Universitat Politècnica de Catalunya

Tesis Doctoral

Control de congestión multipunto en redes IP y ATM. Diseño de protocolos de transporte multipunto fiable.

Autora: Marta Solera Delgado
Director: Dr. Sebastià Sallent Ribes

2006

Programa de Doctorado de Ingeniería Telemática
Departamento de Ingeniería Telemática

RESUMEN

Las comunicaciones multipunto ofrecen, tanto a usuarios como a proveedores, mayor eficiencia, permitiendo desarrollar e implantar nuevos servicios. Para poder desplegarlos adecuadamente sobre la redes de comunicaciones, es necesario contar con protocolos adecuados a todos los niveles.

El soporte de conexiones multipunto, que es inmediato en muchas redes de ordenadores, por la existencia de un medio compartido, no lo es en una red ATM o en una red IP. En este tipo de redes, ofrecer una comunicación multipunto requiere de mecanismos complejos que coordinen y controlen la transmisión de información entre las fuentes y los receptores. Esta tesis doctoral estudia las comunicaciones punto a multipunto sobre las redes ATM e IP. En concreto, se diferencian dos objetivos:

- Estudiar, analizar y proponer un control de congestión punto a multipunto en la categoría de servicio *Available Bit Rate* (ABR) en redes ATM.
- Diseñar, analizar y simular un protocolo de transporte punto a multipunto fiable con control de congestión de tasa única para redes IP.

El control de flujo de ABR fue diseñado para comunicaciones punto a punto. Para el caso multipunto, los conmutadores deben desarrollar mecanismos que limiten y agreguen el tráfico de realimentación. Se ha desarrollado un algoritmo de consolidación que asegura que la agregación de la información de realimentación se realiza de forma correcta, mejorando la convergencia de propuestas anteriores. Este mecanismo se ha modelado matemáticamente, y se ha validado mediante simulación.

En cuanto a las comunicaciones multipunto en Internet, en este trabajo se ha desarrollado un protocolo de transporte punto a multipunto fiable, RCCMP, diseñado para ser escalable, fiable y con un control de congestión de tasa única que comparta el ancho de banda equitativamente con TCP.

El control de congestión ha sido planteado como una parte esencial del protocolo, y no como ocurre en muchas propuestas, como un componente adicional que debe ser ajustado a un protocolo de transporte. En RCCMP se combinan los objetivos de regular la tasa de transmisión y de conseguir una comunicación fiable, con el fin de simplificar y limitar el número de confirmaciones negativas que se envían desde los receptores. Para la evaluación de las prestaciones de RCCMP, se ha implementado el protocolo en el simulador ns-2.

El caudal de este protocolo de transporte multipunto se ha modelado en función de la tasa de pérdidas y del tiempo de ida y vuelta. La principal contribución de nuestro modelo radica en la caracterización del caudal ante cambios de representante. También, se ha desarrollado un método de análisis para estimar el ancho de banda consumido por cualquier protocolo de transporte. La principal diferencia con otros trabajos es que éstos se centran únicamente en el coste de los procesos del control de errores.

Para mejorar el rendimiento de RCCMP, se ha definido y simulado otro protocolo de transporte punto a multipunto fiable de tasa única, RVCMP, que incluye un control de congestión que emula al de la implementación TCP Vegas.

ÍNDICE

5. EVALUACIÓN DEL PROTOCOLO RCCMP

6. MODELADO DEL CAUDAL DEL PROTOCOLO RCCMP

ÍNDICE DE FIGURAS

CAPÍTULO 1

INTRODUCCIÓN

1.1 MOTIVACIÓN, OBJETIVOS Y CONTRIBUCIONES

Las comunicaciones multipunto aumentan notablemente la eficiencia en el uso de la red al compartir recursos entre todos los agentes participantes. Es por ello que cada día surgen más aplicaciones que requieren que uno o más emisores transmitan hacia un conjunto de usuarios. Algunos de los ejemplos más habituales son las aplicaciones colaborativas para trabajo en grupo, la distribución de audio/video para conferencias y presentaciones, y la transmisión multipunto de ficheros, programas y datos. Esta diversidad de aplicaciones da como resultado que existan multitud de requerimientos que deben satisfacer los protocolos. Además, para que puedan implementarse en cualquier red, todos los niveles de la torre de protocolos deben ofrecer un cierto grado de compatibilidad con este tipo de comunicaciones.

El soporte de conexiones multipunto, que es inmediato en muchas redes de ordenadores, por la existencia de un medio compartido, no lo es en una red ATM o en una red IP, en la que los terminales se conectan a conmutadores que los aíslan del resto de tráfico de la red. En general, en este tipo de redes, ofrecer una conexión multipunto involucra mecanismos más complejos que para las técnicas punto a punto, como son los mecanismos de enrutamiento que llevan un paquete de la fuente hasta cada uno de los receptores, los procedimientos de señalización para formar el grupo multipunto, y permitir que los usuarios se asocien y disocien en cualquier momento, el direccionamiento de estas agrupaciones de receptores, la ordenación de los paquetes que provienen de fuentes diferentes, etc. También, son más complejos los controles de flujo y congestión que deben regular la tasa de la fuente en función de algún criterio que sea adecuado para el grupo y la aplicación.

Los controles de flujo y congestión han sido ampliamente estudiados para comunicaciones punto a punto. En este caso, se ajusta la tasa de la fuente en función de la velocidad de recepción del terminal destino o de las condiciones de saturación de la conexión entre la fuente y el receptor. Otra de las funciones implícitas de estos mecanismos es el reparto equitativo de los recursos entre los protocolos que compiten por ellos. En multipunto, los objetivos son los mismos, sin embargo determinar la tasa de la fuente repartiendo de forma equitativa los recursos ante un

grupo de usuarios heterogéneos disgregados por toda la red, no es una tarea sencilla. Este va a ser una de las problemáticas que se van a estudiar y analizar en este trabajo de tesis.

En una red ATM se distinguen varias categorías de servicio que son apropiadas para distintos tipos de tráfico y aplicaciones. Una de ellas es *Available Bit Rate* (ABR) que garantiza una tasa mínima equitativa entre las conexiones competidoras, minimizando la tasa de pérdida de células, a partir de un control de flujo que periódicamente determina a qué velocidad debe transmitir la fuente. En principio, este mecanismo se diseñó para conexiones punto a punto, siendo necesario extenderlo para comunicaciones multipunto. En este caso, para garantizar una tasa de pérdida de células mínima, la tasa a la que transmite la fuente, se corresponde con la mínima que pueden soportar todos los conmutadores que se encuentran en el camino desde la fuente hasta cada uno de los receptores. Además, es necesario el desarrollo de ciertos algoritmos que limiten y agreguen el tráfico de realimentación, ya que esta información tiene que reflejar el estado de las conexiones multipunto y no debe crecer de forma proporcional al número de receptores.

En esta tesis doctoral se propone un algoritmo de consolidación que controla el orden de llegada de las células para determinar adecuadamente la información que llevará la célula de retroalimentación. Ésta incluye las condiciones de saturación de las conexiones que cuelgan del conmutador y están asociadas al grupo multipunto. De esta manera, el algoritmo propuesto intenta evitar el problema de consolidación, es decir, errores en la agregación de la información de realimentación, mientras se mantiene la escalabilidad del control de flujo.

Existen en la literatura diferentes trabajos que han modelado los controles de flujo punto a punto basados en tasa, y en concreto, sobre la clase de servicio ABR. Sin embargo, muy pocos han analizado el problema desde el punto de vista punto a multipunto. En este trabajo se obtiene un modelo analítico para el comportamiento del control de congestión multipunto, teniendo en cuenta que se utiliza el algoritmo de consolidación propuesto. Este análisis se desarrolla en base a considerar un modelo continúo, que se valida mediante simulación. Para ello, se ha modificado el simulador NIST ATM/HFC Simulator Versión 3.0, que no soporta conexiones punto a multipunto, incorporando un nuevo componente que es el conmutador multipunto.

En la arquitectura de protocolos TCP/IP, los controles de flujo y congestión que se van a estudiar son los asociados al nivel de transporte. En el caso de comunicaciones punto a punto, el protocolo dominante en Internet es TCP, que desarrolla un control de flujo y congestión para evitar la saturación de la red. En comunicaciones multipunto es muy importante que aparte de reducir y agregar la información de realimentación, los recursos se compartan de forma equitativa con TCP, para asegurar una perfecta convivencia entre todos los protocolos existentes en Internet. En esta tesis doctoral se propone un protocolo de transporte fiable para comunicaciones punto a multipunto que incluye un control de congestión de tasa única: RCCMP *(Reliable Congestion Controlled Multicast Protocol).*

En RCCMP se combinan los objetivos de regular la tasa de transmisión y de conseguir una comunicación fiable, con el fin de obtener una buena escalabilidad y simplificar el diseño del protocolo. El control de congestión de tasa única propuesto imita el comportamiento de la implementación TCP Reno, la más usada en Internet, para asegurar la máxima equidad. Para ello, entre todos los miembros del grupo se selecciona a un receptor, el de menores recursos, que será el representante. Entre éste

y la fuente se crea un lazo cerrado que regula una ventana de transmisión. De esta manera, la tasa del grupo se corresponde con la velocidad mínima que pueden soportar todos los enlaces involucrados en el grupo multipunto. Además, seleccionando a un solo receptor se evita el problema de la multiplicidad de caminos de pérdida (*Loss Path Multiplicity)*, reduciendo la cantidad de información de realimentación que llega a la fuente. El control de congestión propuesto no requiere de asistencia por parte de la red para la agregación de la información de realimentación, estimación del tiempo de ida y vuelta (RTT) o la modificación de las estrategias de las colas de los nodos intermedios. Es un control de congestión extremo a extremo, y solamente es necesaria la colaboración de los receptores.

Algunos autores han demostrado que la implementación TCP Vegas, que modifica el control de congestión de TCP Reno, consigue un mejor aprovechamiento del canal en ciertos entornos. En base a estos estudios se ha transformado el control de congestión de RCCMP para asemejarlo a TCP Vegas. Así, en este trabajo se propone un nuevo protocolo de transporte multipunto fiable RVCMP *(Reliable Vegas Congestion controlled Multicast Protocol).*

Para comprender el funcionamiento del protocolo de transporte multipunto propuesto, se ha modelado el comportamiento del control de congestión de RCCMP en función de los parámetros de la pérdida de paquete y del RTT. Como el control de congestión de RCCMP emula al de la implementación TCP Reno, el estudio está basado en el trabajo de [Padhye98]. La principal contribución de nuestro modelo radica en la caracterización del caudal ante cambios de representante. En una sesión multipunto, la aparición de un receptor con menos recursos que los del representante actual, modifica el caudal de la fuente, y este efecto debe ser incluido en el modelo.

El control de congestión no es el único requerimiento que tiene que cumplir un protocolo de transporte multipunto, sino que ha de satisfacer las necesidades de las diversas aplicaciones punto a multipunto. Uno de estos requerimientos es la fiabilidad, ya que algunas de ellas necesitan asegurar que la información transmitida llegue a todos los receptores.

La mayoría de los protocolos de transporte multipunto fiable usan el mecanismo de confirmaciones negativas para controlar la correcta recepción de los paquetes, y así, aumentar la escalabilidad. Sin embargo, cuando muchos receptores pierden el mismo paquete, es posible recibir un alto número de respuestas, pudiendo ver reducida la escalabilidad. Para disminuir el número de confirmaciones negativas existen diferentes mecanismos. El protocolo propuesto utiliza un esquema de temporizadores exponenciales, que ofrece diferentes ventajas: no necesita la colaboración de nodos intermedios, es una solución que no implica una estructura jerarquizada y es muy simple de implementar.

Para evaluar el comportamiento de los protocolos propuestos, se han implementado utilizando el simulador ns-2 *(network simulator).*

Otra de las contribuciones de este trabajo de tesis es el desarrollo de un método de análisis para estimar el ancho de banda consumido por cualquier protocolo de transporte. Este estudio no se limita a contabilizar el número de paquetes de realimentación que los receptores envían a la fuente, sino que tiene en cuenta los procesos que involucran un intercambio de paquetes, como son la selección del nuevo representante, el control de congestión y el control de errores. Aunque este análisis pretende ser un marco de referencia válido para cualquier protocolo, para simplificar el modelo se utiliza como caso de estudio el protocolo TCP y RCCMP.

La principal diferencia con otros trabajos a nivel de transporte es que éstos se centran únicamente en el coste de los procesos del control de errores.

1.2 DESARROLLO DE LA TESIS

La memoria de esta tesis doctoral se estructura en dos partes claramente diferenciadas y asimétricas. La primera está compuesta por el capítulo 3, y la segunda por el resto de capítulos. A continuación se detalla brevemente el contenido.

En el capítulo 2 se describen las bases para el estudio de la distribución de datos multipunto sobre las redes ATM e IP. Por un lado, se explica desde la arquitectura TCP/IP como una red como Internet es capaz de soportar comunicaciones multipunto; se analizan los requerimientos que deben cumplir los protocolos de transporte, y los principales esquemas para lograrlos. Por otro lado, se describe una de las categorías de servicio que ATM ofrece como es ABR (*Available Bit Rate*), sobre la que se va a trabajar en esta tesis doctoral, y sus principales problemas para soportar comunicaciones multipunto.

En el capítulo 3 se estudia y analiza el control de flujo multipunto sobre la clase de servicio ABR en redes ATM. Se propone un algoritmo de agregación que controla el orden de llegada de las células de control para establecer la información de realimentación adecuada. A continuación, se modela el comportamiento del control de congestión multipunto, en concreto se obtienen las expresiones de la tasa de la fuente y la ocupación de las colas del conmutador. Este análisis supone un modelo continúo que se resuelve mediante ecuaciones diferenciales. Para validar el modelo se utilizan los resultados obtenidos a través de simulaciones.

El capítulo 4 propone un nuevo protocolo de transporte multipunto fiable RCCMP *(Reliable Congestion Controlled Multicast Protocol).* Se explican los diferentes mecanismos que permiten que el protocolo cumpla con los requerimientos de fiabilidad, escalabilidad y equidad. Seguidamente, se describe el control de errores basado en una estrategia de confirmaciones negativas que utiliza un esquema de temporizadores exponenciales. Se detalla el control de congestión de tasa única que requiere de un receptor especial, el de menores recursos del grupo, para regular la tasa. En este capítulo también se detallan los formatos de los diferentes paquetes.

En el capítulo 5 se muestra y evalúa el comportamiento del protocolo RCCMP. Este estudio se lleva a cabo mediante simulaciones, utilizando el simulador de redes ns-2 (*network simulator*). A partir de topologías sencillas, se pretenden describir diferentes aspectos de su funcionamiento. Primero, se analiza la respuesta del protocolo ante diferentes valores de los parámetros de diseño. Segundo, se estudia el comportamiento en función del tiempo de ida y vuelta. Posteriormente, se muestra la conducta del protocolo ante cambios de representante. A continuación, se estudia la equidad entre los protocolos RCCMP y TCP, y se evalúa el comportamiento de RCCMP cuando comparte enlaces con otras instancias del mismo protocolo. Finalmente, se estudia la escalabilidad.

En el capítulo 6 se modela el mecanismo de control de congestión de RCCMP. Para ello, se analiza, en estado permanente (*congestion avoidance),* el caudal inyectado por la fuente en función de la probabilidad de pérdida y del RTT. Posteriormente, se valida el modelo mediante simulaciones.

El capítulo 7 plantea un método de análisis para hallar el ancho de banda consumido por un protocolo de transporte multipunto fiable. Se estudia la utilización del enlace en función del número de receptores para distintas probabilidades de error. Para simplificar el modelo analítico, se utiliza como caso de estudio el protocolo RCCMP. Este trabajo se completa con un estudio comparativo del ancho de banda utilizado por un protocolo de transporte punto a punto como TCP.

El objetivo del capítulo 8 es proponer un nuevo protocolo de transporte fiable, RVCMP *(Reliable Vegas Congestion controlled Multicast Protocol)*, que modifica el control de congestión de RCCMP, que emula al de TCP Reno, para asemejarlo al del protocolo TCP Vegas. De esta manera, se pretende mejorar la eficiencia. A continuación, se evalúa mediante simulación, y los resultados obtenidos son comparados con RCCMP.

En el capítulo 9 se resumen las conclusiones más relevantes de esta tesis doctoral y se apuntan las líneas futuras de trabajo.

Finalmente se adjuntan cinco apéndices, cuyos contenidos se describen a continuación.

En el apéndice A se caracteriza en valores medios y mediante el modelo de fluidos el mecanismo de control de congestión multipunto, basado en umbrales, para la categoría de servicio ABR. Se determina la probabilidad de pérdida de paquetes del sistema, el caudal y otros parámetros que caracterizan el control de congestión. Además, se estudia cuales son los valores de los umbrales más adecuados.

El apéndice B especifica utilizando un lenguaje estructurado de alto nivel el protocolo RCCMP. Se diferencian muy claramente dos procesos totalmente independientes: el emisor y el receptor.

El apéndice C analiza la complejidad del protocolo, evaluando los recursos de espacio de almacenamiento y de tiempo de cómputo que RCCMP necesita para su desarrollo.

En este apéndice D se estudian los temporizadores exponenciales que han sido utilizados en el protocolo RCCMP.

En el apéndice E se analizan y evalúan el protocolo de transporte multipunto PGM, y los controles de congestión PGMCC y ORMCC. Éstos se comparan con RCCMP.

CAPÍTULO 2

COMUNICACIONES MULTIPUNTO

2.1 INTRODUCCIÓN

Una de las clasificaciones más importantes en las que se puede dividir una comunicación es aquella que hace referencia al número de participantes y al papel desempeñado por cada uno de éstos. Así, dicha clasificación permite distinguir entre:

- Comunicaciones punto a punto, en la que intervienen únicamente dos terminales, siendo posible distinguir comunicación punto a punto unidireccional y bidireccional. Ejemplos clásicos pueden ser la conversación telefónica o la transmisión de paquetes de datos mediante TCP en la red Internet.
- Comunicación punto a multipunto, en este caso participa un emisor que transmite a muchos receptores, pudiendo existir información de retroalimentación entre los receptores y el emisor. Los sistemas de difusión de ondas, como la radio o la televisión, son ejemplos típicos de este tipo de comunicación.
- Comunicación multipunto a multipunto, es el caso más genérico, donde en un conjunto de terminales pueden distinguirse varios emisores y varios receptores. Las comunicaciones de radio-aficionado o los sistemas de chateo, donde conceptualmente todos los terminales son a la vez emisores y receptores, son quizás los ejemplos más intuitivos de comunicación multipunto a multipunto.

Una de las principales ventajas de las comunicaciones multipunto es que aumentan notablemente la eficiencia de uso de la red al compartir recursos entre todos los agentes participantes. Así, por ejemplo, la transmisión de una conferencia usando mecanismos de comunicación punto a punto implica generar un paquete para cada uno de los posibles receptores, aún cuando parte del camino sea común. Los sistemas de comunicación punto a multipunto ofrecen mecanismos que permiten transmitir un único paquete, que es desdoblado en aquellos nodos intermedios en los que los caminos hacia los distintos receptores divergen. Para que la comunicación

punto a multipunto resulte posible, es necesario disponer de nodos intermedios expresamente diseñados para permitir este tipo de transmisión.

Por otra parte, la transmisión punto a multipunto permite la aparición de nuevos servicios multimedia, que aprovechen al máximo las capacidades de difusión de este tipo de comunicaciones. Cabe dividir a estos nuevos servicios en varias categorías:

- Colaborativos o de trabajo en grupo: estos servicios usan las capacidades de transmisión de uno a muchos para permitir a un grupo de usuarios de tamaño no restringido, trabajar conjuntamente en la elaboración de documentos, bases de datos, hojas de cálculo o cualquier otra actividad. En la red de comunicaciones punto a multipunto de Internet, una de las aplicaciones más populares es la “pizarra compartida”, donde varios usuarios pueden pintar y escribir sobre un mismo lienzo en tiempo real.

- De difusión o no interactivos: usando las capacidades de transmisión eficiente de las comunicaciones multipunto, estos servicios ofrecen a un alto número de usuarios la posibilidad de seguir la retransmisión de audio (radio por Internet) o de audio y video (televisión por Internet) sin limitaciones en cuanto al número de espectadores y sin obligar a los difusores o transmisores a usar una gran capacidad de transmisión.

- Interactivos o de juego en grupo: escalando los juegos multimedia por Internet actuales, los servicios interactivos pretenden ofrecer una nueva experiencia multimedia en tiempo real, donde el número de usuarios no dependa directamente de la capacidad de los servidores para transmitir una copia de cada paquete de información a todos los usuarios.

Otras aplicaciones punto a multipunto son la distribución de titulares de noticias, actualizaciones de las previsiones del tiempo u otros tipos de información no esencial y dinámica; la emisión de información útil para un gran grupo de usuarios como el tiempo de red, la programación de sesiones multipunto o claves para seguridad; las aplicaciones para monitorizar sistemas sensores para actividades sísmicas, meteorológicas o de control; y la distribución de ficheros.

Dentro de las aplicaciones multipunto-multipunto se puede añadir la sincronización de recursos, como el uso compartido de cualquier tipo de bases de datos; el procesado concurrente; la tele-enseñanza; y aplicaciones para simulaciones interactivas distribuidas (DIS) o sesiones de audio compartido.

El resto del capítulo trata de describir las diferentes particularidades de las comunicaciones multipunto sobre redes IP y ATM. Primero, se comienza explicando como los diferentes niveles de la arquitectura de protocolos TCP/IP proporcionan servicios para este tipo de comunicaciones. A continuación, se detallan los diferentes requerimientos de los protocolos de transporte multipunto y los principales esquemas para lograrlos. Posteriormenete, se introducen las comunicaciones multipunto en las redes ATM. Finalmente, se resume el funcionamiento de la categoría de servicio ABR *(Available Bit Rate)*, y se analizan cuales son sus principales problemas a la hora de afrontar este tipo de conexiones.

2.2 LAS COMUNICACIONES MULTIPUNTO SOBRE INTERNET

La arquitectura de protocolos TCP/IP es el resultado de la investigación y del desarrollo llevado a cabo en la red experimental de conmutación de paquetes ARPANET. A partir de los protocolos estándar desarrollados, todas las funciones involucradas en la comunicación se pueden organizar en cinco capas: aplicación, transporte, red, enlace y física.

La transmisión punto a multipunto necesita que varios de estos niveles proporcionen servicios específicos para este tipo de comunicación. Concretamente, se puede dividir la comunicación punto a multipunto en comunicación intrared, entre terminales dentro de una misma red local, y comunicación interred, entre terminales que pueden estar situados en distintas redes o subredes.

El nivel de enlace es el encargado de ofrecer servicios para la comunicación intrared. Así, por ejemplo, Ethernet dispone de mecanismos de difusión, mediante los cuales un terminal puede enviar un paquete de información a todos los terminales conectados en la misma red local.

Para la comunicación interred se usan los niveles de red y de transporte. El nivel de red, encargado de encaminar los paquetes, debe ofrecer los mecanismos necesarios para poder transmitir un paquete a un conjunto de receptores. Para Internet, donde a nivel de red se usa el protocolo IP (*Internet Protocol*) [RFC791] [RFC1180], se ha desarrollado una variante de este protocolo, conocido como IP Multicast [RFC1112]. Las principales diferencias entre estos dos protocolos se basan en la extensión del esquema de direccionamiento, en la modificación de los mecanismos de encaminamiento y en la incorporación de un protocolo para la gestión del grupo.

IP Multicast se basa en un modelo abierto de grupos de receptores, donde cada grupo se identifica con una dirección multipunto (en IP versión 4 se usa la clase D de direcciones, cuyo rango varía entre 224.0.0.0 y 239.255.255.255). Las principales características de estos grupos son las siguientes:

- Pertenencia libre: cualquier terminal puede unirse a cualquier grupo, no siendo necesaria autorización alguna.
- Pertenencia múltiple: un terminal puede pertenecer a varios grupos simultáneamente.
- Transmisión no restringida: cualquier terminal puede transmitir a cualquier grupo, no siendo necesaria la pertenencia a él para enviar datos al grupo.
- Anonimato: IP Multicast no proporciona mecanismos para conocer el número de terminales pertenecientes a un grupo o la identidad de éstos.
- Grupos dinámicos: cualquier terminal puede unirse o abandonar un grupo en cualquier momento.

Para realizar la gestión local de grupos y el encaminamiento local se usa el protocolo IGMP (*Internet Group Management Protocol*) [RFC1112] [RFC2236] [RFC3376]. Con IGMP se puede establecer una comunicación entre terminales y su nodo intermedio local, permitiendo que los terminales locales expresen su deseo de unirse, permanecer o retirarse de un grupo de comunicación multipunto. Existen

diferentes configuraciones de IGMP, que se diferencian principalmente en si los terminales locales se suponen miembros del grupo por omisión, lo que optimiza el número de paquetes transmitidos en subredes con un alto número de terminales pertenecientes al mismo grupo multipunto, o por indicación, lo que optimiza el número de paquetes transmitidos en subredes con pocos agentes de comunicación punto a multipunto.

Para realizar tanto el encaminamiento global como la gestión global de grupos, han surgido varios protocolos. Uno de los más exitosos y que ha sido usado en M-BONE [Deering93] es DVMRP (*Distance Vector Multicast Protocol Routing*) [RFC1075], extensión del protocolo de encaminamiento de comunicaciones punto a punto RIP [RFC1058]. MOSPF (*Multicast Open Shortest Path First*) [RFC1584] es otro protocolo de encaminamiento basado en OSPF V2 [RFC2328], que permite un cálculo rápido de las rutas con un mínimo intercambio de información entre nodos intermedios. Estos dos protocolos presentan problemas de escalabilidad, por ello el IETF (*Internet Engineering Task Force*) ha definido otros protocolos para extender el soporte multipunto a un área mayor: PIM-DM (*Protocol Independent Multicast-Dense Mode*) y PIM-SM (*Protocol Indepent Multicast- Sparse Mode*) [Deering94]. Ambos son independientes de los mecanismos de encaminamiento punto a punto subyacentes y emplean mensajes de control similares.

Hay que resaltar que los protocolos de encaminamiento multipunto son, por lo general, bastante más complejos que sus homólogos en punto a punto, y por otro lado, su desarrollo ha sido más tardío, por lo que aún presentan mayores deficiencias, sobre todo cuando se aplican a redes complejas y extensas. Sin embargo, su evolución ha sido más rápida, debido en gran medida, al gran interés que ha despertado en función de sus enormes posibilidades de aplicación, y a la presión de usuarios finales y empresas que demandan una oferta de servicios multimedia de forma eficaz y económica.

En cuanto al nivel de transporte, debe cubrir aquellos requisitos exigidos por las aplicaciones que no hayan sido satisfechos en los niveles inferiores. Entre éstos cabe destacar:

- Fiabilidad: algunas aplicaciones necesitan asegurar que la información transmitida llega a todos los receptores.
- Recepción ordenada: relacionado con el punto anterior, consiste en asegurar que la información llega a la aplicación destino en el mismo orden en la que fue transmitida.
- Control de flujo y congestión: es quizás, uno de los requisitos más importantes, ya que sin el control de congestión existente actualmente en protocolos de transporte como TCP, Internet no sería una red funcional. Con el control de flujo y congestión se asegura que la velocidad de transmisión se adapte a las características dinámicas de la red.
- Gestión de grupo: algunas aplicaciones necesitan más información o un mayor grado de control sobre los grupos que el ofrecido por IP Multicast. En esos casos, el protocolo de transporte debe aportar los mecanismos necesarios para poder realizar dichas funciones.

- Requisitos de tiempo real: algunas aplicaciones imponen límites temporales al retardo máximo que un receptor puede asumir o a la varianza de retardo entre distintos paquetes para un mismo receptor.

Debido a que no todas las aplicaciones tienen las mismas necesidades, y a que la gran variedad de características entre aplicaciones multipunto es mayor que entre aplicaciones punto a punto, no ha sido posible establecer un único protocolo de transporte multipunto estándar, equivalente a lo que representa TCP para las comunicaciones punto a punto. Este es el motivo de que existan distintas propuestas de protocolos de transporte fiable que intenten cubrir parte de estas necesidades.

El nivel de aplicación impone los requisitos que el protocolo de transporte ha de cumplir. Se ha sugerido un esquema de negociación entre las aplicaciones y los niveles de transporte, similar al que utiliza el protocolo RTP (*Real Time Protocol*) [RFC1889]. Este esquema, bautizado ALF (*Application Level Framing*) [Clark90], fue descrito por primera vez en el protocolo SRM (*Scalable Multicast Protocol*) [Floyd95].

2.2.1 Los protocolos de transporte en las comunicaciones multipunto

Los protocolos de transporte punto a multipunto han de satisfacer las necesidades de las aplicaciones, pero sin olvidar cumplir los requerimientos básicos para poder transmitir eficientemente a través de Internet, propios también de los protocolos punto a punto.

El principal de estos requisitos es la escalabilidad. Los protocolos multipunto pueden ser usados en circunstancias donde hay un alto número de terminales que intervienen en la comunicación, y por lo tanto, deben contar con mecanismos que eviten que tanto el tráfico como la información de estado a almacenar, crezca proporcionalmente al número de receptores.

El otro gran requisito de los protocolos de transporte multipunto es el comportamiento equitativo con respecto a TCP. En Internet ya existe un tráfico dominante, que es el tráfico TCP. Para que un protocolo de transporte multipunto pueda ser usado en Internet sin afectar al tráfico existente, debe comportarse de manera equitativa con respecto a este protocolo.

2.2.1.1 Mecanismos para asegurar la fiabilidad

En las comunicaciones punto a punto, el protocolo de transporte fiable TCP utiliza confirmaciones positivas, ACKs, transmitidas por el receptor para saber que paquetes de información han llegado correctamente a su destino. Sin embargo, en las comunicaciones multipunto no resulta posible aplicar un esquema de ACKs generalizado, ya que al recibir una confirmación por cada posible receptor, el protocolo resultante no sería escalable. Sin embargo, existen algunos primeros trabajos que se basan en esta estrategia [Ram92], [Sabnani85], [Towley85], [Towley87] y [Wang89].

La mayoría de los protocolos de transporte multipunto usan el mecanismo de confirmaciones negativas, NAKs, para controlar la correcta recepción de los paquetes. Según este esquema, cada vez que un receptor detecte la ausencia de un

paquete, debe enviar una confirmación negativa al emisor para solicitar la retransmisión.

La estrategia de confirmaciones negativas, aunque aumenta la escalabilidad de los protocolos multipunto, también presenta sus propios problemas. En concreto, cuando muchos receptores pierden el mismo paquete, es posible recibir un alto número de NAKs. Este fenómeno es conocido como implosión o tormenta de NAKs. Para reducir el número de confirmaciones negativas se pueden usar uno o varios de los siguientes mecanismos:

- Repetidores locales: según esta estrategia, determinados receptores son seleccionados como "repetidores". Cada repetidor sirve a un conjunto de receptores, que envían sus peticiones de retransmisión a dicho receptor. Sólo si el repetidor es incapaz de satisfacer la petición de retransmisión se enviará un NAK al emisor. PGM (*Pragmatic General Multicast*) [RFC3208] y SRM (*Scalable Reliable Multicast*) [Floyd95] son ejemplos de protocolos que usan repetidores locales.

- Jerarquización de los receptores: esta solución lleva la estrategia de repetidores locales al extremo, imponiendo una jerarquía entre todos los receptores, en la cual cada receptor sólo puede enviar un NAK al eslabón superior. Uno de los protocolos con jerarquización de receptores más extendido es RMTP (*Reliable Multicast Transport Protocol*) [Lin96].

- Colaboración de los elementos de red: esta estrategia se basa en que los elementos de red realicen un filtrado, de manera que de varias peticiones de retransmisión, sólo una llegue al emisor. Aunque aumenta notablemente la eficiencia de la comunicación, el uso de este mecanismo impone requisitos a los nodos intermedios, dispositivos que ya de por sí tienen una carga de trabajo significativa. El protocolo PGM necesita la colaboración de los nodos intermedios para lograr una transmisión eficiente.

- Temporizadores: el uso de temporizadores disminuye el número de NAK que el emisor recibe, logrando así, una mayor eficiencia en la comunicación. Estos mecanismos son exclusivamente locales por lo que consumen pocos recursos. Cuando un receptor detecta la falta de un paquete de información, genera un tiempo de espera al azar. Si ese tiempo transcurre sin que ningún otro receptor haya enviado un NAK, el receptor transmite su NAK directamente al emisor. Existen muchos protocolos basados en temporizadores, que a su vez puede tener una función de densidad uniforme, exponencial, etc., siendo TFMCC (*TCP-friendly Multicast Congestion Control*) [Widmer01b], uno de los ejemplos más recientes.

2.2.1.2 El control de congestión en los protocolos de transporte multipunto

Los mecanismos de control de congestión de un protocolo de transporte se encargan de regular el tráfico, generado en el emisor, a las circunstancias cambiantes de la red. El control de congestión de las comunicaciones TCP en Internet se basa principalmente en el trabajo de Van Jacobson [Jacobson88], que en 1986, ante la

primera disrupción seria de Internet, desarrolló ciertos mecanismos para evitar que se volviera a producir esa situación.

Los algoritmos de control de congestión de los protocolos para transporte multipunto son más complejos y variados que los disponibles en comunicaciones punto a punto. Una de las principales diferencias entre los controles de congestión de los protocolos multipunto y los punto a punto es como tratar la información de realimentación, para evitar una reducción excesiva de la tasa de la fuente, respuesta a los mensajes de congestión desde los distintos receptores. Este problema es conocido como la multiplicidad de caminos de pérdida (*Loss Path Multiplicity Problem), crying baby o drop-to-zero* [Bhatta99]. La solución consiste en agregar o filtrar, de alguna manera, los paquetes de realimentación que llegan desde los receptores. Así, se distinguen protocolos que utilizan estructuras jerárquicas que con la cooperación de receptores y nodos intermedios agregan la información de realimentación, como por ejemplo [Golestani99] y [Rhee99]. Mientras que otros algoritmos seleccionan a uno o varios receptores como los representantes del grupo multipunto [Rizzo00] y [Kasera00].

Los controles de congestión multipunto admiten diferentes clasificaciones. Una de ellas es función de cómo los algoritmos utilizan la información de realimentación para regular la transmisión. Así, se distinguen dos grandes grupos:

- Transmisión regulada por tasa: los protocolos clasificados de este modo transmiten la información a la velocidad indicada por una tasa calculada según las circunstancias de la red. Esta velocidad normalmente va aumentando lentamente hasta que llega al emisor una señal de congestión, momento en el que se reduce drásticamente, en lo que se conoce como incremento lineal, reducción exponencial. Generalmente, la tasa de estos protocolos se escoge de forma que imiten la fórmula simplificada de TCP [Mathis97]. ORMCC [Li02] y TFMCC [Widmer01b] son ejemplos de esquemas regulados por tasa.
- Transmisión regulada por ventana: en este caso, el protocolo dispone de una ventana de transmisión, y la información desde los receptores se usa para ir desplazándola conforme los paquetes de información enviados son recibidos. PGMCC [Rizzo00] incluye un algoritmo de ventana de este tipo para regular su transmisión.

Los protocolos regulados por tasa son más comunes en multipunto que en punto a punto, probablemente porque requieren una interacción menos frecuente que los protocolos regulados por ventana, que deben transmitir continuamente qué paquetes han sido recibidos y cuales no.

Otro de los criterios más comunes de clasificación se basa en diferenciar si la fuente transmite uno o varios flujos de información. De esta manera, se distinguen dos grandes subgrupos:

- Control de congestión unitasa: el emisor transmite a una única tasa, que varía según las circunstancias de la red. Normalmente dicha tasa se corresponde con la máxima velocidad que el peor receptor es capaz de asimilar. La principal ventaja de estos esquemas es que son muy sencillos tanto conceptualmente como de implementar; sin embargo, existe una cierta ineficiencia en la comunicación, ya que muchos receptores recibirán la información a una tasa muy por debajo de su

máxima capacidad. Ejemplos de esquemas de congestión unitasa son PGMCC [Rizzo00], TFMCC [Widmer01b], MTCP [Rhee99] y ORMCC [Li02].

- Control de congestión multitasa: en estos esquemas se distinguen varias tasas, y cada receptor puede seleccionar la que más adecuada le resulte. Así, el emisor divide los datos a transmitir, en distintos niveles o capas y transmite cada uno de ellos a un grupo multipunto distinto. Los receptores se asocian a uno o más grupos multipunto en función de sus recursos. De esta manera, se consigue una asignación de ancho de banda más flexible y una mejor escalabilidad que en los primeros, sin embargo, son mucho más complejos y pueden sufrir de falta de equidad con TCP. Ejemplos de estos protocolos son RLC (*Receiver-Driven Layered Congestion Control*) [Vicisiano98] y FLID-DL (*Fair Layered Increase/Decrease with Dynamic Layering*) [Byers00].

En general, aunque ya se ha dicho que la eficiencia de los esquemas multitasa es mayor que los de unitasa, aún no se ha logrado desarrollar un esquema multitasa enteramente satisfactorio; y aunque es probable que en el futuro se logre, siempre será más complicado de implementar que los esquemas unitasa. Todo esto lleva a que los protocolos unitasa tienen un futuro, al menos a corto plazo, más esperanzador que los multitasa.

El diseño de un protocolo de control de congestión, y particularmente, el reparto equitativo de los recursos, puede ser facilitado considerablemente introduciendo inteligencia en la red a modo de nodos intermedios o agentes especializados. Generalmente, el soporte que requieren de la red se traduce en la agregación de la información de realimentación, la estimación del RTT, la gestión de los subgrupos de receptores o la modificación de las estrategias de las colas de los nodos intermedios. Por otra parte, los controles de congestión multipunto extremo-extremo, que no requieren de asistencia por parte de la red, necesitan de la colaboración de los receptores. Algunas experiencias en Internet, demuestran que determinadas aplicaciones, para obtener un mayor ancho de banda, no utilizan mecanismos que contemplan el reparto equitativo de los recursos con TCP; por lo que sería necesario en aras de prevenir un colapso de la red, que los nodos intermedios fueran capaces de identificar flujos y regularlos correctamente. Sin embargo, estos mecanismos son difíciles de desarrollar, ya que requiere cambios en la infraestructura de red que llevan tiempo e importantes inversiones económicas y de esfuerzo.

2.3 LAS COMUNICACIONES MULTIPUNTO SOBRE ATM

La tecnología ATM presenta dos importantes ventajas como facilitar la conmutación a alta velocidad y permitir la gestión de flujos de información en la red. Por esta razón, ATM se ha desarrollado como red troncal en las redes de los operadores, y en menor medida, en redes privadas que requieren prioridades en los flujos de información. Aunque, no ha penetrado en las redes de área local, ni se hayan desarrollado aplicaciones especificas para esta tecnología, un importante porcentaje, alrededor del 80%, del tráfico de Internet pasa a través de estas redes.

Como se ha comentado, una de las aportaciones más importantes de ATM son las referidas a la gestión de tráfico y la calidad de servicio. Para ello, se distinguen ciertas categorías de servicio que son apropiadas para distintos tipos de tráficos y aplicaciones. Las seis categorías de servicio son: *Constant Bit Rate* (CBR), *real-time Variable Bit Rate* (rt-VBR), *non real-time Variable Bit Rate* (nrt-VBR), *Available Bit Rate* (ABR), *Guaranteed Frame Rate* (GFR) y *Unspecified Bit Rate* (UBR). Las características de tráfico que se garantizan se encuentran claramente especificadas para cada una de estas clases. Además, ATM ofrece una serie de funciones para la gestión de tráfico que permiten optimizar los recursos y garantizar la calidad de servicio que el usuario ha contratado. Entre éstas destacan los mecanismos de control de admisión (CAC) y de vigilancia de tráfico (UPC), técnicas para conformar el tráfico (*traffic shapping*), funciones para la gestión inteligente de las colas y el control de flujo para ajustar la tasa de la fuente en función de las condiciones de la red. Esta última técnica es la utilizada por la clase de servicio ABR.

El soporte de comunicaciones tipo multipunto, que es inmediato en muchas redes de ordenadores, por la existencia de un medio compartido, no lo es en una red ATM, en la que los terminales se conectan a conmutadores que los aíslan del resto de tráfico de la red. Sin embargo, ATM cuenta con una señalización específica, que permite crear árboles multipunto, a los cuales los terminales pueden asociarse o disociarse dinámicamente. Este procedimiento se encuentra descrito en el estándar ATM UNI 3.1 [UNI3.1]. En él, solamente el terminal raíz puede establecer el árbol de distribución, controlar qué terminales se añaden a la comunicación y enviar datos. En una versión posterior, ATM UNI 4.0 [UNI4.0], los terminales pueden tomar la iniciativa para asociarse al árbol multipunto.

Actualmente, solamente se soportan conexiones punto a multipunto a partir de circuitos virtuales (VCCs). Las conexiones multipunto a multipunto pueden ser obtenidas de las dos formas siguientes:

- Configuración de N conexiones punto a multipunto para conseguir conectar todos los nodos en una topología completamente mallada. El principal problema de esta solución es que no escala bien cuando el número de participantes es elevado.
- Uso de un servidor que actúa a modo de nodo raíz en el árbol multipunto. Este método requiere de un terminal raíz para almacenar la información. La desventaja radica en la saturación de este servidor cuando debe encargarse de envíos y retransmisiones de las conexiones multipunto a multipunto.

Para solventar las limitaciones de UNI 3.1, que soportan conexiones punto a multipunto, pero no nativamente conexiones multipunto a multipunto, ATM Forum, ITU-T e IETF han realizado varias propuestas al actual mecanismo de señalización (UNI 3.1, UNI 4.0). Por ejemplo, una mejora del protocolo PNNI (*Private network network interface*) que define mecanismos de enrutamiento para conexiones multipunto en la UNI 4.0.

Uno de los problemas que se encuentran al utilizar circuitos virtuales multipunto-multipunto, en los que existen múltiples fuentes en el mismo circuito lógico, es la identificación de células generadas por diferentes fuentes que pertenecen a la misma conexión.

La capa de adaptación 5 (AAL5) es la más utilizada para la encapsulación de datos. Ésta no introduce ningún identificador para la multiplexación de flujos o número de secuencia para las unidades de datos de ATM. Si las células de diferentes fuentes se unen o intercalan, por ejemplo en una conexión multipunto a punto, el receptor AAL5 no es capaz de reensamblar los datos, debido a que todo el tráfico utiliza los mismos identificadores VPI/VCI (*Virtual Path Identification/Virtual Channel Identification*). Se han encontrados diferentes soluciones para este problema, como evitar los procesos de fusión de VCCs -utilizando diferentes conexiones-, previniendo el intercalado de las células en los conmutadores, o introduciendo más identificadores en la cabecera de las células.

No existen excesivas propuestas de protocolos que ofrezcan un servicio multipunto sobre un árbol de distribución compartido. Entre éstas cabe destacar SMART (*Shared many-to-many ATM Reservations*) [Gauthier97] y SEAM (*Scalable and Efficient ATM Multicast*) [Gross97].

El primero es un protocolo para controlar un árbol de distribución compartido soportando comunicaciones multipunto. SMART controla el turno de transmisión de las fuentes, para evitar el problema del intercalado de las células, permitiendo que un emisor transmita solamente si tiene el testigo. Solamente existe un testigo por VCC. El protocolo garantiza la calidad de servicio asociada a los circuitos virtuales, a partir de mensajes de control especiales. Esta propuesta reside completamente en la capa ATM y no requiere de ningún servidor. SMART puede ser entendido como un protocolo completamente distribuido para coordinar la distribución de VPIs/VCIs.

SEAM propone una arquitectura escalable, eficiente y multipunto para redes ATM, que usa un solo VCC multipunto como árbol de distribución compartido para todos los emisores y receptores. Todos los paquetes asociados a un grupo multipunto llevan asociados un identificador, además cada grupo tiene un nodo que es utilizado como punto focal para todos los mensajes de señalización. Esta propuesta permite a los grupos multipunto aprovechar el soporte de calidad de servicio y la escalabilidad del ancho de banda. También realiza aportaciones para conseguir soportar IP Multicast sobre redes ATM extensas.

2.3.1 El Servicio ABR

El mecanismo de control de flujo ABR permite que la red divida el ancho de banda disponible de una manera equitativa y eficiente entre las diferentes fuentes activas. Para ello, se utiliza un control de flujo basado en tasa, de lazo cerrado y extremo a extremo.

Para regular la tasa a la que se envía tráfico a la red, la fuente genera una célula de control llamada *Resource Management* (RM) cada *Nrm –1* células de datos. El destino devolverá estas células hacia la fuente. Las células RM que van desde la fuente hacia el destino se llaman células FRM (*Forward* RM), mientras las que fluyen desde el destino a la fuente se llaman BRM (*Backward* RM). Las células RM pueden ser examinadas y modificadas por los elementos intermedios de red en cualquiera de las dos direcciones. La figura 2.1 muestra los componentes del control de congestión descrito.

Los campos más importantes de la células RM son: CCR (*Current Cell Rate*), ER (*Explicit Rate*), CI (*Congestion Indication*) y NI (*No Increase*). Estos tres últimos campos llevan la información de control necesaria para que la fuente modifique su tasa en función del estado de la red. El formato detallado puede encontrarse en [UNI4.1].

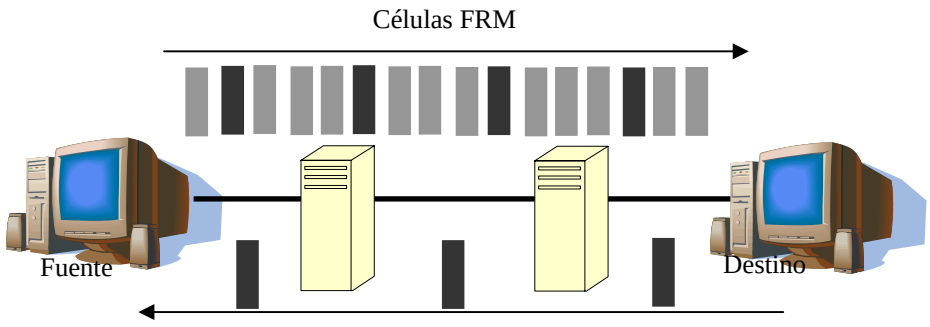

Figura 2.1 Esquema del control de flujo ABR.

Los conmutadores deben implementar al menos uno de los siguientes métodos de control de congestión en los puntos de encolamiento:

a) *EFCI marking*. Cada célula de datos contiene un bit EFCI *(Explicit Forward Congestion Indication)* que puede activarse si el conmutador está congestionado. Este mecanismo es una modificación del DECbit [Jain98]. A estos conmutadores se les denomina conmutadores EFCI o binarios de primera generación.

b) *Relative Rate Marking*. El conmutador puede modificar el campo CI y NI de la célula RM. A estos conmutadores se les llama *Relative Rate Marking* o *Binary Enhanced Switches* (BES).

c) *Explicit Rate Marking*. El conmutador puede reducir el valor del campo ER de la célula RM en función del estado de congestión. A estos conmutadores se les llama *Explicit Rate Switches* (ERS).

Los conmutadores a) y b) son mucho más sencillos que los últimos, pero sin embargo pueden sufrir de problemas de equidad, inestabilidad y lentitud en la respuesta a la congestión. El estándar define los comportamientos de la fuente y el destino, pero permite que los conmutadores implementen cualquiera de los tres métodos anteriormente descritos.

La fuente transmite células a tasa ACR *(Allowed Cell Rate)* que es inicializada a ICR *(Inicial Cell Rate)* al comienzo de la transmisión con valores entre PCR *(Peak Cell Rate)* y MCR *(Minimum Cell Rate)*. Estos tres parámetros se negocian antes de que se acepte la conexión. Junto con las células de datos se transmiten las células RM. La fuente introduce en el campo CCR de la célula el valor de ACR y en el campo ER, el valor de la tasa a la cual la fuente desea transmitir, generalmente PCR. Las células RM atravesarán la red hasta llegar al destino, que las devolverá hacia la fuente. Los conmutadores intermedios notificarán congestión hacia la fuente marcando el bit EFCI de las células de datos, modificando los bits CI y NI, o reduciendo el valor ER de la célula RM.

Una vez que la célula RM llega a la fuente, se actualiza el valor ACR dependiendo de la información que ésta lleve. Si los bits CI y NI no están activados, la fuente puede incrementar su tasa en un valor fijo, de manera que la tasa no exceda de PCR. Si la célula RM tiene activado el bit NI, no se incrementará el valor de ACR. Si el bit CI está a 1, la fuente reducirá su velocidad, pero teniendo en cuenta que no puede ser menor que MCR. Si la tasa ACR calculada es mayor que la tasa ER del campo de la célula BRM recibida, la fuente reducirá su ACR a ER, siempre que ésta sea mayor que MCR.

2.3.2 El Control de flujo para conexiones punto a multipunto sobre ABR

Cuando se tratan conexiones punto a multipunto, dos cuestiones inmediatas, que resultan más o menos obvias en conexiones punto a punto, deben ser resueltas. La primera es a qué velocidad debe transmitir la fuente y la segunda cómo se limita el tráfico de realimentación que llega a la fuente desde cada uno de los destinos. Para dar respuesta a estas dos preguntas, en esta sección se va a definir el criterio de equidad en el caso de tener múltiples receptores, y se van a analizar diferentes propuestas para evitar la saturación de células BRM.

Uno de los criterios de equidad más utilizados en ABR es una extensión del criterio Max-Min [UNI4.1]. Según este principio, la fuente debe transmitir a la capacidad máxima que permita el enlace que represente el cuello de botella de todas las conexiones involucradas. De esta manera, se minimizan la tasa de pérdidas de todas las ramas, ya que la velocidad de la fuente coincide con el enlace más restrictivo del árbol multipunto. Este criterio resulta muy adecuado para aquellos receptores que no puedan tolerar pérdidas. Sin embargo, puede resultar ineficiente en cuanto a la utilización del ancho de banda.

Se han desarrollado otros criterios que intentan maximizar otros parámetros como la equidad entre el grupo, que muestra la relación de tasas entre un receptor del grupo y los otros miembros, la equidad intergrupo, como la relación de tasas entre el flujo multipunto y otros flujos competidores, y la equidad interreceptores, que mide el cociente entre la tasa mínima de la fuente y la tasa que puede aceptar un receptor para un tolerancia de pérdidas determinada [Jiang98b]. Sin embargo, para el trabajo que aquí se presenta se utilizará la extensión del criterio Max-Min.

En una conexión punto a multipunto, los conmutadores deben ser capaces de replicar cada una de las células provenientes desde la fuente, hacia cada uno de los puertos de salida que llevan hacia un miembro del grupo. Además, deben ser capaces de realizar el proceso inverso, es decir, de consolidar las células BRM que provienen de los destinos, para evitar que el flujo de realimentación crezca de forma proporcional al número de receptores (figura 2.2). A este problema se le conoce como sobrealimentación o implosión de realimentación. Diferentes algoritmos de consolidación han sido propuestos, algunos de los más interesantes son [Roberts94b], [Yokotani95], [Siu95], [Ren96], [Cavendish96], [Moh97], [Cho97], [Moh98], [Jiang98a] y [Ren98].

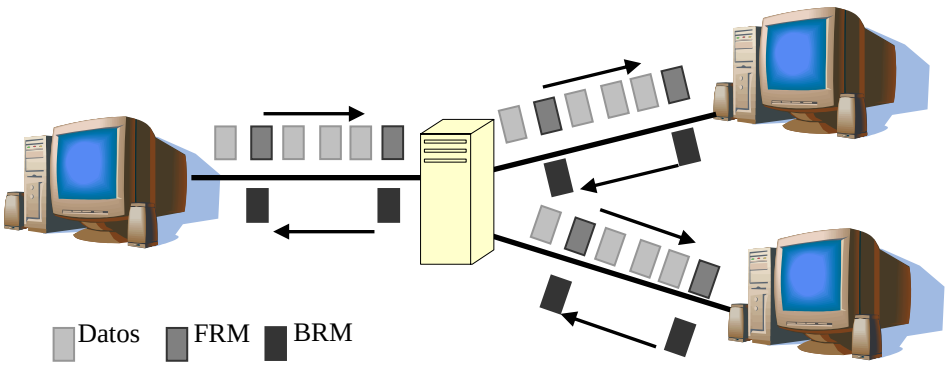

Figura 2.2 Problema de consolidación de realimentación

La primera propuesta publicada por L. Roberts [Robert94b] estudia una extensión para conexiones punto a multipunto del control de congestión ABR. El algoritmo propuesto está basado en el algoritmo EPCRA [Robert94a] [Siu96]. Cada vez que una celda FRM se recibe, se replica en cada una de las ramas que llevan hacia los receptores y una copia adicional se devuelve hacia la fuente. Por otra parte, el conmutador absorbe todas las células BRM que le llegan desde las ramas, y genera una célula BRM con el campo ER igual a la tasa mínima de realimentación de todas las células BRM recibidas, o en modo binario, marca el campo de la célula BRM generada, si alguna de las células recibidas lo tiene activado. Este algoritmo presenta problemas de consolidación, si dos o más células FRM se reciben consecutivamente.

Para evitar este problema de consolidación, una simple mejora consiste en introducir unos contadores para controlar el orden de llegada de las células y evitar que el retardo en la llegada de información de realimentación se traduzca en información de realimentación errónea. Por ejemplo, la propuesta de Yokotani [Yokotani95] introduce números de secuencia en las células RM y temporizadores para crear una tabla de estado de cada puerto de la conexión, y determinar cual de ellos sufre congestión. La principal desventaja de este trabajo es la complejidad operacional.

2.4 CONCLUSIONES

Las redes IP y ATM no fueron creadas para soportar comunicaciones multipunto, por lo que es preciso extender y diseñar nuevos protocolos que permitan este tipo de conexiones. Sobre Internet, IPMulticast proporciona toda una serie de procedimientos para el direccionamiento, la gestión y el enrutamiento de los paquetes hacia el grupo multipunto. Los protocolos de transporte se centran en otra serie de tareas como la fiabilidad, el control de flujo y congestión, la recepción ordenada y la gestión de grupo, etc., en función de las características de las aplicaciones.

La mayoría de los protocolos de transporte envían confirmaciones negativas para asegurar la fiabilidad y la escalabilidad; y resuelven de formas diferentes los posibles problemas de sobrecarga de realimentación. En cuanto al control de congestión, existen dos grandes divisiones según el número de flujos que transmite la fuente:

unitasa y multitasa. Otra importante clasificación es cómo regulan la velocidad del emisor mediante tasa o ventana.

ATM soporta conexiones punto a multipunto a partir de circuitos virtuales, y cuenta con un protocolo de señalización para la creación de grupos dinámicos. ABR (*Available Bit Rate*) es una de las categorías de servicio que ofrece ATM que desarrolla un control de flujo para regular la tasa de la fuente. Este mecanismo tiene que extenderse para ofrecer comunicaciones punto a multipunto. Existen diferentes propuestas que tratan de modificarlo con diferentes complejidades y eficiencias.

CAPÍTULO 3

CONTROL DE CONGESTIÓN MULTIPUNTO SOBRE LA CLASE DE SERVICIO ABR

3.1 INTRODUCCIÓN

La clase de servicio ABR reparte los recursos de forma equitativa entre las diferentes conexiones que compiten por ellos, garantiza una tasa mínima (MCR), que puede ser diferente de cero, y minimiza la tasa de pérdidas de células, a partir de un control de flujo que periódicamente determina a qué velocidad debe transmitir la fuente. Este servicio es adecuado para el transporte de datos, y con la apropiada implementación y selección de parámetros es capaz de soportar tráfico en tiempo real.

Diferentes autores han desarrollado propuestas de algoritmos para extender el control de flujo de ABR para conexiones punto a multipunto. En este caso, los conmutadores deben replicar las células a cada una de las ramas de las que cuelgan miembros del grupo. También, es necesario el desarrollo de ciertos mecanismos que limiten y agreguen el tráfico de realimentación, ya que esta información no debería crecer de forma proporcional al número de receptores.

En conexiones punto a punto, la fuente transmite a la velocidad mínima que pueden soportar todos los conmutadores en el camino desde el origen hasta el destino para garantizar una tasa de pérdidas mínima [UNI4.0]. En el entorno multipunto, esta estrategia se puede extender, de manera que la fuente transmita a la tasa mínima que puedan soportar todos los conmutadores que se encuentren en el camino desde la fuente hasta cada uno de los receptores. De esta manera, se garantiza una probabilidad de células perdidas mínima. Sin embargo, otro tipo de estrategias también son utilizadas [Jiang98b].

Algunos de los algoritmos de consolidación propuestos más interesantes son [Roberts94b], [Yokotani95], [Siu95], [Ren96], [Cavendish96], [Moh97], [Cho97], [Moh98], [Jiang98a] y [Ren98]. A grandes rasgos estos mecanismos difieren sobre qué dispositivo y en qué instantes se generan las células BRM consolidadas.

El primer algoritmo fue publicado por L. Roberts, [Roberts94b], y está basado en el algoritmo EPCRA [Robert94a] [Siu96]. Este algoritmo presenta problemas de consolidación, debido a problemas de sincronización o pérdidas de células BRM.

Las propuestas de [Siu95] y [Ren96] solucionan este problema a partir de una variable indicadora de que ha llegado una nueva célula desde las ramas. De manera que solamente se genere la célula BRM consolidada, si ha llegado nueva información de realimentación. En [Cho97] se propone que solamente se envíe la célula BRM consolidada, si desde todas las ramas se ha recibido una nueva célula BRM. Sin embargo, la respuesta de este algoritmo puede ser demasiado lenta para adaptarse a las condiciones de la red.

La propuesta de [Yokotani95] introduce números de secuencia en las células RM y temporizadores en los conmutadores, para crear una tabla de estado de cada puerto de la conexión y determinar cual de ellos sufre congestión. La principal desventaja de este trabajo es la complejidad operacional.

Los siguientes trabajos [Cavendish96], [Jiang98a] y [Moh97] utilizan temporizadores en los conmutadores, y las células BRM se envían cuando éstos expiran. También, el principal problema de estos algoritmos es su complejidad para implementarlos.

El control de flujo de ABR requiere de la información de retroalimentación, indicativa de las condiciones de la red, para adaptar la tasa de la fuente. Al evaluar estos mecanismos de retroalimentación hay que tener en cuenta muchas variables: la frecuencia y rango de las oscilaciones que determinan la estabilidad y velocidad de reacción ante los cambios en el sistema, la eficiencia en el caudal y la equidad al compartir el ancho de banda disponible entre las distintas conexiones.

Los controles de flujo punto a punto basados en tasa han sido ampliamente analizados en gran cantidad de trabajos. Algunos de ellos estudian su estabilidad y convergencia, [Altman94], [Fendick91], [Izmailov95] y [Keshav91], otros la equidad, [Bonomi95] y [Mukherjee91], otros se centran en modelar el comportamiento dinámico en régimen permanente, [Bolot90], [Ritter95], [Yin94], [Ram95] y [Ohsaki95], y en el transitorio [Cavendish95] [Elwalid95] y [Tipper90].

El objetivo de este capítulo es el estudio y el análisis matemático del control de flujo multipunto sobre la clase de servicio ABR. Primero, se analiza en detalle el algoritmo propuesto por L. Roberts y se propone un algoritmo de consolidación, que evite el problema de consolidación. Después, se modela el comportamiento del control de congestión multipunto, teniendo en cuenta, que se utiliza este algoritmo para la agregación de células de realimentación. Este primer análisis, se desarrolla en base a considerar un modelo continúo, y se resuelve mediante ecuaciones diferenciales. Los resultados obtenidos mediante simulación validarán el modelo. En el apéndice A, se analiza el control de flujo multipunto por medio de sus valores medios y mediante el modelo de fluidos [Solera01].

3.2 ALGORITMO DE CONSOLIDACIÓN DE L. ROBERTS

El primer algoritmo de consolidación de células BRM fue publicado por L. Roberts, [Roberts94b], y está basado en el algoritmo EPCRA [Robert94a] [Siu96]. Cada vez que una celda FRM se recibe, se copia en cada una de las ramas que llevan

hacia los receptores, y una copia adicional se devuelve hacia la fuente. Por otra parte, el conmutador absorbe todas las células BRM que le llegan desde las ramas, y genera una célula BRM con el campo ER igual a la tasa mínima de realimentación de todas las células BRM recibidas, o en modo binario, marca el campo de la célula BRM generada si alguna de las células recibidas lo tiene activado. A continuación se transcribe el pseudo-código:

```
If receive RM (ACR,Forward,ER,CI) from source
   Multicast RM to all branches;
   Set MXR=ER; MXI=CI;
   Send RM (ACR,DIR=reverse,ER=MER,CI=MCI)to source
   MCI=MXI;MER=MXR
If receive RM (ACR,Reverse,ER,CI) from leaves
   MCI= OR (MCI,CI);
   MER= MIN (MER,ER);
   Discard RM cell;
```

Es necesario un registro MER y el bit MCI para cada circuito virtual. Inicialmente MER se inicializa a PCR y MCI a cero. Cada vez que llega una célula BRM se modifican los valores de estas variables. Cuando se recibe una célula FRM, se genera una copia hacia la fuente cuyos campos ER y CI contienen el valor de MER y MCI. En este momento es cuando los valores MCI y MER se actualizan con los valores CI y ER de la célula FRM.

Este algoritmo presenta problemas de consolidación. Si dos o más células FRM se reciben consecutivamente, debido a problemas de sincronización, o pérdidas de células BRM, los valores de MER y MCI no se actualizarán correctamente. La célula BRM generada presentará en sus campos ER y CI, los valores de la célula FRM recibida, y no los resultantes de la consolidación de las células BRM, que provienen de las ramas. De manera que a la fuente le llega información errónea de realimentación. Una configuración, en la que se producen estos problemas, es la que muestra la figura 3.1, donde un enlace troncal de gran capacidad se divide en otras ramas de menor capacidad. La desincronización entre las células FRM generadas por la fuente y las células BRM devueltas desde los enlaces producen problemas de consolidación.

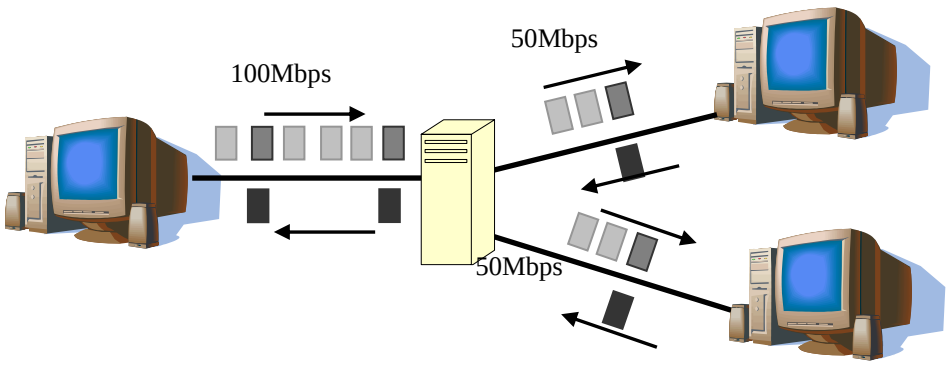

Figura 3.1 Configuración con ruido de consolidación para el algoritmo de L. Roberts

3.3 ALGORITMO DE CONSOLIDACIÓN PROPUESTO

Como se ha visto en el apartado anterior, el algoritmo de L. Roberts presenta problemas de convergencia, para evitarlos se propone el algoritmo Modelo I [Solera00]. Si el conmutador recibe dos o más células FRM sin que ninguna célula BRM haya sido recibida, entonces se genera una célula BRM con el campo CI igual a 1. De manera, que el retardo en la llegada de la información de realimentación de las ramas que están río abajo se modela como estado de congestión. Este algoritmo trabaja con las ramas que se encuentran activas, por lo que aquellas que estén en estado de no respuesta no se tendrán en cuenta.

El algoritmo propuesto converge con la configuración de la figura 3.1. La tasa de la fuente se ajusta a la condiciones de la red cuando un enlace de mayor capacidad se diversifica en otros de capacidad menor. A continuación se muestra el pseudo-código:

```
If receive RM (ACR,Forward,ER,CI) from source
  Multicast RM to all branches;
  Set t_frm_new=time_now();
  Set MXI=CI;MXR=ER;
  t_dif_frm=t_frm_new - t_frm_old;
  t_dif_brm= t_frm_new - t_brm;
  If (t_dif_frm<t_dif_brm) no_arrive_brm=1;
  If (no_arrive_brm) CI=1 else CI=MCI;
  Send RM(ACR, Reserse,MER,MCI) to source
  t_frm_old=t_frm_new;
  Set MCI=MXI;MER=MXR;
If receive RM(ACR,Reserve,ER,CI) from leaves
  Set t_brm=time_now();
  MCI= OR(MCI,CI);
  MER=MIN(MER,ER);
  Discard RM cell;
```

t_frm_new y *t_frm_old* guardan el instante de tiempo en el que el conmutador recibe la célula FRM. *t_brm* almacena el momento de llegada de la célula BRM. *t_dif_frm* contiene el intervalo de tiempo entre dos células FRM consecutivas y *t_dif_brm* es el tiempo entre la llegada de la célula FRM y la BRM. El uso de estas variables determina el orden de llegada. Si el conmutador recibe dos células FRM consecutivas, sin haber recibido ninguna célula BRM, la célula generada llevará el bit CI activado, debido a la falta de realimentación durante este intervalo.

3.4 MODELADO DEL CONTROL DE CONGESTIÓN MULTIPUNTO MEDIANTE ECUACIONES DIFERENCIALES

En este apartado se va presentar un modelo matemático para el control de congestión multipunto, que utiliza como algoritmo de consolidación el propuesto anteriormente. Para ello, se va a determinar la evolución a lo largo del tiempo de la tasa de la fuente, ACR(t), y de la longitud de la cola, Q(t), en estado permanente. Se van a caracterizar estas variables mediante ecuaciones diferenciales y utilizando una aproximación de tiempo continúo.

G. Ramamurthy y Q. Ren han desarrollado un modelo analítico detallado para analizar el comportamiento de un control de congestión de lazo cerrado para comunicaciones punto a punto [Ram95]. El modelo que se presenta a continuación está basado en su trabajo.

3.4.1 El modelo matemático

En la clase de servicio ABR, cada vez que la célula RM llega a la fuente, se actualiza el valor ACR dependiendo de la información que ésta lleve. Si los bits CI y NI no están activados, CI=0 y NI=0, la fuente puede incrementar su tasa en un valor fijo igual a RIF*PCR (*Rate Increase Factor* tiene un valor por defecto de 1 que se negocia durante la conexión). La tasa no puede exceder de PCR.

$$\mathrm{ACR} = \max\left(\min\left(\mathrm{ER}, \mathrm{ACR} + \mathrm{RIF}\cdot\mathrm{PCR}, \mathrm{PCR}\right), \mathrm{MCR}\right) \tag{3.1}$$

Si la célula RM tiene activado el bit NI, NI=1 y CI=0, no se incrementará ACR.

$$\mathrm{ACR} = \max\left(\min\left(\mathrm{ER}, \mathrm{ACR}\right), \mathrm{MCR}\right) \tag{3.2}$$

Si el bit CI está activado, CI=1, la fuente reducirá la tasa ACR en un valor de ACR*RDF (*Rate Decrease Factor* tiene un valor por defecto de 1/32768, que se negocia durante la conexión). La tasa no puede ser menor que MCR.

$$ACR = \max\left(\min\left(ER, ACR - ACR\cdot RDF\right), MCR\right) \tag{3.3}$$

Si la tasa ACR calculada es mayor que la tasa ER del campo de la célula BRM recibida, la fuente reducirá su ACR a ER, siempre que ésta sea mayor que MCR.

Estas ecuaciones van a ser alteradas para que regulen el incremento y decremento de la tasa de la fuente teniendo en cuenta la hipótesis de que la modificación de la tasa de la fuente sólo va a ocurrir en los instantes de generación de la célula RM. Para ello, se definen las constantes AIR (*Additive Increase Rate*) y Mdf (*Multiplicative decrease factor*) como:

$$\mathrm{AIR} = \mathrm{RIF}\cdot\frac{\mathrm{PCR}}{\mathrm{Nrm}} \tag{3.4}$$

$$\mathrm{Mdf} = \frac{\mathrm{Nrm}}{\mathrm{RDF}} \tag{3.5}$$

Donde RIF, RDF, PCR y Nrm son parámetros definidos por la fuente. La constante AIR modela el incremento y Mdf el decremento de la tasa cada vez que se

genera una célula RM. Por lo tanto, las ecuaciones que regulan la tasa de la fuente en función de la información de realimentación quedan:

Si CI=0,

$$ACR = \max\left(\min\left(ER, ACR + Nrm \cdot AIR, PCR\right), MCR\right) \quad (3.6)$$

Si CI=1,

$$ACR = \max\left(\min\left(ER, ACR - ACR * \frac{Nrm}{Mdf} \cdot PCR, PCR\right), MCR\right) \quad (3.7)$$

Si se considera que en la red sólo existen conmutadores binarios del tipo BES, las ecuaciones anteriores se pueden simplificar. Sea t_{i+1} el instante en el cual se genera una célula RM. Si en el intervalo de generación de dos células FRM, se ha recibido una BRM con CI=0, la velocidad de la fuente se incrementará aditivamente:

$$ACR(t_{i+1}) = ACR(t_i) + Nrm \cdot AIR \quad (3.8)$$

Si se recibe una célula BRM con CI=1, la velocidad de la fuente se decrementará de forma multiplicativa:

$$ACR(t_{i+1}) = ACR(t_i) \cdot \left(1 - \frac{Nrm}{Mdf}\right) \quad (3.9)$$

ACR(t) está limitada entre el valor máximo PCR y el valor mínimo MCR. Inicialmente la tasa de la fuente es igual a ICR que se determina en el momento del establecimiento de la conexión.

Para modelar el comportamiento de la tasa de la fuente, ACR(t), y de la longitud de la cola, Q(t), se va asumir un modelo simplificado formado por un solo conmutador, una fuente y N terminales destino. Se va a considerar que el enlace con menor capacidad es el enlace 1 con C1 bps. Se considerará que la fuente es persistente, por lo que siempre tiene células para transmitir y que la longitud de las colas es infinita. En la cola se definen dos umbrales que determinan el grado de congestión del conmutador: Q_H y Q_L. Cuando la longitud de la cola es mayor que el primer umbral, el conmutador está congestionado, si la longitud de la cola cae por debajo de Q_L, el nodo intermedio no está saturado. El tiempo de ida y vuelta entre la fuente y el destino es RTT. La representación de este modelo puede verse en la figura 3.2.

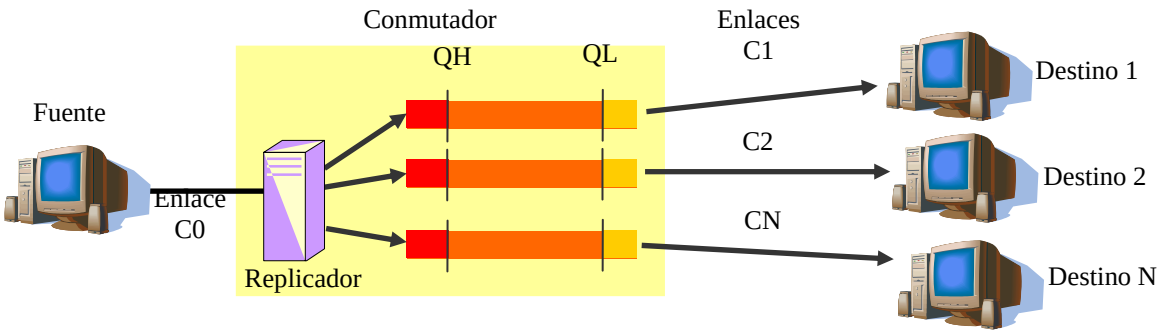

Figura 3.2 Modelo analítico

3.4.2 Análisis del control de congestión multipunto

El estudio del control de congestión se realiza a través de las variables ACR(t) y Q(t). Estas variables muestran un comportamiento periódico, por lo que se van a estudiar durante un ciclo en el régimen permanente. Para facilitar el análisis, se van a diferenciar dos etapas que se modelan de forma independiente. En la primera, el conmutador no está congestionado, la tasa de transmisión se incrementa y las colas están vacías; en la segunda, el conmutador está congestionado, la tasa de transmisión se reduce y las colas están llenas. La figura 3.3 ilustra este comportamiento para la tasa de transmisión.

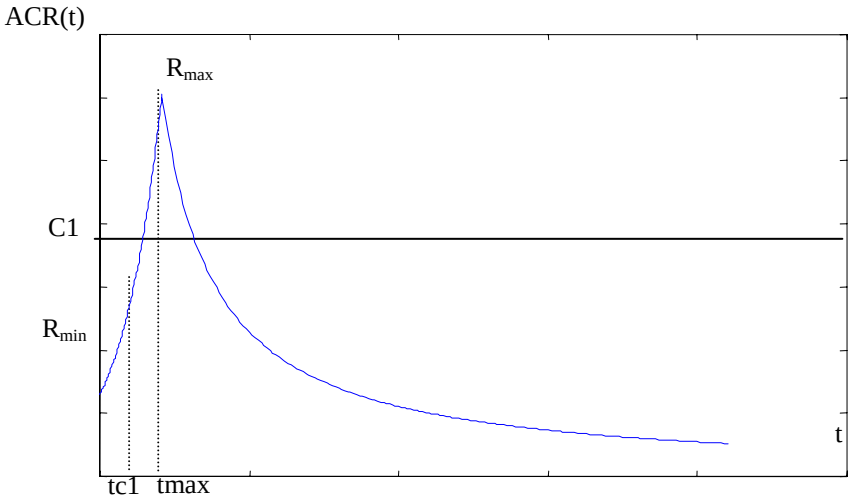

Figura 3.3 Comportamiento de ACR(t) en un ciclo.

3.4.2.1 Modelado de la tasa de la fuente

A continuación se va a analizar el comportamiento periódico de ACR(t) durante un ciclo en régimen permanente.

3.4.2.1.1 Primera etapa: la tasa se incrementa

Durante esta etapa, ACR empieza con el valor mínimo R_{min} y alcanzará su valor máximo R_{max}. El comportamiento de ACR(t) es creciente. Se puede dividir esta fase en dos subetapas: en la primera, la fuente incrementa su velocidad de transmisión en unas circunstancias de baja carga, ACR(t) está por debajo del enlace de menor capacidad, C1, y las colas están vacías; en la segunda subetapa, ACR(t) continúa incrementándose por encima de C1, por lo que la longitud de esta cola comienza a crecer. En esta segunda fase se considera que la red está cargada.

La tasa se incrementa durante el periodo de baja carga

La situación en esta subetapa es la siguiente. La velocidad inicial de la fuente es mínima y está por debajo de la capacidad de los enlaces. Las colas del conmutador están vacías, éste no está congestionado, y las células BRM que envía a la fuente tienen el bit CI=0. Durante este periodo, ACR(t) se incrementa según la ecuación (3.8).

Por otra parte, la velocidad de las células BRM será proporcional a la tasa de la fuente e igual en todos los enlaces.

$$\text{TasaBRM}_{\text{enlaceCi}} = \frac{\text{ACR(t)}}{\text{Nrm}} \tag{3.10}$$

Al dividir la ecuación (3.8) entre el intervalo de tiempo de generación de dos células RM consecutivas, se tiene

$$\frac{\text{ACR}(t_{i+1}) - \text{ACR}(t_i)}{\Delta t_{i+1}} = \frac{\text{Nrm} \cdot \text{AIR}}{\Delta t_{i+1}} \tag{3.11}$$

Si se sustituye en la ecuación anterior, el valor de la tasa de las células BRM, (ecuación (3.10)):

$$\frac{\text{ACR}(t_{i+1}) - \text{ACR}(t_i)}{\Delta t_{i+1}} = \text{Nrm} \cdot \text{AIR} \cdot \frac{\text{ACR}}{\text{Nrm}} \tag{3.12}$$

En el límite cuando Δt_i tiende a cero, ACR(t) puede ser descrita mediante la siguiente ecuación diferencial de primer grado

$$\frac{d\text{ACR(t)}}{dt} = \text{ACR(t).AIR} \tag{3.13}$$

Si se resuelve la ecuación anterior se tiene

$$\text{ACR(t)} = \text{ACR(0)} \cdot e^{\text{AIR}\cdot t} \tag{3.14}$$

Esta subetapa termina cuando ACR(t) alcanza el valor del enlace de menor capacidad C1.

La tasa se incrementa durante el periodo de alta carga

En esta subetapa la tasa de la fuente continúa incrementándose desde el valor del enlace más limitado, C1, hasta alcanzar el valor máximo, R_{max}. La longitud de la cola del enlace 1 empieza a crecer, aunque durante esta etapa no alcanza el valor del umbral, Q_H, que marca el estado de congestión en el conmutador.

La tasa de las células BRM sobre el enlace C1 estarán limitadas por su capacidad y no será proporcional a la velocidad de la fuente.

$$\text{TasaBRM}_{\text{enlaceC1}} = \frac{\text{C1}}{\text{Nrm}} \tag{3.15}$$

En la fuente se generaran células RM cada Nrm/ACR(t). Al conmutador llegan células BRM procedentes de los N destinos a una tasa que depende de la capacidad de los enlaces. Si ACR(t) es menor que la capacidad del enlace i, Ci, la velocidad a la que circulan las células BRM en el enlace viene definido por la ecuación (3.10), en

cambio si ACR(t) es mayor que la capacidad del enlace Ci, la tasa de BRM sobre este enlace vendrá definido por Ci/Nrm.

El algoritmo de consolidación define cual es la política para coordinar la información de realimentación que llega al conmutador. El mecanismo de consolidación propuesto, Modelo I, descrito en la apartado anterior, trabaja de la siguiente manera en esta subetapa. Cada vez que llega una célula FRM se devuelve otra célula BRM con el campo CI=0. Por lo que ACR(t) continúa incrementándose y se describe a partir de la ecuación (3.14).

Esta subetapa termina cuando la fuente detecta que el estado de la red es de congestión.

3.4.2.1.2 Segunda etapa: la tasa se decrementa

En esta etapa, las células BRM llegan a la fuente con CI=1, por lo tanto la tasa se decrementa siguiendo el comportamiento que describe la ecuación (3.9). Al conmutador llegan células BRM procedentes de los N terminales destino con el campo CI función del estado de congestión. El primer enlace que se satura es el enlace 1. En el momento que se recibe la primera célula BRM con CI=1, se actualiza el valor de la variable MCI a 1, y se activa el campo CI de la célula BRM consolidada, con lo que la tasa de transmisión se reduce.

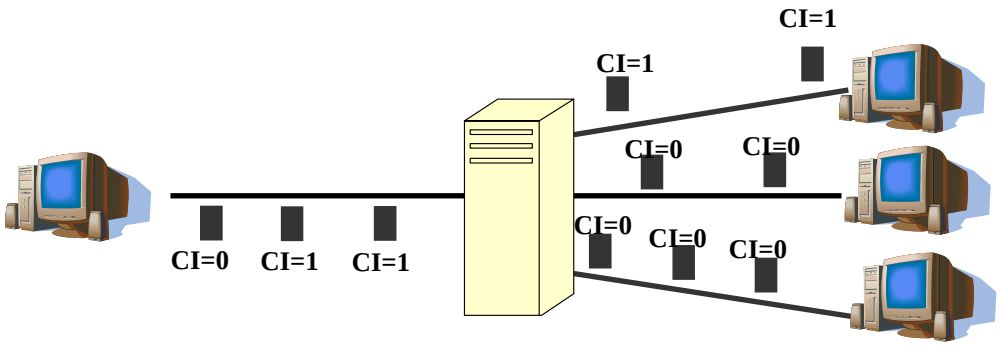

Figura 3.4 Funcionamiento del algoritmo de consolidación en la etapa 2.

En el instante t_{i+1} la tasa de la fuente se reduce multiplicativamente. Si se divide la expresión (3.9) entre Δt_i y se sustituye la tasa de generación de células FRM por Nrm/ACR(t) se tiene

$$\frac{ACR(t_{i+1}) - ACR(t_i)}{\Delta t_i} = -ACR^2(t_i) \cdot \frac{1}{Mdf} \qquad (3.16)$$

Si Δt_i tiende a cero, se obtiene la siguiente ecuación diferencial

$$\frac{dACR(t)}{dt} = -\frac{ACR^2(t)}{Mdf} \qquad (3.17)$$

Resolviendo la ecuación anterior, se tiene el comportamiento de ACR(t) durante la segunda etapa:

$$ACR(t) = \frac{1}{\frac{t}{Mdf} + \frac{1}{R_{max}}} \tag{3.18}$$

Esta etapa termina cuando a la fuente llegan células BRM indicando el fin de la congestión.

3.4.2.2 Modelado de la longitud de las colas

La evolución de la longitud de la cola tiene un comportamiento periódico. Para el análisis sólo se va a tener en cuenta la cola del enlace de menor capacidad, porque ésta es la que gobierna el control de flujo multipunto.

Durante la primera subetapa, ACR(t) es menor que la capacidad del enlace 1, C1, por lo que la longitud de la cola es igual a cero. Cuando la tasa de la fuente supera la capacidad de este enlace, ACR(t)>C1, la cola se incrementa de la siguiente manera

$$Q(t) = \int_{tc1}^{t} \left(R_{min} \cdot e^{AIR \cdot x} - C1 \right) dx \tag{3.19}$$

donde t_{c1} es el instante en el que ACR(t) es igual a la capacidad del enlace 1 (C1).

Durante el periodo de congestión, el comportamiento de la cola puede aproximarse por

$$Q(t) = \int_{t\,max}^{t} \left(\frac{1}{\frac{x}{Mdf} + \frac{1}{R\,max}} - C1 \right) dx \tag{3.20}$$

donde t_{max} es el instante en el que ACR(t) es igual a la velocidad máxima Rmax.

Durante esta etapa la cola se incrementa aproximadamente en

$$Mdf \cdot \left(\log \frac{R_{max}}{C1} - \left(1 - \frac{C1}{R_{max}} \right) \right) \tag{3.21}$$

La cota máxima de la longitud de la cola durante un ciclo puede aproximarse por:

$$Q_H + (R_{max} - C1) * RTT + Mdf \cdot \left(\log \frac{R_{max}}{C1} - \left(1 - \frac{C1}{R_{max}} \right) \right) \tag{3.22}$$

donde Q_H es el umbral superior de la cola que hace que se genere una célula BRM hacia la fuente indicando congestión, el segundo sumando hace referencia al tamaño de la cola cuando todavía no se ha advertido a la fuente de la congestión, y por lo tanto, sigue transmitiendo a tasa máxima, y el tercer sumando es el grado de ocupación en la fase de congestión.

3.5 SIMULACIONES

Para evaluar y comparar el comportamiento del algoritmo de consolidación de L. Roberts y del propuesto en este trabajo, se han realizado simulaciones utilizando el simulador NIST ATM/HFC Simulator Versión 3.0 [Golmie98], elaborado por el *National Institute of Standards and Technology*. Esta herramienta ha sido desarrollada para el estudio de redes ATM y HFC, permitiendo la creación de diferentes topologías de red, el control de los parámetros de los diversos dispositivos, la monitorización de la actividad de la red y el almacenado de los parámetros interesantes en un fichero para su posterior análisis. El simulador se ejecuta bajo plataforma UNIX y está escrito en lenguaje C y X Window System. Dispone de una interfaz gráfica con el usuario sencilla y amigable.

La versión 3 del simulador no soporta conexiones punto a multipunto, por lo que se ha modificado el código fuente para este fin. Con este objetivo, se ha programado un nuevo componente: el conmutador multipunto, capaz de operar con este tipo de conexiones. Esta herramienta desarrolla funcionalidades de duplicación de células de datos y de control FRM hacia las distintas ramas que conforman la conexión multipunto, y de fusión de células BRM para evitar la sobrecarga de realimentación. La tabla de enrutamiento local ha de ser modificada para poder albergar las conexiones multipunto, de manera que indique hacia que puertos de salida se deben redireccionar las copias de la células de entrada. Si el enlace de salida está libre, el conmutador pondrá la célula en el enlace, sí no deberá esperar en la cola ABR. Si se excede el valor de umbral alto de esta cola, el indicador de congestión de este puerto se activará, para reinicializarlo el grado de ocupación de la cola deberá caer por debajo del umbral bajo. Si se excede el tamaño de la cola, las células se perderán. El algoritmo que establece el estado de congestión del conmutador será el EPRCA [Roberts94a].

Los terminales destino se van a comportar como sumideros, no van a generar tráfico de datos, y van a devolver las células RM hacia la fuente, pudiendo modificar sus campos. Su comportamiento al igual que el de todos los componentes de la conexión está recogido en Traffic Management Specification Version 4.0.

Para el estudio del mecanismo de consolidación que se ha descrito anteriormente se ha utilizado la configuración de la figura 3.5. Está formada por una fuente que da servicio a tres terminales destino. Sus capacidades se encuentran descritas en la figura. Se observa que el enlace C1 tiene una capacidad menor que los otros enlaces de la conexión. Esto nos da una idea de donde se encuentran las restricciones del sistema. Se va a estudiar como evoluciona el modelo para longitudes de enlaces diferentes y con conmutadores BES y ERS.

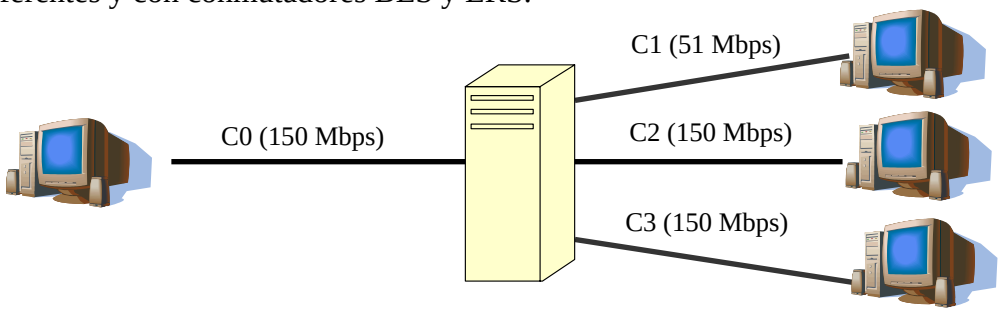

Figura 3.5 Modelo de simulación

3.5.1 Resultados

A continuación se presentan los resultados obtenidos mediante simulación del comportamiento de las variables: ACR y de la ocupación de las colas de los enlaces. El tamaño de las colas de los enlaces de mayor capacidad no tiene ningún interés, ya que van a estar vacías, debido a que la capacidad de esos enlaces es tres veces superior a la tasa de generación de células de la fuente.

En la figura 3.6 y 3.7 se observa la tasa de emisión de la fuente y la ocupación de la cola del enlace más restrictivo en regimen permanente, para conmutadores ERS, que implementan el algoritmo de L. Roberts. Se describen los comportamientos en función de la distancia del enlace, d, entre el conmutador y el terminal. Los umbrales de congestión, utilizados por el algoritmo EPRCA, y que se han escogido para todas las simulaciones son DQT=500 y Q_H=Q_L=100. (DQT es el umbral que indica congestión severa).

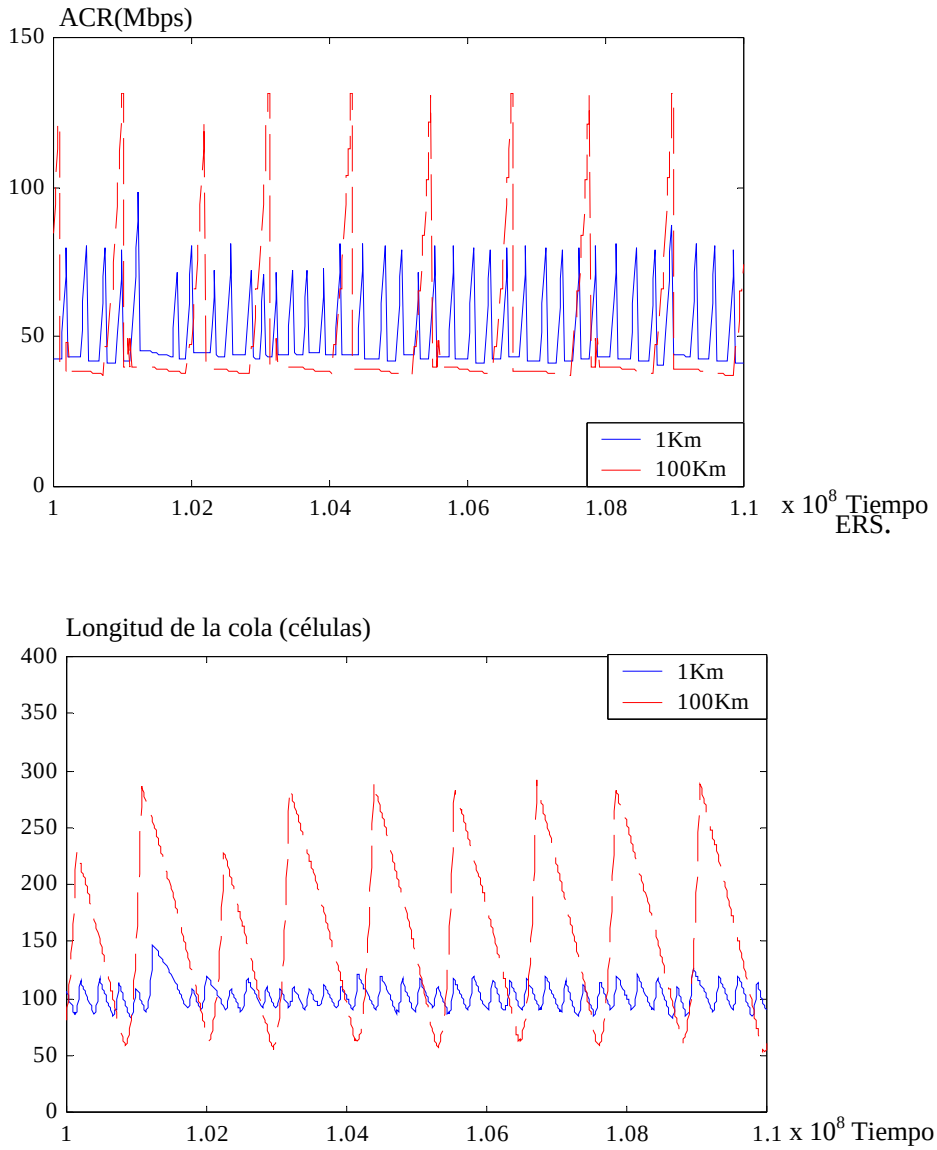

Figura 3.7. Efecto del retardo de propagación en Q(t) en conmutador ERS.

Para la convergencia del sistema la sintonización de los parámetros debe ser muy cuidadosa. De hecho, el valor del parámetro MRF *(Major Reduction Factor)*, utilizado en el algoritmo EPRCA, que asegura la convergencia es 0.5, más restrictivo que el que recomienda el estándar de 0.95 para conexiones punto a punto.

En general, se observa para todas las simulaciones que al aumentar la distancia de los enlaces, la amplitud y el periodo de oscilación aumenta. Esto es debido a que aumenta el retardo de propagación por lo que la notificación de congestión llega más tarde, lo que resulta en el aumento de la longitud de la cola.

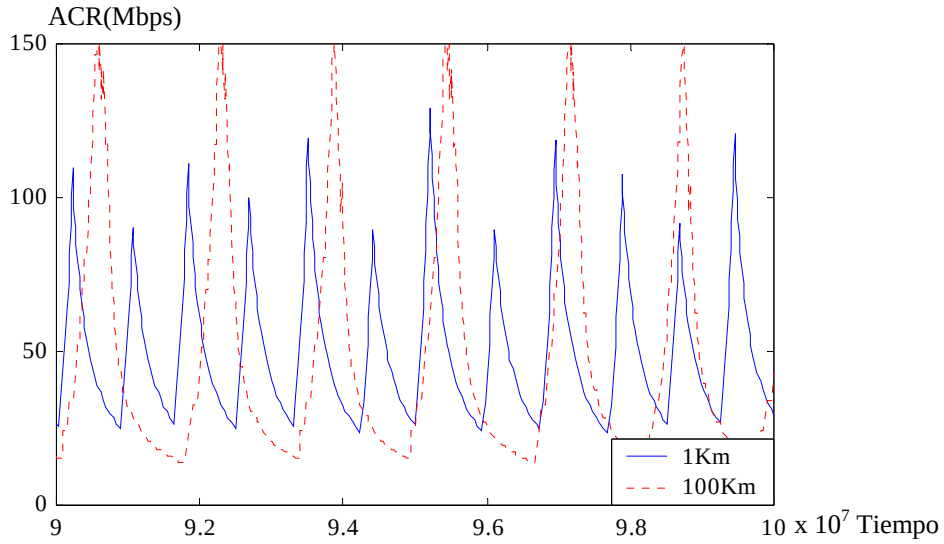

Figura 3.8. Comportamiento ACR(t) sobre conmutador BES.

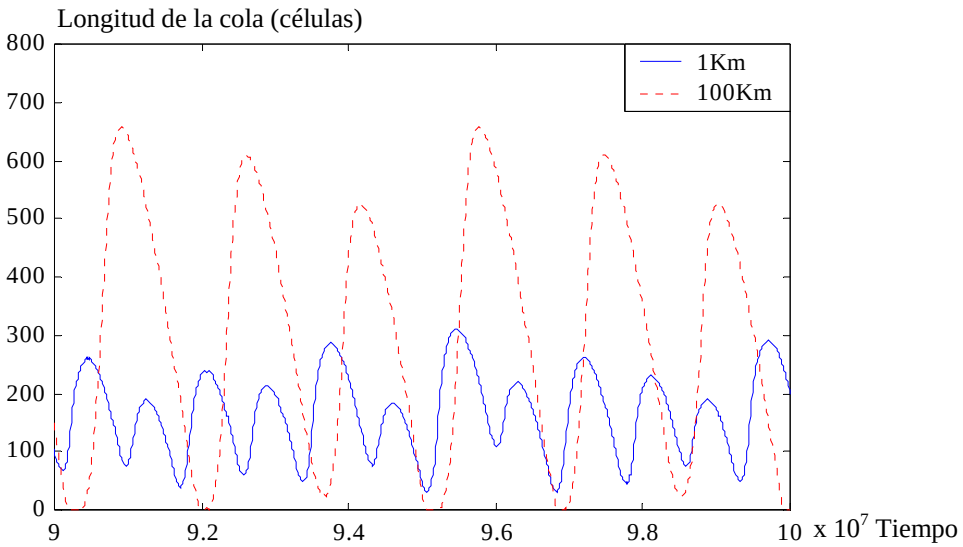

Figura 3.9. Comportamiento de Q(t) en conmutador BES.

En la figura 3.8 y 3.9 se presenta el comportamiento del algoritmo de L. Roberts para conmutadores BES. Comparando estas gráficas con las anteriores se observa

que los conmutadores ERS son mucho más nerviosos, de manera que el seguimiento del estado de la red es más ajustado. Con los conmutadores ERS se consiguen longitudes de ocupación media menores.

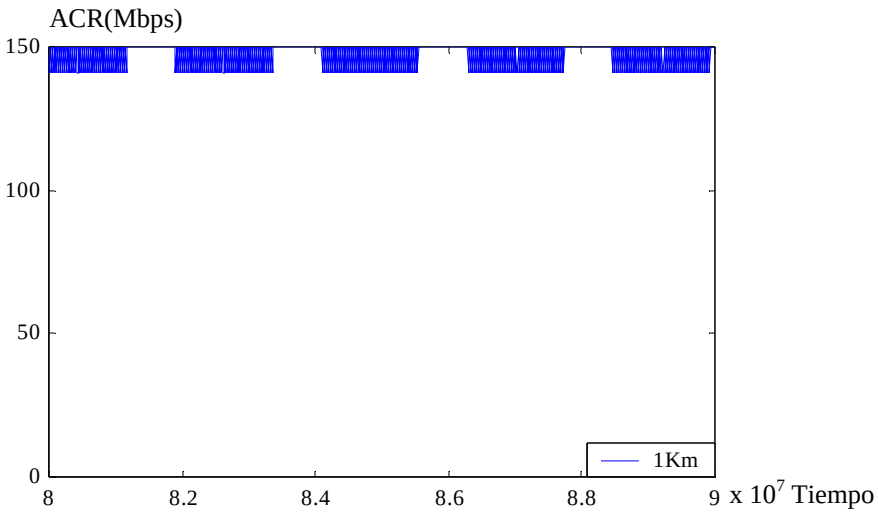

Figura 3.10. Comportamiento divergente de ACR(t) con C0>Ci en conmutador BES.

En la figura 3.10 se muestra el comportamiento del algoritmo de L. Roberts sobre conmutadores BES, cuando la capacidad del enlace troncal es mayor que la de las ramas. Se comprueba que en esta situación el algoritmo no converge, ya que no se consigue ajustar la tasa de emisión a la capacidad del enlace más restrictivo.

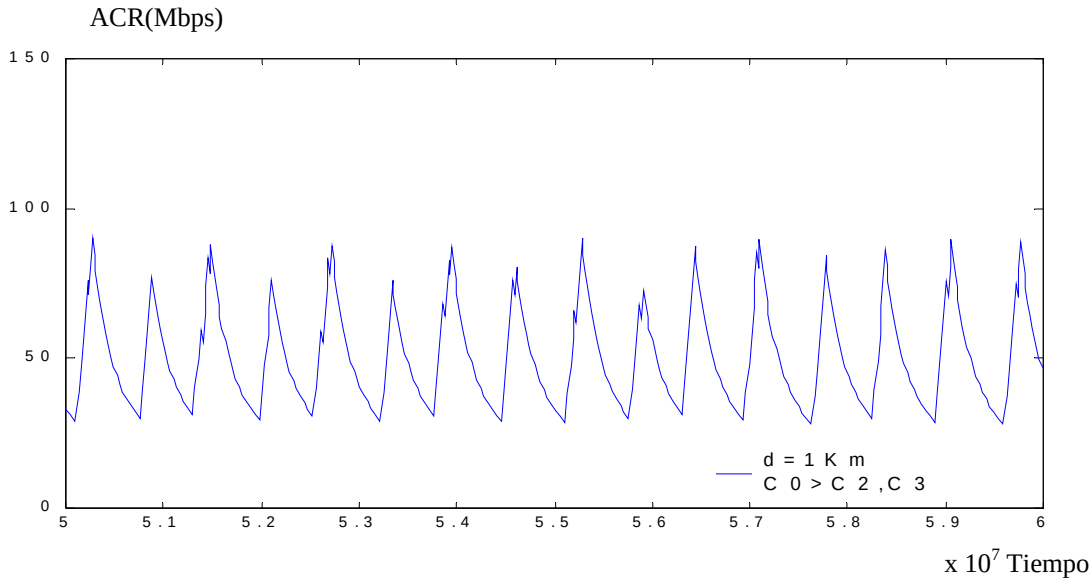

Figura 3.11. ACR(t) para Modelo I con C0>Ci en conmutador BES.

En la figura 3.11 y 3.12 se presenta el funcionamiento del sistema para el algoritmo de consolidación propuesto sobre conmutadores BES cuando C0>Ci. Se observa, que a diferencia con la implementación anterior, el sistema converge. El

comportamiento de este algoritmo, en los otros escenarios, coincide con el anterior para la configuración de la figura 3.5.

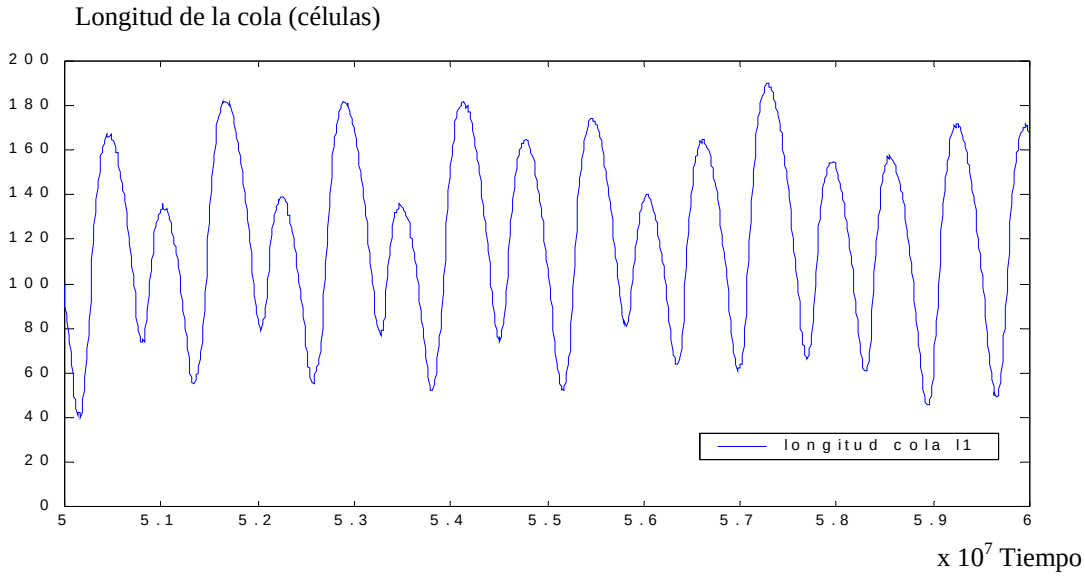

Figura 3.12. Q(t) para Modelo I con C0>Ci en conmutador BES.

3.6 CONCLUSIONES

En este capítulo se ha estudiado el soporte de conexiones punto a multipunto sobre la clase de servicio ABR en redes ATM. Para ello, se ha analizado y modelado matemáticamente la recomendación formulada por L. Roberts en 1994, que recoge la extensión del mecanismo de control de flujo utilizado en conexiones punto a punto a conexiones punto a multipunto. En su trabajo propone mantener el comportamiento de los terminales fuente y destino, y modificar el comportamiento de los conmutadores para que sean capaces de fusionar la información de realimentación que proviene de las ramas. Tras un análisis del algoritmo que propone, se observa que bajo ciertas condiciones no converge. El estudio realizado modifica ese algoritmo para evitar el problema de consolidación. Se comprueba mediante simulación y matemáticamente que el algoritmo Modelo I converge en situaciones en las que anteriormente el sistema divergía.

CAPÍTULO 4

DESCRIPCIÓN DEL PROTOCOLO RCCMP

4.1 INTRODUCCIÓN

El protocolo RCCMP [Solera05a] [Solera05b] ha sido diseñado para comunicaciones punto a multipunto por lo que se considera que una sola fuente transmite datos a un conjunto de receptores. Sin embargo, para comunicaciones multipunto a multipunto, donde intervienen varias fuentes, se puede extender la solución RCCMP para cada una de ellas de manera independiente, debido a sus características de baja complejidad y buena eficiencia en el consumo de ancho de banda.

RCCMP puede ser clasificado como un protocolo de transporte multipunto fiable con un control de congestión de tasa única. Todos los receptores que forman parte del grupo multipunto reciben el mismo caudal, que viene controlado por el receptor de menos recursos. Este receptor es denominado el representante. Aunque estos esquemas pueden resultar ineficientes debido a que un receptor puede reducir drásticamente el flujo de datos a todo el grupo, introduciendo mecanismos que limiten la tasa mínima de transmisión, se pueden solventar estas situaciones.

Entre la fuente y el representante se establece un lazo cerrado para controlar una ventana de transmisión, que ajusta la tasa del emisor al grupo multipunto. Este control de congestión intenta emular al máximo posible al de TCP. De esta manera, se puede conseguir un reparto del ancho de banda de manera equitativa entre el protocolo multipunto propuesto y TCP. Cumplir este requerimiento es muy importante para implantar RCCMP en un entorno como Internet.

Otros de los requisitos que han de cumplir los protocolos de transporte multipunto son la fiabilidad y la escalabilidad. Aunque existen propuestas que utilizan confirmaciones positivas desde todos los receptores para asegurar la fiabilidad, estos esquemas presentan importantes problemas de escalabilidad. En general, la mayoría de los protocolos de transporte multipunto usan el mecanismo de confirmaciones negativas para controlar la correcta recepción de los paquetes. Aunque estos esquemas aumentan la escalabilidad de los protocolos multipunto, también presentan sus propios problemas. Uno de los más importantes es limitar el

número de mensajes de realimentación cuando un paquete se pierde para un gran número de receptores.

El protocolo propuesto utiliza las confirmaciones negativas para asegurar la fiabilidad; por lo tanto, se engloba dentro de estos últimos. RCCMP diferencia el mecanismo para recuperar un paquete en función del receptor que lo pierde. Si es el representante, el emisor detecta la pérdida a través de los mecanismos del control de congestión (duplicación de confirmaciones positivas o expiración del temporizador de retransmisión), y reenvía el paquete. Si cualquier receptor que no es el representante pierde un paquete, debe esperar a recibir la confirmación positiva desde el representante, antes de enviar un NAK solicitando la retransmisión del paquete. De esta manera, el protocolo minimiza el número de NAKs generados.

Sin embargo, si un paquete es perdido por un conjunto de receptores que no incluye al representante, se producirá una implosión de NAKs. Para evitarlo, RCCMP incluye un esquema de temporizadores para suprimir esa respuesta. Así, antes de enviar un NAK, el receptor deberá esperar un tiempo aleatorio. Si recibe la retransmisión del paquete antes de que se agote el temporizador, el NAK se cancelará. En caso contrario, se enviará. Estos mecanismos han sido profundamente estudiados y mejoran la escalabilidad de los protocolos, reduciendo el número de paquetes de realimentación que se transmiten.

Por otro lado, RCCMP es un protocolo no orientado a conexión, por lo que carece de los procedimientos para establecer y liberar la conexión. Un protocolo multipunto que esté diseñado para transmitir datos a miles de receptores, no puede establecer conexiones con cada uno de los miembros del grupo multipunto si pretende ser escalable.

Este capítulo tiene como objetivo la descripción del protocolo de transporte multipunto fiable RCCMP. Para ello, el capítulo se estructura de la siguiente manera. Primero se detalla el proceso de selección de representante. Segundo, se exponen los criterios que el protocolo utiliza para comparar receptores. En tercer lugar, se explica el control de congestión, y se calcula el tiempo de espera del temporizador de retransmisión. A continuación, se describe el control de errores, y el mecanismo que utiliza el protocolo para la supresión de la información de realimentación. Por último, se detallan los formatos de los paquetes. Además, en el apéndice B se especifica el protocolo utilizando un lenguaje de alto nivel, y en el apéndice C se evalúa su complejidad.

4.2 DESCRIPCIÓN FUNCIONAL DEL PROTOCOLO

En esta sección se van a describir los mecanismos que intervienen en el protocolo RCCMP. Este protocolo de tasa única ha sido diseñado para ser simple, fiable, escalable y equitativo, por lo que los distintos procesos han sido desarrollados para cumplir estos objetivos. Al ser un protocolo de tasa única, todos los receptores reciben la misma tasa, independientemente de sus recursos, que vendrá determinada por el receptor con peor conexión de red. Así, es necesario establecer el mecanismo para elegir al representante entre todos los miembros del grupo multipunto. Entre este receptor y el emisor se crea un lazo cerrado que regula la tasa de la fuente por medio de un mecanismo de ventana deslizante. Este control de congestión también tendrá que ser especificado. Finalmente se describirá el procedimiento de control de

errores que debe proveer fiabilidad y escalabilidad al protocolo. Todos estos procesos serán ampliamente detallados en los siguientes apartados.

4.2.1 Selección del representante

Para cualquier protocolo multipunto es muy importante determinar cuál es la tasa a la que debe transmitir la fuente. Generalmente, en protocolos donde se define una única tasa para todo el grupo, ésta coincide con la del receptor con menos recursos. Algunas propuestas, como [Widmer01b] y [Rizzo00], utilizan uno o varios receptores como representantes del grupo multipunto para controlar la tasa. Otras propuestas, como [Lui03], permiten que cualquier receptor envíe paquetes de realimentación a la fuente utilizando mecanismos de supresión efectivos. RCCMP es un protocolo que se engloba dentro de los primeros.

Es propio de los protocolos multipunto el problema de la multiplicidad de caminos de pérdida (*Loss Path Multiplicity*) [Bhatta99], que consiste en que la fuente sobrereaccione a pérdidas que son detectadas por varios receptores. Esto se debe a que no se realiza una correcta agregación o filtrado de los mensajes de realimentación, que llegan desde los receptores que han detectado congestión. En protocolos en los que se selecciona a un receptor como representante, por ejemplo RCCMP, este problema se solventa, porque el control de la tasa solamente se realiza en función de los paquetes de realimentación que le llegan desde este receptor.

Como tanto las condiciones de la red como los miembros del grupo multipunto pueden variar a lo largo de la sesión, RCCMP necesita determinar cuál es el receptor más lento de entre todo el grupo multipunto en cada momento.

A continuación se describe el mecanismo para la selección de representante que desarrolla el protocolo propuesto. Se diferencia una primera etapa al comienzo de la comunicación, cuando no ha empezado la transmisión de paquetes y todos los receptores resultan homogéneos; y una segunda etapa durante la sesión, donde los receptores ya contabilizan paquetes perdidos y se observa un escenario heterogéneo.

4.2.1.1 Proceso de inicialización

Para comenzar la sesión es necesario un proceso de inicialización que encuentre al representante más lento entre todos los miembros del grupo. Como en un primer momento todos los receptores tienen las mismas condiciones, porque no se ha comenzado la transmisión, encontrar al representante se traduce en seleccionar al receptor que se encuentre más alejado de la fuente. Así, se asegura que el representante tendrá el mayor retardo.

Entonces, el transmisor solicita a todos los receptores, mediante un paquete de control de *petición de informe de congestión* (PIC), que respondan inmediatamente mediante otro paquete de control de *informe de congestión* (IC). Este paquete respuesta incluye el número de secuencia del último paquete de datos recibido y la tasa de pérdidas. En el proceso de inicialización estos campos son iguales para todos los receptores. Por ello, al recibir el primer paquete respuesta, se selecciona como representante al receptor que ha enviado este mensaje. Este primer informe proviene de uno de los receptores más cercanos a la fuente. A partir de este momento se inicia un periodo en el que todo informe que llega con las mismas características que el

representante actual, provoca un cambio de representante. A este periodo se le denomina *periodo de gracia*. De esta manera, las nuevas respuestas que llegan a la fuente generan cambios de representante sucesivos, hasta que todos los receptores respondan o finalice este tiempo. Al expirar el periodo de gracia, el último paquete IC recibido determinará el representante elegido para la sesión, que será modificado si cambian las condiciones de la red o los miembros del grupo.

Es importante determinar la duración del periodo de gracia, que viene determinado por el parámetro *grace_period_factor*. Si se escoge un tiempo demasiado corto, puede no seleccionar al receptor con mayor tiempo de ida y vuelta (RTT) como representante al finalizar este periodo. Un tiempo demasiado largo provoca excesivos cambios, que revierten en ineficiencia para la comunicación. RCCMP ajusta este periodo inicial a dos veces el RTT. Como no se dispone de una estimación del RTT al comenzar la transmisión, este parámetro debe inicializarse a un valor que dependerá del dominio de la sesión.

También es importante tener acotado el tiempo de espera entre que se envía un PIC y se recibe la primera respuesta IC, ya que si ésta no se produce, es porque no existe ningún miembro activo en el grupo multipunto. La variable *report_factor* controla este tiempo. Si después de varios intentos de enviar un paquete de control solicitando respuesta, ésta no se produce, la sesión deberá ser cancelada. El número de reintentos se ajusta a partir del parámetro *report_maxtrx_factor*.

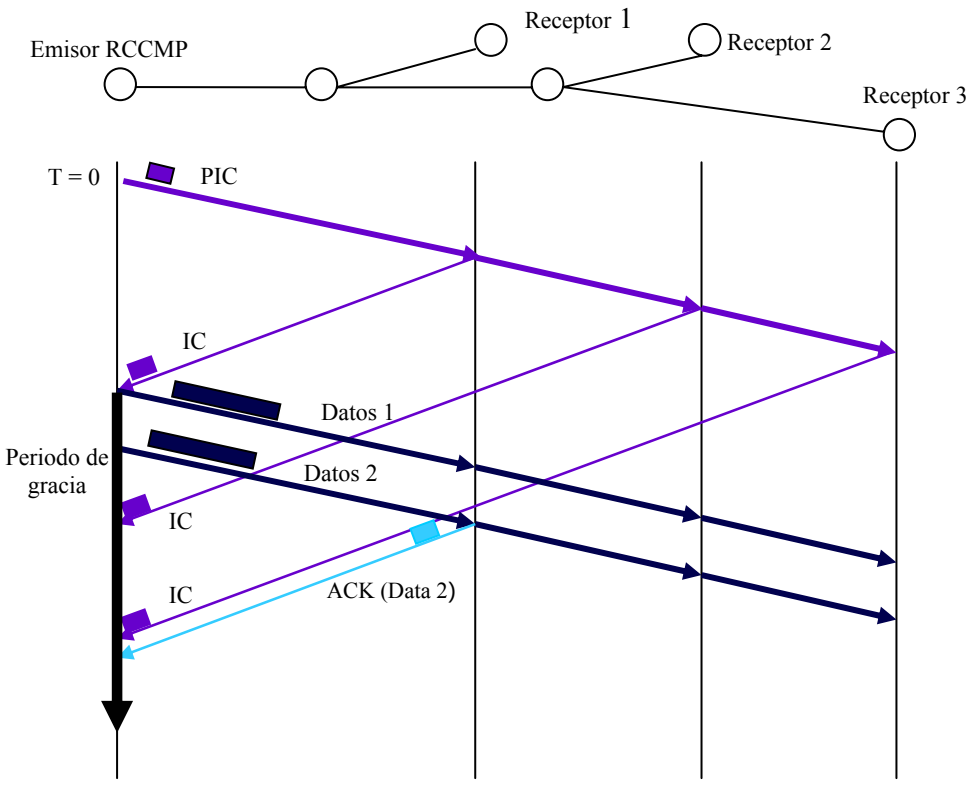

Figura 4.1 Proceso de inicialización de búsqueda de representante.

Como a partir de la recepción del primer informe de congestión se ha seleccionado a un representante, aunque sea de forma transitoria, se inicia la

transmisión de los paquetes. La figura 4.1 ilustra el proceso inicial de selección de representante.

4.2.1.2 Cambio de representante durante la sesión

Durante la sesión, un cambio de representante solamente se produce si existe un receptor con peor conexión de red que él. Cuando un receptor, que no es el representante, detecta la pérdida de un paquete, debe esperar a recibir el mensaje de confirmación para determinar qué receptor debe encargarse del proceso de recuperación de este paquete. Si el receptor que ha detectado la pérdida recibe el ACK confirmándolo, entonces envía un NAK que incluye la tasa de pérdidas, calculada localmente, y la secuencia del último paquete de datos, que utilizará el transmisor para el cálculo del tiempo de ida y vuelta (RTT). La fuente, al recibir este NAK, calcula el caudal de este receptor en base a la ecuación de equilibrio de TCP [Mathis97], y lo compara con el obtenido por el representante. Si el resultado es significativamente peor, el receptor será elegido como nuevo representante. Un ejemplo de este proceso puede verse en la figura 4.2.

En este caso, el grupo está formado por tres miembros, y el representante es el receptor número 3. Éste ha recibido todos los paquetes correctos hasta el 209, que confirma mediante ACKs. El receptor número 1 detecta congestión con la pérdida del paquete 184. Entonces, debe esperar a recibir el ACK de ese paquete, para generar y enviar el NAK. Cuando la fuente recibe la confirmación negativa desde el receptor 1, en el que se han incluido la tasa de pérdidas y el número de secuencia máximo recibido, calcula el caudal estimado para ese usuario y comprueba que es inferior al obtenido para el representante, por lo que debe realizarse el cambio. A partir de que el receptor 1 conozca que es el nuevo representante, comenzará a confirmar los paquetes de datos.

Para evitar un número excesivo de oscilaciones en el protocolo, el proceso de cambio de representante tiene un sesgo a favor del representante actual, y además un tiempo de permanencia mínimo. Este tiempo es controlado mediante un temporizador de duración *minimum_period_factor*. La permanencia mínima en este estado debe ser lo suficientemente corta, para que el protocolo observe los cambios que se producen en las condiciones de la red y en los miembros del grupo, pero a la vez suficientemente larga, para absorber los excesivos cambios que pueden llevar al protocolo a situaciones inestables.

Un cambio de representante también puede tener lugar porque este receptor abandone la sesión. En este caso, el emisor solicita a los receptores que envíen los paquetes de control IC para escoger al nuevo representante. Este informe se puede solicitar a todos los receptores, u opcionalmente, a aquellos que tengan una tasa de pérdidas igual o superior a la indicada en el paquete de control PIC, para evitar que receptores que no sean candidatos a representante respondan.

En cada paquete de datos, la fuente introduce la secuencia del último paquete confirmado. Un representante recién electo debe comenzar a transmitir a partir de ese paquete, no pudiendo usar los ACKs para solicitar la retransmisión de paquetes anteriores al último confirmado. Esto es así, para que paquetes que han sido recibidos y confirmados por el anterior representante, no se vuelvan a retransmitir evitando retransmisiones innecesarias que lleven a comunicaciones ineficientes.

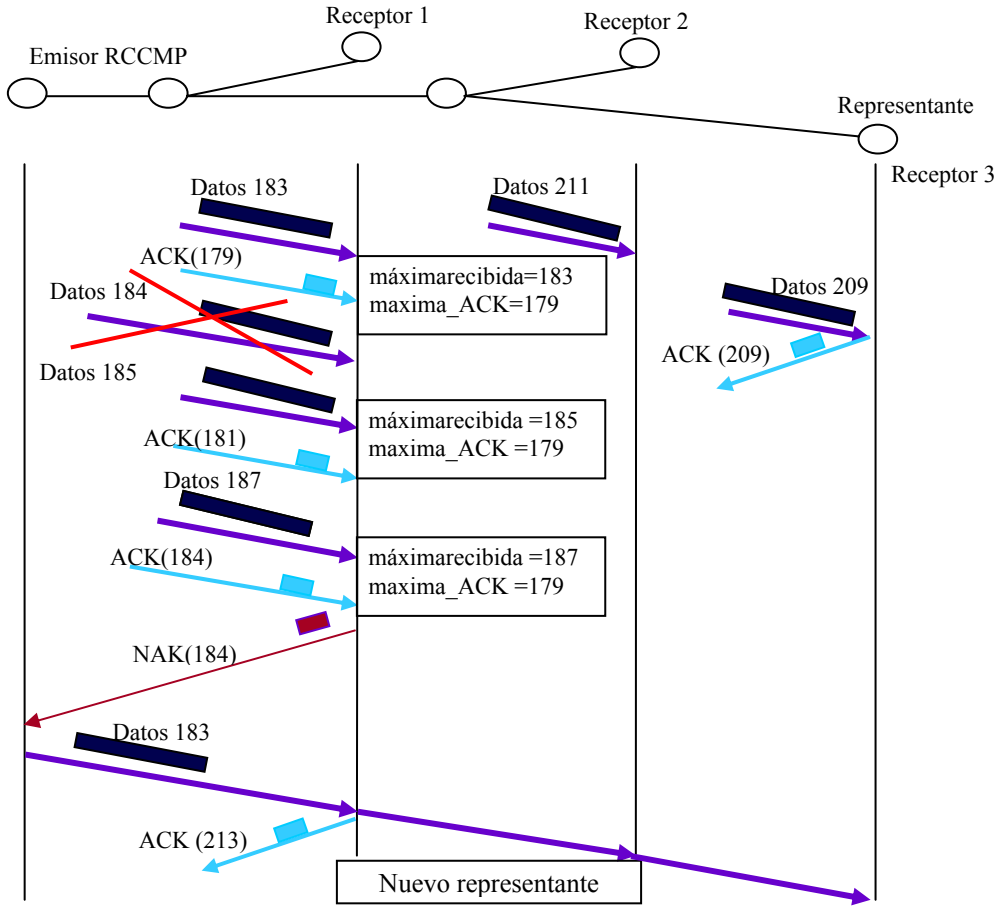

Figura 4.2 Selección de representante durante la sesión.

4.2.2 Comparación entre los receptores

Para conocer qué receptor tiene la peor conexión de red es necesario establecer una serie de parámetros, que permitan calcular y comparar el caudal aceptado por estos receptores. Para la estimación de este caudal se utiliza la ecuación de equilibrio de TCP simplificada [Mathis97], que se muestra a continuación:

$$Caudal = \frac{s \cdot c}{RTT\sqrt{p}} \tag{4.1}$$

donde s es el tamaño de paquete, c es una constante que tiene un valor de $\sqrt{3/2}$, RTT es el tiempo de ida y vuelta ente la fuente y el receptor, y p es la tasa de pérdidas. El peor receptor es aquel que obtiene el menor resultado al computar esta ecuación.

La calidad de un receptor vendrá determinada por los parámetros RTT y tasa de pérdidas. Para evitar las derivas de los relojes de los receptores, el RTT se mide en paquetes. El emisor calcula este parámetro restando al mayor número de secuencia enviada el mayor número de secuencia recibida. La tasa de pérdidas se calcula

localmente en cada uno de los receptores, a partir de un filtro paso bajo de primer orden de ecuación:

$$Y_i = Y_{i-1}W + (1-W)X_i \tag{4.2}$$

donde Y_i representa la tasa actual de pérdidas, Y_{i-1} es la tasa anterior de pérdidas, X_i es una variable booleana que vale 1 cuando el paquete actual no se ha recibido correctamente, y W es una constante cuyos valores están dentro del intervalo [0.05,0.95]. Experimentalmente se ha demostrado que este rango da lugar a un buen comportamiento.

Para descubrir qué receptor debe ser el representante, basta con aplicar la ecuación (4.1). De hecho, no es necesario computar completamente esta ecuación para simplemente comparar las tasas de los receptores. Para facilitar la implementación, RCCMP utiliza la siguiente expresión

$$p_i \cdot RTT_i \cdot RTT_i \qquad i = [0...N] \tag{4.3}$$

donde el *RTT* se mide en paquetes, *p* es el resultado del filtro paso bajo que mide la tasa de pérdidas e *i* identifica a los receptores que han enviado un NAK. (Al utilizar la expresión (4.3), el receptor elegido como representante será el que maximice esta expresión que se corresponde con el caudal mínimo).

Por lo tanto, una vez la fuente ha calculado el caudal para el receptor candidato, lo compara con el obtenido para el representante. Si éste es significativamente peor, entonces se producirá el cambio. Este sesgo a favor del representante es para evitar excesivos cambios, que lleven a una comunicación ineficiente. El parámetro, que controla cuanto peor ha de ser el caudal del receptor que el del representante, es *switch_factor*. Éste se debe ajustar de forma conveniente para captar las condiciones de la red y del grupo de usuarios.

RCCMP utiliza un mecanismo llamado *estadísticas retardadas* que permite comparar la tasa de pérdidas de todos los receptores con la del representante. Cada vez que éste transmite un ACK, se hace de forma multipunto, por lo que llega a todos los miembros del grupo. Al recibir esta confirmación, los receptores actualizan su tasa de pérdidas hasta el paquete confirmado por el ACK. Como estos paquetes incluyen la tasa de pérdidas del representante, todos los receptores pueden comparar su tasa con la de él. Este mecanismo, aunque introduce cierto retardo al reaccionar ante las pérdidas, soluciona el problema de que un receptor sobrereaccione ante pérdidas que también se han producido en el representante. Con las estadísticas retardadas, se puede asegurar, que si un receptor reacciona es porque ha sufrido más pérdidas que el representante.

4.2.3 Control de congestión

Cualquier control de congestión tiene como objetivo evitar saturar la red y conseguir un reparto del ancho de banda de manera equitativa. A la hora de diseñarlo para un entorno punto a multipunto, se deben tener en cuenta dos cuestiones que no se plantean en el entorno punto a punto. La primera hace referencia a la implosión de realimentación, debido a que las señales de congestión pueden llegar desde todos los miembros del grupo multipunto. La segunda se refiere a la sobrereacción de la

fuente, debido a la recepción de estas señales de congestión desde distintos receptores. Así, en multipunto, el control de congestión debe incluir mecanismos para filtrar y agregar estas señales.

RCCMP resuelve las dos cuestiones anteriores, ajustando la velocidad de transmisión del grupo en base a un representante que es el receptor con peores recursos del grupo. A partir de los paquetes de control que envía este receptor, la fuente ajusta la tasa de transmisión.

Por otro lado, RCCMP debe coexistir y repartir el ancho de banda con otros protocolos punto a punto y multipunto. En concreto, debe compartir equitativamente los recursos de red con TCP, el protocolo dominante en Internet. Por ello, el control de congestión de RCCMP imita lo máximo posible el algoritmo estándar de TCP Reno [RFC2581], para asegurar un comportamiento equitativo con respecto a las comunicaciones punto a punto, y por ser la implementación de referencia con la que todas las implementaciones actuales deben compararse. La mayoría de los mecanismos de control de congestión, basados en ventana que se han desarrollado para protocolos de transporte multipunto, usan algoritmos muy similares a éste.

Uno de los puntos que se debe tener en cuenta a la hora de diseñar el control de congestión es como va a responder a los cambios de representante. RCCMP resuelve esta situación, haciendo que el nuevo representante comience a controlar la transmisión a partir del último paquete confirmado por el anterior representante. De esta manera, se consigue un cambio de representante más eficiente evitando retransmisiones de paquetes inútiles.

A continuación se detalla el control de congestión de RCCMP, que tiene lugar entre el representante y la fuente. Para ello, se describen los algoritmos que se ejecutan en cada uno de ellos.

4.2.3.1 El representante

El representante debe comportarse de manera equivalente a un receptor TCP Reno en comunicaciones punto a punto. Al recibir un paquete de datos, se aplica el siguiente algoritmo:

1. Si el número de secuencia es el esperado, se inicializa un temporizador de espera (100 milisegundos es un valor típico de TCP Reno en Internet). Cuando dicho temporizador expira, se envía un ACK de confirmación del paquete recibido; sin embargo, si se recibe un nuevo paquete de datos con el número de secuencia adecuado antes de que el temporizador expire, se cancela el temporizador, y se envía un ACK correspondiente al último paquete recibido.
2. Si el número de secuencia es inferior al esperado, se trata de un paquete de reparación, respuesta a un NAK enviado por algún receptor. Este paquete de reparación no se confirma.
3. Si el número de secuencia es superior al esperado, se envía inmediatamente un ACK que confirme el último paquete recibido con el número de secuencia esperado, para que el emisor sea consciente de que se han perdido paquetes.

Como se puede observar, el algoritmo es el típico de cualquier implementación TCP. RCCMP incluye un retardo al transmitir ciertos ACKs que disminuye el ancho

de banda consumido por el tráfico de realimentación, ya que un único ACK confirma dos paquetes de datos.

4.2.3.2 El emisor

En el emisor RCCMP se aplica también un algoritmo TCP estándar. Sin embargo, a diferencia de este protocolo no divide la información que le entrega la aplicación en segmentos de longitud variable, sino que la encapsula en un paquete de longitud fija. Por lo tanto, para controlar la ventana de transmisión se utilizarán estos paquetes en vez de los segmentos de TCP.

Para poder transmitir un paquete, es necesario que esté dentro de la ventana, es decir, que se cumpla que el número de secuencia de dicho paquete sea menor o igual al último número de secuencia confirmado más el tamaño de la ventana. El protocolo propuesto inicializa la ventana siguiendo las especificaciones de TCP Reno [RFC2414]. El tamaño inicial depende del tamaño de los paquetes:

1. Si el tamaño de paquete es menor o igual que 1095 octetos, el tamaño de ventana inicial es 4.
2. Si el tamaño de paquete está entre 1095 y 2190 octetos, el tamaño inicial de ventana es 3.
3. Si el tamaño de paquete es mayor que 2190 octetos, el tamaño inicial de ventana es 2.

La ventana se incrementa con cada ACK recibido en un número de paquetes que depende de si se encuentra en la fase transitoria *(slow-start)* o en fase permanente *(congestion avoidance)*. En el primer caso, la ventana se incrementa en uno, mientras que en el segundo caso el incremento es igual a la inversa del tamaño de la ventana actual.

RCCMP, al igual que TCP Reno, detecta congestión si un paquete no ha sido recibido. Para ello, cada vez que se envía un paquete, se inicializa un temporizador de retransmisión. Si éste expira sin que se haya recibido la confirmación, o se reciben *ndupack* ACKs consecutivos con el mismo número de secuencia, se detecta congestión (*ndupack* es un parámetro de TCP, cuyo valor estándar es 3).

Si la congestión se detecta mediante la recepción de tres ACKs duplicados, entonces se retransmite solamente el paquete con número de secuencia no confirmado, y el tamaño de la ventana se disminuye a *ssthreshold* más el número de ACKs duplicados recibidos. (*Ssthreshold* es un parámetro de TCP que determina el tamaño de la ventana para pasar de fase transitoria a la permanente y viceversa). En cuanto se recibe un nuevo ACK, el tamaño de la ventana se vuelve a reducir, esta vez hasta *ssthreshold.*

Si la congestión se detecta mediante la expiración del temporizador de retransmisión, el tamaño de la ventana disminuye a uno y se vuelve a la fase transitoria *(slow-start),* retransmitiéndose paquetes desde el primero no confirmado. Al igual que en TCP, RCCMP dobla el valor del temporizador de retransmisión (RTO) cada vez que se produce una retransmisión. En TCP si el temporizador alcanza el valor de sesenta y cuatro segundos y no recibe ninguna respuesta, la comunicación aborta. En comunicaciones multipunto, un elevado número de retransmisiones puede significar que el representante ha abandonado el grupo

multipunto. Por lo que después de *rtos_new_representative_factor* retransmisiones consecutivas, se inicia el proceso de búsqueda de nuevo representante.

Se puede resumir el algoritmo que controla la ventana de transmisión en los siguientes puntos:

1. Inicialización de la ventana según el tamaño de paquete.
2. Se transmiten paquetes hasta que la ventana esté completa. Se inicializa el temporizador de retransmisión.
3. Se espera a recibir ACKs, y se usa su información para actualizar el RTT.

    a. Si el número de secuencia es el esperado, se incrementa el tamaño de la ventana en una cantidad que depende de la variable *ssthreshold*.

    b. Si el número de secuencia no es el esperado, y ya se han recibido *ndupacks* ACKs duplicados, se detecta congestión. Se decrementa el tamaño de la ventana, se retransmite solamente el último paquete no confirmado, y se vuelve al paso 3.

    c. Si no se recibe ningún ACK y el temporizador de retransmisión expira, se duplica el valor del temporizador, se reduce el tamaño de ventana a uno, se retransmiten todos los paquetes desde el último no confirmado y se vuelve al paso 3.

4.2.3.2.1 Cálculo del temporizador de retransmisión

RCCMP, al igual que TCP, retransmite los paquetes de datos que no son confirmados dentro del periodo de retransmisión (RTO). Es muy importante determinar adecuadamente este tiempo para evitar retransmisiones innecesarias, y así mejorar el rendimiento del protocolo. El RTO se calcula estimando la media y la varianza del RTT entre la fuente y el representante. Algunas implementaciones de TCP hacen un cálculo de RTT por ventana. Esto es adecuado para ventanas pequeñas, alrededor de unidades de segmentos, sin embargo resulta un cálculo muy poco preciso para ventanas mayores. En estos últimos casos, es preciso el cálculo del RTT cada vez que llega un ACK. RCCMP utiliza un cálculo de RTT basado en las extensiones de TCP [RFC1323] para mejorar el rendimiento.

Para poder inicializar adecuadamente el temporizador de retransmisión es necesario conocer el RTT que existe entre el emisor y el representante. Para estimarlo, se usa el algoritmo de TCP que utiliza la opción *timestamp,* donde cada paquete de datos es acompañado de una marca temporal, que el representante copia en los ACKs. Al recibir un ACK, el emisor calcula la diferencia entre el instante actual y la marca temporal del ACK, obteniendo una estimación del RTT. Para cada muestra, se aplican las fórmulas de Jacobson [Jacobson90] para el cálculo del RTO.

RCCMP modifica ligeramente la fórmula de Jacobson para el cálculo del RTO, incluyendo una constante igual a uno, para mejorar la equidad entre RCCMP y aquellas implementaciones TCP que no hacen un cálculo muy exhaustivo del RTT

$$RTO = RTT + 4 * varianza\_RTT + 1 \tag{4.4}$$

donde *RTT* y *varianza_RTT* son una media calculada según [Jacobson90]. El análisis de RCCMP mediante simulaciones ha mostrado que si se elimina la constante 1 en el cálculo del RTO, RCCMP tiene una respuesta excesivamente rápida ante la congestión, produciendo situaciones de falta de equidad con TCP.

4.2.4 Control de Errores

Para asegurar la fiabilidad, los protocolos de transporte multipunto no suelen usar un mecanismo de confirmación positiva por parte de cada uno de los receptores, debido a los problemas de escalabilidad que esta solución presenta para un número elevado de receptores [Pingali93]. Generalmente, estos protocolos utilizan un mecanismo de confirmación negativa, mediante el cual un receptor se comunica con el emisor siempre que detecte la pérdida de un paquete. Sin embargo, estos esquemas también presentan tres importantes inconvenientes. Primero, la fuente es incapaz de determinar, con garantía, cuando puede liberar los paquetes transmitidos de la memoria. No existe un mecanismo explicito, en este tipo de protocolos, que determine cuando ejecutar esta acción. De hecho, el trabajo de [Levine96] demuestra que estos protocolos necesitarían un espacio de memoria infinito para trabajar correctamente. Es decir, la fuente debería almacenar en memoria cada paquete que transmite durante toda la sesión a la espera de que un receptor reclame un paquete perdido. La segunda desventaja es que estos protocolos trabajan correctamente solo si la red no reordena los paquetes. El último inconveniente, pero quizás, en el que más esfuerzos se han invertido y más propuestas se han publicado para intentar resolverlo, es el caso en el que se produzcan pérdidas en muchos receptores. Esto puede dar lugar a la transmisión de un alto número de NAKs, generando una sobrecarga de información de realimentación, también conocido como implosión de NAKs.

Evitar dicha implosión es fundamental para el éxito de un protocolo multipunto de transporte fiable. Algunas de las soluciones propuestas requieren de la colaboración de los elementos intermedios de red para la agregación de la información de realimentación, por ejemplo PGM [RFC3208], o para el envío de señales específicas de congestión, por ejemplo los protocolos presentados por [Barcellos05]. Otras propuestas requieren una cierta jerarquización de los receptores como RMTP [Lin96], LMS [Papa98] o MTCP [Rhee99]. Mientras que otros trabajos abogan por utilizar temporizadores como SRM [Floyd99] o MDP [Macker99]. El protocolo presentado en este artículo utiliza un esquema basado en temporizadores, que no necesita la colaboración de los elementos de red ni la jerarquización de los receptores.

El uso de temporizadores para controlar la supresión de la información de realimentación no es la solución más óptima, ya que tanto la colaboración de los elementos intermedios de red como la jerarquización de los receptores asegura resultados idénticos o superiores. Sin embargo, el uso de temporizadores tiene las siguientes ventajas:

1. No implica carga a los nodos intermedios, por tanto, puede desplegarse en cualquier segmento, sin tener que esperar a que se instalen nodos compatibles.

2. Es una solución más robusta que la jerarquización de los receptores, que presenta problemas cuando alguno de los receptores, situados más cerca del emisor en el árbol de jerarquía, abandona súbitamente la comunicación.

3. Es mucho más simple de implementar que cualquiera de las otras dos y no implica comunicación más que entre el emisor y cada receptor, es decir, extremo a extremo.

Sin embargo, también presenta algunos inconvenientes frente a estos esquemas, como mayor latencia y más requerimientos de memoria, y la necesidad de ajustar convenientemente los temporizadores en cada terminal.

En general, el uso de temporizadores garantiza probabilísticamente, que tanto el número medio como el retardo medio de los NAKs van a estar acotados por unos límites manejables por el protocolo, sin incurrir en problemas de escalabilidad. Hay que tener en cuenta, sin embargo, que las garantías probabilísticas no evitan la posibilidad remota de sufrir una tormenta de NAKs, por lo que RCCMP incluye un mecanismo adicional de protección al limitar el número de retransmisiones.

Otro de los criterios a tener en cuenta a la hora de diseñar un protocolo de transporte fiable es el modo de transmisión punto a punto o multipunto de los NAKs. Las propuestas SRM [Floyd99] y MDP [Macker99] utilizan distribución multipunto, mientras que la modificación de MDP propuesta en [Adamson02] y RMTP [Lin96], utilizan transmisión punto a punto. Estos últimos protocolos son adecuados cuanto es posible establecer comunicaciones multipunto entre la fuente y los receptores, pero sin embargo, el camino desde los receptores a la fuente está restringido a conexiones punto a punto.

El control de error diseñado para RCCMP puede resumirse como sigue. Cuando un receptor se da cuenta de que el representante ha confirmado un paquete que no había sido recibido por él, compara su tasa de pérdidas con la del representante, que está incluida en el ACK. Si la primera es significativamente peor, el receptor podrá enviar inmediatamente un NAK a la fuente. Sin embargo, si no se cumple esta condición, se deberá retardar el envío del NAK un tiempo aleatorio que dependerá del temporizador. Si el paquete perdido se recibe antes de que expire el temporizador, entonces el NAK será cancelado, si no será enviado. Si se pierde un paquete que no es recibido ni por el representante ni por el receptor, no se genera un NAK, ya que ese paquete será recuperado a través del lazo de control entre el representante y la fuente. En la figura 4.3 se ilustra este mecanismo.

En este ejemplo, el grupo está formado por tres miembros y el representante es el receptor número 3. En un momento dado, un paquete de datos se pierde para los receptores uno y dos, mientras que llega correctamente al representante. Por lo que éste lo confirma mediante un ACK. Los receptores deben esperar a recibir esta confirmación para activar los temporizadores. El tiempo de espera expira para el receptor 2, y como no ha llegado el paquete de reparación, envía un NAK. La fuente,

al recibirlo, reenvía el paquete solicitado. Como éste llega al receptor 1 antes de que su temporizador expire, el NAK se cancela.

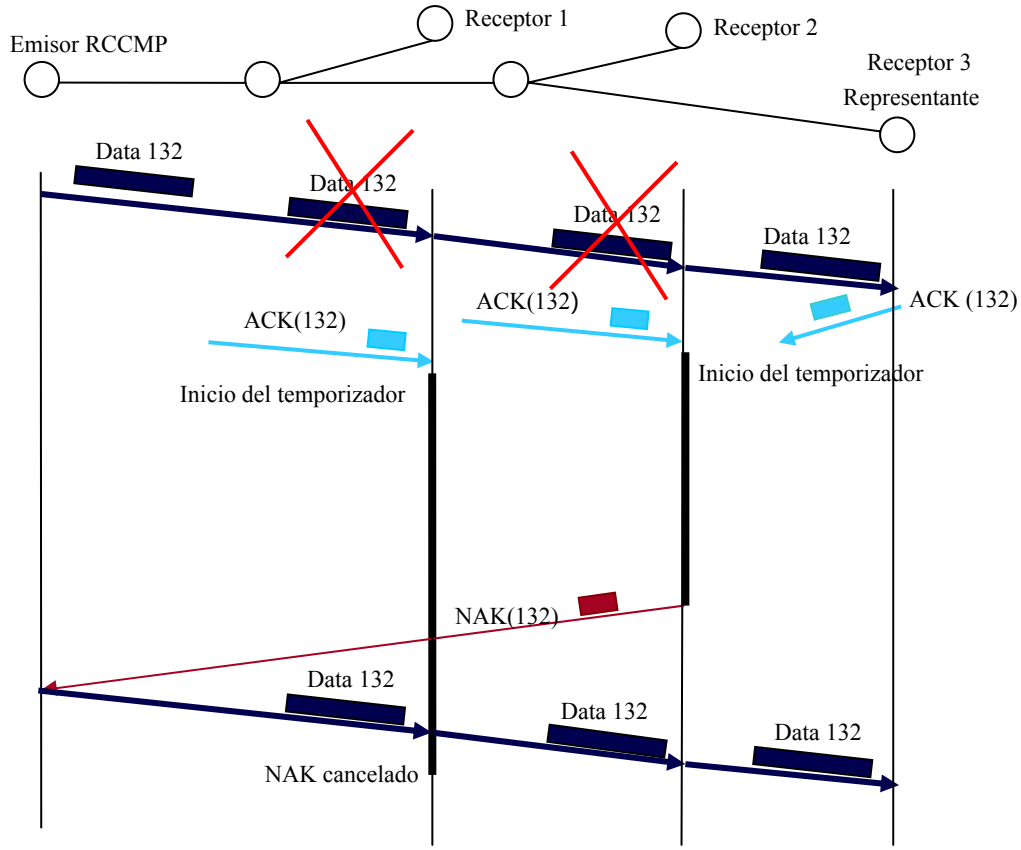

Figura 4.3 Gestión de NAKs.

4.2.4.1 Mecanismo de supresión de NAKs

RCCMP usa, para asegurar la fiabilidad, un mecanismo de confirmaciones negativas. Para garantizar la escalabilidad de este procedimiento, el número de NAKs generado por cada paquete perdido se limita mediante un esquema de temporizadores exponenciales.

El uso de temporizadores para controlar la escalabilidad de protocolos de comunicación punto a multipunto ha sido analizado con detalle en [Nonnen98a], [Nonnen98b] y [Nonnen99]. Este mecanismo se basa en que antes de transmitir una confirmación negativa, el receptor espera un tiempo aleatorio. Si durante este tiempo de espera se recibe el paquete perdido, el NAK puede cancelarse.

El objetivo de todo esquema de temporizadores es, por tanto, reducir al mínimo el número de NAKs enviados, produciendo la cancelación de la mayoría de los mismos. Para ello, los temporizadores deben asegurar que dado un conjunto de receptores que van a enviar un NAK, uno o unos pocos de ellos generen un tiempo de espera muy inferior a la media, de tal manera que el paquete sea retransmitido antes de que la mayoría de los receptores envíen su NAK. De esta manera, se logra la supresión de la información de realimentación, garantizado al mismo tiempo la

fiabilidad. Sin este esquema de temporizadores, el protocolo RCCMP no podría evitar las tormentas de NAKs y, por tanto, difícilmente podría calificarse de escalable.

Sin embargo, para reducir los efectos de una tormenta de NAKs o simplemente para reducir el número de paquetes retransmitidos, que son solicitados por varios NAKs consecutivos, RCCMP reduce el número de retransmisiones. El parámetro *nak_factor* limita el tiempo en el que se puede volver a reenviar un paquete, con lo que por una parte evita que una implosión de NAKs se responda con una implosión de paquetes retransmitidos, y por otra, mejora el rendimiento del protocolo.

El algoritmo para generar y suprimir NAKs en los receptores RCCMP se puede resumir a través de los siguientes puntos:

1. El emisor calcula los parámetros necesarios para ajustar los temporizadores.
2. Una copia de estos parámetros se incluye en cada paquete de datos. Los receptores ajustan sus temporizadores con estos valores.
3. Si un receptor detecta la pérdida de un paquete de datos, compara su tasa de pérdidas con la del representante.
   a. Si es significativamente peor, se transmite un NAK directamente.
   b. En caso contrario, activa el temporizador. Si este temporizador se agota sin recibir el paquete cuya pérdida se detectó, se transmite un NAK.
4. Transcurrido un cierto tiempo, se realiza un nuevo cálculo de los parámetros de los temporizadores.

Para el desarrollo de este algoritmo hace falta ajustar ciertas constantes, por ejemplo cuanto peor debe ser un receptor con respecto al representante para poder transmitir directamente el NAK. El parámetro *automatic_send_factor* se encarga de este ajuste. Un valor elevado inhibe las transmisiones de NAK instantáneas, limitando el número de NAK, sin embargo retarda la respuesta del protocolo. Otro parámetro a ajustar es *feedback_factor* que determina el tiempo necesario para recalcular los valores de los temporizadores. Como se verá en el siguiente apartado, estos valores dependen del número de receptores activos que forman parte de la sesión, por lo que este tiempo debe ser el adecuado para incluir el comportamiento de los receptores.

En cuanto a la función de distribución de los temporizadores, los trabajos de [Nonnen98a] y [Adamson02] demuestran que los temporizadores exponenciales ofrecen una mejor respuesta que los uniformes. Los primeros muestran una menor latencia, incluso si los parámetros de los temporizadores se encuentran ligeramente desajustados. Sin embargo, los primeros protocolos, SRM [Floyd95] y MDP [Macker99], que utilizan estos esquemas para limitar la realimentación, utilizan temporizadores uniformes. RCCMP utiliza aquellos que generan un periodo con distribución exponencial.

4.2.4.1.1 Los temporizadores exponenciales

Un temporizador exponencial es aquel que genera un tiempo de espera con la siguiente función densidad de probabilidad:

$$f(z) = \begin{cases} \frac{1}{e^{\lambda} - 1} \frac{\lambda}{T} e^{\frac{\lambda}{T} z} & 0 < z < T \\ 0 & \textit{en otro caso} \end{cases} \tag{4.5}$$

donde λ y T, el parámetro de la exponencial truncada y el tiempo de espera máximo, son dos parámetros que deben ajustarse de manera adecuada para minimizar el tiempo de respuesta y el número de NAKs enviados. Nonnenmacher and Biersack en [Nonnen98a], [Nonnen98b] y [Nonnen99] han estudiado ampliamente estos temporizadores, y ajustan convenientemente estos parámetros en función del número de receptores de la sesión, R, y del número de NAKs que se desean recibir, N. Este último es ajustado por el programador a través del parámetro *desired_feedback*. Estos trabajos asumen que el RTT es igual para todos los receptores.

En el apéndice D se muestra el desarrollo del cálculo de estos parámetros, en base al estudio de Nonnenmacher and Biersack, modificado para su adecuación al protocolo RCCMP. Aquí se muestran los valores de λ y T:

$$\lambda = ln(R) + 1 \tag{4.6}$$

$$T \approx \frac{RTT}{\ln(N)} (\ln(R) + 1) \tag{4.7}$$

Aunque estos parámetros son función del número de receptores que forman parte del grupo multipunto, no es necesaria una estimación muy precisa por su dependencia logarítmica. Por otro lado, el esquema de temporizadores exponenciales tiene la ventaja de que aunque el número de receptores sea alto, el número de NAK recibidos puede mantenerse acotado sin aumentar el tiempo de respuesta, asegurando la escalabilidad y dinamismo del protocolo.

4.2.4.1.2 La estimación de miembros del grupo multipunto

Para poder calcular los valores adecuados de λ y T es necesario disponer de una estimación del número de receptores. El estimador propuesto por Nonnenmacher y Biersack [Nonnen98a] tiene varios inconvenientes:

1. Se basa en la suposición de que el retardo entre cualquiera de los receptores y el emisor es idéntico para todos los receptores.
2. Presenta un sesgo en la estimación que depende del número de receptores.

Posteriores estudios [Friedman99] han logrado eliminar estos dos problemas. El algoritmo para estimar el número de receptores propuesto en [Friedman99] y adaptado para su uso en RCCMP es el siguiente:

1. El emisor solicita a todos los receptores, mediante un paquete de control PIC, que generen un tiempo de espera usando los temporizadores exponenciales. Cuando expiren, los receptores deben enviar un paquete de control IC, que adjunte una copia del tiempo que se ha esperado antes de enviar dicho informe.
2. En cuanto se recibe el primer paquete de control IC, se extrae el tiempo de espera que contiene dicho informe, y se envía un paquete de datos que incluye este valor.

3. Los receptores, que aún no han enviando su informe, comparan el tiempo de espera generado con el tiempo de espera del receptor que envío el primer informe. Si éste es inferior, mandan inmediatamente su informe, aunque el tiempo no se haya agotado aún; si no, se cancela.
4. El emisor cuenta el número de informes recibidos cuyo tiempo de espera es igual o inferior al tiempo del primer informe y aplica el siguiente estimador [Friedman98]:

$$R' = \alpha\left(r\frac{1 - F_z(m)}{F_z(m + RTT/2) - F_z(m)} + 1\right) + (1 - \alpha)R'' \qquad (4.8)$$

donde *R'* es el número de receptores estimado, α es una constante con un valor de 0.2, *R''* es la anterior estimación del número de receptores, *RTT* es el tiempo de ida y vuelta entre el emisor y el representante, *m* es el tiempo de espera del receptor que envió el primer informe, *r* es el número de receptores que enviaron un informe con un tiempo de esperar inferior o igual al primero (sin contar el primero) y $F_z(z)$ es la función de distribución de los temporizadores exponenciales.

Para indicar a los receptores que es necesario transmitir un informe para estimar el número de receptores o que es necesario cancelarlo en función de su tiempo de espera, se usan unos determinados campos en el paquete de datos.

4.3 DESCRIPCIÓN SINTÁCTICA DEL PROTOCOLO

En esta sección se van a describir los formatos de los diferentes paquetes que intervienen en el protocolo. En RCCMP se pueden distinguir paquetes de datos y control. Entre estos últimos se clasifican a los ACKs y NAKs, y a los paquetes de solicitud de información PIC y de respuesta IC.

4.3.1 Formato de los paquetes

En RCCMP se utilizan paquetes de datos para transmitir información a todos los receptores que forman parte del grupo multipunto. Las confirmaciones positivas se reservan para la comunicación entre la fuente y el representante, y tienen la misma función que en TCP: confirmar que el paquete ha llegado correctamente al destino. Las confirmaciones negativas, NAKs, son utilizadas por los receptores que no son el representante para solicitar la retransmisión de algún paquete perdido. En cuanto a los paquetes de control, la petición de informes de congestión, PIC, se envía desde la fuente hacía los receptores para solicitar la respuesta de éstos como medio de seleccionar al representante o para estimar el número de miembros activos del grupo multipunto. Los receptores responden al paquete de control anterior con un paquete de informe de congestión IC. Todos los paquetes del protocolo de transporte RCCMP viajan encapsulados en un datagrama IP como muestra la figura 4.4.

En principio, el protocolo RCCMP se ha diseñado pensando en comunicaciones punto a multipunto, es decir, considerando solamente una fuente. Sin embargo, la extensión a comunicaciones multipunto a multipunto es posible sin más que extender el protocolo RCCMP a cada uno de esos emisores. La simplicidad de RCCMP hace posible que en un esquema de múltiples transmisores, cada uno de ello actúe de forma independiente. Sin embargo, será necesario identificar de manera conveniente cada una de las sesiones, para esto se utiliza el campo *Identificador de sesión* asociado a todos los paquetes de RCCMP de una misma fuente. Este identificador de longitud 16 bits se genera de manera aleatoria, igual que el número de puerto origen en una aplicación TCP.

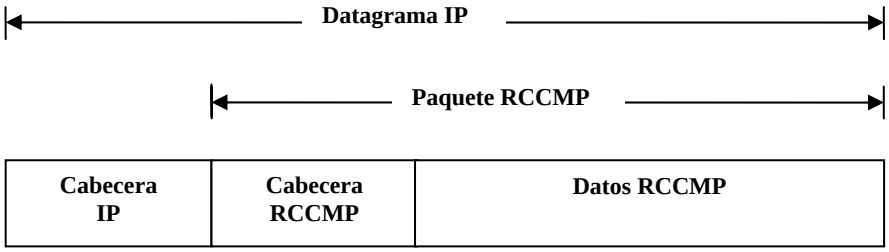

Figura 4.4 Encapsulación de un paquete de datos RCCMP en un datagrama IP.

4.3.1.1 Paquete de datos

El formato de los paquetes de datos de RCCMP es el mostrado en la figura 4.5.

32 24 16	8 0
Identificador de sesión	Dirección IP del representante
Dirección IP del representante	Número de secuencia
Número de secuencia	Número de secuencia confirmado
Número de secuencia confirmado	Marca temporal
Marca temporal	λ
λ	T
T	Comando
Opciones relativas a comando	

Figura 4.5 Cabecera de un paquete de datos RCCMP.

El tamaño de la cabecera es de 32 octetos. Este paquete se transmite mediante mecanismos multipunto. Los campos son:

- Identificador de sesión. Junto con la dirección IP del grupo multipunto, identifica de manera unívoca la aplicación emisora.

- Dirección IP del representante. Junto con el campo anterior permite identificar al representante actual. En el futuro, deberá actualizarse al estándar IPv6.
- Número de secuencia. Representa el número de secuencia del paquete transmitido.
- Número de secuencia confirmado. Indica el último número de secuencia para el que se recibió un ACK, permitiendo así que cuando se produzca un cambio de representante, éste sepa a partir de que paquete debe comenzar a confirmar.
- Marca temporal. Copia del reloj del sistema del emisor en el momento de transmitir el paquete. Al ser copiado en los ACKs, permite estimar el RTT.
- λ y T. Son los valores para los parámetros del mismo nombre en los temporizadores exponenciales.
- Comando. Puede tener uno de los siguientes valores:
  - (0): Con este valor se indica que no se transmite ninguna orden especial.
  - Cancelar respuesta (1): Usado en la estimación del número de receptores, indica a todos los receptores que aquellos que han generado un tiempo de espera superior al especificado en el campo Opciones deben cancelar el envío de IC.
  - Cerrar comunicación (2): Indica que es el último paquete que la fuente desea transmitir.
- Opciones. Este campo tiene un valor que depende del campo Comando.

4.3.1.2 Confirmaciones positivas

Los paquetes de confirmación positiva, ACK, tienen el siguiente formato que se ilustra en la figura 4.6:

32	24	16	8	0
Identificador de sesión			Número de secuencia confirmado	
Número de secuencia confirmado			Tasa de pérdidas del representante	
Tasa de pérdidas del representante			Número de secuencia máximo	
Número de secuencia máximo			Copia de marca temporal	
Copia de marca temporal				

Figura 4.6 Formato de un paquete ACK.

El tamaño del paquete es de 18 octetos y se transmiten mediante mecanismos multipunto. Se compone de los siguientes campos:

- Identificador de sesión. Identifica la aplicación emisora.
- Número de secuencia confirmado. Indica el último paquete recibido sin errores.
- Tasa de pérdidas del representante. Representa la tasa de pérdidas en el representante.
- Número de secuencia máximo. Indica el mayor número de secuencia recibido. A partir de este dato el emisor estima el RTT medido en paquetes, utilizado para comparar receptores.
- Copia de la marca temporal. Representa una copia del campo marca temporal del último paquete recibido correctamente, y sirve para que el emisor pueda estimar el RTT medido en segundos. Se utiliza para actualizar el valor del temporizador de retransmisión.

4.3.1.3 Confirmaciones negativas

La figura 4.7 muestra el formato de las confirmaciones negativas.

32	24	16	8	0
Identificador de sesión			Número de secuencia	
Número de secuencia			Tasa de pérdidas	
Tasa de pérdidas			Número de secuencia máximo	
Número de secuencia máximo				

Figura 4.7 Formato de un paquete NAK

El tamaño total de los paquetes es de 14 octetos y se transmite mediante mecanismos punto a punto desde el receptor hasta el emisor. El paquete se compone de los siguientes campos:

- Identificador de sesión. Identifica la aplicación emisora.
- Número de secuencia. Indica para que paquete se está solicitando retransmisión.
- Tasa de pérdidas. Representa la tasa de pérdidas del receptor que genera el NAK.
- Número de secuencia máximo. Indica el máximo número de secuencia recibido, lo que permite estimar el RTT medido en paquetes.

4.3.1.4 Petición de informes de congestión

Las peticiones de informes de congestión tienen el aspecto que se muestra en la figura 4.8:

32	24	16	8	0
Identificador de sesión			Tasa de pérdidas mínima	
Tasa de pérdidas mínima			Número de secuencia máximo	
Número de secuencia máximo			B	

Figura 4.8 Formato de un paquete PIC

Su tamaño real es de 81 bits, pero para redondear hasta el tamaño estándar de procesado, se transmite como una cabecera de 12 octetos mediante mecanismos multipunto. Se compone de los siguientes campos:

- Identificador de sesión. Identifica la aplicación emisora.
- Tasa de pérdidas mínima. Indica cuál es la tasa mínima de pérdidas para responder con un informe de congestión. Si es 0, todos los receptores deben responder.
- Máximo número de secuencia. Indica cuál es el máximo número de secuencia para responder con un informe de congestión. Si es 0, todos los receptores deben responder.
- B. Es un dígito que si tiene un valor de 0, indica que la respuesta debe ser inmediata. En este caso, se usa para escoger un nuevo representante y al comienzo de la comunicación. Con un valor igual a 1, indica que la respuesta debe esperar un tiempo aleatorio y se usa para estimar el número de receptores.

4.3.1.5 Informes de congestión

Los informes de congestión tienen el siguiente formato (figura 4.9):

32	24	16	8	0
Identificador de sesión			Tasa de pérdidas	
Tasa de pérdidas			Número de secuencia máximo	
Número de secuencia máximo			Temporizador	
Temporizador				

Figura 4.9 Formato de un paquete IC

Tiene un tamaño de 14 octetos y se transmiten mediante mecanismos punto a punto desde el receptor a la fuente. El paquete de control se compone de los siguientes campos:

- Identificador de sesión. Identifica la aplicación emisora.
- Tasa de pérdidas. Representa la tasa de paquetes perdidos del receptor que envía el informe.

- Máximo número de secuencia. Número de secuencia del último paquete de datos recibido por este receptor. El emisor lo emplea para estimar el RTT medido en paquetes.
- Temporizador. Si el informe de congestión es respuesta a una petición de informe de congestión con el bit B a 1, este campo contiene el tiempo que se ha esperado antes de enviar el informe de congestión. En otro caso, este campo contiene 0.

4.4 CONCLUSIONES

RCCMP ha sido diseñado para ser un protocolo escalable, fiable y con un control de congestión de tasa única que comparta el ancho de banda equitativamente con TCP.

El protocolo escoge al receptor con peores recursos del grupo como representante. Para evaluar la calidad de diferentes receptores, RCCMP utiliza las mismas métricas que PGMCC, cuya principal ventaja es que se pueden aplicar directamente a la ecuación de equilibrio de TCP, y por lo tanto, resulta muy fácil comparar el caudal que pueden aceptar distintos receptores. En aras de escoger convenientemente al representante, al comienzo de la comunicación, se incluye un tiempo de espera que permite sesgar el proceso de selección hacia los receptores con mayores retardos.

El control de congestión ha sido planteado como una parte esencial del protocolo y no, como ocurre en muchas propuestas, como un componente adicional que debe ser ajustado a un protocolo de transporte. En RCCMP se combinan los objetivos de regular de la tasa de transmisión y conseguir una comunicación fiable, con el fin de simplificar y limitar el número de confirmaciones negativas que se envían desde los receptores. El control de congestión diseñado emula al máximo posible a TCP para conseguir un comportamiento lo más parecido a él, y cumplir con el requisito indispensable de equidad.

La fiabilidad viene garantizada por la posibilidad de retransmitir los paquetes perdidos mediante confirmaciones negativas. Aunque, el uso de estos mensajes puede provocar efectos no deseados, como las tormentas de NAKs, los temporizadores exponenciales y la limitación del número de retransmisiones en un intervalo de tiempo, aseguran que el protocolo es fiable y escalable. Todo esto sin recurrir a la colaboración de los elementos intermedios de red, y manteniendo la complejidad del protocolo acotada.

CAPÍTULO 5

EVALUACIÓN DEL PROTOCOLO RCCMP

5.1 INTRODUCCIÓN

En este capítulo se muestra y evalúa el comportamiento del protocolo RCCMP. Este estudio se lleva a cabo mediante simulaciones, utilizando el simulador de redes ns-2 (*network simulator*) [ns-2]. A partir de topologías sencillas y adecuadas, se pretende exhibir diferentes aspectos del funcionamiento del protocolo de transporte multipunto.

Los objetivos de este capítulo son los siguientes:

- Analizar la respuesta del protocolo ante diferentes valores de los parámetros de diseño. Algunos de ellos afectan al comportamiento de la fuente y otros al de los receptores. A partir de una topología en árbol de treinta y nueve receptores, se calcula el número de paquetes enviados, retransmitidos y de control en función de estos parámetros. El ajuste correcto de estas variables es muy importante para que el protocolo cumpla con los requerimientos de estabilidad y escalabilidad.
- Estudiar el comportamiento del protocolo cuando funciona de forma aislada. En este caso, se analiza el caudal que inyecta la fuente a la red y el tamaño de ventana, en función del tiempo de ida y vuelta entre la fuente y el representante (RTT). Se modifican los retardos de propagación de los enlaces de los distintos receptores y el tamaño de grupo para comprobar cual es su efecto en el caudal.
- Mostrar el comportamiento del protocolo ante cambios de representante. La aparición de receptores, con recursos más limitados en el grupo multipunto, hace que se genere un cambio de representante, y que el caudal que se transmite se vea afectado. Se caracteriza el tiempo de respuesta del protocolo ante estos cambios. También se ilustra como actúa el protocolo ante el abandono del representante actual. En este caso, la sesión es desatendida por uno de los terminales que controla la ventana de transmisión.

- Estudiar la equidad entre los protocolos RCCMP y TCP. Evaluar y comparar el comportamiento de estos dos mecanismos es muy importante, ya que uno de los principales requisitos, para que un protocolo multipunto fiable sea implementado en Internet, es que sea equitativo con TCP [RFC2357].
- Analizar el comportamiento de RCCMP cuando comparte enlaces con otras instancias del mismo protocolo, para analizar el grado de equidad entre ellas.
- Evaluar la escalabilidad, es decir, como responde el protocolo al aumentar el número de receptores.
- Comparar RCCMP con otras propuestas. Para ello, se clasifica el protocolo propuesto aplicando los criterios del trabajo de Atwood [Atwood04]. El control de congestión se estudia de forma separada, en función de la clasificación que propone J. Widmer en [Widmer01a].

El capítulo se estructura a partir de los puntos anteriormente descritos, comenzando con una breve introducción a la herramienta de simulación.

5.2 EL SIMULADOR

El simulador ns-2 (*network simulator*) [ns-2] es una herramienta para el estudio de redes y protocolos. Este simulador es la principal contribución del proyecto VINT (*Virtual InterNetwork Tested*) [Breslau00], que desarrolla herramientas de simulación para la comunidad investigadora, y protocolos de área extensa en Internet. Uno de sus objetivos, al desarrollar este simulador, es ofrecer a los investigadores un entorno de simulación común, en la que se prueben y evalúen nuevos protocolos y algoritmos.

Algunos de los protocolos y algoritmos que han sido investigados y desarrollados mediante ns-2 son: distintas implementaciones de TCP, disciplinas de colas para nodos intermedios, y protocolos de transporte y enrutamiento multipunto como SRM (*Scalable Reliable Multicast*) [Floyd99] o PIM (*Protocol Independent Multicast*) [Deering96]. También, se han estudiado protocolos de enrutamiento en redes inalámbricas, protocolos a nivel de aplicación, la transmisión de servicios de audio y video con calidad de servicio o la codificación multicapa de servicios en tiempo real.

Ns-2 es una evolución del simulador REAL (*Realistic and Large*) [Keshav88], que también deriva del NEST (*Network Simulation Testbed*) [Dupuy90]. A partir de 1995, el desarrollo de ns-2 lo ha llevado a cabo DARPA a través del proyecto VINT, aunque siempre ha contado con contribuciones de otros programadores.

Ns-2 es un simulador de eventos discretos. Dispone de dos niveles de programación, a bajo nivel se utiliza el lenguaje de programación C++, en el que están programadas las operaciones a nivel de evento, por ejemplo el acceso a tablas de enrutamiento, el envío de paquetes o el desarrollo de los protocolos TCP; a alto nivel se utiliza un lenguaje no compilado como el OTcl (*Object Tcl*) en el que se especifica, por ejemplo, la configuración de las fuentes de tráfico, el modelo de error

de los enlaces, los cambios de enrutamiento o las estadísticas que van a ser estudiadas. Es decir, mientras que con C++ se desarrolla el núcleo del simulador, OTcl controla y configura la simulación.

Ns-2 dispone de una interfaz gráfica para visualizar las simulaciones que es la aplicación Nam [Estrin99]. Esta herramienta también ofrece un editor gráfico, que permite crear escenarios de simulación sencillos, sin tener que programarlos en OTcl.

El protocolo RCCMP se ha implementado en el simulador ns-2. A continuación, y a partir de las simulaciones realizadas, se va a proceder a la evaluación del comportamiento del protocolo de transporte multipunto fiable.

5.3 AJUSTE DE PARÁMETROS

Ajustar convenientemente los parámetros de diseño del protocolo es vital para obtener un comportamiento más eficiente, que minimice los paquetes de control y el número de NAKs, y maximice el número de paquetes transmitidos [Solera5c]. Para ello, se van a estudiar seis parámetros que controlan diversos procedimientos y tiempos del protocolo. Estos parámetros establecen cuando realizar un cambio de representante, el tiempo de permanencia mínimo de un receptor en este estado, el tiempo de espera mínimo para retransmitir un paquete, la frecuencia para estimar el número de receptores activos, el número medio de NAKs que se desean recibir, o cuando enviar un NAK sin esperar un retardo aleatorio.

La topología utilizada para este estudio se muestra en la figura 5.1. Es de tipo arbolada con tres niveles de profundidad. El número total de receptores es de treinta y nueve. Las ramas son todas iguales con un ancho de banda de 500Kbps, un retardo de propagación de 10ms y un tamaño de cola de 3 paquetes. La duración de la simulación es de 100s. Los resultados obtenidos son el promedio de diez realizaciones.

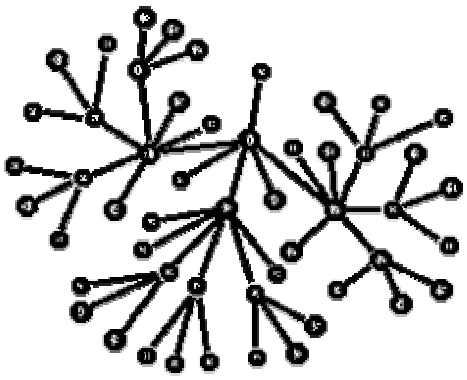

Figura 5.1 Topología para el estudio del ajuste de parámetros de RCCMP.

Los resultados se muestran en forma de gráficos de barras. Se va a representar el número de paquetes totales enviados y reenviados, los paquetes reenviados mediante solicitud por NAK, los paquetes útiles, los NAKs, ACKs y el número de paquetes de control IC. Para facilitar la representación gráfica, se van a codificar estos conceptos

mediante números del 1 al 7. De manera, que en la gráfica 5.2, la abscisa 1 representa el número de paquetes enviados totales, la abscisa 2 el número de paquetes reenviados, y así sucesivamente y siguiendo el orden expuesto.

5.3.1 Switch_factor

Switch_factor es un parámetro propio del emisor que se utiliza para tomar la decisión de si se va a realizar el cambio de representante. Cuando un receptor detecta un paquete perdido, envía un NAK con su tasa de pérdidas y el último número de secuencia recibido. El emisor, a partir de estos valores, calcula el caudal según la ecuación de equilibrio de TCP [Mathis97], y lo compara con el del representante. *Switch_factor* se utiliza en esta comparación. Exactamente, controla cuanto peor debe ser el caudal del receptor que el del representante, para que se realice el cambio. Este parámetro tiene un rango de valores que va desde 0 a 1. Un valor cercano a cero, indica que para que se produzca el cambio, el caudal del receptor debe ser bastante peor que el representante.

Se han realizado pruebas con los valores de *switch_factor* siguientes: 0.5, 0.75 y 1. Los resultados muestran que el número de cambios de representante es de 27.5, 21 y 32.4 respectivamente. Parece lógico que un valor de *switch_factor* de 1 genere más cambios de representante. En cuanto al número de NAKs que llegan a la fuente, es mayor para un *switch_factor* de 0.5, lo que provoca que también el número de paquetes retransmitidos por solicitud de NAK sea mayor. De hecho, en este caso, el 51% de los paquetes retransmitidos son debidos a pérdidas en otros receptores, que no son el representante. Si se utiliza un valor de *switch_factor* igual a 0.75, este porcentaje queda reducido al 23%. A continuación, pueden observarse en la gráfica 5.2 los resultados.

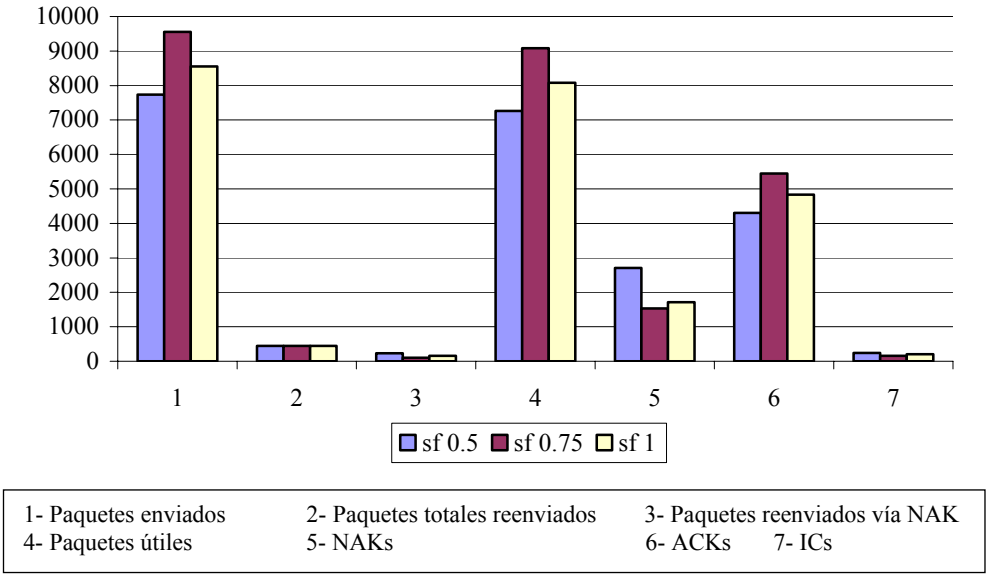

Figura 5.2 Número de paquetes en función del parámetro *switch_factor* (sf).

Un valor de *switchfactor* igual a 0.75 ofrece los mejores resultados, ya que un valor de 1 supone más cambios de representante, que influyen en la eficiencia del protocolo, mientras que un valor de 0.5, hace que la tasa de la fuente no se ajuste con suficiente rapidez a los cambios de la red, y eso pueda llevar a situaciones de congestión.

5.3.2 Nak_factor

Este parámetro, también es propio del emisor, y controla cuanto tiempo debe esperarse para que la retransmisión de un paquete solicitado tenga lugar. Desde que se envió el paquete hasta su retransmisión existe un tiempo mínimo igual a *nak_factor*RTT_representantive* (donde *RTT_representative* es el tiempo de ida y vuelta entre la fuente y el representante). Este periodo intenta evitar que múltiples solicitudes del mismo paquete se respondan con múltiples retransmisiones.

Se han realizado pruebas con los valores de *nak_factor* de 2, 3 y 5. Cuanto mayor es *nak_factor,* mayor es el número de NAKs que se reciben, ya que el paquete de reparación tarda más en llegar, y tienen lugar menos cancelaciones. Esto puede observase en la figura 5.3. Sin embargo, el tráfico retransmitido es menor para un valor de 5. El porcentaje de tráfico retransmitido mediante solicitud de NAK es de 37.8%, 23% y 28% para los valores de *nak_factor* 2, 3 y 5. El valor que se utiliza en el simulador es de 3, ya que como puede observarse en la gráfica se obtienen los mejores resultados.

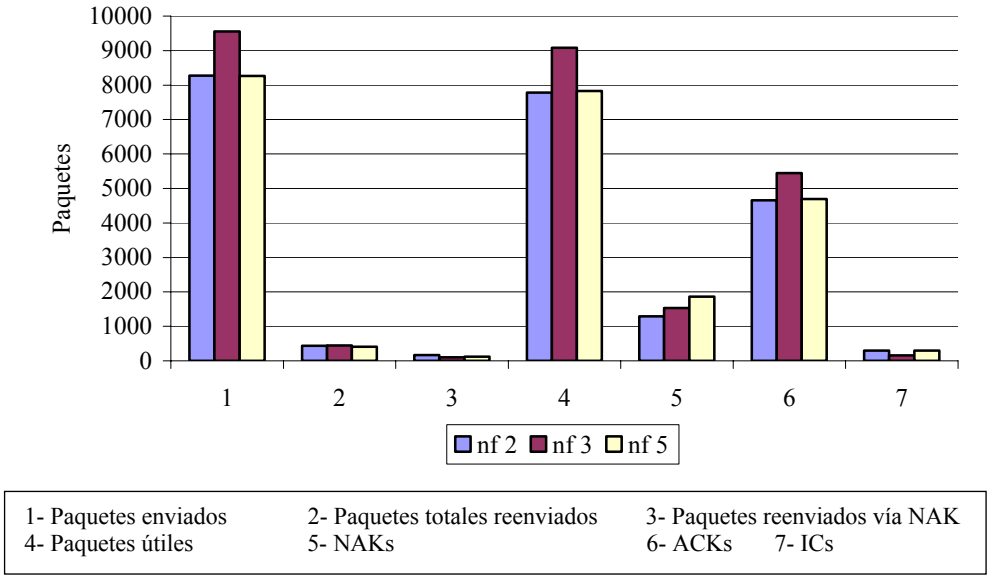

Figura 5.3 Número de paquetes en función del parámetro *nak_factor* (nf).

5.3.3 Minimum_period

Este parámetro hace referencia al tiempo de permanencia mínimo, *minimum_period*RTT_representantive*, del representante en este estado. Los resultados obtenidos son para valores de 0, 3 y 5, y se muestran en la figura 5.4. Como es de suponer, valores más pequeños implican un mayor número de cambios de representante, y un porcentaje de paquetes retransmitidos debido a NAKs menor, porque los cambios se realizan de forma más dinámica. El mayor número de paquetes útiles se obtiene para un valor de *minimum_period* igual a 3.

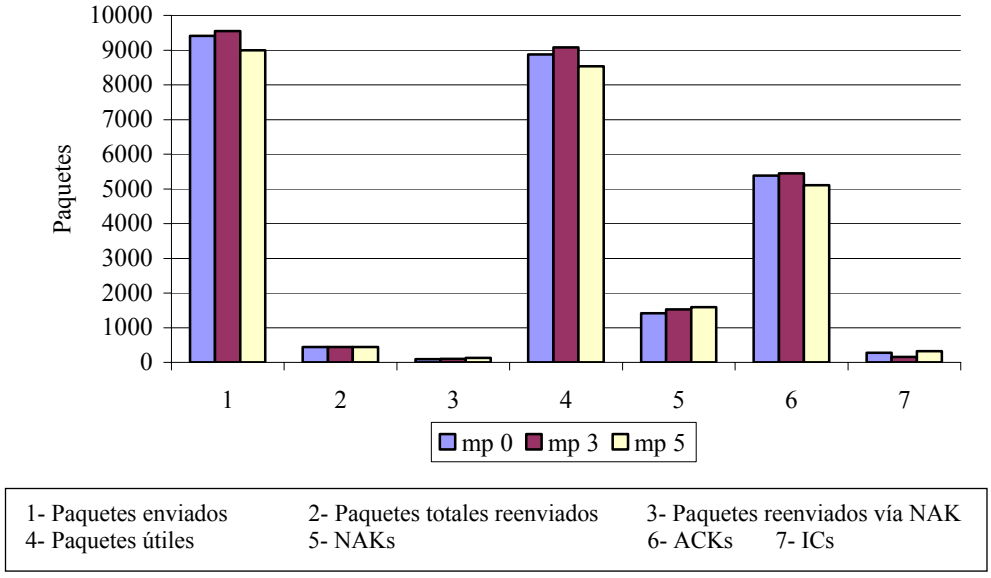

Figura 5.4 Número de paquetes en función del parámetro *minimum_period* (mp).

5.3.4 Feedback_factor

Este parámetro controla cada cuanto tiempo debe realizarse una estimación del número de receptores. Conocer el número de miembros activos de la sesión es necesario para ajustar convenientemente los temporizadores exponenciales. En el capítulo anterior, se describieron los parámetros de la exponencial, y se mostró que tienen una dependencia logarítmica con el número de receptores, por lo que no es necesaria una estimación muy exacta para su correcto funcionamiento. Así, el proceso para calcular el número de miembros del grupo consiste en enviar un paquete de control, que debe ser respondido por los receptores, una vez expiren sus temporizadores, y en contar las respuestas obtenidas, si no han sido canceladas. En principio, cuan frecuentemente debe desarrollarse este proceso, puede depender del modelo de tráfico de la fuente.

El número de paquetes de control obtenidos para valores de *feedback_factor* de 50, 100 y 200 es de 19.3, 6 y 5.8. Los resultados muestran como a medida que aumenta la frecuencia el número de paquetes crece.

5.3.5 Desired_feedback

Desired_feedback, o el número medio de respuestas NAK, es un parámetro de ajuste de los temporizadores exponenciales. Cuanto menor es este valor, el número de cancelaciones de NAKs es mayor, pero a costa de un mayor retardo. En la gráfica 5.5, se observa que el número de NAK recibidos durante el tiempo de simulación es casi la mitad, si se utiliza un *desired_feedback* de 2 frente a un valor de 6. Sin embargo, si se estudia el tiempo medio de retardo para valores de 2, 4 y 6 se obtienen 0.66, 0.32 y 0.23 segundos respectivamente. Las respuestas más lentas hacen que el número de cambios de representante sean menores. Así, los resultados obtenidos son de 18.5, 21 y 30.8 para valores de *desired_feedback* de 2, 4 y 6. En la gráfica, se observa que el mejor comportamiento del protocolo se consigue para un valor igual a 4.

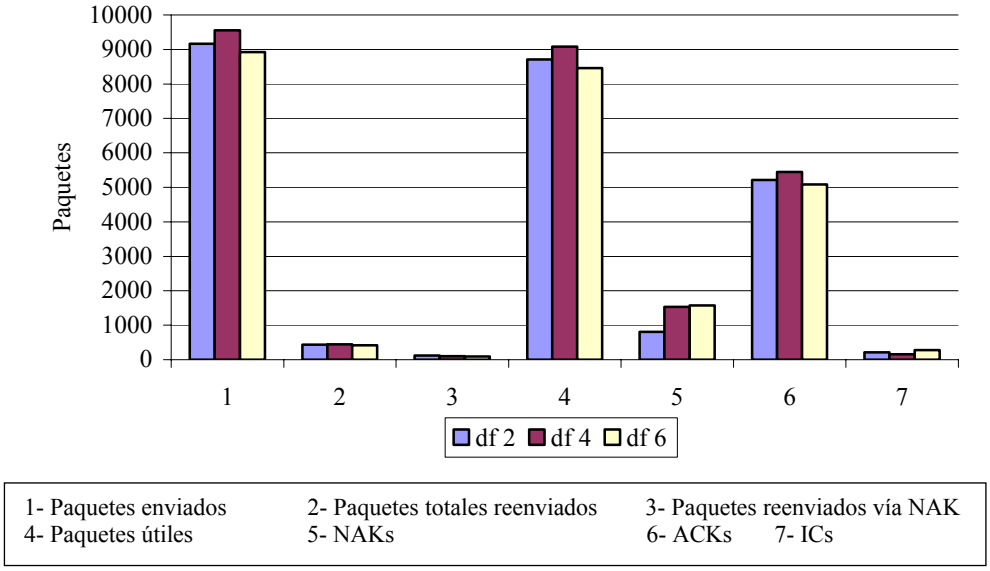

Figura 5.5 Número de paquetes en función del parámetro *desired_feedback* (df).

5.3.6 Automatic_send_factor

Automatic_send_factor es un parámetro del receptor que determina cuando se puede enviar un NAK inmediatamente. Si la tasa de pérdidas del receptor multiplicada por *automatic_send_factor* es mayor o igual a la tasa de pérdidas del representante, el receptor podrá enviar un NAK inmediatamente. Este mecanismo permite reducir el tiempo de respuesta del protocolo.

Las pruebas se han realizado para valores de 1, 0.75 y 0.5. Ajustar *automatic_send_factor* a 1 es prescindir del esquema de temporizadores que reduce el número de NAKs. Se observa en la figura 5.6, que si se asigna a uno esta variable, el número de NAK es mucho mayor que para los otros valores. Sin embargo, como la retransmisión no se realiza de forma inmediata, si no que está controlada por el

intervalo de tiempo *nak_factor*RTT_representative*, los resultados muestran que no se reenvían más paquetes que con los otros valores. Por otro lado, como el cambio de representante está controlado por el parámetro *switch_factor*, el mayor número de NAKs, tampoco lleva a un mayor número de cambios de representante.

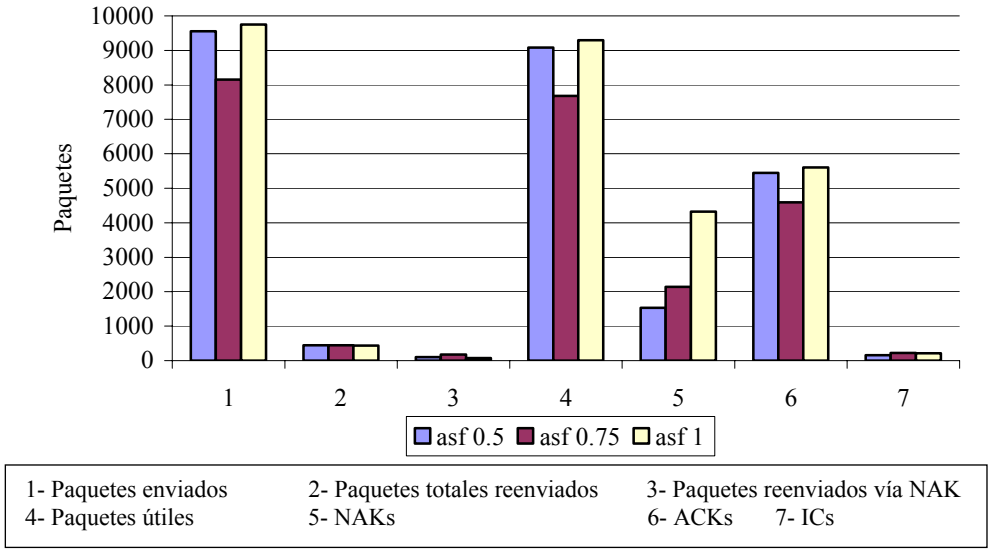

Figura 5.6 Resultados en función del parámetro *automatic_send_factor* (asf).

5.3.7 Resumen de los parámetros

Se va a resumir en la tabla 5.1 los parámetros más importantes que intervienen en el protocolo y su valor por defecto.

Parámetro	**Descripción**	**Valor**
Switch_factor	Cuanto peor debe ser el caudal del representante que el de receptor para realizar el cambio.	0.75
Nak_factor	Tiempo de espera para la retransmisión de un paquete solicitado mediante NAK	3
Minimum_period	Tiempo de permanencia mínimo como representante	3
Feedback_factor	Tiempo de espera para llevar a cabo la estimación del número de receptores	500
Desired_feedback	Número medio de NAKs por paquete perdido	4
Automatic_send_factor	Cuanto peor debe ser la tasa de pérdidas del receptor que la del representante para enviar inmediatamente un NAK	0.5

Tabla 5.1 Resumen de los parámetros de RCCMP y sus valores de ajuste.

5.4 UNA SOLA SESION RCCMP

En este apartado se describe el comportamiento del protocolo cuando una sesión RCCMP se considera de forma aislada. En concreto, se estudia como responde el protocolo a grupos multipunto de diferentes tamaños, y ante cambios de representante. El estudio del primer punto, está centrado en el análisis del caudal y del tamaño de ventana, cuando se incrementa el número de receptores, y con ellos el retardo de propagación entre la fuente y el representante. El objetivo de este primer punto es comprender ante una topología sencilla los principales mecanismos que rigen el protocolo. En esta sección no se analiza la escalabilidad del protocolo. El segundo punto, pretende mostrar de forma clara, la respuesta de RCCMP ante la aparición de receptores de menos recursos en el grupo multipunto. Además, se cuantifica el tiempo de respuesta del protocolo para adaptarse a las nuevas condiciones del grupo.

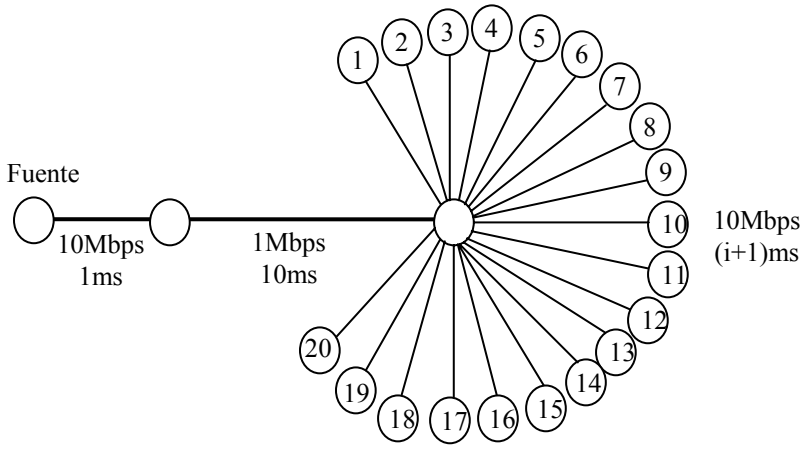

Figura 5.7 Topología para el estudio del comportamiento RCCMP para una sola sesión.

5.4.1 Tiempo de ida y vuelta entre la fuente y el representante

En función de distintos tamaños de grupos, con receptores conectados a enlaces con diferentes retardos de propagación, se analiza el caudal que inyecta a la red y el tamaño de ventana de un emisor RCCMP. Así, se compara el comportamiento del protocolo cuando varía el retardo de ida y vuelta entre la fuente y el representante.

La topología de la figura 5.7 es la que se va utilizar para este estudio. El enlace que une la fuente con el nodo intermedio tiene unas características de 10Mbps y 1ms de retardo de propagación. El enlace entre los dos nodos intermedios está caracterizado por 1Mbps y 10ms. Los enlaces que van desde el nodo intermedio hasta los terminales destino, tienen una capacidad de 10Mbps y un retardo que varía en función del número de receptor. Así, el retardo de cada enlace responde a la siguiente ecuación (*identificación_receptor*+1) ms. En una topología con un tamaño de grupo de tres miembros, los retardos de propagación en esos enlaces serían de 1, 2 y 3ms para el primer, segundo y tercer receptor. Las colas son FIFO con capacidad para 30 paquetes.

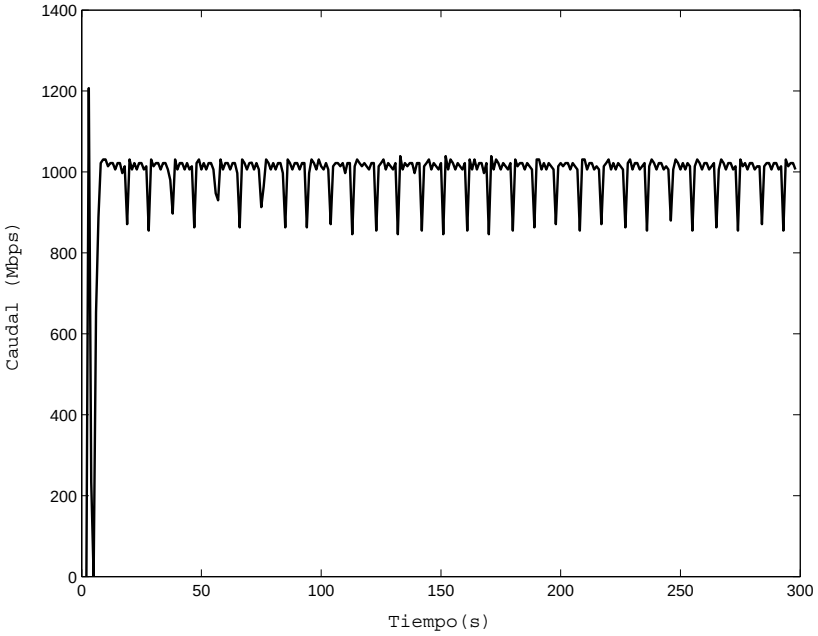

Figura 5.8 Caudal para una sesión RCCMP con grupo de tamaño 1 y sobre un canal sin pérdidas.

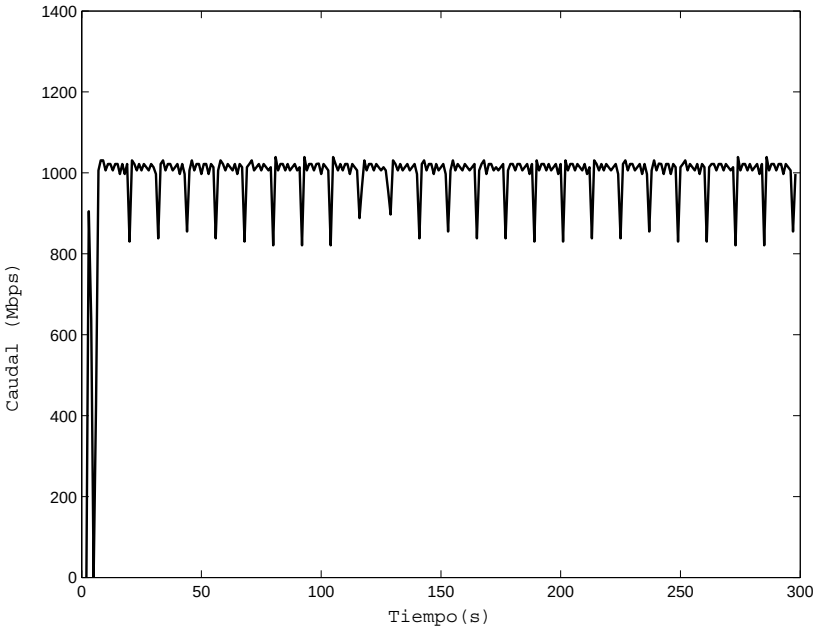

Figura 5.9 Caudal para una sesión RCCMP con grupo de tamaño 20 y sobre un canal sin pérdidas.

Las figuras 5.8 y 5.9 muestran el caudal de una sesión RCCMP con tamaño de grupo multipunto de uno y veinte receptores respectivamente. El representante se elige como el receptor con mayor tiempo de ida y vuelta. Este tiempo es de 24ms para el caso de un receptor y de 64ms para el caso de veinte. Las gráficas están dibujadas tomando un punto cada segundo. Al aumentar el tiempo de ida y vuelta, las pérdidas por congestión disminuyen. Mientras que en el primer caso se contabilizan

34 indicaciones de pérdidas, para el segundo se reducen a 26. En ambos casos, tan solo una pérdida es detectada por medio de la expiración del temporizador. Este resultado demuestra que al incrementarse el tiempo de ida y vuelta, aumenta la capacidad en bits del canal. El ancho de banda promedio es prácticamente igual para el primer y segundo caso, los valores obtenidos son de 987.8Kbps y 987.3Kbps respectivamente.

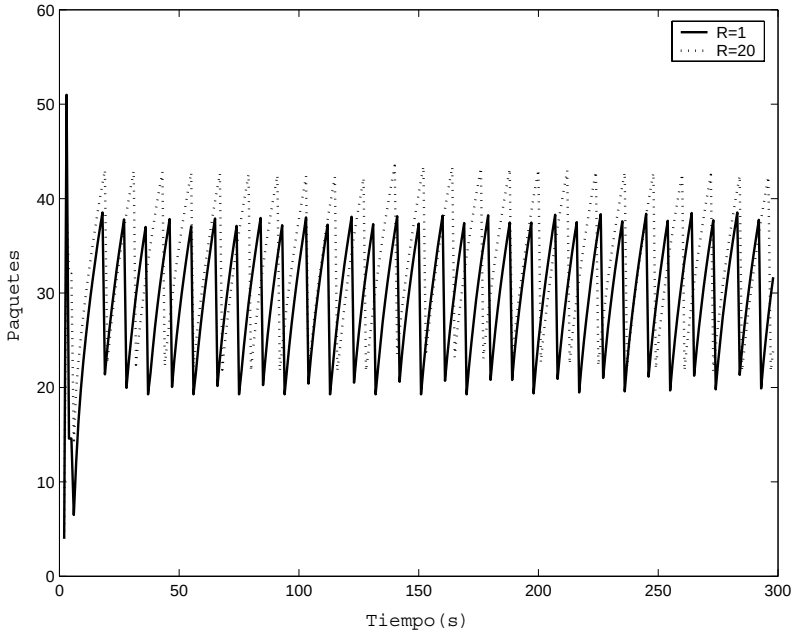

Figura 5.10 Tamaño de ventana para RCCMP en un canal sin pérdidas para distintos tamaños del grupo.

La evolución del tamaño de ventana a lo largo de la simulación se muestra en la figura 5.10. Los márgenes del tamaño de la ventanas son de [19,38] y [22,43] paquetes para un tamaño de grupo de uno y veinte receptores respectivamente. Cuanto mayor sea el tiempo de ida y vuelta entre la fuente y el representante, mayor es el número de paquetes que están circulando en la red. La conducta oscilante de las ventanas es debido a la pérdida periódica de paquetes en las colas. En la gráfica no se aprecia el tamaño de la ventana cuando finaliza la fase transitoria *(slow-start)*, ya que al expirar el temporizador de retransmisión, el tamaño de ventana se fija a uno. Esto es así, porque se han tomado los puntos de la gráfica cada segundo.

Una importante diferencia, entre el primer y segundo caso, es el comportamiento del protocolo en el transitorio *(slow-start).* Mientras, que para un tamaño de grupo de un receptor, la ventana alcanza su valor máximo en 51 paquetes y se observa un pico en el caudal; cuando el número de receptores aumenta y con ellos el tiempo de ida y vuelta, la evolución de la ventana en el transitorio *(slow-start)* es más progresiva. Al inicio de la sesión, el receptor más cercano se convierte en el representante temporal y la ventana inicial se iguala a cuatro. A continuación, a la fuente llega el paquete de control respuesta desde el segundo receptor más cercano, que se convertirá en el nuevo representante, y la ventana aumentará en una unidad, y así sucesivamente,

hasta que llegue el paquete desde el receptor más alejado. A partir de este instante, éste será el terminal que gobierne la tasa del grupo multipunto.

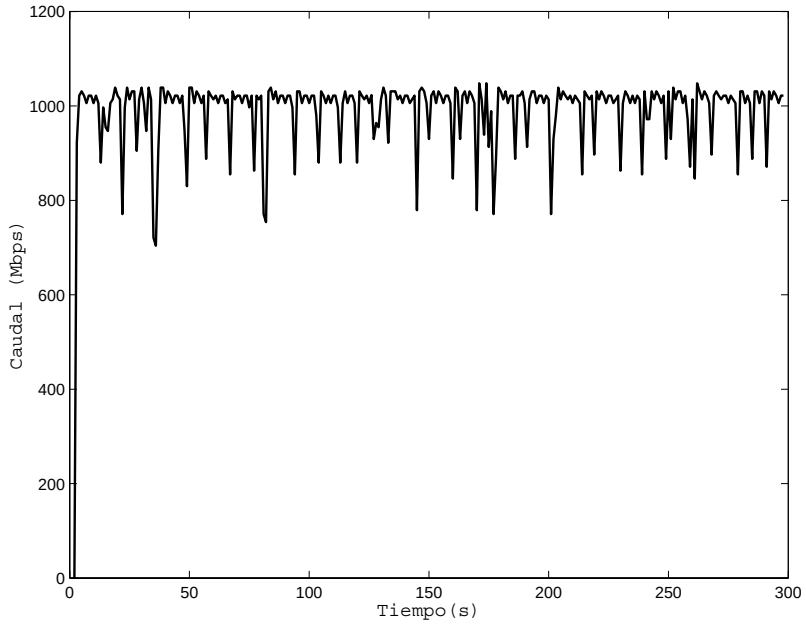

Figura 5.11 Caudal para una sesión RCCMP con grupo de tamaño 1 y sobre un canal con pérdidas.

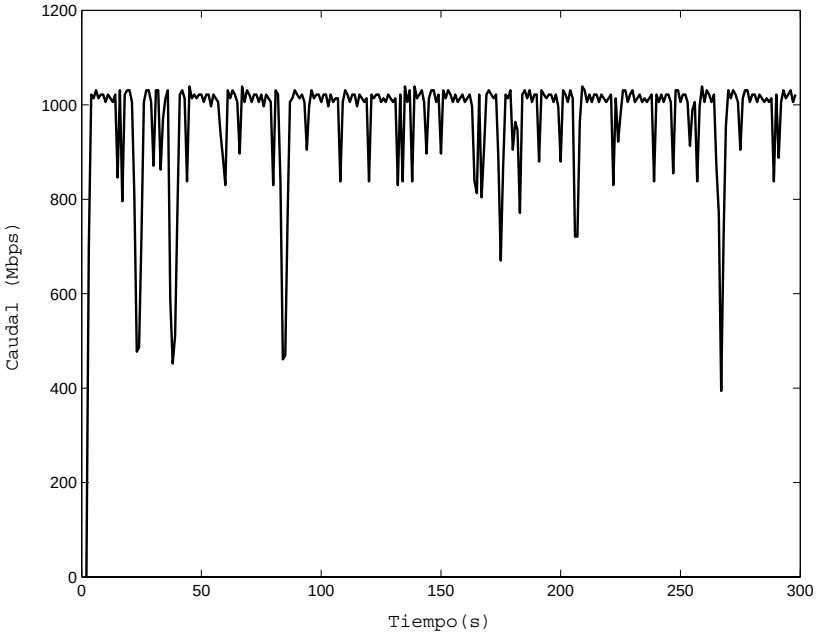

Figura 5.12 Caudal para una sesión RCCMP con grupo de tamaño 20 y sobre un canal con pérdidas.

Las gráficas 5.11 y 5.12 muestran el caudal de una sesión RCCMP, si se introducen pérdidas de canal con función de distribución uniforme de 0.001 en el

enlace que forma el cuello de botella. El número de indicaciones de pérdida aumenta respecto al caso anterior a 64 y 55, para una sesión con uno y veinte receptores respectivamente. En este caso, hay que añadir a las pérdidas por congestión, las pérdidas aleatorias que se introducen en el enlace. Todas son detectadas mediante la recepción de tres ACKs duplicados. Los caudales medios obtenidos son de 988.2Kbps y de 965.9Kbps para uno y veinte receptores. En este caso, se puede apreciar en las gráficas, como cuanto mayor es el tiempo de ida y vuelta, el tiempo para recuperar las pérdidas es mayor. La evolución del tamaño de la ventana se muestra en la figura 5.13. Cuando se consideran pérdidas, el margen de valores que alcanza la ventana es mayor en el caso de tiempos de propagación mayores. El tamaño medio de ventana obtenido es de 25.2 y 27.2 paquetes para uno y veinte receptores.

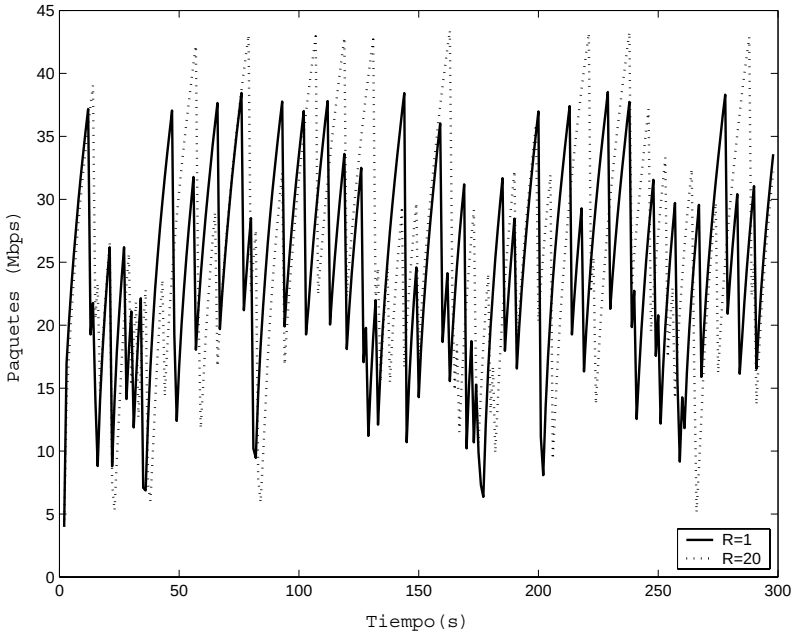

Figura 5.13 Tamaño de ventana para RCCMP en un canal con pérdidas en función del tamaño del grupo.

5.4.2 Cambio de representante

Un cambio de representante se realiza cuando en el grupo multipunto aparece un receptor con peores recursos que el representante actual. Este receptor debe darse a conocer a la fuente para que se produzca el cambio, y se regule la tasa de datos del grupo, a partir de este nuevo receptor. En este apartado se estudian detalladamente los procesos de cambio de representante. Para ello, se utilizan topologías sencillas que ilustran el comportamiento del protocolo RCCMP. En primer lugar, se estudia una topología con solamente dos receptores. Uno de ellos pertenece al grupo multipunto desde el inicio de la sesión, mientras que el otro, que cuelga del enlace más restrictivo, se adhiere al grupo a los 30s. En segundo lugar, se analiza una topología un poco más compleja, con tres receptores. En este escenario, se adhieren

al grupo en instantes diferentes, y en un orden que permite mostrar el comportamiento de RCCMP, ante sucesivos cambios de representante. En tercer lugar, se estudia una topología en árbol con seis receptores. En este caso, todos los enlaces son iguales y todos los receptores se asocian al grupo al inicio de la sesión. Por último, se ilustra el proceso de recuperación, cuando el representante abandona el grupo multipunto. En este caso, primero la fuente debe asegurarse de que el representante ya no está, y después debe iniciar el mecanismo para la búsqueda de un nuevo representante.

A continuación, se va a trabajar con el escenario que se presenta en la figura 5.14. El caudal que inyecta la fuente se muestra en la gráfica 5.15. Al inicio de la sesión, el grupo multipunto solamente está formado por el receptor RCCMP1, que será el representante. El caudal está limitado por los enlaces a 1Mbps. A los 30s, el receptor RCCMP2 se adhiere al grupo multipunto. En el instante 31.83s llega el primer NAK a la fuente solicitando la retransmisión de un paquete perdido. La fuente, a partir de la tasa de pérdidas y del número de secuencia más alto recibido, calcula el caudal estimado de este receptor, a partir de la fórmula de equilibrio de TCP [Mathis97], y lo compara con él del representante. El resultado indica que se debe ejecutar el cambio. Entonces, la fuente envía dos paquetes de datos informando del nuevo representante, y queda a la espera de recibir el paquete de reconocimiento ACK desde el nuevo representante. Esto se produce a los 32.42s. A partir de este instante, el caudal de la fuente está regulado por el receptor RCCMP2, y por ello se reduce a unos 800Kbps. En este ejemplo, el proceso de cambio de representante ha tardado unos 0.59 segundos.

Puede observarse, que durante la fase transitoria (*slow-start*) o durante el cambio de representante, cuando expira el temporizador de retransmisión, el caudal de la gráfica 5.15 no llega a cero, esto es debido a la falta de resolución al tomar los puntos de la gráfica (se ha considerado de un segundo). Sin embargo, la representación permite observar la estabilización del caudal cuando se realizan cambios de representante.

Si se aumenta el retardo de propagación de los enlaces de la topología 5.14 a 15ms, se observa como el cambio de representante se ralentiza. A los 30s, RCCMP2 se introduce en el grupo multipunto, y a los 31.80s llega el primer NAK, que determina el cambio. En el instante 32.59s se recibe el ACK desde el nuevo representante. En este caso, el proceso ha tardado 0.79s.

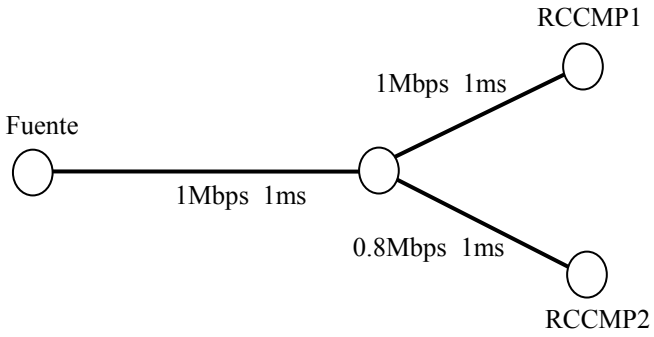

Figura 5.14 Topología con dos receptores RCCMP sobre enlaces de distinta capacidad.

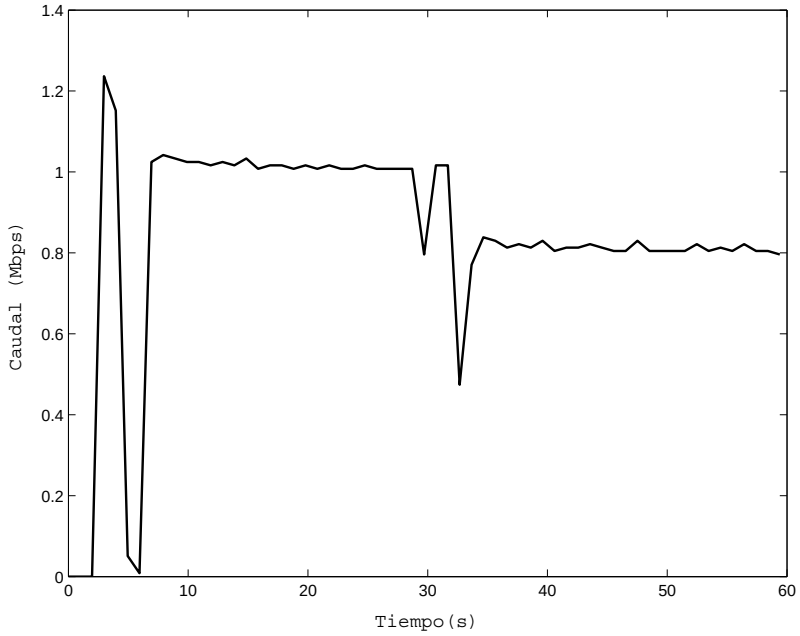

Figura 5.15 Caudal RCCMP. (RCCMP2 entra al grupo multipunto a los 30s).

La topología de la figura 5.16 cuenta con tres receptores que se asocian al grupo multipunto en distintos instantes, y que están unidos al nodo intermedio a través de enlaces independientes de diferentes capacidades. El receptor RCCMP1 está asociado al grupo desde el inicio de la transmisión, y será el representante hasta que un receptor con recursos más limitados aparezca. Esto ocurre en el instante 32.12s, momento en el cual, llega a la fuente el NAK desde el receptor RCCMP2, solicitando la retransmisión de un paquete perdido. En este momento, la fuente determina el cambio. Después de 0.6s, este receptor controla la comunicación con la fuente. El receptor RCCMP3 se asocia al grupo en el instante 40s. A los 42.25s llega a la fuente el NAK procedente de este receptor, que producirá el nuevo cambio de representante. En el instante 42.81s, RCCMP3 gobierna la comunicación. La figura 5.17 muestra el caudal a lo largo de los 80s que dura la simulación. Se observa claramente como afectan los cambios de representante al flujo de datos.

En el proceso de cambio de representante se diferencian dos tiempos. El primero, se define desde el instante que el receptor se adhiere al grupo hasta que se recibe el primer NAK en la fuente; y el segundo, mide el tiempo desde que la fuente determina el cambio de representante hasta que se recibe el primer ACK. Este segundo tiempo no puede reducirse, ya que depende de la dinámica de la ventana de transmisión. Sin embargo, el primer tiempo podría acortarse, a partir del ajuste de los parámetros de RCCMP, para que el receptor envíe lo antes posible el NAK a la fuente. Sin embargo, este ajuste puede afectar la escalabilidad del protocolo.

Se han modificado los parámetros *automatic_send_factor* y *switch_factor,* que regulan, cuanto peor ha de ser la tasa del receptor que la del representante para enviar un NAK inmediatamente, y cuanto peor ha de ser el caudal del candidato con respecto al del representante, para que se produzca un cambio de representante. El nuevo valor de estas variables se ha puesto a uno. En este caso, a la fuente llega el NAK a los 31.67s (unos 0.37 segundos antes que sin modificar los parámetros). El

primer ACK desde RCCMP2 llega a los 32.22s. Para el receptor RCCMP3, que se incorpora al grupo a los 40s, en el instante 41.70s llega el NAK a la fuente (un 1.1s antes), y a los 42.06s controla la comunicación. Aunque, existe una mejora en los tiempos de respuesta, el número de NAKs que se generan también ha aumentado extraordinariamente, de 102 NAKs durante la duración de toda la sesión en el primer experimento, se ha pasado a 613 en este segundo.

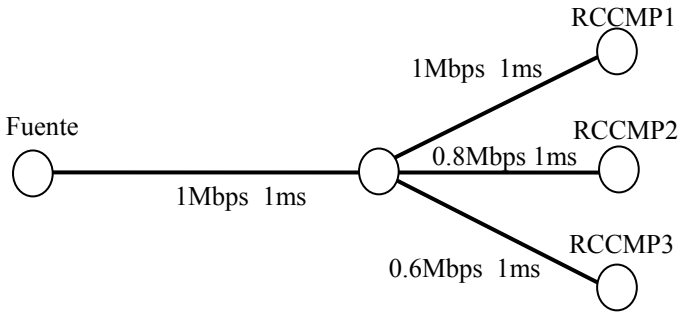

Figura 5.16 Topología con tres receptores RCCMP sobre enlaces de distinta capacidad.

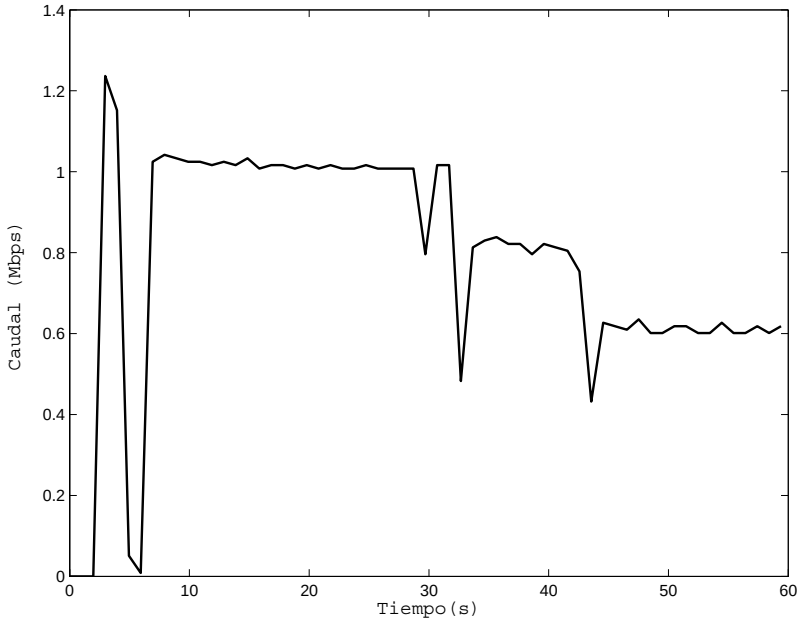

Figura 5.17 Caudal RCCMP. (RCCMP2 entra al grupo multipunto a los 30s y RCCMP3 a los 40s).

En los ejemplos anteriores, todos los cambios de representante siguen el siguiente procedimiento.

- La fuente recibe el NAK, con los parámetros adecuados, para comparar el caudal estimado del candidato con el del representante. El resultado determina que se debe realizar un cambio de representante.
- La fuente envía paquetes de datos, en función del tamaño de ventana, que incluyen un campo que informa de la modificación del representante.

- Se reciben los ACKs, desde el representante anterior, que confirman los paquetes enviados. Estos paquetes de confirmación no son válidos, ya que no provienen del representante actual, y no hacen avanzar la ventana.
- Se produce la expiración del temporizador de retransmisión.
- Se reenvían los siguientes paquetes al último paquete de datos, confirmado por un representante válido.
- Finalmente, se recibe el primer ACK desde el nuevo representante.

Como se observa en el procedimiento descrito, para los ejemplos anteriores, todos los cambios de representante tienen asociado la expiración del temporizador de retransmisión.

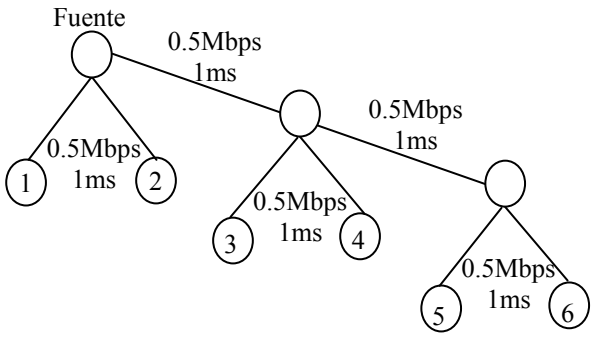

Figura 5.18 Topología arbolada con seis receptores. Todos los enlaces son iguales a 0.5Mbps y 1ms.

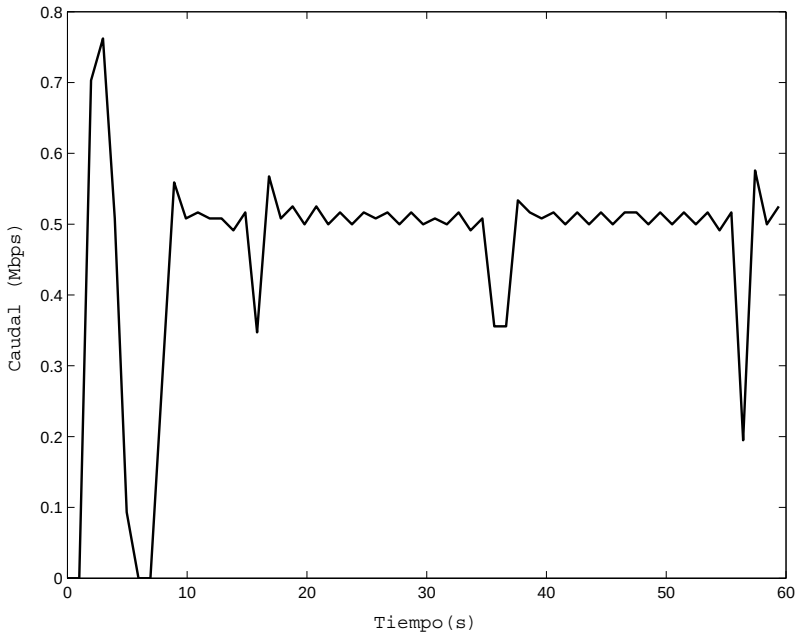

Figura 5.19 Caudal de una fuente RCCMP para una topología arbolada.

A continuación, se estudian los procesos de cambio de representante que se desarrollan en una estructura arbolada, figura 5.18, con todos los enlaces iguales a 0.5Mbps y retardo de propagación de 1ms. Los seis receptores se asocian al grupo multipunto desde el inicio de la sesión. En esta topología sólo tienen lugar pérdidas por congestión. En este experimento, se producen cinco cambios de representante de los que en solamente tres de ellos, se produce la expiración del temporizador de retransmisión. Los instantes en los que se producen los cambios son 11.84s, 13.41s, 15.17s, 35.34s y 55.78s. En la figura 5.19 se presenta el caudal. Los tres picos, que aparecen en la gráfica, muestran las pérdidas, detectadas por medio de la expiración del temporizador, y que son debidas a cambios de representante. Con este ejemplo, se quiere mostrar, que no todos los cambios de representante, llevan asociados la expiración del temporizador. Esto dependerá de la dinámica de la ventana de transmisión.

Para finalizar esta sección, se va ilustrar el proceso cuando un representante abandona el grupo multipunto. Esto puede ser debido a que voluntariamente el usuario decide dejar la conexión multipunto, o porque se produzca algún fallo en la red que impida la conectividad. En RCCMP, la tasa de la sesión está regulada por un mecanismo de ventana deslizante entre la fuente y el representante. Cuando el segundo abandona la comunicación, se producirá una primera expiración del temporizador de retransmisión. Entonces, se transmitirá el paquete siguiente al último debidamente confirmado, y se actualizará el temporizador al valor anterior multiplicado por un factor igual a 2. Si no se obtiene respuesta, se volverá a repetir el proceso, pero modificando el valor del temporizador. Simultáneamente, se enviará un paquete de control PIC, para seleccionar un nuevo representante. Las sucesivas respuestas, por parte de los receptores activos, determinarán cual de ellos es el nuevo representante. Al expirar el temporizador de retransmisión, se enviará un nuevo paquete, que será confirmado por el nuevo representante, y que permitirá que la sesión se restablezca.

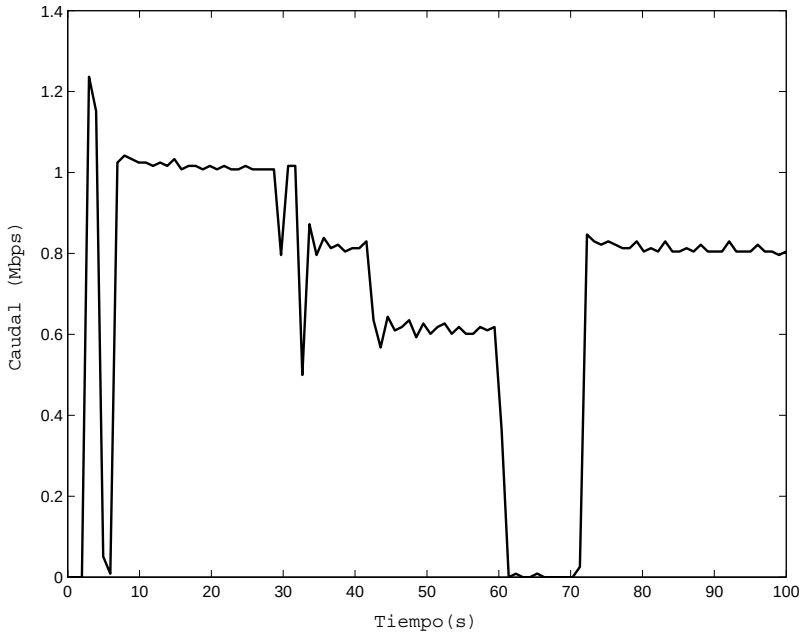

Figura 5.20 Caudal RCCMP. (RCCMP2 entra al grupo multipunto a los 30s, RCCMP3 a los 40s y abandona el grupo a los 60s cuando es representante).

La figura 5.20 muestra el caudal de una sesión RCCMP, para el caso de la topología de la figura 5.16. En este ejemplo, los receptores se adhieren al grupo multipunto en los mismos instantes que en el caso anterior (0.5s, 30s y 40s). El representante, RCCMP3, abandona la sesión en el instante 60s. La gráfica muestra como a partir del abandono del representante, el mecanismo de ventana deslizante no puede continuar, y el caudal cae hasta cero. Después de la segunda expiración del temporizador, se inicia el proceso de búsqueda de representante, y a partir de la tercera expiración del temporizador, ya con el nuevo representante, continúa la comunicación.

Se va a calcular la relación entre el caudal medio teórico y el obtenido en la figura 5.20. El primero se obtiene considerando que los cambios y la reposición de un nuevo representante se realizan de forma inmediata. Por lo tanto, se supone que durante los primeros 30s la tasa de la fuente es de 1Mbps, a continuación está 10s a 0.8Mbps, los 20s siguientes a 0.6Mbps, y a partir del abandono del representante, la tasa se ajusta a 0.8Mbps. Esto resulta en un caudal medio de 0.82Mbps. Si se calcula el caudal medio obtenido mediante simulación se obtiene un valor de 0.71Mbps. La eficiencia de RCCMP, calculada como el cociente entre el caudal real y el teórico, es del 86.6%.

5.5 COMPORTAMIENTO INTERPROTOCOLO

Para implantar cualquier nuevo protocolo en Internet es muy importante analizar su conducta frente a los protocolos ya existentes, y en concreto, frente al protocolo dominante: TCP. Un buen diseño debe conseguir, que los nuevos protocolos se comporten frente a la congestión tal como lo haría TCP, para evitar que los recursos de red se compartan de forma poco equitativa. A continuación, a través de una serie de experimentos, se va a comparar el comportamiento de RCCMP frente a TCP. Los diferentes escenarios, que se van a utilizar, están basados en el trabajo de Seada y Helmy [Seada02], y tratan de evaluar diferentes aspectos.

Para medir el grado de equidad se utilizan diferentes métricas. Una de las más utilizadas es el índice de equidad [Chiu89], definido como

$$\mathrm{FI} = \frac{\left(\sum_{i=0}^{N-1} X_i\right)^2}{N \cdot \sum_{i=0}^{N-1} X_i^{\,2}} \tag{5.1}$$

donde X_i es el caudal de la instancia i del protocolo y N es el número de conexiones que atraviesan el enlace. Este índice está acotado entre 0 y 1. Un escenario totalmente equitativo, con todos los X_i iguales, tiene un índice de equidad igual a 1. En el escenario menos equitativo, donde todos los recursos son utilizados por un solo usuario, el índice de equidad es igual a 1/N. En este caso, si el número de conexiones es suficientemente grande, el índice de equidad tiende a 0. FI tiene un valor de K/N, si sólo K de las N conexiones comparten equitativamente el enlace. Las ventajas de este índice son que es independiente de la unidad de tiempo, y que cualquier pequeño cambio en la asignación de ancho de banda es capturado.

Otro parámetro que mide la equidad entre distintos protocolos es la proporción del ancho de banda que toma cada uno de ellos. Se define el factor de equidad, *f*, como el cociente entre el caudal de TCP y el de RCCMP.

En las topologías sobre las que se van a realizar los distintos experimentos, todos los enlaces tienen un retardo de propagación de 1ms y un ancho de banda de 10Mbps, a menos que se especifique lo contrario. Las colas tienen una disciplina de servicio FIFO, y en caso de saturación, el paquete que se descarta es el último en llegar. Su capacidad es de 30 paquetes. El modelo de fuente responde a una aplicación que genera tráfico a una tasa de 1Mbps, superior a la capacidad del enlace más restrictivo. Esto es así, para que la tasa de la fuente esté controlada en todo momento por el control de congestión. Los paquetes de datos son de tamaño 1048 octetos. Los resultados se expresan mediante gráficas, que representan el número de paquetes enviados y el tamaño de la ventana en función del tiempo. El primer resultado es fácilmente convertible a tráfico enviado.

El primer experimento analiza el comportamiento de RCCMP y diferentes implementaciones de TCP en una topología muy sencilla con solo un cuello de botella. Para ello, se va a utilizar la topología de la figura 5.21, en la que el enlace congestionado, tiene un retardo de propagación de 50ms y un ancho de banda de 500Kbps.

En la figura 5.22, se observa el número de paquetes enviados por RCCMP y la implementación TCP Reno para tres experimentos sobrepuestos. El primero, consiste en activar una instancia RCCMP y otra TCP en el mismo instante de tiempo, en el segundo la fuente TCP-1 se retrasa 0.5s respecto a RCCMP-1, y en el tercero RCCMP-2 se retrasa 0.5s respecto a TCP-2. En estos experimentos solamente se producen pérdidas por saturación en las colas. Puede verse, como en el primer caso, TCP obtiene una mejor respuesta frente a RCCMP (97143 paquetes frente a 82477); sin embargo, en los experimentos sucesivos, al retrasar o avanzar una fuente respecto a otra, el comportamiento de las dos conexiones es más equitativo. Esto es debido al uso de colas FIFO, que no están tratando a las dos conexiones por igual. Al desincronizar las conexiones, la cola FIFO tiene un mejor comportamiento.

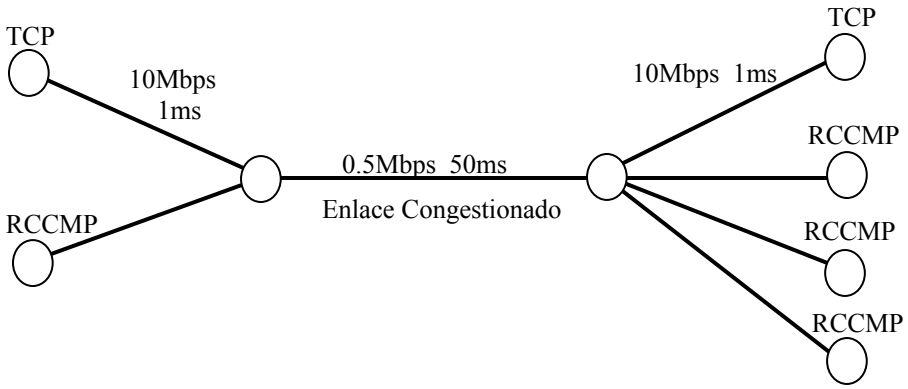

Figura 5.21 Una sesión RCCMP con tres receptores compitiendo con una sesión TCP en un enlace congestionado.

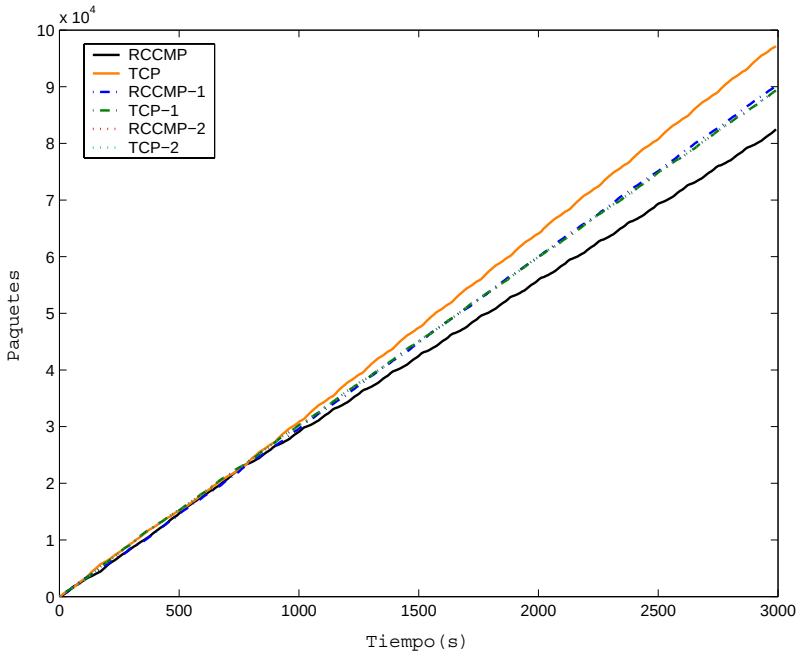

Figura 5.22 Secuencia de paquetes de RCCMP versus TCP Reno.

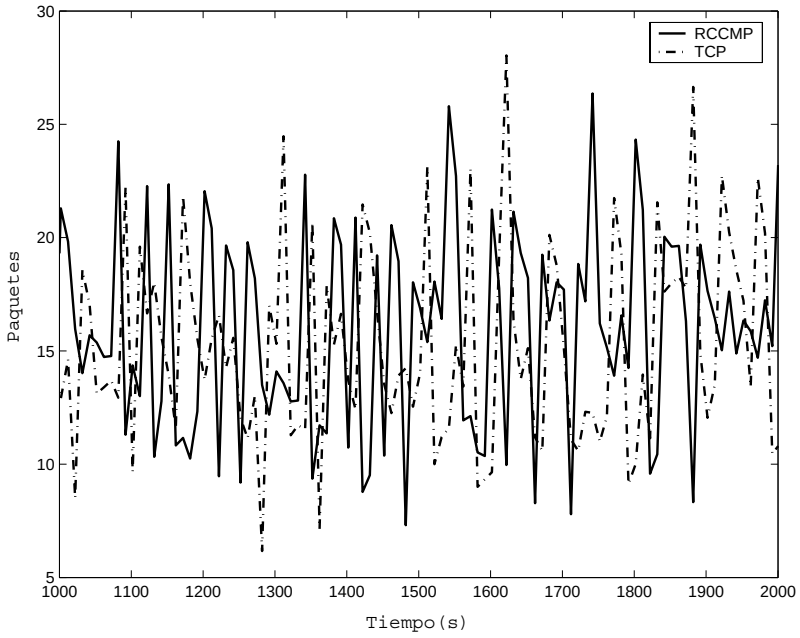

Figura 5.23 Tamaño de la ventana de para RCCMP y TCP Reno.

El índice de equidad es de 0.993 para el primer experimento y de 0.999 para el segundo y tercero. El porcentaje de tráfico TCP frente a RCCMP es del 117.7%, 99% y 98% para las tres pruebas respectivamente. Además, el tamaño de la ventana para los dos protocolos en cualquiera de los tres experimentos se mantiene dentro de los mismos márgenes. En la figura 5.23 se muestra el tamaño de ventana para el caso en el que la fuente RCCMP empiece 0.5s antes que la TCP.

El diseño del control de congestión de RCCMP trata de emular al máximo el de TCP Reno, para obtener una respuesta lo más parecida posible. Como RCCMP y TCP Reno reaccionan de la misma manera a las pérdidas y a los ACKs, los resultados muestran un comportamiento totalmente equitativo.

Si se introduce en el enlace congestionado pérdidas de canal con función de distribución uniforme de un 1%, y se realizan otra vez los tres experimentos, en los que simultáneamente se lanzan las conexiones RCCMP y TCP, y se retarda una respecto a otra 0.5s; los resultados muestran un comportamiento equitativo con un índice de equidad de 0.999 en los tres experimentos, y un porcentaje de tráfico TCP sobre RCCMP de 95%, 96% y 95% respectivamente. Aunque, estos resultados son buenos, en la figura 5.24 se muestra como RCCMP se comporta de forma ligeramente más agresiva que TCP. Esto es debido, a diferencias en la gestión de los temporizadores, ya que una de las principales diferencias, en el control de congestión entre RCCMP y TCP es la manera de calcular el tiempo de ida y vuelta entre la fuente y el representante. Mientras que TCP utiliza un cálculo muy burdo, RCCMP hace un cálculo más exhaustivo del RTT, cada vez que recibe un ACK. Otra diferencia entre RCCMP y TCP, es el proceso para la recuperación de un paquete perdido, cuando la detección se produce por medio de tres ACKs duplicados. Mientras que la versión de TCP Reno, implementada en el simulador ns-2, solamente envía un paquete cuando se recibe el primer y el segundo ACK duplicado, RCCMP sigue el procedimiento descrito en el [RFC2581], en el que para cada ACK duplicado adicional, se incrementa la ventana de transmisión convenientemente, y se envía un paquete si es posible. Estas dos diferencias entre el control de congestión de RCCMP y TCP, suponen que para largos periodos de tiempo, y en un escenario de fuertes pérdidas, los protocolos muestran una respuesta ligeramente diferenciada.

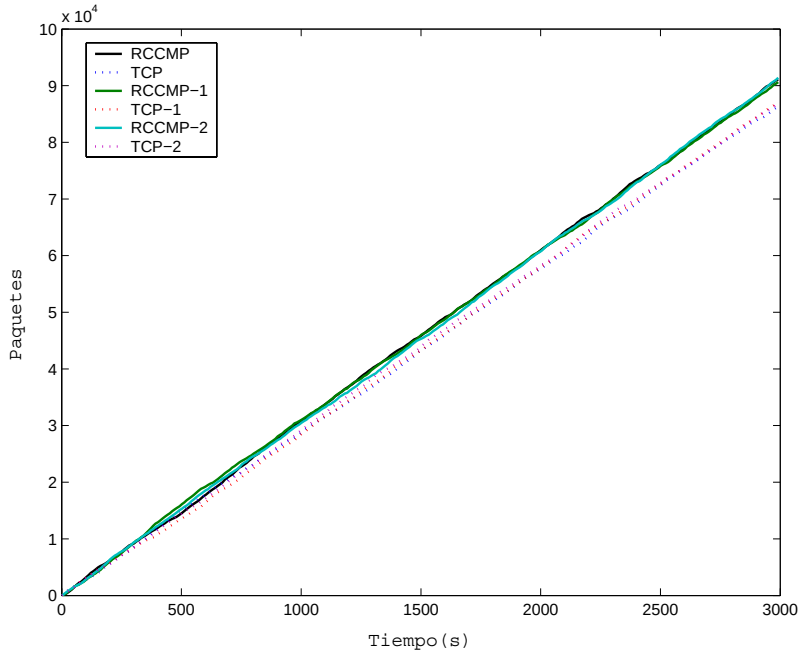

Figura 5.24. Secuencia de paquetes de RCCMP versus TCP Reno para un enlace con 1% pérdidas.

Conexiones	FI	f
RCCMP-TCP1	0.97	70.3%
RCCMP-TCP2	0.994	85.96%
RCCMP-TCP3	0.993	85.01%

Tabla 5.2. Índices de equidad para RCCMP compitiendo con TCP Vegas.

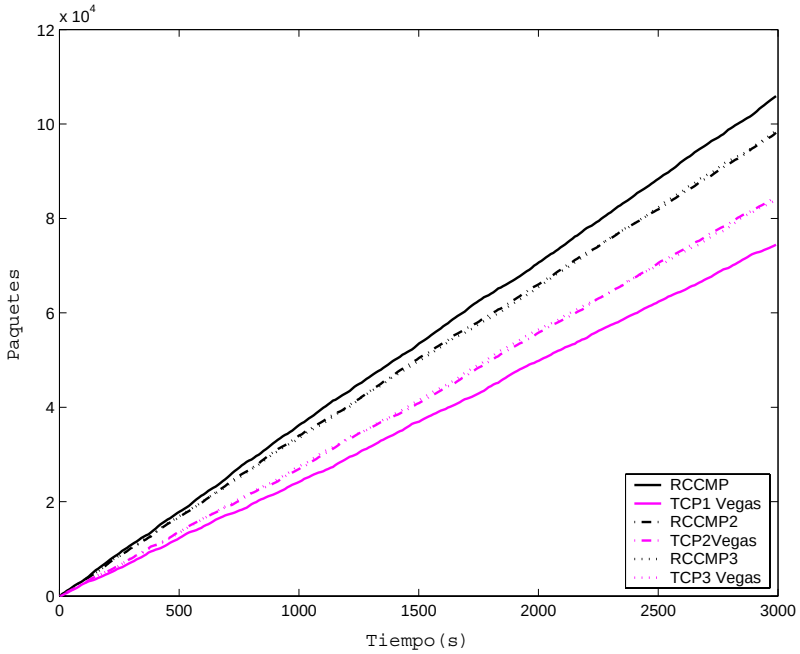

Figura 5.25 Secuencia de paquetes de RCCMP versus TCP Vegas.

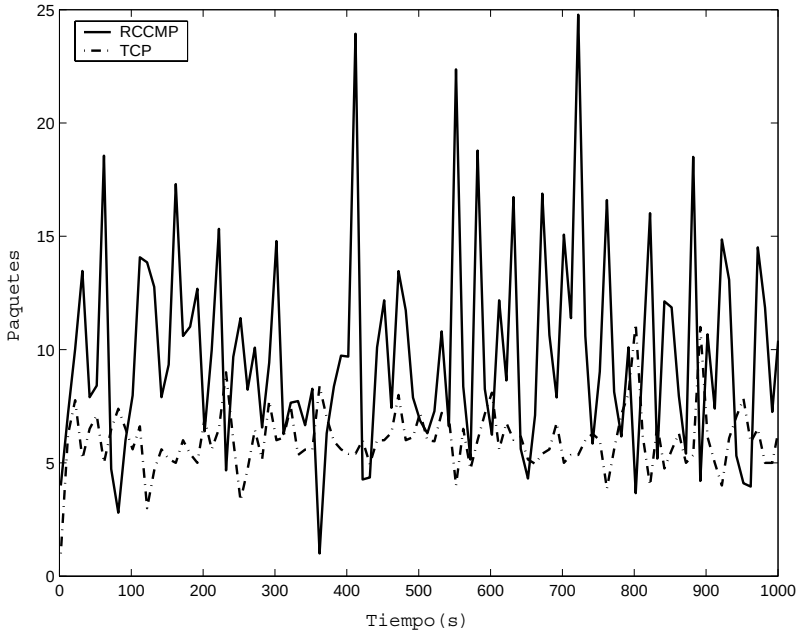

Figura 5.26 Tamaño de la ventana de RCCMP versus la ventana de TCP Vegas.

En las figuras 5.25 y 5.26 se compara la conducta de RCCMP frente a TCP Vegas. La gráfica 5.25 muestra el número de paquetes para tres simulaciones en las que se han modificado los umbrales de TCP Vegas. TCP1 tiene unos valores iguales a $\alpha=\gamma=1$ $\beta=3$, TCP2 de $\alpha=\gamma=\beta=3$ y TCP3 de $\alpha=\gamma=3$ $\beta=5$. Los resultados de estas simulaciones presentan un escenario menos equitativo que el observado para TCP Reno. RCCMP se comporta de forma más agresiva que TCP Vegas. El índice de equidad y el factor de ancho de banda entre RCCMP y los distintos TCP Vegas pueden verse en la tabla 5.2.

En la figura 5.26 se muestra el tamaño de ventana para RCCMP y TCP Vegas con $\alpha=\gamma=1$ $\beta=3$. Se observa como la ventana del protocolo multipunto permanece con valores superiores a la del protocolo punto a punto. En este caso, el comportamiento predictivo de TCP Vegas hace que el tamaño de su ventana tenga un comportamiento más suave, menos oscilante.

A continuación, se van a analizar los efectos al considerar receptores conectados a enlaces con diferentes pérdidas de canal y retardos. En este caso, solamente se consideran pérdidas independientes, es decir, no se consideran pérdidas debidas a congestión, por lo que se va a utilizar para las simulaciones, tamaños de colas de 3000 paquetes. La topología utilizada queda descrita en la figura 5.27, donde aparecen dos receptores RCCMP, uno de ellos unido a un enlace con alto retardo (400ms) y baja tasa de pérdidas (0.4% ó 2%), y el otro con bajo retardo (200ms) y alta tasa de pérdidas (1.6% ó 8%). Los demás enlaces se configuran a 1Mbps y 1ms de retardo. Los modelos de fuente son aplicaciones que generan tráfico continúo a 2Mbps. Los valores de retardo y de pérdidas de canal han sido escogidos para obtener los mismos valores de caudal en ambos enlaces según la ecuación simplificada de equilibrio de TCP [Mathis97].

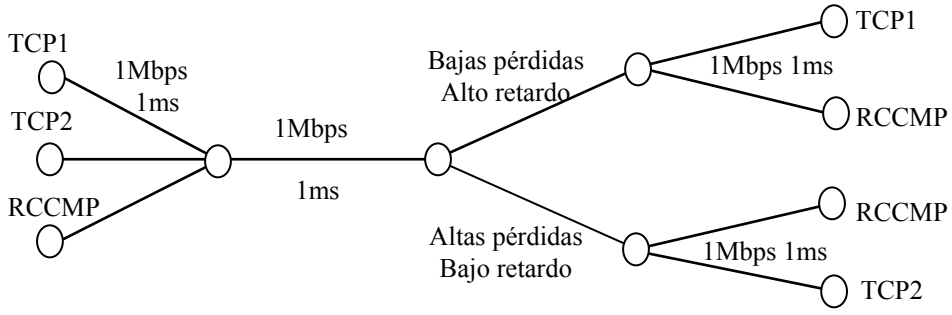

Figura 5.27 Una sesión RCCMP compitiendo con dos sesiones TCP sobre enlaces con diferentes retardos y pérdidas.

Se observa, en la figura 5.28, que para el caso en el que se apliquen las tasas de error bajas en los enlaces, 0.4% y 1.6%, la conexión TCP, TCP-1, que circula por el enlace de mayor retardo, es la que acapara mayor ancho de banda, mientras que la conexión TCP, TCP-2, que circula por el enlace de altas pérdidas, tiene una respuesta más desfavorecida. La conexión RCCMP se sitúa más cercana a la conexión TCP desfavorecida, por lo que el receptor que durante más tiempo ha ejercido como representante es el RCCMP2. Los índices de equidad obtenidos son *FI*=0.988, *f1*=101.3% y *f2*=108.5%. Se han contabilizado 108 cambios de representante. Las indicaciones de pérdida que ha detectado la fuente son de 555, de las que 81 son

debidas a expiraciones del temporizador, 274 a la recepción de ACKs duplicados, y 200 por solicitud de NAKs. Los paquetes retransmitidos suponen el 2.2% de los paquetes totales.

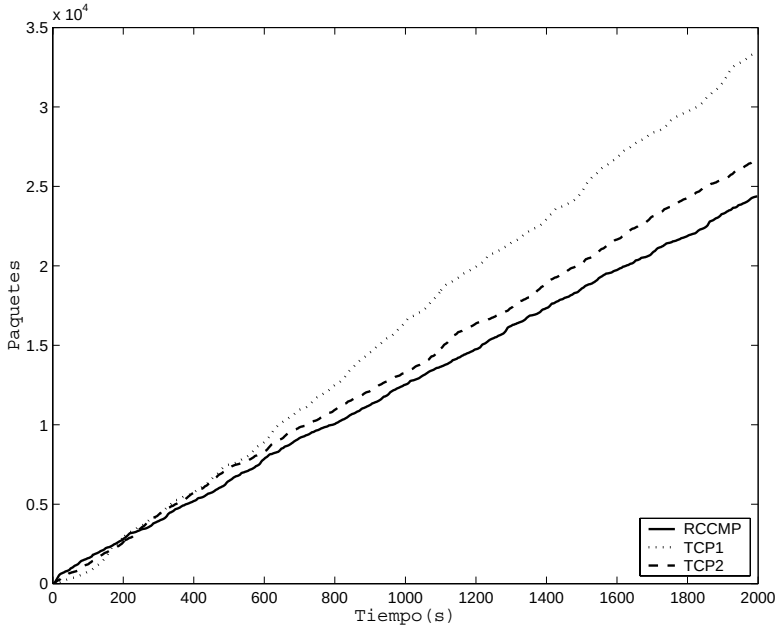

Figura 5.28 Secuencia de paquetes de RCCMP y dos conexiones TCP con tasas de error bajas.

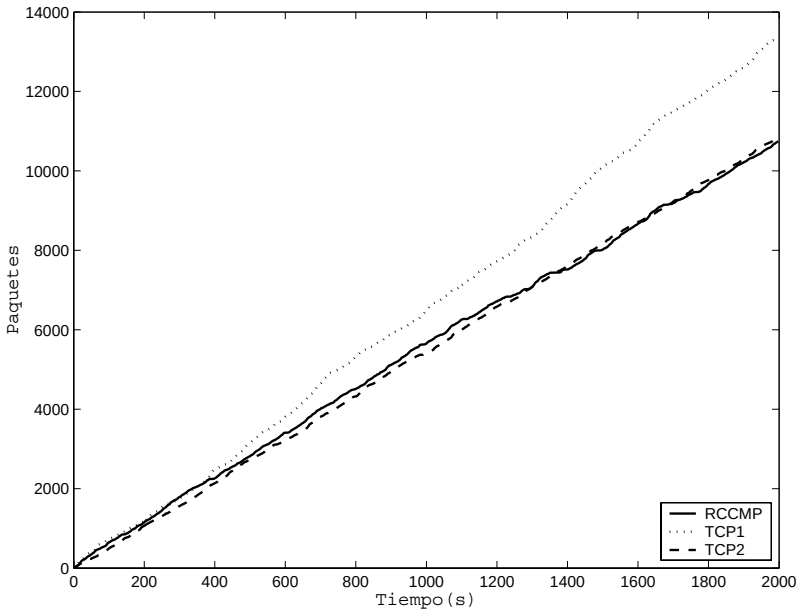

Figura 5.29 Secuencia de paquetes de RCCMP y dos conexiones TCP con tasas de error altas.

En la figura 5.29 se presentan los resultados cuando se utiliza la topología con enlaces con tasas de error del 2% y 8%. RCCMP, otra vez, muestra un comportamiento en el que se alinea con la conexión TCP más desfavorecida, es

decir, la conexión TCP con menor retardo y mayores pérdidas. En este caso, los índices toman los siguientes valores 0.99, 101.24% y 101.99% para FI, f1 y f2 respectivamente. Se observa que la conexión TCP, TCP-1, con mayor retardo sale favorecida. Aunque, los valores de retardos y tasas hayan sido escogidos para obtener el mismo caudal, a partir de la ecuación simplificada de TCP, la simplicidad de esta aproximación que no considera las pérdidas por expiración del temporizador, hace que existan conexiones claramente privilegiadas en un escenario con pérdidas elevadas. En este caso, las indicaciones de pérdida son de 998 de las que 259 son detectadas mediante la expiración del temporizador, 405 mediante ACKs duplicados y 334 a través de NAK. La tasa de paquetes retransmitidos es del 9.4%. El número de cambios de representante se eleva a 121.

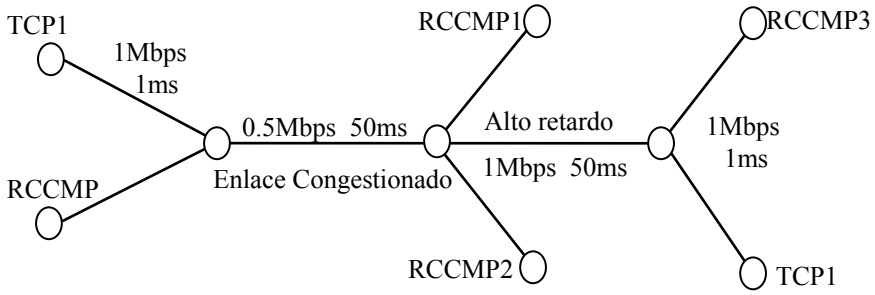

Figura 5.30 Una sesión RCCMP con tres receptores con las mismas pérdidas pero diferentes retardos.

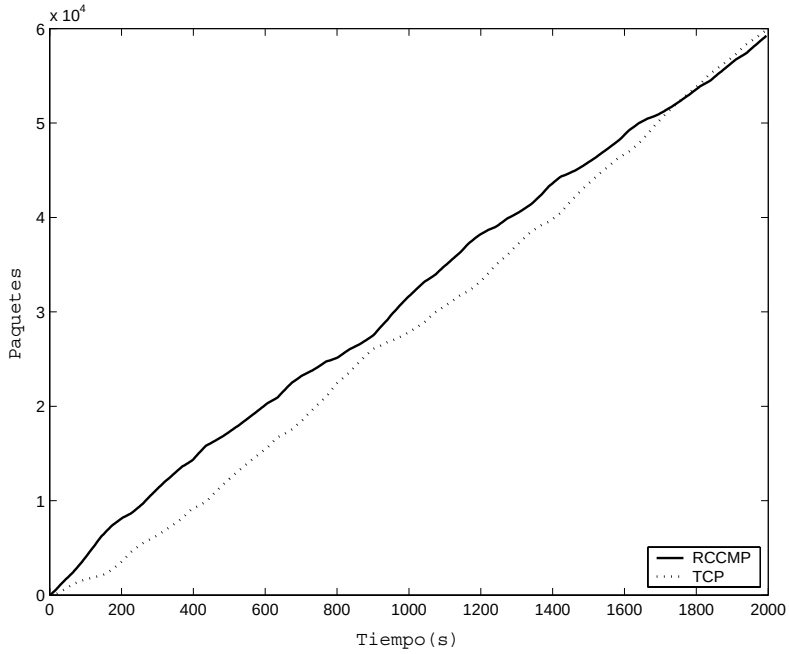

Figura 5.31 Secuencia de paquetes de TCP y RCCMP para receptores con diferentes retardos.

Con la topología mostrada en la figura 5.30, se pretende estudiar, el efecto de la supresión de información de realimentación. En este caso, los enlaces son de 1Mbps, 1ms de retardo de propagación, y tamaño de cola de 30 paquetes, excepto el enlace

congestionado que responde a 0.5Mbps y 50ms. El enlace etiquetado como de alto retardo, también tiene un tiempo de propagación de 50ms. Los receptores RCCMP se situarán detrás del cuello de botella con diferentes retardos. Los resultados expuestos en las figuras 5.31 y 5.32 muestran un comportamiento equitativo entre los dos protocolos. Los índices de equidad son de 0.999 y 106.8% para *FI* y *f*.

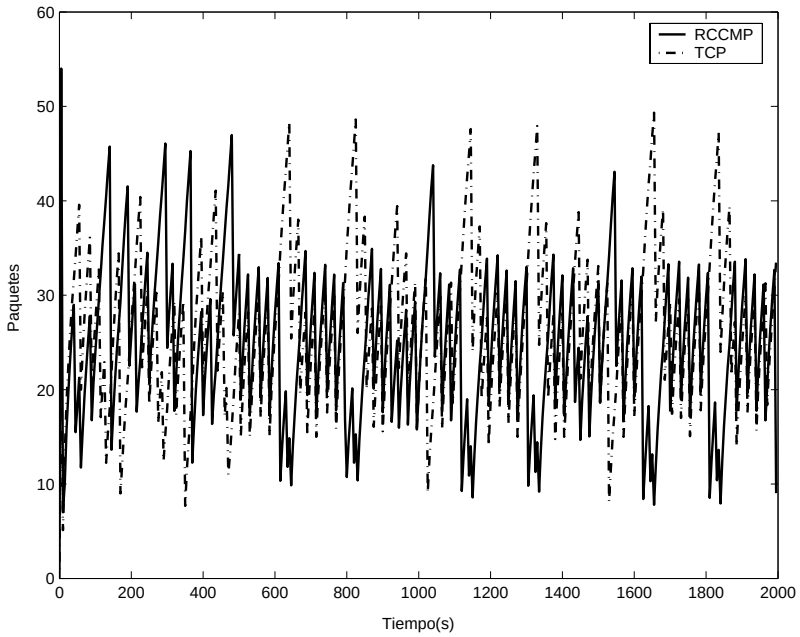

Figura 5.32 Tamaño de ventana para TCP Y RCCMP con receptores con diferentes retardos.

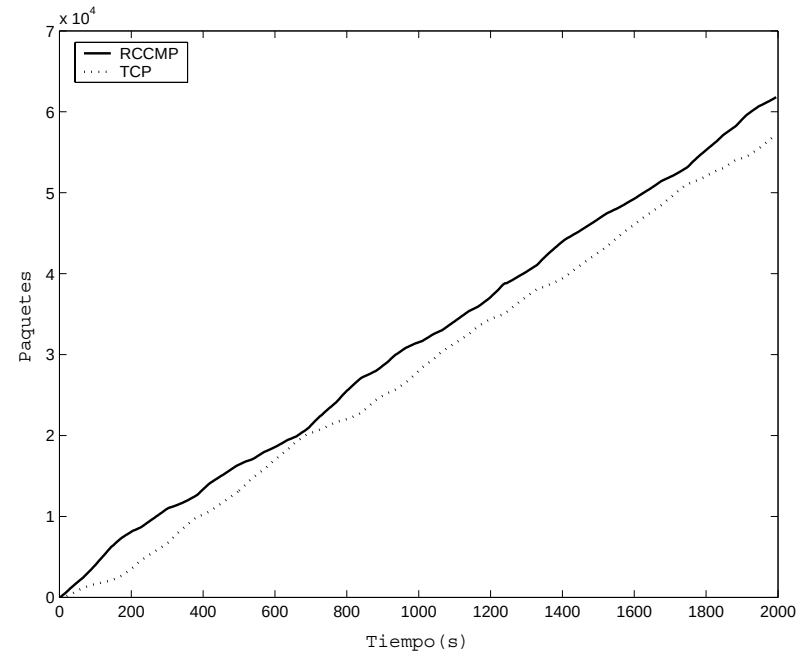

Figura 5.33 Secuencia de paquetes de TCP y RCCMP para receptores con diferentes retardos.

Por último, para acabar este tipo de experimentos, se va a estudiar la misma topología propuesta en la figura 5.30, pero ahora, el enlace llamado de alto retardo, tiene un valor de 200ms. En este caso, figura 5.33, el comportamiento es equitativo entre los dos protocolos. En estas simulaciones no se producen cambios de representante, desde el primer momento se elige como tal al receptor que está más alejado de la fuente. Los índices de equidad son *FI* igual a 0.999 y *f* de 101.03%.

Otros experimentos, que muestran la calidad de RCCMP en cuanto a la equidad, evalúan el comportamiento del protocolo cuando el número de conexiones, que compiten por el ancho de banda de un enlace, aumenta. Para ello, se va a utilizar la topología descrita en la figura 5.21, en la que el enlace congestionado tiene un retardo de propagación de 50ms, y un ancho de banda que responde a la siguiente función: $0.25 * N$, donde N es el número de conexiones RCCMP y TCP que atraviesan el enlace. Así, si el numero de flujos TCP y RCCMP suman cuatro, el ancho de banda del enlace congestionado tendrá una capacidad de 1Mbps. El modelo de fuente responde a una aplicación que genera datos de forma continua a una tasa de 1Mbps. Por lo tanto, las pérdidas que van a producirse son solamente por congestión.

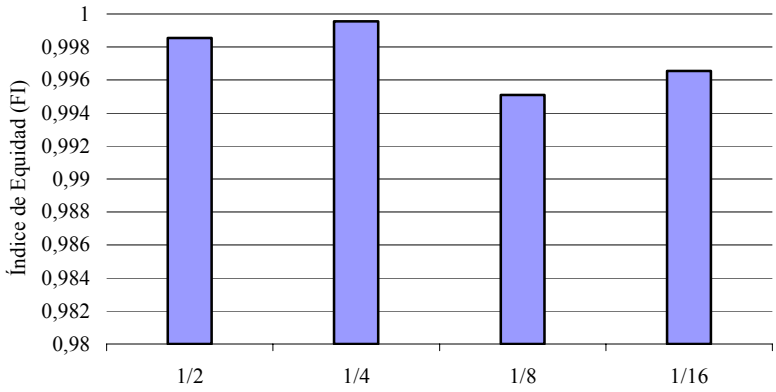

Figura 5.34 Índice de equidad para una sesión RCCMP que compite con diferentes sesiones TCP.

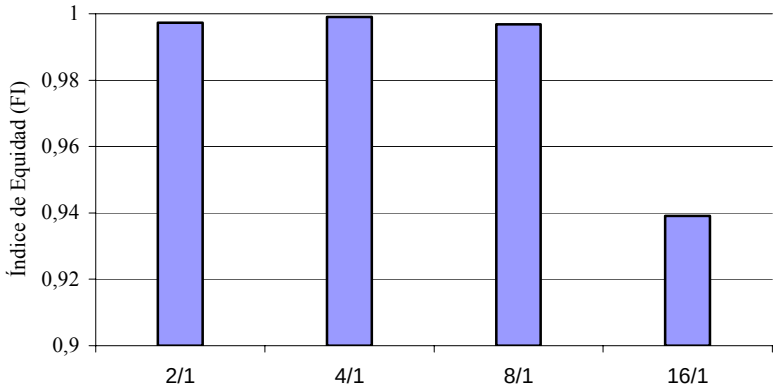

Figura 5.35. Índice de equidad para diferentes sesiones RCCMP que compiten con una sesión TCP.

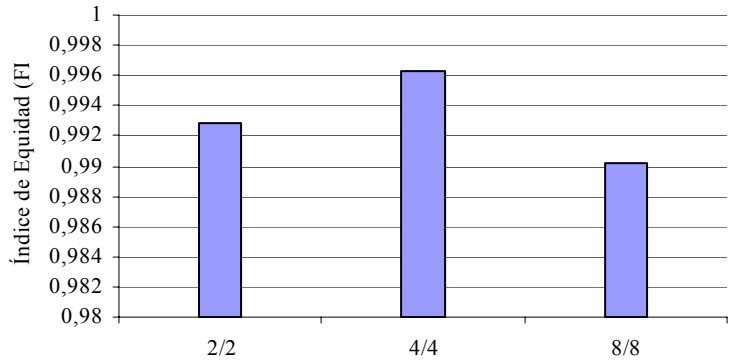

Figura 5.36 Índice de equidad para *i* sesiones RCCMP que compiten con *i* sesiones TCP.

Se han considerado diferentes escenarios: en el primero, una sesión RCCMP comparte el ancho de banda con distintas sesiones TCP; en el segundo, una sesión TCP comparte el enlace con diferentes sesiones RCCMP; y por último, el mismo número de sesiones TCP y RCCMP compiten por el mismo ancho de banda. Los resultados se expresan a través del índice de equidad para las diferentes pruebas, y se muestran en las gráficas 5.34, 5.35 y 5.36. Como puede observarse, el protocolo se comporta de forma equitativa ante cualquier número de conexiones TCP.

5.6 COMPORTAMIENTO INTRAPROTOCOLO

Es importante conocer como se comporta un protocolo cuando distintas sesiones comparten un enlace. En esta sección, se presentan los resultados que muestran la equidad entre conexiones RCCMP. La topología que se va a utilizar es la misma que aparece en la figura 5.21. Para estas simulaciones, todos los enlaces tienen las mismas características: 10Mbps de ancho de banda, 10ms de retardo de propagación y una cola con capacidad para 50 paquetes (incluido el enlace etiquetado como enlace congestionado). El tamaño del grupo multipunto de una sesión RCCMP será de 3 receptores. Los experimentos consisten en aumentar el número de sesiones RCCMP para comprobar como comparten la capacidad del enlace congestionado. El índice de equidad y el porcentaje de ancho de banda, que cada sesión consume, son los indicadores de la equidad entre las diferentes sesiones. Se han hecho pruebas sin introducir pérdidas, e introduciendo pérdidas de canal con función de distribución uniforme con una tasa de 0.001 en el enlace congestionado. Además, para comprobar el comportamiento de las colas, se ha variado el tiempo de inicio de las fuentes RCCMP, comenzando todos los emisores en el mismo instante o desplazándolos 0.5s uno respecto a otro.

En estas experiencias no se producen cambios de representante, ya que todos los receptores ven las mismas pérdidas producidas por desbordamiento de las colas o por pérdidas aleatorias en el enlace congestionado.

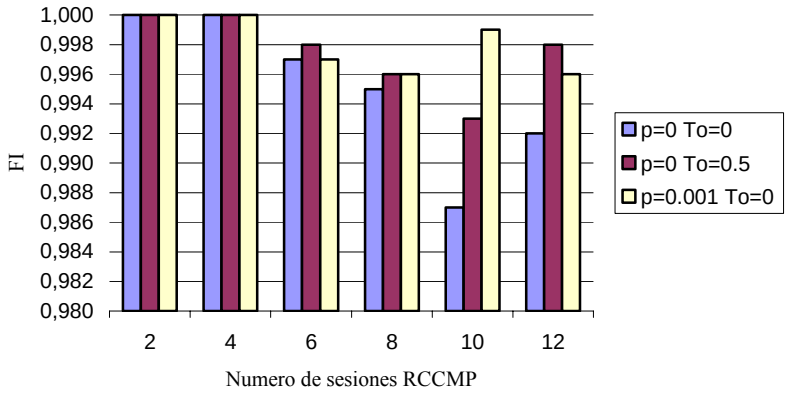

Figura 5.37 Índice de equidad para *i* sesiones RCCMP considerando enlaces sin pérdidas, con pérdidas y fuentes que comienzan en instantes distintos.

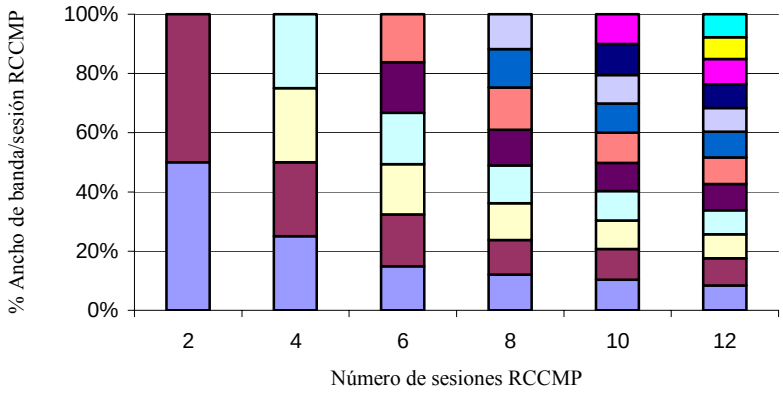

Figura 5.38 Reparto del ancho de banda para i sesiones RCCMP compartiendo el enlace sin pérdidas.

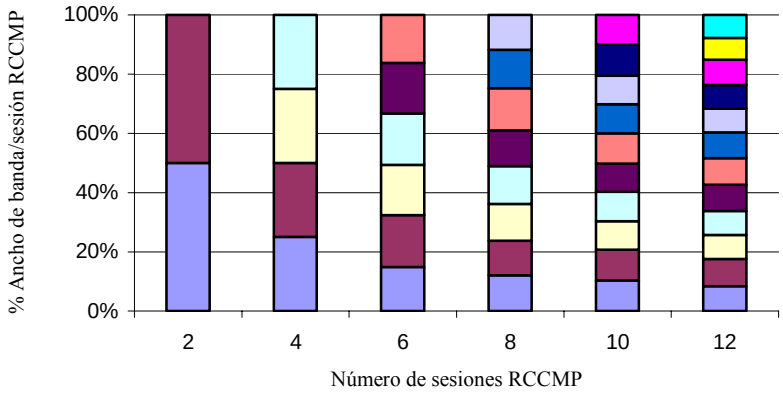

Figura 5.39 Reparto del ancho de banda para i sesiones RCCMP compartiendo el enlace con pérdidas de canal de 0.001.

Los resultados presentados en las figuras 5.37, 5.38 y 5.39 muestran que el enlace se comparte de forma equitativa entre todas las sesiones RCCMP. Si no se introducen pérdidas, cuando el enlace se reparte entre dos o cuatro sesiones, no se producen desbordamientos y el ancho de banda se reparte equitativamente entre todas las instancias. A partir de seis instancias el enlace comienza a saturarse.

Si se considera, que todas las fuentes comienzan simultáneamente, los índices de equidad, para las experiencias en las que se han introducido pérdidas, son ligeramente mejores que cuando no se introducen. Esto es debido al comportamiento de la cola FIFO que no trata por igual a todas las conexiones. Si se desplaza el comienzo de las fuentes en 0.5s, se observa que el reparto del ancho de banda es más equitativo y mejora el índice de equidad.

5.7 ESCALABILIDAD

Se puede considerar que un protocolo de transporte multipunto es escalable, cuando ni la información que se transmite desde los miembros del grupo, ni el número de variables de estado que almacenan emisores y receptores, crece de forma proporcional al número de usuarios que intervienen en la comunicación. En ese sentido, RCCMP es un protocolo en el que se define un número constante y pequeño de variables de estado por cada emisor y receptor. Además, la información de realimentación para controlar tanto el flujo de la comunicación como la posibilidad de congestión, proviene exclusivamente del representante, y por tanto, es perfectamente escalable. Sin embargo, las peticiones de retransmisión, NAKs, que pueden llegar desde todos los receptores, podrían generar un problema de escalabilidad. Para controlarlo, RCCMP incluye un esquema de temporizadores exponenciales.

Los experimentos, que se llevan a cabo en esta sección, tienen el objetivo de demostrar que el protocolo es totalmente escalable. Para ello, las diferentes pruebas determinan el número y tipo de paquetes de una sesión RCCMP, en función del número de receptores que forman el grupo multipunto para distintos valores de pérdida de canal y del tamaño de las colas.

Los diferentes tipos de paquetes se clasifican de la siguiente manera. Los enviados por la fuente son: paquetes de datos, paquetes de datos retransmitidos y paquetes de control (PIC). Dentro de la clasificación de paquetes de control se distingue a los PIC y a los IC, enviados estos últimos desde los receptores. Se llama tara a aquellos paquetes, que RCCMP debe incorporan para cumplir con los requerimientos de escalabilidad y fiabilidad, y que no aparecen en TCP. Así, se englobarían en este grupo los paquetes de control (IC y PIC) y los NAKs.

La topología, que se ha utilizado, para realizar los diferentes experimentos, es la mostrada en la figura 5.40. La fuente, que se sitúa en la raíz del árbol de distribución, se corresponde con una aplicación, que genera datos continuamente a una velocidad de 1Mbps. Los enlaces son todos iguales, y se caracterizan por un retardo de 1ms y una capacidad de 0.5Mbps. Las simulaciones tienen una duración de 400s. En general, el número de receptores varía desde 3 hasta 900.

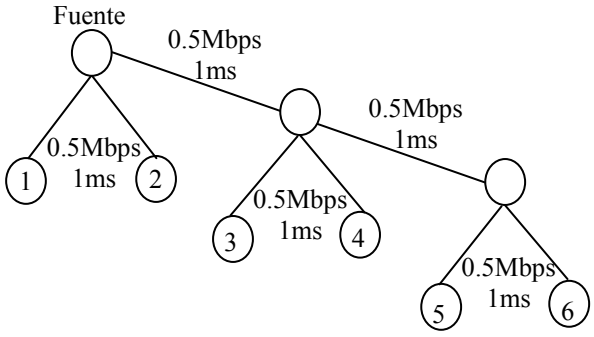

Figura 5.40 Topología con seis receptores para el estudio de la escalabilidad del protocolo RCCMP.

Para el primer experimento, se va a utilizar la topología anterior, en la que el número de receptores varía desde 3 hasta 3600. No se introducen pérdidas de canal en los enlaces, por lo que solamente se van a tener en cuenta pérdidas por saturación. Se consideran tamaños de colas de 90 paquetes. Los resultados se muestran en la figura 5.41.

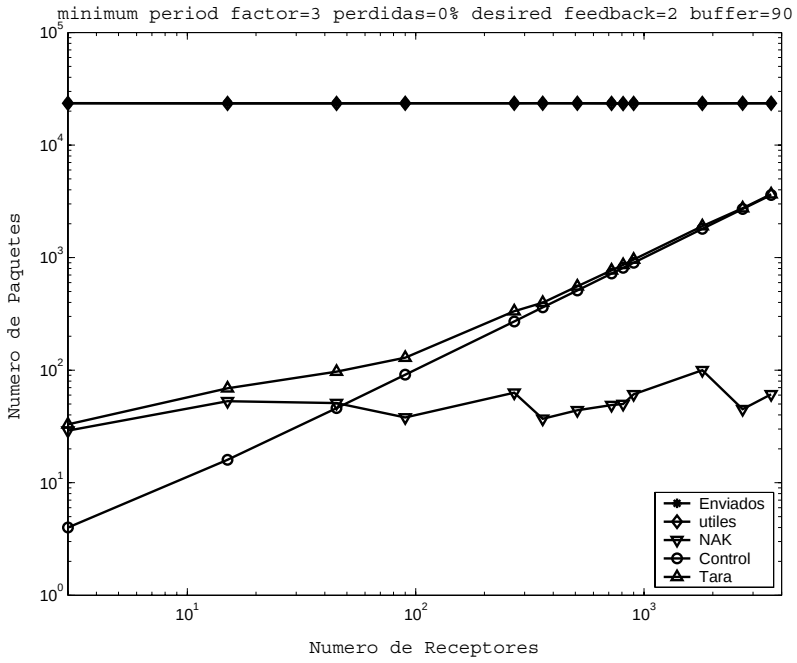

Figura 5.41 Número de paquetes de una sesión RCCMP para p=0 y tamaño de cola=90.

En esta gráfica, puede observarse como el número de paquetes enviados y útiles es independiente del número de receptores, y muestran valores muy cercanos. El porcentaje de paquetes retransmitidos obtenido es del 0.2%. En este escenario, las pérdidas debidas a saturación alcanzan el $1.5\ 10^{-3}$. Éstas se calculan como el cociente entre las indicaciones de pérdidas y el número de paquetes enviados. El número de NAKs no muestra dependencia con el número de receptores. Los cambios de representante producidos durante la sesión son en media de 12.

Los paquetes de control, por el contrario, crecen proporcionalmente al número de miembros del grupo multipunto. Los paquetes responsables de esta conducta son los informes de congestión (IC). La mayor aportación se produce en el proceso inicial de selección de representante. En este procedimiento, la fuente envía un primer paquete, PIC, que es respondido por todos los receptores, mediante un paquete IC. Otro proceso que también involucra a estos paquetes es la estimación del número de receptores activos del grupo multipunto, sin embargo RCCMP implementa un proceso para reducir el número de respuestas.

En la figura 5.41, se observa, como el principal contribuidor a la tara son los paquetes de control. Por lo tanto, al igual que estos paquetes, presenta un crecimiento lineal con el número de receptores. La tara supone en promedio un 0.35% de los paquetes útiles para grupos multipunto de tamaño hasta los 90 receptores, un 2.75% para grupos de hasta 900 miembros y un 11.8% hasta 3600 terminales.

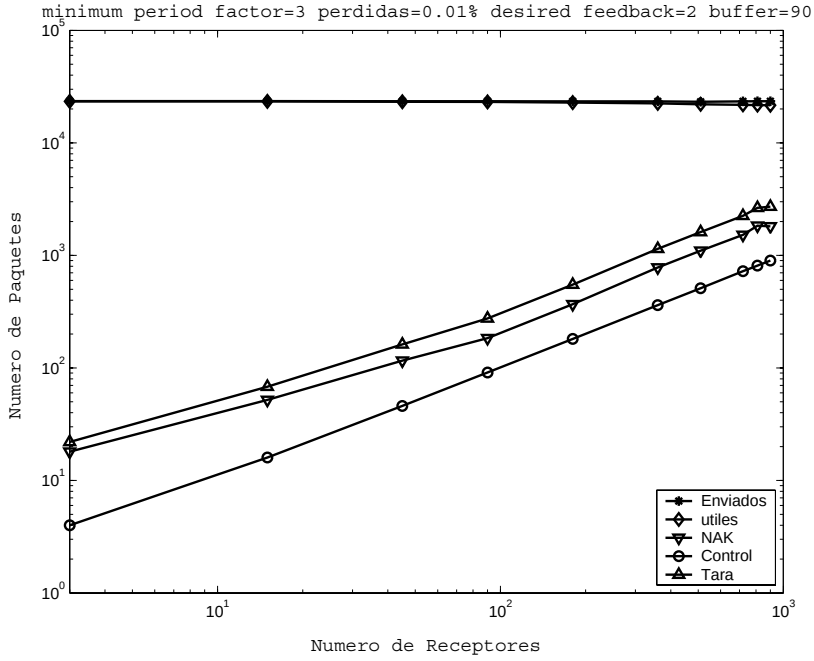

Figura 5.42 Número de paquetes de una sesión RCCMP para p=0.01% y tamaño de cola=90.

Si se consideran pérdidas en el canal, el número de paquetes en función del tamaño del grupo cambia. En la figura 5.42, se muestra el comportamiento de RCCMP, considerando pérdidas de canal con función de distribución uniforme e igual a 0.01% en todos los enlaces. En este caso, la principal diferencia con el experimento anterior, es el crecimiento lineal de los NAKs con el número de receptores. Si se contabiliza el número de estos paquetes por receptor, se obtiene en media unos 2.68. Lo que supone un ancho de banda de 0.0067 paquetes/s. Estos valores demuestran la escalabilidad del protocolo, en un escenario, en el que todos los receptores cuentan con los mismos recursos, por lo que no existe un representante claro. En este caso, las pérdidas totales, considerando las que se producen en las colas y en los enlaces, alcanzan en media un valor de 3.44%. Si se diferencia en función de los tamaños de los grupos, se obtiene que para un grupo multipunto de hasta 45 receptores, las pérdidas se mantienen en un 0.41%, mientras que si se

consideran grupos mayores a 720 receptores, éstas se elevan hasta el 7.18%. El número de cambios de representante es en media de unos 204 por sesión.

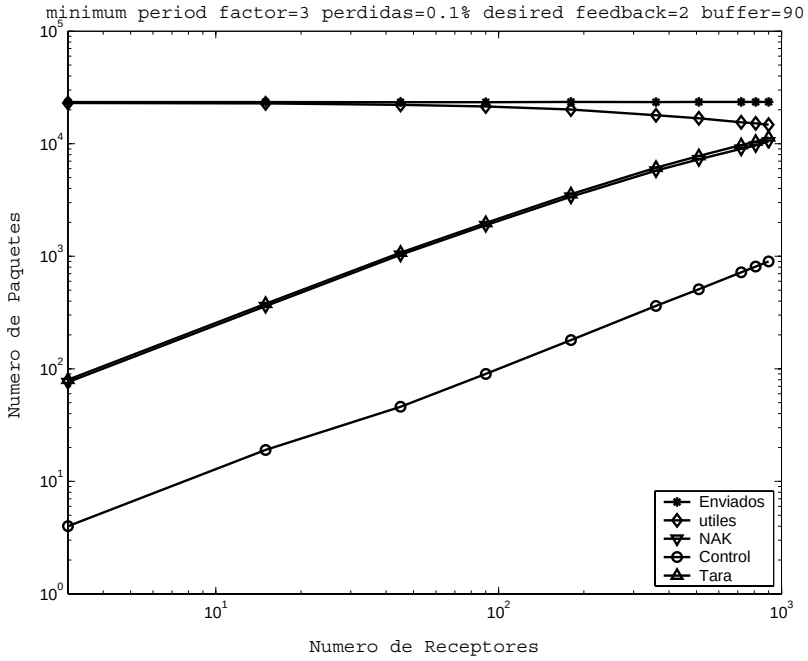

Figura 5.43 Número de paquetes de una sesión RCCMP para p=0.1% y tamaño de cola=90.

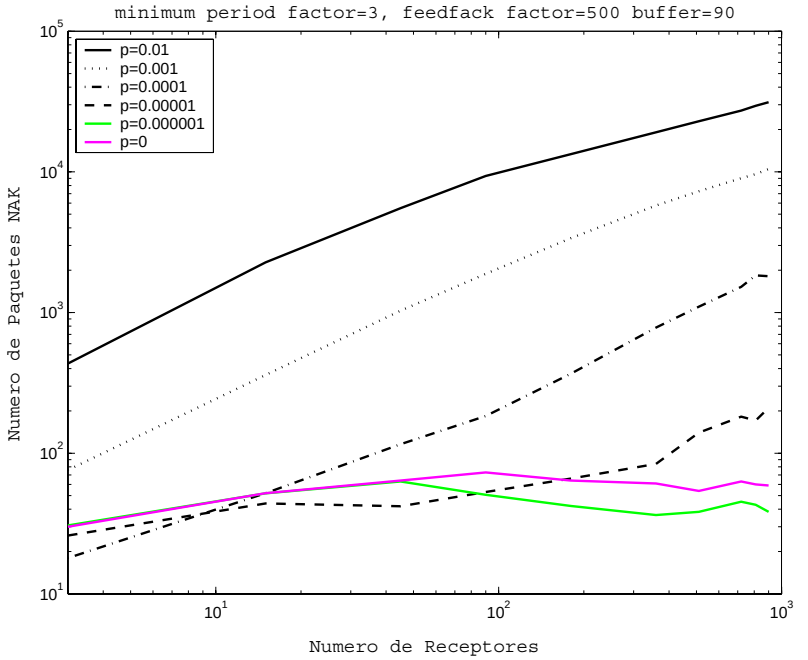

Figura 5.44 Paquetes NAK para distintos valores de pérdida de canal para un tamaño de cola=90.

Si se introducen pérdidas de canal con distribución uniforme del 0.1%, se obtiene la gráfica 5.43. En este experimento se observa el efecto anterior agudizado. El número de NAKs se incrementa drásticamente, y es que las pérdidas alcanzan el 18.7% de los paquetes enviados, de las que el 18.3% son detectadas en receptores que no son el representante. El número de NAKs por receptor es en media de 13.46 (0.033 paquetes/s). La fuente reduce el número de paquetes útiles al aumentar el número de receptores, ya que tiene que atender a las solicitudes de retransmisión. El número medio de cambios de representante por sesión es de 375, llegando a valores de 540 en el caso de grupos multipunto de 720, 810 o 900 miembros.

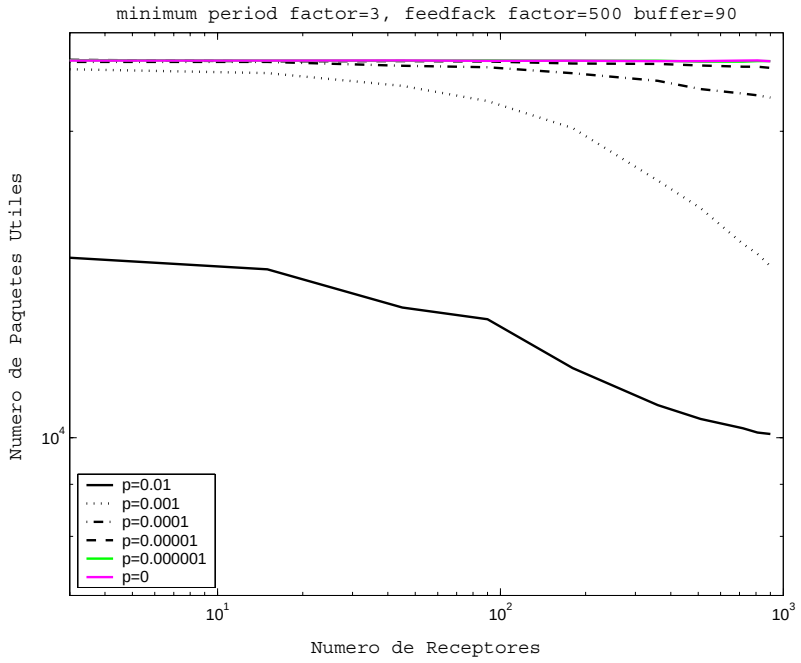

Figura 5.45 Paquetes útiles para distintos valores de pérdida de canal para un tamaño de cola=90.

A continuación, se presentan una serie de gráficas que muestran el número de paquetes útiles, de control y de NAKs para distintos valores de probabilidad de pérdidas que se introduce en el canal.

La figura 5.44 muestra el número de NAK, en función del número de receptores, para diferentes probabilidades de pérdida. Se puede ver como a partir de pérdidas de canal mayores a 0.01%, el número de NAKs crece drásticamente. La figura 5.45 muestra el número de paquetes útiles. Aquí, puede observarse, como al aumentar la probabilidad de pérdidas, la eficiencia de la comunicación decrece. Para pérdidas de canal mayores al 0.1%, los paquetes útiles se reducen de forma importante. La figura 5.46 muestra el número de paquetes de control. En este caso, se observa que para una probabilidad de pérdida del 1%, el número de paquetes de control para grupos multipunto de tamaño inferior a la centena de usuarios es superior a la del resto de simulaciones. Esto es debido, a que las altas pérdidas, hacen que no lleguen a la fuente los paquetes de control respuesta, y haya que volver a repetir el proceso de

solicitud de informes. Para el resto de experimentos, se comprueba que el número de paquetes de control es independiente de las pérdidas.

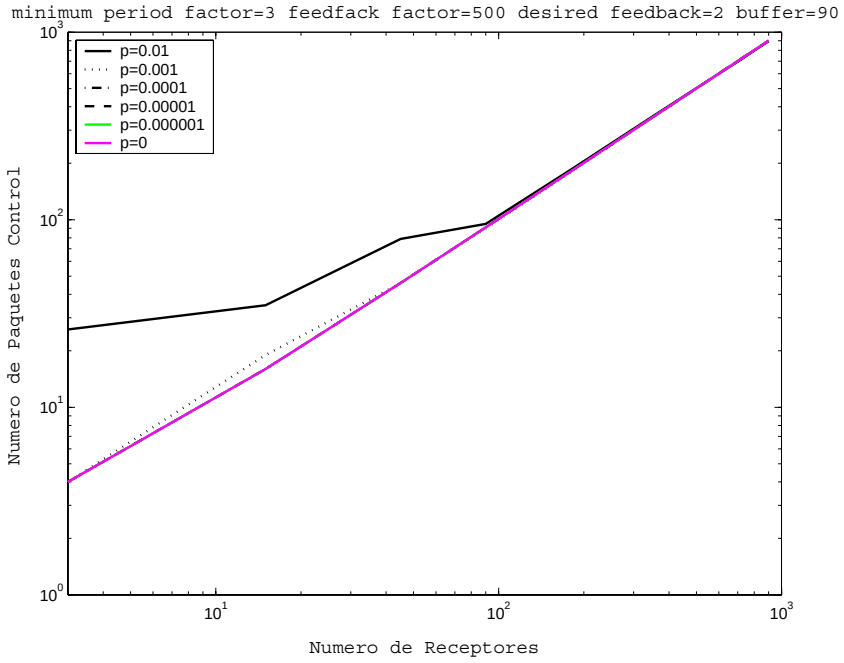

Figura 5.46 Paquetes control para distintos valores de pérdida de canal para un tamaño de cola=90.

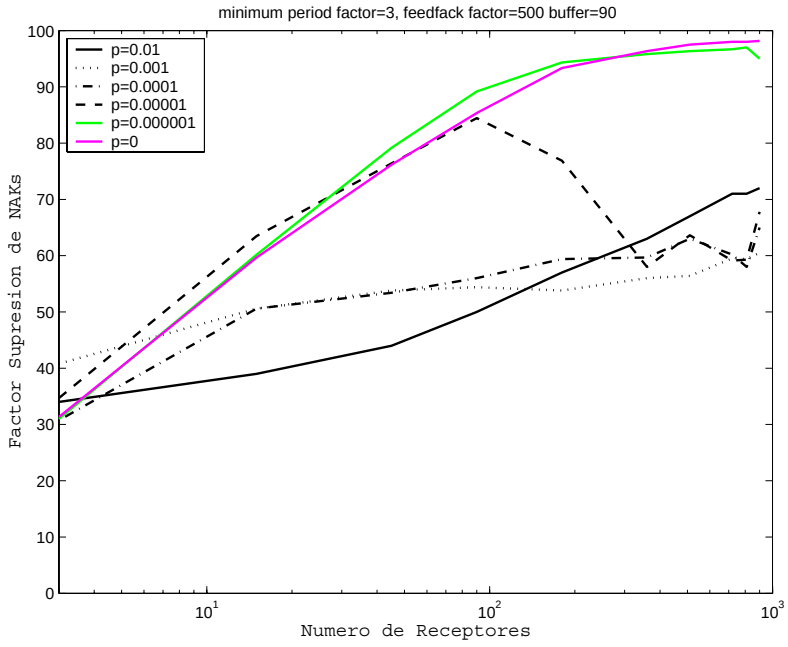

Figura 5.47 Factor de supresión de NAKs para distintos valores de pérdida de canal para un tamaño de cola=90.

La figura 5.47 muestra el factor de supresión de NAKs en función del número de receptores para distintas probabilidades de pérdida. Este índice se calcula como el cociente entre los NAKs suprimidos y los generados. Lo primero que salta a la vista de estas curvas es que cuanto mayor es el número de receptores se suprimen más NAKs. Además, la supresión es mayor cuando las pérdidas de canal tienen una tasa menor, es decir, cuando las pérdidas totales que se producen en la sesión están dominadas por las pérdidas que se producen en las colas. Sin embargo, puede observarse que para pérdidas del 1%, el factor de supresión supera el 70% para tamaños de grupos importantes. En un escenario, como el que se está estudiando, en el que todos los enlaces son iguales, y con fuertes pérdidas, el esquema de temporizadores exponenciales asegura la escalabilidad del protocolo.

Otro punto de vista a la hora de evaluar la escalabilidad de RCCMP es estudiar como se comporta el protocolo ante la modificación del tamaño de las colas, que afecta a la probabilidad de pérdida por saturación. Se va a utilizar la topología de la figura 5.40 con tamaños de buffer de 30, 60, 90 y 180 paquetes. Los resultados obtenidos muestran el número de paquetes útiles, NAKs y de control para distintos valores de pérdida de canal en el enlace. Como resulta lógico, los mejores resultados se obtienen para mayores tamaños de cola.

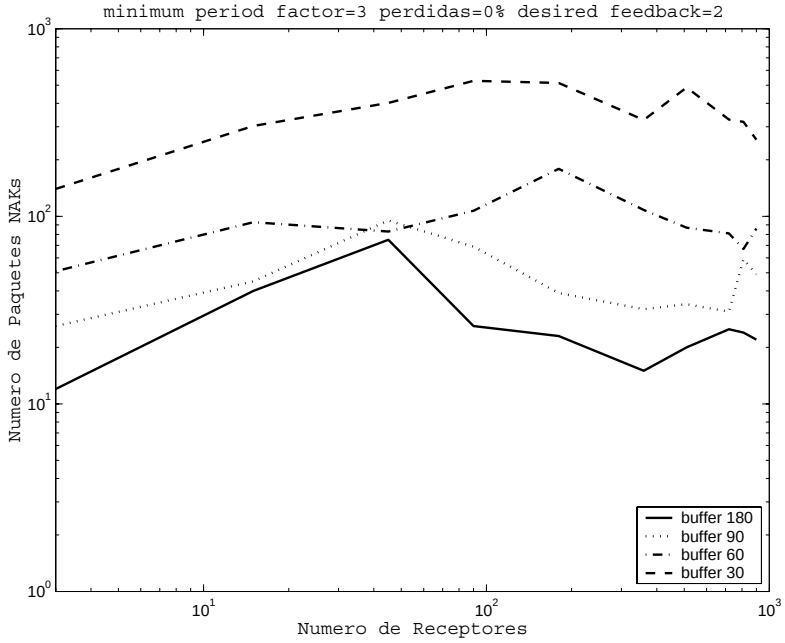

Figura 5.48 Paquetes NAKs para distintos tamaños de cola y sin pérdidas en el canal.

En primer lugar, se presentan las gráficas en el caso de que no se introduzcan pérdidas de canal. La figura 5.48 muestra el número de NAKs en función del número de receptores para distintos tamaños de cola. Puede observarse, como el número de paquetes aumenta para capacidades más pequeñas.

El número de paquetes útiles también resulta depender del tamaño de las colas. A mayor capacidad, mayor número de paquetes útiles, ya que la fuente reduce el

número de paquetes retransmitidos. Este comportamiento puede observarse en la figura 5.49. La siguiente gráfica, 5.50, muestra el número de paquetes de control. Éstos se muestran independientes del tamaño de la cola para tamaños del grupo multipunto superiores a 45 miembros. Sin embargo, para tamaños de grupo y de colas pequeños, se observa que el número de paquetes de control es superior que si se utiliza un tamaño de cola mayor. En estos casos, las peticiones de respuesta que solicita la fuente durante la sesión se multiplican por tres. Este proceso está regulado por el tiempo *RTT_representante*feedback_factor*. Como *feedback_factor* es una variable que está ajustada a 500 para todas las simulaciones, el retardo de ida y vuelta entre la fuente y el representante para tamaños de cola y de grupo pequeños, es menor, por lo que se solicita más veces a estos receptores que respondan.

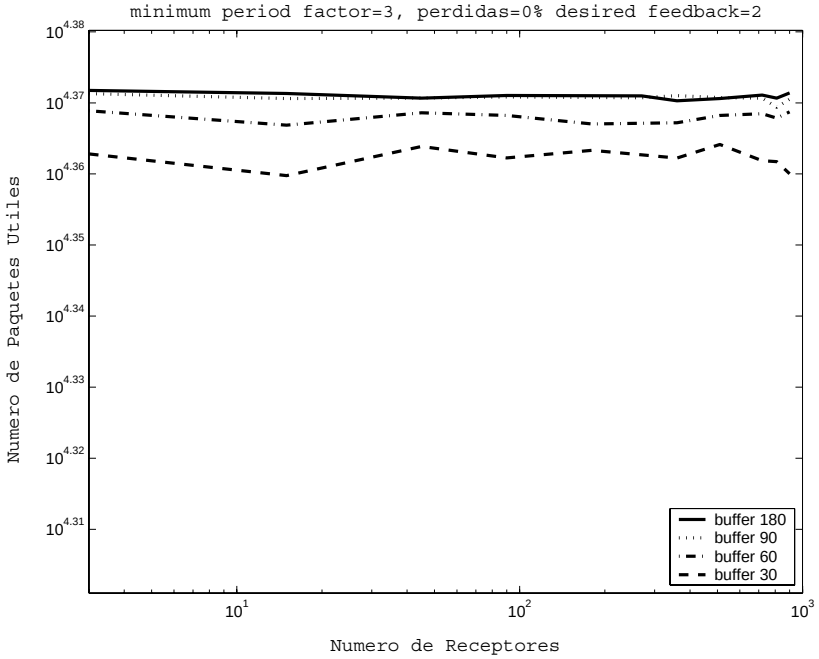

Figura 5.49 Paquetes útiles para distintos tamaños de cola y sin pérdidas en el canal.

Si se introducen pérdidas de canal de valores 0.01% y 0.1%, se obtienen las gráficas 5.51 y 5.52, que muestran el número de NAKs para diferentes tamaños de cola. Para pérdidas de canal del 0.01%, el número de confirmaciones negativas es creciente con el número de receptores y con menores capacidades. Sin embargo, para grupos multipunto de tamaño superior a las centenas de usuarios, el número de NAKs se muestra independiente de la longitud de la cola. Este comportamiento coincide con el que aparece en la figura 5.52, simplemente, que en este caso, se avanza para grupos multipunto de tamaño mayor a alguna decena de usuarios. La causa de esta respuesta responde a que cuando el número de receptores aumenta, el término dominante de las pérdidas es debido a las pérdidas aleatorias que se introducen en el canal. Por lo tanto, el tamaño de la cola, que regula las pérdidas por saturación, no afecta el número de NAKs cuando las pérdidas de canal son importantes. De hecho, el número de paquetes perdidos, en otros receptores que no son el representante, aumenta con el número de receptores. Así, este factor se

convierte en el término dominante de las pérdidas, y esto, como se ve en las figuras 5.53 y 5.54, hace disminuir el número de paquetes útiles.

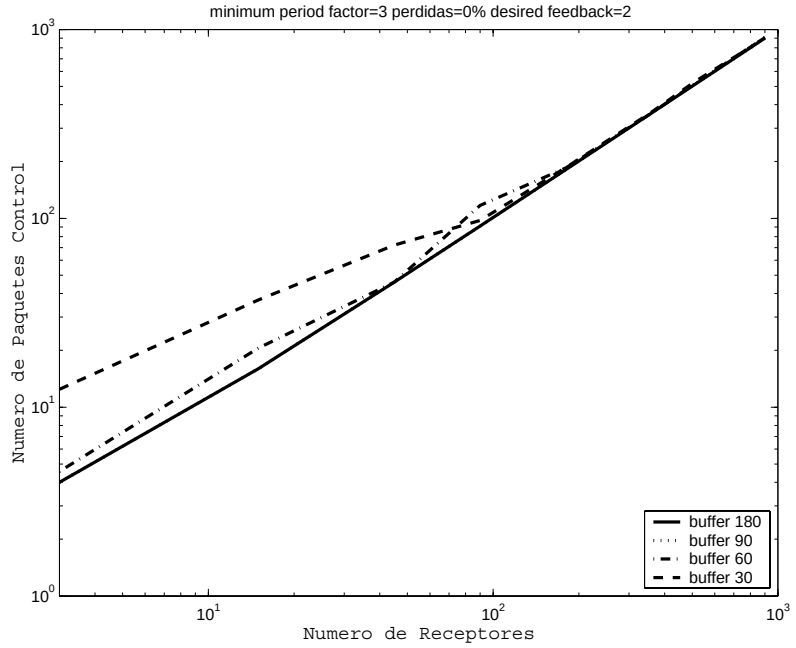

Figura 5.50 Paquetes control para distintos tamaños de cola y sin pérdidas en el canal.

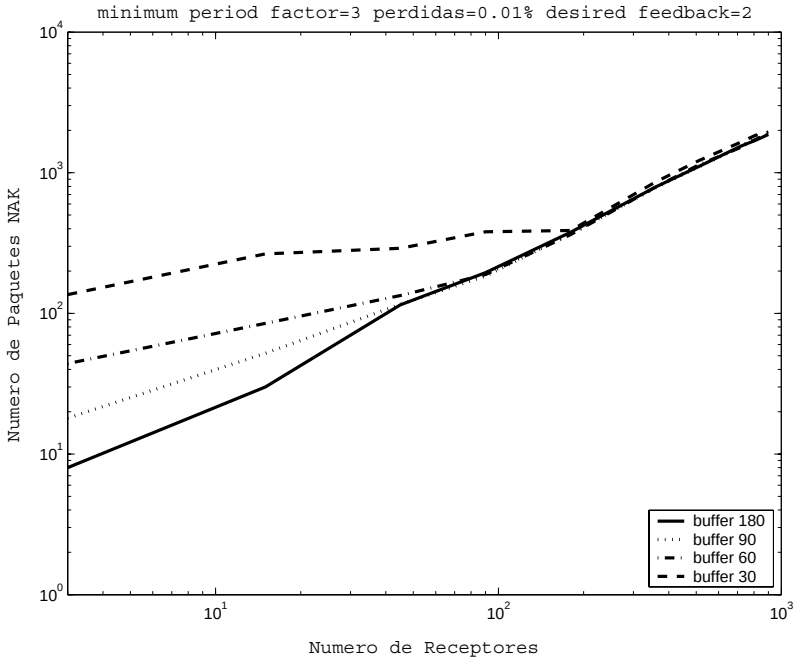

Figura 5.51 Paquetes NAKs para distintos tamaños de cola y pérdidas en el canal de 0.01%.

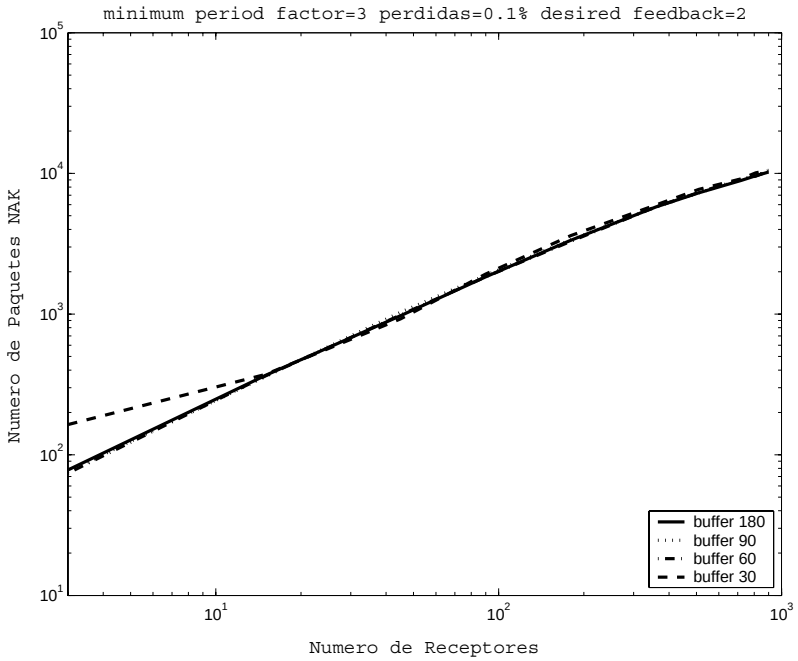

Figura 5.52 Paquetes NAKs para distintos tamaños de cola y pérdidas en el canal de 0.1%.

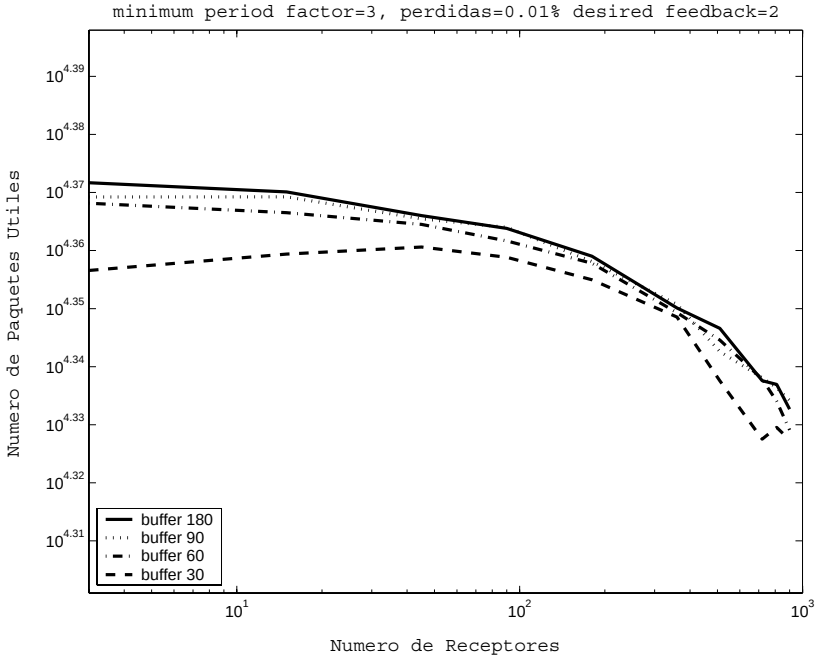

Figura 5.53 Paquetes útiles para distintos tamaños de cola y pérdidas en el canal de 0.01%.

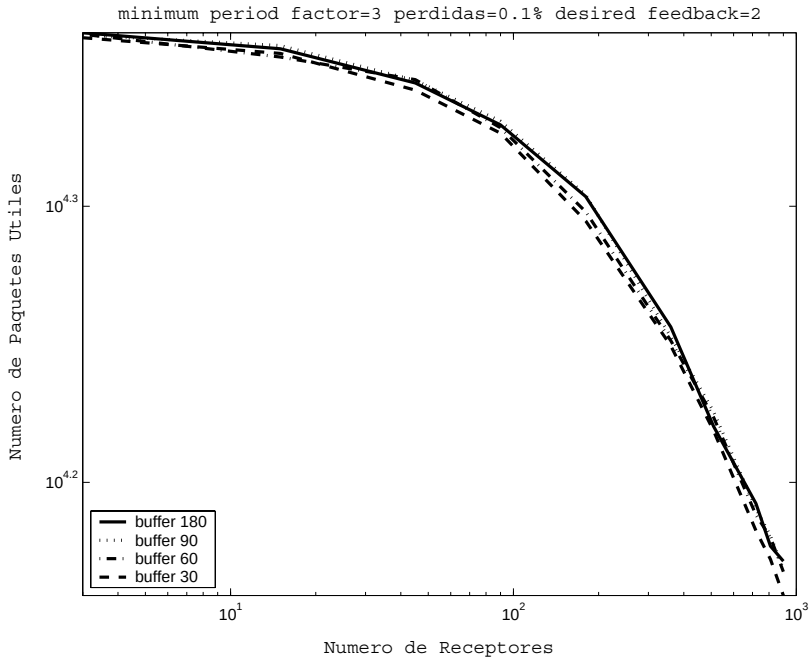

Figura 5.54 Paquetes útiles para distintos tamaños de cola y pérdidas en el canal de 0.1%.

Si bien, se ha observado que en general, los resultados obtenidos mejoran con el tamaño de la cola, también se ha de tener en cuenta que esto introduce mayores retardos en el protocolo. Además, en un entorno con pérdidas, los resultados han mostrado, que un número suficiente de receptores, anula el efecto de reducir la probabilidad de pérdida de las colas, ya que el mayor número de paquetes perdidos es debido a las pérdidas aleatorias que se producen en los enlaces.

5.8 COMPARACIÓN DE RCCMP CON OTRAS PROPUESTAS

Distintas propuestas de protocolos han sido diseñadas para proveer un servicio multipunto fiable. Diversos autores, [Diot97], [Obraczka98] y [Atwood04], han sintetizado y clasificado las más interesantes. El primer trabajo resume los protocolos multipunto en términos de funciones y mecanismos para la transmisión multipunto fiable. El segundo, revisa y clasifica las diferentes propuestas a partir de unos determinados criterios. Y el tercer trabajo, más reciente, divide en diferentes categorías los protocolos multipunto fiable, en base a los requerimientos del usuario, y a las arquitecturas, mecanismos y modelos de comunicaciones usados para satisfacer esas necesidades.

Los requerimientos, propuestos por J. W. Atwood, que caracterizan y diferencian a los protocolos de transporte multipunto fiable son: el número de fuentes, el grado de escalabilidad y fiabilidad, si incluyen control de congestión, el desarrollo de un protocolo de gestión de grupo y el nivel de ordenación de los paquetes. En base a estos criterios, clasifica distintos protocolos multipunto. En la tabla 5.3 se transcribe su clasificación a la que se ha añadido el protocolo RCCMP.

Protocol	# Fuentes	Escalabilidad	Fiabilidad					Control Congestión	Gestión grupo	Orden
			Débil				Fuerte			
			BE	BL	MR	RC	AB			
UDP	M	Grande	X					No	No	No
XTP 3.6	1	Media	X			X		No	Débil	No
XTP 4.0	1	Pequeña	X				X	Si	Fuerte	No
RMP	M	Media					X	Si	Fuerte	Si
SRM	M	Grande				X		No	Débil	Appl
PGM	M	Media				X		No	No	No
LGMP	1	Grande				X		Si	Débil	No
RMTP	1	Grande				X		Si	No	No
RMTP-II	M	Grande		X		X		Si	Débil	Appl
LPC	1	Enorme				X		No	No	No
RCCMP	1	Grande				X		Si	Débil	No

BE= Best Effort — BL= Bounded latency

MR= Most recent — RC= Receiver-Centered

AB= Absolute — Appl= application

Tabla 5.3 Clasificación de protocolos multipunto fiable en base a los requerimientos de usuario [Atwood04].

En principio, RCCMP ha sido diseñado para trabajar con una única fuente, sin embargo, debido a su sencillez y al coste de la comunicación, podría trabajar con N fuentes independientes. En cuanto a la escalabilidad, el protocolo es orientado a NAK, y el uso de temporizadores para suprimir y evitar una sobrecarga de información de realimentación, hacen que su escalabilidad sea grande en función de los criterios que establece Atwood.

El autor define un servicio fiable fuerte, donde todos los paquetes enviados por la fuente se reciben en orden y sin duplicación por los receptores. Este tipo de fiabilidad tiene sentido para grupos pequeños. Para grupos más numerosos, este concepto debe ser relajado, por lo que aparece la noción de fiabilidad débil. De las clasificaciones que propone el autor, el protocolo RCCMP se ajusta a la denominada fiabilidad centrada en el receptor (*Receiver-Centered Reliability*). En los protocolos que están bajo esta denominación, el emisor no tiene otra responsabilidad en la recuperación del error que la de proveer la retransmisión del paquete, dejando la responsabilidad del proceso en manos de los receptores.

El mecanismo de control del error de RCCMP es función de qué receptor detecta la pérdida. Si no es el representante, el proceso de recuperación que se incluye en el protocolo, se ajusta adecuadamente a la clasificación anterior. Sin embargo, si es el representante quien detecta la pérdida, el emisor debe tomar un papel más activo en el control del error, ya que el proceso que se ejecuta es similar al del protocolo TCP. Así, si un paquete es perdido por varios receptores en los que se incluye el

representante, solamente este último deberá realizar la recuperación del error, suprimiendo las confirmaciones negativas, y por lo tanto mejorando la escalabilidad.

Protocol	Detección del Error							Recuperación Error		
	Distribución		Política					Distribución		Política
	uc	mc	ACK	NACK	Supp	SNACK	TRACK	uc	mc	
XTP 3.6		X	X	X	X				X	SR
XTP 4.0	X		X	X					X	SR
RMP		X	X	X					X	SO
SRM		X		X	X				X	FRO
PGM		X		X	X				X	HRO
LGMP	X	X	X	X		X		X	X	HRO
RMTP	X		a	X				X	X	HRO
RMTP-II	X		X	X			X	X	X	HRO
LPC	X			X					X	HFEC
RCCMP	X	X	X	X	X				X	SO

uc: Unicast mc: Multicast Supp: Suppresion of acknowledgments

SNACK: Semi-NACK TRACK: Tree based acknowledgment SR: Sender-reliable

SO: Receiver-reliable, sender oriented FRO: Receiver-reliable, flat receiver-oriented

HRO: Receiver-reliable, hierarchinal receiver-oriented HFEC: Hybric forward error correction

Tabla 5.4 Clasificación de protocolos multipunto fiable en función del control de error [Atwood04].

Atwood propone otra clasificación más exhaustiva de la fiabilidad de los protocolos en función de cómo se realiza la detección y la recuperación del error. La tabla 5.4, modificada al introducir las características de RCCMP, es una reproducción de su trabajo.

En RCCMP, el receptor, a partir del número de secuencia del paquete, detecta que ha habido una pérdida, entonces envía vía punto a punto a la fuente un NAK para solicitar el paquete perdido. Sin embargo, también utiliza los ACKs, que envía de forma multipunto el representante, para informar de qué paquetes están siendo recibidos correctamente. Estos ACKs son utilizados para regular la tasa del emisor en base a las condiciones del peor miembro del grupo multipunto, y para preservar la fiabilidad, en caso de que las pérdidas se produzcan en este receptor. Por tanto, se puede clasificar a RCCMP como un protocolo que utiliza la distribución punto a punto y multipunto para informar de que se ha producido un error. En el control de error involucra a NAKs y ACKs, y además incluye un mecanismo de supresión de realimentación.

A propósito de cómo se produce la recuperación del error es por medio de la retransmisión del paquete perdido que se distribuye de forma multipunto. El esquema que mejor describe a RCCMP, según Atwood, sería *Sender-oriented*. Bajo esta clasificación se agrupan los protocolos fiables multipunto cuyos receptores envían un

NAK a la fuente, que es quien realiza la retransmisión. Sin embargo, como ya se ha visto, esta clasificación no es representativa de todo el proceso de recuperación del error del protocolo.

El trabajo de Atwood no clasifica los controles de congestión, simplemente, determina si las diferentes propuestas los incluyen. En cuanto a la clasificación en función de la gestión de grupo, el autor determina diferentes escalas en función del grado de conocimiento que se requiere de los miembros del grupo. RCCMP necesita conocer el número de receptores que forman parte del grupo multipunto, en aras de ajustar convenientemente los temporizadores exponenciales. Por ello, se podría clasificar a esta propuesta como que realiza una gestión del grupo débil. Sin embargo, como se ha visto en el capítulo anterior, la dependencia con el número de usuarios es logarítmica, por lo que no es necesario un conocimiento muy exacto.

Por otro lado, la necesidad de un mecanismo que realice un ordenamiento global tiene sentido para comunicaciones multipunto a multipunto, donde los paquetes de los diferentes emisores deben ser convenientemente organizados. RCCMP no incluye ningún tipo de mecanismo a tal propósito.

5.8.1 Clasificación del control de congestión de RCCMP

El control de congestión de RCCMP ha sido diseñado con el objetivo principal de mostrar un comportamiento lo más parecido a TCP para asegurar la equidad. Este es uno de los requerimientos fundamentales que debe cumplir cualquier protocolo que vaya a ser desarrollado en Internet. Jörg Widmer en [Widmer01a] examina y evalúa las principales propuestas de controles de congestión, equitativos con TCP, para punto a punto y multipunto. La tabla 5.5 clasifica los mecanismos multipunto y de tasa única en base a ciertos criterios propuestos por el autor. Se ha modificado esta clasificación para añadir el control de congestión que incluye el protocolo RCCMP.

Protocolo	Tasa /ventana	Asistencia de la red	Complejidad	Perfil de la tasa	Sesgo contra altos RTT	Equidad con TCP
TEAR	tasa	extremo-extremo	Baja	Suave	Si	Buena
RLC & LPR	ventana	extremo-extremo	Baja	Sierra	Si	Buena
MTCP	ventana	extremo-extremo	Baja	Sierra	Si	Buena
NCA	ventana	Requerida	Baja	Sierra	Si	Buena
PGMCC	ventana	Requerida	Media	Sierra	Si	Buena
RCCMP	ventana	extremo-extremo	Baja	Sierra	Si	Buena

Tabla 5.5 Clasificación de controles de congestión multipunto de tasa única [Widmer01a].

Basado en los puntos que propone J. Widmer, el control de congestión de RCCMP puede ser clasificado como de ventana deslizante, ya que éste es el método que utiliza para regular la tasa de la fuente. Por otra parte, no requiere de asistencia

por parte de la red para la agregación de la información de realimentación, estimación del tiempo de ida y vuelta (RTT) o la modificación de las estrategias de las colas de los nodos intermedios. Es un control de congestión extremo a extremo, y solamente es necesaria la colaboración de los receptores, en especial, del representante.

La complejidad del control de congestión es el propio de un mecanismo de ventana deslizante punto a punto, ya que este mecanismo está gobernado por la fuente y el representante, mediante los ACKs acumulativos que este último envía. Con respecto a un mecanismo para punto a punto, RCCMP añade el procedimiento para el cambio de representante. Por lo tanto, la complejidad es baja.

El autor clasifica los diferentes mecanismos en función de su comportamiento en régimen permanente en un entorno con pérdidas. Esto es útil, para establecer qué controles de congestión son capaces de soportar aplicaciones de difusión de audio o video, que envían flujos de información a tasa aproximadamente estable. RCCMP, como emula al máximo posible el comportamiento de TCP, tiene una conducta parecida a este protocolo, por lo que presenta un comportamiento oscilatorio de diente de sierra. Sin embargo, el protocolo RVCMP (*Reliable Vegas Congestion controlled Multicast Protocol*) [Solera05d] permite suavizar este perfil, ya que estima el ancho de banda disponible.

Otro de los criterios de clasificación es si los controles de congestión, muestran como TCP, una tasa que se degrada con el RTT. Como se observa en la tabla todos los mecanismos evaluados por J. Widmer cumplen esta condición, RCCMP también. Finalmente, los resultados presentados muestran que la equidad con TCP es buena.

En el apéndice E se compara de forma más exhaustiva el protocolo RCCMP con otros protocolos de transporte como PGM, y otros controles de congestión multipunto como PGMCC y ORMCC.

5.9 CONCLUSIONES

Los experimentos presentados mediante simulación muestran que RCCMP es un protocolo de transporte multipunto fiable que funciona razonablemente bien en cuanto a dos aspectos fundamentales como la equidad y la escalabilidad.

RCCMP cuenta con una serie de parámetros, que deben ser ajustados por el administrador, para conseguir que el comportamiento del protocolo se ajuste convenientemente al escenario donde se va a producir la comunicación. Sin embargo, se ofrece un conjunto de valores para estas variables que cubren un extenso rango de trabajo. Las pruebas realizadas muestran que existe un compromiso entre la dinámica del protocolo y escalabilidad.

El control de congestión ajusta la tasa de la fuente a la capacidad del receptor más lento. Éste hay que elegirlo mediante el proceso de selección de representante. Los resultados muestran que el caudal, que la fuente RCCMP inyecta a la red, se adecua a la aparición de receptores más lentos. La velocidad con la que el protocolo se adapta a las nuevas condiciones, depende por un lado del tiempo que tarda la fuente en conocer que existe un receptor de menos recursos, y por otro de la dinámica de la ventana de transmisión.

En cuanto a la equidad con TCP, el control de congestión ha sido diseñado con esta especificación, por lo que las simulaciones muestran unos buenos resultados en diversas circunstancias de ancho de banda, retardo y pérdidas. La equidad entre sesiones RCCMP también ha sido evaluada, y las pruebas presentan unos buenos índices.

En cuanto a la escalabilidad, la información de realimentación tanto para controlar la tasa del grupo como para indicar congestión, proviene únicamente del representante, por lo que es perfectamente escalable. Además, el protocolo incluye un mecanismo de supresión de la información de realimentación para prevenir una respuesta masiva por parte de los receptores, cuando un paquete ha sido perdido por un conjunto de terminales que no incluyen al representante. Con este fin, se han incorporado los temporizadores exponenciales. Éstos son exclusivamente locales a los receptores, y funcionan en colaboración con el control de congestión. Los resultados muestran que el factor de supresión de NAKs crece en función del número de receptores, y que en escenarios con fuertes pérdidas, el número de NAK cancelados supera el 70%. Por lo que RCCMP es un protocolo adecuado para trabajar en entornos de grupos multipunto de tamaño de cientos de miles de usuarios.

CAPÍTULO 6
MODELADO DEL CAUDAL DEL PROTOCOLO RCCMP

6.1 INTRODUCCIÓN

Distintos estudios han tratado de conocer con precisión el protocolo TCP para mejorar la utilización de los recursos en Internet. Tradicionalmente, estos trabajos se basaban en simulaciones y en medidas sobre la propia red; sin embargo, a mediados de los noventa, aparecen los primeros modelos analíticos que estudian el caudal de un flujo TCP en función del tiempo de ida y vuelta y de la probabilidad de pérdida: [Ott97], [Madhavi97], [Mathis97], [Padhye98], y [Samios03], entre otros. Los tres primeros desarrollan un modelo analítico para la fase permanente (*congestion avoidance*) del control de congestión TCP, teniendo en cuenta solamente que el emisor detecta pérdidas de paquetes mediante la recepción de más de tres reconocimientos con el mismo número de secuencia. El trabajo de J. Padhye, más completo, incorpora los efectos del mecanismo de retransmisión por efecto de la expiración del temporizador. El trabajo de [Samios03], más reciente, estudia analíticamente la implementación TCP Vegas, y obtiene un modelo, teniendo en cuenta los dos mecanismos de recuperación anteriores.

Uno de los principales objetivos, a la hora de estudiar los flujos TCP, es modelar su comportamiento en función de los parámetros de la pérdida de paquete y del RTT, para incorporar este funcionamiento en tráfico no TCP. Tanto los protocolos punto a punto como multipunto deben competir por los recursos de red con TCP, por lo que es importante que incluyan mecanismos de control de congestión compatibles con este protocolo. Así, una de las especificaciones de diseño más importantes, para que un protocolo funcione en Internet, es que se comporte ante la congestión como lo haría una conexión TCP. Y es que, en general, las aplicaciones multipunto tienen el potencial de saturar la red más de lo que lo haría una aplicación punto a punto. Mientras que un flujo punto a punto se distribuye entre un emisor y un receptor, un árbol de distribución multipunto podría alcanzar toda la red.

Las aplicaciones de distribución de video y audio multipunto, que ofrecen un servicio no fiable, generalmente, cuentan con usuarios que monitorizan la calidad de la comunicación, y abandonan la aplicación, cuando las imágenes o la voz se degradan. Estos usuarios actúan como controles de congestión, ya que responden a la saturación de la red, abandonando la conexión. Sin embargo, en aplicaciones como transmisión de

ficheros o descarga de ficheros ejecutables, que necesitan de un servicio fiable, no es habitual tener usuarios al otro extremo del terminal. Además, la duración de la transmisión puede ser muy larga, y podría darse una sobrecarga de paquetes de control, ACKs y NAKs. Todos estos condicionantes hacen que sea muy importante desarrollar controles de congestión en los protocolos de transporte multipunto fiable para evitar el colapso de la red. Hoy en día, la manera habitual de prevenir la saturación de Internet es mediante los mecanismos de control de congestión de la familia TCP.

No todos los protocolos de transporte multipunto fiable publicados como RFCs incluyen controles de congestión. Esto es debido a que se desarrollaron antes de que este requerimiento se considerara como uno de los aspectos más importantes para su estandarización [RFC2357]. Sin embargo, en los últimos años las propuestas de controles de congestión para entornos multipunto se han ido sucediendo. En [Widmer01a] se encuentra un resumen.

Los controles de congestión pueden ser clasificados en base a diferentes características. En función del número de flujos que envía la fuente, se distinguen los controles de congestión de tasa única o multitasa. En los primeros, los datos se envían en un solo flujo a todos los receptores a la misma velocidad, mientras que en los segundos, el emisor divide los datos a transmitir en distintos niveles o capas, y transmite cada uno de ellos a un grupo multipunto distinto. Los receptores se asocian a uno o más grupos multipunto en función de sus recursos. De esta manera, se consigue una asignación de ancho de banda más flexible y una mejor escalabilidad que en los primeros; sin embargo, son mucho más complejos y pueden sufrir de falta de equidad con TCP. Otra posible división se basa en si los controles de congestión utilizan un mecanismo basado en tasa o de ventana deslizante, para ajustar la velocidad de la fuente. Las propuestas que se engloban dentro de la clasificación de tasa única y basados en ventana deslizante sufren de dos problemas principalmente: el problema de la reducción drástica de la tasa de la fuente debido a las señales de congestión que llegan de los receptores, y el problema de la implosión de NACKs. Las siguientes propuestas PGMCC [Rizzo00], NCA [Kasera00], MTCP [Rhee99], RLA [Wang98] y la desarrollada por Golestani [Golestani99] tratan de superar de diferentes formas estas dificultades. Todas estas propuestas ajustan la ventana de transmisión de forma similar a TCP. Algunas crean estructuras jerárquicas con la cooperación de receptores y nodos intermedios, para agregar la información de realimentación que proviene de los receptores, [Golestani99] y [Rhee99]. Otras seleccionan al receptor de menos recursos como el representante del grupo para controlar la ventana, [Rizzo00] y [Kasera00]. El control de congestión, que incluye el protocolo de transporte multipunto fiable RCCMP, también emula a TCP a partir de la información de realimentación que le llega desde un representante.

En este capítulo se va a modelar el control de congestión que incluye RCCMP. Para ello, se analizará, en estado permanente *(congestion avoidance)*, el caudal inyectado por la fuente RCCMP en función de la probabilidad de pérdida y del RTT. Como el control de congestión de RCCMP emula al de TCP, el estudio que aquí se propone está basado en el trabajo de [Padhye98]. La principal diferencia con este trabajo radica en la caracterización del caudal ante cambios de representante. Ante la aparición de un receptor con menos recursos que los del representante actual, la tasa de la fuente debe ser convenientemente ajustada. Finalmente, para comprobar la bondad del modelo matemático, se validarán los resultados mediante simulaciones.

6.2 ANÁLISIS DEL CAUDAL EN FUNCIÓN DEL TIEMPO DE IDA Y VUELTA Y DE LA PROBABILIDAD DE PÉRDIDA

El análisis del caudal en función del tiempo de ida y vuelta (RTT) entre la fuente y el representante y de la probabilidad de pérdida se va a estructurar de la siguiente manera. Primero, se describe el modelo de control de congestión. En este apartado, se introducen los parámetros y variables que se van a utilizar a lo largo del estudio. Segundo, se modela el caudal bajo la hipótesis de que todas las pérdidas se detectan en el representante, independientemente, de si se producen también en otros receptores. Tercero, se construye un modelo suponiendo que las pérdidas se producen en cualquier receptor. En este caso, la detección de pérdidas puede producirse en el representante y en otros receptores, o solamente en otros receptores. Finalmente, se analiza el caudal en el caso de cambio de representante. Este último apartado es el que mayores diferencias implica respecto a las comunicaciones punto a punto, ya que en este caso, se producen variaciones en el caudal, cuando aparece un receptor de menos recursos. Por lo tanto, se va a construir un modelo estocástico, que modele el comportamiento de RCCMP en régimen permanente *(congestion avoidance)*, con bajas o moderadas pérdidas, en el caso de que la fuente siempre tenga datos pendientes de ser transmitidos, los cuales podrán ser enviados si lo permite la ventana de congestión.

6.2.1 El Modelo

En régimen permanente *(congestion avoidance)*, RCCMP utiliza un control de congestión basado en la implementación TCP Reno. La ventana de transmisión de tamaño W paquetes, se incrementa por 1/W cada vez que recibe un ACK. Si el emisor detecta que ha habido pérdida de paquete, la ventana disminuye en una cantidad, que es función de cómo se ha detectado esa pérdida: por medio de la recepción de tres ACK duplicados o por la expiración del temporizador. En RCCMP también es importante distinguir qué receptor ha perdido el paquete, ya que los procedimientos involucrados para la recuperación son diferentes. Además, la aparición de un receptor con menos recursos que el representante, también hace modificar el caudal, y por lo tanto, habrá que tenerlo en cuenta a la hora de modelar la conducta del protocolo.

Al igual que en los estudios anteriores, para modelar el caudal se va a utilizar el hecho de que la ventana de transmisión del control de congestión muestra cierta periodicidad que se corresponde con el RTT entre la fuente y el representante. El inicio de un periodo coincide con la transmisión de W paquetes, donde W es el tamaño actual de la ventana de transmisión, y acaba con la recepción del primer ACK. Por lo tanto, la duración de este periodo es igual al tiempo de ida y vuelta entre la fuente y el representante. Se asume, igual que en los trabajos anteriores, que este tiempo es independiente del tamaño de la ventana y que es mayor que el tiempo necesario para transmitir W paquetes.

En RCCMP, cada ACK confirma la recepción correcta de dos paquetes de datos, por consiguiente, si al inicio de un periodo se transmiten W paquetes, se recibirán W/2 ACKs. Como cada reconocimiento hace aumentar la ventana de transmisión en 1/W paquetes, se tiene que al comienzo del segundo periodo, el tamaño de la ventana es igual a $W + 1/2$. Por lo que, en fase permanente *(congestion avoidance)*, y en ausencia de pérdidas, la ventana se incrementa linealmente por $1/2$ paquete cada RTT.

Para determinar cual es el comportamiento de RCCMP ante pérdidas, es necesario, describir como éstas se van a producir. Se va a considerar que un paquete se pierde dentro de un periodo de forma independiente a otro paquete perdido en cualquier otro periodo. Esta hipótesis es utilizada en otros estudios como [Bolot96] y [Padhye98]. Por otra parte, se asume que las pérdidas de paquetes están correladas dentro del mismo periodo, por lo que si un paquete se pierde, se asumirá que todos los paquetes desde que se detecta el primer paquete perdido hasta el último en transmitirse en ese periodo, también se perderán. Esta conducta de pérdidas en ráfagas concuerda con el comportamiento de las colas con disciplina FIFO, que muchos nodos intermedios adoptan.

A continuación, se va a modelar el comportamiento de RCCMP si las pérdidas se producen en el representante y en otros receptores; en caso de que éstas se detecten por medio de cualquiera de los dos mecanismos anteriormente citados. Seguidamente, al modelo anterior se le añadirá el caso de que las pérdidas se produzcan en algunos receptores y no en el representante. Finalmente, se considerará el caso de cambio de representante. En este último modelo, solamente se tendrán en cuenta las pérdidas detectadas por recepción de tres ACK duplicados.

6.2.2 Caso 1: pérdidas detectadas en el representante y en otros receptores

En este caso, se asume que todas las pérdidas son detectadas por el representante, independientemente, de si ocurren también en otros receptores. Entre el representante y la fuente se crea un control de congestión igual al que se realiza en la implementación TCP Reno entre la fuente y el destino. Por lo tanto, se tiene la misma situación que para punto a punto, por lo que el modelo descrito en [Padhye98] es válido y adecuado en este caso.

Se transcribe aquí la ecuación obtenida por estos autores que modela el caudal de una sesión TCP, con las restricciones e hipótesis asumidas en el apartado anterior, en función de la probabilidad de pérdida de un paquete, *p*, y del tiempo de ida y vuelta, *RTT*.

$$B(p) \approx \frac{1}{RTT\sqrt{\frac{2bp}{3}} + T_o \min(1, 3\sqrt{\frac{3bp}{8}})\, p(1+32p^2)} \qquad (6.1)$$

En esta expresión *b* es igual a número de paquetes confirmados por un ACK y T_o es el valor del temporizador de retransmisión.

Para RCCMP, bajo la hipótesis de que todas las pérdidas son detectadas en el representante, está aproximación es válida. La ecuación anterior puede ajustarse igualando b a 2, considerando p_r como la probabilidad de que el representante pierda un paquete, y como *RTT* el tiempo de ida y vuelta entre la fuente y el representante.

$$B(p_r) \approx \frac{1}{RTT\sqrt{\frac{4p_r}{3}} + T_o \min\left(1, 3\sqrt{\frac{6p_r}{8}}\right) p_r(1+32p_r^2)} \qquad (6.2)$$

6.2.3 Caso 2: pérdidas detectadas en otros receptores que no son detectadas en el representante

En este apartado se va a añadir al modelo de caudal calculado en la sección anterior, que sólo tenía en cuenta las pérdidas detectadas en el representante, las pérdidas que ocurren en otros receptores y que no son detectadas por el representante. Por lo tanto, al caudal calculado en la ecuación (6.2) hay que añadirle un término debido a la retransmisión de paquetes que han sido solicitados mediante NAKs. La retransmisión de paquetes solicitados mediante NAKs es independiente de la transmisión de paquetes al grupo multipunto, que está controlada por un mecanismo de ventana que gestiona el emisor y el representante.

En el apartado anterior se ha definido p_r como la probabilidad de que el representante pierda un paquete. Ahora, se va a definir $p_{receiver}$ como la probabilidad de pérdida de un paquete en algún receptor que no es el representante.

Cuando un paquete es correctamente recibido por el representante, pero sin embargo, se pierde para un grupo de receptores, provoca en cada uno de estos terminales la generación de un NAK. Solamente aquellos receptores en los que expire el temporizador exponencial antes de recibir el paquete perdido, podrán enviar las respuestas hacia la fuente.

Varios receptores pueden perder el mismo paquete, por lo que pueden llegar a la fuente diferentes NAKs, solicitando la retransmisión del mismo paquete. En este modelo se va a asumir que el proceso de retransmisión se hará una vez por cada paquete perdido, aunque existan varias solicitudes de retransmisión.

RCCMP utiliza el parámetro *nak_factor* para ajustar la duración del intervalo de tiempo en el que no está permitido volver a transmitir el mismo paquete. El objetivo de incluir esta variable en el protocolo es evitar que una sobrecarga de NAKs, se responda con una multitud de paquetes retransmitidos iguales. Por lo tanto, se supondrá que si varios receptores pierden el mismo paquete, y envían NAKs, solicitando su retransmisión, se considerará que todos estos paquetes de control se recibirán en un intervalo de tiempo de longitud máxima igual a *nak_factor* por el tiempo de ida y vuelta del representante, para que el paquete perdido se recupere con una única retransmisión. El ajuste de *nak_factor* de manera apropiada, permitirá que esta hipótesis no suponga ninguna pérdida de generalidad del modelo. Por consiguiente, cada paquete perdido en cualquier receptor, excepto en el representante, implica un paquete retransmitido. Este paquete es inmediatamente reenviado, no siguiendo los procedimientos del mecanismo de ventana deslizante.

Por tanto, para el cálculo del caudal del protocolo RCCMP, teniendo en cuenta que las pérdidas pueden darse en cualquier receptor, se añadirá al modelo obtenido por la ecuación (6.2), el ancho de banda debido a la retransmisión de estos paquetes $B(p_{receiver})$. Considerando que ambos caudales son independientes se tiene que

$$B(p_r, p_{receiver}) = B(p_r) + B(p_{receiver}) \tag{6.3}$$

Para hallar la expresión de $B(p_{receiver})$, es necesario calcular el número medio de retransmisiones de un paquete en función de $p_{receiver}$. Por lo que si la probabilidad de que un paquete sea transmitido k veces es

$$p_{receiver}^{k-1}\left(1-p_{receiver}\right) \tag{6.4}$$

el número medio de transmisiones se calcula como

$$\sum_{i=1}^{\infty} i \cdot p_{receiver}^{i-1} \cdot (1 - p_{receiver}) = \frac{1}{1 - p_{receiver}} \tag{6.5}$$

y el número medio de retransmisiones resulta:

$$N_{rx} = \frac{1}{1 - p_{receiver}} - 1 = \frac{p_{receiver}}{1 - p_{receiver}} \tag{6.6}$$

El caudal retransmitido, función de $p_{receiver}$, es una fracción del tráfico inyectado por la fuente al grupo multipunto, que depende del número medio de retransmisiones por paquete. Así, la expresión para ese caudal queda

$$B(p_{receiver}) = B(p_r) \frac{p_{receiver}}{1 - p_{receiver}} \tag{6.7}$$

donde $B(p_r)$ no es más que el caudal inyectado por la fuente a la red, que está regulado por el control de congestión establecido entre la fuente y el representante. Así, $B(p_{receiver})$ representa el caudal retransmitido debido a pérdidas en otros receptores, que no está controlado por ningún mecanismo de ventana.

Por lo tanto, el caudal total se expresa

$$B(p_r, p_{receiver}) = B(p_r)\left(1 + \frac{p_{receiver}}{1 - p_{receiver}}\right) = B(p_r) \frac{1}{1 - p_{receiver}} \tag{6.8}$$

Si se sustituye la ecuación (6.2) en la ecuación de arriba, se obtiene

$$B(p_r, p_{receiver}) \approx \frac{1}{1 - p_{receiver}} \frac{1}{RTT\sqrt{\frac{2bp_r}{3}} + T_o \min\left(1, 3\sqrt{\frac{3bp_r}{8}}\right) p_r (1 + 32 p_r^2)} \tag{6.9}$$

Esta expresión modela el caudal de una sesión RCCMP cuando se consideran pérdidas en cualquier receptor.

6.2.4 Caso 3: cambio de representante

En este apartado se va a modelar el caudal de una sesión RCCMP, considerando que las pérdidas van a producirse en el representante, que la detección va a ser mediante la recepción de tres ACK duplicados, y que se producen cambios de representante. Un cambio de representante se realiza cuando hay un receptor cuyo caudal estimado, calculado mediante la ecuación de equilibrio simplificada de TCP [Mathis97], es inferior al que se obtiene con el representante actual.

A continuación, se va a estudiar detalladamente, sobre la topología de la figura 5.14 estudiada en el capítulo 5, todos los procesos involucrados en el cambio de representante, para que posteriormente se construya el modelo del caudal. Esta topología presenta una fuente que sirve a dos receptores, RCCMP1 y RCCMP2, los cuales están conectados a enlaces de 1Mbps y 0.8Mbps respectivamente. El receptor RCCMP1 se une al grupo multipunto desde el segundo 0.5s, mientras que el receptor RCCMP2 se asocia en el segundo 30. Hasta este instante el gobierno de la tasa recae en

RCCMP1. Cuando RCCMP2 detecta su primera pérdida envía un paquete NAK que incluye la tasa de pérdidas y el número de secuencia del último paquete de datos recibidos. A partir de estos parámetros, la fuente calcula el caudal estimado, y si éste es suficientemente inferior, provoca un cambio de representante. Si no habrá que esperar a otro NAK que cumpla con la condición. En la figura 5.15 del capítulo 5 se puede observar como se modifica el caudal de la fuente. En los primeros segundos, el representante es el receptor RCCMP1, y el caudal es de 1Mbps, mientras que a partir del segundo 32.6s aproximadamente, el control de la sesión recae en RCCMP2. Se observa como el caudal se reduce hasta los 0.8Mbps, que es la velocidad máxima que este receptor puede aceptar.

Se ha descrito de forma intuitiva cómo es el procedimiento para el cambio de representante. A continuación, a partir del gráfico de la figura 6.1, se va explicar detalladamente el intercambio de paquetes que tiene lugar entre la fuente y los receptores. El gráfico recoge las acciones desde que se recibe el NAK enviado por RCCMP2, hasta el instante en el que el nuevo representante confirma la recepción del primer paquete. Las acciones que forman parte del procedimiento de cambio de representante se puede describir cronológicamente de la siguiente manera:

1. Llegada del NAK desde el receptor RCCMP2. La fuente compara los caudales del representante (RCCMP1) y de RCCMP2, y determina el cambio.

2. Llegada del último paquete de confirmación, ACK(944), validando el último paquete que recibe el receptor RCCMP1 como representante.

3. Envío de los dos últimos paquetes de datos con número de secuencia 971 y 972, indicando que hay un nuevo representante (RCCMP2).

4. El receptor RCCMP1 continúa confirmando los paquetes que va recibiendo, ya que desconoce que no es el representante. Llegan paquetes de confirmación hasta el ACK(970). Como estas confirmaciones provienen de un representante que no es válido, no se aumenta la ventana de transmisión.

5. No hay más paquetes de datos que confirmar. El temporizador de retransmisión expira.

6. Se reenvía todos los paquetes a partir del último debidamente confirmado que es el 945. La ventana de transmisión se iguala a 1 y el control de congestión entra en la fase transitoria (*slow-start*).

7. Se recibe el primer ACK(946) desde el nuevo representante RCCMP2.

Aunque se ha visto en el capítulo 5, que no todos los cambios de representante llevan asociados la expiración del temporizador de retransmisión, éste es el esquema que se va a seguir para llevar a cabo el modelo del caudal. En principio, esta suposición puede resultar en un análisis más pesimista, ya que en todos los cambios de representante se asume que el tamaño de la ventana se reduce drásticamente. El estudio que se presenta a continuación se basa en el estudio de [Padhye98].

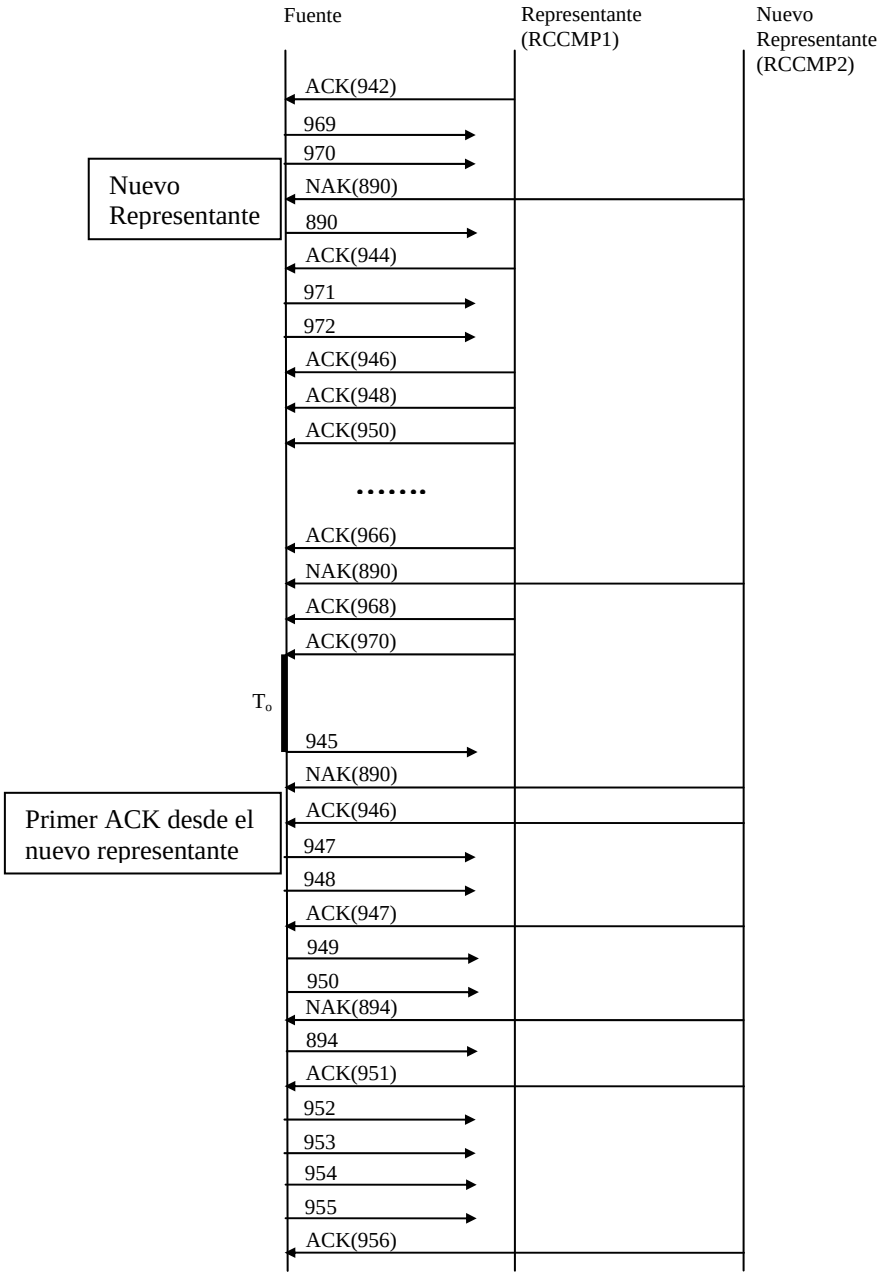

Figura 6.1 Intercambio de paquetes entre la fuente y los receptores en el proceso de cambio de representante.

En la figura 6.2 se observa la evolución de la ventana de transmisión entre dos cambios de representante. Z_i^{TD} representa el tiempo desde que toma el control el nuevo representante hasta que se recibe el último ACK, confirmando el último paquete de datos, que precede a la expiración del temporizador de retransmisión. La duración de la fase controlada por los temporizadores de retransmisión es de Z_i^{TO}. Cuando expira el

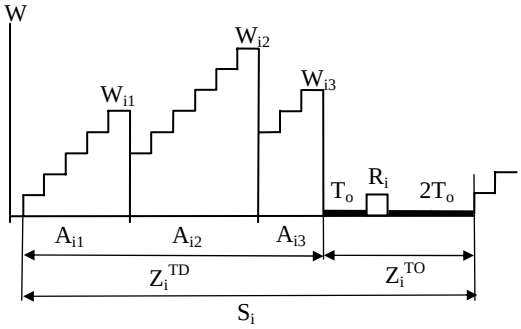

Figura 6.2 Evolución del tamaño de la ventana cuando la detección de pérdidas se produce por 3 ACKs duplicados y cambios de representante.

primer temporizador, que tiene un valor de T_o, se iguala la ventana de transmisión a 1, se retransmite el primer paquete no confirmado, y se actualiza el valor del nuevo temporizador a $2*T_o$. Este proceso se realiza hasta que se recibe el ACK o hasta que el valor del temporizador alcance un valor determinado. Para cada periodo i se define S_i como la duración total de estas dos fases.

$$S_i = Z_i^{TD} + Z_i^{TO} \tag{6.10}$$

Por otro lado, se define M_i como el número de paquetes transmitidos durante S_i. Si se considera $\{(S_i, M_i)\}_i$ como una secuencia de variables aleatorias independientes e idénticamente distribuidas, se puede escribir que el caudal B se puede calcular como

$$B(p_r) = \frac{E(M)}{E(S)} \tag{6.11}$$

A partir de la figura 6.2 se van a definir variables que serán útiles en el análisis. Se define A_{ij} como el intervalo entre dos pérdidas de paquete consecutivas, que son detectadas por el mecanismo de retransmisión rápida. Como se observa, en la gráfica coincide con el decremento a la mitad del tamaño del valor de la ventana. Al final del intervalo A_{ij}, el tamaño de la ventana de transmisión es W_{ij}. X_{ij} representa el número de periodos o RTTs de un A_{ij}. Por otra parte, se define como n_{ij} al número de A_{ij} que ocurren en Z_i^{TD}. Al número de paquetes que se envían en el intervalo A_{ij} se le denomina Y_{ij}. Mientras que R_i es el número de paquetes enviados durante Z_i^{TO}.

Se puede calcular fácilmente M_i, que expresa el número de paquetes totales enviados durante S_i, en función del número de paquetes que se envían en cada intervalo A_{ij} como

$$M_i = \sum_{j=1}^{n_i} Y_{ij} + R_i \tag{6.12}$$

Por otro lado, S_i se puede descomponer como

$$S_i = \sum_{j=1}^{n_i} A_{ij} + Z_i^{TO} \tag{6.13}$$

Si se calculan las esperanzas de estas dos expresiones se tiene,

$$E(M) = E(n)E(Y) + E(R) \tag{6.14}$$

$$E(S) = E(n)E(A) + E(Z^{TO}) \tag{6.15}$$

donde E(n) es el número de veces que se repiten los intervalos A_{ij} en el intervalo Z^{TD}.

Por tanto, el cálculo del caudal se reduce a calcular las esperanzas que aparecen en las expresiones anteriores. Del estudio de [Padhye98] se pueden extraer directamente los valores de E(Y) y E(A), ya que éstos modelan el número de paquetes medio y la duración media del intervalo entre pérdidas de paquetes consecutivas del control de congestión RCCMP, cuyo funcionamiento coincide con el de TCP. E(R) y $E(Z^{TO})$ coinciden también con las expresiones de [Padhye98], puesto que el procedimiento de RCCMP es el mismo que el de TCP, cuando expiran los temporizadores de retransmisión. Por consiguiente, la particularidad de nuestro estudio radica en el cálculo de E(n).

Como Padhye observa en su trabajo, durante Z_i^{TD}, existen n_i-1 intervalos A_{ij} que terminan con la detección de una pérdida por medio de tres ACKs duplicados, mientras que el último intervalo n_i, finaliza con un cambio de representante. Así, E(n) representa el número de intervalos A_{ij} que existen entre dos cambios de representante. Para facilitar el cálculo de esta esperanza, se define $Q = 1/E(n)$ como la probabilidad de que un intervalo A_{ij} termine con un cambio de representante.

En la figura 6.3 se ilustra las acciones que suceden antes de un cambio de representante. Éstas se pueden resumir en los siguientes puntos:

1. Se envían W paquetes en el penúltimo periodo.
2. En el último periodo se recibe $(W-2)/2$ ACKs que confirman los W-2 paquetes enviados en el penúltimo periodo. No llega el ACK que confirma los paquetes W-1 ni W.
3. Aunque se han recibido $(W-2)/2$ ACKs, no se avanza la ventana de transmisión, ya que estos paquetes de confirmación provienen de un representante que no es válido.
4. Tiene lugar la expiración del temporizador.

Estas acciones marcan el cambio de representante, por lo tanto marcan el final del intervalo Z_i^{TD}.

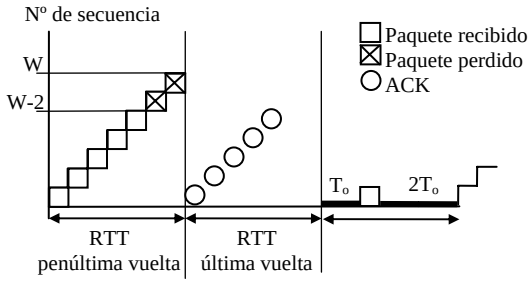

Figura 6.3 Paquetes involucrados antes de un cambio de representante.

Se puede definir A(W,W-2) como la probabilidad de que solamente los primeros W-2 paquetes sean confirmados en una ventana de W paquetes, sabiendo que habrá una

o más pérdidas en esa vuelta. Se considerará que las pérdidas dentro del mismo periodo están correladas, por lo que si se pierde el paquete W-1, también se perderá el W.

$$A(W,W-2)=\frac{(1-p_r)^{W-2}p_r}{1-(1-p_r)^W} \tag{6.16}$$

La probabilidad de que haya en el grupo multipunto un receptor que sea el representante y que además este representante no sea válido se puede calcular como

$$\frac{1}{N}\frac{N-1}{N} \tag{6.17}$$

donde N representa el número de receptores que forman el grupo multipunto y $1/N$ es la probabilidad de que el receptor que envía los ACK sea un representante válido.

Si Q(W) se define como la probabilidad de que en una ventana de tamaño W haya un cambio de representante, ésta viene dada por la siguiente expresión

$$Q(W)=\frac{(1-p_r)^{W-2}p_r}{1-(1-p_r)^W}\frac{N-1}{N} \tag{6.18}$$

Donde la probabilidad de que haya un cambio de representante, la determina el que habiendo enviado W paquetes se reciban $(W-2)/2$ paquetes de confirmación desde un receptor no válido. Se considera que estos dos sucesos son independientes.

Si se supone que la probabilidad de pérdida es pequeña, se puede resolver el límite utilizando la regla de L'Hopital y se tiene

$$\lim_{p\to 0}Q(W)=\frac{N-1}{N^2}\frac{1}{W} \tag{6.19}$$

Por lo tanto, se puede aproximar Q(W) a

$$Q(W)=\frac{N-1}{N^2}\frac{1}{W} \tag{6.20}$$

Si Q es la probabilidad de que haya un cambio de representante, entonces

$$Q=\sum_{i=1}^{\infty}Q(W)P(W=w)=E(Q) \tag{6.21}$$

Sin embargo, se va utilizar la aproximación de que $Q\approx Q(E(W))$ donde E(W) ya se calculó en el estudio de [Padhye98].

$$Q=\frac{N-1}{N^2}\sqrt{\frac{3bp_r}{8}} \tag{6.22}$$

Una vez obtenido Q, se tienen todos los términos para calcular el caudal

$$B(p_r)=\frac{E(Y)+QE(R)}{E(A)+QE(Z^{TO})} \tag{6.23}$$

Si se sustituyen los valores de E(Y), E(R), E(A) y E(Z^{TO}) se obtiene

$$B(p_r) = \frac{\frac{1-p_r}{p_r} + E(W) + Q(E(W))\frac{1}{1-p_r}}{RTT(E(X)+1) + Q(E(W))T_o\frac{f(p_r)}{1-p_r}} \tag{6.24}$$

Finalmente esta expresión puede aproximarse por

$$B(p_r) \approx \frac{1}{RTT\sqrt{\frac{4p_r}{3}} + T_o p_r (1+32p_r^2)\frac{N-1}{N^2}\sqrt{\frac{6p_r}{8}}} \tag{6.25}$$

Notar que en esta ecuación sí se observa una dependencia del caudal con el número de receptores.

Este resultado puede extenderse al caso de que las pérdidas se den en cualquier receptor, es decir, que puedan darse en el representante y otros receptores o simplemente en otros receptores. Para ello, basta aplicar el razonamiento desarrollado en el apartado 6.2.3, donde la ecuación (6.7) representa el caudal debido a retransmisiones por pérdidas en receptores que no son el representante. Si se aplica esa ecuación al caudal que se ha calculado en esta sección, teniendo en cuenta los cambios de representante (6.25), se obtiene:

$$B(p_r, p_{receiver}) \approx \frac{1}{1-p_{receiver}} \frac{1}{RTT\sqrt{\frac{4p_r}{3}} + T_o p_r (1+32p_r^2)\frac{N-1}{N^2}\sqrt{\frac{6p_r}{8}}} \tag{6.26}$$

Esta expresión muestra la tasa de paquetes enviados totales teniendo en cuenta pérdidas en cualquier receptor y cambios de representante.

6.3 RESULTADOS

A continuación se van a presentar los resultados que validarán a través de simulaciones la ecuación (6.26), la cual estima el caudal que la fuente RCCMP introduce en la red en función de la probabilidad de pérdida y del tiempo de ida y vuelta entre la fuente y el receptor más lento.

Para las simulaciones se ha utilizado una topología arbolada de 30 receptores, figura 6.4, en la que todos los enlaces tienen la misma capacidad y el mismo retardo igual a 0.5Mbps y 1ms respectivamente. Se ha considerado una fuente que emite constantemente a una velocidad mayor que la capacidad de los enlaces, para que se cumpla la hipótesis de que la fuente siempre tenga datos para transmitir. De esta manera, el caudal solamente se verá restringido por el control de congestión.

Se han realizado series de 10 simulaciones de 150 segundos, donde se ha modificado el valor de la probabilidad de pérdida en el canal y el tamaño de las colas. El

valor de RTT y del temporizador de retransmisión, que se utilizará en el modelo analítico, se calcula como el promedio entre las diez realizaciones. La probabilidad de pérdida en el representante se obtiene como el cociente de las indicaciones de pérdidas divido entre el número de paquetes totales enviados. Estas indicaciones se producen mediante tres ACKs duplicados o la expiración del temporizador. La probabilidad de pérdida en otros receptores que no son el representante se calcula como el número de paquetes retransmitidos a partir de las solicitudes explicitas dividido por el número total de paquetes enviados.

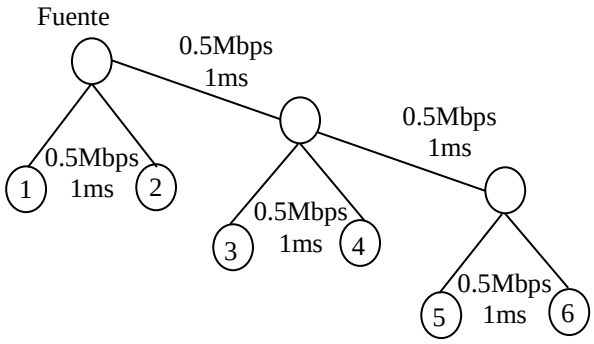

Figura 6.4 Topología para el estudio del modelo de caudal.

Para una mejor representación, los resultados de las diferentes simulaciones se han agrupado en función de su RTT. Así, los resultados analíticos para probabilidades de pérdida en el canal igual a 0, 10^{-6}y 10^{-5} se han promediado y se presentan bajo una sola curva. Las gráficas 6.5 hasta 6.8 comparan el número de paquetes estimados mediante el modelo analítico y los valores obtenidos mediante simulación para distintas probabilidades de error y tamaño de cola.

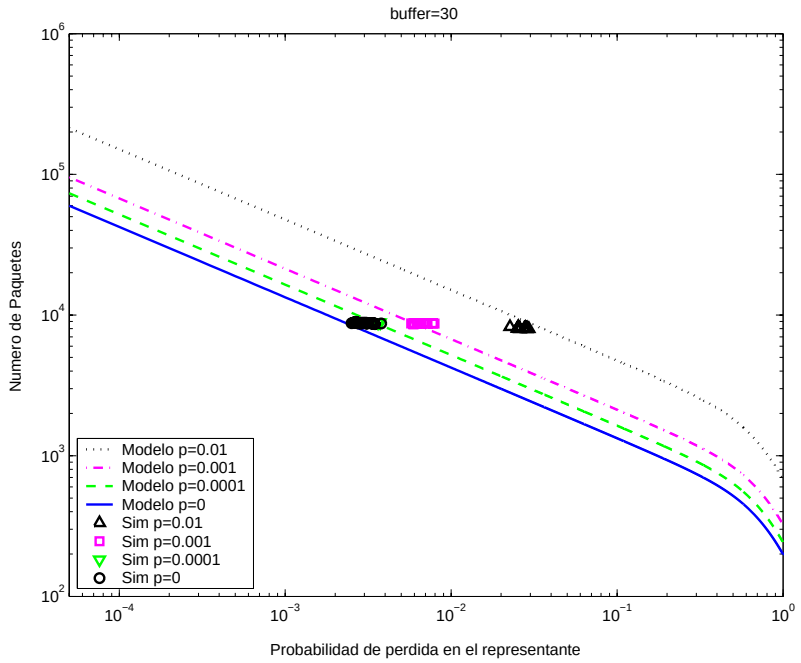

Figura 6.5 Comparación entre el caudal analítico y simulado para un tamaño de cola de 30 paquetes.

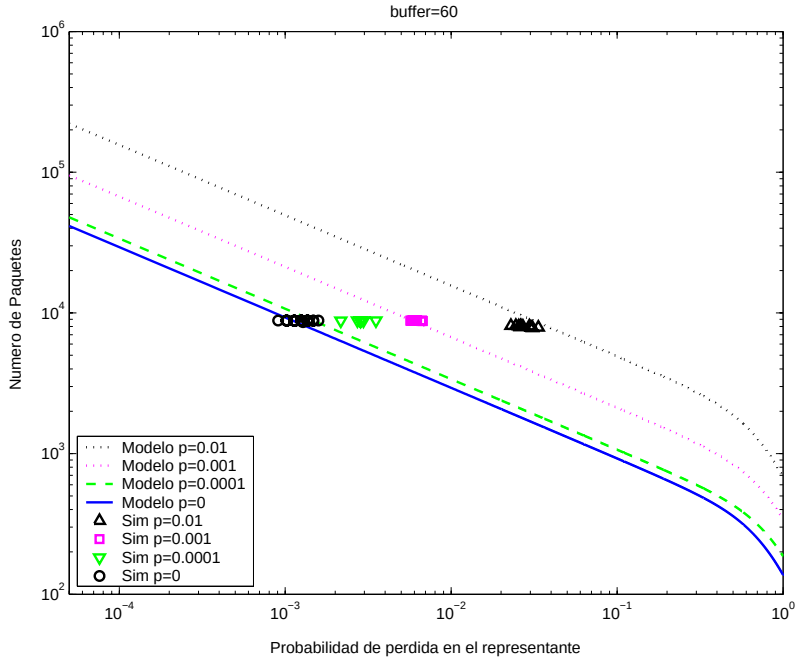

Figura 6.6 Comparación entre el caudal analítico y simulado para un tamaño de cola de 60 paquetes.

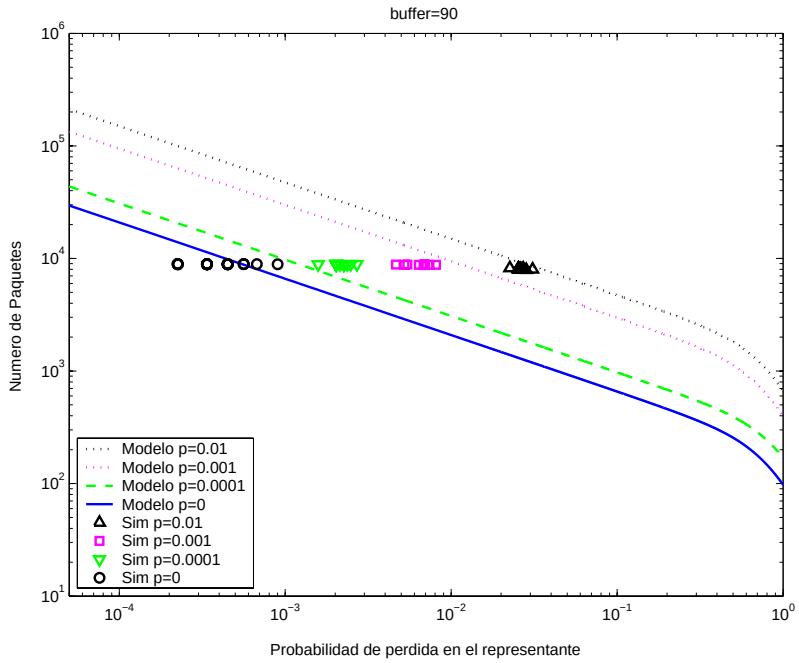

Figura 6.7 Comparación entre el caudal analítico y simulado para un tamaño de cola de 90 paquetes.

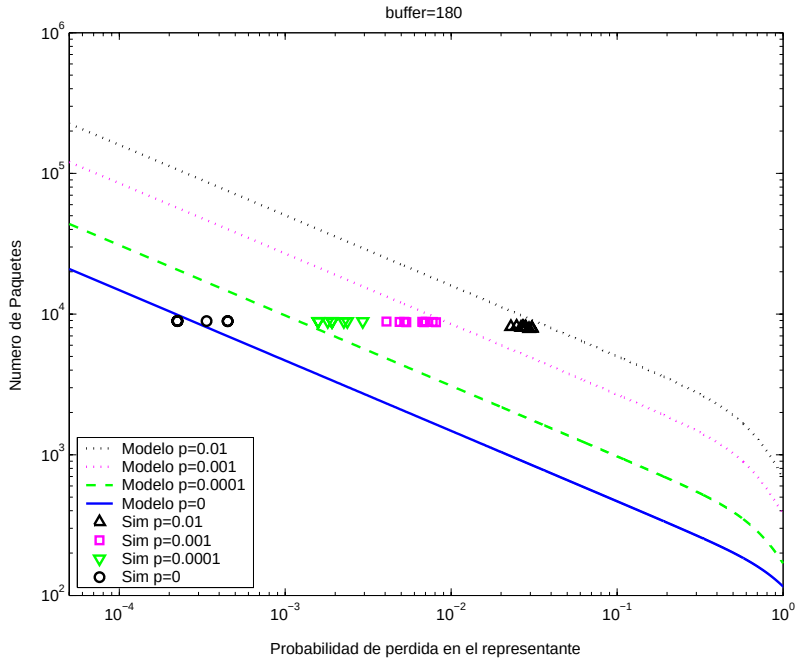

Figura 6.8 Comparación entre el caudal analítico y simulado para un tamaño de cola de 180 paquetes.

Para probabilidades de pérdida pequeña, se cumple que a mayor tamaño de la cola se obtiene un tiempo de ida y vuelta entre la fuente y el representante mayor, y una probabilidad de pérdida en otros receptores menor. También se cumple que para tamaño de colas pequeños las curvas aparecen más agrupadas ya que el valor del RTT es menos disperso. Por otro lado, el introducir mayores pérdidas en el canal redunda en mayores pérdidas en el representante y en los otros receptores, y por lo tanto en más cambios de representante. En la tabla 7.1 aparecen el número de cambios de representante y el RTT promedio en función del tamaño de la cola y de la probabilidad de pérdida en el canal.

	Cola 30		Cola 60		Cola 90		Cola 180	
Probabilidad	RTT	Cambios	RTT	Cambios	RTT	Cambios	RTT	Cambios
10^{-2}	0,37	294	0,37	286	0,11	296	0,10	292
10^{-3}	0,20	99	0,20	102	0,14	98	0,16	96
10^{-4}	0,25	34	0,39	28	0,42	24	0,42	21
10^{-5}	0,30	26	0,43	12	0,67	6	0,88	3
10^{-6}	0,30	27	0,44	13	0,55	4	1,07	0
0	0,32	24	0,46	12	0,65	4	1,07	0

Tabla 6.1 Número de cambios de representante y RTT promedios.

Para evaluar la bondad del modelo, se calcula el error relativo promediado entre todas las realizaciones para la misma longitud de cola en función de las pérdidas de canal. El error se calcula mediante la siguiente expresión:

$$\frac{\sum_{simulaciones} \frac{\left|Caudal_{analitico} - Caudal_{simulado}\right|}{Caudal_{simulado}}}{numero\,de\,simulaciones} \tag{6.27}$$

En la gráfica 6.9 se representa el valor de este error. Un menor error nos indica una mejor adecuación del modelo a las simulaciones. Como se observa en la gráfica el modelo se ajusta mejor para tamaños de cola de 30 y 60 paquetes.

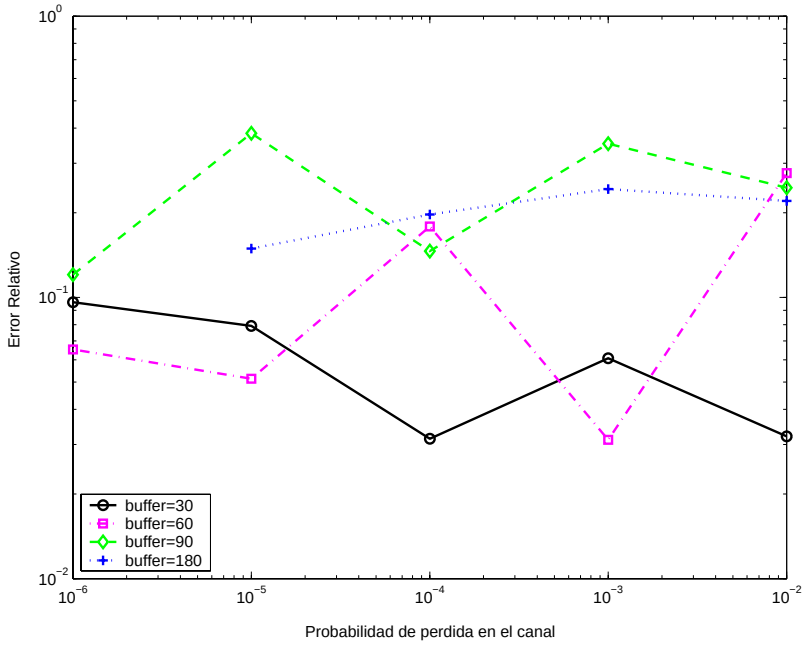

Figura 6.9 Error relativo.

Otra manera de comprobar la validez del modelo es obtener el número de paquetes enviados en función del número de receptores para distintas tasas de pérdidas de canal. En este caso, se va a utilizar la topología en árbol de la figura 5.40, donde todos los enlaces son iguales, y se caracterizan por un ancho de banda de 0.5Mbps y un retardo de propagación de 1ms. Las simulaciones tienen una duración de 400s.

La gráficas 6.10, 6.11 y 6.12 muestran el número de paquetes en un escenario en el que las pérdidas de canal son del 0%, 0.001% y 1% respectivamente. Aunque el ajuste es bastante bueno, pueden observarse que el modelo tiende a sobreestimar el número de paquetes enviados. Para probabilidades de pérdida de canal pequeñas, 0 y 0.001%, la aportación de los paquetes retransmitidos por solicitud de NAKs a los paquetes enviados totales es también pequeña. Sin embargo, para pérdidas de canal del 1%, el modelo sobredimensiona el número de paquetes retransmitidos vía NAKs. Este efecto puede verse en la gráfica 6.12, en la que aparecen tres curvas que representan los resultados obtenidos mediante simulación (línea azul), ecuación (6.26) (línea negra) y ecuación (6.25) (línea verde).

La figura 6.13 muestra el error relativo. El peor ajuste es el obtenido cuando se consideran pérdidas de canal del 1%.

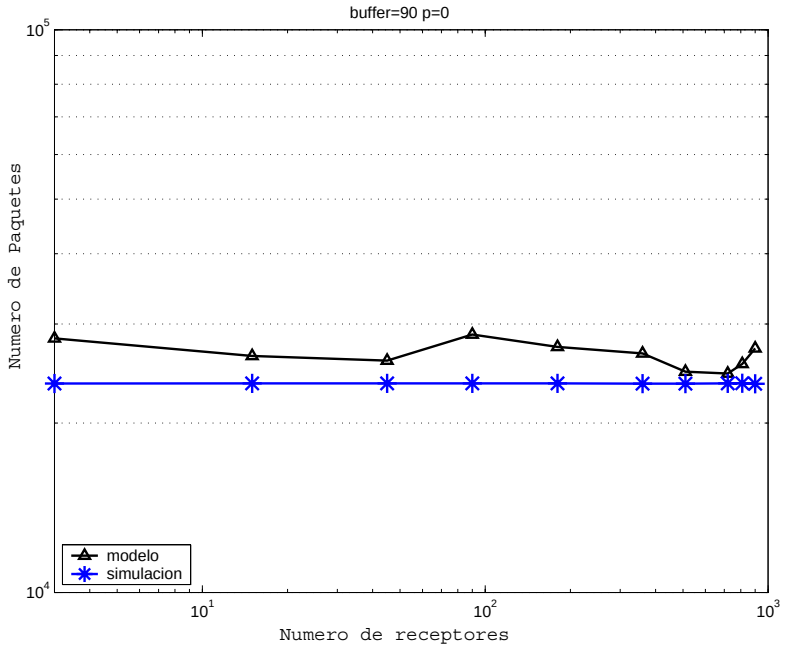

Figura 6.10 Comparación del número de paquetes enviados mediante simulación y analíticamente sin pérdidas de canal.

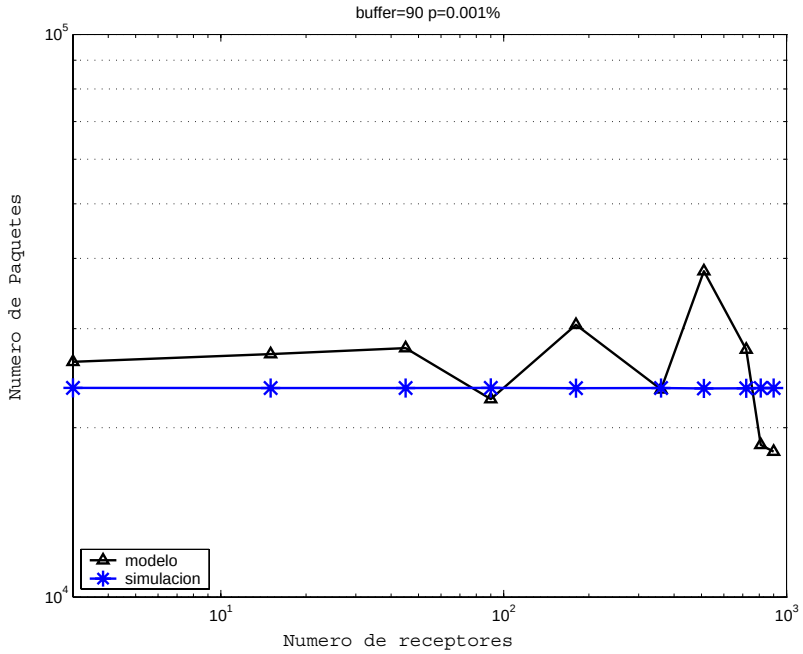

Figura 6.11 Comparación del número de paquetes enviados mediante simulación y analíticamente para pérdidas de canal de 0.001%.

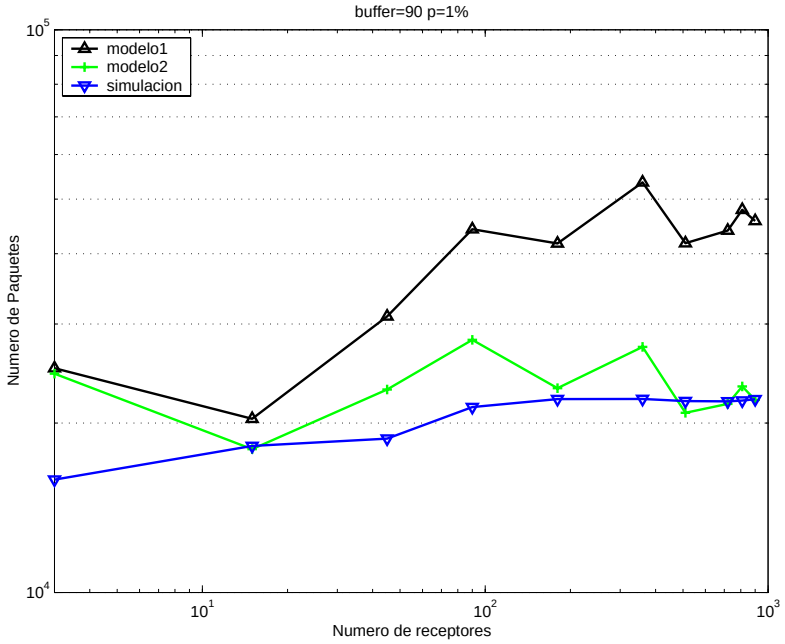

Figura 6.12 Comparación del número de paquetes enviados mediante simulación y analíticamente para pérdidas de canal del 1%.

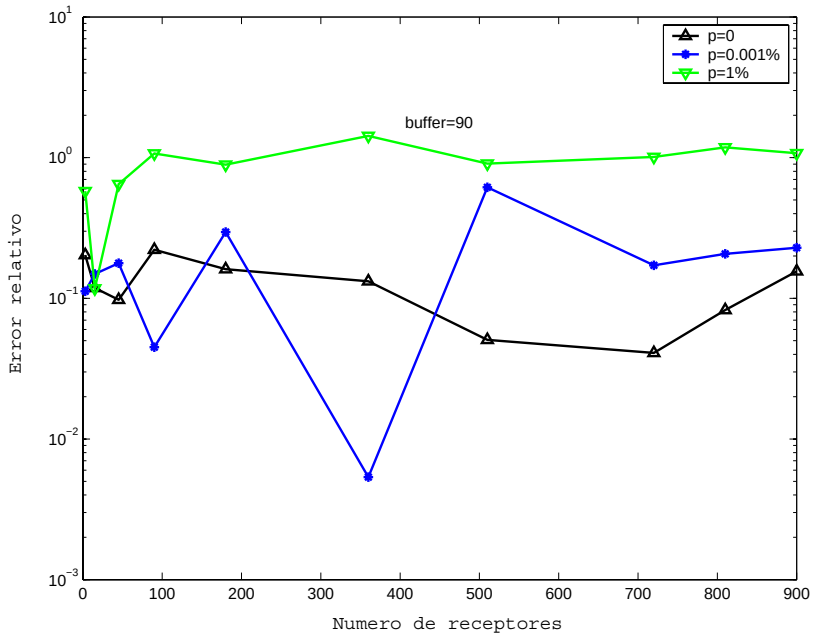

Figura 6.13 Error relativo.

6.4 CONCLUSIONES

En este capítulo se ha obtenido una expresión matemática que modela el caudal del protocolo de transporte multipunto fiable, en función de la tasa de pérdidas y del tiempo de ida y vuelta entre la fuente y el representante. El trabajo realizado por J. Padhye ha sido un importante punto de partida, ya que el control de congestión de RCCMP es totalmente asimilable al que desarrolla el protocolo TCP. Como en una conexión punto a punto, entre la fuente y el representante se crea un lazo cerrado que controla la tasa de la fuente en función de la realimentación que le llega desde este receptor.

Sin embargo, existen diferencias entre los entornos punto a punto y multipunto. Las dos principales causas del por qué se modifica el caudal en un entorno multipunto, y que difieren respecto al escenario punto a punto, son las siguientes. La primera es que se pueden detectar pérdidas en cualquier receptor, generándose paquetes retransmitidos fuera del control de flujo, que han de ser tenidos en cuenta en el modelo del caudal. Segunda, la aparición de un receptor de menos recursos en el grupo multipunto provoca un cambio de representante, y también una modificación en la tasa.

A partir del estudio de diferentes topologías se han obtenido resultados mediante simulación que validan el modelo matemático para un alto rango de valores.

CAPÍTULO 7
MODELADO DE LA CARGA DEL PROTOCOLO RCCMP

7.1 INTRODUCCIÓN

Es importante conocer profundamente el comportamiento, las ventajas, limitaciones y debilidades, de cada una de las propuestas de protocolos multipunto fiables, que han sido publicadas durante estos últimos años, para seleccionar adecuadamente para qué aplicaciones y en qué entornos deben ser utilizadas. En el [RFC2357], se describen los puntos que deben detallarse cuidadosamente en cualquier propuesta: la escalabilidad, el control de congestión, la recuperación de errores y la robustez. Este capítulo se centra en determinar la escalabilidad del protocolo RCCMP, a partir del análisis del ancho de banda que consume. Para ello, se calcula detalladamente el coste de cada uno de los procesos que este protocolo incluye.

Existen diferentes trabajos que estudian el coste de distribuir datos a nivel de red y a nivel de transporte. Entre los primeros se encuentran [Chuang98], [Phillips99], [Billhartz97] y [Maciá04], que calculan y comparan eficiencias y costes para diferentes protocolos de enrutamiento multipunto. En [Chuang98] y [Maciá04] se incluye un análisis comparativo entre las distribuciones punto a punto y multipunto, para determinar en qué escenarios es más eficiente utilizar una u otra forma de distribuir la información. Si bien, en la transmisión multipunto un solo paquete es enviado a todo los miembros del grupo multipunto, y la duplicación del paquete solo se realiza cuando los caminos a los distintos receptores divergen, incluye otros costes no presentes en las comunicaciones punto a punto, como la gestión y el mantenimiento del árbol de distribución. La importancia del estudio de [Chuang98] radica en que es el primero que de forma general cuantifica las ventajas de la distribución multipunto.

A nivel de transporte, los trabajos se centran en el estudio del coste de los procesos de control de errores, que aseguran la transmisión fiable, en los protocolos multipunto fiable. En el estudio [Pingali94] se determina el caudal máximo para el emisor y los receptores para protocolos genéricos basados en confirmaciones implícitas y explicitas. Los primeros protocolos consiguen una transmisión fiable a partir del uso de los ACKs. Cada receptor debe enviar un paquete de confirmación por cada paquete correctamente recibido. La fuente utiliza temporizadores para detectar los paquetes perdidos. Estos protocolos sufren de sobrecarga de ACKs a medida que el número de receptores

aumenta. En los protocolos que utilizan respuesta explícita, el receptor al detectar un paquete perdido, envía un NAK al emisor para solicitar la retransmisión. Aunque a primera vista, pueda parecer que los segundos protocolos consiguen una mejor eficiencia, también pueden sufrir de sobrecarga, si por ejemplo, se produce una pérdida de paquete en uno de los enlaces comunes del árbol multipunto. Pingali et al. calculan el caudal máximo de estas dos clases de protocolos desde el enfoque del tiempo de procesado requerido por un paquete en los nodos emisor y receptor.

El trabajo de Levine et al. [Levine96] clasifica los protocolos multipunto fiable en cuatro categorías en función de sus procesos de control de errores, y calcula para cada uno de ellos el caudal máximo limitado, al igual que en el estudio anterior, por la capacidad de proceso de los nodos.

En [Adamson02] se cuantifica la cantidad de información de realimentación que generan los protocolos multipunto, que utilizan respuesta explicita, y que aseguran la escalabilidad del protocolo en base a un esquema de temporizadores. Adamson et al. obtienen diferentes resultados si los NAKs son transmitidos de forma punto a punto o multipunto, y dependiendo de la función de distribución de los temporizadores.

Otros trabajos estudian en menor o mayor medida la escalabilidad de un protocolo de transporte multipunto fiable. La escalabilidad de SRM [Floyd95] ha sido profundamente estudiada en [Raman98]. En este trabajo, se cuantifica el número de NAKs en función del tamaño del grupo multipunto para diferentes topologías y parámetros del protocolo.

En este capítulo se va plantear un método de análisis para cuantificar el ancho de banda de un protocolo de transporte multipunto fiable. Este estudio no se limitará al estudio del número de paquetes de realimentación que los receptores envían a la fuente, sino que tendrá en cuenta los procesos que involucran un intercambio de paquetes, como la selección del nuevo representante, el control de congestión o el control de errores. Se estudiará la utilización del enlace en función del número de receptores para distintas probabilidades de error. Para simplificar el modelo analítico se va utilizar como caso de estudio el protocolo RCCMP. Sin embargo el método desarrollado pretende ser un punto de referencia para calcular el ancho de banda consumido por cualquier protocolo de transporte multipunto. Además en este estudio, también se comparará el ancho de banda de un protocolo de transporte multipunto con uno punto a punto como TCP.

El capítulo se organiza de la siguiente manera. Primero, se cuantifica el ancho de banda consumido por un protocolo de transporte punto a punto como TCP en un escenario sin y con pérdidas. Segundo, se amplia el modelo para el protocolo multipunto fiable RCCMP. Tercero, se valida el análisis propuesto a partir del estudio del protocolo realizado en el capítulo 5 mediante simulaciones. Por último, se compara al ancho de banda consumido para un protocolo punto a punto y multipunto.

7.2 ANÁLISIS DE LA CARGA DEL PROTOCOLO

En esta sección se va a desarrollar un método para el análisis de la escalabilidad de un protocolo de transporte multipunto fiable. Para ello, se comienza estudiando el protocolo de transporte punto a punto TCP. En este entorno, los paquetes que se han de contabilizar son los paquetes de datos, los retransmitidos y los ACKs. Como el

protocolo RCCMP, controla la tasa de la sesión multipunto a partir de un solo receptor, este modelo simplificado será un buen punto de partida para el análisis. Además, permitirá determinar cuando es más rentable utilizar un protocolo punto a punto o multipunto.

Para cuantificar el ancho de banda consumido por el protocolo RCCMP es importante distinguir los distintos procesos que involucra y analizarlos por separado. Al igual que en el caso punto a punto, hay que contabilizar los paquetes de datos, los retransmitidos y los ACKs. Sin embargo, hay que añadir las solicitudes de retransmisión de paquetes, NAKs, y los paquetes de control que estiman el número de receptores necesario para ajustar los temporizadores exponenciales. También habrá que tener en cuenta el número de paquetes de control que consume la elección de representante.

Para el siguiente estudio, se supondrá que el tamaño del grupo multipunto es de N receptores y que la fuente inyecta tráfico a una tasa de E(α) paquetes útiles/s. El coste se medirá en paquetes por salto/s. Para que los resultados sean fácilmente comparables en el entorno punto a punto y multipunto, en el caso punto a punto se analiza el coste de N conexiones.

7.2.1 Caso 1: conexión punto a punto sin pérdidas

Se ha escogido a TCP para analizar y cuantificar el ancho de banda, porque es el protocolo de transporte fiable de referencia para punto a punto. Sin embargo, para comparar la eficiencia entre un protocolo multipunto como RCCMP y un protocolo punto a punto como TCP, es necesario calcular el ancho de banda que consumen N sesiones TCP con una sesión RCCMP que dé servicio a N receptores. Para comenzar, se va a considerar un escenario sin pérdidas.

En una sesión TCP, sobre enlaces con probabilidad de error por bit igual a cero y tamaños de cola suficientemente grandes, los únicos paquetes que intervienen son los de datos y los ACKs. En este primer caso, no es necesario contabilizar paquetes retransmitidos o paquetes de control para coordinar a múltiples receptores. Así, el coste que suponen N sesiones TCP se reduce a multiplicar por N el coste de una sola sesión. El coste de N sesiones TCP sobre enlaces sin pérdidas se puede dividir en

$$C_{unicast\_nonlosses} = C_{data} + C_{ACK} \tag{7.1}$$

donde C_{data} y C_{ACK} hacen referencia al coste que introducen los diferentes paquetes.

Se define $\overline{L}_u$ como el número de saltos medio entre la fuente y cualquier receptor. En [Chuang98], este parámetro describe el número medio de saltos entre dos nodos cualquiera. Esta constante es necesaria para el cálculo del coste en unidades de paquetes por salto/s.

El coste de que N conexiones punto a punto TCP transmitan paquetes de datos es

$$C_{data} = E(\alpha)\overline{L}_u N \tag{7.2}$$

donde *E(α)* representa la tasa útil (en paquetes/s) a la que la fuente envía los datos, *N* es el número de sesiones punto a punto y $\overline{L}_u$ es el número de saltos medio entre la fuente y el receptor.

El receptor TCP devuelve un ACK por un paquete correctamente recibido. Sin embargo, existen ciertas implementaciones TCP que envían un ACK por cada dos paquetes consecutivamente recibidos (*delayed ACK*). Por lo tanto, para una mayor generalidad se define *b* como el número de paquetes que reconoce un ACK (b=1 en el primer caso y b=2 en el segundo). El coste que introducen los paquetes ACK, es decir, el coste de la fiabilidad puede ser calculado como

$$C_{data} = \frac{E(\alpha)}{b}\overline{L_u}N \tag{7.3}$$

Por lo tanto el ancho de banda que consumen N conexiones TCP punto a punto en un escenario sin pérdidas se puede expresar como

$$C_{unicast\_nonlosses} = E(\alpha)\overline{L_u}N(1+1/b) \tag{7.4}$$

A continuación se va extender el modelo para incluir el comportamiento de TCP ante pérdidas para el cálculo del ancho de banda que consumen las N conexiones punto a punto.

7.2.2 Caso 2: conexión punto a punto con pérdidas

Para calcular el coste que añaden las pérdidas, se han de tener en cuenta cómo afectan éstas al protocolo punto a punto fiable. En principio, cualquier paquete perdido debe ser recuperado, por lo que es necesario que cuando el receptor detecte la pérdida de un paquete, informe a la fuente y ésta lo retransmita.

Se supondrá, en una primera aproximación, que cuando un receptor detecta un salto de secuencia, envía un NAK a la fuente para solicitar la retransmisión del paquete perdido. Se asumirá que se emplea un mecanismo de ventana deslizante selectiva, de manera que el paquete perdido es el único que se reenviará, y que los ACKs y los NAKs no se perderán.

Se define *p* como la probabilidad de que un paquete se pierda. Por lo tanto, la probabilidad de que un paquete sea transmitido exactamente *k* veces se expresa mediante la siguiente ecuación

$$N_{tx} = p^{k-1}\left(1 - p\right) \tag{7.5}$$

La esperanza del número de transmisiones de un paquete se calcula mediante

$$E[N_{tx}] = \sum_{i=1}^{\infty} i\, p^{i-1}\left(1-p\right) = \frac{1}{1-p} \tag{7.6}$$

Por lo tanto, a partir de (7.6) resulta fácil calcular la esperanza del número de retransmisiones, simplemente, restando una unidad a la expresión anteriormente obtenida.

$$E[N_{rtx}] = \frac{p}{1-p} \tag{7.7}$$

Para calcular el coste que suponen los paquetes retransmitidos, se multiplica la tasa de paquetes retransmitidos por el número de sesiones, y por la longitud media en saltos de las conexiones.

$$C_{retransmittedpacket} = E(\alpha)\frac{p}{1-p}\overline{L_u}N \tag{7.8}$$

Cuando un paquete se retransmite es porque un NAK ha llegado al emisor. Por consiguiente, cada NAK se corresponde con un paquete retransmitido. Así que el coste, que supone el envío de estos paquetes de control, coincide con el coste de los paquetes retransmitidos.

Consecuentemente, el consumo de ancho de banda para N sesiones punto a punto en un escenario con pérdidas, se puede expresar como:

$$\begin{aligned} C_{unicast\_losses} &= E(\alpha)\overline{L_u}N(1+\frac{1}{b}) + 2E(\alpha)\frac{p}{1-p}\overline{L_u}N \\ &= E(\alpha)\overline{L_u}N\frac{b(1+p)+1-p}{b(1-p)} \end{aligned} \tag{7.9}$$

en el que se diferencia un primer término que engloba el coste sin considerar pérdidas, y un segundo, en el que se contabilizan los NAKs y los paquetes retransmitidos.Esta ecuación representa el consumo en paquetes por salto/s del ancho de banda de N conexiones de un protocolo punto a punto fiable que utiliza respuesta explícita.

Para calcular el coste de N sesiones TCP en presencia de pérdidas, es necesario considerar el comportamiento de este protocolo ante ellas. El emisor TCP detecta que ha habido pérdida de un paquete, cuando recibe más de tres réplicas consecutivas de un ACK, o cuando ha expirado el temporizador de retransmisión. En este modelo no se consideran peticiones de solicitud de retransmisión explícitas, por lo tanto, la detección de pérdidas se lleva a cabo sin añadir ningún coste adicional. Así, sólo es necesario sumar al coste del modelo sin pérdidas, el coste de la retransmisión de paquetes, para obtener la utilización del enlace de N sesiones TCP.

$$\begin{aligned} C_{unicast\_losses} &= E(\alpha)\overline{L_u}N(1+\frac{1}{b}) + E(\alpha)\frac{p}{1-p}\overline{L_u}N \\ &= E(\alpha)\overline{L_u}N\frac{b+1-p}{b(1-p)} \end{aligned} \tag{7.10}$$

7.2.3 Caso 3: conexión multipunto sin pérdidas

A diferencia de TCP, el protocolo RCCMP, necesita de más tipos de paquetes y más procedimientos, para alcanzar los objetivos de fiabilidad y escalabilidad, en un entorno que podría llegar a centenas de miles de receptores. Para calcular el consumo de ancho de banda de una sesión multipunto, se deben conocer a priori los mecanismos y los paquetes que intervienen. Por ello, primero, se van a resumir brevemente los procesos implicados en el protocolo, cuando no se consideran pérdidas.

Los procedimientos, englobados en RCCMP, son el control de congestión, que incluye la selección del representante y los mecanismos para asegurar la escalabilidad. El primero utiliza un mecanismo de ventana deslizante que se asemeja al de TCP, ya que cada paquete correctamente recibido por el representante, se reconoce mediante un ACK. Sin embargo, este paquete de reconocimiento es enviado a todos los miembros del grupo multipunto, para que cualquier receptor conozca que paquetes han sido

recibidos por el representante. Para asegurar la escalabilidad del protocolo, es necesario ajustar convenientemente los temporizadores exponenciales de cada terminal. Este ajuste se realiza en base a la estimación del número de receptores que participan en la sesión. Por esta razón, el emisor envía periódicamente a todos los miembros del grupo, paquetes de control PIC, que son respondidos mediante paquetes IC, que se transmiten desde el receptor hacia la fuente, de forma punto a punto. A partir del número de respuestas IC, el emisor estima el número de receptores, y calcula los parámetros de los temporizadores. El protocolo RCCMP también utiliza paquetes de control en la fase de inicialización para seleccionar al representante. En este procedimiento, a la solicitud de un paquete PIC todos los receptores del grupo contestan con paquetes IC. Aquel receptor cuyo paquete llegue el último antes de expirar el periodo de gracia, será elegido como representante.

Puede observarse, como en un escenario sin pérdidas, el consumo de ancho de banda se puede dividir en tres términos que se corresponden con los costes debido a los datos, a los ACK y a los paquetes de control.

$$C_{multicast\_nonlosses} = C_{data} + C_{ACK} + C_{control} \tag{7.11}$$

Como los paquetes de datos se distribuyen de forma multipunto a todos los receptores, su coste puede expresarse mediante

$$C_{data} = E(\alpha)E[L_m] \tag{7.12}$$

donde $E[L_m]$ expresa la esperanza del número de saltos del árbol de distribución.

Los ACKs también se distribuyen de manera multipunto. Si cada ACK confirma dos paquetes de datos, se tiene, que el coste que suponen los ACKs, se expresa mediante

$$C_{ACK} = \frac{E(\alpha)}{2} E[L_m] \tag{7.13}$$

Para estimar el tamaño del grupo multipunto, cada $\tau_{request}$ segundos, el emisor distribuye mediante transmisión multipunto un paquete de control de PIC. En RCCMP, este tiempo se ajusta a 500 veces el tiempo de ida y vuelta entre la fuente y el representante. Las simulaciones muestran, que este valor, consigue una buena estimación del número de receptores, introduciendo un coste mínimo [Solera05a]. El coste que introducen estas peticiones periódicas se traduce en

$$C_{request} = \frac{E[L_m]}{\tau_{request}} \tag{7.14}$$

Para calcular, cual es el coste de los paquetes respuesta a estas peticiones periódicas, es conveniente, revisar el procedimiento de estimación del número de miembros del grupo, para un mejor entendimiento. Una vez se recibe el paquete de petición PIC, los receptores activan los temporizadores exponenciales. Cuando cada uno de ellos expira, se envía un informe IC, vía punto a punto, que contiene el tiempo que ha tardado en expirar el temporizador. La fuente envía de forma multipunto un paquete de datos que contiene el valor del temporizador mínimo entre las respuestas recibidas. Los miembros del grupo, que reciben este paquete de datos antes de que su

temporizador expire, cancelan su respuesta, si el valor de su temporizador es mayor que el observado en el paquete de datos.

Se define como E[N'] la esperanza del número de receptores que envían paquetes de control respuesta IC, o que cuyo temporizador tiene un valor entre [z_{min}, z_{min} +2c], donde z_{min} es el temporizador de menor valor, y c es el tiempo de propagación entre la fuente y cualquier receptor [Nonnenmacher98]. Si los temporizadores exponenciales utilizan λ y T como parámetros, la esperanza del número de receptores, que envían un paquete respuesta, se puede expresar como

$$E[N'] = E[N]\frac{e^{\lambda\frac{2c}{T}} - 1}{e^{\lambda} - 1} + e^{\lambda\frac{2c}{T}} \tag{7.15}$$

donde *E[N]* expresa la esperanza del número de receptores activos, ya que los miembros de un grupo multipunto pueden abandonar o adherirse a la sesión de forma dinámica. Así, el consumo de ancho de banda de los paquetes respuesta, para el procedimiento de la estimación de los receptores, es de

$$C_{replies} = \frac{1}{\tau_{request}}\overline{L_u}E[N'] \tag{7.16}$$

A esta expresión hay que añadir el coste del proceso de inicialización de selección de representante que también involucra paquetes IC. Como éste es un proceso que solamente se ejecuta al comienzo de la sesión, se va a expresar mediante una constante que tiene un valor que es función del tamaño del grupo multipunto. El proceso de búsqueda de representante consiste en que la fuente, al inicio de la sesión, envía un paquete de petición PIC para que sea respondido por todos los receptores. A medida que van llegando las respuestas a la fuente, se van produciendo cambios de representante hasta que el tiempo de gracia finalice. El receptor, que envió el último paquete antes de expirar este tiempo, será el representante hasta que aparezca un receptor con peores recursos. El coste debido a los paquetes IC puede describirse mediante

$$C_{replies} = \frac{1}{\tau_{request}}\overline{L_u}E[N'] + C_{inicialization} \tag{7.17}$$

Finalmente, la expresión del coste de una conexión multipunto de E[N] miembros, y cuyo árbol de distribución tiene una longitud de E[L_m] saltos, en un escenario sin pérdidas, se puede resumir mediante la siguiente ecuación:

$$\begin{aligned} C_{multicast\_nonlosses} &= E(\alpha)E[L_m] + \frac{E(\alpha)}{2}E[L_m] \\ &+ \frac{E[L_m]}{\tau_{request}} + \frac{\overline{L_u}E[N']}{\tau_{request}} + C_{inicialization} \end{aligned} \tag{7.18}$$

donde el primer término hace referencia a los paquetes de datos, el segundo a los paquetes de confirmación, y el tercer y cuarto término, contabilizan los paquetes de control para la estimación del número de receptores de la sesión y la selección inicial del representante.

7.2.4 Caso 4: conexión multipunto con pérdidas

Al considerar un escenario con pérdidas, los paquetes retransmitidos y los NAKs, se han de añadir al consumo de ancho de banda de la sesión multipunto. Los primeros se envían de forma multipunto, mientras que los segundos se transmiten de forma punto a punto entre el receptor que ha detectado la pérdida y la fuente.

Existen dos mecanismos, con coste diferentes, para recuperar un paquete perdido que depende de qué receptor ha detectado el salto de secuencia: el representante u otro receptor. Así, el coste de una sesión multipunto en un escenario con pérdidas puede dividirse en los siguientes términos:

$$C_{multicast\_losses} = C_{multicast\_nonlosses} + C_{representative\_losses} + C_{anyreceiver\_losses} \tag{7.19}$$

el primer término hace referencia al coste calculado en el apartado anterior, al que hay que sumar los costes debido a pérdidas ocurridas en el representante, y en otros receptores.

Se va a comenzar calculando el coste que suponen las pérdidas detectadas en el representante, para ello se va a revisar el procedimiento, para derivar más intuitivamente las expresiones. La fuente detecta que el representante ha sufrido pérdida de paquete, porque recibe más de tres ACKs con el mismo número de secuencia, o porque expira el temporizador de retransmisión. Como estos mecanismos no introducen ningún paquete adicional, el coste debido a pérdidas detectadas por el representante, se reduce a calcular el coste de los paquetes retransmitidos.

Se define p_r como la probabilidad de que el representante pierda un paquete. Así, la probabilidad de que un paquete sea transmitido k veces es

$$N_{tx\_r} = p_r^{\,k-1}\left(1 - p_r\right) \tag{7.20}$$

Como en el modelo punto a punto con pérdidas, la esperanza del número de retransmisiones entre la fuente y el representante se calcula como

$$E[N_{rtx\_r}] = \frac{p_r}{1 - p_r} \tag{7.21}$$

Por lo tanto, el coste debido a pérdidas detectadas en el representante se expresa como la tasa de paquetes retransmitidos por el número de saltos del árbol de la distribución multipunto

$$C_{representative\_losses} = E(\alpha)\frac{p_r}{1 - p_r}E[L_m] \tag{7.22}$$

Esta ecuación es similar a la ecuación (7.8) que modela el coste de paquetes retransmitidos en una conexión punto a punto. La diferencia radica que en este caso la distribución se realiza de forma multipunto.

Si las pérdidas se detectan en cualquier receptor excepto en el representante, el proceso de recuperación es diferente. En este caso, cuando un receptor pierde un paquete debe esperar hasta la recepción del ACK, para observar si ese paquete ha sido perdido también por el representante. Si ha sido así, entonces el procedimiento es el descrito anteriormente; si no, se activa el temporizador exponencial, que regula el tiempo que hay que esperar para poder enviar una solicitud de retransmisión. Si el

paquete perdido no se ha recibido y el temporizador expira, éste se envía. La fuente retransmite el paquete al recibir el paquete de control. Si el paquete perdido es recibido antes de que el temporizador expire, entonces se cancela el NAK.

En este apartado se va a utilizar la misma suposición utilizada en el capítulo 6 para modelar el caudal de RCCMP. Se considera que solamente se generará un paquete retransmitido por cada paquete perdido, ya que se supondrá que los NAKs, solicitando el envío de un mismo paquete, van a llegar a la fuente, agrupados en un intervalo de tiempo no superior al parámetro *nak_factor* por el tiempo de ida y vuelta entre la fuente y el representante. Por lo tanto, cada paquete perdido se recuperará con un paquete retransmitido.

Así, la expresión que contabiliza el consumo de ancho de banda del proceso de detección y recuperación de pérdidas ocasionadas en cualquier receptor, debe tener en cuenta el ancho de banda consumido por los paquetes retransmitidos y los NAKs.

Se define $p_{receiver}$ como la probabilidad de que cualquier receptor excepto el representante pierda un paquete. Entonces, la probabilidad de que un paquete sea transmitido exactamente *k* veces se expresa como

$$N_{tx\_receiver} = p^{k-1}_{receiver}\left(1 - p_{receiver}\right) \tag{7.23}$$

y la esperanza del número de retransmisiones es

$$E[N_{rtx\_receiver}] = \frac{p_{receiver}}{1 - p_{receiver}} \tag{7.24}$$

Esta expresión es equivalente a la obtenida en (7.21).

La ecuación, que modela el coste debido a los paquetes retransmitidos, es similar a la obtenida en (7.22), pero ahora se utiliza la probabilidad de que cualquier receptor pierda un paquete ($p_{receiver}$)

$$C_{retransmittedpackets} = E(\alpha)\frac{p_{receiver}}{1 - p_{receiver}} E[L_m] \tag{7.25}$$

Para calcular el consumo de ancho de banda de los NAKs, se debe contabilizar el número de confirmaciones negativas recibidas por paquete perdido. Esto dependerá del número de receptores que detectan la pérdida y de los temporizadores exponenciales. Se definen E[N’’] y E[N’’’], como la esperanza del número de receptores que detectan el paquete perdido, y la esperanza del número de receptores que envían un NAK, respectivamente. E[N’’’] depende de los parámetros λ y T de los temporizadores exponenciales, y se puede expresar como

$$E[N'''] = E[N'']\frac{e^{\lambda\frac{2c}{T}} - 1}{e^{\lambda} - 1} + e^{\lambda\frac{2c}{T}} \tag{7.26}$$

donde *E[N’’’]* expresa el número medio de receptores que envían respuesta, bajo la hipótesis de que el número de receptores, *E[N’’],* que detectan la pérdida sea suficientemente grande.

Una vez calculado el número de NAKs, y teniendo en cuenta que estos se transmiten de forma punto a punto, se puede fácilmente contabilizar su coste.

$$C_{NAK} = E(\alpha)\frac{p_{receiver}}{1 - p_{receiver}} E[N''']\overline{L_u} \tag{7.27}$$

Esta ecuación expresa que cada paquete perdido por E[N'’] receptores, que no son el representante, se corresponde con E[N'’’] paquetes de solicitud de retransmisión.

Si se agrupa el coste que suponen las pérdidas para una sesión multipunto se tiene la siguiente expresión

$$\begin{aligned} C_{losses} = {} & E(\alpha)\frac{p_r}{1 - p_r} E[L_m] + \\ & E(\alpha)\frac{p_{receiver}}{1 - p_{receiver}} E[L_m] + E(\alpha)\frac{p_{receiver}}{1 - p_{receiver}} E[N''']\overline{L_u} \end{aligned} \tag{7.28}$$

que se corresponden a los paquetes de datos retransmitidos y a los NAKs.

Finalmente, el consumo de ancho de banda total de una sesión multipunto de un tamaño de E[N] receptores, en un escenario con pérdidas es el siguiente:

$$\begin{aligned} C_{multicast\_losses} = {} & E(\alpha)E[L_m] + \frac{E(\alpha)}{2} E[L_m] \\ & + \frac{E[L_m]}{\tau_{request}} + \frac{\overline{L_u} E[N']}{\tau_{request}} + C_{inicialization} \\ & + E(\alpha)\frac{p_r}{1 - p_r} E[L_m] + E(\alpha)\frac{p_{receiver}}{1 - p_{receiver}} E[L_m] \\ & + E(\alpha)\frac{p_{receiver}}{1 - p_{receiver}} E[N''']\overline{L_u} \end{aligned} \tag{7.29}$$

7.2.5 Cálculo de la esperanza del número de receptores que detectan pérdidas

E[N'’] se define como la esperanza del número de receptores que detectan el mismo paquete perdido y puede ser calculado como

$$E[N''] = \sum_{i=1}^{N} i P(i\_receivers) \tag{7.30}$$

donde N es el número de receptores activos, y la probabilidad de que *i* receptores detecten la pérdida de un paquete se expresa como *P(i_receivers)*. Esta probabilidad depende de la topología, por lo tanto se acotará E[N'’] a partir de la topología con más y menos enlaces (figura 7.1 y figura 7.2).

Primero, se va a calcular *P(i_receivers)* en la topología con menos enlaces posibles (figura 7.1). Para ello, es necesario definir p_L como la probabilidad de error por bit sobre cada enlace. La expresión que se obtiene es la siguiente:

$$P(i\_receivers) = \left(\overline{L_u} - 1\right) p_L \sum_{k=0}^{i} {p_L}^k (1 - p_L)^{i-k} + (1 - p_L)^{\left(\overline{L_u} - 1\right)} {p_L}^i \tag{7.31}$$

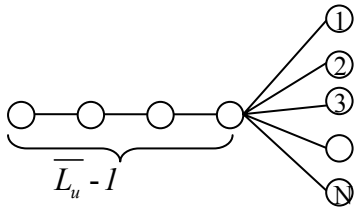

Figura 7.1 Topología con el menor número de enlaces posibles.

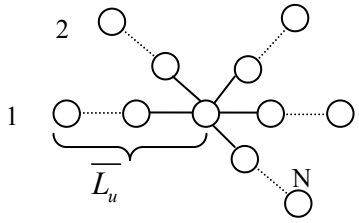

Figura 7.2 Topología con el mayor número de enlaces posibles.

Para pequeños valores de p_L, la expresión de (7.31) puede simplificarse de la siguiente manera:

$$P(i\_receivers) \approx \left(\overline{L_u} - 1\right) p_L (1-p_L)^i + (1-p_L)^{\overline{L_u}-1} {p_L}^i \tag{7.32}$$

Por otro lado, la topología con mayor número de enlaces es una estrella cuyas ramas tienen una longitud de $\overline{L_u}$ enlaces. En este caso, la probabilidad de que *i* receptores detecten un paquete perdido puede ser aproximado por

$$P(i\_receivers) \approx \left(\overline{L_u} p_L\right)^i \tag{7.33}$$

Cualquier otra topología tiene un número de enlaces que está acotado entre estas dos. Por lo tanto, para calcular el número medio de receptores que detectan un mismo paquete perdido, que es un factor que se utiliza para computar el consumo de ancho de banda, se busca una cota superior de este valor. A partir del estudio de las ecuaciones (7.32) y (7.33), se observa que para valores adecuados de p_L, la ecuación (7.32) obtiene valores mayores que (7.33). Por lo tanto, se va usar esa expresión para acotar E[N''']. La expresión que se obtiene es la siguiente:

$$E[N''] \le \sum_{i=1}^{N} i((\overline{L_u} - 1) p_L (1-p_L)^i + (1-p_L)^{\overline{L_u}-1} {p_L}^i) =$$

$$= (\overline{L_u} - 1) p_L \sum_{i=1}^{N} i(1-p_L)^i + (1-p_L)^{\overline{L_u}-1} \sum_{i=1}^{N} i {p_L}^i \tag{7.34}$$

En el caso de que se trabaje con gran número de receptores, es decir, para valores elevados de N se tiene

$$E(N'') \le (\overline{L_u} - 1) p_L (1 - p_L) \frac{1}{1 - (1 - p_L)^2} + (1 - p_L)^{\overline{L_u} - 1} p_L \frac{1}{1 - p_L^{\,2}} \tag{7.35}$$

7.3 RESULTADOS

Los resultados, que se muestran en este apartado, pretenden validar el modelo matemático, que se ha desarrollado en la sección anterior, a partir de los datos obtenidos mediante simulación y expuestos en el capítulo anterior. Las simulaciones realizadas contabilizan la cantidad y tipo de paquetes de una sesión RCCMP en función del número de receptores para distintas probabilidad de error. Estos resultados serán muy útiles para determinar la validez del análisis de la carga del protocolo.

También, se comparará el coste de la distribución multipunto frente a la punto a punto. Para ello, se confrontará el ancho de banda que consume una sesión RCCMP con N receptores, con N conexiones punto a punto. La comparación se realizará en base a resultados obtenidos mediante el modelo matemático y mediante simulaciones.

7.3.1 Ajuste del modelo matemático

La ecuación (7.29) expresa el ancho de banda consumido por una sesión RCCMP. Para evaluar la bondad del modelo matemático, se van a comparar los resultados obtenidos mediante esta expresión con los obtenidos mediante simulación, y que han sido expuestos en el apartado 5.7. Para que estos resultados sean equiparables, se va a trabajar sobre la misma topología en árbol con tres niveles con las que se realizaron las simulaciones. La figura 5.40 muestra la topología.

El modelo matemático propuesto es función de la tasa de datos útil, de la probabilidad de pérdida, del número medio de NAKs, de los paquetes de control y de la topología. Por lo tanto, es necesario calcular la probabilidad de pérdidas en el representante, p_r, y la probabilidad de pérdidas en otros receptores $p_{receiver}$, para introducirlos en el modelo. Estos valores se obtienen por medio de las simulaciones. La primera, p_r, se calcula como el cociente entre el número de indicaciones de pérdida en el representante y el número total de paquetes enviados. La segunda, $p_{receiver}$, responde al cociente entre los paquetes retransmitidos por solicitud mediante NAK y el número de paquetes totales.

E[N'] y E[N''], que expresan el número medio de NAKs por paquete perdido, y el número medio de paquetes de control IC, que se responden por paquete de control PIC, se han igualado al parámetro *desired_feedback*. Éste es un parámetro del temporizador exponencial, que controla el número medio de respuestas. Un valor igual a 2 es el que se ha utilizado en las simulaciones, y es el que se asigna a estas dos esperanzas, que se introducirán en el modelo.

La constante de inicialización, $C_{inicialization}$, que contabiliza el número de paquetes de control respuesta en el proceso inicial de búsqueda de representante, se iguala al tamaño

del grupo multipunto. La tasa útil en paquetes por segundo, E(α), se ha igualado a 56 paquetes/s, lo que supone una tasa de 469.5Kbps.

En la topología de la figura 5.40, el parámetro $\overline{L_u}$ vale 2. El valor de L_m, utilizado para evaluar la expresión matemática, se calcula contabilizando el número de ramas que forman el árbol de distribución. Esta variable es función del número de receptores. Los resultados se obtienen considerando tamaños de grupo desde 3 a 900 receptores.

Para comenzar a evaluar la bondad del modelo, se evalúa la ecuación (7.29) en un escenario donde no se introducen pérdidas de canal en los enlaces y se considera un tamaño de buffer de 90 paquetes. A partir del cálculo de las probabilidades $p_{receiver}$ y p_r, y con las demás variables ajustadas según se ha referido, se estima el consumo de ancho de banda de una sesión RCCMP. En este caso, se ha considerado que las unidades del coste son paquetes, y no paquetes por salto/s como en el estudio teórico, para que estos resultados sean fácilmente comparables con los obtenidos mediante simulación.

La figura 7.3 muestra el número de paquetes estimados para una sesión RCCMP, considerando un tiempo de 400s. En estas curvas se presentan los diferentes términos que contribuyen al coste: los paquetes útiles, los paquetes retransmitidos debido a pérdidas en el representante y a pérdidas en otros receptores, los paquetes de control y los NAKs. En esta figura, se observa que el número de paquetes retransmitidos por solicitud explícita supera al número de paquetes retransmitidos solicitados por el representante. Esto es debido a que los valores obtenidos para $p_{receiver}$ son mayores que para p_r. En concreto, la primera alcanza el 68% de las pérdidas totales. La topología, que se está considerando, se caracteriza por tener todos los enlaces de la misma capacidad y retardo, por lo que no existe un receptor, que claramente pueda ser seleccionado como representante. Por esta razón, las pérdidas se van a distribuir entre todos los receptores. Puede observarse en la gráfica, que el número de NAKs se muestra independiente al número de receptores. En este escenario, también los paquetes útiles contribuyen de forma constante al coste en función del número de receptores.

El número total de paquetes de control se puede dividir entre los PIC, que dependen del parámetro *feedback_factor,* que determina la frecuencia para reenviar un mensaje de este tipo, y está ajustado a 500 veces el RTT; los paquetes respuesta IC, que se utilizan para estimar los receptores activos del grupo, y se ajustan a partir del número medio de respuestas (*desired_feedback);* y a los paquetes IC, que se utilizan para seleccionar al representante en el instante inicial, y se igualan al número de receptores. Por lo tanto, la curva de paquetes de control es proporcional al tamaño del grupo, debido a este último término, que se representa por medio de la constante de inicialización.

En la figura 7.4 se compara el número de paquetes obtenidos mediante simulación y utilizando el modelo matemático obtenido de la expresión (7.29). Como puede observarse, el coste calculado a partir del modelo es más optimista que el obtenido a partir de las simulaciones. Sin embargo, el modelo es capaz de capturar la tendencia del coste en función del número de receptores.

Si se introducen pérdidas de canal con función de distribución uniforme de 0.01% en todos los enlaces, la figura 7.5 muestra el número de paquetes estimados. En este escenario, puede observarse como el número de NAKs y el de paquetes retransmitidos mediante solicitud explicita aumenta en función del número de receptores. Al aumentar las pérdidas también crece el número de paquetes retransmitidos detectados en el representante, pero en menor medida. Los paquetes útiles y de control no se ven afectados al aumentar la probabilidad de pérdidas. La comparación del coste calculado

mediante el modelo y mediante simulación puede verse en la figura 7.6. En este caso el modelo presenta unos resultados más pesimistas a partir de un tamaño de grupo de 360 miembros.

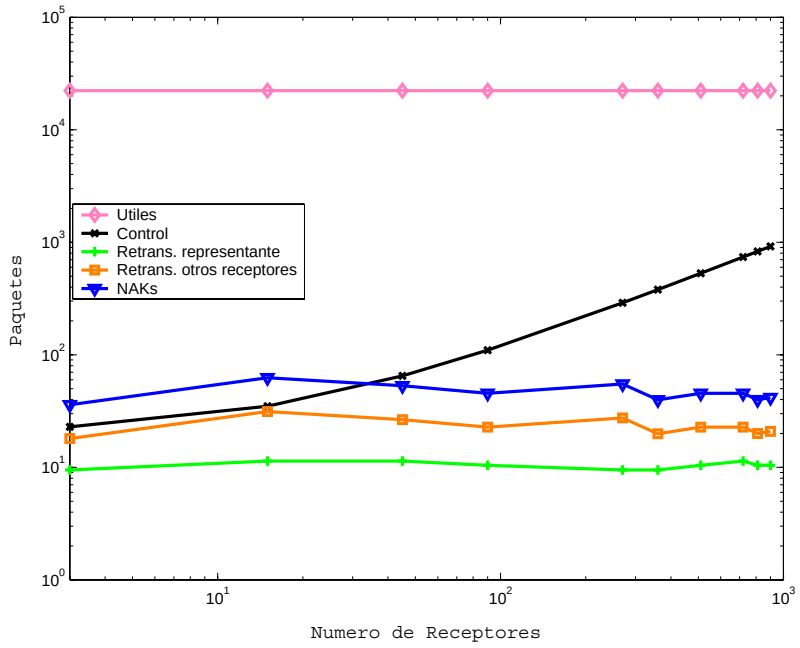

Figura 7.3 Número de paquetes de una sesión RCCMP sin pérdidas y tamaño de cola=90, según el modelo matemático.

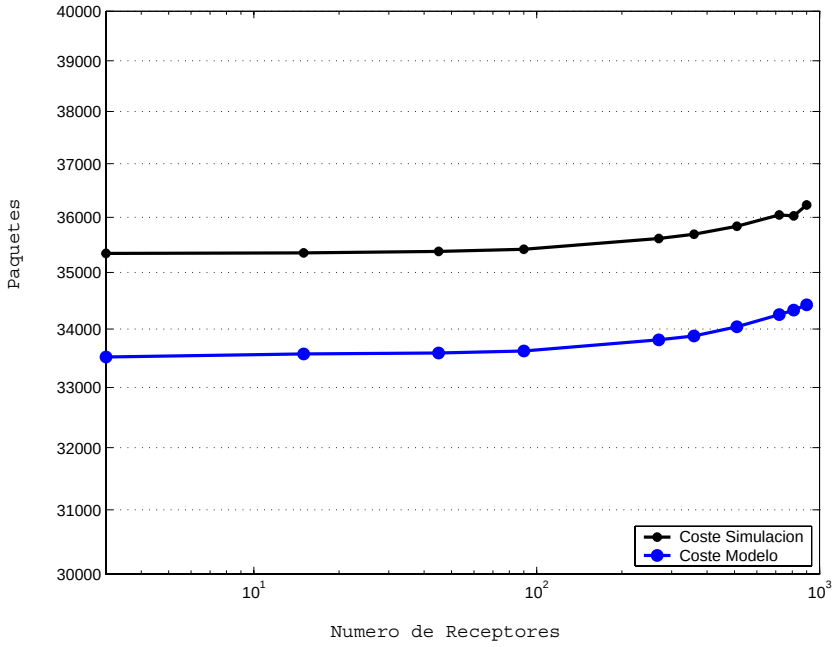

Figura 7.4 Coste calculado mediante simulación y analíticamente para una sesión RCCMP sin pérdidas y tamaño de cola=90.

Si se aumentan las pérdidas de canal que se introducen en los enlaces al 0.1%, los resultados se presentan en las gráficas 7.7 y 7.8. En este caso, las tendencias del experimento anterior se ven agudizadas. El modelo matemático se muestra mucho más pesimista que las simulaciones a partir de tamaños de grupo de 45 miembros.

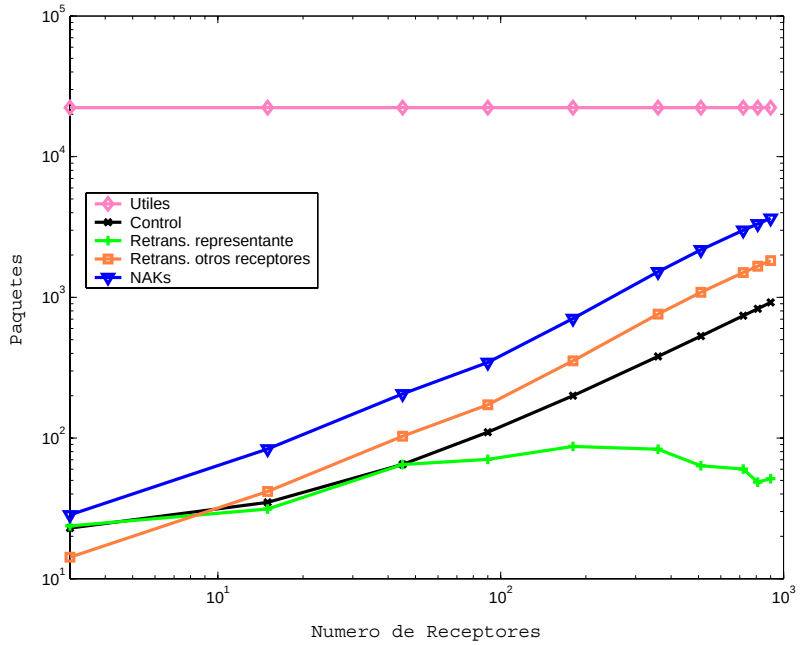

Figura 7.5. Número de paquetes para una sesión RCCMP con p=0.01% y tamaño de cola=90.

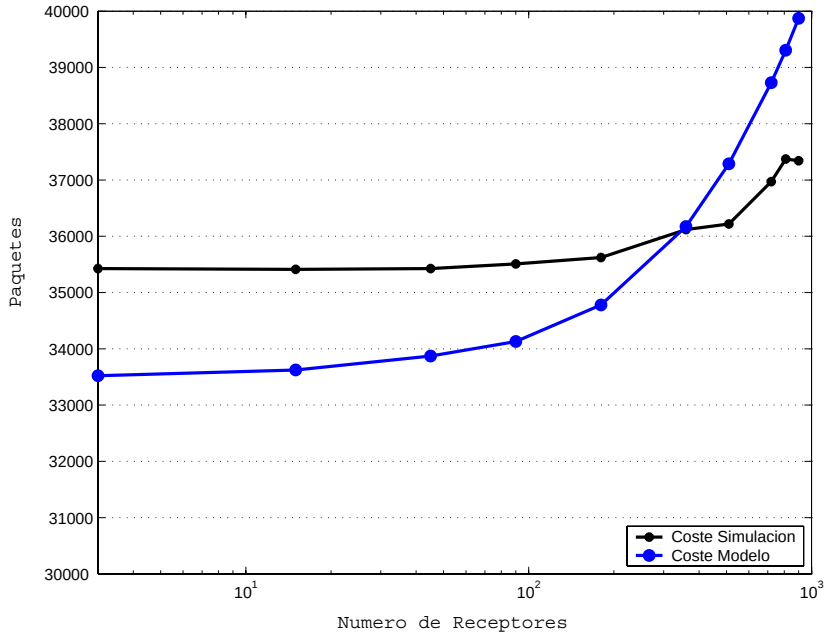

Figura 7.6 Coste calculado mediante simulación y analíticamente para una sesión RCCMP con p=0.01% y tamaño de cola=90.

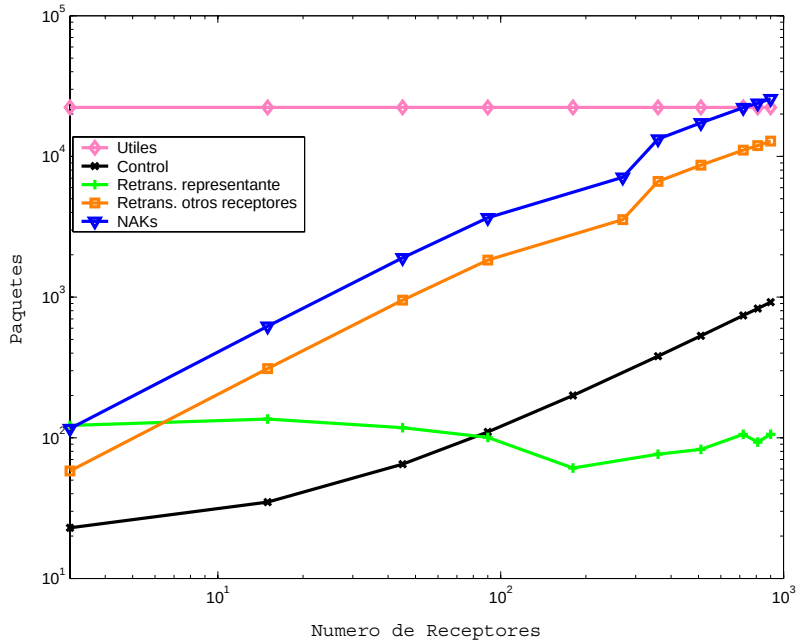

Figura 7.7 Número de paquetes para una sesión RCCMP con p=0.1% y tamaño de cola=90.

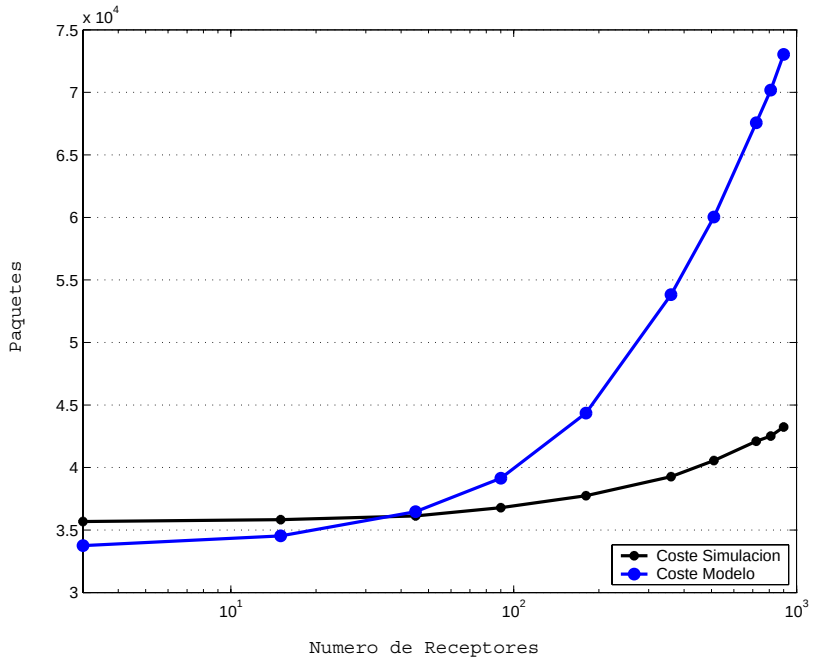

Figura 7.8 Coste calculado mediante simulación y analíticamente para una sesión RCCMP p=0.1% y tamaño de cola=90.

Para evaluar la bondad del modelo, se calcula el error relativo como:

$$\frac{\left|Coste_{modelo} - Coste_{simulacion}\right|}{Coste_{simulacion}} \tag{7.36}$$

En la figura 7.9 se muestra el valor del error para diferentes probabilidades de pérdida en el canal. Un error más pequeño muestra una mejor adecuación del modelo a los resultados obtenidos mediante simulación.

La gráfica del error relativo muestra que para probabilidades de pérdida superiores al 0.1%, el modelo matemático se ajusta peor a los resultados obtenidos por simulación. Sin embargo, para probabilidades de pérdida más pequeñas, el error relativo es de un 5%.

La principal limitación del modelo matemático reside en contabilizar el número de paquetes útiles. Para el desarrollo del análisis se ha considerado que E(α) es constante e independiente del número de receptores. Sin embargo, en las simulaciones se observa que cuando el número de receptores del grupo multipunto aumenta, el número de paquetes útiles disminuye (figuras 5.42 y 5.43). Esta limitación es más importante cuantas más pérdidas se introduzcan en el canal. Si en vez de considerar E(α) constante e igual a 56 paquetes/s, como en las experiencias anteriores, se utiliza en el modelo matemático el valor de E(α) obtenido de las simulaciones, como el número de paquetes útiles dividido entre el tiempo de simulación, se obtienen mejores resultados. Las gráficas 7.10, 7.11 y 7.12 muestran el coste estimado mediante el modelo y el obtenido mediante simulación.

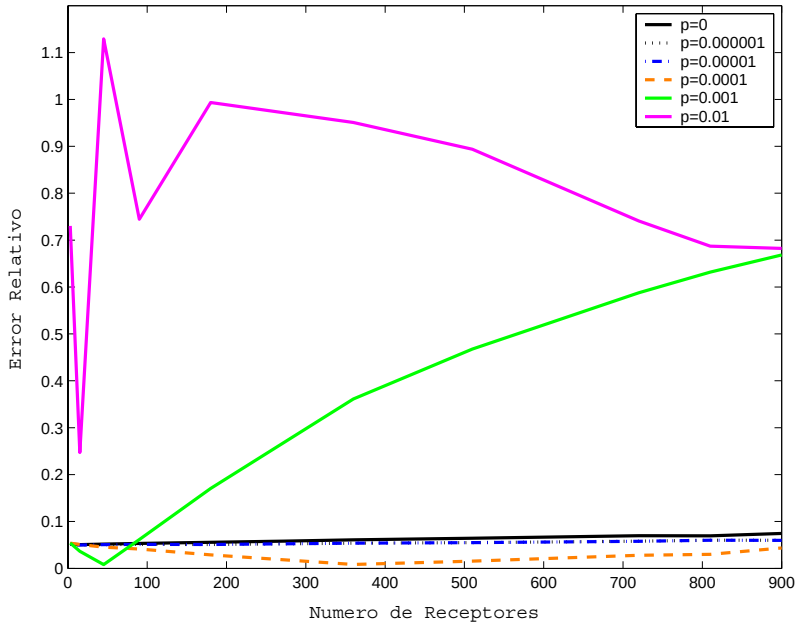

Figura 7.9 Error relativo para una sesión RCCMP con tamaño de cola=90.

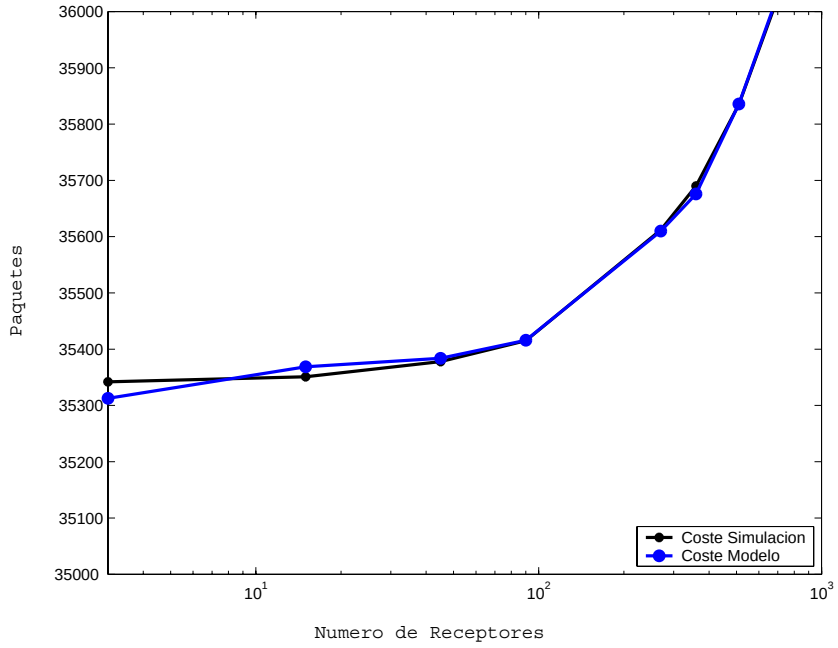

Figura 7.10 Coste calculado mediante simulación y analíticamente para una sesión RCCMP con p=0% y tamaño de cola=90.

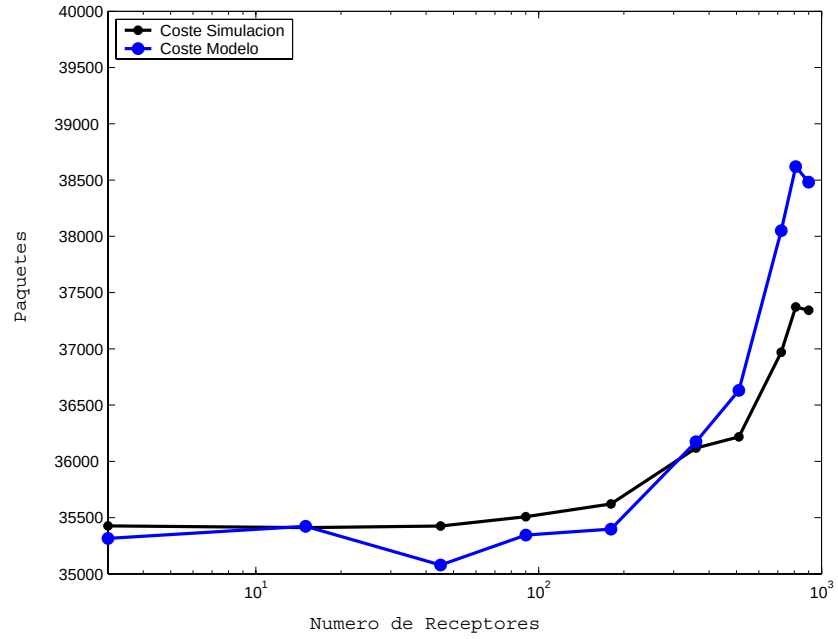

Figura 7.11 Coste calculado mediante simulación y analíticamente para una sesión RCCMP con p=0.01% y tamaño de cola=90.

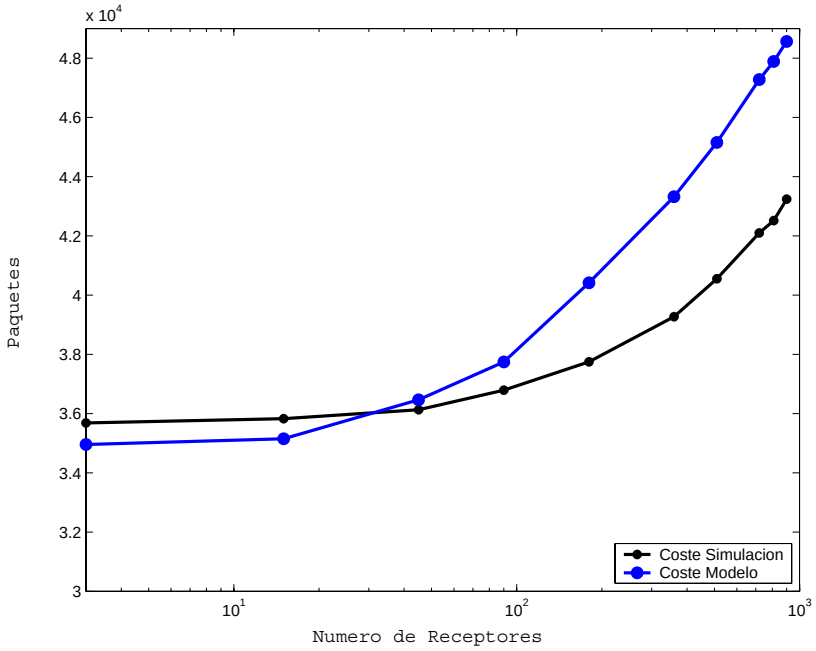

Figura 7.12 Coste calculado mediante simulación y analíticamente para una sesión RCCMP con p=0.1% y tamaño de cola=90.

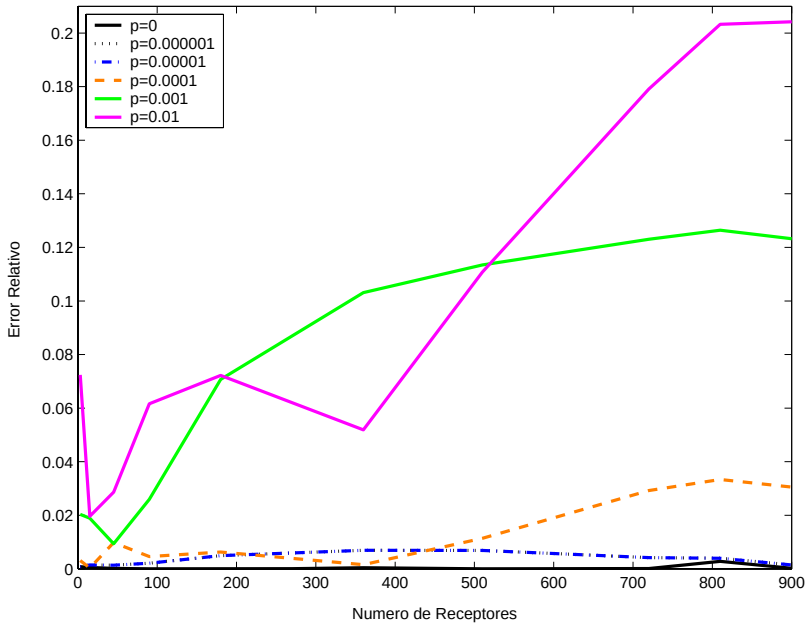

Figura 7.13 Error relativo para una sesión RCCMP con tamaño de cola=90.

La gráfica 7.13 muestra el error relativo. Aunque, como en el caso anterior, se muestra que para probabilidades de pérdida superiores al 0.1%, el modelo matemático se ajusta peor a los resultados obtenidos por simulación, el valor de ese error es más pequeño para cualquier probabilidad. Para pérdidas de canal pequeñas, el error relativo es inferior al 0.34%.

7.3.2 Comparación del coste para conexiones multipunto y punto a punto

A priori, la transmisión multipunto resulta más eficiente que la punto a punto, ya que solamente es necesario duplicar los paquetes cuando las rutas a los distintos receptores divergen. Sin embargo, este tipo de comunicación es más compleja, y requiere de más paquetes de control para coordinar a los distintos miembros del grupo. A continuación, se compara el coste de dar servicio a N receptores mediante transmisión punto a punto y multipunto. Primero, se estimará el coste a partir del modelo matemático sobre una topología general. Para ello, se supone que el número medio de enlaces entre dos receptores cualquiera (L_u) es 5. Para calcular la longitud del árbol multipunto, se utilizará la expresión de Chuang [Chuang98]. Aquí, se reproduce la formula:

$$L_m = L_u N^{0.8} \tag{7.37}$$

Se considera que el retardo en cada enlace es de 0.01ms. En cuanto a los paquetes de control, se supone que la fuente envía un paquete cada 100**RTT_representative*, y que el número medio de paquetes respuesta es de 4. Con estas condiciones, la figura 7.14 muestra el coste de enviar 1 paquete en un escenario donde no hay pérdidas. No se contemplan pérdidas de ningún tipo, ni por saturación, ni introducidas en el canal.

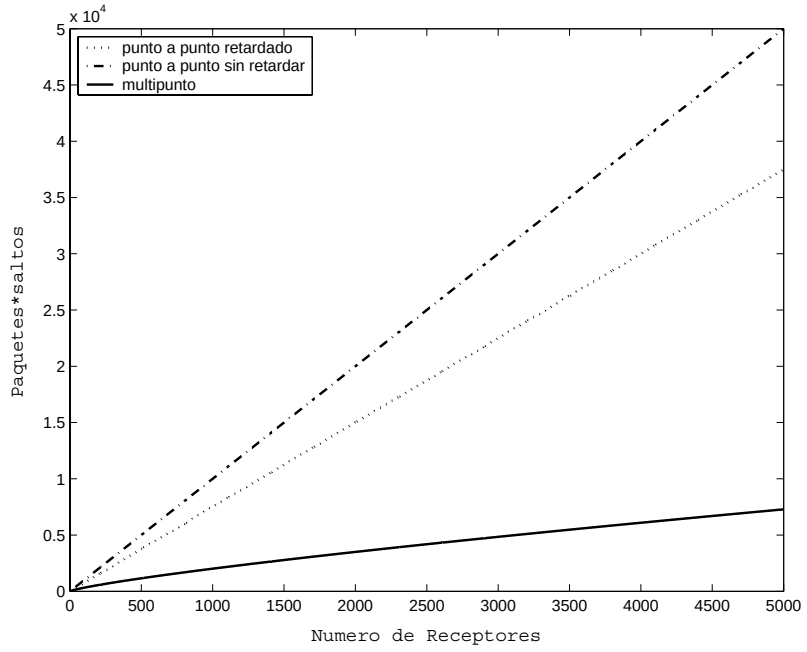

Figura 7.14 Comparando el coste de enviar 1 paquete con distribución multipunto y punto a punto sin pérdidas.

Como puede observarse, el coste de enviar un paquete a N receptores mediante distribución punto a punto es mucho menos eficiente que una transmisión multipunto, aunque esta última introduzca más paquetes de control. La figura 7.15 compara el coste, cuando la tasa de paquetes útiles es de 5Kbps.

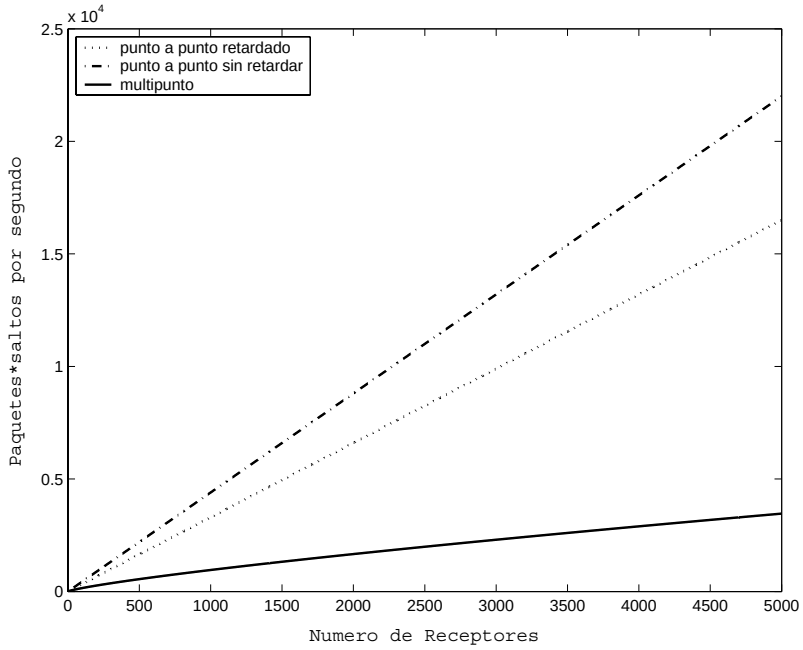

Figura 7.15 Comparando el coste de enviar 5Kbps con distribución multipunto y punto a punto sin pérdidas.

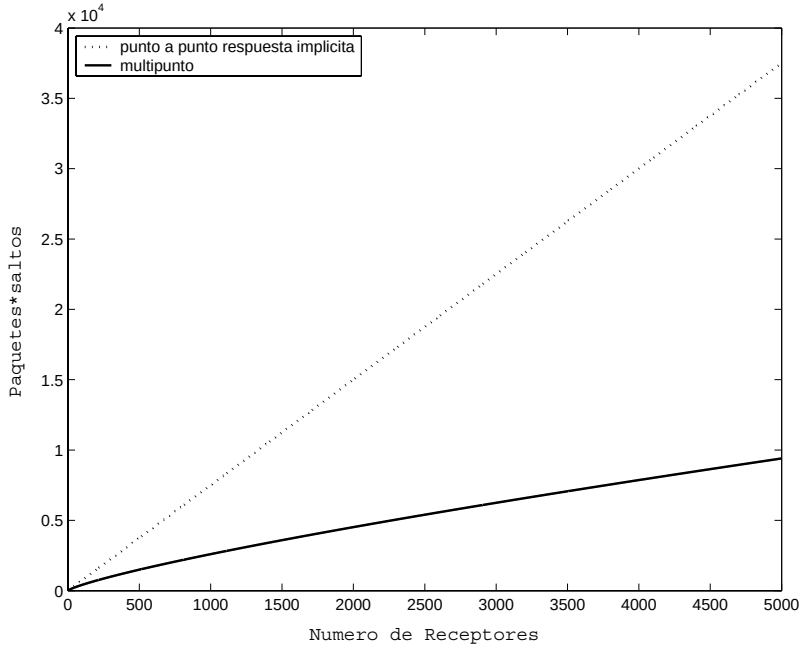

Figura 7.16 Comparando el coste de enviar 1 paquete con distribución multipunto y punto a punto con $p=10^{-8}$.

Si se considera un escenario con enlaces con una probabilidad de error por bit igual a 10^{-8}, se obtienen los resultados mostrados en las gráficas 7.16 y 7.17. En este caso, la tendencia es la misma que para el caso anterior, la transmisión multipunto presenta mejor eficiencia. Cuando se consideran pérdidas, hay que añadir a los paquetes de datos y los ACKs, los paquetes retransmitidos, y en el caso multipunto, los NAKs.

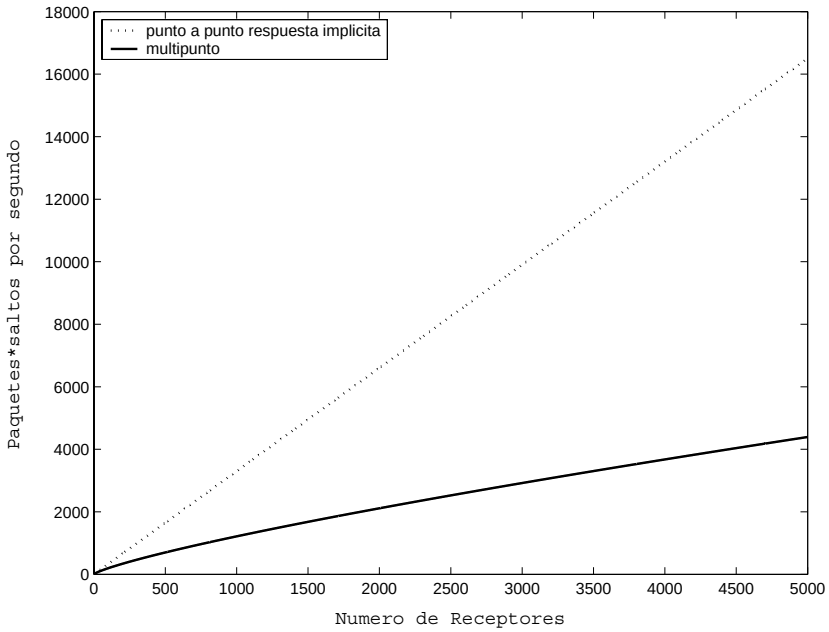

Figura 7.17 Comparando el coste de enviar 5Kbps con distribución multipunto y punto a punto con $p=10^{-8}$

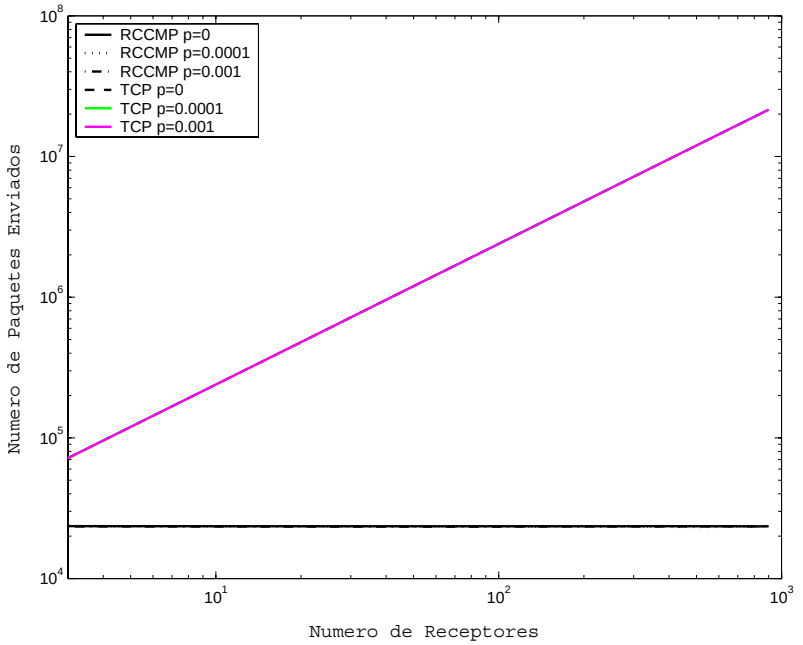

Figura 7.18 Número de paquetes para una distribución multipunto y punto a punto para distintas probabilidades de error.

Los resultados obtenidos por medio del modelo matemático destacan la gran ventaja de las comunicaciones multipunto.

A continuación, se va a realizar el mismo estudio, pero a través del simulador ns-2, utilizando la topología arbolada de la figura 5.40. En este caso, también, se va a comparar el número de paquetes enviados para el caso de un grupo multipunto de N receptores, y N conexiones punto a punto. Todos los enlaces se configuran con una capacidad de 0.5Mbps y 1ms de retardo de propagación. Como la topología de la figura 5.40 tiene un valor de $\overline{L}_u$ igual a 2, este árbol de distribución debe compararse con N conexiones punto a punto que tengan un número de saltos entre el receptor y el emisor de 2.

Como puede observarse en la figura 7.18, los resultados mediante simulación confirman que la distribución multipunto lleva asociado un coste menor. Cuanto mayor es el número de receptores la eficiencia de la transmisión multipunto aumenta con respecto a la punto a punto.

7.4 CONCLUSIONES

Uno de los factores que afectan a la escalabilidad de un protocolo multipunto es la cantidad de paquetes de control transmitidos entre los receptores y la fuente, que no han de aumentar de forma proporcional al número de miembros del grupo. Al evaluar un protocolo y determinar hasta que punto es escalable, es importante, estimar el ancho de banda consumido. En este capítulo se ha propuesto un método de análisis para calcular el ancho de banda que consume un protocolo de transporte punto a punto o multipunto en un escenario con o sin pérdidas. Las expresiones obtenidas se han validado a partir de los resultados obtenidos mediante simulación que muestran el número de paquetes para una sesión RCCMP en función del número de receptores para distintas probabilidades de error.

CAPÍTULO 8

DESCRIPCIÓN Y EVALUACIÓN DEL PROTOCOLO RVCMP

8.1 INTRODUCCIÓN

Tan importante como evitar la saturación de la red es que los protocolos de transporte multipunto fiable se comporten de forma equitativa con los protocolos existentes en Internet, en concreto, con TCP. Por ello, los controles de congestión multipunto desarrollados tienden a emular el comportamiento de este protocolo. PGMCC [Rizzo00], TMFCC [Widmer01b], ORMCC [Li02] son algunas propuestas de controles de congestión en que por medio de un mecanismo de ventana o de tasa emulan el comportamiento de TCP.

Desde la primera versión de TCP [RFC793], distintas implementaciones han surgido con el motivo de mejorar su comportamiento a medida que el número de usuarios de Internet crecía. Así, la implementación TCP Tahoe [RFC2001] aparece en 1988 y perfecciona el control de congestión, introduciendo la fase transitoria *(slow-start)*, mejorando la estimación del cálculo del tiempo de ida y vuelta, modificando los algoritmos de retransmisión y ajustando de forma dinámica el tamaño de la ventana de transmisión. En 1990 aparece la implementación TCP Reno [RFC2581] que incorpora los mecanismos de recuperación y retransmisión rápida. En el [RFC1323] se especifican una serie de extensiones para mejorar el rendimiento de TCP en redes con altas velocidades y tiempos de propagación elevados. En 1994 aparece la implementación TCP Vegas [Brakmo95] que incorpora un mecanismo proactivo que evita la congestión.

En concreto TCP Vegas modifica los siguientes procedimientos de TCP Reno.

- Desarrolla un nuevo mecanismo para la retransmisión basado en calcular el tiempo de ida y vuelta de forma más frecuente y precisa, para avanzar el instante de la retransmisión del segmento perdido.
- Modifica el mecanismo de control de congestión en fase permanente *(congestion avoidance)*. La ventana de transmisión varía en base a la observación del caudal que es comparado con un caudal esperado. Si estos valores están próximos, significa que hay suficiente ancho de banda

disponible y que la ventana de transmisión puede ser incrementada. Sin embargo, si el caudal medido es inferior al caudal esperado, entonces es señal de una posible saturación de la red, con lo que la ventana será reducida.

- Modifica el mecanismo transitorio del control de congestión para poder detectar también en esta fase la congestión incipiente.

Como han demostrado ciertos estudios estas modificaciones en el control de congestión de TCP Reno redundan en un mejor aprovechamiento del canal. Los trabajos de [Hengar00], [Brakmo94], [Brakmo95] muestran que TCP Vegas consigue incrementar el caudal en un 40% - 70% más que TCP Reno con un menor número de paquetes retransmitidos.

El protocolo RVCMP [Solera05d] *(Reliable Vegas Congestion controlled Multicast Protocol)* modifica el control de congestión de RCCMP para asemejarlo al de TCP Vegas, y de esta manera mejorar la eficiencia. Otras propuestas para entornos multipunto fiable que incorporan este tipo de control de congestión son MTCP [Rhee99] para unitasa y [Ait04] [Mahanti05] para multitasa. El primer trabajo incorpora los mecanismos de TCP Vegas en la fase transitoria *(slow-start)* y permanente *(congestion avoidance)* de un protocolo que emplea un árbol jerárquico lógico para agregar el tráfico de realimentación. Por otro lado, las propuestas para multitasa utilizan la ecuación que modela el caudal para TCP Vegas [Samios03] como criterio para unirse o abandonar una capa.

Este capítulo se estructura de la siguiente manera. La próxima sección propone un nuevo mecanismo de control de congestión para RVCMP basado en la implementación TCP Vegas. A continuación, se evalúa mediante simulación el comportamiento del protocolo, y finalmente, los resultados obtenidos son comparados con RCCMP.

8.2 CONTROL DE CONGESTIÓN DE RVCMP

El control de congestión de RVCMP emula el comportamiento de la implementación TCP Vegas. Las fases transitoria *(slow-start)* y permanente *(congestion avoidance)* del control de congestión del protocolo RCCMP van a resultar modificadas para introducir los nuevos mecanismos que esta implementación propone.

RVCMP estima el ancho de banda mínimo disponible en el árbol de distribución multipunto estableciendo un lazo de control cerrado entre la fuente y el representante. Como en TCP Vegas, en la etapa permanente *(congestion avoidance)* se calcula la diferencia entre la tasa esperada y la actual para estimar el nivel de congestión en la red, y así, ajustar convenientemente la ventana. La tasa esperada se calcula como

$$Tasa\_esperada = \frac{W}{BaseRTT} \tag{8.1}$$

donde *W* es el tamaño de la ventana, que representa la cantidad de segmentos que hay en tránsito, y *BaseRTT* se define como el mínimo de todos los RTTs entre la fuente y el representante. Esta última variable se actualiza por dos razones, porque se observa un valor más pequeño de RTT o porque se produce un cambio de representante. Cada RTT se calcula la diferencia entre la tasa esperada y la actual, y se almacena en la variable *Diff*. Al igual que en TCP Vegas, se definen dos umbrales α y β, con $\alpha \leq \beta$, que

representan la cantidad de tráfico extra que circulan por la red. Cuando *Diff* < α, la ventana de transmisión se incrementa en $1/W$, y cuando *Diff* > β, la ventana se decrementa en un paquete en el próximo RTT. La ventana permanece inalterada si *Diff* tiene un valor entre los dos umbrales. Intuitivamente, se puede observar que cuanto mayor es la diferencia entre la tasa esperada y la actual, las condiciones de la red son más desfavorables, mientras que si está diferencia se aproxima, la fuente puede no estar aprovechando convenientemente el ancho de banda disponible.

TCP Vegas también modifica la fase transitoria *(slow-start)* de TCP Reno para detectar y prevenir la congestión en esta fase. RVCMP utiliza esta versión modificada. El algoritmo inicializa la ventana de transmisión dependiendo del tamaño del paquete. Cuando llega un ACK, la ventana de transmisión se incrementa en un paquete. El crecimiento exponencial propio de esta fase solamente se permite una vez cada dos RTT. Entre dos incrementos de ventana, la ventana de transmisión permanece fija para que se pueda comparar la tasa esperada y la actual. Cuando la diferencia entre estas dos tasas es mayor que γ, se sale de esta fase.

RVCMP no incluye el mecanismo de retransmisión que propone TCP Vegas. Este consiste en calcular de forma más exacta y frecuente el RTT de cada segmento enviado y en modificar el proceso de retransmisión rápida para que las pérdidas puedan ser detectadas más pronto. Además ante múltiples segmentos perdidos y más de un procedimiento de retransmisión rápida, la ventana de congestión solamente se reduce una vez. RCCMP ya introducía un cálculo más adecuado del tiempo de ida y vuelta entre la fuente y el representante que TCP Reno. Por otro lado, en el estudio de Hengartner [Hengar00] se evalúa por separado cada uno de los mecanismos que TCP Vegas implementa. Este trabajo concluye que aunque uno de los procesos que más importancia tiene en la mejora de la eficiencia, en condiciones de alta carga en la red, es debido a este nuevo mecanismo de retransmisión, el principal responsable de este aumento de rendimiento es que TCP Vegas reduce la ventana de transmisión es un factor menor que TCP Reno. Por lo que en el diseño de RVCMP no se consideró interesante incorporar los mecanismos que avanzan el instante para la retransmisión del paquete. Otros autores como [Ahn96] y [Bolliger99] también han trabajado y evaluado la versión de TCP Vegas que no incorpora estos mecanismos de retransmisión rápida.

Como puede observarse, las modificaciones del control de congestión sólo afectan a la fuente. En el representante se ejecuta el mismo algoritmo que se presentó en la sección 4.2.3 y que imita el comportamiento de un receptor TCP con confirmaciones retardadas.

8.3 EVALUACIÓN DEL PROTOCOLO RVCMP

El objetivo de este apartado es mostrar y evaluar el comportamiento de RVCMP, bajo la misma perspectiva que se utilizó para el protocolo RCCMP. Por ello, se van a estudiar los mismos aspectos que se analizaron en el capítulo 5, salvo el procedimiento para el cambio de representante que es el mismo que para RCCMP. Los puntos que se van a tratar son: el análisis del comportamiento de RVCMP cuando aumenta el tamaño del grupo y el retardo de ida y vuelta; el estudio de la equidad entre RVCMP y TCP; y el examen de la conducta del protocolo cuando varías instancias RVCMP comparten el mismo enlace. A estos aspectos se va a añadir el análisis de la escalabilidad del

protocolo, donde se contabilizará el número de paquetes de control y de datos en función del número de receptores para distintas probabilidades de error.

8.3.1 Tiempo de ida y vuelta entre la fuente y el representante

Es importante comprender los mecanismos que gobiernan el control de congestión del protocolo ante una topología sencilla. En esta sección se analiza el caudal y el tamaño de ventana cuando se incrementa el tamaño del grupo y el retardo de propagación de ida y vuelta entre la fuente y el representante.

La topología que se va a utilizar es la misma que se utilizó en el capítulo 5 para el estudio del protocolo RCCMP. La figura 5.7 muestra la configuración que se va a analizar. Todos los enlaces tienen una capacidad de 10Mbps. Los retardos de propagación son de 1ms en el caso del enlace que une la fuente con el primer nodo intermedio, de 10ms para el enlace entre los dos nodos intermedios, y (*identificación_receptor*+1) ms para los enlaces desde el nodo intermedio hasta los receptores. Así, para una topología con un receptor, el tiempo de ida y vuelta es de 24ms, mientras que para 20 receptores, el tiempo entre la fuente y el receptor más alejado es de 64ms.

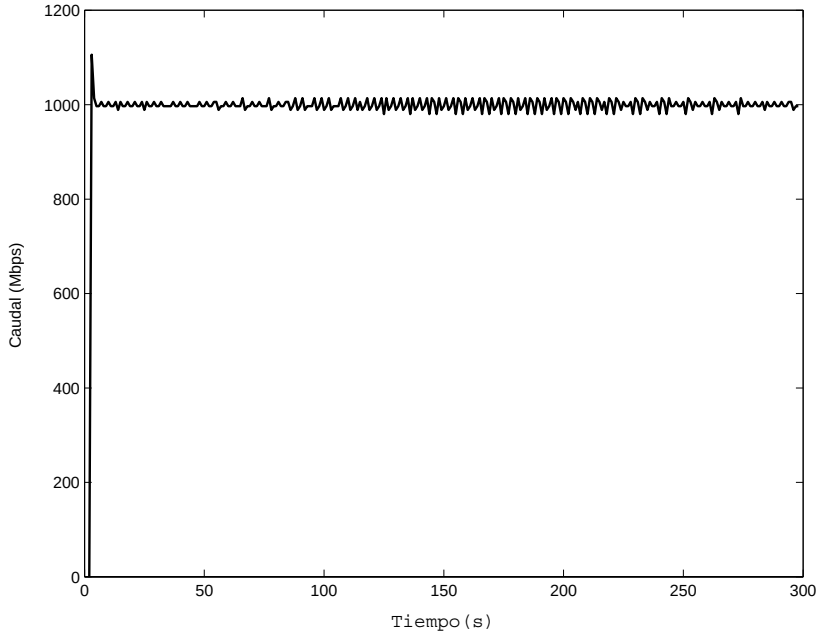

Figura 8.1 Caudal para una sesión RVCMP (α=γ=β=20) con un grupo de tamaño 1 y en un canal sin pérdidas.

Las gráficas 8.1, 8.2 y 8.3 muestran el caudal y el tamaño de ventana para un tamaño de grupo de uno y veinte receptores. En estos experimentos no se detectan pérdidas, por lo tanto no se generan NAKs, ni se producen cambios de representante. El control de congestión de RVCMP, a partir del cálculo de la tasa actual y la tasa estimada, consigue avanzarse a la congestión y adaptar suavemente la ventana de transmisión al ancho de banda disponible. Los caudales medios obtenidos son de

993.7Kbps y 993.2Kbps, mientras que los tamaños de ventana medio son de 22.12 y 26.32 paquetes para uno y veinte receptores respectivamente.

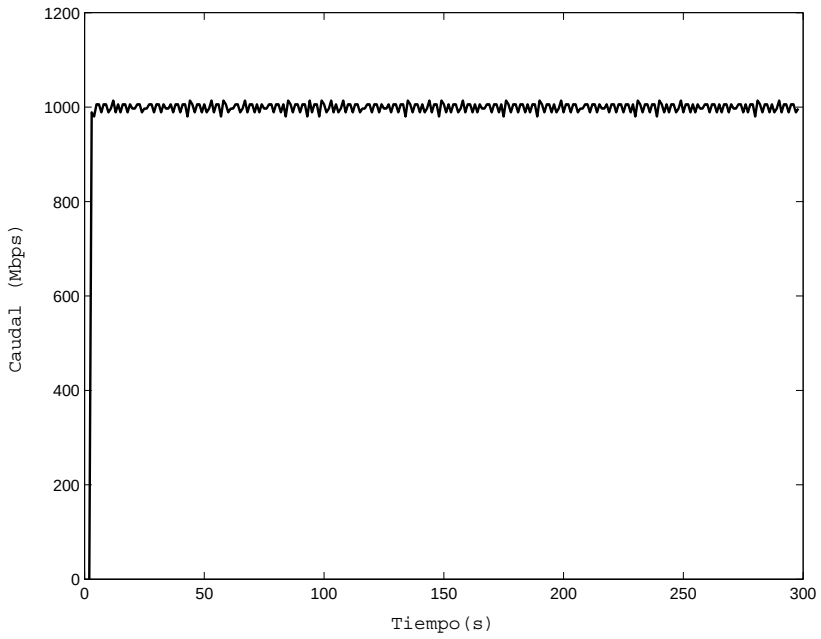

Figura 8.2 Caudal para una sesión RVCMP (α=γ=β=20) con un grupo de tamaño 20, sin pérdidas.

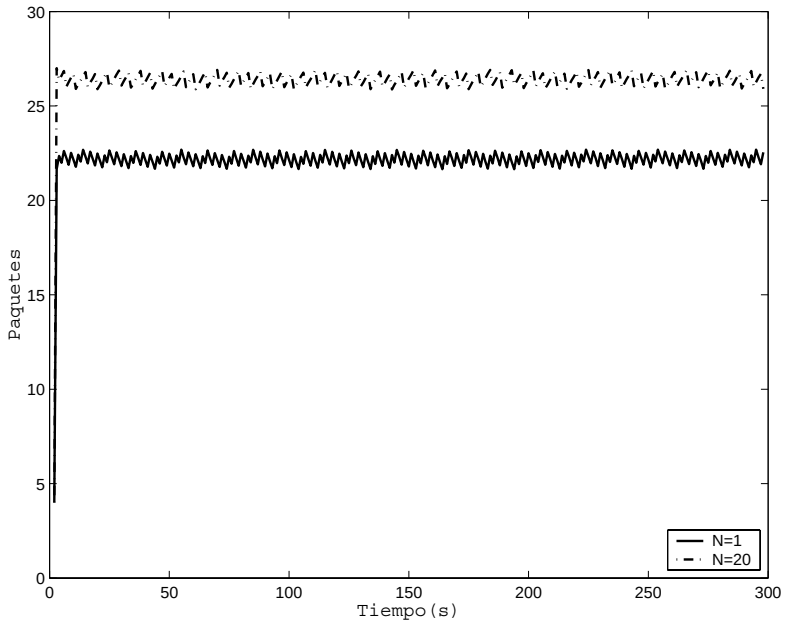

Figura 8.3 Ventana para RVCMP (α=γ=β=20) con 1 y 20 receptores, en un canal sin pérdidas.

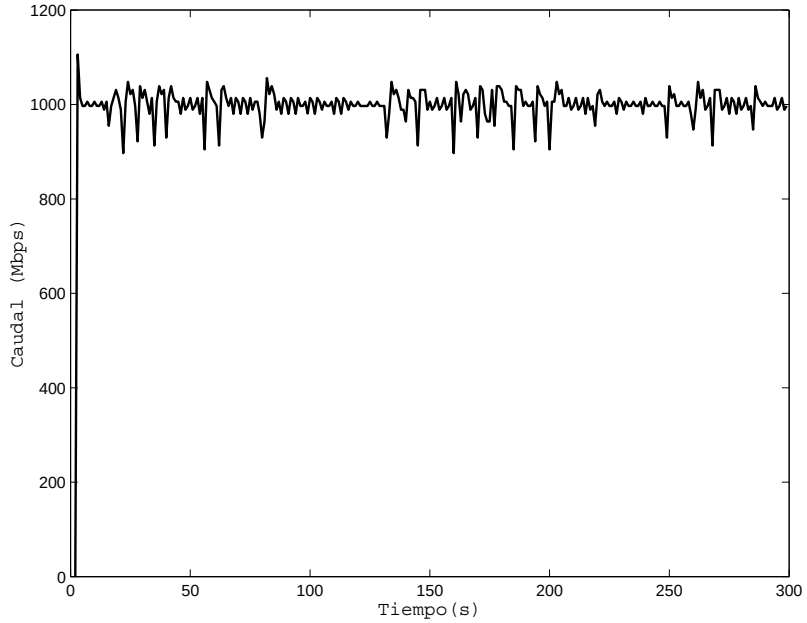

Figura 8.4 Caudal para una sesión RVCMP (α=γ=β=20) con un grupo de tamaño 1 y en un canal con p=0.001.

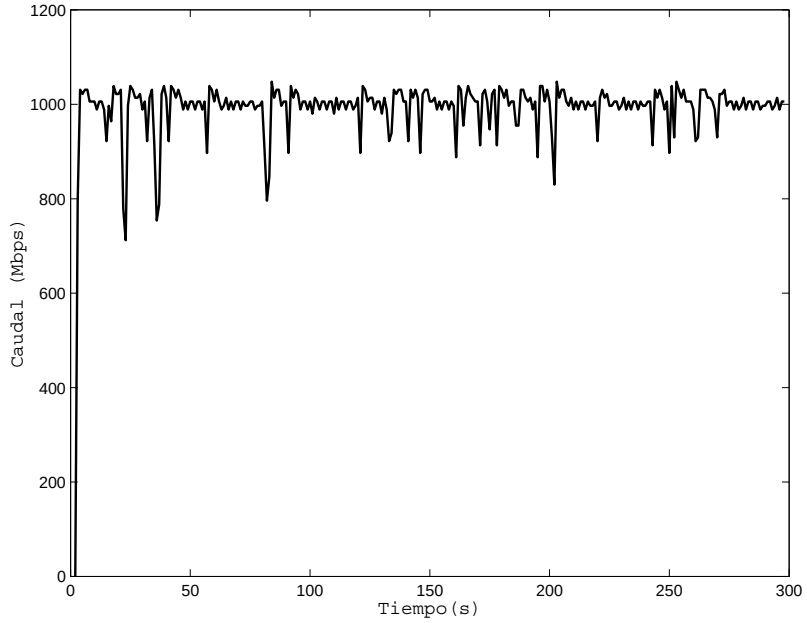

Figura 8.5 Caudal para una sesión RVCMP (α=γ=β=20) con un grupo de tamaño 20 y en un canal con p=0.001.

Los resultados mostrados en las figuras 8.4, 8.5 y 8.6 muestran el caudal y el tamaño de ventana en el caso de que se introduzcan pérdidas de canal con función de distribución uniforme a una tasa de 0.001 en el enlace que une los dos nodos intermedios. En este caso, los receptores detectan pérdidas que son observadas también por el representante. La fuente reconoce todas estas pérdidas a partir de la recepción de ACKs duplicados. En estos experimentos se han observado 35 y 37 indicaciones de

pérdida. Los caudales medios son de 997.2Kbps y 992Kbps, y los tamaños de ventana medios de 20.2 y 23 paquetes para uno y veinte receptores. En este caso, al igual que para RCCMP, al detectarse una pérdida la ventana de transmisión se reduce a la mitad.

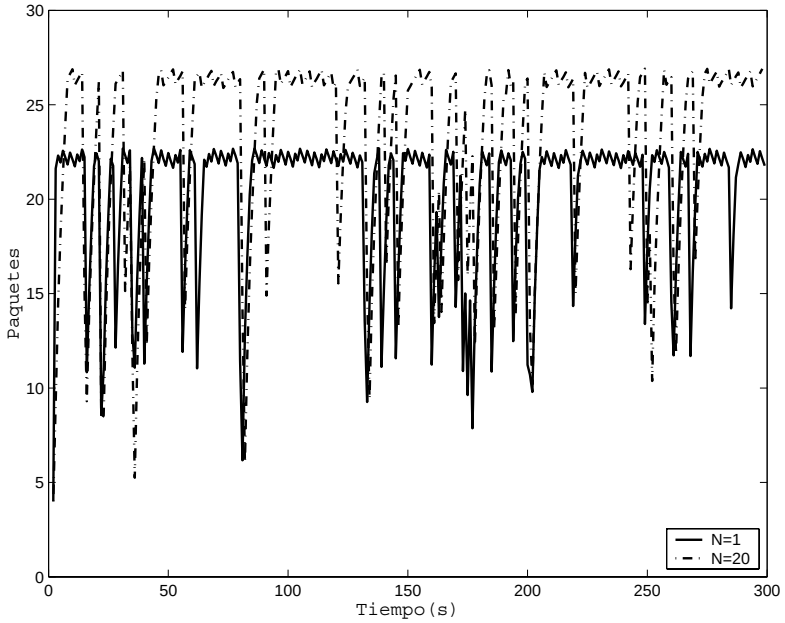

Figura 8.6 Tamaño de ventana para una sesión RVCMP (α=γ=β=20) con un grupo de tamaño 1 y 20 en un canal con p=0.001.

8.3.2 Comportamiento interprotocolo

En este apartado se va a evaluar el comportamiento de RVCMP frente a diferentes implementaciones de TCP. El estudio que se presenta está basado en el trabajo de Seada [Seada02], y sigue la misma estructura que se utilizó para evaluar la equidad del protocolo RCCMP en el apartado 5.5. Las métricas que miden el grado de equidad son el índice de equidad, *FI*, (5.1) y el factor de equidad, *f*, definido como el cociente entre el caudal TCP y el de RVCMP.

Se van a realizar los mismos experimentos y se van utilizar las mismas topologías que se utilizaron en el apartado 5.5. El primer experimento consiste en comprobar la equidad entre la implementación TCP Reno y RVCMP en un enlace que actúa como cuello de botella de 50ms y 500Kbps. La topología con la que se va a trabajar es la mostrada en la figura 5.21. Como RVCMP utiliza los umbrales α, γ y β para controlar la ventana de transmisión, se han realizado experimentos con distintos valores. En la figura 8.7 se presentan simultáneamente dos resultados que se corresponden con valores de α=γ=β=20 y α=γ=10 β=30. En estos experimentos solamente se consideran pérdidas por saturación de las colas. La gráfica muestra que dependiendo del ajuste de los umbrales el comportamiento de RVCMP es diferente. Para valores de α=γ=β=20, RVCMP muestra un comportamiento más cercano a TCP Reno. Aunque puede observarse, que TCP acapara más ancho de banda. Los índices de equidad son de 0.997

y 0.982, mientras que el factor de equidad es de 110.7% y 130.85% para α=γ=β=20 y α=γ=10 β=30.

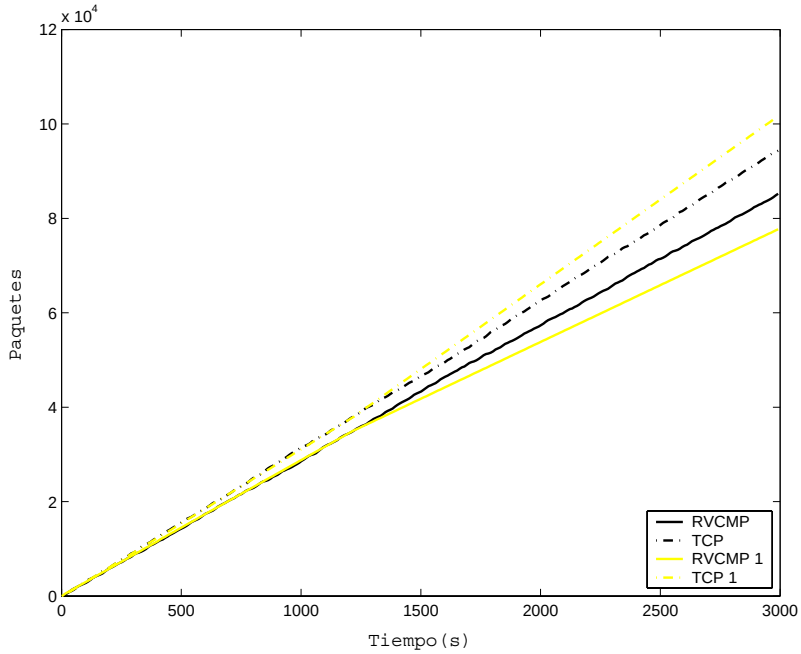

Figura 8.7 RVCMP (α=γ=β=20) y RVCMP1 (α=γ=10 β=30) versus TCP Reno en un canal sin pérdidas.

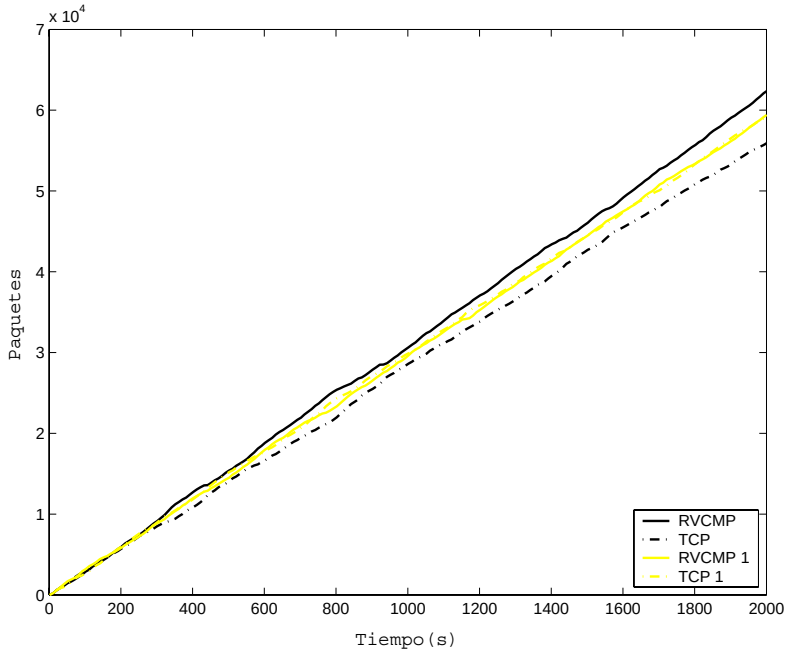

Figura 8.8 RVCMP (α=γ=β=20) y RCCMP1 (α=γ=10 β=30) versus TCP Reno en un enlace con 1% pérdidas de paquete.

Si se desplaza el inicio de las fuentes 0.5s se observa un mejor reparto del ancho de banda del enlace. Si se retarda 0.5s el inicio de RVCMP respecto a la conexión TCP, para α=γ=β=20 se obtiene un *FI* de 1 y un *f* de 102.99%. Si se retarda TCP, el valor del

índice de equidad resulta también en 1 y el factor de equidad en 103.52%. Estos resultados vuelven a mostrar como influye la disciplina de la cola en el justo reparto del ancho de banda entre las dos sesiones.

En este segundo experimento, se van a introducir, en el enlace etiquetado como congestionado, pérdidas de canal de función de distribución uniforme de un 1%. Los resultados se muestran en la figura 8.8. En este caso, RVCMP (α=γ=β=20) acapara más ancho de banda que la implementación TCP Reno. *FI* tiene un valor de 0.998 y *f* de 91.85%. El comportamiento predictivo de la ventana de congestión hace que el caudal se vea menos afectado por las pérdidas. Si se escogen unos valores de α=γ=10 β=30 se observa que el comportamiento de los dos protocolos se ajusta perfectamente. En este caso, los índices toman unos valores de *FI* 1 y *f* de 97.58%.

A continuación sobre la topología 5.21, se va a estudiar la equidad de RVCMP con TCP Vegas. Para ello, se han realizado múltiples experimentos con diferentes valores de umbrales. La tabla 8.1 muestra los resultados de los índices de equidad.

	RVCMP (α=γ=β=20)		RVCMP (α=γ=10 β=30)	
Protocolo	FI	f	FI	f
TCP (α=γ=1 β=3)	0.695	20.34%	0.804	33.93%
TCP (α=γ=β=3)	0.687	19.37%	0.915	53.32%
TCP (α=γ=3 β=5)	0.804	25.59%	0.864	43.21%

Tabla 8.1 Métricas de equidad para RVCMP (α=γ=β=20) y RVCMP1 (α=γ=10 β=30) versus TCP Vegas.

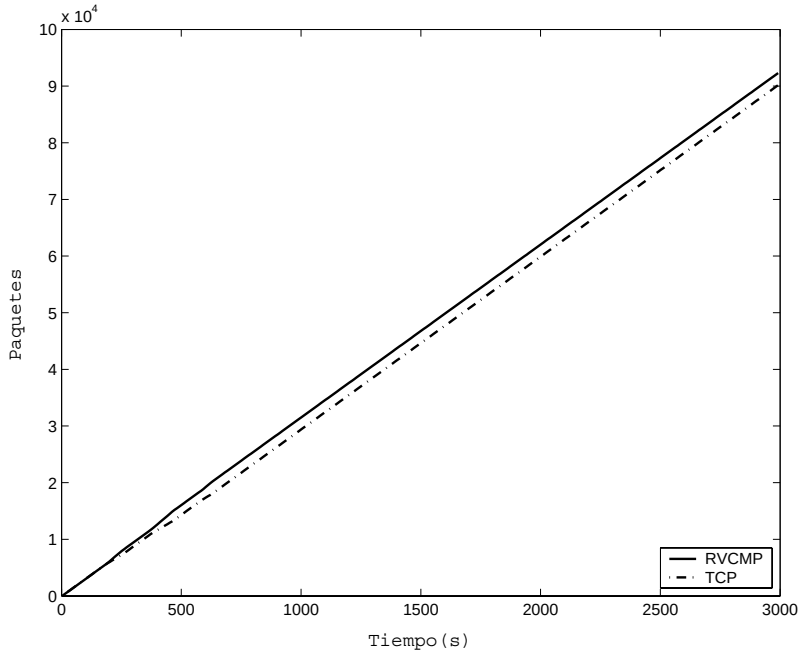

Figura 8.9 RVCMP (α=γ=β=4) versus TCP Vegas (α=γ=1 β=3).

Como puede observarse en la tabla, RVCMP no muestra un comportamiento equitativo con TCP Vegas. Esto es debido a que los umbrales de RVCMP han sido ajustados para que la conducta equilibrada se produzca con la implementación TCP Reno, que es la más usual en Internet. Por eso, cuando RVCMP comparte enlace con TCP Vegas acapara el ancho de banda. Si se modifican los umbrales de RVCMP para suavizar su comportamiento a un valor de $\alpha=\gamma=\beta=4$, se obtienen mejores resultados. Estos se presentan en la figura 8.9. En este caso, los índices toman un valor de *FI* de 1 y *f* de 97.67%.

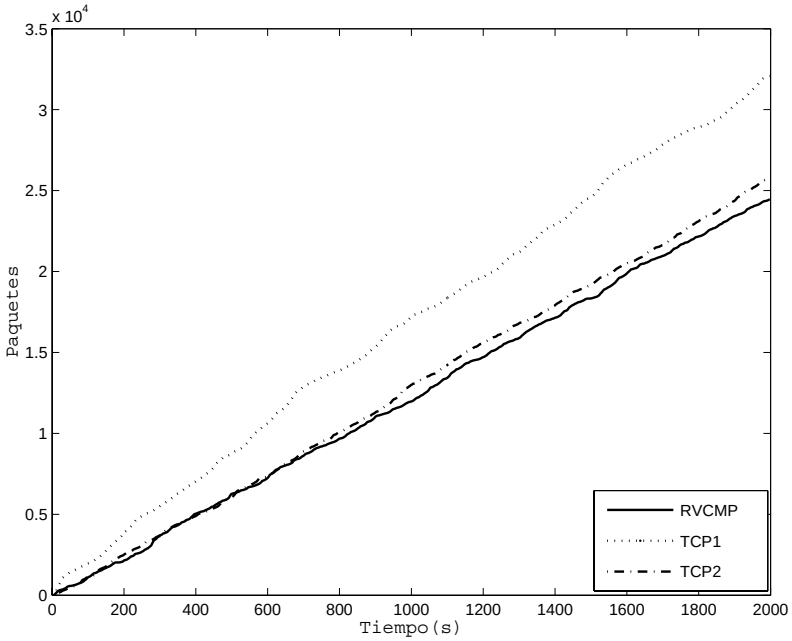

Figura 8.10 RVCMP ($\alpha=\gamma=\beta=20$) versus dos conexiones TCP Reno con tasas de error bajas.

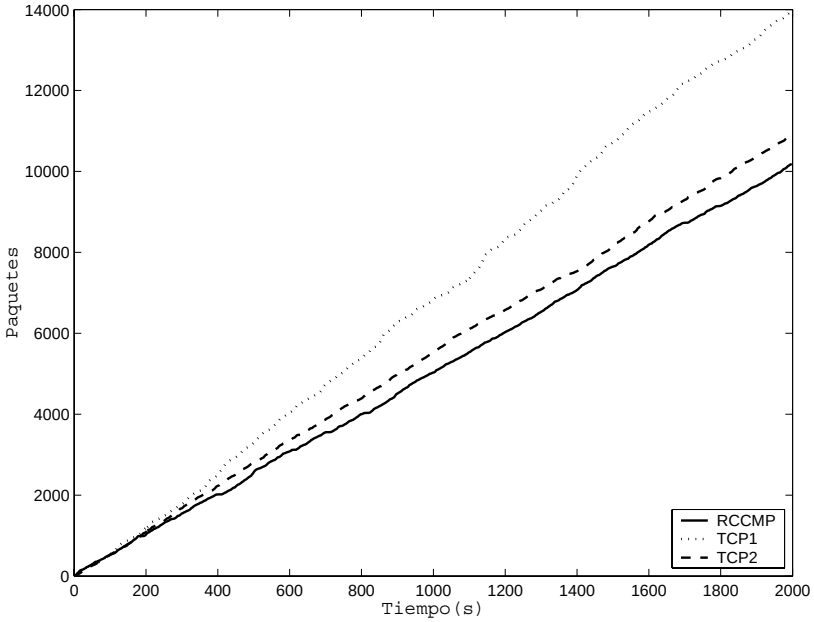

Figura 8.11 RVCMP ($\alpha=\gamma=\beta=20$) versus dos conexiones TCP Reno con tasas de error altas.

En el siguiente experimento se analizan los efectos al considerar receptores conectados a enlaces independientes con diferentes retardos y tasa de pérdidas. La topología que se considera es la mostrada en la figura 5.27. En este caso, una sesión RVCMP comparte enlaces con dos sesiones de la implementación TCP Reno. En la topología existe un enlace con alto retardo (400ms) y baja tasa de pérdidas (0.4%), y otro con bajo retardo (200ms) y alta tasa de pérdidas (1.6%). Los resultados pueden verse en la figura 8.10. En este caso, la conexión TCP que va por el camino de alto retardo y bajas perdidas acapara más ancho de banda. RVCMP se ajusta al receptor de menos recursos, RCCMP2, y muestra un comportamiento que coincide con la sesión de TCP más desfavorecida, que es la que selecciona el camino de bajo retardo y altas pérdidas. En este caso, el índice de equidad es de 0.987. Las indicaciones de pérdida son de 528 de las que 75.5 son detectadas mediante la expiración del temporizador. El porcentaje de tráfico reenviado es del 2.42%. El número de cambios de representante contabilizados es de 101.5.

Cuando se aumentan las tasas de error en los enlaces de la topología 5.27, el comportamiento que se obtiene es el mostrado en la figura 8.11. En este caso, la tasa de pérdidas del enlace etiquetado como de alto retardo y baja tasa es del 2%, y la del etiquetado como de bajo retardo y alta tasa es del 8%. Los resultados muestran, que otra vez la sesión TCP que va por el camino de mayor retardo y menores pérdidas, obtiene un mayor ancho de banda. RVCMP muestra la misma conducta que la sesión TCP más desfavorecida. En este caso, el índice de equidad es de 0.987. En este experimento, el emisor detecta 996.6 indicaciones de perdida de paquete, de los que 340.75 son detectadas por expiraciones del temporizador de retransmisión. Se producen 111.2 cambios de representante, y el porcentaje de paquetes retransmitidos es de 11.99%.

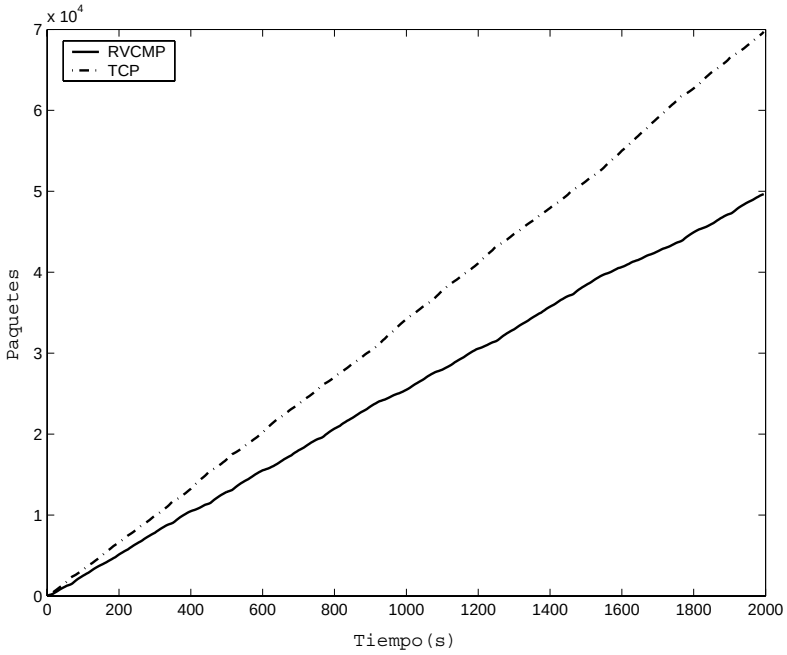

Figura 8.12 Secuencia de paquetes de TCP Reno y RVCMP (α=γ=β=20) con receptores con diferentes retardos.

La topología de la figura 5.30 estudia el comportamiento de RVCMP cuando existen en el grupo multipunto receptores con diferentes retardos. En este caso, el protocolo cuenta con dos receptores separados 52ms de la fuente, y un tercer receptor separado 102ms. El protocolo RVCMP selecciona como representante desde el inicio al receptor más alejado. La figura 8.12 muestra el comportamiento de TCP y RVCMP. Se observa que el protocolo multipunto es menos agresivo. RVCMP se ve muy afectado por el retardo. En este caso, los índices de equidad son *FI* de 0.973 y *f* 140.39%. Si se modifican los umbrales de RVCMP a α=γ=10 β=30 se obtienen peores resultados. Los índices son *FI* de 0.853 y *f* de 242.17%.

El siguiente experimento consiste en aumentar el retardo del enlace etiquetado como de alto retardo de la topología 5.30 a 200ms. En este caso, el protocolo desde la fase de inicialización escoge como representante al receptor más alejado. RVCMP (α=γ=β=20) muestra unos índices de equidad de *FI* 0.920 y *f* 184.04%, estos valores se pueden mejorar modificando los umbrales del protocolo a α=γ=10 β=30. En este caso, los resultados son *FI* 0.982 y *f* 131.73% (figura 8.13).

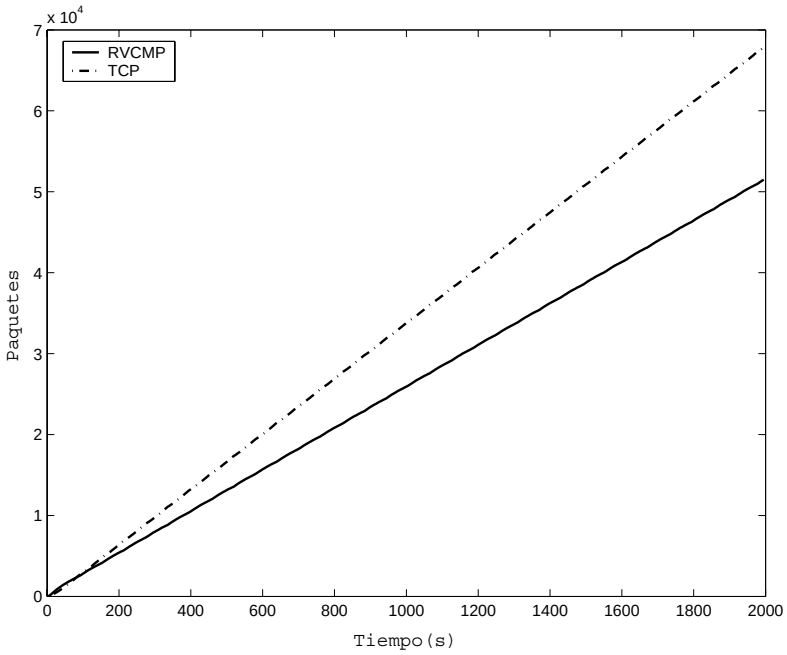

Figura 8.13 Secuencia de paquetes de TCP Reno y RVCMP (α=γ=10 β=30) con receptores con diferentes retardos.

Por último, se muestran los índices de equidad obtenidos cuando diferentes sesiones de la implementación TCP Reno y RVCMP (α=γ=β=20) comparten el enlace congestionado de la figura 5.21. En este escenario, el ancho de banda de este enlace se corresponde con la siguiente función $0.25 * N$, donde N es el número de conexiones RVCMP y TCP que atraviesan el enlace. Así, si el numero de flujos TCP y RVCMP suman cuatro, el ancho de banda del enlace congestionado tendrá una capacidad de 1Mbps. Se han considerado diferentes experimentos: en el primero una sesión RVCMP comparte el ancho de banda con distintas sesiones TCP; en el

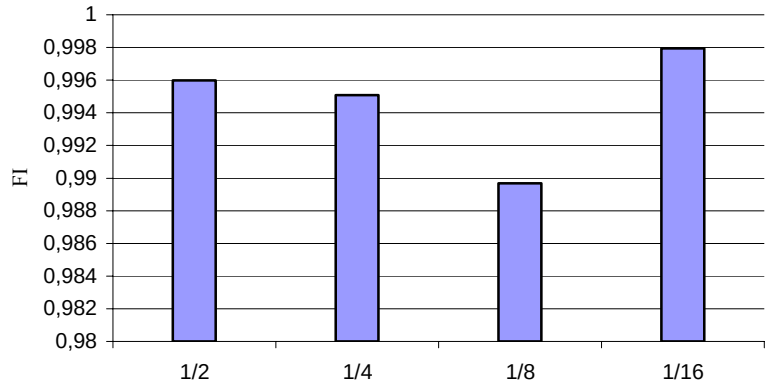

Figura 8.14 Índice de equidad para una sesión RVCMP que compite con diferentes sesiones TCP.

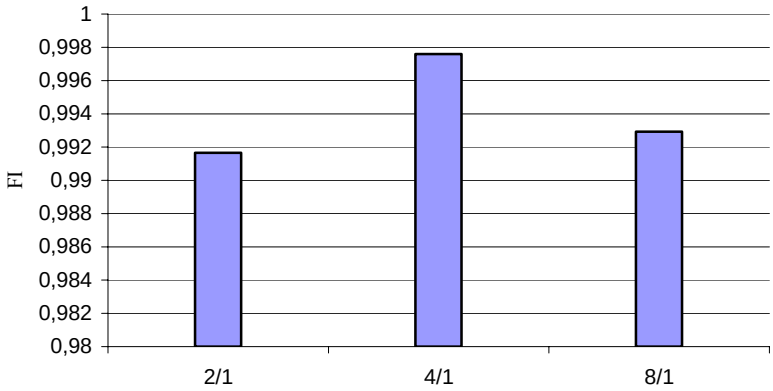

Figura 8.15 Índice de equidad para diferentes sesiones RVCMP que compite con una sesión TCP.

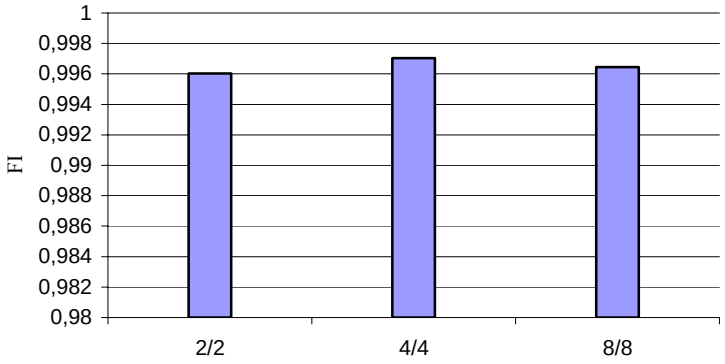

Figura 8.16 Índice de equidad para i sesiones RVCMP que compite con i sesiones TCP.

segundo una sesión TCP comparte el enlace con diferentes sesiones RVCMP, y por último, el mismo número de sesiones TCP y RVCMP compiten por el mismo ancho de banda. Los resultados se muestran en las gráficas 8.14, 8.15 y 8.16. Como puede observarse, el protocolo se comporta de forma equitativa ante cualquier número de conexiones TCP.

8.3.3 Comportamiento intraprotocolo

A continuación se muestra el estudio que evalúa como se comparte el ancho de banda de un enlace que es atravesado por distintas sesiones RVCMP. La topología que se ha utilizado es la de la figura 5.21. En este caso, todos los enlaces tienen una capacidad de 10Mbps, 10ms de retardo y una cola con capacidad para 30 paquetes. Se ha considerado un tamaño de grupo de tres receptores por sesión. Los experimentos consisten en aumentar el número de sesiones que comparten el enlace, que forma el cuello de botella, y observar como se reparte el ancho de banda. El indicador de la equidad será *FI*.

En la figura 8.17 se muestran los índices de equidad para tres experimentos: sin pérdidas, introduciendo pérdidas de canal de 0.001 en el enlace congestionado, y sin pérdidas, pero introduciendo un retardo entre el comienzo de una fuente y otra de 0.5s. Los resultados muestran que los recursos se reparten de forma equitativa entre las diferentes sesiones RVCMP. Si bien, se observa que a medida que aumenta el número de sesiones hay unas fuentes que acaparan más ancho de banda que otras.

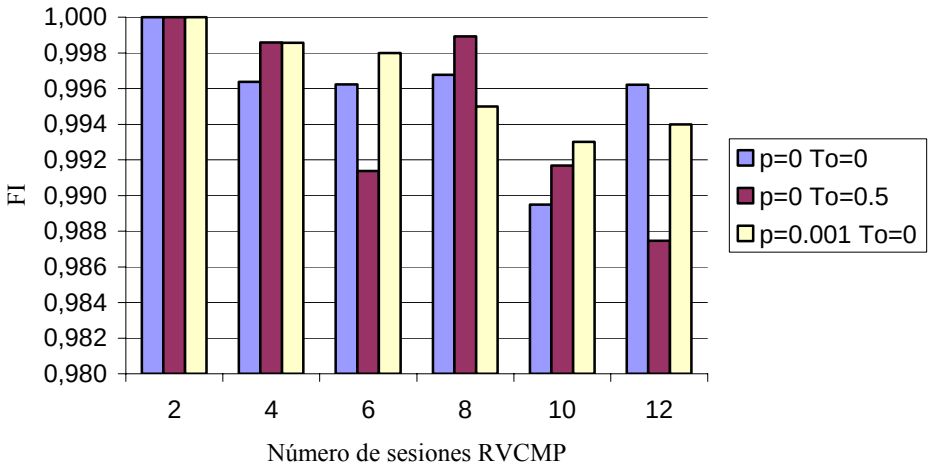

Figura 8.17 Índice de equidad para i sesiones RVCMP considerando enlaces sin pérdidas, con pérdidas y fuentes que comienzan en instantes distintos.

8.3.4 Escalabilidad

Para la evaluación de la escalabilidad, se muestran los resultados obtenidos mediante simulaciones, que presentan la cantidad y tipo de paquetes que una sesión

RVCMP genera en función de las pérdidas de canal. La topología utilizada es la misma que se utilizó para estudiar la carga del protocolo RCCMP, y que se corresponde con la figura 5.40. Los enlaces son todos iguales y se caracterizan por un retardo de 1ms y una capacidad de 0.5Mbps. La fuente, que se sitúa en la raíz del árbol de distribución, se corresponde con una aplicación que genera datos de forma continúa a una velocidad de 1Mbps. El número de receptores varía desde 3 hasta 900.

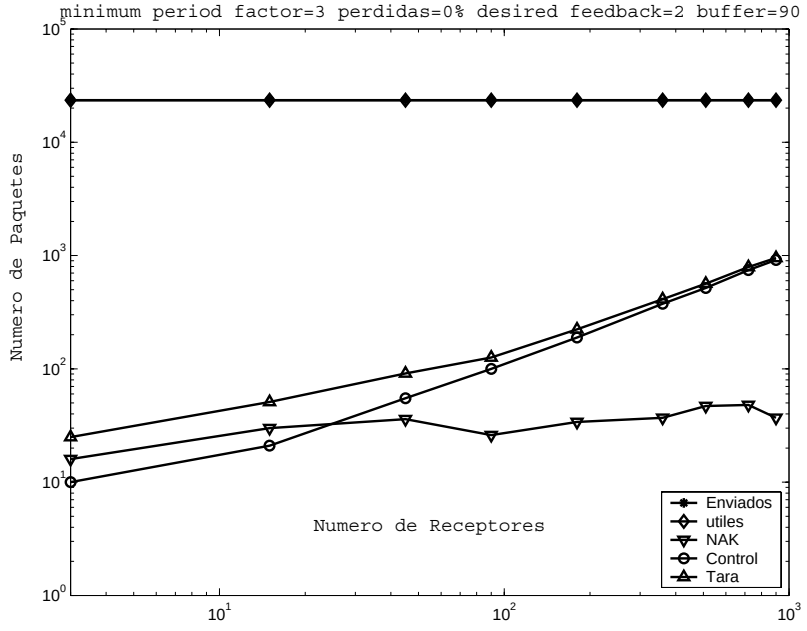

Figura 8.18 Número de paquetes de una sesión RVCMP(α=γ=β=20) sin pérdidas.

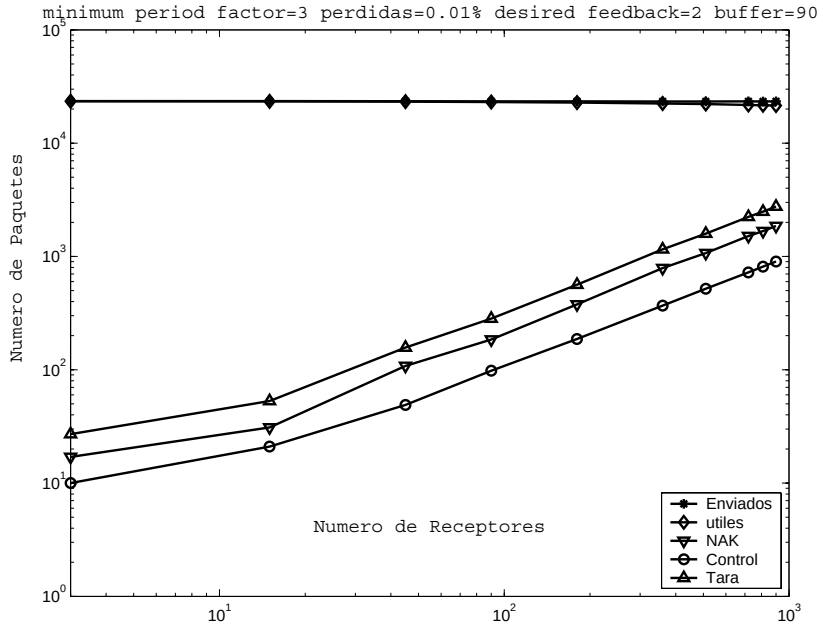

Figura 8.19 Número de paquetes de una sesión RVCMP(α=γ=β=20) con pérdidas de canal de 0.01%.

En la figura 8.18 se presenta el número y tipo de paquetes que intervienen en una sesión de 400s de duración cuando no se introducen pérdidas en el canal. En este caso, solamente se consideran pérdidas por saturación de las colas. Los paquetes que forman la tara son los de control, tanto los enviados por la fuente como por los receptores, y los NAKs. Si no se introducen pérdidas en el canal, el número de NAKs se mantiene controlado e independiente del número de miembros del grupo. Los paquetes de control, sin embargo, crecen en función del número de receptores, debido al proceso de inicialización, en el que todos los receptores deben responder al paquete de control PIC. En cuanto a los paquetes retransmitidos representan el 0.094% de los paquetes enviados.

Los resultados de la figura 8.19 muestra el número y tipo de paquetes al introducir una tasa de pérdidas de canal de 0.01% en los enlaces. En este caso, el número de NAKs crece respecto al escenario sin pérdidas, y se muestra dependiente del tamaño de grupo. La tara viene dominada por estos paquetes. El número de paquetes útiles decrece ligeramente para tamaños de grupo grandes. Si bien, el porcentaje de paquetes retransmitidos es en promedio del 3.972%, el rango va desde 0.135% hasta 8.111% para un tamaño de grupo de 3 y 900 receptores.

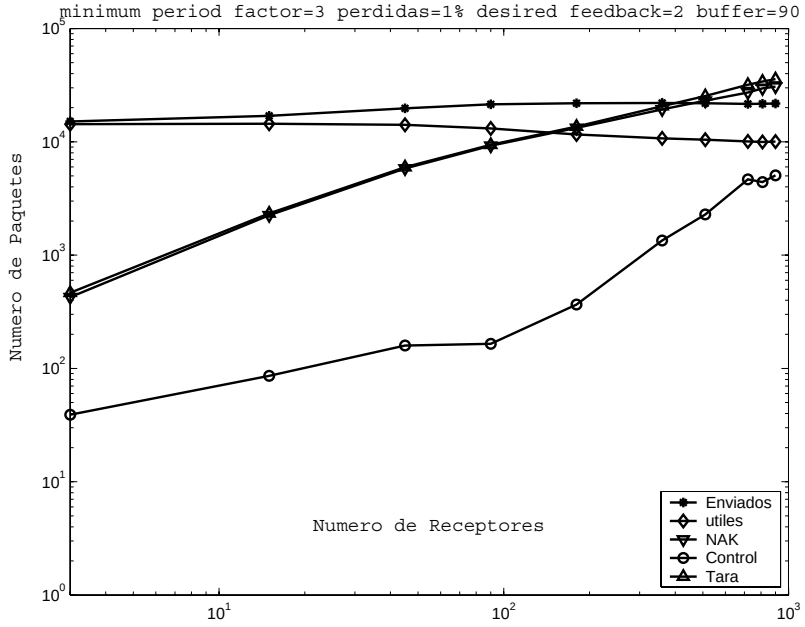

Figura 8.20 Número de paquetes de una sesión RVCMP(α=γ=β=20) con pérdidas de canal del 1%.

La gráfica 8.20 muestra el número de paquetes al introducir pérdidas de canal del 1% en los enlaces. En este caso, se observa que el número de NAKs crece drásticamente respecto a los escenarios anteriores. Sin embargo, si se considera el número de paquetes por receptor, se obtiene, haciendo el promedio para todos los tamaños de grupo, un valor de 80.50 paquetes/receptor que se corresponde con una tasa del 0.20 paquetes/s. El número de NAKs por receptor disminuye cuando aumenta el número de receptores, porque ocurren más cancelaciones. Así, por ejemplo, para un tamaño de grupo de 900 receptores se envían en total unos 30692 NAKs, que se corresponden con 34.1 paquetes/receptor (0.085 paquetes/s). En promedio, solamente el 43.08% de los NAKs generados son enviados, reduciéndose este valor para tamaños de grupo mayores a 500

receptores al 30% aproximadamente. Estos datos demuestran que el esquema de temporizadores funciona y mantiene el número de paquetes de solicitud de retransmisión acotados. En cuanto al porcentaje de paquetes retransmitidos es en media del 39.87% de los paquetes enviados. Para grupos de 3 receptores este valor es del 5.24%, mientras que para grupos de 900 receptores alcanza el 53.83%.

La topología que se está estudiando es posiblemente uno de los peores escenarios que se pueden encontrar. Todos los enlaces muestran pérdidas elevadas y no existe un receptor con peores recursos que los demás, por lo que no se puede seleccionar a un representante estable que controle la sesión. Los resultados muestran que para tamaños de grupos superiores a la centena de receptores el número de indicaciones de pérdida es de 11199 de las que 10822 ocurren en receptores que no son el representante. Esto lleva a que el número de cambios de representante sea en media de unos 881.7.

8.4 COMPARACIÓN DE RESULTADOS ENTRE RCCMP Y RVCMP

En este capítulo se ha propuesto un nuevo control de congestión multipunto basado en la implementación TCP Vegas. En el capítulo 4 se ha descrito el control de congestión del protocolo RCCMP basado en la implementación TCP Reno. Ambos se han evaluado por separado. Ahora es el momento de comparar los resultados. Para ello, se va a seguir la misma estructura que para su evaluación, se comenzará comparando su comportamiento respecto al tiempo de ida y vuelta, después se analizará la equidad interprotocolo e intraprotocolo, y por último, se estudiará la escalabilidad.

Se comienza comparando el comportamiento del control de congestión de RVCMP y RCCMP sobre una topología en la que todos los receptores cuelgan del mismo nodo intermedio, y no se considera tráfico de fondo (figura 5.7). En este escenario, RVCMP no detecta pérdidas y regula su ventana de transmisión en base a la estimación del ancho de banda disponible. Por otra parte, RCCMP detecta pérdidas que se producen por la saturación de las colas. Esto ocasiona la reducción de la ventana de transmisión. Esto supone que el caudal medio de RVCMP resulta en un 0.84% más que el de RCCMP en el caso de considerar un tamaño de grupo de un receptor. Si se considera un tamaño de grupo de veinte receptores se obtiene una mejora en el caudal del 0.83%. Al introducir pérdidas de canal con una tasa del 0.001 en el enlace, se observa que el número de indicaciones de pérdidas es de 35 y 37 para RVCMP con uno y veinte receptores, mientras que para RCCMP es de 64 y 55. Por lo tanto, el tráfico retransmitido representa para RVCMP el 0.098% y 0.105%, y para RCCMP el 0.16% y 0.14%.

En cuanto a la equidad con TCP, los dos protocolos multipunto presentan un buen comportamiento. RCCMP y RVCMP con $\alpha=\gamma=\beta=20$ presentan prácticamente los mismos índices de equidad bajo la topología de la figura 5.21. Sin embargo, si se modifican los parámetros de los umbrales de RVCMP a $\alpha=\gamma=10$ $\beta=30$, los resultados empeoran. Si se introducen pérdidas de canal bajas (0.001), RCCMP y RVCMP ($\alpha=\gamma=\beta=20$) vuelven a tener el mismo comportamiento equitativo respecto a TCP. En este caso, los protocolos multipunto consiguen algo más de ancho de banda que la sesión TCP. Esto es debido al mecanismo para la recuperación de un paquete perdido, que estos dos protocolos implementan, y que consiste en enviar nuevos paquetes a la red, si la ventana de transmisión lo permite, al recibir un ACK duplicado.

Si se compara la equidad de RCCMP y RVCMP con la implementación TCP Vegas, los resultados obtenidos son bastante dispares. RCCMP consigue un

comportamiento más equitativo que RVCMP. Aunque RCCMP acapara más ancho de banda que TCP Vegas, los índices de equidad para distintos umbrales son superiores al 0.97. Sin embargo, RVCMP (α=γ=β=20) tiene un comportamiento muy agresivo, acaparando prácticamente todo el ancho de banda y con valores máximos de *FI* de 0.8. Si se modifican los umbrales de RVCMP se consiguen mejores resultados. De hecho, si se ajustan estos umbrales a α=γ=β=4, el comportamiento de RVCMP y TCP Vegas resulta ser totalmente equitativo con un índice de equidad de 1.

Cuando se utilizan topologías en la que los receptores cuelgan de caminos independientes con diferentes características de retardo y pérdidas, como la mostrada en la figura 5.27, el comportamiento de RVCMP y RCCMP respecto a TCP coincide. La conexión TCP que atraviesa el enlace de mayor retardo siempre es la que consigue mayor ancho de banda, mientras que la sesión TCP que tiene en su camino el enlace de menor retardo y más pérdidas siempre resulta desfavorecida. La sesión multipunto, con dos receptores asociados, tiene un comportamiento que coincide con la sesión TCP más perjudicada. Como es lógico, la sesión multipunto es controlada la mayor parte del tiempo por el receptor más desfavorecido. Como los dos receptores detectan pérdidas se producen del orden de cientos de cambios de representante.

Se analizan los resultados de una configuración con receptores con retardo de propagación diferentes, como en la figura 5.30, donde una sesión TCP Reno y una multipunto comparten enlaces. En este escenario, el retardo entre la fuente y el representante es de 102ms ó de 302ms, que coincide con el retardo entre la fuente y el receptor de la conexión TCP. RCCMP comparte el ancho de banda de forma equitativa con TCP. El índice de equidad es de 0.999. Sin embargo, RVCMP muestra un comportamiento menos equitativo, TCP acapara la mayor parte del ancho de banda. Los índices de equidad son de 0.973 y 0.92 para los dos retardos. Si se modifican los umbrales se puede conseguir variar la conducta de RVCMP.

En cuanto a la escalabilidad, se van a comparar los resultados en dos escenarios: en el que sólo se consideran pérdidas por congestión, y cuando se introducen pérdidas de canal del 1% en todos los enlaces del árbol. Los valores que se muestran en las tablas 8.2 y 8.3 son el promedio de los resultados obtenidos en las simulaciones para tamaños de grupo de 3 a 900 receptores. Como puede observarse en las tablas, RVCMP presenta un mejor comportamiento en el caso de que las pérdidas se produzcan solo por congestión. En este caso, este protocolo detecta menos pérdidas, se dan menos cambios de representante y se genera menos tráfico retransmitido. Cuando se introducen pérdidas elevadas en los enlaces, el comportamiento de los protocolos tiende a igualarse. Esto es debido a que gana en protagonismo el mecanismo de recuperación mediante NAKs que es el mismo para ambos protocolos.

Tasa de pérdida en los enlaces = 0		
	RCCMP	RVCMP
Pq. enviados	23516	23531
Indicaciones de pérdidas	37.89	16.24
%Pq reenviados	0.223	0.094

Pq. reenviados por NAK	26.58	10.83
Cambios de representante	12.65	8.38
Caudal (Kbps)	495.058	495.465

Tabla 8.2 Resultados para RCCMP y RVCMP(α=γ=β=20) en un escenario en el que sólo se consideran pérdidas por congestión.

Tasa de pérdida en los enlaces = 1%		
	RCCMP	RVCMP
Pq. enviados	20501	20415
Indicaciones de pérdidas	8430.7	8398.4
%Pq reenviados	39.90%	39.87%
Pq. reenviados por NAK	8046.2	8016.5
Cambios de representante	728.5	748.7
Caudal (Kbps)	431.426	429.533

Tabla 8.3 Resultados para RCCMP y RVCMP(α=γ=β=20) en un escenario con pérdidas de canal del 1%.

8.5 CONCLUSIONES

El protocolo RCCMP incluye un control de congestión que intenta emular al máximo posible el de la implementación TCP Reno, que es la más utilizada en Internet, con el propósito de obtener unos buenos resultados en lo relativo a la equidad. Diversos estudios han comprobado que la que la implementación TCP Vegas consigue incrementar el caudal hasta el 70% en ciertos escenarios, por lo que se pensó en modificar convenientemente el control de congestión de RCCMP para asemejarlo al de TCP Vegas. Esta modificación resulta en un nuevo protocolo llamado RVCMP.

RVCMP adopta los cambios en lo relativo a la estimación del ancho de banda disponible para modificar el tamaño de la ventana, y a la fase transitoria *(slow-start);* pero no incluye los mecanismos de retransmisión rápida que propone TCP Vegas.

Los resultados muestran, que en los escenarios estudiados, el caudal de RVCMP supera el de RCCMP, pero en un pequeño porcentaje que no supera el 1%, reduciéndose también el tráfico de paquetes retransmitidos. En cuanto a la equidad con TCP, si se escogen los umbrales de RVCMP adecuadamente, se obtienen unos buenos resultados. La escalabilidad también está asegurada para este protocolo.

CAPÍTULO 9

CONCLUSIONES Y LINEAS FUTURAS

9.1 CONCLUSIONES

Las comunicaciones multipunto prometen ofrecer, tanto a usuarios como a proveedores, mayor eficiencia y nuevos servicios. Sin embargo, para poder desplegar adecuadamente este tipo de comunicaciones en las redes, es necesario contar con protocolos adecuados a todos los niveles. Esta tesis doctoral se ha centrado en las comunicaciones multipunto desde dos puntos de vista, el control de flujo de la categoría de servicio ABR en redes ATM, y los protocolos de transporte fiable en las redes IP, y concretamente, en las soluciones de control de congestión de tasa única.

La clase de servicio ABR (*Available Bit Rate*) reparte los recursos de forma equitativa entre las diferentes conexiones, garantizando una tasa mínima y minimizando la tasa de pérdidas de células. Todo ello, a partir de un control de flujo que periódicamente determina a qué velocidad debe transmitir la fuente. Este mecanismo fue diseñado para conexiones punto a punto, siendo necesario extenderlo para punto a multipunto. En este caso, los conmutadores tienen que copiar las células a cada una de las ramas de las que cuelgan miembros del grupo, y deben desarrollar mecanismos que limiten y agreguen el tráfico de realimentación. El primer algoritmo para la consolidación de células fue publicado por L. Roberts y está basado en el algoritmo EPCRA. En su trabajo propone mantener el comportamiento de los terminales fuente y destino, y modificar el comportamiento de los conmutadores para que sean capaces de agregar la información de realimentación que proviene de las ramas. Tras un análisis del algoritmo que propone, se observa que bajo ciertas condiciones no converge. Como contribución de este trabajo de tesis se ha propuesto un algoritmo de consolidación que determina el orden de llegada de las células, para asegurar que la agregación de la información de realimentación, se realiza de forma correcta. El mecanismo propuesto se ha validado mediante el análisis matemático desarrollado en el capítulo 3, que obtiene las expresiones para el caudal de la fuente y la ocupación de las colas en estado permanente. También, se han realizado simulaciones utilizando una modificación del simulador NIST ATM/HFC

Simulator Versión 3.0, al que se le ha añadido un nuevo componente: el conmutadores multipunto. Con este algoritmo se consigue mejorar la convergencia.

En cuanto a las comunicaciones multipunto en Internet, esta tesis ha propuesto y evaluado un protocolo de transporte punto a multipunto fiable, RCCMP, diseñado para ser un protocolo escalable, fiable y con un control de congestión de tasa única que comparta el ancho de banda equitativamente con TCP.

El control de congestión ha sido planteado como una parte esencial del protocolo, y no como ocurre en muchas propuestas, como un componente adicional que debe ser ajustado a un protocolo de transporte. En RCCMP se combinan los objetivos de regular la tasa de transmisión y conseguir una comunicación fiable, con el fin de simplificar y limitar el número de confirmaciones negativas que se envían desde los receptores. El control de congestión diseñado emula al máximo posible al de la implementación TCP Reno, para conseguir un comportamiento lo más parecido a él, y así, cumplir con el requisito indispensable de equidad.

La fiabilidad viene garantizada por la posibilidad de retransmitir los paquetes perdidos mediante confirmaciones negativas. Aunque, el uso de estos mensajes puede provocar situaciones de sobrecarga de NAKs, los temporizadores exponenciales y la limitación del número de retransmisiones en un intervalo de tiempo, aseguran la escalabilidad y fiabilidad del protocolo. Todo esto sin recurrir a la colaboración de los elementos intermedios de red, y manteniendo la complejidad del protocolo reducida.

Para la evaluación de las prestaciones de RCCMP, se ha implementado el protocolo en el simulador ns-2. Los experimentos presentados muestran que RCCMP funciona razonablemente bien en cuanto a dos aspectos fundamentales como la equidad y la escalabilidad.

El protocolo propuesto cuenta con una serie de parámetros, que deben ser ajustados por el administrador, para conseguir que el comportamiento se adecúe convenientemente al escenario donde se va a producir la comunicación. Sin embargo, se ofrece un conjunto de valores para estas variables, que cubren un extenso rango de trabajo. Las pruebas realizadas muestran que existe un compromiso entre la dinámica del protocolo y la escalabilidad.

El control de congestión ajusta la tasa de la fuente a la capacidad del receptor más lento. Éste hay que elegirlo mediante el proceso de selección de representante. Los resultados muestran que el caudal sigue los cambios que se producen en el grupo. La velocidad con la que el protocolo se adapta a las nuevas condiciones, depende por un lado del tiempo que tarda la fuente en conocer que existe un receptor más lento, y por otro de la dinámica de la ventana de transmisión.

Este protocolo ha sido diseñado con el requerimiento de equidad, por lo que los resultados muestran unas buenas presentaciones con diversos valores de ancho de banda, retardo y pérdidas. La equidad entre sesiones RCCMP también ha sido evaluada, y las pruebas presentan unos buenos índices.

En cuanto a la escalabilidad, la información de realimentación tanto para controlar la tasa del grupo como la posibilidad de congestión, proviene únicamente del representante, por lo que es perfectamente escalable. Además, el protocolo incluye un mecanismo de supresión de la información de realimentación, que evita una respuesta masiva por parte de los receptores. Con este fin, se han incorporado los temporizadores exponenciales. Éstos son exclusivamente locales a los receptores, y funcionan en colaboración con el control de congestión. Los resultados muestran que el factor de supresión de NAKs crece en función del número de receptores, y que en escenarios con fuertes pérdidas, el número de NAK cancelados supera el 70%. Por lo que RCCMP es un protocolo adecuado para trabajar en entornos de grupos de tamaño de cientos de miles de usuarios.

Otra contribución de esta tesis ha sido la obtención de una expresión analítica para modelar el caudal del protocolo de transporte multipunto RCCMP, en función de la tasa de pérdidas, y del tiempo de ida y vuelta entre la fuente y el representante. El trabajo desarrollado por J. Padhye para punto a punto, se ha extendido a punto multipunto. Las dos principales causas de la modificación del caudal en un entorno multipunto, y que difieren respecto al escenario punto a punto, son las siguientes:

- La detección de pérdidas en cualquier receptor, generándose paquetes retransmitidos fuera del control de flujo, que han de ser tenidos en cuenta en el modelo del caudal.
- La aparición de un receptor de peores recursos en el grupo multipunto provoca un cambio de representante, y también una modificación en la tasa.

Estos dos procesos han sido analizados detalladamente, y se han englobado en el modelo.

Uno de los factores que afectan a la escalabilidad de un protocolo multipunto es la cantidad de paquetes de control transmitidos por la fuente y los receptores para asegurar la fiabilidad, controlar la tasa de la fuente, estimar el número de miembros del grupo, etc. Para determinar el nivel de escalabilidad de un protocolo es importante contabilizar el ancho de banda que consume. Otra de las contribuciones de esta tesis doctoral es el desarrollo de un método de análisis para hallar el ancho de banda consumido por un protocolo de transporte punto a punto o multipunto. Las expresiones obtenidas se han validado a partir de los resultados obtenidos mediante simulación.

Finalmente, se ha diseñado y simulado otro protocolo de transporte punto a multipunto fiable de tasa única, RVCMP, que incluye un control de congestión que emula al de la implementación TCP Vegas. El protocolo adopta los cambios en lo relativo a la estimación del ancho de banda disponible para modificar el tamaño de la ventana, y a la fase transitoria (*slow-start*). Los resultados muestran que se consigue aumentar el caudal de RVCMP respecto a RCCMP en un pequeño porcentaje, reduciéndose también el tráfico de paquetes retransmitidos. En cuanto a la equidad con TCP, si se escogen los umbrales de RVCMP adecuadamente, se obtienen unos buenos resultados.

9.2 LÍNEAS FUTURAS

En el área de investigación de las comunicaciones multipunto todavía quedan campos por explorar. Además de los temas que quedan fuera del propósito de esta tesis, el estudio de los puntos abordados ha dado lugar a diferentes aspectos susceptibles de ser tratados con más profundidad. A continuación, se enumeran las líneas de trabajo más relevantes, comenzando por aquellas que son una consecuencia más directa del trabajo realizado:

- Aunque se han programado en el simulador ns-2 los dos protocolos de transporte multipunto fiable, queda pendiente su implementación real, para poder realizar pruebas sobre entornos de redes locales o de Internet, y así, verificar los resultados obtenidos mediante simulación y analíticamente.

- Para ajustar los temporizadores exponenciales, imprescindibles para asegurar la escalabilidad de los protocolos propuestos, es necesario conocer de manera aproximada el número de receptores activos en la sesión. Aunque se ha adoptado la solución de [Friedman99], adecuándola a RCCMP, esta propuesta requiere de un conocimiento parcial del grupo multipunto, y genera un intercambio de paquetes de control entre la fuente y los receptores. Una posible línea futura es el estudio en más profundidad de estos mecanismos de estimación del número de receptores, para reducir la carga que implican.

- La implementación TCP SACK ha sido desarrollada para mejorar el rendimiento cuando se producen múltiples pérdidas de segmentos en la misma ventana. Este control de congestión puede ser adaptado a RCCMP para comprobar su comportamiento en punto a multipunto. Esta propuesta ya está siendo abordada, y los resultados muestran, que en entornos con pérdidas de este tipo, se consiguen mejores resultados.

- En cuanto al modelo desarrollado del caudal del protocolo de transporte multipunto RCCMP en función de la tasa de pérdidas y del tiempo de ida y vuelta, queda por completar el modelo incluyendo las pérdidas detectadas por la expiración del temporizador. El análisis planteado incorpora las pérdidas detectadas mediante la duplicación de los ACKs y el cambio de representante.

- Las últimas propuestas de controles de congestión para entornos multipunto [Jiang03], [Kwon03] y [Kwon04] proponen mecanismos híbridos que conjugan las técnicas multitasa y unitasa. La fuente divide la información en capas, donde la tasa de cada grupo multipunto, que está asociada a una capa, se ajusta a través de un algoritmo unitasa. La sencillez de RCCMP hacen que este protocolo resulte adecuado para gobernar la velocidad de cada uno de estos grupos, quedando pendiente un algoritmo transversal que determine los límites entre capas.

- Se ha demostrado que las comunicaciones multipunto son más eficientes que las punto a punto para transmitir información hacia muchos receptores. Aunque, la tesis doctoral se ha centrado en las redes cableadas, los protocolos propuestos pueden ser extendidos y adaptados a otros entornos como el *Universal Mobile Telecommunication System* (UMTS). En este entorno móvil, se introducen fenómenos que no aparecían en las redes IP o ATM, como los traspasos y la jerarquía de la red, que deben ser tenidos en cuenta.

APÉNDICE A

MODELADO DEL CONTROL DE CONGESTIÓN MULTIPUNTO SOBRE LA CLASE DE SERVICIO ABR

A.1 INTRODUCCIÓN

Debido al gran desarrollo de aplicaciones multipunto se hace necesario el estudio de mecanismos de congestión que eviten la saturación de la red. El desarrollo de estos controles de congestión es mucho más complejo en comunicaciones multipunto que en punto a punto. En conexiones multipunto, los distintos receptores están conectados a la fuente a través de enlaces con capacidades distintas, lo que dificulta determinar la tasa a la que debe transmitir la fuente. Existen diferentes políticas para dar respuesta a este problema, pero la más común, es adaptar la tasa de la fuente al receptor de menor capacidad para garantizar una tasa de pérdida de células mínima.

El control de flujo de la clase de servicio ABR *(Available Bit Rate)* para punto a multipunto se basa en establecer un mecanismo de realimentación que informe del estado de congestión de los receptores y elementos de la red. En este caso, si la realimentación es demasiado lenta no será útil para conocer el estado de la red en un instante determinado, pero si es demasiado frecuente puede aparecer el problema de implosión de información de realimentación.

En este apéndice se va a modelar el estudio de un control de congestión para comunicaciones punto a multipunto. El mecanismo se basa en que un nodo intermedio reconoce congestión cuando detecta que la longitud de alguna de las colas de los enlaces, que pertenecen a la conexión multipunto, alcanza el umbral alto (Q_H). En este instante, el nodo congestionado informa a la fuente mediante una señal de control para que pare de transmitir. Mientras la fuente no envía tráfico a la red, las longitudes de las colas disminuyen hasta alcanzar el umbral bajo (Q_L). Cuando todas las colas tengan una longitud por debajo de Q_L, el nodo informa a la fuente que puede volver a transmitir mediante una señal de control de activación. Este control de congestión podría perfectamente implementarse en los conmutadores ATM para ofrecer un servicio multipunto en la clase de servicio ABR.

En la literatura podemos encontrar diversos estudios para el análisis de mecanismos de control de congestión reactivos para comunicaciones punto a punto. En [Wang91] y [Kato97] se analiza un control de congestión basado en dos umbrales a partir de un complejo análisis matemático.

En este trabajo se va a caracterizar en valores medios y mediante el modelo de fluidos este mecanismo de control de flujo basado en umbrales. Se determinará la probabilidad de pérdida de paquetes del sistema, el caudal y otros parámetros que caracterizan el control de congestión [Solera01]. Se estudiará cuales son los valores de los umbrales Q_H y Q_L más adecuados. Para ello, primero se describirá el modelo de control de congestión utilizado para el análisis; segundo se determinarán los parámetros que caracterizan al sistema en valores medios, y se evaluarán las expresiones obtenidas; seguidamente, se analizará el sistema mediante el modelo de fluidos y se estudiará su validez.

A.2 EL MODELO

El modelo que se va a considerar para caracterizar el sistema de control de congestión se presenta en la figura A.1. Está compuesto por una fuente y un nodo intermedio con N colas de longitud finita e igual a B paquetes. Las colas se sirven a diferentes velocidades dependiendo de la capacidad del enlace de salida. Los paquetes llegarán al nodo después de un tiempo igual al tiempo de transmisión más el tiempo de propagación. Al llegar al nodo, los paquetes se duplicarán y se encolarán, si los servidores de salida están ocupados.

Para simplificar el sistema se va a considerar que el control de congestión está gobernado por el receptor de menos recursos. El conmutador envía la señal de control de inactivación, cuando la longitud de la cola del enlace más lento alcanza el umbral $Q_{H,}$ y envía la señal de control de activación, cuando la longitud de la cola del enlace de menor capacidad alcanza el umbral Q_L. De esta manera, se minimiza la probabilidad de pérdidas. Se va a considerar que la fuente genera tráfico a una tasa media de V paquetes/s, y que el enlace de capacidad V*C paquetes/s, con C<1, representa al enlace de menos recursos involucrado en la conexión punto a multipunto. D representa el tiempo de ida y vuelta entre la fuente y el nodo.

Para simplificar el análisis, se va a suponer que $(Q_H - Q_L)/VC > D$. Esta hipótesis es razonable cuando la carga de tráfico es alta, lo que coincide con la región de interés para el estudio de la congestión. Con esta condición, el comportamiento de la ocupación de la cola queda ilustrado en la figura A.2. El sistema pasa cíclicamente por cuatro fases. Durante los periodos I y IV, la fuente está inactiva, y por lo tanto, la longitud de la cola disminuye. Cuando alcanza el umbral $Q_{L,}$ se envía una señal de control para la activación de la fuente. En el instante $X_{sig\text{-}on}$ el sistema pasa de la fase IV a la fase I. Durante un tiempo, D, continúa disminuyendo la longitud de la cola hasta la llegada del primer paquete generado por la fuente, momento $X_{lleg\text{-}on}$, en el que el sistema pasa de la fase I a la fase II. En los periodos II y III, la fuente genera tráfico. Cuando la cola alcanza el umbral Q_H, se envía la señal de control de inactivación a la fuente, instante $X_{sig\text{-}off}$, que coincide con el paso de la fase II a la fase III; todavía, durante un tiempo D y hasta el instante $X_{lleg\text{-}off}$ se reciben paquetes.

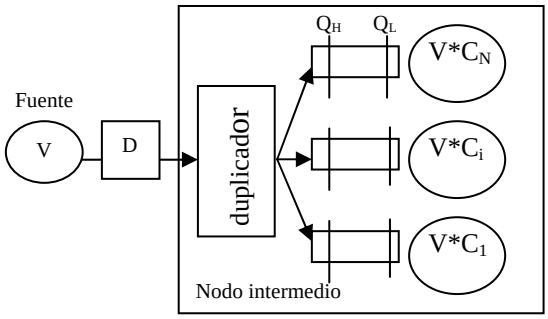

Figura A.1. Modelo analítico

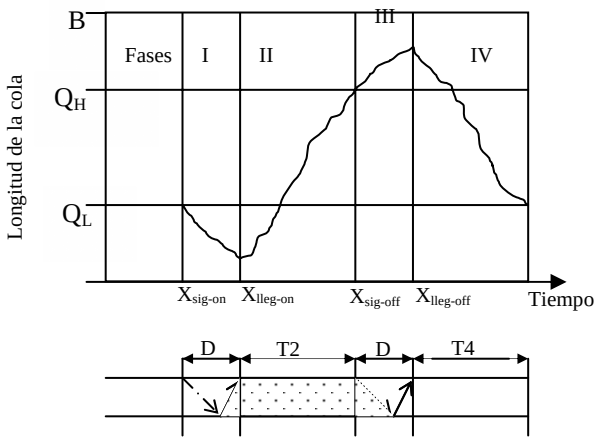

Figura A.2. Comportamiento de la cola del nodo intermedio

A.3 ANÁLISIS EN VALORES MEDIOS DEL SISTEMA

De la definición del sistema, se obtiene que el número medio de paquetes al inicio de la fase II es

$$Q_{sig\_on} = Q_L \tag{A.1}$$

$$Q_{lleg\_on} = \max(Q_L - VCD, 0) \tag{A.2}$$

Por lo tanto, la duración del tiempo de actividad, T2, queda determinado cuando el número de paquetes llega a Q_H

$$T2 = \frac{Q_H - \max(Q_L - VCD, 0)}{V(1-C)} \tag{A.3}$$

La fase III tiene una duración de D. En este periodo es donde pueden producirse pérdidas. En valores medios, la probabilidad de pérdidas puede calcularse como

$$P_B = \frac{\max(Q_H + DV(1-C) - B, 0)}{V(T2 + D)} \tag{A.4}$$

Al comienzo del periodo IV, en el instante $X_{lleg\text{-}off}$, se tiene un número medio de paquetes igual a,

$$Q_{lleg\_off} = Q_H + (V(1-P_B)-VC)D \quad \text{(A.5)}$$

por lo que la duración del periodo T4 es

$$T4 = \frac{Q_H + (V(1-P_B)-VC)D - Q_L}{VC} \quad \text{(A.6)}$$

La expresión del caudal se obtiene a partir de la duración media del ciclo T2+T4+2D y de la probabilidad de pérdidas (A.4):

$$T(H,L) = \frac{(T2+D)V(1-P_B)}{T4+T2+2D} \quad \text{(A.7)}$$

A.4 EVALUACIÓN DE LAS EXPRESIONES

A partir de las expresiones obtenidas en el apartado anterior, se va a analizar la probabilidad de pérdida y el caudal en función del retardo de propagación. Además, se va a estudiar cómo escoger los valores de los umbrales Q_H y Q_L, para minimizar la probabilidad de pérdida y la frecuencia de envío de señales de control.

A.4.1 Impacto del retardo sobre la probabilidad de pérdidas

Con B, Q_H, Q_L y C fijados, la figura A.3 muestra el efecto del tiempo de ida y vuelta, D, en la probabilidad de pérdidas, P_B, para distintos valores de carga. Como se podía preveer, un aumento del tiempo de propagación supone un aumento en la duración del periodo III, única fase del sistema donde se pueden producir pérdidas. Por lo tanto, el sistema está trabajando en los límites de la cola durante más tiempo. Desde otro punto de vista, suponiendo fija la probabilidad de pérdidas a la que el sistema debe trabajar, se observa que el sistema puede cursar más tráfico si el tiempo de propagación es mayor.

A.4.2 Elección de los umbrales

Una elección adecuada de los valores umbrales, Q_L y Q_H, es importante para maximizar el caudal, minimizar las pérdidas y reducir el envío de señales de control. En la figura A.4 se observa como el valor del caudal se mantiene constante hasta un determinado tiempo D en el que cae drásticamente. Esto es debido a que el umbral Q_L está por debajo del valor VCD. En está situación, la cola permanece vacía durante algún tiempo en la fase I, lo que hace disminuir el caudal. Por otro lado, un valor del umbral Q_L mayor que VCD no aumenta el caudal, y sin embargo, es posible que genere el envio de más señales de control. Además, puede inducir a mayores pérdidas en el sistema, si esto nos lleva a aumentar el valor del umbral Q_H.

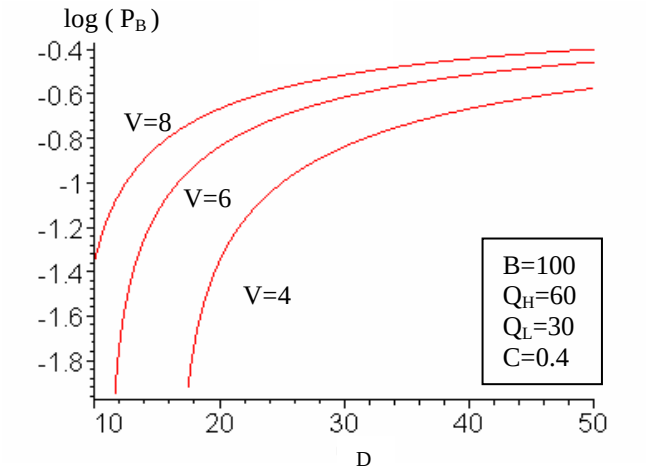

Figura A.3. Probabilidad de pérdida en función del tiempo de propagación para varios valores de carga.

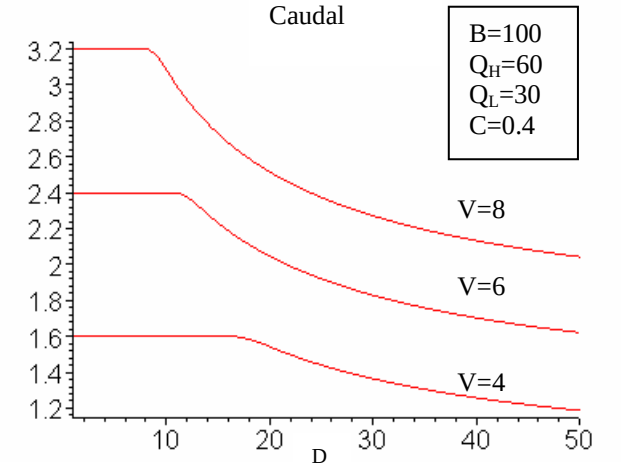

Figura A.4. Efecto en el caudal del umbral Q_L< VCD.

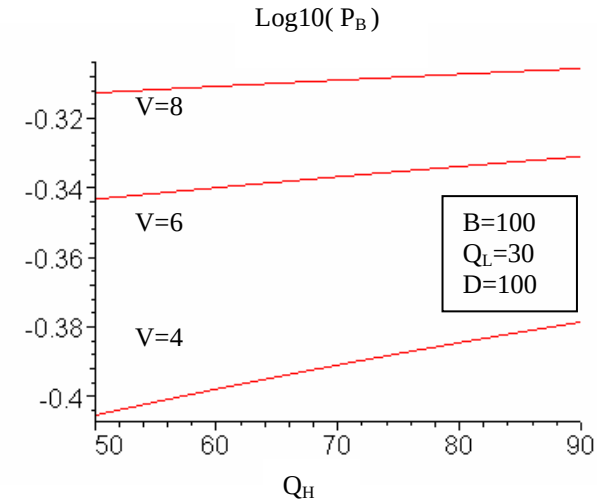

Figura A.5. Probabilidad de pérdida en función del valor del umbral Q_H.

La elección del umbral Q_H deberá ser lo más bajo posible para minimizar la probabilidad de pérdidas, llegando al compromiso entre la frecuencia de envío de señales de control, que vendrá dado por el tiempo de ciclo del sistema y las pérdidas. En las gráficas A.5 y A.6 pueden observarse estos comportamientos.

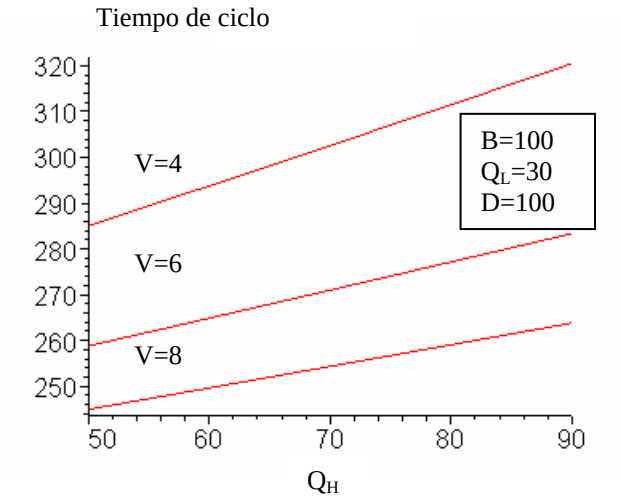

Figura A.6. Tiempo medio de ciclo en función del umbral alto.

A.5 ANÁLISIS MEDIANTE EL MODELO DE FLUIDOS

Otra manera de describir el mecanismo de control de congestión es caracterizarlo como una fuente de dos estados, *on-off*, con un periodo de actividad T2+D, en el que la fuente genera V paquetes/s a tasa constante, y un periodo de inactividad de duración T4+D. Se supondrá que el tiempo de permanencia en los estados activo, (estado 1), e inactivo (estado 0), sigue una distribución exponencial. Esto nos lleva a un modelo de nacimiento y muerte de dos estados. En la figura A.7 está representado el modelo, donde λ y α son las tasas de transición, y se calculan como la inversa del tiempo medio de permanencia en el estado.

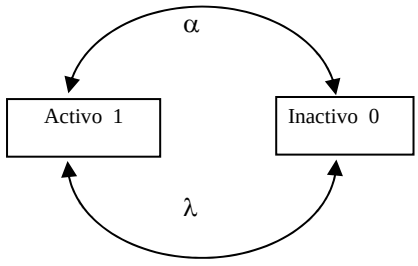

Figura A.7. Modelo de dos estados: activo e inactivo.

$$\lambda = \frac{1}{T4 + D} \tag{A.8}$$

$$\alpha = \frac{1}{T2 + D} \tag{A.9}$$

Πo y Π_1 representan la probabilidad de que el sistema se encuentre en el estado 0 y 1 respectivamente, y se escriben como:

$$\Pi_o = \frac{\alpha}{\lambda + \alpha} \tag{A.10}$$

$$\Pi_1 = \frac{\lambda}{\lambda + \alpha} \tag{A.11}$$

La ecuación de balance que resulta del modelo descrito es:

$$\lambda\Pi_0 = \alpha\Pi_1 \tag{A.12}$$

A partir de la caracterización de la fuente y mediante el modelo de fluidos se va a estudiar el mecanismo de control de flujo descrito en el apartado A.2. Para ello, se define *x* como la variable aleatoria continua que describe el número de paquetes que llegan a la cola durante el periodo de actividad, y αC como la capacidad normalizada respecto al número de paquetes que llegan en el periodo de actividad. La variable aleatoria que representa la ocupación de la cola en paquetes la denominaremos *m*. La relación entre *x* y *m* vendrá dada por $x=m*\alpha/V$. Además, se considerará que la cola es infinita.

Se define $F_i(t,x)$, con i=0,1, como la función de distribución de *x* en el instante *t* con el sistema en el estado *i*. A partir del análisis del modelo de la figura A.7, se puede escribir:

$$F_0(t+\Delta t, x) = \alpha\Delta t F_1(t,x) + (1-\lambda\Delta t)\,F_0(t, x+c\alpha\Delta t) + o(\Delta t) \tag{A.13}$$

$$F_1(t+\Delta t, x) = \lambda\Delta t F_0(t,x) + (1-\alpha\Delta t)\,F_1(t, x-(1-c)\alpha\Delta t) + o(\Delta t) \tag{A.14}$$

Se supone que $F_{-1}(\ )$ y $F_2(\ )$ son cero, y que $F_i(t,x)$ es continua y derivable. Desarrollando en serie de Taylor $F_i(t+\Delta t,x)$ y $F_i(t,x-\Delta x)$, y haciendo tender Δt a cero, quedan unas ecuaciones en derivadas parciales respecto a *t* y *x*. Si se supone estacionariedad, $\partial F_i(t,x)/\partial t = 0$, entonces queda el sistema de ecuaciones siguiente:

$$-C\alpha\frac{dF_0(x)}{dx} = -\lambda F_0(x) + \alpha F_1(x) \tag{A.15}$$

$$(1-C)\alpha\frac{dF_1(x)}{dx} = \lambda F_0(x) - \alpha F_1(x) \tag{A.16}$$

La solución de este sistema es una suma de exponenciales, donde el cálculo de las constantes se determina a partir de las condiciones de contorno. Finalmente, y siguiendo el desarrollo de M. Schwartz en [Schwartz96] se obtiene:

$$F_o(x) = \frac{\alpha}{\alpha+\lambda} - \rho(1-C)e^{-\frac{(1-\rho)(1+\gamma)}{(1-C)}x} \tag{A.17}$$

$$F_1(x) = \frac{\lambda}{\alpha + \lambda} - \rho C e^{-\frac{(1-\rho)(1+\gamma)}{(1-C)}x} \quad \text{(A.18)}$$

con $\rho = \frac{\lambda}{(\lambda + \alpha)C}$ y $\gamma = \frac{\lambda}{\alpha}$.

Para caracterizar el mecanismo de control de congestión, se ha de determinar la duración del ciclo del sistema y la probabilidad de pérdidas. El tiempo medio de actividad de la fuente definido como T2+D es igual al obtenido en la expresión (A.3). Como la fuente genera V paquetes/s y la duración de este periodo está limitado entre $m=Q_L$ y $m=Q_H$ se obtiene la igualdad anterior. El cálculo de la duración del tiempo de inactividad, T4+D, vendrá dado a partir de imponer en la función de distribución de la ocupación de la cola, $F(m)=F_0(m) + F_1(m)$, la siguiente condición:

$$F(m \le Q_L) = 1 - \rho e^{-\frac{(1-\rho)(1+\gamma)\alpha Q_L}{(1-C) \quad V}} \quad \text{(A.19)}$$

De aquí se obtiene T4 en función de T2 como:

$$T4 = \frac{H/VC}{\ln\left(\frac{D}{T2+D}\right) + \frac{H}{V(1-C)(T2+D)}} - D \quad \text{(A.20)}$$

El mecanismo de control de congestión sólo presenta pérdidas durante la fase III, por lo tanto la probabilidad de pérdidas se escribe como el producto entre la probabilidad de estar en esa fase por la probabilidad de pérdida condicionada a estar en la fase III.

$$P_B = \frac{D}{T2 + T4 + 2D} \rho e^{-\frac{(1-\rho)(1+\gamma)\alpha(B-Q_H)}{(1-C)V}} \quad \text{(A.21)}$$

A.6 VALIDEZ DEL MODELO

El análisis del mecanismo de control de congestión mediante el modelo de fluidos es válido para unos valores del tiempo de propagación que van desde $D1 \le D \ge L/VC$, donde D1 es el instante que minimiza la probabilidad de pérdidas F(m). En este rango, el análisis matemático se adecua al funcionamiento del sistema. La probabilidad de pérdidas aumenta al aumentar la carga del sistema y el tiempo de propagación, como puede observarse en la figura A.8. Al estudiar el caudal, definido en (A.7), en función del tiempo de propagación se observa que a partir de un valor de D igual a L/VC, el caudal disminuye drásticamente (figura A.9), comportamiento esperado si el umbral Q_L se fija a un valor inferior a VCD.

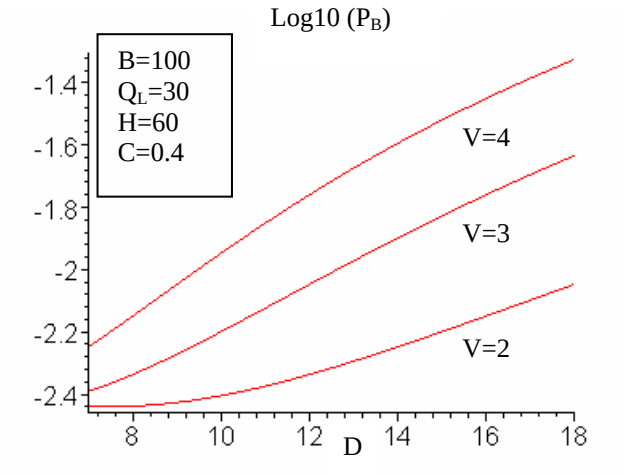

Figura A.8. Probabilidad de pérdidas en función del tiempo de ida y vuelta para el modelo de fluidos.

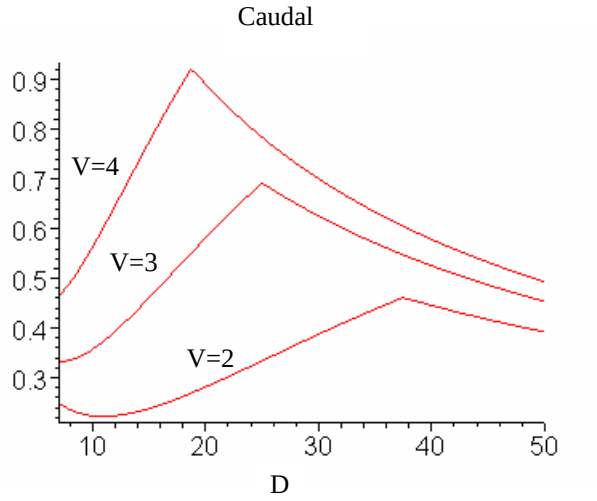

Figura A.9. Caudal en función del tiempo de ida y vuelta para el modelo de fluidos.

En este apéndice se ha analizado un mecanismo de control de congestión basado en la ocupación de la cola del enlace de menos recursos. Esta política que minimiza la probabilidad de pérdidas del sistema, y podía resultar viable en entornos homogéneos, es poco flexible y poco dinámica para entornos heterogéneos. Se ha obtenido a partir del estudio de los valores medios del sistema, un ágil y sencillo mecanismo para analizar las características del sistema y seleccionar los valores de los umbrales adecuados. También, se ha utilizado el modelo de fluidos para analizar el sistema como una fuente de dos estados.

APÉNDICE B

ESPECIFICACIÓN DEL PROTOCOLO RCCMP

B.1 INTRODUCCIÓN

A continuación, se va especificar el protocolo RCCMP utilizando un lenguaje estructurado de alto nivel. Éste diferencia muy claramente dos procesos totalmente independientes: el *rccmpsender*, que se ejecuta en la fuente y cuyas principales funciones son enviar paquetes de datos y procesar los paquetes que recibe: ACKs, NAKs e IC; y el *rccmpreceiver*, que se ejecuta en los receptores. Dependiendo del tipo de receptores, si son o no son el representante, sus funciones van desde recibir paquetes y ACKs, y solicitar paquetes perdidos, hasta enviar ACKs para confirmar que el paquete ha sido recibido correctamente.

B.2 EL EMISOR

```
Program   rccmpsender;
const     sstherhold; {umbral de la ventana para pasar de fase slow-start a
          a fase congestion avoidance}
          ndupack; {numero de ACK duplicados para detectar pérdida de pq}
          rtos_new_representative_factor; {numero de RTOs consecutivos
          antes de iniciar el proceso para seleccionar un nuevo
          representante}
          report_factor; {tiempo desde que se envía un PIC hasta recibir
          respuestas IC}
          report_maxtrx_factor; {Número de intentos para esperar respuesta,
          después del cual la simulación aborta}
          grace_period_factor; {tiempo del periodo de gracia}
          minimum_period_factor; {tiempo de permanencia mínimo como
          representante}
          switch_factor; {controla cuanto peor ha de ser el caudal del
          receptor que él del representante para que se produzca un cambio
          de representante}
```

```
            nak_factor; {tiempo mínimo para reenviar un paquete solicitado
            mediante NAK}

            desired_feedback; {parámetro de ajuste del temporizador que
            representa el número medio de NAKs recibidos}

var         struct ackh{

               int seq_acked; {número de secuencia confirmada}

               double losses;

               int highest_received;

               double time_stamp_echo;

            }

            struct nakh{

               int seq_naked; {número de secuencia confirmada}

               double losses;

               int highest_received;

            }

            struct ich{

               double losses;

               int highest_received;

               double timer;

            }

            struct packeth{

               int seq; {número de secuencia}

               int seq_acked; {último número de secuencia confirmada}

               direccion del representante;

               double time_stamp;

               double λ;

               double T;

               int command;

               double option;

            }

            window.last_acked; {último número de secuencia confirmada}

            window.pointer; {puntero que apunta al siguiente paquete a
            transmitir}

            cwnd; {tamaño de la ventana de transmisión}

            dupack; {número de ACK duplicados}

            numdupack; {número de ACK duplicados que se utiliza para mover la
            ventanta}

            representative.metrics; {métricas del representante}

            representative.backoff; {controla el proceso de backoff del RTO}

            representative.address.addr; {dirección del representante}

            estimador.first_timer_received; {valor del temporizador mínimo
            que se utiliza para estimar el numero de receptores}

            estimador.last_sample_estimated; {numero de respuestas IC}

            state; {estado del emisor al llegar un paquete de informe de
            congestión}
```

```
        state.temporizador; {estado del temporizador}
procedure Initialize; {Construye Agent/RCCMP/Sender object, inicializa
        todas las variables y temporizadores}
prodedure askForReport(bool estimation, double losses); {Envía un PIC a
        todos los receptores. Si bool= true, las respuestas de los
        receptores serán retardadas, si bool=false las respuestas serán
        inmediatas. Sólo los receptores con tasa de pérdidas mayor que
        losses pueden responder a este paquete}
procedure IP_address(packet p); {Extrae el campo de dirección IP del
        paquete}
procedure try_to_send; {Envía todos los paquetes posibles hasta completar
        la ventana}
procedure getMetrics(double losses, int highest); {Computa una aproximación
        de la ecuación de equilibrio de TCP que servirá para comparar la
        calidad de los receptores}
procedure updateParameters(double last_rtt); {actualiza cuando hay una
        nueva medida de rtt los parámetros rtt, varianza rtt, λ y T}
procedure updateReportTimer; {actualiza el temporizador utilizado para
        esperar ICs después de haber enviado un PIC. Si expira sin
        respuesta se reactualiza. Si no hay respuesta después de un
        número de intentos report_maxtrx_factor, la simulación aborta}
procedure updateStopEstimatorTimer; {Actualiza el temporizador que se
        utiliza para parar la estimación de los receptores. Esta función
        se llama después del primer IC respuesta a la PIC para la
        estimación de receptores}
procedure updateMinimumPeriodTimer; {Actualiza el temporizador que
        comprueba que el receptor ha sido representante un tiempo mínimo}
procedure updateStartEstimatorTimer; {Actualiza el temporizador que
        comprueba si es necesario hacer una nueva estimación de
        receptores}
procedure updateRtoTimer; {Actualiza el temporizador RTO, utilizado para la
        retransmisión de paquetes si no se ha recibido el ACK del
        representante}
procedure packets_getByIndex(int index); {Extrae el paquete index}
procedure start_grace_period_timer;{inicializa el temporizador para el
        proceso de inicialización de selección de representante}
procedure cancelar(temporizador);
procedure calcular_num_receiver_estimated (int respuestas); {estima el
        número de receptores según la fórmula de Towsley}
begin   Inicializate;
        askForReport(false,0); {Envía PIC, todos los receptores pueden
        responder}
        repeat Esperar un suceso;
         tipo_suceso={paquete,expiración_temporizador}
         tipo_paquete= tipo de paquete;
         tipo_temporizador= tipo de temporizador;
         switch (tipo_paquete) of

          case ACK:
           acked=ackh.seq_acked; {número de secuencia confirmada}
           {Comprobar que el ACK proviene del representante actual}
           if(representative_address.addr=IP_address(ackh))
```

```
if(acked>window.last_acked) {comprobar que la secuencia del
ACK es mayor a la última confirmada}
 window.last_acked=acked;
 {calculo aprox. caudal TCP del representante}
 representative.metrics=getMetrics(ackh.losses,ackh.acked);
 UpdateParameters(rtt);{actualiza valores rtt, var rtt,T,λ}
 {actualiza tamaño ventana}
 if(cwnd>=sstherhold)
  {Fase congestion-avoidance}
  cwnd+=1/cwnd;
 else
  {Fase show_start}
  cwnd= cwnd+1;
 end;
 try_to_send; {se envían paquetes hasta completar la
 ventana}
 updateRTOTimer; {Se actualiza el valor de RTO y se inicia}
end;
else if (acked= window.last_acked) {ha llegado un ACK con
numero de secuencia duplicado}
 ++dumpack; {contabiliza el num de ACKS duplicados}
 if (dumpack<ndupack) {Si ACKs duplicados menor que 3}
  numdupack++;
  try_to_send; {se envía 1 paquete de datos)
 end
 else if (dupack=ndupack) {Si 3 ACKs duplicado, se detecta
 pérdida de paquete}
  {se modifican los valores de la ventana}
  sstherhold=cwnd/2;
  cwnd=sstherhold;
  p_loss=packets_getByIndex(window.last_acked); {paquete
  perdido}
  send(p_loss); {se retransmite paquete perdido}
 end;
 else
  numdupack++; {si la ventana no esta completa pueden
  enviarse paquetes}
 end;
end;
end;

case NAK:
 naked=nakh.seq_naked; {num. de paquete para el que se
 solicita retransmision}
 p_loss=packets_getByIndex(naked); {paquete perdido}
 now=clock();
```

```
  if(now-p_loss.time_stamp>nak_factor*rtt) {si el tiempo entre
  que se envío originalmente el pq y ahora es suficientemente
  grande, entonces se retransmite el pq}
   send(p_loss);
  end;
  {comprobar si el receptor que ha enviado el NAK puede ser el
  nuevo representante}
  sender_metrics=getMetrics(nakh.losses,nakh.highest_received)
  if(sender_metrics*switch_factor>representative_metrics)
   {si el caudal del receptor es suficiente inferior, cambio de
   representante}
   representative.address=IP_address(nakh); {actualiza la @}
   representative.metric=sender_metrics; {actualiza las
   métricas del nuevo representante}
  end;

 case IC:
  switch(state) of

   case WAITING_FOR_NON_DELAYED_REPORT: {Primer informe
   recibido}
    state=GRACE_PERIOD;
    start_grace_period_timer;{inicializar el periodo de gracia}
    cancelar(report_timer); {cancela el temporizador que
    controla el tiempo hasta la recepción de un respuesta IC}

   case GRACE_PERIOD:
    sender_metrics=getMetrics(ich.losses,ich.highest_received);
    if(sender_metrics*switch_factor>=representative_metrics)
    {si llega un informe con características iguales o peores
    al representante, entonces se produce un cambio de
    representante}
    representative.address= IP_address(ich);
    representative.metric=sender_metrics;
   end;

   case STARTING_ESTIMADOR:
    state=STOPING_ESTIMATOR;
    updateStopEstimatorTimer; {se inicia el temporizador para
    contabilizar el num. de IC retardados recibidos}
    estimador.first_timer_received=ich.timer; {se coge el
    tiempo del temporizador del primer IC}

   case STOPING_ESTIMADOR:
    ++estimador.last_sample_estimated; {se incrementa el número
    de respuestas ICs}
  end;
end;
```

```
switch (tipo_temporizador) of

  case RTO_TIMER:
  {El temporizador RTO_TIMER ha expirado. No ha llegado el ACK
  desde el representante}
  {modificación del tamaño de ventana}
  sstherhold=cwnd;
  cwnd=1;
  numdupack=0;
  p_loss=packets_getByIndex(window.last_acked);
  send(p_loss);
  window.pointer=window.last_acked+1; {se actualiza el puntero a
la posición siguiente a la última confirmada. Permitirá el
reenvío de paquetes desde el último confirmado}
  if(representative_backoff<64)
   {si expira RTO sin recibir ACK. Se aumenta el tiempo de RTO}
   representative_backoff=representative_backoff*2;
  end;
  if(representative_backoff>rtos_new_representative_factor)
  {Si el backoff ha alcanzado un valor alto}
   askForReport(false,0);{seleccionamos un nuevo representante}
  end;
  try_to_send; {se envia toda la ventana}
  updateRTOTimer();{se actualiza RTO y se inicia}


  case REPORT_TIMER:
   {El emisor envía un PIC y no recibe respuestas de los
  receptores. Se envía una nueva petición}
   askForReport(false,0);


  case GRACE_PERIOD_TIMER:
   updateMinimumPeriodTimer; {El periodo de gracia ha acabado,
   existe un representante. Este tiene que permanecer en este
   estado un periodo de tiempo mínimo}
   updateParameters;
   updateStartEstimatorTimer;
   {Temporizador que controla la próxima vez que hay que enviar
   un PIC}


  case MINIMUM_PERIOD_TIMER:
   {Ha expirado el tiempo de permanencia mínima como
   representante}
   state.temporizador=None;


  case START_ESTIMATOR_TIMER:
```

```
            {se lanza el PIC para estimar el num. de receptores}
            estimator.last_sample.estimated=0; {inicialización de
              respuestas IC}
            askForReport(true,0);

          case STOP_ESTIMATOR_TIMER:
          {se procede a estimar el número de receptores y se actualizan
           las variables del temporizador}
          calcular_num_receiver_estimated
             (estimator.last_sample.estimated); {calcula el num de
             receptores según la fórmula de Towsley}
          updateParameters;

      until Eternament;
end.
```

B.3 EL RECEPTOR

```
Program   rccmpreceiver;
const     automatic_send_factor; {parámetro que regula cuanto peor ha de
          ser la tasa de pérdidas del receptor que la del representante
          para enviar inmediatamente un NAK}
          weight_losses_last_sample; {peso del filtro paso bajo utilizado
          para calcular las pérdidas}
var       struct ackh{
             int seq_acked; {número de secuencia confirmada}
             double losses;
             int highest_received;
             double time_stamp_echo;
          }
          struct nakh{
             int seq_naked; {número de secuencia confirmada}
             double losses;
             int highest_received;
          }
          struct pich{
             double losses;
             int highest_received;
             bool delayed; {indica si la respuesta se ha de retardar}
          }
          struct packeth{
             int seq; {número de secuencia}
             int seq_acked; {último número de secuencia confirmada}
             direccion del representante;
             double time_stamp;
```

```
            double lambda;
            double T;
            int command;
            double option;
          }p_datah;
          window.last_seq_ack; {último número de secuencia confirmada}
          first_when_representative; {paquete a partir del cual el nuevo
          representante controla la comunicación}
          last_when_representative; {paquete a partir del cual el actual
          representante deja de serlo}
          metric.losses_representative; {tasa de pérdidas del
          representante}
          metric.losses; {tasa de pérdidas del receptor}
          metrics.highest_received; {secuencia del último paquete recibido}
          last_time_stamp; {almacena el timestamp del paquete de datos
          recibidos}
          addr here.addr; {dirección IP del receptor}
procedure Initialize; {Construye Agent/RCCMP/Receiver object, inicializan
          todas las variables}
procedure updateReceivedList(int num_seq); {actualiza la lista de paquetes
          recibidos y no recibidos}
procedure updateLosses(int new_seq); {actualiza la tasa de pérdidas hasta
          el último paquete confirmado}
procedure getIndextoAck; {devuelve el índice que el representante debe
          confirmar}
procedure setupReport(double timer); {crea un IC y lo envía}
procedure CheckNak; {comprueba si hay que enviar un NAK}
procedure SetupNak; {construye y envía un NAK}
procedure inicia_temporizador_exponencial(double lambda, double T);
procedure IP_Address(p packet); {extrae la dirección IP del paquete}
procedure representative (packeth p, addr here.addr); {determina si el
          receptor es el representante}
begin     Inicializate;
          repeat Esperar un suceso;
           tipo_suceso={paquete,expiración_temporizador}
           tipo_paquete= tipo de paquete;
           tipo_temporizador= tipo de temporizador;
           switch (tipo_paquete) of

            case ACK:
             {si soy el representante, no necesito escuchar los ACKs}
             if(IP_Address(ackh)==here.adr) return;
             else
              {si no soy el representante. Actualizo la tasa de pérdidas,
              el último número de secuencia confirmado y la tasa de
              pérdidas del representante}
              updateLosses(ackh.seq_acked);
```

```
  window.last_seq_ack=ackh.seq_acked;
  metric.losses_representative=ackh.losses;
  checkNak; {comprueba si ha hay algún paquete perdido que el
  reperesentante haya confirmado}
 end;

 case P_DATA:
  {si este receptor es ahora el representante}
  if(representative(p_datah, here.addr)
   first_when_representative=P_DATA.seq_acked; {actualiza
  desde que paquete debe empezar a controlar la comunicación}
  else
  {si este receptor ya no es el representante}
   last_when_representative= p_datah.seq_acked; {actualiza
  desde que paquete debe dejar de controlar la comunicación}
  end;
  {actualiza los parámetros del temporizador}
  λ=p_datah.lambda;
  T=p_datah.T;
  last_time_stamp= p_data.timestamp; {almacena el timestamp
  del paquete que controlará cuando enviar un ACK retardado}
  {si este receptor es el representante}
  if(representative(p_datah, here.addr)
    {se actualiza la tasa de pérdidas y la lista de los
    paquetes recibidos. Se comprueba si hay que generar un NAK
    y se envía un ACK retardado}
    updateLosses(metrics.highest_received);
    updateReceivedList(p_data.seq);
    checkNak;
    setupAck(last_time_stamp);
  else
    {si este receptor no es el representante}
    {actualiza la lista de los paquetes recibidos y comprueba
    si hay que generar un NAK}
    updateReceivedList(p_data.seq);
    checkNak;
  end;

case PIC:
  {comprobar si este receptor debe responder con un paquete IC}
  if((pich.losses=0 || pich.losses<=metrics.losses) &&
  pich.non_delayed)
  {Si el PIC contiene los campos de tasa de perdidas igual a
  cero o menores que las que este receptor computa y además
  solicita una respuesta no retardada}
   setupReport(0); {envía respuesta IC}
  else if(pich.delayed)
```

```
            inicia_temporizador_exponencial(λ,T);
          end;

        end;

       switch (tipo_temporizador) of

        case NAK_TIMEOUT:
          {expira el temporizador que controla el tiempo de espera de
      un NAK. Se envía el NAK hacia la fuente}
          setupNak;

      case ACK_TIMEOUT:
          {temporizador que controla los ACK retardados}
          setupAck(last_time_stamp);

      case FEEDBACK_TIMEOUT:
          {expira el temporizador que controla el tiempo de espera de
      un IC. Se envía la respuesta}
          setupReport;

       end;
      until Eternament;
end.
```

APÉNDICE C

EVALUACIÓN DE LA COMPLEJIDAD DEL PROTOCOLO

C.1 INTRODUCCIÓN

A partir de la especificación del protocolo se puede pasar a su evaluación, para analizar los recursos de espacio de almacenamiento y de tiempo de cómputo que RCCMP necesita para su desarrollo. Una primera observación de su código permite observar que RCCMP resulta escalable, ya que el número de variables no se incrementa con el tamaño del grupo multipunto. A continuación se justificará que el protocolo propuesto es implementable, ya que es un protocolo simple, que no requiere de grandes cantidades de memoria y que su tiempo de ejecución responde a una función de orden constante.

C.2 EL EMISOR

A continuación, se van a analizar los recursos que consume la implementación del *rccmpsender*.

C.2.1 Tamaño de la cola del emisor

RCCMP utiliza ACK acumulativos, por lo que si el emisor recibe el paquete de confirmación positiva con un número de secuencia igual a *i*, eso significa que todos los paquetes anteriores a éste han sido correctamente recibidos por el representante. Sin embargo, un receptor al recibir un ACK con número de secuencia *i*, debe comprobar si ha recibido ese paquete de datos. Si no lo ha recibido, debe generar un NAK y retardarlo un tiempo aleatorio antes de enviarlo hacia la fuente. Por lo tanto, el tamaño de la cola de la fuente debe almacenar todos los paquetes enviados, para

que cualquier receptor pueda recuperar un paquete perdido. Este comportamiento es propio de los protocolos que utilizan NAKs.

De todas formas, se va a tratar de acotar el valor máximo. Para ello, se considera que el tamaño de la cola del emisor debe ser igual al tamaño de ventana actual, ya que son paquetes que no han sido confirmados, más un número de paquetes que dependen de los mecanismos de recuperación de RCCMP. El tamaño de la cola puede ser aproximado mediante la siguiente ecuación

$$tamaño\_buffer = W + (t + \frac{RTT}{2}) \cdot tasa\_de\_fuente \qquad \text{(C.1)}$$

donde *W* es el tamaño de la ventana, *t* simboliza el tiempo de espera aleatorio, *RTT* es el tiempo de ida y vuelta entre el receptor que solicita el NAK y la fuente, y la *tasa_de_fuente* es el caudal inyectado por el emisor en el canal. Por lo tanto, el segundo sumando representa el número de paquetes que deben ser almacenados por si algún receptor solicita la retransmisión de un paquete.

En el apéndice D se muestra un estudio de los temporizadores exponenciales. De ahí podemos extraer el tiempo medio de respuesta de un temporizador exponencial, por el cual se podría aproximar *t*

$$t \approx \frac{RTT}{\ln(N)} e + \frac{RTT}{2} \qquad \text{(C.2)}$$

donde se considera que RTT es dos veces el retardo de propagación entre la fuente y cualquier receptor y N es el número de respuestas que llegan al emisor. Para calcular una cota del tamaño de la cola, se va a considerar la hipótesis de que todos los RTTs son iguales al tiempo de ida y vuelta entre el representante y el emisor.

Por otro lado, si el tamaño de ventana, *W*, está bien ajustado se tiene la siguiente expresión [Stevens95]

$$W = anchodebanda \cdot RTT \qquad \text{(C.3)}$$

donde el *anchodebanda* es la capacidad entre la fuente y el representante, y puede ser aproximado por la *tasa_de_fuente.* Por lo que el tamaño de la cola se puede aproximar a la siguiente expresión:

$$tamaño\_buffer = 2 \cdot W + (\frac{RTT}{\ln(N)} e) \cdot tasa\_de\_fuente \qquad \text{(C.4)}$$

RCCMP ajusta el número de respuestas, N, para un buen funcionamiento a un valor de 4, para que el número de NAKs se mantenga acotado, pero no aumente el tiempo de respuesta. Así, se aproxima $\frac{e}{\ln(N)} \approx 2$, por lo que el tamaño de la cola se puede acotar a cuatro veces el tamaño de la ventana de transmisión. Si como en el protocolo TCP, se considera un tamaño máximo de ventana de igual a 64 paquetes, tenemos que RCCMP necesita de una cola máxima de transmisión de 256 paquetes.

C.2.2 Temporizadores

El emisor cuenta con seis temporizadores:

1. *grace_period_timer.* El periodo de gracia, es decir, el tiempo que dura el proceso inicial de búsqueda de representante.
2. *minimum_period_timer.* El tiempo de permanencia mínimo que debe pasar un receptor como representante.
3. *rto_timer.* El temporizador RTO. Controla el tiempo desde que se envía un paquete hasta que se recibe un ACK.
4. *report_timer*. Controla el tiempo desde que se envía un PIC hasta que se recibe la primera respuesta IC. Asegura que haya receptores en la sesión.
5. *star_estimator_timer*. Controla el tiempo de espera para volver a iniciar el proceso de estimación del tamaño del grupo multipunto.
6. *stop_estimator_timer*. Controla el tiempo desde que se envía un PIC que solicita respuestas retardadas hasta la llegada de las respuestas IC. Una vez que expira se estima el número de receptores en la sesión.

C.2.3 Variables y funciones

El emisor RCCMP define cinco clases de variables:

1. *class Representative* de tamaño 74 octetos. Almacena las características que la fuente debe conocer del representante: dirección IP, tasa de pérdidas, RTT, varianza del RTT.
2. *class Window* de tamaño 41 octetos. Almacena las características propias para los procesos de envío y recepción de paquetes como tamaño de la ventana, ventana máxima de transmisión, sstherhold, número de secuencia enviada, número de secuencia recibida.
3. *class State* de tamaño 10 octetos. Guarda los diferentes estados de la fuente.
4. *class Estimator* de tamaño 40 octetos. Almacena parámetros del temporizador y otros contadores: T, λ, receptores estimados, número de respuestas recibidas.
5. *class Parameter* de tamaño 97 octetos. Guarda los parámetros de diseño del protocolo: *switch_factor, nak_factor, rtt_initial_value…*

Si se suma el coste total de estas variables obtenemos 262 octetos. Como puede observarse, estas variables no aumentan con el número de receptores.

El código de RCCMP utiliza funciones que actualizan variables y temporizadores, que se encargan del envío y de la recepción de paquetes, y por último emplea ciertas funciones matemáticas. Las cuatro funciones matemáticas, que implementa el código RCCMP, se encargan de calcular la función de distribución exponencial, de computar el caudal aproximado en función de la ecuación (4.3), de ajustar los parámetros de los temporizadores, de actualizar el valor del temporizador de retransmisión, y por último, de estimar el número de receptores activos en la sesión utilizando la ecuación (4.8).

C.2.4 Orden del código del emisor

Para calcular el tiempo de ejecución del programa *rccmpsender,* el tiempo se expresa en término de pasos de computación elementales (asignaciones, comparaciones, multiplicaciones, etc). Por ejemplo, una operación de asignación ocupa una unidad de tiempo para ejecutarse, mientras que un ciclo ocupa el número de iteraciones en el que está definido. La duración del programa no vendrá dada en función de una determinada máquina, sino en función del tamaño de la entrada que define la magnitud, que al aumentar incrementa la complejidad del algoritmo. También hay que tener en cuenta la naturaleza de los datos de entrada, ya que dependiendo cuales sean estos datos se ejecutarán o no determinadas instrucciones de decisión y será distinto el número de iteraciones de los bucles. Generalmente, se opta por el peor caso, es decir, el que consume mayor tiempo de ejecución. Para simplificar el estudio se obtendrá una cota del tiempo de ejecución expresada mediante la notación O.

En el emisor el proceso que más tiempo consume es el procesado de un paquete ACK, que confirma a los paquetes anteriormente enviados. Este procedimiento incluye el cálculo del caudal del representante según la ecuación de equilibrio de TCP, la actualización del cálculo del RTT, de su varianza y de los parámetros del temporizador, además de enviar los paquetes necesarios según el tamaño de ventana y el inicio del temporizador de retransmisión. Los procesos que más tiempo consumen son los relacionados con ordenar, buscar y actualizar los paquetes almacenados. Con el empleo de tablas Hash, la complejidad de estos procedimientos en función del número de paquetes almacenados es de orden lineal. Sin embargo, si se acota el número de paquetes máximo que puede almacenar el emisor, una estructura tipo array es suficiente y el tiempo de acceso es constante e independiente del número de paquetes almacenados. Por lo tanto, si se utiliza la primera opción se obtiene que la complejidad computacional del código *rccmpsender* es lineal, O(n), y por lo tanto es un problema implementable y tratable. Si se utiliza la segunda opción la complejidad computacional queda reducida a O(k), es decir, de orden constante. Esto significa que el tiempo de proceso es independiente del tamaño de la entrada, en este caso, del número de paquetes que hay almacenados en el emisor.

C.3 EL RECEPTOR

A continuación se van a analizar los recursos que consume la implementación del *rccmpreceiver.*

C.3.1 Tamaño de la cola del receptor

La capa de transporte receptora debe entregar los paquetes en orden a la aplicación, por lo que la cola de recepción debe tener un tamaño suficiente que permita la recuperación de cualquier paquete perdido. Por ejemplo, si se detecta la pérdida del paquete con número de secuencia *i,* el receptor debe generar un NAK, iniciar el temporizador y cuando expire, enviarlo. Por lo tanto, el NAK tarda en

llegar al emisor el tiempo de espera aleatorio *(t)* más el tiempo de propagación hasta la fuente. En el peor de los casos, la fuente tardará *nak_factor*RTT* (*RTT* es el tiempo de propagación entre la fuente y el representante) en reenviar el paquete solicitado. Por lo que el paquete retransmitido llega al receptor en el siguiente tiempo:

$$t + RTT + nak\_factor \cdot RTT \tag{C.5}$$

donde *t* es el tiempo de espera aleatorio, y *RTT* es el tiempo de ida y vuelta entre la fuente y el receptor que solicita el NAK. Al igual que para el cálculo del tamaño de la cola de emisión, se va a suponer que todos los RTT son iguales al tiempo de ida y vuelta entre la fuente y el representante. Si se multiplica el tiempo de la ecuación (C.5) por la tasa de la fuente, y se sustituye el valor del tiempo de espera medio, calculado en la ecuación (C.2), se obtiene una expresión aproximada del tamaño mínimo de la cola de recepción:

$$tamaño\_buffer = (\frac{RTT}{\ln(N)}e + \frac{3 \cdot RTT}{2} + nak\_factor \cdot RTT) \cdot tasa\_de\_fuente \tag{C.6}$$

Utilizando la misma suposición que se utilizó para calcular la longitud de la cola del emisor, de manera que se aproxima $\frac{e}{\ln(N)} \approx 2$ y el parámetro *nak_factor* se recomienda ajustarlo para un buen funcionamiento de RCCMP a 3. Se tiene que el tamaño de la cola de recepción tiene un tamaño mínimo de 6.5 veces la ventana de transmisión. Si como antes consideramos W a 64, el tamaño del buffer de recepción es de 416 paquetes.

C.3.2 Temporizadores

El receptor cuenta con tres temporizadores:

1. *ack_timer*. Controla el tiempo para la transmisión de ACK retardados.
2. *nak_timer*. Temporizador exponencial que controla el tiempo para enviar el NAK.
3. *feedback_timer*. Cuando se recibe un PIC con solicitud de respuestas retardadas, este temporizador genera el tiempo de espera.

C.3.3 Variables y funciones

El receptor RCCMP define cuatro clases de variables:

1. *class State* de tamaño 16 octetos. Determina las direcciones IP de la fuente y el representante.
2. *class Window* de tamaño 17 octetos. Guarda variables para la recepción y transmisión de paquetes: última secuencia recibida, última secuencia confirmada, *timestamp…*

3. *class Metrics* de tamaño 34 octetos y 1 RNG (generador de números aleatorios). Almacena variables como la tasa de pérdidas del receptor, del representante y los parámetros del temporizador.
4. *class Parameter* de tamaño 16 octetos. Almacena las características de diseño del receptor RCCMP.

El coste total de estas cuatro variables es de 83 octetos.

Las funciones, que intervienen en el código del receptor, se pueden clasificar en aquellas que actualizan parámetros y temporizadores; las que controlan la recepción de paquetes, y las funciones matemáticas. Estas últimas se encargan de generar una variable aleatoria exponencial y de calcular las pérdidas por medio de un filtro paso bajo de primer orden según la ecuación (4.2).

C.3.4 Orden del código del receptor

Se va a calcular una cota del tiempo de ejecución del programa *rccmpreceiver* del mismo modo que se realizó para el emisor. En este caso el proceso que más tiempo consume es el procesado de un paquete de datos para un receptor que es el representante. Este procedimiento incluye la actualización de la tasa de pérdidas, la comprobación de que el paquete que se esperaba recibir es el paquete recibido, se debe generar un paquete de confirmación, comprobar si se debe enviar un NAK solicitando algún paquete que no haya sido recibido durante la fase en que este receptor no era representante, y por último actualizar los temporizadores. Al igual que en el caso del emisor, los procesos que más tiempo consumen son los relacionados con buscar y ordenar paquetes. Si estos procesos se realizan de forma eficiente a través de tablas Hash o limitando la capacidad de almacenamiento mediante un array, la complejidad computacional del *rccmpreceiver* es de O(n) y O(k). Si el receptor no es el representante, la complejidad es también lineal o constante en función de la opción elegida para procesar los paquetes almacenados.

APÉNDICE D

TEMPORIZADORES EXPONENCIALES

D.1 INTRODUCCIÓN

RCCMP, para asegurar la fiabilidad, usa un mecanismo de confirmaciones negativas. Para garantizar la escalabilidad del protocolo, el número de NAKs generado por cada paquete perdido se limita mediante un esquema de temporizadores exponenciales. Este mecanismo se basa en que antes de transmitir una confirmación negativa, el receptor espera un tiempo aleatorio. Si durante este tiempo de espera se recibe el paquete perdido, el NAK puede cancelarse.

El objetivo de todo esquema de temporizadores es reducir al mínimo el número de NAKs enviados, produciendo la cancelación de la mayoría de ellos. Así, los temporizadores deben asegurar que dado un conjunto de N receptores, que van a enviar un NAK, uno o unos pocos de ellos, han de generar un tiempo de espera muy inferior a la media, de tal manera que el paquete sea retransmitido antes de que la mayoría de los receptores envíen su NAK. De esta manera, se logrará la supresión de la información de realimentación, garantizado al mismo tiempo la fiabilidad.

Los temporizadores usados en la mayoría de los protocolos eran en un principio de tipo uniforme, sin embargo en el [RFC3941] se recomienda el uso de los exponenciales. De hecho, diversos trabajos, [Nonnen98a] [Adamson02], comparan la eficiencia de uno y otros, concluyendo que los temporizadores exponenciales tienen una mejor respuesta.

En este apéndice se van a estudiar los temporizadores exponenciales que han sido utilizados en el protocolo RCCMP. Este análisis está basado en el trabajo [Nonnen98a].

D.2 TEMPORIZADORES EXPONENCIALES

Un temporizador exponencial es aquel que genera un tiempo de espera con la siguiente función densidad de probabilidad:

$$f(z)=\begin{cases}\dfrac{1}{e^{\lambda}-1}\dfrac{\lambda}{T}e^{\frac{\lambda}{T}z} & si\ 0\le z\le T\\ 0 & en\ otro\ caso\end{cases} \tag{D.1}$$

donde T es el máximo tiempo de espera y λ es el parámetro de la exponencial truncada. Si se asume la simplificación de que el retardo entre el emisor y cada uno de los receptores es constante e igual a *c*, se puede estimar fácilmente el número de respuestas recibidas para R receptores según el valor de T.

Para ello, se considera que un NAK tarda c segundos en llegar al emisor, y que éste reacciona inmediatamente transmitiendo el paquete. De manera que transcurren *2*c* segundos desde que se genera el NAK hasta que llega la retransmisión a todos los receptores. Teniendo esto en cuenta, y designando como m al tiempo mínimo entre los tiempos aleatorios generados por los R receptores, se puede deducir que la probabilidad de que el receptor iésimo envíe una respuesta al emisor es la probabilidad de que genere un tiempo de espera inferior a *m+2*c*, ya que si el tiempo de espera fuera superior, recibiría la retransmisión antes de que pudiera enviar la respuesta. Denotando con $E(X_i)$ la probabilidad de recibir respuesta del receptor R, se tiene:

$$E(X_i)\big|_{para\ un\ valor\ de\ m\ dado}=p(tiempo\ de\ espera\ z_i\le m+2c) \tag{D.2}$$

Haciendo la media para todos los valores posibles de m, se obtiene

$$E(X_i)=\int_o^T p(z_i\le m+2c)\,p(m)dm \tag{D.3}$$

La probabilidad de que un valor m concreto sea el mínimo es la probabilidad de que cualquier receptor genere un valor de espera m y el resto, un valor superior. Teniendo esto en cuenta para los R receptores, queda:

$$p(m)=R\cdot p(z_i=m)\cdot p(z_i\ge m)^{R-1} \tag{D.4}$$

Aplicando la función de densidad exponencial, (D.1), a la ecuación (D.4), se deduce:

$$\begin{aligned}&p(z_i\le m+2c)=\frac{e^{\frac{(m+2c)\lambda}{T}}-1}{e^{\lambda}-1} && si\ T\ge 2c+m\\ &p(z_i\le m+2c)=1 && si\ T\le 2c+m\end{aligned} \tag{D.5}$$

$$p(m)=\frac{R\lambda}{T}\frac{e^{\frac{m}{T}\lambda}}{e^{\lambda}-1}\left(\frac{e^{\lambda}-e^{\frac{m}{T}\lambda}}{e^{\lambda}-1}\right)^{R-1} \tag{D.6}$$

Aplicando estos resultados a la ecuación (D.3), se obtiene

$$\begin{aligned}E(X_i)=&\frac{R\lambda}{T(e^{\lambda}-1)^{R+1}}\int_o^{T-2c}\left(e^{\frac{m}{T}\lambda}\right)\left(e^{\lambda}-e^{\frac{m}{T}\lambda}\right)^{R-1}\left(e^{\frac{(m+2c)}{T}\lambda}-1\right)dm+\\ &\frac{R\lambda}{T(e^{\lambda}-1)^{R}}\int_{T-2c}^{T}\left(e^{\lambda}-e^{\frac{m}{T}\lambda}\right)^{R-1}\left(e^{\frac{m\lambda}{T}}\right)dm \quad si\ T\ge 2c\end{aligned} \tag{D.7}$$

$$E(X_i) = \frac{R\lambda}{T(e^{\lambda}-1)^R}\int_0^T \left(e^{\lambda} - e^{\frac{m}{T}\lambda}\right)^{R-1}\left(e^{\frac{m\lambda}{T}}\right)dm \quad si\ T \le 2c \tag{D.8}$$

Para integrar estas ecuaciones, resulta muy útil el cambio de variable:

$$e^{\frac{m}{T}\lambda} = l \rightarrow dm = \frac{T}{l\lambda}dl \tag{D.9}$$

Así,

$$\int_0^{T-2c}\left(e^{\lambda} - e^{\frac{m}{T}\lambda}\right)^{R-1}\left(e^{\frac{(m+2c)}{T}\lambda} - 1\right)dm =$$

$$\frac{Te^{\frac{2c}{T}\lambda}}{\lambda R(R+1)(e^{\lambda}-1)^{R+1}}\left[\left(e^{\lambda}-1\right)^R - e^{(R+1)\lambda}\left(1-e^{-\frac{2c}{T}\lambda}\right)^{R+1}\right] + \tag{D.10}$$

$$\frac{Te^{\frac{2c}{T}\lambda}}{\lambda R(e^{\lambda}-1)^{R+1}}\left[\left(1-e^{-\lambda\frac{2c}{T}}\right)\left(e^{\lambda}-1\right)^R - e^{-\frac{2c}{T}\lambda}(e^{\lambda}-1)e^{R\lambda}\left(1-e^{-\frac{2c}{T}\lambda}\right)^R\right]$$

$$\int_{T-2c}^{T}\left(e^{\lambda} - e^{\frac{m}{T}\lambda}\right)^{R+1}\left(e^{\frac{m}{T}\lambda}\right)dm = \frac{Te^{R\lambda}}{\lambda R(e^{\lambda}-1)}\left(1-e^{-\frac{2c}{T}\lambda}\right)^R \tag{D.11}$$

$$\int_0^{T}\left(e^{\lambda} - e^{\frac{m}{T}\lambda}\right)^{R-1}\left(e^{\frac{m}{T}\lambda}\right)dm = \frac{T(e^{\lambda}-1)^R}{\lambda R} \tag{D.12}$$

Puede concluirse que

$$E(X_i) = \frac{e^{\lambda\frac{2c}{T}}-1}{e^{\lambda}-1} - \frac{1}{R+1}e^{\lambda\frac{2c}{T}}\left[\left(\frac{1-e^{-\lambda\frac{2c}{T}}}{1-e^{-\lambda}}\right)^{R+1} - 1\right] si\ T \ge 2c \tag{D.13}$$

$$E(X_i) = 1 \qquad si\ T \ge 2c \tag{D.14}$$

Calculada la probabilidad de que receptor iésimo reciba respuesta, es posible deducir el número medio de respuestas, E(X):

$$E(X) = \sum_{j=0}^{R} E(X_j) = R \cdot E(X_i) \tag{D.15}$$

Por lo tanto, aplicando a la ecuación (D.15) la expresión (D.13) y (D.14)

$$E(X) = R\frac{e^{\frac{2c}{T}\lambda}-1}{e^{\lambda}-1} - \frac{R}{R+1}e^{\frac{2c}{T}\lambda}\left[\left(\frac{1-e^{-\frac{2c}{T}\lambda}}{1-e^{-\lambda}}\right)^{R+1} - 1\right] \quad si\ T \ge 2c \tag{D.16}$$

$$E(X) = R \qquad si\ T \le 2c \tag{D.17}$$

Suponiendo que el número de receptores es suficientemente grande, se cumplen las siguientes aproximaciones

$$\frac{R}{R+1} \approx 1 \tag{D.18}$$

$$\left(\frac{1-e^{-\lambda\frac{2c}{T}}}{1-e^{-\lambda}} \right)^{R+1} \approx 0 \; si \; T \geq 2c \tag{D.19}$$

Con esto, el número medio de respuestas recibidas, queda, aproximadamente como

$$E(X) \approx R\frac{e^{\frac{2c}{T}\lambda}-1}{e^{\lambda}-1} + e^{\frac{2c}{T}\lambda} \qquad si \; T \geq 2c \tag{D.20}$$

$$E(X) = R \qquad si \; T \leq 2c \tag{D.21}$$

El tiempo medio de espera para recibir respuesta en el emisor no es más que el valor medio del mínimo temporizador, *m*, más el retardo entre emisor y receptor, c. Así, el valor medio del mínimo temporizador, E(M), se puede calcular como

$$E(M) = \int_0^T m \cdot p(m) dm = \int_0^T m \frac{R\lambda}{T} \frac{e^{\frac{m}{T}\lambda}}{e^{\lambda}-1} \left(\frac{e^{\lambda}-e^{\frac{m}{T}\lambda}}{e^{\lambda}-1} \right)^{R+1} dm \tag{D.22}$$

donde

$$\int_0^T e^{\frac{m}{T}\lambda} \left(\frac{e^{\lambda}-e^{\frac{m}{T}\lambda}}{e^{\lambda}-1} \right)^{R-1} dm = \left(\frac{T}{\lambda} \right)^2 \frac{(e^{\lambda}-1)^{R+1}}{R(R+1)} \tag{D.23}$$

Así, puede concluirse que

$$E(M) = \frac{T}{\lambda} \frac{e^{\lambda}-1}{R+1} \tag{D.24}$$

Y, por tanto, el tiempo medio para recibir un NAK es de

$$Tiempo \; medio \; de \; respuesta = \frac{T}{\lambda} \frac{e^{\lambda}-1}{R+1} + c \tag{D.25}$$

Es evidente que las ecuaciones (D.20) y (D.25) son más complicadas de manejar que sus homólogas para el caso de temporizadores uniformes. Sin embargo, es posible simplificar enormemente el manejo de los temporizadores exponenciales si se define un número de respuestas deseadas *N*, tal que

$$N = E(X_i) \approx R\frac{e^{\lambda\frac{2c}{T}}-1}{e^{\lambda}-1} + e^{\lambda\frac{2c}{T}} \; si \; T \geq 2c \tag{D.26}$$

Despejando *T* se obtiene la fórmula

$$T = \frac{2\lambda c}{\ln(R + N(e^{\lambda}-1)) - \ln(R + e^{\lambda}-1)} \tag{D.27}$$

En esta fórmula se obtiene un valor de T que viene determinado por el retardo c y el parámetro λ. Con objeto de encontrar un valor para λ y determinar T de manera exacta, debe derivarse E(X) en función de λ y buscar el valor de λ que minimiza el número de respuestas medias recibidas:

$$\frac{d}{d\lambda}E(X)=0 \Rightarrow \frac{2c}{T}e^{\frac{2c}{T}\lambda}\left(e^{\lambda}-1\right)^2 = R\left[e^{\lambda}\left(e^{\lambda\frac{2c}{T}}-1\right)-\frac{c}{T}e^{\frac{c}{T}\lambda}(e^{\lambda}-1)\right] \quad \text{(D.28)}$$

La ecuación (D.28) no es lineal, y ha de resolverse mediante métodos no lineales. Según se indica en [Nonnen98a] y [Nonnen98b], interpolando es posible obtener la siguiente expresión:

$$\lambda = \ln(R)+1 \quad \text{(D.29)}$$

Y, por tanto,

$$T = \frac{2(\ln(R)+1)c}{\ln(R+N(e^{\ln(R)+1}-1))-\ln(R+e^{\ln(R)+1}-1)} \quad \text{(D.30)}$$

Suponiendo que R es mucho mayor que N, podemos simplificar y obtener que

$$T \approx \frac{2c}{\ln(N)}\left(\ln(R)+1\right) \quad \text{(D.31)}$$

Con estas expresiones de T y λ, basta con definir un número de respuestas deseadas N, y conocer el valor del retardo c para determinar unívocamente los parámetros del temporizador exponencial. El tiempo medio de respuesta, una vez ajustados T y λ, queda:

$$\textit{Tiempo medio de respuesta} \approx \frac{2c}{\ln(N)}\frac{Re-1}{R+1}+c \quad \text{(D.32)}$$

Nuevamente, debido a que R es mucho mayor que 1, es posible hacer:

$$\textit{Tiempo medio de respuesta} \approx \frac{2c}{\ln(N)}e+c \quad \text{(D.33)}$$

Esta aproximación muestra que el tiempo medio de respuesta depende de la inversa del logaritmo del número de respuestas deseadas. En [Nonenn98a] existen gráficas[1] detalladas, calculadas sin aproximaciones, que muestran como, para un millón de receptores (R=1.000.000) es posible reducir el número de respuestas deseadas a sólo 4 (N=4), y que el retardo se encuentre por debajo de 8c (es decir, 4 veces el RTT). Incrementando N a 10, el retardo disminuye por debajo de 3 veces el RTT. Además, al depender λ del logaritmo de R, la magnitud del error cometido al estimar el número de receptores se traslada de forma logarítmica al retardo y al número medio de respuestas. Los temporizadores exponenciales son mucho menos sensibles que los uniformes a errores de estimación de R y a cambios bruscos en el número de receptores.

Es posible concluir, por tanto, que el uso de temporizadores exponenciales tiene las siguientes ventajas:

[1] Las gráficas de [Nonnen98a] se han calculado asumiendo un modelo ligeramente distinto, donde el retardo entre receptor y emisor es c, y el retardo entre dos receptores cualesquiera es también c. Las conclusiones de este modelo son muy similares a las del usado en este trabajo, aunque, es necesario multiplicar c por 2 antes de trasladarlas al modelo aquí empleado.

- Permite un número medio de respuestas recibidas muy bajo sin incrementar significativamente el retardo, incluso para comunicación con cientos de miles o millones de receptores.
- Sólo es necesario ajustar dos parámetros, λ y T, para los cuales existen dos expresiones sencillas.
- No requiere mucha exactitud a la hora de estimar el número de receptores.

APÉNDICE E

ESTUDIO DE PGM, PGMCC Y ORMCC

E.1 INTRODUCCIÓN

En este apéndice, se presenta uno de los protocolos de transporte multipunto fiable, PGM, que más éxito ha tenido. Éste no incluye ningún control de congestión, por lo que se estudiará PGMCC, propuesto por L. Rizzo [Rizzo00], también muy novedoso y referenciado. Como este protocolo utiliza para regular la tasa un mecanismo de ventana deslizante, se describirá otro control de congestión basado en tasa como ORMCC, propuesto por J. Li [Li02], que muestra también ideas originales. Estos protocolos han servido de inspiración para el diseño de RCCMP [Solera03].

E.2 ESTUDIO DE PROTOCOLO DE TRANSPORTE FIABLE PGM

PGM (*Pragmatic Generic Multicast*) [RFC3208] es una iniciativa conjunta de las compañías Microsoft y Cisco para desarrollar un protocolo de transporte multipunto fiable y genérico. La principal novedad de PGM es el uso de los elementos intermedios de red para facilitar la tarea de retransmitir los paquetes perdidos, sin provocar una tormenta de confirmaciones negativas en el emisor. Sus principales objetivos de diseño se pueden resumir en los siguientes puntos:

- Múltiples emisores: PGM ha sido diseñado para permitir la existencia de más de un emisor en cada comunicación. Sin embargo, no garantiza ordenamiento global, es decir, los paquetes recibidos de cada transmisor son ordenados, pero el orden entre paquetes de distintos emisores no se mantiene.
- Fiabilidad: garantiza que un receptor, o recibe todos los paquetes correctamente, o es capaz de detectar una pérdida de paquete irreparable.

- Gestión de grupos: no ofrece mecanismos para gestión de grupos, ni limita la pertenencia a las sesiones, por lo que los grupos de PGM son totalmente abiertos y anónimos.
- Simplicidad: la especificación de PGM tiene como objetivo garantizar la mayor simplicidad posible sin perder funcionalidad.
- Control de congestión: la especificación de PGM no incluye ningún algoritmo específico de control de congestión; sin embargo, si define los mecanismos que deben usar los receptores y los nodos intermedios para comunicar situaciones de congestión al emisor. Estos mecanismos, para ser eficientes, requieren que los nodos intermedios sean compatibles con PGM.

Debido a la falta de un algoritmo específico de control de congestión, así como de la dependencia que tienen los receptores con los nodos intermedios para indicarla, se han sugerido numerosos esquemas de control de congestión para PGM. En los siguientes apartados se analizarán los esquemas PGMCC y ORMCC.

E.2.1 La colaboración de los nodos intermedios en PGM

Para evitar la sobrecarga de NAKs, PGM requiere de la colaboración de los elementos intermedios de red. El proceso que se sigue es el siguiente.

Los nodos intermedios necesitan conocer desde que nodo les llegó el paquete de datos. Para ello, cada cierto tiempo, el emisor transmite un paquete de control que es modificado por estos dispositivos, para determinar cuál es el elemento de red más cercano.

En caso de que un receptor detecte que no ha recibido un paquete, solicita enviando un NAK al nodo más cercano, mediante mecanismos punto a punto, la retransmisión de dicho paquete. Cuando el nodo intermedio recibe la confirmación negativa, la reenvía a su nodo más cercano, nuevamente vía punto a punto, mientras que transmite al resto de los receptores que de él cuelgan un aviso de retransmisión solicitada (NFC). Al recibir este aviso, que se distribuye de forma multipunto, los receptores conocen para qué paquetes se ha solicitado la retransmisión, y anulan cualquier petición que tuvieran para esos mismos paquetes.

Esta secuencia de NAKs y NFC se repite nivel a nivel, ascendiendo en el árbol de transmisión hasta que por fin un NAK alcanza al emisor, momento en el cual se produce la retransmisión. Evidentemente, este mecanismo necesita que los nodos intermedios presentes en la ruta hasta el emisor sean compatibles con PGM, perdiendo mucha eficiencia en caso contrario.

La colaboración de la compañía Cisco en la elaboración del protocolo, principal fabricante mundial de encaminadores, asegura que una buena parte de Internet podría ser en el futuro compatible con PGM. Aunque al mismo tiempo, esto puede retraer a otros fabricantes de colaborar en futuros desarrollos de este protocolo.

E.3 ESTUDIO DEL CONTROL DE CONGESTIÓN PGMCC

PGMCC es un esquema de control de congestión para PGM que pretende ser: simple, escalable y equitativo con TCP. Este mecanismo de tasa única se basa en escoger a un representante, encargado de enviar confirmaciones positivas al emisor por cada paquete correctamente recibido, para regular un mecanismo de ventana basada en testigos de comportamiento similar a TCP. A continuación se detallan los aspectos más importantes.

E.3.1 Selección de representante

Para seleccionar al representante, PGMCC compara el resultado de calcular la ecuación de equilibrio de TCP simplificada [Mathis97] entre los diferentes receptores candidatos, y escoge al que menor caudal compute. Los parámetros que incluye la fórmula, retardo de ida y vuelta y la tasa de pérdidas, se calculan igual que en RCCMP.

Al comienzo de la transmisión, para saber qué representante se debe elegir, el emisor recaba de cada receptor tanto la tasa de pérdidas como el mayor número de secuencia recibido. Como en dicho instante todos los receptores tendrán idénticos parámetros, PGMCC siempre escoge a los de menor retardo, puesto que entre dos receptores con idénticas características, PGMCC siempre favorece a aquel cuyos paquetes lleguen antes al emisor.

Cuando un nodo no recibe un paquete, transmite un NAK, que acompaña de sus estadísticas de retardo y pérdidas. Si al calcular la ecuación de equilibrio de TCP, el caudal resultante es por lo menos un veinticinco por ciento peor que el actual, se producirá un cambio de representante. Esto, que favorece ligeramente al representante actual frente a los candidatos, evita que se produzcan cambios constantes debidos a pequeñas fluctuaciones en las condiciones de la red.

La principal novedad de PGMCC es que no interpreta estos cambios de representante como un cambio en las condiciones de la red. La información del nuevo representante se usa para evaluar la situación de la red, sin tomar ninguna decisión a priori. Otros esquemas anteriores a PGMCC asimilaban un cambio de representante con una variación en las condiciones de la red, lo que parecía lógico, puesto que un cambio sólo puede producirse tras un NAK, es decir, tras una pérdida de paquete, pero que provocaba que sobrerreaccionaran y disminuyeran la tasa de emisión más de lo que era necesario.

E.3.2 Modulación de la velocidad de transmisión

Un protocolo de control de congestión multipunto de tasa única no sólo debe proveer los mecanismos necesarios para poder realizar una selección efectiva de un representante entre todos los receptores, sino que también debe cumplir su objetivo

básico, es decir, debe proveer un mecanismo para detectar condiciones de congestión, a través de la información de este receptor, y actuar en consecuencia.

PGMCC usa un mecanismo basado en ventana, que pretende imitar el comportamiento de TCP. En concreto, se genera un número determinado de testigos por cada ACK enviado por el representante; un testigo se consume cada vez que un nuevo paquete es transmitido. Si W es el tamaño de la ventana y T el número de testigos disponibles, se puede resumir el algoritmo en los siguientes pasos:

1. W y T se inicializan a 1.
2. Al recibir un ACK, se aplican las fórmulas:

$$W = W + \frac{1}{W} \tag{E.1}$$

$$T = T + 1 + \frac{1}{W} \tag{E.2}$$

3. Si se detecta una pérdida de paquetes mediante la duplicación consecutiva de ACKs, se reduce el tamaño de ventana a la mitad y se hace caso omiso de los W/2 ACKs siguientes.
4. Por cada paquete transmitido se substrae un testigo hasta que no quede ninguno.
5. Se repite el algoritmo desde el paso 2.

Existen algunas diferencias entre los ACKs usados en PGMCC y los tradicionales de TCP, ya que los primeros no son acumulativos, es decir, recibir un reconocimiento para el paquete enésimo no implica que todos los anteriores a él se hayan recibido correctamente. Esta característica es necesaria habida cuenta de que en PGM, al contrario que en TCP, los paquetes de reparación pueden no transmitirse inmediatamente. Además, para poder tener más información sobre el estado de la red, cada ACK es acompañado de una máscara que informa del estado (recibido o no) de los 32 paquetes anteriores.

Uno de los comportamientos de TCP más difíciles de emular en transmisiones multipunto es lo que se ha venido en llamar apertura exponencial de la ventana. En efecto, cuando al recibir ACKs duplicados TCP detecta pérdidas en la red, disminuye el tamaño de la ventana; sin embargo, cuando se reciben nuevos ACKs, es decir, ACKs de distintos paquetes, comprueba el tamaño de la ventana, y si éste es inferior a un límite calculado según las circunstancias de la red, la ventana se incrementa no de forma lineal, sino de forma exponencial.

PGMCC intenta imitar este comportamiento propio de TCP sin calcular el límite de incremento exponencial dinámicamente, sino asignándole un valor fijo igual a seis paquetes, lo que implica un comportamiento que puede resultar demasiado agresivo o pasivo, según las circunstancias.

E.3.3 Colaboración de los elementos intermedios de red en PGMCC

En PGM, el papel de los elementos intermedios de red es fundamental para lograr una supresión efectiva de los NAKs redundantes. Por otro lado, PGMCC

añade información sobre las estadísticas de cada receptor en las confirmaciones negativas. Por lo que estos paquetes no pueden ser suprimidos sin impedir que llegue información vital al emisor. PGMCC ofrece varios mecanismos para solventar este problema:

- No modificar el comportamiento de los nodos intermedios: esta solución, que parece inadecuada a primera vista, se basa en el principio de que los peores receptores, al menos en cuanto a pérdidas, enviarán en media un mayor número de NAKs, y por tanto, tendrán menos problemas con el filtrado de los nodos intermedios.
- Desechar la supresión de NAKs: si la compatibilidad con PGM se desactiva en los elementos intermedios de red, toda la información llegará sin problemas al emisor, aunque el problema de las tormentas de NAKs, que los nodos intermedios se encargaban de eliminar, volverá a aparecer.
- Modificar el algoritmo de supresión de NAKs: según el algoritmo sugerido por los autores de PGMCC, los nodos intermedios deberían almacenar, por cada NAK recibido, la tasa de pérdidas correspondiente al receptor que lo emitió. Si un futuro NAK para el mismo paquete perdido llega al nodo, éste debe comparar las estadísticas almacenadas con las del NAK que llega, y, en caso de que correspondan a un receptor de peores características, debe dejar pasar este último NAK. Este esquema sólo tiene en cuenta las pérdidas y no el retardo a la hora de realizar filtrado.

Ninguna de estas soluciones es completamente satisfactoria. Si se mantiene el comportamiento de los nodos intermedios PGM sin modificar, entonces, sólo el primer NAK que llegue a un nodo pasará sin ser filtrado. Está solución mantiene la escalabilidad, pero evita el control de congestión, ya que paquetes NAK con información sobre nodos muy congestionados podrían filtrarse, al menos al principio. En caso de existir una gran diferencia entre un receptor y el resto, el primero debería enviar más paquetes, y en media, alguno de ellos debería llegar al emisor, pero esto no está garantizado. Por todo lo anterior, esta solución no se considera compatible con un control de congestión adecuado.

La segunda solución se basa en eliminar el carácter de filtro de los nodos, garantizado que todos los NAKs llegan al emisor. Así pues, se garantiza el control de congestión al disponer el emisor de toda la información; sin embargo, el protocolo PGM deja de ser escalable, puesto que no hay forma de evitar las tormentas de NAKs. Éstas podrían ser aceptables, en algunos casos, si no fuera por la sobrerreacción que presenta PGM, que por cada NAK recibido envía un paquete de reparación. Por tanto, al recibir un tormenta de NAKs, el emisor responde enviando una tormenta de paquetes de reparación, lo que no es un comportamiento adecuado en ninguna circunstancia.

La modificación del comportamiento de los nodos, según el algoritmo propuesto, también presenta algunos inconvenientes. Primero, aumenta la cantidad de información a almacenar en los nodos intermedios. Concretamente, se almacena una nueva variable por cada NAK recibido. Aunque no es un incremento desmesurado, significa una carga aún mayor para unos elementos de red, ya bastante cargados en el protocolo PGM. Segundo, al tener en cuenta sólo la tasa de pérdidas y no el retardo, no garantiza que el peor receptor sea seleccionado como representante.

Por ejemplo, en el caso de dos receptores que cuelgan del mismo nodo con la misma tasa de pérdidas y con retardos bien diferenciados. Los NAKs que llegan desde el receptor con menos retardo serán los que sean transmitidos hacia la fuente, mientras que los del peor receptor, el del mayor retardo, serán filtrados. Además, esta solución no garantiza que no se produzcan tormentas de NAKs. En el improbable caso de que todos los receptores tuvieran una tasa de pérdidas diferente, y se generarán los temporizadores con tan mala fortuna, que se enviarán primero los de mejor tasa de pérdidas, y posteriormente los de peor, se produciría una tormenta de NAKs en el emisor. Aunque esta circunstancia es extremadamente improbable, no se producía con el algoritmo original de PGM.

E.3.4 Análisis de la equidad de PGMCC

Una de las características deseables para cualquier esquema de control de congestión en protocolos multipunto fiables es que su actividad no interfiera con la de las comunicaciones punto a punto ya existentes. El comportamiento de PGMCC frente a TCP ha sido ampliamente estudiado en los siguientes trabajos: [Iannac01], [Seada02] y [Seada04]. De estos estudios y de las simulaciones realizadas puede deducirse, que aunque el comportamiento de PGMCC es bastante parecido al de TCP, se observan ciertas diferencias, que se explican porque el algoritmo de PGMCC tiene una gestión inicial de la ventana menos dinámica, utiliza unos ACKs no acumulativos y calcula el temporizador de retransmisión de forma diferente a TCP. De hecho, el trabajo de K. Seada, [Seada02], demuestra que en escenarios de muy altas pérdidas, PGMCC tiene una conducta poco equitativa debido al tratamiento que realiza de las pérdidas, ya que el emisor sigue transmitiendo nuevos paquetes, aunque haya paquetes perdidos que no hayan sido recibidos todavía.

E.3.5 Análisis de la escalabilidad de PGMCC

El código de PGMCC para ns-2, [ns-2], no contiene una subrutina dedicada a los nodos intermedios, por lo que las simulaciones se realizaron con el filtrado de NAKs deshabilitado. Con esta configuración, y habida cuenta de que los temporizadores aleatorios de PGM tampoco funcionan correctamente en el código para ns-2 de PGMCC, los resultados muestran, lógicamente, que no se puede hablar de escalabilidad en el mecanismo de control de congestión.

La topología empleada para estas simulaciones es una topología en árbol de dos niveles, figura E.1, donde la raíz es un emisor PGMCC y hay un total de 20 receptores. Todos los enlaces son de capacidad 500Kbps, 30ms de retardo de propagación y sin pérdidas.

Si la fuente emite a 0.6Mbps, superando la capacidad de los enlaces, se obtiene la gráfica E.2 que muestra el ancho de banda consumido. El tráfico retransmitido, figura E.3, es inferior a 16Kbps.

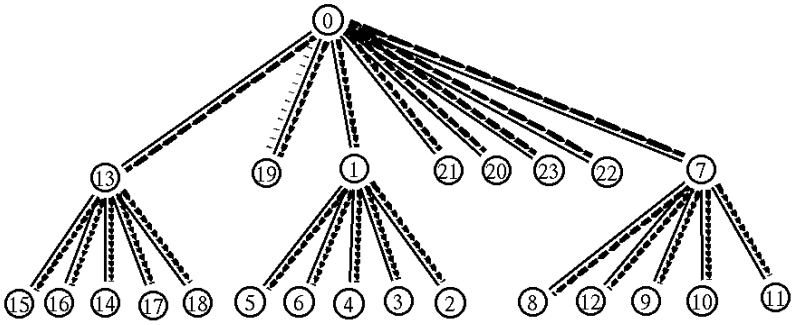

Figura E.1 Topología en árbol con 20 receptores con el recorrido de los paquetes. El nodo 19 es el representante, por lo que existe un tráfico de paquetes ACK desde ese nodo a la fuente.

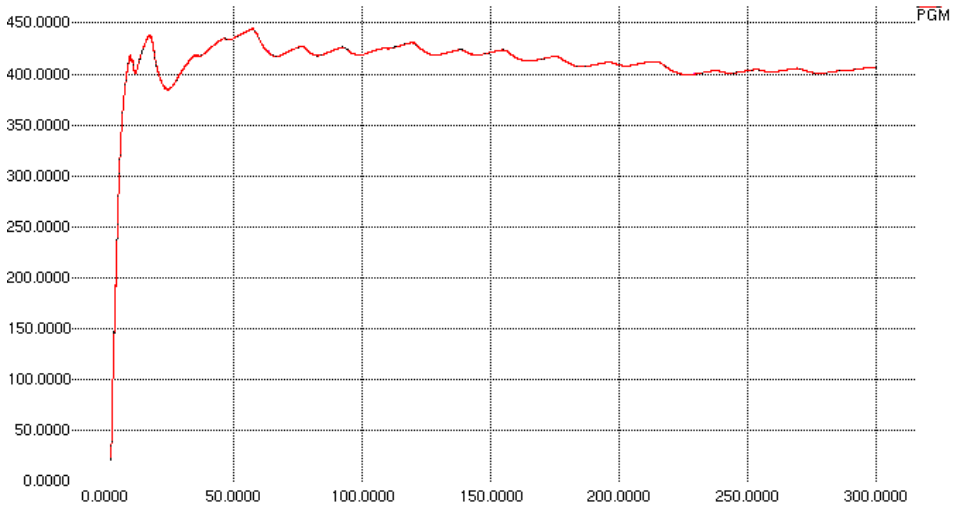

Figura E.2 Ancho de banda empleado total ocupado para 20 receptores.

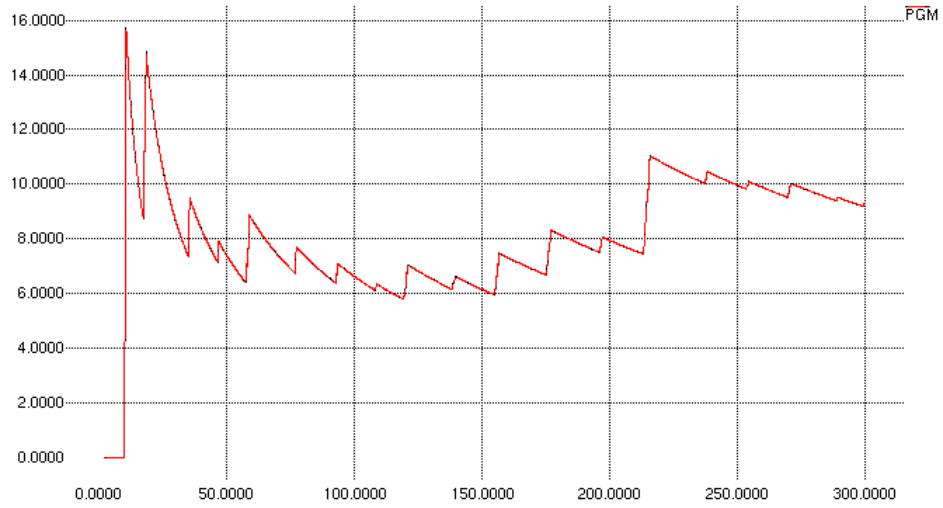

Figura E.3 Ancho de banda empleado para retransmisiones.

Es destacable que al comienzo de la transmisión se escoge como representante siempre a un nodo del nivel más cercano a la raíz, pues todos tienen las mismas estadísticas y, en ese caso, PGMCC favorece a los que llegan primero al emisor. Al abrirse demasiado la ventana, se comienzan a producir pérdidas en los receptores situados en los niveles más profundos del árbol, y se produce un cambio de representante.

De estos experimentos y otros que no se exponen aquí, puede deducirse, obviamente, que el ancho de banda ocupado disminuye conforme aumenta el número de receptores, y, además, un mayor porcentaje de éste es empleado en retransmisiones. ¿Por qué se ve afectado el ancho de banda ocupado para emitir cuando crece el número de receptores?

La principal razón es que al aumentar el número de receptores también aumenta el número de NAKs que llegan al emisor. Esto, en principio, debería influir sólo al canal de subida no al de bajada, pero al reaccionar a cada NAK con una retransmisión, PGM responde a una tormenta de NAKs con una tormenta de retransmisiones. Como PGMCC no aplica mecanismos de control de flujo y de congestión a las retransmisiones, estas tormentas pueden saturar el canal, disminuyendo el rendimiento efectivo de la comunicación, en mayor grado cuanto mayor sea la tormenta.

Además, el hecho de que al principio se escoja como representante a un nodo del primer nivel, que tiene menos retardo, no ayuda a controlar este problema, pues al imponer su ritmo, provoca que los receptores, situados en los niveles más profundos del árbol, sufran la imposición de un flujo de transmisión que sobrepasa sus capacidades.

E.4 ESTUDIO DEL CONTROL DE CONGESTIÓN ORMCC

ORMCC (*Output Rate Multicast Congestion Control*) [Li02] es un esquema genérico de control de congestión, aplicable a cualquier protocolo de transporte punto a multipunto. Como cualquier otro mecanismo unitasa, se basa en escoger un representante y usar su información para controlar, mediante un algoritmo de tasa, el flujo de transmisión. Para seleccionar a este receptor especial, se dispone de una métrica desarrollada especialmente para este mecanismo: la tasa de rendimiento en congestión, TRAC (*Throughput Rate At Congestion*).

E.4.1 Selección del representante

Entre todos los representantes posibles, ORMCC se queda con aquel que en situaciones de congestión presente el peor rendimiento según la media de la TRAC.

ORMCC supone que los receptores son capaces de diferenciar cuando una pérdida de paquete se debe a congestión y cuando a otra razón. Tras producirse una pérdida por saturación de las colas, un receptor debe calcular el máximo flujo que es capaz de soportar en situación de máxima congestión; eso es el valor instantáneo de

TRAC, que se define como U. Para calcular la TRAC promedio, denotada por E(U), se usa la siguiente fórmula:

$$E(U) = (1-\alpha) * E(U) + \alpha * U \quad \text{(E.3)}$$

Por otra parte, para calcular la varianza de la TRAC, expresada por U_σ, se utiliza la siguiente ecuación

$$U_\sigma = (1-\alpha) * U_\sigma + \alpha * \left| E(U) - U \right| \quad \text{(E.4)}$$

La especificación de ORMCC no concreta el valor de la constante α, la forma de calcular cada U, ni algún mecanismo para detectar que pérdidas se deben exactamente a congestión. Sin embargo, la implementación para ns-2 de ORMCC, define el valor de α como 0.05 y un algoritmo para el cálculo de U. En este procedimiento se calcula cada muestra del valor instantáneo de TRAC, como el cociente entre el número de octetos recibidos desde la última vez que se midió este valor, y el tiempo que ha transcurrido entre las dos muestras. También incluye un algoritmo de suavizado, que tiene en cuenta las muestras anteriores cuando se actualiza U; sin embargo, este algoritmo se encuentra deshabilitado por defecto.

Cuando hay que seleccionar un nuevo representante, por ejemplo, al principio de la comunicación, ORMCC solicita a todos los receptores que le envíen su informe de E(U). Al llegar el mensaje de un receptor con un valor peor al del actual representante, se produce un cambio y comienza un tiempo de espera llamado periodo de gracia.

Si durante este periodo llega un informe correspondiente a un receptor con las mismas características que el representante recién seleccionado, se vuelve a producir un cambio de representante. Este periodo permite, por tanto, que ORMCC favorezca entre dos receptores de similares características a aquel de mayor retardo. La duración del periodo de gracia es el doble del máximo retardo de ida y vuelta.

La especificación de ORMCC no establece un criterio claro sobre cuando un representante debe ser desechado por otro de peor retardo y E(U) similar. Según los autores, una de las razones para trabajar con las nuevas métricas U y E(U) es evitar computar las fórmulas de equilibrio de TCP. Sin embargo, en la definición de ORMCC, no especifica un criterio claro para comparar el retardo con la tasa de rendimiento en situación de congestión, quedando ese criterio libre para definirse en implementación. El código sugerido en el artículo no tiene en cuenta la TRAC durante el periodo de gracia: cualquier informe que llegue durante este periodo provoca un cambio de representante.

Con cada paquete transmitido por el emisor se indica el representante actual, su E(U) y U_σ, y se adjunta una marca temporal. Si un receptor observa que su E(U) es peor que la diferencia entre la E(U) y la U_σ del representante, puede enviar un informe de congestión para solicitar ser el nuevo representante. Ese informe de congestión se acompaña de la marca temporal, con lo que el emisor puede estimar el RTT. Esta estimación se utiliza para calcular el periodo de gracia y para controlar la tasa de transmisión.

E.4.2 Modulación de la velocidad de transmisión

ORMCC usa un algoritmo de control de flujo y congestión basado en tasa, que incrementa y disminuye la tasa según las condiciones de la red para imitar el comportamiento de TCP. Concretamente, el algoritmo se puede resumir en los siguientes puntos:

1. Se comienza con una tasa inicial arbitrariamente seleccionada.
2. Se selecciona un representante inicial.
3. Se espera a que transcurra un RTT sin recibir un informe de congestión. Sin embargo, si se recibe un informe de este tipo:
    a. Se analiza qué receptor lo envió:
        i. Si el informe de congestión es del actual representante, se ajusta la tasa al mínimo entre la tasa actual y 0.75 veces la tasa que soporta el actual representante. No se permite una nueva reducción de la tasa hasta que transcurra un RTT.
        ii. Si el informe de congestión corresponde a un receptor distinto al actual representante, se produce un cambio y se adapta la tasa a 0.75 veces la tasa que soporta el nuevo representante. No se permite una nueva reducción de la tasa hasta que transcurra un RTT.
    b. Se vuelve al paso 3.
4. Al transcurrir un RTT sin recibir el informe de congestión, se incrementa la tasa en tamaño del paquete/RTT unidades.

Se puede comprobar, por tanto que es un algoritmo de tasa que usa incrementos aditivos y decrementos multiplicativos, lo que imita tanto el crecimiento como la reducción de la ventana en TCP.

Al contrario que en PGMCC, no se envían confirmaciones positivas desde el representante hacia el emisor, sino confirmaciones negativas, lo que imposibilita detectar si el representante ha sufrido algún problema. Por ello, el emisor ORMCC realiza una medida tanto del tiempo medio entre informes de congestión del representante E(T) como de su varianza T_σ.

Se asume que cuando la tasa de emisión alcanza el valor $E(U)+4*U_\sigma$, se está alcanzando la capacidad máxima del canal del representante, y puede darse una situación de congestión. Para controlar esta situación se inicializa un temporizador. Todos los informes de congestión que lleguen con esa tasa (o superior), se utilizan para actualizar tanto E(T) como T_σ. Si transcurren $E(T)+8* T_\sigma$ segundos sin recibir un informe de congestión, se supone que el actual representante tiene problemas y comienza una ronda para una nueva elección.

E.4.3 Análisis de la equidad de ORMCC

ORMCC es un mecanismo de control de congestión basado en tasa bastante novedoso. Los experimentos realizados demuestran que su comportamiento es equitativo con respecto a TCP, cediéndole normalmente un mayor porcentaje de ancho de banda que el que se reserva para él mismo. En este sentido, los resultados obtenidos coinciden con los expuestos por los autores en [Li02].

E.4.4 Análisis de la escalabilidad de ORMCC

ORMCC es un mecanismo de control de congestión que puede usarse con cualquier protocolo de transporte. Por tanto, esta propuesta no se encarga de gestionar aspectos como el control de flujo o las retransmisiones de paquetes erróneos, lo que implica que el estudio de la escalabilidad queda limitado al estudio de algunos puntos concretos que sí son responsabilidad de ORMCC:

- Elección del representante: si ORMCC ha de considerarse un mecanismo escalable, la selección del representante no debe verse influida por el número de receptores.
- Algoritmo de control de congestión: tampoco debe verse influido por el número de receptores; es decir, debe siempre adaptarse a las circunstancias de la red, independientemente del tamaño del grupo multipunto.
- Supresión de los informes de congestión: para evitar inundar al agente emisor con una tormenta de informes de congestión que bloquearía el canal de subida, ORMCC debe proveer un mecanismo de realimentación que pueda considerarse escalable.

De estos tres aspectos, el segundo es el de más fácil análisis. El algoritmo de control de congestión usa exclusivamente la información del representante para gestionar la velocidad de transmisión; por tanto, si el representante ha sido correctamente seleccionado, el algoritmo de control de congestión funcionará con independencia del número de receptores.

Como se comentó en apartados anteriores, la selección del representante no está completamente especificada en ORMCC, ya que no se detalla un criterio para comparar y elegir a dos receptores que tienen valores similares de TRAC y RTT. En ese sentido, ORMCC adolece de la falta de una fórmula equivalente a la ecuación de equilibrio de TCP que usa PGMCC o RCCMP.

El mecanismo sugerido en el apéndice del artículo [Li02] consiste en que dentro del periodo de gracia, se seleccione como representante siempre al de RTT mayor, sin tener en cuenta la TRAC. La implementación de ORMCC para ns-2 actúa de forma similar, aunque exige que la TRAC sea al menos igual y el RTT mayor para provocar un cambio de representante. Ninguna de estas dos soluciones es adecuada, ya que la primera no tiene en cuenta la TRAC, y por tanto, puede seleccionar como representante a un nodo con muy pocas pérdidas pero un RTT ligeramente superior al del representante actual, mientras que la segunda no permite que un receptor con

un RTT mucho mayor que el representante actual y con una TRAC sólo ligeramente inferior pueda ser considerado a la hora de escoger a un nuevo representante.

Según los diseñadores de ORMCC, el mecanismo de supresión de informes de congestión es una de sus mejores características. Sin embargo, nuestros resultados no concuerdan con los publicados en el artículo. Concretamente, un problema que se presenta en ORMCC es que a la fuente no llega una información actualizada de las condiciones de la red. A continuación, a partir del análisis detallado de un escenario particular se analizará la eficacia del mecanismo de supresión de informes, y se realizarán simulaciones para verificar los resultados.

Se supone que en un conjunto de receptores de características muy similares, uno de ellos sufre una serie de pérdidas de paquete debidas a congestión momentánea, y se convierte en el nuevo representante. Después de ser seleccionado representante, la situación de congestión desaparece, lo que le permite recibir a una tasa mayor. El emisor ORMCC aumentará ligeramente la velocidad de transmisión hasta que alcance la tasa máxima que el representante es capaz de soportar. Esta velocidad de transmisión máxima es enviada al resto de receptores para informar de la TRAC del representante.

Transcurridos unos segundos, se produce una congestión general en la red. Todos los receptores, incluido el representante, ven su TRAC disminuir drásticamente, muy por debajo de la TRAC que ellos conocen del representante -ya que la información de la TRAC no se actualiza constantemente, sólo en caso de pérdidas-, y todos envían un informe de congestión. Este proceso de inundación de mensajes no era necesario, ya que el representante iba a informar de la situación de saturación. Sin embargo, los receptores no pueden saberlo, y como reaccionan inmediatamente, todos envían un informe de congestión.

En estas circunstancias, la supresión de los informes de congestión ha sido muy baja o inexistente. En general, cuando las pérdidas son correladas entre distintos receptores de características muy similares, la supresión de informes de congestión no funciona. Esto se debe a que la TRAC cae para todos los receptores, incluido el representante, pero al no actualizarse inmediatamente la información, los receptores creen que su TRAC está muy por debajo de la del representante y, efectivamente, la TRAC del representante que conocen es superior a la que ellos tienen en ese momento.

Se han realizado experimentos para comprobar el comportamiento del protocolo en el escenario anterior, y se ha constatado que la supresión puede caer hasta porcentajes tan bajos como el 4%. La topología que se ha utilizado es de tipo árbol con 200 receptores distribuidos en tres niveles de profundidad, enlaces con una capacidad media de 500Kbps, colas de tamaño medio de 30 paquetes y un retardo medio de 30ms. En la tabla E.1 se puede comprobar como el porcentaje de supresión es muy bajo cuando los receptores son muy parecidos.

Las características del enlace de cada receptor con el nivel superior del árbol varían con una probabilidad uniforme entre los límites indicados en la tabla. Cuando esta variación no es excesiva, los receptores son todos muy parecidos, y la tasa de supresión va cayendo hasta niveles muy bajos. Sin embargo, al crecer la variación, el número de receptores con características muy similares disminuye, aumentando por tanto el porcentaje de supresión. El ancho de banda medio consumido también disminuye al aumentar la variación, ya que existen enlaces cada vez peores.

Máxima variación de las características del enlace	Porcentaje de informes de congestión suprimidos	Ancho de banda medio del canal de bajada en el peor receptor
±0%	58,59%	470kbps
±0.5%	43,67%	450kbps
±1%	23,20%	450kbps
±1,25%	13,89%	450kbps
±2,5%	4%	440kbps
±5%	6,3%	400kbps
±7,5%	4,72%	390kbps
±10%	4,84%	370kbps
±12,5%	14,11%	350kbps
±15%	14,50%	320kbps
±17,5%	15,21%	300kbps
±20%	21,20%	280kbps

Tabla E.1 Porcentaje de supresión para una topología arbolada de 200 receptores en función del grado de variación de los enlaces.

Número de receptores	Porcentaje de informes de congestión suprimidos	Ancho de banda medio del canal de bajada en el peor receptor
20	76,49%	470kbps
200	58,59%	460kbps
400	30,56%	450kbps
800	30,45%	450kbps

Tabla E.2 Porcentaje de supresión para una topología arbolada de enlaces con las mismas características en función del número de receptores.

Cabe preguntarse por qué no cae la supresión tan significativamente como en otros casos cuando todos los enlaces son idénticos, y por tanto, todos los receptores de un mismo nivel del árbol tienen las mismas estadísticas. En realidad, la supresión (inferior al 60%) es baja, según los resultados que reportan los autores. Sin embargo, para ese caso, lo que ocurre es que un receptor sólo transmite un informe de congestión si la TRAC es inferior, y no igual, a la del representante. Si todos los receptores son iguales, hay muchos casos en los que la TRAC es la misma para todos los receptores de un mismo nivel del árbol, y por tanto, en esas circunstancias la supresión es alta.

Sin embargo, aún para todos los receptores idénticos, a medida que crece el número de receptores el porcentaje de supresión disminuye. En la tabla E.2 se

pueden observar los resultados variando el número de receptores en la topología anterior, pero con enlaces de características iguales.

E.5 COMPARACIÓN DE RCCMP CON PGM, PGMCC Y ORMCC

Los principales problemas que presentan los esquemas de control de congestión son habitualmente:

- Como realizar una supresión adecuada de la información de realimentación redundante.
- Como comparar receptores y escoger al más adecuado como representante.
- Como usar la información de realimentación del representante para regular el flujo de transmisión.

La mayoría de los esquemas de control de congestión se diseñan o en abstracto, sin tener en cuenta ningún protocolo en concreto, o sobre un protocolo de transporte ya existente. Esto provoca que el mecanismo de control de congestión no influya en la información que transmite el protocolo de transporte, y por tanto, que no se pueda aprovechar sus mecanismos de supresión de información de realimentación, lo que le ocurre a PGMCC, y que la gestión de las retransmisiones sea independiente del control de congestión, cuando son aspectos de un protocolo bastante ligados entre sí.

Al diseñar el protocolo de transporte multipunto RCCMP con el control de congestión desde el principio, la información transmitida por el representante se usa para regular la tasa, y también para el mecanismo de fiabilidad. De esta manera, se logra una supresión efectiva de la realimentación, que provoca que sólo se generen NAKs para aquellos paquetes que han sido recibidos por el representante y no por otros receptores.

El problema de qué métricas utilizar para escoger al representante es mucho más simple, ya que sólo atañe al control de congestión, y no a ningún otro bloque funcional de los protocolos de transporte multipunto. Aunque prácticamente cada esquema de control de congestión sugerido incluye sus propias métricas, éstas en general no se apartan de la ecuación de equilibrio de TCP [Mathis97] para poder comparar fácilmente a los receptores entre sí.

ORMCC, en ese sentido, es una excepción, ya que define su propia métrica, la tasa de rendimiento en congestión, TRAC (*Throughput Rate At Congestion*). El análisis realizado muestra algunos posibles problemas relacionados con el peso relativo que ha de tener el RTT con respecto a la TRAC.

PGMCC presenta unas métricas mucho más fáciles de implementar, comprender y comparar, ya que son directamente trasladables a la ecuación de TCP. Sin embargo, el estudio detallado ha mostrado que este esquema presenta un sesgo que favorece, entre dos receptores con las mismas estadísticas a aquel que tenga un menor retardo. Para evitar esto, en RCCMP, se han usado las métricas de PGMCC añadiendo el periodo de gracia definido en ORMCC.

A la hora de controlar el flujo y la congestión, existen dos grandes clases de algoritmos: los de ventana y los de tasa. Aunque probablemente los de tasa son más

eficientes y requieren menos información de realimentación, como es posible deducir del análisis de ORMCC, los algoritmos de ventana se encuentran mejor situados para imitar el comportamiento de TCP. En ese sentido, los resultados de las simulaciones realizadas muestran que el comportamiento de PGMCC se parece bastante al de TCP, mientras que ORMCC suele dejar que TCP ocupe más ancho de banda.

En RCCMP se ha emulado el algoritmo de control de congestión de TCP Reno, lo que permite unos muy buenos resultados en cuanto a como se comparte el ancho de banda con este protocolo punto a punto. Además, la respuesta de RCCMP también se ve favorecida por implementar retransmisiones inmediatas, que permiten que el representante pueda usar ACKs acumulativos, imitando el comportamiento de un receptor TCP.

Otro aspecto a estudiar de un protocolo de transporte multipunto es la escalabilidad. Un protocolo no escalable no puede implementarse en Intenet, ya que afectaría negativamente al tráfico ya existente. Los principales problemas de escalabilidad en los protocolos fiables vienen relacionados con los NAKs, y la posibilidad de que se produzca una sobrecarga de estos paquetes. Las soluciones a estos problemas no corresponden al bloque funcional del control de congestión, aunque pueden verse afectados por este.

El análisis de PGMCC ha determinado que este mecanismo tiene problemas a la hora de escoger al peor receptor como representante, y simultáneamente mantener el mecanismo de supresión de NAKs, basado en la colaboración de los nodos intermedios del protocolo de transporte PGM.

ORMCC fue diseñado para minimizar la información de control de congestión generada en los receptores que no están funcionando como representante. Sin embargo, el estudio sobre la escalabilidad de ORMCC demuestra que al actuar los receptores inmediatamente, aunque no dispongan de estadísticas actualizadas, el comportamiento cuando todos los receptores tienen características similares no es el más deseable. Este problema fue la inspiración para el desarrollo del mecanismo de estadísticas retardadas, que evita que en RCCMP aparezca este problema.

Mediante el uso de temporizadores exponenciales, RCCMP es capaz de controlar el número medio de respuestas y su retardo. Además, los temporizadores exponenciales son perfectamente compatibles con el esquema de control de congestión. Y por otro lado, es un mecanismo que se desarrolla localmente en cada uno de los receptores. Características que no cumple PGM ya que por un lado requiere de la asistencia de los elementos de red, y por otro, la supresión de NAK en los nodos entra en conflicto con el control de congestión.

REFERENCIAS BIBLIOGRÁFICAS

[Adamson02] R. B. Adamson, J. P. Macker, "Quantitative Prediction of NACK-Oriented Reliable Multicast (NORM) Feedback",Proc. IEEE MILCOM 2002. pp. 221-230. Anaheim, California. USA. October 2002.

[Ahn96] J. Ahn, P. Danzing, "Packet Network Simulation: Speedup and Accuracy Versus Timing Granularity", Proc. IEEE Transactions on Networking. Vol 4 (5). pp. 743-757. October 1996.

[Ait04] O. Ait-Hellal, G. Leduc, "TCP Vegas-like algorithm for layered multicast transmission", Proc. CIC 2004. pp. 58-64. Las Vegas. Nevada. USA. June 2004.

[Altman94] E. Altman, F. Baccelli, J. C. Bolot, "Discrete-Time Analysis of Adaptive Rate Control Mechanisms", High Speed Networks and Their Performance. H.G. Perros and Y. Viniotis (Ed.). North-Holland. pp. 121-140. 1994.

[Atwood04] J. W. Atwood, "A Classification of Reliable Multicast Protocols", IEEE Network. pp. 24-34. May/June 2004.

[Barcellos05] M. Barcellos, A. Detsch, "Congestion Control with ECN Support in Poll-based Multicast Protocols", Proc. ISCC. La Manga del Mar Menor. Spain. June 2005.

[Bhatta99] S. Bhattacharyya, D. Towley, J. Kurose, "The Loss Path Multiplicity Problem in Multicast Congestion Control", Proc. IEEE INFOCOM 99. Vol. 2. pp. 856-863. New York. USA. March 1999.

[Billhartz97] T. Billhartz, J.B. Cain, E. Farrey-Goudreau, D. Fieg, S. Batsell, "Performance and resource cost comparisons for the CBT and PIM multicast routing protocols", IEEE Journal on Selected Areas in Communications 15(3). 1997.

[Bolliger99] J. Bolliger, U. Hengartner, T. Gross, “The Effectiveness of End-to-End Congestion Control Mechanism”, Technical Report 313. ETH Zürich. February 1999.

[Bolot90] J.C. Bolot, A.U. Shankar, “Dynamical Behavior of Rate-based Flow Control Mechanisms”, ACM SIGCOMM Computer Communication Review. Vol. 20(2). pp. 165-170. 1990.

[Bolot96] J. Bolot, A. Vega-García, “Control mechanisms for packet audio in the Internet”, Proc. IEEE INFOCOM96. San Francisco. USA. March 1996.

[Bonomi95] F. Bonomi, D. Mitra, J.B. Seery, “Adaptative Algorithms for Feedback-Based Flow Control in High-Speed”, Wide Area ATM Networks”, IEEE Journal on Selected Areas in Communications. Vol. 13 (7). pp. 1267-1283. 1995.

[Brakmo94] L. S. Brakmo, S. W. O’Malley, L. Peterson, “TCP Vegas: New Techniques for Congestion Detection and Avoidance”, Proc. ACM SIGCOMM ‘94. pp. 24-35. October 1994.

[Brakmo95] L. S. Brakmo, L. Peterson, “TCP Vegas: End to End Congestion Avoidance on a Global Internet”, IEEE Journal on Selected Areas in Communications. Vol. 3(8). pp. 1465-1480. October 1995.

[Breslau00] L. Breslau, D. Estrin, K. Fall, J. Heidemann, A. Helmy, P. Huang, S. McCanne, K. Varadhan, Y. Xu, H. Yu, “Advances in Network Simulation”, IEEE Computer 33(5). pp. 59-67. 2000.

[Byers00] J. Byers, M. Frumin, G. Horn, M. Luby, M. Mitzenmacher, A. Roetter, W. Shaver, “FLID-DL: Congestion Control for Layered Multicast”, Proc. NGC 2000 on Networked group communication. pp. 71-81. Palo Alto. California. USA. November 2000.

[Cavendish95] D. Cavendish, Y. Oie, M. Murata, H. Miyahara, “Proportional Rate-Based Congestion Control Under Long Propagation Delay”, International Journal of Communication Systems. Vol. 8. pp. 79-89. 1995.

[Cavendish96] D. Cavendish, S. Mascolo and M. Gerla, “Rate Based Congestion Control for multicast ABR traffic”, Proc. IEEE Globecom. Vol. 2. pp. 1114-1118. November 1996.

[Chiu89] D. M. Chiu, R.Jain, “Analysis of the Increase and Decrease Algorithm for Congestion Avoidance in Computer Networks”, Computer Networks and ISDN Systems. Vol. 17. pp. 1-14. 1989.

[Cho97] Y.Z.Cho, M. Y Lee, “An efficient rate-based algorithm for point to multipoint ABR service”, Proc. IEEE Globecom. Vol 2. pp. 790-795. November 1997.

[Chuang98] J. Chuang, M. Sirbu, “Pricing Multicast Communication: A Cost-Based Approach”, Proc. INET’98. Geneva. Switzerland. July 1998.

[Clark90] D.D. Clark, D.L. Tennenhouse, “Architectural Considerations for a New Generation of Protocols”, Proc. ACM SIGCOMM 1990, Computer Communication Review. Vol 20(4). pp. 200-208. September 1990.

[Deering93] S. Deering, “MBONE: The multicast backbone”, CERFNet Seminar. May 1993.

[Deering94] S. Deering, D. Estrin, D. Farinacci, V. Jacobson, C. G. Liu, L. Wei, “Protocol independent multicast (PIM): Sparse mode protocol specification”, Internet Draft. March 1994.

[Deering96] S. Deering, D. Estrin, D. Farinacci, V. Jacobson, C. Liu, L. Wei, “The PIM architecture for wide-area multicast routing”, IEEE/ ACM Transactions on Networking. Vol. 4(2). pp. 153-163. 1996.

[Diot97] C. Diot, W. Dabbous, J. Crowcroft, “Multipoint Communications: A Survey of Protocols, Functions, and Mechanisms”, IEEE Journal on Selected Areas in Communications. Vol. 15 (3). pp. 27-90. April 1997.

[Dupuy90] A. Dupuy, J. Schwartz, Y. Yemini, D. Bacon, “NEST: A Network Simulation and Prototyping Testbed”. Comm. ACM. pp. 64-74. October 1990.

[Elwalid95] I. Elwalid, “Analisis of Rate-Based Congestion Control for High-Speed Wide-Area Networks”, Proc. IEEE ICC’95. pp. 1943-1948. Seattle. 1995.

[Estrin99] D. Estrin, M. Handley, J. Heidemann, S. McCanne, Y. Xu, H. Yu, “Network Visualization with the VINT Network Animator Nam”, Tech. Report 99/703. Dept. Computer Science. University of Southern California. Los Angeles. 1999.

[Fendick91] K.W. Fendick, M. A. Rodrigues, A. Weiss, “Analisis of a Rate-based Control Strategy with Delayed Feedback”, IEEE Communications Magazine. pp. 74-82. October 1991.

[Floyd95] S: Floyd, V. Jacobson, S. McCanne, C. Liu, L. Zhang, “A Reliable Multicast Framework for Light-weight Sessions and Application Level Framing”, Proc. ACM SIGCOM 1995. Cambridge, Massachusetts. USA. August 1995.

[Floyd99] S. Floyd, K. Fall, “Promoting the Use of End-to-End Congestion Control in the Internet”. IEEE/ACM Transactions on Networking. August 1999.

[Friedman98] T. Friedman, D. Towsley “Multicast session membership size estimation”. Tech Report 98-32. Universidad de Massachusset. Amherst Computer Science Dept. September 1998.

[Friedman99] T. Friedman, D. Towsley “Multicast session membership size estimation” Proc. IEEE INFOCOM 1999. Vol. 2. pp. 965-972. New York. USA. March 1999.

[Gauthier97] E. Gauthier, J. Le Boudec, P. Oechslin, “SMART : A many-to-many Multicast protocol for ATM”, IEEE Journal on Selected Areas in Communications. Vol. 15(3). pp. 458-472. April 1997.

[Golestani99] S. J. Golestani, K.K. Sabnani, “Fundamental Observations on Multicast Congestion Control in the Internet”, Proc. INFOCOM. vol. 2. pp. 990-1000. New York, USA. March 1999.

[Golmie98] N. Golmie, Y.Saintillan, A. Koenig, D. Su, "NIST ATM/HCF Simulator Version 3.0". March1998; available through http://w3.antd.nist.gov/Hsntg/req_atm-sim.html

[Gross97] M. Grossglauser, K.K. Ramakrishnan, “SEAM: Scalable and Efficient ATM multipoint-to-multipoint multicasting”, IEEE INFOCOM. April 1997.

[Hengar00] U. Hengartner, J. Bolliger, Th. Gross, “TCP Vegas Revisited”, Proc. IEEE INFOCOM. Vol. 3. pp. 1546-1555. March 2000.

[Iannac01] G. Iannaccone, L. Rizzo, “Fairness of a single-rate multicast congestion control scheme”, Proc. International Workshop on Digital Communications. Taormina. Italia. September 2001.

[Izmailov95] R. Izmailov, “Adaptive Feedback Control Algorithms for Large Data Transfers in High Speed Networks”, IEEE Transactions on Automatic Control. Vol. 40(8). pp. 1469-1471. 1995.

[Jacobson88] V. Jacobson, “Congestion Avoidance and Control”, Proc. ACM SIGCOMM’88. pp. 314-329. August 1988.

[Jacobson90] V. Jacobson, "Modified TCP Congestion Avoidance Algorithm”, end2end-interest mailing list. April 1990.

[Jain98] R. Jain, K. Ramakrishnan, D. Chiu, “Congestion Avoidance in Computer Networks with Connectionless Network Layer”, Innovations in Internetworking, Ed. Artech House, October 1998.

[Jiang98a] T. Jiang, E. Zegura, M. H. Ammar, “Improved consolidation algorithms for point-to-multipoint ABR service”, Proc. IEEE Workshop on ATM. pp. 195-201. June 1998.

[Jiang98b] T. Jiang, M. H. Ammar, E. Zegura, “Inter-receiver fairness: a novel performance measure for multicast ABR sessions”, Proc. ACM. pp. 202–211. June 1998.

[Kato97] M.Kato, Y.Oje, M.Murata, H. Miyahara, “Performance Analysis of reactive congestion control based upon queue length threshold values”, Performance Evaluation. Vol.29, nº 2. March 1997.

[Kasera00] S. Kasera, S. Bhattacharyya, M. Keaton, D. Kiwior, J. Kurose, D. Towsley, S. Zabele, “Scalable Fair Reliable Multicast Using Active Service”, IEEE Network. Vol. 14(1). pp. 48-57. January/February 2000.

[Keshav88] S. Keshav, “REAL: A Network Simulator”, Tech. Report 88/472. University of California. Berkeley. 1988.

[Keshav91] S. Keshav, “A Control-Theoretic Approach to Flow Control”, Proc. ACM SIGCOMM Computer Communication Review. Vol. 21(4). pp. 3-15. 1991.

[Kwon03] G. Kwon, J. Byers, “Smooth Multirate Multicast Congestion Control”, Proc. IEEE INFOCOM 2003. pp. 1022-1032. March 2003.

[Kwon04] G. Kwon, J. Byers, “Leveraging Single Rate Schemes in Multiple Rate Multicast Congestion Control Design”, IEEE Journal on Selected Areas in Communications (J-SAC), Special Issue on Design, Implementation, and Analysis of Communication Protocols. Vol. 22(10). pp. 1975-1986. December 2004.

[Levine96] B. N. Levine, J. J. Garcia-Luna-Aceves, “A Comparison of Known Classes of Reliable Multicast Protocols”, Proc. International Conference on Network Protocols. Columbus, Ohio. USA. October 1996.

[Li02] J. Li, S. Kalyanaraman, “ORMCC: A Simple and Effective Single-Rate Multicast Congestion Control Scheme”, in submitted http://www.cs.rpi.edu/ lij6/Research/index.html. July 2002.

[Li03] J. Li, S. Kalyanaraman, “Generalized Multicast Congestion Control: An Efficient Multi-rate Scheme Using Single-rate Control”, Proc. of COST 264/ACM International Workshop on Network Group Communications (NGC 2003) .Vol. 2816. pp. 155-167. Munich. Germany. September 2003.

[Lin96] J. C. Lin, S. Paul, “RMTP: A Reliable Multicast Transport Protocol”, Proc. IEEE INFOCOM 1996. pp. 1414-1424. March 1996.

[Liu03] C. Liu, X. Shan, “Self-suppressed NACK-based Multicast Congestion Control”, Proc. ICT 2003. pp. 456-461. Tahiti. French Polynesia. March 2003.

[Maciá04] G. Maciá, J.E. Díaz-Verdejo, J.M. Estévez-Tapiador, “PIM-DM Cost análisis in Loop Free Topologies”, Proc. ISCC. La Manga del Mar Menor. Murcia. June 2004.

[Macker99] J. Macker, R. Adamson, “The Multicast Dissemination Protocol (MDP) Toolkit”, Proc. IEEE MILCOM 99. October 1999.

[Madhavi97] J. Mahdavi, S. Floyd, “TCP-friendly unicast rate based flow control”. Note sent to end2end-interest mailing list. January 1997.

[Mahanti05] Mahanti, D. Eager, M. Vernon, "Improving Multirate Congestion Control Using a TCP Vegas Throughput Model", Computer Networks Journal. Vol. 48(2). pp. 113-136. June 2005.

[Mathis97] M. Mathis, J. Semske, J. Madhavi, T. Ott, “The macroscopic behaviour of the TCP congestion avoidance algorithm”. Computer Communication Review. Vol 27(3). July 1997.

[Moh97] W. M. Moh, “On multicasting ABR protocols for wireless ART networks”, Proc.International Conference on Network Protocols (ICNP)’97. pp. 24-31. 1997.

[Moh98] W. M. Moh, Y. Chen, B. Niizawa, “Branch-point algorithms for multicasting flow control”, Proc. IEEE Workshop on ATM. May 1998.

[Mukher91] Mukherjee, J.C. Strikwerda, “Analysis of Dynamic Congestion Control Protocols- A Fokker-Plank Approximation”, ACM SIGCOMM Computer Communication Review. Vol. 21(4). pp. 159-169. 1991.

[Nonnen98a] J. Nonnenmacher, E. W. Biersack, “Optimal multicast feedback“, Proc. IEEE INFOCOM 1998. San Francisco. March 1998.

[Nonnen98b] J. Nonnenmacher, “Reliable Multicast Transport to Large Groups”, Ph. D. dissertation. EPFL. Laussane. Switzerland. July 1998.

[Nonnen99] J. Nonnenmacher, E. W. Biersack, “Scalable Feedback for Large Groups”, IEEE/ACM Transactions on Networking. Vol. 7(3). pp. 375-386. June 1999.

[ns-2] Ns development team, “Ns manual”, http://www.isi.edu/nsnam/ns/ns-documentation.html

[Obraczka98] K. Obraczka, “Multicast Transport Protocols: A Survey and Taxonomy”, IEEE Communications Magazine. pp. 94-102. January 1998.

[Ohsaki95] H. Ohsaki, M. Murata, H. Suzuki, C. Ikeda, H. Miyakara, “Rate-Based Congestion Control for ATM Networks”, ACM SIGCOMM Computer Communication Review. Vol. 25(2). pp. 60-72. 1995.

[Ott97] T. Ott, J. Kemperman, M. Mathis, “The stationary behaviour of ideal TCP Congestion avoidance” available at ftp://ftp.bellcore.com/pub/tjo/TCPwindow.ps. August 1996.

[Padhye98] J. Padhye, V. Firoiu, D. Towsley, J. Kurose, “Modeling TCP Throughput: A Simple Model and its Empirical Validation”, Proc. ACM SIGCOMM. 1998. pp. 303-314.Vancouver. Canada. September 1998.

[Papa98] C. Papadopoulos, G. Parulkar, G. Varghese, “An Error Control Scheme for Large Multicast Applications”, Proc. IEEE INFOCOM. March 1998.

[Phillips99] G. Phillips, S.Shenker, “Scaling of Multicast Trees: Comments on the Chuang-Sirbu Scaling Law”, Proc. ACM SIGCOM’99. pp. 41-51. Massachusetts. USA. September 1999.

[Pingali93] S. Pingali, D. Towsley, J.F. Kurose, “A Comparison of Source-Initiated and Receiver-Initiated Reliable Multicast Protocols”, Proc. IEEE INFOCOM. San Francisco. California. USA. October 1993.

[Pingali94] S. Pingali, D. Towsley, J.F. Kurose, “A Comparison of Known Classes of Reliable Multicast Protocols”, Proc. International Conference on Network Protocols. pp. 221-230. Nashville, Tennesse. USA. May 1994.

[Ram92] S. Ram, S. Lin, “Selective-repeat-ARQ Schemes for Broadcast Links”, IEEE Trans. on Communications. Vol. 40. pp. 12-19. January 1992.

[Ram95] G. Ramamurthy, Q. Ren, “Analysis of the Adaptative Rate Control for ABR in ATM Networks”, Proc. IEEE GLOBECOM. pp. 1083-1088. 1995.

[Raman98] S. Raman, S. McCanne, “Asymptotic Behavior of Global Recovery in SRM”, Proc. ACM SIGMETRICS 98. pp. 90-99. Madison, Wisconsin. USA. June 1998.

[Ren96] W. Ren, K. Y. Siu, H. Suzuki, “On the performance of congestion control algorithms for multicast ABR service in ATM”, Proc. IEEE ATM Workshop. San Francisco. USA. August 1996.

[Ren98] W. Ren, K. Y. Siu, H. Suzuki, M. Shinohara, “Multipoint-to-multipoint ABR service in ATM”, Proc. Computer Networks and ISDN Systems. Vol. 30(19). pp. 1793-1810. 1998.

[Rhee99] Rhee, N. Balaguru, G. Rouskas, “MTCP: Scalable TCP-like Congestion Control for Reliable Multicast”. Proc. IEEE INFOCOM. Vol. 3. pp. 1265-73. New York, USA. March 1999.

[Ritter95] M. Ritter, “Steady-State Analysis of the Rate-based Congestion Control Mechanisms for ABR Service in ATM Networks”, University of Würzburg, Institute of Computer Science, Research Report Series, Report nº 114. 1995.

[Rizzo00] L. Rizzo. “pgmcc: A TCP-Friendly Single-Rate Multicast Congestion Control Scheme”. Proc. ACM SIGCOMM 2000. pp. 17-28. Stockholm, Sweden. August 2000.

[Roberts94a] L. Roberts, “Enhanced PRCA (Proportional Rate Control Algorithm)”, ATM Forum cont. 94-0735R1. August 1994.

[Roberts94b] L. Roberts, “Rate based algorithm for point to multipoint ABR service", ATM Forum/94-0772R1. November 1994.

[Sabnani85] K. Sabnani, M. Schwartz, “Multidestination Protocols for Satellite Broadcast Channels”, IEEE Trans. on Communications. Vol. 33. pp. 232-240. March 1985.

[Samios03] C. Samios, M. K. Vernon, "Modeling the Throughput of TCP Vegas", Proc. ACM SIGMETRICS 2003. San Diego, USA. June 2003.

[Schwartz96] M. Schwartz, “Broadband Integrated Networks”, Prentice Hall. New Jersey. 1996.

[Seada02] K. Seada, A. Helmy, “Fairness Evaluation Experiments for Multicast Congestion Control Protocols”, Proc. GLOBECOM 2002. pp. 2621-2625. Taipei, Taiwan. November 2002.

[Seada04] K. Seada, A. Helmy, S. Gupta “A Framework for Systematic Evaluation of Multicast Congestion Control Protocols”, IEEE Journal on Selected Areas in Communications. Vol. 22(10). pp. 2048-2061. December 2004.

[Siu95] K.Y. Siu, T. Tzeng, “Congestion Control for multicast service in ATM networks”, Proc. IEEE GLOBECOM. Vol. 1. pp. 310-314. 1995.

[Siu96] K.Siu, H. Tzeng, “Intelligent Congestion Control for ABR Service in ATM Networks”, ACM SIGCOMM. Comp. Commun. Rev. pp. 81-106. October 1996.

[Solera00] M. Solera, S. Sallent, "Analysis of ABR Service for Multipoint Communications in ATM Networks", Proc. PROMS 2000 (Protocols for Multimedia Systems). pp. 285-293. Cracrow. Polonia. October 2000.

[Solera01] M. Solera, I. Barbancho, J. Yufera, S. Sallent, “Estudio de un mecanismo de congestión para comunicaciones punto a multipunto”, Proc. JITEL 2001. pp. 45-50. Barcelona. September 2001.

[Solera03] M.Solera, J. Muñoz, “Protocolo de Transporte Multipunto Fiable con Control de Congestión (PMFCC)”, Proc. JITEL 2003. pp. 105-111. Gran Canaria. September 2003.

[Solera05a] M.Solera, J. Muñoz, S. Sallent, "Reliable Multicast Transport Protocol with a Single-Rate Congestion Control", Proc. IEEE Infocom 2005 Student Workshop. Miami. EEUU. March 2005.

[Solera05b] M.Solera, J. Muñoz, S. Sallent, "RCCMP: Reliable Congestion Controlled Multicast Protocol", Proc. NGI 2005. Roma. Italia. April 2005.

[Solera05c] M.Solera, J. Muñoz, S. Sallent, "RCCMP: A TCP-friendly Reliable Multicast Transport Protocol", Proc. ISCC 2005. La Manga del Mar Menor. Murcia. June 2005.

[Solera05d] M. Solera, S. Sallent, “RVCMP: TCP Vegas-like Congestion Control for Reliable Multicast”, Proc. IEEE GLOBECOM’05. Stº Louis. Missouri. USA. November 2005.

[Stevens95] W. R. Stevens, “TCP/IP Illustrated, Volume 1; The Protocols.” Addison Wesley. Boston. Massachusetts. 1995.

[Tipper90] D. Tipper, M.G. Hlunchyj, “A Dynamic Rate Control Mechanism for Source Coded Traffic in a Fast Packet Network”, IEEE Journal on Selected Areas in Communications. Vol. 9(7). pp. 1003-1012. 1991.

[Towley85] D. Towley, “An Analysis of a Point-to-Multipoint Channel Using a Go-Back-N Error Control Protocol”, IEEE Trans. on Communications. Vol. 33. pp. 282-285. March 1985.

[Towley87] D. Towley, S. Mithal, “A Selective Repeat ARQ Protocol for a Point to Multipoint Channel”, Proc. IEEE INFOCOM. pp. 521-526. March 1987.

[Vicisiano98] L. Vicisiano, J. Crowcroft, L. Rizzo, “TCP-like Congestion Control for Layered Multicast Data Transfer”, Proc. IEEE INFOCOM. pp. 996-1003. San Francisco. USA. March 1998.

[Wang89] J. L. Wang, J. A. Silvester, “Delay Minimization of the Adaptative Go-Back-N ARQ Protocols for Point-to-Multipoint Communication”, Proc. IEEE INFOCOM. pp. 584-593. April 1989.

[Wang91] Y.T. Wang, B. Sengupta, “Performance Analysis of a feedback Congestion Control Policy Under Non-Negligible Propagation Delay”, Proc. ACM SIGCOMM. pp. 149-157. 1991.

[Wang98] H. A. Wang, M. Schwartz, “Achieving Bounded Fairness for Multicast and TCP Traffic in the Internet”. Proc. ACM SIGCOMM. Vancouver, Canada. August 1998.

[Widmer01a] J. Widmer, R. Denda, M. Mauve, “A Survey on TCP-Friendly Congestion Control”, IEEE Network. pp. 28- 37. May/June 2001.

[Widmer01b] J. Widmer, M. Handley, "Extending Equation-based Congestion Control to Multicast Applications", Proc. ACM SIGCOMM 2001. pp. 275-285 August 2001.

[Yin94] N. Yin, M.G. Hluchyj, “On Closed-Loop Rate Control for ATM Cell Relay Networks”, Proc. IEEE INFOCOM. pp. 99-108. Toronto. 1994.

[Yokotani95] T. Yokotani, T. Ichihashi and M. Ishizaka, “Congestion Control of Multi-cast Connections in ATM Networks”, Proc. 20th Conference on Local Computer Networks (LCN'95). October 1995.

Request For Comments (RFC) y otras especificaciones

[RFC791] Defense Advanced Research Projects Agency. Information Processing Techniques Office “Internet Protocol” DARPA Internet Program Protocol Specification. RFC 791. September 1981.

[RFC793] Defense Advanced Research Projects Agency. Information Processing Techniques Office “Transmission Control Protocol” DARPA Internet Program Protocol Specification. September 1981.

[RFC1058] C. L. Hedrick, “Routing Information Protocol”. RFC1058. June 1988.

[RFC1075] D. Naitzman, C. Partidge, S. Deering, “Distance Vector Multicast Routing Protocol”. RFC 1075. November 1988.

[RFC1112] S. Deering, “Host Extensions for IP multicasting”. RFC 1112. August 1989.

[RFC1180] T. Socolofsky, C. Kale, “TCP/IP tutorial”. RFC 1180. January 1991.

[RFC1323] V. Jacobson, R. Braden, D. Borman,” TCP Extensions for High Performance”. RFC 1323. May 1992.

[RFC1584] J. Moy, “Multicast Extensions to OSPF”. RFC 1584. March 1994.

[RFC1889] H. Schulzrine, S. Casner, R. Fredreick, V. Jacobson, “RTP: A Transport Protocol for Real-Time Applications”. RFC1889. January 1996.

[RFC2001] W. Stevens, “TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms”. RFC 2001. January 1997.

[RFC2236] W. Fenner, “Internet Group Management Protocol, Version 2”. RFC 2236. November 1997.

[RFC2328] J. Moy, “OSPF Version 2”. RFC 2328. April 1998.

[RFC2357] Mankin, A. Romanow, S. Bradner, V. Paxson, “IETF Criteria for Evaluating Reliable Multicast Transport and Application Protocols”. RFC 2357. June 1998.

[RFC2414] M. Allman, S. Floyd, C. Partridge. “Increasing TCP’s Initial Window”. RFC 2414. September 1998.

[RFC2581] M. Allman, V. Paxson, W. Stevens. “TCP Congestion Control”. RFC 2581. April 1999.

[RFC3170] B. Quinn, Celox Networks, K. Almeroth, “IP Multicast Applications: Challenges and Solutions”. RFC 3170. September 2001.

[RFC3208] T. Speakman, D. Farinacci, S. Lin, A. Tweedly, “PGM Reliable Transport Protocol Specification”. RFC 3208. December2001.

[RFC3376] B. Cain, S. Deering, I. Koulevas, B. Fenner, A. Thyagarajan “Internet Group Management Protocol, Version 3”. RFC 3376. October 2002.

[RFC3941] B. Adamson, C. Bormann, M. Handley, J. Macker, “Negative-Acknowledgment (NACK)-Oriented Reliable Multicast (NORM) Building Blocks”. RFC 3941. November 2004.

[UNI3.1] “ATM user-network interface version 3.1 specification”, ATM Forum. 1994.

[UNI4.0] “Traffic management Specification version 4.0 specification”, ATM Forum. April 1996.

[UNI4.1] “Traffic management Specification version 4.1 specification”, ATM Forum. March 1999.