfit2 <- fitted(model.ip.mac.locfit2)
RSS.ip.mac.locfit2<-mean((fit.ip.mac.locfit2-ip.mac$jitter)^2)
RSS.ip.mac.locfit2
aic(model.ip.mac.locfit2)
residual.model.ip.mac.locfit2 <- resid(model.ip.mac.locfit2)

modelo_locfit2.ip.mac <- cbind("locfit_jitter", round(sum(residual.model.ip.mac.
    locfit2),digits = 2),
                               round(mean(residual.model.ip.mac.locfit2), digits = 2),
                               round(var(residual.model.ip.mac.locfit2), digits = 2))

model.ip.mac.locfit3<-locfit(ip.mac$pl ~ lp (ip.mac$mcs+ip.mac$model_phy_mac),
                             scale = TRUE)
model.ip.mac.locfit3
summary(model.ip.mac.locfit3)
fit.ip.mac.locfit3 <- fitted(model.ip.mac.locfit3)
RSS.ip.mac.locfit3 <- mean((fit.ip.mac.locfit3-ip.mac$pl)^2)
RSS.ip.mac.locfit3
aic(model.ip.mac.locfit3)
residual.model.ip.mac.locfit3 <- resid(model.ip.mac.locfit3)

modelo_locfit3.ip.mac <- cbind("locfit_pl", round(sum(residual.model.ip.mac.locfit3),
    digits = 2),
                               round(mean(residual.model.ip.mac.locfit3), digits = 2),
                               round(var(residual.model.ip.mac.locfit3), digits = 2))

#gam - Additive Nonparametric Regression######################
model.ip.mac.gam1 = gam(delay ~ mcs+model_phy_mac, data = ip.mac )
model.ip.mac.gam1
summary(model.ip.mac.gam1)
fit.ip.mac.gam1 <- model.ip.mac.gam1$fitted.values
RSS.ip.mac.gam1<-mean((fit.ip.mac.gam1-ip.mac$delay)^2)
RSS.ip.mac.gam1
```

```
aic(model.ip.mac.gam1)
residual.model.ip.mac.gam1 <- resid(model.ip.mac.gam1)

modelo_gam1.ip.mac <- cbind("gam_delay", round(sum(residual.model.ip.mac.gam1), digits
    = 2),
                            round(mean(residual.model.ip.mac.gam1), digits = 2),
                            round(var(residual.model.ip.mac.gam1), digits =2))

model.ip.mac.gam2 = gam(jitter ~ mcs+model_phy_mac, data = ip.mac )
model.ip.mac.gam2
summary(model.ip.mac.gam2)
fit.ip.mac.gam2 <- model.ip.mac.gam2$fitted.values
RSS.ip.mac.gam2<-mean((fit.ip.mac.gam2-ip.mac$jitter)^2)
RSS.ip.mac.gam2
aic(model.ip.mac.gam2)
residual.model.ip.mac.gam2 <- resid(model.ip.mac.gam2)

modelo_gam2.ip.mac <- cbind("gam_jitter", round(sum(residual.model.ip.mac.gam2),digits
    = 2),
                            round(mean(residual.model.ip.mac.gam2), digits = 2),
                            round(var(residual.model.ip.mac.gam2), digits = 2))

model.ip.mac.gam3 = gam(pl ~ mcs+model_phy_mac, data = ip.mac )
model.ip.mac.gam3
summary(model.ip.mac.gam3)
fit.ip.mac.gam3 <- model.ip.mac.gam3$fitted.values
RSS.ip.mac.gam3<-mean((fit.ip.mac.gam3-ip.mac$pl)^2)
RSS.ip.mac.gam3
aic(model.ip.mac.gam3)
residual.model.ip.mac.gam3 <- resid(model.ip.mac.gam3)

modelo_gam3.ip.mac <- cbind("gam_pl", round(sum(residual.model.ip.mac.gam3),digits =
    2),
                            round(mean(residual.model.ip.mac.gam3), digits = 2),
                            round(var(residual.model.ip.mac.gam3), digits = 2))

#sm.regression - Local linear Regression#####################
ipmacmatrizbivarianteCE1<-cbind(ip.mac$mcs,ip.mac$model_phy_mac)
model.ip.mac.sm1<-sm.regression(ipmacmatrizbivarianteCE1, ip.mac$delay, poly.index=1,
    method="cv",
                                eval.grid=FALSE, eval.points=ipmacmatrizbivarianteCE1)
model.ip.mac.sm1


summary(model.ip.mac.sm1)
fit.ip.mac.sm1<-model.ip.mac.sm1$model.y
RSS.ip.mac.sm1<-mean((fit.ip.mac.sm1-ip.mac$delay)^2)
RSS.ip.mac.sm1
aic(model.ip.mac.sm1)
residual.model.ip.mac.sm1 <- ip.mac$delay-fit.ip.mac.sm1
```

```
modelo_sm1.ip.mac <- cbind("sm_delay", round(sum(residual.model.ip.mac.sm1), digits =
    2),
                           round(mean(residual.model.ip.mac.sm1), digits = 2),
                           round(var(residual.model.ip.mac.sm1), digits =2))

ipmacmatrizbivarianteCE2<-cbind(ip.mac$mcs,ip.mac$model_phy_mac)
model.ip.mac.sm2<-sm.regression(ipmacmatrizbivarianteCE2, ip.mac$jitter, poly.index=1,
     method="cv",
                                eval.grid=FALSE, eval.points=ipmacmatrizbivarianteCE2)
model.ip.mac.sm2
summary(model.ip.mac.sm2)
fit.ip.mac.sm2<-model.ip.mac.sm2$model.y
RSS.ip.mac.sm2<-mean((fit.ip.mac.sm2-ip.mac$jitter)^2)
RSS.ip.mac.sm2
aic(model.ip.mac.sm2)
residual.model.ip.mac.sm2 <- ip.mac$jitter-fit.ip.mac.sm2

modelo_sm2.ip.mac <- cbind("sm_jitter", round(sum(residual.model.ip.mac.sm2),digits =
    2),
                           round(mean(residual.model.ip.mac.sm2), digits = 2),
                           round(var(residual.model.ip.mac.sm2), digits = 2))

ipmacmatrizbivarianteCE3<-cbind(ip.mac$mcs,ip.mac$model_phy_mac)
model.ip.mac.sm3<-sm.regression(ipmacmatrizbivarianteCE3, ip.mac$pl, poly.index=1,
    method="cv",
                                eval.grid=FALSE, eval.points=ipmacmatrizbivarianteCE3)
model.ip.mac.sm3
summary(model.ip.mac.sm3)
fit.ip.mac.sm3<-model.ip.mac.sm3$model.y
RSS.ip.mac.sm3<-mean((fit.ip.mac.sm3-ip.mac$pl)^3)
RSS.ip.mac.sm3
aic(model.ip.mac.sm3)
residual.model.ip.mac.sm3 <- ip.mac$mcs-fit.ip.mac.sm3

modelo_sm3.ip.mac <- cbind("sm_pl", round(sum(residual.model.ip.mac.sm3),digits = 2),
                           round(mean(residual.model.ip.mac.sm3), digits = 2),
                           round(var(residual.model.ip.mac.sm3), digits = 2))

#polymars - Multivariate Additive Regression Models###########
model.ip.mac.polymars1 =polymars(ip.mac$delay, ip.mac$mcs+ip.mac$model_phy_mac)
summary(model.ip.mac.polymars1)
fit.ip.mac.polymars1 <-(ip.mac$delay-model.ip.mac.polymars1$residuals)
RSS.ip.mac.polymars1 <-mean((fit.ip.mac.polymars1-ip.mac$delay)^2)
RSS.ip.mac.polymars1
aic(model.ip.mac.polymars1)
residual.model.ip.mac.polymars1 <- resid(model.ip.mac.polymars1)

modelo_polymars1.ip.mac <- cbind("polymars_delay", round(sum(residual.model.ip.mac.
    polymars1), digits = 2),
```

```
                                round(mean(residual.model.ip.mac.polymars1), digits =
    2),
                                round(var(residual.model.ip.mac.polymars1), digits
    =2))

model.ip.mac.polymars2 =polymars(ip.mac$jitter, ip.mac$mcs+ip.mac$model_phy_mac)
summary(model.ip.mac.polymars2)
fit.ip.mac.polymars2 <-(ip.mac$jitter-model.ip.mac.polymars2$residuals)
RSS.ip.mac.polymars2 <-mean((fit.ip.mac.polymars2-ip.mac$jitter)^2)
RSS.ip.mac.polymars2
aic(model.ip.mac.polymars2)
residual.model.ip.mac.polymars2 <- resid(model.ip.mac.polymars2)

modelo_polymars2.ip.mac <- cbind("polymars_jitter", round(sum(residual.model.ip.mac.
    polymars2), digits = 2),
                                round(mean(residual.model.ip.mac.polymars2), digits =
    2),
                                round(var(residual.model.ip.mac.polymars2), digits
    =2))

model.ip.mac.polymars3 =polymars(ip.mac$pl, ip.mac$mcs+ip.mac$model_phy_mac)
summary(model.ip.mac.polymars3)
fit.ip.mac.polymars3 <-(ip.mac$pl-model.ip.mac.polymars3$residuals)
RSS.ip.mac.polymars3 <-mean((fit.ip.mac.polymars3-ip.mac$pl)^2)
RSS.ip.mac.polymars3
aic(model.ip.mac.polymars3)
residual.model.ip.mac.polymars3 <- resid(model.ip.mac.polymars3)

modelo_polymars3.ip.mac <- cbind("polymars_pl", round(sum(residual.model.ip.mac.
    polymars3), digits = 2),
                                round(mean(residual.model.ip.mac.polymars3), digits =
    2),
                                round(var(residual.model.ip.mac.polymars3), digits
    =2))

#mars- Multivariate Adaptive Regression Splines###############
model.ip.mac.mars1 = mars(cbind(ip.mac$mcs,ip.mac$model_phy_mac), ip.mac$delay)
model.ip.mac.mars1
summary(model.ip.mac.mars1)
fit.ip.mac.mars1 <- fitted(model.ip.mac.mars1)
RSS.ip.mac.mars1 <-mean((fit.ip.mac.mars1-ip.mac$delay)^2)
RSS.ip.mac.mars1
aic(model.ip.mac.mars1)
residual.model.ip.mac.mars1 <- resid(model.ip.mac.mars1)

modelo_mars1.ip.mac <- cbind("mars_delay", round(sum(residual.model.ip.mac.mars1),
    digits = 2),
                            round(mean(residual.model.ip.mac.mars1), digits = 2),
                            round(var(residual.model.ip.mac.mars1), digits =2))
```

```
model.ip.mac.mars2 = mars(cbind(ip.mac$mcs,ip.mac$model_phy_mac), ip.mac$jitter)
model.ip.mac.mars2
summary(model.ip.mac.mars2)
fit.ip.mac.mars2 <- fitted(model.ip.mac.mars2)
RSS.ip.mac.mars2 <-mean((fit.ip.mac.mars2-ip.mac$jitter)^2)
RSS.ip.mac.mars2
AIC(model.ip.mac.mars2)
residual.model.ip.mac.mars2 <- resid(model.ip.mac.mars2)

modelo_mars2.ip.mac <- cbind("mars_jitter", round(sum(residual.model.ip.mac.mars2),
    digits = 2),
                             round(mean(residual.model.ip.mac.mars2), digits = 2),
                             round(var(residual.model.ip.mac.mars2), digits = 2))

model.ip.mac.mars3 = mars(cbind(ip.mac$mcs,ip.mac$model_phy_mac), ip.mac$pl)
model.ip.mac.mars3
summary(model.ip.mac.mars3)
fit.ip.mac.mars3 <- fitted(model.ip.mac.mars3)
RSS.ip.mac.mars3 <-mean((fit.ip.mac.mars3-ip.mac$pl)^2)
RSS.ip.mac.mars3
AIC(model.ip.mac.mars3)
residual.model.ip.mac.mars3 <- resid(model.ip.mac.mars3)

modelo_mars3.ip.mac <- cbind("mars_pl", round(sum(residual.model.ip.mac.mars3),digits
    = 2),
                             round(mean(residual.model.ip.mac.mars3), digits = 2),
                             round(var(residual.model.ip.mac.mars3), digits = 2))

#cox - Cox survival-Regression Models#########################
model.ip.mac.cox1 <- coxph(Surv(ip.mac$delay)~ip.mac$mcs+ip.mac$model_phy_mac)
summary(model.ip.mac.cox1)
fit.ip.mac.cox1 <-(ip.mac$delay-model.ip.mac.cox1$residuals)
RSS.ip.mac.cox1 <-mean((fit.ip.mac.cox1-ip.mac$delay)^2)
RSS.ip.mac.cox1
AIC(model.ip.mac.cox1)
residual.model.ip.mac.cox1 <- resid(model.ip.mac.cox1)

modelo_cox1.ip.mac <- cbind("cox_delay", round(sum(residual.model.ip.mac.cox1), digits
     = 2),
                            round(mean(residual.model.ip.mac.cox1), digits = 2),
                            round(var(residual.model.ip.mac.cox1), digits =2))

model.ip.mac.cox2 <- coxph(Surv(ip.mac$jitter)~ip.mac$mcs+ip.mac$model_phy_mac)
summary(model.ip.mac.cox2)
names(model.ip.mac.cox2)
fit.ip.mac.cox2 <-(ip.mac$jitter-model.ip.mac.cox2$residuals)
RSS.ip.mac.cox2 <-mean((fit.ip.mac.cox2-ip.mac$jitter)^2)
RSS.ip.mac.cox2
AIC(model.ip.mac.cox2)
residual.model.ip.mac.cox2 <- resid(model.ip.mac.cox2)
```

```
modelo_cox2.ip.mac <- cbind("cox_jitter", round(sum(residual.model.ip.mac.cox2),digits
    = 2),
                            round(mean(residual.model.ip.mac.cox2), digits = 2),
                            round(var(residual.model.ip.mac.cox2), digits = 2))

model.ip.mac.cox3 <- coxph(Surv(ip.mac$pl)~ip.mac$mcs+ip.mac$model_phy_mac)
summary(model.ip.mac.cox3)
names(model.ip.mac.cox3)
fit.ip.mac.cox3 <-(ip.mac$pl-model.ip.mac.cox3$n)
RSS.ip.mac.cox3 <-mean((fit.ip.mac.cox3-ip.mac$mcs)^2)
RSS.ip.mac.cox3
AIC(model.ip.mac.cox3)
residual.model.ip.mac.cox3 <- resid(model.ip.mac.cox3)

modelo_cox3.ip.mac <- cbind("cox_", round(sum(residual.model.ip.mac.cox3),digits = 2),
                            round(mean(residual.model.ip.mac.cox3), digits = 2),
                            round(var(residual.model.ip.mac.cox3), digits = 2))

## MODELOS NO PARAMETRICOS IP-MAC_DELAY ######################
NoParametricos_IP_MAC_Delay <- rbind(cbind ("loess",round(RSS.ip.mac.loess1, digits =
    2)),
                                     cbind ("ppr",round(RSS.ip.mac.ppr1, digits = 2)),
                                     cbind ("locfit",round(RSS.ip.mac.locfit1, digits
    = 2)),
                                     cbind ("gam",round(RSS.ip.mac.gam1, digits = 2)),
                                     cbind ("sm",round(RSS.ip.mac.sm1, digits = 2)),
                                     cbind ("polymars",round(RSS.ip.mac.polymars1,
    digits = 2)),
                                     cbind ("mars",round(RSS.ip.mac.mars1, digits = 2)
    ),
                                     cbind ("cox",round(RSS.ip.mac.cox1, digits = 2)))

colnames(NoParametricos_IP_MAC_Delay)[1] <-"Modelo"
colnames(NoParametricos_IP_MAC_Delay)[2] <-"RRS"
colnames(NoParametricos_IP_MAC_Delay)[1] <-"Modelo"
NoParametricos_IP_MAC_Delay

## MODELOS NO PARAMETRICOS IP-MAC_JITTER ######################
NoParametricos_IP_MAC_Jitter <- rbind(cbind ("loess",round(RSS.ip.mac.loess2, digits =
    2)),
                                      cbind ("ppr",round(RSS.ip.mac.ppr2, digits = 2))
    ,
                                      cbind ("locfit",round(RSS.ip.mac.locfit2, digits
    = 2)),
                                      cbind ("gam",round(RSS.ip.mac.gam2, digits = 2))
    ,
                                      cbind ("sm",round(RSS.ip.mac.sm2, digits = 2)),
                                      cbind ("polymars",round(RSS.ip.mac.polymars2,
    digits = 2)),
```

```
                                    cbind ("mars",round(RSS.ip.mac.mars2, digits =
    2)),
                                    cbind ("cox",round(RSS.ip.mac.cox2, digits = 2))
    )

colnames(NoParametricos_IP_MAC_Jitter)[1] <-"Modelo"
colnames(NoParametricos_IP_MAC_Jitter)[2] <-"RRS"
colnames(NoParametricos_IP_MAC_Jitter)[1] <-"Modelo"
NoParametricos_IP_MAC_Jitter

## MODELOS NO PARAMETRICOS IP-MAC_PL ########################
NoParametricos_IP_MAC_PL <- rbind(cbind ("loess",round(RSS.ip.mac.loess3, digits = 2))
    ,
                                  cbind ("ppr",round(RSS.ip.mac.ppr3, digits = 2)),
                                  cbind ("locfit",round(RSS.ip.mac.locfit3, digits =
    2)),
                                  cbind ("gam",round(RSS.ip.mac.gam3, digits = 2)),
                                  cbind ("sm",round(RSS.ip.mac.sm3, digits = 2)),
                                  cbind ("polymars",round(RSS.ip.mac.polymars3, digits
     = 2)),
                                  cbind ("mars",round(RSS.ip.mac.mars3, digits = 2)),
                                  cbind ("cox",round(RSS.ip.mac.cox3, digits = 2)))

colnames(NoParametricos_IP_MAC_PL)[1] <-"Modelo"
colnames(NoParametricos_IP_MAC_PL)[2] <-"RRS"
colnames(NoParametricos_IP_MAC_PL)[1] <-"Modelo"
NoParametricos_IP_MAC_PL

## TABLA DE RESIDUOS IP-MAC_DELAY ###########################
Residuos_IP_MAC_delay <- rbind.data.frame(modelo_g1.ip.mac,
                                          #modelo_p1.ip.mac,
                                          modelo_loess1.ip.mac,
                                          modelo_ppr1.ip.mac,
                                          modelo_locfit1.ip.mac,
                                          modelo_gam1.ip.mac,
                                          modelo_sm1.ip.mac,
                                          modelo_polymars1.ip.mac,
                                          modelo_mars1.ip.mac,
                                          modelo_cox1.ip.mac)
colnames(Residuos_IP_MAC_delay)[1] <-"Modelo"
colnames(Residuos_IP_MAC_delay)[2] <-"Residuos"
colnames(Residuos_IP_MAC_delay)[3] <-"Media"
colnames(Residuos_IP_MAC_delay)[4] <-"Varianza"

Residuos_IP_MAC_delay

## TABLA DE RESIDUOS IP-MAC_JITTER ##########################
Residuos_IP_MAC_jitter <- rbind.data.frame(modelo_g2.ip.mac,
                                           #modelo_p2.ip.mac,
                                           modelo_loess2.ip.mac,
```

```
                                        modelo_ppr2.ip.mac,
                                        modelo_locfit2.ip.mac,
                                        modelo_gam2.ip.mac,
                                        modelo_sm2.ip.mac,
                                        modelo_polymars2.ip.mac,
                                        modelo_mars2.ip.mac,
                                        modelo_cox2.ip.mac)
colnames(Residuos_IP_MAC_jitter)[1] <-"Modelo"
colnames(Residuos_IP_MAC_jitter)[2] <-"Residuos"
colnames(Residuos_IP_MAC_jitter)[3] <-"Media"
colnames(Residuos_IP_MAC_jitter)[4] <-"Varianza"

Residuos_IP_MAC_jitter

## TABLA DE RESIDUOS IP-MAC_PL ##############################
Residuos_IP_MAC_PL <- rbind.data.frame(modelo_g3.ip.mac,
                                       #modelo_p3.ip.mac,
                                       modelo_loess3.ip.mac,
                                       modelo_ppr3.ip.mac,
                                       modelo_locfit3.ip.mac,
                                       modelo_gam3.ip.mac,
                                       modelo_sm3.ip.mac,
                                       modelo_polymars3.ip.mac,
                                       modelo_mars3.ip.mac,
                                       modelo_cox3.ip.mac)
colnames(Residuos_IP_MAC_PL)[1] <-"Modelo"
colnames(Residuos_IP_MAC_PL)[2] <-"Residuos"
colnames(Residuos_IP_MAC_PL)[3] <-"Media"
colnames(Residuos_IP_MAC_PL)[4] <-"Varianza"

Residuos_IP_MAC_PL

## TABLA DE KS IP-MAC_DELAY #################################
KS_IP_MAC_Delay <- rbind.data.frame(ks.test(ip.mac$delay,fit.ip.mac.g1),
                                    #ks.test(ip.mac$delay,fit.ip.mac.p1),
                                    ks.test(ip.mac$delay,fit.ip.mac.loess1),
                                    ks.test(ip.mac$delay,fit.ip.mac.ppr1),
                                    ks.test(ip.mac$delay,fit.ip.mac.locfit1),
                                    ks.test(ip.mac$delay,fit.ip.mac.gam1),
                                    ks.test(ip.mac$delay,fit.ip.mac.sm1),
                                    ks.test(ip.mac$delay,fit.ip.mac.polymars1),
                                    ks.test(ip.mac$delay,fit.ip.mac.mars1),
                                    ks.test(ip.mac$delay,fit.ip.mac.cox1))
colnames(KS_IP_MAC_Delay)[1] <-"D"
colnames(KS_IP_MAC_Delay)[2] <-"P-VALUE"
KS_IP_MAC_Delay$alternative <- NULL
KS_IP_MAC_Delay$method <- NULL

KS_IP_MAC_Delay
```

```
## TABLA DE KS IP-MAC_JITTER ##################################
KS_IP_MAC_Jitter <- rbind.data.frame(ks.test(ip.mac$jitter,fit.ip.mac.g2),
                                     #ks.test(ip.mac$jitter,fit.ip.mac.p2),
                                     ks.test(ip.mac$jitter,fit.ip.mac.loess2),
                                     ks.test(ip.mac$jitter,fit.ip.mac.ppr2),
                                     ks.test(ip.mac$jitter,fit.ip.mac.locfit2),
                                     ks.test(ip.mac$jitter,fit.ip.mac.gam2),
                                     ks.test(ip.mac$jitter,fit.ip.mac.sm2),
                                     ks.test(ip.mac$jitter,fit.ip.mac.polymars2),
                                     ks.test(ip.mac$jitter,fit.ip.mac.mars2),
                                     ks.test(ip.mac$jitter,fit.ip.mac.cox2))
colnames(KS_IP_MAC_Jitter)[1] <-"D"
colnames(KS_IP_MAC_Jitter)[2] <-"P-VALUE"
KS_IP_MAC_Jitter$alternative <- NULL
KS_IP_MAC_Jitter$method <- NULL

KS_IP_MAC_Jitter

## TABLA DE KS IP-MAC_PL ######################################
KS_IP_MAC_PL <- rbind.data.frame(ks.test(ip.mac$pl,fit.ip.mac.g3),
                                 #ks.test(ip.mac$pl,fit.ip.mac.p3),
                                 ks.test(ip.mac$pl,fit.ip.mac.loess3),
                                 ks.test(ip.mac$pl,fit.ip.mac.ppr3),
                                 ks.test(ip.mac$pl,fit.ip.mac.locfit3),
                                 ks.test(ip.mac$pl,fit.ip.mac.gam3),
                                 ks.test(ip.mac$pl,fit.ip.mac.sm3),
                                 ks.test(ip.mac$pl,fit.ip.mac.polymars3),
                                 ks.test(ip.mac$pl,fit.ip.mac.mars3),
                                 ks.test(ip.mac$pl,fit.ip.mac.cox3))
colnames(KS_IP_MAC_PL)[1] <-"D"
colnames(KS_IP_MAC_PL)[2] <-"P-VALUE"
KS_IP_MAC_PL$alternative <- NULL
KS_IP_MAC_PL$method <- NULL

KS_IP_MAC_PL
```

- Script con la función E-MODEL

```
####
# Script de calculo estimado de calidad de experiencia segun E-Model y conversion a
    MOS
# Preparado por:  Joaquin Chavez B.
# Modificado por: Johanna Ortega B.
# Fecha:          03.11.2016
####

# IMPORTAR DATOS ##############################################
mos.ip.mejormodelo_delay <- predict(<mejor modelo de delay IP-MAC>)
mos.ip.mejormodelo_jitter <- predict(<mejor modelo de jitter IP-MAC>)
mos.ip.mejormodelo_pl <- predict(<mejor modelo de packetloss IP-MAC>)

Flow_Monitor_Stats <- read.table("F:/Flow_Monitor_Stats_<scheduler>.txt",
                                 header=TRUE, sep=",", na.strings="NA", dec=".", strip
    .white=TRUE)

mos.ip.flowmonitor_delay  <- Flow_Monitor_Stats$delayAvg
mos.ip.flowmonitor_jitter <- Flow_Monitor_Stats$jitterAvg
mos.ip.flowmonitor_pl <- Flow_Monitor_Stats$packetLossPerc

# FUNCION MOS #################################################
# Funcion para convertir R a MOS.
RtoMOS <- function(R)
{
  MOS <- c()
  for(i in 1:length(R))
  {
    if(R[i] < 0)
    {
      MOS[i] = 1
    }
    else if(R[i] > 0 && R[i] < 100)
    {
      MOS[i] = 1 + 0.035 * R[i] + R[i] * (R[i] - 60) * (100 - R[i]) * 7 * 0.000001
    }
    else if(R[i] > 100)
    {
      MOS[i] = 4.5
    }
    else
    {
      MOS[i] = NULL
    }
  }
  return(MOS)
}
```

```
# CALCULO MOS CON MEJOR MODELO ##############################
# Codigo que extrae la informacion de los flujos IP para calcular R.
m_loss1 <- c()
m_mos1 <- c()

# AMR
Bpl = 10
Ie = 5

m_delay1 = c()
m_jitter1 = c()

m_delay1 <- mos.ip.mejormodelo_delay
m_jitter1 <-mos.ip.mejormodelo_jitter
m_loss1 <- mos.ip.mejormodelo_pl

burstr = 1
Ie_eff1 = Ie + (95 - Ie)*m_loss1/((m_loss1/burstr) + Bpl)

# Id Parameter calculation
jitter1 = m_jitter1 # Miliseconds
jitter1[is.na(jitter1)] <- 9999999
Drtcp1 = m_delay1# Miliseconds
Drtcp1[is.na(Drtcp1)] <- 9999999
Dj1 = 20 + 0.9 * jitter1 # 20ms is codec frame size in AMR
Dj1[is.na(Dj1)] <- 9999999
for (i in 1:length(Dj1))
{
  if(Dj1[i] > 300)  {
    Dj1[i] = 300
  }

}
# Estimation of De
De = 1.2 * 0.020 # Encoding delay is +20% of frame size
# Da and Dr is set to zero
Dr = 0

# Calculation of T
T1 = Drtcp1 + Dj1 + De + Dr

# Id Calculation
TERV1 = 65 + (6 * exp(-0.3 * T1^2 )) - 40 * log10((1 + T1/10)/(1 + T1/150))

# Idte
Re1 = 80 + 2.5 * (TERV1 - 14)
Roe1 = -1.5 * (-61.18 - 2)
Xdt1 = (Roe1 - Re1) / 2
Idte1 = Xdt1 + sqrt(Xdt1^2 + 100)
Idte1 = (Idte1 - 1) * (1 - exp(-T1))
```

```
# Idle
Tr1 = 2*Drtcp1+Dj1+De
Rle1 = 1228.5 * (Tr1 + 1)^(-0.25)
Xdl1 = (94.77 - Rle1) / 2
Idle1 = Xdl1 + sqrt(Xdl1^2 + 169)

# Idd
Ta1 = Drtcp1+Dj1+De
Idd1 = 0
X1 <- c()
for (i in 1:length(Ta1))
{
  if (Ta1[i] > 100)
  {
    X1[i] = log(Ta1[i]/100)/log(2)
    Idd1[i] = 25.0 * ((1.0 + X1[i]^6.0)^(1.0/6.0) - 3.0 * ((1.0 + (X1[i]/3.0)^6.0)
    ^(1.0 / 6.0)) + 2.0)
  }
  else
  {
    Idd1[i] = 0
  }
}

Id1 = Idte1 + Idle1 + Idd1

# R value calculation
R1 = 93.34 - Id1 - Ie_eff1

# Conversion a MOS
modelo_mos = RtoMOS(R1)
MOS_modelo <- cbind(modelo_mos)

# CALCULO MOS CON FLOWMONITOR ################################
m_loss <- c()
m_mos <- c()

# AMR
Bpl = 10
Ie = 5

m_delay = c()
m_jitter = c()

Flow_Monitor_Stats <- Flow_Monitor_Stats[Flow_Monitor_Stats$destinationPort == 1234,]
row.names(Flow_Monitor_Stats) <- NULL
m_delay <- Flow_Monitor_Stats$delayAvg
m_jitter <- Flow_Monitor_Stats$jitterAvg
m_loss <- Flow_Monitor_Stats$packetLossPerc
```

```
burstr = 1
Ie_eff = Ie + (95 - Ie)*m_loss/((m_loss/burstr) + Bpl)

# Id Parameter calculation
jitter = m_jitter # Miliseconds
jitter[is.na(jitter)] <- 9999999
Drtcp = m_delay# Miliseconds
Drtcp[is.na(Drtcp)] <- 9999999
Dj = 20 + 0.9 * jitter # 20ms is codec frame size in AMR
Dj[is.na(Dj)] <- 9999999
for (i in 1:length(Dj))
{
  if(Dj[i] > 300)  {
    Dj[i] = 300
  }

}
# Estimation of De
De = 1.2 * 0.020 # Encoding delay is +20% of frame size
# Da and Dr is set to zero
Dr = 0

# Calculation of T
T = Drtcp + Dj + De + Dr

# Id Calculation
TERV = 65 + (6 * exp(-0.3 * T^2 )) - 40 * log10((1 + T/10)/(1 + T/150))

# Idte
Re = 80 + 2.5 * (TERV - 14)
Roe = -1.5 * (-61.18 - 2)
Xdt = (Roe - Re) / 2
Idte = Xdt + sqrt(Xdt^2 + 100)
Idte = (Idte - 1) * (1 - exp(-T))

# Idle
Tr = 2*Drtcp+Dj+De
Rle = 1228.5 * (Tr + 1)^(-0.25)
Xdl = (94.77 - Rle) / 2
Idle = Xdl + sqrt(Xdl^2 + 169)

# Idd
Ta = Drtcp+Dj+De
Idd = 0
X <- c()
for (i in 1:length(Ta))
{
  if (Ta[i] > 100)
  {
```

```
    X[i] = log(Ta[i]/100)/log(2)
    Idd[i] = 25.0 * ((1.0 + X[i]^6.0)^(1.0/6.0) - 3.0 * ((1.0 + (X[i]/3.0)^6.0)^(1.0 /
     6.0)) + 2.0)
  }
  else
  {
    Idd[i] = 0
  }
}

Id = Idte + Idle + Idd

# R value calculation
R = 93.34 - Id - Ie_eff

# Conversion a MOS
flowmonitor_mos = RtoMOS(R)
MOS_flowmonitor <- flowmonitor_mos

# GUARDAR DATOS ################################################
write.csv(MOS_modelo,file = "F:/MOS(Modelo)_<scheduler>.txt", row.names=TRUE)
write.csv(MOS_flowmonitor,file = "F:/MOS(Flowmonitor)_<scheduler>.txt", row.names=TRUE
    )

# DIBUJO HISTOGRAMA ############################################
# Histograma de MOS estimado para los flujos VoIP.
path <- file.path("F:/", paste("MOS(MODELO)_<scheduler>", ".jpeg", sep = ""))
jpeg(file=path, width = 600, height = 400)
hist(MOS_modelo,
     col ="darkorange1",
     main = " ",
     xlab = "MOS",
     ylab = "Frecuencia")
grid()
legend("topright",
       c(paste("Media =", as.name(mean(MOS_modelo))),
         paste("Desv.Estandar =", as.name(sd(MOS_modelo)))
       )
)
dev.off()

# Histograma de MOS estimado para los flujos VoIP.
path <- file.path("F:/", paste("MOS(FLOWMONITOR)_<scheduler>", ".jpeg", sep = ""))
jpeg(file=path, width = 600, height = 400)
hist(MOS_flowmonitor,
     col ="mediumblue",
     main = " ",
     xlab = "MOS",
     ylab = "Frecuencia")
grid()
```

```
legend("topleft",
       c(paste("Media =", as.name(mean(MOS_flowmonitor))),
         paste("Desv.Estandar =", as.name(sd(MOS_flowmonitor)))
       )
)
dev.off()

# TABLA DE KS MOS-IP ########################################
KS_MOS_IP_DELAY <- ks.test(mos.ip.mejormodelo_delay, mos.ip.flowmonitor_delay)
KS_MOS_IP_DELAY
KS_MOS_IP_JITTER <- ks.test(mos.ip.mejormodelo_jitter, mos.ip.flowmonitor_jitter)
KS_MOS_IP_JITTER
KS_MOS_IP_PL <- ks.test(mos.ip.mejormodelo_pl, mos.ip.flowmonitor_pl)
KS_MOS_IP_PL
```

Anexo C

Script en NS-3

- Input Default

```
default ns3::LteSpectrumPhy::DataErrorModelEnabled "true"
default ns3::LteSpectrumPhy::CtrlErrorModelEnabled "true"

default ns3::LteEnbPhy::TxPower "40"
default ns3::LteEnbPhy::NoiseFigure "5"
default ns3::LteEnbPhy::MacToChannelDelay "2"
default ns3::LteEnbPhy::UeSinrSamplePeriod "1"
default ns3::LteEnbPhy::InterferenceSamplePeriod "1"

default ns3::LteUePhy::TxPower "10"
default ns3::LteUePhy::NoiseFigure "9"
default ns3::LteUePhy::TxMode1Gain "0"
default ns3::LteUePhy::TxMode2Gain "4.2"
default ns3::LteUePhy::TxMode3Gain "-2.8"
default ns3::LteUePhy::TxMode4Gain "0"
default ns3::LteUePhy::TxMode5Gain "0"
default ns3::LteUePhy::TxMode6Gain "0"
default ns3::LteUePhy::TxMode7Gain "0"

default ns3::LteEnbRrc::SrsPeriodicity "320"

default ns3::LteEnbNetDevice::UlBandwidth "25"
default ns3::LteEnbNetDevice::DlBandwidth "25"
default ns3::LteEnbNetDevice::DlEarfcn "100"
default ns3::LteEnbNetDevice::UlEarfcn "18100"

default ns3::LteHelper::PathlossModel "ns3::FriisPropagationLossModel"
default ns3::LteHelper::FadingModel ""
default ns3::LteHelper::UsePdschForCqiGeneration "true"

default ns3::RadioBearerStatsCalculator::DlRlcOutputFilename "DlRlcStats.txt"
default ns3::RadioBearerStatsCalculator::UlRlcOutputFilename "UlRlcStats.txt"
```

```
default ns3::RadioBearerStatsCalculator::DlPdcpOutputFilename "DlPdcpStats.txt"
default ns3::RadioBearerStatsCalculator::UlPdcpOutputFilename "UlPdcpStats.txt"

default ns3::PhyStatsCalculator::DlRsrpSinrFilename "DlRsrpSinrStats.txt"
default ns3::PhyStatsCalculator::UlSinrFilename "UlSinrStats.txt"
default ns3::PhyStatsCalculator::UlInterferenceFilename "UlInterferenceStats.txt"

default ns3::MacStatsCalculator::DlOutputFilename "DlMacStats.txt"
default ns3::MacStatsCalculator::UlOutputFilename "UlMacStats.txt"

default ns3::PhyTxStatsCalculator::DlTxOutputFilename "DlTxPhyStats.txt"
default ns3::PhyTxStatsCalculator::UlTxOutputFilename "UlTxPhyStats.txt"
default ns3::PhyRxStatsCalculator::DlRxOutputFilename "DlRxPhyStats.txt"
default ns3::PhyRxStatsCalculator::UlRxOutputFilename "UlRxPhyStats.txt"
```

- Sript de la Simulación de Schedulers

```
#include "ns3/lte-helper.h"
#include "ns3/epc-helper.h"
#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/ipv4-global-routing-helper.h"
#include "ns3/internet-module.h"
#include "ns3/mobility-module.h"
#include "ns3/lte-module.h"
#include "ns3/applications-module.h"
#include "ns3/point-to-point-helper.h"
#include "ns3/config-store.h"
#include "ns3/flow-monitor-module.h"
#include <ns3/flow-monitor-helper.h>
#include "ns3/http-client.h"
#include "ns3/http-server.h"
#include "ns3/http-client-server-helper.h"

using namespace ns3;

// Declaraciones iniciales.
void ThroughputMonitor (FlowMonitorHelper *fmhelper, Ptr<FlowMonitor> flowMon);

vector<vector<double> > readCordinatesFile (std::string node_coordinates_file_name);
vector<uint32_t> readNumUeFile (std::string node_num_ue_file_name);

NS_LOG_COMPONENT_DEFINE ("tests");
int
main (int argc, char *argv[])
 {
  // Tiempo de simulacion por defecto.
  double simTime = 5;

  // Se anadieron los siguientes parametros para la simulacion de trafico VoIP.

  // Codec por defecto: AMR 0.
  // Tamano de Payload de voz: voicePayload = 32;
  // Tasa de paquetes por segundo. Depende de codec utilizado.
  int pps = 50;

  // Tasa de transferencia de trafico base en la bajada.
  std::string tasaTraficoBaseDl ("1Mb/s");
  // Tasa de transferencia de trafico base en la subida.
  std::string tasaTraficoBaseUl ("200Kb/s");
  // Numero de Ue en condicion base de trafico.
  double numUeBase = 20;
  // Declaracion de archivo con coordenadas para los eNodeB's.
  std::string node_coordinates_file_name ("node_coordinates.txt");
```

```
std::string node_num_ue_file_name ("node_num_ue.txt");

CommandLine cmd;
cmd.AddValue("simTime", "Duracion total de la simulacion [s])", simTime);
cmd.AddValue("tasaTraficoBaseDl", "Tasa de transferencia asociada al trafico base en
   el DL. Ej: 1Mb/s", tasaTraficoBaseDl);
cmd.AddValue("tasaTraficoBaseUl", "Tasa de transferencia asociada al trafico base en
   el UL. Ej: 200Kb/s", tasaTraficoBaseUl);
cmd.AddValue("numUeBase", "Numero de UE en condicion base de trafico. Ej: 20",
  numUeBase);
cmd.Parse(argc, argv);

double ber = 0.000001;
Config::SetDefault ("ns3::LteAmc::AmcModel", EnumValue (LteAmc::PiroEW2010));
Config::SetDefault ("ns3::LteAmc::Ber", DoubleValue (ber));
Config::SetDefault ("ns3::LteEnbRrc::EpsBearerToRlcMapping",EnumValue(LteEnbRrc::
  RLC_UM_ALWAYS));


Ptr<LteHelper> lteHelper = CreateObject<LteHelper> ();
Ptr<PointToPointEpcHelper> epcHelper = CreateObject<PointToPointEpcHelper> ();

lteHelper->SetEpcHelper (epcHelper);

//Ingreso de parametros a traves de archivo de configuracion.
ConfigStore inputConfig;
inputConfig.ConfigureDefaults();

cmd.Parse(argc, argv);

  //Configuracion del Scheduler LTE MAC
lteHelper->SetSchedulerType ("ns3::<Scheduler>");
//lteHelper->SetSchedulerType ("ns3::RrFfMacScheduler");
//lteHelper->SetSchedulerType ("ns3::PfFfMacScheduler");
//lteHelper->SetSchedulerType ("ns3::FdMtFfMacScheduler");
//lteHelper->SetSchedulerType ("ns3::TdMtFfMacScheduler");
//lteHelper->SetSchedulerType ("ns3::TtaFfMacScheduler");
//lteHelper->SetSchedulerType ("ns3::FdTbfqFfMacScheduler");
//lteHelper->SetSchedulerType ("ns3::TdTbfqFfMacScheduler");
//lteHelper->SetSchedulerType ("ns3::PssFfMacScheduler");
//lteHelper->SetSchedulerType ("ns3::CqaFfMacScheduler");

NS_LOG_UNCOND ("Configurando LTE MAC Scheduler");

//Tamano de paquete = Payload de Voz + MP + RTP Header. No esta implementada la
  compresion de //header.
int dlPacketSize = 32 + 6 + 12;

// Tasa de transmision de datos. Depende de codec utilizado.
DataRate dlDataRate = DataRate(dlPacketSize * 8 * pps);
```

```
NS_LOG_UNCOND ("Tiempo de simulacion: " << simTime << "[seg]");
NS_LOG_UNCOND ("Tasa de paquetes por segundo: " << pps << "[paquetes/seg]");

Ptr<Node> pgw = epcHelper->GetPgwNode ();
NS_LOG_UNCOND ("Creando red de Internet");
// Crear un Host remoto para utilizar como servidor HTTP.
NodeContainer remoteHostContainer;
remoteHostContainer.Create (1);
Ptr<Node> remoteHost = remoteHostContainer.Get (0);
InternetStackHelper internet;
internet.Install (remoteHostContainer);

// Crear conexion a internet.
PointToPointHelper p2ph;
p2ph.SetDeviceAttribute ("DataRate", DataRateValue (DataRate ("100Gb/s")));
p2ph.SetDeviceAttribute ("Mtu", UintegerValue (1500));
p2ph.SetChannelAttribute ("Delay", TimeValue (Seconds (0.010)));
NetDeviceContainer internetDevices = p2ph.Install (pgw, remoteHost);
Ipv4AddressHelper ipv4h;
ipv4h.SetBase ("1.0.0.0", "255.0.0.0");
Ipv4InterfaceContainer internetIpIfaces = ipv4h.Assign (internetDevices);
// interface 0 is localhost, 1 is the p2p device
Ipv4Address remoteHostAddr = internetIpIfaces.GetAddress (1);

Ipv4StaticRoutingHelper ipv4RoutingHelper;
Ptr<Ipv4StaticRouting> remoteHostStaticRouting = ipv4RoutingHelper.GetStaticRouting
  (remoteHost->GetObject<Ipv4> ());
remoteHostStaticRouting->AddNetworkRouteTo (Ipv4Address ("7.0.0.0"), Ipv4Mask ("
  255.0.0.0"), 1);

// Creacion de containers para los nodos UE y nodos eNb
NodeContainer ueNodes;
NodeContainer enbNodes;

// Creacion de contenedor para UE's representativos de trafico base.
NodeContainer ueTrafico;

NS_LOG_UNCOND ("Leyendo archivo de coordenadas y bandas de operacion de los eNodeB."
  );
// ---Lectura de archivos de coordenadas
     vector<vector<double> > coord_array;
coord_array = readCordinatesFile (node_coordinates_file_name);
int numberOfeNodeB = coord_array.size ();
// ---Fin de Lectura de archivos de coordenadas
NS_LOG_UNCOND ("Leyendo archivo de cantidad de UE por celda.");
// ---Lectura de archivo de numero de UE por celda
vector<uint32_t> num_ue_array;
num_ue_array = readNumUeFile (node_num_ue_file_name);
uint32_t numberOfUe = 0;
```

```
for (size_t m = 0; m < num_ue_array.size(); m++)
{
  numberOfUe += num_ue_array[m];
}
// ---Fin de Lectura de archivo de numero de UE por celda
enbNodes.Create(numberOfeNodeB);

// Configuraciones de movilidad

// ---Setear posiciones para los nodos

// eNodeB's
NS_LOG_UNCOND ("Ubicando " << numberOfeNodeB << " eNodeB's para la simulacion.");
MobilityHelper eNodeMobility;
NetDeviceContainer enbLteDevs = NetDeviceContainer();

Ptr<ListPositionAllocator> positionAlloc = CreateObject<ListPositionAllocator> ();
int nCel=0;
double orientation = 0;

// Ciclo que ubica celdas y UEs representativos de trafico base para cada una.
for (size_t m = 0; m < coord_array.size (); m++)
{
  if (nCel == 0)
  {
    orientation = 0;
    nCel++;
  }
  else if (nCel == 1)
  {
    orientation = 120;
    nCel++;
  }
  else if (nCel == 2)
  {
    orientation = -120;
    nCel = 0;
  }

  positionAlloc->Add (Vector (coord_array[m][0], coord_array[m][1], coord_array[m
  ][2]));
  Ptr<Node> n0 = enbNodes.Get (m);
  Ptr<ConstantPositionMobilityModel> nLoc =  n0->GetObject<
  ConstantPositionMobilityModel> ();

  if (nLoc == 0)
  {
    nLoc = CreateObject<ConstantPositionMobilityModel> ();
    n0->AggregateObject (nLoc);
  }
```

```
  Vector nVec (coord_array[m][0], -coord_array[m][1], coord_array[m][2]);
  nLoc->SetPosition (nVec);

  lteHelper->SetEnbAntennaModelType ("ns3::CosineAntennaModel");
  lteHelper->SetEnbAntennaModelAttribute ("Orientation", DoubleValue(orientation));

  // Seteo de banda de operacion de la celda. Frecuencia en formato EARFCN y ancho
  de banda en numero de PRBs.

  lteHelper->SetEnbDeviceAttribute ("DlEarfcn", UintegerValue(coord_array[m][3]));
  lteHelper->SetEnbDeviceAttribute ("DlBandwidth", UintegerValue(coord_array[m][4]))
  ;
  lteHelper->SetEnbDeviceAttribute ("UlEarfcn", UintegerValue(coord_array[m][5]));
  lteHelper->SetEnbDeviceAttribute ("UlBandwidth", UintegerValue(coord_array[m][6]))
  ;

  enbLteDevs.Add (lteHelper->InstallEnbDevice (n0));
}

eNodeMobility.SetPositionAllocator (positionAlloc);
eNodeMobility.Install (enbNodes);

// UE's

NetDeviceContainer ueLteDevs = NetDeviceContainer();
NetDeviceContainer ueBaseLteDevs = NetDeviceContainer();
MobilityHelper ueNodeMobility;
Ptr<RandomDiscPositionAllocator> uePositionAlloc = CreateObject<
  RandomDiscPositionAllocator> ();

for (size_t m = 0; m < num_ue_array.size (); m++)
{
  NS_LOG_UNCOND ("Posicionando " << num_ue_array[m] << " UE's en torno a la celda "
  << m << ".");

  NodeContainer uesPerCell = NodeContainer();

  NodeContainer ueTraficoAux = NodeContainer();

  uesPerCell.Create(num_ue_array[m]);

  // Crear nodo de trafico representativo
  ueTraficoAux.Create(1);
  // Agregar nodo de trafico representativo a contenedor de todos los nodos
  representativos.
  ueTrafico.Add(ueTraficoAux);

  // Ubicar los UE aleatoriamente dentro de un disco de radio 10-250 [m] en torno a
  el eNodeB.
```

```
  ueNodeMobility.SetPositionAllocator("ns3::RandomDiscPositionAllocator",
                      "X", DoubleValue(coord_array[m][0]),
                      "Y", DoubleValue(coord_array[m][1]),
                      "Rho", StringValue("ns3::UniformRandomVariable[Min=10|Max=250]
  "));
  // Setear movilidad Random Walk a los UE.
  // Los limites del rectangulo deben ser mayores que las posibles posiciones para
  los UE.
  ueNodeMobility.SetMobilityModel("ns3::RandomWalk2dMobilityModel", "Bounds",
  RectangleValue(Rectangle(0,20000,0,20000)));
  ueNodeMobility.Install(uesPerCell);
  ueNodeMobility.Install(ueTraficoAux);

  ueNodes.Add (uesPerCell);
  // Instalar LTE devices con su banda de operacion correspondiente en los nodos y
  agregarlos al contenedor de LTE devices para los UE.
  lteHelper->SetUeDeviceAttribute ("DlEarfcn", UintegerValue(coord_array[m][3]));
  ueLteDevs.Add (lteHelper->InstallUeDevice (uesPerCell));
  ueBaseLteDevs.Add (lteHelper->InstallUeDevice (ueTrafico));

}

// ---Fin de Setear posiciones para los nodos

// Instalar el stack de protocolos de internet en los nodos UE.
internet.Install (ueNodes);
internet.Install (ueTrafico);
Ipv4InterfaceContainer ueIpIface;
Ipv4InterfaceContainer ueIpIfaceBase;
// Asignar IP's a los UE.
NS_LOG_UNCOND ("Asignando direcciones IP a los UE.");
ueIpIface = epcHelper->AssignUeIpv4Address (NetDeviceContainer (ueLteDevs));
ueIpIfaceBase = epcHelper->AssignUeIpv4Address (NetDeviceContainer (ueBaseLteDevs));
// Definir puertos para aplicaciones VoIP y HTTP
NS_LOG_UNCOND ("Definiendo puertos para aplicaciones");
// Puerto por defecto de subida para aplicaciones On-Off: 49153
// Puerto de bajada On-Off.
uint16_t dlPort = 1234;
uint16_t ulPort = 2000;

// Puerto de subida para app de trafico base.
uint16_t ulBasePort = 8000;
uint16_t dlBasePort = 8000;
// Generar filtro para clasificar paquetes VoIP y posteriormente enviarlos a traves
  de distinto eps bearer.

Ptr<EpcTft> voipTft = Create<EpcTft> ();

EpcTft::PacketFilter dlpf;
dlpf.localPortStart = dlPort;
```

```
dlpf.localPortEnd = dlPort;
voipTft->Add (dlpf);

EpcTft::PacketFilter ulpf;
ulpf.remotePortStart = dlPort;
ulpf.remotePortEnd = dlPort;
voipTft->Add (ulpf);

NS_LOG_UNCOND ("Activando EPS Bearer dedicado para VoIP. (GBR_CONV_VOICE)");

EpsBearer voipBearer (EpsBearer::GBR_CONV_VOICE);

for (uint32_t u = 0; u < ueNodes.GetN (); ++u)
{
  Ptr<Node> ueNode = ueNodes.Get (u);
  // Set the default gateway for the UE
  Ptr<Ipv4StaticRouting> ueStaticRouting = ipv4RoutingHelper.GetStaticRouting (
  ueNode->GetObject<Ipv4> ());
  ueStaticRouting->SetDefaultRoute (epcHelper->GetUeDefaultGatewayAddress (), 1);

  // Activar diferentes bearers para cada servicio utilizando filtros definidos
  anteriormente.
  lteHelper->ActivateDedicatedEpsBearer (ueLteDevs.Get(u), voipBearer, voipTft);
}

// Permite el attachment automatico de los UE.
NS_LOG_UNCOND ("Habilitando attachment automatico.");
lteHelper->Attach (ueLteDevs);
lteHelper->Attach (ueBaseLteDevs);

// Habilitar interfaz X2 entre eNodeB's.
NS_LOG_UNCOND ("Habilitando interfaz X2.");
lteHelper->AddX2Interface (enbNodes);

// Instalar e inciar aplicaciones en los UE y servidores.
NS_LOG_UNCOND ("Instalando aplicaciones en UE's.");

ApplicationContainer serverApps;
ApplicationContainer clientApps;

ApplicationContainer baseTrafficClientApps;
ApplicationContainer baseTrafficServerApps;
for (uint32_t u = 0; u < ueTrafico.GetN (); ++u)
{
  ++ulBasePort;

  // Aplicaciones de trafico base.
  PPBPHelper ppbpDl=PPBPHelper("ns3::UdpSocketFactory",InetSocketAddress (
  ueIpIfaceBase.GetAddress (u), dlBasePort));
  ppbpDl.SetAttribute ("BurstIntensity", DataRateValue (DataRate (tasaTraficoBaseDl)
```

```
  ));
  ppbpDl.SetAttribute ("MeanBurstArrivals", RandomVariableValue (ConstantVariable (
  numUeBase)));
  baseTrafficClientApps.Add (ppbpDl.Install (remoteHost));

  PacketSinkHelper dlPacketSinkHelperBase ("ns3::UdpSocketFactory",
  InetSocketAddress (ueIpIface.GetAddress (u), dlPort));
  baseTrafficServerApps.Add (dlPacketSinkHelperBase.Install (ueNodes.Get(u)));

  PPBPHelper ppbpUl=PPBPHelper("ns3::UdpSocketFactory",InetSocketAddress (
  remoteHostAddr, ulBasePort));
  ppbpUl.SetAttribute ("BurstIntensity", DataRateValue (DataRate (tasaTraficoBaseUl)
  ));
  ppbpUl.SetAttribute ("MeanBurstArrivals", RandomVariableValue (ConstantVariable (
  numUeBase)));
  baseTrafficClientApps.Add (ppbpUl.Install (ueTrafico.Get(u)));

  PacketSinkHelper ulPacketSinkHelperBase ("ns3::UdpSocketFactory",
  InetSocketAddress (remoteHostAddr, ulBasePort));
  baseTrafficServerApps.Add (dlPacketSinkHelperBase.Install (remoteHost));
}

baseTrafficClientApps.Start (Seconds(0.01));
baseTrafficClientApps.Stop (Seconds(simTime));
baseTrafficServerApps.Start (Seconds (0.01));
baseTrafficServerApps.Stop (Seconds(simTime));

for (uint32_t u = 0; u < ueNodes.GetN (); ++u)
{
  //++ulHttpPort;
  ++ulPort;
  // Aplicaciones que modelan el trafico VoIP.
  OnOffHelper onoff=OnOffHelper("ns3::UdpSocketFactory",InetSocketAddress (ueIpIface
  .GetAddress (u), dlPort));
  onoff.SetAttribute ("OnTime", StringValue("ns3::ExponentialRandomVariable[Mean
  =0.352]"));
  onoff.SetAttribute ("OffTime", StringValue("ns3::ExponentialRandomVariable[Mean
  =0.65]"));
  onoff.SetAttribute("PacketSize", UintegerValue(dlPacketSize));
  onoff.SetAttribute("DataRate", DataRateValue(dlDataRate));
  clientApps.Add (onoff.Install (ueNodes.Get(ueNodes.GetN() - u - 1)));

  PacketSinkHelper dlPacketSinkHelper ("ns3::UdpSocketFactory", InetSocketAddress (
  ueIpIface.GetAddress (u), dlPort));
  serverApps.Add (dlPacketSinkHelper.Install (ueNodes.Get(u)));


}

clientApps.Start (Seconds(1));
```

```
  clientApps.Stop(Seconds(simTime));
  serverApps.Start (Seconds (1));

  // Habilitar registro de resultados en archivos de texto.
  NS_LOG_UNCOND ("Habilitando Tracing PHY, MAC, PDCP, RLC.");
  lteHelper->EnableTraces ();

  Ptr<RadioBearerStatsCalculator> rlcStats = lteHelper->GetRlcStats ();
    rlcStats->SetAttribute ("EpochDuration", TimeValue (Seconds (simTime)));

  // Uncomment to enable PCAP tracing
  // NS_LOG_UNCOND ("Habilitando PCAP Tracing.");
  // p2ph.EnablePcapAll("test");

  // Setear tiempo maximo de simulacion.
  Simulator::Stop(Seconds(simTime));

  // Se agrega monitor de flujo para hacer analisis posterior de delay, packet loss,
    jitter, etc.
  // Flow monitor
  NS_LOG_UNCOND ("Habilitando Flowmonitor.");
  NodeContainer monitoredNodes;
  monitoredNodes.Add (ueNodes);
  monitoredNodes.Add (remoteHost);

  Ptr<FlowMonitor> flowMonitor;
  FlowMonitorHelper flowHelper;
  flowHelper.SetMonitorAttribute("MaxPerHopDelay", TimeValue(Seconds(1.0)));
  flowMonitor = flowHelper.Install(monitoredNodes);

  // Desplegar informacion de simulacion en consola.
  NS_LOG_UNCOND ("Iniciando simulacion. Esta puede tomar varias horas. Por favor
    espere...");
  Simulator::Run();

  // Exportar datos de monitoreo a archivo xml.
  NS_LOG_UNCOND ("Exportando datos de flujo de paquetes.");
  flowMonitor->SerializeToXmlFile("Flowmon_VoIP.xml", false, true);

  NS_LOG_UNCOND ("Simulacion finalizada con exito.");
  Simulator::Destroy();
  return 0;
}

// Funcion de monitoreo de flujo para consola.

void ThroughputMonitor (FlowMonitorHelper *fmhelper, Ptr<FlowMonitor> flowMon)
{
  std::map<FlowId, FlowMonitor::FlowStats> flowStats = flowMon->GetFlowStats();
  Ptr<Ipv4FlowClassifier> classing = DynamicCast<Ipv4FlowClassifier> (fmhelper->
```

```
    GetClassifier());

  for (std::map<FlowId, FlowMonitor::FlowStats>::const_iterator stats = flowStats.
    begin (); stats != flowStats.end (); ++stats)
  {
    Ipv4FlowClassifier::FiveTuple fiveTuple = classing->FindFlow (stats->first);
    std::cout<<"Flow ID     : " << stats->first <<" ; "<< fiveTuple.sourceAddress <<"
    -----> "<<fiveTuple.destinationAddress<<std::endl;
    std::cout<<"Tx Packets = " << stats->second.txPackets<<std::endl;
    std::cout<<"Rx Packets = " << stats->second.rxPackets<<std::endl;
    std::cout<<"Rx Bytes = " << stats->second.rxBytes<<std::endl;
    std::cout<<"Duration    : "<<stats->second.timeLastRxPacket.GetSeconds()-stats->
    second.timeFirstTxPacket.GetSeconds()<<std::endl;
    std::cout<<"Last Received Packet  : "<< stats->second.timeLastRxPacket.GetSeconds
    ()<<" Seconds"<<std::endl;
    std::cout<<"Lost Packets = "<< stats->second.lostPackets<<std::endl;
    std::cout<<"Throughput: " << stats->second.rxBytes * 8.0 / (stats->second.
    timeLastRxPacket.GetSeconds()-stats->second.timeFirstTxPacket.GetSeconds())
    /1024/1024  << " Mbps"<<std::endl;
    std::cout<<"
    ---------------------------------------------------------------------------"<<std
    ::endl;
  }

  Simulator::Schedule(Seconds(0.1),&ThroughputMonitor, fmhelper, flowMon);
}

// Funcion de lectura de archivo de posiciones.
vector<vector<double> > readCordinatesFile (std::string node_coordinates_file_name)
{
  ifstream node_coordinates_file;
  node_coordinates_file.open (node_coordinates_file_name.c_str (), ios::in);

  if (node_coordinates_file.fail ())
  {
        NS_FATAL_ERROR ("File " << node_coordinates_file_name.c_str () << " not found"
    );
  }
  vector<vector<double> > coord_array;
  int m = 0;

  while (!node_coordinates_file.eof ())
  {
    string line;
    getline (node_coordinates_file, line);

    if (line == "")
    {
      NS_LOG_WARN ("WARNING: Ignoring blank row: " << m);
      break;
```

```
    }

    istringstream iss (line);
    double coordinate;
    vector<double> row;
    int n = 0;

    while (iss >> coordinate)
    {
      row.push_back (coordinate);
      n++;
    }

    if (n != 7)
    {
      NS_LOG_ERROR ("ERROR: Number of elements at line#" << m << " is "  << 7 << "
    which is not equal to 3 for node coordinates file");
      exit (1);
    }

    else
    {
      coord_array.push_back (row);
    }
    m++;
  }
  node_coordinates_file.close ();
  return coord_array;
}

// Funcion de lectura de archivo de cantidad de usuarios por nodo.
vector<uint32_t> readNumUeFile (std::string node_num_ue_file_name)
{
  ifstream node_num_ue_file;
  node_num_ue_file.open (node_num_ue_file_name.c_str (), ios::in);

  if (node_num_ue_file.fail ())
    {
      NS_FATAL_ERROR ("File " << node_num_ue_file_name.c_str () << " not found");
    }

  vector<uint32_t> num_ue_array;
  int m = 0;

  while (!node_num_ue_file.eof ())
    {
      string line;
      getline (node_num_ue_file, line);

      if (line == "")
```

```
        {
          NS_LOG_WARN ("WARNING: Ignoring blank row: " << m);
          break;
        }

      istringstream iss (line);
      uint32_t numUe;

    if (iss >> numUe)
      {
        num_ue_array.push_back (numUe);
      }
      m++;
  }

  node_num_ue_file.close ();
  return num_ue_array;
}
```