UNIVERSIDAD DE VALLADOLID

E.T.S.I. TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

GRADO EN INGENIERÍA DE TECNOLOGÍAS DE TELECOMUNICACIÓN

Métodos de control del retardo medio en redes PON

Autor:
Dña Anaïs Andrés Fraile
Tutor:
Dña Noemí Merayo Álvarez

TÍTULO: **Métodos de control del retardo medio en redes PON**

AUTOR: **D. Anais Andrés Fraile**

TUTOR: **D. Noemí Merayo Álvarez**

DEPARTAMENTO: **Teoría de la Señal y Comunicaciones e Ingeniería Telemática**

TRIBUNAL

PRESIDENTE: **D. Ignacio de Miguel Jiménez**

VOCAL: **D. Ramón J. Durán Barroso**

SECRETARIO **Dña Noemí Merayo Álvarez**

SUPLENTE **D. Juan Carlos Aguado Manzano**

SUPLENTE **Dña Patricia Fernández Reguero**

FECHA: **12 de Septiembre de 2014**

CALIFICACIÓN:

Resumen de TFG

El estudio de investigación realizado y descrito en este Proyecto Fin de Carrera, se basa en el análisis de diversos aspectos relacionados con la gestión de recursos en las redes de acceso PON (*Passive Optical Network*), cuya arquitectura está implementada en un simulador bajo el entorno *OMNET++*.

Así pues, inicialmente, se realizó un análisis de las prestaciones de los algoritmos de asignación dinámica de recursos en redes PON, en concreto algoritmos de asignación dinámica de ancho de banda (DBA, *Dynamic Bandwidth Allocation)*, bajo patrones de tráfico realista, esto es, tráfico auto-semejante o *self-similar*. Para ello se utilizaron algunos algoritmos DBA considerando diferentes parámetros de ejecución asociados al diseño de dichos algoritmos en la red PON, para analizar el impacto de dicho patrón de

tráfico sobre diferentes parámetros de red, tales como, sobre el retardo medio. A continuación, se desarrolló un controlador de admisión de paquetes (CAD, *Control Admision Delay-aware*) en las colas de los usuarios asociados a una estación ONU (*Optical Network Unit*) con la finalidad de gestionar de forma eficiente el retardo máximo de las clases de servicio prioritarias.

Palabras clave

EPON (red óptica pasiva Ethernet), QoS (calidad de servicio), retardo medio, IPACT (*Interleaved Polling with Adaptive Cycle Time*), DaSPID (*Delay aware Service level agreement PID)*, controlador de admision de paquetes (CAD).

Abstract

The research developed and described in this project is based on the analysis of several aspects regarding resources management in Passive Optical Networks (PON), whose architecture is implemented in an optical network simulator in the OMNET++ environment.

Firstly, we carried out an analysis of the performance of dynamic resources allocation algorithms in PON networks, specifically dynamic bandwidth allocation algorithms (DBA) under realistic traffic partners, that is, self-similar traffic. In this way, some DBA algorithms were studied, considering different execution parameters associated with the design of these algorithms in PON networks, in order to analyze the impact of that kind of traffic in different network parameters such as, the mean packet delay. After that, an admission delay-aware controller (CAD) was developped inside the user's queues associated with each ONU (Optical Network Unit) in order to guarantee the estipulated delay requirements that satisfy the subscribers' needs.

Keywords

EPON (*Ethernet Passive Optical Networking*), QoS (*Quality of Service*), delay-aware, IPACT (*Interleaved Polling with Adaptive Cycle Time*), DaSPID (*Delay aware Service level agreement PID), admission delay-aware controller* (CAD).

Agradecimientos

Quiero aprovechar este espacio para agradecer a todas aquellas personas que me han ayudado y apoyado a lo largo de estos cuatro años de carrera.

Me gustaría empezar agradeciéndole a mis padres Alfredo y Toñy, quienes han sido mi pilar fundamental durante este tiempo, todo el esfuerzo que han hecho para poder estar hoy aquí, así como el ánimo que me han dado siempre. También quiero destacar al resto de familiares, en especial mi hermana Esther, abuela Sagrario, abuelos Jesús y Micaela, tíos Maria Jesús, Gonzalo, Paco, Rubén y Julia, así como mis primos Belén y Jona, quienes aportaron su granito de arena. He de agradecer también toda la ayuda y paciencia que ha tenido conmigo Roberto, mi gran compañero durante toda la carrera porque sin su ayuda habría sido mucho más difícil mi paso por la Universidad. Por otro lado, nombrar a todos los amigos y compañeros de clase tales como Raquel, Ana, y Javier, ya que siempre han estado ahí dispuestos a echarme una mano y sin los cuales todo este recorrido habría sido completamente distinto. Finalmente, me gustaría dar las gracias a las tutoras que he tenido durante este Proyecto Fin de Carrera, Noemí y Tamara, por haberme dado la oportunidad de llevar a cabo esta tarea y por todo el ánimo y ayuda que me han dado durante la realización de todo este trabajo.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE ECUACIONES

1

Introducción

1.1 Motivación

En este Proyecto Fin de Carrera se ha llevado a cabo una gestión de la calidad de servicio en redes PON. Para ello se ha partido de un simulador de Redes de Acceso Ópticas basado en el estándar Ethernet (EPON), ya implementado en el entorno de simulación OMNeT++.

La red de acceso, popularmente conocida como bucle de abonado, conecta a los clientes o abonados finales a la oficina central correspondiente. Hasta los últimos años las tecnologías más utilizadas en el despliegue de estas redes de acceso, han sido las basadas en líneas xDSL (*x Digital Subscriber Loop/Line*) o redes HFC (*Hybrid Fibre-Coaxial*), pero en la actualidad están en desuso. Esto es debido a que la acogida masiva de las aplicaciones ofrecidas por Internet así como la aparición de nuevos servicios, alguno de los cuales requiere respuesta en tiempo real, han disparado enormemente la demanda de ancho de banda de los clientes de modo que estas tecnologías no son capaces de hacer frente a las prestaciones demandadas por las redes troncales, produciendo un "cuello de botella" a la hora de enfrentarse a estos nuevos servicios emergentes.

La tecnología candidata que cuenta con un perfil adecuado a las necesidades demandadas se basa en la fibra óptica como medio de transmisión para llegar al abonado final. La incorporación de la fibra óptica hasta el usuario final es conocida como FTTH (*Fiber To The Home*, Fibra hasta el hogar), la cual supone una infraestructura de acceso de gran capacidad que ofrece a los abonados servicios de banda ancha a grandes tasas de velocidad. Hoy en día, FTTH está creando un gran impacto comercial a nivel mundial y ya son muchos los operadores que están compitiendo en el despliegue de fibra hasta el

hogar en diferentes países. Las arquitecturas más habituales para el despliegue de redes FTTH son las redes ópticas pasivas (PONs, *Passive Optical Networks*). Este tipo de red está basada en una arquitectura punto-multipunto, resulta ser muy económica puesto que utiliza una única fibra desde la oficina central hasta el punto de distribución. Sin embargo, esta tecnología punto-multipunto presupone que todos los abonados finales se vean obligados a compartir el canal de acceso ascendente, por lo que será necesario aplicar protocolos de contienda en dicho canal [1][2][3][4].

Gran parte de la investigación llevada a cabo actualmente en el campo de las redes PON [5] se centra en el desarrollo de esquemas que gestionen el acceso al medio compartido y que sean capaces de distribuir el ancho de banda disponible de un modo equitativo y eficiente. Al mismo tiempo, la calidad de servicio (QoS, *Quality of Service*) es otro de los puntos fuertes en los que se focaliza gran parte de la actual investigación. De forma adicional, también se espera que las redes de nueva generación proporcionen movilidad al usuario final. Se puede concluir por lo tanto, que este tipo de investigación presenta una gran relevancia en el marco actual del despliegue de fibra en el acceso.

Lo que se persigue en este Proyecto Fin de Carrera, es proponer nuevos métodos para optimizar la gestión de recursos y la calidad de servicio ofrecida en las actuales redes de acceso PON. Para ello, se parte de una arquitectura de red EPON desarrollada previamente por el Grupo de Comunicaciones Ópticas (GCO).

Así, el eje principal de este proyecto será el análisis de las prestaciones de algunos algoritmos de asignación dinámica de recursos en redes EPON bajo un patrón de tráfico realista, esto es, tráfico auto-semejante o *self-similar*. Una vez llevado a cabo este estudio, se pondrá en marcha el desarrollo e implementación de nuevas estrategias que mejoren el comportamiento de ciertos parámetros en una red EPON. En concreto, el proyecto se centrará fundamentalmente en el diseño y desarrollo de un sistema de control de admisión de paquetes cuyo objetivo es la gestión eficiente del retardo máximo experimentado en el tráfico prioritario.

1.2 Objetivos

1.2.1 Objetivo General

El objetivo principal de este proyecto es la gestión de la calidad de servicio en redes PON. De forma paralela, llevar a cabo medidas que permitan mejorar las situaciones críticas relacionadas con esta calidad de servicio, es una tarea importante que también es necesaria. Estos objetivos generales se pueden desglosar en otros más específicos que a continuación se listan.

1.2.2 Objetivos Específicos

Con la realización de este proyecto se han cubierto los siguientes objetivos específicos:

1. Análisis de prestaciones de algoritmos de asignación dinámica de recursos en redes PON, en concreto algoritmos de asignación dinámica de ancho de banda (DBA, *Dynamic Bandwidth Allocation)*, bajo patrones de tráfico realista, que como ya hemos dicho se denomina también tráfico auto-semejante o *self-similar*.

2. Análisis de prestaciones de algunos algoritmos DBA considerando diferentes parámetros de ejecución asociados al diseño de dichos algoritmos de gestión de recursos dentro de la PON y a diferentes características del tráfico rafagoso.

3. Desarrollo de un controlador de admisión de paquetes (CAD, *Control Admision Delay-aware*) en las colas de los usuarios asociados a una estación *ONU* (*Optical Network Unit*) con la finalidad de gestionar de forma eficiente el retardo máximo de las clases de servicio prioritarias.

1.3 Fases y Métodos

La metodología a seguir para el desarrollo de los objetivos del Proyecto Fin de Carrera ha constado fundamentalmente de las siguientes fases:

1.3.1 Fase de Investigación

Esta fase tiene la doble finalidad de adquirir familiaridad con el entorno de simulación OMNeT++ [6][7] y de estudiar el estado del arte de las redes PON.

De este modo, se pretende empezar con una iniciación al simulador que se va a utilizar, siendo éste OMNeT++, realizando para ello algunos ejercicios para adquirir soltura con el mismo [6][7]. Así pues, se llevaron a cabo las siguientes actividades:

- Análisis y comprensión del manual de usuario de la versión 4.1 de OMNeT++, con el cual se realizará la implementación del simulador.
- Análisis y comprensión de la guía de usuario de la versión 4.1 de OMNeT++ para manejar con soltura el simulador.
- Realización y estudio de dos ejemplos en el entorno de simulación de OMNeT++:
    - TicToc: Ejemplo de transmisión de mensajes entre varias estaciones que se mandan mensajes entre sí.
    - FIFO o Cola M/M/1: Ejemplo de inserción de paquetes en una cola con el principio de "primero en entrar, primero en salir" (FIFO, *First Come, First Served*).

Por otro lado, en esta fase de investigación también se realizó un análisis del estado del arte de las arquitecturas de redes de acceso ópticas PON.

1.3.2 Fase de Diseño

En esta fase del proyecto se partirá de la arquitectura ya implementada en el simulador, y tras el análisis de las prestaciones de algunos algoritmos ya implementados, en concreto IPACT (*Interleaved Polling with Adaptive Cycle Time*) [8][9], ante patrones de tráfico *self-similar*, se diseñarán y adoptarán nuevas estrategias para la mejora de los resultados de dichos algoritmos en la gestión de recursos de la red PON.

Así pues se definirán una serie de parámetros en el archivo de inicialización de la plataforma de OMNeT++ (*omnetpp.ini*) para poder analizar el funcionamiento de la red ante la implementación de los distintos algoritmos bajo patrones de tráfico con cierto nivel de rafagosidad.

Por otro lado, también se desarrollará un controlador de admisión de paquetes (CAD, *Control Admision*) en las colas de los usuarios asociados a una estación *ONU* con la finalidad de gestionar de forma eficiente el retardo máximo de las clases de servicio prioritarias. Para ello se implementarán dos tipos de CAD, uno estático y otro dinámico. Finalmente se procederá a la variación de algunos de los parámetros del CAD dinámico para una mejor adaptación a las necesidades y condiciones de la red.

1.3.3 Fase de Implementación

En esta fase se implementará los controladores de admisión de paquetes diseñados para las colas de los usuarios que se encuentran asociados a una estación *ONU*, con la finalidad de reducir el retardo máximo de las clases de servicio prioritarias. En concreto, se llevarán a cabo las siguientes tareas:

- Análisis de prestaciones de algunos algoritmos de asignación dinámica de ancho de banda en redes PON ya implementados, en concreto IPACT, bajo patrones de tráfico auto-semejante.

- Diseño e implementación de un CAD de estático dentro de un algorimo que controla el retardo del tráfico prioritario. En concreto, el objetivo será optimizar el comportamiento y prestaciones del algoritmo DaSPID (*Delay aware Service level agreement PID*), con su correspondiente estudio bajo patrones de tráfico auto-semejante.

- Diseño, implementación y verificación de resultados de un CAD dinámico en el algoritmo DaSPID bajo un patrón de tráfico *self-similar*. También se modificarán y se añadirán una serie de parámetros relacionados con el mismo para optimizar los resultados obtenidos de nuestra red en factores tales como el retardo medio, la probabilidad de bloqueo, etc.

- Recogida de estadísticas de la red sobre retardo medio, probabilidad de pérdida de paquetes, evolución del CAD, etc.

1.3.4 Fase de Simulación

En esta fase del proyecto se procederá a la realización de las simulaciones de todos los algoritmos implementados con los parámetros definidos teóricamente, todos ellos bajo un patrón de tráfico *self-similar*, para el análisis de los resultados obtenidos. En concreto, se realizarán dicho proceso de la siguiente manera:

- Simulación del algoritmo IPACT con variaciones en algunos parámetros

definidos, tales como el número de *streams* o el tamaño de la ventana:

  - Recogida de los resultados de simulación del retardo extremo a extremo en función de la carga de red.
  - Realización de gráficas y análisis de los resultados obtenidos.

- Simulación del algoritmo DaSPID sin CAD ante distintos valores de algunos parámetros definidos para este algoritmo como por ejemplo el tamaño de la ventana:
  - Recogida de los resultados de simulación del retardo extremo a extremo en función de la carga de red.
  - Realización de las gráficas y estudio de los resultados obtenidos.
- Simulación del algoritmo DaSPID con CAD estático en función de una serie de parámetros del algoritmo:
  - Recogida de los resultados de simulación del retardo extremo a extremo, probabilidad de pérdida de paquetes y evolución del CAD.
  - Realización de las gráficas y estudio de los resultados obtenidos.
- Simulación del algoritmo DaSPID con CAD ante diversos valores de los parámetros del algoritmo:
  - Recogida de los resultados de simulación del retardo extremo a extremo, probabilidad de pérdida de paquetes y evolución del CAD.
  - Realización de las gráficas y estudio de los resultados obtenidos.

1.3.5 Fase de Realización de los Informes

En esta fase se procedió a realizar los informes del Proyecto Fin de Carrera:

- Documentación de todos los archivos que forman la red.
- Realización de la memoria del Proyecto Fin de Carrera.

1.4 Estructura de la Memoria del PFC

En el Capítulo 2 se realiza un estudio del estado del arte en el ámbito de las redes de acceso ópticas PON. Como este proyecto parte de la red PON desarrollada en años anteriores, este capítulo expone una breve descripción de estas redes.

El Capítulo 3 está enfocado en el entorno de trabajo empleado para la realización e implementación de este proyecto. De este modo, se presenta una breve introducción al entorno de trabajo OMNet++, para que así el lector pueda adquirir nociones básicas sobre

el funcionamiento de este simulador para entender los posteriores capítulos de este PFC, en los que se describirá con más detalle la red implementada en OMNeT++.

En el Capítulo 4 se presenta la arquitectura de la red de acceso PON implementada en el simulador OMNeT++, pero no se entrará en profundidad en la misma puesto que ha sido analizada ampliamente en proyectos anteriores.

En el Capítulo 5 se lleva a cabo una breve introducción del tráfico *self-similar* que se va a implementar para analizar el comportamiento de algunos algoritmos DBA. Para ello en primer lugar, se describen brevemente las técnicas para la generación de tráfico auto-semejante en la parte óptica de la red. Así pues, se realiza un análisis de prestaciones del algoritmo IPACT, ante dicho patrón de tráfico descrito.

El Capítulo 6 presenta un algoritmo de asignación dinámica de recursos de *polling* basado en un sistema de PID (*Proportional-Integral-Derivative*, Proporcional-Integral-Derivativo) para controlar el retardo en la red implementada, previamente desarrollado por el GCO en el simulador, analizando el comportamiento del retardo extremo a extremo ante un patrón de tráfico auto-semejante. Además se realiza una breve introducción a los métodos de inserción y extracción de paquetes utilizados dentro del simulador.

El Capítulo 7 expone el diseño e implementación de un control de admisión de paquetes, CAD, en el algoritmo DaSPID, de modo que se desarrollan dos tipos de CAD, uno estático y otro dinámico. De esta forma, se analizan las prestaciones del algoritmo mediante ambos sistemas de control ante un patrón de tráfico auto-semejante.

En el Capítulo 8 se recogen las conclusiones derivadas de todo el trabajo realizado en este Proyecto Fin de Carrera, así como las líneas futuras que se abren a partir de él para próximas investigaciones.

Finalmente, en el Capítulo 9 se incluyen todas las referencias bibliográficas citadas a lo largo de la presente memoria.

2

Redes de Acceso Ópticas Pasivas (PON)

2.1 Introducción

En este capítulo se realiza una exposición del estado del arte en el área de las redes de acceso ópticas. El tráfico en la red ha experimentado un notable incremento en los últimos años, debido en buena medida a la proliferación de dispositivos móviles con acceso a Internet. De entre las diferentes tecnologías de acceso que pueden encontrarse desplegadas en la actualidad, tales como línea digital de abonado (DSL, *Digital Subscriber Line*), cable coaxial o fibra óptica, la que se postula como mejor alternativa para dar soporte a la alta demanda de ancho de banda por parte de los usuarios finales de la red es la fibra óptica. Esto es así porque la tecnología óptica permite altas tasas de transmisión (de 50 Mbps a 100 Gbps) a mayores distancias que las otras dos opciones.

Así pues, en este capítulo se explicarán las características generales de las tecnologías de redes de acceso, centrándonos con posterioridad en la red con la que se trabajará para la realización de la presente memoria, esto es, la red de acceso óptico pasiva. Finalmente, se llevará a cabo una breve reseña del estándar adoptado, así como de los mecanismos de control de acceso al medio implementados. No se entrará en profundidad en ambos temas, pues éstos han sido ampliamente discutidos en Proyectos Fin de Carrera de años anteriores.

2.2 Definición de la Red de Acceso

Una red de acceso es el conjunto de elementos que permiten que cada abonado pueda conectarse con la central local de la que es dependiente. De este modo, tal y como ilustra la Figura 1, esta red se compone de una oficina central (*CO, Central Office*) en la

que se situará el módulo *OLT* (*Optical Line Terminal*), y en el otro extremo de la red se encuentra el usuario final conectado, emplazamiento donde se encuentra el dispositivo *ONU*, cuya descripción será abordada en más detalle en capítulos posteriores. Debido al gran aumento experimentado en el tráfico que circula en las redes existentes, el tipo de enlace empleado para unir ambos extremos de la red es la fibra óptica, ya que ésta se ha impuesto como la mejor solución para hacer frente a los retos planteados en la red de acceso debido a características tales como:

- Mayor velocidad de propagación de una señal.
- Mayor capacidad de transmisión.
- Inmunidad ante interferencias electromagnéticas.
- Menor atenuación y mayor ancho de banda.
- Menores tasas de error.

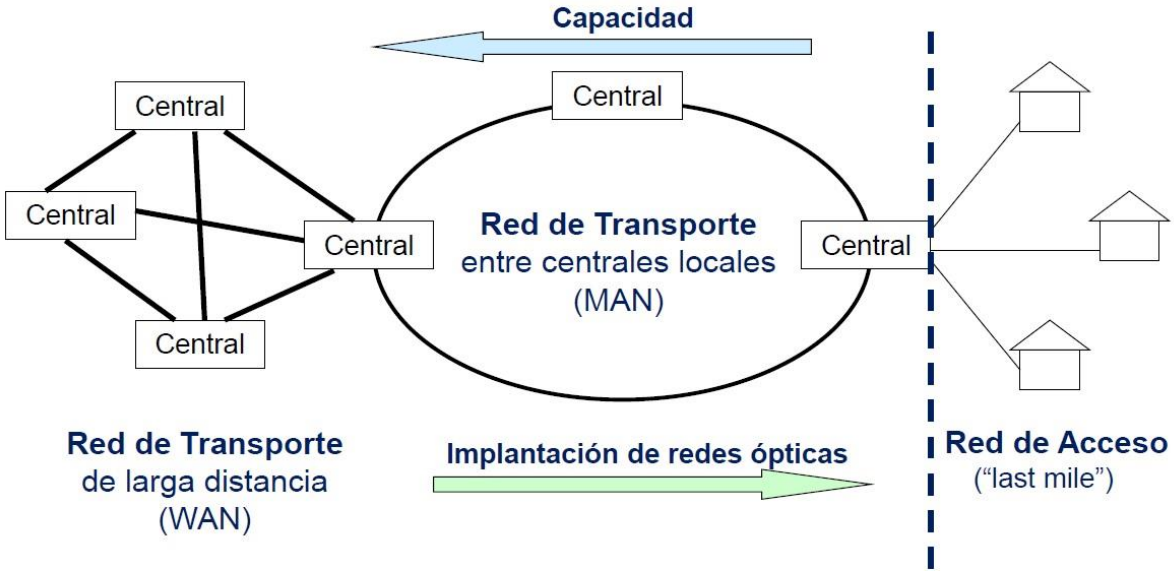


Figura 1. Evolución de la Red de Acceso y de la Red de Transporte.

Por lo tanto, la red de acceso está basada en la tecnología *FTTx (Fiber to the x)*, siendo éste un término con el que designamos cualquier acceso de banda ancha sobre fibra óptica que sustituye parcial o totalmente el cobre del cable de acceso. De todas las infraestructuras *FTTx* existentes, nuestro estudio se centra *FTTH* o Fibra hasta el Hogar. Esta infraestructura de red propone la utilización de fibra óptica hasta el domicilio del usuario y se basa en la utilización de cables de fibra óptica y sistemas de distribución ópticos adaptados a esta tecnología para la distribución de servicios avanzados tales

como telefonía, internet de banda ancha y televisión a los hogares y negocios de los abonados. Un resumen de todas las infraestructuras *FTTx se* observa en la Figura 2.

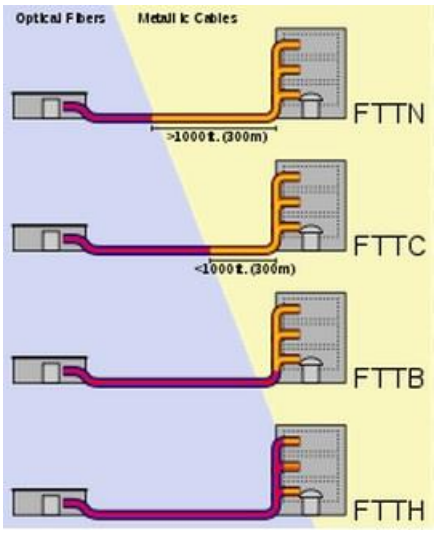


Figura 2. Distintas tecnologías FTTx

2.3 Redes de Acceso Ópticas Pasivas (PON, *Passive Optical Network)*

Una red de acceso PON se caracteriza por constar únicamente de elementos pasivos en el camino entre la fuente y el destino. El funcionamiento básico de una red PON consiste en una comunicación bidireccional entre una unidad de línea óptica u *OLT*, localizada dentro de la oficina central, un divisor óptico pasivo (*Splitter*), y varias unidades de red óptica u *ONUs* localizadas dentro o cerca de las dependencias del abonado final [10][11]. El *OLT* se conecta punto-a-punto con el *Splitter*, y éste último se conecta punto-a-multipunto con las *ONUs*, de manera que divide la fibra troncal en varias ramas de distribución que conectan a los usuarios finales con la oficina central. Un esquema simplificado de la arquitectura de una red PON es el que muestra la Figura 3.

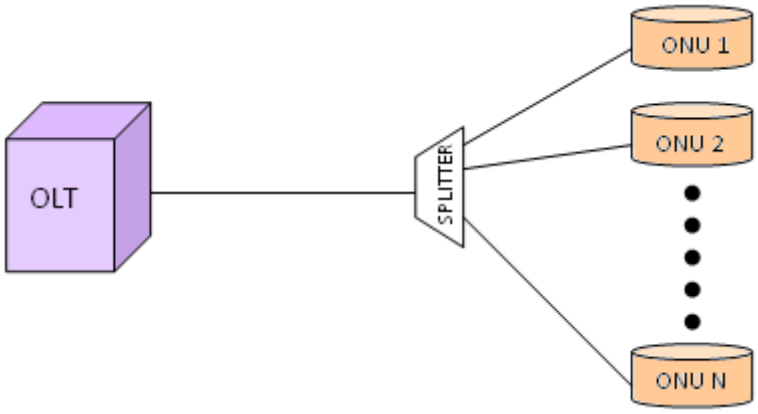


Figura 3. Topología típica de una red de acceso PON.

Las principales razones que motivan la implantación de este tipo de redes de acceso son las siguientes:

- Permiten alcanzar a usuarios localizados a distancias de hasta 20 kilómetros desde la oficina central, superando así la máxima cobertura de las tecnologías DSL utilizadas en los últimos tiempos (máximo 5 kilómetros desde la central).

- La fibra óptica dispone de mayor capacidad para transportar información, proporcionando así un mayor ancho de banda por usuario que alternativas más antiguas como xDSL (*xDigital Subscriber Line*) y CATV (*Community Antenna Television*).

- Permite disponer de una mayor calidad de servicio a la vez que reduce el mantenimiento de la red. Esto es debido a características tales como la inmunidad a ruidos electromagnéticos, la no propagación de descargas eléctricas procedentes de rayos, etc.

- Las tasas de transmisión son más elevadas ya que puede superponer longitudes de onda adicionales, utilizando para ello la tecnología WDM (*Wavelength Division Multiplexing*).

De entre las diferentes topologías en que puede implementarse una red de acceso PON, la que predomina es la topología en árbol. Esta topología es tomada como referencia por todos los estándares PON en la definición de su arquitectura, a la vez que multitud de vendedores de *software* y de *hardware* la soportan para su implementación. En una red en árbol, todos los usuarios comparten el ancho de banda en el canal

ascendente, mientras que en el sentido descendente cada usuario tiene el total de la capacidad de su canal individual.

Aunque la configuración de una topología en árbol puede ser más difícil en comparación con otras topologías, lo cierto es que es muy popular en el despliegue de redes PON, por lo que la parte óptica de la red simulada en OMNeT++ para este PFC también sigue una topología en árbol.

2.4 Estándares en Redes de Acceso PON

Respecto a los estándares disponibles para las redes de acceso PON, se ha elegido trabajar en este PFC con el estándar EPON, ya que es una de las alternativas más prometedoras a nivel de despliegue y en la que se está investigando en la actualidad. Puesto que en PFCs anteriores este tema ya ha sido tratado en profundidad, únicamente trataremos superficialmente algunos de sus aspectos más destacados.

Así pues, el estándar EPON está basado en el protocolo Ethernet, y se encuentra definido dentro de la especificación IEEE.802.3ah, aprobada por la *IEEE Standard Association* en junio de 2004. Está definido para soportar tasas de transmisión por encima de 1Gbit/s, siendo ésta la empleada en nuestra red. Además, este estándar requiere el uso de algún protocolo adicional con el objetivo de alcanzar una compatibilidad total con el estándar IEEE 802.1D [12], de modo que por ejemplo, emplea el protocolo de comunicación MPCP (*Multi-Point Control Protocol*) para transmitir los datos entre el *OLT* y las *ONUs*.

2.5 Control de Acceso al Medio en Redes PON

Respecto al protocolo de control de acceso al medio empleado en nuestra red EPON, decir que se utiliza el llamado TDMA (*Time Division Multiplexing Access*, Acceso Múltiple por División en el Tiempo), el cual también ha sido ampliamente descrito en anteriores PFC. Este protocolo será necesario en el canal ascendente para que diversos usuarios no colisionen al acceder al tramo de red común.

Este protocolo se basa en la utilización de una única longitud de onda en el canal de acceso compartido, asignando un cierto intervalo de tiempo a cada usuario a lo largo de un periodo. Trabajando sobre una sola fibra, se utilizan longitudes de onda diferentes mediante técnicas WDM. De esta manera, dada nuestra topología con dos canales

(ascendente y descendente), emplearemos una única longitud de onda diferente para cada uno de ellos.

Finalmente, se utilizan algoritmos de asignación dinámica de ancho de banda (DBA) para adaptar la capacidad de la red a las condiciones de tráfico existentes en todo momento, modificando la distribución el ancho de banda asignado a cada *ONU* en función de la demanda actual. Con este mecanismo se consigue no desaprovechar la capacidad del canal y será el *OLT* el encargado de controlar esta asignación del ancho de banda ciclo tras ciclo en función de los requisitos de demanda del ancho de banda actual de las *ONUs*. En el Capítulo 5, se describirá con más detenimineto en qué consiste el principio de funcionamiento de los algoritmos de asignación dinámica de ancho de banda de *polling*, los cuales se emplearán en este proyecto.

2.6 Conclusiones

Las tecnologías implementadas en los últimos años no cubrían toda la demanda de ancho de banda que los abonados exigían para la transmisión de información, por lo que se han propuesto las tecnologías *FTTxk*, basadas en fibra óptica, para cubrir la demanda actual. En la actualidad, la opción por excelencia que está empezando a implementarse es la fibra hasta el hogar (*FFTH*).

Mediante esta tecnología, la fibra óptica llega hasta el hogar del usuario, permitiendo así disponer de un gran ancho de banda y calidad de servicio, y es por ello por lo que ésta es elegida para el estudio en el que se centra este proyecto. Para su implementación se han elegido las arquitecturas de Redes de Acceso Ópticas Pasivas (PON) con una topología en árbol cuya configuración es punto-multipunto en el sentido descendente y punto-a-punto en el canal ascendente. Esta infraestructura de red requiere un control de acceso al medio que gestione el canal compartido, por lo que se implementará TDMA siguiendo el estándar de paquetes Ethernet (EPON).

3

Entorno y Herramientas de Trabajo

3.1 Introducción

En este capítulo se describe el entorno de trabajo utilizado para desarrollar el simulador de redes de acceso EPON. Puesto que en anteriores Proyectos Fin de Carrera se ha tratado este tema en profundidad, se expondrán unas nociones básicas que permitirán una aproximación a los distintos elementos que conforman el entorno de trabajo del simulador de redes OMNeT++.

3.2 Simulador de redes OMNeT++

El entorno de trabajo OMNeT++ es un simulador de redes de eventos discretos, que está basado en el lenguaje de programación *C++*. Se basa en módulos orientados a objetos, y es de código abierto [13]. Conviene destacar que puesto que es un simulador de eventos discretos, en OMNeT++ los objetos cambian de estado únicamente en instantes discretos de tiempo, es decir, nada sucede entre dos eventos consecutivos, los cuales tardan un tiempo nulo en ser realizados.

3.2.1 Creación de un proyecto en OMNeT++

Con el objetivo de empezar a trabajar con el entorno de trabajo OMNeT++ y crear así un proyecto en el mismo, se necesitan al menos un fichero **.ned* y uno o más ficheros **.cc*/**.h* que modelen el funcionamiento de los módulos simples y las conexiones básicas de la red.

Así pues, Los ficheros **.ned* están escritos en el lenguaje de alto nivel NED (*NEtwork Description*), el cual se emplea para la creación de la arquitectura de las redes, y puede definirse de dos formas, o bien con cualquier editor de texto (código fuente), o con el editor visual GNED (bloques de módulos). Para poder ser compilado, OMNeT++

lo pasa a código *C++* mediante el compilador llamado *nedtool*. De esta manera se define la topología de la red, los módulos, puertas, conexiones, y posición en la visualización del entorno a partir de estos ficheros **.ned*.

Por otro lado, los ficheros ficheros **.cc/*.h* se encargan de modelar el comportamiento de un módulo y para ello lo primero que se hace es crear una clase derivada de la clase base *cSimpleModule*. Se podrán introducir nuevas funciones o modificar las funcionalidades incorporadas en este módulo para modelar el comportamiento deseado del sistema. Si además se quieren crear mensajes propios para un sistema concreto, pueden incluirse nuevos ficheros **.msg*, que serán luego pasados a *C++* por el compilador de mensajes.

Un aspecto clave a tener en cuenta cuando para poder llevar a cabo un proyecto en OMNeT++, es el fichero de configuración *omnetpp.ini*. En él se define cómo va a ser ejecutada la simulación y en él aparecen los valores de los parámetros declarados en el modelo de la red. Finalmente, OMNeT++ permite definir diversas configuraciones para la ejecución de la simulación (*[Run 1]*, *[Run 2]*,..., *[Run N]*), de entre las cuales habrá de elegirse una.

3.2.2 Concepto de modelado en OMNeT++

Un módulo OMNeT++ consiste en una jerarquía de modelos que permite reflejar la estructura de una red donde la comunicación entre módulos se basa en el intercambio de mensajes. El diseño de estos modelos se realiza de forma jerárquica, anidando los módulos entre sí, de forma que el nivel más alto es el denominado módulo de sistema y a partir de éste se van encontrando módulos compuestos de otros módulos, y así hasta el nivel más bajo, formado por los módulos simples. Los módulos simples son el nivel más bajo de la jerarquía y la unidad principal de simulación. En ellos reside todo el peso del modelado y funcionalidad del sistema, por lo que son la parte más importante del mismo. Las funcionalidades de los módulos compuestos se añaden en los archivos desarrollados en *C++* para los módulos simples, derivados de la clase *cSimpleModule*. Esta clase contiene algunas funciones mínimas que deben implementarse para que la simulación pueda funcionar (*initialize()*, *handleMessage()* y *finish()*), al mismo tiempo que permite la inclusión de nuevas funciones.

3.2.3 Conexiones entre los módulos

Para poder enlazar módulos entre sí se requiere de una serie de enlaces o canales. De esta manera, cada uno de los enlaces o canales que conectan módulos son instancias de las clases *cIdealChannel*, *cDelayChannel* o *cDatarateChannel*, que a su vez se extienden de la clase *cChannel*. Si bien *cIdealChannel* modela un canal ideal (sin retardo ni pérdida de paquetes), *cDelayChannel* y *cDatarateChannel* por ejemplo, permiten añadir al canal parámetros propios de conexiones reales, tales como:

- *Propagation delay:* Tiempo que tarda en llegar el mensaje al destino en relación a lo que debería tardar, es decir, en función de la longitud del canal atravesado y del medio en que se transmite.
- *BER o bit error rate:* Probabilidad de que un *bit* transmitido sea erróneamente identificado en recepción.
- *Data rate:* Tasa de transmisión de la red.

Los extremos de todo canal en OMNeT++ terminan en objetos de la clase *cGate* denominados "puertas". Estas puertas se definen en el fichero *.ned* de cada módulo y forman parte de la interfaz de éste con el resto de módulos. Según el tipo de conexión, se distinguen tres tipos de puertas: de entrada (*input*), de salida (*output*) o de entrada y salida (*inout*).

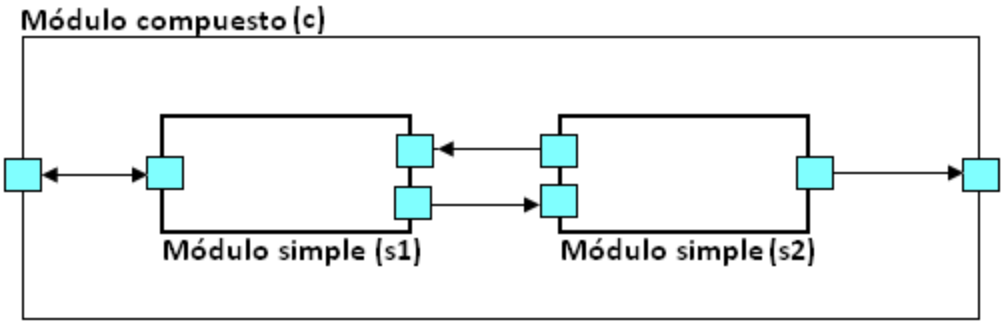


Figura 4. Posibles conexiones entre módulos simples y módulos compuestos.

A modo de ejemplo, en la Figura 4 el módulo compuesto *c* tiene dos puertas para conectarse con el exterior con otros módulos de la red, de tipo entrada/salida (*inout*) y de tipo salida (*out*). Al mismo tiempo, estas puertas están conectadas internamente a los módulos simples *s1* y *s2*, que tienen sendas puertas de entrada/salida y de salida para

conectar con el módulo padre, *c*. Por otra parte, los módulos simples también pueden conectarse entre sí mediante enlaces bidireccionales o unidireccionales. Los módulos simples podrán intercambiarse mensajes entre ellos, al mismo tiempo que podrán enviar/recibir mensajes hacia/del módulo compuesto con destino/origen algún módulo externo perteneciente a la misma red que se simula.

3.2.4 Intercambio de mensajes

La comunicación entre módulos se consigue mediante el intercambio de mensajes o paquetes. Los mensajes pueden representar tramas o paquetes de una red real, pudiendo incluir diferentes estructuras de datos. Asimismo, el intercambio de mensajes puede realizarse de dos formas, bien mediante un camino predefinido que seguirá el mensaje desde su creación en el módulo origen hasta su llegada al módulo destino, bien a través de puertas y conexiones directas entre dos módulos.

Para enviar un mensaje hacia otro módulo, es necesario emplear la función *send()* o *sendDelayed()*. La primera envía el mensaje al módulo destino en el mismo instante en que se invoca este método; mientras que la segunda lo envía cierto tiempo después, como si hubiera experimentado un retardo pasado como parámetro. En el caso de la función *send(),* existen tres posibles definiciones, con diferentes parámetros según la forma de referenciar la puerta por la que se envía el mensaje, que se definen a continuación:

*send (cMessage * msg, const char * gateName, int index = 0)*

*send(cMessage * msg, cGate * gate)*

*send (cMessage * msg, int gateId)*

Las posibles definiciones de la función *sendDelayed()* difieren con respecto a las tres anteriores en la inclusión de un parámetro de tipo *simtime_t* que especifica el retardo a añadir en la transmisión del mensaje. Existe también una tercera función para el envío de mensajes, *sendDirect()*. Esta función se emplea para enviar un mensaje a un módulo sin que se tenga en cuenta la puerta ni la conexión.

Cabe destacar que OMNeT++ no permite a un mismo módulo enviar más de una vez el mismo objeto mensaje, por lo que para enviar a varios destinos un mismo mensaje es necesario realizar copias del original mediante el método *dup()* de la clase *cMessage*

antes de realizar el primer envío. Esto se puede aplicar para retransmisiones, o para el caso de envío de mensajes a varios módulos por difusión o *broadcast*.

Otra posibilidad que ofrece el entorno de simulación es la creación de mensajes que sean enviados y recibidos por el mismo módulo, es decir, auto-mensajes o *self-messages*. La creación y asignación de parámetros son iguales que para el resto de mensajes, pero la forma de enviarlos es diferente, pues se realiza mediante la función *scheduleAt(cMessage *msg, simtime_t time)*. Esta función envía un auto-mensaje para el mismo módulo donde se encuentra implementado en el instante de tiempo que se pase como parámetro en la variable "*time*".

En capítulos posteriores se abordará el tema de generación de mensajes para la simulación de tráfico real desde los abonados conectados a las unidades ópticas (*ONUs*) hasta la estación principal (*OLT*), explicando también el generador de tráfico empleado.

3.3 Conclusiones

El entorno de trabajo OMNeT++ es un simulador de redes de eventos discretos, basado en módulos orientados a objetos, siendo programado en *C++*. La elección de esta plataforma de trabajo para la realización del proyecto se sustenta en el hecho de que es un entorno multiplataforma de libre distribución para propósitos académicos de investigación, además de que su programación por módulos favorece la simulación de procesos en paralelo y distribuidos.

Dentro de la simulación de redes de acceso que se ha desarrollado en OMNeT++ para este proyecto, un aspecto fundamental es el comportamiento de los diversos algoritmos implementados en función de diversos parámetros de configuración, ante un patrón de tráfico realista, denominado auto-semejante o *self-similar*, que será explicado en capítulos posteriores.

4

Arquitectura de la Red de Acceso Óptica Simulada en OMNeT++

4.1 Introducción

En el presente capítulo de esta memoria del Proyecto Fin de carrera, se presenta y detalla la arquitectura completa del simulador de redes de acceso desarrollado en el entorno de trabajo OMNeT++.

El simulador se ha implementado siguiendo el estándar EPON diseñado para trabajar a una tasa de dados de 1 Gbit/s. El protocolo de control de acceso al medio elegido es TDMA, el cual considera una longitud de onda de bajada o *downstream* y otra de subida o *upstream*. Además, se desarrollará una topología en árbol con una tecnología FTTH o fibra hasta el hogar para la red diseñada. De este modo, nuestra red constará de múltiples *ONUs* conectadas a un *OLT* central según la topología descrita.

En las próximas secciones de este capítulo, se explicará en primer lugar la arquitectura de la red implementada. Seguidamente se describirán los módulos y submódulos que la componen, así como sus funcionalidades, aunque sin entrar en gran profundidad ya que éstos fueron objeto de estudio en Proyectos de Fin de Carrera de años anteriores. Finalmente se definirán una serie de parámetros que aparecen especificados en el fichero de configuración *omnetpp.ini*.

4.2 Arquitectura General de la Red de Acceso EPON implementada

4.2.1 Esquema de la subred EPON

La red implementada en nuestro sistema es una red de acceso PON (*Passive Optical Network*) y cómo tal, no existen componentes activos entre el *OLT* y las *ONUs*. De este modo, solamente existen en toda la planta externa (20 km) componentes ópticos pasivos para guiar el tráfico por la red, tales como divisores o combinadores ópticos (*splitters*).

El estándar utilizado para desarrollar la subred óptica será EPON, diseñado para trabajar a una tasa de datos máxima de 1 Gbit/s, con la posibilidad de implementar dos mecanismos distintos de control de acceso al medio, TDMA y WDMA. Para nuestras simulaciones se elegirá TDMA, el cual considera una única longitud de onda para el sentido descendente y otra para el canal ascendente. Así pues el acceso de cada *ONU* al canal ascendente se rige por la asignación de un cierto intervalo de tiempo a lo largo de un período, como ilustra la Figura 5.

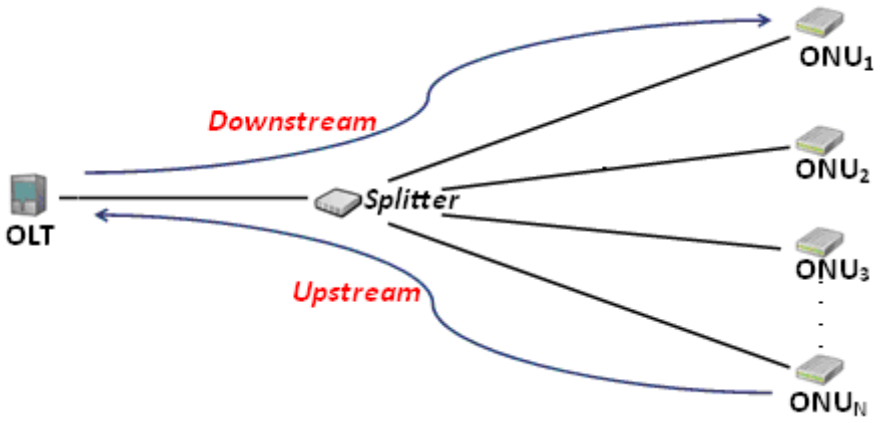


Figura 5. Acceso al Medio por División en Tiempo en la subred óptica simulada.

Junto con TDMA, la subred óptica emplea el protocolo de control multipunto MPCP (*Multi-Point Control Protocol*) para la comunicación entre el *OLT* y las *ONUs*, relacionada con la demanda y asignación de ancho de banda. Esta asignación se realiza a partir de algoritmos dinámicos (DBA), que pueden ser tanto centralizados como distribuidos. Se han implementado diversos algoritmos para la asignación del ancho de banda óptico en el simulador, de los cuales se eligirá el que se va a usar en cada

simulación mediante el valor asignado al parámetro *oltmethod_centralized0_Polling1* en el fichero de configuración *omnetpp.ini*. En nuestro caso en concreto, se trabajará con el algoritmo IPACT y DaSPID, tal y como se verá en los siguientes capítulos.

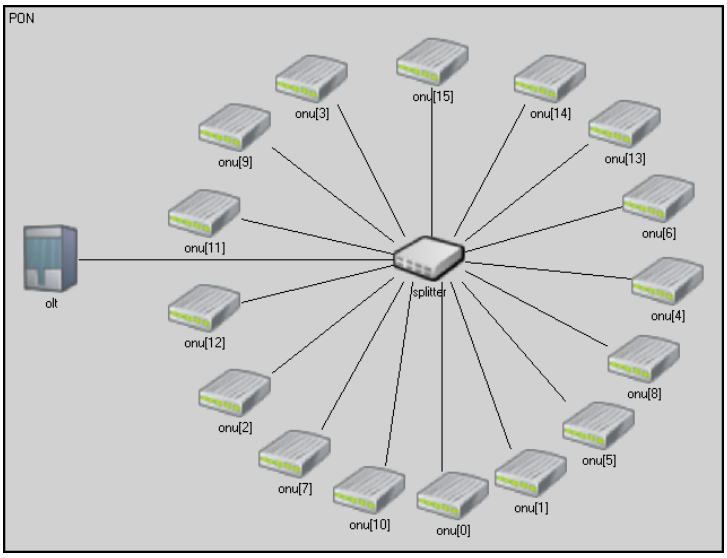


Figura 6. Arquitectura de la parte óptica de la red simulada en OMNeT++.

La Figura 6 ilustra la arquitectura de la subred óptica implementada en OMNeT++, donde se aprecia la topología en árbol entre el *OLT* y las 16 *ONUs*, que se logra con la presencia del *splitter* óptico. Todas las *ONUs* están configuradas para recibir tráfico de usuarios conectados directamente mediante fibra óptica.

4.3 Diseño de los módulos de la Red EPON

4.3.1 Nodo Terminal de Línea Óptico (OLT, Optical Line Terminal)

El *OLT* es un módulo compuesto que constituye el elemento central de la red EPON. Este terminal de línea óptico tiene como tareas principales el envío y recepción de mensajes de control con los que obtener información del estado de las colas de las *ONUs*, y así, a partir de alguno de los algoritmos DBA implementados en su capa MAC, asignar el ancho de banda que utilizará cada *ONU* para transmitir. Por este motivo se le puede denominar como el "cerebro" de la red de acceso.

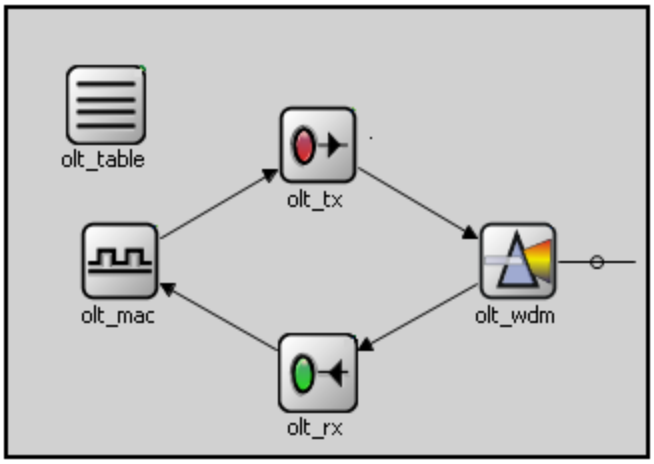


Figura 7. Módulo *OLT*.

El módulo *OLT* está formado por los siguientes módulos simples, como se observa en la Figura 7:

- *olt_mac*: Realiza todas las operaciones lógicas de asignación dinámica de ancho de banda y controla los problemas de contienda del canal ascendente en la parte óptica de la red, siendo así el módulo simple más importante del *OLT*. Además, se encarga de inicializar la subred EPON mediante el envío de un mensaje *Gate* a cada una de las *ONUs*. Estos mensajes *Gate* contienen información acerca del ancho de banda inicial asignado a las *ONUs* y el instante de tiempo en que se tienen que iniciar la transmisión de sus paquetes *Ethernet*. Este módulo está conectado con los módulos *olt_rx* y *olt_tx*, para gestionar la entrada y salida respectivamente de mensajes en el *OLT*, que pueden ser *Gate, Ethernet* o *Report*.

- *olt_rx*: Módulo simple conectado a la capa MAC (*olt_mac*) y al *splitter* interno del *OLT* (*olt_wdm*). Se encarga de recibir los paquetes que las *ONUs* envían al *OLT*, que pueden ser de tipo *Ethernet* o *Report*. Si el *olt_rx* recibe mensajes *Ethernet*, con datos de los usuarios finales, los borra para no sobrecargar la red (en una red real, estos mensajes se enviarían hacia la red troncal que conecta con el resto de la red). Si, en cambio, los mensajes que le llegan son de tipo *Report*, con la información necesaria para la gestión del ancho de banda, los envía hacia la capa MAC (módulo *olt_mac*) para su procesamiento.

- *olt_table*: Este módulo contiene una tabla en la que se almacena el valor del estado de las colas, del ancho de banda demandado y del ancho de banda

asignado para cada *ONU* en el tiempo de ciclo actual. Los parámetros guardados en esta tabla son actualizados por el módulo *olt_mac* con la llegada de cada mensaje *Report*, para lo cual accede de forma remota al módulo *olt_table*.

- *olt_tx*: Módulo simple que tiene como única función enviar los mensajes *Gate* que le llegan de la capa MAC (*olt_mac*) hacia el s*plitter* interno del *OLT* (*olt_wdm*), para que éstos sean enviados a las *ONUs* destino a través del *Splitter* óptico.

- *olt_wdm*: Este módulo simple actúa de divisor óptico pasivo o s*plitter* dentro de la estructura interna del *OLT*. El funcionamiento principal de este módulo se basa en separar los canales *downstream* y *upstream* dentro de la arquitectura del *OLT*. En el sentido ascendente, envía lo que le llega desde el *Splitter* de la red EPON hacia el módulo *olt_rx*. Por el contrario, en el sentido descendente, envía los mensajes que le llegan desde el módulo *olt_tx* hacia de fuera de la red para dirigirlos hacia las *ONUs*.

4.3.2 Nodo Splitter

El módulo simple *Splitter* es el divisor óptico pasivo encargado de reenviar los paquetes que le llegan en un sentido por todas las salidas del otro, sin realizar encaminamiento uniendo así *OLT* y las *ONUs*. Para el tráfico que va desde el *OLT* hacia las *ONUs* (recordemos que esta es una configuración punto-multipunto), actúa como duplicador de los paquetes, enviando cada uno de ellos a todas las *ONUs* de la red EPON. En el sentido *upstream* (cuya configuración es punto-a-punto), combina y reenvía el tráfico que recibe de las *ONUs* hacia el *OLT*, manteniendo el orden de llegada de los paquetes.

Este módulo está conectado directamente con el módulo *OLT* y el array de módulos *ONU*, mediante canales bidireccionales (puertas *inout*). Dado que, como se ha indicado, el simulador soporta WDM, la conexión entre el *Splitter* y cada nodo de la parte óptica consta de tantos canales como longitudes de onda se empleen en la red.

4.3.3 Nodo Unidad de Red Óptica (ONU, Optical Network Unit)

El módulo compuesto *ONU* modela la estación óptica situada en el extremo final de cada rama de la topología en árbol de la subred EPON, correspondiéndose con los usuarios finales de la red. Las unidades ópticas están situadas en las dependencias de los

clientes de la red de acceso que acceden a Internet a través de enlaces de fibra óptica (FTTH).

El número de *ONUs* presentes en la simulación se introduce como parámetro en el fichero de configuración *omnetpp.ini*, de modo que al inicializarse la ejecución se crean de forma dinámica tantas *ONUs* como se haya indicado. En todas las simulaciones que se han realizado en este Proyecto Fin de Carrera, este parámetro vale 16.

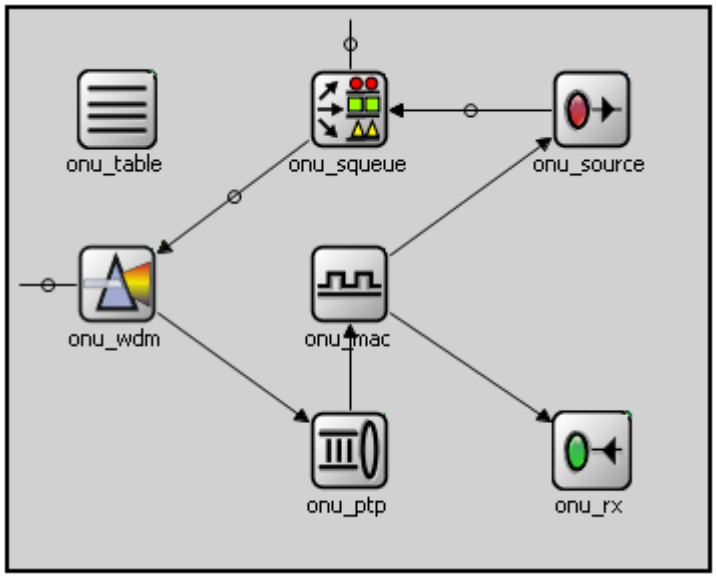


Figura 8. Módulo *ONU*.

La Figura 8 muestra el esquema del módulo compuesto *ONU*, formado por los siguientes módulos:

- *onu_mac*: Este módulo simple es el más importante de la *ONU* y representa la capa de control de acceso al medio de la *ONU*, encargándose de gestionar el tráfico que entra y sale de la misma. Controla la mecánica de transmisión de paquetes *Ethernet* de la *ONU* hacia el *OLT*. Este módulo posee una entrada, por la que se une al módulo *onu_ptp*; y dos salidas, por las que se comunica con los módulos *onu_rx_report* (contenido en el módulo compuesto *onu_source*) y *onu_rx*.

- *onu_ptp*: Este módulo se encarga de comprobar el identificador de los mensajes que recibe la *ONU* desde el *OLT* (mensajes *Gate* y *Ethernet*) y discernir si ésta es la destinataria de los mismos, ya que todo mensaje que ha llegado al *Splitter* procedente del *OLT* es duplicado y enviado a todas las *ONUs*. Si la *ONU* en

cuestión es el destino de un mensaje, éste se envía hacia el módulo *onu_mac*. En caso contrario, el mensaje es eliminado.

- *onu_rx*: Módulo simple cuyo único cometido es recibir y eliminar los paquetes *Ethernet* que la capa MAC de la *ONU* recibe desde el *OLT*, procedentes de la red troncal. En las simulaciones realizadas, sólo se genera y envía tráfico en el sentido ascendente, esto es, de los usuarios finales al *OLT*, por lo que este módulo no llega a utilizarse. Sin embargo, en el caso de que se quisieran analizar las prestaciones de la red en el canal descendente, este módulo sería el encargado de calcular las estadísticas del tráfico que llega desde el *OLT* para luego analizar las prestaciones de la red EPON en dicho canal.

- *onu_source*: Módulo compuesto por el módulo simple *onu_rx_report* y el array de módulos simples *onu_gentraffic[]*. Los módulos de este array se crean dinámicamente, en función del número de clases de servicio soportadas por la red de acceso (parámetro definido en el archivo de configuración). Tienen como función generar los paquetes Ethernet mediante distintas estrategias (fuentes *Self-Similar* y CBR) e insertarlos en el módulo *onu_sistqueue[]* de la misma prioridad, según la política de inserción de paquetes elegida (por colas separadas o por prioridad de colas). Cabe indicar que el índice de cada módulo *onu_gentraffic[]* dentro del array corresponde con la prioridad de los paquetes que genera, siendo "0" la más alta. Por su parte, el módulo *onu_rx_report* se encarga de extraer paquetes *Ethernet* de los módulos *onu_sistqueue[]* cuando recibe el mensaje *Report* que crea el módulo *onu_mac*, según la política de extracción de paquetes elegida (centralizado o de prioridad estricta). Además, actualiza el campo correspondiente al tamaño de las colas en el paquete *Report* antes de insertarlo en la cola que almacena este tipo de mensajes.

- *onu_squeue*: Módulo compuesto por un array de módulos simples *onu_sistqueue[]*. La longitud de este array es igual al número de prioridades o clases de servicio definidas en el fichero de configuración, más uno. Cada módulo *onu_sistqueue[]* modela una cola que almacena los paquetes destinados al *OLT* con una misma prioridad (procedentes del mismo módulo *onu_gentraffic[]*), coincidiendo ésta con el índice del módulo dentro del array (*0, 1, ..., n-1*). El

último módulo, en cambio, almacena los mensajes *Report* que genera la capa MAC de la *ONU* periódicamente, con destino al *OLT*.

- *onu_table*: Módulo simple que contiene como atributo principal un array bidimensional en el que se almacenan los datos relacionados con el ancho de banda asignado para transmitir y el instante de tiempo de inicio de la transmisión para cada *ONU*. El módulo *onu_mac* es el encargado de actualizar los valores de esta tabla, lo cual hace cada vez que recibe un mensaje *Gate*, accediendo de forma remota.

- *onu_wdm*: Este módulo simple funciona como divisor óptico pasivo o s*plitter* dentro de la estructura interna de la *ONU*. El funcionamiento principal de este módulo se basa en separar el canal *downstream* del *upstream* dentro de la estructura de la *ONU*. En el sentido ascendente, envía los paquetes *Ethernet* y *Report* que le llegan desde el módulo compuesto *onu_squeue* hacia fuera de la *ONU*. Por el contrario, en el sentido descendente, envía los mensajes *Gate* que le llegan desde el *Splitter* de la red hacia el módulo *onu_ptp* para que compruebe si esos mensajes van dirigidos a dicha *ONU* o no.

4.4 Parámetros definidos en el archivo de configuración *omnetpp.ini*

En este apartado se presentan las características de los parámetros definidos en el archivo de configuración *omnetpp.ini*, y las funciones que ejecutan dependiendo del valor que se asigne a su definición. De esta manera, se obtiene toda la información necesaria para poder simular la red EPON con las características deseadas.

- *numOnu*: Número de módulos *ONU[]* que contiene la red simulada. Para las simulaciones realizadas en este proyecto, se ha asignado el valor de “16” a este parámetro.

- *numlong*: Número de longitudes de onda con las que trabaja la red. Para trabajar con el protocolo de contienda TDMA, ha de darse a este parámetro el valor numérico “1”. Si por el contrario, se quisiera emplear WDMA, se daría a este parámetro el valor del número de longitudes de onda con el que se quiera trabajar.

- *longpon1*: Longitud, en metros, del canal *pon1*, que conecta los módulos *OLT* y *Splitter*. Este parámetro se ha definido de manera genérica para que dicho canal pueda tener la longitud que se desee.

- *longpon2*: Longitud, en metros, del canal *pon2*, que conecta el módulo *Splitter* con cada uno de los módulos *ONU[]*. Este parámetro se ha definido de manera genérica para que dicho canal pueda tener la longitud que se desee.

- *tambuffer*: Valor numérico, en bytes, asignado al tamaño total del *buffer* utilizado para insertar paquetes en los módulos *onu_sistqueue[]*. Se le puede asignar cualquier valor, pero se ha elegido fijar su tamaño en 10 Mbytes para el algoritmo de *polling* IPACT.

- *node_load*: Parámetro que define la carga de cada fuente de tráfico óptico en los submódulos *onu_gentraffic[]*. Se le puede asignar cualquier valor numérico comprendido entre 0 y 1.

- *txrate*: Capacidad de transmisión de la subred óptica, en bits. Al seguir el estándar EPON, este parámetro vale "1*1000000000" (1 Gbit/s).

- *numqueue*: Número de clases de servicio o prioridades que soporta la parte óptica de la red. Definiendo este parámetro, también se determina el número módulos *onu_sistqueue[]* y *onu_gentraffic[]* que tendrá cada *ONU*.

- *numSLA*: Número de *SLAs* definidos en la subred EPON. Los *SLA* son niveles de servicio asociados a los usuarios, y definen una jerarquía para que los abonados al *SLA* más prioritario reciban mayor asignación de ancho de banda.

- *w_sla*: Parámetro que define el peso asociado a cada *SLA*. En el archivo de configuración este parámetro se define como *w_slaN*, siendo *N* el subíndice del *SLA* al que está asociado. Estos pesos están directamente relacionados con el ancho de banda o requisitos de calidad asociados a nivel de prioridad de abonado.

- *BW_garantizado*: Parámetro que define el ancho de banda mínimo garantizado a cada *SLA*, en MBytes. En el archivo de configuración este parámetro se define como *BW_garantizadoN*, siendo *N* el subíndice del *SLA* al que está asociado.

- *numonu_sla*: Parámetro que define el número de módulos *ONU[]* asociados a cada *SLA*. En el archivo de configuración este parámetro se define como *numonu_slaN*, siendo *N* el subíndice del *SLA* correspondiente.

- *insercionmethod_separatequeue0_priorityqueue1*: Parámetro para elegir el método de inserción de paquetes en los submódulos *onu_sistqueue[]*. Si se le asigna el valor 0, se implementa el mecanismo de inserción en colas separadas; mientras que el valor 1 hace que se implemente el método de inserción de paquetes con prioridad de colas.

- *extractionmethod_StrictPQ0_Centralized1*: Parámetro que define el método de extracción de paquetes de los submódulos *onu_sistqueue[]*. Si vale 0, se implementa el método de extracción de colas por prioridad estricta; y si vale 1 se implementa el método de extracción centralizado.

- *oltmethod_Centralized0_Polling1_wdm2_PollingPID3_DaSPID4*: Parámetro que determina el algoritmo DBA centralizado o de *polling* que se quiere ejecutar en la parte óptica de la red simulada. Si se le asigna el valor 0, se implementa el algoritmo centralizado DMB. En cambio, si se le asigna los valores 1, 2, 3 ó 4, se implementa los algoritmos de *polling* implementados en el simulador.

- *methodlength_longvariable0_longfija1*: Parámetro para elegir el método para establecer la longitud de los canales *pon2*. Si se le da el valor 0, se ejecuta el método de longitud variable, por lo que cada canal entre cada *ONU[]* con el *Splitter* tendrá un retardo de propagación distinto. Si se le da el valor 1, se ejecuta el método de longitud fija para todos los canales *pon2*, por lo que todos tendrán el mismo retardo de propagación.

- *numstreamV2_32_128_256*: Parámetro que fija el número de *streams* definidos para cada fuente generadora de tráfico en la parte óptica de la red. Estos *streams* se utilizan para generar los paquetes *Ethernet* a ráfagas en los generadores *Self-Similar* implementados en los módulos *onu_gentraffic[]*. Los posibles valores de este parámetro que se han definido en la red son 32, 128 y 256.

- *longpacketfixed0_trimodal1*: Parámetro que define el tamaño de los paquetes *Ethernet* generados en los módulos *onu_gentraffic[]*. Si vale 0, todos los paquetes

generados tendrán el mismo tamaño; mientras que si vale 1, el número de *streams* definidos en cada fuente se divide para generar paquetes de tres tamaños diferentes.

4.5 Conclusiones

El simulador de redes de acceso ópticas con el que se trabaja en esta memoria, está constituido por una red PON formada por un elemento óptico central, denominado O*L*T, con el que se conecta un array de 16 unidades ópticas, llamadas *ONUs*. Estos elementos se disponen siguiendo una topología en árbol, para lo cual se incluye un divisor óptico pasivo o *splitter*. Todos los enlaces de la parte óptica poseen las características definidas en el estándar EPON para redes de acceso ópticas pasivas. Todo ello aparece recogido en los Apartados 4.2 y 4.3 de este capítulo.

Más adelante, en el Apartado 4.3 de este capítulo se ha descrito superficialmente cada uno de los módulos que componen el modelo de red simulado. Todos estos módulos se comunican entre sí mediante el envío y recepción de mensajes de diferente tipo (*Ethernet*, *Gate* y *Report*), para lo cual es necesario implementar unos mecanismos que regulen la inserción y la extracción de paquetes de cada una de sus colas, los cuales serán explicados en capítulos posteriores. Por último, en el Apartado 4.5 se han definido los parámetros que recogidos en el fichero de configuración de la red óptica.

5

Análisis del tráfico *self-similar* en algoritmos DBA

5.1 Introducción

Este capítulo se centra en el análisis de prestaciones de algoritmos de asignación dinámica de recursos en redes PON, en concreto algoritmos de asignación dinámica de ancho de banda (DBA, *Dynamic Bandwidth Allocation)*, bajo patrones de tráfico realista, esto es, tráfico auto-semejante o *self-similar.* Más adelante, en este mismo capítulo, se presenta una breve reseña de las características de este tipo de tráfico junto a la implementación de la fuente de tráfico *Self-Similar* en la red óptica.

Análogamente, se estudiarán las prestaciones de uno de estos algoritmos DBA, llamado IPACT (*Interleaved Polling with Adaptative Cycle Time*), considerando diferentes parámetros de la fuente de tráfico auto-semejante y otros parámetros de ejecución asociados al diseño de dicho algoritmo de gestión de recursos dentro de la red PON.

Así mismo se expondrán las distintas simulaciones realizadas con el algoritmo IPACT para poder analizar el impacto del mismo ante el tráfico *self-similar*. Se describirán las características de los diversos escenarios para cada simulación realizada, ya que cada una se configura individualmente y está relacionada con diversos parámetros de la red.

Por último, se expondrán los resultados obtenidos para las diversas simulaciones, profundizando concretamente en el retardo medio de los paquetes para los distintos perfiles de usuario y clases de servicio.

5.2 Fuentes de tráfico *Self-Similar* en la red PON

En esta sección se presenta el funcionamiento y características de la fuente de tráfico *Self-Similar* implementada en nuestro simulador de redes de acceso PON. En concreto, se ha implementado el generador de tráfico desarrollado por *Glen Kramer,* para la simulación del tráfico real transmitido en el canal *upstream*, es decir, desde los abonados hasta la estación principal.

El tráfico *Self-Similar* o auto-semejante [14] se basa en ráfagas de paquetes, generados siguiendo siempre un patrón similar. Para ello, se implementan múltiples fuentes generadoras de tráfico siguiendo una distribución de Pareto [15], y de la combinación de todas estas fuentes de Pareto, se obtiene el tráfico auto-semejante. Más concretamente, en la literatura pueden encontrarse tres versiones distintas del generador de tráfico *Self-Similar,* cada una más compleja pero también más eficiente computacionalmente que la anterior [16][17][18].

La versión implementada en nuestra red óptica simulada es la segunda, realizando así un compromiso entre complejidad y eficiencia. En ella, se trabaja con números aleatorios obtenidos a partir del generador *MersenneTwister*. Este generador permite la creación de números aleatorios y permite la reducción de los efectos de truncamiento de la cola en los generadores de Pareto.

De esta manera para generar cada una de las distribuciones de Pareto se requiere pasar una serie de parámetros, tales como la carga de tráfico, la duración de las ráfagas (que son los periodos en *ON*) y el tamaño de los paquetes. Respecto a este último parámetro conviene destacar que la versión implementada en el simulador se ha visto modificada de tal manera que permita la generación de paquetes con distinta longitud.

5.3 Descripción del algoritmo IPACT

En este apartado se procede a la descripción de uno de los algoritmos de asignación dinámica de ancho de banda (DBA) de *polling* conocido como IPACT.

Dentro de los algoritmos DBA existen dos políticas adoptadas, *polling* y centralizada. Mediante la política de *polling* se lleva a cabo un testeo continuo, de modo

que se aprovecha el ancho de banda entre ciclos consecutivos. Además permite la asignación del ancho de banda en función del estado de cada *ONU*. Estos aspectos resultan realmente interesantes si lo comparamos con la política centralizada, donde el ancho de banda entre ciclos consecutivos se desperdicia, ya que se realiza la asignación teniendo en cuenta el estado de todas las *ONUs*, por lo que puede resultar una técnica más ineficiente que la de *polling.* Para ejemplificar el funcionamiento de la política de *polling* acudimos a la Figura 9.

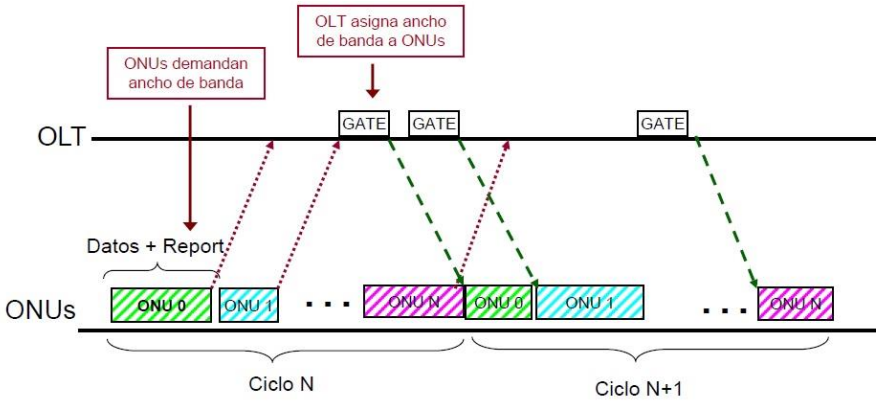


Figura 9. Comportamiento de la política de *polling*.

En ella podemos ver que no es necesaria la recepción de todos los *Report* por parte de las *ONUs* para ir asignando el ancho de banda disponible. De este modo en función de la demanda de ancho de banda de las *ONUs*, recogida en el mensaje *Report*, el *OLT* les asigna su capacidad respectiva a través del envió del mensaje *Gate*.

IPACT fue diseñado para realizar una asignación dinámica de ancho de banda ciclo tras ciclo sin tener que esperar a recibir la demanda de todos los usuarios, lo cual permite no desperdiciar tiempo entre ciclos consecutivos.

Este algoritmo es implementado en la capa MAC del *OLT* en una red EPON y se ejecuta en dicha capa cada vez que recibe un mensaje *Report* de cada *ONU*. Dentro de la simulación, para que se ejecute este algoritmo y no otro de los ya implementados, el parámetro *oltmethod_centralized0_Polling1* tiene que tener asignado en el archivo de configuración el valor numérico “1”.

Con el objetivo de soportar diferentes tipos de tráfico de distintos tamaños y prioridades de los paquetes, este algoritmo de *polling* ofrece diferenciación de servicios. Además, IPACT soporta la diferenciación de usuarios en niveles de servicio denominados *SLA* (*Service Level Agreement*). Estos niveles soportan una jerarquía de

prioridad para asignar mayor ancho de banda al abonado asociado al *SLA* más prioritario. Para la implementación de nuestro algoritmo de *polling* IPACT, se ha determinado la utilización de tres *SLAs* asignados a todas las *ONUs* de la red EPON con el peso asociado de valor "1", lo cual quiere decir que todos tienen la misma prioridad.

5.3.1 Asignación de Ancho de Banda

La asignación de ancho de banda en el algoritmo IPACT se realiza de manera dinámica en la capa MAC del *OLT*. Para ello, en primer lugar se ha de conocer el ancho de banda demandado por la *ONU* (recibido en el mensaje *Report*) y, como cada *ONU* puede soportar varias clases de servicio, tenemos que calcular el ancho de banda total demandado por la *ONU* sumando el tamaño total de sus colas. Esto se muestra en la Ecuación 1, donde Q_{ij} representa el tamaño de la cola perteneciente a la clase de servicio *j* para la ONU_i:

$$B_{demand}^{onu_i} = \sum_j Q_{i,j} \qquad \text{Ecuación 1}$$

Por otro lado, es necesario conocer el ancho de banda máximo que es posible asignar a cada *ONU* en un ciclo dependiendo del *SLA* que tenga asociado, el cual es fijo en cada ciclo. Para calcular el ancho de banda, recurrimos a la Ecuación 2.

$$B_{demand}^{onu_i} = \frac{B_{cycle} \times W_k}{\sum_m W_k \times N_{ONUs}^{m_i}} \qquad \text{Ecuación 2}$$

Donde $N_{ONUs}^{m_i}$ es el número de *ONUs* asociadas al SLA_m, W_m es el peso asociado al SLA_m, B_{cycle} es el ancho de banda de transmisión contenido en cada ciclo máximo (siendo este de 2 ms en el estándar EPON) y W_k es el peso asociado a SLA_k.

En el momento en que tenemos estos dos parámetros, se lleva a cabo la asignación de ancho de banda en cada ciclo. Para ello, en el momento en que la capa MAC del OLT recibe un mensaje *Report* de una ONU_i, se compara el ancho de banda demandado de la ONU_i con el máximo que se le puede asignar a dicha *ONU* según el *SLA* al que esté asociada:

- Si $B_{demand}^{onu_i} \leq B_{\max}^{k}$, el ancho de banda demandado para la ONU_i es menor que el máximo para esa *ONU* asociada al SLA_k. Por lo tanto, el ancho de banda asignado corresponde con el demandado por dicha *ONU*, tal y como se observa en la Ecuación 3:

$$B_{allocated}^{onu_i} = B_{demand}^{onu_i} \qquad \text{Ecuación 3}$$

- Si $B_{demand}^{onu_i} > B_{\max}^{k}$, entonces en este caso el ancho de banda demandado para la ONU_i es mayor que el máximo que se le puede dar a esa *ONU* asociada al SLA_k. De este modo, el ancho de banda asignado será el máximo que se le pueda dar a dicha *ONU*, según al *SLA* al que corresponda, tal y como vemos en la Ecuación 4:

$$B_{allocated}^{onu_i} = B_{\max}^{k} \qquad \text{Ecuación 4}$$

Se observa entonces que el ancho de banda se asignará a cada *ONU* en función de la demanda de todas sus colas. Una vez realizada la asignación de ancho de banda, nos queda asignar a cada *ONU* un tiempo de inicio de transmisión, ya que IPACT utiliza TDMA en el canal compartido ascendente.

5.3.2 Asignación del tiempo de inicio de transmisión en cada ciclo

Además de asignar el ancho de banda en IPACT, se necesitar asignar el instante de tiempo en el que las *ONUs* empiezan a transmitir los paquetes Ethernet insertados en sus colas para evitar la colisión de paquetes en el canal *upstream*. Recordemos que esto es debido a que los usuarios de nuestra red comparten la misma longitud de onda para la transmisión de datos hasta la central (*OLT*), y por tanto no pueden transmitir datos simultáneamente. De este modo, mediante la asignación del tiempo de inicio de la transmisión, se consigue una compartición libre de colisiones en el canal, permitiendo además un uso más eficiente de los recursos del mismo.

Esta asignación, depende del instante de tiempo de finalización de la transmisión de la *ONU* previa (ONU_{i-1}) en el orden de transmisión fijo ciclo tras ciclo (T_{tx_ONUi-1}).

Para calcular el instante de tiempo de inicio de la ONU_i, $T_{ini\_tx\_ONUi}$, se comparará el instante de finalización de la *ONU* anterior, $T_{tx\_ONUi-1,}$ con el instante de tiempo compuesto por la suma del tiempo de simulación en el instante en el que se crea el

mensaje *Gate* (simTime()), más el retardo de ida de la red (*RTT/2*), más la tasa de transmisión del mensaje creado (T_{GATE}). Esta asignación del tiempo inicial de transmisión depende de dos casos:

- Si la ONU_i no tiene que esperar a la transmisión de ONU_s previas.
  En este caso, el instante de finalización de la transmisión de la ONU_{i-1}, T_{tx_ONUi-1} es menor que la suma de $simTime() + RTT/2 + T_{GATE}$. Por lo tanto, el instante de tiempo de inicio de la transmisión de la ONU_i ($T_{ini\_tx\_ONU}$), es el tiempo que nos indica la Ecuación 5:

$$T_{ini\_tx\_ONU} = simTime() + RTT/2 + T_{GATE} \qquad \text{Ecuación 5}$$

  Este tiempo nos indica que cuando llegue el mensaje *Gate* de la ONU_i a la capa MAC de la *ONU*, ésta comienza a transmitir debido a que el canal de transmisión en ese instante de tiempo está libre, tal y como se observa en la Figura 10.

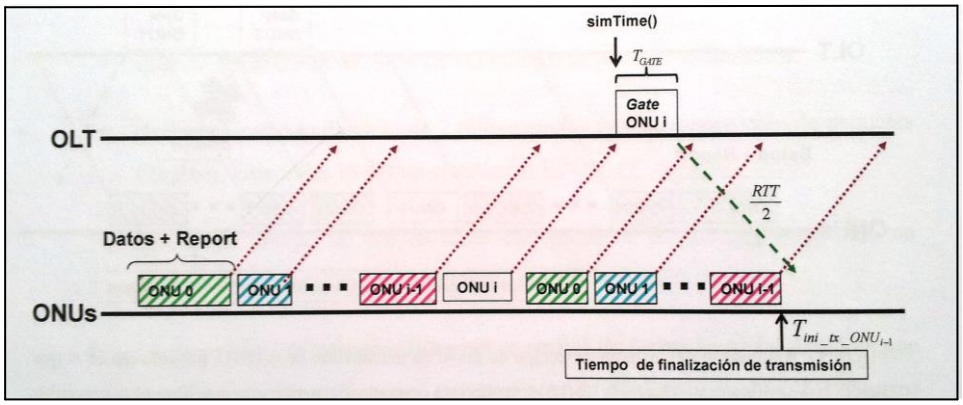


Figura 10. Asignación del instante de tiempo de inicio de transmisión de la ONU_i para el caso en el que cuando llegue un mensaje *Gate* a la *ONU*, el medio está libre y puede transmitir.

  Una vez asignado este tiempo de inicio de la transmisión para la *ONU* e introducido en el mensaje *Gate*, junto con el ancho de banda asignado para ella, se envía dicho mensaje para la *ONU* correspondiente.

- Si la ONU_i tiene que esperar a que terminen de transmitir ONU_s previas.
  En este caso, el instante de finalización de la transmisión de la ONU_{i-1} T_{tx_ONUi-1} es mayor que la suma de $simTime() + RTT/2 + T_{GATE}$. Por lo tanto, el instante de tiempo de inicio de la transmisión de la ONU_i ($T_{ini\_tx\_ONU}$), es el instante de tiempo

en el que finaliza la transmisión de la ONU_{i-1} previa, tal y como se observa en la Ecuación 6:

$$T_{ini\_tx\_ONU} = T_{tx\_ONUi-1} \qquad \text{Ecuación 6}$$

Este tiempo nos indica que cuando llegue el mensaje *Gate* de la ONU_i a la capa MAC de la *ONU*, observa que el canal está ocupado, tal como se observa en la Figura 11. Esta *ONU* espera a que se termine de transmitir la ONU_{i-1} y que el medio esté libre, y comienza a transmitir sus paquetes debido a que el canal de transmisión en ese instante de tiempo está libre, lo cual es visible en la Figura 11.

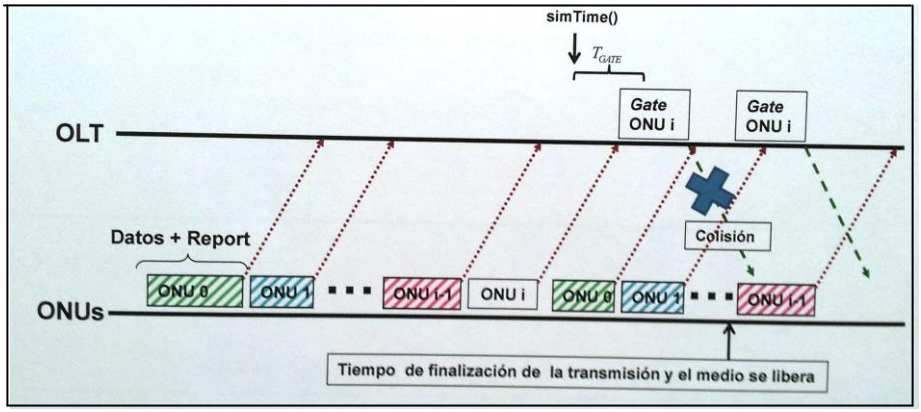


Figura 11. Asignación del instante de tiempo de inicio de transmisión de la ONU_i para el caso en el que cuando llegue un mensaje *Gate* a la *ONU*, el medio está ocupado y espera a que esté libre para transmitir.

Una vez asignado este tiempo de inicio de transmisión para la *ONU* e introducido en el mensaje *Gate*, junto con el ancho de banda asignado para ella, se envía dicho mensaje para la *ONU* correspondiente.

5.4 Escenario de simulación genérico para IPACT

Para ejecutar el algoritmo IPACT, se ha planteado un escenario de simulación en el que los parámetros más significativos toman los valores que a continuación se definen:

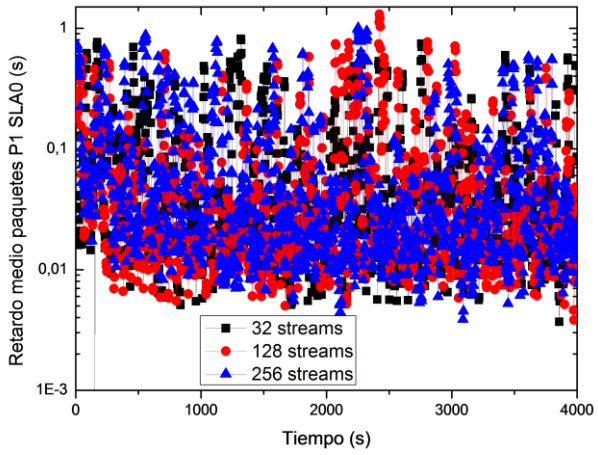
- El método DBA implementado en la capa MAC del *OLT* es IPACT (*oltmethod_Centralized0_Polling1_wdm2_PollingPID3_DaSPID4=1*).
- La subred óptica consta de 16 *ONU*s (*numOnu=16*), cada una de ellas con su correspondiente índice, un número entero entre 0 y *numOnu*-1.

- Se definen 3 niveles de servicio o *SLAs* (*numSLA=3*), todos con peso unitario (*w_sla$_0$=w_sla$_1$=w_sla$_2$=1*).
- Tres servicios de prioridad, de modo que cada *ONU* consta de 3 fuentes de tráfico óptico (*numqueue=3*), cada una perteneciente a una clase de servicio distinta (P_0, P_1 y P_2). La clase P_0, siendo ésta la más prioritaria, genera paquetes de 70 Bytes a una tasa constante de 4.48 Mbps, mientras que las fuentes P_1 y P_2 generan tráfico *Self-Similar* con paquetes de tamaño de 64, 594 y 1500 bytes (*longpacketfixed0_trimodal1=1*).
- La carga de la fuente P_0 (tráfico CBR) es de 0,0448 y la de las fuentes P_1 y P_2 (tráfico self-similar) se dividen el resto de la carga total equitativamente.
- Las 16 *ONU*s se distribuyen entre los *SLAs* de la siguiente manera: 2 en el SLA_0 (*numonu_sla0=2*), 6 en el SLA_1 (*numonu_sla1=6*) y 8 en el SLA_2 (*numonu_sla2=8*).
- La tasa de transmisión de datos en la subred EPON es de 1 Gbps (*txrate=1*1000000000*).
- La duración de cada ciclo de transmisión de paquetes Ethernet por parte de las 16 *ONUs* hacia el *OLT* es de 2 ms (*T_cycle=0.002*).
- La longitud de la ventana deslizante (T_{window}) será de 10, 50, 100 ó 150 segundos.
- El número de streams asociado al tráfico *Self-Similar* generado por el usuario (*numstreamV2_32_128_256=32)* tendrá los valores de “32”, “128”, ó “256”.

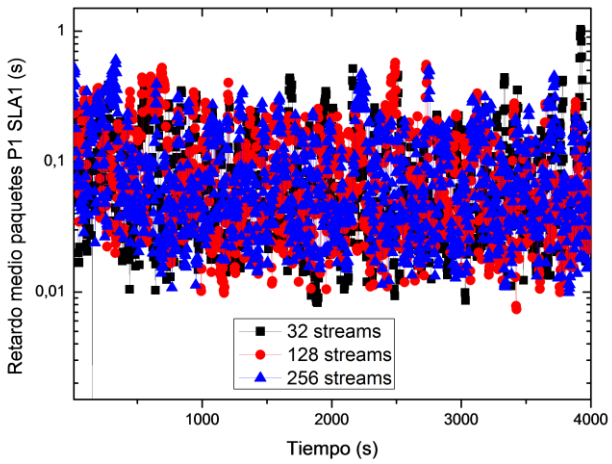
5.5 Análisis de resultados

En un primer momento procedemos al estudio del retardo medio de la ventana del servicio P_1, con tráfico *Self-Similar*, para cada uno de los tres *SLAs* existentes, cuando se modifica el número de *streams* asociados a la fuente de tráfico auto-semejante. El motivo de llevar a cabo el análisis para el servicio P_1 es que éste experimentará cierta variación por tener una característica rafagos y al ser tráfico prioritario su caracterización es importante para poder garantizar unos determinados niveles de QoS.

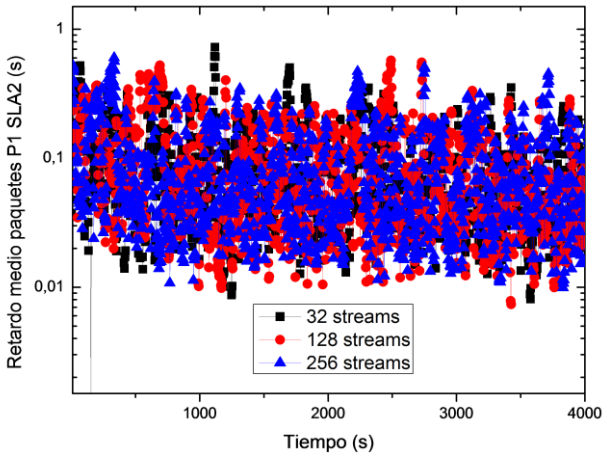
En la Figura 12 (a), (b) y (c) se muestra la evolución del retardo medio de P_1 para cada uno de los tres *SLAs* (en (a) se muestra respecto al SLA_0, en (b) respecto al SLA_1 y finalmente en (c) para el SLA_2), cuando se considera diferente número de streams.



(a)



(b)



(c)

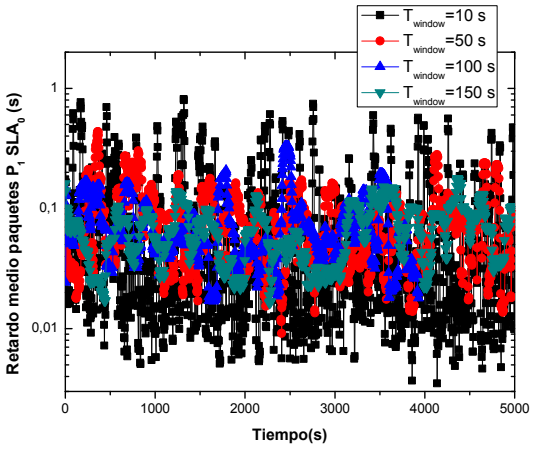
Figura 12. Comparativa de la evolución del retardo medio de los paquetes de clase P_1 para (a) SLA_0 (b) SLA_1 (c) SLA_2 dados distintos valores de *streams*.

Tal y como vemos en las gráficas la conclusión más importante que se puede extraer es que con independencia del número de *streams* que se utilicen para generar la fuente *self-similar*, los resultados obtenidos para el retardo medio van a ser prácticamente iguales. Así pues, el comportamiento del algoritmo IPACT es similar ante la variación del número de *streams* empleados. Por lo tanto, a partir de este momento y para agilizar el tiempo de simulación, tomaremos el menor número de *streams*, esto es, 32.

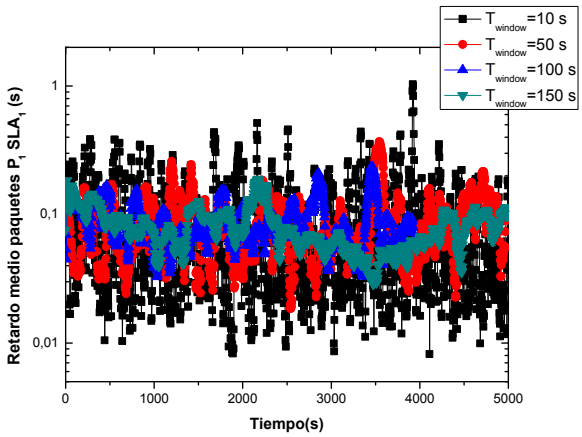
Por otro lado, los resultados obtenidos demuestran que IPACT obtiene una gran fluctuación de los niveles de retardo entre un máximo y un mínimo, debido fundamentalmente al carácter rafagoso de dicha fuente de tráfico.

A continuación procedemos a analizar cómo se comporta el algoritmo IPACT en función del tamaño de la ventana (T_{window}) y considerando 32 *streams*.

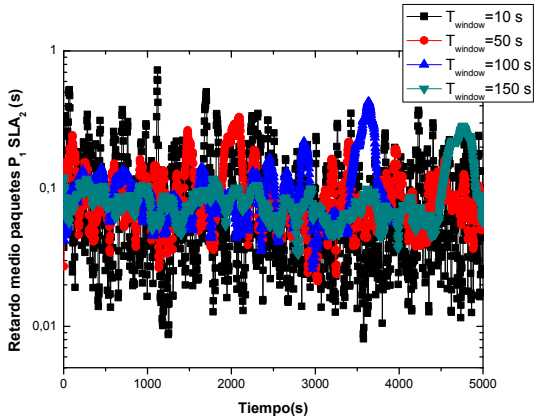
Por consiguiente, en la Figura 13 (a) (b) (c) se refleja la dependencia del retardo medio del tráfico P_1 para cada uno de los tres *SLAs* existentes, dados unos tamaños de ventana de 10, 50, 100 y 150 segundos.



(a)



(b)



(c)

Figura 13. Comparativa de la evolución del retardo medio de los paquetes de clase P_1 para (a) SLA_0 (b) SLA_1 (c) SLA_2 dados unos tamaños de ventana de 10, 50, 100 y 150 segundos.

Si observamos las gráficas mostradas, vemos una variación distinta de los valores alcanzados por el retardo en función del tamaño de ventana que se seleccione. Vistos los resultados de las gráficas, resulta conveniente la selección de un valor de la ventana intermedio, por ejemplo de 50 segundos. Esto es debido a que, para dicho valor del tamaño de la ventana, los valores alcanzados por el retardo no fluctúan demasiado. Además, la selección de este valor intermedio (ni muy bajo como puede ser 10 segundos, ni muy alto como ya lo son a partir de 100 segundos) permitirá una adaptación más o menos rápida del algoritmo a nuevas condiciones de red, ya que si dicha ventana fuera muy grande, el número de paquetes con el que se trabajaría podría hacer que cambios abruptos en las características del tráfico no se vieran reflejados con suficiente rapidez en el valor medio de la ventana. Por otro lado, si la ventana fuera muy pequeña, el número de muestras no sería demasiado significativo y la media del retardo fluctuaría demasiado, tal y como se aprecia para una ventana de 10 segundos.

5.6 Conclusiones

En este capítulo se han analizado las prestaciones de un algoritmo de asignación dinámica de ancho de banda en redes PON, bajo patrones de tráfico auto-semejante modificando diferentes características asociadas al tráfico auto-semejante. Por otro lado, se ha llevado a cabo una explicación de este tipo de tráfico así como de la fuente

generadora del mismo, para aclarar y exponer en qué consisten y por qué han sido utilizados en este estudio.

Además de todo ello, se ha llevado a cabo un estudio del comportamiento del algoritmo de *polling* conocido como IPACT también bajo el patrón de tráfico auto-semejante. Este algoritmo permite la asignación dinámica de ancho de banda ciclo tras ciclo para redes EPON en OMNeT++

Con el fin de analizar el comportamiento del algoritmo ante distintos tamaños de *streams* y bajo un patrón de tráfico auto-semejante, se ha realizado una simulación en la que se exponían los diferentes resultados, extrayendo de ellos que el retardo medio para los tres *SLAs* es independiente del número de *streams* empleado, de modo que el algoritmo IPACT no se ve afectado por el número de *streams* utilizado. Así pues, se selecciona el valor de 32 *streams* para los posteriores estudios con este algoritmo, evitando tener una carga computacional elevada con un número mayor, puesto que los resultados obtenidos serían aproximadamente los mismos.

Finalmente, debido al carácter rafagoso de la fuente de tráfico auto-semejante empleada, se ha visto que IPACT presentaba una gran fluctuación de los niveles de retardo, de modo que se llevó a cabo un análisis del comportamiento del algoritmo ante este tipo de tráfico, considerando diferentes tamaños de ventana donde se almacenan las muestras del retardo medio. De esta manera se llegó a la conclusión de que el tamaño de ventana sí influye en gran medida. Por lo tanto, resulta conveniente la selección de tamaños de ventana intermedios para tener estabilidad del algoritmo, ya que para ventanas pequeñas se produce mucha fluctuación en el retardo medio. Sin embargo, para tamaños de ventana demasiado grandes la adaptación a diferentes condiciones de red y tráfico podría ser un proceso adaptativo más lento y costoso.

6

Análisis del algoritmo DaSPID para el control de retardo bajo patrones de tráfico auto-semejante

6.1 Introducción

Uno de los objetivos principales dentro de la subred óptica basada en tecnología PON, con la que se trabaja en este documento, es la integración de mecanismos de control de QoS para poder garantizar los requisitos de todos los usuarios extremo a extremo. De esta manera, en este Proyecto Fin de carrera se va a llevar a cabo un estudio de las prestaciones proporcionadas por un algoritmo ya implementado en el simulador de redes PON. Este algoritmo trata la asignación de recursos para controlar en la parte óptica el retardo de servicios y usuarios prioritarios acordes a unos umbrales preestablecidos por los proveedores de servicio. Así pues, en este capítulo, se presenta el algoritmo DaSPID (*Delay aware Service level agreement PID*), basado en un controlador PID (*Proportional Integral Derivate*) para gestionar y controlar los recursos disponibles en la subred EPON con la finalidad de garantizar unos mínimos requisitos de QoS (*Quality of Service*). Dicho algoritmo ha sido desarrollado previamente por el GCO en el simulador OPNET *Modeler* y es capaz de gestionar de forma automática el ancho de banda disponible para garantizar unos requisitos mínimos retardo a ciertos abonados y servicios mediante un controlador PID.

Análogamente, una vez analizadas las prestaciones del algoritmo en sí, considerando diferentes parámetros de ejecución asociados al diseño de dicho algoritmo, se exponen los aspectos más significativos observados, así como una serie de mejoras a dicho algoritmo con el escenario de simulación propuesto.

6.2 Descripción e implementación del algoritmo DaSPID (*Delay aware Service level agreement PID*)

En la presente sección, se explican inicialmente los métodos que se han utilizado para la extracción e inserción de paquetes en las colas de los nodos ópticos de la red (*ONUs*). Además, se procede a la descripción del algoritmo de control de retardo en redes EPON, denominado DaSPID, desarrollado por el GCO. Éste es un algoritmo basado en un controlador PID que gestiona de forma óptima la asignación del ancho de banda a todos los usuarios finales (*ONUs*) teniendo en cuenta tanto el retardo máximo permitido a la clase de servicio a la que pertenece el tráfico, así como el *SLA* contratado por el usuario que lo genera. En la literatura pueden encontrarse otros algoritmos que proporcionan garantías respecto al retardo, aunque no cubren tantos aspectos como DaSPID [19][20][21]. Por lo tanto, este algoritmo goza de una gran potencialidad ya que es capaz de ofrecer una Calidad de Servicio óptima a partir del control robusto y eficaz del retardo de servicios y usuarios prioritarios a través de un controlador PID.

6.2.1 Métodos de inserción y extracción de paquetes en la red EPON

En esta sección se explican brevemente los métodos para insertar y extraer paquetes de las colas de los nodos ópticos (módulos *ONU*), implementados en el simulador de redes EPON disponible, ya que serán necesarios para la ejecución del algoritmo DaSPID bajo patrones de tráfico *self-similar*.

6.2.1.1 Método de inserción de paquetes de prioridad estricta

Los usuarios (*ONUs*) de la red estudiada generan una serie de datos, los cuales son almacenados en paquetes, que serán enviados a la estación central (*OLT*). Estos paquetes han de ser insertados en colas y para ello se dispone de los métodos de inserción de paquetes, los cuales se encargan de insertar los paquetes generados en las colas pertinentes. Estos métodos se encuentran en los módulos *onu_sistqueue[]* y *onu_gentraffic[]* generados de forma dinámica, y pretenden controlar el estado de las colas del módulo *onu_sistqueue[]* para insertar en ellas nuevos paquetes sin superar el tamaño máximo del *buffer* (*tambuffer*) indicado en el fichero de configuración.

El módulo *onu_gentraffic*[] tiene por objetivo la invocación un método de inserción cada vez que se genera un nuevo paquete *Ethernet*, para enviarlo a la cola correspondiente del módulo *onu_squeue*.

La invocación al método de prioridad estricta se produce considerando el parámetro *insercionmethod_separatequeue0_priorityqueue1=1*. Además, se siguen los siguientes pasos para la implementación de este método:

1. Primeramente se obtiene del archivo de configuración *omnetpp.ini* el ancho de banda total del *buffer*.

2. A continuación se accede de manera remota al número de bytes que ocupan cada una de las colas, añadiendo cada valor a una variable auxiliar que almacena el número total de bytes presentes en todas las colas.

3. Se compara la capacidad máxima total del *buffer* con el número de bytes en todas las colas, al que se añade los bytes del paquete a insertar. En función del resultado de la comparación, y de la prioridad del nuevo paquete, aparecen diversas opciones:

    a) Si el tamaño del *buffer* es mayor que el número de bytes ocupados entre todas las colas, se inserta el nuevo paquete en la cola correspondiente.

    b) Si el tamaño del *buffer* es menor que la cantidad de bytes ocupados en las colas, según la prioridad del nuevo paquete:

        i. Si el nuevo paquete es de la prioridad más baja, se elimina.

        ii. Ahora bien, si el nuevo paquete no es de la prioridad más baja, se eliminan paquetes de las colas de prioridad inferior, hasta que quede espacio suficiente para insertar el nuevo paquete en la cola de la prioridad que le corresponde. En caso de haber eliminado todos los paquetes de las colas de menos prioridad, se elimina el nuevo paquete.

La Figura 15 representa un ejemplo del funcionamiento del método de inserción de paquetes de prioridad de colas. En este ejemplo, todas las colas están completas, por

lo que al intentar incluir el nuevo paquete, se excede el ancho de banda del *buffer*. Como el nuevo paquete es de prioridad P_1, se borran paquetes de la cola P_2 hasta que queda en el *buffer* suficiente espacio como para insertar el paquete en la cola P_1.

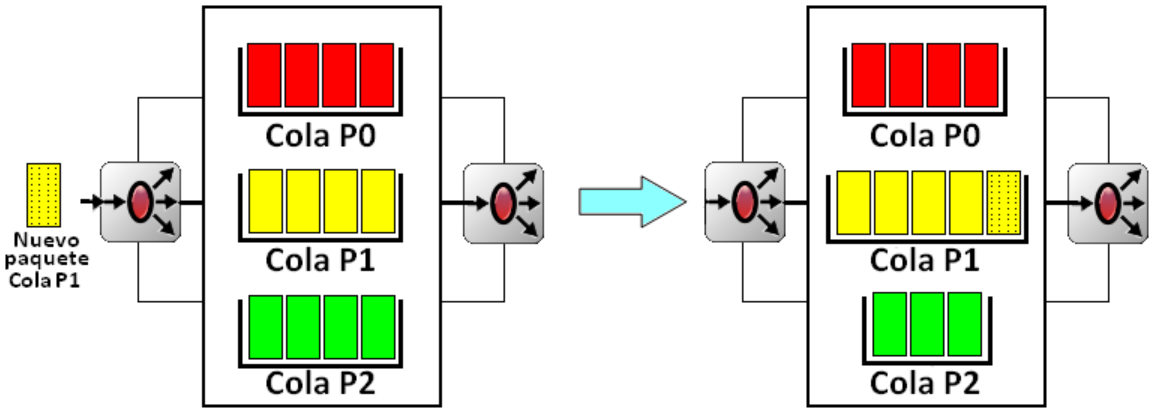


Figura 15. Método de inserción de paquetes de prioridad de colas.

6.2.1.2 Método de extracción de paquetes de prioridad estricta

Los métodos de extracción de paquetes son los encargados de extraer de las colas los paquetes almacenados, y se implementan en el módulo *onu_rx_report* para enviar hacia el siguiente nodo de la red (*ONU* u *OLT*) los paquetes permitidos en el ancho de banda asignado. Para las *ONU*s, el ancho de banda de transmisión en cada ciclo TDMA es asignado por el *OLT* según el algoritmo DBA implementado en su capa MAC.

El modelo de red EPON implementado, permite elegir entre dos métodos de extracción distintos, el método de colas de prioridad estricta y el método de colas centralizado, según el valor asignado al parámetro *extactionmethod_StrictPQ0_Centraliced1* en el fichero de configuración *omnetpp.ini*. Si se da valor "0" a este parámetro, se implementa el método de colas de prioridad estricta; mientras que si se elige "1", entonces se ejecuta el método de colas centralizado. En nuestro caso nos quedaremos con el de prioridad estricta.

El método de extracción de paquetes de prioridad estricta se invoca a la llegada del mensaje *Report* al módulo *onu_rx_report* en el instante de tiempo de inicio de la transmisión y sigue las siguientes fases:

1. Se calcula el ancho de banda correspondiente a un ciclo de transmisión, multiplicando el tiempo de ciclo por la tasa binaria del enlace.

2. Al iniciarse un nuevo ciclo de transmisión en un módulo, se extraen paquetes de sus colas, empezando por la cola de prioridad más alta, siempre que el número de bytes extraídos no exceda el ancho de banda de transmisión.

La Figura 16 muestra un esquema explicativo del método de extracción de paquetes de prioridad de colas estricta. Como puede apreciarse en este ejemplo, el tiempo de transmisión establecido permite el envío de siete paquetes, mientras que entre las tres colas se tienen once. Al extraer paquetes con el método de colas de prioridad estricta, las colas de prioridad más alta quedan vacías. Sin embargo, no se han extraído paquetes de la cola P_2, porque se excedía el ancho de banda total.

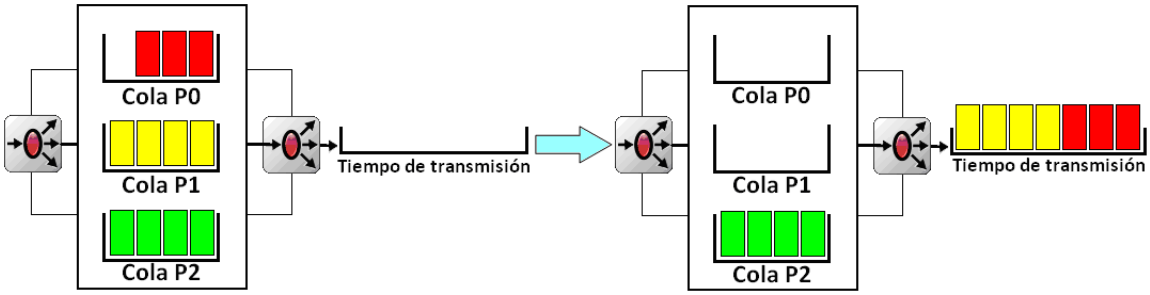


Figura 16. Método de extracción de paquetes de prioridad de colas estricta.

6.2.2 Diferenciación de servicios y usuarios en la red EPON

Uno de los puntos clave del algoritmo DaSPID es la diferenciación de servicios. De este modo, con el fin de poder llevarla a cabo, se emplea un esquema de prioridad estricta de colas con compartición de memoria, que incluye la implementación en las *ONUs* (siendo éstas los usuarios finales) de los métodos de inserción de prioridad de colas y de extracción de colas con prioridad estricta, explicados previamente en este capítulo. La idea de estos métodos es otorgar una mayor prioridad a aquellas clases de servicio que así lo requieren, de modo que el retardo medio conseguido para estas clases de servicio, sea menor.

Para lograr la clasificación del tráfico según el usuario que lo genera, DaSPID establece que los usuarios finales deben contratar un *SLA*, y cada *SLA* tiene asociados unos requisitos de *QoS* que el algoritmo ha de proporcionar. Ya que el objetivo es garantizar que cada tipo de tráfico esté situado por debajo de su umbral máximo, se establecerán diferentes niveles de retardo máximos en función de los requisitos

estipulados para cada clase de servicio y cada *SLA* considerado, de modo que pueda garantizarse un buen nivel de *QoS* a los usuarios finales.

6.2.3 Control de retardo y asignación de ancho de banda en DaSPID

Con el fin de cumplir con los requisitos mínimos de *QoS* demandados por los usuarios finales, este algoritmo realiza sondeos periódicos en los que las *ONUs* informan al elemento central, el *OLT*, del ancho de banda que van a requerir en el siguiente ciclo para transmitir los paquetes almacenados en sus colas, mediante mensajes del tipo *Report* (definidos en el fichero *REPORT.msg*). Para cada mensaje de este tipo que recibe el *OLT*, su capa de control de acceso al medio asigna a la *ONU* emisora de dicho mensaje el ancho de banda solicitado, en caso de ser menor que el máximo permitido para el *SLA* al que pertenezca ($B_{alloc}^{onu_i} = B_{demand}^{onu_i}$ *if* $B_{demand}^{onu_i} \leq B_{max}^{onu_i}$). En caso de ser mayor que su ancho de banda máximo estipulado, se le asignará dicho máximo ($B_{alloc}^{onu_i} = B_{max}^{onu_i}$ *if* $B_{demand}^{onu_i} > B_{max}^{onu_i}$). Por otro lado, es necesario asignar un valor inicial al citado ancho de banda máximo por *SLA*, para lo que se usa la Ecuación 7, que tiene en cuenta el peso asociado a cada *SLA* (W^{sla_k}), es decir, el nivel de prioridad del *SLA*.

$$B_{max}^{onu_i} = \frac{B_{ciclo} \cdot W^{sla_k / onu_i \in sla_k}}{\sum_k W^{sla_k} \cdot N_{onus}^{sla_k}} \qquad \text{Ecuación 7}$$

De forma paralela, DaSPID activa un controlador PID que ajusta los niveles máximos de ancho de banda para cada *SLA* en función del retardo medio obtenido en los ciclos anteriores para el tráfico de distintas prioridades. En la red EPON se han definido tres niveles de servicio o *SLAs* (*SLA*$_0$, *SLA*$_1$ y *SLA*$_2$ siendo los dos primeros los más prioritarios) contratados por las *ONUs*, y tráfico de tres prioridades (P_0, P_1 y P_2), constituyendo P_0 y P_1 el tráfico prioritario y P_2 el tráfico de tipo *Best Effort*. De este modo, la Tabla 1 muestra los límites de retardo que deben cumplir los paquetes según su prioridad y el *SLA* de la *ONU* emisora, según recomendaciones de los estándares o estimaciones realizadas, tanto extremo a extremo como en la parte óptica de la red de acceso. En el algoritmo implementado, se toman como retardos máximos permitidos los que vienen definidos para el segmento del acceso en la Tabla 1. Por tanto, DaSPID controla y garantiza que el retardo de las clases de servicio P_0 y P_1, siendo éstas las clases

prioritarias, no superan estos umbrales según el *SLA* contratado. Sin embargo, se observa que no se garantiza un retardo máximo para la clase P_2, considerada *Best-Effort*, ya que esta clase de servicio se considera no prioritaria.

Segmento de red	**Recomendación**	**Clase de Servicio**	**Requisitos de retardo**	**Aplicaciones**
Extremo a extremo	ITU-T G.1010	P_0 (interactivo)	150 ms	VoIP, videoconferencia, juegos interactivos, *Telnet*
Acceso	ITU-T G.114	P_0 (interactivo)	1.5 ms	
Extremo a extremo	ITU-T G.1010	P_1 (respuesta rápida)	2 s	Mensajería de voz, navegador Web, HTML, transacciones, Correo electrónico
Acceso	Estimación	P_1 (respuesta rápida)	SLA_2 60 ms SLA_1 20 ms SLA_0 5 ms	
Extremo a extremo	ITU-T G.1010	P_2 (no crítico)	-	Datos sin prioridad
Acceso	Estimación	P_2 (no crítico)		

Tabla 1. Parámetros de red considerados en el entorno de simulación de DaSPID.

La manera en la que el PID diseñado ajusta los valores máximos de ancho de banda para cada *SLA* se explica a continuación. Cada cierto tiempo, denominado *tiempo_PID*, el submódulo *olt_mac* se envía un auto mensaje que le insta a ejecutar el método *PID_control_delay(cMessage *msg)*. Entonces, para los *SLAs* en los que se ha activado el controlador PID, (es decir, aquéllos en los que en el último ciclo de transmisión se ha superado el retardo máximo en sus servicios prioritarios), calcula el sumatorio de los errores individuales cometidos a la hora de garantizar el retardo de las clases de servicio prioritarias (diferencia entre el retardo máximo garantizado y el retardo medio real obtenido para cada SLA_k y servicio P_j, $R_{P_j}^{sla_k} - \overline{r_{P_j}^{sla_k}[n]}$) y calcula la señal de control del PID, usando la Ecuación 8 y la Ecuación 10, respectivamente. Cabe destacar que para el cálculo del retardo medio de las clases de servicio por cada *SLA*, se mantiene un esquema de ventana deslizante de T_{window} segundos por cada *SLA* y servicio prioritario (P_0, P_1). Dado que las ventanas deslizantes se actualizan constantemente con el retardo medio de los paquetes pertenecientes a cada *SLA* y clase de servicio en el último ciclo de transmisión, éstas sólo contienen las muestras más recientes, lo cual lleva a un mejor control en tiempo real del retardo medio de los paquetes y una mejor gestión de los cambios en la red o en el tráfico.

$$e[n] = \sum_j (R_{P_j}^{sla_k} - \overline{r_{P_j}^{sla_k}[n]}) \qquad \text{Ecuación 8}$$

$$u[n] = K_p \cdot e[n] + K_p \cdot \frac{T_{sample}}{T_i} \cdot \sum_{m=0}^{n} e[m] + K_P \cdot \frac{T_d}{T_{sample}} \cdot (e[n] - e[n-1]) \qquad \text{Ecuación 9}$$

$$u[n] = K_p \cdot e[n] \qquad \text{Ecuación 10}$$

Aunque de forma general la señal de control en un PID se calcula considerando los términos P, I y D (Ecuación 9), en esta aplicación del control del retardo se utilizó un controlador simple P (Ecuación 10), ya que ofrecía un comportamiento más eficiente que las políticas PI y PID. Es importante resaltar que la señal de control calculada está definida en tiempo, por lo que será necesario aplicarle un factor de conversión para expresarla en bits y emplearla entonces para ajustar el nivel de ancho de banda. Una vez obtenida la señal de control en bits, se actualiza el ancho de banda máximo permitido por el PID para cada *ONU*. Para ello, el controlador P diseñado resta al anterior ancho de banda máximo por *SLA* el valor de la señal de control actualizado, teniendo en cuenta que si este nuevo ancho de banda resultase negativo, se le adjudicaría un valor mínimo por defecto. Por otro lado, el ancho de banda calculado por el controlador en cada ciclo debe estar delimitado para evitar que la suma del ancho de banda máximo permitido para todas las *ONUs* exceda al ancho de banda contenido en un tiempo de ciclo máximo de 2 ms, que es el máximo definido por el estándar EPON. Para ello, el sistema P implementado incluye un delimitador que escala los anchos de banda máximos calculados por el controlador P en el caso de que el ancho de banda total máximo permitido supere el ancho de banda total de un ciclo. Todo el proceso del sistema P para realizar el control del retardo se visualiza en el diagrama de bloques de la Figura 17.

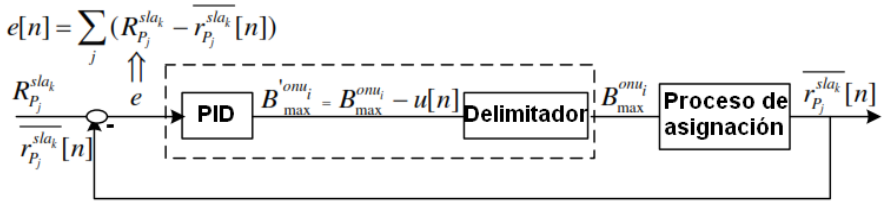


Figura 17. Diagrama de bloques del proceso controlado por un PID para el control del retardo.

En los siguientes apartados de la memoria se expondrán y analizarán los resultados obtenidos al simular el algoritmo DaSPID en el simulador de redes EPON

desarrollado en OMNeT++, evaluando distintos aspectos del mismo cuando se introducen fuentes de tráfico auto-semejante.

6.3 Escenario de simulación genérico para DaSPID

Para ejecutar el algoritmo de asignación dinámica de ancho de banda basado en *polling* con PID para el control del retardo, DaSPID, se ha planteado un escenario de simulación en el que los parámetros más significativos toman los valores que a continuación se definen:

- El método DBA implementado en la capa MAC del *OLT* es DaSPID (*oltmethod_Centralized0_Polling1_wdm2_PollingPID3_DaSPID4=4*).
- La red EPON consta de 16 *ONU*s (*numOnu=16*), cada una de ellas con su correspondiente índice, un número entero entre 0 y *numOnu*-1.
- Se definen 3 niveles de servicio o *SLAs* (*numSLA=3*), todos con peso unitario (*w_sla0=w_sla1=w_sla2=1*).
- Cada *ONU* consta de 3 fuentes de tráfico óptico (*numqueue=3*), cada una perteneciente a una clase de servicio distinta (P_0, P_1 y P_2). La fuente P_0 genera paquetes de 70 Bytes a una tasa constante de 4.48 Mbps, mientras que las fuentes P_1 y P_2 generan tráfico *Self-Similar* usando 32 *streams* (*numstreamV2_32_128_256=32*), con paquetes de tamaño de 64, 594 y 1500 bytes (*longpacketfixed0_trimodal1=1*).
- La carga de la fuente P_0 (tráfico CBR) es de 0,0448 y la de las fuentes P_1 y P_2 (tráfico self-similar) será equiprobable e igual a la mitad del resto de la carga.
- Las 16 *ONU*s se distribuyen entre los *SLAs* de la siguiente manera: 2 en el SLA_0 (*numonu_sla0=2*), 6 en el SLA_1 (*numonu_sla1=6*) y 8 en el SLA_2 (*numonu_sla2=8*).
- La tasa de transmisión de datos en la subred EPON es de 1 Gbps (*txrate=1*1000000000*).
- La duración de cada ciclo de transmisión de paquetes Ethernet por parte de las 16 *ONUs* hacia el *OLT* es de 2 ms (*T_cycle=0.002*).

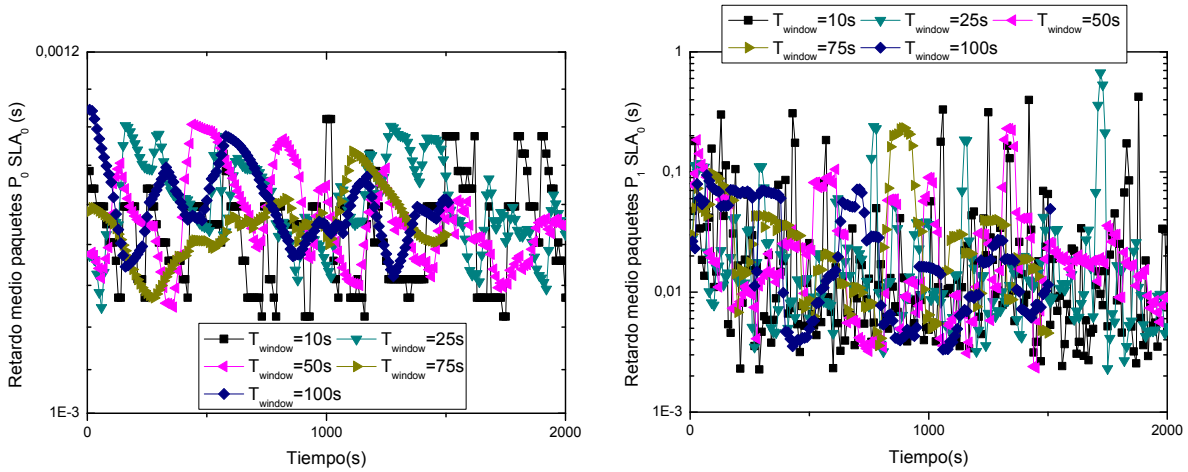
- La longitud de la ventana deslizante será modificada a lo largo del análisis (T_{window}).

6.4 Análisis de resultados de DaSPID bajo patrones de tráfico auto-semejante

En esta subsección se procede a estudiar los resultados obtenidos al simular el algoritmo DaSPID en OMNeT++ mediante el uso de ventanas de diferentes tamaños bajo un patrón de tráfico auto-semejante o *self-similar*.

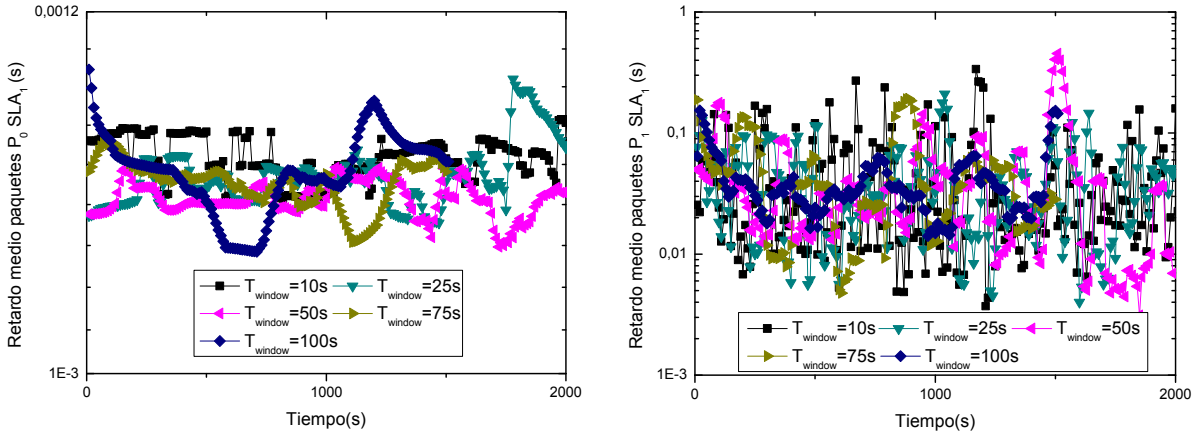
Los tamaños de ventana (T_{window}) estudiados serán de 10, 25, 50, 75 y 100 segundos. Esta ventana es utilizada por DaSPID para promediar y actualizar el retardo de cada una de las clases de servicio prioritario (P_0, P_1,), con la finalidad de minimizar el error cometido por el controlador P a la hora de controlar el retardo de dicho tráfico y mantenerlo por debajo de las cotas máximas estipuladas por el proveedor de servicios. Dicho de otra manera, DaSPID emplea un controlador P para ajustar periódicamente el ancho de banda máximo asignado a cada *SLA* con el fin de mantener el retardo de las clases de servicio prioritarias, P_0 y P_1, por debajo de unos límites que garanticen al usuario la *QoS* requerida para cada aplicación. De este modo, si la ventana es demasiado pequeña, el número de muestras que contiene no será suficiente, por lo que el valor de retardo medio no será muy fiable. Por el contrario, si el valor de la ventana es demasiado grande, cambios bruscos en el patrón de tráfico no se actualizarán a tiempo y por lo tanto el valor de retardo no se actualizará adecuadamente en tiempo real. Así pues, el tiempo de la ventana deslizante es un parámetro muy importante y por lo tanto resulta muy interesante hacer un análisis de su comportamiento modificando dicho valor. Dicho análisis tendrá un mayor impacto cuando se utilicen fuentes de tráfico rafagoso, como en el escenario de red que nos atañe.

Las Figuras 18, 19 y 20 muestran la evolución del retardo medio de ventana del servicio P_0 y del servicio P_1 (siendo estos los servicios prioritarios) para cada uno de los tres *SLAs* considerados en la red EPON, considerando las cotas máximas de la Tabla 1.



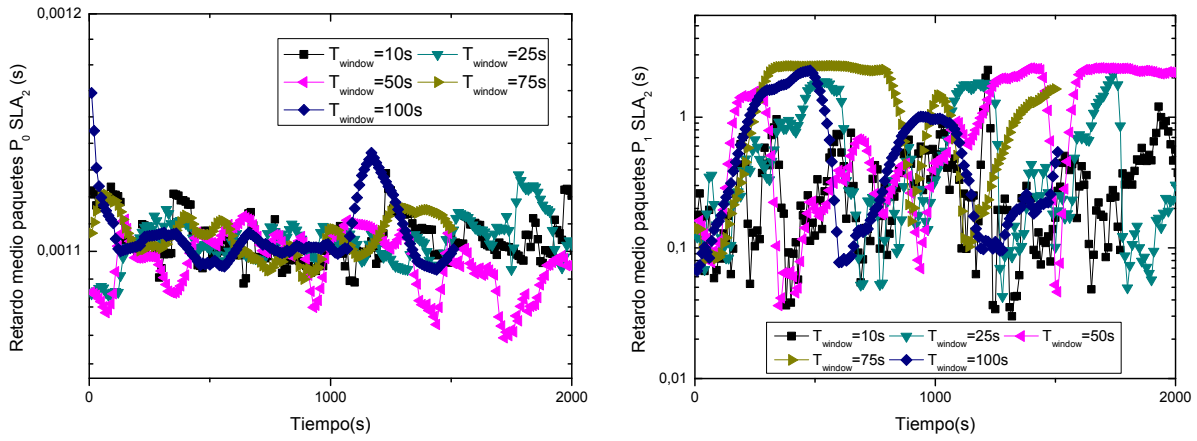
(a) (b)

Figura 18. Comparativa de la evolución del retardo medio de los paquetes para distintos tamaños de ventana para *SLA₀* con (a) la clase P_0 y (b) la clase P_1.



(a) (b)

Figura 19. Comparativa de la evolución del retardo medio de los paquetes para distintos tamaños de ventana para *SLA₁* con (a) la clase P_0 y (b) la clase P_1.



(a) (b)

Figura 20. Comparativa de la evolución del retardo medio de los paquetes para distintos tamaños de ventana para *SLA*$_2$ con (a) la clase P$_0$ y (b) la clase P$_1$.

La selección de estos tamaños de ventana tiene por objetivo observar el impacto del tamaño de la ventana en la evolución en tiempo real del retardo medio para cada una de las clases de servicio, para ver así qué valor es más adecuado para obtener un nivel medio de retardo más fiable y estable. A tenor de los resultados obtenidos podemos afirmar que tamaños elevados de ventana provocan una peor adaptación del algoritmo a las condiciones de nuestra red y, por otro lado, tamaños pequeños de ventana provocan alta variabilidad y fluctuación del retardo medio, pudiendo provocar el descarte de demasiados paquetes.

Por otro lado, viendo los resultados mostrados en las gráficas podemos concluir que DaSPID consigue controlar eficientemente el retardo de la clase de servicio más prioritaria P$_0$, correspondiente a tráfico de tasa binaria constante, manteniéndolo siempre por debajo de 1.2 ms para los tres *SLAs*, por lo tanto, por debajo de su nivel límite de 1.5 milisegundos para el tráfico interactivo fijado anteriormente en la Tabla 1.

Respecto a la clase de servicio P$_1$ es reseñable destacar que para todos los perfiles obtenidos, considerando todos los tamaños de ventana analizados, se aprecia que DaSPID no es capaz de mantener el retardo por debajo del límite estipulado para dicha clase de servicio esto es, para el *SLA*$_0$ 5 ms, para *SLA*$_1$ 20 ms, y para *SLA*$_2$ 60 ms. Este comportamiento será objeto de análisis a continuación.

La explicación de lo ocurrido se encuentra en la gran variabilidad que sufre el retardo instantáneo del tráfico de prioridad P_1, debido a su origen altamente rafagoso, pues dicho tráfico es generado a partir de fuentes autosemejantes o *self-similar*. Por lo tanto, en este punto se decidió realizar el resto de simulaciones considerando un tiempo de ventana intermedio de 50 ms (T_{window}=50) pues hemos visto que su comportamiento es más estable con el tiempo que en el caso de las otras ventanas más pequeñas, y además es lo suficientemente pequeño para responder a cambios en las condiciones de red.

Así pues, se observa una alta fluctuación de los niveles medios de retardo para todos los tamaños de ventana considerados y para todos los perfiles de abonado en esa clase de servicio, debido fundamentalmente al carácter rafagoso de las fuentes. De este modo y con el fin de asegurarnos que se garantiza en todo momento que el retardo medio para cada una de las clases de servicio esté siempre por debajo de los niveles máximos permitidos, se propuso el diseño dentro de las *ONUs* de un sistema de control de admisión de paquetes de las clases prioritarias (P_0 y P_1), que limite la salida de paquetes hacia el *OLT* cuyo retardo estimado supere unos valores máximos preestablecidos en las *ONUs*. Este control de admisión, que se describirá en el siguiente capítulo, solamente se activará para el tráfico de prioridad media P_1, puesto que dicho tráfico, como ya hemos mencionado antes, ha sido implementado mediante concatenación de fuentes *Self-Similar* y es el que más variabilidad sufre. Así pues, como P_0 cumple con los requisitos estipulados, el sistema de admisión on se activará, por lo que no se realizará el estudio de dicho servicio.

6.5 Conclusiones

En este capítulo se ha implementado un algoritmo de asignación dinámica de ancho de banda para redes EPON en el simulador *OMNeT++*, previamente desarrollado por el GCO. Este algoritmo, denominado DaSPID, se basa en un controlador PID para gestionar el retardo de diferentes clases de servicios y perfiles de abonado. El fin de este algoritmo es controlar el retardo máximo de los paquetes en el canal ascendente o *upstream*, es decir, aquel que va de las *ONUs* al *OLT*. Hay que destacar también que lleva a cabo una diferenciación entre usuarios y clases de servicio y distingue los requisitos de *QoS* que demanda el tráfico según su prioridad (P_0 y P_1) y el *SLA* al que pertenezca el usuario (SLA_0, SLA_1 y SLA_2).

En este Proyecto Fin de Carrera se ha analizado el impacto en dicho algoritmo cuando se emplean patrones de tráfico auto-semejante en las clases de servicio prioritarias P_0 y P_1. De este modo, se ha visto que DaSPID consigue controlar eficientemente el retardo de la clase de servicio más prioritaria P_0, correspondiente a tráfico de tasa binaria constante, cosa que no ocurre para el tráfico de prioridad P_1. La explicación a este hecho reside en la naturaleza rafagosa del tráfico P_1 ya que este tráfico se genera a partir de fuentes auto-semejantes o *self-similar*. A tenor de los resultados obtenidos, se concluye que la mejor opción es la elección de tamaños de ventana intermedios (50 segundos) para conseguir un comportamiento estable en dicho parámetro y al mismo tiempo capaz de responder de forma rápida ante cambios en las características del tráfico.

Ahora bien, puesto que el objetivo perseguido es garantizar en todo momento el cumplimiento del retardo máximo permitido para cada una de las clases de servicio, se expone la necesidad de diseñar un sistema de control de admisión de paquetes en las *ONUs* para los paquetes de las clases prioritarias P_0 y P_1, que se activará automáticamente para la segunda de ellas puesto que es la que mayor inestabilidad presenta, puesto que incumple los requisitos de *QoS* perseguidos.

7

Diseño e implementación de un controlador de admisión de paquetes en DaSPID

7.1 Introducción

En el capítulo anterior ha quedado patente la necesidad de añadir al algoritmo DaSPID un control de admisión de paquetes en las *ONUs*, con el fin de garantizar que el retardo medio de servicios prioritarios no superen los umbrales definidos en la Tabla 1 ante patrones de tráfico *self-similar*, y así proporcionar a los usuarios de cada *SLA* la *QoS* (*Quality of Server*) correspondiente.

En los próximos apartados de este capítulo se procederá al diseño e implementación de dos sistemas de control de admisión de paquetes, en concreto, un CAD de tipo dinámico y otro de tipo estático. Para ambos casos se procederá a una presentación y explicación del CAD implementado y posteriormente, tendremos un apartado en el que analizaremos los resultados de ambos. Para ello, en primer lugar, extraeremos conclusiones del análisis del CAD estático y en segundo lugar, haremos lo mismo para el CAD dinámico pero, en este caso, analizando otros factores extra, tales como el nivel de variación del CAD o los niveles de acotación máxima de su valor (parámetros que se explicarán más adelante).

Finalmente se aportarán conclusiones que engloben todos los estudios llevados a cabo, dando una visión global y final de los resultados obtenidos.

7.2 Diseño de sistemas de control de admisión de paquetes

En esta sección se procede al diseño y explicación teórica de dos sistemas de control de admisión de paquetes, siendo éstos un CAD estático en primer lugar y un CAD dinámico en segunda posición. Más tarde, en los siguientes apartados, se analizarán los resultados obtenidos con cada uno de ellos.

7.2.1 Diseño de un CAD estático

Puesto que nuestro objetivo es garantizar que el retardo medio del servicio P_1 no supere los umbrales definidos en la Tabla 1 ante patrones de tráfico rafagoso para el algoritmo DaSPID, se propone complementar el controlador PID diseñado en la capa MAC del *OLT* con un sistema de control de admisión de paquetes en la capa MAC de las *ONUs*. Este sistema será capaz de estimar el retardo que tendrá cada paquete de prioridad P_1 para descartarlo o insertarlo en la cola correspondiente según supere o no un límite de retardo máximo aceptado. Este control de admisión sensible al retardo o CAD (Control de Admisión *Delay-aware*) se aplica únicamente sobre el tráfico P_1 porque el controlador PID implementado garantiza el retardo máximo para el tráfico de P_0, por no tener una naturaleza rafagosa sino constante (tráfico CBR); pero en cambio no logra mantener el retardo de P_1 por debajo de los límites máximos. El funcionamiento de este controlador dentro de las *ONUs* es el siguiente. Cada vez que una *ONU* recibe un paquete perteneciente a P_1, realiza un cálculo del retardo medio que, en estimación, sufrirá este nuevo paquete antes de llegar al *OLT*, aplicando la Ecuación 11. Para dicho cálculo, la capa MAC de la *ONU* estima el tiempo que transcurrirá desde el instante actual hasta el momento en que se transmita el último bit presente en ambas colas de prioridad P_0 y P_1, pues al emplear un esquema de prioridad de colas estricta todos éstos se transmitirán antes que el nuevo paquete entrante. Cabe destacar que para dicha estimación se tiene en cuenta el ancho de banda asignado en el ciclo actual ($B_{alloc\_previous}$), pues no se conoce la asignación en ciclos futuros, al igual que el tiempo de ciclo actual ($T_{cycle\_previous}$).

$$delay_{estimated} = T_{cycle\_previous} * \left(\frac{bits\_queue[P_0] + bits\_queue[P_1]}{B_{alloc\_previous}} \right) \qquad \text{Ecuación 11}$$

Si este retardo estimado ($delay_{estimated}$) supera el máximo admitido para dicha clase de servicio y *SLA* ($CAD\_delay_{max}[sla]$), el paquete será descartado ($if\ delay_{estimated} \geq CAD\_delay_{max}\ [sla] \rightarrow delete\ packet$). En caso contrario, el paquete será insertado en la cola P_1 ($if\ delay_{estimated} < CAD\_delay_{max}\ [sla] \rightarrow insert\ packet$). Cabe destacar que el citado retardo máximo para que una *ONU* admita un nuevo paquete P_1 no es igual al retardo máximo estipulado en el *OLT* para garantizar la *QoS* requerida para dicho servicio, pues estamos trabajando con estimaciones y predicciones, y por lo tanto será necesario introducir un margen de error por encima. En concreto, este retardo del control de admisión lo hemos considerado menos restrictivo y fijo a lo largo de la simulación. De este modo, se ha decidido considerar valores del retardo máximo permitido por el CAD proporcionales al máximo retardo aceptable para garantizar una mínima *QoS* a cada *SLA*. En concreto, se ha elegido un factor de proporcionalidad "3" de modo que los valores del CAD para cada *SLA* aparecen recogidos en la Tabla 2. Cabe destacar que en otro proyecto anterior se ha hecho un estudio más exhaustivo del valor de proporcionalidad óptimo a escoger.

	Retardo máximo de admisión
SLA_0	3x5 ms = 15 ms
SLA_1	3x20 ms = 60 ms
SLA_2	3x60 ms = 180 ms

Tabla 2. Retardos máximos de admisión para los paquetes de clase P_1 y para cada *SLA* en los tres esquemas simulados en OMNeT++ para el CAD de DaSPID.

Es importante indicar que si el valor límite del CAD es muy alto, es decir poco restrictivo, los retardos medios reales podrían superan fácilmente el límite establecido para esa clase de servicio. Por el contrario, si este límite es demasiado bajo, se pueden llegar a borrar muchos paquetes, ya que el nivel será demasiado restrictivo y se pueden llegar a borrar paquetes de forma innecesaria. Es por este motivo por lo que los valores recogidos en la Tabla 2 podrían ser una buena aproximación a nuestro problema.

7.2.2 Diseño de un CAD dinámico

Tal y como ya hemos visto en el apartado 7.2.1, la incorporación de un control de admisión al algoritmo DaSPID aporta grandes ventajas en relación a las prestaciones del algoritmo, pero bien es cierto que, como ya mencionábamos, estas ventajas son mejorables. Esto es debido a que el hecho de tener umbrales fijos no permite una alta adaptación a las necesidades de la red EPON, puesto que ésta no presenta las mismas

características en la red o en el tráfico en todo momento, de modo que habrá situaciones que no permitirán sacarle un óptimo partido a los recursos de la red, desaprovechando muchos de los mismos. Esto traerá consigo problemas tales como anchos de banda desaprovechados, mayores retardos, o probabilidades de bloqueo superiores. De este modo, si los umbrales del CAD son fijos, cuando las condiciones del tráfico se vean modificadas, el CAD no se podrá adaptar de forma automática a las nuevas circunstancias de red, lo cual hará que el sistema pierda potencialidad y flexibilidad.

Ahora bien, este mecanismo puede ser optimizado de forma que los umbrales máximos de admisión del CAD varíen en tiempo real y de forma automática según las necesidades de la red EPON. Así pues se procedió al diseño de un CAD dinámico, teniendo en cuenta los retardos existentes en la red. Esta mayor flexibilidad permite una mejor adaptación de los recursos a los usuarios de los mismos, de modo que se espera conseguir mejores resultados en parámetros tales como el retardo medio, el ancho de banda que cada *ONU* utiliza, los cuales están relacionados entre sí. Para ello se programó la funcionalidad mencionada en el simulador *OMNeT++*.

A continuación se procede a presentar el funcionamiento e implementación del CAD dinámico diseñado. La idea que se persigue es adaptar el valor máximo del CAD, para la clase de servicio P_1, aumentándole o disminuyéndole una cantidad α de su valor de forma periódica y observando el comportamiento en tiempo real de la carga de la red. Para ello nos fijaremos en los siguientes factores:

- El retardo medio estimado que tendrán los paquetes de las *ONUs* para cada uno de los *SLAs* existentes (*retardo_max_ONU[número de la ONU][prioridad]*) para dicho servicio P_1.

- El retardo medio de la ventana para cada uno de los *SLAs* existentes (*retardo_alg_SLA[SLA][prioridad]*) para dicho servicio P_1.

El valor de la prioridad mencionado en ambos parámetros es “1” en este caso debido a que el estudio se realiza para P_1.

De este modo, en primer lugar, se recoge el valor del retardo medio estimado en una de las *ONUs* (ya que este valor es el mismo para todas las *ONUs* de un mismo *SLA*). Esto se puede ver en la primera parte de la Figura 21 en la que se muestra el código fuente empleado.

```
for(int m=0;m<(int)par("numSLA");m++)
{

        if(m==0)
        {
            delay_max_permitido=olt_mac->retardo_max_ONU[0][1];
            delay_ventana=olt_mac->retardo_alg_SLA[0][1];
        }

        if(m==1)
        {
            delay_max_permitido=olt_mac->retardo_max_ONU[2][1]
            delay_ventana=olt_mac->retardo_alg_SLA[1][1];
        }

        if(m==2)
        {
            delay_max_permitido=olt_mac->retardo_max_ONU[8][1];
            delay_ventana=olt_mac->retardo_alg_SLA[2][1];

        }

        if(delay_ventana>delay_max_permitido)
        {
                max_control_admision[m]=max_control_admision[m]-(α*max_control_admision[m]);
       }

      if(delay_ventana<delay_max_permitido)
     {
                max_control_admision[m]=max_control_admision[m]+( α*max_control_admision[m]);
     }

}
```

Figura 21. Cálculo y actualización del umbral máximo del CAD.

Cabe destacar que *numSLA* es el número de *SLAs* existentes en la red EPON y también que para el SLA_0 se está seleccionando la *ONU* 0 (*retardo_max_ONU**[0]**[1]*) debido a que el SLA_0 está constituido por las *ONUs* 0 y 1. Por otro lado para el SLA_1, se elige la ONU_2 (*retardo_max_ONU**[2]**[1]*) puesto que este SLA_1 está formada por las *ONUs* de 2 a 7 y finalmente para SLA_2 tomamos la *ONU* 8 (*retardo_max_ONU**[8]**[1]*) ya que el SLA_2 engloba las *ONUs* de 8 a 15.

A continuación, tal y como vemos en la Figura 21, para cada perfil de abonado y la clase de servicio P_1, se realiza una comparación entre el retardo medio de la ventana y el retardo medio estimado calculado, que pasa a llamarse retardo máximo permitido. En caso de que el retardo medio real de la ventana sea mayor que el retardo máximo establecido por el CAD, entonces el valor del CAD tiene que disminuir un cierto valor α para adaptarse a los requerimientos de *QoS* ya que estamos sobrepasando los límites de retardo medio. Ahora bien, si el retardo de la ventana tiene un valor menor que el retardo máximo permitido, el valor del CAD se ve incrementado la cantidad α para aprovechar los recursos de la red. Todo este procedimiento se ejecuta de manera periódica cada cierto tiempo, aconsejablemente relacionado con el tamaño de la ventana utilizado.

Por otro lado, conviene destacar que para evitar grandes oscilaciones del CAD (que se verán y se analizarán más adelante en el apartado de análisis de resultados) se ha tomado la decisión de acotar su valor, dando para ello una cota inferior y otra superior al mismo. Recordemos que esto es un aspecto a tener en cuenta puesto que oscilaciones del CAD se verán reflejadas en fluctuaciones de ciertos parámetros tales como el retardo medio, de modo que a mayor oscilación, peores resultados obtendremos puesto que las variaciones elevadas impiden una buena adaptación a las condiciones de la red.

De este modo una vez actualizados los valores del CAD como en la Figura 21, se procede a comparar este valor del CAD con las cotas mínimas y máximas establecidas, que serán:

- La cota mínima, la cual viene determinada por el retardo máximo (*delay_max_permitido*) para la clase de servicio P_1 para cada perfil de abonado.
- La cota máxima que viene dada por diez veces el retardo máximo (*10*delay_max_permitido*).

Distinguimos por tanto dos situaciones.

1) En el caso en que el valor del CAD tenga que ser reducido porque el retardo de la ventana es superior al máximo permitido, se compara con la cota inferior para que esta no sea rebasada, de manera que pueden ocurrir dos cosas:
    a. Si el valor del CAD es menor que el retardo máximo permitido, entonces su valor pasa a ser el de la cota inferior, siendo éste como ya decíamos el retardo máximo permitido, evitando así estar por debajo de la cota inferior.

b. Si el valor del CAD es mayor que el retardo máximo permitido, su valor se queda tal cual, permaneciendo el valor entre la cota máxima y la mínima.

2) En el caso de que el retardo de la ventana sea inferior al máximo permitido, se compara con la cota superior, distinguiéndose dos opciones.

a. Si el valor del CAD es mayor el retardo máximo permitido (fijado a diez veces), su valor pasa a ser el de la cota superior, evitando así que ésta sea sobrepasada.
b. En caso contrario, permanece igual, estando comprendido su valor entre ambas cotas.

Finalmente decir que, con el fin de poder ver el funcionamiento del algoritmo *DaSPID* mediante la implementación de un CAD dinámico, tanto en el caso en que su valor no se encuentre acotado, como en el que sí, todo ello ante un patrón de tráfico *self-similar*, se ha llevado a cabo un estudio que se expondrá en el apartado de análisis de resultados del CAD dinámico.

7.3 Escenario de simulación genérico para DaSPID

Para ejecutar el algoritmo de asignación dinámica de ancho de banda basado en *polling* con PID para el control del retardo, DaSPID, mediante la implementación de un CAD, tanto estático como dinámico, se ha planteado un escenario de simulación en el que los parámetros más significativos toman los valores que a continuación se definen:

- El método DBA implementado en la capa MAC del *OLT* es DaSPID (*oltmethod_Centralized0_Polling1_wdm2_PollingPID3_DaSPID4=4*).
- La subred óptica consta de 16 *ONU*s (*numOnu=16*), cada una de ellas con su correspondiente índice, un número entero entre 0 y *numOnu*-1.
- Se definen 3 niveles de servicio o *SLAs* (*numSLA=3*), todos con peso unitario (*w_sla0=w_sla1=w_sla2=1*).
- Cada *ONU* consta de 3 fuentes de tráfico óptico (*numqueue=3*), cada una con perteneciente a una clase de servicio distinta (P_0, P_1 y P_2). La fuente P_0 genera paquetes de 70 Bytes a una tasa constante de 4.48 Mbps, mientras que las fuentes

P_1 y P_2 generan tráfico *Self-Similar* usando 32 *streams* (*numstreamV2_32_128_256=32*), con paquetes de tamaño de 64, 594 y 1500 bytes (*longpacketfixed0_trimodal1=1*).

- La carga de las fuentes *Self-Similar* de cada *ONU* se ha fijado en 0.4, para tener una carga total en cada *ONU* de 0.85.
- Las 16 *ONU*s se distribuyen entre los *SLAs* de la siguiente manera: 2 en el SLA_0 (*numonu_sla0=2*), 6 en el SLA_1 (*numonu_sla1=6*) y 8 en el SLA_2 (*numonu_sla2=8*).
- La tasa de transmisión de datos en la subred EPON es de 1 Gbps (*txrate=1*1000000000*).
- La duración de cada ciclo de transmisión de paquetes Ethernet por parte de las 16 *ONUs* hacia el *OLT* es de 2 ms (*T_cycle=0.002*).
- La longitud de la ventana deslizante (T_{window}) tomará distintos valores.

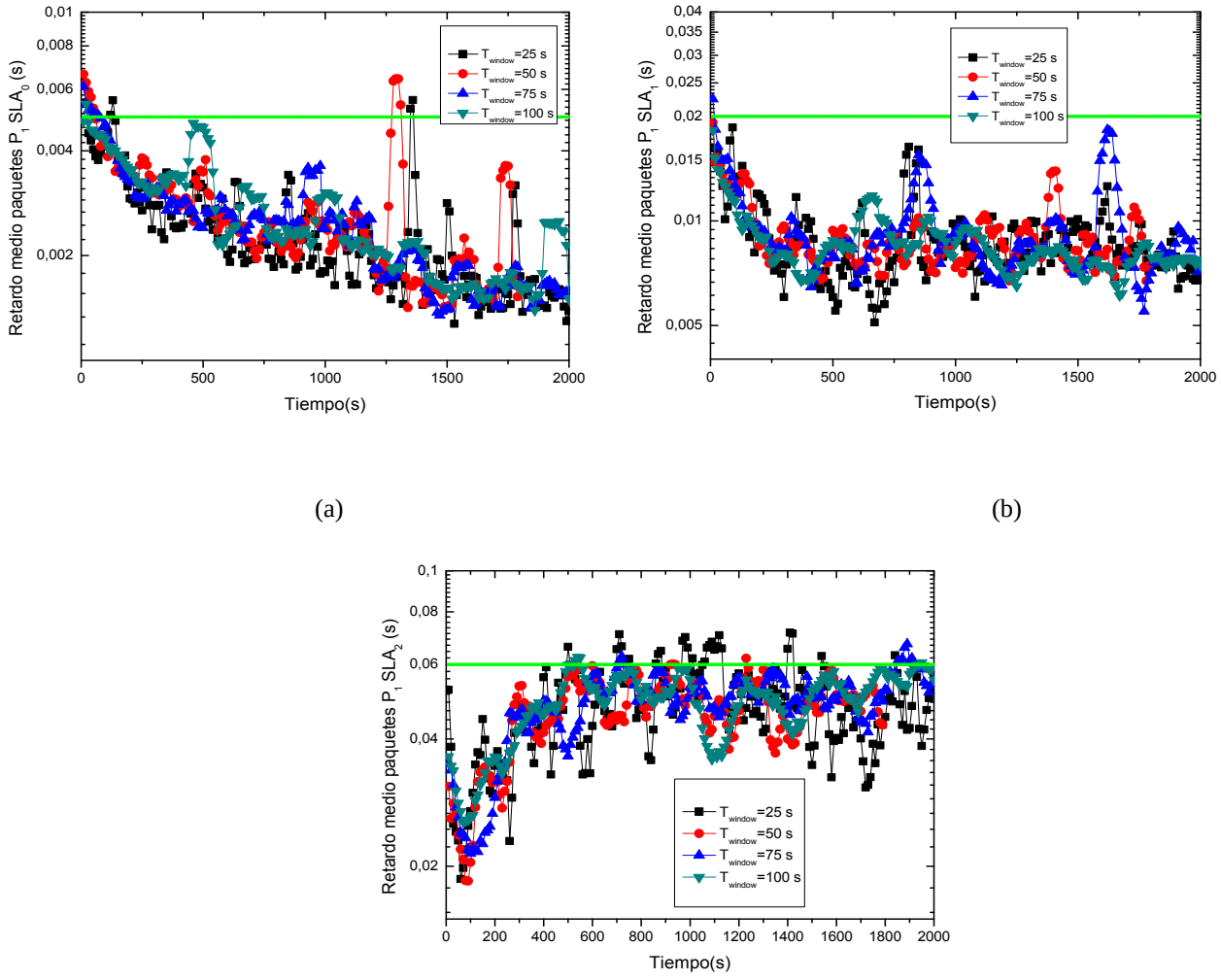
7.4 Análisis de resultados

En esta subsección se procede a analizar el comportamiento del algoritmo DaSPID, al cual se le ha implementado un control de acceso al medio, bajo un patrón de tráfico *self-similar*.

En primer lugar se estudia su comportamiento con la implementación de un CAD de tipo estático y seguidamente se observa el funcionamiento del algoritmo bajo la implementación de un CAD dinámico, para observar y comparar las diferencias.

7.4.1 Análisis de resultados del CAD estático bajo un patrón de tráfico auto-semejante

Una vez elegido el nivel del CAD fijo, el cual aparece reflejado en la Tabla 2, se procede a realizar un estudio del comportamiento del algoritmo DaSPID en función del tamaño de ventana seleccionado para así analizar su impacto en el retardo medio ante un patrón de tráfico *self-similar*. Por consiguiente, en la Figura 22 (a), (b) y (c), se representa la evolución del retardo medio de P_1, para el SLA_0, SLA_1 y SLA_2, respectivamente, al considerar distintos tamaños de ventana y una carga de *ONU* de 0.85 (*ONUs* transmitiendo a 85 Mbit/s).



(a) (b)

(c)

Figura 22. Evolución de la probabilidad de bloqueo de la clase P_1 con carga de 0.85 en OMNeT++ para (a) *SLA$_0$* (b) *SLA$_1$* (c) *SLA$_2$*.

Con el fin de comparar el impacto del tamaño de la ventana en el algoritmo DaSPID, una vez implementado el CAD y bajo patrones de tráfico *self-similar*, se ha procedido a escoger una serie de tamaños de ventana comprendidos entre los 25 y 100 segundos ya que, tal y como veíamos previamente, valores por debajo de 25 segundos provocan altas variaciones en el retardo, lo que conlleva a la eliminación de muchos paquetes. Sin embargo, cifras superiores a los 100 segundos pueden llevar una lenta adaptación a las condiciones reales de la red.

A tenor de los resultados obtenidos, se puede afirmar que al complementar DaSPID con el CAD estático, el retardo medio de los paquetes de prioridad P_1 se mantiene la mayor parte del tiempo por debajo de los umbrales máximos de *QoS*

establecidos en la Tabla 1, los cuales se ven reflejados mediante la recta de color verde. De forma adicional, también se demuestra que DaSPID ha conseguido garantizar un retardo máximo diferenciado de forma conjunta, escogiendo distintos tamaños de ventana, en clases de servicio y perfiles de abonado usuarios, pues el límite de retardo para la prioridad P_1 es diferente según el *SLA* al que pertenezca la *ONU* que los genera.

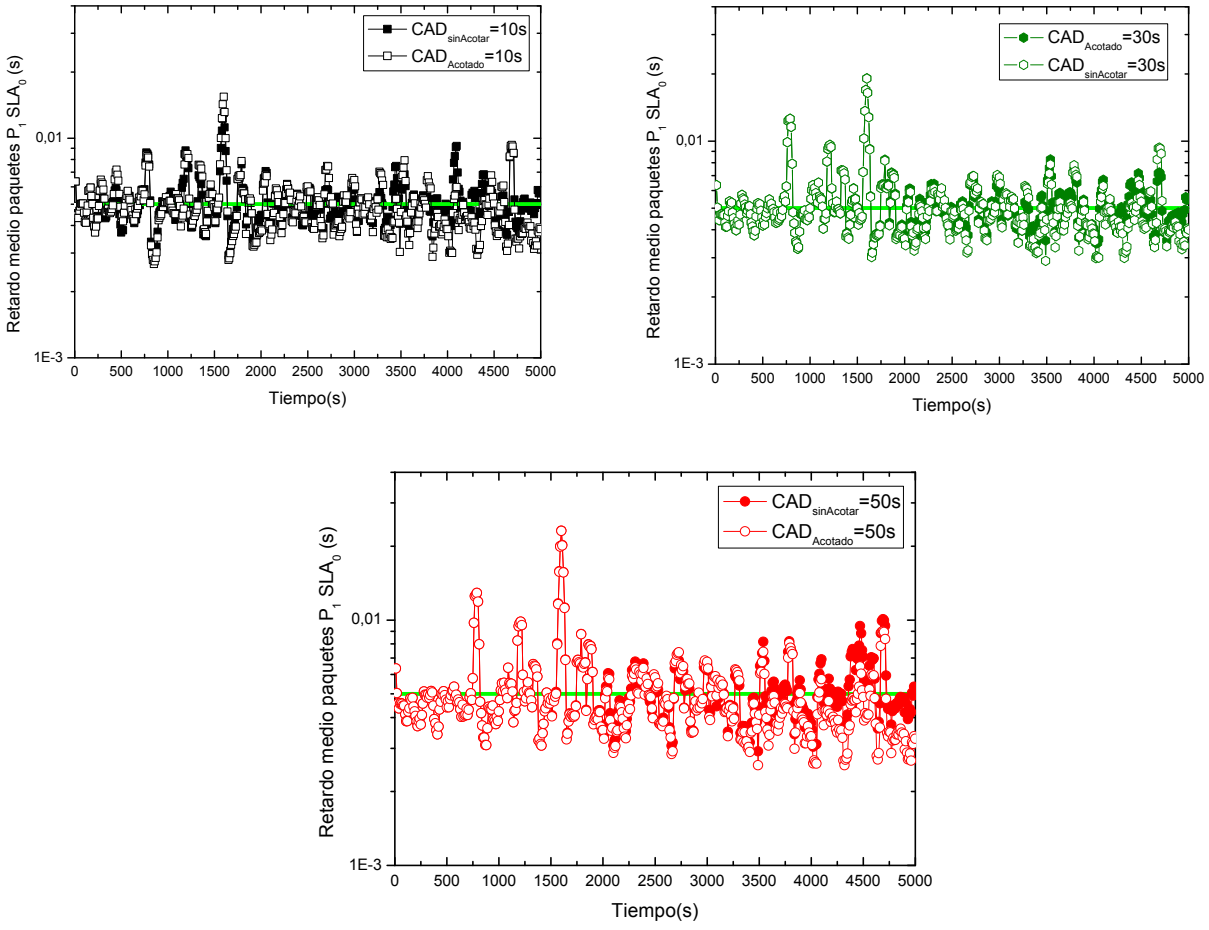
Como conclusión al estudio realizado, con el fin de cumplir los requerimientos de *QoS* analizados respecto al retardo medio, decidimos quedarnos con un tamaño de ventana de 50 segundos, de modo que no se eliminan demasiados paquetes (T_{window} pequeño) y la adaptación no resulta muy lenta (T_{window} alto) aunque es altamente mejorable como veremos a continuación.

7.4.2 Análisis de resultados del CAD dinámico bajo un patrón de tráfico auto-semejante

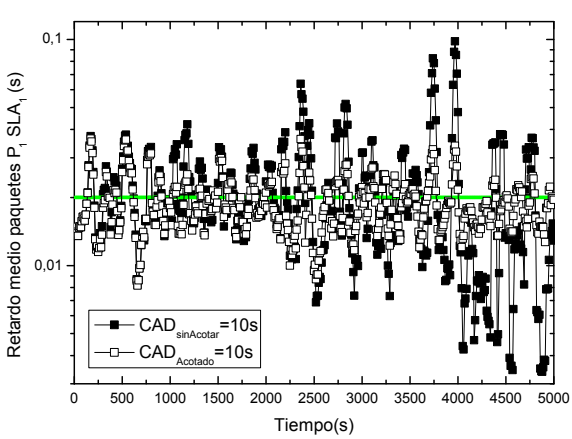
Llegados a este punto, se procede a la eliminación del CAD estático estudiado y se implementa el CAD dinámico diseñado. Antes de llevar a cabo cualquier simulación para el CAD dinámico, conviene preguntarse qué valor de α resulta más apropiado para la variación del CAD. De este modo, se ha llegado a la conclusión de que ha de ser un porcentaje pequeño, para poder tener una buena adaptación (que no sea ni muy lenta ni muy rápida) a las necesidades de la red EPON sin tener que borrar muchos paquetes. Así pues, tomamos por ahora el valor de α del 10%, no sin antes decir que más adelante se comprobará su eficiencia respecto a otros valores.

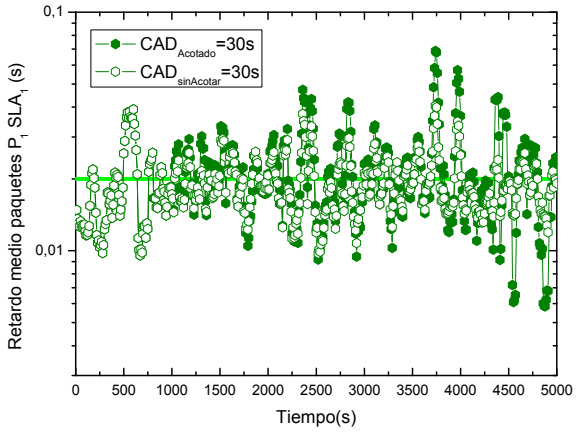
En primer lugar, se realiza un estudio comparativo del retardo medio experimentado mediante el algoritmo DaSPID, bajo patrones de tráfico *self-similar*, para dos situaciones, una en el que el valor del CAD no está acotado y la otra en la que sí lo está. Para ello, se modifica la semilla de números aleatorios de nuestro simulador de modo que se empiece siempre con los mismos valores, para poder así comparar las distintas políticas (estática y dinámica) con un patrón de números aleatorios idéntico, viendo por tanto la respuesta de cada una de ellas ante una misma situación de tráfico.

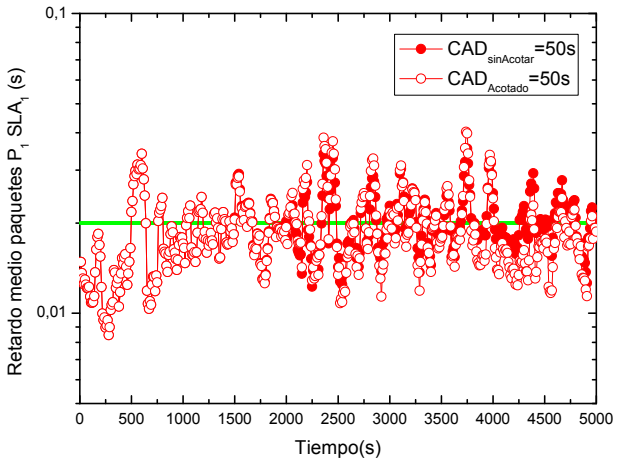
Así pues, en la siguiente Figura 23 (a), (b), y (c), se establece una comparación entre la respuesta del algoritmo DaSPID, complementado mediante un CAD dinámico, y este mismo algoritmo complementado con un CAD estático. Todo ello se realiza bajo un patrón de tráfico *self-similar* y un valor de α del 10%, considerando distintos tiempos de actualización del CAD, comprendidos entre 10 y 50 segundos, los cuales además coinciden con el tamaño de la ventana seleccionada.



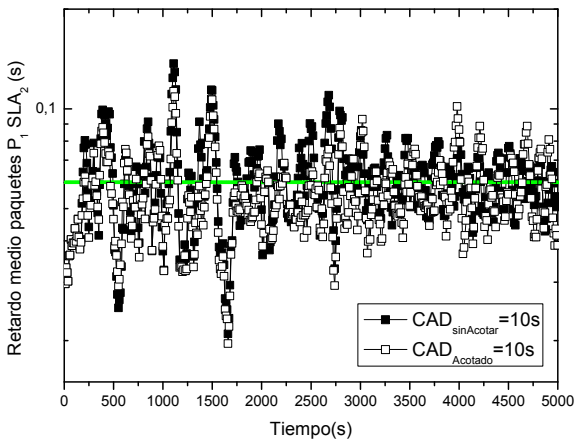
(a)

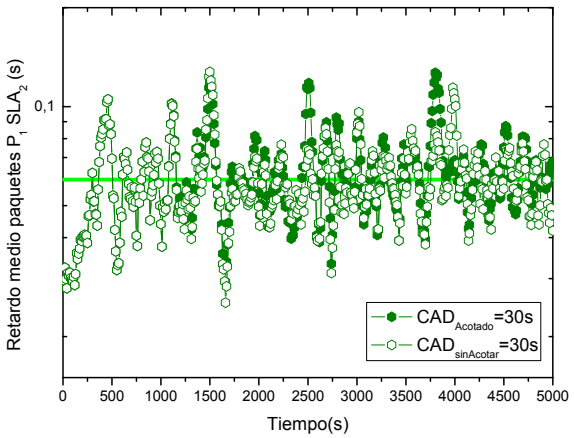

Retardo medio paquetes P1 SLA1 (s)
0,1
0,01
0 500 1000 1500 2000 2500 3000 3500 4000 4500 5000
Tiempo(s)
CADsinAcotar=10s
CADAcotado=10s

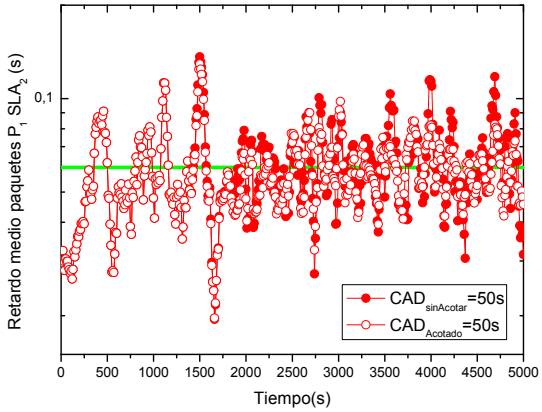

Retardo medio paquetes P1 SLA1 (s)
0,1
0,01
0 500 1000 1500 2000 2500 3000 3500 4000 4500 5000
Tiempo(s)
CADAcotado=30s
CADsinAcotar=30s


Retardo medio paquetes P1 SLA1 (s)
0,1
0,01
0 500 1000 1500 2000 2500 3000 3500 4000 4500 5000
Tiempo(s)
CADsinAcotar=50s
CADAcotado=50s

(b)


Retardo medio paquetes P1 SLA2 (s)
0,1
0 500 1000 1500 2000 2500 3000 3500 4000 4500 5000
Tiempo(s)
CADsinAcotar=10s
CADAcotado=10s


Retardo medio paquetes P1 SLA2 (s)
0,1
0 500 1000 1500 2000 2500 3000 3500 4000 4500 5000
Tiempo(s)
CADAcotado=30s
CADsinAcotar=30s

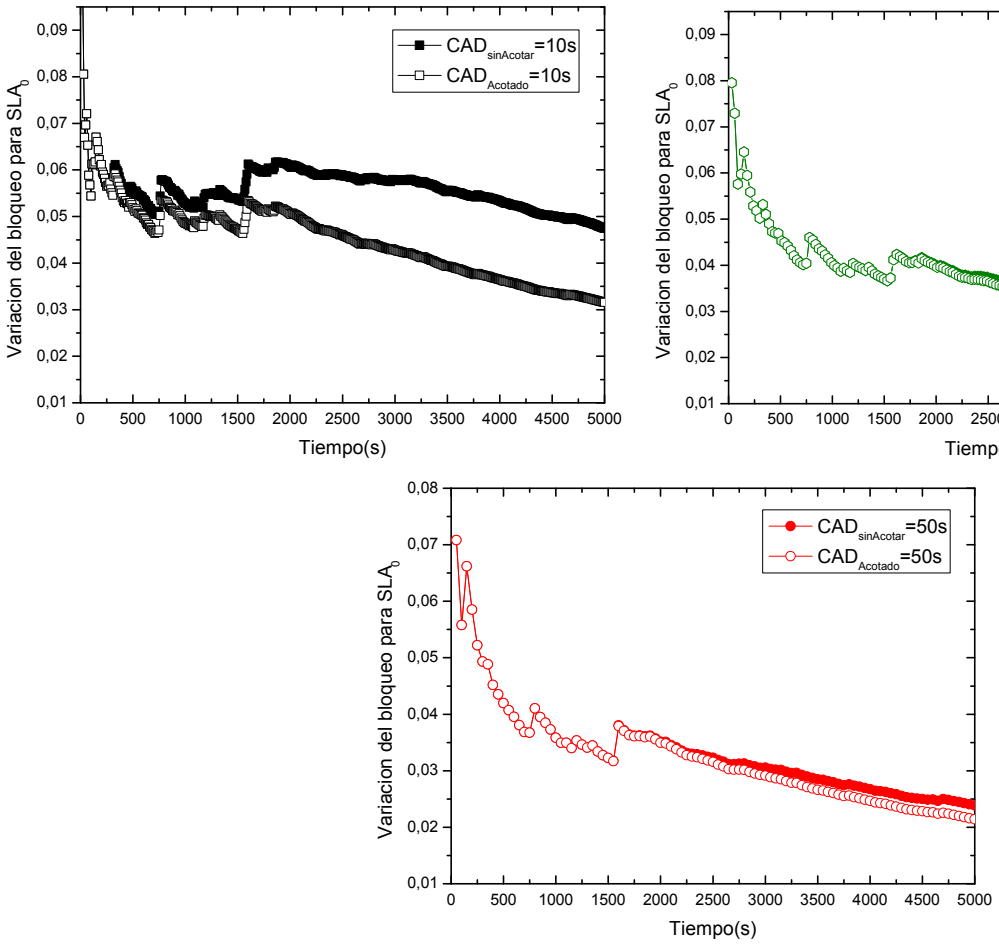


(c)

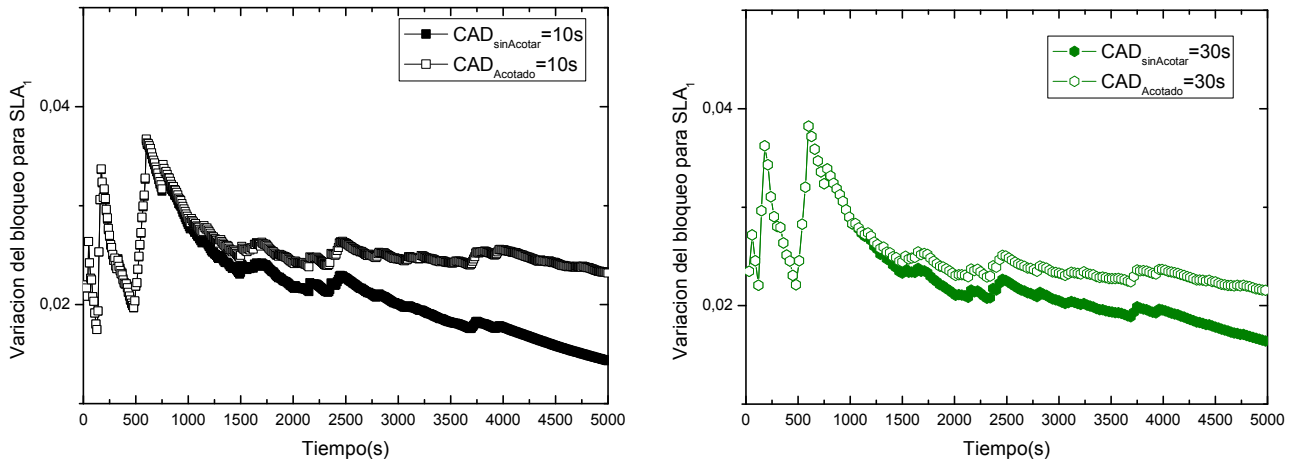
Figura 23. Comparación de la evolución del retardo medio de la clase P_1 entre la implementación de un CAD estático y otro dinámico, para distintos tiempos de actualización del CAD dados (a) SLA_0 (b) SLA_1 (c) SLA_2.

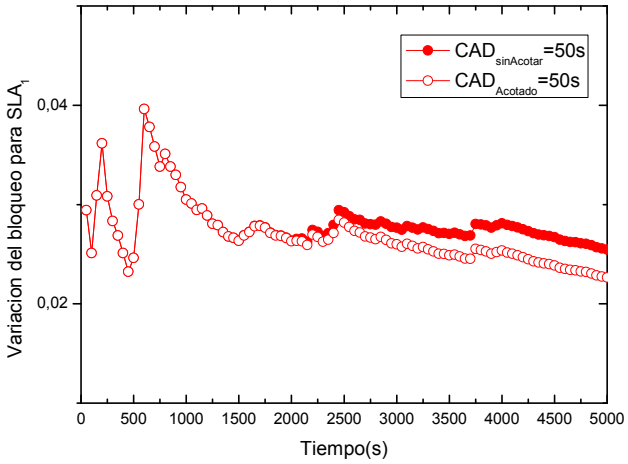
Si nos fijamos en los resultados obtenidos, podemos destacar por un lado que el retardo medio de los paquetes de prioridad P_1 logra situarse de media, para ambos tipos de CAD, por debajo de los umbrales máximos de QoS, que tal y como ya hemos mencionado en otras ocasiones, están establecidos en la Tabla 1. Ahora bien, este comportamiento se ve claramente influenciado por el tamaño de la ventana y por tanto, del tiempo de actualización del CAD seleccionado. Así pues, conviene fijarse en segundo lugar, en el hecho de que con valores pequeños de actualización de nuestro CAD, no podremos afirmar cumplir los requisitos de QoS con tanta seguridad y durante tanto tiempo como para otros valores superiores. Además, para dichos valores tan bajos, se pone de manifiesto la necesidad de acotar el CAD para obtener así unos resultados mucho mejores respecto a las prestaciones de nuestro algoritmo, puesto que, tal y como afirmábamos anteriormente, altas variaciones de CAD influyen negativamente a nuestro algoritmo. Destacar que las conclusiones extraídas han sido también obtenidas para una ventana de 70 segundos.

Por otro lado si comparamos los resultados obtenidos cuando el CAD está acotado y cuando no lo está, se puede ver en la Figura 23 que el retardo medio que se tiene para un CAD cuyo valor se encuentra acotado es menor que el otro, situándose éste en una posición ventajosa por dicho motivo respecto al otro.

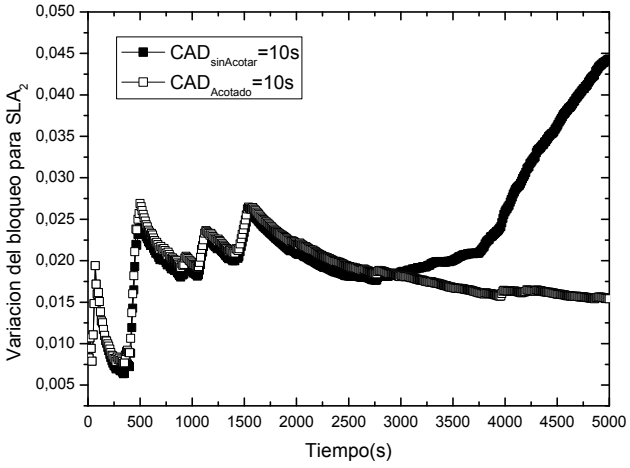

Variacion del bloqueo para SLA0
Tiempo(s)
CADsinAcotar=10s
CADAcotado=10s
CADsinAcotar=30s
CADAcotado=30s
CADsinAcotar=50s
CADAcotado=50s

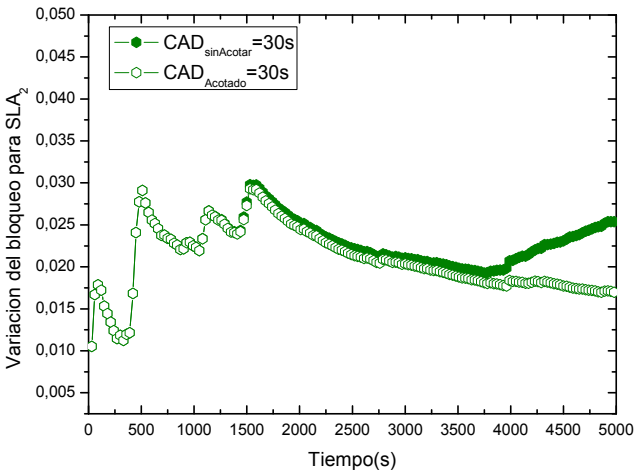
(a)

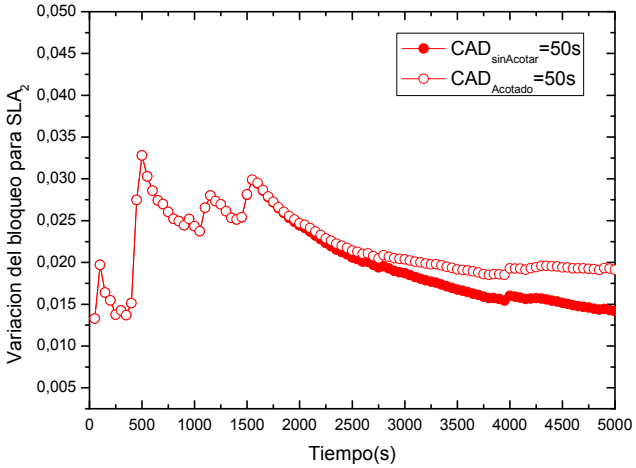

Variacion del bloqueo para SLA1
Tiempo(s)
CADsinAcotar=10s
CADAcotado=10s
CADsinAcotar=30s
CADAcotado=30s



(b)





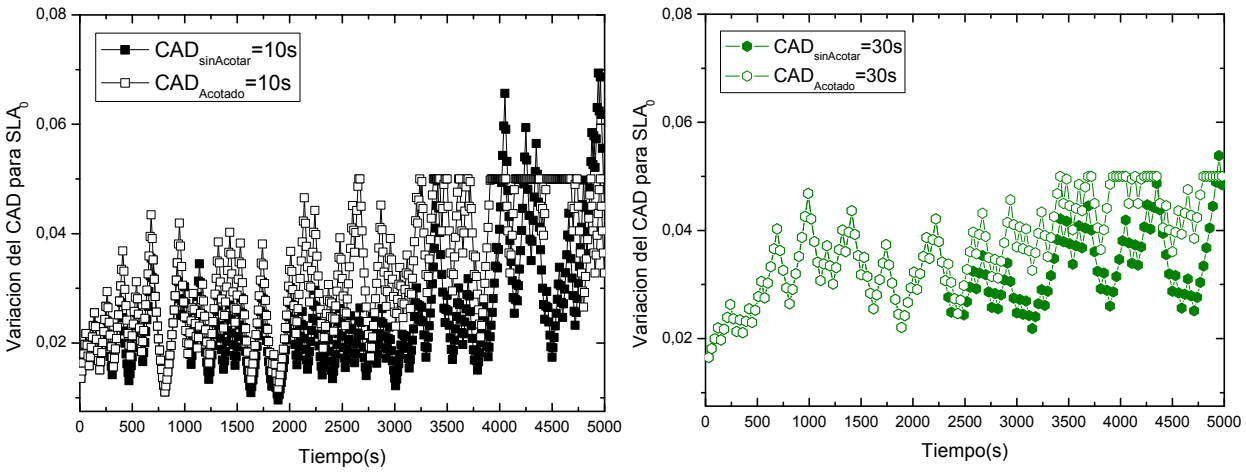


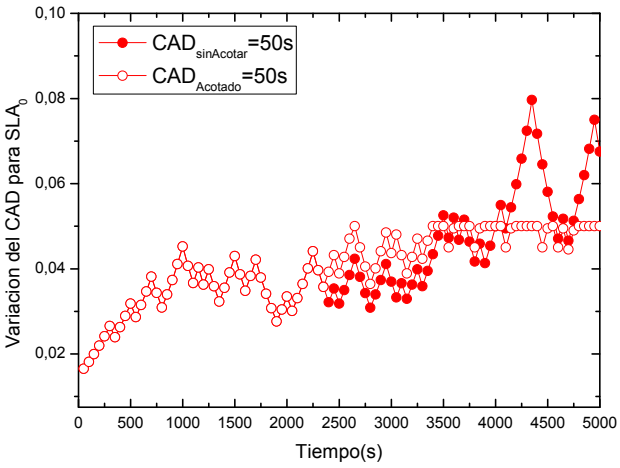
(c)

Figura 24. Comparación de la evolución de la probabilidad de bloqueo entre la implementación de un CAD estático y otro dinámico, para distintos tiempos de actualización del CAD dados (a) SLA_0 (b) SLA_1 (c) SLA_2.

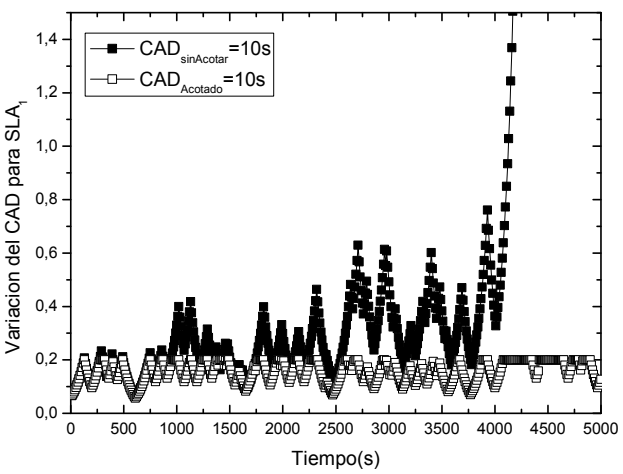
Para tratar de elegir entre un CAD acotado o sin acotar, nos fijamos ahora en la evolución de la probabilidad de bloqueo de nuestro algoritmo mostrada en la Figura 24, bajo las mismas circunstancias de simulación que para el estudio del retardo medio, teniendo en cuenta que para instantes en el que el parámetro de retardo es más estable (al principio de la simulación el algoritmo experimenta mayor inestabilidad), los retardos experimentados por DaSPID con implementación de un CAD dinámico acotado, resultan menores que para un CAD dinámico no acotado. Así pues, como consecuencia a las variaciones del CAD, obtenemos los resultados de bloqueo de la Figura 24, donde se observa que para el CAD acotado la probabilidad de pérdida de paquetes es ligeramente menor que para el CAD sin acotar, independientemente del tamaño de ventana analizado y para todos los perfiles de abonado.

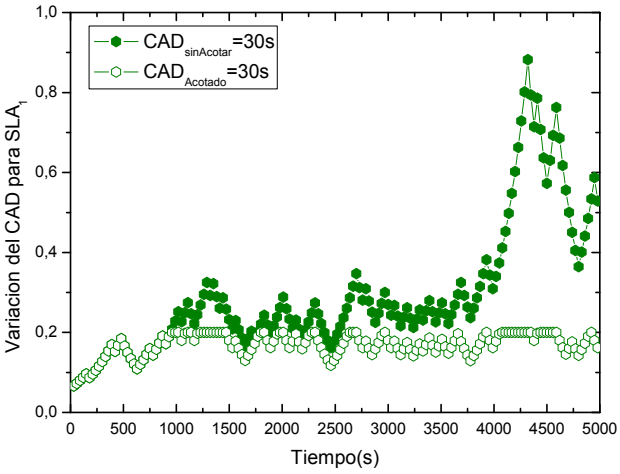
Finalmente, respecto a la variación del CAD, para el caso de valores del CAD acotados, en la Figura 25 se puede apreciar cómo se llega a estos umbrales máximos y mínimos, siendo de 0.6 segundos para el SLA_2, 0.2 segundos para el SLA_1 y 0.05 segundos para el SLA_0 ya que estos umbrales son 10 por el retardo máximo permitido para cada una de las clases de servicio. Sin embargo, se observan muchas más fluctuaciones de estos niveles cuando se considera el CAD sin acotar. Este mayor nivel de fluctuación del nivel del CAD es el que provoca una mayor inestabilidad en el retardo medio de esta clase de servicio prioritaria, así como un peor comportamiento en la probabilidad de pérdida de paquetes.

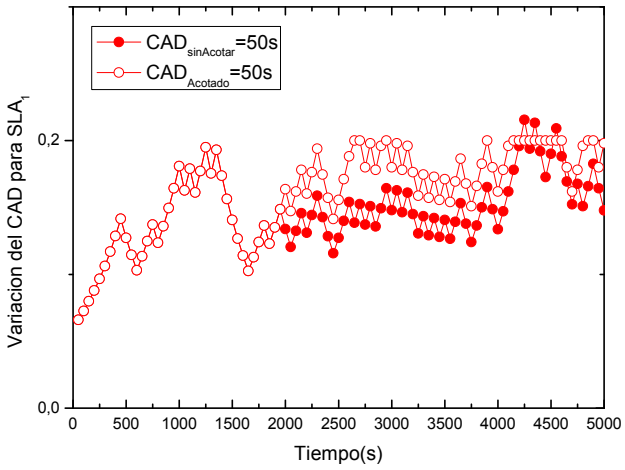



CAD sinAcotar =50s
CAD Acotado =50s
Variacion del CAD para SLA0
Tiempo(s)

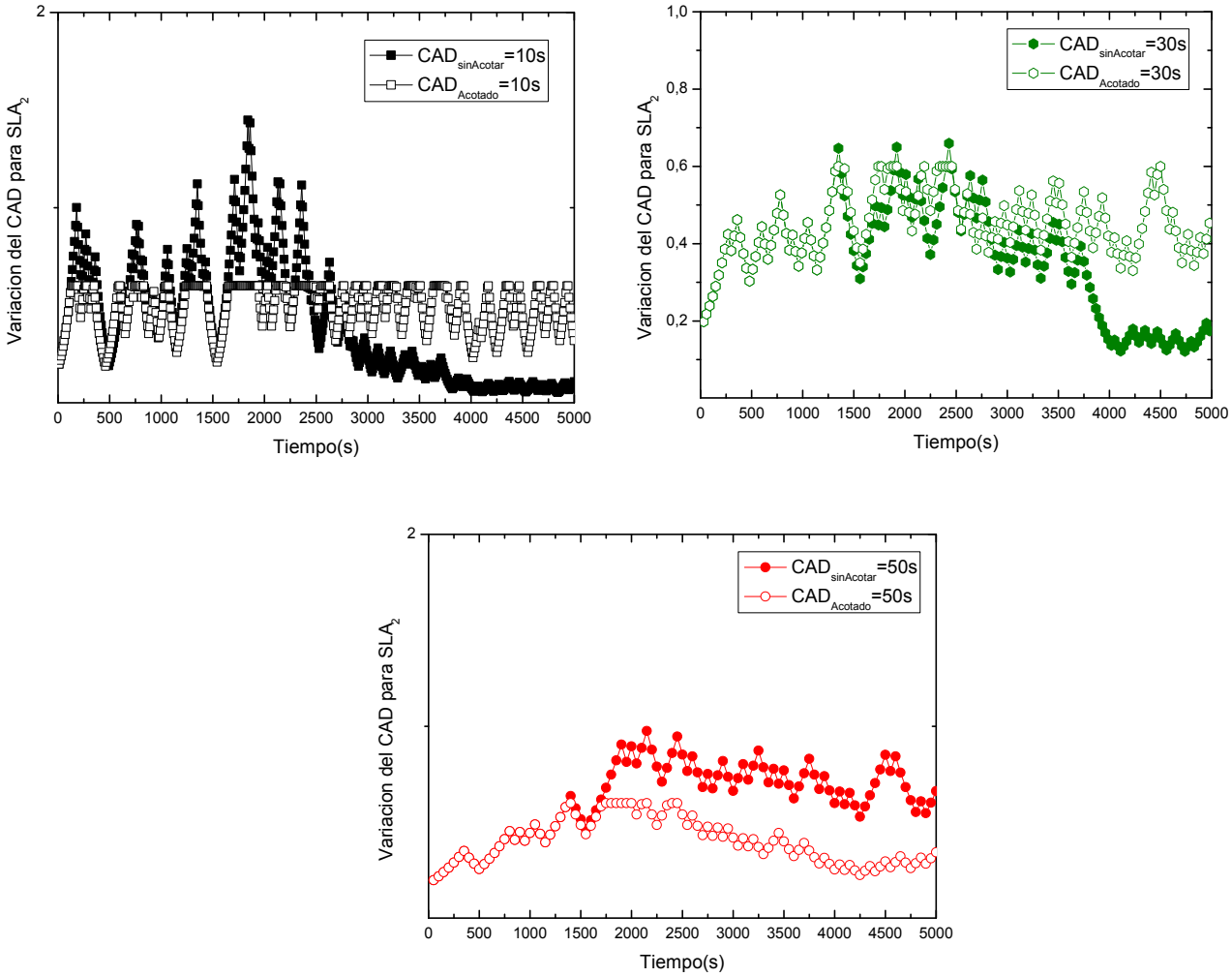
(a)


CAD sinAcotar =10s
CAD Acotado =10s
Variacion del CAD para SLA1
Tiempo(s)


CAD sinAcotar =30s
CAD Acotado =30s
Variacion del CAD para SLA1
Tiempo(s)


CAD sinAcotar =50s
CAD Acotado =50s
Variacion del CAD para SLA1
Tiempo(s)

(b)



(c)

Figura 25. Comparación de la evolución del CAD entre la implementación de un CAD estático y otro dinámico, para distintos tiempos de actualización del CAD dados (a) SLA_0 (b) SLA_1 (c) SLA_2.

A tenor de los resultados obtenidos, se pone de manifiesto la selección de un CAD acotado como forma de optimizar las prestaciones de nuestro algoritmo. Así pues, bajo estas condiciones de CAD acotado, llevamos a cabo una simulación de nuestro algoritmo teniendo en cuenta que el valor permanece acotado entre un umbral máximo y otro mínimo, para distintos tamaños de ventana. De esta manera, analizamos la influencia de dicho valor de ventana en el retardo medio, aunque viendo los resultados ya analizamos previamente, prevemos que las ventanas óptimas serán las de 30 ó 50 segundos.

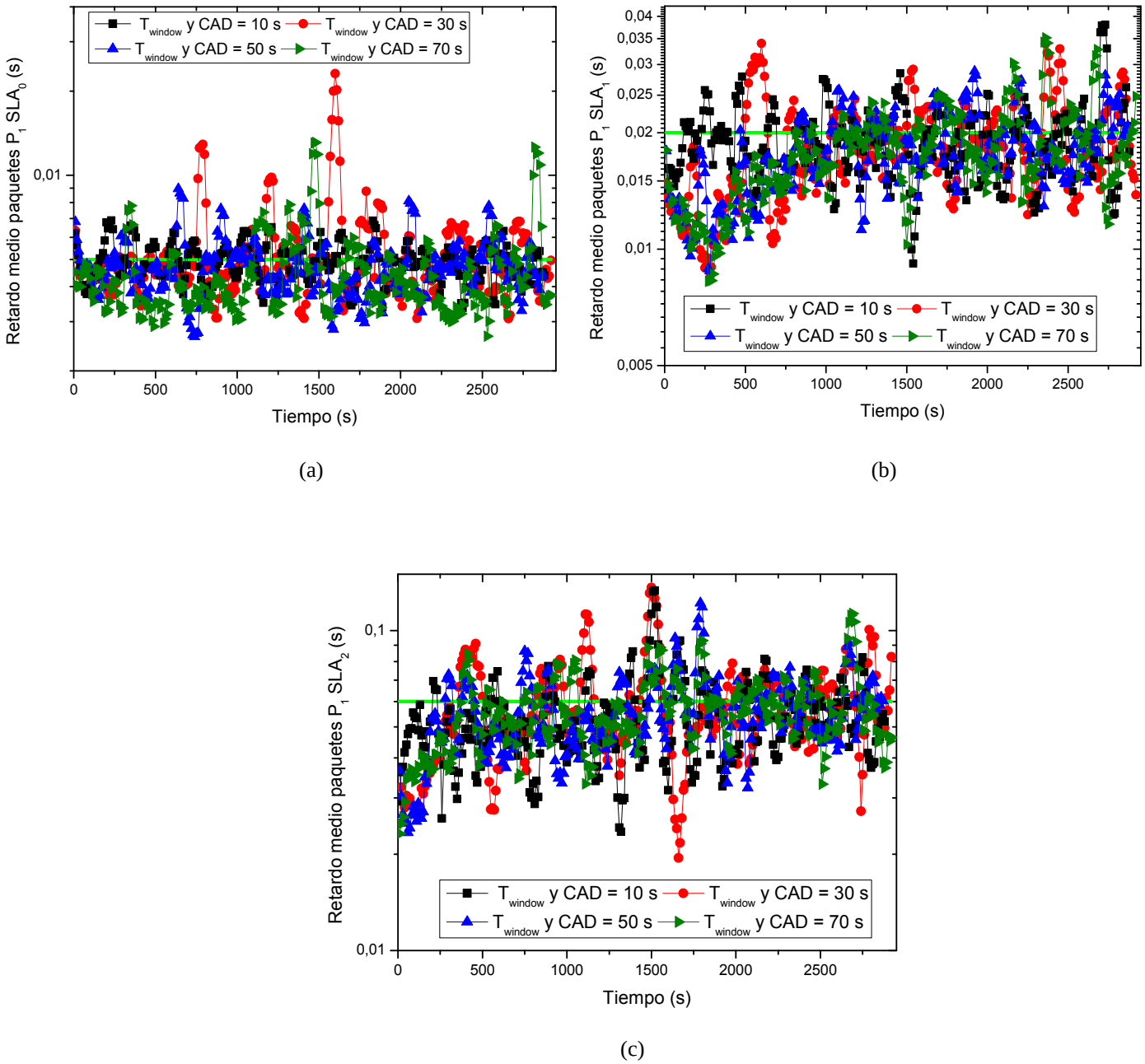


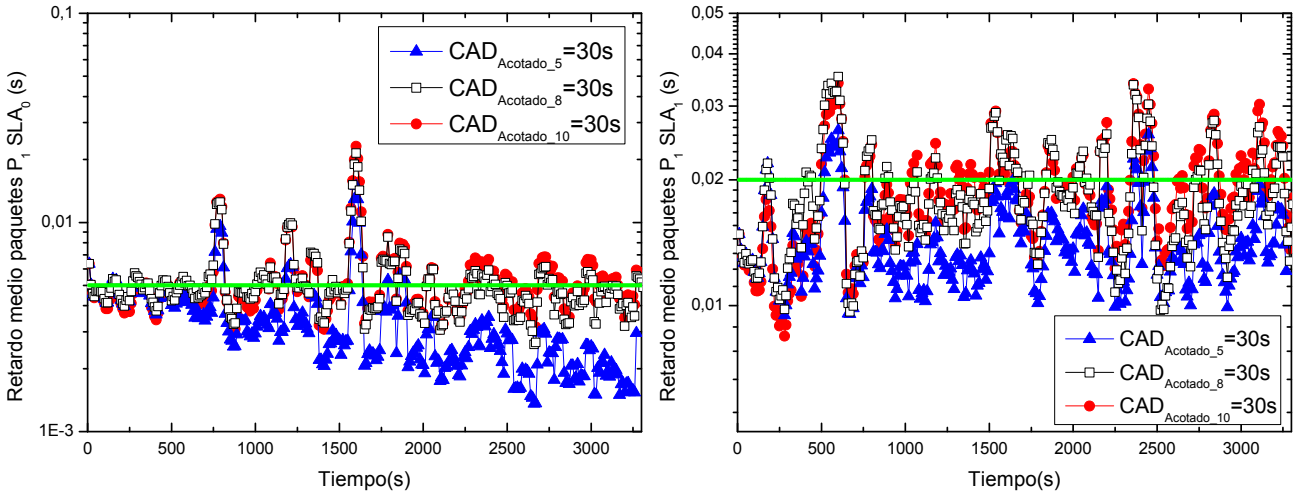
Figura 26. Evolución del retardo medio de la clase P_1, para distintos tiempos de actualización del CAD dados (a) SLA_0 (b) SLA_1 (c) SLA_2 para distintos tamaños de ventana.

Aunque los resultados obtenidos muestren que se sobrepasan los umbrales máximos de QoS, se puede decir que el retardo medio de los paquetes de prioridad P_1 logra situarse de media, por debajo de dichos umbrales para los tipos de ventana observados. Ahora bien, el caso de 10 segundos tenía unas limitaciones nombradas en numerosas ocasiones, tales como alta inestabilidad y eliminación de paquetes, de modo que lo ideal sería quedarnos con una ventana de 30 ó de 50 segundos para obtener las

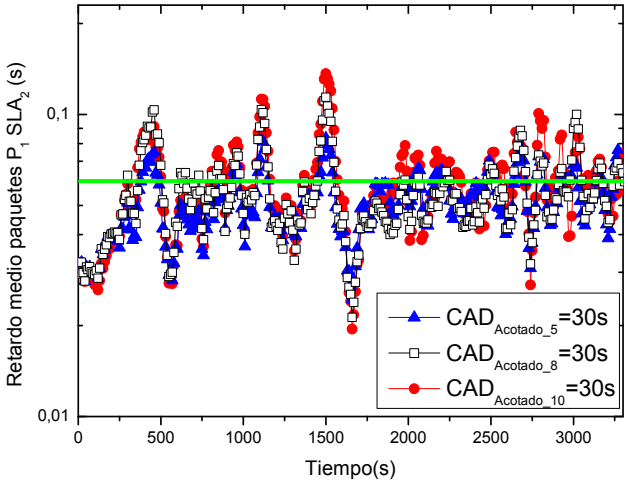
mejores prestaciones en cuanto al retardo medio. Por otro lado, hay que destacar que la obtención de los datos se ha efectuado mediante el lanzamiento de varias simulaciones, teniendo así en este caso una semilla diferente para cada una de ellas, lo cual explica la posición de los picos y los valores de los mismos de cada tipo de ventana. Más adelante se mostrará a modo de ejemplo, como empezando en semillas idénticas los resultados no son tan diferentes entre las ventanas de 30 y 50 segundos como pudiera aparentar la Figura 26.

Para concluir el estudio del algoritmo DaSPID con la implementación de un CAD de tipo dinámico, retomamos un tema ya mencionado anteriormente, esto es, la influencia del valor de α en los parámetros de nuestro algoritmo. De este modo, llevamos a cabo una simulación del algoritmo DaSPID, al cual hemos implementado un CAD dinámico acotado, bajo un patrón de tráfico *self-similar* para unos valores de α del 10%, 5% y 8% para unos tamaños de ventana de 30 y 50 segundos.

En primer lugar llevamos a cabo el análisis del retardo medio y de la evolución del valor del CAD y de la probabilidad de bloqueo experimentada por el algoritmo DaSPID bajo un patrón de tráfico *self-similar*, al cual se le ha implementado un CAD dinámico cuyo valor se encuentra acotado en función del parámetro α, para el que daremos distintos valores de estudio, siendo estos, 5%, 8% y 10% para un tamaño de ventana igual al tiempo de actualización del CAD, siendo este de 30 segundos.



(a) (b)

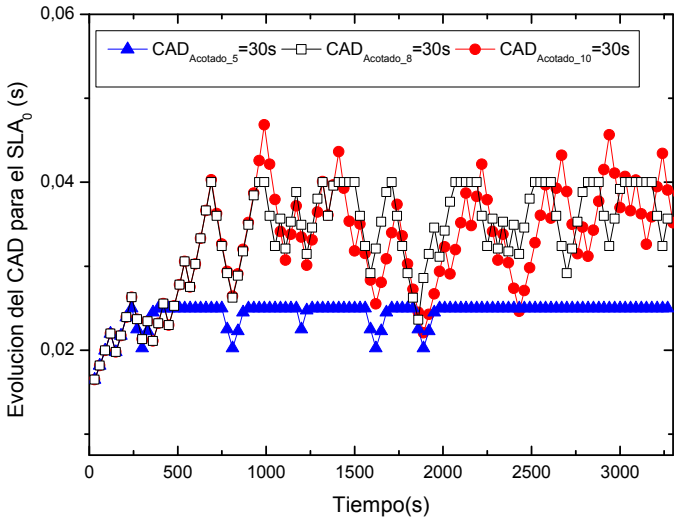


(c)

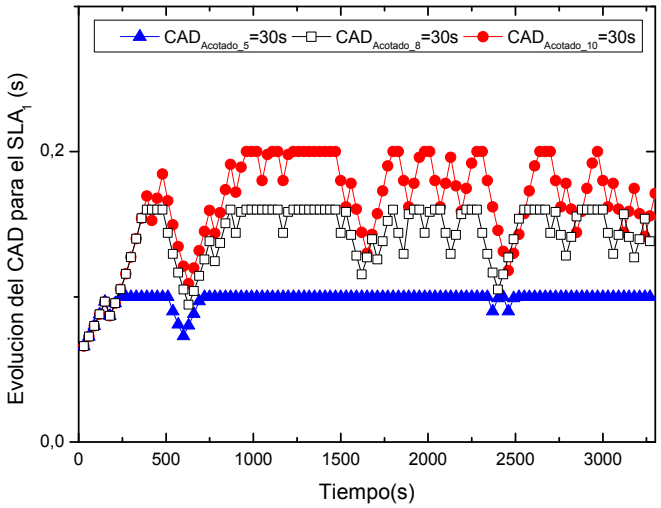
Figura 27. Evolución del retardo medio de la clase P_1, para un tiempo de actualización del CAD de 30 segundos, para (a) SLA_0 (b) SLA_1 (c) SLA_2 dados distintos valores de α.

A tenor de los resultados obtenidos en la Figura 27, se observa que el retardo medio de los paquetes de prioridad P_1 logra situarse de media, para todos los valores de α, por debajo de los umbrales máximos de QoS establecidos en la Tabla 1. Por otro lado, se puede apreciar que los valores máximos alcanzados para los tres valores de α resultan muy similares, siendo levemente superiores para el caso de α=10%. Ahora bien, los valores mínimos alcanzados son mucho menores para el caso de 5%, lo cual supone la eliminación de más paquetes que en los otros dos casos, lo cual es lógico, puesto que el umbral del CAD es menor. De este modo, se podría decir que existe un fuerte compromiso entre el nivel medio de retardo y la probabilidad de pérdida de paquetes a la hora de elegir el umbral máximo del CAD dinámico. Por todo esto, se deberá escoger un valor que permita que el retardo se sitúe de forma estable por debajo de la cota máxima estipulada por el proveedor de servicios, mientras que la probabilidad de pérdida de paquetes no sea demasiado alta.

Si respaldamos estos resultados con los mostrados por la Figura 28, donde se analiza la variabilidad periódica del nivel máximo de CAD, se puede ver que para un valor de α=10%, el CAD experimenta una mayor variabilidad que en los otros casos, siendo mucho más estable para α=5%. Finalmente, si se observa la Figura 29, en la que se presenta la probabilidad de bloqueo para todos los perfiles de abonado, tal y como era de esperar, cuando mayor es el nivel de α, mayor es la variabilidad y el nivel medio del retardo, y en contra, mayor será la probabilidad de pérdida de paquetes. De nuevo, queda patente la necesidad de buscar un compromiso en el valor del parámetro α, para obtener una compensación adecuada entre los niveles de retardo medio y la probabilidad de pérdida de paquetes.



(a)



(b)

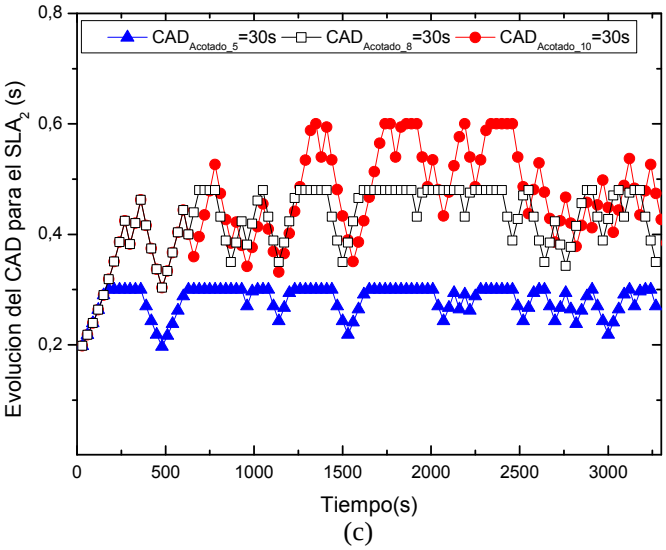


Figura 28. Evolución del valor del CAD para un tiempo de actualización del CAD de 30 segundos para (a) SLA_0 (b) SLA_1 (c) SLA_2 dados distintos valores de α.

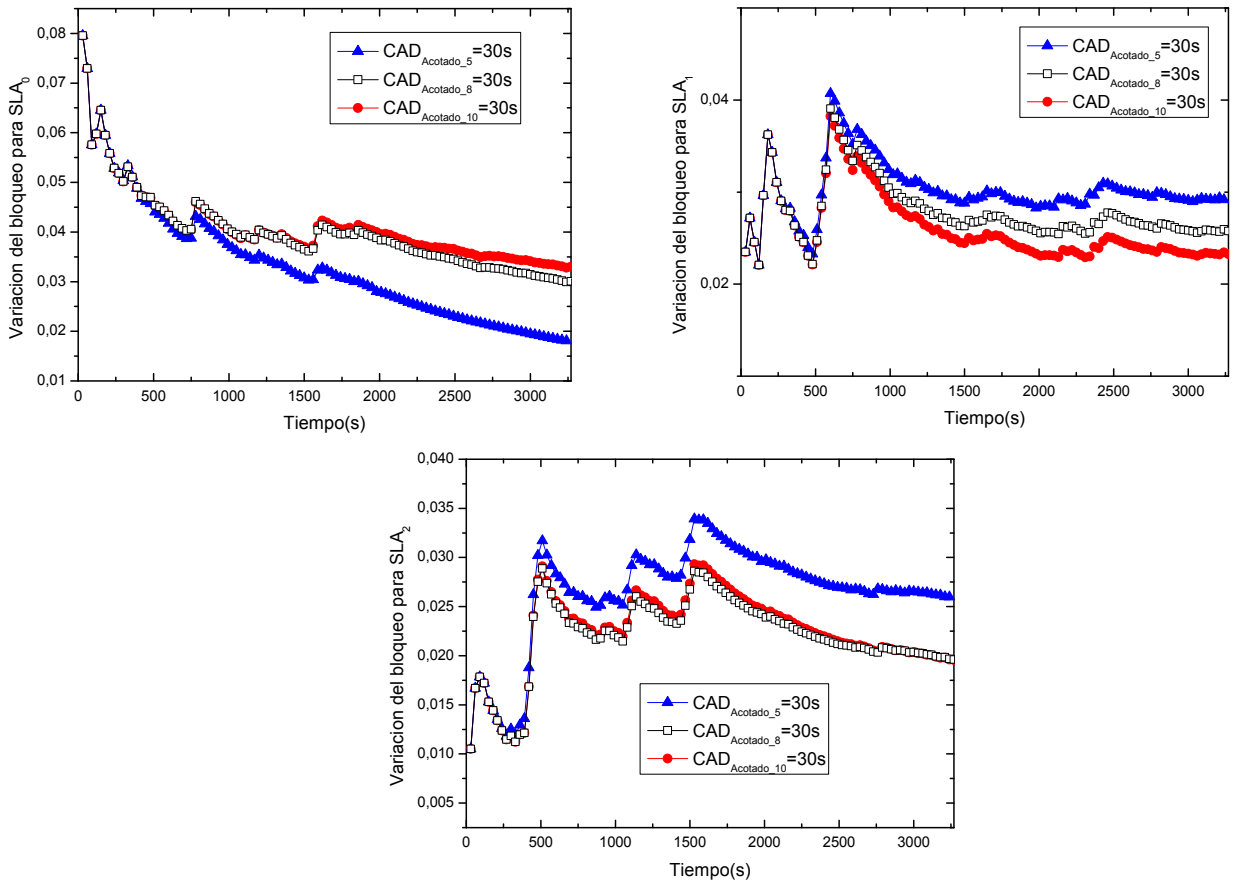
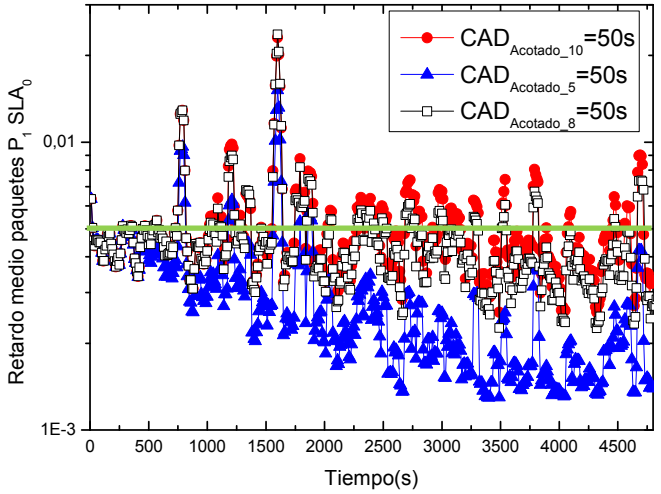


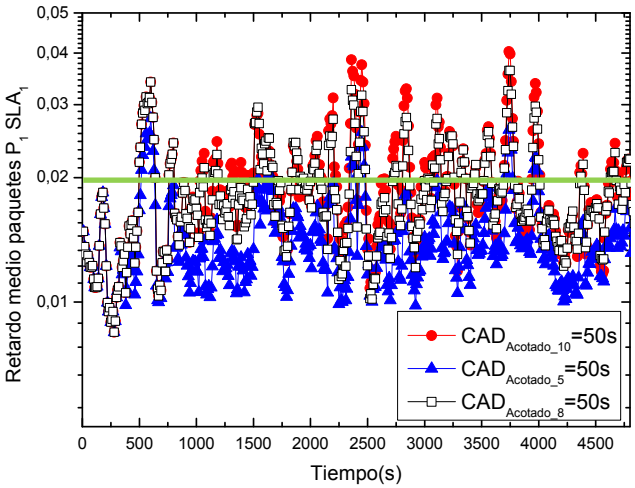
Figura 29. Evolución de la probabilidad de bloqueo para un tiempo de actualización del CAD de 50 segundos para SLA_0, SLA_1 y SLA_2 dados distintos valores de α.

Repetimos ahora el experimento modificando el tamaño de ventana y el tiempo de actualización del CAD, siendo ahora ambos valores de 50 segundos. De este modo, obtenemos la evolución del retardo medio, así como la del valor del CAD reflejados en la Figura 30 y en la Figura 31.

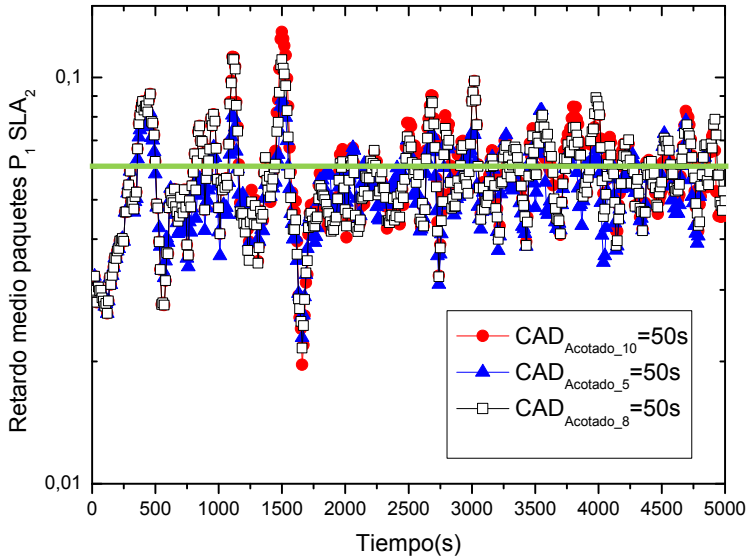
Si nos fijamos por tanto en los resultados obtenidos, vemos que éstos son prácticamente iguales que para el caso anterior. En ambos casos se cumple el mantenimiento del retardo medio de los paquetes de prioridad P_1 por debajo de los umbrales máximos de *QoS* (aunque se observan picos, sobre todo cuanto más alto sea el nivel del CAD). Sobre la evolución del CAD, así como la evolución de la probabilidad de bloqueo vistas en la Figura 31 y en la Figura 32, no podemos aportar nuevas conclusiones que las ya dichas para la Figura 28, ya que su comportamiento es el similar.



(a)

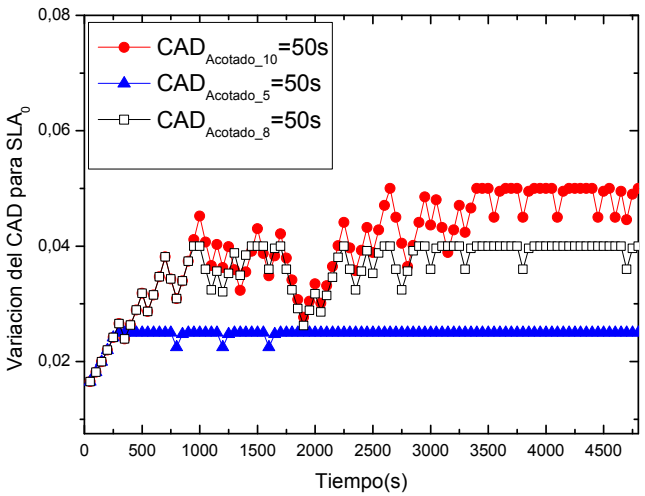


(b)

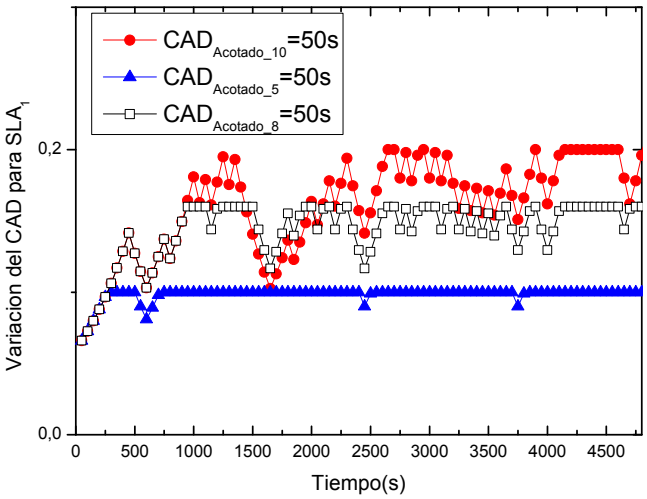


(c)

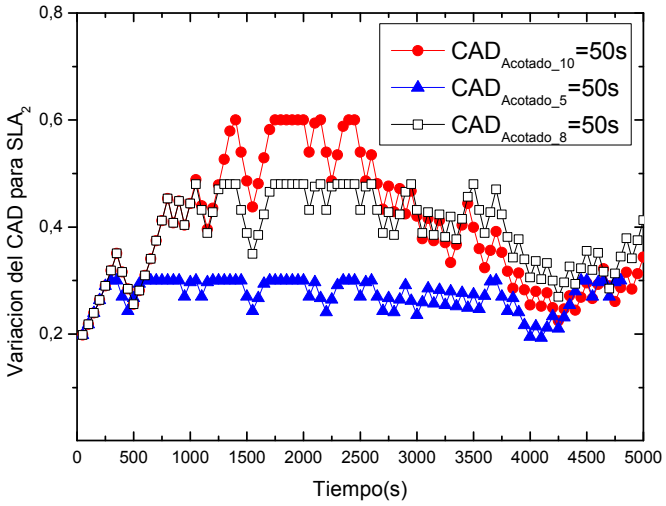
Figura 30. Evolución del retardo medio de la clase P_1, para un tiempo de actualización del CAD de 50 segundos, para (a) SLA_0 (b) SLA_1 (c) SLA_2 dados distintos valores de α.



(a)



(b)



(c)

Figura 31. Evolución del valor del CAD para un tiempo de actualización del CAD de 50 segundos para (a) SLA_0 (b) SLA_1 (c) SLA_2 dados distintos valores de α.

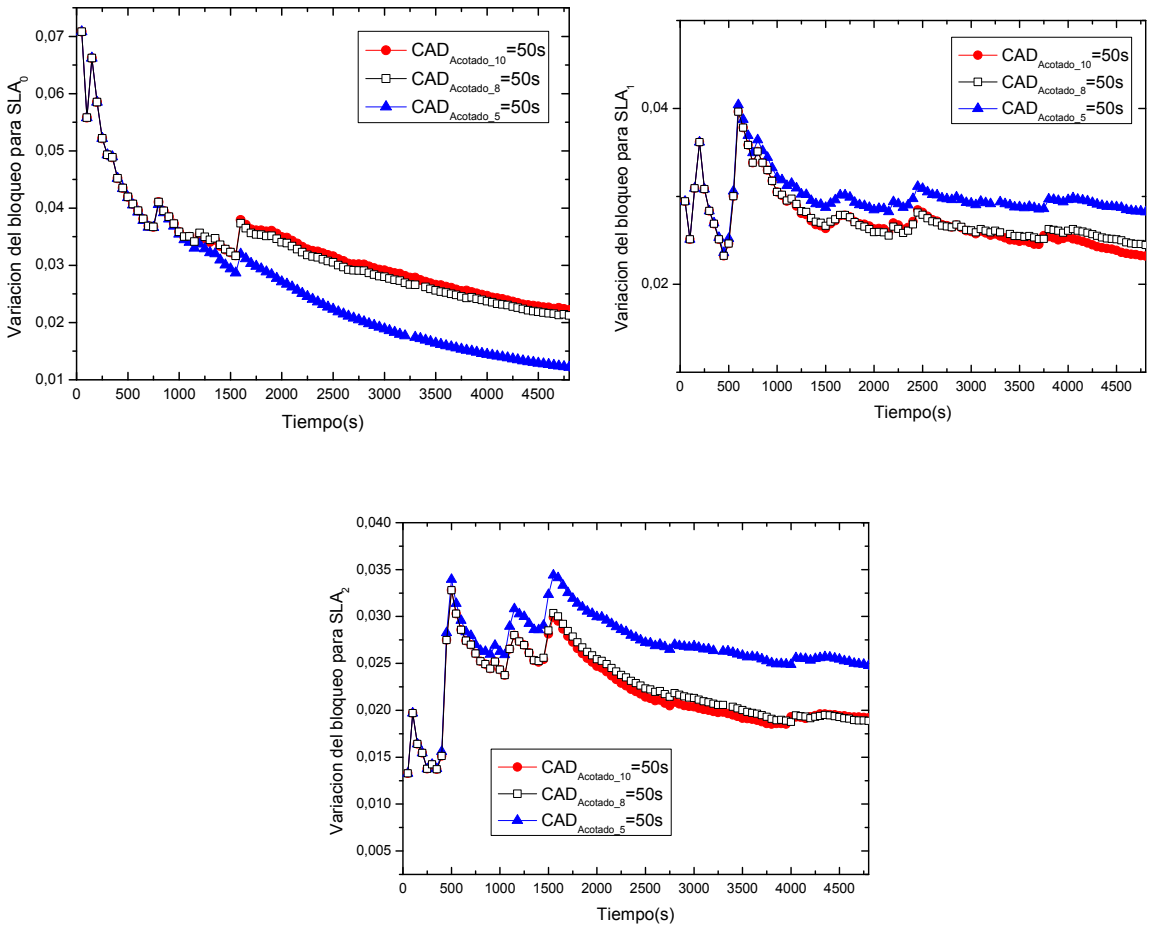


Figura 32. Evolución de la probabilidad de bloqueo para un tiempo de actualización del CAD de 50 segundos para SLA_0, SLA_1 y SLA_2 dados distintos valores de α.

Podemos concluir que la elección de distintos valores de α no aporta diferencias decisivas en el retardo medio de los paquetes para los tiempos de actualización del CAD de 30 segundos y 50 segundos.

7.5 Conclusiones

En este capítulo se han diseñado e implementado dos tipos de control de admisión en las *ONUs*, siendo el primero de ellos un CAD de tipo estático y el segundo de tipo dinámico. Con este control se persigue controlar el retardo del tráfico de prioridad P_1 debido a su carácter altamente rafagoso, el cual fue analizado y mostrado en el capítulo anterior.

El funcionamiento de este sistema de control, se basa en realizar una estimación del retardo de cada paquete de prioridad P_1 entrante (ya que para P_0 este análisis es innecesario al ser un tráfico constante que cumple los requisitos de QoS) y seguidamente este retardo es comparado con un umbral prefijado y distinto según la prioridad del *SLA* que aparece en la Tabla 2.

De este modo, en primer lugar se ha analizado el comportamiento del algoritmo ante la implementación de un CAD estático para diferentes tamaños de ventana ante un patrón de tráfico *self-similar*, consiguiendo así un retardo medio de los paquetes de prioridad P_1 situado de media por debajo de los retardos máximos mencionados, obteniendo los resultados mejores para un tamaño de ventana de 50 segundos.

A continuación, con el fin de mejorar las prestaciones del algoritmo llevando a cabo una mejor adaptación a las necesidades reales de la red EPON en todo momento, se diseñó y se implementó un CAD de tipo dinámico. Con el fin de evitar grandes oscilaciones en el valor de este CAD, se decidió acotar su valor alcanzado, comparando así los resultados aportados por el algoritmo para el CAD acotado y sin acotar para distintos tamaños de ventana y ante un tráfico *self-similar.* Los resultados obtenidos con el segundo de ellos fueron mejores, tales como un retardo medio de P_1 por debajo de los umbrales permitidos de media. En concreto, se decidió seleccionar un valor de 30 ó 50 segundos para la actualización del CAD, coincidiendo este valor con el del tamaño de ventana.

Una vez decidida la mayor viabilidad del CAD dinámico frente al estático, así como la necesidad de acotar el valor de este primero, es en este momento donde se ha estudiado el porcentaje adecuado de variabilidad que debe experimentar el valor del CAD en cada actualización para lograr que la red cumpla los requerimientos de QoS. De esta manera se ha visto que el valor de este porcentaje se debe elegir para obtener un compromiso adecuado entre el retardo medio y la probabilidad de paquetes.

8

Conclusiones y Líneas Futuras

8.1 Conclusiones

En este Proyecto Fin de Carrera se ha llevado a cabo un análisis de algunas de las prestaciones de los algoritmos de asignación dinámica de ancho de banda en redes PON, los cuales habían sido implementados previamente por el GCO, bajo patrones de tráfico *self-similar.* Así pues, la principal motivación de este Proyecto ha sido la gestión de la calidad de servicio en redes PON. Para ello se ha partido de una red de acceso óptica desarrollada e implementada previamente por el GCO en la plataforma de simulación OMNeT++.

Para llevar a cabo esta tarea, en un principio se ha recopilado información sobre el estado del arte en el ámbito de redes de acceso ópticas cuyo despliegue a nivel real se está llevando a cabo de forma masiva en los últimos años. De este modo, se ha procedido a trabajar en el análisis del comportamiento de una red de acceso EPON, como la descrita en el Capítulo 2, ya implementada en el simulador de redes ópticas OMNeT++ en otros proyectos de años anteriores. Así mismo se ha llevado a cabo un análisis de las prestaciones del algoritmo DBA llamado IPACT, explicado en el Capítulo 5, considerando diferentes parámetros de ejecución asociados al diseño de dicho algoritmo dentro de la red bajo un patrón de tráfico *self-similar*. Las conclusiones extraídas de este estudio han sido por un lado, que el algoritmo no se ve influenciado por el número de *streams* empleado, eligiéndose por tanto un valor de 32 *streams* (menos tiempo de computación), y por el otro, que para este valor elegido de *streams* obtenemos los resultados más ventajosos para tamaños de ventana intermedios, capaces de obtener un compromiso entre estabilidad y rapidez de reacción en cualquier parámetro de red (retardo, probabilidad de bloqueo, etc).

Posteriormente se ha realizado un estudio similar para el algoritmo DaSPID, explicado en el Capítulo 6, bajo el mismo patrón de tráfico de carácter rafagoso.

Seguidamente, en relación a este último algoritmo, se ha observado que DaSPID podía controlar eficientemente el retardo de la clase de servicio más prioritaria P_0, correspondiente a tráfico de tasa binaria constante, pero no se refleja ese mismo comportamiento para el tráfico de prioridad P_1, debido fundamentalmente a que este tráfico es generado a partir de fuentes auto-semejantes. Por consiguiente, se ha desarrollado e implementado un controlador de admisión de paquetes, CAD, el cual ha sido aplicado a las colas de los usuarios asociados a una estación *ONU* para así lograr una gestión eficiente del retardo máximo de las clases de servicio prioritarias.

De este modo se ha elaborado en primer lugar un CAD de tipo estático, es decir, aquel cuyos parámetros son inmutables, pero se ha llegado a la conclusión que un carácter adaptativo del mismo podía ser altamente ventajoso para satisfacer las necesidades reales de la red. Siguiendo esta conclusión se creó el CAD de tipo estático, cuyos resultados, obtenidos mediante la variación de algunos parámetros que lo componen, han sido más ventajosos que el anterior. Como conclusión final a esta sección destacar que el algoritmo DaSPID con un CAD dinámico implementado se ha perfilado como la óptima opción.

8.2 Líneas Futuras

La implementación y validación de nuevas estrategias de control en el simulador de redes de acceso en la plataforma de simulación OMNeT++, ha propiciado la aparición de nuevas líneas de investigación que se pueden seguir una vez concluido este Proyecto Fin de Carrera.

Por otra parte, en cuanto al algoritmo de gestión dinámica de ancho de banda desarrollado para controlar el retardo medio de tráfico prioritario, sería de gran interés seguir implementando nuevas modificaciones de la técnica de control de admisión diseñada, para mejorar los resultados de QoS obtenidos en este Proyecto Fin de Carrera.

Así mismo, sería de gran interés incorporar esta técnica de control de admisión de paquetes para gestionar de forma óptima otros parámetros de red, tales como el ancho de banda medio o la probabilidad de pérdida de paquetes.

Finalmente, sería de gran interés plantearse el diseño y desarrollo de otros algoritmos de gestión de recursos en los que se tenga en cuenta el control simultáneo de diferentes parámetros de red, para así realizar una gestión más eficiente y global del comportamiento de la misma.

9

Bibliografía

[1] C.-H. Lee, W. V. Sorin, and B. Y. Kim, "Fiber to the home using a PON infrastructure", *IEEE/OSA Journal of Lightwave Technology*, vol. 24, no. 12, pp. 4568–4583, Dec. 2006.

[2] K.Grobe and J.-P. Elbers, "PON inadolescence: From TDMA to WDM-PON", *IEEE Communications Magazine*, vol. 46, no. 1, pp. 26–34,Jan. 2008.

[3] G.-K. Chang, A. Chowdhury, Z. Jia,H.-C. Chien, M.-F. Huang, J. Yu, and G. Ellinas, "Key technologies of WDM-PON for future converged optical broadband access networks", *Journal of Optical Communication Networking*, vol. 1, no. 4, pp. C35–C50, 2009.

[4] Y. Li, J. Wang, C. Qiao, A. Gumaste, Y. Xu,and Y. Xu, "Integrated fiber-wireless (FiWi) access networks supporting inter-ONU communications", *IEEE/OSA Journal of Lightwave Technology*, vol. 28, no. 5, pp. 714–724, Mar. 2010.

[5] A. Banerjee, G. Kramer, Y. Ye, S. Dixit y B. Mukherjee, "Advances in Passive Optical Networks (PONs)", en Emerging optical network technologies: Architectures, Protocols and Performance, K. M. Sivalingnan y S. Subramaniam, Eds. Spring Street, Nueva York: Springer 2005.

[6] A. Vargas and OpenSim Ltd., "Omnet++ User Manual". Disponible *on-line* en: http://www.omnetpp.org/doc/omnetpp/manual/usman.html

[7] A. Vargas and OpenSim Ltd., "Omnet++ User Guide". Disponible *on-line* en: http://www.omnetpp.org/doc/omnetpp/UserGuide.pdf

[8] G. Kramer, B. Mukherjee, G. Pesavento, "IPACT: A dynamic protocol for an Ethernet PON (EPON)", IEEE Communications Magazine, vol. 40(2), pp. 74-88, Dic. 2002.

[9] G. Kramer, B. Mukherjee, G. Pesavento, "Interleaved Polling with Adaptive Cycle Time (IPACT): A Dynamic Bandwidth Distribution Scheme in an Optical Access Network," *Photonic Network Communications*, vol. 4(1), pp. 89-107, Dec. 2002.

[10] B. Lung, "PON architecture Future proofs FTTH," *Lightwave*, vol. 16(10), pp. 104-107, Nov. 1999.

[11] M. Pesavento y A. Kesley, "PONs for the Broadband Local Loop," *Lightwave*, vol. 16(10), pp. 68-74, Nov. 1999.

[12] IEEE 802.3 (2006) Call For Interest: 10 Gbps PHY for EPON. Disponible en: http://www.ieee802.org/3/cfi.

[13] Página Web Oficial de OMNeT++: http://www.omnetpp.org/

[14] Generador de tráfico *Self-Similar* desarrollado por Kramer. Disponible en: http://wwwcsif.cs.ucdavis.edu/~kramer/research.html.

[15] "Pareto Distribution", Princeton University Repository, 2012, [Online], http://www.princeton.edu/~achaney/tmve/wiki100k/docs/Pareto_distribution.html.

[16] Versión 1 del Generador de tráfico *Self-Similar* desarrollado por Kramer. Disponible en: http://wwwcsif.cs.ucdavis.edu/~kramer/code/trf_gen1.html.

[17] Versión 2 del Generador de tráfico *Self-Similar* desarrollado por Kramer. Disponible en: http://wwwcsif.cs.ucdavis.edu/~kramer/code/trf_gen2.html.

[18] Versión 3 del Generador de tráfico *Self-Similar* desarrollado por Kramer. Disponible en: http://wwwcsif.cs.ucdavis.edu/~kramer/code/trf_gen3.html.

[19] T. Berisa, A. Bazant, V. Mikac, "Bandwidth and delay guaranteed polling with adaptative cycle time (BDGPACT): a scheme for providing bandwidth and delay guarantees in passive optical networks", *Journal of Optical Networking*, Vol. 8, Issue 4, April, 2009.

[20] B. Kantarci, H.T. Mouftah, "Delay-Constrained Admission and Bandwidth Allocation for Long-Reach EPON", *Journal of Networks*, Vol. 7, Issue 5, May 2012.

[21] B. Kantarci, H.T. Mouftah, "On SLA constraints in dynamic bandwidth allocation for long-reach passive optical networks", *Proceedings of the 12th International Conference on Transparent Optical Networks* (ICTON 2010), pp. 1,7, July 2010.