UNIVERSIDAD DE GRANADA

DEPARTAMENTO DE

TEORÍA DE LA SEÑAL, TELEMÁTICA Y COMUNICACIONES

DESCARTE SELECTIVO DE PAQUETES EN MECANISMOS DE GESTION ACTIVA DE COLAS

ERIKA PATRICIA ALVAREZ FLORES

TESIS DOCTORAL

Granada, España. Abril, 2009

Editor: Editorial de la Universidad de Granada
Autor: Erika Patricia Álvarez Flores
D.L.: GR 2311-2009
ISBN: 978-84-692-3113-5

D. Juan Manuel López Soler
Profesor Titular de Ingeniería Telemática del
Departamento de Teoría de la Señal, Telemática y Comunicaciones de la
Universidad de Granada

CERTIFICA:

Que la presente memoria, titulada *Descarte Selectivo de Paquetes en Mecanismos de Gestión Activa de Colas*, ha sido realizada por Dña. Erika Patricia Alvarez Flores bajo su dirección en el Departamento de Teoría de la Señal, Telemática y Comunicaciones de la Universidad de Granada. Esta memoria constituye la Tesis que Dña. Erika Patricia Alvarez Flores presenta para optar al grado de Doctor por la Universidad de Granada.

Granada, abril de 2009

_____

Fdo.:
Dr. D. Juan Manuel López Soler
Director de la Tesis

Cada selección que tú hagas asume un resultado final.

- Zig Ziglar

Si buscas resultados distintos, no hagas siempre lo mismo.

- Albert Einstein

A mis padres y hermanos

Resumen

La adopción del protocolo IP para servir diversas aplicaciones pone de manifiesto la necesidad de nuevos mecanismos de prevención de la congestión en escenarios con diferentes tipos de tráfico (reactivo y no-reactivo) compartiendo limitados recursos de red. Basados en este problema, una plétora de algoritmos de gestión activa de colas (AQM) ha sido propuesto en la literatura. Sin embargo, la mayoría de ellos ignora el conocimiento del tipo de tráfico asociado con los paquetes. Como consecuencia, en términos generales, los algoritmos AQM controlan eficientemente el problema de la congestión sin tomar en cuenta el efecto sobre la QoS para los distintos tráficos proporcionando un tratamiento no diferenciado a las distintas fuentes. En este sentido, el servicio proporcionado puede no coincidir con los requerimientos característicos de los diferentes tipos de tráficos, tal como los servicios de VoIP, que demandan retardos extremo-a-extremo de paquete limitados y tasas de pérdida acotadas.

Motivados en reforzar las medidas para el control de congestión que impliquen acentuar las prestaciones al tráfico multimedia en cuanto a los parámetros que definen la QoS (pérdidas de paquetes y retardo extremo-a-extremo), un esquema de descarte de paquetes ha sido definido dentro de la estructura de un *router* AQM.

La primera contribución de esta tesis es un procedimiento de selección de víctima en estrategias AQM denominado *Drop-Sel*. Considerando que los flujos con más alta carga de trabajo son los principales contribuyentes al evento de congestión, el esquema propuesto se basa fundamentalmente en identificar y penalizar la clase de tráfico con la más alta consumición de memoria en la cola. *Drop-Sel* es analizado y evaluado con simulaciones en escenarios con diferentes grados de combinación de tráficos UDP y TCP. Se proporcionan medidas objetivas y subjetivas de funcionamiento. La evaluación

experimental ha demostrado que este esquema aporta beneficios en términos de equidad -entre diferentes clases de tráficos- que no pueden proporcionar los tradicionales algoritmos (*Drop-Tail*, *Drop-Random* y *Drop-Front*) utilizados en el contexto de descarte de paquetes.

En la segunda contribución se ha considerado reforzar la QoS y QoE de la clase *real-time* (VoIP). Se han desarrollado *Drop-Sel-delay* y *Drop-Tail-delay,* la primera de éstas es una variante de *Drop-Sel* y otra de *Drop-Tail.* Ambas alternativas consideran un criterio basado en la utilidad del paquete cuando la clase seleccionada como víctima corresponde a VoIP. Estos esquemas facilitan que los paquetes de audio considerados no-aprovechables sean detectados y descartados anticipadamente y el número de pérdidas de paquetes útiles pueda ser reducido. En consecuencia, mejor QoS puede ser esperada para el tráfico VoIP.

La tercera contribución son dos esquemas denominados *Drop-Sel+fair* y *Drop-Tail+fair.* Estas variantes respectivamente de *Drop-Sel* y *Drop-Tail* garantizan equidad a diversos flujos de tráfico VoIP que utilizan diferentes *codecs* o diferentes tasas de envío.

Experimentalmente, es mostrado y verificado que las alternativas, *Drop-Sel-delay*, *Drop-Tail-delay, Drop-Sel+fair* y *Drop-Tail+fair* contribuyen a reforzar las aportaciones propias del descarte selectivo.

Agradecimientos

Mi agradecimiento a todas aquellas personas que han contribuido, de una forma u otra, a hacer posible esta Tesis.

A Juan Manuel López Soler, por su dedicación en la dirección de ésta y su ayuda a lo largo de estos años.

Al Centro de Estudios Superiores del Estado de Sonora que, junto con el *Programa de Mejoramiento del Profesorado (PROMEP)*, permitió y financió este proyecto académico.

Sin duda tampoco habría sido posible realizar esta Tesis sin la colaboración del grupo de investigación del Departamento de Teoría de la Señal, Telemática y Comunicaciones de la Universidad de Granada. En especial, a JuanJo que siempre estaba disponible cuando lo necesitaba. Por sus innumerables opiniones científicas que más de una vez me han sacado de algún atolladero.

A mi familia, por haber confiado siempre en mí y en mis planes. Gracias por apoyarme en todos los momentos y decisiones de mi vida.

A Paco, por todo su cariño y los ánimos brindados para seguir adelante.

A mis amigos (en México y España), por su aliento, confianza y ayuda para saltar los problemas del camino académico y personal durante todos estos años.

Y finalmente a Dios, que ha sido mi guía espiritual y ha permitido que llegase hasta esto.

Contenido

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Contextualización del problema

Internet ha crecido vertiginosamente desde sus orígenes tanto en número de usuarios como en prestaciones y servicios ofrecidos. Su despliegue, inicialmente circunscrito al ámbito militar, se ha generalizado progresivamente alcanzando tanto a los ámbitos académicos, como empresariales y, finalmente, a toda la población. Este éxito se ha fundamentado en la generalización del servicio *World Wide Web* así como el correo electrónico, ambos basados en tecnología IP. El enorme aumento del número de conexiones y la demanda cada vez mayor de nuevos servicios, ha propiciado que los servicios ofrecidos por Internet no sólo sean mejores y más especializados, lo que ha significado un proceso continuo de innovación, sino que además, ha significado un desafiante *test* de escalabilidad a los diseños originales.

Recientemente, la tecnología IP se está erigiendo como la base para conseguir la convergencia de servicios, siempre deseada por los operadores de telecomunicación. De esta forma el protocolo IP, aunque inicialmente no fue concebido para ello, está siendo utilizado para transportar información multimedia con requisitos de tiempo real, proporcionando servicios cada vez más extendidos como son los de telefonía, televisión, radio y videoconferencia. Aunque se han conseguido grandes aportaciones en el desarrollo de las técnicas de codificación (mejorando el compromiso entre calidad y *bit-rate*), el creciente número de usuarios hace que las demandas de ancho de banda crezcan constantemente. El buen funcionamiento de las aplicaciones se puede ver afectado al no

poder garantizar el ancho de banda solicitado, su deterioro puede causar problemas significativos a los usuarios multimedia, lo que se materializa en una pobre calidad de servicio.

La calidad de servicio (QoS, *Quality of Service*) es un concepto multidimensional que se caracteriza por el retardo extremo a extremo introducido, su fluctuación (o *jitter*), por la probabilidad de pérdidas y por su disponibilidad. El servicio que ofrece el protocolo IP [Postel, 1981a], no orientado a conexión y de mejor esfuerzo (*best effort*), no siempre satisface la calidad de servicio demandada por todas las aplicaciones, especialmente por aquellas con requerimientos de tiempo real. Este tipo de aplicaciones no pueden tolerar grandes variaciones de retardo o grandes probabilidades de pérdidas de paquetes, por lo que, sin menoscabar las prestaciones de los otros servicios, la transmisión de audio y vídeo de buena calidad en tiempo real sobre IP ha demandado y demanda grandes esfuerzos de investigación y desarrollo.

Los recursos disponibles (ancho de banda y capacidad de almacenamiento en *routers*) son siempre limitados, mientras que las demandas por lo general siempre han sido crecientes. Y lo que es más, en ciertas condiciones de pico, incluso cuando el dimensionado de recursos es adecuado, nada impide que localmente la demanda sea superior a la disponibilidad. Para paliar estos problemas, desde sus inicios, se incorporaron medidas para el control de congestión. A pesar de ello, fundamentado por las diferencias entre demanda y oferta, así como por las características singulares que las aplicaciones multimedia exigen, uno de los temas que más investigación y recursos ha suscitado ha sido la mejora de prestaciones de los procedimientos para control de congestión. Entre los muchos propuestos, por su éxito, cabe mencionar a los mecanismos de gestión activa de colas *(*AQM, *Active Queue Management*) [Braden *et al*., 1998]. Resumidamente, estos procedimientos consisten en la detección temprana de la congestión reaccionando en base a criterios probabilísticos, aproximación ésta en la que se desarrolla la investigación realizada.

1.2. Motivación

La mayoría de las aplicaciones de Internet sin requisitos de tiempo real, utilizan el protocolo TCP para recuperar los paquetes perdidos en situaciones de congestión, así como para adoptar medidas reactivas en caso de congestión regulando el volumen de tráfico generado. En cambio, con el fin de establecer el menor retardo posible, el protocolo de transporte para las aplicaciones multimedia en Internet utiliza por lo general los servicios de UDP. Al no ofrecer control de congestión, UDP no es sensible a la dinámica de la red, como si lo hace TCP. Por lo que, cuanto mayor sea la cantidad de flujos UDP que se estén transmitiendo en la red, aumenta la probabilidad de sobrecarga en los *routers*.

La severidad del problema de la congestión crece en magnitud al crecer el volumen de las demandas de tráfico. De ahí que el problema de la congestión se ve agravado por el hecho de la aparición de aplicaciones que utilizan audio y vídeo, las cuales generan enormes cantidades de tráfico a veces de forma regular, pero otras de forma impredecible (dependiendo de que el codificador sea de *bit-rate* constante o no).

El protocolo de Internet (IP), como previamente se ha comentado, es un protocolo no orientado a la conexión, proporciona un servicio de paquetes no fiable denominado *"best effort*", su función sólo es hacer el mejor esfuerzo para entregar un paquete.

En el intento de entrega al nodo destino, en cada *router* (basados en *store-and-forward*) pueden converger desde uno hasta miles de flujos de paquetes. Si la razón de llegadas de tráfico en un *router* es superior a la razón de salidas (impuesta por el ancho de banda disponible en el correspondiente enlace), los paquetes recibidos, tras ser inspeccionados y conmutados son temporalmente almacenados en la cola de salida. Dichos paquetes encolados en la memoria de la interfaz de salida (de naturaleza FIFO) serán posteriormente reenviados a su correspondiente IP destino de acuerdo con la tabla de encaminamiento. Esta cola puede estar constituida por paquetes de distintos tipos de flujos (reactivos y no-reactivos) y en situaciones de congestión, simplemente serán descartados aquellos paquetes que impliquen una saturación de las memorias intermedias o *buffers* sin importar a qué tipo de flujo pertenecen (si el *router* sigue una estrategia no diferenciada) o independientemente si son los causantes o no de la congestión.

Una vez que un determinado *router* sufre congestión, la calidad del servicio de transporte se deteriora, aunque exista exceso de ancho de banda en todo el resto de la ruta. Por lo que, la dinámica del comportamiento de reenvío de los *routers* es determinante en la calidad del servicio ofrecido a nivel de red, lo que finalmente es

decisivo en la calidad percibida por los usuarios de aplicaciones multimedia.

La técnica tradicional para controlar el tamaño de las colas en *router*s (establecer un determinado tamaño máximo de la cola especificado en número de paquetes) denominada *drop tail* fue sustituida por la gestión activa de colas (AQM). El esquema AQM proporciona un mejor rendimiento en la red en términos de utilización de las líneas, la probabilidad de pérdidas de paquetes y del retardo extremo-a-extremo respecto a su predecesor. Evaluaciones realizadas a nivel de red [Parris *et al*., 1999] y evaluaciones de calidad subjetiva [Reguera *et al*., 2008] hacen evidente que el esquema AQM tiene un impacto significativo en la calidad de servicio de aplicaciones con requerimientos de tiempo real (VoIP).

La probabilidad de descarte del paquete en los esquemas AQM esta directamente relacionada con la carga del tráfico de entrada. Sin embargo, por lo general los esquemas AQM no proporcionan un control de congestión que considere los requerimientos del tráfico. Las políticas de colas comúnmente adoptadas en AQM para elegir el paquete a ser descartado son, selección directa del paquete recién llegado (denominada *Drop-Tail*), elegir alternativamente un paquete en forma aleatoria de entre los almacenados temporalmente (*Drop-Random*) [Hashem, 1989] o seleccionar el paquete más próximo a salir de la cola (*Drop-Front*) [Lakshmann *et al*, 1996; Yin and Hluchyj, 1993]. Todas ellas, ignoran los distintos tráficos que convergen en el *router*.

En relación a esto, hemos considerado necesario un esquema de control de congestión que proporcione un servicio conforme a los requerimientos característicos de los diferentes tipos de tráficos, tal como las aplicaciones de VoIP, que demandan retardos extremo-a-extremo de paquete limitados y tasas de pérdida acotadas.

1.3. Contribución del trabajo de tesis

En la investigación realizada, motivados principalmente en reforzar las medidas para el control de congestión que implique acentuar las prestaciones al tráfico multimedia en cuanto a los parámetros que definen la QoS (pérdidas de paquetes, utilización del enlace, retardo y *jitter*), un esquema de descarte de paquetes ha sido definido dentro de la estructura de un *router* AQM.

En particular, esta investigación aporta esencialmente una técnica de selección de víctima, a ser aplicado en el AQM, que permite suministrar distintos niveles de servicios a distintos tipos de tráfico (*real-time* VoIP, otros UDP y elástico TCP).

Se ha desarrollado el esquema de descarte selectivo (*Drop-Sel*) que sin necesitar la cooperación de un planificador, éste aporta beneficios en términos de equidad que no pueden proporcionar los tradicionales algoritmos (*Drop-Tail*, *Drop-Random* y *Drop-Front*) actualmente utilizados en el contexto de descarte de paquetes. Adicionalmente, cuando las cargas de tráfico lo permitan, incrementa la calidad del servicio para flujos multimedia, en particular a paquetes de VoIP.

Con este propósito en mente, aunado a *Drop-Sel* se desarrollan otros algoritmos referidos como *Drop-Sel-delay*, *Drop-Tail-delay*, *Drop-Sel+fair* y *Drop-Tail+fair* cada uno con sus propios beneficios que refuerzan las aportaciones propias del descarte selectivo.

1.4. Organización del documento

Durante esta Investigación se realizaron distintos tipos de estudios, los cuáles son desarrollados en los siguientes capítulos. En los capítulos 2, 3, y 4 se muestra una visión general de elementos, tecnologías y estudios que están directamente relacionados con el tema de investigación. Por otra parte, en los capítulos 5, 6 y 7 se muestran las aportaciones propias de esta tesis y las conclusiones del trabajo. Para más detalle, a continuación se hace un bosquejo general de la tesis.

El capítulo 2, *Control de Congestión en Internet*, se enfoca principalmente en el estudio de mecanismos para solventar los problemas de congestión. Entre ellos, se describen algunos mecanismos de control de congestión que han sido necesarios incorporarlos al protocolo TCP y, por otra parte, se describe el mecanismo de gestión activa de colas (AQM). Adicionalmente, se estudian otros protocolos (RTP, TFRC y DCCP) que permiten combinar la transferencia de flujos no-reactivos con funcionalidades de control de congestión.

En el capítulo 3, *Calidad de Servicio*, se analizan los elementos esenciales que determinan la calidad en Internet y se discuten algunos esquemas que se han implementado con el fin de conseguir la QoS necesaria según el usuario y el tipo de aplicación. Se describen algunos puntos en relación a la transmisión de paquetes de audio en redes IP. Básicamente, se trata de identificar las causas que provocan el deterioro de la calidad percibida por el usuario, de un extremo a otro de la red. Para finalizar el capítulo, se observan algunas herramientas que permiten evaluar la QoS recibida por usuarios en flujos de VoIP.

El capítulo 4, *Algoritmos de gestión activa de cola*, proporciona un resumen de mecanismos de AQM que se han propuesto con éxito con el fin de controlar la congestión. Simultáneamente, algunos de estos esquemas intentan maximizar la utilización del enlace, minimizar las pérdidas de paquetes y el impacto del *jitter* sobre la calidad de entrega de paquetes a su destino. Entre ellos, se estudia especialmente uno de los primeros y más relevantes esquemas para la prevención de la congestión en la arquitectura de Internet: *Random Early Detection* (RED) [Floyd and Jocobson, 1993]. En concreto, se ilustran los mecanismos sobre los que se basa fundamentalmente el trabajo de investigación realizado.

En el capítulo 5, *Descarte selectivo de paquetes*, esencialmente se expone el esquema general, características y beneficios de la nueva propuesta de selección de víctima denominada *Drop-Sel*. Este algoritmo pretende proporcionar un servicio equitativo tanto a UDP como al tráfico elástico TCP en una cola sencilla, sin necesidad de utilizar un planificador. Se sitúa al algoritmo dentro del conjunto de técnicas tradicionales de descarte y se compara con ellas basándose en los resultados de la experimentación obtenidos tras simulaciones.

El capítulo 6, *Alternativas de Drop-Sel y Drop-Tail*, se enfoca en el estudio de algunas alternativas que componen el descarte selectivo, principalmente las denominadas *Drop-Sel-delay*, *Drop-Tail-delay*, *Drop-Sel+fair* y *Drop-Tail+fair*. Adicionalmente, se proporciona un análisis de los efectos en la calidad de servicio para flujos de paquetes de audio, que dichas alternativas pueden producir en *Drop-Sel*.

En el capítulo 7 se muestran las conclusiones propias del trabajo de investigación. Y para finalizar la memoria de trabajo, se citan las principales referencias bibliográficas analizadas para el desarrollo de los diferentes estudios efectuados.

Capítulo 2

Control de Congestión en Internet

Uno de los problemas más discutidos y analizados en la literatura es el causado por la congestión en la red. La congestión es un problema omnipresente en todas las redes de paquetes en general y en Internet (basada en protocolo IP) en particular, relacionado con la disponibilidad limitada de recursos y se manifiesta en la aparición de retardos variables entre paquetes e incluso en pérdidas de los mismos. Se produce cuando la capacidad de las líneas se encuentra a su máximo nivel, con dificultades para satisfacer la demanda de ancho de banda de los usuarios, es decir, los *routers* no son capaces de procesar y transmitir todos los paquetes que reciben durante un tiempo determinado.

Este capítulo esta enfocado en el problema de la congestión en redes IP y en el estudio de la evolución de los mecanismos para solventarlo. Entre ellos, se describirán algunos métodos de control de congestión que han sido necesarios incorporarlos al protocolo TCP y, por otra parte, la adopción de mecanismos de gestión activa de colas en los *routers* participantes.

2.1. Antecedentes

2.1.1. Redes IP actuales

Hoy en día, las redes IP facilitan el acceso de cientos de millones de usuarios a múltiples

servicios de información, comercio y entretenimiento sin restricción alguna. Muchos de estos servicios llevan contenidos de audio y vídeo en tiempo real, para los que el usuario demanda determinadas calidades de servicio.

La introducción tiempo atrás de aplicaciones de audio y vídeo sobre IP, como por ejemplo *Vat* [Floyd *et al*.], generó un alto interés. Este interés se ha ido incrementando por la disponibilidad de nuevas herramientas que soporten estas aplicaciones, el aumento de ancho de banda y por iniciativas tales como MBONE (*Multicast backBONE*) [Thyagarajam *et al*., 1995] que ha facilitado trabajar con audio y vídeo en Internet. Actualmente, dentro de *The World Wide Web* (WWW) encontramos fácilmente *plug-ins* [Youtube, 2009] diseñados para transmitir *streaming* de vídeo y de audio entre PC. Además de lo anterior, por su gran popularidad, cabe mencionar aplicaciones como *RealAudio* [Mena and Heidemann, 2000] y *RealVídeo* [Realnetworks, 2008], así como servicios de transmisión multimedia incrustados en aplicaciones de mensajería instantánea como *MSN Messenger* [Microsoft, 2009], *Yahoo Messenger* [Yahoo, 2009], *ICQ* [ICQ, 2009], e incluso aplicaciones basadas en tecnologías *peer-to-peer* como por ejemplo *Skype* [Skype, 2009] y *VoIPstunt* [VoIPStunt, 2009] entre otras.

Por otra parte, cada día se busca crear nuevos equipos que aporten mejores servicios a los usuarios en cuestión de calidad de audio. Aunque muchos de éstos no sean a bajo coste, podemos encontrar por ejemplo el desarrollo de un equipo para conferencia multimedia denominada *Speak Conference Director* [Sipforum, 2006] basado en el protocolo IP.

A pesar de todo este progreso, Internet tiene todavía una serie de limitaciones que dificultan las aplicaciones interactivas o en tiempo real. La naturaleza *best effort* de IP hace que a veces se logre transmitir contenidos multimedia con buena calidad, y en otras, dependiendo de la dinámica de la red, se perciba con calidad inaceptable, causada por retardos, pérdidas de paquetes o lo que es peor, pérdidas consecutivas, denominadas en este contexto *ráfagas*. A medida que se van incrementando los servicios, el número de sitios a los que se puede tener acceso y los usuarios, Internet también incrementa sus problemas de inestabilidad y de rendimiento impredecible. Por tanto, el carácter *best effort* de IP no satisface las demandas de las aplicaciones multimedia, y en definitiva los requerimientos de los usuarios.

Internet esta integrada por pequeñas y grandes redes que se interconectan mediante el protocolo IP. Todas estas redes se encuentran conectadas mediante miles de enlaces denominados NAPs (*Network Access Points*), más recientemente denominados Puntos de Intercambio (*Exchange Points*) o también Intercambiadores de Internet (o IXs, *Internet*

Exchanges), todos ellos gestionados a través de acuerdos entre los correspondientes operadores, denominados contratos de *peering.*

Los protocolos de encaminamiento (o de intercambio de tablas de *routing*) en la red se clasifican en dos tipos: protocolos IGP (*Interior Gateway Protocol*) y protocolos EGP (*Exterior Gateway Protocol*) [Mills, 1984]. Los protocolos IGP hacen posible el intercambio de información de encaminamiento entre nodos y *routers* dentro de un sistema autónomo. La información generada es utilizada por el protocolo IP, u otros protocolos de red, para especificar cómo se llevará a cabo el encaminamiento de los datagramas. Existen algunos IGP comúnmente utilizados como son RIP (*Routing Information Protocol*) [Malkin, 1998], IGRP (*Interior Gateway Routing Protocol*) [Hedrick, 1991], y el OSPF (*Open Shortest Path First*) [Moy, 1998].

Mientras que el encaminamiento de un sistema autónomo es gestionado por un *Internal Gateway Protocol*, por otra parte, el protocolo EGP permite que distintos sistemas autónomos puedan intercambiar información de encaminamiento entre ellos. Para este fin, los más utilizados son: EGP (*Exterior Gateway Protocol)* y BGP *(Border Gateway Protocol*) que está sustituyendo progresivamente a EGP. El BGP versión 4 [Rekhter *et al*., 2006] es considerado actualmente en Internet como el estándar EGP. Está desarrollado para distribuir información entre los sistemas autónomos, indicando dónde se encuentran localizados y la ruta para acceder a diferentes prefijos de red o destinos finales IP.

Los protocolos interior y exterior tienen diferentes metas y por tanto, también requerimientos diferentes. La interacción entre los *Exchange Points* y la forma de trabajar de estos protocolos de encaminamiento IP puede variar como consecuencia de las diferentes tecnologías existentes para realizar la conexión entre *routers*.

El diseño de la arquitectura de Internet ha permitido la escalabilidad de la red y de los servicios proporcionados. Actualmente, Internet está constituida por múltiples *backbones*, los cuales interconectan un gran número de lugares distribuidos alrededor del mundo. Sin embargo, existen cuellos de botella, que son inherentes a esta arquitectura y que producen la congestión. La congestión y en definitiva, el rendimiento de la red, están afectados por problemas denominados como el problema de *backbone* y el problema de *peering* (Figura 1.1).

El problema de *Backbone* está generado como consecuencia de la falta de capacidad de las redes principales que forman la columna vertebral de Internet, cuando se solicita excesivamente información. Cada vez que se requiere información, ésta tiene que transmitirse a través de varias redes principales, cuya capacidad no ha aumentado en la

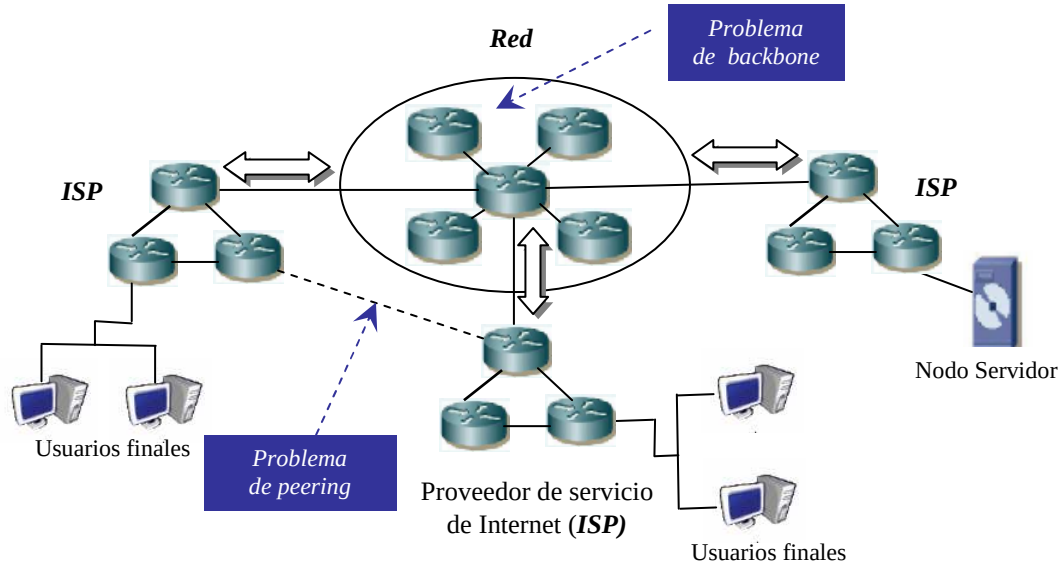


Figura 1.1. Cuellos de botella en Internet.

misma proporción que la generación de tráfico. La capacidad de la red se encuentra determinada esencialmente por dos factores: La capacidad del medio físico de transporte y de la capacidad de procesamiento y de conmutación de los *routers*.

Las tecnologías de conexión tienen diferentes características en función de la disponibilidad de ancho de banda y de lo predecible que sea su retardo o latencia. Los enlaces de redes de área local *Ethernet*, por ejemplo, pueden tener tasas de transmisión de 10Mbit/seg a 1Gbit/seg, pero su disponibilidad de ancho de banda dependerá del acceso de los *Exchange Points* establecidos en la red. Los enlaces de redes de área extendidas pueden variar de T1 (1.5Mbit/seg), E1 (2.048Mbits/seg) y E3 (34Mbits/seg) a OC-3/STM-1 (155Mbit/seg) o superiores basados en tecnología *GigabitEthernet*. Este tipo de enlaces suelen tener baja latencia y mínimo *jitter* pero pueden resultar en algunas ocasiones inflexibles y de alto costo. Otro de los tipos de enlaces que comúnmente se están utilizando son los enlaces virtuales, por ejemplo en MBONE.

Hoy en día, vemos que la implantación de fibra óptica ha mermado este problema, pero aún queda seguir mejorando las capacidades de conmutación y almacenamiento de los *routers*. Si bien son muchos los avances que se han generado en las arquitecturas de conmutación de los *routers*, éstos no han ido en proporción a los volúmenes del tráfico que se generan como resultado del aumento continuo de usuarios y de las nuevas

aplicaciones en tiempo real.

Las distintas redes que conforman Internet están operando entre ellas, como se ha especificado previamente, bajo ciertos acuerdos denominados de *peering.* Estos acuerdos consisten en que se puede transmitir información libremente entre varias redes distintas para reducir costos de servicio. Actualmente Internet opera con una mayor cantidad de puntos *peering* con respecto al pasado, mejorando con ello el funcionamiento y rutas de acceso para la mayoría del tráfico. La gran mayoría de estos operadores de *Exchange Points* ofrecen tecnología *Ethernet* como *Gigabit Ethernet* que viene a ser una selección predominante para reducir costos e incrementar la capacidad ofrecida. Pero aún así, el problema de *peering* se manifiesta como consecuencia de que muchos de los *Exchange Points* pueden tener cientos de miles de usuarios, lo que puede provocar puntualmente un momento dado problemas de congestión.

Los cuellos de botella en Internet siguen existiendo a pesar de las mejoras continuas que se realizan en cuestión de tecnología y del aumento de ancho de banda. La pérdida de paquetes y retardos provocados por la congestión siguen degradando la calidad que el paquete de audio y de vídeo demanda. Por lo que es de suma relevancia continuar buscando alternativas para aminorar la congestión y/o alternativas que disipen los problemas que desencadena.

2.1.2. Protocolos de transporte TCP y UDP

En la arquitectura de Internet sobre el protocolo IP, entre las aplicaciones y el nivel de red se prevén dos protocolos de transporte: TCP (*Transmission Control Protocol*) [Postel, 1981b] y UDP (*User Datagram Protocol*) [Postel, 1980], ambos operando extremo-a-extremo, es decir, involucrando a sistemas finales.

TCP es el protocolo de transporte predominante en Internet que acarrea entre 60% y 90% del total del volumen de tráfico [Fomenkov *et al*., 2004]. Este protocolo orientado a conexión es utilizado por un rango de diferentes aplicaciones como son HTTP/HTTPS (WWW), FTP (transferencia de archivos), SMTP/POP3/IMAP (correo electrónico) y en ocasiones, es además utilizado para transmitir *streaming* de audio y de vídeo (p.e. [Youtube, 2009]).

Una serie de mecanismos claves determinan la fiabilidad y robustez del protocolo TCP. Los segmentos TCP son transportados en paquetes IP y con el fin de asegurar la entrega de cada segmento, cada paquete contiene 16 bits de datos de información (*checksum*) del contenido de la cabecera y del segmento TCP (Figura 2.1), con los cuales

Cabecera de TCP

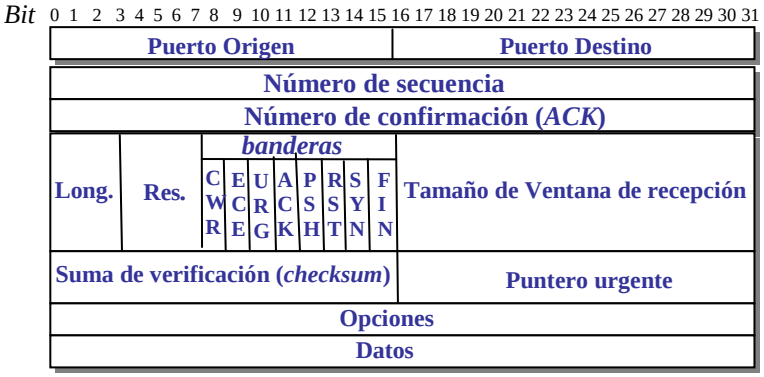


Figura 2.1. Formato de la cabecera de TCP.

el receptor puede verificar si el dato recibido es correcto o no. Aunado a esto, los segmentos pueden perderse como consecuencia de errores (fallas en pruebas de *checksum*) o por congestión en la red. Si el paquete es recibido con éxito el receptor envía un paquete de confirmación (segmentos con el *bit de la bandera ACK activado*) de regreso a la fuente. En caso de que el dato recibido sea defectuoso, será descartado por el receptor. Para compensar errores, paquetes perdidos o retrasos, la fuente retransmite cada paquete que no ha sido confirmado después de un tiempo apropiado (por ser considerado un *time-out*, expiración de los temporizadores arbitrados).

Alternativamente a TCP, otro protocolo de transporte utilizado en Internet es UDP. Típicamente, el tráfico UDP es entre 10% y 40% del total del volumen de tráfico [Fomenkov *et al*., 2004]. Éste está especialmente indicado para las aplicaciones multimedia (audio y vídeo), las cuales prefieren pocos retardos (es decir, no incluir mecanismos de realimentación para la recuperación de errores), aún asumiendo que la capa de transporte no proporcione fiabilidad alguna. Es decir, UDP no ofrece control de congestión ni control de errores, lo cual puede ser indicado para aplicaciones sencillas que no pueden implementar toda la complejidad que implica el protocolo TCP. Al ser enviados de esta manera, los segmentos UDP no tienen garantía alguna de fiabilidad en la entrega, por lo que no van protegidos frente a las pérdidas en la red durante congestión. Esto se puede traducir en una posible degradación de la calidad ya sea de audio, de vídeo o ambos inclusive, hasta incluso hacerlos ininteligibles. Y lo que es más, una demanda excesiva no sólo afectará a los propios flujos multimedia, sino que definitivamente deteriorará igualmente a otros flujos de tráfico (TCP).

2.2. Control de congestión en TCP

A diferencia de UDP, el protocolo TCP garantiza la transmisión fiable de los paquetes. Para ello, además de mecanismos que proporcionan control de errores, incorpora mecanismos para el control de congestión con el propósito de lograr que TCP reaccione rápidamente en condiciones de alta congestión (sin menoscabar con ello el rendimiento obtenido).

Las primeras especificaciones de TCP en RFC 793 [Postel, 1981b] no incluyen ningún ajuste dinámico del tamaño de la ventana de congestión (control en respuesta a la congestión), sólo fue concebido como un protocolo capaz de proporcionar una transmisión de datos fiable entre dos puntos de una red de comunicación. Ante la eventualidad de una serie de colapsos de congestión sufridos en Internet, TCP ha sido mejorado a lo largo de los años, en paralelo a la cada vez más alta capacidad de las redes. Entre las primeras propuestas para aplicar control de la congestión en Internet se encuentran las desarrolladas por *Van Jacobson* [Jacobson, 1988]. Estas sugieren incorporar al protocolo TCP algunos mecanismos de control de la carga generada en la red, de forma tal que las fuentes ajusten sus tasas de transferencia al nivel de congestión existente y así provocar que se adapten a la dinámica de la red.

Si una fuente de información TCP detecta que en la transmisión de sus paquetes empieza a haber problemas (es decir, detecta pérdidas mediante la expiración de los temporizadores arbitrados para la recepción de las correspondientes confirmaciones), asume que dichas pérdidas son siempre debidas a problemas de congestión. Como medida reactiva, en ese caso, TCP reduce drásticamente la tasa de transferencia con la esperanza de reducir el número de paquetes perdidos o descartados. Resultado de este trabajo es la implementación denominada TCP *Tahoe* que combina los así denominados algoritmos de *arranque lento*, *prevención de la congestión* y *retransmisión rápida*. Los algoritmos de *Jacobson* (denominados también mecanismos de recuperación de la congestión) constituyen ahora la base en todas las implementaciones modernas de TCP. A TCP *Tahoe* le fue añadido el algoritmo denominado de *recuperación rápida*, esta inclusión forma la desarrollada implementación de TCP *Reno* [Stevens, 1997; Allman, 1999], versión de TCP dominante en la transmisión por Internet [Ho *et al*., 2008].

2.2.1. Arranque lento y prevención de la congestión

Arranque lento (*slow-start*) y *prevención de la congestión* (*congestion avoidance*) son dos de los algoritmos incorporados a TCP utilizados para controlar la tasa de

transferencia de paquetes generados, monitorizando permanentemente las prestaciones de la red y determinando la capacidad disponible, para así evitar la congestión incluso cuando se generen grandes ráfagas de paquetes. Estos mecanismos de control tienen objetivos diferentes pero son implementados en conjunto. Para su operación requieren la definición de dos variables por cada conexión TCP: la ventana de congestión *cwnd*, que determina la cantidad de *bytes* que el remitente puede transmitir sin necesidad de ser confirmados, y la ventana de recepción *rwnd* que especifica el número de *bytes* que el receptor tiene capacidad para recibir.

Para una conexión dada, la cantidad máxima a transmitir estará acotada por el mínimo de valor de *cwnd* y de *rwnd*. Al establecerse la conexión TCP, el transmisor asigna a *cwnd* el tamaño máximo de segmento para la conexión, y por lo tanto envía un paquete con ese tamaño de *payload*. Al mismo tiempo, mantiene un temporizador de retransmisión que se activa al enviar cada segmento.

El algoritmo *Arranque lento* es implementado de la siguiente forma: un segmento es enviado y si un ACK es recibido (antes de la expiración del temporizador), el valor de *cwnd* es incrementado en un segmento. Por lo que ahora, dos segmentos pueden ser enviados y causar dos ACKs (Figura 2.2a). Por cada uno de esos ACKs, *cwnd* es incrementada en un segmento de ahí que ahora permite que cuatro segmentos sean enviados. Este proceso de crecimiento exponencial de la ventana de congestión permanece con ese ritmo de crecimiento hasta que ocurre una de las dos posibles alternativas, se alcance el valor acotado de *rwnd* o se alcance un umbral establecido de antemano denominado *ssthresh.*

Cuando la ventana alcanza el valor *ssthresh*, TCP pasa a la fase de *prevención de la congestión* (Figura 2.2b) dónde en lugar de incrementar *cwnd* en un segmento adicional por cada *ACK* recibido, ahora la ventana será incrementada en un segmento por cada RTT. Se está en fase de *prevención de la congestión* hasta que se detecta la pérdida de paquetes (se asume por expiración del temporizador) y es entonces cuándo la fuente TCP regresa a la fase de *arranque lento*, estableciendo *cwnd* a un segmento y *ssthresh* a la mitad del tamaño actual de la ventana. En definitiva, lo que se pretende con el arranque lento y la prevención de la congestión es incrementar la tasa de generación de tráfico, mientras no haya problemas y la capacidad de salida lo permitan.

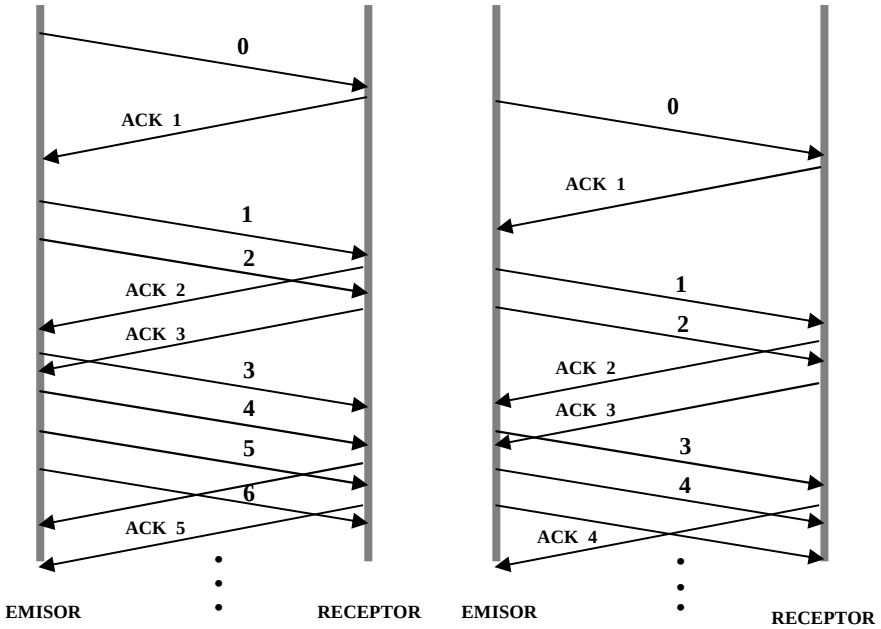


Figura 2.2. Algoritmos de control Arranque lento (a) y Prevención de la congestión (b).

2.2.2. Retransmisión rápida/ recuperación rápida

Además de los anteriores, en TCP se prevén los algoritmos denominados *retransmisión rápida* (*fast retransmit*) y *recuperación rápida* (*fast recovery*), utilizados para detectar y reparar paquetes perdidos. TCP adopta estos algoritmos de control cuando se reciben consecutivamente tres confirmaciones duplicadas (segmentos con el *bit ACK activado*), que pueden ser provocadas por diversos problemas en la red como la congestión (que causa descarte de segmentos en *routers* intermedios).

Tras esta eventualidad, TCP infiere que ha habido una pérdida, por lo que reacciona (entra en fase de *retransmisión rápida*) transmitiendo nuevamente el que parece ser el segmento perdido, sin esperar a que el correspondiente temporizador de retransmisión expire. Tras realizar el procedimiento *retransmisión rápida*, la entidad TCP emisora entra en la fase denominada *recuperación rápida* que es ahora quien controla la transmisión de nuevos segmentos hasta que llega un *ACK* que no sea duplicado. Durante este intervalo por cada paquete *ACK* duplicado (*DupAck*) que llegue al emisor, la ventana *cwnd* es incrementada en un segmento. En el instante en que TCP detecta la llegada de un nuevo

ACK, asume que la transferencia de paquetes ya es normal, y establece *cwnd* al umbral establecido al inicio de la transmisión. En caso de que nunca llegue un nuevo *ACK*, el emisor permanecerá en la fase de *recuperación rápida* hasta que ocurra la expiración de un temporizador. Estos algoritmos permiten la recuperación rápida de paquetes que se han perdido esporádicamente, sin embargo suele haber problemas para recuperar cuando múltiples paquetes son descartados.

2.2.3. Alternativas en TCP

Al retransmitir solamente un segmento por RTT en respuesta a tres *DupAcks*, los algoritmos de *retransmisión rápida* y *recuperación rápida* de TCP *Reno* han mostrado problemas cuando numerosos segmentos son descartados en una única ventana de datos. Siguiendo el trabajo de *Jacobson*, el modelo TCP ha sido utilizado en numerosos estudios realizados por otros investigadores, especialmente en nuevas implementaciones de TCP (que mejoran TCP *Tahoe* y TCP *Reno*), cada una con sus ventajas e inconvenientes, que difieren en la forma de llevar a cabo la estimación y las reglas respecto a cómo adaptar el tamaño de la ventana *cwnd*.

Algunos de estos protocolos consideran que la red esta congestionada basándose (al igual que TCP *Reno* y TCP *Tahoe*) en la pérdida de paquetes, por ejemplo, TCP *SACK* [Mathis *et al*., 1996] y TCP *New Reno* [Floyd and Henderson, 1999]. Mientras que otras implementaciones se basan en el retardo como es el caso de TCP *Vegas* [Brakmo and Peterson, 1995] y *FAST* TCP [Jin *et al*., 2004]. Estas últimas se basan en la hipótesis de que la medida del retardo experimentada está en función de la carga en la red. Con el objetivo de desarrollar un mecanismo de control de la congestión reactivo tanto a la pérdida de paquetes como al retardo en la cola, existen otros protocolos que adoptan ambas estrategias. Ejemplos de este tipo de implementaciones son TCP *Africa* [King *et al*., 2005] y TCP *Illinois* [Liu *et al*., 2006].

Para fijar el contexto de la investigación realizada, y sin ánimo de exhaustividad se resumen algunas de las implementaciones más representativas de TCP.

TCP Vegas

Para incrementar el rendimiento y reducir las pérdidas en la red, TCP *Vegas* [Brakmo and Peterson, 1995] extiende el mecanismo de retransmisión establecido en TCP *Reno*. Esta técnica ejecuta una más rápida detección de pérdidas y recuperación en el caso de múltiples descartes. Por lo que, en cada paquete que envía, TCP *Vegas* establece el

timestamp para calcular su RTT. De ahí que utiliza una constante estimación de RTT para decidir la retransmisión en los dos sentidos.

Primero, cuando un *ACK* duplicado sea recibido, observa si el intervalo de tiempo desde que el paquete fue enviado es superior al valor del período *timeout*. Si es así, TCP *Vegas* retransmite el segmento sin esperar que lleguen tres *ACKs* duplicados. Segundo, si se recibe un *ACK* que no es duplicado y además es el primero o segundo posteriormente a la retransmisión, nuevamente se calcula si el intervalo de tiempo desde que el paquete fue enviado es superior al valor del período *timeout*. Si es así, se considera que otros segmentos pueden haberse perdido previos a la retransmisión y se retransmite nuevamente otro segmento. TCP *Vegas* solamente reduce *cwnd* si la retransmisión del segmento fue después de la última reducción. Ninguna pérdida que suceda antes de la reducción de la ventana implica que la red esté congestionada.

Por otra parte, con el fin de ser capaz de detectar y evitar la congestión durante la fase de *arranque lento*, la implementación TCP *Vegas* cambia el modo de *arranque lento* a un modo lineal de incremento/reducción.

TCP SACK

La implementación referida como *reconocimiento selectivo* (*SACK*) [Mathis *et al*., 1996] es otra de las estrategias que intentan dar respuesta al problema que se presenta con múltiples segmentos descartados. Con *ACK* Selectivo, el receptor le informa a la fuente mediante confirmaciones selectivas que segmentos han sido recibidos satisfactoriamente. De ahí que las fuentes TCP tienen la información justa para tomar una decisión inteligente acerca de que paquetes deben retransmitirse y cuales no deben retransmitirse durante la fase de *recuperación rápida*. Siendo no necesario esperar todo el RTT para identificar que debe retransmitirse otro segmento después de entrar en las fases de *retransmisión rápida* y de *recuperación rápida*. TCP *SACK* es en principio similar a su predecesor TCP *Reno*, sólo difiere en la forma de responder al evento de múltiples paquetes perdidos. TCP *SACK* modifica la fase de *retransmisión rápida* y *recuperación rápida* para que pueda abordar múltiples pérdidas en una única ventana.

Para su operación, establece una lista de algunos de los bloques de secuencias contiguas ocupadas por datos que han sido recibidos. Cuando se recibe un *ACK* conteniendo la opción *SACK*, la fuente debe recordar los *ACK* selectivos para futuras referencias. La fuente deberá retransmitir, en orden secuencial, los segmentos que han sido transmitidos pero no aún confirmados.

Existen alternativas desarrolladas para reforzar aspectos específicos de esta implementación. Con el fin de hacer frente a paquetes duplicados en la implementación de TCP *SACK*, se ha desarrollado TCP *D-SACK* (duplicado-*SACK*) [Floyd *et al*., 2000]. Esta implementación sólo incorpora un cambio en el contenido de la opción *SACK* cuando un *ACK* es enviado. El bloque de *D-SACK* es utilizado por lo tanto, para informar sobre secuencias duplicadas de datos contiguos recibidos. Si el segmento duplicado es parte de un gran bloque de datos no contiguos en la cola del receptor, entonces los siguientes bloques deben ser utilizados para especificar este largo bloque.

Por otro lado, [Blanton *et al*. 2003], presenta un conservador algoritmo de recuperación de pérdidas que remplaza el algoritmo de *recuperación rápida* establecido por [Allman, 1999]. Este algoritmo permite que *SACK* trabaje satisfactoriamente bajo grandes cambios de pérdidas de varios segmentos para una ventana sencilla a consecuencia de congestión. Basándose también en TCP *SACK*, en [Kim *et al*. 2004] se establece una simple modificación para detectar las pérdidas de retransmisión y evitar los siguientes RTO (*retransmission timeout*), desarrollando con ello la implementación TCP *SACK*+. Una de las características de esta implementación es que si la congestión es demasiado alta, TCP *SACK*+ no transmite ningún paquete adicional al detectar una perdida de retransmisión por considerar que también puede ser perdido en esa situación de congestión.

TCP New Reno

Para dar respuesta al problema de múltiples descartes y responder a *ACKs* parciales o incompletos, un sistema mejorado del algoritmo de *recuperación rápida* (establecido en TCP *Reno*) es incorporado en TCP *New Reno* [Floyd and Henderson, 1999]. En esta modificación se introduce una variable denominada *recover* utilizada para distinguir un *ACK* completo o un *ACK* incompleto. Cuando llega un nuevo *ACK* a la fuente (estando en la fase de *recuperación rápida*) se analiza, y si es considerado un *ACK* incompleto, ésta asume que otro segmento de la misma ventana se ha perdido y por consiguiente lo retransmite. Para el primer *ACK* incompleto que recibe, sin salir de la fase de *recuperación rápida*, reestablece el temporizador. En el instante en que TCP detecta la llegada de un nuevo *ACK*, asume que la transferencia de paquetes ya es normal, y establece *cwnd* al umbral *ssthresh* (en lugar del determinado al inicio de la transmisión), pasando a la fase de prevención de la congestión.

Para evitar innecesarias retrasmisiones rápidas múltiples que pueden ocurrir después de un *timeout*, el procedimiento de TCP *New Reno* fue adaptado por [Floyd *et al*. 2004],

para mejorar el uso de la variable *recover.*

Scalable TCP

Con el objetivo de ofrecer un mecanismo robusto que mejore el funcionamiento de redes de alta velocidad, se propuso el *Scalable* TCP (STCP) [Kelly, 2003]. STCP, basado en pérdidas, presenta un sencillo cambio al algoritmo de adaptación de la ventana respecto al tradicional TCP.

Por cada *ACK* recibido en un RTT (cuando la congestión aún no ha sido detectada), la ventana se incrementará de forma que $cwnd \leftarrow cwnd + 0.01$. Sin embargo, si la primera detección de congestión se produce en un RTT, la ventana se reduce: $cwnd \leftarrow cwnd - [0.125 * cwnd]$. Este algoritmo tiende a ser más agresivo en el incremento y menos agresivo en la reducción. De esta manera, el algoritmo de adaptación de la ventana es escalable. Con ello, logra tener impacto sobre el tráfico, las propiedades de asignación de ancho de banda, la varianza en la tasa de flujos y el control de estabilidad.

FAST TCP

Otra versión de protocolo de congestión diseñada para alta velocidad es establecida en *Fast* TCP [Jin *et al*., 2004]. Este protocolo también intenta mejorar el pobre rendimiento de TCP *Reno* en redes con grandes anchos de banda y largos retardos. Para lo cual, *Fast* TCP es un algoritmo que utiliza el retardo experimentado como medida de congestión. Esta implementación se basa en que el retardo de la cola puede ser más acertadamente estimado que la probabilidad de perdidas en redes con una larga relación ancho de banda-retardo y que las pérdidas proporcionan menos información que las medidas del retardo en la cola. Asimismo, *Fast* TCP asume que la dinámica del retardo en la cola tiende a comportarse en relación a la capacidad de la red. Por consiguiente, utilizar la medida del retardo en cola le permite a *Fast* TCP que se eliminen las oscilaciones de nivel de paquete, estabilizar la ventana cerca del punto donde el buffer es grande y el retardo pequeño, y además, estabilizar la dinámica de los flujos.

La dinámica de la ventana de congestión de *Fast* TCP toma la misma forma que TCP *Reno*. Sin embargo, difiere en la manera de elegir la función de ganancia, la función de utilidad marginal y la medida de congestión extremo-a-extremo.

TCP AFRICA

Motivados por el bajo rendimiento de TCP en redes de alta velocidad, se desarrolló TCP *África* [King *et al*., 2005]. Este protocolo puede mantener un apropiado comportamiento escalable al observar el efecto de eventos periódicos de congestión en lugar de la probabilidad de pérdidas, como así se hace en TCP *Reno.*

TCP *África* es un protocolo hibrido que predice la congestión considerando las medidas del RTT del paquete (información del retardo). En ausencia de congestión, bajo condiciones favorables de retardo, entra en el denominado modo rápido que utiliza una regla agresiva de incremento de ventana escalable, lo que permite utilizar el ancho de banda disponible. En presencia de congestión, entra en el denominado modo lento que utiliza la regla más conservadora de *TCP Reno*, esto es, un incremento lineal de un paquete adicional por RTT.

Por consiguiente, TCP *África* es rápido al utilizar el ancho de banda disponible, y lento al detectar la posible congestión.

TCP Illinois

Otro algoritmo de control de congestión para redes de alta velocidad, propuesto en los últimos años es TCP *Illinois* [Liu *et al*., 2006]. Este protocolo adopta una estrategia AIMD (incremento aditivo – reducción multiplicativa) denominada cóncavo-AIMD o C-AIMD. TCP *Illinois* se basa en que la curva de la ventana ideal debe ser cóncava. Por lo que, combina la pérdida y el retardo de paquetes para lograr un mejor rendimiento que el TCP estándar. TCP *Illinois* utiliza la información de paquetes perdidos como principal señal de congestión, y con ello, determina si el tamaño de *cwnd* debe ser incrementado o reducido. Además, con el fin de ajustar los cambios de incremento o reducción de la ventana, observa la información del retardo en la cola como una segunda señal.

TCP *Illinois* conserva las fases de *recuperación rápida* y *retrasmisión rápida* de TCP *New Reno* estándar. Y si los receptores permiten *ACK* selectivo, este algoritmo es capaz de adoptar características de SACK TCP. Otra de las principales características de TCP *Illinois* es que suele ser más robusto contra fluctuaciones del retardo provocadas por el ruido o por el arribo de ráfagas de paquetes.

2.3. Gestión Activa de colas

Los mecanismos de control de congestión incorporados a TCP funcionan en los extremos finales y pueden llegar a ser insuficientes para proporcionar un buen servicio en todo tipo de circunstancias. De forma complementaria, en los puntos intermedios de la red, o *routers*, pueden ser incorporados mecanismos de prevención de la congestión. Estos mecanismos introducidos en el interior de la red son comúnmente llamados gestión activa de colas (*Active Queue Management*, AQM) [Braden *et al*., 1998]. Si bien los mecanismos incorporados a TCP tratan de restaurar el estado de operación cuando la demanda ya ha excedido la capacidad, los mecanismos de control denominados de prevención de la congestión tratan naturalmente de evitarla permaneciendo la demanda de la red cerca del punto máximo, pero evitando que se produzca la congestión.

2.3.1. Fundamentos de la Gestión Activa de Colas

La gestión de colas es un procedimiento para compartir la línea de salida de forma controlada, distribuyendo el tráfico en distintas colas en función de los requisitos asociados. Tradicionalmente, la técnica de gestión de colas más habitual utilizada en los *routers* de Internet se denomina *drop tail* (dado que los últimos paquetes encolados son descartados). En los *routers* que adoptan esta técnica, los paquetes que van llegando son temporalmente aceptados y almacenados hasta que un determinado tamaño máximo de la cola, especificado en número de paquetes, sea alcanzado. Cuando la cola se llena, es decir en condiciones de congestión, los siguientes paquetes que llegan serán descartados mientras que la ocupación de la cola no decrezca y sea capaz de almacenar a otros. El *drop tail* si así lo permite el tráfico de llegada, tiende a mantener la cola al máximo nivel de ocupación durante largos períodos de tiempo, ya que el descarte se inicia cuando la cola esté llena, no obstante, como desventaja puede que el descarte se inicie demasiado tarde, implicando una merma significativa en las prestaciones.

Con el fin de mitigar estas limitaciones de funcionamiento de TCP (colas saturadas) sobre redes *drop tail*, se desarrolla la gestión activa de colas. En AQM, cuando un *router* detecta una incipiente congestión, los paquetes son descartados (a pesar que la cola no este completamente a su capacidad) para notificar a las fuentes que deben reducir la carga en la red y con ello poder controlar la velocidad de llegada de los paquetes. Si el paquete es descartado tempranamente, las colas se incrementarán significativamente menos (conserva una baja ocupación media en los *routers*) evitando colas sobresaturadas, de forma tal que el AQM es capaz de soportar mayores ráfagas de tráfico. Este beneficio de

reducir el número de paquetes descartados no es posible lograrlo si se utiliza la técnica de *drop tail* que mantiene casi al máximo la capacidad de la cola, y no mantiene espacio suficiente para albergar rápidos crecimientos o ráfagas. A pesar que los mecanismos implementados en TCP restringen la tasa de envío, no son capaces de recuperar tan fácilmente los paquetes descartados cuando ocurre una ráfaga, cosa que es menos costosa cuando se trata de recuperar un sólo paquete descartado. El problema de congestión puede estar agravado aún más por la presencia en las redes de flujos que no sean lo suficientemente reactivos a la congestión. Es más, desde que se detecta la congestión hasta que se reacciona, puede transcurrir un intervalo no despreciable de tiempo que se traduce en pérdidas consecutivas o ráfagas, tanto del flujo TCP involucrado como para los otros flujos no reactivos.

Por otra parte, operar con una baja ocupación en los *routers* reduce el retardo de flujo extremo-a-extremo, dado que cuando la ocupación media de las colas de salida tienda a cero, el retardo introducido tenderá a cero. De esta forma se proporcionará un menor retardo en servicios interactivos, lo cual es especialmente relevante para el tráfico sensible al retardo como es el caso de VoIP (un valor superior del retardo considerado puede degradar la calidad del servicio que se está proporcionando).

Otra de las características de gestión activa de colas es conseguir establecer políticas de disciplinas de colas. Por lo general, el AQM contempla tres posibles alternativas para decidir el paquete a ser elegido como víctima de descarte cuando la cola está llena: la selección directa del paquete recién llegado (denominada *Drop-Tail*), elegir alternativamente un paquete en forma aleatoria de entre los almacenados temporalmente en la cola congestionada (denominada *Drop-Random*) o seleccionar el paquete más próximo a salir de la cola (denominada *Drop-Front).*

La idea esencial de la implementación del AQM consiste en reemplazar la técnica tradicional de gestión de colas *drop tail* en favor de proporcionar un mejor rendimiento en la red en términos de utilización de las líneas, probabilidad de pérdidas de paquetes y del retardo extremo-a-extremo. Ejemplos de este tipo de aproximación (propuestos para complementar los mecanismos de control de congestión incorporados a TCP) son los mecanismos RED [Floyd and Jacobson, 1993] y ARED (*Adaptive RED*) [Floyd *et al*., 2001] los cuales serán abordados en detalle en el capítulo 4, al igual que otros esquemas AQM que han sido recientemente introducidos.

2.3.2. Notificación Explícita de la Congestión (ECN)

Para el control de congestión, junto con AQM se incorpora la así denominada Notificación Explicita de la Congestión (ECN, *Explicit Congestion Notification*) [Floyd, 1994; Ramakrishnan and Floyd, 1999]. Implementando AQM en los *routers*, los paquetes TCP pueden ser marcados (en lugar de descartados) si se sufre congestión utilizando un *bit* ECN en la cabecera IP (1 es igual a congestión) de cada uno de ellos y ser enviados a los nodos. Con esta alternativa, se está notificando a los siguientes *routers* que ese paquete ha sufrido congestión. El número y selección de paquetes que son marcados durante una congestión dependen de las políticas AQM que se establezcan.

Con la implementación de ECN, es de esperar que el número de paquetes descartados en la red se reduzca y en consecuencia también el número de paquetes retransmitidos. Sin embargo, experimentos realizados por [Long Le *et al*. 2005] han demostrado que en algunos servicios en Internet no siempre se proporciona un mejor rendimiento al utilizar ECN en ciertos AQM. AQM se desarrolló para trabajar con mecanismos de notificación explícita, pero aún así, dependiendo del escenario, es posible que opere sin realizar ECN. No todos los protocolos de transporte (como UDP) reaccionan ante eventos ECN, en este caso, AQM utiliza el mecanismo de descarte de paquetes (a modo de notificación implícita) para indicar que se ha detectado y sufrido congestión.

2.4. Otros Protocolos en Internet

El protocolo de control de transmisión (TCP) ha servido como procedimiento para el control de la congestión en Internet desde su concepción y despliegue inicial. Sin embargo, las demandas de Internet en tamaño, capacidad y heterogeneidad lo han llevado a rebasar sus límites. Cada vez es más notable la presencia de aplicaciones que utilizan UDP, incluyendo *streaming* de audio y vídeo, telefonía por Internet y juegos en línea, las cuales son más sensibles a los retardos que a la fiabilidad en las entregas. Este crecimiento de tráfico sin control de congestión ha inspirado un intenso esfuerzo en investigación y desarrollo de nuevos protocolos que fluctúen menos que TCP. En particular, protocolos que permitan combinar la transferencia de flujos no reactivos con funcionalidades de control de congestión.

2.4.1. RTP

El protocolo RTP, *Real Time Transport Protocol* [Schulzrinne *et al.*, 2003], es esencialmente un mecanismo de señalización de los denominados *in-band*, ya que incorpora señal de control a través del mismo canal que se usa para el flujo de datos. Es un mecanismo genérico para soportar la integración de audio, video y datos. RTP permite que el receptor pueda tener control de los paquetes que está recibiendo colocando datos pertinentes en la cabecera, por ejemplo, el número de secuencia y la marca de tiempo (*timestamp*) del paquete, información necesaria para reagrupar los *streams* de tiempo real. El formato de la cabecera de RTP se muestra en la Figura 2.3.

Cabecera de RTP

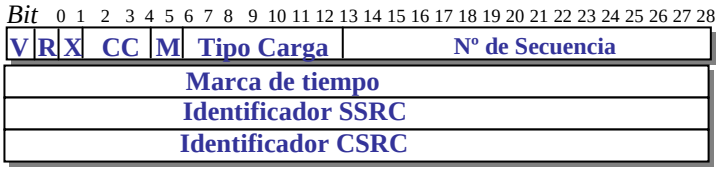


Figura 2.3. Formato de la cabecera de RTP.

Donde,

- versión de RTP (**V)**: 2 bits
- *bytes* de relleno **(R)**:1 bit
- extensión de cabecera **(X)**:1 bit
- número de fuentes que contribuyen (**CC)**: 4 bits
- eventos significativos **(M)**: 1 bit
- formato de la carga útil (**Tipo de carga)**: 7 bits
- **(Nº de Secuencia)**: 16 bits. Se incrementa cada vez que se envía un paquete
- instante de envío del paquete **(Marca de tiempo)**: 32 bits. Esta marca de tiempo se utiliza para medir el *jitter*.
- fuente de sincronización (**Identificador SSRC,** *synchronization source***)**: 32 bits
- fuente que ha contribuido a la carga útil (**Identificador CSRC,** *contributing source***)**: 32 bits.

Con RTP, el receptor puede sincronizar fuentes así como detectar si recibe un paquete fuera de orden o si alguno se ha perdido. RTP no proporciona garantías de QoS.

El protocolo RTCP (*Real-Time Transport Control Protocol*) [Schulzrinne *et al.*, 2003] es un protocolo asociado a RTP para monitorizar la QoS observada por los participantes. Su principal función consiste en intercambiar mensajes de información de estado entre los extremos de la sesión para informar sobre la calidad recibida. De tal manera que el receptor puede usar RTCP como un mecanismo de retroalimentación para notificarle al emisor acerca de la calidad de la sesión que se está realizando. Además también se incluyen mensajes RTCP desde las fuentes para sincronizar relojes e informar sobre el número de paquetes transmitidos.

2.4.3. TFRC

Control de tasa TCP-*Friendly* (TFRC, *TCP Friendly Rate Control*) [Handley *et al*., 2003] es un mecanismo de control de congestión que puede ser adoptado por un protocolo de trasporte como RTP, a nivel de una aplicación para incorporar control de congestión extremo-a-extremo o en el contexto de gestor de congestión en puntos finales. TFRC tiene una más baja variación de rendimiento sobre el tiempo con respecto a TCP, permitiendo que responda más despacio a los cambios de ancho de banda disponibles.

TFRC se ha diseñado especialmente para aplicaciones con determinado tamaño de paquete donde su tasa de envío varía en paquetes por segundo en respuesta a la congestión. Por ende, este protocolo es más adecuado para aplicaciones tales como telefonía o media *streaming*, donde una tasa de envío relativamente más suave es importante. Este mecanismo, basado en el receptor, utiliza una ecuación de rendimiento que proporciona una tasa de envío en función de la tasa de pérdidas y el RTT estimado. Específicamente, el receptor mide la tasa de pérdidas y proporciona esta información a la fuente. Con estos datos retroalimentados, la fuente mide el RTT. A continuación, la tasa de perdidas y el RTT son considerados en la ecuación de rendimiento, generando entonces con ello la tasa de transmisión a la que debe ajustarse la fuente.

Como TFRC es utilizado junto con un protocolo de transporte, el formato del paquete dependerá del protocolo considerado. Sin embargo, cada paquete enviado por la fuente debe contener: un número de secuencia, una marca de tiempo indicando cuando el paquete ha sido enviado y la última estimación de RTT realizada. Por otra parte, cada paquete retroalimentado por el receptor debe incluir: la marca de tiempo del último paquete recibido, el tiempo transcurrido desde que fue recibido hasta que generó la retroalimentación, y finalmente, la tasa a la cual el receptor estima que el dato ha sido recibido, considerando desde el último paquete de retroalimentación enviado.

2.4.4. DCCP

Un protocolo de transporte más reciente que incorpora control de congestión extremo-a-extremo ha sido desarrollado. Este protocolo, denominado Protocolo de control de congestión de paquete de datos (DCCP, *Datagram Congestion Control Protocol*) [Kohler *et al*., 2006], proporciona control de congestión para flujos de datos no reactivos, evitando los retardos asociados con TCP. De forma tal, que está diseñado para utilizarse en muchas de las aplicaciones que actualmente utilizan UDP, tales como *streaming* de audio y vídeo, o juegos *on-line*.

DCCP permite a las aplicaciones seleccionar una alternativa de un grupo de mecanismos de control de congestión. El primero de ellos, Control de Congestión TCP-*Like* [Floyd and Kohler, 2006a], reduce a la mitad la ventana de congestión en respuesta al descarte de paquetes, de forma similar a como se lleva a cabo en TCP. Una aplicación utilizando este mecanismo de control responderá rápidamente a los cambios de disponibilidad de ancho de banda. Este mecanismo en contraste con TCP, podrá tolerar fuertes cambios en la ventana de congestión. Un segundo mecanismo alternativo es el propio TFRC [Floyd and Kohler, 2006b] comentado en el apartado anterior, el cual utilizando una ecuación de rendimiento es capaz de minimizar los cambios abruptos en la tasa de envío, mientras mantiene equidad con TCP. En consecuencia, las aplicaciones seleccionarán la forma de control de congestión más apropiada de acuerdo con sus preferencias.

La cabecera genérica de DCCP toma diferentes formas dependiendo del valor del *bit* (X). Por ejemplo, la Figura 2.4 muestra una cabecera de 16 *bytes*, si el valor especificado es uno. En este caso, el campo de número de secuencia es de una longitud de 48 *bits*.

Cabecera de DCCP

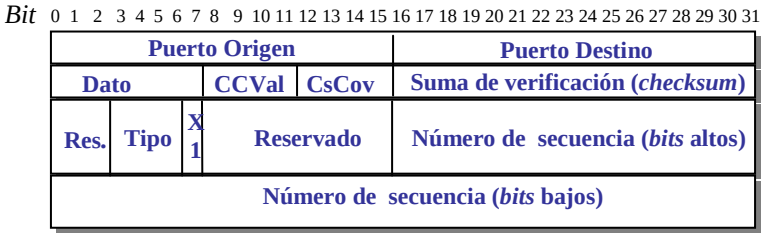


Figura 2.4. Formato de la cabecera de DCCP.

2.5. Conclusiones de Control de Congestión en Internet

En este capítulo se proporciona un breve resumen que sintetiza el intenso esfuerzo en investigación y desarrollo que los investigadores han llevado a cabo en los últimos años, con el fin de mitigar la congestión en Internet.

Se ha iniciado con una breve descripción de las causas que provocan la congestión y posteriormente, se han abordado algunos mecanismos desarrollados para combatirla. Entre ellos, los mecanismos básicos de control de congestión incorporados a TCP, que funcionan en los extremos finales. Sin embargo, estos mecanismos han llegado a ser insuficientes para proporcionar un buen servicio en todo tipo de circunstancias. De forma tal, como complemento a éstos, se incorporan mecanismos de prevención de la congestión en los puntos intermedios de la red, o *routers*, denominados Gestión Activa de colas.

Por otro lado, dada la alta demanda de aplicaciones que utilizan UDP, algunos protocolos recientes han sido también considerados. Los cuales, han sido diseñados para ajustar el servicio proporcionado a los requisitos de los datos multimedia.

Capítulo 3

Calidad de Servicio

Con el propósito de establecer los fundamentos para la búsqueda de nuevas alternativas que cumplan ó mejoren los requisitos de QoS de aplicaciones en tiempo real que contienen audio, en este capítulo nos centraremos esencialmente en el estudio de los elementos relacionados con la transmisión de paquetes de audio en redes IP.

Básicamente, se describe cómo se lleva a cabo la transmisión extremo-a-extremo, se identifican elementos que pueden deteriorar la calidad percibida por el usuario final y se discuten algunos esquemas de referencia que se han implementado con el fin de solventar esas necesidades.

3.1. Introducción

El establecimiento de IP como protocolo de red que permite abarcar redes de cualquier tecnología, ha generado un esfuerzo considerable para el desarrollo de arquitecturas que proporcionen calidad de servicio (QoS) en redes IP. Calidad de Servicio es un concepto que caracteriza las prestaciones y el funcionamiento de una comunicación extremo-a-extremo entre dos puntos finales de una red. La calidad de servicio se define por la UIT (Unión Internacional de Telecomunicaciones) como

> "*El resultado global de las prestaciones de un servicio que determina el grado de satisfacción de un usuario del servicio*" [ITU-T, 1994].

QoS esta relacionada, por lo tanto, con la calidad experimentada (QoE) o satisfacción subjetiva del usuario final. Por ejemplo, una transmisión de audio o vídeo de un extremo

a otro de la red no se puede considerar realizada con éxito total a menos que la calidad de las señales recibidas satisfaga al usuario.

Sin embargo, en Internet como ya se ha comentado la tecnología dominante proporciona el servicio denominado *best effort*, es decir un tratamiento FIFO no diferenciado. Por lo general, cada vez que se requiere información en Internet, ésta tiene que transmitirse a través de varias redes, por tanto, la calidad de servicio extremo-a-extremo depende principalmente de las características de QoS de los enlaces y *routers* por los que tiene que atravesar a su paso. En consecuencia, varias herramientas son imprescindibles para conseguir la QoS necesaria según el tipo de aplicación y usuario.

Para sustentar la QoS extremo-a-extremo es importante conocer más acerca del comportamiento dinámico de las redes. Esto es posible lograrlo mediante algunos parámetros que pueden ser medidos y monitoreados en la red. Entre los parámetros más importantes, que determinan si se cumple con el nivel de servicio que se ofrece, se encuentran: el ancho de banda, el retardo, el *jitter* y la probabilidad de pérdidas de paquetes.

3.2. Mecanismos de QoS

La interconexión de redes mediante protocolo IP (*best-effort*) ofrece un servicio no orientado a conexión y sin garantías de QoS. El protocolo IP se ha expandido vertiginosamente y sobre él ha emergido una gran diversidad de tráfico para el que inicialmente no fue concebido, como por ejemplo el multimedia con requisitos de tiempo real. Las aplicaciones en tiempo real si requieren de altos niveles de calidad ya que poseen necesidades específicas, son muy sensibles al retardo y *jitter* de transmisión, además requieren un ancho de banda garantizado.

El problema principal de calidad de servicio en voz y vídeo no estriba en la transmisión en si misma sino en las demandas de interactividad del servicio en tiempo real. Es decir, aun disponiendo de ancho de banda, en caso de pérdidas, por lo general, las aplicaciones multimedia (por cuestiones de interactividad) no tienen disponibilidad temporal para recuperar la información perdida mediante la solicitud de una retransmisión. Para que una red pueda proporcionar servicios en forma selectiva a ciertas aplicaciones, requiere de mecanismos que puedan diferenciar entre las diversas aplicaciones. En la Tabla 3.1 [ITU-T, 2001a] se muestran los requerimientos de algunas aplicaciones.

Tabla 3.1.
REQUERIMIENTOS DE APLICACIONES

Requerimientos de Funcionamiento			
Aplicación	**Retardo en una dirección**	***Jitter***	**Pérdida**
VoIP	< 150 ms (de preferencia) < 400 ms (límite)	< 1 ms	< 3% tasa de pérdidas de paquetes
Audio Streaming	< 10 s	<< 1 ms	< 1% tasa de pérdidas de paquetes
Videoconferencia	< 150 ms (de preferencia) < 400 ms (límite)		< 1% tasa de pérdidas de paquetes
Video Streaming	< 10 s		< 1% tasa de pérdidas de paquetes
HTML	< 2 s / página (de preferencia) < 4 s / página (aceptable)		Ninguna pérdida
Servicios de transacción	< 2 s (de preferencia) < 4 s (aceptable)		Ninguna pérdida
Juegos interactivos	< 200 ms		Ninguna pérdida
Correo	< 2 s (de preferencia) < 4 s (aceptable)		Ninguna pérdida

Ante la necesidad de establecer nuevas arquitecturas y protocolos que proporcionaran QoS en redes IP, varios grupos de la *Internet Engineering Task Force (IETF*) han desarrollado una serie de modelos, entre los cuales están principalmente el así denominado modelo de Servicios Integrados [Braden *et al.*, 1994] y el modelo de Servicios Diferenciados [Blake *et al.*, 1998].

3.2.1. Servicios Integrados (*IntServ*)

Servicios integrados o *IntServ* define un modelo que se basa en garantizar QoS a las aplicaciones a través de reservar recursos extremo-a-extremo de la red para cada flujo antes de ser transmitido. Las aplicaciones solicitan el nivel de servicio necesario para poder trabajar apropiadamente y las reservas son mantenidas hasta que la aplicación termina o mientras cumpla con las demandas solicitadas. En este modelo, se definen dos clases de servicios: servicios de carga controlada [Wroclawski, 1997] y servicios

garantizados [Shenker *et al*., 1997].

Servicios de carga controlada

Proporciona al flujo de datos una calidad de servicio lo más cerca posible a la calidad que el mismo flujo recibiría en una red *best effort* sin cargas. En este sentido, se denomina *better than best effort.* Para ello se utiliza un procedimiento de control de admisión en cada nodo para asegurarse que hay recursos para el flujo y supervisión a pesar de haber sobrecarga. Está especialmente indicado para implementaciones altamente simples que no tengan demandas precisas de QoS. El servicio de carga controlada no proporciona ninguna garantía cuantitativa respecto a retardo y ancho de banda. Este tipo de servicio es propicio para aplicaciones que pueden tolerar grandes rangos de variación en el retardo, pero sensibles a condiciones de sobrecarga en la red, por lo que no es indicado para aplicaciones con requisitos explícitos de tiempo real.

Servicios garantizados

Este tipo de reserva proporciona un retardo extremo-a-extremo de la red y garantiza un ancho de banda para el tráfico conformado con las especificaciones prefijadas. Para lo cuál, se necesita conocer las especificaciones del tráfico (*TSpec*, *traffic specification*) como son los parámetros del cubo de *token* del emisor, y las especificaciones de requerimiento de servicio (*Rspec, service request specification*) como el ancho de banda y el retardo solicitado. Este servicio está sujeto a un procedimiento de control de admisión en cada nodo. Los *routers* admitirán la solicitud del servicio de reserva dependiendo del *TSpec* (mensajes *PATH* de RSVP), del *RSpec* (en los mensajes *RESV* de RSVP) y de los recursos existentes. Es este el indicado para aplicaciones como audio y vídeo que requieren una entrega en tiempo real.

La reserva de ancho de banda puede ser hecha dinámicamente utilizando el protocolo RSVP (*Resource Reservation Protocol*) [Braden *et al*., 1997]. Una vez hecha la reserva, la aplicación puede iniciar el envío de tráfico en la ruta sobre la cual se han solicitado los recursos. Cada fuente se ha comprometido con cierto patrón de generación el cual debe ser cumplido. Esta arquitectura si bien garantiza QoS, no es lo suficientemente escalable debido a que el tratamiento por flujos puede dificultar su despliegue en escenarios de grandes demandas. Es muy costoso mantener en cada nodo una tabla de estados y reservas por cada flujo para el control de admisión, esto conduce a un considerable tráfico de señalización (por el protocolo RSVP) a lo largo del camino, y ocupación de recursos en cada *router.*

En definitiva, está típicamente limitada en tamaño y operación, por lo que resulta difícil su despliegue en los *backbones* de Internet que requieren de un gran número de reservas.

3.2.2. Servicios Diferenciados (*DiffServ*)

Para solucionar el problema de escalabilidad de *IntServ* se desarrolla la arquitectura de servicios diferenciados o *DiffServ*. En contraste a la orientación por flujo del protocolo RSVP, este modelo se basa en considerar el tráfico en diferentes clases de servicios (*class of service*, CoS), controlar la cantidad de tráfico de cada clase que cada cliente envía a la red y asegurar los requerimientos de QoS de varios tipos de aplicaciones utilizando mecanismos de colas en cada nodo con políticas claramente definidas (y dependientes del contrato o compromiso adquirido con el usuario) de *scheduling* y *dropping*.

El modelo de servicios diferenciados no utiliza la comunicación extremo-a-extremo para la reserva de recursos como lo hace *IntServ*. Este modelo se aplica a una región de la red y usa un sistema de clasificación agregado basado en reglas predefinidas para agrupar a los paquetes en clases. Las garantías de QoS se establecen considerando que el envío de tráfico se lleve dentro de ciertos valores de media, pico y tamaño máximo de ráfagas para cada clase considerada. Para aplicar los parámetros de QoS a las clases, los paquetes son clasificados con algún criterio, marcándolos usando la clave de servicios diferenciados (DSCP, *Differentiated Services Code Point*), utilizando para ello el campo *tipo de servicio* (*type of service*, ToS) en IPv4 o el campo *clase de tráfico* en IPv6, establecido en la cabecera del paquete IP.

El término PHB (*Per-Hop Behavior)* describe en este modelo el tratamiento específico de envío que recibirá un paquete en el interior de la red después de que es clasificado en una clase dada. Este tratamiento proporciona al paquete de un apropiado retardo, ancho de banda, *jitter*, etc. De tal manera, que todos aquellos paquetes que contengan igual DSCP recibirán idéntico tratamiento durante su envío. Se han definido cuatro PHBs en *DiffServ: Default* PHB que proporciona el tradicional servicio *best effort* [Nichols *et al*., 1998], *Class-Selector* PHB que conserva el comportamiento de un nodo basado en clasificación y envío [Nichols *et al*., 1998] con diferentes niveles de prioridad, *Assured Forwarding* PHB en el cual se definen clases y a cada una de ellas se le proporciona un servicio especifico de carga controlada [Heinanen *et al*.,1999] y *Expedited Forwarding* PHB que proporciona una baja pérdida, bajo retardo, bajo *jitter* y un servicio garantizado de ancho de banda [Davie *et al*., 2002].

El servicio ofrecido en *DiffServ* puede ser definido en términos cuantitativos o cualitativos considerando garantizar que el acuerdo de nivel de servicio establecido (SLA, *Service Level Agreement*) entre el proveedor del servicio y el cliente sea siempre respetado. Un SLA puede ser estático o dinámico y especifica los compromisos o niveles de servicio que serán proporcionados al cliente por parte del proveedor, por ejemplo el límite de ancho de banda. En *DiffServ*, hay una clara distinción entre lo que son *routers* externos (próximos a los usuarios finales) y los internos (también denominados *core routers*).

Como aproximación genérica y con el objetivo de mejorar la escalabilidad del modelo, se adopta la aproximación de realizar las tareas costosas en los *routers* externos (por ejemplo, la clasificación y la supervisión) mientras que las tareas menos costosas se realizan en los *routers* internos. Por lo que, los *routers* externos son los responsables de determinar de qué forma se ejecutará el envío de un paquete por la red, considerando como ya se ha comentado anteriormente, las clases especificadas y el SLA establecido para cada cliente.

Para realizar tal función, estos nodos clasifican el tráfico a retransmitir y realizan tareas de supervisión del tráfico:

- El tráfico que va llegando se dividirá en múltiples grupos basados en reglas predefinidas. La clasificación puede ser teniendo en cuenta el protocolo de capa superior, las direcciones de red, el interfaz de ingreso, los puertos origen o destino, o cualquier otro tipo de criterio al que se pueda tener fácil acceso y que pueda ser utilizado para proporcionar QoS.
- La supervisión del tráfico implica verificar y controlar los acuerdos o perfil del tráfico suscrito, midiendo el tráfico cursado, haciendo que los paquetes estén sujetos a acciones de marcado o adaptación para que se les permita entrar en la red. Además de considerar en su caso, la acción de descarte que puede ser aplicada a paquetes que no cumplan con el perfil esperado.

Esta arquitectura logra dar servicios diferenciados a tráfico agregado, es decir, se proporciona un tratamiento por clase, no por flujos; sin embargo se requieren la consideración de mecanismos complicados para negociar los acuerdos del nivel de servicio. Todo ello, además de requerir supervisión de tráfico, clasificación, conformación, marcado y, posiblemente, descarte de paquetes.

3.3. QoS en transmisión de VoIP

A pesar de las distintas soluciones propuestas de QoS para redes IP como *IntServ* y *DiffServ*, en Internet no se ha generalizado la aplicación de alguna de éstas, con lo cual el nivel de prestaciones depende del sobredimensionado de la capacidad de las redes. Sin embargo, esta opción no siempre resulta adecuada para proporcionar QoS y varias herramientas aún siguen siendo imprescindibles para conseguir la QoS necesaria según el tipo de aplicación y usuario. Las aplicaciones multimedia, en particular las de audio, dependen para su funcionamiento de distintos parámetros, por lo que éstos tienen una gran influencia en la calidad percibida por el usuario final. Veamos en detalle estos parámetros y algunos esquemas que han sido desarrollados para satisfacer a las aplicaciones de VoIP.

3.3.1. Elementos que deterioran la QoS percibida por el usuario

Retardo extremo-a-extremo

El esquema *best effort* no esta diseñado para enviar información con garantías de retardo o fiabilidad y las aplicaciones de audio, como ya se ha comentado previamente, son muy sensibles al retardo que se produce desde que la señal es enviada hasta que ésta es recibida por el usuario. La transmisión de paquetes de audio en tiempo real puede verse alterada si este retardo extremo-a-extremo excede un límite, que típicamente es de 100 a 300 ms para aplicaciones de audio interactivas [Wang, 2001]. Si el paquete de audio no es recibido antes de ese límite, éstos no pueden ser aprovechados por la aplicación y son considerados como no-útiles. La calidad del audio percibida por el usuario final se verá comprometida cuando los retardos sufridos sean elevados, ésta se deteriora significativamente por arriba de los 400ms.

Las causas que contribuyen al retardo extremo-a-extremo que pueden provocar dificultades en la transmisión de paquetes de audio en Internet y en definitiva, a la QoS que en mayor o menor medida percibe el usuario final son: el procesamiento digital de la señal (codificación y decodificación), el proceso de manipulación de paquetes (empaquetado, desempaquetado, encolamiento), la interfaz de la red y el protocolo de acceso al medio (Figura 3.1). Algunos de los retardos son determinados y conocidos anticipadamente (como los de codificación y decodificación), otros son variables e impredecibles (como los de encolamiento) que dependen de las condiciones de la red.

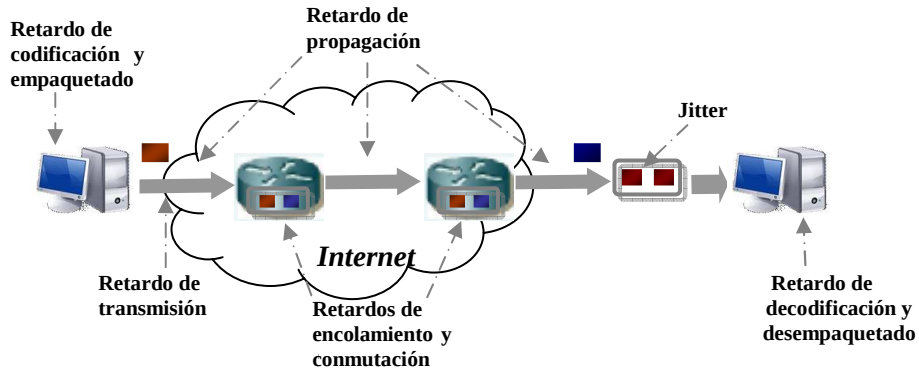


Figura 3.1. Retardos inherentes al retardo extremo-a-extremo del paquete.

- **Retardo de propagación.** Ésta es una fuente de retardo ineludible y se define como el tiempo que se produce durante la transmisión de los datos desde su origen hasta su destino final, directamente relacionada con la velocidad de la luz. Por lo tanto, es función de la distancia que tiene que recorrer. El retardo total de propagación vendrá dado por *Retardo total = distancia / velocidad de la luz.* La distancia es la clave de esta medida, puesto que el retardo de propagación puede verse afectado por un cambio de ruta desde un camino más corto hasta un camino más largo durante el tiempo de vida de la conexión. De ahí que cuando largas distancias están involucradas, los datos son enviados sobre enlaces satelitales. La velocidad de propagación en el cable es aproximadamente de 4 a 6 ms/km, mientras que con una transmisión satelital dependerá de la altitud del satélite. Por ejemplo, para un satélite con altitud de 14000 km, el retardo es 110 ms y para uno con altitud de 36000 km es de 260 ms.

- **Retardo de procesamiento**. Este retardo es consecuencia de los sucesivos procesos que tiene que pasar el paquete durante su trayecto desde el envío hasta que es recibido por el usuario final. Estos procesos son codificación y empaquetado que se llevan a cabo en el emisor, y decodificación y desempaquetado que se realizan en el receptor. Por otra parte, existen otros factores que contribuyen a este retardo, el retardo de encolamiento y el retardo de conmutación introducidos por los *routers* como resultado de clasificación, almacenamiento temporal (colas de entrada y salida) y conmutación.

  Retardo de codificación y decodificación. Es el tiempo requerido por el emisor y el receptor para digitalizar y comprimir los datos antes de ser enviados o

viceversa. El retardo introducido depende del *codec* utilizado, una alta compresión tiende a producir largos retardos algorítmicos. El retardo de decodificación es típicamente la mitad del retardo producido en la codificación [Kostas *et al.,* 1998]. Por ejemplo, para G.711 los retardos de codificación y decodificación impuestos son igual a cero y para G.729 de 15 ms y 7.5 ms respectivamente. En el caso del G.723.1, el retardo de codificación es igual a 37.5 ms y de 18.75 ms para decodificación. Inherente al retardo de codificación, se manifiesta el retardo algorítmico que está determinado por el mismo *codec*. Por ejemplo para el G.726 es de 1 ms mientras que para el G.728 varía de 3 a 5 ms.

Retardo de empaquetado y desempaquetado. Es el tiempo empleado en el emisor y en el receptor para encapsular y desencapsular un paquete de datos con tramas de audio codificadas y comprimidas, incluyendo las cabeceras y campos pertinentes. Este retardo esta en función del bloque requerido por el codificador y del número de bloques en una sola trama. El tiempo de paquetización de un paquete de audio debe tener aproximadamente una duración de 20 ms [Schulzrinne and Casner, 2003]. Sin embargo, algunos *codec* generan un retardo mayor a éste cuando su carga útil aumenta.

Retardo de encolamiento. Este retardo es el tiempo aleatorio de espera producido por las colas en los *routers* como consecuencia de que la razón de llegadas (salida del conmutador) es superior a la capacidad de la línea. Se genera por cada nodo que atraviesa el paquete y depende de la naturaleza del proceso de llegada, la utilización del enlace y del tamaño promedio de paquete.

Retardo de conmutación de la red. Considerado el tiempo que tarda en transportar el paquete desde su origen hasta el destino, es debido a los procesos dentro de los *routers.* En estos procesos se consideran las tareas de extracción de la dirección del destino del paquete, consultas de tablas de encaminamiento, etc.

- **Retardo de transmisión.** Es el tiempo que se tarda para transmitir un paquete. Este retardo está en función de la velocidad de la línea de transmisión y la longitud del paquete (tamaño de la trama utilizada por el *codec*). Cuanto mayor sea el tamaño del paquete, mayor será el retardo de transmisión.

Aunado a todos los retardos especificados, el retardo extremo-a-extremo se ve afectado además por otro factor, que no es sino su fluctuación o *jitter.* El transmisor envía cada paquete de una aplicación a un mismo ritmo de salida (con una periodicidad dada) pero la red (debido a la naturaleza *store and forward* de los *routers* IP) puede provocar

que el ritmo de llegada en el receptor no sea constante (fluctuaciones en los retardos). La medida de la variación del retardo al llegar al receptor (o extremo final) entre paquetes consecutivos de un flujo transmitido se denomina como *jitter*.

En todas las redes *store and forward* siempre se genera *jitter* distinto de cero a consecuencia de la variabilidad de retardos producidos cada vez que un paquete es encolado en un punto de la red. La calidad de audio percibida por el usuario se verá tanto más deteriorada cuanto mayor sea esta variación. Las aplicaciones multimedia con requisitos de tiempo real se caracterizan por demandar una variación del retardo máxima acotada, para así garantizar un flujo sin discontinuidades, tal que el *stream* de salida no tenga interrupciones no deseadas al ser reproducidas a través del correspondiente interfaz (audio o vídeo) con el usuario. Si aumenta el *jitter* considerablemente, el paquete no llega en el tiempo esperado, por lo que desde el punto de vista del receptor final es inservible o lo que es equivalente, es una pérdida. Si esto sucede con demasiada frecuencia, la calidad de audio será afectada significativamente, además del consumo inútil de recursos implicado.

Pérdida de paquetes y corrupción de datos

Un problema en la transmisión de audio es la pérdida de paquetes además de la corrupción de los datos por errores. Cuanto mayor sea el número de paquetes consecutivos (o longitud de la ráfaga) que se pierdan o se dañen, mayor será la degradación de la calidad suscitada [Ramos, 2009]. Las pérdidas de paquetes pueden ocurrir por el descarte generado en una situación de congestión en un *router* intermedio o como consecuencia de errores provocados por el medio físico de transmisión. Aunado a éstas, las pérdidas generadas por la llegada de paquetes inservibles en el receptor, como previamente se ha especificado.

Es tarea del *codec* prever mecanismos de suplantación de la información perdida con un coste mínimo sobre la degradación de la calidad de audio. Las pérdidas de paquetes pueden ser toleradas hasta cierto punto, sin ello no implica una degradación excesiva de la información percibida. Esto dependerá del impacto de la información perdida, del tipo de medio y de la técnica de codificación utilizada.

3.3.2. Proceso de transmisión de audio

La transmisión de paquetes de audio a través de la red puede efectuarse de dos formas, en tiempo real o bajo demanda. Para cualquiera de los dos casos, los datos son transportados

por la red por alguno de los dos métodos de distribución: *unicast* y *multicast.*

- La transmisión en tiempo real reproduce en el PC del usuario el audio de un evento a medida que éste se desarrolla en el sitio de origen.
- La transmisión bajo demanda reproduce en cualquier momento que se desee, los archivos disponibles para consultarse de audio pre-grabado, pre-codificado y almacenado dentro de un servidor de *streaming*.

El consumo de recursos dependerá del método de transmisión adoptado. Si se utiliza el *unicast*, éste tiene un efecto de consumo acumulativo, es decir, por cada usuario que se conecte a una transmisión de audio consumirá tantos *kbits/seg* como la codificación del contenido lo exija. En cambio, con el *multicast* el consumo de ancho de banda de la red (para la parte de la ruta que comparten todos los usuarios) es equivalente al de un único usuario, independientemente de cuantos sean los que se conecten. Este último método no es apropiado para transmitir bajo demanda puesto que cada usuario desea escuchar el contenido de audio en instantes diferentes.

El proceso de transmisión de audio se logra mediante la participación adecuada de tres elementos esenciales: emisor, *routers* y receptor, como se muestra en la Figura 3.2.

Emisor

La tarea principal del emisor consiste en enviar paquetes de audio por un canal de transmisión hasta el receptor o receptores. El emisor inicia el proceso de comunicación muestreando, digitalizando, codificando, empaquetando y enviando el paquete al receptor. Por otra parte, ante la necesidad de disminuir los efectos de pérdidas que se puedan generar durante la transmisión, el emisor recurre a ejecutar también otro tipo de funciones, como pueden ser, control de errores (añadiendo redundancia), control de congestión, clasificación y posiblemente, conformación de paquetes.

- **Generación y adquisición de la señal.** Como el sonido es una magnitud física analógica es preciso la conversión a señal digital para poder procesarlas mediante la computadora. Para generar señales digitalizadas de cualquier forma de onda, periódicas o no periódicas se requiere de filtrado para limitar la señal en banda, de muestreo y retención de la señal a través del circuito correspondiente y la cuantificación y codificación mediante un convertidor analógico-digital. Tanto el muestreador como el conversor serán controlados por el circuito de control que genera la señal muestreadora y las temporizaciones para generar la palabra código correspondiente a cada muestra, e inicio y finalización de la conversión.

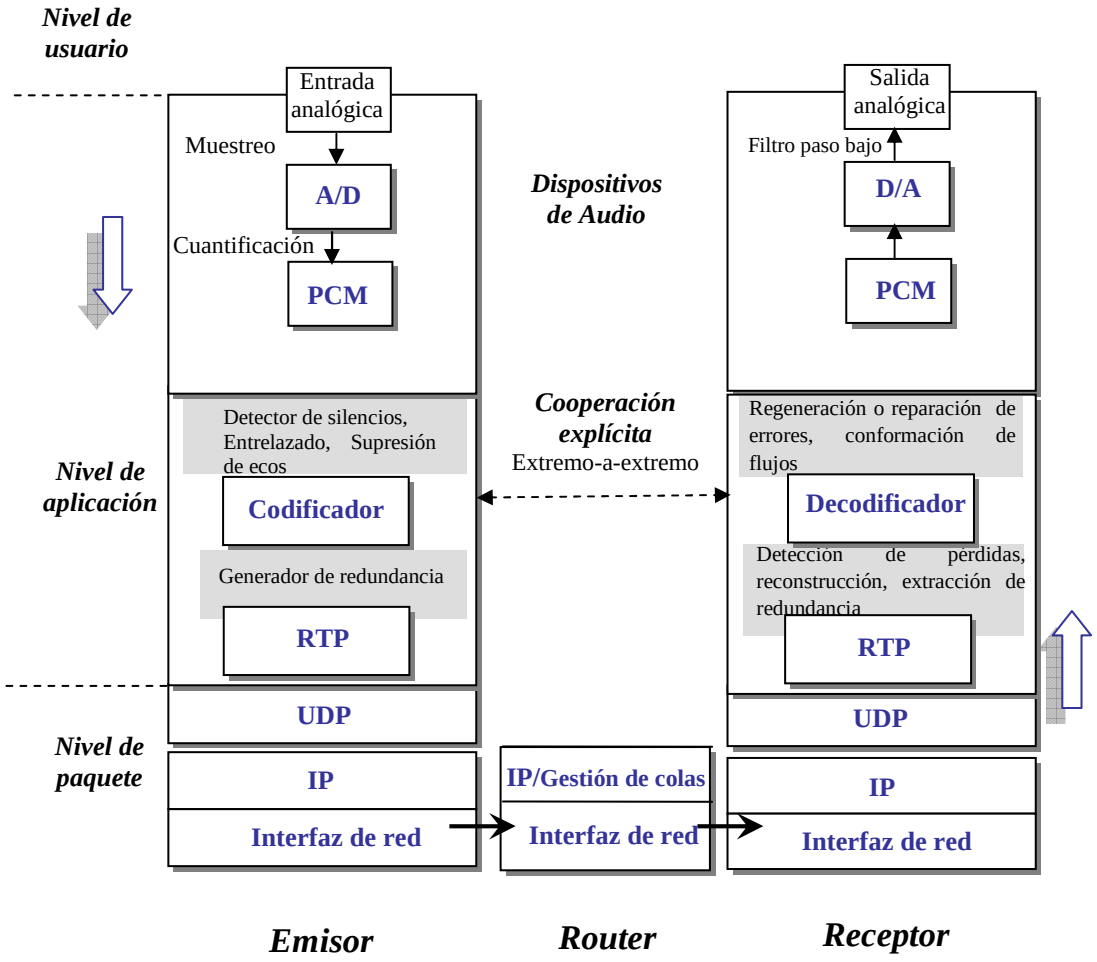


Figura 3.2. Proceso de transmisión de audio.

- **Preprocesado de la señal**. A la hora de procesar una señal de audio es importante considerar previamente ciertas técnicas que pueden otorgar una mejor preparación para el procesado. Las señales son fácilmente contaminadas por ruido e interferencias. Es por ello por lo que el primer eslabón del procesado consiste en una serie de pasos que pretenden limpiar las señales y eliminar componentes de interferencia. Entre las técnicas se encuentran: supresión de ecos, eliminación de silencios y entremezclado de segmentos.

Supresión de ecos. El eco es el fenómeno consistente en que una parte de la señal que ha sido enviada vuelve al origen retrasada y distorsionada. El eco que puede tener impacto en VoIP puede generarse como consecuencia de la acústica del micrófono y el altavoz, y puede incrementarse con el retardo y el *jitter*. El eco puede afectar muy negativamente a la calidad recibida, por lo cuál, se hace necesario el uso de canceladores de eco para minimizarlo y mejorar la calidad de audio resultante. El cancelador de eco substrae una señal corregida de la señal recibida basándose en la respuesta de una pequeña porción del sonido. El desempeño de los canceladores de eco se encuentra definido en *ITU-T Recommendation G.165* de la UIT [ITU-T, 1993].

Eliminación de silencios. Se determinan como silencios a aquellas tramas detectadas en las que no aparece información de la señal de audio. La detección y eliminación de silencios permite reducir el retardo extremo-a-extremo y reduce los requerimientos de ancho de banda, por lo que puede ayudar si no hay el suficiente ancho de banda en la red para soportar la transmisión de audio. Por otra parte, la eliminación de silencios contribuye para determinar exactamente dónde empieza y donde termina una trama. VAD (*Voice Activity Detection*) es un método que analizando la actividad de voz detecta intervalos de silencio [Yamada]. Puede trabajar en conjunto con un CNG (*Comfort Noise Generador*) para reducir la tasa de transmisión y salvar los recursos de procesamiento durante los intervalos de silencios. Estos también pueden ser utilizados efectivamente con canceladores de eco.

Entremezclado de segmentos. El entremezclado de segmentos es un método utilizado para incrementar la inteligibilidad de señales distorsionadas [Perkins *et al.,* 1998; Perkins, 1999]. Esta técnica envía partes segmentadas de una misma señal de audio en diferentes paquetes para posteriormente ser re-ensambladas a su orden original al ser recibidas. De esta manera, una pérdida de una longitud dada sobre un *stream* entremezclado, tras el re-ensamblado, se dispersarán evitando así el efecto no deseado de las pérdidas consecutivas. Si se pierde un segmento, todavía se dispondrá de la mayor parte de la señal. El entremezclado necesita siempre un *buffer* que genere datos en el emisor y un elemento de resecuenciador en el receptor, de tal manera que dependiendo de la longitud máxima de la ráfaga a dispersar puede introducir una alta latencia.

- **Codificación de la señal**. Para reducir el consumo de ancho de banda, el codificador es empleado para comprimir las señales de audio. Se intenta recurrir

a la menor cantidad de *bits* posible, pero tratando siempre de mantener ciertos niveles deseados de calidad. La reconstrucción de la señal dependerá por tanto, del tipo de técnica de codificación que se utilice. Las técnicas de codificación de voz están divididas en tres categorías: codificación en forma de onda, codificación fuente y codificación híbrida.

La codificación en forma de onda típicamente es utilizada para altas tasas de transmisión, no es compleja y proporciona muy buena calidad de audio. Cuando la señal es reconstruida tiende a ser una muy buena aproximación de la original. Consume grandes cantidades de ancho de banda, de tal forma que si es utilizado en sistemas con bajo ancho de banda, la calidad de la voz se degrada significativamente. Una sencilla técnica de codificación en forma de onda es el PCM (*Pulse Code Modulation*) [ITU-T, 2000], la cuál codifica cada trama independientemente. Esta técnica PCM ha sido estandarizada por la UIT en el *codec* G.711. Este *codec*, permite una señal de audio de calidad con ancho de banda de 3.4 KHz, que debe ser codificado para tasas de transmisión de 56 Kbps o 64 Kbps. Los codificadores fuente, también conocidos como *vocoder*, operan a muy baja tasa de transmisión, pero tienden a producir voz con sonidos artificiales. No ofrecen mejoría si son operados a una tasa de transferencia más alta. Los codificadores híbridos usan la combinación de técnicas de codificación fuente y de forma de onda, de tal manera que proporcionan buena calidad de voz a una tasa de transmisión intermedia.

- **Protección del contenido.** El emisor recurre a las técnicas FEC [Carle and Biersack, 1997] para dar una protección al contenido del paquete ante las posibles pérdidas. Estas técnicas se dividen en dos clases, las que son independientes del contenido del *stream* (denominadas FEC algebraico) y las que usan una referencia del *stream* (repeticiones de tramas anteriores con inferior calidad) [Hardman *et al*., 1995; Perkins *et al.*, 1998]. FEC algebraico hace una exacta reposición de un paquete perdido utilizando bloques de códigos para producir paquetes adicionales que serán transmitidos para ayudar en la corrección de las pérdidas. La gran desventaja de este esquema es el retardo adicional que impone (puede causar que los paquetes recuperados lleguen demasiado tarde), el incremento en el ancho de banda y una dificultad en la implementación del decodificador. Por consecuencia, no es especialmente recomendable para aplicaciones interactivas.

  El otro esquema de FEC, consiste en transmitir junto con el paquete de

audio original, uno o varios paquetes anteriores, generalmente con inferior calidad. De tal forma, si un paquete se pierde, otro paquete posterior podrá proporcionar la información perdida, aunque eso si, con menor calidad, por lo que en determinadas condiciones se podrá suplir la pérdida. Este esquema tiene la ventaja de ser computacionalmente poco complejo (dependiendo de los diferentes *codecs* utilizados). Es aceptable para aplicaciones interactivas además, nada impide, al igual que FEC algebraico ser usado junto técnicas de regeneración en el receptor.

En investigaciones hechas, por ejemplo en [Cidon *et al*. 1993], se ha demostrado que el esquema FEC tiende a no ser lo suficientemente efectivo en la recuperación cuando el proceso de pérdidas de paquetes se presenta a ráfagas. Por lo cual, se han desarrollado técnicas de envío múltiple de redundancia de *stream* de voz sobre diferentes e independientes rutas. Con la transmisión multi-*stream* de voz a lo largo de diferentes rutas en la red, se tiene más libertad para variar retardo, pérdidas tardías y calidad de reconstrucción de voz [Liang *et al*., 2002].

- **Empaquetado.** El empaquetado consiste en rellenar un paquete de datos con tramas de audio codificadas y comprimidas, incluyendo la cabecera y campos pertinentes. Los paquetes de audio se encapsulan habitualmente con el protocolo RTP (*Real Time Transport Protocol*, ver apartado 2.4), el cual esta encapsulado a su vez en UDP-IP (El formato de la cabecera de RTP ha sido mostrado previamente en capítulo 2, específicamente en la Figura 2.3). Si se aplica algún tipo de mecanismo de protección de pérdidas, por ejemplo, introduciendo redundancia o utilizando entrelazado, el formato de encapsulado debe contener la información correspondiente para que el receptor pueda hacer uso de ella.

- **Conformación de tráfico.** Los paquetes pueden ser enviados según el patrón de tráfico generado (tasa de envío variable) o independientemente de como se ha generado (tasa de envío constante). Antes de transmitir, el emisor puede adecuar el tráfico de salida según lo pactado con el receptor o los requerimientos de la aplicación. Como consecuencia de limitar la tasa de transmisión de paquetes, este método introduce latencia adicional a la conexión. Ejemplo de este tipo de mecanismo, el esquema *Token Bucket* [Shenker and Wroclawski, 1997].

Routers o nodos intermedios

La tarea principal del *router* es interconectar redes y encaminar paquetes entre ellas. Cuando un transmisor envía un paquete IP a un destino situado en una red remota, éste siempre pasará al menos por un *router* o nodo intermedio antes de llegar al receptor. En orden de proporcionar cualquier clase de servicio a los paquetes (ejemplo: reserva de ancho de banda), éstos deben ser primeramente clasificados por los *routers* de acuerdo a las características especificadas en la cabecera. Aunado a esto, dependiendo del ancho de banda que haya en las líneas de transmisión y el tráfico existente, el *router* debe tomar la decisión de si los paquetes deben enviarse, almacenarse en colas, descartarse o marcarse, y de qué manera debe de hacerse para evitar excesivos descartes y en definitiva pérdidas de paquetes.

Existen dos clases de esquemas, no excluyentes entre sí, que pueden ser implementados en los *routers*: la gestión activa de colas y algoritmos de planificación o *scheduling.*

- Los procedimientos de planificación o *scheduling* determinan, dada una línea de salida y un instante dado, qué paquete servir. La estrategia de planificación seleccionará un paquete para ser transmitido de entre todos los paquetes que estén esperando ser transmitidos. Existen varios tipos de mecanismos *de planificación*, cada uno de ellos busca un balance entre complejidad, control y equidad.

  Por ejemplo, el mecanismo *Fair Queueing* permite a varios flujos compartir equitativamente la capacidad del enlace. A diferencia de las colas FIFO, este planificador no consiente que flujos predominantes tomen más de la porción de ancho de banda que les corresponde. La capacidad del *buffer* es dividida en varias colas, donde cada una de ellas es utilizada para mantener los paquetes de un flujo, definido por ejemplo por la dirección IP de origen o de destino. Por otra parte, *Priority Queueing* (PRIQ) (Figura 3.3) donde se determinan colas múltiples con un nivel de prioridad dado. A medida que van llegando los paquetes, se clasifican y se envían a la cola correspondiente. En cada una de ellas los paquetes ya son tratados bajo el criterio FIFO (*First Input First Out*, el primer paquete que llega a la cola es el primero que se sirve). Una cola con un nivel de prioridad más alto será servida siempre antes que una cola con un nivel de prioridad más bajo. Este tipo de cola proporciona diferenciación en ancho de banda y retardo entre las distintas clases de tráfico.

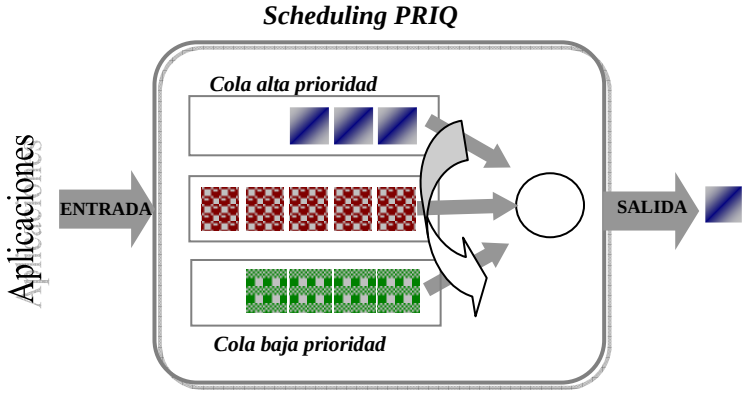


Figura 3.3. Mecanismo de planificación *Priority Queueing* (PRIQ).

- Para controlar las pérdidas producidas por la congestión, se han desarrollado mecanismos de prevención como la gestión activa de cola. Los mecanismos AQM han sido previamente introducidos en el capítulo 2 (específicamente en apartado 2.3), y éstos serán comentados en detalle en el capítulo 4.

Receptor

El último elemento por describir es el receptor. La tarea principal del receptor consiste en recibir, desempaquetar y decodificar paquetes para reproducir señales de audio que fueron enviados por un transmisor. Por otra parte, ante la necesidad de garantizar la inteligibilidad del audio original, el receptor efectúa otro tipo de funciones, entre ellas, la reserva de recursos necesarios para la transmisión que esta solicitando (en caso de optar por el modelo *IntServ*) y la detección de pérdidas o errores de paquetes para su posterior reconstrucción, la mitigación o amortiguamiento del *jitter*, y en su caso, técnicas de regeneración o realce.

- **Reserva de recursos**. La reserva de recursos puede hacer que una red basada en tecnología IP, pueda garantizar al receptor tiempos de transmisión extremo-a-extremo acotados, pero no evita los errores a consecuencia de fallos en los nodos. La reserva de recursos se hace con RSVP (*Reservation Protocol*) e implica una reserva para cada flujo de datos de usuario, y como ya se explicó en el apartado 3.2.1 es parte esencial del modelo *IntServ*. RSVP es un protocolo *soft-state* que debe mantener en cada nodo los requerimientos de reserva, por lo que define un conjunto de mensajes de señalización. Esto conduce a un considerable tráfico de señalización y ocupación de recursos en cada *router* para

cada flujo, por lo que la escalabilidad no es una de sus virtudes. Por las razones anteriores, además de la necesidad de interoperatividad que se exige entre distintos operadores en la ruta para ir de un extremo a otro han dificultado su despliegue en Internet.

- **Desempaquetado.** Consiste en realizar inversamente el proceso de empaquetado, desencapsular el paquete de datos con tramas de audio codificadas y comprimidas, incluyendo la cabecera y campos pertinentes. Durante el desencapsulado se efectúa la interpretación de la cabecera y el formato del paquete, y tiene lugar una comprobación de errores sobre los datos. Si no se ha producido ningún error de transmisión, se comprueba el campo de dirección para comprobar si esa trama iba dirigida a ese nodo.

- **Detección de pérdidas.** La detección oportuna de pérdidas en el receptor permite que las tramas puedan repararse o recuperarse. Utilizando el número de secuencia establecido en la cabecera del paquete RTP, el receptor detecta pérdidas, detecta la recepción fuera de orden o los duplicados de paquetes. Como ya se explicó en el apartado 3.3.1, si los paquetes de audio no llegan en el límite de tiempo establecido (300 ms) se consideran como pérdidas en el sistema. Para determinar cuanto tiempo tiene de haberse enviado el paquete, el receptor utiliza la marca de tiempo señalada en la cabecera del paquete RTP (Figura 2.3). Si se descubre la pérdida de un paquete individual, y el emisor utilizó un generador de redundancia, el receptor podrá mitigarla siempre y cuando la redundancia introducida lo permita.

- **Decodificación.** Consiste en realizar inversamente el proceso de codificación**.** La decodificación total del paquete depende de la técnica de codificación que se utilizó y de la pérdida de tramas que se generó. Si se utilizó un codificador de trama independiente, la pérdida de una de ellas puede no afectar para la decodificación total del paquete. Por otra parte, en caso de haber utilizado una técnica de codificación diferencial, la falta de una trama puede evitar la correcta decodificación de las subsecuentes tramas.

- **Reparación de tramas erróneas.** Cuando las técnicas de recuperación aplicadas en el emisor fallan o son incapaces de reparar correctamente, aún pueden utilizarse las técnicas de regeneración o realce aplicadas en el receptor. Las técnicas de reparación o regeneración basadas en el receptor son totalmente complementarias a las basadas en el emisor. Si se utilizan ambos métodos, se puede lograr el mejor funcionamiento posible.

Los esquemas de regeneración de errores basados en la inserción consisten en sustituir las tramas faltantes por ruido blanco, silencio o repitiendo el paquete anterior. Sin embargo, estudios realizados [Perkins *et al.,* 1998] han revelado que utilizar ruido blanco es mejor que sustituirlo por silencio. Por otra parte, existen los esquemas basados en la interpolación, utilizan un patrón similar de las características de los paquetes cercanos al paquete perdido e interpolan para obtener el paquete a reemplazar. Este esquema funciona para tasas de pérdidas relativamente pequeñas (<15%) y será tanto mejor cuanto menor sean los paquetes (4-40 ms). Para ráfagas de pérdidas que aproximen al tamaño de un fonema (5-100 ms) o superiores, la regeneración es menos eficiente. Esta técnica puede proporcionar mejor calidad que la obtenida al utilizar sustitución por silencio o repetición de paquetes [Goodman *et al*., 1986]. Los más sofisticados esquemas de regeneración pueden tolerar altas tasas de pérdidas pero a cambio de altos costes computacionales. Investigaciones en [Liao *et al.,* 2001] revelan que son pocas las técnicas de recuperación de pérdidas para aplicaciones de *streaming* audio que son capaz de sacarle provecho al trueque entre la calidad de recuperación y la complejidad computacional.

- **Procesamiento posterior de la señal.** Como el sonido es una magnitud física analógica, el receptor hace la conversión de señal digital a analógica para adecuarla al entorno del usuario. La información previamente manipulada puede tomar nuevamente su valor analógico utilizando un CDA (Convertidor digital-analógico). Dependiendo del tipo de resolución (número máximo de bits por salida) que se tenga establecida, será la exactitud de la conversión.

Para contrarrestar los efectos de las fluctuaciones de la red, se han desarrollado mecanismos denominados *jitter buffers*, también conocidos como *playout buffer*. Algoritmos de *playout buffers* han sido investigados, por ejemplo, en [Moon *et al*., 1998; Boutremans and Boudec, 2003]. Estos mecanismos se utilizan para cambiar la llegada asíncrona del paquete a un *stream* síncrono para tornar los retardos variables de la red en retardos fijos en el destino final del sistema. La supresión del *jitter* requiere el almacenamiento de los paquetes en una cola hasta que los paquetes más lentos puedan ser recibidos. Como los paquetes de audio son secuenciales y periódicos generalmente, el *jitter buffer* también asegura que sean recibidos por la aplicación en el orden correcto, cosa que como es conocida no puede ser garantizada por el protocolo IP. Un algoritmo *playout buffer* compensa en el receptor las fluctuaciones entre las llegadas pero usualmente a cambio implica la introducción de un retardo adicional.

3.4. Métodos de evaluación de QoS

Ante la necesidad de medir la calidad extremo-a-extremo en algunos sistemas de codificación y transmisión de paquetes, se han desarrollado varias alternativas. Estas alternativas creadas con el fin de obtener el grado de rendimiento de un servicio se definen como Métodos Objetivos y Métodos Subjetivos.

3.4.1. Métodos Objetivos

Aún cuando dos rutas son simétricas, éstas pueden tener radicalmente diferentes características de rendimiento debido a un encolamiento asimétrico. En consecuencia, la QoS de una aplicación provista en una dirección puede ser radicalmente diferente a la que se proporciona en dirección contraria, por lo que las garantías de QoS difieren. Si se miden las rutas independientemente se permite la verificación de ambas garantías. Basándose en factores como éstos, el grupo de trabajo de la *IETF* denominado IPPM (IP *Performance Metric*) ha desarrollado un marco general de medidas y métodos [Paxson *et al.*, 1998] que relacionan el rendimiento y fiabilidad de operación de aplicaciones en Internet.

Una de estas medidas es una métrica analítica relativamente sencilla que considera la marca de tiempo establecida en la cabecera del paquete denominada *Retardo de una sola dirección de paquetes a través de rutas de Internet* [Almes *et al*., 1999a]. Esta medida determina si el valor del retardo es simplemente muy largo, es decir si es superior a un límite (por ejemplo el límite superior teórico de 255 segundos sobre el tiempo de vida de los paquetes IP en el cual puede ser utilizado). Basándose en esta medida, se establecen otras tres métricas: (1) *Paquetes perdidos en una sola dirección* [Almes *et al*., 1999b] que determina si hay una excesiva pérdida de paquetes que impida soportar ciertas aplicaciones en tiempo real, (2) *Variación del retardo del paquete IP* [Demichelis and Chimento, 2002] una medida para la variación del retardo de paquetes de un mismo flujo desde un nodo a otro a través de una ruta IP, y finalmente (3) *Retardo de ida y vuelta* [Almes *et al*., 1999c] que proporciona un indicador del retardo debido solamente a la retardo de propagación y transmisión, además de proporcionar una indicación de la presencia de congestión en la ruta.

Aún más, un método y métricas relevantes para estimar el rendimiento de ciertas aplicaciones de redes IP son establecidos en *Medidas de rendimiento de red con periódico stream* [Raisanen *et al*., 2002]. Este método es utilizado para medir QoS de tráfico interactivo multimedia. De ahí que esta metodología determina el retardo y su

variación experimentada por el *stream* multimedia de una aplicación.

Por otra parte, métodos objetivos para medir la calidad de la voz, han sido desarrollados en *ITU-T Recommendation P.563* [ITU-T, 2004]. Este método intenta estimar la calidad analizando la señal de voz recibida, sin requerir la señal original. Este método no proporciona una evaluación completa de la calidad de transmisión, únicamente mide unidireccionalmente los efectos de la voz y del ruido sobre la calidad vocal. En *ITU-T Recommendation P.564* [ITU-T, 2007] se proporciona un método para evaluar específicamente la calidad de transmisión de VoIP. Este método es basado en la información de los paquetes RTCP-SR, RR y XR.

3.4.2. Métodos Subjetivos

Este tipo de métodos están basados en la evaluación de la percepción del usuario. En un sistema de comunicación de voz dado, la evaluación de la calidad percibida es un proceso costoso, consume mucho tiempo y puede ser muy difícil reproducirlo. Sin embargo, para el caso general de comunicación multimedia, y para aplicaciones de VoIP en particular, proporcionar excelente rendimiento en términos de calidad de servicio (QoS) es inaprovechable si ésta no está definitivamente de algún modo correlacionada con la asociada calidad de experiencia (QoE).

Dentro del ámbito de las comunicaciones de voz y aplicaciones multimedia, UIT-T ha desarrollado modelos subjetivos para evaluar la percepción de usuario de la calidad de servicio.

Método de Puntuación de Opinión Media

Uno de los primeros métodos subjetivos fue el de Puntuación de Opinión Media (*Mean Opinion Score*, MOS) definido en *ITU-T Recommendation P.800* [ITU-T, 1996]. MOS representa un índice de referencia para evaluar la calidad de la voz resultante en sistemas de comunicación. En este método, el evaluador oyente, dada una locución, decide de acuerdo con su opinión una puntuación numérica definida entre 1 (calidad pobre) hasta 5 (calidad excelente). Dado un sistema de comunicación, su evaluación MOS consiste por tanto en la elección de una base de datos de locuciones, la cual tras ser transmitidas por el sistema (mediante simulación o sobre un sistema real) es reproducida y evaluada por una serie de oyentes, a partir de los cuales se obtienen estadísticas de la opinión percibida.

Por otra parte, *ITU-T Recommendation P.862* [ITU-T, 2001b] proporciona un

modelo basado en la señal denominado PESQ (*Perceptual evaluation of speech quality*) que permite generar una predicción de la calidad percibida comparando la señal de voz de referencia transmitida con la señal degradada que se recibe. Una de las principales características del modelo es el hecho de tener en cuenta los efectos de compresión de voz, la variación del retardo y los errores del canal. Sin embargo, PESQ no proporciona una evaluación exhaustiva de la calidad de transmisión, mide solamente los efectos del ruido y la distorsión de voz unidireccionales sobre la calidad de voz. Este método calcula dos parámetros de error, lo cuales se combinan para dar una MOS de calidad de escucha objetiva.

La metodología MOS, en principio destinada a la valoración subjetiva de la calidad de voz, se ha ampliado para incluir aplicaciones multimedia. *ITU-T Recommendation P.911* [ITU-T, 1998] presenta un modelo de evaluación subjetiva para sistemas de comunicación unidireccional (no interactiva). La medida de la calidad percibida en este modelo se basa en tres métodos de escala subjetiva a utilizar según la aplicación determinada. Por ejemplo, en el método de índice por categorías de degradación (*degradation category rating,* DCR), el evaluador observa la presentación de secuencias de prueba por pares y emite su opinión de forma independiente dando una puntuación en una escala entre 1 (Muy molesta) hasta 5 (Imperceptible). Los resultados de calidad dependen no sólo de la calidad real del vídeo y audio sino también de otros factores como la experiencia y expectativas de los evaluadores.

En *ITU-T Recommendation P.920* [ITU-T, 2000] se amplía el método definido en *ITU-T Recommendation P.800* incorporando la evaluación de comunicación de audio y vídeo interactiva. En este modelo se definen métodos de evaluación para cuantificar los efectos de compresión de voz, del retardo de la transmisión y el deterioro causado por los factores de transmisión (como pérdida de paquetes) en las comunicaciones punto a punto o multipunto.

Los esquemas MOS por tanto proporcionan una evaluación genérica del concepto subjetivo de la calidad percibida.

Modelo-E

Como el proceso de puntuación MOS es un procedimiento con poca reproducibilidad y de alto coste, como alternativa o complemento, se han propuesto e implementado métodos que de forma objetiva intentan estimar la calidad subjetiva oral en sistemas de transmisión de voz, en general, y de voz sobre IP en particular. Entre estos métodos objetivos destaca en cuanto a la relación entre calidad percibida y calidad de

funcionamiento de la red el denominado Modelo-E, definido en *ITU-T Recommendation G.107* [ITU-T, 2005].

El modelo-E fue inicialmente concebido para el propósito de planificación y diseño de redes, éste predice los efectos subjetivos que pueden ser experimentados por un promedio de interlocutores. Sin embargo, este ha sido también adoptado para estimar la calidad subjetiva percibida por los usuarios en muchos sistemas de transmisión de voz, combinando el deterioro causado por parámetros de transmisión dentro de una evaluación sencilla. Este modelo asume que los distintos factores que pueden influir en la degradación de la calidad de voz son aditivos e independientes entre si. Los factores que deterioran la calidad resultante considerada en este modelo son los retardos de transmisión, el eco, la distorsión introducida por el uso del equipo y las pérdidas de paquetes, entre otros factores.

El resultado (o puntuación) final es la suma de los factores considerados por el modelo resultando en un factor escalar *R* que varía de 0 (el peor de los casos) a 100 (excelente). Este factor puede ser convertido a escala de puntuación MOS a través de las siguientes tres expresiones [ITU-T, 2005]:

- Para $R < 0$: MOS =1
- Para $0 \leq R \leq 100$: $MOS = 1 + 0{,}035*R + 7*10^{-6}*R*(R-60)*(100-R)$
- Para $R > 100$: MOS = 4,5

Usualmente, el factor *R* es descrito en categorías de valores, donde cada rango corresponde a un nivel de calidad bien diferenciado, tal como se muestra en la Tabla 3.2. Si una conversación es evaluada por debajo de 60 ($R \leq 60$) indica calidad inaceptable, por lo que ese sistema no es recomendable. Por otra parte, si la conversación es evaluada por encima de 70 ($R \geq 70$) corresponderá a una buena calidad.

El valor *R* en el Modelo-E está dado por

$$R = Ro - I_s - I_d - I_e + A \qquad [3.1]$$

Donde,

- **Ro** incluye los efectos de ruido introducidos por el sistema de transmisión.
- $\mathbf{I_s}$ (*simultaneous factor*) este factor está relacionado con la SNR (*signal to noise ratio*) del canal de transmisión.
- $\mathbf{I_d}$ (*delay factor*) es el deterioro asociado con el retardo de la señal. Este puede ser debido a la presencia de eco en el emisor, eco en el receptor y retardo

Tabla 3.2.
NIVEL DE CALIDAD DE LA TRANSMISIÓN EN MODELO-E

Nivel de calidad de la transmisión		
Rango Factor R	**MOS**	**Calidad de Transmisión**
90≤R<100	4,34 - 4,50	Muy alta (la mejor)
80≤R<90	4,03 - 4,34	Alta
70≤R<80	3,60 - 4,03	Media
60≤R<70	3,10 - 3,60	Baja
0≤R<60	1,00 - 3,10	Pobre

extremo-a-extremo.

- $\mathbf{I_e}$ (*equipment factor*) es el deterioro asociado con los mecanismos de codificación. Este puede ser causado por bajas tasas de transmisión de *codecs* y pérdidas de paquetes.
- **A** (*expectation factor*) es un valor subjetivo que corresponde al beneficio de tener acceso al servicio de comunicación oral proporcionado por el sistema de transmisión a evaluar. Este valor es igual a 0 para sistemas basados en cableado (sin movilidad), 5 para sistemas inalámbricos DECT (*Digital Enhanced Cordless Telecommunications*), 10 para GSM (*Global System for Mobile communications*) y de 20 para sitios solamente alcanzados por conexión satelital.

3.4.3. Otras alternativas

Sistemas Automáticos de Reconocimiento de Voz sobre IP

El modelo-E a pesar de proporcionar una aproximación reproducible y de bajo coste en relación a la evaluación MOS, tiene algunos inconvenientes que motivan la necesidad de considerar medidas complementarias. Entre éstas, dada la definición del Modelo-E y en definitiva de la puntuación MOS, se observa una clara dificultad en valorar la inteligibilidad final que cabe esperar del sistema de transmisión oral. Por tanto, en un intento de proporcionar un mecanismo de evaluación complementario, se propone la consideración de un esquema basado en reconocimiento automático de voz para analizar la inteligibilidad de los flujos (medida en términos de tasas de palabras ó exactitud de frases) [Jiang and Schulzrinne, 2002; Ramos *et al*., 2005], aspecto éste verdaderamente relevante para las aplicaciones que implican la transmisión de información multimedia.

Con esta aproximación, es de esperar que cuanto mayor sea la inteligibilidad, nitidez y en definitiva la calidad resultante mayor será la puntuación o tasa de acierto obtenida en un reconocedor automático situado en el receptor.

Una vez definida la base de datos de voz de *test*, el procedimiento de evaluación consistirá simplemente en transmitir (realmente o por simulación) dicho *corpus* a través del sistema de comunicación y la señal obtenida reconocerla en el destino proporcionando como medida objetiva alguna medida del acierto obtenido.

Evidentemente, una de las propiedades destacables de esta metodología es su gran reproducibilidad, respecto a MOS, ya que no precisa ningún conjunto de oyentes normalmente especializado. Respecto al Modelo-E no precisa realizar los ajustes y sintonización (factores de peso) de parámetros necesarios en aquel. Más concretamente, se propone predecir la calidad estimando la tasa de error de palabras (WER, *Word Error Rate*) [Ramos *et al*., 2005], en un sistema de reconocimiento de voz continua.

La tasa de error de palabras está definida por

$$WER = \frac{n_i + n_s + n_d}{n_t} * 100 \qquad [3.2]$$

Donde n_i es el número de palabras falsas insertadas, n_s es el número de palabras sustituidas, n_d es el número de palabras borradas y finalmente, n_t es el número total de palabras.

Antes de que se registre el conteo de palabras sustituidas, es utilizada una programación dinámica para alinear las oraciones reconocidas con su correcta trascripción. Adicionalmente, la precisión de palabras (WA de 0 a 100%) es estimada mediante

$$WA = 100 - WER \qquad [3.3]$$

Este sistema tiene un bajo coste comparado con las pruebas de MOS con un significante número de interlocutores.

3.5. Conclusiones de QoS

En este capítulo se ha proporcionado una visión general de los principales mecanismos implementados para solventar y medir la calidad (objetiva y subjetiva) de transmisión en redes IP. Se han caracterizado los distintos modelos propuestos *IntServ* y *DiffServ* haciendo hincapié en sus dificultades o beneficios. Además de los modelos propuestos en

IETF, en este capítulo se muestra que las demandas de QoS pueden aproximarse con un conjunto de mecanismos y técnicas que pueden ser configurados para inter-operar en la red en una forma estable y consistente. Por ejemplo, el emisor puede recurrir a las técnicas FEC para dar una protección al contenido del paquete ante las posibles pérdidas (parámetro que tiene una gran influencia en la calidad) o implementar esquemas en los *routers* como la gestión activa de colas y algoritmos de planificación o *scheduling.*

Para finalizar se revisa brevemente distintas metodologías para evaluar la QoS recibida por usuarios en flujos de VoIP.

Capítulo 4

Algoritmos de gestión activa de colas

Este capítulo intenta proporcionar una visión general de los algoritmos de gestión activa de colas más relevantes que han sido presentados en la literatura. Los cuáles se han concebido no sólo con el fin de detectar, evitar y controlar la congestión si no también con el objetivo de mejorar las prestaciones en cuanto a los parámetros que definen la QoS (pérdidas de paquetes, utilización del enlace, retardo y *jitter).*

Entre ellos, se presenta *Random Early Detection* (RED), considerado como punto de partida o esquema de referencia, el cual, ha motivado un gran número de interesantes trabajos.

4.1. Introducción

Uno de los retos cruciales que afrontan las tecnologías basadas en IP es el hecho de prevenir la congestión en la red. Para tratar este problema, un número de destacados sistemas de Gestión Activa de Colas (AQM, *Active Queue Management*) han sido propuestos. La Gestión Activa de Colas es una clase de algoritmos que intentan prevenir la congestión descartando o marcando paquetes en el *router* comprometido. Estos utilizan una aproximación probabilística para reaccionar a las condiciones de congestión.

La gestión activa de colas ha generado gran interés por parte de la comunidad toda vez que no afecta a la interoperatividad entre equipos, y puede suponer un valor añadido

para los fabricantes. En términos generales, la QoS extremo-a-extremo depende de las líneas de transmisión y del funcionamiento de los *routers*. Ésta es también sujeta al patrón de tráfico generado y, por último pero no menos importante, a los errores de encaminamiento y a los algoritmos de congestión. La dinámica del comportamiento de reenvío de los *routers* es determinante en la calidad del servicio ofrecido a nivel de red, lo que finalmente es decisivo en la calidad percibida por los usuarios de aplicaciones multimedia. En cada *router* pueden converger desde uno hasta miles de flujos de paquetes, que en muchas ocasiones, también pueden llegar a involucrar un número significativo de enlaces (o saltos). Por lo general, para amortiguar las diferencias entre la razón de llegadas y la razón de salidas, las interfaces de salida en los *routers* disponen de un *buffer* o memoria temporal donde los paquetes serán almacenados para ser enviados posteriormente. Esta cola puede estar constituida por paquetes de distintos tipos de flujos, los cuales pueden haber seguido rutas diferentes. En situaciones de congestión, es decir cuando la cola se llena, si el *router* sigue una estrategia no diferenciada, los paquetes son descartados o marcados sin importar a qué tipo de flujo pertenecen.

La QoS experimentada por los paquetes de audio puede deteriorarse durante su transferencia de un extremo a otro de la red, a consecuencia de la variabilidad de retardos y descartes producidos cada vez que un paquete es encolado en cada uno de los *routers* atravesados. Sustentándonos en estos factores, la investigación realizada se ha enfocado esencialmente en el estudio y mejora de mecanismos de gestión activa de colas para el caso de paquetes con información multimedia.

Diferentes aspectos han sido ampliamente estudiados en la literatura desde que RED fue propuesto. Por ejemplo, algunas de las propuestas se han enfocado en la estabilidad de la ocupación de la cola, como *Stabilized* RED (SRED) [Ott *et al*., 1999] y RaQ [Sun and Zukerman, 2007]. La configuración de los parámetros RED también ha sido considerada, dando como resultado esquemas como *Adaptive* RED [Floyd *et al.*, 2001] o nuevas estructuras para encontrar el valor optimo del parámetro de máxima probabilidad (Max_p) en *routers* RED [Zheng and Atiquzzaman, 2008]. Otros esquemas como *Flow Random Early Drop* (FRED) [Lin and Morris, 1997] y *Stochastic Fair Blue* (SFB) [Feng *et al.*, 2002] han sido propuestos para otorgar equidad entre los flujos. Por otra parte, en [Lakshman *et al*., 1996] alternativas de estrategias de descarte de paquetes, como *Drop Front*, han sido investigadas. Finalmente, alternativas para proporcionar un mejor control sobre niveles de tráfico a ráfagas han sido consideradas en [Feng *et al*., 2004].

Puesto que obviamente es imposible cubrir en esta investigación todos los esfuerzos que han sido realizados, sólo se abordarán algunos de los más representativos algoritmos de gestión activa de colas. Entre los algoritmos descritos podrán encontrarse los más

populares así como las más recientes alternativas.

4.2. Random Early Detection (RED)

El algoritmo *Random Early Detection* desarrollado en [Floyd and Jocobson, 1993], es uno de los primeros y más relevantes esquemas para evitar la congestión en la arquitectura de Internet. Trabajos previos como el esquema DECbit [Jain *et al*., 1988] y *Early Random Drop* (ERD) [Hashem, 1989] anteceden al trabajo de Detección Temprana Aleatoria. RED detecta la congestión estimando la ocupación media de la cola siempre y cuando arribe un paquete a ella. Este algoritmo pretende evitar el descarte y posterior pérdida de paquetes IP en los *routers* sin menoscabar con ello las prestaciones de la red.

Cada vez que llega un paquete y siempre que la ocupación media de la cola excede un umbral predeterminado, el mecanismo provoca que se descarte o se marque con una cierta probabilidad, donde la probabilidad exacta está en función de la ocupación media de la cola del *router*. Con RED, un *router* puede realizar un descarte antes que la cola se sature. La idea por tanto es, suponiendo que la congestión está relacionada con la ocupación media de la cola, situar al algoritmo en un punto de funcionamiento óptimo que involucre un compromiso adecuado entre la utilización del ancho de banda de salida de la cola y la reacción temprana o evitación de la congestión.

Una vez determinada la necesidad de notificar congestión (esto es cuando la cola supera una ocupación determinada) RED selecciona con algún criterio una fuente (de entre las presentes en la cola) y le notifica la congestión para que ésta pueda bajar su tasa de transmisión. Observamos que RED es independiente del mecanismo de notificación adoptado, el cual puede ser tanto explícito (devolviendo un paquete de control ICMP al origen) como implícita (descartando un paquete). RED logra mantener un bajo nivel de ocupación media de la cola mientras permite ocasionalmente ráfagas de paquetes. Téngase en cuenta, que el objetivo último de todo algoritmo de gestión activa de colas es mantener la ocupación media de la cola siempre al menor valor posible, ya que así el retardo que sufra cada paquete será menor.

Estudiemos con más detalle el funcionamiento de RED. Para su operación, se designan dos umbrales, un umbral mínimo (Min_{th}) y un umbral máximo (Max_{th}), tal y como se muestra en la Figura 4.1. Éstos son comparados con una estimación de la ocupación media de la cola (*Qavg*), que se calcula mediante un filtrado paso bajo (*moving average*) de los valores de ocupación instantáneos. Éste es un procedimiento similar, por ejemplo, al usado en la estimación del RTT (*Round Trip Time*) en el protocolo TCP

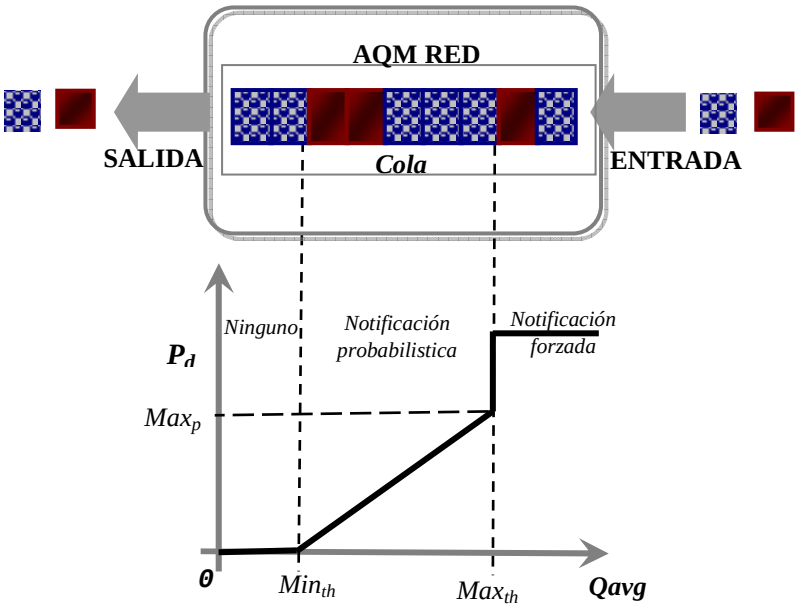


Figura 4.1 Probabilidad de notificación P_d en función de la ocupación media estimada de la cola *Qavg.*

[Postel, 1981b].

En la Figura 4.1 se muestra un ejemplo de curva que caracteriza el comportamiento de RED, representándose para cada valor de la ocupación media (*Qavg*) la correspondiente probabilidad de notificación (P_d). Nótese, que en general cuanto mayor es la ocupación media mayor será la probabilidad de notificación.

La ocupación media de la cola se estima cada vez que llega un paquete. Considerando los parámetros ilustrados en Tabla 4.1, el cálculo se realiza de la siguiente manera:

$$Qavg = (1 - wq)Qavg_{old} + wqQ_{inst} \qquad [4.1]$$

RED establece tres estados de operación que se determinan haciendo la comparación entre los dos umbrales y la *Qavg* estimada. El comportamiento de RED dependerá del resultado obtenido:

- *Cuando la ocupación media de la cola* (*Qavg, expresión [4.1]*) *es menor que* Min_{th}. Esta situación es considerada como el estado normal de operación. Aquí, ningún paquete es marcado o descartado, simplemente se encola.

Tabla 4.1.
PARÁMETROS ESTANDAR DEL MECANISMO RED: *RANDOM EARLY DETECTION*

Parámetros de RED	
Umbral mínimo en la cola	Min_{th}
Umbral máximo en la cola	Max_{th}
Máxima probabilidad de descarte de paquetes	Max_p
Ocupación media obtenida	$Qavg$
Ocupación media obtenida previamente	$Qavg_{old}$
Factor de peso para estimación de $Qavg$	wq
Ocupación instantánea en la cola	Q_{inst}
Probabilidad de notificación de paquetes	P_d

- *Cuando la ocupación media de la cola esté entre* Min_{th} *y* Max_{th} ($Min_{th} < Qavg < Max_{th}$) el mecanismo de protección de la congestión será implementado. Los paquetes que arriben serán notificados con una probabilidad (P_d), en función de la ocupación media de la cola ($Qavg$, Figura 4.1).
- *Cuando la ocupación media de la cola sea mayor que* Max_{th} ($Qavg > Max_{th}$) cada paquete recibido disparará con probabilidad 1 la notificación de congestión. Este evento se denomina notificación forzada.

La probabilidad de descarte de paquetes está dada por:

$$P_d = Max_p (Qavg - Min_{th}) / (Max_{th} - Min_{th}) \qquad [4.2]$$

En lo que sigue, supondremos que se adopta una notificación implícita (descarte de paquetes), no obstante, sin pérdida de generalidad todo es trasladable al caso de adoptar una estrategia de notificación explícita. La ejecución del algoritmo RED depende por lo tanto, de dos procedimientos diferentes, de cómo se estime $Qavg$ y de la probabilidad de descarte.

En RED tanto la probabilidad de descarte como el descarte forzado del paquete no dependerán en ningún momento del tipo de flujo al que pertenezca el paquete en cuestión, sino que sólo está determinada por el nivel de congestión que se encuentre en ese instante.

Una modificación a este diseño original introduce un modo "*gentle*" [Floyd, 2000] donde la probabilidad de descarte se incrementa lentamente entre Max_p y 1, mientras que la ocupación media de la cola varia entre Max_{th} y 2 Max_{th}.

Si bien, RED es considerado un buen mecanismo que previene la congestión, está especialmente indicado para operar sobre flujos TCP y no para UDP ya que RED provoca que las fuentes TCP ajusten sus tasas de transferencia de acuerdo al nivel de congestión existente, mientras que los flujos UDP no son capaces de responder. El algoritmo RED, por la propia naturaleza del protocolo UDP, no es tan eficaz para flujos multimedia en general y de audio en particular, ya que como es sabido se transmiten (por cuestiones de interactividad) usando el servicio no orientado a conexión de UDP.

4.3. AQM alternativos a RED

Si bien, el algoritmo *Random Early Detection* es uno de los más populares esquemas AQM, es conocido que éste tiene imperfecciones en su funcionamiento. El principal problema es que bajo algunas circunstancias RED puede permitir oscilaciones en la cola y reaccionar lento, puesto que su efectividad es altamente dependiente de la configuración de sus parámetros [May *et al*., 1999] y de la carga de tráfico [Lin and Morris, 1997]. Tomando en consideración este hecho, investigadores han generado una plétora de alternativos esquemas de Gestión Activa de Colas para mejorar RED y hacer frente a estas cuestiones.

Para proporcionar en detalle algunos de los algoritmos AQM más significativos bajo una estructura clara, se han clasificado en principio acorde al criterio sobre el cuál basan la probabilidad de descarte del paquete en la cola. Para esto, tres diferentes estrategias pueden ser identificadas, (1) Algoritmos basados en la ocupación media o instantánea de la cola, (2) Algoritmos basados en la tasa de pérdida de paquetes o tasa de tráfico de entrada (denominados algoritmos basados en tasas) y finalmente, (3) Algoritmos basados en la ocupación media de la cola y pérdida de paquetes ó la tasa de tráfico de entrada, es decir, una combinación de las dos primeras.

4.3.1. AQM basados en la ocupación de la cola

Esta clase de algoritmos AQM basan la probabilidad de descarte en observar la ocupación media o instantánea de la cola, con el objetivo principal de estabilizar el tamaño de la cola. RED y sus modificaciones pertenecen a esta categoría. Algunos de ellos, sólo tienen una modificación básica al diseño original de RED, como es el caso del algoritmo *Adaptive* RED (ARED) [Floyd *et al.,* 2001] y *Proportional-Derivative* RED (PD-RED) [Sun *et al.,* 2003a]. Otros algoritmos de esta categoría como *Flow Random Early Drop* (FRED) [Lin and Morris, 1997] y *Stabilized* RED (SRED) [Ott *et al*., 1999], intentan ser

más equitativos con los distintos flujos al distribuir el ancho de banda disponible. En cambio, el algoritmo *Dynamic -Class Based Thresholds* (D-CBT) [Chung and Claypool, 2000] pone mayor interés en proporcionar un trato diferenciado entre clases. Por otro lado, existe un número de diferentes algoritmos de esta clase que no están basados en el diseño original de RED como es el caso de *Fast and Autonomic Fuzzy AQM Controller* (FAFC).

El principal inconveniente de estos AQM es que crean innecesarios retardos en cola y *jitter* de encolamiento así como bajo rendimiento de salida.

Flow Random Early Drop (FRED)

El algoritmo FRED (*Flow Random Early Drop*) [Lin and Morris, 1997] fue propuesto para paliar el efecto de falta de equidad del esquema RED. Este algoritmo protege flujos frágiles de otros flujos más agresivos (que tienden a monopolizar la cola y consumir todo el ancho de banda disponible) imponiendo una probabilidad de descarte que depende de la ocupación de la cola de cada uno de los flujos activos. El algoritmo intenta proporcionar igual ancho de banda manteniendo sólo información de los flujos que tengan paquetes en la cola.

FRED trabaja de forma similar a RED, sólo difiere en algunos aspectos, introduce cinco nuevos parámetros para cada flujo: Min_f, Max_f, $Qavg_f$, qf_{len}, *strike*. Estos parámetros identifican la ocupación máxima y mínima en la zona de notificación probabilística, la ocupación media de paquetes por flujo, la cantidad instantánea de paquetes del flujo en la cola y el número de ocasiones que el flujo no responde a la notificación de congestión respectivamente. FRED calcula *Qavg* por cada paquete recibido si la cola está vacía, en caso que esté ocupada, se hace el cálculo hasta que el paquete sea admitido. Además, por cada paquete que sale de la cola se calcula *Qavg* y $Qavg_f$.

FRED penaliza un paquete si cree que éste pertenece a un flujo agresivo. Se considera un flujo agresivo si

$$qf_{len} \geq Max_f \;\|\; (QAvg \geq Max_{th} \;\&\&\; qf_{len} > 2QAvg_f) \;\|\; qf_{len} \geq QAvg_f \;\&\&\; strike > 1) \qquad [4.3]$$

Este esquema acepta preferentemente paquetes de flujos con pocos paquetes en la cola, permitiendo un valor Min_f de paquetes a cada conexión sin pérdidas. De ahí que los flujos considerados frágiles serán descartados de forma probabilística sólo cuando

$$qf_{len} \geq MAX(Min_f, Qavg_f) \qquad [4.4]$$

Stabilized RED (SRED)

La idea básica detrás del algoritmo *Stabilized* RED (SRED) [Ott *et al*., 1999] es reducir las fluctuaciones de la cola, es decir estabilizar la ocupación de la cola a un nivel dado, independientemente del número de flujos activos TCP. Este algoritmo difiere de RED en el sentido de que no computa la ocupación media de la cola, por lo que la probabilidad de descarte depende solamente de la ocupación instantánea y del número estimado de flujos activos. Principalmente, el algoritmo identifica qué flujos TCP están tomando más ancho de banda de la porción que les corresponde y los penaliza para asignar así en forma más equitativa el ancho de banda entre todos los flujos.

Para su operación, SRED utiliza una pequeña lista de tamaño *M* denominada *Zombie* donde básicamente cada entrada en la lista contiene el identificador del flujo y dos variables asociadas, la variable denominada *count* almacena un valor y la segunda denominada *timestamp* un tiempo determinado. De tal manera, tan pronto llega un paquete a la cola, si la lista no esta llena, identifica el flujo del paquete y lo agrega a la lista. De igual forma, la variable *count* de ese flujo es establecida a cero y el *timestamp* es establecido con el tiempo de llegada del paquete. En caso de que la lista esté llena, el algoritmo compara el paquete que va llegando con un paquete que ha seleccionado en forma aleatoria de los que previamente han llegado a la cola. Cuando los dos paquetes son del mismo flujo, se declara un "*hit*". Es decir, la variable *count* del flujo es incrementada en uno y el *timestamp* es establecido al tiempo de llegada del paquete encolado. Por otra parte, si los dos paquetes no son del mismo flujo, se declara un "*no hit*". En este caso la variable *count* del flujo seleccionado de la lista es establecida a cero y el *timestamp* es establecido con el tiempo de llegada del nuevo paquete. La secuencia de *hits* es utilizada en dos sentidos y con dos diferentes objetivos: estimar el número de flujos activos y encontrar candidatos de flujos que no estén actuando correctamente. Si un paquete causa "*hit*" es evidente que su flujo puede ser el causante de la congestión. Aún más, si el valor de la variable *count* de ese flujo es alto, la evidencia tiende a ser más sólida.

De ahí que la probabilidad de descartar el paquete P_d depende de

$$Hit(t) = \begin{cases} 0, si\ "nohit" \\ 1, si\ "hit" \end{cases}$$

Esto permite que, si B es la capacidad de la cola y $0<\alpha<1$, las funciones $P_{sred}(q)$ y $P(t)$ se definan como

$$P(t) = (1-\alpha)P(t-1) + \alpha Hit(t) \quad [4.5]$$

$$P_{sred}(q) = \begin{cases} p_{\max}, \ si \frac{1}{3}B \le q < B, \\ \frac{1}{4} * p_{\max}, \ si \frac{1}{6}B \le q < \frac{1}{3}B, \\ 0, \ si 0 \le q < \frac{1}{6}B \end{cases} \quad [4.6]$$

Finalmente, la probabilidad de descarte es dada por

$$P_d = P_{sred}(q) * \min(1, \frac{1}{(256 * P(t))^2}) \quad [4.7]$$

Adaptive RED (ARED)

ARED (*Adaptive* RED) [Floyd *et al.*, 2001] se propuso para resolver el problema de la alta sensibilidad de los parámetros en RED, ajustando la probabilidad de descarte de acuerdo con las condiciones del tráfico. El algoritmo ARED utiliza la tasa media de paquetes encolados como un indicador de la congestión con el objetivo de permitir una alta utilización del ancho de banda disponible, reducir la tasa de pérdidas y simultáneamente proporcionar bajo retardo medio de encolamiento. Por lo que, adapta el parámetro RED Max_p en función de los cambios en el tamaño de la cola y configura automáticamente los parámetros *wq* y Max_{th}.

Para conservar la ocupación media de la cola entre un rango objetivo (entre Min_{th} y Max_{th}) y reducir la sensibilidad de los parámetros RED, el algoritmo *Adaptive* RED (Figura 4.2) difiere de RED en el sentido de que utiliza la política de Incrementos Aditivos y Decrementos Multiplicativos (AIMD, Additive-increase Multiplicative-decrease) para adaptar Max_p. Es decir, incrementa Max_p aditivamente cuando la ocupación media de la cola está por debajo del Min_{th} y reduce Max_p en forma multiplicativa cuando la ocupación media de la cola está por encima de Max_{th}.

La adaptación de Max_p es lenta e infrecuente y está restringida a no reducirse por debajo del 1% de la probabilidad de pérdida de paquetes y a no excederse del 50% de ésta misma probabilidad.

Cada *intervalo* de segundo:
Si ($Qavg > target$ y $Max_p \leq 0{,}5$)
  Incrementa Max_p :
  $Max_p \leftarrow Max_p + \alpha$;
Si ($Qavg < target$ y $Max_p \geq 0{,}01$)
  Reduce Max_p;
  $Max_p \leftarrow Max_p * \beta$

Variables:
Qavg : ocupación media de la cola

Parámetros fijos:
intervalo: tiempo; 0,5 segundos
target: referencia para *Qavg*;
  $[Min_{th} + 0{,}4 * (Max_{th} - Min_{th}),$
  $Min_{th} + 0{,}6 * (Max_{th} - Min_{th})]$.
α: factor de incremento; min (0,01, Max_p /4)
β: factor de reducción; 0,9

Figura 4.2. Pseudo-código de *Adaptive* RED.

[Kim and Lee, 2006] y [Seol *et al*., 2006] incorporan cambios al esquema ARED y respectivamente establecen las alternativas denominadas *Refined Adaptive* RED (RARED) y *Queue Variation Adaptive* RED (QVARED). Específicamente, las dos implementaciones afinan Max_p utilizando las variaciones de ocupación media de la cola registradas.

Dynamic -Class Based Thresholds (D-CBT)

Dynamic -Class Based Thresholds (D-CBT) [Chung and Claypool, 2000] es una variante de RED que detecta la congestión clasificando y aplicando diferentes políticas según el tipo de flujo al que pertenece el paquete. D-CBT clasifica los flujos en tres clases, TCP, UDP (multimedia) y otros UDP. Asumiendo que los flujos UDP no responden (por definición no incorporan mecanismos de control de congestión) a las notificaciones de congestión, en esta aproximación se calcula un determinado umbral para cada una de las dos clases de flujos UDP considerados. Los umbrales que se establecen determinan el número máximo de paquetes en la cola que cada clase puede almacenar durante la congestión. De esta manera, el mecanismo D-CBT proporciona una mejor utilización de recursos en cola para el tráfico (clase) UDP que para la otra, proporcionando así un

tratamiento diferenciado.

En la Figura 4.3 se muestra el diseño de un gestor de cola D-CBT. Cuando un paquete arriba, éste es clasificado con una de las tres posibles etiquetas: TCP, UDP (multimedia) u otros UDP. El flujo es contabilizado de acuerdo con la clase a la que pertenezca en el sistema denominado *contador de flujos activos por clase*. El contador, además de registrar la cantidad de flujos, actualiza una estructura de datos por cada clase, teniendo en cuenta el tiempo local por cada paquete recibido. Cada una de las estructuras de datos mantiene ordenadamente información de los flujos (identificador de la fuente y dirección IP destino) y el tiempo de recepción del último paquete. Posterior a la actualización de la estructura de datos por clase, D-CBT actualiza la ocupación media de la cola (*Qavg*) de RED, la estimación de la media de los paquetes etiquetados como UDP (multimedia) y la estimación de la media de los paquetes etiquetados como otros UDP. Para cada uno de los paquetes UDP que llegan, la estimación de las medias es actualizada usando el mismo factor de peso.

Cuando la cola RED indica que es necesaria la prevención de la congestión (*Qavg* > Min_{th}), se envía el paquete UDP a la unidad de prueba de umbral. Este mecanismo de prevención de la congestión contiene dos unidades de cálculo de umbral (ver Figura 4.3), una para cada clase UDP, en la que el umbral es estimado (antes de pasar a la unidad de prueba) utilizando la media correspondiente y la ocupación media de la cola (*Qavg*) de RED previamente obtenidas, además de la información del contador de flujos activos por clase. Según sea la clase del paquete (UDP u otros UDP), la media de la cola correspondiente es comparada (prueba de umbral) con el umbral calculado para esa clase

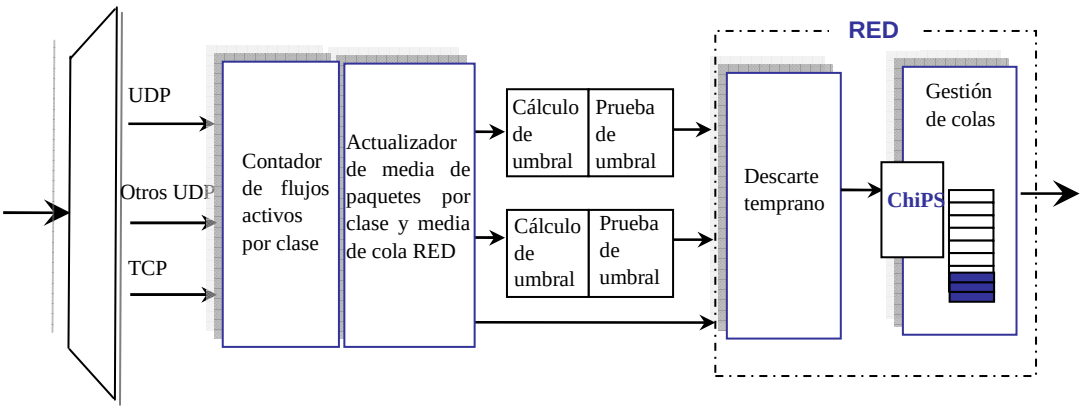


Figura 4.3. Modelo *Dynamic* -CBT.

y decide si se descarta o no el paquete antes de pasar al algoritmo de RED. Para la clase de TCP, no se aplica una comparación con un especificado umbral, por lo tanto, el paquete pasa directamente al algoritmo RED. Los paquetes UDP que sobreviven a la prueba del umbral apropiado, son pasados a la unidad de RED junto con los flujos TCP.

Proportional-Derivative RED (PD-RED)

[Sun *et al.,* 2003a] introducen mínimos cambios a RED y proponen un esquema denominado PD-RED (*Proportional-Derivative* RED), con el objetivo de lograr una mayor utilización de recursos. Basándose en la teoría de control (en el principio de Control Proporcional-Derivado), PD-RED intenta conservar estabilizada la ocupación de la cola cerca de un valor de referencia entre Min_{th} y Max_{th}.

Para ese propósito, el algoritmo adapta la máxima tasa de descarte de RED (Max_p) utilizando un controlador Proporcional-Derivado (Figura 4.4) donde específicamente el tamaño de la cola en el *router* es regulada al valor de referencia q_{ref}. Este controlador considera la magnitud del error de *Qavg* para adaptar la Max_p, es decir, la diferencia entre la ocupación de la cola Q_{avg} y el valor de referencia $q_{ref.}$

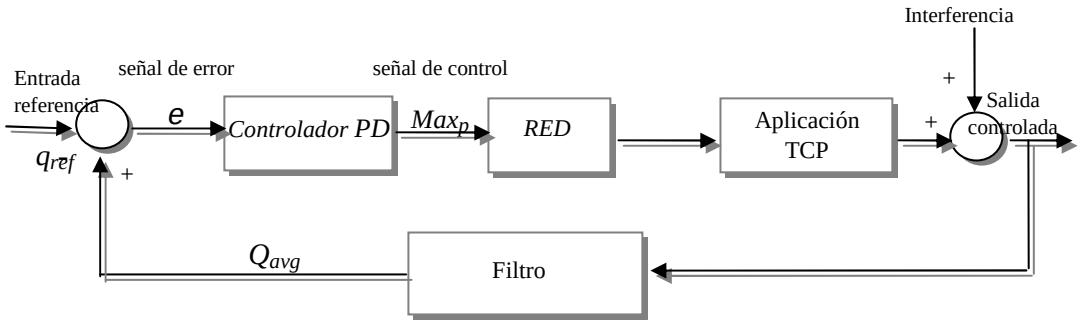


Figura 4.4. Diagrama de mecanismo de control de la congestión TCP como un sistema de control de retroalimentación cerrado.

Penalizing Unresponsive Flows with Stateless Information (PUNSI)

Un algoritmo más reciente llamado PUNSI (*Penalizing Unresponsive flows with Stateless Information*) [Yamaguchi and Takahashi, 2007], ha sido desarrollado con el fin de prevenir que flujos no-reactivos dominen el ancho de banda disponible que comparten con los flujos reactivos. PUNSI es un algoritmo de baja complejidad enfocado a proteger los flujos reactivos sin almacenar información de los flujos que tengan paquetes en la cola.

Cuando un paquete llega a la cola, PUNSI verifica el nivel de congestión de la misma forma que RED, y difiere de éste sólo cuando la ocupación media de la cola esté entre Min_{th} y Max_{th} ($Min_{th} < Qavg < Max_{th}$). Por lo que, contrario a RED, PUNSI adopta en esta situación dos diferentes estrategias de descarte dependiendo del tipo de tráfico de que se trate (UDP o TCP). Si el paquete que ha llegado a la cola es TCP, el algoritmo utiliza la misma estrategia que en RED. Sin embargo, si el paquete es UDP, el siguiente mecanismo de protección será implementado. Primeramente, siguiendo una distribución geométrica, otro paquete es seleccionado de la cola. Y en segundo término, ejecuta una comparación entre paquetes. Si el paquete seleccionado de la cola pertenece al mismo flujo que el paquete que va llegando, los dos paquetes son descartados. En caso de ser diferentes, el paquete que va llegando es admitido y ningún paquete es descartado. De ahí que PUNSI es más agresivo con los flujos no-reactivos, permitiendo otorgar más ancho de banda a los flujos reactivos.

Fast and Autonomic Fuzzy AQM Controller (FAFC)

Además de los métodos basados en el diseño original de RED, existen diferentes mecanismos (no fundamentados en RED) que basan también la probabilidad de descarte en la ocupación de la cola, como por ejemplo, FAFC (*Fast and Autonomic Fuzzy AQM Controller*) [Aoul *et al.*, 2007]. La idea detrás de FAFC es minimizar la fluctuación de la cola, optimizar el rendimiento sin importar la variación de carga de tráfico y la presencia de tráfico no-reactivo y aún más, otorgar el mejor compromiso entre la utilización del enlace y el retardo en cola, basándose en un robusto algoritmo de lógica difusa y un mecanismo auto-configurado.

A diferencia de los métodos de control convencionales, este algoritmo se centra en la estructura de un controlador que utiliza un modelo de sistema. Tal como se muestra en la Figura 4.5, se adoptan dos entradas del mecanismo de control proporcional-derivado difuso (MISO, *Multi Inputs Single Output*). La primera entrada (el valor proporcional)

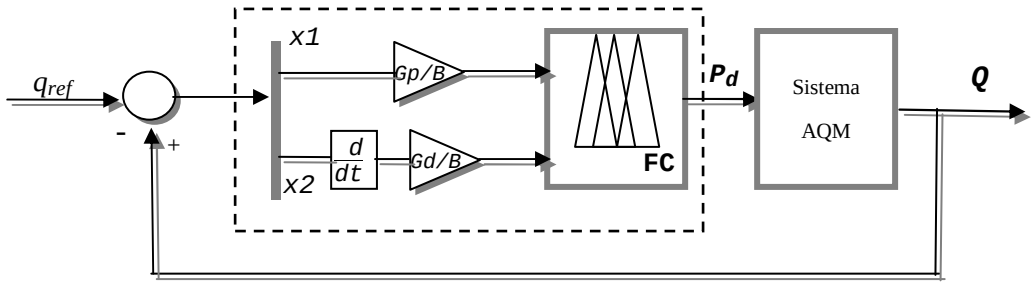


Figura 4.5. Modelo cerrado de control de colas para gestión de la congestión.

determina la reacción del error actual, el error *x1* (la diferencia entre la ocupación instantánea Q_{inst} y el tamaño de cola de referencia q_{ref}) y la segunda (el derivativo) determina la reacción del tiempo en el que el error se produce, el error *x2*. La ocupación máxima de la cola *B* es utilizada para normalizar los valores *x1* y *x2*. Por otra parte, los parámetros *Gp* y *Gd* representan la ganancia proporcional y derivada respectivamente.

FAFC establece nueve funciones agrupadas para cada entrada. Las reglas básicas para un AQM difuso utilizando esas funciones son mostradas en la Figura 4.6.

X1 \ X2	0	1	2	3	4	5	6	7	8
0-3	0	0	0	0	0	0	0	0	0
4	0	1	2	3	4	5	5	6	6
5	4	4	5	5	5	6	6	7	7
6	4	5	6	6	6	7	7	7	8
7	5	6	7	7	7	8	8	8	8
8	6	7	8	8	8	8	8	8	8

Figura 4.6. Reglas básicas para un controlador difuso.

Donde la probabilidad de descarte del paquete está determinada empíricamente excepto para los valores 0 (más bajo), 4(medio) y 8 (más grande), los cuáles corresponden respectivamente al mínimo, estable y máxima probabilidad de descarte. Así, de ahí se deducen los valores:{0=0, 1=0,003, 2=0,008, 3=0,01, 4=0,02, 5=0,06, 6=0,12, 7=0,2 y 8=1} para la salida de la probabilidad de descarte del paquete, la variable P_d.

4.3.2. AQM basados en medida de tasas

Alternativamente a utilizar la información de la ocupación media o instantánea de la cola, otros algoritmos AQM detectan o previenen la congestión basándose estratégicamente en la tasa de entrada del tráfico o en la tasa de pérdidas de paquetes. Esta clase de algoritmos mantiene una única probabilidad que es utilizada para descartar el paquete cuando éste es encolado. Los AQM basados en tasas son más rápidos para reaccionar a la congestión por lo que intentan reducir la diferencia de la tasa entre encolamiento y des-encolamiento, lograr baja tasa de pérdidas, bajo retardo y mayor utilización del enlace. *Adaptive Virtual Queue* (AVQ) [Kunniyur and Srikant, 2001], *Stochastic Fair Blue* (SFB) [Feng *et al.*, 2002], GREEN [Wydrowski and Zukerman, 2002], *CApture-REcapture fair sharing* (CARE) [Chan and Hamdi, 2003] y R*ate-based* AQM (RAQM) [Wang *et al*., 2005] son algunos de los más representativos AQM que pertenecen a esta clase.

Adaptive Virtual Queue (AVQ)

Adaptive Virtual Queue (AVQ) [Kunniyur and Srikant, 2001] remplaza calcular la probabilidad de descarte del esquema RED -observando la ocupación media de la cola - para imponer una probabilidad que se basa en la capacidad de una cola virtual controlada por una utilización esperada del enlace. Es decir, la capacidad de la cola virtual implícitamente impone la probabilidad de descarte de los paquetes en la cola real. Este AQM minimiza el retardo en la cola.

Para este propósito, AVQ mantiene una cola virtual cuya capacidad o velocidad de salida *Cv* es menor que la capacidad del enlace real *C*. Por cada paquete que llega a la cola real, la cola virtual es también adaptada para reflejar una nueva llegada. La capacidad de la cola virtual es adaptada entonces mediante

$$Cv = \alpha(\gamma C - \chi) \qquad [4.8]$$

Donde χ es la tasa de llegada en el enlace, α el factor de atenuación y γ la utilización esperada del enlace. Si el nuevo paquete satura la cola virtual, entonces el paquete es descartado en la cola virtual y el paquete real es marcado o descartado en la cola real, dependiendo del mecanismo de notificación utilizado en el *router*. Cuando la utilización del enlace excede el valor esperado, el algoritmo será más agresivo e inversamente menos agresivo cuando la utilización del enlace permanece por debajo de la utilización esperada.

Sin embargo, no es necesario encolar o desencolar los paquetes en la cola virtual, es posible que se lleve a cabo a través de un *token bucket* donde cada *token* es generado a la

tasa $\alpha\gamma C$ y reducirse por cada llegada de paquete en una cantidad igual al tiempo α. La Figura 4.7 describe como el algoritmo AVQ realiza esta implementación. Donde *B* expresa la ocupación de la cola, *s* el tiempo de llegada del paquete previo, *b* el número de *bytes* del paquete que va llegando y finalmente, *VQ* indica el número de *bytes* registrados en la cola virtual.

Por cada paquete que llega:
VQ ← *max (VQ- CV(t-s),0)* Adapta el tamaño de la cola virtual
Si *VQ + b > B*
Marca o descarta el paquete en la cola real
En caso contrario
VQ ← *VQ + b* Adapta el tamaño de la cola virtual.

*Cv=max(min(Cv+α*γ* C(t-s),C)- α*b,0)* Adapta la capacidad de cola virtual

s ← *t* Adapta el tiempo de llegada del último paquete.

Figura 4.7. Pseudo-código de AVQ.

El algoritmo AVQ ha sido mejorado por los propios autores en un trabajo posterior en [Kunniyur and Srikant, 2004].

Existen algunas alternativas desarrolladas para reforzar aspectos específicos de esta implementación. Por ejemplo, con el fin de estabilizar la dinámica de la cola a la vez que se mantiene una alta utilización del enlace, se ha desarrollado *Stabilized Adaptive Virtual Queue* (SAVQ) [Long *et al*., 2005a]. Por su parte, *Enhanced Adaptive Virtual Queue* (EAVQ) [Yanping *et al*., 2007] intenta resolver problemas como la rigidez al configurar parámetros, la pobre habilidad de estabilidad y la capacidad de pérdidas del enlace. A diferencia de AVQ estándar, estas dos implementaciones incorporan un cambio en el método para obtener γ (la utilización esperada del enlace).

Stochastic Fair Blue (SFB)

Para remediar algunos problema de RED, el algoritmo SFB (*Stochastic Fair Blue*) [Feng *et al.*, 2002] gestiona la congestión observando la pérdida de paquetes y la utilización del enlace en lugar de la ocupación de la cola. La idea principal detrás de SFB consiste en detectar flujos agresivos (flujos que no reaccionan a la congestión) y limitar su tasa de transmisión para proteger a los flujos frágiles (flujos TCP). Para este propósito, SFB mantiene *N x L* áreas de registro que están organizados en *L* niveles con *N* número de

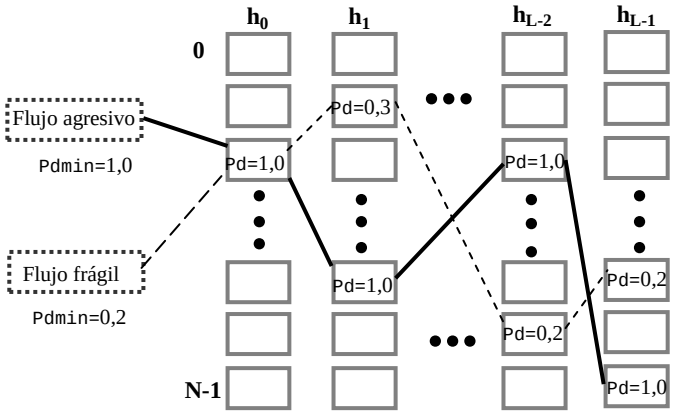


Figura 4.8. Ejemplo de SFB.

áreas por nivel. Además mantiene *L* funciones *hash* independientes, cada una asociada con un nivel de área de registro (Figura 4.8). Cada función *hash* dirige un flujo según su ID de conexión a alguna de las *N* áreas de registro en cada nivel. Las áreas son utilizadas para conservar un registro de la ocupación de la cola de los paquetes pertenecientes a un flujo en particular. Cada área de registro conserva una probabilidad de descarte P_d, la cual es adaptada basándose en su ocupación.

Tan pronto como un paquete llegue a la cola, este es distribuido dentro de una de las *N* áreas de registro en cada uno de los *L* niveles. Si el número de paquetes registrados en una área sobrepasa un cierto umbral, P_d es incrementado para esa área, en caso de que el número de paquetes descartados en la área sea cero, P_d se reduce. Si P_d tiende rápidamente a 1, significa que es un flujo agresivo. La decisión de descartar un paquete se basa en P_{dmin}, el mínimo valor de P_d de todas las áreas en que el flujo fue registrado. Si P_{dmin} es 1, el paquete es identificado como que pertenece a un flujo agresivo y por tanto su tasa se limita (Figura 4.8).

GREEN

La idea detrás del algoritmo GREEN [Wydrowski and Zukerman, 2002] es mantener una tasa de pérdidas baja y minimizar los retardos de encolamiento. Para este fin, el algoritmo ajusta la probabilidad de descarte de acuerdo a una tasa estimada de entrada del tráfico denominada X_{est} y en base a la capacidad esperada del enlace, C_e.

La tasa de entrada es obtenida mediante

$$X_{est} = (1 - \exp(-Del / K)) * (B / Del) + \exp(-Del / K) * X_{est} \quad [4.9]$$

Donde *Del* es el retardo entre paquetes, *B* el tamaño del paquete y la variable *K* una constante de tiempo. La capacidad de C_e es asignada con un valor menor que la capacidad del enlace real *C*.

De tal manera, GREEN ajusta la tasa de notificación de la congestión incrementándola o reduciéndola en función de la capacidad esperada del enlace y la tasa estimada de entrada del tráfico. Es decir, si X_{est} es superior a C_e la tasa de notificación de la congestión P_d es incrementada por ΔP_d a una tasa de $1/\Delta T$. Inversamente, si X_{est} es inferior a C_e, P_d es reducida por ΔP_d a una tasa de $1/\Delta T$. Por lo que, el algoritmo GREEN aplica descartar o marcar en forma probabilística el paquete que va llegando a una tasa

$$P_d = P_d + \Delta P_d - U(X_{est} - C_e) \quad [4.10]$$

Donde, $U(x)=+1$ *cuando* $x \geq 0$ y $U(x)=-1$ *cuando* $x<0$.

CApture-REcapture fair sharing (CARE)

El algoritmo CARE (*CApture-REcapture fair sharing*) [Chan and Hamdi, 2003] es una técnica distinta utilizada para proporcionar ancho de banda equitativo entre flujos, estimando el número de flujos en la cola y la tasa de envío de flujos individuales basándose en el modelo Captura-Recaptura (CR). El modelo CR simplemente captura/recaptura en forma aleatoria los paquetes que van llegando a la cola. En lugar de explorar en completo la cola para encontrar el n_f número de flujos existentes, este algoritmo captura en una lista y en una cola virtual, con probabilidad P_{cap}, un paquete que va llegando para obtener un grupo de identificadores de flujos (*IDs).* Observando el grupo de *IDs* de los paquetes, construye el dato de frecuencia de captura "*x*" de acuerdo a los flujos capturados en la lista. Donde finalmente, este grupo es usado para estimar el número total de flujos en la cola n_f apoyándose del estimador *jackknife*.

La probabilidad de capturar en forma aleatoria un paquete del flujo *i* en la cola virtual, está dado por

$$P_{cap\,i} = x_i / (x_1 + x_2 + ... + x_n) \quad [4.11]$$

donde *n* refleja los flujos en la cola y x_1 representa el número de paquetes en la cola del flujo con *ID* 1, por lo que la cola debe contener $x_1 + x_2 + \ldots + x_n$ paquetes. Como la tasa

de datos de los diferentes flujos es diferente, la probabilidad de captura varía por flujo.

La tasa de envío de un cierto flujo es representada por la cantidad de paquetes del flujo en la cola virtual. CARE estima el número de paquetes pertenecientes a un flujo (n_{pf}) utilizando nuevamente el modelo CR. Para este propósito captura con probabilidad P_{cap}, n_{tp} paquetes de la cola. Posteriormente, cuenta en la lista de captura el número de paquetes "*x*" con *ID* del flujo en particular, y el estimado n_{pf} es calculado si el tamaño de la cola virtual *B* es dado. En definitiva

$$n_{pf} = B*(x/n_{tp}) \qquad [4.12]$$

Para evitar tener múltiples capturas para las estimaciones, los paquetes guardados en la lista de captura son utilizados tanto para la estimación de equidad como para la estimación de tasas de datos. Ahora, para proporcionar igual ancho de banda, cada porción *S* correspondiente a cada flujo es estimada utilizando por consiguiente

$$S = B/n_f \qquad [4.13]$$

Finalmente, la probabilidad de descarte está dada por

$$P_d = 1-(S/n_{pf}) \qquad [4.14]$$

Rate-based AQM (RAQM)

Para obtener estabilidad y rápida respuesta a la congestión bajo distintas condiciones de la red, [Wang *et al.*, 2005] proponen la implementación denominada RAQM (*rate-based* AQM). Este algoritmo calcula la probabilidad de descarte basándose en el exceso de tasa de entrada del tráfico. RAQM tiene dos formas distintas de operación, el modo independiente de la ocupación de la cola y el modo dependiente de la ocupación de la cola.

En el modo independiente de la cola, RAQM solamente mide periódicamente la tasa de entrada del tráfico (r_k) utilizando

$$r_k = (1-e^{-T/f})\frac{l_k}{T} + e^{-T/f} r_{k-1} \qquad [4.15]$$

donde *T* es el período en que se lleva acabo la medición, l_k el total de *bytes* que van llegando durante el período *k* y *f* es una constante que influirá en la exactitud del resultado. Específicamente, $e^{-T/f} < 1$.

Con r_k, interactivamente adapta la probabilidad de descarte del paquete de acuerdo a una regla exponencial como,

$$P_{dk} = e^{\alpha(r_k - r_0)} P_{dk-1} \quad [4.16]$$

donde $\alpha>0$ y r_o es la tasa esperada de entrada del tráfico. r_o puede ser establecida a un valor igual a la capacidad del enlace C ó como γC (donde $0<\gamma\leq 1$).

En el modo dependiente de la cola, RAQM utiliza conjuntamente el tamaño instantáneo de la cola para refinar la probabilidad de descarte cada vez que llega un paquete, de esta manera regula el tamaño de la cola a un valor esperado. Por lo que,

$$P_d = \max(0, \min(1, P_{dk} \frac{Q_{inst}}{q_0})) \quad [4.17]$$

donde Q_{inst} es el tamaño instantáneo de la cola y q_o es el esperado tamaño de la cola. En particular, con este segundo modo de operación, RAQM se puede clasificar dentro de la clase de algoritmos AQM basados en ocupación de la cola y medida de tasas.

4.3.3. AQM basados en ocupación de la cola y medida de tasas

Combinando los métodos anteriores se identifica un nuevo tipo de esquemas AQM. Más precisamente, estos esquemas monitorean la ocupación instantánea de la cola en adición a la tasa de entrada del tráfico, la medida de tasa de pérdida o la medida de retardos, para determinar la probabilidad de descarte del paquete. Algunos de los ejemplos más representativos de esta clase de esquemas AQM son *Random Exponential Marking* (REM) [Athuraliya *et al*., 2000], RED-Boston [Phirke *et al.* 2002], *Loss Ratio Based RED* (LRED) [Wang *et al.,* 2004], *Yellow* [Long *et al.,* 2005b], *Jitter Detection* (JD) [Chan *et al.,* 2005] y RaQ [Sun and Zukerman, 2007].

Random Exponential Marking (REM)

REM (*Random Exponential Marking*) [Athuraliya *et al*., 2000] utiliza una probabilidad exponencial para ajustar el tamaño de la cola y minimizar el retardo de encolamiento, definiendo primeramente una función de precio que se basa en la información de la cola y en la tasa de llegada de paquetes.

Este algoritmo puede estabilizar la tasa de entrada a la capacidad del enlace y el tamaño de la cola a un valor pequeño independientemente del número de usuarios que

estén compartiendo el enlace. En definitiva, la función de precio para un enlace en un período *t* es adaptada de acuerdo a

$$p(t+1) = [\, p(t) + \gamma(\alpha(q(t) - q_{ref}) + y(t) - C]^{+} \qquad [4.18]$$

Donde γ y α son constantes mayores a cero y *q(t)* y q_{ref} son respectivamente el tamaño de la cola y el valor objetivo. Correspondientemente, la probabilidad de descartar es dado por

$$P_d(t) = 1 - \phi^{-p(t)} \qquad [4.19]$$

cuando la constante ϕ >1.

RED-Boston

RED-Boston [Phirke *et al.* 2002] es un mecanismo de detección de la congestión desarrollado para satisfacer requerimientos de QoS a todas las aplicaciones introduciendo en sus paquetes un retardo sugerido. Este algoritmo conserva la estructura básica de RED y ajusta la Max_p para mantener una ocupación media de la cola en un objetivo determinado (entre el Min_{th} y el Max_{th}) basándose en un suavizado exponencial de los valores instantáneos de retardos sugeridos. El retardo sugerido indica el grado de retardo que la aplicación puede tolerar sin afectar significativamente su rendimiento. RED-Boston no utiliza el retardo sugerido para garantizar un retardo en cola. Lo usa para calcular la media de requerimientos de QoS que necesita el paquete en ella. Este mecanismo proporciona relativamente un menor retardo extremo-a-extremo a los flujos sensibles a él y un relativo incremento de rendimiento a los flujos sensibles a este parámetro.

Para cada paquete recibido, RED-Boston lee la etiqueta que contiene el retardo sugerido (R_{ref}) y calcula el objetivo de ocupación media que éste puede tolerar, el cual es definido por

$$O_{avg} = (1 - w_f) * O_{avg} + w_f * C * R_{ref} / P \qquad [4.20]$$

Donde, *C* representa la capacidad de la línea, *P* el tamaño del paquete y finalmente, w_f el factor de peso para la estimación. De esta forma RED-Boston mantiene la ocupación media de la cola a un objetivo tomando en consideración los requerimientos de los paquetes que van llegando. Otra de las características de este mecanismo es que cuando O_{avg} se encuentra entre el Min_{th} y el Max_{th} ajusta la probabilidad de descarte basándose

en el retardo de encolamiento correspondiente a la ocupación media de la cola (R_q) y en el retardo sugerido del paquete, adaptando

$$P_d = P_d * R_q / R_{ref} \quad [4.21]$$

De tal manera que RED-Boston determina que un paquete tiene alta probabilidad de descarte si trae especificado un bajo retardo sugerido, como puede ser el caso de un paquete de audio. Mientras que un paquete FTP con alto retardo sugerido, tendrá una baja probabilidad de descarte. Este mecanismo de detección de la congestión no sólo ajusta la ocupación media de la cola y la probabilidad de descarte, si no que además, establece cómo será la salida de los paquetes de la cola.

Cada paquete que llega y que no es descartado, es ordenado en la cola considerando un peso, el cual es calculado utilizando

$$W = tiempo\ \ de\ \ llegada + R_{ref} \quad [4.22]$$

En definitiva, si el paquete tiene marcado un bajo retardo sugerido, es insertado al frente de la cola. De otra manera, un paquete con alto retardo sugerido es colocado en la parte final de la cola. Es decir, un paquete con un vencimiento más temprano puede ser servido primero que cualquier otro.

Loss Ratio Based RED (LRED)

Para regular el tamaño de la cola a un valor esperado, el AQM LRED (*Loss Ratio Based RED*) [Wang *et al.,* 2004] ajusta dinámicamente la probabilidad de descarte del paquete. Para lo cual, mide la tasa de pérdidas de paquetes y la utiliza junto con la ocupación instantánea de la cola. LRED calcula la tasa de pérdidas periódicamente como

$$lp(k) = lp(k-1) * mw + (1-mw) * l(k) \quad [4.23]$$

utilizando,

$$l(k) = \frac{\sum_{i=0}^{M-1} N_d(k-i)}{\sum_{i=0}^{M-1} N_a(k-i)} \quad [4.24]$$

Donde $l(k)$ es la tasa de paquetes perdidos durante los últimos *M* períodos, *mw* es el peso para estimar la media, $N_d(k)$ es el número de paquetes descartados en el período *k-th*

y *Na(k)* es el número de paquetes que llegan en el período *k-th*.

Teniendo la tasa de pérdidas, el algoritmo adapta la probabilidad de descarte cada vez que llega un paquete estableciendo

$$P_d = lp(k) + \beta\sqrt{lp(k)}(Q_{inst} - q_{ref}) \qquad [4.25]$$

Donde el parámetro β>0 es una constante preestablecida, Q_{inst} el tamaño de la cola y q_{ref} el valor esperado.

En definitiva, LRED designa dos reglas para su funcionamiento:1) Cuando Q_{inst} = q_{ref}, la probabilidad de descarte P_d será igual a la tasa de paquetes perdidos $lp(k)$ y 2) P_d mayor o menor que $lp(k)$, cuando $Q_{inst} > q_{ref}$ o $Q_{inst} < q_{ref}$ respectivamente.

Yellow

Para mejorar el rendimiento de control de congestión, el algoritmo *Yellow* es propuesto en [Long *et al.*, 2005b]. Este algoritmo garantiza un buen funcionamiento en términos de utilización del enlace utilizando simultáneamente un factor de carga y una función de control de cola. El AQM *Yellow* monitorea periódicamente la carga de cada enlace y adapta la función de control de la cola utilizando,

$$f(q) = \begin{cases} \max(QDLF, \dfrac{\gamma\alpha q_{ref}}{(\alpha-1)Q_{inst} + q_{ref}}) & for\ \ Q_{inst} > q_{ref}, \\ \dfrac{\gamma\beta q_{ref}}{(\beta-1)Q_{inst} + q_{ref}} & for\ \ 0 \le Q_{inst} \le q_{ref} \end{cases} \qquad [4.26]$$

Donde Q_{inst} determina el tamaño de la cola, *QDLF* es el factor que determina la reducción máxima de la cola, q_{ref} es el tamaño de la cola de referencia, α y β son los parámetros que tienen efecto sobre la tasa de reducción de la cola. Y finalmente, γ es el factor de utilización del enlace.

Con la función de control de la cola *f(q)* y C indicando la capacidad del enlace, el algoritmo adapta la capacidad virtual (*cv*) disponible, en particular

$$cv(q) = f(q) * C \qquad [4.27]$$

Por consiguiente, el factor de carga (*z)* es calculado y la probabilidad de descarte del paquete es adaptada,

$$z = (Tasa\ \ de\ \ entrada\ \ enlace) / cv \qquad [4.28]$$

$$p_d = \begin{cases} p_d + z\Delta / C & si\ \ z \geq 1 + \delta, \\ p_d - \Delta / (zC) & si\ \ z < 1, \\ p_d & otros \end{cases} \qquad [4.29]$$

De esta manera, *Yellow* responde rápido a la variación entre la tasa de entrada y la capacidad del enlace.

Jitter Detection (JD)

Jitter Detection (JD) [Chan *et al.,* 2005] es un mecanismo que proporciona calidad de servicio en redes multimedia detectando y descartando paquetes que han acumulado grandes *jitter*. *Jitter Detection* trata de reducir la media del retardo de los paquetes multimedia, a la vez que mantiene un alto rendimiento de utilización de éstos mismos paquetes. Este mecanismo conserva la estructura básica de RED e introduce un nuevo esquema JD tal como se muestra en la Figura 4.9.

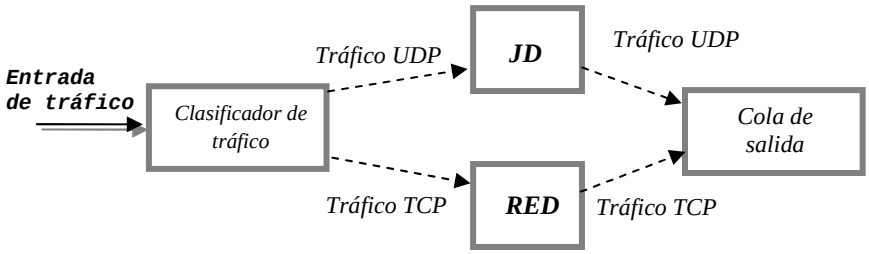


Figura 4.9. Modelo *Jitter Detection* para redes multimedia.

Cuando un paquete es recibido, éste es clasificado como una de las dos posibles categorías: TCP o UDP. Este paquete es insertado en el esquema que le corresponde de acuerdo a su identificación de flujo. El tráfico UDP es sujeto al esquema JD mientras que el TCP es sujeto a RED estándar. Después de pasar por alguno de los esquemas, el paquete es puesto en la salida de una cola de tipo FIFO.

Para cada paquete multimedia, JD calcula un retardo R_p que depende de la ocupación

actual de la cola de salida Q_{inst}, el tamaño del paquete *P* y de la capacidad de la línea *C*, utilizando

$$R_p = q * P / C \quad [4.30]$$

Posteriormente, recurriendo a un suavizado exponencial del retardo, obtiene el retardo medio mediante

$$R_{avg} = (1 - w_d) * R_{avg} * w_d * R_p \quad [4.31]$$

donde w_d es el factor de peso para la estimación. Por lo que el *jitter* del paquete multimedia estará dado por

$$J_p = R_p - R_{avg} \quad [4.32]$$

Este valor J_p determinará ahora cuál de los paquetes multimedia deberá ser descartado o enviado a la cola de salida. Por lo que, el algoritmo JD estima un umbral de descarte (U_d) para cada paquete de acuerdo con la expresión

$$U_d = J_p - \frac{F_{th}}{2} * (1 + \frac{(v + [v/3])}{3}) \quad [4.33]$$

En particular, si el valor del umbral obtenido se encuentra entre un determinado rango de valores, $\frac{-F_{th}}{2} \le U_d \le \frac{F_{th}}{2}$, donde F_{th} identifica el factor de límite, el paquete multimedia no será descartado y se enviará entonces a la cola de salida junto con los paquetes TCP.

RaQ

En RaQ [Sun and Zukerman, 2007], se estabiliza el tamaño de la cola a un objetivo de referencia dentro del contexto de teoría de control. Para calcular la probabilidad de descarte de un paquete, este algoritmo utiliza simultáneamente un controlador proporcional de la tasa y un controlador proporcional e integral del tamaño de la cola. La probabilidad de descartar o marcar un paquete viene dada por

$$p_d(t) = \psi\{\ r_{kp}(r(t) - C) + q_{kp}(Q_{inst}(t) - q_{ref}) + q_{ki}\int_0^l (Q_{inst}(t) - q_{ref})dt\ \} \quad [4.34]$$

Donde *r(t)* es la tasa de entrada de la cola, *C* es la capacidad del enlace, Q_{inst} *(t)* es el tamaño de la cola instantánea, q_{ref} el tamaño de la cola de referencia, r_{kp} es el coeficiente

del controlador proporcional de la tasa de tráfico, q_{kp} y q_{ki} son los coeficientes del controlador proporcional e integral de la cola, respectivamente.

La función ψ{x}es definida por

$$\psi(x) = \begin{cases} 1, x > 1 \\ x, 1 \geq x \geq 0 \\ 0, x < 0 \end{cases} \quad [4.35]$$

De ahí, que el esquema RaQ puede ser considerado como un sistema de control de doble retroalimentación. Donde, el control de retroalimentación interior está basado en la tasa de tráfico y el exterior está basado en la cola, como se muestra en la Figura 4.10.

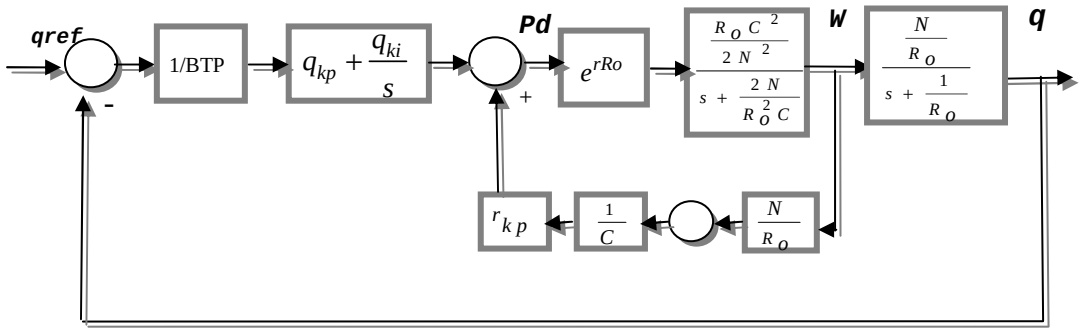


Figura 4.10. Sistema de control de congestión TCP con RaQ.

4.4. Otras formas de clasificar a los AQM

En esta investigación, como en [Aoul *et al*., 2007], se ha llevado a cabo la clasificación de los mecanismos de gestión activa de colas adicional siguiendo el enfoque adoptado por cada uno de los esquemas. Para esto, cuatro diferentes estrategias son identificadas: (1) enfoque heurístico, (2) enfoque de teoría de control, (3) enfoque basado en optimización y finalmente, 4) enfoque híbrido.

Los mecanismos con enfoque heurístico y de teoría de control son métodos que usualmente consideran un modelo lineal. Los mecanismos AQM diseñados en el contexto de técnicas de teoría de control, mejoran el compromiso entre la utilización del enlace y el retardo de la cola reajustando la ocupación media de la cola del *router* como un sistema variable. Este criterio hace al sistema independiente de la carga de trabajo. Ejemplos de

AQM pertenecientes a esta clasificación son los esquemas *Yellow*, LRED y RaQ.

Por su parte, los esquemas AQM con un enfoque de optimización intentan encontrar el valor óptimo de la probabilidad de descarte considerando a la vez, minimizar el retardo de encolamiento y maximizar el rendimiento. Esta clase de mecanismos permite mantener un bajo nivel de ocupación de cola comparado con los AQM con enfoque de teoría de control y heurístico, lo cual hace que con frecuencia padezcan el problema de baja utilización del enlace. Algunos ejemplos de esta clasificación son REM y RAQM. Finalmente, con el objetivo de permitir un buen rendimiento en el *router* y una mejor gestión de la ocupación de la cola, se han diseñado mecanismos AQM basados en un enfoque híbrido como los algoritmos PD-RED y FAFC.

Alternativamente, los algoritmos AQM pueden ser clasificados sólo en dos categorías basándose en si son reactivos o si toman medidas preventivas. Los algoritmos AQM reactivos se centran en evitar la congestión (la congestión puede ocurrir pero es detectada tempranamente) y las acciones a seguir se basan en la congestión actual. Ejemplos de esta categoría son ARED, SFB y AVQ. Por otra parte, los AQM preventivos como el esquema GREEN se enfocan en prevenir la congestión, y las acciones a seguir son consideradas basándose en la congestión esperada.

4.5. Conclusiones de AQM

En este capítulo se ha presentado una muestra de los mecanismos de gestión activa de colas más representativos encontrados en la literatura. Aparte de los citados, queda un numeroso grupo de esquemas, entre los cuáles podemos citar algoritmos que intentan maximizar la utilización del enlace como *BLUE* [Feng *et al.,* 2002]. Otros como *PD-Controller* [Sun *et al.* 2003b] que tratan de estabilizar la ocupación de la cola adoptando un enfoque de teoría de control o quizás, *Dynamic* RED (DRED) [Aweya *et al.,* 2001] y *Hybrid* RED (HRED) [Joo *et al.,* 2003] que comparten la misma meta que *PD-Controller* pero basándose en un enfoque híbrido. Por su parte, SFC (S*tate feedback controller*) [Gao and Hou, 2003] y *Predictive* AQM (PAQM) [Gao *et al.,* 2002] buscan mejorar el funcionamiento global de la red adoptando un enfoque de optimización. Y aún más, esquemas como CHOKe (*CHOose and Keep for responsive flows, CHOose and Kill for unresponsive flows*) [Pan *et al.,* 1999] y *Balanced* RED (BRED) [Anjum and Tassiulas, 1999] que abordan el problema de falta de equidad en ancho de banda al compartir flujos TCP y UDP. Una visión general de algunos otros esquemas AQM puede ser encontrado en [Aoul *et al*., 2007; Thiruchelvi and Raja, 2008].

Los AQM han sido principalmente abordados bajo dos contextos, primeramente según el criterio utilizado para estimar la probabilidad de descarte del paquete en la cola y en segundo término, según el enfoque adoptado por cada uno de ellos. La categorización aplicada pretende proporcionar un selectivo y diversificado grupo de AQM bajo una estructura lo más clara posible. Las Tablas 4.2, 4.3 y 4.4 sintetizan los esquemas AQM seleccionados y los caracterizan según su enfoque y métrica de la congestión.

De la exploración a los AQM, se ha encontrado que la probabilidad de descarte del paquete esta directamente relacionada con la carga del tráfico de entrada. Sin embargo, si flujos reactivos y no-reactivos (tráfico TCP y UDP) son mezclados en una cola sencilla, los esquemas AQM previenen o evitan la congestión independientemente de la naturaleza reactiva de los tráficos involucrados. En definitiva, son pocos los esquemas AQM (como *Dynamic -Class Based Thresholds*, RED-Boston y *Jitter Detection*) que proporciona un control de congestión que se adapte a los requerimientos del tráfico.

Tabla 4.2.
ALGORITMOS AQM SEGÚN ENFOQUE HEURÍSTICO

Enfoque	AQM	Medida de congestión	Meta
Heurístico	RED	Ocupación media de la cola	Eliminar el problema de sincronización global.
	FRED	Ocupación media de la cola	Reducir el efecto de falta de equidad del esquema RED, imponiendo una probabilidad de descarte de paquetes que depende de la ocupación de la cola de cada uno de los flujos.
	SRED	Ocupación instantánea de la cola	Reducir las altas fluctuaciones de la cola instantánea, estabilizando la ocupación de la cola a un nivel dado independientemente del número de flujos activos TCP.
	ARED	Ocupación media de la cola	Resolver el problema de la alta sensibilidad de los parámetros en RED. Adapta el parámetro Max_p en función de los cambios del tamaño de la cola.
	D-CBT	Ocupación media de la cola	Aplicar diferentes políticas según el tipo de flujo al que pertenece el paquete.
	PUNSI	Ocupación media de la cola	Proporcionar una asignación de ancho de banda equitativa entre flujos reactivos y flujos no-reactivos.
	AVQ	Tasa de entrada	Dar una respuesta rápida y minimizar el retardo en la cola.
	SFB	Tasa de pérdida de paquetes y utilización de enlace	Reducir el efecto de falta de equidad. Detecta flujos agresivos y disminuye su tasa de transmisión para proteger a los flujos frágiles.
	GREEN	Tasa de entrada	Mantener una tasa de pérdidas baja y minimizar los retardos de encolamiento.

Tabla 4.3.
ALGORITMOS AQMSEGÚN ENFOQUE HEURÍSTICO, TEORÍA DE CONTROL Y OPTIMIZACIÓN

Enfoque	AQM	Medida de congestión	Meta
Heurístico	CARE	Tasa de entrada	Proveer ancho de banda equitativo entre flujos basándose en el modelo Captura-Recaptura.
	RED-Boston	Ocupación media de la cola y retardos sugeridos	Satisfacer requerimientos de QoS a todas las aplicaciones, ajustando Max_p para mantener una ocupación media de la cola en un objetivo determinado entre el Min_{th} y el Max_{th}.
	JD	Ocupación instantánea de la cola y *jitter*	Reducir el retardo medio de los paquetes multimedia, a la vez que mantiene un alto rendimiento de utilización de éstos mismos.
Teoría de Control	Yellow	Ocupación instantánea de la cola y tasa de entrada	Estabilizar la ocupación de cola y garantizar un buen funcionamiento en términos de utilización del enlace.
	LRED	Ocupación instantánea de la cola y tasa de pérdidas	Estabilizar la ocupación de la cola a un valor esperado.
	RaQ	Ocupación instantánea de la cola y tasa de entrada	Estabilizar la ocupación de la cola a un valor de referencia.
Optimización	RAQM	Tasa de entrada (primer modo de operación). Tasa de entrada y ocupación de la cola (segundo modo)	Dar una respuesta rápida, estabilizar la tasa de entrada a la capacidad del enlace y el tamaño de la cola a un valor esperado.
	REM	Ocupación instantánea de la cola y tasa de entrada	Estabilizar la tasa de entrada a la capacidad del enlace y el tamaño de la cola a un valor pequeño independientemente del número de usuarios que estén compartiendo el enlace. Minimizar retardo en cola.

Tabla 4.4.
ALGORITMO S AQM SEGÚN ENFOQUE HIBRIDO

Enfoque	AQM	Medida de congestión	Meta
Híbrido	PD-RED	Ocupación de la cola	Estabilizar la ocupación de la cola y lograr una mayor utilización de recursos basándose en el principio de Control Proporcional-Derivado
	FAFC	Ocupación de la cola	Minimizar la fluctuación de la cola, optimizar el rendimiento sin importar la variación de carga de tráfico y otorgar el mejor compromiso entre la utilización del enlace y el retardo en la cola, basándose en un robusto algoritmo de lógica difusa y un mecanismo auto-configurado.

Capítulo 5

Descarte selectivo de paquetes

Para servir diversas aplicaciones el protocolo IP necesita de nuevos mecanismos de prevención de la congestión en escenarios con diferentes tipos de tráfico compartiendo recursos de red limitados. Para dar solución a este problema, en la literatura se han propuesto, como ya ha sido expuesto en el capítulo 4, un número de algoritmos de gestión activa de colas. Sin embargo, estos esquemas en general hacen caso omiso del conocimiento del tipo de tráfico asociado con los paquetes presentes en la cola. Como consecuencia, en términos generales, los algoritmos AQM controlan eficientemente el problema de congestión sin tomar en cuenta el efecto sobre la QoS para los distintos tráficos involucrados. En este sentido, el servicio proporcionado puede no coincidir con los requerimientos característicos de los diferentes tráficos, tal como los servicios de VoIP, que demandan retardo extremo-a-extremo limitado y tasa de pérdidas acotadas.

La ausencia de garantías de QoS en Internet y motivados por la necesidad de obtener mejores rendimientos durante la transmisión, en la investigación realizada se han llevado a cabo varios estudios que consideran las peculiaridades de las diferentes clases de tráfico compartiendo los recursos de la red. Estos con el objetivo de mejorar las prestaciones en cuanto a parámetros que definen la QoS (pérdidas de paquetes y retardo) con el objetivo último de mejorar la calidad subjetiva percibida por el usuario final sin olvidar la equitatividad en el reparto de los recursos (ancho de banda) entre los distintos tráficos involucrados.

En este capítulo se propone un nuevo método a ser aplicado en *routers* AQM que considera la naturaleza o tipo de tráfico en la selección del paquete a ser descartado. Como se expone más adelante, la propuesta denominada Descarte Selectivo (*Drop-Sel)*, observa y clasifica el tráfico de la red en la cola y, en correspondencia, selecciona el paquete a ser descartado de la clase de tráfico con mayor carga en ella (implícitamente identifica la clase de tráfico a ser penalizada). En consecuencia, este método balancea la penalización aplicada entre las clases, particularmente entre TCP y UDP, mejorando la equitatividad sin necesitar la participación de un algoritmo explícito de *scheduling* (o planificador).

En el estudio se evalúa mediante simulación las mejoras introducidas por el algoritmo tanto en términos de QoS, como en términos de la calidad subjetiva experimentada (QoE) por el usuario final.

5.1. Introducción

Cuando un esquema AQM determina que un paquete debe ser marcado o descartado, éste implícitamente ejecuta un algoritmo de selección de víctima que elige el paquete a marcar o descartar. Por simplificación, en este trabajo se asume que la política adoptada es descartar paquetes en lugar de marcar.

Estos algoritmos acostumbran ser tan simples como seleccionar el último, el primero o un paquete aleatoriamente de la cola, siendo referidos de aquí en adelante como *Drop-Tail*, *Drop-Front* y *Drop-Random* respectivamente (Figura 5.1). Muchos esquemas AQM utilizan esta clase de algoritmos para seleccionar el paquete a descartar. Sin embargo, esta

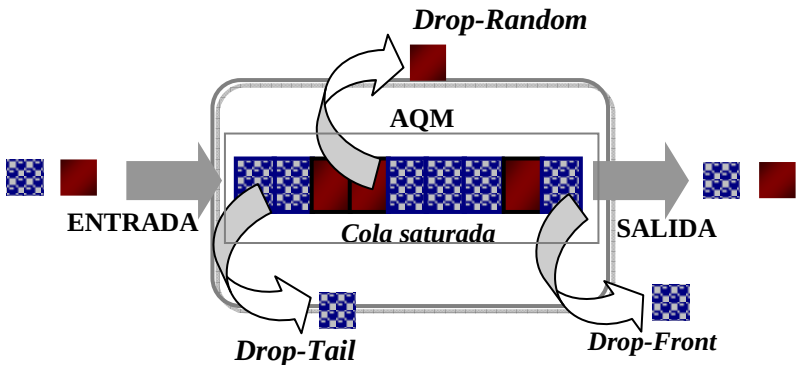


Figura 5.1. Algoritmos clásicos de selección de víctima.

decisión puede ocasionar un tratamiento injusto a los diferentes tráficos - generando un desequilibrio en la tasa de pérdidas- y posiblemente, infrautilizando los recursos de la red.

Otra cuestión importante a considera para los flujos VoIP es el retardo medio experimentado por el paquete. En aplicaciones de voz que exijan interactividad, si la latencia o retardo extremo-a-extremo de un paquete sobrepasa los 300 ms, éste es considerado una pérdida. Más aún, como la calidad de los flujos de VoIP depende altamente del retardo medio del paquete [Perkins *et al*., 1998], es esencial que todo procedimiento en la ruta trate de reducirlo. En este respecto, investigaciones previas han demostrado que el esquema *Drop-Front* es mejor reduciendo el retardo medio [Lakshmann *et al*, 1996; Yin and Hluchyj, 1993]. Sin embargo, a pesar de este beneficio, este esquema de selección no distingue entre clases de tráfico, y como resultado, una clase de tráfico puede tener una mayor oportunidad de ser seleccionada y por tanto ser forzada a tener una fracción menor del ancho de banda que le corresponde. Un algoritmo de selección de victima más efectivo puede ser concebido si la selección es realizada considerando la clase de tráfico a la cuál pertenece el paquete. Haciendo esto, mejor rendimiento para todas las clases de tráfico involucradas puede ser esperanzadoramente obtenido sin necesidad de utilizar ningún *scheduler* adicional.

En la investigación realizada se propone esencialmente un algoritmo de selección de victima, referido de aquí en adelante como *Drop-Sel*. El objetivo es proporcionar equidad y proveer, si es posible, una mejor utilización general de la red cuando se mezclan tráficos correspondientes a fuentes reactivas a la congestión (TCP) como no-reactivas (UDP). *Drop-Sel* evita un *scheduler* que utilice múltiples colas. En lugar de eso, el esquema propuesto considera una cola sencilla en la cual todos los tráficos son servidos.

En concreto, este capítulo expone el algoritmo de descarte selectivo (*Drop-Sel*), las condiciones experimentales de los escenarios de simulación considerados, así como los resultados obtenidos, finalizando con las conclusiones extraídas.

5.2. Esquema de descarte selectivo (*Drop-Sel*)

Considerando que los flujos con más alta carga de trabajo son los principales contribuyentes al evento de congestión, el algoritmo *Drop-Sel* simplemente identifica en la cola los paquetes pertenecientes a dicha clase de tráfico, penalizándolos -en los momentos que sea necesario- intentando así proteger a los otros flujos. Por lo que, este algoritmo intenta penalizar a los flujos causantes de la congestión, asignando a éstos una alta probabilidad a ser descartados. Como resultado, un trato diferenciado para cada

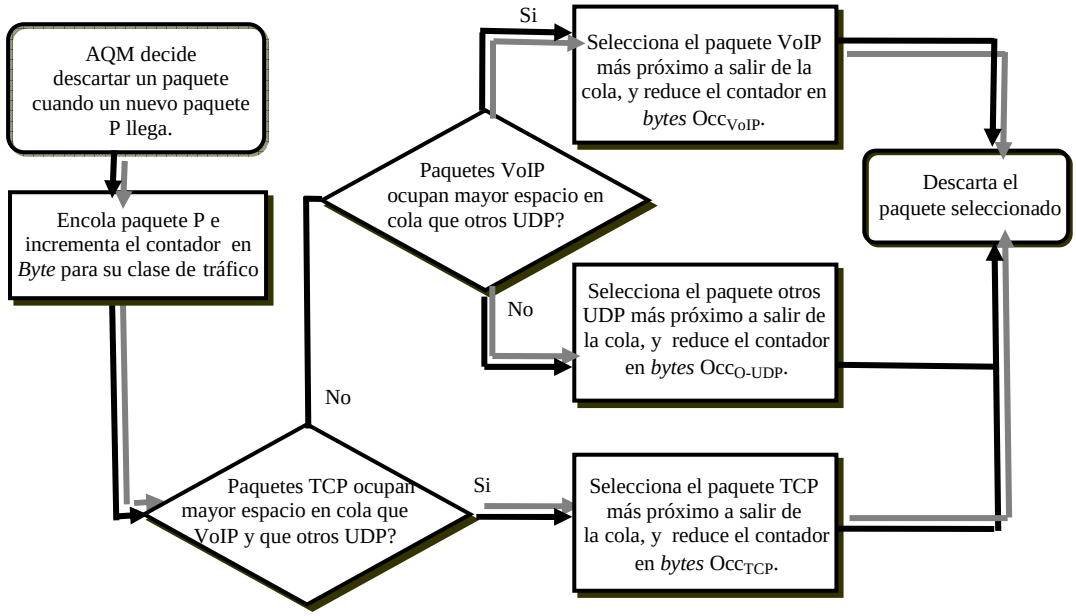


Figura 5.2. Algoritmo *Drop-Sel*.

paquete basado en su clase de tráfico es proporcionado.

En esta investigación, de forma similar a [Chung and Claypool, 2000], se han considerado tres clases de tráficos, clase *real-time* correspondiente a fuentes no-reactivas UDP (flujos VoIP), clase otros UDP no *real-time* y finalmente TCP.

Para proporcionar un servicio equitativo, el algoritmo *Drop-Sel* tiene en cuenta la cantidad de *bytes* encolados para cada clase de tráfico. *Drop-Sel* seleccionará el paquete más cercano a la cabecera de la cola de la clase de tráfico con mayor presencia en la cola. Con este propósito, cuando un paquete llega a la cola, éste es clasificado como *real-time* (VoIP), otros UDP o elástico (TCP), y en consecuencia el correspondiente contador es adaptado según sea el caso. Simultáneamente, el paquete es encolado o descartado con una probabilidad de acuerdo con lo que determine la disciplina AQM adoptada. Si el algoritmo AQM decide descartar el paquete, éste ejecuta el algoritmo de selección de víctima. Primero, los contadores *real-time*, otros UDP y TCP son comparados para determinar cuál de las tres clases de tráfico ocupa mayor espacio en la cola. Entonces, el algoritmo busca el paquete perteneciente a la clase seleccionada que esté más próximo a salir de la cola, y consecuentemente lo descarta. En la Figura 5.2, se representa el algoritmo *Drop-Sel* mediante un esquema simple.

Drop-Sel preferentemente acepta paquetes de la clase de tráfico con menos *bytes* en la cola durante los períodos de congestión. En consecuencia, como va a ser mostrado, el algoritmo proporciona una mejor oportunidad a la clase de tráfico que consume menos recursos.

Específicamente, las ventajas de *Drop-Sel* son las siguientes: primeramente, el algoritmo sólo considera la ocupación instantánea de la cola en un esquema basado por-clase en lugar de uno basado por-flujo. En este sentido, el algoritmo de selección de víctima es escalable. Segundo, para distintas clases de tráfico, éste utiliza una cola sencilla, por lo que no requiere un *scheduler* para proporcionar equidad. Adicionalmente, aún si alguna de las clases no requiere utilizar su porción equitativa de los recursos de la red (memoria y ancho de banda), *Drop-Sel* podrá otorgar equitativamente los recursos extras no utilizados a otra de las clases de tráficos. Tercero, las fuentes no requieren insertar información de señalización extra dentro de la cabecera de los paquetes.

5.3. Evaluación del modelo propuesto

Para medir el impacto a nivel de red y a nivel de usuario del esquema *Drop-Sel*, un número de simulaciones fueron realizadas con *Network Simulator* (*ns-2*). Más precisamente, los seleccionados esquemas AQM (RED, REM y AVQ, descritos en apartados 4.1 y 4.2) fueron evaluados bajo varios niveles de congestión, implementando los algoritmos de selección de víctima descritos en el apartado 5.1 y 5.2.

Para medir adecuadamente los beneficios facilitados a los flujos VoIP por el algoritmo propuesto, se midieron los parámetros de calidad a nivel de red, específicamente el retardo medio por paquete y la tasa de pérdidas, además de la calidad percibida por el usuario utilizando el Modelo-E [ITU-T, 2005] y un sistema automático de reconocimiento de voz.

5.3.1. Herramientas utilizadas

Network Simulator

Network Simulator (*ns*) [NS2, 2008] es una herramienta flexible y fácil de usar que ha sido ampliamente adoptada por la comunidad científica como herramienta de simulación de redes basada en eventos discretos. *ns* proporciona un amplio espectro de facilidades para evaluar (a bajo coste) muchos de los aspectos que pueden influir en el rendimiento

de la red. Además de otros muchos componentes, para el interés del trabajo realizado, *ns* integra algoritmos de encaminamiento y control de congestión con protocolos de transporte, sesión y aplicación. Este simulador es capaz de trabajar con un gran número de aplicaciones, modelos de tráfico, tipos y elementos de red. Está basado en dos lenguajes: un simulador orientado a objetos escrito en C++ y un intérprete orientado a objetos en *Tcl* (*OTcl*). *Tcl* (*Tool Command Language*) [TCL] permite a los usuarios elaborar de forma sencilla *scripts* que especifiquen una topología de red en particular, con diferentes fuentes de información, protocolos, aplicaciones así como distintas líneas de transmisión con las diferentes tecnologías de transmisión más relevantes. El *script Tcl* determina en qué momento deberá ser simulado un evento, y con ello obtener la salida de datos requerida permitiendo así la evaluación de cualquier mejora o la determinación de la influencia de muchos de los aspectos que pueden afectar al rendimiento de la red.

ns genera archivos de trazas en formato *ascii*. En estos archivos se registran cada uno de los eventos ocurridos (tipo de evento, tiempo de cuando éste ocurrió, etc.). Estos datos pueden ser manipulados para extraer información relevante. Por otra parte, existen otros objetos en *ns* que facilitan la obtención de estadísticas acerca del tráfico que se recibe en cualquier punto de la red, como es el caso del monitor de colas, el monitor de flujos y el monitor de pérdidas. Los datos que pueden generar estos objetos incluyen al tamaño de la cola instantánea, los instantes de llegadas, salidas y pérdidas de paquetes.

Además, dentro de *ns* se puede trabajar con algunos algoritmos de gestión activa de colas (base fundamental en la investigación realizada), tales como RED (*Random Early Detection*) REM (*Random Exponential Marking*), AVQ (*Adaptive Virtual Queue*), ARED (*Adaptive* RED), entre otros. Lo cuál, facilita la evaluación de los modelos propuestos.

Modelo-E

Para de evaluar la calidad que un usuario percibiría bajo unas determinadas condiciones, en este trabajo se ha utilizado el Modelo-E y la recomendación *G.107* descrita previamente en el apartado 3.3.2 además se ha adoptado la configuración propuesta en [Cole and Rocenbluth, 2001]:

- R_o y I_s (véase expresión [3.1]) son igual a *94.2*, ya que estas variables no dependen fundamentalmente de la red de transporte.
- I_d = 0.024*d* + 0.11*(*d* -177.3)**H*, expresión analítica en términos de retardo extremo a extremo (***d***), asumiendo que en entornos puramente IP, todas las fuentes de retardo se suman en una y la influencia del eco es despreciable.

En este caso H modela la contribución del retardo de acuerdo a

$$H = \begin{cases} 0, si(d - 177.3) < 0 \\ 1, si(d - 177.3) \geq 0 \end{cases} . \qquad [5.7]$$

- I_e = 30ln(1+ 15p), para ello se supone que el *codec* no introduce ninguna distorsión debido a la compresión. En la expresión $\boldsymbol{p}$ es la probabilidad de pérdidas totales (pérdidas de la red y pérdidas debido a paquetes que llegan después de su tiempo límite).

Por lo tanto, a partir de la expresión [3.1] la puntuación a evaluar se realizó de acuerdo con las simplificaciones adoptadas, obteniéndose

$$\boldsymbol{R} = 94.2 - (0.024d + 0.11*(d - 177.3)*H) - 30\ln(1+ 15p) \qquad [5.8]$$

Reconocedor automático de voz (ASR)

Adicional a la evaluación en términos de puntuación MOS, se evalúa la bondad de *Drop-Sel* de referencia *y Drop-Sel-delay* estimando la inteligibilidad de los flujos de audio en el usuario final, utilizando una máquina de reconocimiento automático de voz (ASR) extremo-a-extremo. El valor de reconocimiento continúo extremo-a-extremo (medido en términos de tasa de palabras o precisión de frases) es utilizada para medir la calidad de una conversación reconstruida (ver apartado 3.3.3.). El ASR tiene en cuenta elementos que con otras aproximaciones pueden ser difíciles de medir y, adicionalmente como ventajas añadidas nótese que es de bajo costo y provee medidas reproducibles.

La metodología utilizada consiste en la utilización de un sistema automático de reconocimiento de voz continua que proporciona un índice de inteligibilidad obtenido a partir de la tasa de palabras erróneas definida en la expresión [3.2]. Para la investigación realizada, la tarea de reconocimiento es fundamentada en la base de datos de voz Aurora 2 [Hirsch and Pearce, 2000], un *corpus* que consiste en secuencias de dígitos conectados para locutores de inglés americano. Después de la transmisión, la señal de voz es procesada para reducir su variabilidad inherente. Un extractor de características divide la señal de voz recibida en tramas solapadas de 25 ms de duración cada 10 ms. Cada trama de voz es representada por un vector de características de dimensión 14 que contiene 13 coeficientes en escala *MEL Frequency Cepstrum* (MFCCs) más el logaritmo de la energía *log-Energy*. Finalmente, los vectores de características son extendidos con su primera y segunda derivada.

El reconocer utilizado está basado en Modelos Ocultos de Markov (*Hidden Markov*

Models, HMM). Emplea 11 modelos de palabra HMM continuos de 16 estados con 3 gausianas por estado. Estos modelos HMM son entrenados con un sistema de 8440 frases libres de ruido grabadas en condiciones de alta SNR y pronunciadas por 55 locutores adultos masculinos y 55 femeninos, mientras el *corpus* para el *test* comprende 4004 frases libres de ruido.

5.3.2. Entorno de simulación

Los esquemas AQM seleccionados, adoptando los algoritmos de selección de víctima, fueron evaluados bajo una variedad de topologías de red, fuentes de tráfico y diferentes niveles de congestión.

Primeramente, se consideró la topología estándar *dumbbell* con un solo "cuello de botella" mostrada en la Figura 5.3. En este escenario de red, un número de fuentes TCP, flujos VoIP (UDP) y otros flujos UDP no-reactivos compiten por compartir recursos en el *router* AQM (R0). Las fuentes FTP generan segmentos TCP con una longitud igual a 1500 *byte*s, y las fuentes VoIP generan paquetes RTP (encapsulados dentro de paquetes UDP) que representan el *stream* de voz codificados con G.711 [ITU-T, 1998]. El tamaño de los paquetes UDP no-VoIP es también igual a 1500 *bytes*.

Los flujos VoIP simulados son generados utilizando cuatro aplicaciones de VoIP (A,

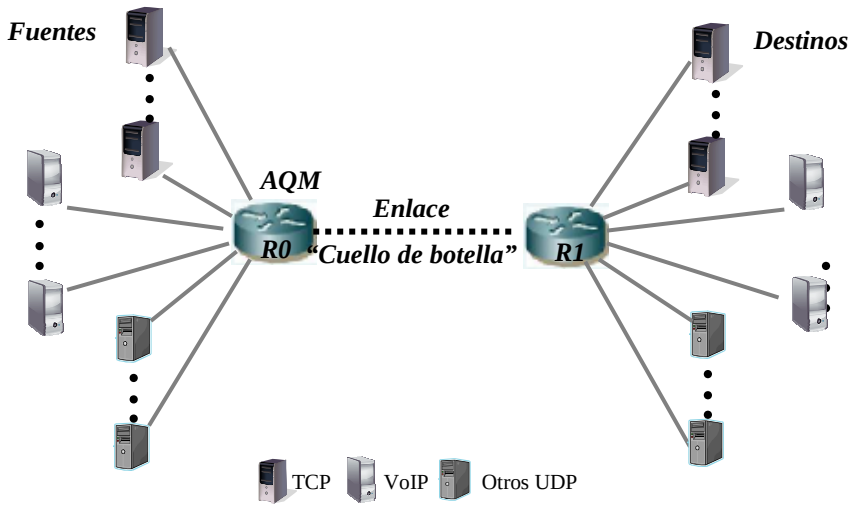


Figura 5.3. Topología *dumbbell*.

Tabla 5.1.
ESPECIFICACIÓN DE FLUJOS VOIP PARA TOPOLOGÍA *DUMBBELL*

Aplicación	Período entre paquetes	Tamaño del paquete	S1 retardo extremo a extremo	S2 retardo extremo a extremo
A	10 ms	92 *bytes*	39 ms	39 ms
B	30 ms	252 *bytes*	39 ms	79 ms
C	60 ms	492 *bytes*	39 ms	169 ms
D	30 ms	252 *bytes*	39 ms	239 ms

Tabla 5.2.
CARGAS DE TRABAJO DE FLUJOS UDP Y TCP PARA EL ESCENARIO S1

Caso	Carga VoIP	Carga otros UDP	Carga total UDP	Carga TCP
1	25%	50%	75%	25%
2	50%	25%	75%	25%
3	25%	25%	50%	50%
4	12%	12%	24%	76%

Tabla 5.3.
CARGAS DE TRABAJO DE FLUJOS UDP Y TCP PARA EL ESCENARIO S2

Caso	Carga VoIP	Carga otros UDP	Carga total UDP	Carga TCP
5	47%	36%	83%	17%
6	54%	19%	73%	27%
7	13%	52%	65%	35%
8	29%	6%	35%	65%

B, C y D) con diferentes tasas de envío y diferentes tamaños de paquetes (ver Tabla 5.1). En relación al retardo extremo-a-extremo, se consideraron dos escenarios diferentes (referidos como S1 y S2) en esta topología. En el primero (S1), todas las aplicaciones VoIP tienen igual retardo (39 ms). En S2 los retardos son diferentes para cada una de las aplicaciones (Tabla 5.1).

En el escenario S1, el *router* R0 fue sometido con cuatro diferentes cargas de trabajo, resumidas en la Tabla 5.2 (casos 1, 2, 3 y 4). Para el escenario S2, en el cuál diferentes retardos son considerados, se simularon cuatro casos extras (etiquetados como 5, 6, 7 y 8 en Tabla 5.3). En ambas tablas (Tabla 5.2 y 5.3), R0 se supone que tiene un esquema AQM RED con *Drop-Tail*.

En los experimentos del escenario S1 todas las fuentes TCP, las fuentes VoIP y los

UDP adicionales inician en el instante de tiempo simulado t=0 y están activas hasta el final de la simulación, el instante de tiempo t=500 seg. Sin embargo, en S2 todas las fuentes son modeladas utilizando un patrón de tráfico ON/OFF. El período ON para la aplicación de VoIP tiene una duración de 180 segundos, y el período OFF se mantiene por 100 segundos [Iacovoni *et al.*, 2002]. El tráfico FTP sigue una distribución de Pareto con un parámetro *shape* de k= 1.4, con un período medio ON igual a 2 segundos y un período OFF que sigue una distribución exponencial con una duración media de 1 segundo [Liu *et al.*, 2005]. Finalmente, para las aplicaciones que generan otros tráficos UDP, el período ON tiene una duración de 300 segundos, y el período OFF es 200 segundos. Cada generador de tráfico empieza a enviar paquetes con una probabilidad uniformemente distribuida durante los primeros 15 segundos de la simulación. De esta forma, asumiendo que los diferentes flujos no están sincronizados, un número de períodos de congestión son generados aleatoriamente.

Para completar la evaluación, en adición a la topología *dumbbell*, se ha considerado una topología más realista con múltiples enlaces. La topología simulada, mostrada en la Figura 5.4, consiste de un total de 7 *routers* (especificados por R0 a R6) y 4 nodos destinos (especificados por NF1 a NF4). El ancho de banda de los enlaces son establecidos a 10 Mbps, excepto para el enlace R2-R3 y R5-R6. Para causar congestión de red en nodos AQM (R2 y R5), el ancho de banda del enlace R2-R3 es restringido a

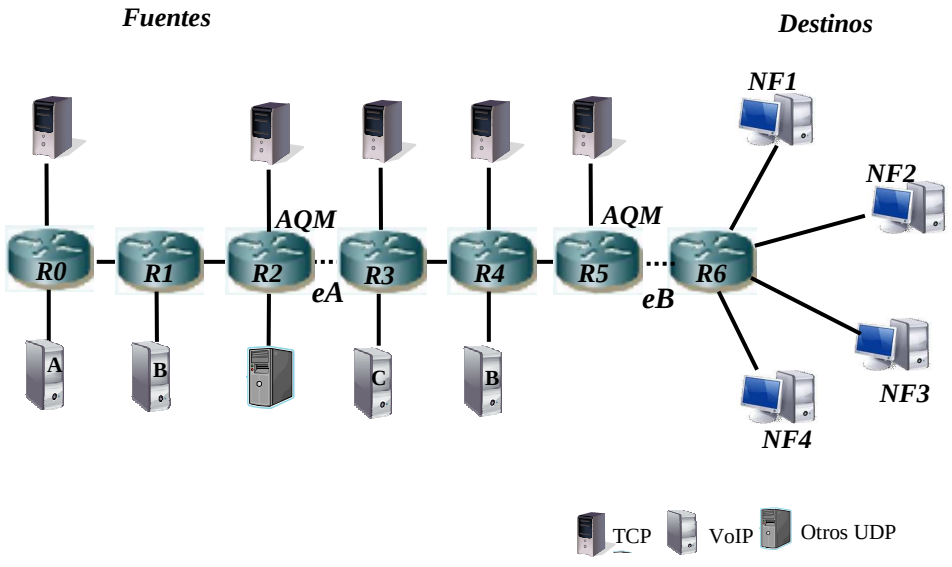


Figura 5.4. Topología compleja.

Tabla 5.4.
CARGAS DE TRABAJO DE FLUJOS UDP Y TCP PARA EL ESCENARIO S3

Caso	Carga VoIP	Carga otros UDP	Carga total UDP	Carga TCP
9	12%	39%	51%	49%
10	12%	21%	33%	67%

3.5 Mbps y el ancho de banda del enlace R5-R6 a 4.5 Mbps. De esta manera, R2-R3 y R5-R6 son los "cuello de botella" (*eA* y *eB* en Figura 5.4). Dado que los otros enlaces tienen suficiente capacidad para hacer frente al tráfico generado, ningún retardo de encolamiento extra será introducido en estos enlaces diferentes a *eA* y *eB*.

Una serie de fuentes TCP y UDP son también establecidas para este escenario complejo. Las aplicaciones de VoIP, otros UDP y FTP son nuevamente modeladas como fuentes de tráfico ON/OFF (igual que en el escenario S2). El número de fuentes es establecido adoptando la distribución relativa analizada en [Fomenkov *et al*., 2004]. En particular, se adoptó este tercer escenario (referido como S3) con las cargas de trabajo sintetizadas en la Tabla 5.4. Los flujos generados van desde las fuentes del router R0 al nodo destino NF1, desde las fuentes en R1, R2 y R5 al nodo destino NF2, desde las fuentes de R3 a NF3, y finalmente desde fuentes conectadas a R4 hasta NF4.

5.4. Impacto de *Drop-Sel*

En esta sección, se evalúa y discute el impacto de incluir el procedimiento de selección de víctima *Drop-Sel* en los esquemas RED, AVQ y REM. En particular, se ha examinado el efecto para diferentes clases de tráfico (TCP, VoIP y otros competidores UDP) bajo diferentes condiciones de tráfico. Como principal resultado se anticipa que *Drop-Sel* mejora la equidad global entre las clases de tráfico aún cuando la carga de tráfico es en su mayor parte generada por fuentes no-reactivas. Ofreciendo así un mejor servicio a las distintas clases de tráfico.

5.4.1. Equidad entre clases de tráficos

En relación al problema de equidad, se ha examinado el impacto sobre el rendimiento medio para las distintas clases de tráfico.

Primeramente, para el escenario S1 (casos 1, 2, 3 y 4 de Tabla 5.2), en las Figuras 5.5, 5.6, 5.7 y 5.8 se muestran el rendimiento medio de entrada y de salida para cada uno

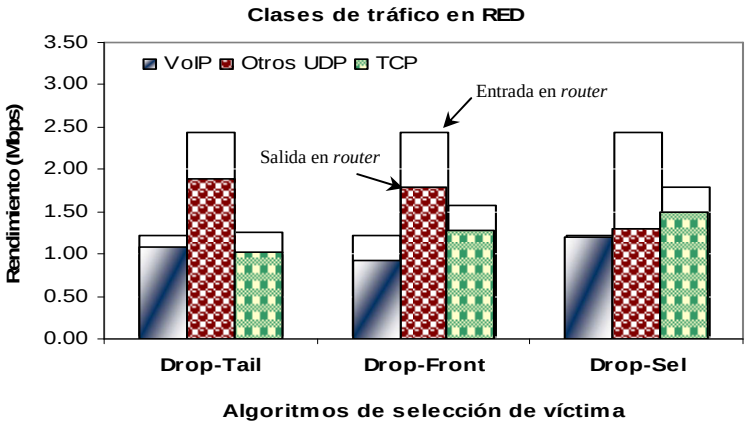
de los algoritmos de selección de víctima aplicados a los diferentes esquemas AQM considerados. En caso 1 (Figura 5.5), puede ser observado que esquemas *Drop-Tail* y *Drop-Front* claramente no son equitativos. Cuando los esquemas RED y REM adoptan los algoritmos *Drop-Tail* y *Drop-Front* (Figuras 5.5a y 5.5c), el tráfico correspondiente a otras fuentes UDP obtiene a la salida del *router* aproximadamente el doble de ancho de banda que las fuentes VoIP y TCP. Más específicamente, *Drop-Tail* en RED, respectivamente proporciona un reparto del 27%, 47%, y 26% del ancho de banda para flujos VoIP, otros flujos UDP y TCP. Por su parte, *Drop-Tail* en AVQ (Figura 5.5b), el tráfico no-reactivo de otras fuentes UDP logra consumir hasta un 34% adicional de ancho de banda respecto al tráfico TCP (52% y 18% respectivamente).

Por el contrario, *Drop-Sel* tiende a ser imparcial para los tres tráficos independientemente de la naturaleza reactiva de las fuentes contendientes y del esquema AQM utilizado. Nótese que para los tres AQM, se observa que las fuentes VoIP obtienen el máximo ancho de banda que ellos necesitan (en torno al 30%) y se otorgan más equitativamente los recursos extras no utilizados a las otras clases de tráficos. Por ejemplo, el esquema selectivo respectivamente proporciona un 32% y un 38% del ancho de banda para flujos otros UDP y TCP en AQM RED (Figura 5.5a).

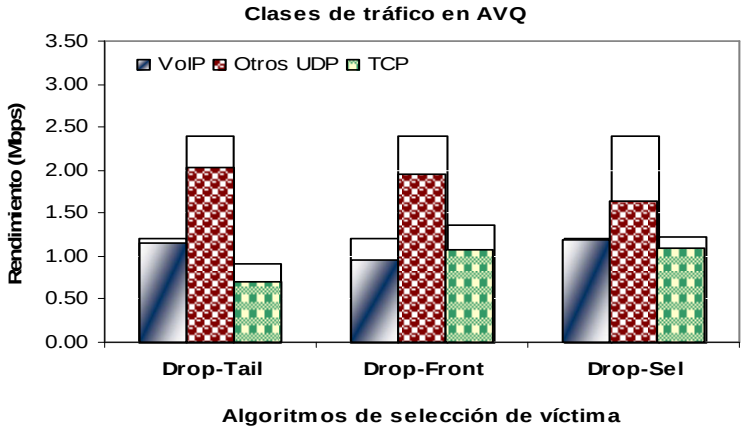
Similarmente, para el caso 2 (Figura 5.6) del escenario S1, en el cual TCP y otros flujos UDP compiten con flujos VoIP cuya carga de tráfico es predominante, se observa que *Drop-Sel* iguala respectivamente los recursos consumidos por el tráfico VoIP a los tráficos (UDP y TCP) con repartos del 34%, 31% y 35% en RED, y del 37%, 31% y 32% en el esquema REM. Nótese en este caso que las fuentes UDP que no son VoIP obtienen el máximo ancho de banda necesitado. Cuando AVQ adopta *Drop-Sel*, en este caso en particular (Figura 5.6b), se observa que no logra proporcionar una asignación tan equitativa para los tráficos TCP y VoIP como en RED y REM. No obstante, *Drop-Sel* tiende a ser el más imparcial entre los esquemas de descarte.

Además, nótese en las Figuras 5.7 y 5.8 para los casos 3 y 4, en la que las fuentes TCP generan respectivamente 50% y 76% de carga de tráfico, que *Drop-Sel* proporciona a los tráficos VoIP y a otros UDP aproximadamente todo el ancho de banda correspondiente a la carga generada, sin penalizar significativamente a las fuentes TCP.

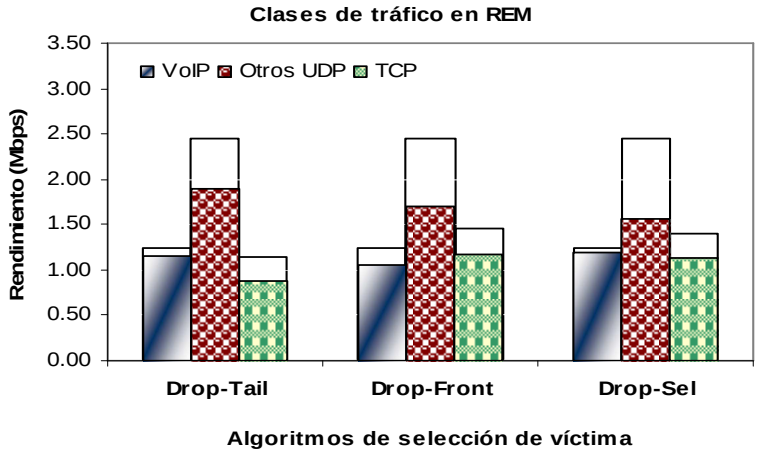
Drop-Sel también puede mejorar la equidad entre clases de tráfico cuando el escenario S2 es utilizado. Recuérdese que S2 corresponde a la topología *dumbbell* con retardos extremo-a-extremo variables. En este caso, se ha considerado particularmente el impacto del rendimiento medio en RED.



a)

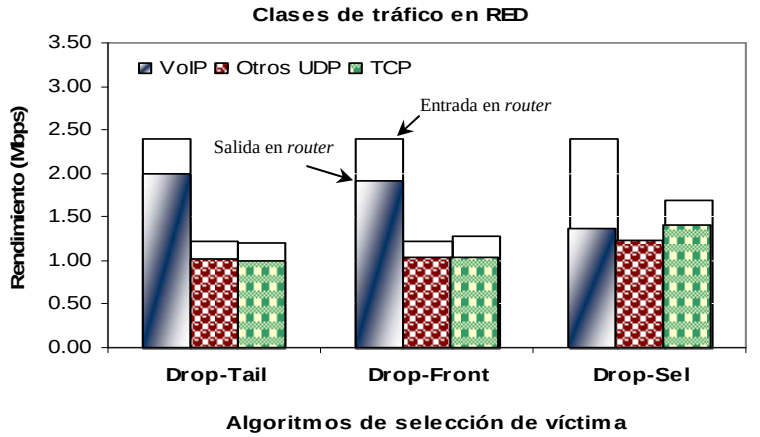


b)

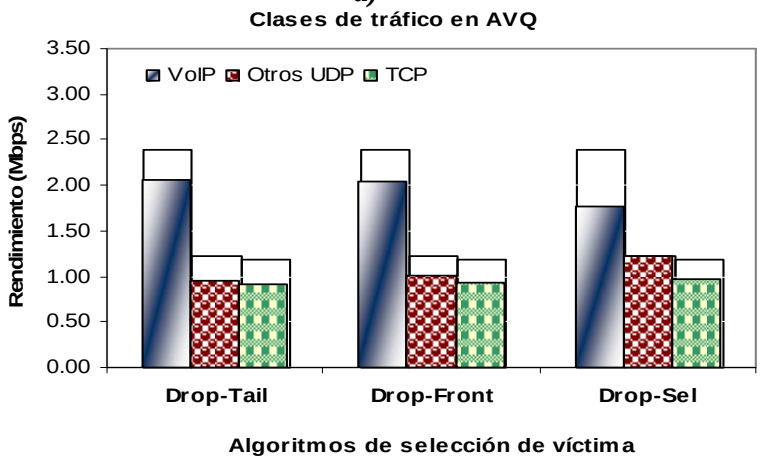


c)

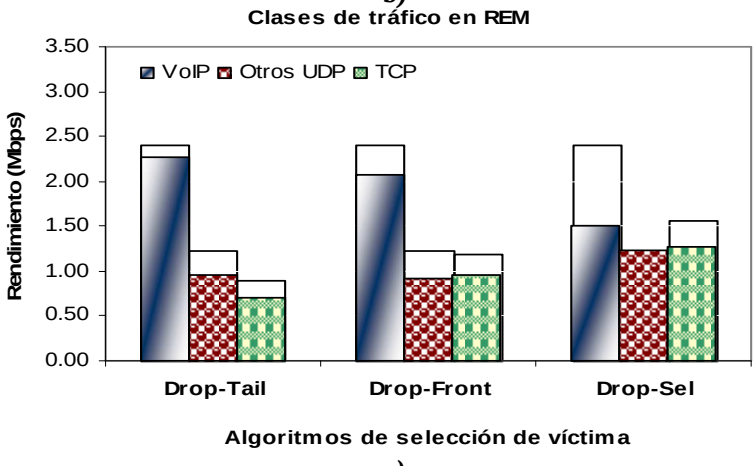
Figura 5.5 Evaluación del rendimiento medio de diferentes clases de tráfico en el caso 1.



a)

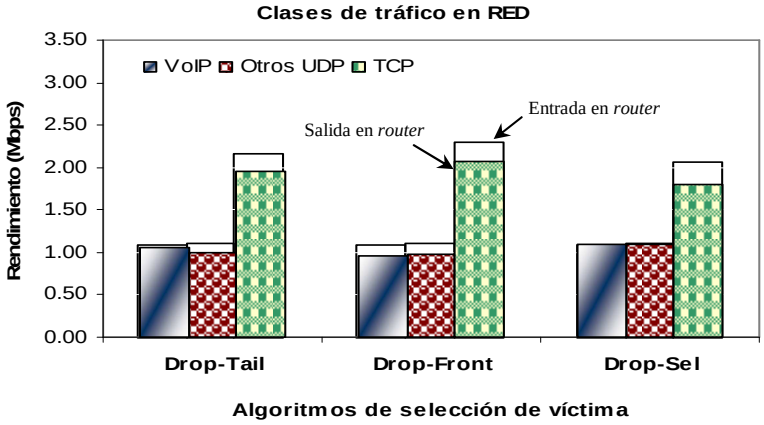


b)

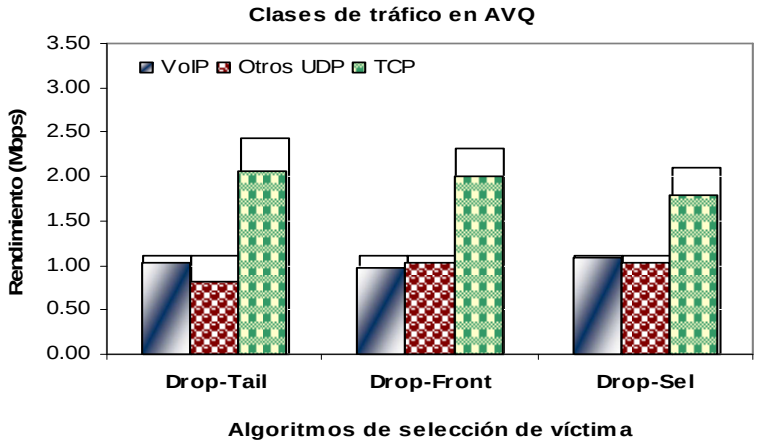


c)

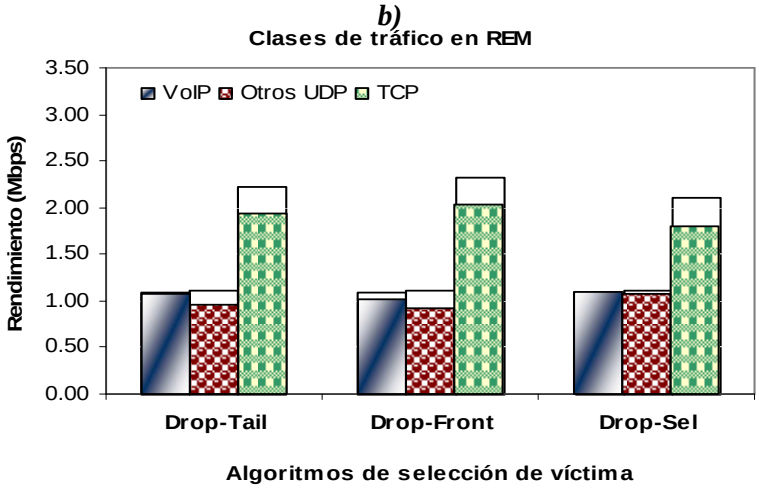
Figura 5.6. Evaluación del rendimiento medio de diferentes clases de tráfico en el caso 2.



a)

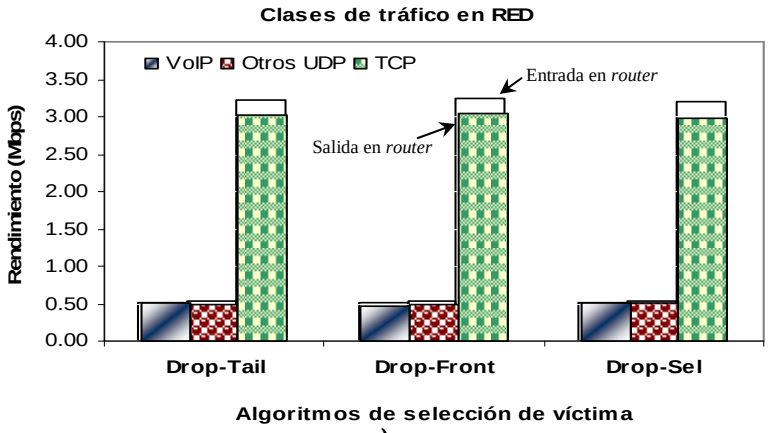


b)

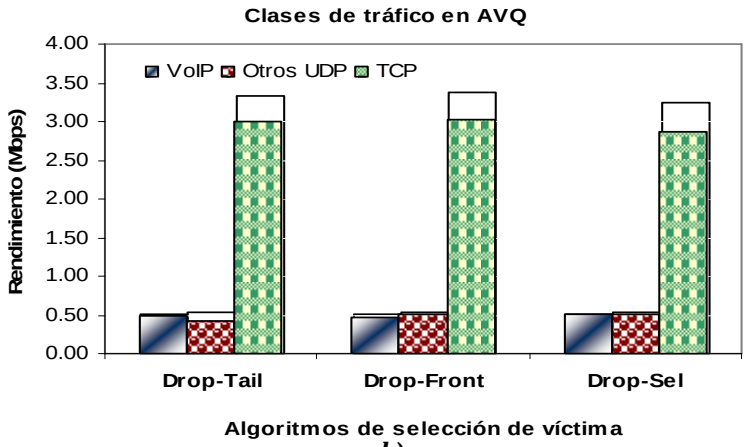


c)

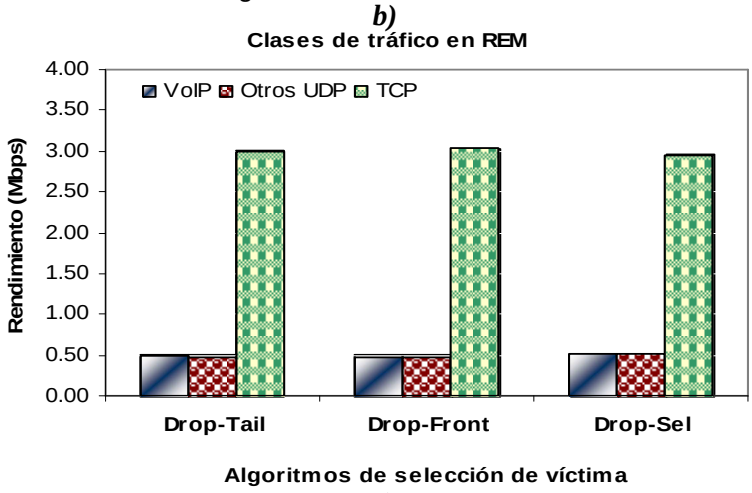
Figura 5.7. Evaluación del rendimiento medio de diferentes clases de tráfico en el caso 3.



a)



b)



c)

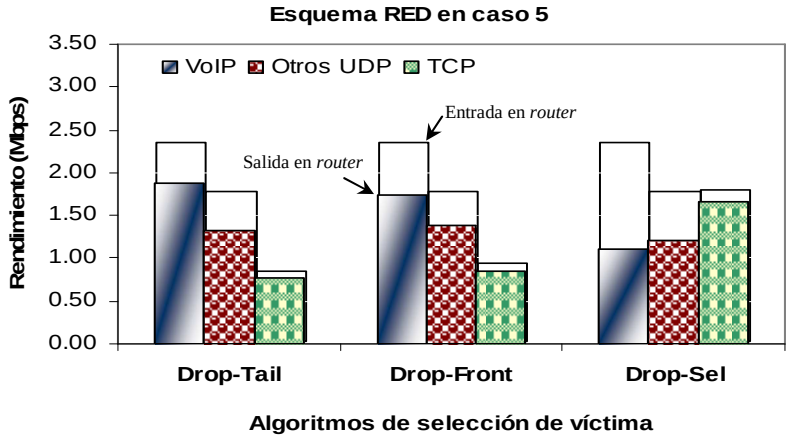
Figura 5.8. Evaluación del rendimiento medio de diferentes clases de tráfico en el caso 4.

Por ejemplo, las Figuras 5.9a y 5.9b corresponden a los casos 5 y 6 en los cuales la carga del tráfico VoIP generado es la que domina (47% y 54% respectivamente). En estos casos, cuando RED adopta *Drop-Sel*, se observa que la diferencia que existe en los tiempos de generación del tráfico en escenario S2 evita que no se vea tan claramente equitativo con las clases como lo hizo en S1. No obstante, se observa que *Drop-Sel* tiende a ser más equitativo. Nótese en las Figuras 5.9c y 5.10c para los casos 7 y 8, en los cuales respectivamente la carga del tráfico no-reactivo de otras fuentes UDP y TCP es la que domina, que *Drop-Sel* proporciona al tráfico VoIP todo el ancho de banda correspondiente a la carga generada (el mismo comportamiento que en casos 3 y 4 de S1).

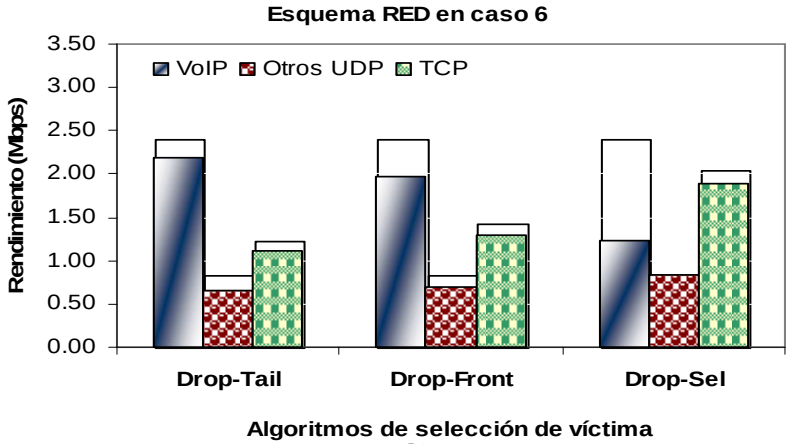
Por simplificación, para los esquemas AVQ y REM se ha considerado el impacto del rendimiento medio en dos de los casos (6 y 7 de Tabla 5.3). En el caso 6 nuevamente la carga del tráfico VoIP generado domina (54%), mientras que fuentes otros UDP y TCP respectivamente generan el 19% y 27% de la carga total. En este caso (Figura 5.10a), cuando AVQ adopta el esquema *Drop-Tail*, el tráfico VoIP aproximadamente obtiene un ancho de banda de más del doble del que obtiene TCP. Sin embargo, *Drop-Sel* respectivamente proporciona un 37% y un 42% del ancho de banda para flujos VoIP y TCP, mientras que las otras fuentes UDP obtienen todo lo generado (21%). Similarmente, el comportamiento equitativo de *Drop-Sel* es también mantenido para el esquema REM en el caso 7, como se observa en la Figura 5.10b, en la que la carga de tráfico dominante es UDP.

Como conclusión, los resultados obtenidos indican que *Drop-Sel* proporciona mucha más equidad para esquemas RED, AVQ y REM, cuando diferentes fuentes de tráfico reactivas y no-reactivas compiten por los recursos de la red sin llegar a penalizar significativamente a fuentes reactivas aún bajo condiciones de tráfico para las cuales éstas dominen.

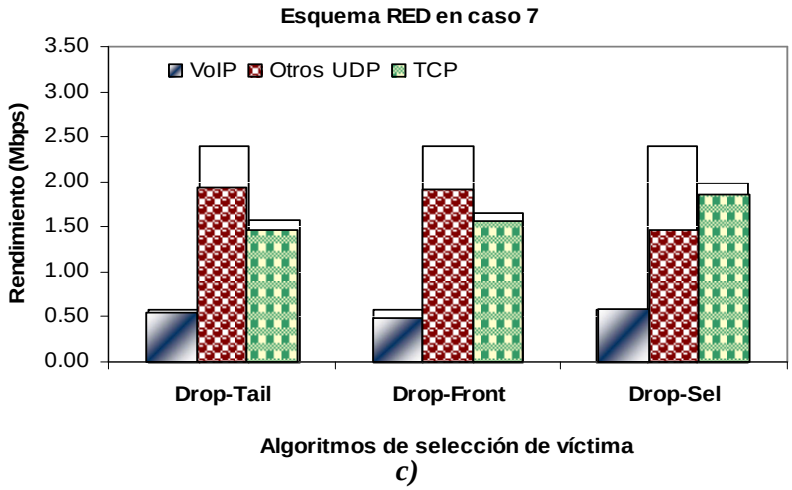
Para finalizar la evaluación de la equidad de *Drop-Sel*, se ha considerado el esquema AQM PUNSI. Éste fue especialmente diseñado para tratar con flujos no-reactivos. PUNSI previene que flujos no-reactivos dominen el ancho de banda disponible y dañen a flujos reactivos (ver apartado 4.3.1). Dos diferentes cargas de trabajo son consideradas para este fin. Primeramente, se ha evaluado el esquema PUNSI para el caso 2 en el escenario S1. En la Figura 5.11a se muestra el rendimiento de entrada y de salida de PUNSI, y para propósitos de comparación se representan nuevamente los resultados de RED con *Drop-Sel* y *Drop-Tail*. En este caso, ya que el algoritmo PUNSI solamente considera dos clases de tráfico, los flujos VoIP y las otras fuentes UDP son considerados en la misma clase de tráfico no-reactivo. Esto hace que, PUNSI no detecte estas diferencias por lo que también penaliza a las otras fuentes UDP.



a)

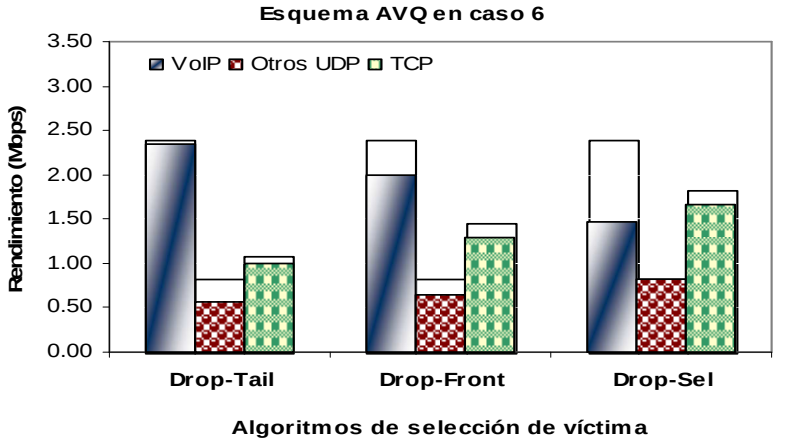


b)

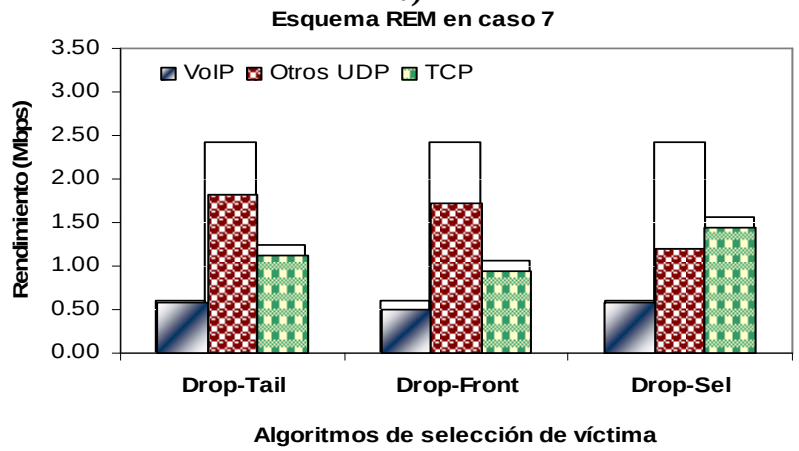


c)

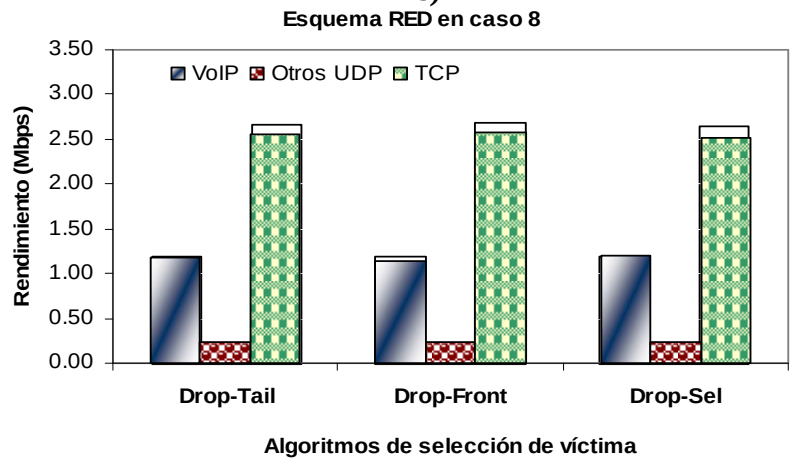
Figura 5.9. Evaluación del rendimiento medio de diferentes clases de tráfico en los casos 5, 6 y 7.



a)



b)



c)

Figura 5.10. Evaluación del rendimiento medio de diferentes clases de tráfico en los casos 6, 7 y 8.

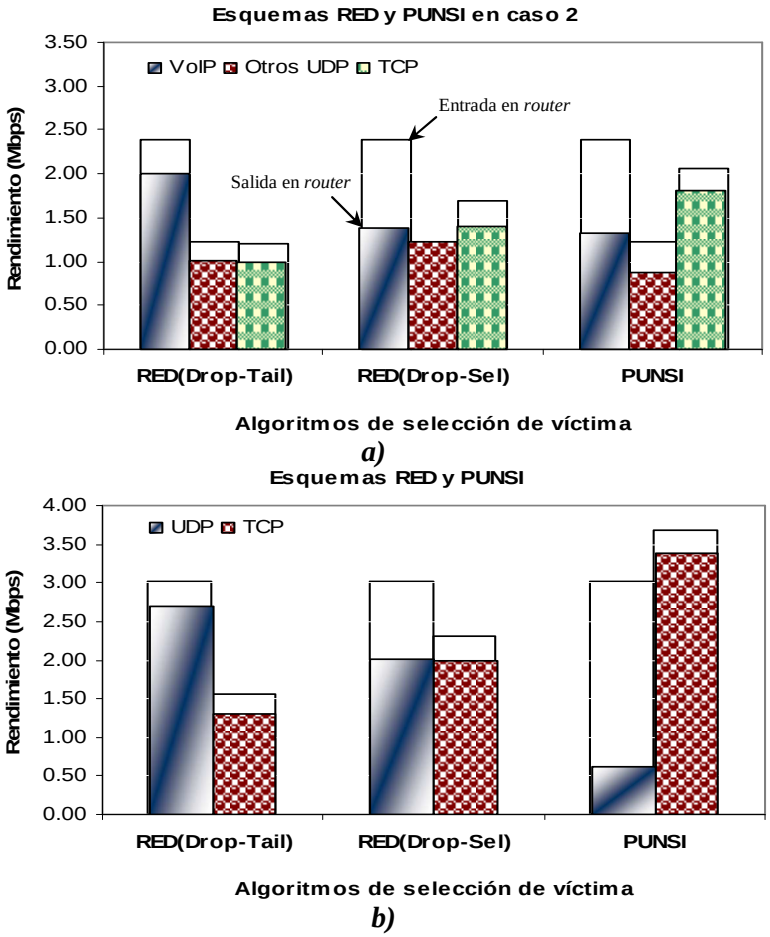


Figura 5.11. Rendimiento medio de diferentes clases de tráfico aplicadas a RED y PUNSI AQM.

Por el contrario *Drop-Sel* en RED obtiene una distribución más justa.

En segundo lugar, para una mejor clarificación, se ha considerado un escenario final en el cual sólo dos clases de tráfico compiten por los recursos de la red, fuentes no-reactivas UDP (no incluye VoIP) y fuentes TCP. El tráfico no-reactivo genera en conjunto el 66% de la carga. La Figura 5.11b muestra los resultados obtenidos. Como puede observarse, PUNSI penaliza severamente a los flujos no-reactivos, siendo tan severo como puede. Al contrario, *Drop-Sel* sigue manteniendo su comportamiento equitativo. Observe que *Drop-Sel* iguala la asignación de UDP a la del tráfico TCP.

5.4.2. Calidad de servicio (QoS) en VoIP

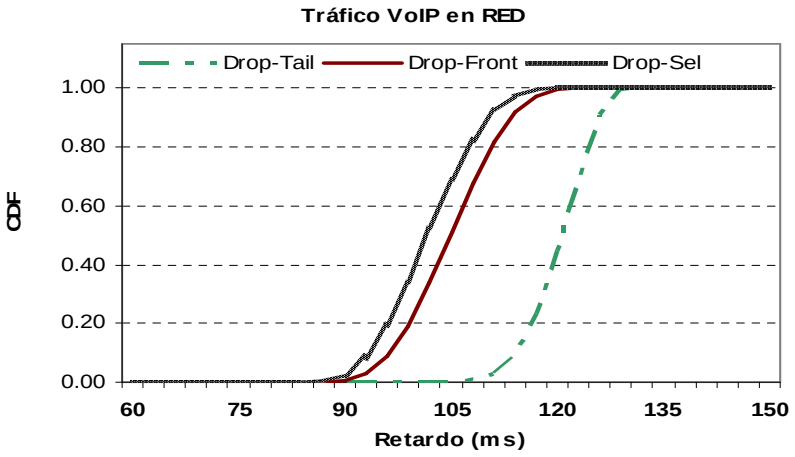
Latencia o retardo extremo-a-extremo

Para el tráfico de VoIP y para un *codec* dado, la calidad de servicio depende firmemente del retardo extremo-a-extremo experimentado por los paquetes. Por supuesto, para cada paquete, adicionalmente al retardo de propagación, el retardo extremo-a-extremo es determinado por la dinámica de la red y, más precisamente, por los intervalos de tiempo de espera en la cola de los *routers*.

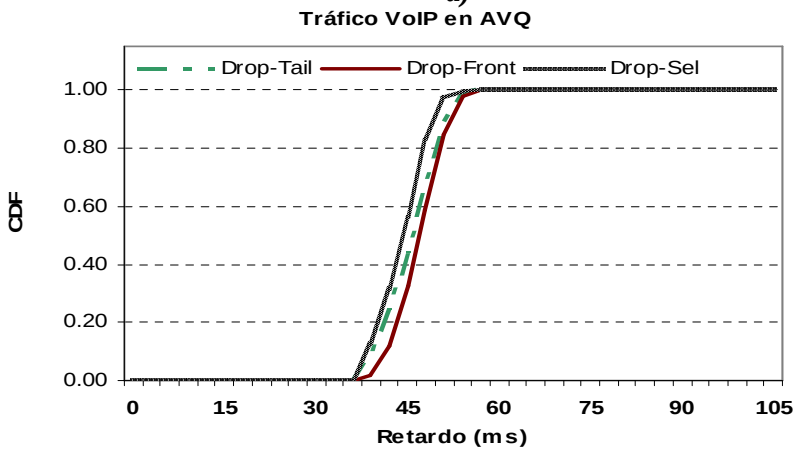
[Yin and Hluchyj, 1993] muestran que la probabilidad de que un paquete tienda a convertirse más viejo en un sistema con estrategia de descarte *Drop-Front* es menor o igual al de un sistema con descarte *Drop-Tail*. Por consiguiente, *Drop-Sel* descarta el paquete (para la clase de tráfico seleccionada) lo más próximo a la salida de la cola. En este sentido, el paquete descartado genera un espacio vacío, un cambio en la cola, y en consecuencia, una reducción en el tiempo de espera en la cola para todos los paquetes posteriores a éste. Por lo que descartar de entre los paquetes más cercanos a salir permite una reducción en el retardo extremo-a-extremo, que es particularmente significativo para el tráfico VoIP, dado que se reduce el número de paquetes no-útiles en el receptor. Adicionalmente, nótese que para fuentes reactivas el descartar de la cabecera acelera la detección de la congestión e implícitamente la reacción de las fuentes.

En este sentido se evalúa el funcionamiento de *Drop-Sel*, las Figuras 5.12, 5.13, 5.14 y 5.15 muestran la función de distribución acumulativa (CDF) del retardo extremo-a-extremo por paquete para cada uno de los algoritmos de selección de víctima considerados. En este caso, se han asumido los esquemas RED, AVQ y REM con las cuatro diferentes cargas del escenario S1 (caso 1, 2, 3 y 4 de Tabla 5.2).

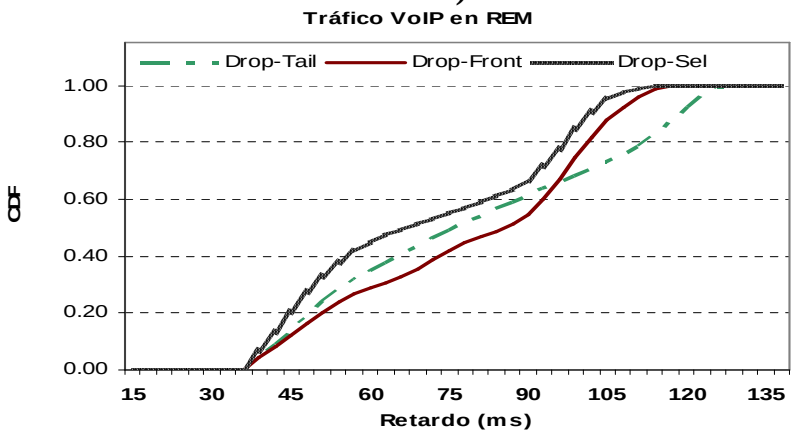
Para los casos 1, 3 y 4 (Figuras 5.12, 5.14 y 5.15), se observa que la estrategia de *Drop-Sel* induce a que los paquetes consuman igual o menos tiempo que los paquetes que han sido manipulados por *Drop-Tail* y *Drop-Front*, independientemente del esquema AQM utilizado. Nótese en las Tablas 5.5, 5.6 y 5.7, que *Drop-Sel* reduce el tiempo de espera de la cola del AQM en comparación con los otros dos algoritmos de selección de víctima. En particular, la mayor reducción de *Drop-Sel* es para el caso 1 con RED (Tabla 5.5), donde *Drop-Sel* reduce el retardo medio en la cola de 83 ms a 65 ms, 18 ms por debajo de *Drop-Tail*. Siendo también significativa la reducción en los casos 3 y 4. Permitiendo una reducción en el retardo extremo-a-extremo en la misma proporción.



a)

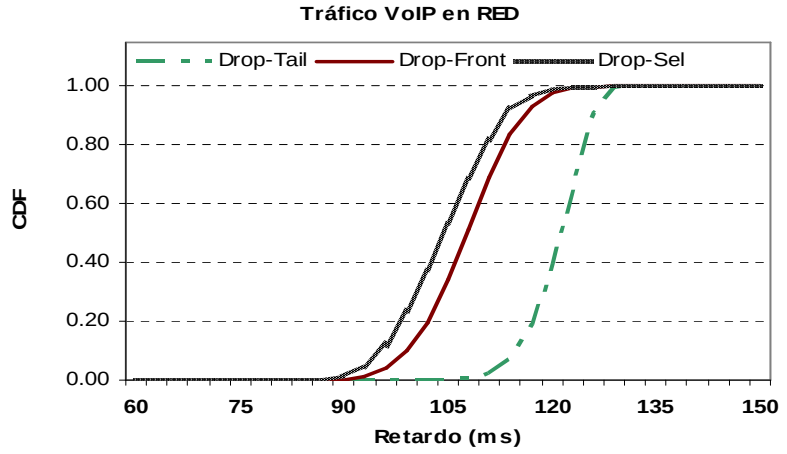


b)

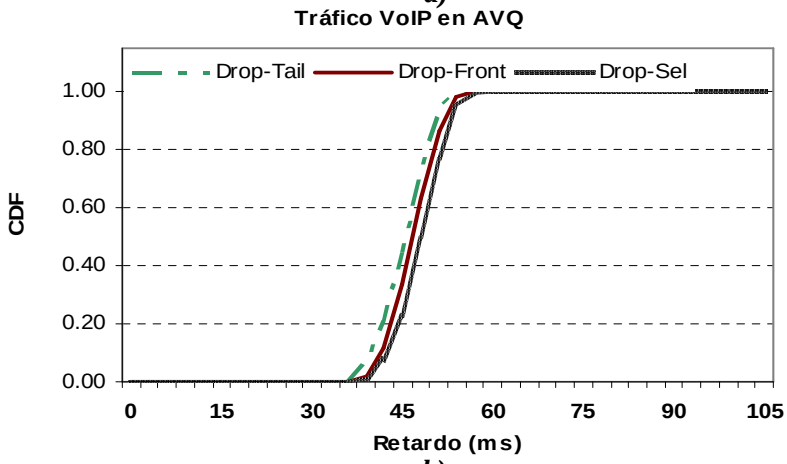


c)

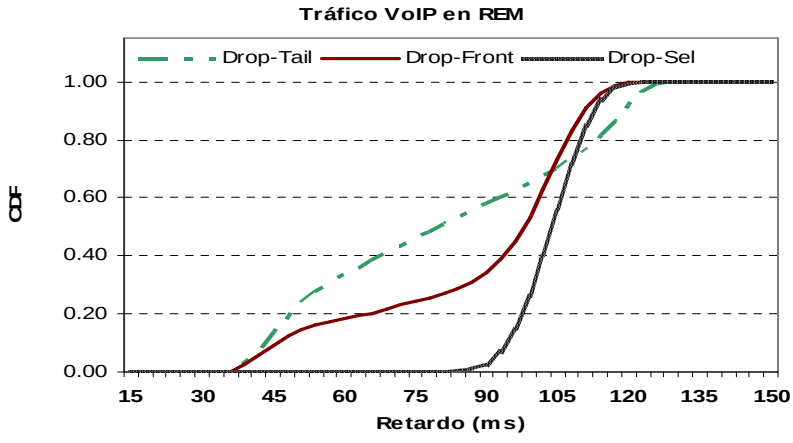
Figura 5.12. Evaluación de función de distribución acumulativa (CDF) del retardo extremo-a-extremo del paquete en el caso 1.



a)

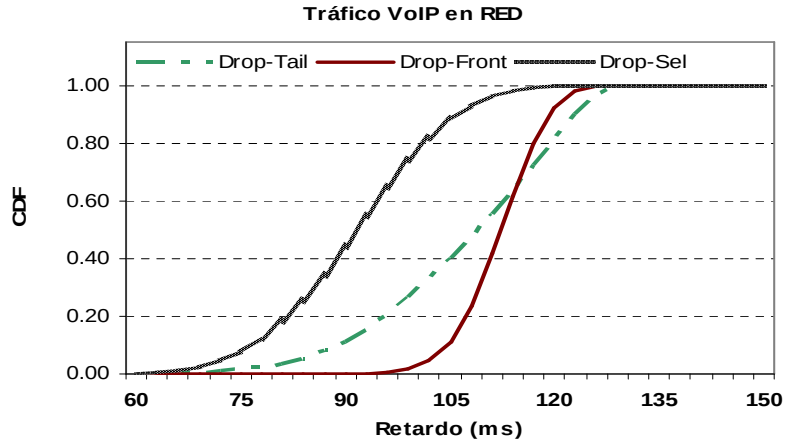


b)

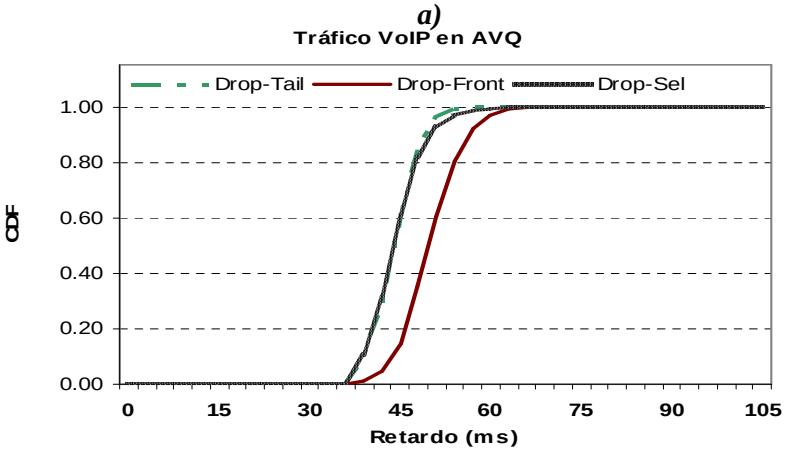


c)

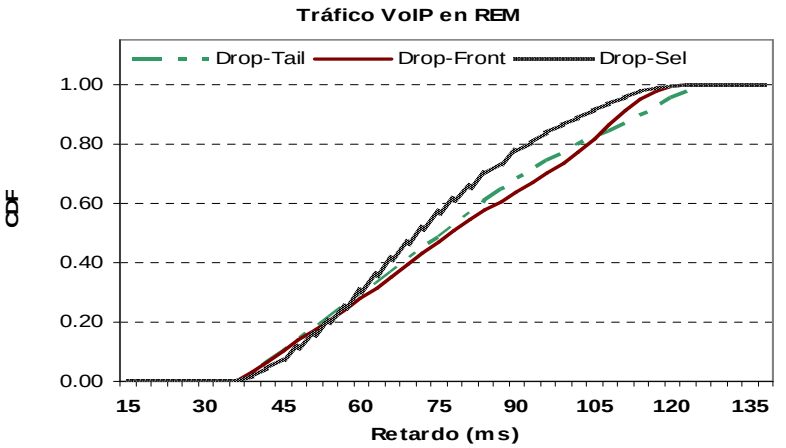
Figura 5.13. Evaluación de función de distribución acumulativa (CDF) del retardo extremo-a-extremo del paquete en el caso 2.



a)

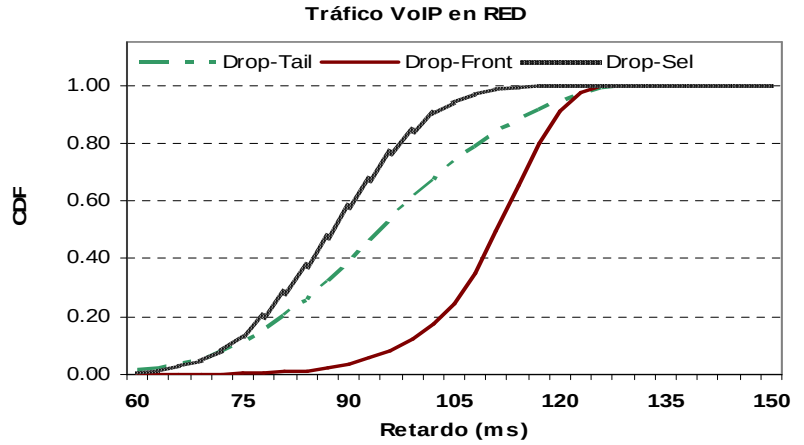


b)

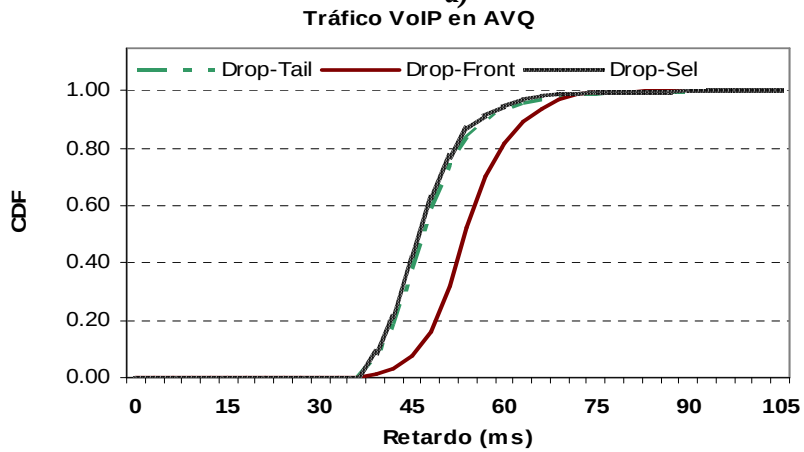


c)

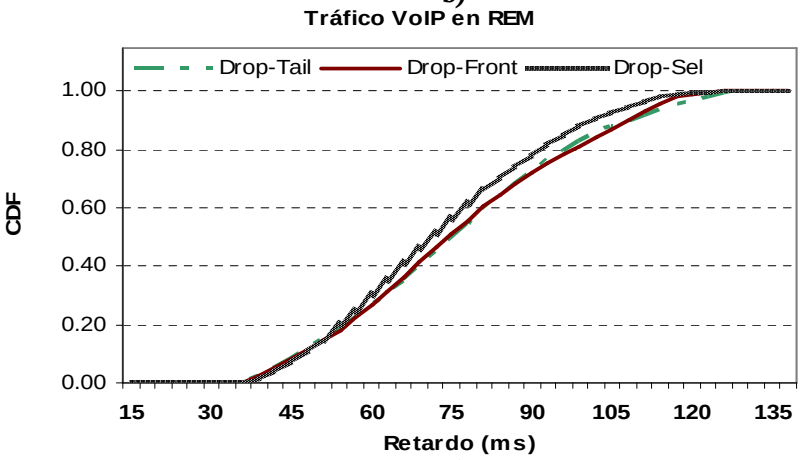
Figura 5.14. Evaluación de función de distribución acumulativa (CDF) del retardo extremo-a-extremo del paquete en el caso 3.



a)



b)



c)

Figura 5.15. Evaluación de función de distribución acumulativa (CDF) del retardo extremo-a-extremo del paquete en el caso 4.

Tabla 5.5.
RETARDO DE PAQUETES DE AUDIO PARA ESQUEMA RED EN ESCENARIO S1

	Algoritmos de selección de víctima								
	Drop-Tail			***Drop-Front***			***Drop-Sel***		
Caso	Retardo en cola (ms)	Retardo extremo-a-extremo(ms)	Varianza extremo-a-extremo	Retardo en cola (ms)	Retardo extremo-a-extremo(ms)	Varianza extremo-a-extremo	Retardo en cola (ms)	Retardo extremo-a-extremo(ms)	Varianza extremo-a-extremo
1	83	123	22e-03	68	108	42e-03	65	105	40e-03
2	84	124	19e-03	71	111	42e-03	67	107	48e-03
3	71	110	17e-02	75	114	34e-03	55	94	13e-02
4	58	97	25e-02	73	112	83e-02	51	90	12e-02

Tabla 5.6.
RETARDO DE PAQUETES DE AUDIO PARA ESQUEMA AVQ EN ESCENARIO S1

	Algoritmos de selección de víctima								
	Drop-Tail			***Drop-Front***			***Drop-Sel***		
Caso	Retardo en cola (ms)	Retardo extremo-a-extremo(ms)	Varianza extremo-a-extremo	Retardo en cola (ms)	Retardo extremo-a-extremo(ms)	Varianza extremo-a-extremo	Retardo en cola (ms)	Retardo extremo-a-extremo(ms)	Varianza extremo-a-extremo
1	9	49	20e-03	10	50	15e-03	7	47	15e-03
2	9	49	16e-03	10	50	14e-03	11	51	15e-03
3	8	47	16e-03	13	52	25e-03	8	47	21e-03
4	11	50	53e-03	18	57	47e-03	11	50	47e-03

Tabla 5.7.
RETARDO DE PAQUETES DE AUDIO PARA ESQUEMA REM EN ESCENARIO S1

	Algoritmos de selección de víctima								
	Drop-Tail			***Drop-Front***			***Drop-Sel***		
Caso	Retardo en cola (ms)	Retardo extremo-a-extremo(ms)	Varianza extremo-a-extremo	Retardo en cola (ms)	Retardo extremo-a-extremo(ms)	Varianza extremo-a-extremo	Retardo en cola (ms)	Retardo extremo-a-extremo(ms)	Varianza extremo-a-extremo
1	42	82	81e-02	42	82	53e-02	34	74	59e-02
2	44	84	83e-02	52	92	54e-02	67	107	50e-03
3	41	81	61e-02	41	81	58e-02	36	76	42e-02
4	39	79	48e-02	39	79	48e-02	36	76	39e-02

Obsérvese que esta tendencia también se manifiesta cuando REM adopta *Drop-Sel* (Tabla 5.7). Nuevamente, esta implementación no involucra un castigo significativo en el retardo medio del paquete. *Drop-Sel* aplicado a AVQ mantiene el retardo del paquete igual o ligeramente inferior al generado cuando se utiliza *Drop-Tail* (Tabla 5.6). Las funciones propias de AVQ ya permiten una reducción considerablemente en el retardo medio del paquete en relación a los esquemas RED y REM. Para caso 1, mostrado en la Figura 5.12b se observa que aproximadamente el 100% de los paquetes consume menos de 95 ms, independientemente del procedimiento de selección de víctima. Además, los paquetes en RED y REM (Figuras 5.12a y 5.12c) consumen aproximadamente hasta 130 ms.

Por el contrario, si el caso 2 es considerado (Figura 5.13), en el cual la carga del tráfico VoIP generado es la que domina (Tabla 5.2), obsérvese que el aumento del tráfico inducido por *Drop-Sel* (por la reducción en la penalización del tráfico reactivo) provoca que los paquetes consuman más tiempo para llegar al receptor que en *Drop-Tail*, especialmente en los esquemas AVQ (Tabla 5.6) y REM (Tabla 5.7). A pesar de que *Drop-Front* siempre considera descartar el paquete más próximo a salir de la cola, éste proporciona un mayor retardo extremo-a-extremo comparado con *Drop-Sel*.

Por ejemplo, en los casos 3 y 4 (Figuras 5.14 y 5.15), en los que las fuentes TCP dominan, *Drop-Front* no distingue entre clases de flujos y puede potencialmente reducir la penalización del tráfico reactivo haciendo que las fuentes TCP no reaccionen. Esto involucra un castigo significativo en el retardo medio del paquete.

Como conclusión, los resultados obtenidos indican que *Drop-Sel* reduce el retardo extremo-a-extremo respecto a *Drop-Front* y *Drop-Tail*. Este comportamiento puede ser explicado porque en promedio *Drop-Sel* selecciona la apropiada clase de tráfico a ser descartada y reduce el retardo medio en la cola del *router* (por las ventajas de descartar paquetes más próximos a la salida que en lugar de descartar los más cercanos a la entrada de la cola), independientemente de las condiciones de carga.

Tasa de pérdidas

Además de evaluar la latencia o retardo extremo-a-extremo, en términos generales y, para el tráfico de VoIP en particular, es también importante la evaluación de la tasa de pérdidas, dado que en el destino, la calidad final también depende de este factor. Existen dos causas por las cuales se pueden perder los paquetes: primeramente, los paquetes son descartados en el *router* AQM para notificar y prevenir congestión; y en segundo término, los paquetes no útiles (aquellos que acumulan un retardo extremo-a- extremo

superior a 300 ms) también son descartados en el usuario final.

Para mostrar el impacto de los procedimientos de selección de víctima de paquetes del AQM sobre la probabilidad de pérdidas para los paquetes de audio, las Tabla 5.8 y 5.9 proporcionan los resultados obtenidos para las diferentes cargas de trabajo de tráfico en la topología *dumbbell* (escenarios S1 y S2). Dado que los paquetes en los casos 1, 2, 3 y 4 acumulan un retardo extremo-a-extremo inferior a 130 ms (Tablas 5.5, 5.6 y 5.7), se genera una tasa de pérdidas en el usuario final con valor de cero. Por simplificación, estos resultados han sido omitidos en la Tabla 5.8. Nótese que en ella sólo se muestra la tasa de descarte en el AQM.

En términos de probabilidad de pérdidas total, RED ha sido el esquema menos favorecedor en todas las cargas de tráfico VoIP consideradas, reaccionando más lentamente a la congestión en comparación a REM y a AVQ. En consecuencia, como se observó en la sección anterior, el tiempo medio en la cola es más alto y por ello, la tasa de pérdidas en el usuario final es significativamente más alta. Por ejemplo, en RED si se considera el caso 6 (Tabla 5.9), la tasa de paquetes no-útiles en el usuario final para el esquema *Drop Tail* es 14.19%. Sin embargo, con algoritmos AVQ y REM, este valor cae a 0.12% y 9.05%, respectivamente.

Por otra parte, *Drop-Front* causa la más alta tasa de descarte de paquetes VoIP en la cola AQM, comparado a cualquier otro procedimiento de descarte (con excepción de *Drop-Sel* en los casos 2, 5 y 6 que penaliza intencionalmente a VoIP). Sin embargo, como AVQ reacciona a la congestión de forma más rápida que RED y REM, éste reduce significativamente la tasa total de pérdidas extremo-a-extremo.

La utilización del descarte selectivo en los casos 7 y 8, comparado con cualquiera de los otros procedimientos de descarte, produce los más bajos retardos en cola. En consecuencia, se minimiza significativamente el número de paquetes que acumulan más de 300 ms. Más precisamente, *Drop-Sel* minimiza la tasa en el usuario final para todos los esquemas AQM simulados. De hecho, la tasa de pérdidas en el usuario final con el esquema REM varia desde 4.04% a 8.71% al utilizar *Drop-Tail*, mientras que con *Drop-Sel*, sólo desde 1.41% a 2.89%. Un beneficio más significativo se obtiene en RED para estos casos, la tasa de pérdidas varía desde 5.63% a 11.84% con *Drop-Tail* y con *Drop-Sel*, se reduce a un rango desde 3.23% a 4.24%.

Adicionalmente, como una tendencia general, se reduce la tasa de descarte en la cola para los paquetes de audio cuando *Drop-Sel* identifica a los flujos VoIP como la clase de tráfico que consume menos espacio de memoria en cola durante los períodos de congestión. En este caso, el mecanismo ofrece potencial garantía de funcionamiento para

Tabla 5.8.
TASA TOTAL DE PÉRDIDAS DE PAQUETES DE AUDIO EN ESCENARIO S1

Caso	Esquema AQM (Método de selección de víctima)								
	RED			**AVQ**			**REM**		
	D-Tail	*D-Front*	*D-Sel*	*D-Tail*	*D-Front*	*D-Sel*	*D-Tail*	*D-Front*	*D-Sel*
1	8.96	22.69	0.00	1.67	19.31	0.58	2.23	11.26	0.68
2	13.44	19.61	43.00	9.40	13.59	23.98	3.28	11.62	37.68
3	3.15	13.00	0.04	2.58	10.19	0.62	1.43	7.02	0.22
4	1.10	10.12	0.00	0.76	7.89	0.00	0.88	6.72	0.02

Tabla 5.9.
TASA TOTAL DE PÉRDIDAS DE PAQUETES DE AUDIO EN ESCENARIO S2

		Esquema AQM (Método de selección de víctima)								
		RED			**AVQ**			**REM**		
Caso		*D-Tail*	*D-Front*	*D-Sel*	*D-Tail*	*D-Front*	*D-Sel*	*D-Tail*	*D-Front*	*D-Sel*
	cola	21.03	23.79	54.01	2.36	24.19	29.37	5.23	26.45	53.21
5	usuario	11.94	7.13	4.33	0.07	0.40	0.57	7.91	5.88	3.90
	Total	32.97	30.92	58.34	2.43	24.59	29.94	13.14	32.33	57.11
	cola	7.88	11.59	48.17	1.26	16.57	38.94	3.17	16.98	47.82
6	usuario	14.19	13.29	5.60	0.12	0.19	0.69	9.05	11.99	5.26
	Total	22.07	24.88	53.77	1.38	16.76	39.63	12.22	28.97	53.08
	cola	8.80	16.14	0.00	0.35	15.51	0.00	2.22	13.64	0.74
7	usuario	11.84	11.78	4.24	0.43	0.39	0.00	8.71	6.23	1.41
	Total	20.64	27.92	4.24	0.78	15.90	0.00	10.93	19.87	2.15
	cola	1.21	4.31	0.26	0.18	5.80	0.02	0.72	5.56	0.28
8	usuario	5.63	11.29	3.23	0.12	0.07	0.10	4.04	8.19	2.89
	Total	6.84	15.60	3.49	0.30	5.87	0.12	4.76	13.75	3.17

la clase de tráfico VoIP sin considerar ningún esquema de reserva de recursos explícita. Por ejemplo, dado el caso 7, mientras *Drop-Tail* en RED produce una tasa de descarte en la cola de 8.80%, *Drop-Sel* reduce ésta a 0.00%. Similarmente, si se consideran los casos 1, 3, 4 y 8, obsérvese que nuevamente el método *Drop-Sel* permite reducir la tasa de descarte en el *router* AQM y así aumentar la tasa de paquetes útiles en el receptor sin importar que tan lejos estén las fuentes del *router* AQM.

Al contrario, cuando la carga del tráfico VoIP generado es la que domina, *Drop-Sel* aumenta su probabilidad de descarte en la cola. Por ejemplo, si se considera el caso 6, RED con *Drop-Tail* produce una tasa de descarte de 7.88%. Nótese que al adoptar *Drop-*

Tabla 5.10.
TASA TOTAL DE PÉRDIDAS DE PAQUETES DE AUDIO EN ESCENARIO S3

Caso		Esquema AQM (Método de selección de víctima)								
		RED			**AVQ**			**REM**		
		D-Tail	*D-Front*	*D-Sel*	*D-Tail*	*D-Front*	*D-Sel*	*D-Tail*	*D-Front*	*D-Sel*
9	cola (A)	1.44	5.16	0.00	0.05	5.65	0.01	0.47	4.81	0.10
	cola (B)	12.02	14.12	0.00	1.87	11.99	0.01	2.00	13.82	0.12
	usuario	8.86	1.85	0.00	0.00	0.00	0.00	8.29	1.11	0.20
	Total	22.32	21.13	0.00	1.92	17.64	0.02	10.76	19.74	0.42
10	cola (A)	0.01	0.01	0.00	0.01	0.00	0.00	0.00	0.00	0.00
	cola (B)	9.61	12.15	0.00	2.14	11.07	0.03	1.88	10.32	0.05
	usuario	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	Total	9.62	12.16	0.00	2.15	11.07	0.03	1.88	10.32	0.05

Sel se penaliza significativamente al tráfico con un 48.17%, un aumento considerable (40.29%) pero necesario para mantener la equitatividad. Con el mismo propósito en el caso 2, *Drop-Sel* aumenta respectivamente un 29.56%, 14.58% y 34.40% la tasa de descarte en RED, AVQ y REM.

Por otro lado, el impacto de los procedimientos de selección de víctima sobre la probabilidad de pérdidas de los paquetes de audio es similar cuando el escenario S3 es utilizado. Básicamente, como se observa en Tabla 5.10, *Drop-Sel* nuevamente proporciona un mejor tratamiento a la clase de tráfico que consume menos recursos.

Los resultados obtenidos de estos experimentos en combinación con los previos corroboran que si un AQM observa el tráfico de la red y selecciona el paquete a descartar del tráfico más pesado, se puede proporcionar mejor QoS al tráfico VoIP en términos de retardo extremo-a-extremo y probabilidad de tasa de pérdida de paquetes, bajo condiciones de tráfico para las cuales las fuentes VoIP no dominen.

5.4.3. Calidad subjetiva (QoE, Quality of Experience) en VoIP

Con el propósito de medir el rendimiento en términos de calidad percibida por el usuario final, obtenida con *Drop-Sel*, se adoptó un reconocedor automático de voz extremo-a-extremo y el Modelo-E. Los resultados experimentales demuestran que el algoritmo propuesto de selección de víctima mejora significativamente al tráfico VoIP en términos de puntuación MOS e inteligibilidad.

MOS

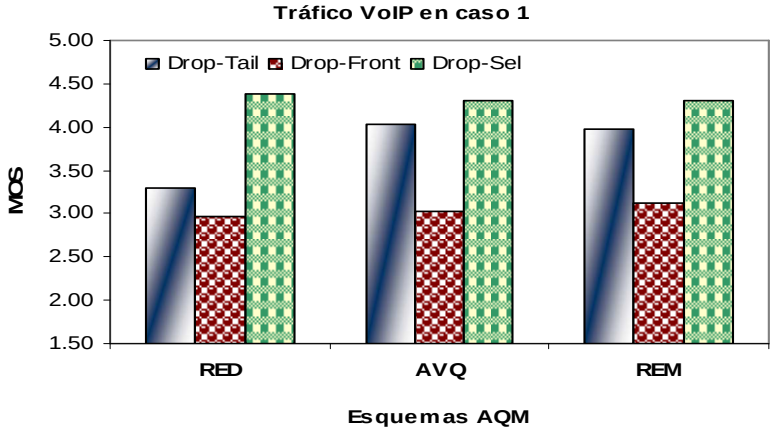
Para proporcionar evaluaciones subjetivas más intuitivas, el factor R es expresado en términos de puntuación MOS (apartado 3.3.2). En este sentido, las Figuras 5.16, 5.17, 5.18 y 5.19 muestran los valores MOS obtenidos para las diferentes condiciones de carga de tráfico y diferentes métodos de descarte.

En las Tablas 5.8, 5.9 y 5.10 se puede observar que RED y REM experimentan tasas más altas de descarte en la cola y en el usuario final que AVQ (excepto REM en el caso 2 y 3). Considerando que las pérdidas de paquetes es uno de los factores que influyen firmemente en la degradación de la calidad de voz y por ende, en la calidad subjetiva percibida por los usuarios, no es de extrañar que los valores MOS obtenidos con estos dos esquemas sean más bajos que los generados en AVQ. De hecho, los resultados de los experimentos muestran que RED con *Drop-Tail* y *Drop-Front* comparado con AVQ y REM obtienen las puntuaciones MOS más baja (valores desde 1.77 a 4.22), independientemente de la carga de tráfico UDP. No obstante, en los casos 1, 3, 4, 7, 8, 9 y 10, *Drop-Sel* proporciona la mejor calidad con valores MOS de hasta 4.38.

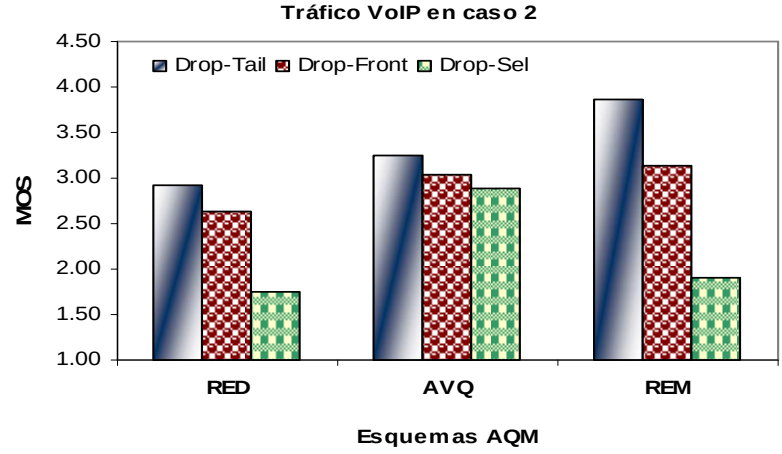
Como REM es menos susceptible a la tasa de pérdidas que RED, el servicio proporcionado a los flujos VoIP en términos de calidad percibida, es superior. REM con *Drop-Tail* y *Drop-Front* obtiene valores MOS desde 1.82 hasta 4.22. El utilizar el método de descarte selectivo en REM, hace posible que en los casos 1, 3, 4, 7, 8, 9 y 10 se provea a los flujos de VoIP una mejor calidad a nivel de usuario al igual que lo hizo en RED. Esencialmente, genera valores MOS hasta del 4.39.

En relación a la evaluación del retardo y tasa de pérdidas reportados en apartado 5.4.2, el esquema AVQ con *Drop-Tail* y *Drop-Front* proporciona la mejor calidad subjetiva respecto a los otros dos esquemas. No obstante, aquí nuevamente *Drop-Sel* mejora la calidad subjetiva de VoIP en todas las condiciones de tráfico simuladas. En la mejor de las situaciones para *Drop-Tail* en AVQ (valor MOS igual a 4.20 en caso 4), *Drop-Sel* aún consigue aumentar 0.40 el valor MOS (valor MOS igual a 4.40).

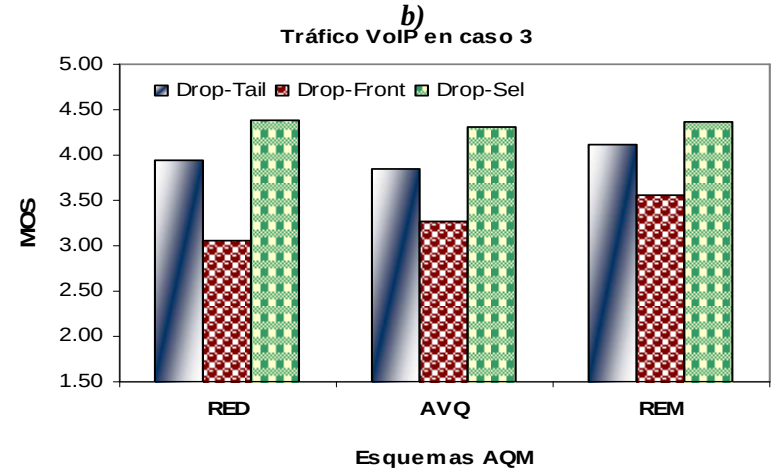
Drop-Sel proporciona la mejor equidad cuando diferentes fuentes de tráfico reactivas y no-reactivas compiten por los mismos recursos de la red. Sin embargo, es imposible conseguir en todos los casos la equidad sin menoscabar la calidad objetiva y subjetiva del tráfico VoIP. En este sentido, la adopción del algoritmo propuesto bajo condiciones de tráfico para las cuales las fuentes VoIP dominen, no es muy favorable para la calidad de servicio del tráfico VoIP. Esto es corroborado con los resultados expuestos en las Figuras 5.16b, 5.17a y 5.17b donde los casos 2, 5 y 6 son considerados.



a)

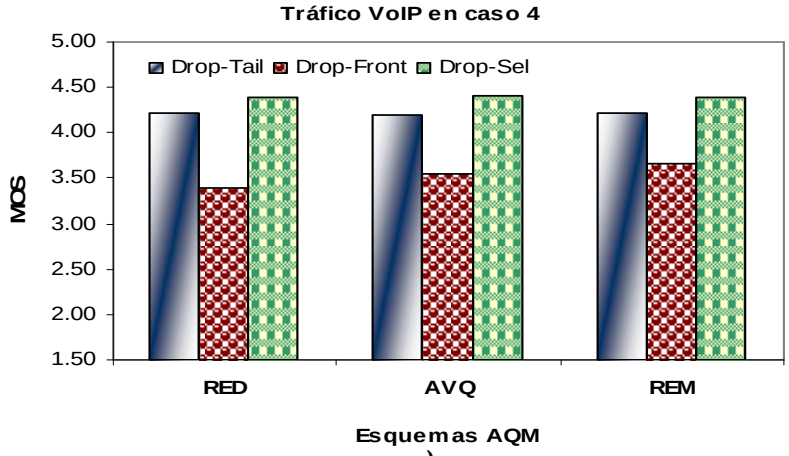


b)

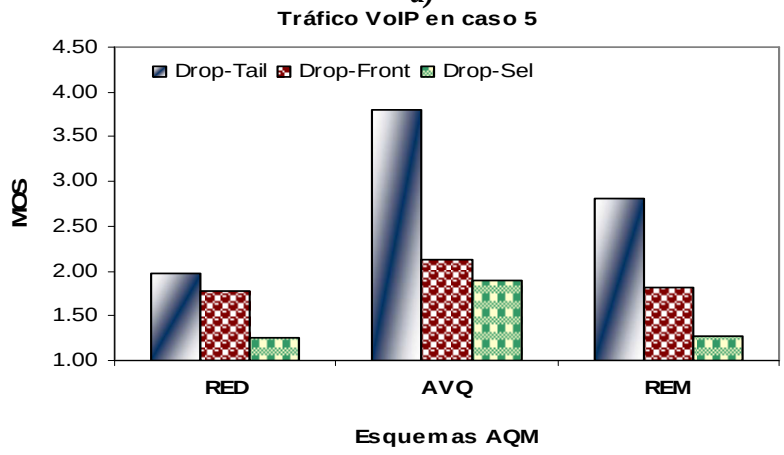


c)

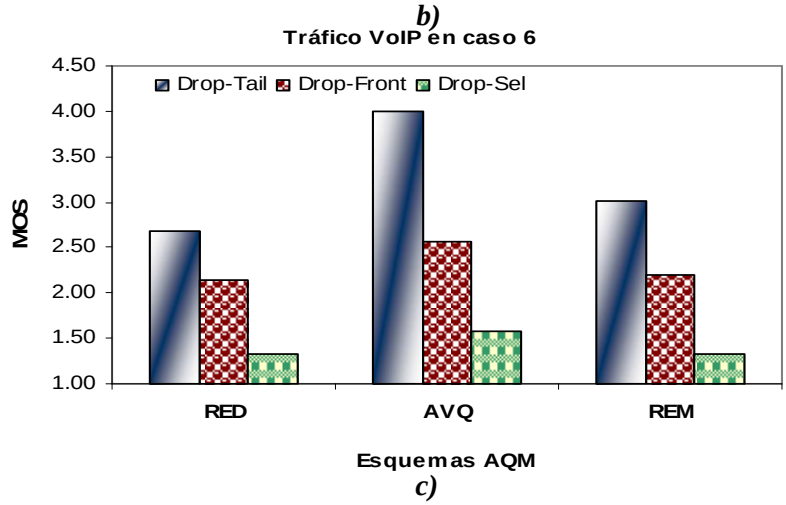
Figura 5.16. Evaluación MOS basado en Modelo-E de *Drop-Front*, *Drop-Tail* y *Drop-Sel* en los casos 1, 2 y 3.



a)

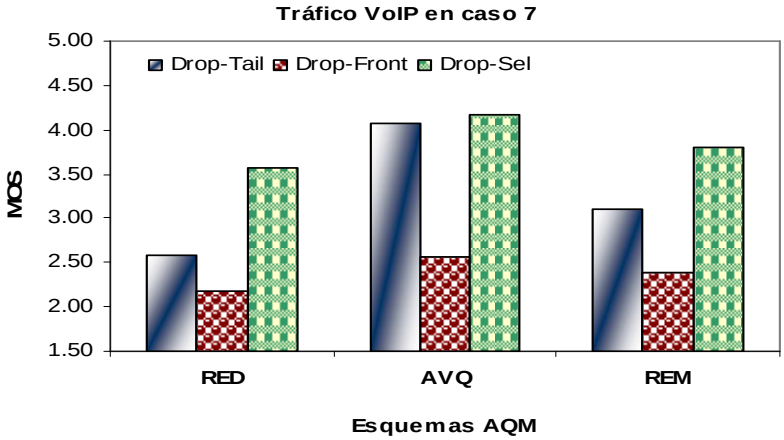


b)

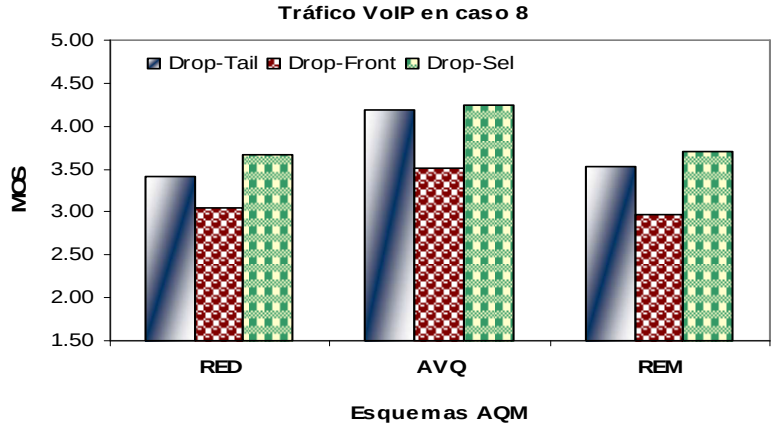


c)

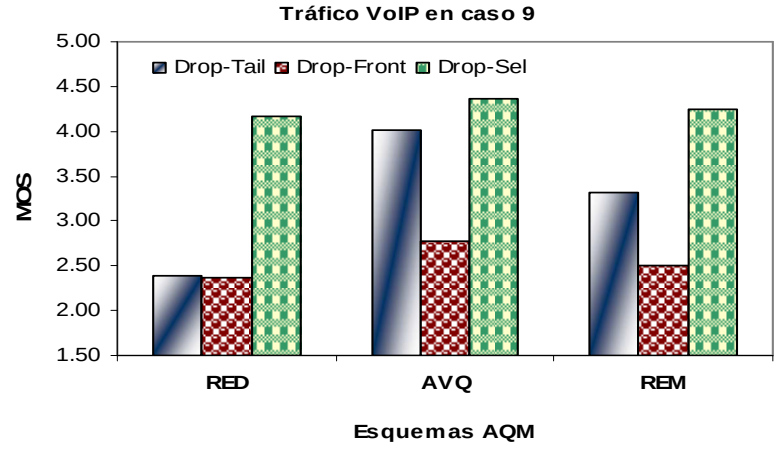
Figura 5.17. Evaluación MOS basado en Modelo-E de *Drop-Front*, *Drop-Tail* y *Drop-Sel* en los casos 4, 5 y 6.



a)



b)



c)

Figura 5.18. Evaluación MOS basado en Modelo-E de *Drop-Front*, *Drop-Tail* y *Drop-Sel* en los casos 7, 8 y 9.

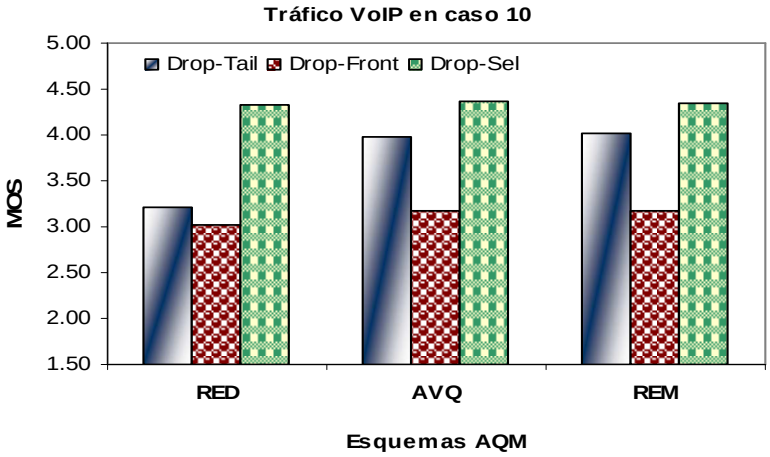


Figura 5.19. Evaluación MOS basado en Modelo-E de *Drop-Front*, *Drop-Tail* y *Drop-Sel* en el caso 10.

La magnitud del deterioro de la calidad del tráfico VoIP está directamente relacionada con los recursos que son otorgados a otras fuentes UDP y TCP al igualar los recursos consumidos de los tres tipos de tráficos. Esto sugiere que la aplicación de *Drop-Sel* preferiblemente debe regularse con un mecanismo de disparo para que pueda permutar a otra de las estrategias tradicionales de selección de víctima (por ejemplo *Drop-Tail*), cuando las fuentes VoIP dominen.

Inteligibilidad

La tasa de palabras o precisión de frases está directamente relacionada con la inteligibilidad final percibida por el usuario. Por lo que, incrementar la inteligibilidad es un objetivo para cualquier algoritmo de transmisión de voz que se proponga. De ahí que esta metodología de evaluación es una herramienta esencial para comprobar las mejoras obtenidas por el esquema de selección de víctima propuesto en esta investigación.

En este sentido se evalúa el funcionamiento de *Drop-Sel* utilizando en el receptor el reconocedor automático de voz descrito en el apartado 3.4.3 y 5.3.1. En este caso, se han asumido los esquemas AQM con la carga del caso 7 (Tabla 5.3) del escenario S2. En particular, un número de fuentes TCP y otros flujos UDP no-reactivos compiten con 10 flujos VoIP (UDP). Específicamente, 2 de los flujos VoIP (A1 y A2) son generados utilizando la aplicación de VoIP tipo A (ver Tabla 5.1), 3 con la aplicación tipo B (B1, B2 y B3), 2 con la aplicación tipo C (C1 y C2) y finalmente, 3 de los flujos VoIP con la

Tabla 5.11.
TASA DE PRECISIÓN DE PALABRAS DE PAQUETES DE AUDIO POR FLUJO PARA CASO 7 EN ESCENARIO S2

	Esquema AQM (Método de selección de víctima)								
	RED			**AVQ**			**REM**		
Flujo	*D-Tail*	*D-Front*	*D-Sel*	*D-Tail*	*D-Front*	*D-Sel*	*D-Tail*	*D-Front*	*D-Sel*
A1	98.58	98.18	99.02	99.02	97.88	99.02	98.96	98.31	98.99
A2	98.73	98.21	99.02	99.02	98.42	99.02	99.00	98.56	99.02
B1	98.60	97.39	99.02	99.01	96.77	99.02	98.95	97.71	98.97
B2	98.35	97.57	99.02	99.02	96.49	99.02	98.94	97.82	98.99
B3	98.50	96.89	99.02	98.99	96.44	99.02	98.90	97.38	98.92
C1	96.70	93.44	99.02	98.78	96.38	99.02	97.65	93.87	98.56
C2	96.98	91.42	99.02	98.59	95.53	99.02	97.65	92.88	98.49
D1	78.54	38.85	90.50	98.04	91.48	98.59	85.18	71.77	97.19
D2	78.30	37.81	90.43	98.09	91.34	98.62	84.98	70.79	96.87
D3	78.55	39.41	91.11	98.10	94.77	98.62	85.40	72.76	97.10
Promedio	92.18	78.92	96.52	98.67	95.55	98.90	94.56	89.19	98.31

Tabla 5.12.
MEJORA RELATIVA DE PRECISIÓN DE PALABRAS RESPECTO A *DROP -TAIL* PARA CADA FLUJO

	Esquema AQM (Método de selección de víctima)					
	RED		**AVQ**		**REM**	
Flujo	*D-Front*	*D-Sel*	*D-Front*	*D-Sel*	*D-Front*	*D-Sel*
A1	-0.41	0.45	-1.15	0.00	-0.66	0.03
A2	-0.53	0.29	-0.61	0.00	-0.44	0.02
B1	-1.23	0.43	-2.26	0.01	-1.25	0.02
B2	-0.79	0.68	-2.56	0.00	-1.13	0.05
B3	-1.63	0.53	-2.58	0.03	-1.54	0.02
C1	-3.37	2.40	-2.43	0.24	-3.87	0.93
C2	-5.73	2.10	-3.10	0.44	-4.88	0.86
D1	-50.53	15.23	-6.69	0.56	-15.74	14.10
D2	-51.71	15.49	-6.88	0.54	-16.70	13.99
D3	-49.83	15.99	-3.39	0.53	-14.80	13.70

aplicación tipo D (D1, D2 y D3).

Las tasas de precisión de palabras obtenidas de la ecuación 3.3 son mostradas en Tabla 5.11. En general, los valores obtenidos por el esquema *Drop-Sel* son superiores o iguales a los obtenidos por *Drop-Tail* y *Drop-Front*. Se observa que los flujos más afectados por el retardo extremo-a-extremo, es decir los flujos D1, D2 y D3, obtienen los peores resultados cuando *Drop-Tail* es adoptado en RED, valores de 78.54%, 78.30% y 78.55% respectivamente. Sin embargo estos son mejorados significativamente por *Drop-*

Tabla 5.13.
TASA DE FRASES CORRECTAS DE PAQUETES DE AUDIO POR FLUJO PARA CASO 7 EN ESCENARIO S2

	Esquema AQM (Método de selección de víctima)								
	RED			**AVQ**			**REM**		
Flujo	*D-Tail*	*D-Front*	*D-Sel*	*D-Tail*	*D-Front*	*D-Sel*	*D-Tail*	*D-Front*	*D-Sel*
A1	95.73	94.78	97.08	97.08	93.79	97.08	96.85	95.06	96.98
A2	96.20	94.73	97.08	97.08	95.38	97.08	97.00	95.68	97.08
B1	95.90	92.49	97.08	97.08	90.84	97.08	96.88	93.26	96.90
B2	95.08	93.11	97.08	97.08	90.41	97.08	96.85	93.66	97.03
B3	95.55	91.04	97.08	96.98	90.41	97.08	96.70	92.36	96.78
C1	90.61	82.85	97.08	96.28	89.74	97.08	93.06	83.37	95.70
C2	91.46	78.80	97.08	95.75	87.81	97.05	92.86	81.62	95.50
D1	53.92	18.91	76.47	94.71	78.95	95.88	76.27	57.07	92.96
D2	53.52	18.86	76.72	94.86	78.47	96.05	75.79	56.67	92.21
D3	54.09	19.86	78.02	94.81	86.31	96.00	76.20	57.84	93.01
Promedio	82.21	68.54	91.08	96.17	88.21	96.75	89.85	80.66	95.42

Tabla 5.14.
MEJORA RELATIVA DE FRASES CORRECTAS RESPECTO A *DROP-TAIL* PARA CADA FLUJO

	Esquema AQM (Método de selección de víctima)					
	RED		**AVQ**		**REM**	
Flujo	*D-Front*	*D-Sel*	*D-Front*	*D-Sel*	*D-Front*	*D-Sel*
A1	-0.99	1.41	-3.39	0.00	-1.85	0.13
A2	-1.53	0.91	-1.75	0.00	-1.36	0.08
B1	-3.56	1.23	-6.43	0.00	-3.74	0.02
B2	-2.07	2.10	-6.87	0.00	-3.29	0.19
B3	-4.72	1.60	-6.77	0.10	-4.49	0.08
C1	-8.56	7.14	-6.79	0.83	-10.41	2.84
C2	-13.84	6.14	-8.29	1.36	-12.10	2.84
D1	-64.93	41.82	-16.64	1.24	-25.17	21.88
D2	-64.76	43.35	-17.28	1.25	-25.23	21.67
D3	-63.28	44.24	-8.97	1.26	-24.09	22.06

Sel (90.50%, 90.43% y 91.11% respectivamente), tal como lo confirman los resultados de la Tabla 5.12 que muestran la mejora relativa para cada flujo respecto a *Drop-Tail*.

Específicamente, *Drop-Sel* respecto a *Drop-Tail*, obtiene una mejora relativa superior al 15% para estos tres flujos. Aunque las mejoras de *Drop-Sel* en los otros dos esquemas AQM no son tan prominentes como en RED, se pueden apreciar para estos flujos en particular, mejoras hasta de 0.56% en AVQ y de 14.10% en REM. Nótese que el incremento de la tasa de pérdidas en el *router* AQM al adoptar *Drop-Front* (Tabla 5.9),

degrada significativamente la inteligibilidad generada por *Drop-Tail* para cada uno de los flujos. Resultando más eficiente el descarte selectivo de paquetes respecto a *Drop-Front*.

Además de la tasa de palabras, la tasa de frases correctas proporciona una medida complementaria de inteligibilidad. Se dice que una frase es correctamente reconocida cuando no se produce ninguna inserción o sustitución de palabras en ella. En este sentido, la Tabla 5.13 muestra las tasas de frases correctas obtenidas para el caso 7. Como se observa, *Drop-Sel* mantiene la inteligibilidad igual o mejor que *Drop-Tail* y *Drop-Front*. Por tanto, este experimento muestra que el esquema propuesto de selección de víctima proporciona mejor inteligibilidad extremo-a-extremo. Este hecho es nuevamente más significativo para los flujos con la peor calidad, es decir, para flujos de voz generados de fuentes más distantes.

Más específicamente, *Drop-Tail* en RED respectivamente proporciona una tasa del 53.92%, 53.52% y 54.09% para los flujos D1, D2 y D3. Nótese que *Drop-Sel* respectivamente incrementa las tasas de frases correctas a un 76.47%, un 76.72% y un 78.06%. Similarmente, el comportamiento de *Drop-Sel* es también mantenido para el esquema REM, haciendo posible que respectivamente obtengan mejoras relativas de 21.88%, 21.67% y 22.06 para los flujos D1, D2 y D3 (Tabla 5.14).

La mejora en la inteligibilidad extremo-a-extremo de los flujos VoIP, como fue previamente observado en la evaluación sobre equidad entre clases de tráfico (Figura 5.9c), no repercute significativamente en el servicio proporcionado a los flujos TCP y otros flujos UDP.

Con estos resultados se corrobora experimentalmente que el algoritmo propuesto *Drop-Sel* es una prometedora alternativa, la cual es capaz de mejorar significativamente la QoS para flujos de audio (en el caso de concurrir tráfico de protocolo TCP con UDP), en término de calidad de voz percibida extremo-a-extremo.

Por último, mencionar que los resultados extraídos muestran una correspondencia entre la evaluación a nivel de red y la calidad estimada a nivel de usuario. Se concluye, por consiguiente, que *Drop-Sel* no sólo mejora la calidad objetiva QoS si no también la estimada calidad subjetiva de experiencia QoE, bajo condiciones de tráfico para las cuales las fuentes VoIP no dominen.

5.4.4. Selección automática del algoritmo

Los resultados de los apartados 5.4.2 y 5.4.3 sugieren que la aplicación de *Drop-Sel* debe regularse con un mecanismo de disparo para que pueda permutar a otra de las estrategias de selección de víctima cuando las fuentes VoIP dominen.

Se han realizado dos pruebas de concepto que consisten en simular de forma concatenada el *caso 3 - caso 2 - caso 3* del escenario S1 y el *caso 8 - caso 6 - caso 8* del escenario S2, considerando un sencillo algoritmo de selección que monitoriza la clase dominante para determinar si debe permutar de *Drop-Sel* a *Drop-Tail* y de *Drop-Tail* a *Drop-Sel* (denominando al esquema de aquí en adelante como *Drop-Sel(PT)*).

Como resultado preliminar en este apartado se comprueba experimentalmente que el sencillo mecanismo de disparo propuesto para adaptarse a las diferentes condiciones de carga en la red puede paliar las deficiencias detectadas en *Drop-Sel* cuando los flujos VoIP son dominantes.

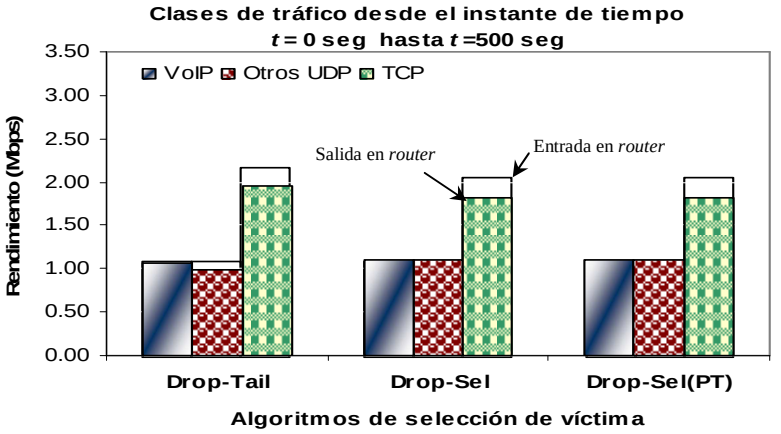
Equidad entre clases de tráfico

Se ha examinado el efecto sobre el rendimiento medio para las distintas clases de tráfico en tres diferentes períodos de la simulación total. Por cada período de tiempo se adopta alguno de los casos concatenados. Para el primer (desde t=0 seg hasta t=500 seg) y el tercer intervalo (desde t=1000 seg hasta t=1500 seg) se adopta el caso 3 si es considerado el escenario S1 y el caso 8 para el escenario S2. Para el segundo intervalo (desde t=500 seg hasta t=1000 seg) se adopta el caso 2 si es considerado el escenario S1 y el caso 6 para el escenario S2. El objetivo de este experimento es modificar las condiciones de carga y verificar que la selección automática permite adaptarse a ellas.

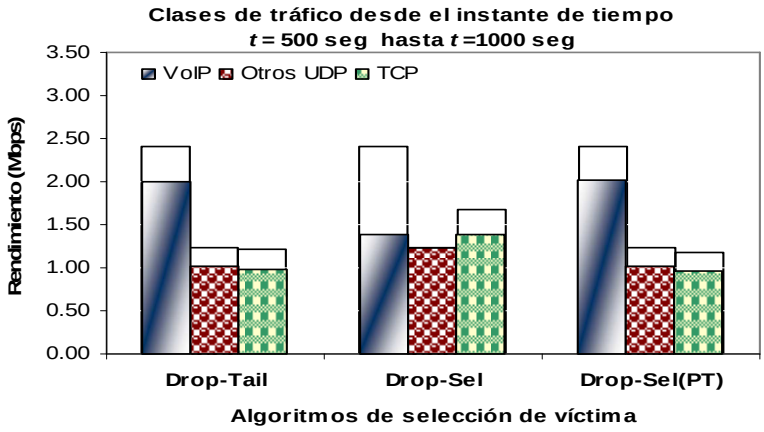
En las Figuras 5.20 y 5.21 se han establecido el rendimiento medio de entrada y de salida de *Drop-Sel(PT)* en el esquema AQM RED y para propósitos de comparación los resultados con *Drop-Tail* y *Drop-Sel* de referencia son también representados.

Primeramente, para el escenario S1 durante el primer intervalo, se observa que *Drop-Sel(PT)* es más equitativo que *Drop-Tail* (Figura 5.20a). En este período de tiempo, *Drop-Sel(PT)* detecta que el tráfico VoIP no es la clase dominante y mantiene el comportamiento equitativo de *Drop-Sel* de referencia.

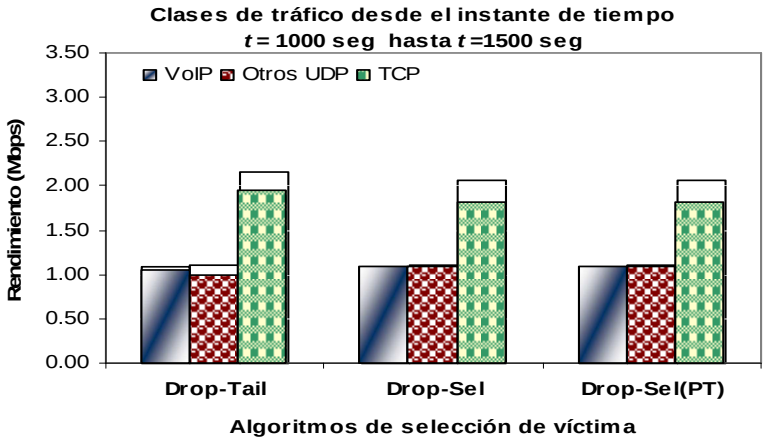
Sin embargo, en el segundo intervalo *Drop-Sel(PT)* detecta que la clase dominante es el tráfico VoIP y determina que debe permutar de *Drop-Sel* a *Drop-Tail*.



a)

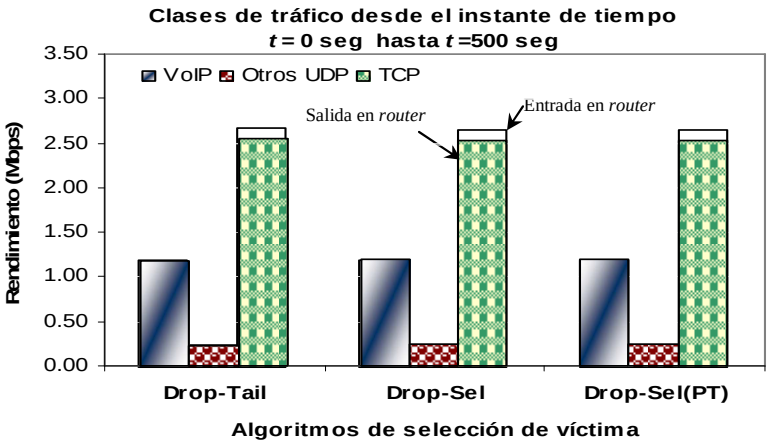


b)

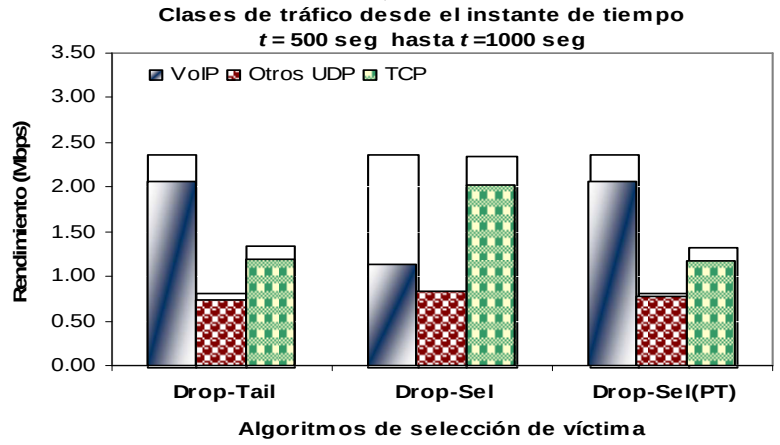


c)

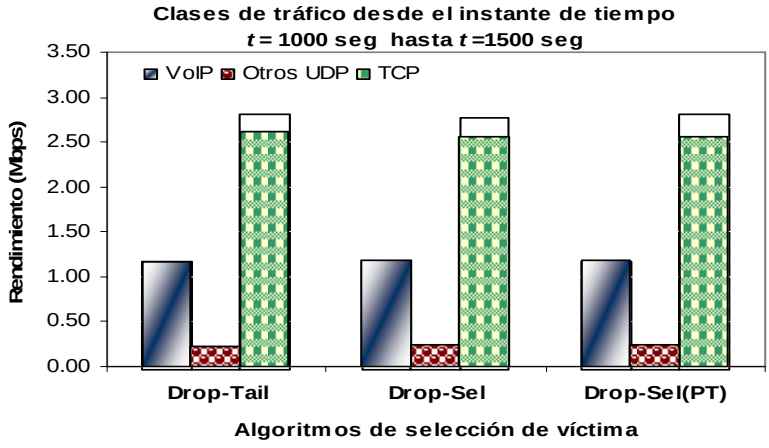
Figura 5.20. Evaluación del rendimiento medio de diferentes clases de tráfico en los casos concatenados 3, 2 y 3 del S1 con RED.



a)



b)



c)

Figura 5.21. Evaluación del rendimiento medio de diferentes clases de tráfico en los casos concatenados 8, 6 y 8 del S2 con RED.

Tabla 5.15.
TASA TOTAL DE PÉRDIDAS DE PAQUETES DE AUDIO EN CASOS CONCATENADOS

	Método de selección de víctima								
	0-500 seg			***500-1000 seg***			***1000 a 1500 seg***		
Casos	***D-Tail***	***D-Sel***	***D-Sel (PT)***	***D-Tail***	***D-Sel***	***D-Sel (PT)***	***D-Tail***	***D-Sel***	***D-Sel (PT)***
Router									
Total S1	3.15	0.04	0.04	13.50	42.38	13.00	3.25	0.04	0.04
Router	1.33	0.25	0.25	13.47	52.26	13.32	2.77	0.32	0.31
usuario	5.73	3.08	3.08	13.95	4.26	13.85	10.48	6.99	8.00
Total S2	7.06	3.33	3.33	27.42	56.52	27.17	13.25	6.31	8.31

En la Figura 5.20b se observa que *Drop-Sel(PT)* mantiene el mismo comportamiento de *Drop-Tail* hasta el principio del tercer período. En este último intervalo, *Drop-Sel(PT)* nuevamente adopta el comportamiento equitativo de *Drop-Sel* (Figura 5.20c).

Similarmente, en el escenario S2, *Drop-Sel(PT)* tiende a ser igual de imparcial que *Drop-Sel* (en el primer y tercer período) para los tres tráficos independientemente de la naturaleza reactiva de las fuentes contendientes (Figura 5.21a y Figura 5.21c). A partir del instante de tiempo t>500 hasta t=1000 seg *Drop-Sel(PT)* adopta el comportamiento de *Drop-Tail* (Figura 5.21b).

Tasa de pérdidas

Para mostrar el impacto de los procedimientos de selección de víctima de paquetes del AQM sobre la probabilidad de pérdidas para los paquetes de audio, la Tabla 5.15 proporciona los resultados obtenidos para los escenarios S1 y S2, cada uno de ellos con los diferentes casos concatenados.

Dado que los paquetes en los casos 2 y 3 del escenario S1 acumulan un retardo extremo-a-extremo inferior a 130 ms, se genera una tasa de pérdidas en el usuario final con valor de cero. Por simplificación, estos resultados han sido omitidos en la Tabla 5.15. Nótese que en ella sólo se muestra la tasa de descarte en el AQM.

En términos de probabilidad de pérdidas totales, *Drop-Sel(PT)* ha sido el esquema más favorecedor en los dos escenarios en comparación a *Drop-Tail* y *Drop-Sel* de referencia. *Drop-Sel(PT)*, en cada período de tiempo, intenta adoptar la estrategia adecuada para favorecer al tráfico VoIP. En consecuencia, como se observa en la Tabla 5.15, *Drop-Sel(PT)* proporciona una tasa de pérdidas en el *router* AQM igual o menor que los otros dos procedimientos.

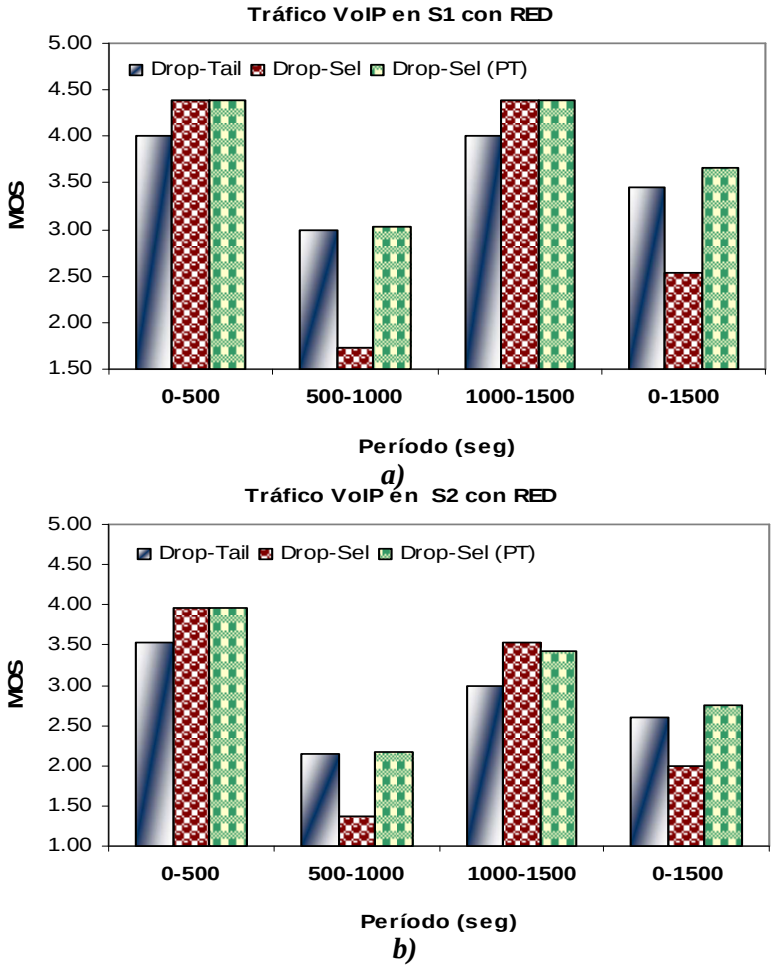


Figura 5.22. Evaluación MOS basado en Modelo-E de *Drop-Tail*, *Drop-Sel* y *Drop-Sel(PT)* en los casos concatenados de S1 y S2.

MOS

Con el propósito de medir el rendimiento en términos de calidad percibida por el usuario final, el esquema *Drop-Sel(PT)* se evalúa en términos de puntuación MOS.

La Figura 5.22 muestra los valores MOS obtenidos en los diferentes períodos de tiempo utilizando los métodos de descarte selectivo *Drop-Tail*, *Drop-Sel* de referencia y *Drop-Sel(PT)*, en los casos concatenados de los escenarios S1 y S2.

Considerando que las pérdidas de paquetes es uno de los factores que influyen firmemente en la degradación de la calidad de voz y por ende, en la calidad subjetiva

percibida por los usuarios, no es de extrañar que los valores MOS obtenidos por *Drop-Sel(PT*) en cada período sean similares a los del esquema adoptado en cada uno de ellos.

En este sentido, como conclusión se observa que la aplicación de *Drop-Sel* regulada con un mecanismo de disparo que le permita permutar a otra de las estrategias tradicionales de selección de víctima (en este caso *Drop-Tail*), cuando los flujos VoIP son dominantes, mejora la calidad subjetiva de los flujos de audio.

5.5. Conclusiones de *Drop-Sel*

En este capítulo se ha analizado y evaluado bajo un rango de amplias posibilidades, el impacto de incluir las peculiaridades de las diferentes clases de tráfico en el procedimiento de selección de víctima para distintos esquemas AQM. En este sentido, se ha considerado como referencia el procedimiento referido como *Drop-Sel.*

En términos generales, entre los esquemas simulados se han extraído las siguientes conclusiones:

- Comparado a los tradicionales esquemas de descarte *Drop-Front* y *Drop-Tail, Drop-Sel* basado en distinguir tres clases de tráfico, clase *real-time* de UDP no reactivo (flujos VoIP), clase otros UDP no *real-time* y clase TCP, proporciona mucha más equidad en los AQM RED, AVQ y REM cuando diferentes fuentes de tráfico reactivas y no-reactivas compiten por los mismos recursos de la red.
- *Drop-Sel* reduce el retardo extremo-a-extremo respecto a *Drop-Front* y *Drop-Tail,* porque a diferencia de éstos, *Drop-Sel* selecciona la clase apropiada de tráfico a ser descartada y reduce el retardo medio en la cola del *router* (por descartar de la cabecera de la cola en lugar de descartar de la parte de atrás de la cola), independientemente de las condiciones de carga. Por lo que, *Drop-Sel* minimiza las tasas de pérdidas en el receptor causadas por paquetes no-útiles (aquellos que acumulan un retardo extremo-a-extremo superior a 300 ms).
- El mecanismo ofrece potencial garantía de funcionamiento para la clase de tráfico VoIP. *Drop-Sel* reduce considerablemente la tasa de descarte en el *router* de los paquetes de audio, bajo condiciones de tráfico para las cuales las fuentes VoIP no dominen los recursos de la red.
- *Drop-Sel* no sólo mejora la calidad objetiva QoS de los flujos VoIP si no también la calidad subjetiva estimada medida en términos de factor MOS e inteligibilidad.

- Bajo condiciones de tráfico para las cuales las fuentes VoIP dominen, no es factible que el algoritmo propuesto consiga la equidad sin menoscabar la calidad de los paquetes de audio.
- Se ha comprobado la conveniencia de adoptar dinámicamente la estrategia de selección de víctima. En particular cuando el tráfico VoIP domine es preferible utilizar las estrategias tradicionales de selección de víctima como *Drop-Tail*.

Resumiendo, se ha corroborado experimentalmente que *Drop-Sel*, basado en la clasificación de tráfico, es una prometedora alternativa capaz de mejorar el grado de equidad entre clases de tráfico (en el caso de concurrir tráfico reactivo y no-reactivo) cuando compiten por los mismos recursos de la red, y cuando las condiciones de carga lo permiten, de incrementar la QoS para flujos de audio en términos de calidad de voz percibida extremo-a-extremo.

Capítulo 6

Alternativas de *Drop-Sel* y *Drop-Tail*

Se ha observado en el capítulo 5 que el algoritmo propuesto *Drop-Sel* mejora la equidad global cuando diferentes fuentes de tráfico reactivas y no-reactivas compiten por los mismos recursos de la red. Sin embargo, la evaluación de la calidad objetiva y subjetiva del tráfico VoIP (apartados 5.4.2 y 5.4.3 respectivamente) ha revelado que es imposible conseguir en todos los casos la equidad sin menoscabar la calidad.

Con el fin de mejorar aún más las prestaciones del servicio ofrecido a flujos multimedia y en particular de audio, además de la propuesta de referencia se presentan cuatro alternativas denominadas *Drop-Sel-delay*, *Drop-Tail-delay*, *Drop-Sel+fair* y *Drop-Tail+fair*. Las dos primeras mejoran aún más la calidad extremo-a-extremo de los paquetes de VoIP y las dos últimas, garantizan un servicio más equitativo a diferentes flujos de VoIP.

En el estudio se evalúa mediante simulación las mejoras introducidas por las variantes de los algoritmos *Drop-Sel* y *Drop-Tail* tanto en términos de QoS, como en términos de la calidad subjetiva experimentada (QoE) por el usuario final.

6.1. *Drop-Sel-delay* y *Drop-Tail-delay*

Los algoritmos de selección de víctima *Drop-Sel-delay* y *Drop-Tail-delay*, variantes que respectivamente mejoran a los esquemas *Drop-Sel* y *Drop-Tail*, están motivados por la limitación en el retardo extremo-a-extremo que demanda el tráfico VoIP interactivo. Como ya se ha comentado si un paquete perteneciente a un flujo de audio interactivo acumula un retardo extremo-a-extremo (típicamente, mayor que 300ms), éste es inaprovechable y produce un desperdicio de recursos de la red. Por lo tanto, un paquete va ser definido como útil siempre y cuando su edad -entendida como el tiempo transcurrido desde su generación en el origen- nunca exceda el máximo tiempo de vida permitido. Bajo tráfico pesado o condiciones de congestión, es posible que un paquete a lo largo de su camino, llegue a ser no-útil y, sin embargo, éste innecesariamente continúa consumiendo memoria y ancho de banda hasta alcanzar su destino. Aún peor, dado un *router*, si un criterio de selección de víctima no es tomado en cuenta, el algoritmo AQM puede descartar un paquete de voz útil mientras conserva en cola uno no-útil.

El algoritmo *Drop-Sel-delay*, conserva el mismo criterio de selección de paquetes que *Drop-Sel* de referencia cuando hay congestión, el cuál determina qué clase de tráfico contribuye más al problema de congestión. Por su parte, *Drop-Tail-delay* conserva el criterio de *Drop-Tail* de referencia, la selección directa del paquete recién llegado.

Adicionalmente, *Drop-Sel-delay* y *Drop-Tail-delay* consideran un criterio basado en la utilidad del paquete cuando la clase seleccionada como víctima corresponde a la clase *real-time* (VoIP). Específicamente, los algoritmos simplemente identifican paquetes no-útiles en la cola pertenecientes a la clase de tráfico a penalizar, descartándolos para así proteger a los paquetes útiles. Tan pronto como los paquetes de audio considerados no-aprovechables sean detectados y descartados anticipadamente, el número de pérdidas de paquetes útiles puede ser reducido. Y en consecuencia, mejor QoS puede ser esperada para el tráfico VoIP.

En esta investigación se asume que los *frames* de audio se encapsulan en un paquete RTP/UDP.

Drop-Sel-delay

Para conseguir un servicio equitativo y además evitar un desperdicio de recursos de la red, el algoritmo *Drop-Sel-delay* (Figura 6.1) ejecuta un procedimiento de dos niveles:

- En el primer nivel se llevan a cabo las funciones propias de *Drop-Sel* de

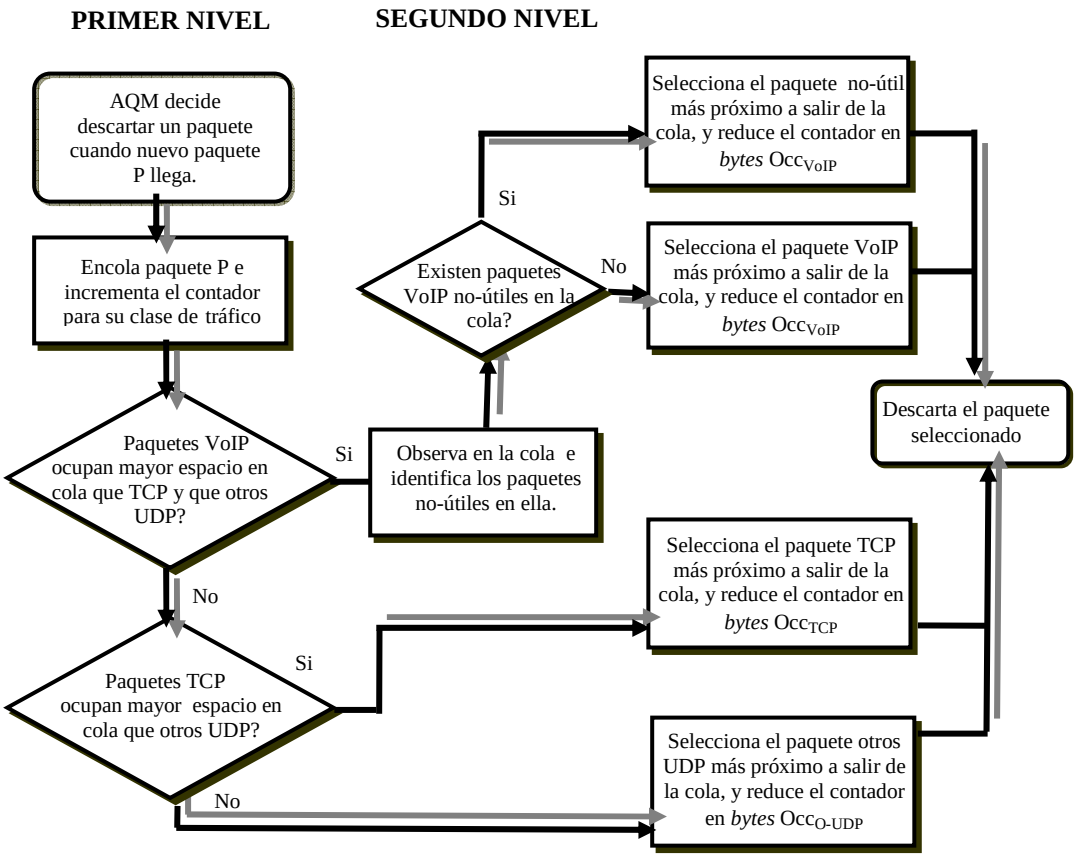


Figura 6.1. Algoritmo *Drop-Sel-delay.*

referencia, se determina cuál de las tres clases de tráfico ocupa mayor espacio en la cola. Si la clase otros UDP o la clase TCP consumen mayor espacio de memoria durante los períodos de congestión, el algoritmo *Drop-Sel-delay* busca el paquete correspondiente a la clase seleccionada que esté más próximo a salir de la cola, y lo descarta. A diferencia de *Drop-Sel*, si la clase *real-time* es la que ocupa mayor espacio en la cola, un segundo nivel es ejecutado.

- *Drop-Sel-delay* en su segundo nivel de operación, observa los paquetes de la cola e identifica el paquete de audio considerado no-útil que esté más próximo a salir de la cola. En caso de no localizar ningún paquete que cumpla con las especificaciones solicitadas (>300ms), simplemente descarta el paquete de audio más próximo a salir de la cola. Es decir, actúa como *Drop-Sel* de referencia.

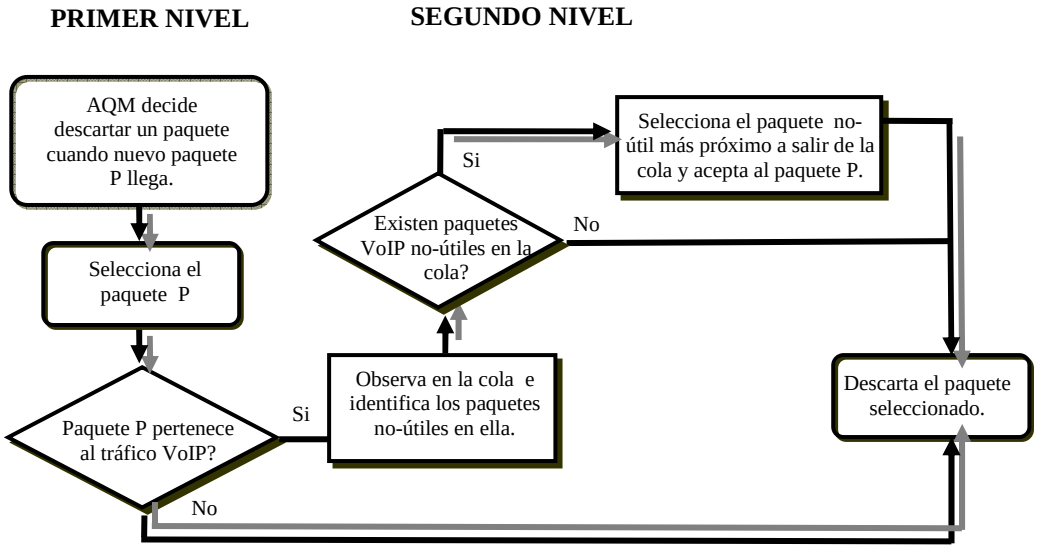


Figura 6.2. Algoritmo *Drop-Tail-delay.*

Finalmente, cada vez que un paquete abandona la cola, el algoritmo *Drop-Sel-delay* actualiza los contadores convenientemente.

Drop-Tail-delay

Similarmente, el algoritmo *Drop-Tail-delay* (Figura 6.2) ejecuta un procedimiento de dos niveles:

- En el primer nivel se llevan a cabo las funciones propias de *Drop-Tail* de referencia, la selección directa del paquete recién llegado. Si el paquete recién llegado pertenece a otros flujos UDP o TCP, lo descarta. A diferencia de *Drop-Tail* de referencia, si el paquete recién llegado pertenece a la clase de tráfico *real-time*, un segundo nivel es ejecutado.
- *Drop-Tail-delay* en su segundo nivel de operación, observa los paquetes de la cola e identifica el paquete de audio considerado no-útil que esté más próximo a salir de la cola. En caso de no localizar ningún paquete que cumpla con las especificaciones solicitadas (>300ms), simplemente descarta el paquete de audio recién llegado a la cola. Es decir, actúa como *Drop-Tail* de referencia.

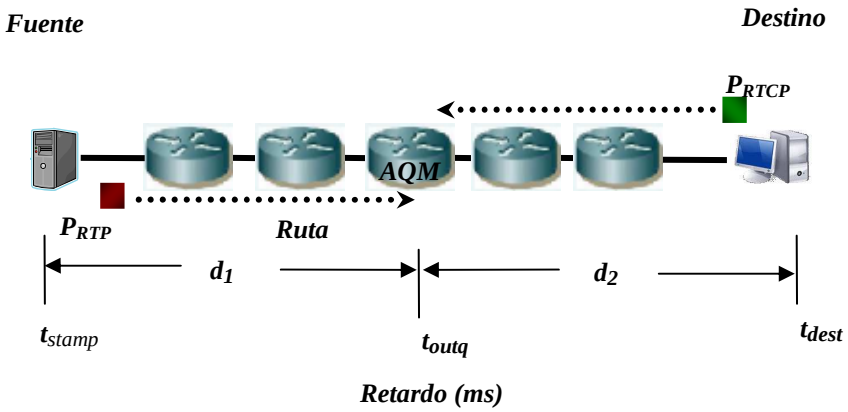


Figura 6.3. Esquema de retardo en una dirección.

Retardo en una dirección

En *Drop-Sel-delay* y *Drop-Tail-delay* se ha incorporado un criterio para seleccionar el paquete VoIP, basándose en el retardo extremo-a-extremo acumulado. Para estimar este valor, inherente a los algoritmos, se proponen esquemas que determinan la edad o retardo acumulado del paquete, utilizando para ello los mensaje del protocolo RTCP [Schulzrinne *et al.*, 2003]. Para ello se supondrá que las entidades involucradas: fuente, *router* AQM y destino están sincronizadas.

Como se observa en la Figura 6.3, el retardo extremo-a-extremo del paquete en una dirección puede ser modelado como la suma de dos variables, el retardo desde su nodo origen hasta el *router* AQM (denominada d_1) y el retardo desde el *router* AQM hasta su nodo destino (denominada d_2). El paquete llega al *router* AQM y al nodo destino con cierto retardo de transmisión y encolamiento derivados de la manipulación a través de los enlaces. La idea principal detrás de *Drop-Sel-delay* y *Drop-Tail-delay* es estimar el retardo que tendrán acumulado cada uno de los paquetes de VoIP encolados cuando éstos lleguen a su destino. Y con ello, predecir posibles paquetes que no serán útiles.

Para que *Drop-Sel-delay* y *Drop-Tail-delay* sean efectivos, a diferencia respectivamente de *Drop-Sel y Drop-Tail*, estos capturan información específica para calcular el retardo acumulado a lo largo de un camino de la red.

Retardo d_2

Por cada paquete de VoIP que sale de la cola AQM, *Drop-Sel-delay* y *Drop-Tail-delay*

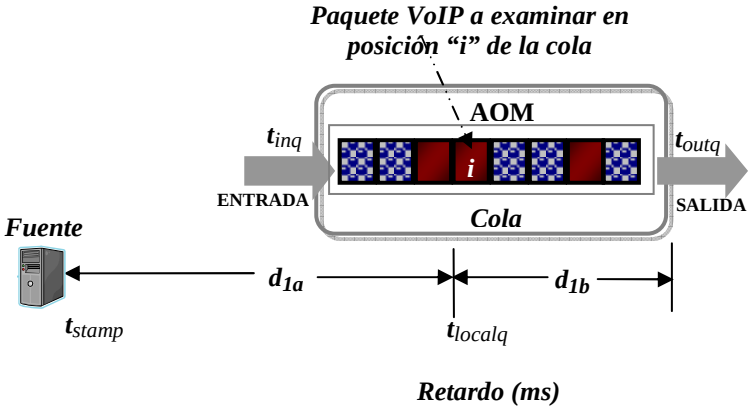


Figura 6.4. Esquema de retardo en cola.

insertan en la cabecera RTP (Figura 2.3), el tiempo de partida de éste. Este valor permitirá estimar una de las variables del retardo, específicamente d_2 (Figura 6.3). Una vez que el nodo destino recibe un paquete RTP, éste examina los campos de la cabecera del paquete y calcula el retardo acumulado entre el *router* AQM y él, determinando simplemente la diferencia entre tiempos:

$$d_2 = t_{dest} - t_{outq} \qquad [6.1]$$

Donde, t_{dest} indica el tiempo cuando el paquete es recibido en el destino y t_{outq} el tiempo de partida de la cola AQM. La medida obtenida (d_2), es entonces retroalimentada a la fuente a través de los mensajes "*receiver report*" de RTCP.

Cuando la fuente recibe el mensaje de control RTCP, ésta extrae la medida del retardo y la incluye en cada uno de los siguientes paquetes RTP a ser transmitidos. Tan pronto como la fuente reciba un nuevo informe RTCP, ésta obtiene la nueva medida de retardo d_2. Con frecuentes mensajes de control llegando a la fuente, se obtendrá información actualizada de los paquetes y con ello, un apropiado retardo adaptado a las fluctuaciones de carga de la red.

Retardo d_1

Por otro lado, para calcular eficazmente el retardo d_1, en particular se propone dividir esta variable en dos componentes: primeramente, estimar el retardo del paquete desde el origen hasta su posición en la cola AQM (d_{1a}) y en segundo término, obtener el retardo que aún necesita emplear en la cola antes de abandonarla (d_{1b}) (Figura 6.4). Para este

propósito, *Drop-Sel-delay* y *Drop-Tail-delay* utilizan información propia del paquete y del mismo *router*. Específicamente, la marca de tiempo del encolado paquete (t_{stamp}), el tiempo local (t_{localq}) en el *router* cuando se hace la estimación, el retardo de encolamiento estimado (d_{queue}), la cantidad de paquetes en la cola (s_q) y la posición del paquete en la cola (i).

Por lo que, por cada paquete que sale de la cola, el esquema obtiene el retardo de encolamiento utilizando

$$d_{queue} = t_{outq} - t_{inq} \qquad [6.2]$$

Donde, t_{inq} es el tiempo de llegada del paquete a la cola AQM y t_{outq} el tiempo de salida de la cola.

Monitorizando el valor d_{queue} se dispondrá de información para caracterizar la dinámica de la red. Con ello, los algoritmos de selección de víctima *Drop-Sel-delay* y *Drop-Tail-delay*, estiman el retardo acumulado para el paquete encolado en la posición *i* utilizando las siguientes expresiones,

$$d_{1a} = t_{localq} - t_{stamp} \qquad [6.3]$$

$$d_{1b} = (\frac{d_{queue}}{s_q}) * (i-1) \qquad [6.4]$$

En definitiva, bajo condiciones de congestión, si el paquete acumula un retardo extremo-a-extremo superior a 300ms ($d_{1a} + d_{1b} + d_2 \geq 300$, *expresiones [6.1, 6.3 y 6.4]*), éste será considerado no-útil.

En la Figura 6.5 se ilustra el procedimiento de selección de víctima *Drop-Sel-delay* con un ejemplo. Se asume que el tráfico de VoIP es el que ocupa mayor espacio en la cola cuando el AQM decide descartar un paquete. En este sentido, el esquema de selección de víctima considera el retardo acumulado de cada uno de los paquetes de audio encolados. Como resultado, determina que dos de los paquetes encolados tendrán un retardo superior al umbral establecido (300 ms). Por ejemplo, supóngase que un paquete acumula 317 ms y otro 308 ms. En este caso, *Drop-Sel-delay* descarta el paquete con 308 ms (el más próximo a salir de la cola) en lugar del paquete que ha acumulado mayor retardo (317 ms). Si la víctima seleccionada fuese el otro paquete, el paquete de 308 ms puede ser eventualmente enviado (antes de poder descartarlo) y esto puede desperdiciar recursos innecesariamente.

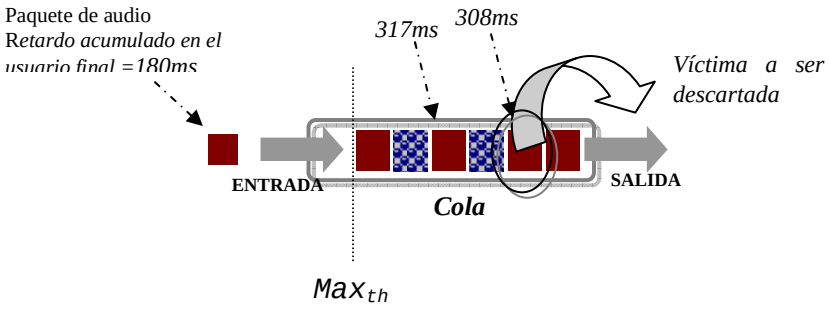


Figura 6.5. Ejemplo de *Drop-Sel-delay*.

En este camino *Drop-Sel-delay* intenta evitar la transmisión de paquetes que no puedan ser aprovechados en el usuario.

6.2. *Drop-Sel+fair* y *Drop-Tail+fair*

Para proporcionar un servicio que garantice equidad a diversos flujos de tráfico VoIP, que utilizan diferentes *codecs* o diferentes tasas de envío, en esta sección, se proponen otras alternativas de selección de víctima denominadas *Drop-Sel+fair* y *Drop-Tail+fair* .

Drop-Sel+fair y *Drop-Tail+fair* son fundamentalmente una mejora a los esquemas *Drop-Sel* y *Drop-Tail* respectivamente. Estos esquemas clasifican los paquetes que van llegando a la cola y aplican diferentes políticas a cada clase de tráfico para mantener igualdad entre diferentes flujos de VoIP con diferente tasa de envío. *Drop-Sel+fair*, al igual que *Drop-Sel-delay*, conserva el mismo criterio de selección de paquetes que *Drop-Sel* de referencia cuando hay congestión, el cual determina que clase de tráfico contribuye más al problema de la congestión. Por su parte, *Drop-Tail+fair* conserva el criterio de *Drop-Tail* de referencia, la selección directa del paquete recién llegado.

Adicionalmente, los métodos *Drop-Sel+fair* y *Drop-Tail+fair* consideran un criterio por flujo que define como víctima el flujo con la tasa de pérdidas de paquetes más baja cuando la clase seleccionada como víctima corresponde a la clase *real-time* (VoIP). De tal manera, cuando un paquete llega al *router*, éste es clasificado. Además, es contabilizado según su clase de tráfico si *Drop-Sel+fair* es adoptado. Si el paquete corresponde a un flujo de VoIP, la información del propio flujo es adaptada correspondientemente. Para este fin, el algoritmo mantiene una tabla con información acerca de cada uno de los flujos activos. Esta información por flujo consiste de un

identificador del flujo, el número de paquetes de voz descartados, el período de envío entre paquetes del flujo y un factor de descarte.

El identificador del flujo puede ser obtenido de la cabecera RTP o, alternativamente de la dirección IP y puertos de la fuente y destino. Para cada identificador de flujo, el número de paquetes de voz descartados son contados y almacenados en la variable $D_{vf.}$

Independientemente a esto, el número de flujos activos es también actualizado. El período de envío entre paquetes puede ser obtenido de la cabecera RTP, o inspeccionando la tasa de llegada del paquete.

Este método hace posible estimar la degradación relativa entre varios flujos VoIP, utilizando solamente la información a nivel de red, particularmente la tasa de paquetes descartados y el número de paquetes perdidos. Por lo que, un factor de descarte que determina el peso de los paquetes perdidos para un flujo dado es también almacenado en la tabla *Drop-Sel+fair* o *Drop-Tail+fair*.

Más precisamente, *Drop-Sel+fair* y *Drop-Tail+fair* estiman el factor de descarte (FD_f) por flujo expresado como

$$FD_f = (\frac{IP_f}{IP_{fl}}) * D_{vf} * R_{ps} \qquad [6.5]$$

en donde IP_{fl} corresponde al período de envío más bajo de todos los flujos activos, IP_f el período de envío del flujo en particular y R_{ps} la relación de tamaño del paquete entre los flujos, la cual es calculada como

$$R_{ps} = \frac{((\frac{IP_{fh}}{IP_{pl}}) * SP_l)}{((\frac{IP_{fh}}{IP_f}) * SP_f)} \qquad [6.6]$$

donde IP_{fh} corresponde al período de envío más alto de todos los flujos activos, IP_{pl} el período de envío del flujo con tamaño de paquete más pequeño, SP_l el tamaño de paquete más pequeño y finalmente, SP_f el tamaño del paquete del flujo en particular.

Simplificando la expresión 6.6,

$$R_{ps} = \frac{(IP_f * SP_l)}{(IP_{pl} * SP_f)} \qquad [6.7]$$

Por lo que, la estimación del factor de descarte por flujo se puede expresar como

$$FD_f = (\frac{IP_f}{IP_{fl}}) * D_{vf} * \frac{(IP_f * SP_l)}{(IP_{pl} * SP_f)} \qquad [6.8]$$

En otras palabras, cuando hay presentes diferentes flujos con diferentes tamaños de paquetes y diferentes periodos de generación, se trata de igualar en el tiempo la cantidad de información descartada por flujo. Para ello, se ha definido el factor de descarte, que permite en un instante dado seleccionar el flujo que menos penalización haya sufrido hasta ese momento.

Drop-Sel+fair

En concreto, para proporcionar un servicio todavía más equitativo, el algoritmo *Drop-Sel+fair* ejecuta un procedimiento de dos niveles de la misma manera que *Drop-Sel-delay*:

- En el primer nivel se llevan a cabo las funciones propias de *Drop-Sel* de referencia, se determina cuál de las tres clases de tráfico ocupa mayor espacio en la cola. Si la clase otros UDP o la clase TCP consumen mayor espacio de memoria durante los períodos de congestión, el algoritmo *Drop-Sel+fair* busca el paquete correspondiente a la clase seleccionada que esté más próximo a salir de la cola, y lo descarta. A diferencia de *Drop-Sel*, si la clase *real-time* es la que ocupa mayor espacio en la cola, un segundo nivel es ejecutado.
- *Drop-Sel+fair* en su segundo nivel de operación, observa la tabla de información de flujos activos, específicamente el factor de descarte FD_f, para identificar y seleccionar el flujo menos degradado. Entonces, el algoritmo busca el paquete correspondiente al flujo seleccionado que esté más próximo a salir de la cola, y lo descarta.

Cada vez que un paquete abandona la cola, el algoritmo actualiza la tabla de flujos activos y los contadores convenientemente. En la Figura 6.6, se representa el algoritmo *Drop-Sel+fair* mediante un sencillo diagrama de flujo.

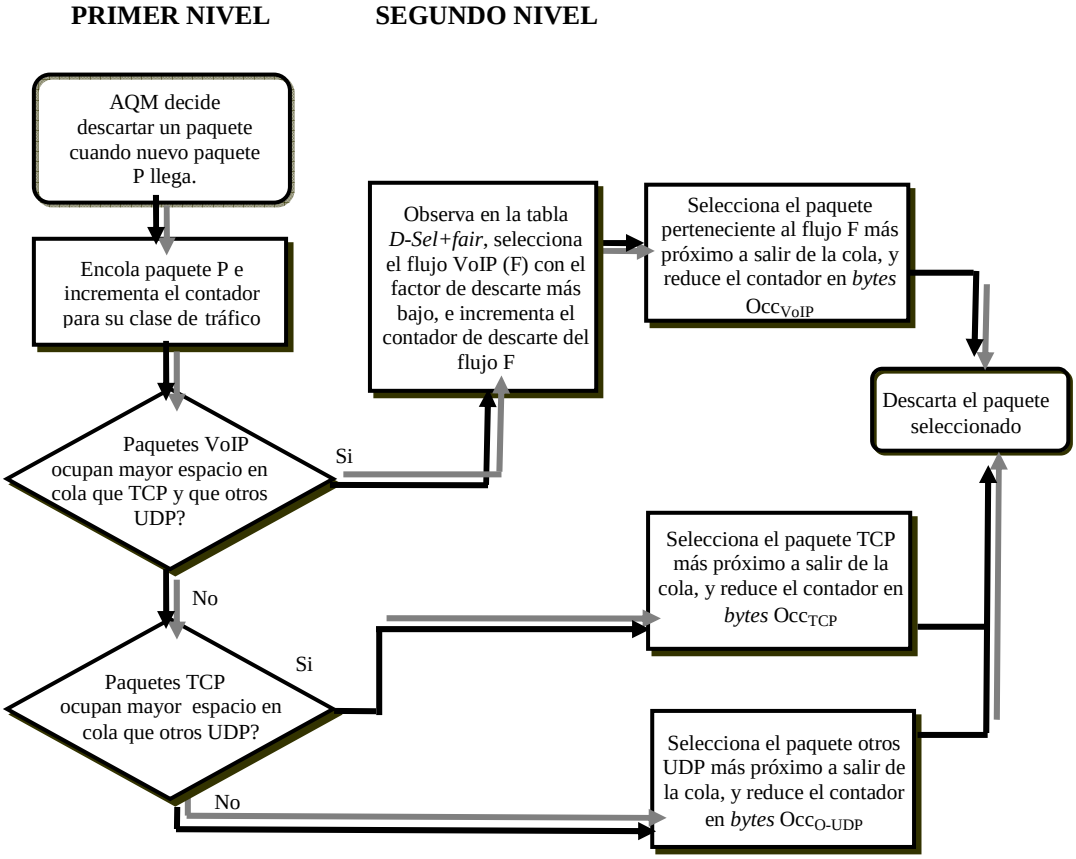


Figura 6.6. Algoritmo *Drop-Sel+fair*.

Drop-Tail+fair

Similarmente, el algoritmo *Drop-Tail+fair* (Figura 6.7) ejecuta un procedimiento de dos niveles:

- En el primer nivel se llevan a cabo las funciones propias de *Drop-Tail* de referencia, la selección directa del paquete recién llegado. Si el paquete recién llegado pertenece a otros flujos UDP o TCP, lo descarta. A diferencia de *Drop-Tail* de referencia, si el paquete recién llegado pertenece a la clase de tráfico *real-time*, un segundo nivel es ejecutado.
- *Drop-Tail+fair* en su segundo nivel de operación, observa la tabla de

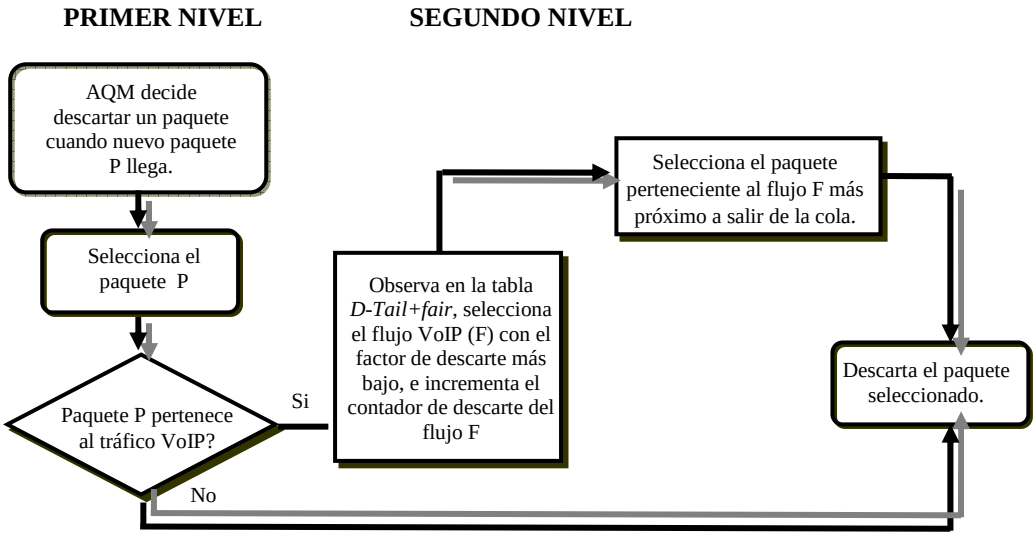


Figura 6.7. Algoritmo *Drop-Tail+fair*.

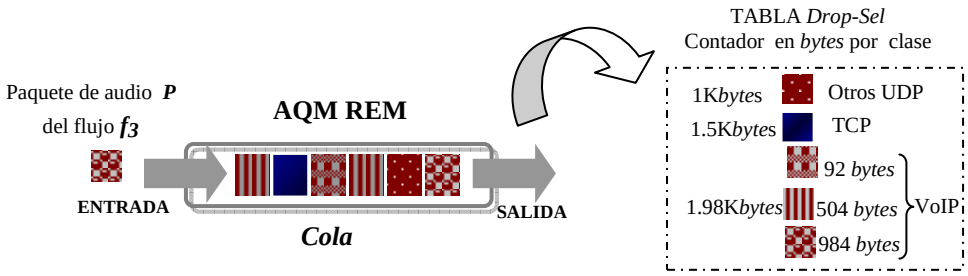


Figura 6.8. Ejemplo de Drop-*Sel+fair* en su primer nivel.

información de flujos activos, específicamente el factor de descarte FD_f, para identificar y seleccionar el flujo menos degradado. Entonces, el algoritmo busca el paquete correspondiente al flujo seleccionado que esté más próximo a salir de la cola, y lo descarta. Cada vez que un paquete abandona la cola, el algoritmo actualiza la tabla de flujos activos.

En las Figuras 6.8 y 6.9 se ilustra el procedimiento de selección de víctima *Drop-Sel-+fair* con un ejemplo. Como se observa en la Figura 6.8, dada una situación de

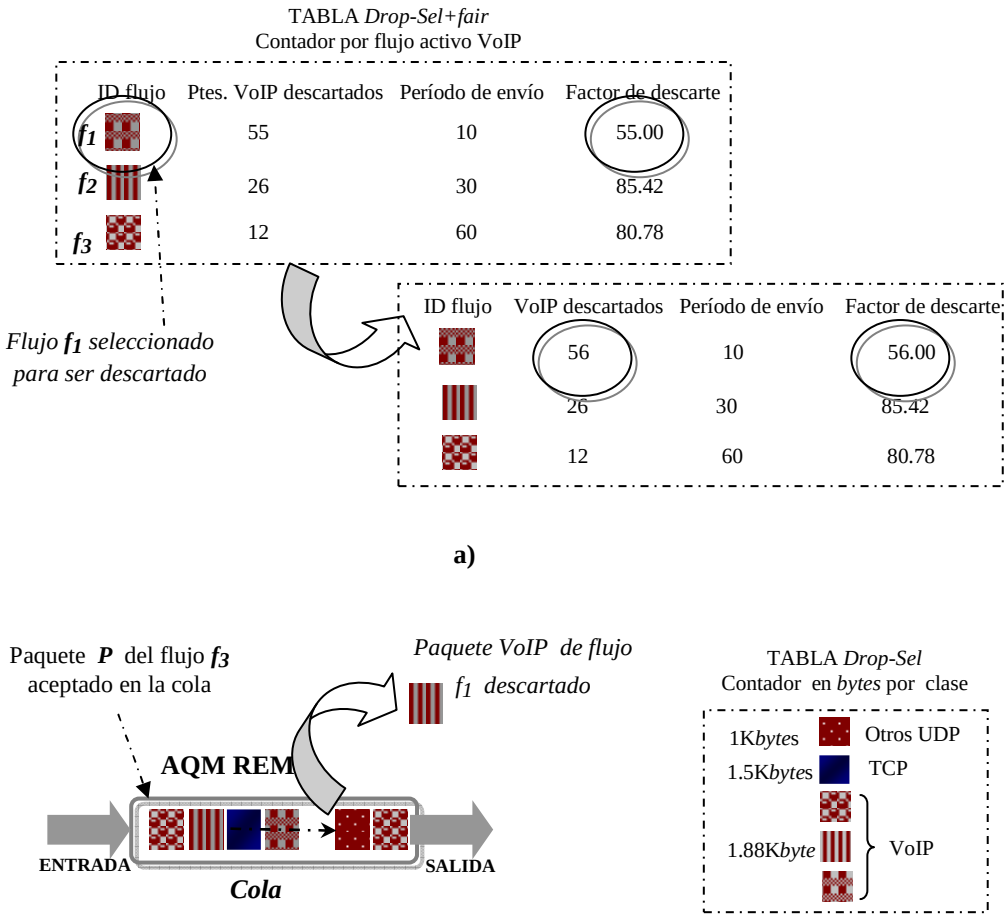


Figura 6.9. Ejemplo de *Drop-Sel+fair* en su segundo nivel.

congestión, el AQM REM decide descartar un paquete cuando el paquete de audio P llega. De forma tal que el algoritmo *Drop-Sel+fair* clasifica e incrementa su respectivo contador de clase. Los contadores por clase de tráfico registran hasta ese momento 1500 *bytes* TCP (1 paquete), 1000 *bytes* otros UDP (1 paquete) y 1980 *bytes* VoIP (por 1 paquete de 92 *bytes*, 2 de 252 *bytes* y 2 de 492 *bytes)* de ocupación en la cola. Por lo que, al compararlos, identifica al tráfico VoIP como la clase que ocupa mayor espacio en la cola. En consecuencia, *Drop-Sel+fair* tiene que ejecutar el segundo nivel.

En el segundo nivel (Figura 6.9), el algoritmo observa la tabla de flujos activos e

identifica al flujo de VoIP denominado f_1, como el flujo con el factor de descarte más bajo. Con ello, busca y selecciona el paquete correspondiente al flujo f_1 más próximo a salir de la cola. Incrementa el contador de descarte de ese flujo y actualiza su factor de descarte (Figura 6.9a). Finalmente, como se muestra en la Figura 6.9b, descarta el paquete seleccionado y reduce su respectivo contador de clase de tráfico.

6.3. Evaluación e impacto de los modelo propuestos

Para medir el impacto a nivel de red y a nivel de usuario de las distintas alternativas de los esquemas *Drop-Sel* y *Drop-Tail*, un número de simulaciones fueron realizadas con *Network Simulator* (*ns-2*). Más precisamente, nuevamente los esquemas AQM seleccionados (RED, REM y AVQ), y adoptando los algoritmos de selección de víctima, fueron evaluados bajo una variedad de topologías de red, fuentes de tráfico y diferentes niveles de congestión descritos en el apartado 5.3.2.

6.3.1. *Drop-Sel-delay*

En los apartados 5.4.2 y 5.4.3 se observó que *Drop-Sel* deteriora significativamente la calidad objetiva y subjetiva de los flujos VoIP cuando esta clase de tráfico consume el mayor espacio de memoria en cola durante los períodos de congestión. En este caso, esto sugiere que la aplicación de *Drop-Sel* preferiblemente debe permutar a otra de las estrategias tradicionales de selección de víctima.

No obstante, si el objetivo es proporcionar equidad entre las distintas clases de tráfico, se deben establecer nuevas alternativas para aminorar el deterioro en la calidad del servicio.

Aunque el esquema *Drop-Sel* de referencia considere descartar por clase de tráfico, éste no proporciona un tratamiento por flujo. Para mejorar el servicio otorgado por *Drop-Sel* para flujos VoIP (preferentemente cuando las fuentes VoIP dominen), la variante *Drop-Sel-delay* considera un criterio adicional basado en la utilidad del paquete y evita con ello descartar un paquete de voz útil mientras conserva en la cola uno no-útil.

En este sentido se evalúa el funcionamiento de *Drop-Sel-delay* considerando particularmente los casos 5 y 6 del escenario S2 (Tabla 5.3) donde intervienen conexiones con largos retardos de propagación y las fuentes VoIP dominan.

En los casos 5 y 6, un número de fuentes TCP y otros flujos UDP no-reactivos

compiten con 40 flujos VoIP (UDP). Específicamente, 10 son generados con la aplicación tipo A (denominados como A1 a A10), 10 con la aplicación tipo B (denominados como B1 a B10), 10 con la aplicación tipo C (denominados como C1 a C10) y finalmente, 10 de los flujos VoIP con la aplicación tipo D (denominados como D1 a D10).

Equidad entre clases de tráfico

Drop-Sel-delay conserva el mismo criterio de selección de paquetes que *Drop-Sel de* referencia cuando hay congestión. En relación a esto, se ha examinado el efecto sobre el rendimiento medio para las distintas clases de tráfico. En las Figuras 6.10 y 6.11 se han establecido el rendimiento medio de entrada y de salida de *Drop-Sel-delay* en los distintos esquemas AQM y para propósitos de comparación los resultados con *Drop-Tail* y *Drop-Sel* son también representados.

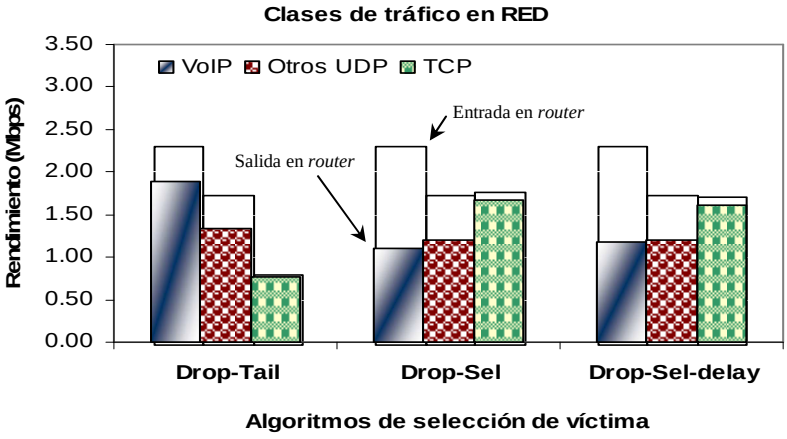
Primeramente, para el caso 5 en RED (Figura 6.10a), se observa que *Drop-Sel* comparado con *Drop-Tail* es equitativo. *Drop-Sel-delay* sigue manteniendo este comportamiento equitativo otorgando un 40% a TCP e igualando a 30% el porcentaje de tráfico cursado de VoIP y otros flujos UDP. Más aún, este comportamiento también se observa para AVQ (Figura 6.10b) y REM (Figura 6.10c) donde *Drop-Sel-delay* otorga equitativamente los recursos a las clases de tráficos.

En el caso 6 nuevamente tiende a ser igual de imparcial que *Drop-Sel* para los tres tráficos independientemente de la naturaleza reactiva de las fuentes contendientes. En RED (Figura 6.11a) y AVQ (Figura 6.11b) logra ser más justo y se observa que las fuentes UDP (no VoIP) obtienen aproximadamente todo el ancho de banda necesario (21%). Por ejemplo, *Drop-Sel* respectivamente proporciona el 32% y el 47% del ancho de banda a los flujos VoIP y TCP en AQM RED. En cambio *Drop-Sel-delay* respectivamente proporciona el 35% y el 44%.

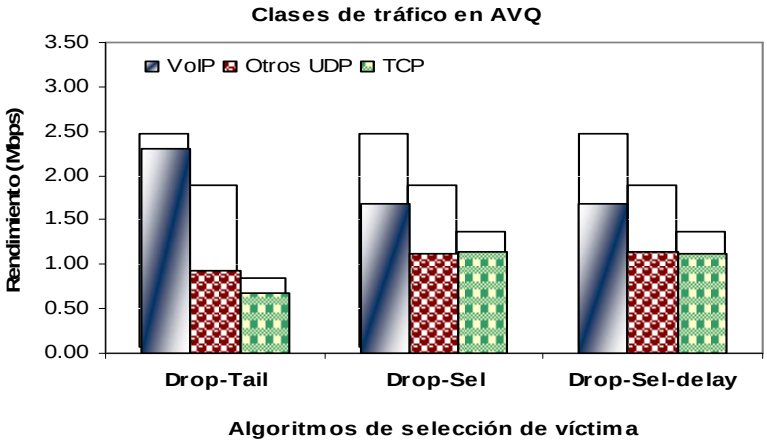
Tasa de pérdidas

Dado que en el destino la calidad final del paquete de audio depende de factores como la probabilidad de pérdidas, se evalúa el impacto de *Drop-Sel-delay* en este respecto.

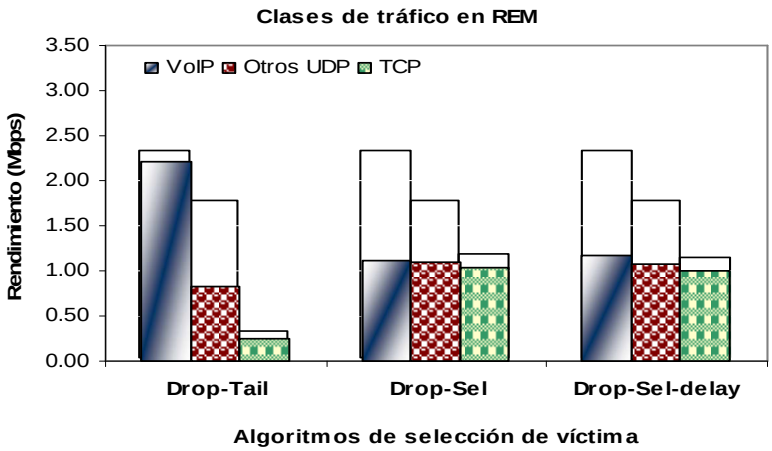
Primeramente, para el caso 5, en las Figuras 6.12, 6.13 y 6.14 se muestran los resultados extraídos para la variante *Drop-Sel-delay* aplicada a los esquemas AQM, y para propósitos de comparación los resultados con *Drop-Sel* son también representados.



a)



b)



c)

Figura 6.10. Evaluación del rendimiento medio en *Drop-Tail*, *Drop-Sel* y *Drop-Sel-delay* en el caso 5.

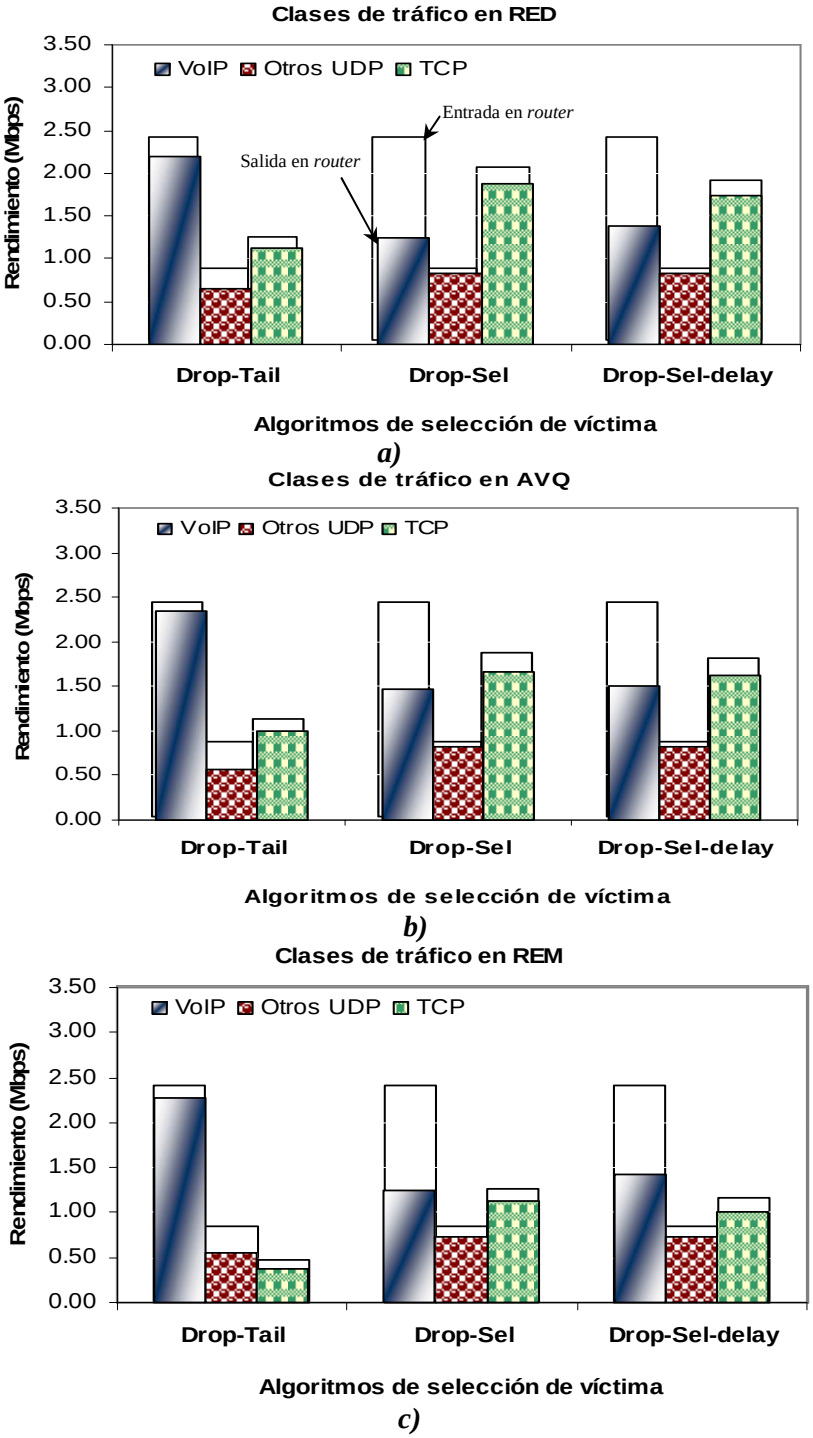


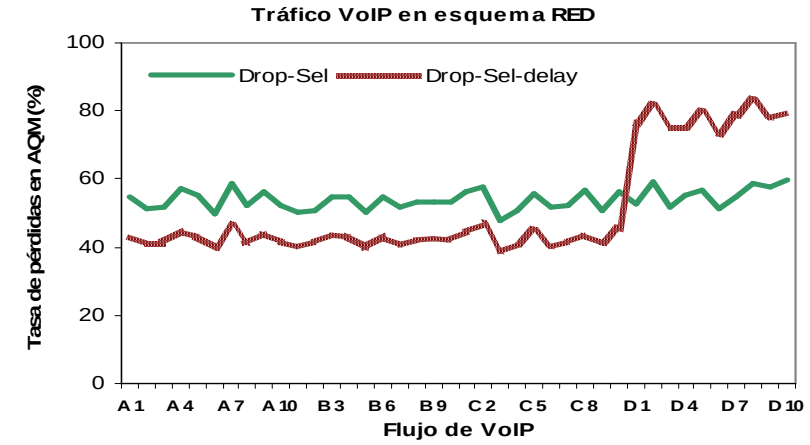
Figura 6.11. Evaluación del rendimiento medio en *Drop-Tail*, *Drop-Sel* y *Drop-Sel-delay* en el caso 6.

En la Figura 6.12b puede ser observado que al no considerar la vida útil del paquete *Drop-Sel* en RED permite que lleguen altas tasas de paquetes no-útiles, típicamente de los flujos más afectados por el retardo extremo-a-extremo. Específicamente, las fuentes más distantes D1 a D10 (en Tabla 5.1) son las que obtienen los peores resultados (desde 25.75% hasta 27.96%). Por el contrario, *Drop-Sel-delay* detecta que los flujos D1 a D10 están generando paquetes no-útiles y aumenta su probabilidad de descarte en el AQM (Figura 6.12a). Tan pronto como los paquetes de audio considerados no-aprovechables sean detectados y descartados anticipadamente, el número de pérdidas de paquetes útiles puede ser reducido. De hecho, la tasa general de paquetes no-útiles desciende a 11.72% (Figura 6.12b).

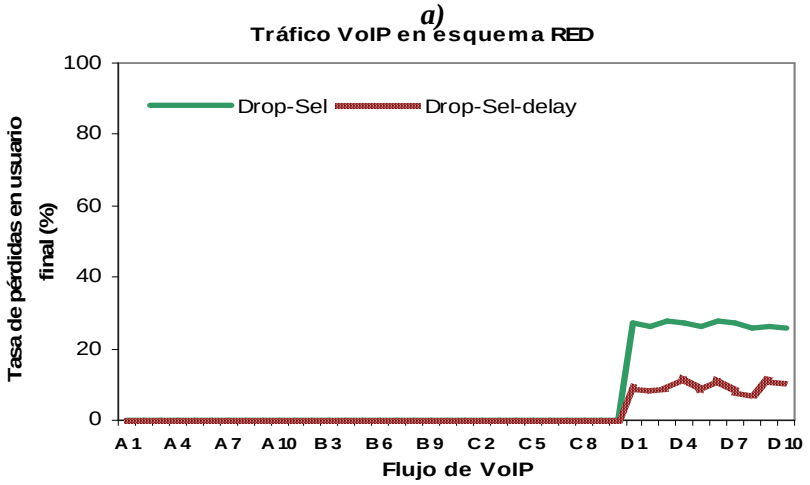
Con el fin de evitar un desperdicio de recursos, el esquema *Drop-Sel-delay* tiene como prioridad descartar los paquetes de audio para los que se prevé acumulan un retardo extremo-a-extremo superior a 300 ms en el usuario final ($d_{1a} + d_{1b} + d_2 \geq 300$, *expresiones [6.1, 6.2 y 6.3]*). Como los dos primeros términos de la predicción ($d_{1b} + d_2$) se basan en el historial de los paquetes que previamente han atravesado por el *router*, se podría llegar a penalizar paquetes que pudieran ser útiles al llegar al receptor. No obstante, estas consecuencias no deterioran más la calidad global del tráfico VoIP. Por ejemplo, en la Figura 5.12c se observa que el deterioro en la tasa de pérdidas total de los flujos D1 a D10 (un incremento adicional desde 4.34% a 5.45%) es menor que el beneficio otorgado a los otros flujos (30 flujos de 40 en total) para los cuales se incrementa la tasa de paquetes útiles desde 8.98% a 12.87%.

La Figura 6.13 corresponde al caso 5 en el cual AVQ adopta *Drop-Sel-delay*. Aunque AVQ experimenta más bajas tasas de descarte en la cola y en el usuario final con respecto a RED (Figuras 6.13a y 6.13b), *Drop-Sel-delay* también detecta en este esquema que los flujos D1 a D10 están generando paquetes no-útiles y aumenta su probabilidad de descarte en el AQM (Figura 6.13a). En este caso, se observa en la Figura 6.13c una reducción de la tasa total de pérdidas del tráfico VoIP del 0.84%. Un beneficio más apreciable es observado para REM en la Figura 6.14c, en la que se reduce un 7.96% la tasa total de pérdidas de paquetes de voz en el receptor. Y una mejora relativa del 18.57% en la tasa de paquetes útiles.

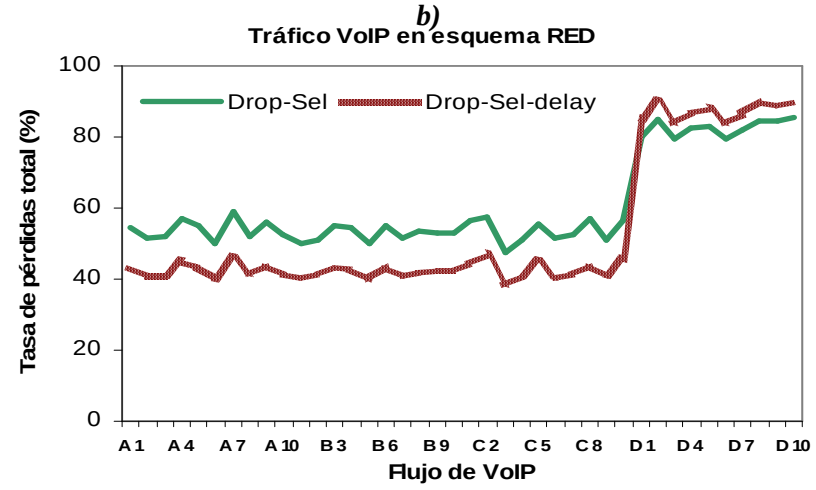
Similarmente, para el caso 6 (Figuras 6.15, 6.16 y 6.17) en el cual TCP y otros UDP flujos compiten con flujos VoIP (dominantes y causantes de la congestión), se observa que *Drop-Sel-delay* consigue disminuir las pérdidas totales de paquetes de VoIP por conservar en la cola los paquetes evaluados como útiles y descartar preferentemente posibles paquetes no-útiles.



a)

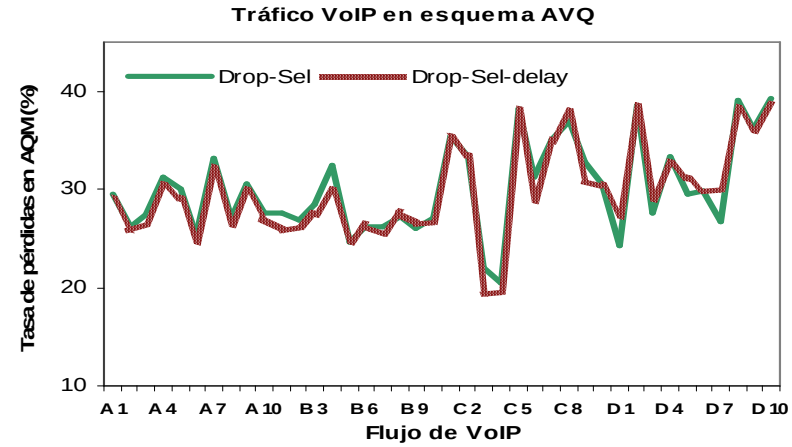


b)

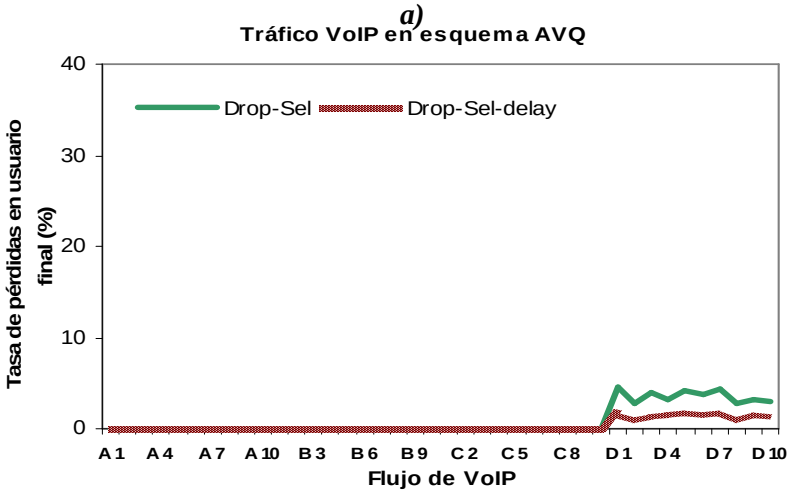


c)

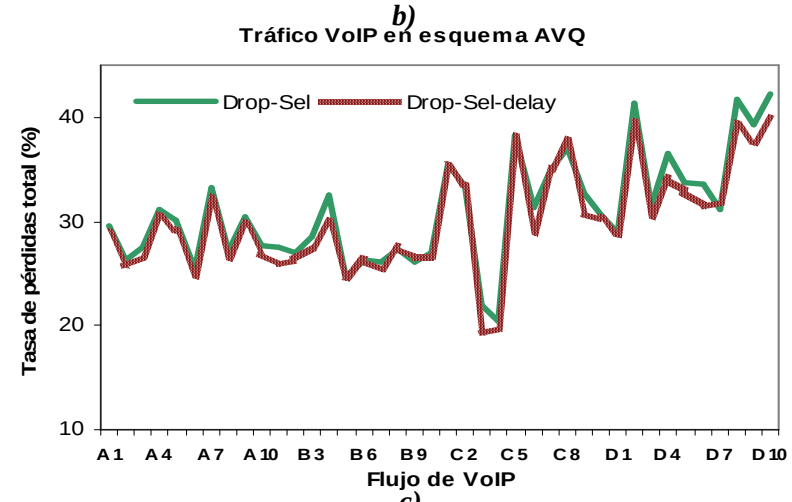
Figura 6.12. Evaluación por flujo bajo *Drop-Sel* y *Drop-Sel-delay* en el caso 5 con RED.



a)

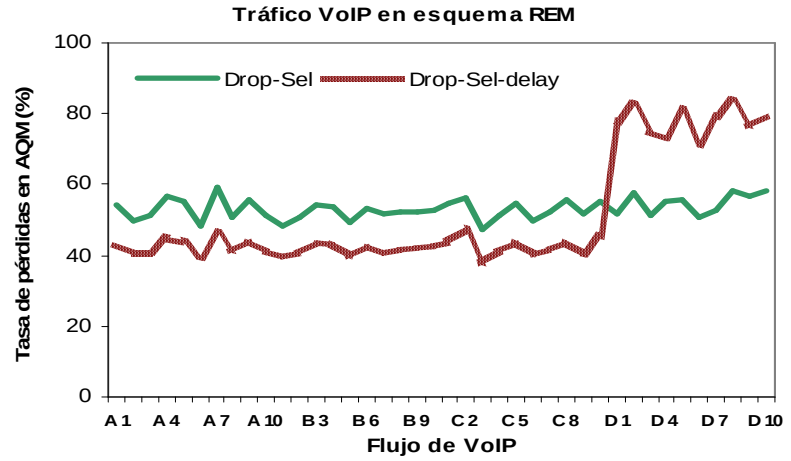


b)

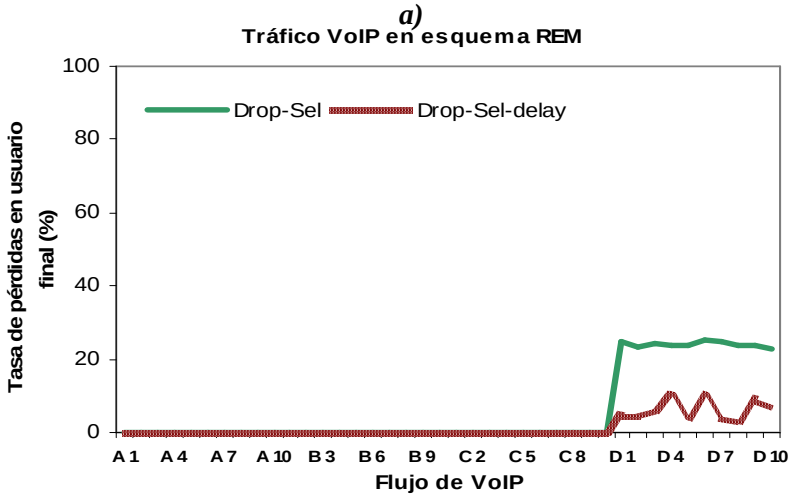


c)

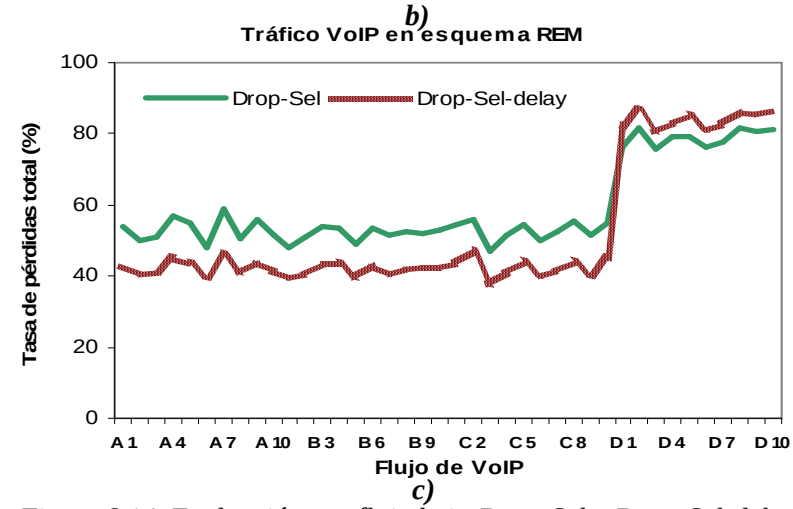
Figura 6.13. Evaluación por flujo bajo *Drop-Sel* y *Drop-Sel-delay* en el caso 5 con AVQ.



a)

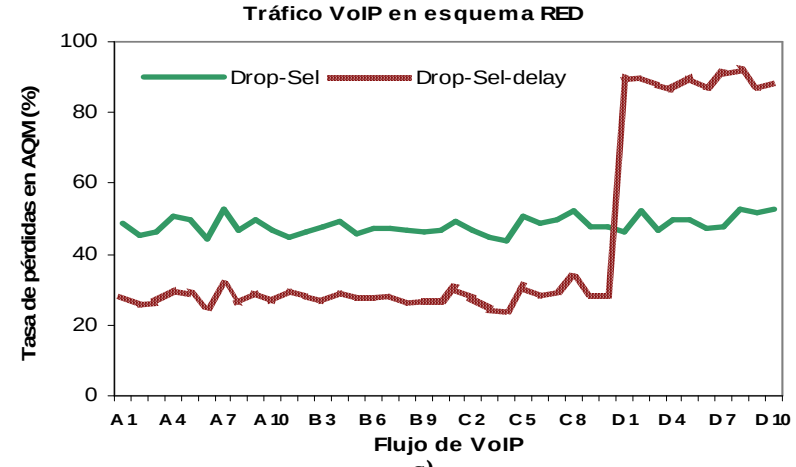


b)

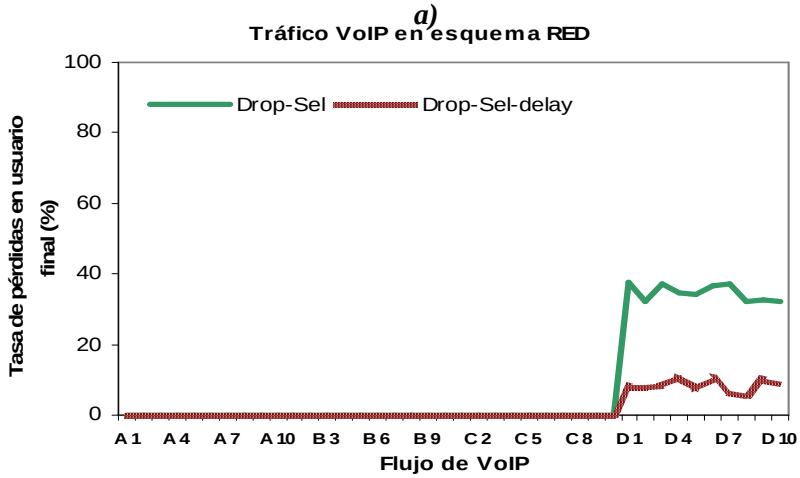


c)

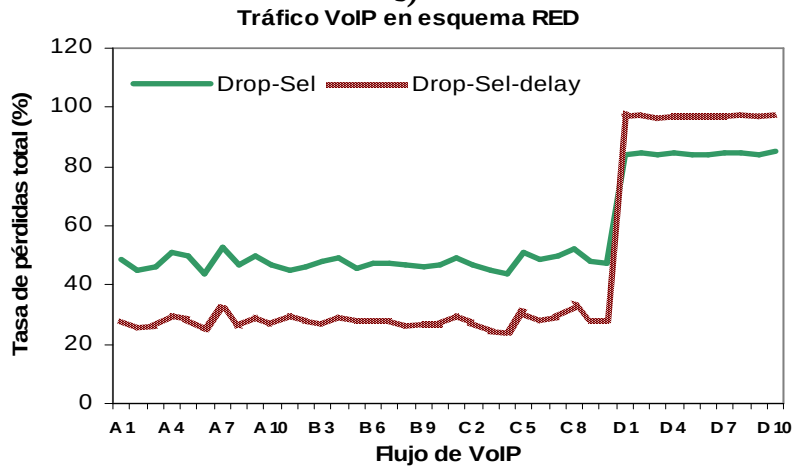
Figura 6.14. Evaluación por flujo bajo *Drop-Sel* y *Drop-Sel-delay* en el caso 5 con REM.



a)

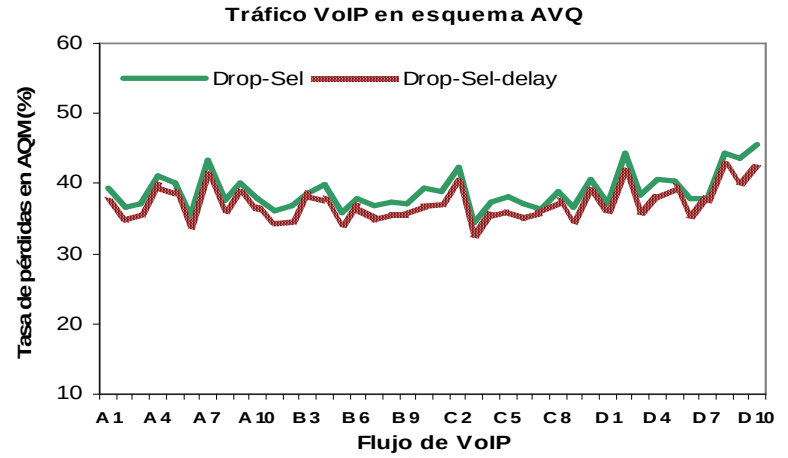


b)

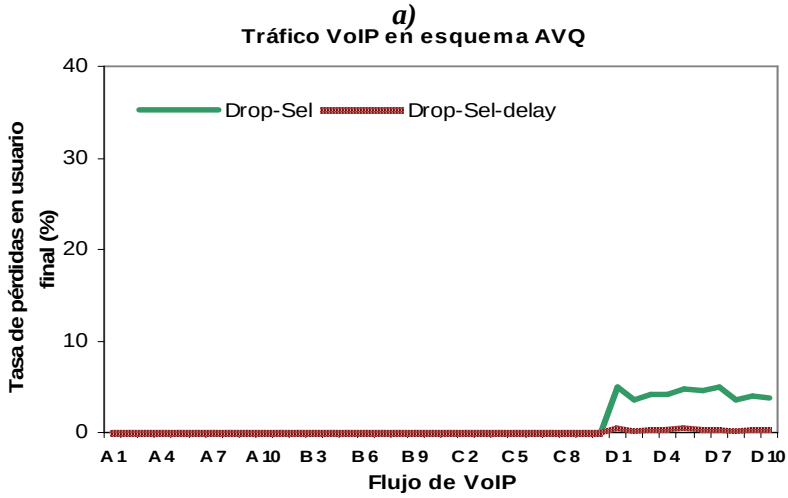


c)

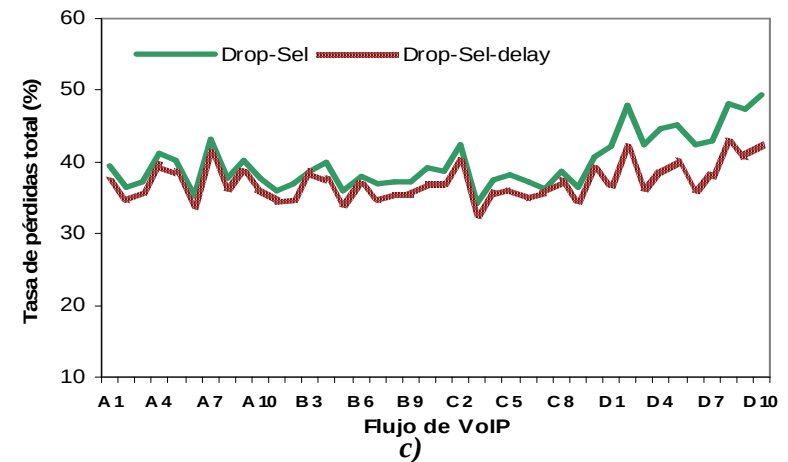
Figura 6.15. Evaluación por flujo bajo *Drop-Sel* y *Drop-Sel-delay* en el caso 6 con RED.



a)

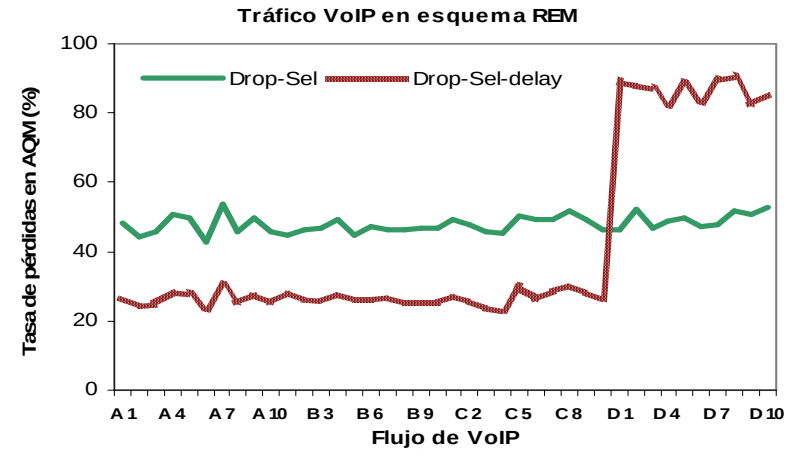


b)

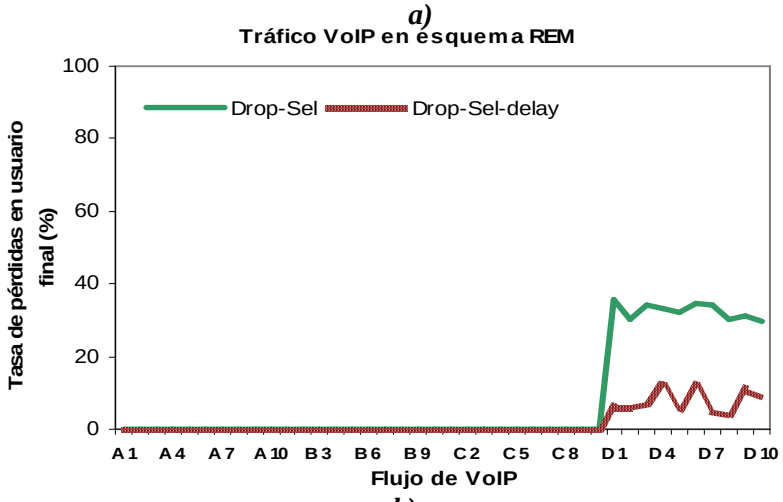


c)

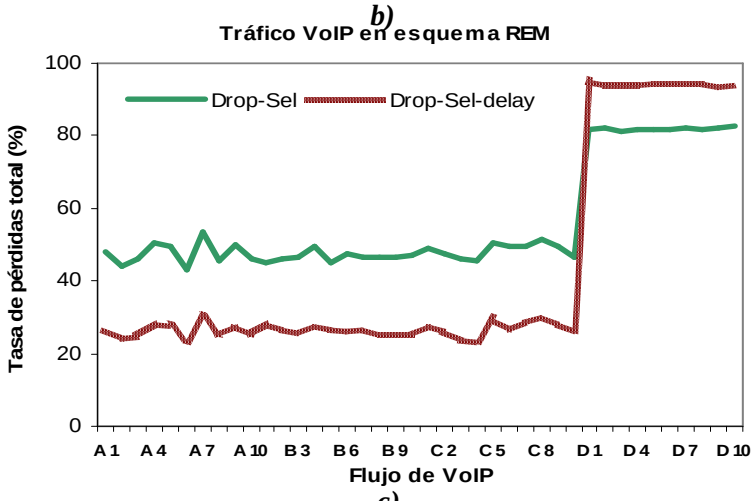
Figura 6.16. Evaluación por flujo bajo *Drop-Sel* y *Drop-Sel-delay* en el caso 6 con AVQ.



a)



b)



c)

Figura 6.17. Evaluación por flujo bajo *Drop-Sel* y *Drop-Sel-delay* en el caso 6 con REM.

Nótese en este caso que las fuentes VoIP obtienen una mayor reducción en la tasa de pérdidas respecto al caso 5. Cuando los AQM RED, AVQ y REM adoptan *Drop-Sel-delay* respectivamente reducen la tasa de pérdidas un 14.64% (Figura 6.15c), un 2.38% (Figura 6.16c) y un 15.67% (Figura 6.17c).

Con el propósito de medir el rendimiento en términos de calidad percibida por el usuario final, el esquema *Drop-Sel-delay* se evalúa en términos de puntuación MOS e inteligibilidad.

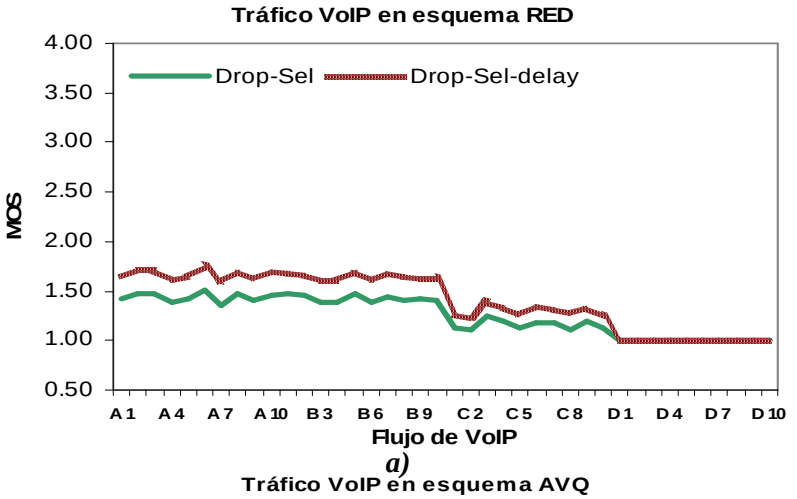
MOS

Las Figuras 6.18 y 6.19 muestran los valores MOS obtenidos para los diferentes flujos utilizando los métodos de descarte selectivo *Drop-Sel* de referencia y *Drop-Sel-delay* en el escenario S2 (casos 5 y 6).

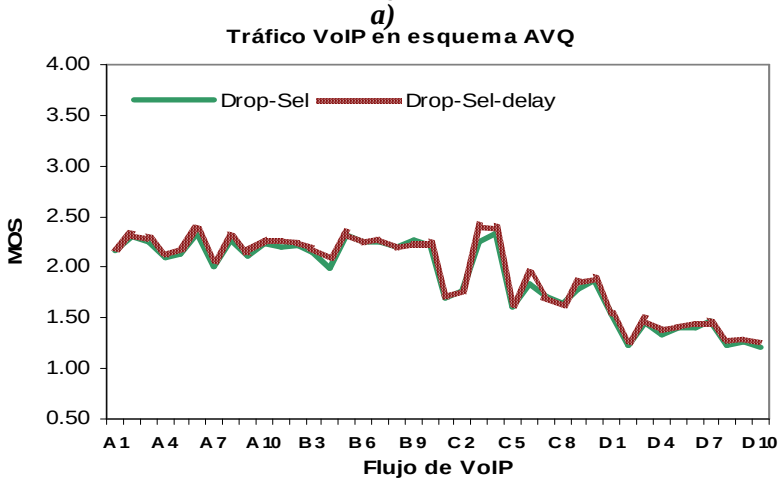
Nótese en las Figuras desde 6.12 a 6.17 que los flujos menos afectados por el retardo extremo-a-extremo (tipos A, B y C) experimentan con *Drop-Sel-delay* las tasas más bajas de descarte en la cola AQM y en el receptor. Como las pérdidas de paquetes es uno de los factores que influyen firmemente en la calidad subjetiva percibida por los usuarios, se observa en las Figuras 6.18 y 6.19 que los valores MOS obtenidos con *Drop-Sel-delay* para los flujos tipos A, B y C son superiores en relación a los que proporciona *Drop-Sel*.

Al contrario, los flujos más afectados por el retardo extremo-a-extremo (tipo D de Tabla 5.1) obtienen la puntuación MOS más baja. No obstante, *Drop-Sel-Delay* proporciona a estos flujos una degradación de la calidad similar a la suministrada por *Drop-Sel*. Más aún, en el esquema AVQ los valores MOS para estos flujos son ligeramente superiores.

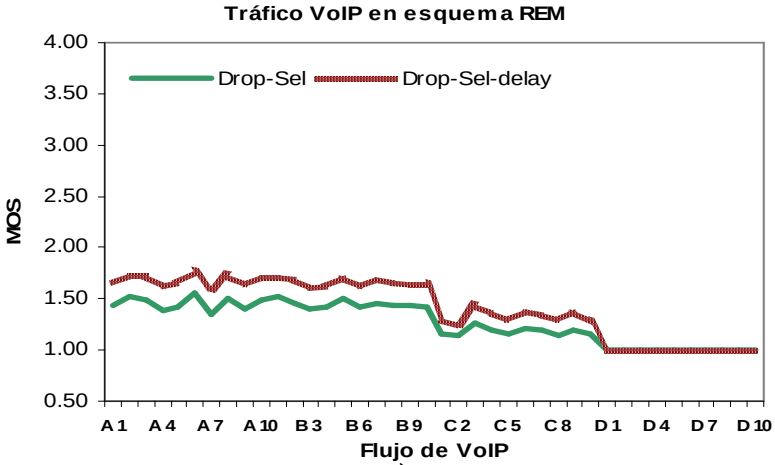
Para finalizar la evaluación MOS, en la Figura 6.20 se han establecido los resultados globales de *Drop-Sel-delay*, y para propósitos de comparación nuevamente son representados los de *Drop-Tail* y *Drop-Sel*. En los resultados expuestos se observa que *Drop-Sel-Delay* aminora la magnitud del deterioro de la calidad del tráfico VoIP bajo condiciones de tráfico para las cuales estas fuentes dominen, en particular en los esquemas RED y REM.



a)

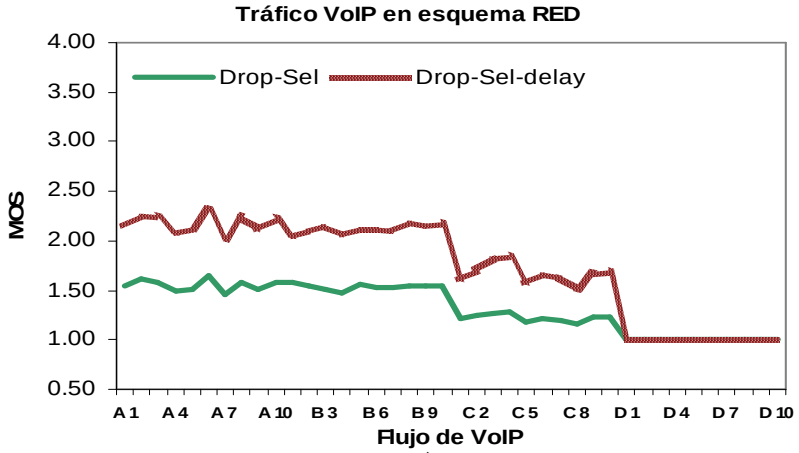


b)

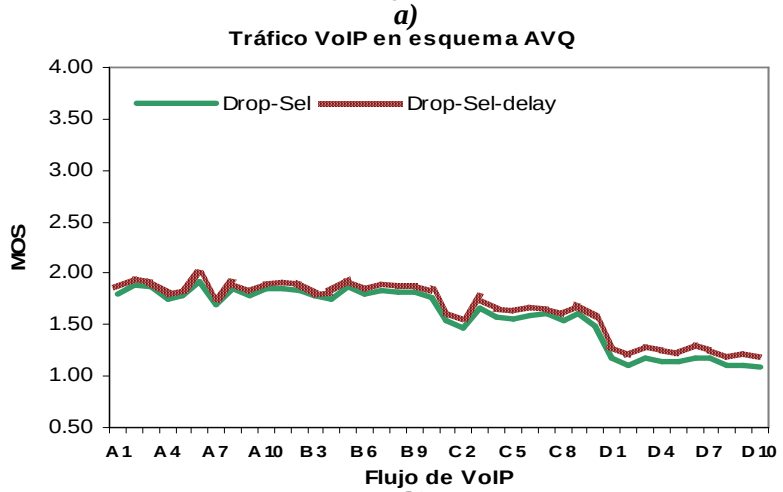


c)

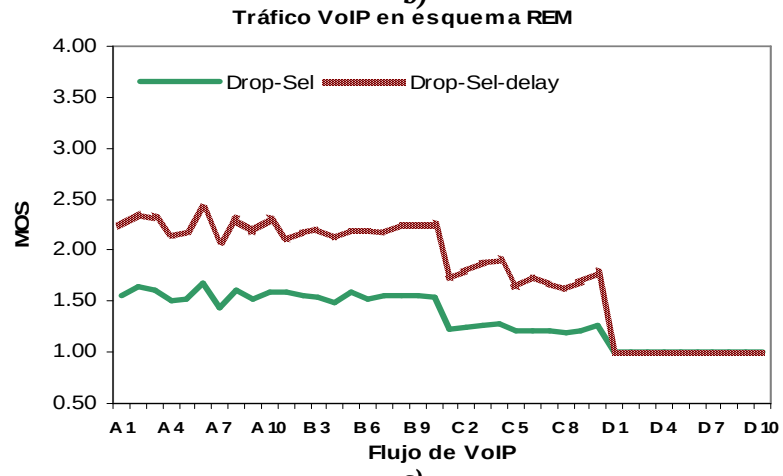
Figura 6.18. Evaluación MOS de *Drop-Sel* y *Drop-Sel-delay* en el caso 5.



a)

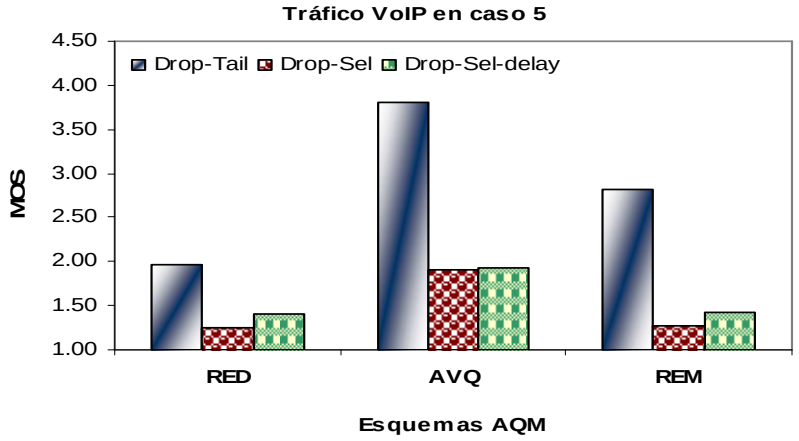


b)

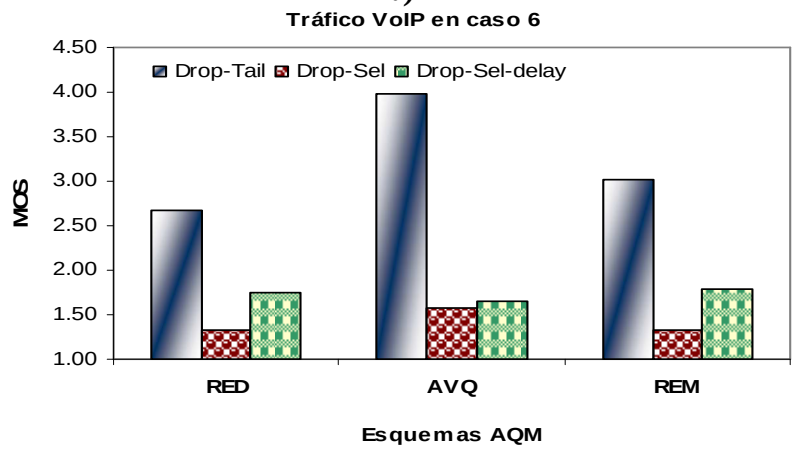


c)

Figura 6.19. Evaluación MOS de *Drop-Sel* y *Drop-Sel-delay* en el caso 6.



a)



b)

Figura 6.20. Evaluación MOS de *Drop-Tail*, *Drop-Sel* y *Drop-Sel-delay* en los casos 5 y 6.

Inteligibilidad

Se evalúa el funcionamiento de *Drop-Sel-delay* en términos de inteligibilidad asumiendo específicamente el esquema AQM RED con el caso 6 del escenario S2. La evaluación experimental ha demostrado que la alternativa *Drop-Sel-delay* proporciona mejor inteligibilidad extremo-a-extremo a los flujos menos afectados por el retardo. Las tasas de precisión de palabras (ver expresión 3.3) para los flujos A, B y C se muestran en la Tabla 6.1. En general, se observa que los valores son superiores a los generados por *Drop-Sel* de referencia.

Tabla 6.1.
TASA DE PRECISIÓN DE PALABRAS DE PAQUETES DE AUDIO POR FLUJO EN RED PARA EL CASO 6

	Método de descarte selectivo							
Flujo	*Drop-Sel*	*Drop-Sel delay*		*Drop-Sel*	*Drop-Sel delay*		*Drop-Sel*	*Drop-Sel delay*
A1	90.50	95.53	B1	78.92	87.45	C1	63.80	79.49
A2	92.09	96.79	B2	77.86	89.61	C2	66.28	81.64
A3	92.11	96.51	B3	77.40	91.18	C3	68.51	84.95
A4	89.26	95.35	B4	74.36	88.06	C4	68.65	84.80
A5	90.06	95.52	B5	78.39	90.06	C5	62.18	79.32
A6	93.19	97.16	B6	77.96	90.62	C6	64.43	81.31
A7	87.75	94.36	B7	76.59	89.50	C7	63.78	79.86
A8	91.52	96.15	B8	78.24	91.05	C8	60.27	77.89
A9	89.33	95.81	B9	78.72	91.10	C9	64.67	81.06
A10	91.66	96.45	B10	77.50	91.57	C10	66.59	82.78

Tabla 6.2.
MEJORA RELATIVA DE PRECISIÓN DE PALABRAS RESPECTO A *DROP -SEL* PARA CADA FLUJO

	Método de descarte *Drop-Sel-delay*					
	Flujo	Mejora (%)	***Flujo***	Mejora (%)	***Flujo***	Mejora (%)
	A1	5.56	B1	10.81	C1	24.59
	A2	5.10	B2	15.09	C2	23.17
	A3	4.78	B3	17.80	C3	24.00
	A4	6.82	B4	18.42	C4	23.53
	A5	6.06	B5	14.89	C5	27.57
	A6	4.26	B6	16.24	C6	26.20
	A7	7.53	B7	16.86	C7	25.21
	A8	5.06	B8	16.37	C8	29.24
	A9	7.25	B9	15.73	C9	25.34
	A10	5.23	B10	18.15	C10	24.31

Como se observa en la Tabla 6.2 *Drop-Sel-delay* permite mejoras relativas siempre superiores al 5.10%. El flujo más favorecido (C8), incrementa su porcentaje de inteligibilidad en cuestión de precisión de palabras de 60.27% a 77.89%, una mejora relativa del 29.24%.

Similarmente, las tasas de frases correctas de los flujos de VoIP en RED adoptando *Drop-Sel–delay* son mejores que en *Drop-Sel* de referencia. Estos resultados son mostrados en las Tablas 6.3 y 6.4. Dónde la mejora mínima obtenida con esta alternativa es de 11.40% para el flujo A6 (Tabla 6.4).

La mejora en la inteligibilidad extremo-a-extremo de los flujos VoIP, como fue

Tabla 6.3.
TASA DE FRASES CORRECTAS DE PAQUETES DE AUDIO POR FLUJO EN RED PARA EL CASO 6

	Método de descarte selectivo							
Flujo	*Drop-Sel*	*Drop-Sel delay*		*Drop-Sel*	*Drop-Sel delay*		*Drop-Sel*	*Drop-Sel delay*
A1	77.40	88.39	B1	60.54	72.41	C1	41.11	61.79
A2	80.42	90.71	B2	59.19	76.17	C2	43.96	64.06
A3	80.10	90.31	B3	57.87	79.12	C3	46.06	68.28
A4	75.60	87.64	B4	54.23	74.03	C4	46.73	68.13
A5	76.07	87.96	B5	58.89	76.85	C5	39.16	61.09
A6	82.40	91.79	B6	57.59	78.05	C6	41.34	64.26
A7	72.16	85.61	B7	57.57	76.07	C7	40.54	62.04
A8	79.32	89.29	B8	58.37	78.12	C8	37.24	59.42
A9	74.58	88.34	B9	59.02	78.77	C9	42.18	63.31
A10	79.50	90.16	B10	58.19	79.62	C10	43.73	65.56

Tabla 6.4.
MEJORA RELATIVA DE FRASES CORRECTAS RESPECTO A *DROP -SEL* PARA CADA FLUJO

	Método de descarte *Drop-Sel-delay*					
Flujo	Mejora (%)		***Flujo***	Mejora (%)	***Flujo***	Mejora (%)
A1	14.20		B1	19.61	C1	50.30
A2	12.80		B2	28.69	C2	45.72
A3	12.75		B3	36.72	C3	48.24
A4	15.93		B4	36.51	C4	45.79
A5	15.63		B5	30.50	C5	56.00
A6	11.40		B6	35.53	C6	55.44
A7	18.64		B7	32.13	C7	53.03
A8	12.57		B8	33.84	C8	59.56
A9	18.45		B9	33.46	C9	50.09
A10	13.41		B10	36.83	C10	49.92

previamente observado en la evaluación sobre equidad entre clases de tráfico (Figura 6.11a), no repercute significativamente en el servicio proporcionado a los flujos TCP y otros flujos UDP.

Como conclusión, estos resultados en combinación con los mostrados anteriormente en términos de MOS corroboran que en un AQM si el procedimiento de descarte selectivo *Drop-Sel* considera adicionalmente como criterio la utilidad o antigüedad de los paquetes de audio encolados, cuando intervienen conexiones con largos retardos de propagación, reduce el deterioro en la inteligibilidad bajo condiciones de tráfico para las cuales las fuentes VoIP dominen.

6.3.2. Selección automática de *Drop-Sel-delay* a *Drop-Tail-delay*

De forma similar al apartado 5.4.4, se ha simulado de forma concatenada el *caso 8* - *caso 6* - *caso 8* del escenario S2, considerando un sencillo algoritmo de selección que monitoriza la clase dominante para determinar si debe permutar de *Drop-Sel-delay* a *Drop-Tail-delay* y viceversa. Esta alternativa de *Drop-Sel* es denominada como *Drop-Sel(PT-delay*).

Equidad entre clases de tráfico

Se ha examinado el efecto sobre el rendimiento medio para las distintas clases de tráfico en tres diferentes períodos de la simulación total. Por cada período de tiempo se adopta alguno de los casos concatenados. Para el primer (desde t=0 seg hasta t=500 seg) y el tercer intervalo (desde t=1000 seg hasta t=1500 seg) se adopta el caso 8. En el segundo intervalo (desde t=500 seg hasta t=1000 seg) se adopta el caso 6.

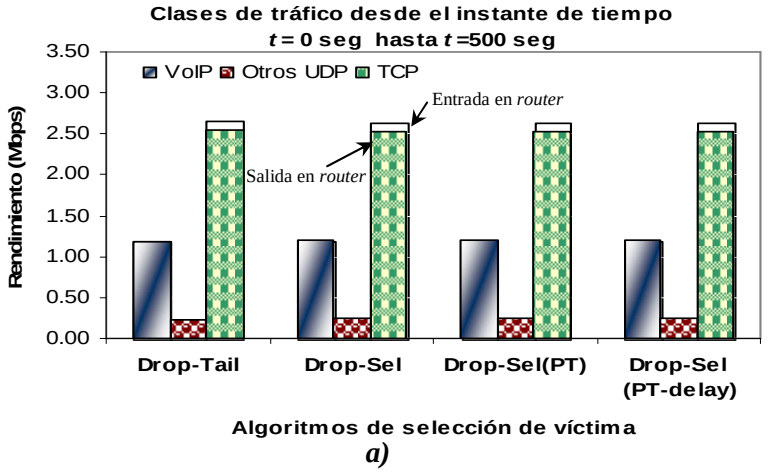
En la Figura 6.21 se ha establecido el rendimiento medio de entrada y de salida de *Drop-Sel(PT-delay)* en el esquema AQM RED y para propósitos de comparación los resultados con *Drop-Tail*, *Drop-Sel* y *Drop-Sel(PT)* son también representados.

Durante el primer y tercer intervalo (Figura 6.21a y 6.21c respectivamente), se observa que *Drop-Sel(PT-delay)* detecta que el tráfico VoIP no es la clase dominante y mantiene el comportamiento equitativo de *Drop-Sel-delay*. Por lo contrario, en el segundo período *Drop-Sel(PT-delay)* detecta al igual que *Drop-Sel(PT)* que el tráfico VoIP es la clase dominante y determina que debe permutar de *Drop-Sel-delay* a *Drop-Tail-delay*. De ahí que *Drop-Sel(PT-delay)* mantiene hasta el principio del tercer intervalo un comportamiento similar al de *Drop-Tail* (Figura 6.21b).

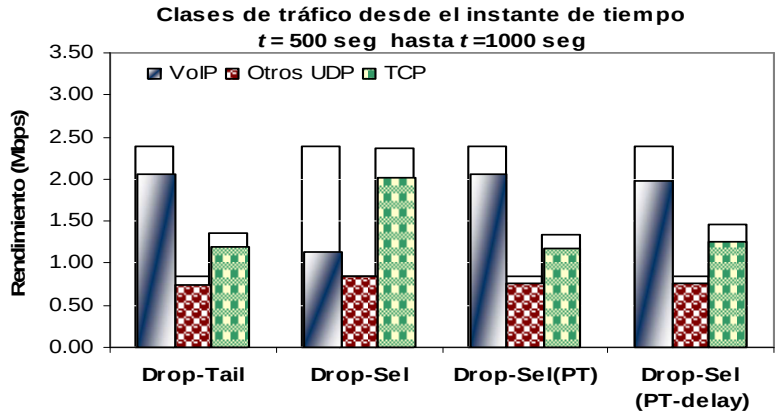
Tasa de pérdidas

Para mostrar el impacto de los procedimientos de selección de víctima de paquetes del AQM sobre la probabilidad de pérdidas para los paquetes de audio, las Figuras 6.22, 6.23 y 6.24 proporcionan los resultados obtenidos para cada uno de los intervalos de la simulación.

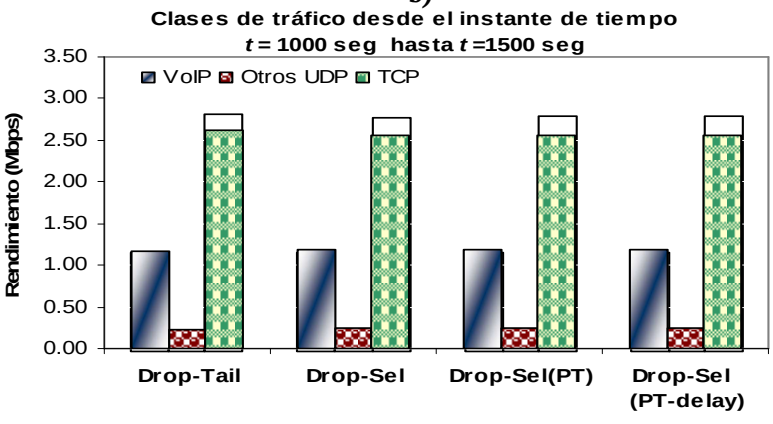
Drop-Sel(PT-delay), en cada período de tiempo, intenta adoptar la estrategia adecuada para favorecer al tráfico VoIP. En consecuencia, en el primer (Figura 6.22) y el tercer período (Figura 6.24), *Drop-Sel(PT-delay)* mantiene el comportamiento equitativo de *Drop-Sel-delay.*



a)



b)



c)

Figura 6.21. Evaluación del rendimiento medio de diferentes clases de tráfico en los casos concatenados 8, 6 y 8 del S2 con RED.

En la Figura 6.22a se observa que *Drop-Sel* y sus variantes tienden a reducir de forma similar la tasa de descarte en el *router* AQM cuando el tráfico VoIP no domina. De ahí que la tasa de pérdidas total de los flujos en los esquemas *Drop-Sel*, *Drop-Sel(PT)* y *Drop-Sel(PT-delay)* es muy similar (Figura 6.22c).

En el segundo intervalo *Drop-Sel(PT)* detecta al igual que *Drop-Sel(PT-delay)* que el tráfico VoIP es la clase dominante y determina que debe permutar a otra estrategia de selección de víctima. Sin embargo, al no considerar la vida útil del paquete *Drop-Sel(PT)* (al igual que *Drop-Tail y Drop-Sel)* permite que lleguen altas tasas de paquetes no-útiles (Figura 6.23b), típicamente de los flujos más afectados por el retardo extremo-a-extremo (D1 a D10).

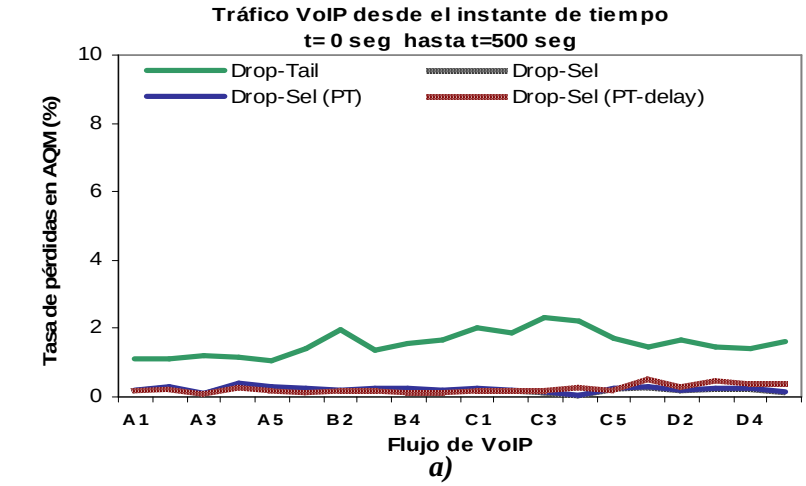
Por el contrario, *Drop-Sel(PT-delay)* detecta que los flujos D1 a D10 están generando paquetes no-útiles y aumenta su probabilidad de descarte en el AQM (Figura 6.23a). Recuérdese que tan pronto como los paquetes de audio considerados no-aprovechables sean detectados y descartados anticipadamente, el número de pérdidas de paquetes útiles puede ser reducido. De hecho, en la Figura 6.23c se observa que con *Drop-Sel(PT-delay)* los flujos de audio menos afectados por el retardo extremo-a-extremo (tipo A, B y C) experimentan más bajas tasas de pérdidas totales, independientemente de que estos sean los posibles causantes de la congestión.

En definitiva, *Drop-Sel(PT-delay)* mantiene el mismo comportamiento selectivo de paquetes que *Drop-Sel-delay* (ver apartado 6.3.1), pero difiere de este algoritmo en las condiciones de carga en las que es adoptado. Específicamente, *Drop-Sel(PT-delay)* es utilizado cuando el tráfico VoIP domina y se desea incrementar la calidad del servicio.

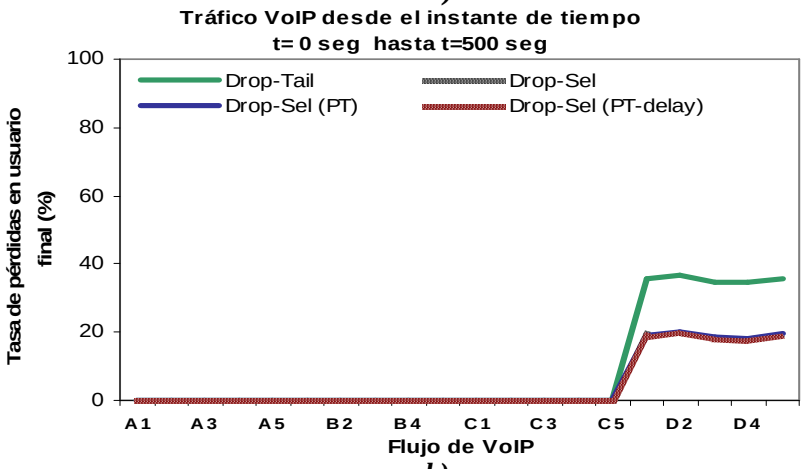
MOS

La Figura 6.25 muestra los valores MOS obtenidos para los diferentes intervalos establecidos y diferentes métodos de descarte. En los resultados expuestos se observa que los valores MOS suministrados por *Drop-Sel(PT-delay)* son iguales o superiores en relación a los que proporcionan los otros métodos adoptados.

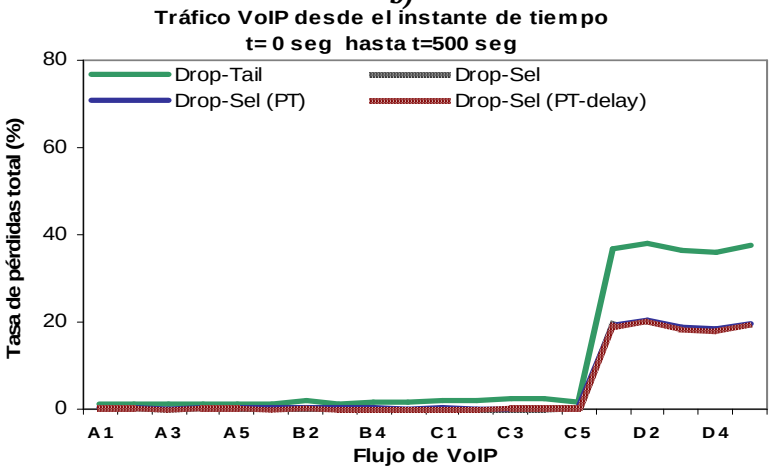
En definitiva, *Drop-Sel(PT-delay)* consigue una menor degradación de la calidad por permutar de un algoritmo de selección de víctima a otro (en el segundo intervalo) y además, por conservar en la cola los paquetes evaluados como útiles y descartar preferentemente posibles paquetes no-útiles.



a)

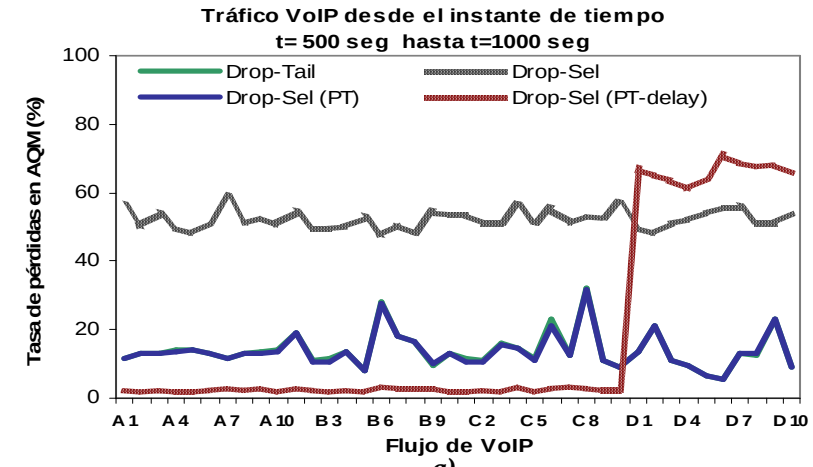


b)

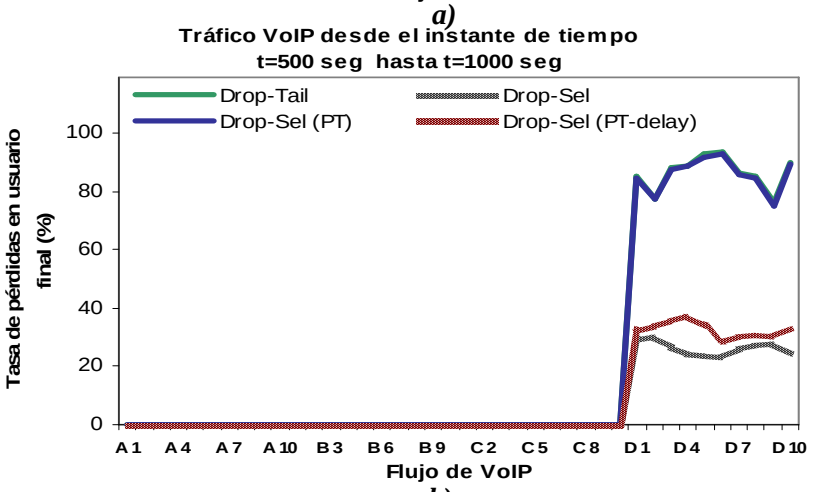


c)

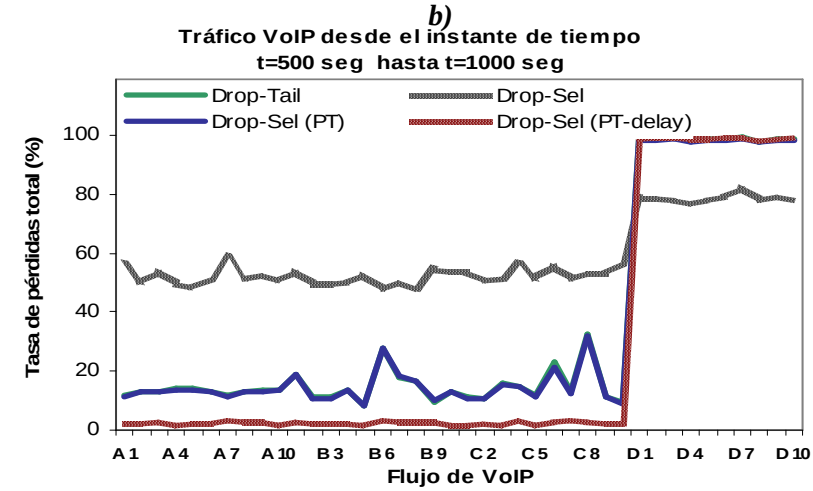
Figura 6.22. Evaluación por flujo en el primer período para los casos concatenados 8, 6 y 8 del S2 con RED.



a)

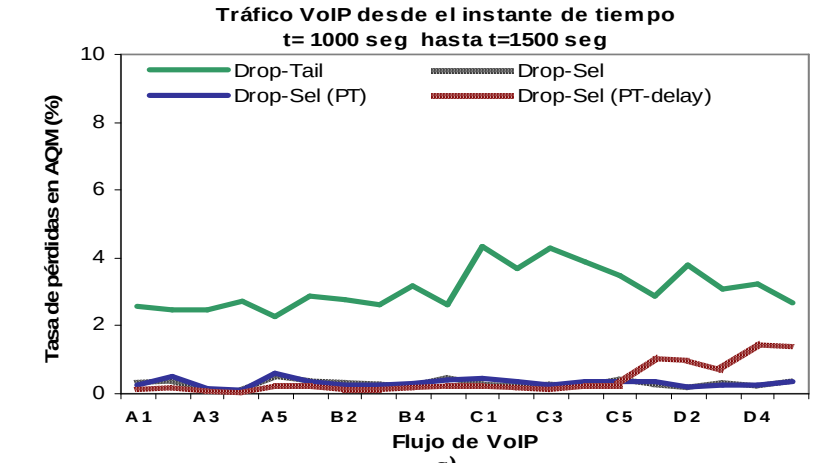


b)

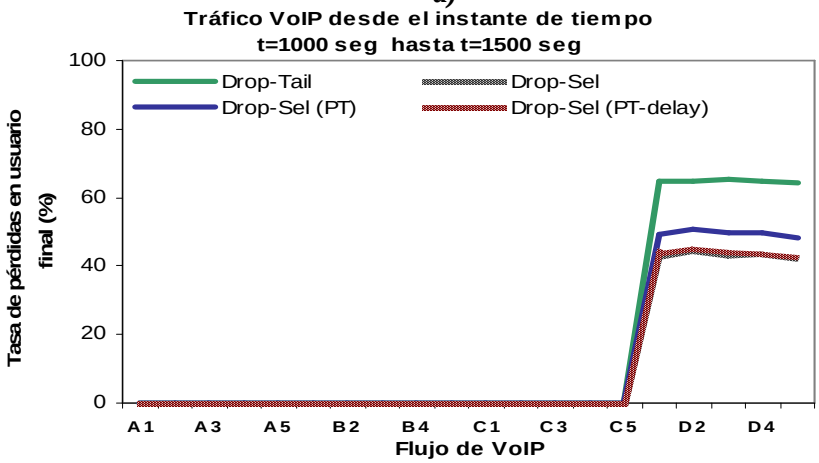


c)

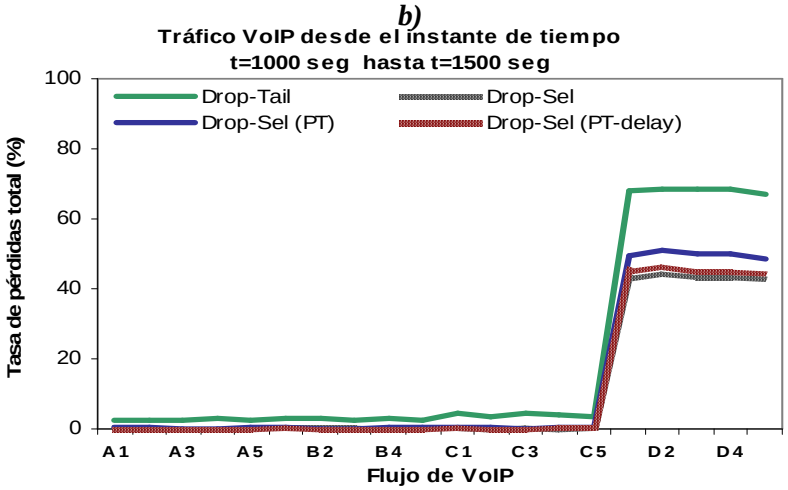
Figura 6.23. Evaluación por flujo en el segundo período para los casos concatenados 8, 6 y 8 del S2 con RED.



a)



b)



c)

Figura 6.24. Evaluación por flujo en el tercer período para los casos concatenados 8, 6 y 8 del S2 con RED.

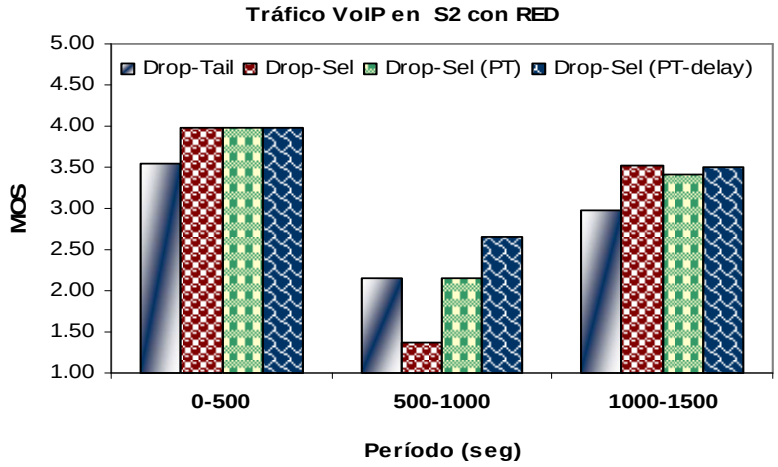


Figura 6.25. Evaluación MOS para los casos concatenados 8, 6 y 8 del S2 con RED.

6.3.3. *Drop-Sel+fair*

Al no proporcionar un tratamiento por flujo, *Drop-Sel* no siempre garantiza un servicio equitativo a diferentes flujos de VoIP, y puede provocar una ligera tasa de descarte más alta para unos flujos con respecto a otros. Para aminorar este problema, *Drop-Sel+fair* da un tratamiento diferenciado a los flujos de VoIP. Al igual que *Drop-Sel-delay*, este esquema busca el paquete de voz idóneo para seleccionarlo como víctima de descarte. En este caso en particular, se busca el paquete de voz del flujo con más baja tasa de descarte.

En este sentido se evalúa el funcionamiento de *Drop-Sel+fair* considerando dos escenarios, específicamente los casos 2 (S1) y 6 (S2).

En el caso 2, un número de fuentes TCP y otros flujos UDP no-reactivos compiten con 13 flujos VoIP (UDP). Específicamente, 4 de los flujos VoIP (denominados como A1 a A4), son generados utilizando la aplicación de VoIP tipo A (ver Tabla 5.1), 3 con la aplicación tipo B (denominados como B1 a B3), 4 con la aplicación tipo C (denominados como C1 a C4) y finalmente, 2 de los flujos VoIP con la aplicación tipo D (denominados como D1 y D2).

Para el caso 6 se consideran 40 flujos VoIP (UDP). Concretamente, 10 son generados con la aplicación tipo A (denominados como A1 a A10), 10 con la aplicación tipo B (denominados como B1 a B10), 10 con la aplicación tipo C (denominados como C1 a C10) y finalmente, 10 de los flujos VoIP con la aplicación tipo D (denominados como D1 a D10).

Equidad entre clases de tráfico

Drop-Sel+fair, al igual que *Drop-Sel-delay,* conserva el mismo criterio de selección de paquetes que *Drop-Sel* de referencia cuando hay congestión. En relación a esto, se ha examinado el efecto sobre el rendimiento medio para las distintas clases de tráfico. En las Figuras 6.26 y 6.27 se han establecido el rendimiento medio de entrada y de salida de *Drop-Sel+fair* en los distintos esquemas AQM y además para propósitos de comparación los resultados con *Drop-Tail* y *Drop-Sel*.

Primeramente, para el caso 2 del escenario S1, se observa en RED (Figura 6.26a) y REM (Figura 6.26c) que *Drop-Sel+fair* equipara el consumo de VoIP a la de los otros tráficos UDP y TCP, similar a como lo hizo *Drop-Sel*. Se puede observar que en este caso las otras fuentes UDP obtienen el máximo ancho de banda requerido (31%) y se reparten más equitativamente los recursos extras no utilizados a las otras clases de tráficos. *Drop-Sel+fair* asigna un 36% (VoIP) y un 33% (TCP) en RED y un 39% (VoIP) y un 30% (TCP) en REM. Al igual que *Drop-Sel,* la variante no consigue una asignación tan equitativa en el esquema AVQ para los tráficos TCP y VoIP (Figura 6.26b) como en RED y REM.

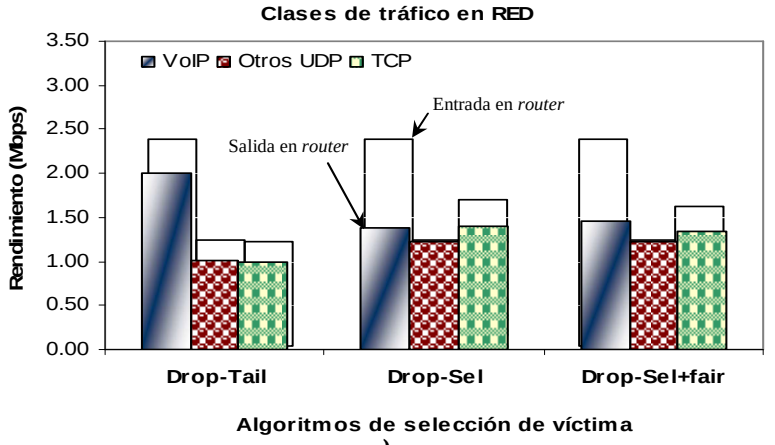
En el caso 6 del escenario S2 (Figura 6.27), *Drop-Sel+fair* nuevamente mantiene el comportamiento equitativo de *Drop-Sel* en los tres AQM. Más aún, se observa que en AVQ (Figura 6.27b) logra ser más justo, y que las otras fuentes UDP obtienen aproximadamente el máximo ancho de banda necesitado, igualando el consumo (39%) de VoIP y TCP.

En conclusión, el servicio medio proporcionado a los flujos VoIP, otros UDP y TCP es similar en ambos esquemas.

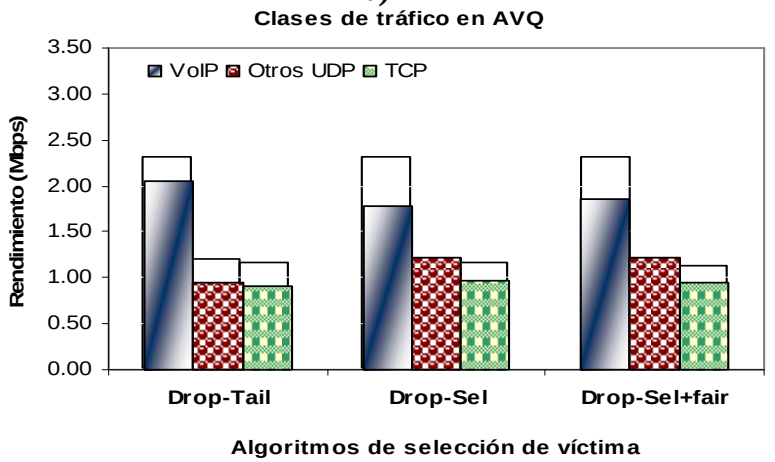
Tasa de pérdidas

Se evalúa el impacto de *Drop-Sel+fair* en términos de probabilidad de pérdidas en el *router*.

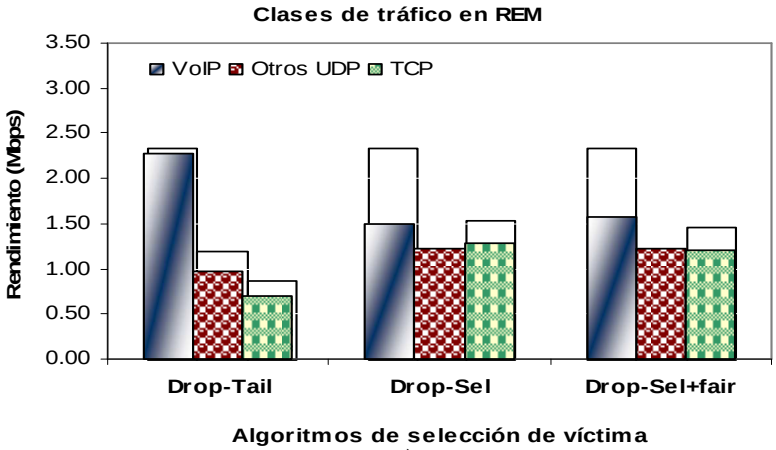
En resultados para el caso 2 del escenario S1 (Figura 6.28) se observa que las tasas de pérdidas de paquetes de audio en el *router* son muy variadas al adoptar *Drop-Sel*, independientemente del tipo de aplicación (A, B, C ó D de la Tabla 5.1) y el esquema AQM utilizado. Se observa que la estrategia de *Drop-Sel* de descartar el paquete más próximo a salir de la cola, cuando las conexiones tienen igual RTT, induce a que las tasas de pérdidas de las fuentes del mismo tipo de aplicación se reduzcan en el orden en que se inició la generación de los paquetes.



a)

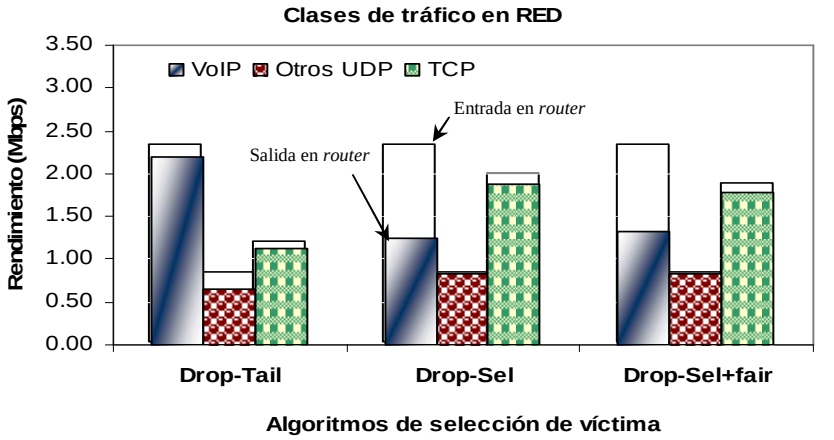


b)

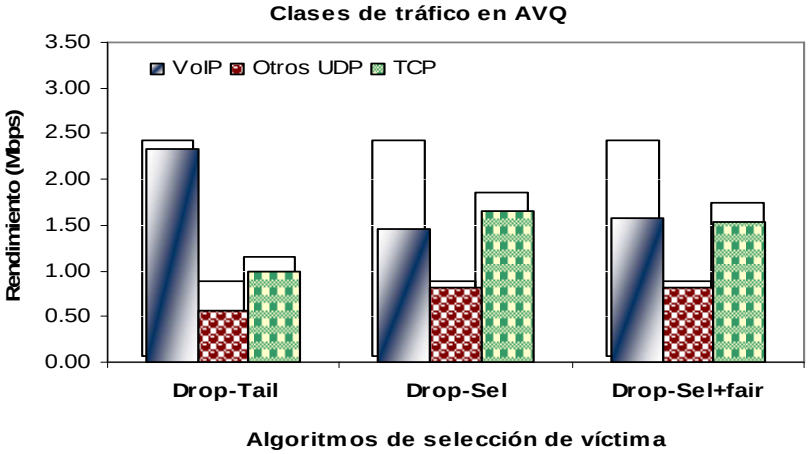


c)

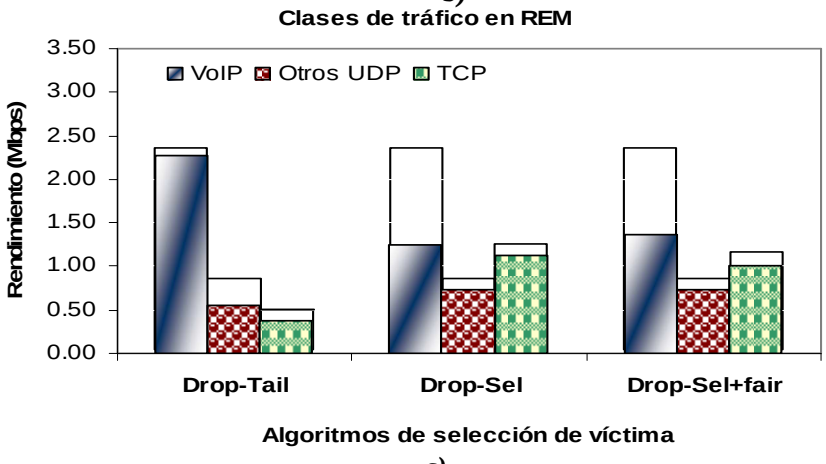
Figura 6.26. Evaluación del rendimiento medio de diferentes clases de tráfico bajo *Drop-Tail*, *Drop-Sel* y *Drop-Sel+fair* en el caso 2.



a)



b)



c)

Figura 6.27. Evaluación del rendimiento medio de diferentes clases de tráfico bajo *Drop-Tail*, *Drop-Sel* y *Drop-Sel+fair* en el caso 6.

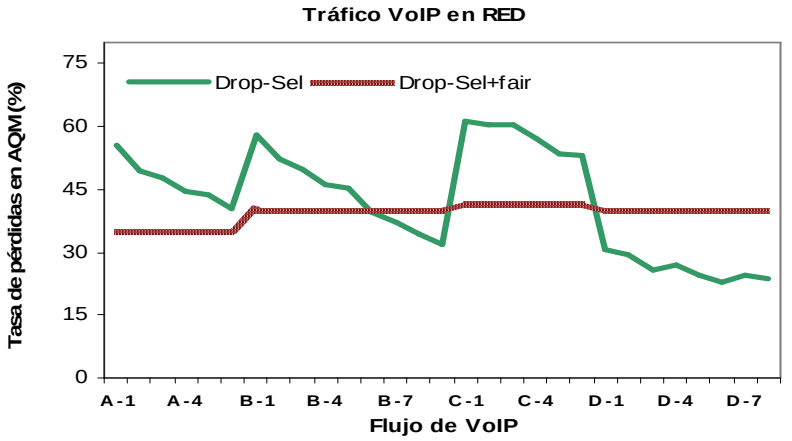
Cuando RED adopta *Drop-Sel* (Figura 6.28a), se observa que las tasas de pérdidas de flujos tipo A varían en un rango desde 40.55% a 55.71% y en flujos tipo B, desde 31.84% a 58.16%. La tasa de descarte del flujo B1 (58.16%) es casi el doble de la proporcionada para B9 (31.84%). Similarmente, la falta de equidad entre flujos de VoIP (de un mismo tipo de aplicación o de distinta) se presenta para los esquemas AVQ (Figura 6.28b) y REM (Figura 6.28c). Por ejemplo, en REM se observa que la tasa de pérdidas del flujo C2 (57.08%) es casi el triple de la tasa de pérdidas en el *router* del flujo D8 (20.96%).

Al contrario, se observa que *Drop-Sel+fair,* al utilizar una mínima información por flujo, otorga más equitativamente los recursos entre los flujos de VoIP independientemente del tipo de aplicación y el esquema utilizado. En párrafos anteriores se mostraba que *Drop-Sel* adoptado en RED tiende a ser menos justo con el flujo B1 en relación con el flujo B9, nótese que *Drop-Sel+fair* iguala las tasas de pérdidas (39.84%) para ambos flujos. Más aún, establece las mismas tasas de pérdidas para los flujos de un mismo tipo de aplicación y tiende a ser más equitativo con los distintos tipos de aplicación. Específicamente, obtiene un 35.24% (A), un 39.84% (B), un 41.67% (C) y un 39.84% (D). Dado que las aplicaciones B y D tienen las mismas características en el escenario S1, ambos tipos de flujos obtienen tasas de descarte idénticas. En AVQ y REM, *Drop-Sel+fair* mantiene el mismo comportamiento equitativo entre los flujos de VoIP.

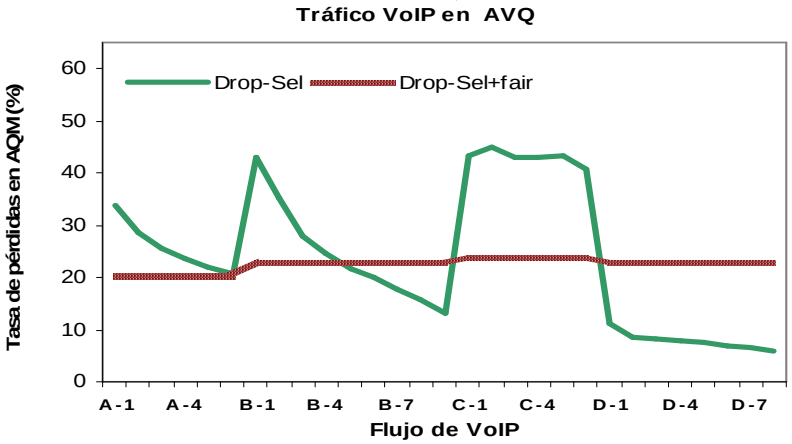
Por otro lado, en el caso 6 del escenario S2 (Figura 6.29), se observa que las tasas de pérdidas obtenidas por los flujos de VoIP en *Drop-Sel* (en los tres AQM) no varían tan drásticamente como en S1. Sin embargo, tiene un comportamiento no-equitativo entre los flujos de un mismo tipo de aplicación. Más específicamente, cuando REM adopta *Drop-Sel* (Figura 6.29c), se observa que las tasas de pérdidas de los flujos tipo B varían en un rango desde 44.81% a 49.47% y los flujos tipo C, desde 45.50% a 51.57%.

Similarmente al escenario S1, una interesante reducción en la desigualdad de tasas de pérdidas (para flujos de un mismo tipo de aplicación) se efectúa para los tres AQM al adoptar *Drop-Sel+fair* en el escenario S2. Por ejemplo, para el esquema REM (previamente analizado, Figura 6.29c), se observa que las tasas de pérdidas para los flujos B se aproximan con porcentajes desde 44.01% a 44.35% y para los flujos tipo C desde 46.23% a 46.56%.

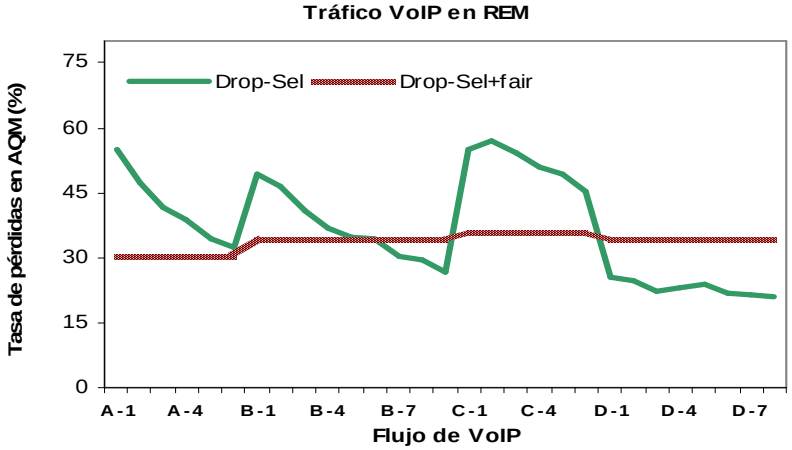
Drop-Sel+fair modifica el orden de los descartes dando más prioridad a paquetes de mayor tamaño que en *Drop-Sel* no se habían considerado, alterando con ello la ocupación del tráfico VoIP en la cola e inherentemente la asignación de descarte entre las clases de tráfico.



a)

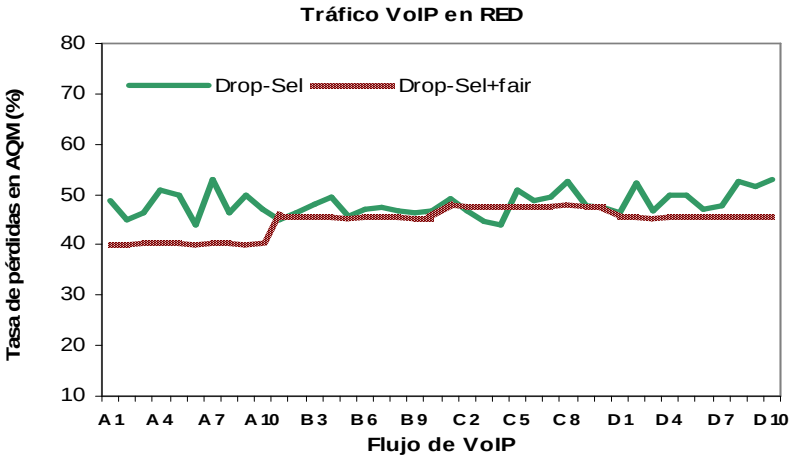


b)

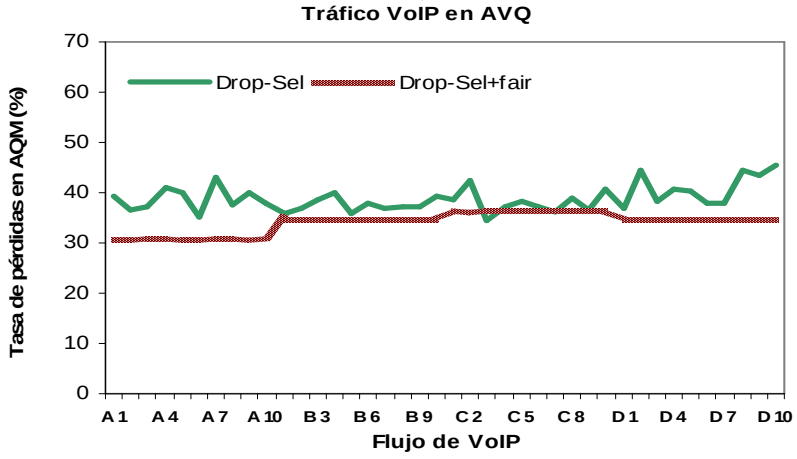


c)

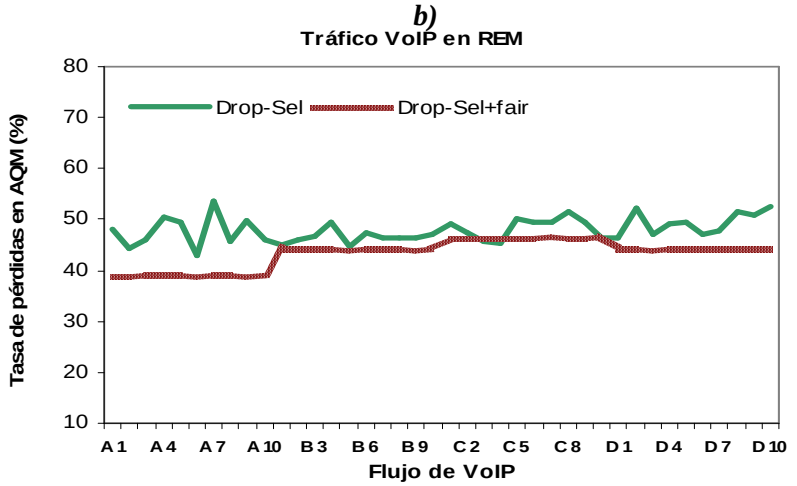
Figura 6.28. Evaluación por flujo de VoIP bajo *Drop-Sel* y *Drop-Sel+fair* en el caso 2.



a)



b)



c)

Figura 6.29. Evaluación por flujo de VoIP bajo *Drop-Sel* y *Drop-Sel+fair* en el caso 6.

En la Figura 6.29 se observa que con *Drop-Sel+fair* el porcentaje de pérdidas en los AQM tienden a bajar ligeramente. No obstante, como fue previamente observado en la evaluación sobre equidad entre clases de tráfico (Figura 6.27), esto no repercute significativamente en el servicio proporcionado a los flujos TCP y otros UDP.

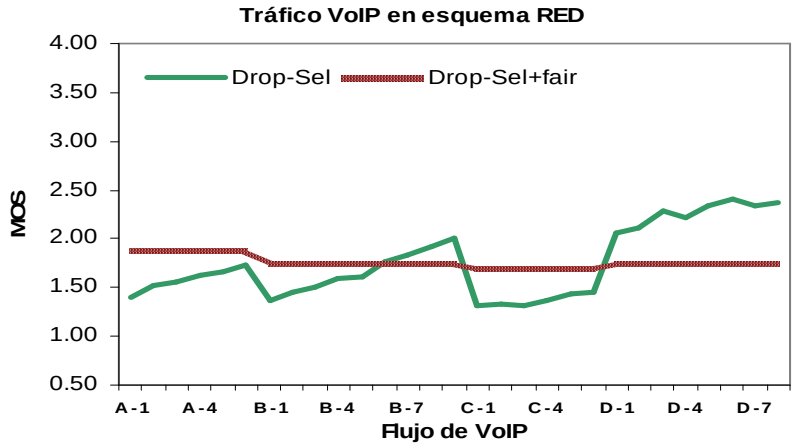
MOS

Con el propósito de medir el rendimiento en términos de calidad percibida por el usuario final, al aplicar *Drop-Sel+fair*, su rendimiento se evalúa en términos de puntuación MOS. Las Figuras 6.30 y 6.31 muestran los valores MOS obtenidos para los diferentes flujos utilizando los métodos de descarte selectivo *Drop-Sel* de referencia y *Drop-Sel+fair* en el escenario S1 (caso 2) y el escenario S2 (caso 6).

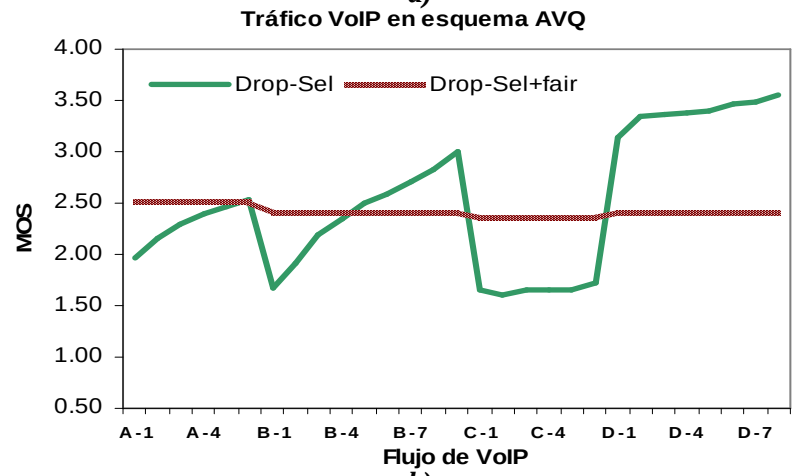
Primeramente, para el caso 2 del escenario S1 (Figura 6.30), se observa que *Drop-Sel* refleja en los valores MOS la variación generada entre tasas de pérdidas en el *router*. Dado que *Drop-Sel+fair* otorga más equitativamente los recursos entre los flujos de VoIP, este algoritmo proporciona a los flujos una calidad muy similar, independientemente del tipo de aplicación (A, B, C ó D) y del esquema AQM utilizado. Por ejemplo, en el esquema RED (Figura 6.30a), proporciona específicamente 1.89 (a flujos tipo A), 1.75 (B), 1.70 (C) y 1.75 (D). Esta tendencia equitativa también se observa para los esquemas AVQ (Figura 6.30b) y REM (Figura 6.30c).

En el caso 6, *Drop-Sel+fair* reduce la desigualdad de las tasas de pérdidas en el *router*. Sin embargo, el retardo extremo-a-extremo es otro de los factores que firmemente influyen en el deterioro de la calidad de los paquetes de voz. En la Figura 6.31 se puede observar que los flujos menos afectados por el retardo extremo-a-extremo (flujos tipo A) experimentan los valores MOS más elevados al adoptar *Drop-Sel+fair*. Al contrario, los flujos más afectados por el retardo (tipo D) obtienen la puntuación MOS más baja. No obstante, *Drop-Sel+fair* proporciona una calidad mejor que la suministrada por *Drop-Sel*. Más aún, la alternativa otorga la misma calidad entre los flujos de un mismo tipo de aplicación.

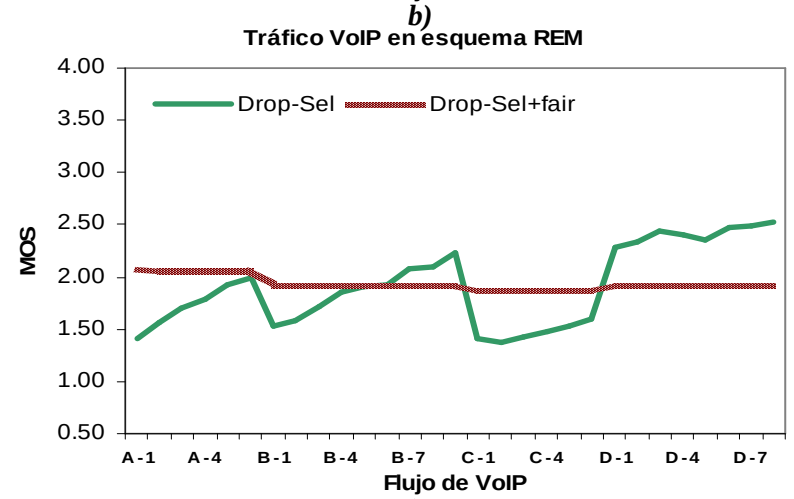
Para finalizar la evaluación MOS, en la Figura 6.32 se muestran los resultados en términos generales de *Drop-Sel+fair*, y para propósitos de comparación son también representados los resultados de *Drop-Tail* y *Drop-Sel*. En estos resultados se observa que *Drop-Sel+fair* reduce ligeramente el deterioro en la calidad del tráfico VoIP en el escenario S2.



a)

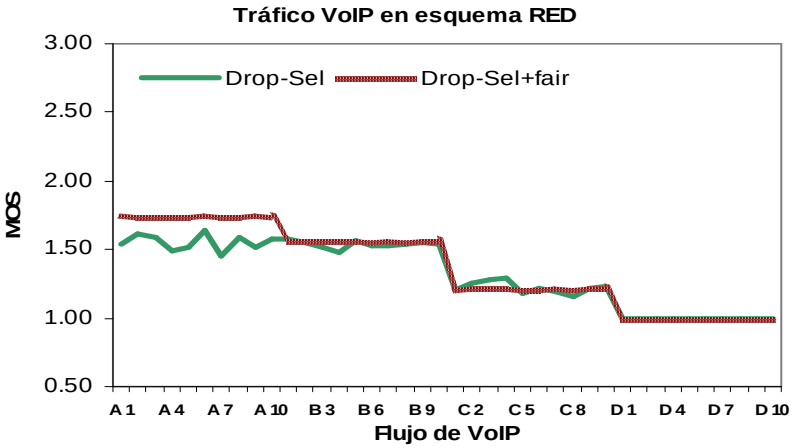


b)

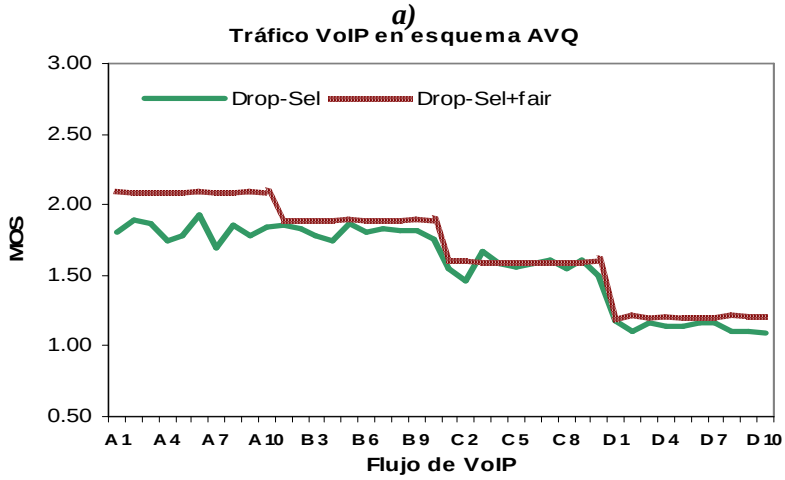


c)

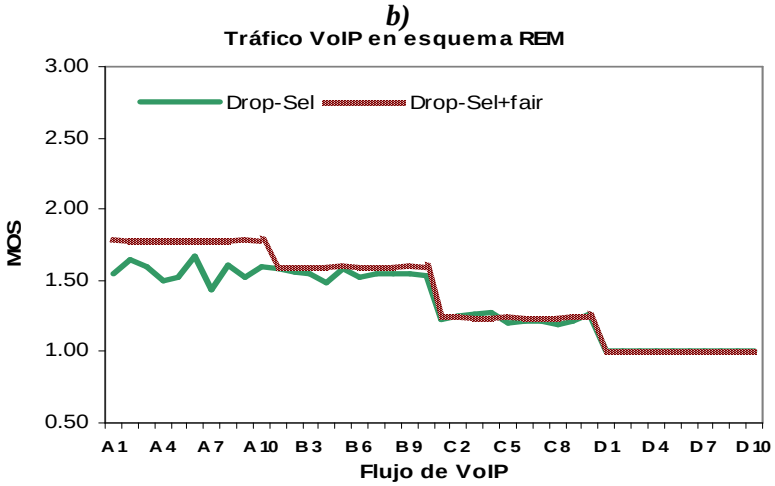
Figura 6.30. Evaluación MOS de *Drop-Sel* y *Drop-Sel+fair* en el caso 2.



a)



b)



c)

Figura 6.31. Evaluación MOS de *Drop-Sel* y *Drop-Sel+fair* en el caso 6.

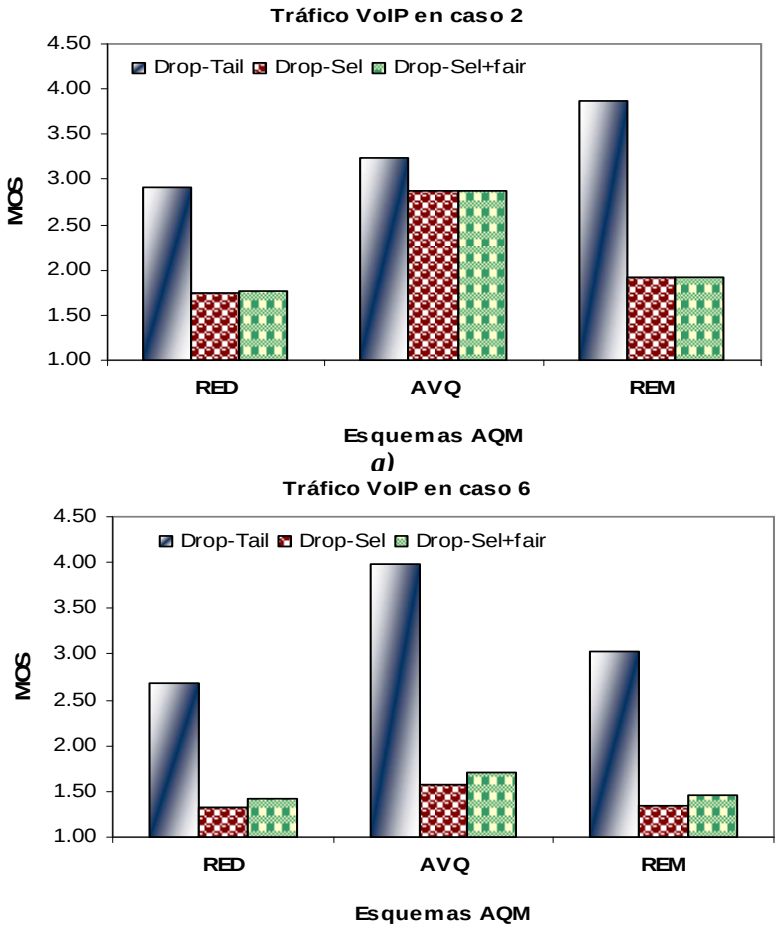


Figura 6.32. Evaluación MOS de *Drop-Tail*, *Drop-Sel* y *Drop-Sel+fair* en los casos 2 y 6.

Como conclusión, si el procedimiento de descarte selectivo *Drop-Sel* considera el número de paquetes descartados por flujo de VoIP bajo condiciones de tráfico para las cuales las fuentes VoIP dominen, se consigue un servicio más equitativo para los paquetes de voz sin degradar significativamente el servicio medio proporcionado al tráfico TCP y otras fuentes UDP.

6.3.4. Selección automática de *Drop-Sel+fair* a *Drop-Tail+fair*

De forma similar al apartado 5.4.4, se ha simulado de forma concatenada el *caso 3* - *caso 2* - *caso 3* del escenario S1, considerando un sencillo algoritmo de selección que monitoriza la clase dominante para determinar si debe permutar de *Drop-Sel+fair* a *Drop-Tail+fair* y viceversa. Esta alternativa de *Drop-Sel* es denominada como *Drop-Sel(PT+fair*).

Equidad entre clases de tráfico

Se ha examinado el efecto sobre el rendimiento medio para las distintas clases de tráfico en tres diferentes períodos de tiempo de la simulación total. Por cada período se adopta alguno de los casos concatenados. Para el primer (desde t=0 seg hasta t=500 seg) y el tercer intervalo (desde t=1000 seg hasta t=1500 seg) se adopta el caso 3. En el segundo intervalo (desde t=500 seg hasta t=1000 seg) se adopta el caso 2.

En la Figura 6.33 se ha establecido el rendimiento medio de entrada y de salida de *Drop-Sel(PT-delay)* en el esquema AQM RED y para propósitos de comparación los resultados con *Drop-Tail*, *Drop-Sel* y *Drop-Sel(PT)* son también representados.

Durante el primer y tercer intervalo (Figura 6.33a y 6.33c respectivamente), se observa que *Drop-Sel(PT+fair)* detecta que el tráfico VoIP no es la clase dominante y mantiene el comportamiento equitativo de *Drop-Sel+fair*. Por lo contrario, en el segundo período el algoritmo detecta al igual que *Drop-Sel(PT)* que el tráfico VoIP es la clase dominante y determina que debe permutar de *Drop-Sel+fair* a *Drop-Tail+fair*. De ahí que *Drop-Sel(PT+fair)* mantiene hasta el principio del tercer intervalo un comportamiento similar al de *Drop-Tail* (Figura 6.33b).

Tasa de pérdidas

Se evalúa el impacto de *Drop-Sel(PT+fair*) en términos de probabilidad de pérdidas en el *router* y en el usuario final para cada uno de los períodos.

Dado que el algoritmo *Drop-Sel* y sus alternativas tienden a reducir al máximo las tasas de pérdidas de los paquetes de audio (cuando las fuentes VoIP no dominan), no es de extrañar que las tasas obtenidas por estos esquemas en el primer (Figura 6.34a) y el tercer intervalo (Figura 6.34c) sean mucho más bajas que las generadas por *Drop-Tail*. No obstante, *Drop-Sel(PT+fair*) tiende a ser más equitativo entre los flujos cuando tiene que descartar paquetes.

En los resultados para el segundo período (Figura 6.34b) se observa que las tasas de pérdidas de paquetes de audio en el *router* son muy variadas al adoptar *Drop-Tail*, *Drop-Sel* y *Drop-Sel(PT)* independientemente del tipo de aplicación (A, B, C ó D de la Tabla 5.1). Al contrario, se observa que *Drop-Sel(PT+fair*), otorga más equitativamente los recursos entre los flujos de VoIP independientemente de que estos sean los causantes de la congestión. Nótese que *Drop-Sel(PT+fair*) establece las mismas tasas de pérdidas para los flujos de un mismo tipo de aplicación y tiende a ser más equitativo con los distintos tipos de aplicación.

En definitiva, *Drop-Sel(PT+fair)* mantiene el mismo comportamiento equitativo entre los flujos de VoIP que *Drop-Sel+fair* (ver apartado 6.3.3), pero difiere de este algoritmo en las condiciones de carga en las que es adoptado. Específicamente, *Drop-Sel(PT+fair)* es utilizado cuando el tráfico VoIP domina.

MOS

Para medir el rendimiento en términos de calidad percibida por el usuario final, al aplicar *Drop-Sel(PT+fair)*, su rendimiento se evalúa en términos de puntuación MOS.

La Figura 6.35 muestra los valores MOS obtenidos para los diferentes períodos utilizando los métodos de descarte. En la Figura 6.35b se observa que *Drop-Tail*, *Drop-Sel* y *Drop-Sel(PT)* proporcionan una calidad subjetiva mucho más variada que *Drop-Sel(PT+fair).* Dado que *Drop-Sel(PT+fair)* otorga más equitativamente los recursos entre los flujos de VoIP, no es de extrañar que este algoritmo proporcione a los flujos una calidad muy similar, independientemente del tipo de aplicación (A, B, C ó D).

En conclusión, *Drop-Sel(PT+fair)* es una prometedora alternativa a utilizarse cuando se desee mantener una satisfactoria y equitativa calidad de servicio para distintos flujos de audio, bajo condiciones de tráfico para las cuales las fuentes VoIP dominen.

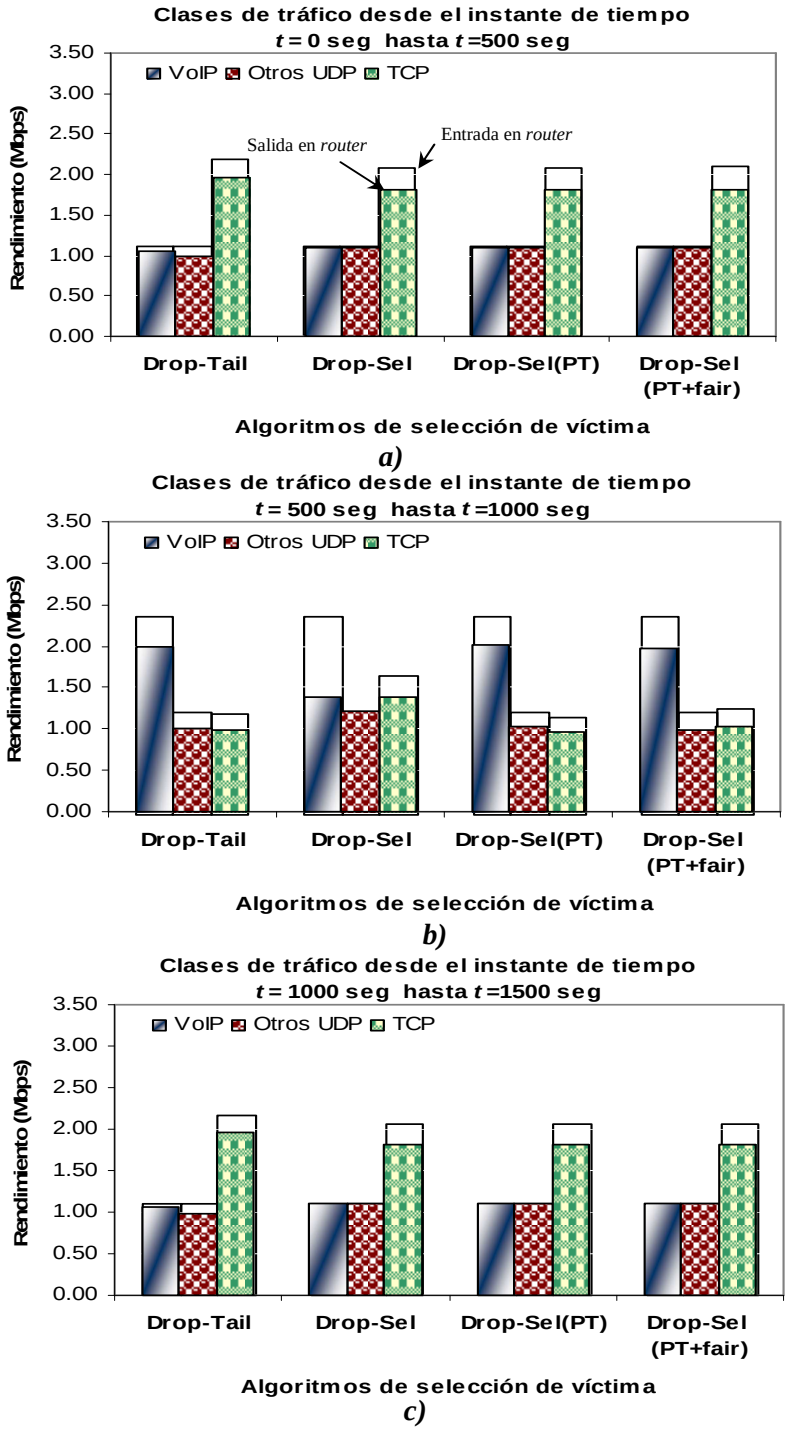
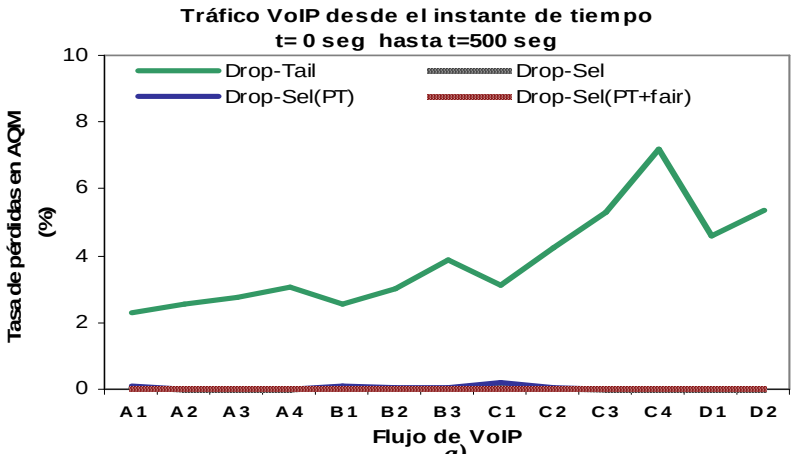
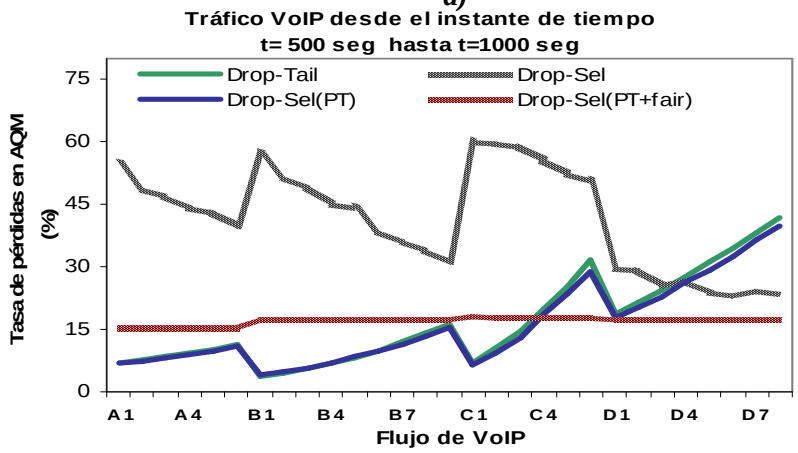


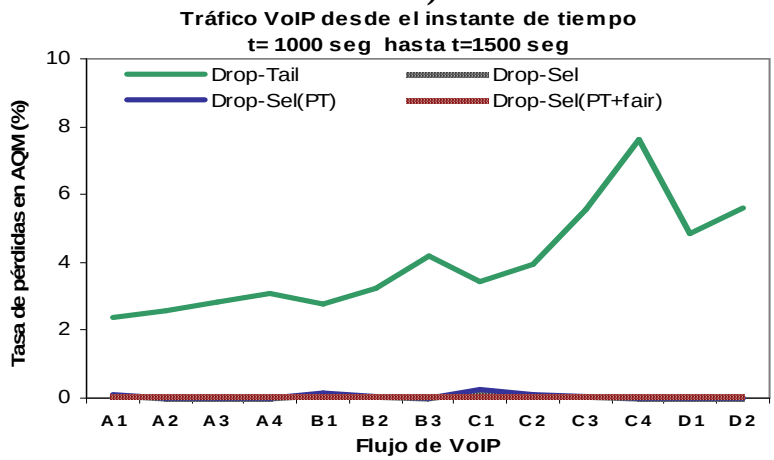
Figura 6.33. Evaluación del rendimiento medio de diferentes clases de tráfico en los casos concatenados 3, 2 y 3 del S2 con RED.



a)

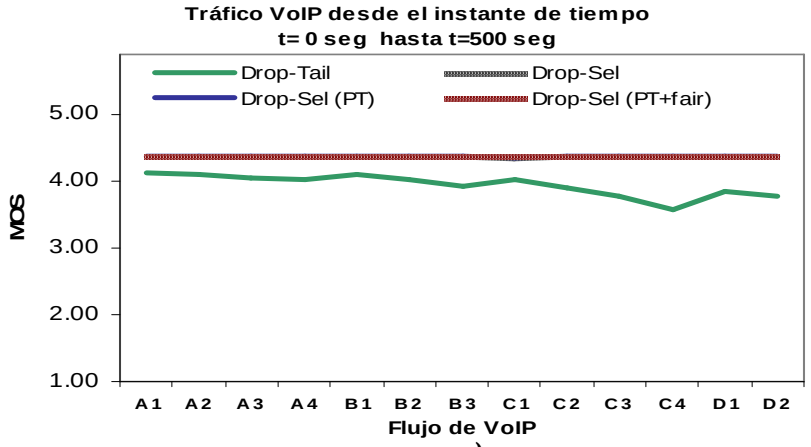


b)

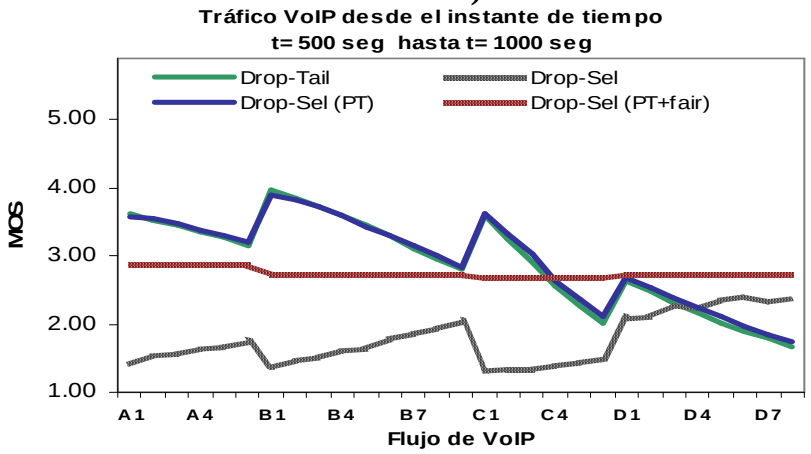


c)

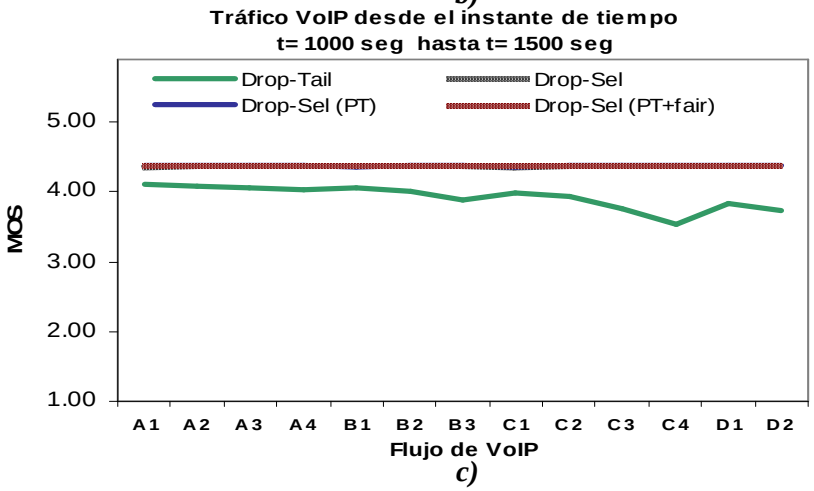
Figura 6.34. Evaluación por flujo de VoIP en los casos concatenados 3, 2 y 3 del S2 con RED.



a)



b)



c)

Figura 6.35. Evaluación MOS en los casos concatenados 3, 2 y 3 del S2 con RED.

6.3.5. Combinación de alternativas de *Drop-Sel*

En los apartados 6.3.1 y 6.3.3 se ha observado que si el procedimiento de descarte selectivo *Drop-Sel* considera adicionalmente como criterio la utilidad del paquete de audio encolado ó el número de paquetes descartados por flujo, se obtiene una reducción del deterioro en la calidad de servicio. Estas alternativas del esquema de descarte selectivo denominadas *Drop-Sel-delay* y *Drop-Sel+fair* han sido adoptadas y evaluadas en forma independiente, otorgando cada una de ellas diferentes beneficios a los flujos de VoIP.

Para finalizar con la evaluación bajo condiciones de tráfico para las cuales las fuentes VoIP dominen, en este apartado serán adoptadas simultáneamente las dos alternativas, referida la combinación de algoritmos como *Drop-Sel-defair.*

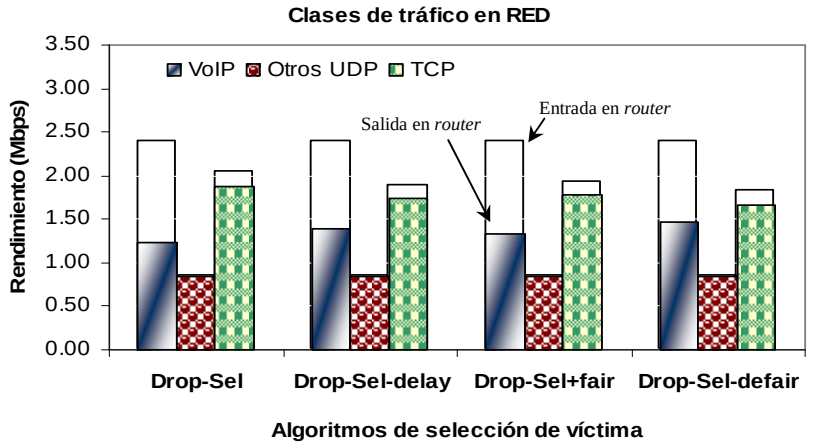
Para proporcionar un servicio todavía más equitativo y además evitar un desperdicio de recursos de la red, el algoritmo *Drop-Sel-defair* ejecuta un procedimiento de tres niveles:

- En el primer nivel se llevan a cabo las funciones generales de *Drop-Sel* de referencia, se determina cuál de las tres clases de tráfico ocupa mayor espacio en la cola. Si la clase *real-time* es la que domina, un segundo nivel es ejecutado.
- En el segundo nivel se llevan a cabo las tareas particulares de *Drop-Sel-delay*, observa los paquetes de la cola e identifica los paquetes de audio considerados no-útiles. Si no se localiza ningún paquete que cumpla con las especificaciones solicitadas (>300ms), un tercer nivel es ejecutado.
- Finalmente, en el tercer nivel se llevan a cabo las funciones particulares de *Drop-Sel+fair*, observa la tabla de información de flujos activos para identificar y seleccionar el flujo menos degradado.

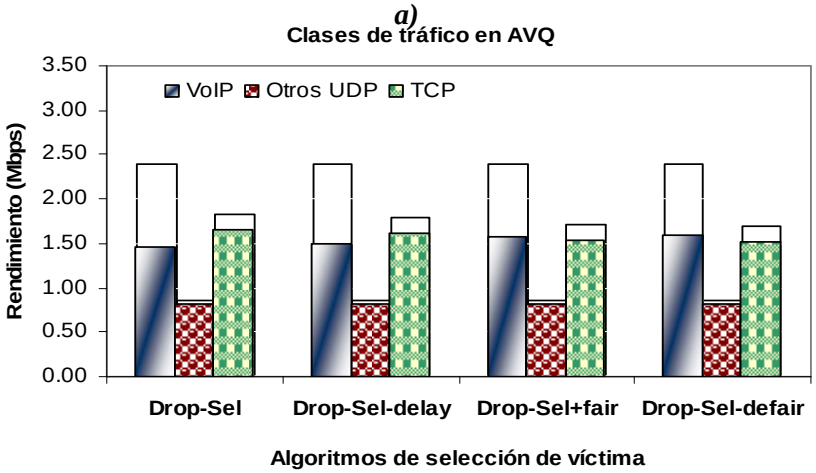
En este sentido se evalúa el funcionamiento de *Drop-Sel-defair* considerando el escenario S2 y específicamente el caso 6.

Equidad entre clases de tráfico

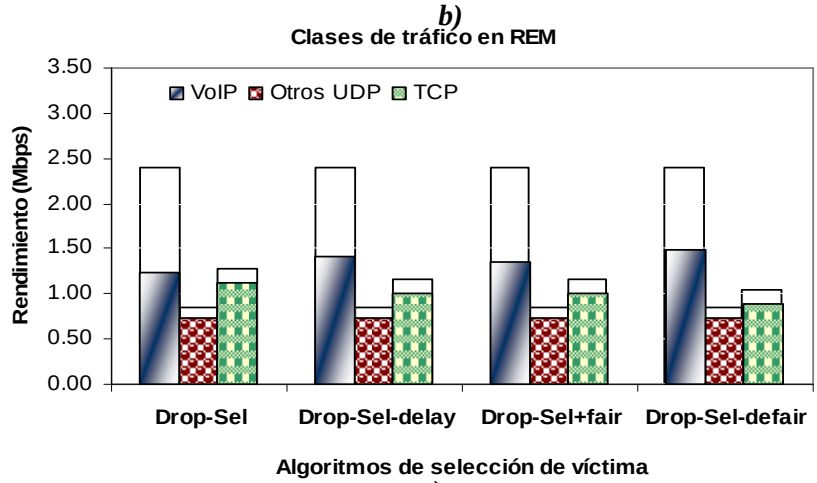
Primeramente, como se ha hecho en evaluaciones de apartados previos, se analizará el comportamiento equitativo de los algoritmos. En la Figura 6.36 se muestra el rendimiento medio de entrada y de salida para las distintas clases de tráfico en los distintos esquemas AQM.



a)



b)



c)

Figura 6.36 Evaluación del rendimiento medio de diferentes clases de tráfico bajo *Drop-Sel* y sus alternativas en el caso 6.

Para propósitos de comparación, en los resultados expuestos se considera al esquema *Drop-Sel* de referencia y a ambas alternativas en forma independiente.

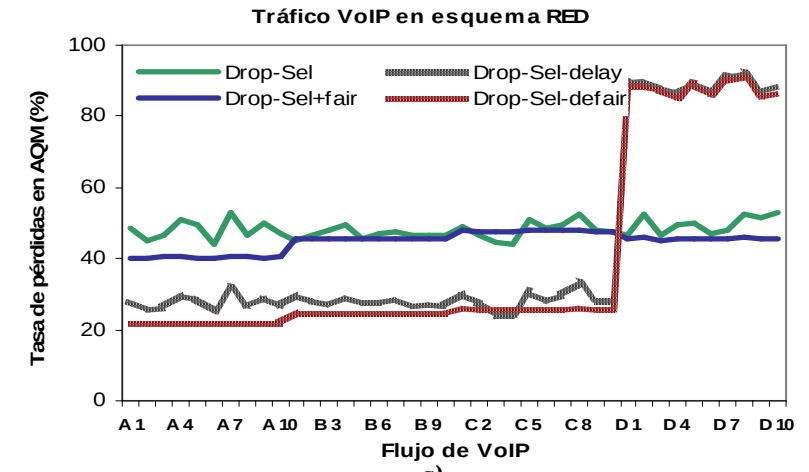
Se observa que la adopción simultánea de los dos algoritmos no impide ser imparcial para los tres tipos de tráficos, independientemente de la naturaleza reactiva de las fuentes y del esquema AQM utilizado. Más aún, en RED (Figura 6.36a) y en AVQ (Figura 6.36b) se logra ser más equitativo y se observa que las fuentes VoIP logran equiparar el consumo de ancho de banda de los flujos TCP mientras que los otros flujos UDP obtienen el ancho de banda requerido. Sin embargo, la combinación de alternativas no logra proporcionar una asignación tan equitativa en el esquema REM, *Drop-Sel-defair* favorece más al tráfico VoIP. En este caso en particular para REM (Figura 6.36c), *Drop-Sel* de referencia (sin utilizar alguna alternativa) proporciona una mejor asignación de recursos a las distintas clases de tráfico.

Tasa de pérdidas

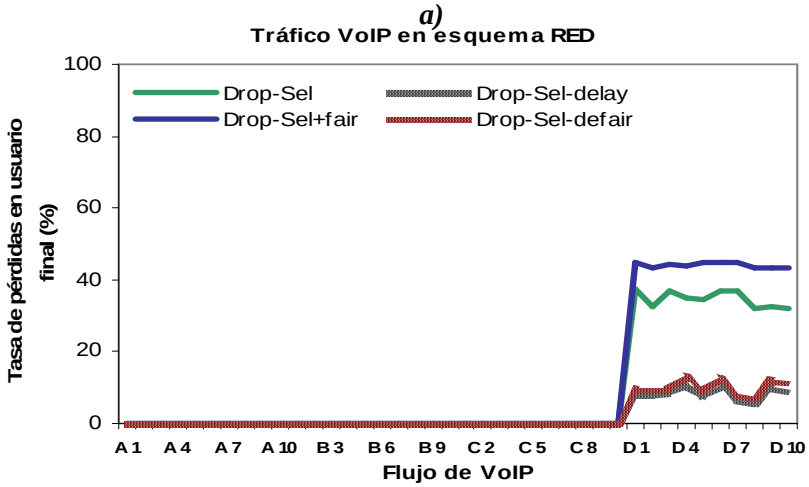
Al evaluar el impacto de *Drop-Sel-defair* en términos de probabilidad de pérdidas en el *router* (Figuras 6.37, 6.38 y 6.39), se observa que el algoritmo está detectando y descartando anticipadamente los paquetes de audio considerados no-aprovechables (de flujos tipo D) al mismo tiempo que asigna más equitativamente los recursos entre los flujos menos afectados por el retardo extremo-a-extremo (tipos A, B y C), independientemente del tipo de aplicación y el esquema utilizado.

Los experimentos ejecutados han mostrado que *Drop-Sel-defair* tiende a proporcionar beneficios mejores a los flujos VoIP en comparación a utilizar las alternativas en forma independiente. Por ejemplo, para RED en la Figura 6.37c, se reduce un 18.91% la tasa total de pérdidas de paquetes de voz en el receptor respecto a *Drop-Sel* de referencia y un 4.27% en relación a *Drop-Sel-delay*. Además se tiende a ser más equitativo entre las tasas de distintos tipos de aplicación. Específicamente, para flujos A, B y C proporciona tasas de pérdidas en un rango desde 21.70% a 26.01% (diferencia de 4.31%) mientras que *Drop-Sel+fair* desde 40.00% a 48.04% (diferencia de 8.04%).

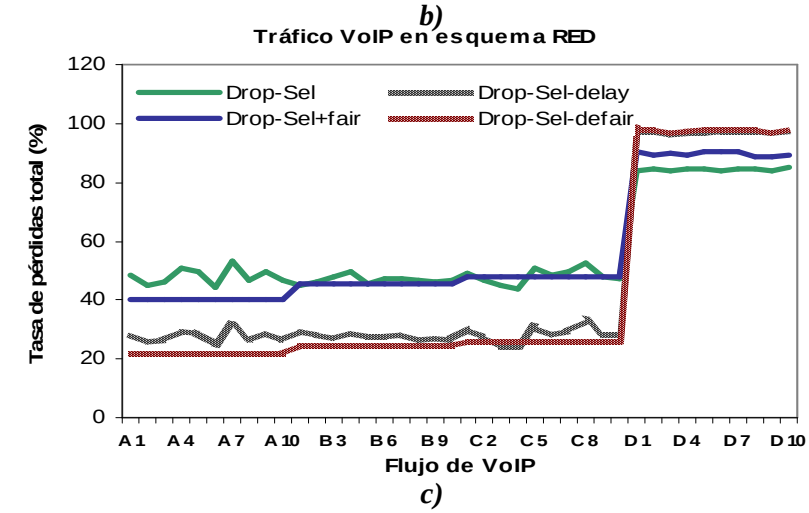
Similarmente, para REM (Figura 6.39c) se observa que *Drop-Sel-defair* consigue reducir las pérdidas totales de paquetes de VoIP. Respectivamente se reduce un 19.55% y un 3.87% en relación a *Drop-Sel* de referencia y *Drop-Sel-delay*. Además, mantiene un mejor comportamiento equitativo entre los flujos de VoIP. Para flujos A, B y C establece tasas de pérdidas en un rango desde 20.72% a 25.07% (diferencia de 4.35%) mientras que *Drop-Sel +fair* desde 38.73% a 46.56% (diferencia de 7.83%).



a)

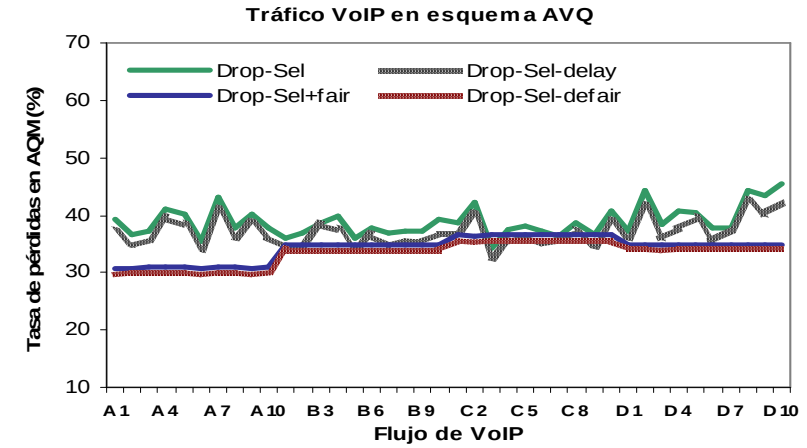


b)

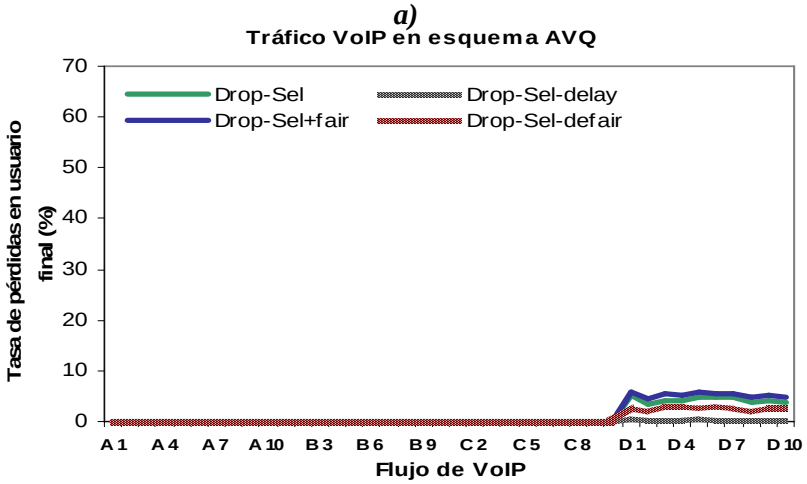


c)

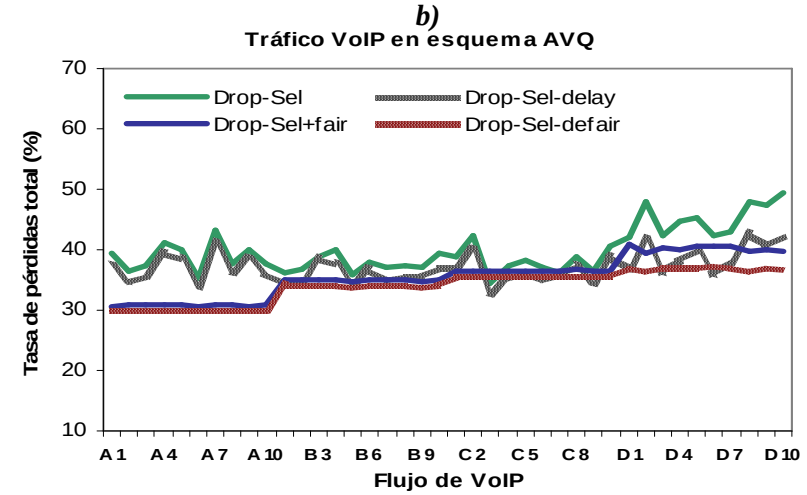
Figura 6.37. Evaluación por flujo bajo *Drop-Sel* y sus alternativas en el caso 6 con RED.



a)

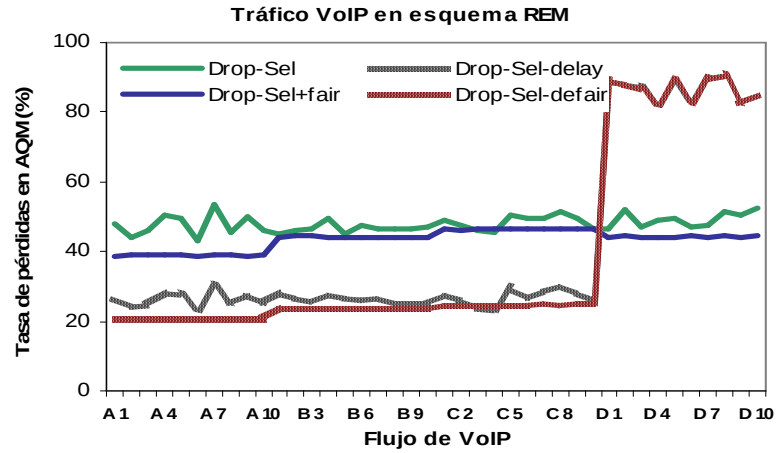


b)

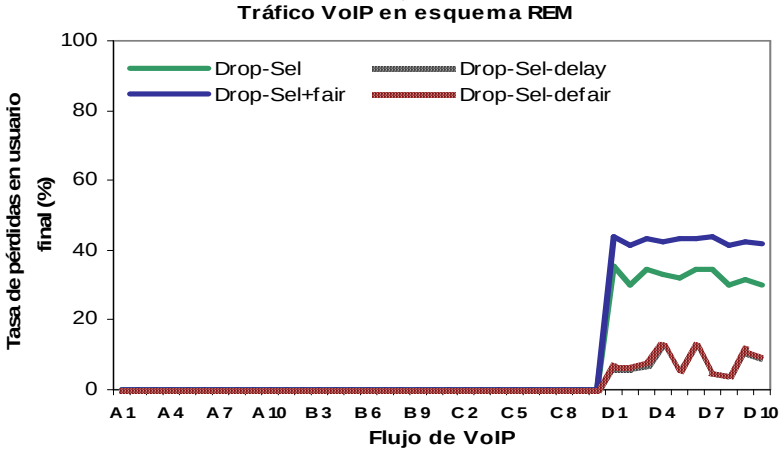


c)

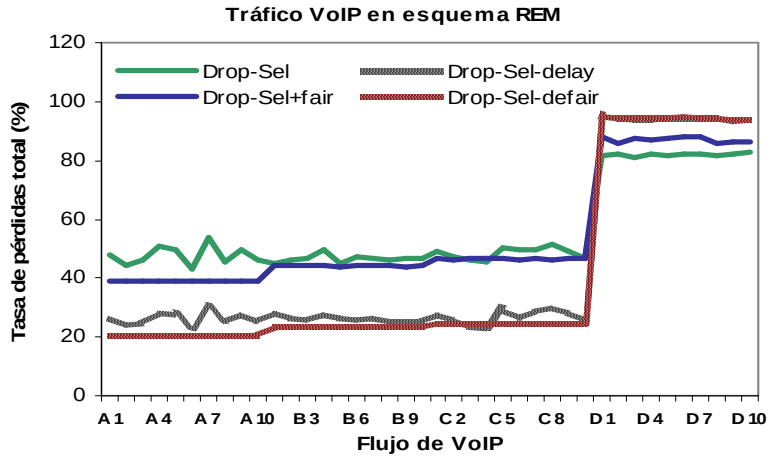
Figura 6.38. Evaluación por flujo bajo *Drop-Sel* y sus alternativas en el caso 6 con AVQ.



a)

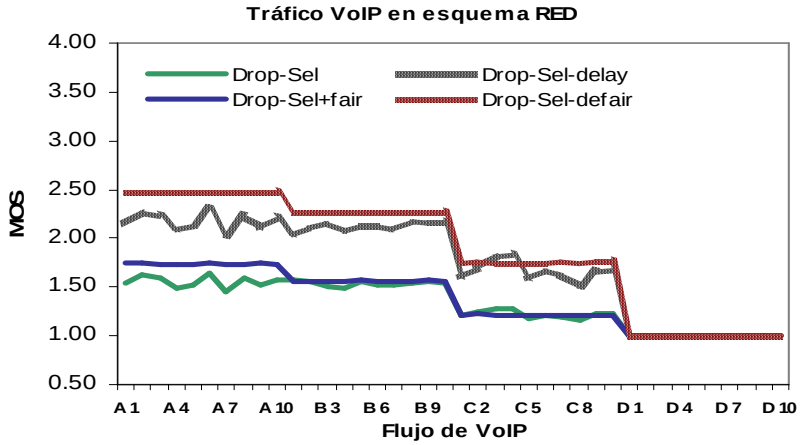


b)

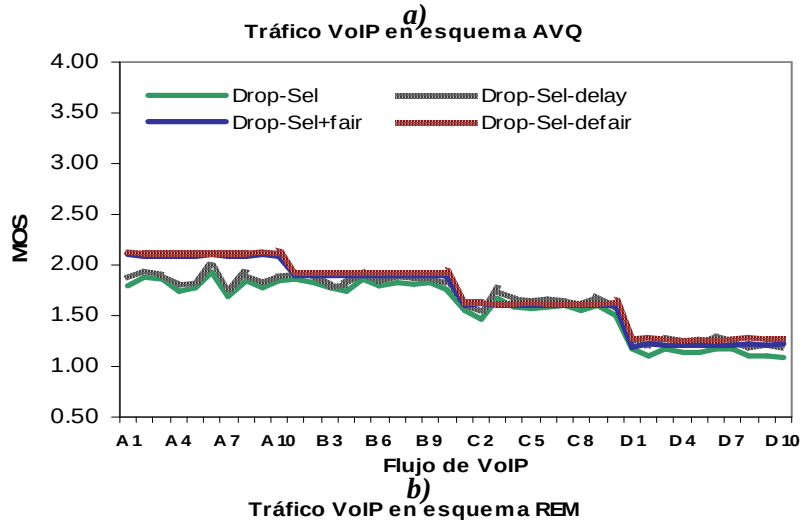


c)

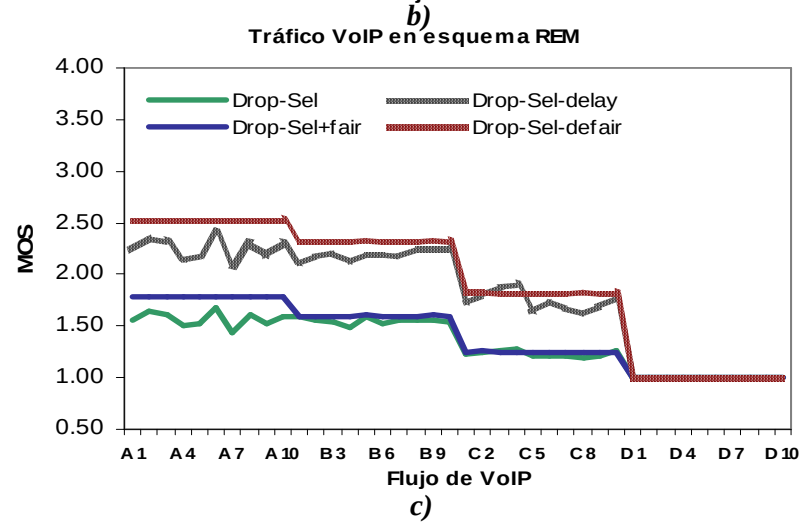
Figura 6.39. Evaluación por flujo bajo *Drop-Sel* y sus alternativas en el caso 6 con REM.



a)



b)



c)

Figura 6.40. Evaluación MOS de *Drop-Sel* y sus variantes en el caso 6.

MOS

La Figura 6.40 muestra el rendimiento en términos de calidad percibida por el usuario final obtenido para los diferentes flujos, utilizando el método de descarte selectivo *Drop-Sel* y sus alternativas.

Se puede observar que *Drop-Sel-defair* al otorgar equitativamente los recursos entre los flujos de VoIP, al igual que *Drop-Sel+fair*, proporciona a los distintos flujos una calidad similar según el tipo de aplicación (A, B, C ó D), independientemente del esquema AQM utilizado. A diferencia de *Drop-Sel+fair*, *Drop-Sel-defair* reduce considerablemente la tasa de pérdidas para los flujos menos afectados por el retardo extremo-a-extremo, experimentando por ende valores MOS para los flujos A, B y C mayores que los que proporciona éste. Además, se observa que la adopción simultánea de los dos algoritmos proporciona una calidad mejor a la suministrada por *Drop-Sel* de referencia y *Drop-Sel-delay*.

Para finalizar la evaluación MOS en la Figura 6.41 se han establecido los resultados en términos generales de *Drop-Sel* de referencia y sus alternativas. *Drop-Sel-defair* nuevamente es la opción que proporciona la mejor calidad comparado con los distintos esquemas de descarte selectivo, independientemente del esquema AQM utilizado.

Los resultados obtenidos en términos de calidad subjetiva en combinación con los previos corroboran que si el procedimiento de descarte selectivo *Drop-Sel* considera simultáneamente la utilidad del paquete de audio encolado y el número de paquetes descartados por flujo, se obtiene una mayor calidad para el tráfico VoIP bajo condiciones

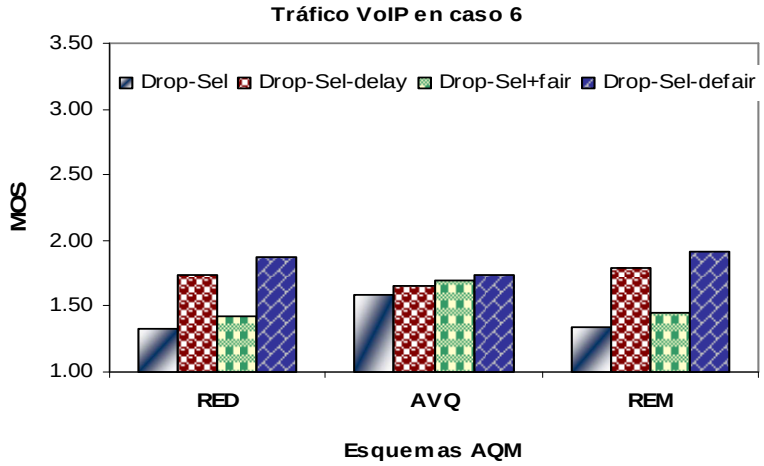


Figura 6.41. Evaluación general MOS de *Drop-Sel* y sus variantes en el caso 6.

de tráfico para las cuales estas fuentes dominen, en particular en los esquemas RED y REM.

6.4. Conclusiones

En este capítulo se han analizado y evaluado una serie de alternativas de descarte que intentan mantener una satisfactoria calidad de servicio a los flujos de VoIP, independientemente de las condiciones de carga existentes.

Los experimentos ejecutados han mostrado que un algoritmo más efectivo de selección de victima (para flujos multimedia y en particular de audio) puede ser concebido. En particular, se presentan las alternativas referidas como *Drop-Sel-delay, Drop-Tail-delay, Drop-Sel+fair* y *Drop-Tail+fair*.

Concretamente, se observa que si el algoritmo *Drop-Sel* considera simultáneamente la utilidad del paquete de audio encolado (*Drop-Sel-delay o Drop-Tail-delay* si adopta dinámicamente la estrategia de selección de víctima) y el número de paquetes descartados por flujo (*Drop-Sel+fair o Drop-Tail+fair),* cuando intervienen conexiones con largos retardos de propagación, un mayor beneficio en términos de QoS y QoE puede obtenerse para el tráfico VoIP bajo condiciones de tráfico para las cuales estas fuentes dominen.

Tabla 6.5 sintetiza las alternativas de descarte selectivo propuestas en esta investigación y las caracteriza identificando su objetivo, funcionamiento y condiciones de efectividad.

Tabla 6.5.
ALGORITMOS DE SELECCIÓN DE VÍCTIMA

Alternativa	Objetivo y funcionamiento	Condiciones de adaptación
Drop-Sel de referencia	**Objetivo.** Reducir en el AQM el efecto de falta de equidad cuando concurre tráfico reactivo y no-reactivo en una cola AQM, sin necesidad de usar un *scheduler* con múltiples colas. **Funcionamiento**. Para proporcionar un servicio equitativo, impone al paquete una probabilidad de ser seleccionado como víctima dependiendo de la ocupación en *bytes* en la cola por cada tipo de tráfico.	Utiliza un contador por clase de tráfico.
Drop-Sel-delay *y* ***Drop-Tail-delay****	**Objetivo.** Evitar desperdicio de recursos de memoria y ancho de banda al mismo tiempo que incrementa la QoS esperada para el tráfico VoIP en el receptor. **Funcionamiento.** Respectivamente llevan a cabo las funciones propias de *Drop-Sel* de referencia y *Drop-Tail.* Además, identifican paquetes de audio considerados no-útiles en la cola, extrayéndolos para proteger a los paquetes útiles de descartes innecesarios.	A diferencia de *Drop-Sel* y *Drop-Tail:* ▪ La fuente, el *router* y el destino deben estar sincronizados. ▪ Se requiere insertar información de señalización extra dentro de la cabecera de los paquetes. Utilización de cabeceras RTP (en fuente) e informes RTCP (en destino).
Drop-Sel*+fair* *y* ***Drop-Tail****+fair**	**Objetivo.** Proveer un servicio que garantice equidad entre diversos flujos de tráfico VoIP que utilizan diferentes *codecs* o diferentes tasas de envío. **Funcionamiento.** Respectivamente llevan a cabo las funciones propias de *Drop-Sel* de referencia y *Drop-Tail.* Además, identifican paquetes de audio del flujo menos degradado.	A diferencia de *Drop-Sel* y *Drop-Tail:* ▪ Utilizan adicionalmente un esquema por-flujo. Mantienen una tabla con información principal acerca de cada uno de los flujos activos.

* Esta alternativa es utilizada cuando se adopta dinámicamente la estrategia de selección de víctima.

Capítulo 7

Conclusiones y resultados

Las redes intercomunicadas basadas en el protocolo IP están suministrando una plétora de diferentes aplicaciones en el usuario final. En este contexto, la tecnología convergente IP, es posiblemente la que controle mediano y largo plazo las futuras comunicaciones sobre Internet. Diferentes tipos de tráficos cada uno con distintos requerimientos deben compartir limitados recursos de la red. Desde el punto de vista extremo-a-extremo, algunas aplicaciones demandan intensamente control de congestión, mientras que otras son menos estrictas al respecto. Como resultado, básicamente dos protocolos de transporte son ampliamente utilizados. Actualmente, son los protocolos UDP y TCP, la alternativa no-reactiva y reactiva, respectivamente.

Para prevenir la congestión en la red, mecanismos de control de congestión incorporados a TCP han sido ampliamente estudiados. Directamente relacionado, la gestión activa de colas (AQM) ha sido un campo muy activo al que la comunidad científica le ha dedicado grandes esfuerzos, resultando en una serie de sistemas muy interesantes. En términos generales, diferentes tipos de tráficos comparten la salida de la cola AQM, si bien por lo general estos son procesados indistintamente. En este trabajo de investigación se ha considerado cómo mejorar el funcionamiento de la red, en términos de medidas objetivas y subjetivas, por el sólo hecho de observar y seleccionar el tráfico para prevenir la congestión en los dispositivos AQM.

Con este objetivo, la principal contribución de esta tesis es un esquema sencillo para la selección de la víctima en los esquemas AQM. El método propuesto considera las demandas de control de congestión de la aplicación. En particular, se establece un

procedimiento sencillo para seleccionar el paquete a ser descartado. El algoritmo propuesto intenta detectar el tráfico que causa la congestión, y la clase de tráfico identificada es penalizada en consecuencia. Entre los esquemas simulados, ha sido experimentalmente comprobado que el algoritmo proporciona un mejor grado de equidad entre clases de tráfico y una mejor QoS extremo-a-extremo para las condiciones en las que las fuentes sin mecanismos de control (no reactivas) no dominen.

El esquema propuesto referido como *Drop-Sel*, es evaluado sobre un conjunto de esquemas AQM relevantes: RED, REM y AVQ. Algunos escenarios simulados con diferentes cargas de tráfico son considerados, en los que flujos de tráfico UDP (flujos VoIP), otros flujos UDP sin requisitos de tiempo real y TCP son generados. En las simulaciones, diferentes fuentes de tráficos y destinos son incluidos en diversas localizaciones para proporcionar un amplio rango de posibilidades. Los experimentos ejecutados han mostrado que *Drop-Sel* logra una significante mejora en la equidad entre diferentes clases de tráfico en todos los esquemas AQM y para las condiciones en las que las fuentes VoIP no dominen, una mejora en el funcionamiento a nivel de red para los flujos de audio. Además, en general *Drop-Sel* también incrementa el rendimiento del tráfico TCP.

Para el tráfico Multimedia, y particularmente para VoIP, es evidente que mejoras obtenidas a nivel de red son insuficientes si no están correlacionadas con la satisfacción del usuario final (alto-nivel). Por consiguiente, para completar la evaluación se han medido los beneficios de adoptar el esquema propuesto, adoptando el modelo-E como soporte de la evaluación. Específicamente, se ha estimado el valor MOS bajo un amplio rango de condiciones experimentales. Adicionalmente, se muestra que *Drop-Sel*, el esquema propuesto de descarte selectivo incrementa la inteligibilidad final adoptando una metodología de evaluación utilizando un reconocedor automático de voz.

Para completar la investigación, también se han evaluado algunas alternativas del esquema propuesto. El procedimiento propuesto de descarte puede ser aún más efectivo al utilizar los recursos y más imparcial si considera una clasificación de grado más fino. En particular, para la clase de tráfico UDP con requisitos de tiempo-real, y para un dado flujo de VoIP en particular, la tasa de descarte depende fuertemente de dos diferentes aspectos, de la utilidad del paquete y del *codec* utilizado por la aplicación. Para hacer frente al desperdicio de consumo de memoria y ancho de banda por enviar paquetes no-útiles, se han considerado dos alternativas denominadas *Drop-Sel-delay* y *Drop-Tail-delay*. Éstas son respectivamente una extensión de los esquemas *Drop-Sel* y *Drop-Tail* que seleccionan el paquete VoIP a descartar basado en observar la utilidad o antigüedad de los paquetes encolados. En cambio, para hacer frente con la imparcialidad de flujos, se

han considerado dos alternativas denominadas *Drop-Sel+fair* y *Drop-Tail+fair*, éstas seleccionan el paquete VoIP a descartar considerando las tasas de descartes de los flujos en la cola.

Después de simulaciones realizadas, se ha evaluado experimentalmente los beneficios de estos procedimientos de selección de víctima. En estos casos, se ha mostrado la evaluación por flujo. En conclusión, se ha observado que con estas variantes del procedimiento de descarte, la equidad y QoS de los flujos VoIP son también mejorados.

Fruto de este trabajo de investigación es el artículo *Selective Packet Dropping for UDP and TCP Flows* remitido a *Telecommunication System (TS)*, y actualmente en proceso de revisión.

Bibliografía

[Allman *et al.*, 1999] M. Allman, V. Paxson and W. Steven. 1999. TCP Congestion Control. RFC 2581. http://www.ietf.org/rfc/rfc2581.txt.

[Almes *et al.*, 1999a] G. Almes, S. Kalidindi, and M. Zekauskas. 1999. A One-way Delay Metric for IPPM. RFC 2679. http://www.ietf.org/rfc/rfc2679.txt.

[Almes *et al.*, 1999b] G. Almes, S. Kalidindi, M. Zekauskas. 1999. A One-way Packet Loss Metric for IPPM. RFC 2680. http://www.ietf.org/rfc/rfc2680.txt.

[Almes *et al.*, 1999c] G. Almes, S. Kalidindi, M. Zekauskas. 1999. A Round-trip Delay Metric for IPPM. RFC 2681. http://www.ietf.org/rfc/rfc2681.txt.

[Anjum and Tassiulas, 1999] F. M. Anjum and L. Tassiulas. 1999. Fair Bandwidth Sharing among Adaptive and Non-Adaptive Flows in the Internet. *IEEE.* pp 1412- 1420.

[Aoul *et al.*, 2007] Y. H. Aoul, A. Mehaoua, C. Skianis. 2007. A fuzzy logic-based AQM for real-time traffic over internet. *Computer Networks* 51. pp 4617-4633.

[Athuraliya *et al.*, 2000] S. Athuraliya, V. H. Li, S. H. Low. 2000. REM: Active Queue Management.

[Aweya *et al.*, 2001] J. Aweya, M. Oullette, and D.Y. Montuno. 2001. A control theoretic approach to active queue management. *Proceedings of Computer Network 36.* pp 203-235.

[Blake, *et al.*, 1998] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. 1998. Architecture for Differentiated Services. RFC 2475. http://www.ietf.org/rfc/rfc2475.txt.

[Blanton *et al.*, 2003] E. Blanton, M. Allman, K. Fall, and L. Wang. 2003. A Conservative selective Acknowledgment (SACK) –based Loss Recovery Algorithm for TCP. RFC 3517. http://www.ietf.org/rfc/rfc3517.txt.

[Boutremans and Boudec, 2003] C. Boutremans and Le Boudec. 2003. Adaptive Joint Playout Buffer and FEC Adjustement for Internet Telephony. *IEEE INFOCOM 2003*.

[Braden *et al*., 1994] R. Braden, D. Clark and S. Shenker. 1994. Integrated Services in the Internet Architecture: an Overview. RFC 1633. http://www.ietf.org/rfc/rfc1633.txt.

[Braden *et al.*, 1997] R. Braden, L. Zhang, S. Berson, S. Herzog, and S.Jamin. 1997. Resource Reservation Protocol (RSVP). RFC 2205. http://www.ietf.org/rfc/rfc2205.txt.

[Braden *et al*., 1998] B. Braden, D. Clark, J. Crowcroft, B. Davie, B. Davie, D. Estrin, S. Floyd, V. Jacobson, G. Minshall, C. Partridge, L. Peterson, K. Ramakrishnan, S. Shenker, J. Wroclawski, and L. Zhang. 1998. Recommendations on Queue Management and Congestion Avoidance in the Internet. RFC 2309. http://www.ietf.org/rfc/rfc2309.txt.

[Brakmo and Peterson, 1995] L.S. Brakmo and L.L. Peterson. 1995. TCP Vegas: End to End Congestión Avoindance on a Global Internet. *IEEE Journal on Selected Areas in Communications*, Vol. 13. No.8, pp 1465- 1480.

[Carle and Biersack, 1997] G. Carle and E. W. 1997. Biersack. Survey of Error Recovery Techniques for IP based Audio-Visual Multicast Applications. *Institute EURECOM*.

[Cidon *et al*., 1993] I. Cidon, A. Khamisy, and M. Sidi. 1993. Analysis of Packet Loss Processes in High-Speed Networks. *IEEE Transactions on Information Theory*.

[Cole and Rocenbluth, 2001] R.G. Cole, and J.H. Rocenbluth. 2001. Voice over IP Performance Monitoring. *In Proceedings of SIGCOMM Computer Communication Vol. 31(2)*. pp 269-275.

[Chan and Hamdi, 2003] M.K. Chan and M. Hamdi. 2003. An Active Queue Management Scheme Based on a Capture-Recapture Model. *IEEE Journal on Selected Areas in Communications*. Vol. 21. pp. 572-583.

[Chan *et al*., 2005] S.P. Chan, C.W. K. and A. K. Wong. 2005. Multimedia Streaming Gateway with Jitter Detection. *IEEE Transactions on Multimedia, Vol. 7, No.3*. pp 585-592.

[Chung and Claypool, 2000] J. Chung and M. Claypool. 2000. Dynamic-CBT and ChIPS- Router Support for improved Multimedia Performance on the Internet. *In Proceedings of the ACM Multimedia Conference*.

[Davie *et al.*, 2002] B. Davie, A. Charny, J.C.R. Bennet, K. Benson, J.Y. Le Boudec, W. Courtney, S. Davari, V. Firoiu, and D. Stiliadis. 2002. An Expedited Forwarding PHB. RFC 3246. http://www.ietf.org/rfc/rfc3246.txt

[Demichelis and Chimento, 2002] C. Demichelis, P. Chimento. 2002. IP Packet Delay Variation Metric for IP Performance Metrics (IPPM). RFC 3393. http://www.ietf.org/rfc/rfc3393.txt.

[Feng *et al*., 2002] W.C. Feng, K. G. Shin, D. D. Kandlur and D. Saha. 2002. The BLUE Active Queue Management Algorithms. *IEEE/ACM Transactions on Networking*. Vol. 10, pp 513-528.

[Feng *et al*., 2004] G. Feng, A. Agawal, A. Jayaraman and C. Siew. 2004. Modified RED gateways under bursty traffic. *IEEE Communications Letters.* pp. 323-325.

[Floyd, 1994] S. Floyd. TCP and Explicit Congestion Notification. 1994. *ACM Computer Communication Review*, V. 24 N. 5, pages 10-23.

[Floyd, 2000] S. Floyd. 2000. Recommendation on using the "gentle_" variant of RED. http://www.icir.org/floyd/red/gentle.html.

[Floyd and Henderson, 1999] S. Floyd and T. Henderson. 1999. The New Reno Modification to TCP's Fast Recovery Algorithm. RFC 2582. http://www.ietf.org/rfc/rfc2582.txt

[Floyd and Jacobson, 1993] S. Floyd and V. Jacobson. 1993. Random Early Detection Gateways for Congestion Avoidance. *IEEE/ACM Transactions on Networking*.

[Floyd and Kohler, 2006a] S. Floyd and E. Kohler. 2006. Profile for Datagram Congestion Control Protocol (DCCP). Congestion Control ID 2: TCP-like Congestion Control. RFC 4341. http://www.ietf.org/rfc/rfc4341.txt.

[Floyd and Kohler, 2006b] S. Floyd and E. Kohler. 2006. Profile for Datagram Congestion Control Protocol (DCCP). Congestion Control ID 3: TCP-Friendly Rate Control (TFRC). RFC 4342. http://www.ietf.org/rfc/rfc4342.txt.

[Floyd *et al*.] Vat - LBNL Audio Conferencing Tool. http://www-nrg.ee.lbl.gov/vat/.

[Floyd *et al*., 2000] S. Floyd, J. Mahdavi, M. Mathis, and M. Podolsky. 2000. An Extension to the Selective Acknowledgment (SACK) Option for TCP. RFC 2883. http://www.ietf.org/rfc/rfc2883.txt.

[Floyd *et al.*, 2001] S. Floyd, R. Gummadi, and S. Shenker. 2001. Adaptive RED: An Algorithm for Increasing the Robustness of RED's Active Queue Management. *AT&T Center for Internet Research at ICSI*. http://www.icir.org/floyd/papers/adaptiveRed.pdf.

[Floyd *et al.*, 2004] S. Floyd, T. Henderson, and A. Gurtov. 2004. The New Reno Modification to TCP's Fast Recovery Algorithm. RFC 3782. http://www.ietf.org/rfc/rfc3782.txt.

[Fomenkov *et al*., 2004] M. Fomenkov, K. Keys, D. Moore and K. Claffy. 2004. Longitudinal study of Internet Traffic in 1998-2003. *In Proceedings ACM International Conference*.

[Gao and Hou, 2003] Y. Gao and J. C. Hou. 2003. A State Feedback Control Approach to Stabilizing Queues for ECN-Enabled TCP Connections. *IEEE INFOCOM 2003.*

[Gao *et al*, 2002] Y. Gao, G. He and J. C. Hon. . 2002. On Exploiting traffic Predictability in Active Queue Management. *IEEE INFOCOM 2002*. pp 1630-1639.

[Goodman, *et al.*, 1986] D. J. Goodman, O. G. Lockart, and W. C. Wong. 1986. Waveform Substitution Techniques for Recovering Missing Speech Segments in Packet Voice Communications. *IEEE Trans. Acoustics, Speech and Sig. Processing*. pp 105-108.

[Gozdecki, *et al.*, 2003] J. Gozdecki, A. Jajszczyk and R. Stankiewicz. 2003. Quality of Service Terminology in IP Networks. *IEEE Communications Magazine*, pp. 153-159.

[Handley *et al.*, 2003] M. Handley, S. Floyd, J. Padhye, J. Widmer. 2003. TCP Friendly Rate Control (TFRC): Protocol Specification. RFC 3448. http://www.ietf.org/rfc/rfc3448.txt.

[Hardman *et al*., 1995] V. Hardman, M. A. Sasse, M. Handley and A. Watson. 1995. Reliable Audio for Use Over the Internet. *In Proceedings INET*.

[Hashem, 1989] E. Hashem. Analysis of Random drop for gateway congestion control. 1989. Report MIT-LCS-TR-465. Laboratory for Computer Science, Cambridge.

[Heinanen *et al.*, 1999] J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski. 1999. Assured Forwarding PHB Group. RFC 2597. http://www.ietf.org/rfc/rfc2597.txt.

[Hirsch and Pearce. 2000] H.G. Hirsch and D. Pearce. 2000. The AURORA experimental framework for the performance evaluation of speech recognition systems under noisy conditions. *In Proceedings of ISCA ITRW ASR: Automatic Speech Recognition: Challenges for the Next Millenium.* pp. 181-188.

[Ho *et al*., 2008] C.Y. Ho, Y.C. Chen, Y.C. Chan, and C.Y. Ho. 2008. Fast retransmit and fast recovery schemes of transport protocols: A survey and taxonomy. *Computer Networks 52*, pp. 1308-1327.

[Iacovoni *et al.*, 2002] G. Iacovoni, S. Morsa, D. Parisi and D. Pierotti. 2002. Broadband FRA scenario and traffic modeling. *Information Society Technologies*.

[Postel, 1980] Postel, J. 1980. User Datagram Protocol. RFC 768. http://www.ietf.org/rfc/rfc0768.txt.

[Postel, 1981a] Postel, J. 1981. Internet Protocol. RFC 791. http://www.ietf.org/rfc/rfc0791.txt.

[Postel, 1981b] Postel, J. 1981. Transmission Control Protocol. RFC 793. http://www.ietf.org/rfc/rfc0793.txt.

[ICQ, 2009] ICQ v6.5. 2009. http://www.icq.com/.

[ITU-T, 1988] International Telecommunication Union Standardization. 1988. Pulse code modulation (PCM) of voice frequencies. Recommendation G.711. http://www.itu.int/rec/T-REC-G.711-198811-I/en.

[ITU-T, 1993] International Telecommunication Union Standardization. 1993. Echo cancellers. Recommendation G.165. http://www.itu.int/rec/T-REC-G.165-199303-I/en.

[ITU-T, 1994] International Telecommunication Union Standardization. 1994. Terms and definitions related to quality of service and network performance including dependability. Recommendation E.800. http://www.itu.int/rec/T-REC-E.800-199408-S/en.

[ITU-T, 1996] International Telecommunication Union Standardization. 1996. Methods for subjective determination of transmission quality. Recommendation P.800. http://www.itu.int/rec/T-REC-P.800-199608-I/en.

[ITU-T, 1998] International Telecommunication Union Standardization. 1998. Subjective audiovisual quality assessment methods for multimedia applications. Recommendation P.911. http://www.itu.int/rec/T-REC-P.911-199812-I/en.

[ITU-T, 2000] International Telecommunication Union Standardization. 2000. Interactive test methods for audiovisual communications. Recommendation P.920. http://www.itu.int/rec/T-REC-P.920-200005-I/en.

[ITU-T, 2001a] International Telecommunication Union Standardization. 2001. End-user multimedia QoS categories. Recommendation G.1010. http://www.itu.int/rec/T-REC-G.1010-200111-I/en.

[ITU-T, 2001b] International Telecommunication Union Standardization. 2001. Perceptual evaluation of speech quality (PESQ): An objetive method for end-to-end speech quality assessment of narrow-band telephone networks and speech codecs. Recommendation P.862. http://www.itu.int/rec/T-REC-P.862-200102-I/en.

[ITU-T, 2004] International Telecommunication Union Standardization. 2004. Single-ended method for objective speech quality assessment in narrow-band telephony applications. Recommendation P.563. http://www.itu.int/rec/T-REC-P.563-200405-I/en

[ITU-T, 2005] International Telecommunication Union Standardization. 2005. The E-Model, a computational model for use in transmission planning. Recommendation G.107. http://www.itu.int/rec/T-REC-G.107-200503-S/en.

[ITU-T, 2007] International Telecommunication Union Standardization. 2007. Conformance testing for voice over IP transmission quality assessment models. Recommendation P.564. http://www.itu.int/rec/T-REC-P.564-200711-I/en.

[Jacobson, 1988] V. Jacobson. 1988. Congestion Avoindance and Control.

[Jain *et al*., 1988] R. Jain, K.K. Ramakrishnan and D. Chiu. 1988. Congestion avoindance in computer networks with a connectionless network layer. Proceedings of SIGCOMM'88.

[Jiang and Schulzrinne, 2002] W. Jiang and H. Schulzrinne. 2002. Speech recognition performance as an effective perceived quality predictor. Tenth IEEE International Workshop on Quality of Service (2002). pp. 269-275.

[Jin, *et al*., 2004] C. Jin, D. X. Wei and S. H. Low. 2004. FAST TCP: Motivation, Architecture, Algorithms, Performance. IEEE INFOCOM (2004)

[Joo *et al.*, 2003] C. Joo, S. Bahk, S.S. Lumetta 2003. Hybrid Active Queue Management, *in: Proceedings of ISCC 2003 Conference.* pp. 999-1004.

[Kelly, 2003] T. Kelly. 2003. Scalable TCP: Improving Performance in Highspeed Wide Area Networks. CERN and Laboratory for Communication Engineering, University of Cambridge.

[Kim *et al.*, 2004] B. Kim, D. Kim, and J.Lee. 2004. Lost Retransmission Detection for TCP SACK. *IEEE Communications Letter*, Vol 8. No.9. pp. 600-602.

[Kim and Lee, 2006] T. Kim and K. Lee. 2006. Refined Adaptive RED in TCP/IP Networks. *SICE-ICASE International Joint Conference.*

[King, *et al.*, 2005] R. King, R. Baraniuk and R. Riedi. 2005. TCP-Africa: An Adaptive and Fair Rapid Increase Rule for Scalable TCP.

[Kohler *et al.*, 2006] E. Kohler, M. Handley and S. Floyd. 2006. Datagram Congestion Control Protocol (DCCP). RFC 4340. http://www.ietf.org/rfc/rfc4340.txt.

[Kostas *et al*., 1998] T. J. Kostas, M. S. Borella, I. Sidhu, G. M. Schuster, J. Grabiec, and J. Machler. 1998. Real-Time Voice over Packet Switched Networks. *IEEE Network*, Vol. 12, pp. 18-27.

[Kunniyur and Srikant, 2001] S. Kunniyur, and R. Srikant. 2001. Analysis and Design of an Adaptive Virtual Queue (AVQ) Algorithm for Active Queue Management. *SIGCOMM'01*. pp 123-134.

[Kunniyur and Srikant, 2004] S. Kunniyur, and R. Srikant. 2004. An Adaptive Virtual Queue (AVQ) Algorithm for Active Queue Management. IEEE/ ACM Transactions on Networking.

[Lakshman *et al*., 1996] T.V. Lakshman, A. Neidhardt and T. J. Ott. 1996. The Drop from Front Strategy in TCP and in TCP over ATM. *IEEE*. pp 1242-1250.

[Le *et al*., 2005] L. Le, J. Aikat, K. Jeffay and F. D. Smith. 2005. The Effects of Active Queue Management and Explicit Congestion Notification on Web Performance. *IEEE/ACM Transactions on Networking*,

[Liao *et al.*, 2001] W.T. Liao, J.C. Chen, and M.S. Chen. 2001. Adaptive Recovery Techniques for Real-Time Audio Streams. *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies.*

[Liang, *et al.*, 2002] Y. J. Liang, E. G. Steinbach, and B. Girod. 2002. Real-time Voice Communication over the Internet Using Packet Path Diversity. *Information Systems Laboratory, Department of Electrical Engineering, Stanford University*.

[Lin and Morris, 1997] D. Lin and R. Morris. 1997. Dynamics of Random Early Detection. *In Proceedings of SIGCOMM'97.*

[Liu *et al.*, 2005] S.G. Liu, P.J. Wang and L.J. Qu. 2005. Modeling and Simulation of self similar data traffic. *In Proceedings of the Fourth International Conference on Machine Learning and Cybernetic*. pp. 3921-3925.

[Liu *et al.*, 2006] S.Liu, T. Basar and R. Srikant. 2006. TCP-Illinois: A Loss and Delay-Based Congestin Control Algorithm for High-Speed Networks. *In Proceedings of First International Conference on Performance Evaluation Methodologies and Tools*.

[Long *et al*., 2005a] C. Long, B. Zhao and X. Guan. 2005. SAVQ: Stabilized Adaptive Virtual Queue Management Algoritm. *IEEE Communications Letters*, Vol. 9. No.1, pp. 78-80.

[Long *et al*., 2005b] C. Long. B. Zhao, X. Guan, and J. Yang. 2005. The Yellow active queue management algorithm. *Computer Networks 47*. pp 525-550.

[Malkin, 1998] G. Malkin. 1998. RIP Version 2. RFC 2453. http://www.ietf.org/rfc/rfc2453.txt.

[Mathis *et al*., 1996] M. Mathis, J. Mahdavi, S. Floyd and A. Romanow. 1996. TCP Selective Acknowledgment Options. RFC 2018. http://www.ietf.org/rfc/rfc2018.txt.

[May *et al*., 1999] M. May, J. Bolot, C. Diot and B. Lyles. 1999. Reasons not to deploy RED. *In Proceedings of International Workshop on Quality of Service.* pp. 260-262.

[Mena and Heidemann, 2000] Art Mena and John Heidemann. 2000. An Empirical Study of Real Audio Traffic. *IEEE INFOCOM*, pages 102-110.

[Microsoft, 2009] Messenger 9.0. 2009. http://www.msn-messenger-9.com.

[Mills, 1984] D.L. Mills. 1984. Exterior Gateway Protocol Formal Specification. RFC 904. http://www.ietf.org/rfc/rfc0904.txt.

[Moon *et al.*, 1998] S. B. Moon, J. Kurose, and D. Towsley. 1998. Packet Audio Playout Delay Adjustment: Performance Bounds and Algorithms. *ACM/Springer Multimedia Systems.*

[Moy, 1998] J. Moy. 1998. OSPF Version 2. RFC 2328. http://www.ietf.org/rfc/rfc2328.txt.

[Nichols *et al.*, 1998] K. Nichols, S. Blake, F. Baker, and D. Black. 1998. Definition of the Differentiated Services Field (DS Field)in the IPv4 and IPv6 Headers. RFC 2474. http://www.ietf.org/rfc/rfc2474.txt.

[Ns2, 2008] Ns2, Network Simulator. http:// www.isi.edu/nsnam/ns/.

[Ott *et al.*, 1999] T. J. Ott, T.V. Lakshman and L. H. Wong. 1999. SRED: Stabilized RED. *IEEE*, pp. 1346-1355.

[Pan *et al*., 1999] R. Pan, B. Prabhakar, and K. Psounis. 1999. CHOKe: A stateless active queue management scheme for approximating fair bandwidth allocation. *Department of Electrical Engineering, Stanford University.*

[Parris *et al*., 1999] M. Parris, K. Jeffay and F.D. Smith. 1999. Lightweight active router-queue management for multimedia networking. *In Proceedings of Conference on Multimedia Computing and Networking*.

[Paxson *et al.*, 1998] V. Paxson, G. Almes, J. Mahdavi, and M. Mathis. 1998. Framework for IP Performance Metrics. RFC 2330. http://www.ietf.org/rfc/rfc2330.txt.

[Perkins, 1999] C. Perkins. 1999. RTP Payload Format for Interleaved Media. *Internet Draft, IETF Audio/Video Transport Working Group.*

[Perkins *et al*., 1998] C. Perkins, O. Hodson, and V. Hardman. 1998. A Survey of Packet Loss Recovery Techniques for Streaming Audio. *IEEE Network*. pp 40-48.

[Phirke *et al.* 2002] V. Phirke, M. Claypool and R. Kinicki. 2002. Traffic Sensitive Active Queue Management for Improved Multimedia Streaming. *Technical Report, Computer Science, Worcester Polytechnic Institute.*

[Raisanen, 2002] V. Raisanen, G. Grotefeld, A. Morton. 2002. Network performance measurement with periodic streams. RFC 3432. http://www.ietf.org/rfc/rfc3432.txt.

[Ramakrishnan and Floyd, 1999] K. Ramakrishnan and S. Floyd. 1999. A Proposal to add Explicit Congestion Notification (ECN) to IP. RFC 2481. http://www.ietf.org/rfc/rfc2481.txt.

[Ramos *et al.*, 2005] J. J. Ramos Muñoz, J. M. Lopez Soler, and A. M. Gomez. 2005. Intelligibility evaluation of a VoIP Multi-flow Block Interleaver. *In Proceedings of IFIP IWAN (2005), Sophia-Antipolis France*.

[Ramos, 2009] J. J. Ramos Muñoz. 2009. Mejoras a la transmisión de voz sobre IP considerando criterios de calidad experimentada. Selección automática de protocolos. Universidad de Granada Tesis doctoral.

[Realnetworks, 2008] RealVídeo 10. 2008. http://www.realnetworks.com/products/codecs/realvideo.html.

[Reguera *et al*., 2008] V.A. Reguera, F.F. Alvarez Paliza, W. Godoy y E.M.García Fernández. 2008. On the impact of active queue management on VoIP quality of service. *Computer Communications 31)*, pp. 73-87.

[Rekhter et al., 2006] Y. Rekhter, T. Li, and S. Hares. 2006. A Border Gateway Protocol 4 (BGP-4). RFC 4271. http://www.ietf.org/rfc/rfc4271.txt.

[Rutgers, 1991] C.L. Hedrick Rutgers. An Introduction to IGRP. 1991. White Paper, Laboratory for Computer Science Research. http://www.cisco.com/warp/public/103/5.html.

[Seol *et al*., 2006] J.H. Seol, K. Y. Lee and Y. S. Hong. 2006. Performance Improvement of Adaptive AQM using the Variation of Queue Length. *IEEE.*

[Schulzrinne *et al.*, 2003] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. 2003. RTP: A Transport Protocol for Real-Time Applications. RFC 3550. http://www.ietf.org/rfc/rfc3550.txt.

[Schulzrinne and Casner, 2003] H. Schulzrinne and S. Casner. 2003. RTP Profile for Audio and Video Conferences with Minimal Control. RFC 3551. http://www.ietf.org/rfc/rfc3551.txt.

[Shenker and Wroclawski, 1997] S. Shenker, and J. Wroclawski. 1997. Network Element Service Specification Template. RFC 2216. http://www.ietf.org/rfc/rfc2216.txt.

[Shenker *et al.*, 1997] S. Shenker, C. Partridge and R. Guerin. 1997. Specification of Guaranteed Quality of Service. RFC 2212. http://www.ietf.org/rfc/rfc2212.txt.

[Sipforum, 2006] Speak Conference Director. 2006. http://www.sipforum.org/content/view/258/170/.

[Skype, 2009] Skype 2009. http://www.skype.com.

[Stevens, 1997] W. Stevens. 1997. TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms. RFC 2001. http://www.ietf.org/rfc/rfc2001.txt.

[Sun and Zukerman, 2007] J. Sun and M. Zukerman. 2007. RaQ: A robust active queue management scheme based on rate and queue length. *Computer Communications 30*, pp. 1731-1741.

[Sun *et al*., 2003a] J. Sun, K.T. Ko, G. Chen, S. Chan, and M. Zukerman. 2003. PD-RED: To Improve the Performance of RED. *IEEE Communications Letters*. Vol 7, no.8, pp. 406-408.

[Sun *et al*., 2003b] J. Sun, G. Chen, K.T. Ko, S. Chan and M. Zukerman. 2003. PD-Controller: A New Active Queue Management Scheme.*IEEE GLOBECOM 2003*. pp. 3103-3107.

[TCL] TCL, Tcl Developer X change, web site. http://www.tcl.tk/.

[Thiruchelvi and Raja, 2008] G. Thiruchelvi and J. Raja. 2008. A Survey On Active Queue Management Mechanisms. *International Journal of Computer Science and Network Security*, Vol. 8 No.12. pp. 130-145.

[Thyagarajam *et al.*, 1995] A. Thyagarajam, S. Casner, and S. Deering. 1995. Making the Mbone real. *INET 1995, Honolulu, HI,* pp. 465-474.

[VoIPStunt, 2009] VoIPstunt. 2009. http://www.voipstunt.com/en/index.html.

[Wang, 2001] Z. Wang. 2001. Internet QoS, Architectures and Mechanisms for Quality of service. *Ed. Morgan Kaufmann Publishers*, pp. 79-132.

[Wang *et al*., 2004] C. Wang, B. Li, Y. T. Hou, K. Sohraby and Y. Lin. 2004. LRED: A Robust Active Queue Management Scheme Based on Packet Loss Ratio. *IEEE INFOCOM 2004.*

[Wang *et al*., 2005] C. Wang, B. Li, Y. T. Hou, K. Sohraby and K. Long. 2005. A stable rate-based algorithm for active queue management. *Computer Communications 28*. pp. 1731-1740.

[Wroclawski, 1997] J. Wroclawski. 1997. Specification of the Controlled-Load Network Element Service. RFC 2211. http://www.ietf.org/rfc/rfc2211.txt.

[Wydrowski and Zukerman, 2002] B. Wydrowski and M. Zukerman. 2002. GREEN: An Active Queue Management Algorithm for a Self Managed Internet. *Proceeding of IEEE International Conference on Communications ICC 2002*, Vol.3. pp. 2368-2372.

[Yahoo, 2009] Yahoo. 2009. http://www.yahoo.com.

[Yanping *et al*., 2007] Q. Yanping, L. Qi, L. Xiangze and J. Wei. 2007. A Stable Enhanced Adaptive Virtual Queue Management Algorithm for TCP Networks. *IEEE International Conference on Control and Automation*.

[Yamada] T. Yamada. Voice Activity Detection in Noisy Environments. *Multimedia Laboratory, Institute of Information Sciences and Electronics, University of Tsukuba.* http://tosa.mri.co.jp/sounddb/nospeech/research/indexe.htm.

[Yamaguchi and Takahashi. 2007] T. Yamaguchi and Y. Takahashi. 2007. A queue management algorithm for fair bandwidth allocation. *Computer Communications 30).* pp 2048-2059.

[Yin and Hluchyj, 1993] N.Yin and M.G. Hluchyj. 1993. Implication of Droping Packets from the Front of a Queue. IEEE Transactions on Communications. Vol. 41. pp. 846-851.

[Youtube, 2009] You tube 2009. http://www.youtube.com.

[Zheng and Atiquzzaman, 2008] B. Zheng and M. Atiquzzaman. 2008. A framework to determine bounds of maximum loss rate parameter of RED queue for next generation routers. *J. Network and Computer Applications.*