UNIVERSIDAD DE CASTILLA-LA MANCHA
ESCUELA POLITÉCNICA SUPERIOR

INGENIERÍA
EN INFORMÁTICA

PROYECTO FIN DE CARRERA

Una Propuesta para Simplificar el Diseño de Encaminadores IP con QoS

Alejandro Martínez Vicente

Julio, 2003

UNIVERSIDAD DE CASTILLA-LA MANCHA

ESCUELA POLITÉCNICA SUPERIOR

Departamento de Informática

PROYECTO FIN DE CARRERA

Una Propuesta para Simplificar el Diseño de Encaminadores IP con QoS

Autor: Alejandro Martínez Vicente
Directores: Francisco José Alfaro Cortés y José Luis Sánchez García

Julio, 2003

Reunido en la fecha el Tribunal evaluador, que más abajo se cita, del Proyecto Fin de Carrera titulado:

_____

_____

presentado por D/Dª

_____

y siendo su/s tutor/es

_____

se otorga la calificación de _____

Y para que así conste, se firma la presente acta en

Albacete a ..... de ..... de 20.....

PRESIDENTE: _____

SECRETARIO: _____

VOCAL: _____

SECRETARIO PRESIDENTE VOCAL

RESUMEN

Durante los últimos años el uso de Internet ha evolucionado notablemente. Se ha pasado de una situación en la que se usaba solamente servicios como la Web o el correo electrónico, a otra muy distinta como es la actual en la que se tiende hacia una Internet multimedia. En este nuevo entorno no es suficiente ofrecer servicio best effort como hasta ahora se ha venido haciendo, sino que será necesario proporcionar garantías a las nuevas aplicaciones de que obtendrán la latencia y el ancho de banda que necesitan.

Para cumplir con estos nuevos requisitos, englobados en lo que se conoce como calidad de servicio o QoS, se han propuesto distintas soluciones, cada una pensada para resolver un problema concreto. Sin embargo, no parece que haya estudios que analicen la interacción entre todas esas soluciones. No hacerlo y llevarlas directamente a los diseños finales de los encaminadores puede desembocar en una complejidad innecesaria. En este proyecto se presentan los primeros resultados de un trabajo más extenso dirigido en esta dirección y que pretende localizar posibles redundancias en el diseño de encaminadores IP.

Los resultados que aquí se incluyen ponen de manifiesto que la eliminación de las redundancias existentes en el tratamiento de QoS que se hace de los paquetes, simplifica el diseño del encaminador sin ocasionar por ello una pérdida de funcionalidad ni de prestaciones. Para llegar a estos resultados se han evaluado diversas configuraciones de encaminador, cada una de ellas con un grado distinto de complejidad y cuyas diferencias se refieren a los mecanismos que incorporan para garantizar QoS. Aunque el estudio, realizado mediante simulación, se ha centrado en un modelo de encaminador concreto, el tratamiento y las conclusiones pueden ser también válidos para cualquier otro en el que se den las mismas circunstancias.

Quiero dedicar este proyecto a toda mi familia, quienes me han formado como persona y han hecho posible que haya llegado hasta este punto. Gracias.

Quiero dedicar también un agradecimiento especial a mis directores de proyecto, José Luis y Francisco, que han colaborado en la realización de este proyecto más allá de lo que su deber les exige. También quiero agradecer la colaboración de José Duato, por su aportación de ideas fundamentales para el desarrollo de este proyecto.

Por último, quiero agradecer también el apoyo de mis compañeros. Este viaje habría sido mucho más duro sin vosotros.

Índice general

Índice de figuras

Capítulo 1

INTRODUCCIÓN

Desde la aparición de Internet, su uso ha sido cada vez mayor. En los años noventa el público en general, más allá de las comunidades científicas, descubrió Internet como una potente herramienta de trabajo y de ocio. Esto ha provocado un crecimiento exponencial del tráfico de Internet, sin que nadie pueda prever cuándo terminará esa tendencia.

Las aplicaciones que popularizaron Internet fueron fundamentalmente la Web y el correo electrónico. También, en menor medida, fueron populares el FTP o las News. Sin embargo, a medida que avanzó la tecnología y el número de usuarios, los servicios de Internet han ido ganando en complejidad e interactividad. Por ejemplo, las páginas Web han pasado de ser texto con alguna imagen a elaboradas presentaciones multimedia, con sonido y animaciones.

Además de este crecimiento de las aplicaciones tradicionales, han surgido nuevas aplicaciones, fundamentalmente orientadas al multimedia, tales como el audio en tiempo real, vídeo bajo demanda o videoconferencia. Estos nuevos usos de Internet están mucho más allá de lo que se había previsto con la definición de las primeras tecnologías. Desgraciadamente, muchas de esas tecnologías, como IPv4, todavía están en uso.

Para el uso previsto de Internet bastaba un servicio best effort, en el que los elementos de la red hacen las cosas lo mejor que pueden, pero no se preocupan de nada más. Sin embargo, las nuevas aplicaciones requieren algo más que ese servicio best effort.

Las aplicaciones multimedia pueden requerir que se les garantice un ancho de banda durante la conexión, como en el caso del vídeo bajo demanda. Algunas otras pueden también requerir que los mensajes tengan una latencia acotada, por ejemplo para dar sensación de tiempo real en una videoconferencia. Otros tipos de aplicaciones pueden

requerir todo lo anterior y además una baja o nula tasa de errores, como la telemedicina.

A estos requisitos que se han presentado se les conoce como requisitos de calidad de servicio o QoS. Para satisfacerlos pueden plantearse fundamentalmente tres estrategias. La primera consistiría en añadir recursos a la red, en este caso Internet, hasta que todo el tráfico fuera tan holgado que se cumplieran todos los requisitos anteriores. Esta alternativa es claramente demasiado costosa.

La segunda alternativa sería dedicar recursos en exclusividad a las conexiones que lo necesiten. Algo similar a la conmutación de circuitos de la red telefónica. De este modo se aseguraría que las conexiones así establecidas gozarían de la calidad requerida, pero ésta vuelve a ser una alternativa demasiado cara.

Por último, se podría intentar aprovechar los recursos de los que se disponen, tratando de utilizarlos del mejor modo posible. Se trataría de dar prioridad a los mensajes que la necesiten, no admitir más tráfico del que se puede soportar y en general gestionar adecuadamente los recursos. Esta alternativa es la más adecuada, pero a la vez la que presenta mayores retos.

Se pueden distinguir dos componentes fundamentales que componen Internet. En primer lugar están los enlaces. Con el descubrimiento de la fibra óptica, los enlaces tienen la velocidad necesaria para soportar las nuevas aplicaciones.

Por otro lado, tenemos los encaminadores[1], encargados de unir las redes que componen Internet. Su principal función es llevar paquetes desde unos enlaces de entrada a otros de salida. Sin embargo, también deben ser capaces de gestionar distintas tecnologías de enlace, proporcionar QoS y participar en algoritmos de encaminamiento distribuidos.

La velocidad de los encaminadores, limitada por la ley de Moore, crece a un ritmo mucho menor que la de los enlaces. Esto obliga a desarrollar nuevas tecnologías para soportar las nuevas aplicaciones.

Por otra parte, las redes que forman Internet pueden clasificarse en tres categorías: las redes de acceso permiten a los hogares y pequeños negocios conectarse a sus ISPs (Proveedores de Servicios de Internet), las redes corporativas conectan decenas de miles de computadores en un campus o corporación, y finalmente el backbone, la columna vertebral de Internet, conecta las redes corporativas y los ISPs. En estas tres categorías los encaminadores juegan un papel fundamental, con distintos retos en cada una de ellas.

[1] A lo largo de este proyecto se utilizará el término encaminador, aunque también es habitual router.

Los encaminadores corporativos, también conocidos como “de campus”, conectan sistemas finales. Al contrario que los encaminadores backbone, donde es prioritaria la velocidad y el coste es secundario, en este tipo de encaminadores se pretende conectar el máximo número de terminales de la forma más barata posible. Además, en estos casos es deseable que haya un soporte para diferentes niveles de servicio, de cara a poder priorizar ciertos tipos de tráfico sobre el resto.

Como se ha puesto de manifiesto, las exigencias demandadas a los encaminadores son cada vez mayores. Para conseguir cumplir con ellas se han planteado diversas líneas de actuación. Así, por ejemplo, se ha propuesto incrementar el número de canales virtuales (CVs), con lo que es más sencillo evitar el HOL Blocking. También se han planteado complejas políticas de calidad de servicio, como asignar un canal virtual a cada flujo. Sin embargo, incrementar el número de canales virtuales implica o bien reducir el espacio de buffer asignado a cada canal virtual con colas a la entrada o a la salida, o bien usar un buffer central que optimice el espacio, pero que debe funcionar a frecuencias de reloj muy altas.

De este modo, parece que para cada problema que ha surgido se han planteado diversas soluciones que, llevadas a la práctica, han dado lugar a soluciones comerciales. Sin embargo, no parece que haya un estudio que examine las interacciones de todas estas soluciones, para ver hasta que punto unas no hacen el trabajo de otras y en conjunto se están realizando tareas de forma redundante, con el desperdicio de recursos que ello conlleva.

El objetivo principal de este trabajo es analizar si una reducción en la complejidad del diseño permite seguir alcanzando niveles de prestaciones similares. Evidentemente reducir complejidad no debe suponer eliminar funcionalidades, y por tanto esta reducción debe estar basada en la eliminación de redundancias.

El contenido de este proyecto fin de carrera es el siguiente: en el Capítulo 2 se hace un resumen del estado del arte en cuanto a encaminadores IP, repasando su evolución y las tecnologías que se han propuesto en este campo. El Capítulo 3 presenta los objetivos del proyecto y la metodología que se ha seguido para llevarlos a cabo. Las modificaciones concretas en el diseño del encaminador se proponen en el Capítulo 4, aunque es en el Capítulo 5 donde se proponen distintas configuraciones a evaluar. En el Capítulo 6 se evalúan las prestaciones de las propuestas. Para terminar, en el Capítulo 7 se presentan las conclusiones obtenidas y se plantea el trabajo futuro.

Capítulo 2

ENCAMINADORES IP

Este capítulo recopila información sobre el estado del arte de los encaminadores. En ella se establecen sus orígenes con la aparición de Internet. A continuación se da una visión general de un encaminador y se establece una clasificación atendiendo al uso. En los puntos siguientes se entra en detalle en la arquitectura de los encaminadores y las tecnologías que han surgido para ellos. Finalmente, se introduce el concepto de Calidad de Servicio (QoS) y cómo influye en el diseño de los encaminadores.

2.1. EVOLUCIÓN DE INTERNET

Los orígenes de Internet se remontan a más de veinticinco años atrás [8], como un proyecto de investigación en redes de conmutación de paquetes. A finales de los años sesenta (1969), en plena guerra fría, el Departamento de Defensa Norteamericano llegó a la conclusión de que su sistema de comunicaciones era demasiado vulnerable por estar basado en la Red Telefónica Conmutada.

Como alternativa, el Departamento de Defensa, a través de su Agencia de Proyectos de Investigación Avanzados (Advanced Research Projects Agency, ARPA) decidió estimular las redes de ordenadores mediante becas y ayudas a departamentos de informática de numerosas universidades y algunas empresas privadas. Esta investigación condujo a una red experimental de cuatro nodos, que arrancó en diciembre de 1969 y se denominó ARPAnet. La idea central de esta red era conseguir que la información llegara a su destino aunque parte de la red estuviera destruida.

ARPA desarrolló la conmutación de paquetes, cuya principal característica reside

en fragmentar la información en paquetes, cada uno con información suficiente para llegar a su destino. El camino a seguir, sin embargo, no está preestablecido.

La ARPAnet original evolucionó hacia Internet. Internet se basó en la idea de que habría múltiples redes independientes, de diseño casi arbitrario, empezando por ARPAnet como la red pionera de conmutación de paquetes, pero que pronto incluiría otros tipos de red. Hasta ese momento había un solo método para relacionar redes. Era el tradicional método de conmutación de circuitos.

DARPA (el nuevo nombre de ARPA, cambiado en 1972) formalizó tres contratos con Stanford, BBN y UCLA para implementar TCP/IP. El equipo de Stanford produjo las especificaciones detalladas y al cabo de un año hubo tres implementaciones independientes de TCP que podían interoperar.

En los años 80, el desarrollo de las LANs, PCs y estaciones de trabajo permitió que la naciente Internet floreciera. La tecnología Ethernet, desarrollada por Bob Metcalfe en el PARC de Xerox en 1973, es la dominante en Internet, y los PCs y las estaciones de trabajo los modelos de ordenador dominantes. El cambio que supone pasar de una pocas redes con un modesto número de hosts (el modelo original de ARPAnet) a tener muchas redes dio lugar a nuevos conceptos y a cambios en la tecnología.

A finales de los ochenta Internet creció hasta incluir el potencial informático de las universidades y centros de investigación. Esto unido a la posterior incorporación de empresas privadas, organismos públicos y asociaciones de todo el mundo, supuso un fuerte impulso para Internet que dejó de ser un proyecto con protección estatal para convertirse en la mayor red de ordenadores del mundo, con más de quinientos millones de usuarios [22].

Actualmente el tráfico de Internet crece de un modo exponencial y nadie puede prever cuando terminará esta tendencia. Además de crecer en intensidad, también aparecen nuevos tipos de tráfico: donde antes sólo había HTTP, FTP y e-mail, ahora surgen necesidades multimedia, tales como audio en tiempo real, vídeo bajo demanda o la videoconferencia. Estos nuevos tipos de aplicaciones necesitan algo más de la red que el servicio best effort que se ha proporcionado hasta ahora.

Este crecimiento de Internet también ha sido un desafío para los encaminadores. Al principio sólo había un sencillo algoritmo de encaminamiento, pero pronto tuvo que ser sustituido por un modelo jerárquico, en el que cada red se podía encaminar internamente de forma autónoma, mientras que en las fronteras se usaba un protocolo común. Además el crecimiento desmesurado de las tablas de direcciones obligó a la aparición

del CIDR (Classless Interdomain Routing, enrutamiento entre dominios sin clase) para la agregación de direcciones, sustituyendo al viejo esquema de redes clase A, B o C.

Por otro lado, las nuevas aplicaciones pueden requerir algo más que un gran ancho de banda. Además de eso, necesitan algún tipo de garantía de que los recursos que demandan estarán disponibles. A esto se le llaman requisitos de calidad de servicio o QoS. Para satisfacer estos requisitos necesitamos encaminadores más rápidos y enlaces de alta velocidad. Los notables avances en el área de la fibra óptica han mejorado significativamente la velocidad y capacidad de los enlaces. Pero por otro lado, limitada por la ley de Moore, la capacidad de cómputo de los encaminadores crece a un ritmo mucho menor.

Por estos motivos, en lugar de sobredimensionar la red, hoy en día la mejor manera de proporcionar QoS es gestionar eficientemente los recursos de que se disponen. Esto lleva a diseños de los encaminadores cada vez más complejos.

2.1.1. Protocolo IP

El protocolo IP (Internet Protocol) es la base de Internet. En el rfc 791 [7] se define en detalle la versión 4 de este protocolo. Actualmente esa es la versión en uso, aunque la versión 6 está terminada, pero no termina de imponerse.

IP se diseñó como protocolo de red de conmutación de paquetes. IP proporciona transmisión de paquetes, llamados datagramas, desde una fuente a un destino, ambos identificados por direcciones de longitud fija. Además, IP también proporciona fragmentación y reensamblado de datagramas, para transmitir sobre redes con tamaño de paquete pequeño.

El ámbito de IP se limita a proporcionar las funciones para la transmisión de los datagramas. No se proporcionan capacidades de fiabilidad, control de flujo o seguridad. En la pila de protocolos de Internet, son los protocolos por encima de IP los encargados de dar estos servicios, en concreto TCP.

2.2. ENCAMINADORES IP

Los encaminadores son los encargados de unir las redes que componen Internet, haciendo posible verlo como un todo. Su principal función es llevar paquetes desde unos enlaces de entrada a otros enlaces de salida. Sin embargo, también deben ser capaces de

gestionar distintas tecnologías de enlace, programar el uso de servicios diferenciados y participar en algoritmos distribuidos de encaminamiento.

Las redes que forman Internet pueden dividirse en tres categorías [23]: las redes de acceso permiten a los hogares y pequeños negocios conectarse a sus ISPs (Proveedores de Servicios de Internet), las redes corporativas conectan decenas de miles de computadores en un campus o corporación, y finalmente el backbone, la columna vertebral de Internet, conecta las redes corporativas y los ISPs. En estos tres niveles los encaminadores juegan un papel fundamental, con distintos retos en cada uno de ellos.

2.2.1. Componentes de un encaminador genérico

Un encaminador genérico tiene básicamente cuatro partes (Figura 2.1): los puertos de entrada, los puertos de salida, un elemento de conmutación (switch fabric o SF) y un procesador de enrutamiento (routing processor). Un puerto de entrada es el lugar donde se conecta el enlace físico y por donde llegan los paquetes entrantes. El elemento de conmutación conecta los puertos de entrada con los de salida. Los puertos de salida almacenan los paquetes salientes y programan el uso del enlace de salida. Por último, el procesador de enrutamiento participa en los protocolos de enrutado y crea la tabla de reenvío (forwarding table). Se describen a continuación estos componentes con algo más de detalle.

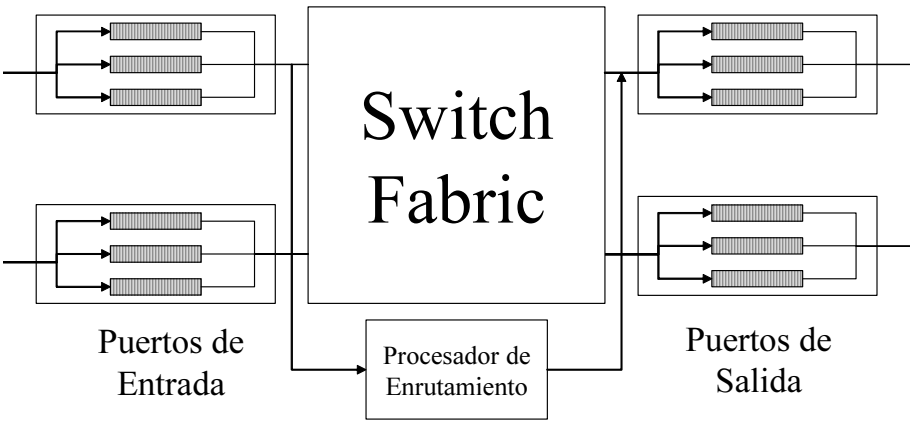


Figura 2.1: Componentes de un encaminador genérico.

Puertos de entrada

Los puertos de entrada proporcionan diversas funcionalidades. En primer, lugar llevan a cabo el encapsulado y desencapsulado de la capa de enlace de datos. En segundo lugar, pueden mirar la dirección destino en una tabla de reenvío para determinar su puerto destino. Este podría ser el primer paso para evitar el HOL Blocking. En tercer lugar, el puerto puede determinar la clase de servicio de los paquetes para proporcionar QoS. En cuarto lugar, el puerto puede que tenga que ejecutar protocolos de nivel de enlace de datos como SLIP o PPP, o incluso de la capa de red como PPTP. Cuando se conoce su destino se envía el paquete al puerto de salida a través del elemento de conmutación.

Elemento de conmutación

El elemento de conmutación se puede implementar de diferentes formas, como veremos en las siguientes secciones. Las más habituales son buses, crossbars, redes multietapa y memorias compartidas. Los buses son simples y conectan todas las entradas con las salidas. Sin embargo, son un recurso compartido que necesita arbitraje y escalan muy mal. Los crossbars permiten varios caminos de datos simultáneos. Sin embargo, su complejidad crece cuadráticamente con el número de entradas que conectan y su velocidad se ve limitada por el planificador del crossbar. Por otra parte, las redes multietapa surgen para resolver los problemas de escalabilidad de los crossbar. Son más lentas pero permiten un mayor número de conexiones. Por último, la memoria compartida puede ser accedida por los puertos de entrada y salida, y sirve para almacenar los paquetes entrantes hasta que son encaminados, siendo su velocidad su punto débil.

Puertos de salida

Los puertos de salida almacenan los paquetes hasta que son transmitidos por el enlace de salida. Pueden implementar algoritmos de programación del enlace para soportar prioridades y garantías de servicio. Estos puertos de salida realizan el encapsulado y desencapsulado de la capa de enlace de datos, al igual que los puertos de entrada. También puede que soporten diversos protocolos de enlace de datos y red.

Procesador de enrutamiento

El procesador de enrutamiento calcula la tabla de reenvío, implementa los protocolos de encaminamiento y ejecuta el software para configurar y gestionar el encaminador. Además, gestiona cualquier paquete cuya dirección no se haya determinado en los puertos de entrada.

2.3. TIPOS DE ENCAMINADORES IP

En la sección anterior se introdujeron los tres tipos de redes que componen Internet. En los siguientes puntos se detallarán los encaminadores usados para cada uno de esos tipos de redes.

2.3.1. Encaminadores del backbone

Internet tiene en la actualidad unas pocas decenas de encaminadores backbone que sirven a miles de redes más pequeñas. Este tipo de encaminadores conectan las redes corporativas, de modo que su coste es compartido entre muchos clientes.

Los principales objetivos de este tipo de encaminadores son fiabilidad y velocidad. Para conseguir la fiabilidad utilizan componentes redundantes, fuentes de energía dobles y caminos de datos duplicados a lo largo del encaminador. Resulta evidente que optimizar el coste no es una prioridad en este tipo de encaminadores.

El principal cuello de botella de rendimiento de este tipo de encaminadores es el acceso a la tabla de reenvío (forwarding table). Cuando se recibe un paquete se comprueba su dirección destino y se consulta la tabla de reenvío para determinar el puerto por el que debe salir. Cada entrada de esta tabla tiene la forma *(dirección/máscara, puerto)*. Una dirección coincide con una entrada de la tabla si los *n* primeros bits de la dirección coinciden, donde *n* es la longitud de la máscara. Para decidir cual es el puerto de salida se elige la entrada que encaje con una mayor longitud de máscara.

Decidir cuál es el puerto destino es costoso por dos motivos: el primero es que la tabla de reenvío puede tener miles de entradas. El segundo es que la dirección de un paquete puede encajar con varias entradas y hay que mirar la longitud de la máscara de cada una. El coste de las búsquedas se incrementa si los paquetes son pequeños y

se dirigen a destinos muy dispares, de modo que se reduce la eficacia de una cache de destinos frecuentes.

En esta breve introducción queda claro que los encaminadores del backbone de Internet presentan varios retos para sus diseñadores.

2.3.2. Encaminadores corporativos

Los encaminadores corporativos, también conocidos como “de campus”, conectan sistemas finales. Al contrario que los encaminadores backbone, donde lo principal es la velocidad y el coste es secundario, en este tipo de encaminadores se pretende conectar el máximo número de terminales de la forma más barata. Además, es deseable que proporcionen soporte para diferentes tipos de servicio.

La mayoría de las redes corporativas están compuestas de segmentos Ethernet conectados por concentradores y puentes. La ventaja de estos dispositivos es que son baratos y fáciles de instalar y configurar. Sin embargo, sus prestaciones se degradan al crecer la red y no proporcionan diferenciación de servicios. Al dividir la red en subredes con el uso de encaminadores también se parten los dominios de colisión de modo que la red escala mejor. Además, la mayoría de los encaminadores soportan algún tipo de diferenciación del tráfico. Sin embargo, los encaminadores suelen ser más caros y difíciles de configurar. El reto de los encaminadores corporativos es tener un bajo coste por puerto, un gran número de puertos, simplicidad de configuración y soporte de QoS.

Por otra parte, los encaminadores corporativos tienen algunos requisitos de diseño adicionales. Al contrario que las redes backbone, las redes corporativas tienen una cantidad significativa de tráfico multicast, que debe ser manejado eficientemente. Por otro lado, los encaminadores corporativos deben soportar otros protocolos además de IP y características de las redes corporativas como cortafuegos, filtros de tráfico, políticas administrativas y de seguridad, VLANs, etc.

2.3.3. Encaminadores de acceso

Las redes de acceso conectan a los consumidores en sus hogares y pequeños negocios con un Proveedor de Servicios de Internet (ISP). Anteriormente, este tipo de redes sólo daba servicio a muchas conexiones telefónicas de baja velocidad vía módem. Actualmente predomina el acceso de alta velocidad mediante ADSL o cable módem. Esto

significa que la carga en los encaminadores de acceso es mucho mayor. Además, ya no es suficiente con la conexión SLIP o PPP, actualmente se demandan servicios de Redes Privadas Virtuales (VPN) mediante PPTP o IPSec. Los retos de los encaminadores de acceso son servir a muchos clientes, en muchos puertos, con un ancho de banda potencialmente grande y distintos protocolos de acceso.

2.4. ARQUITECTURA DE ENCAMINADORES IP

En la sección anterior se examinaron los tipos de encaminadores IP que se pueden encontrar en Internet. En esta sección se hace hincapié en la arquitectura de los encaminadores que son de interés para este proyecto, los de tipo corporativo. En este trabajo se da importancia a este tipo de encaminadores porque son adecuados para probar las mejoras que se propondrán en el Capítulo 4.

Hace ya algunos años los encaminadores se implementaban únicamente con código sobre un computador de propósito general [2]. Para mantener la velocidad del cable eran necesarios un procesador de altas prestaciones y gran cantidad de memoria. Este tipo de conmutador era práctico para velocidades del orden de los 10 Mbps en cada puerto, pero el coste se disparaba a velocidades relativamente bajas como 100 Mbps por puerto.

Afortunadamente la capacidad de integración ha aumentado lo suficiente como para que se puedan implementar encaminadores de un solo chip a un coste bajo, incorporando todo el software y hardware necesario para un rendimiento adecuado.

A lo largo de los siguientes puntos se verán distintas arquitecturas que se han propuesto para los encaminadores IP. Ninguna es mejor que otra porque cada una tiene su propia relación entre rendimiento, funcionalidad y complejidad.

2.4.1. Arquitecturas basadas en bus con un solo procesador

Los primeros encaminadores consistían en un procesador de propósito general y varias tarjetas de interfaz de red conectadas por un bus compartido, tal y como se muestra en la Figura 2.2.

Al llegar un paquete a una interfaz de entrada se analiza en el procesador y se determina la dirección del siguiente salto, enviándose el paquete a la interfaz o interfaces de entrada adecuadas. Los datos esperan a que quede libre su enlace en una memoria

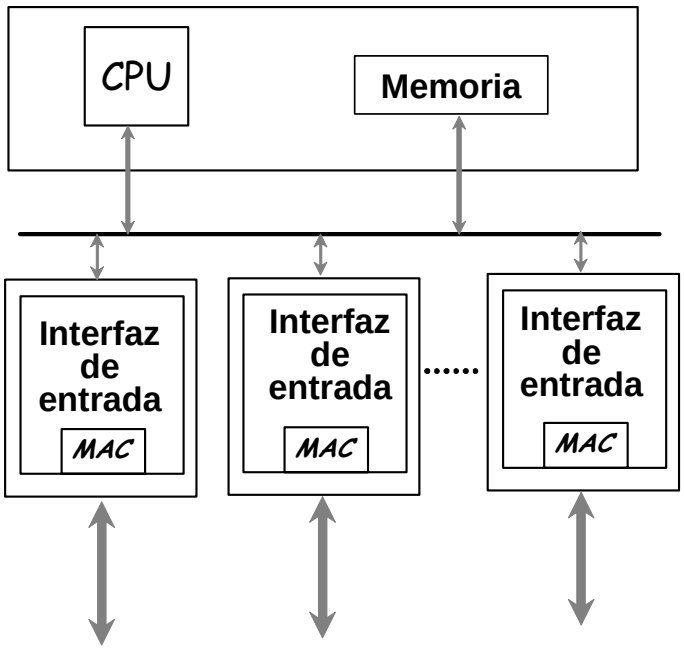


Figura 2.2: Encaminador basado en bus con un solo procesador.

central, lo que implica que los datos cruzan el bus dos veces. Esta arquitectura tiene bajo rendimiento por los siguientes motivos:

- La CPU tiene que procesar todos los paquetes, convirtiéndose en un cuello de botella.
- Algunas tareas en el procesamiento de un paquete implican operaciones intensivas de memoria (mirar la tabla de reenvío, por ejemplo).
- Mover los datos de una interfaz de entrada a otra consume mucho tiempo, a veces más que el tratamiento de la cabecera.

Para mejorar el rendimiento de esta arquitectura se propuso el uso de una cache en la memoria con las últimas traducciones dirección *origen/dirección destino*, aprovechando cierta localidad temporal en las direcciones. A pesar de esto, esta arquitectura sigue teniendo serios problemas para escalar. Un ejemplo de encaminador con esta arquitectura es el DECNIS 500/600 [5], del año 1993.

2.4.2. Arquitecturas basadas en bus con varios procesadores

En la siguiente generación de encaminadores IP se mejora la arquitectura anterior distribuyendo las operaciones de reenvío de los paquetes. Se pueden distinguir dos estrategias distintas a la hora de implementar este tipo de encaminador: arquitecturas con cache de encaminamiento y arquitecturas con varios motores de encaminamiento paralelos. A continuación se verá más detenidamente cada una de estas opciones.

Arquitecturas con cache de encaminamiento

En esta arquitectura se añade una cache de direcciones frecuentes en cada interfaz de entrada, reduciendo el número de operaciones del bus y liberando de algún trabajo a la CPU. Esta estructura se muestra en la Figura 2.3. Cuando llega un paquete se mira primero en la tabla de la interfaz de entrada donde llegó, si hay éxito ya no hace falta consultar a la CPU. Si no estaba la entrada correspondiente, se envía el paquete a la CPU y cuando se determina la dirección de salida, se actualiza la cache de la interfaz de entrada.

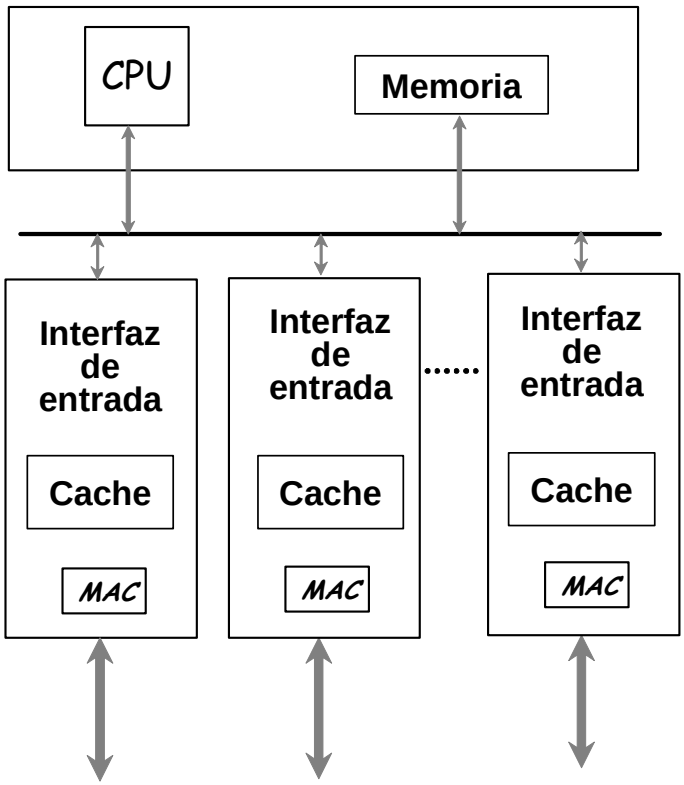


Figura 2.3: Reducción del trabajo del bus con caches en las Interfaces.

El principal inconveniente de esta arquitectura es que el bus y la CPU siguen

siendo, aunque en menor medida, un cuello de botella. Además, la eficacia de este tipo de encaminadores depende del patrón del tráfico.

Arquitecturas con varios motores de encaminamiento paralelos

Otra arquitectura basada en bus es la que se muestra en la Figura 2.4. En ella se conectan varios motores de encaminamiento paralelos para mejorar el tiempo de procesamiento de los paquetes.

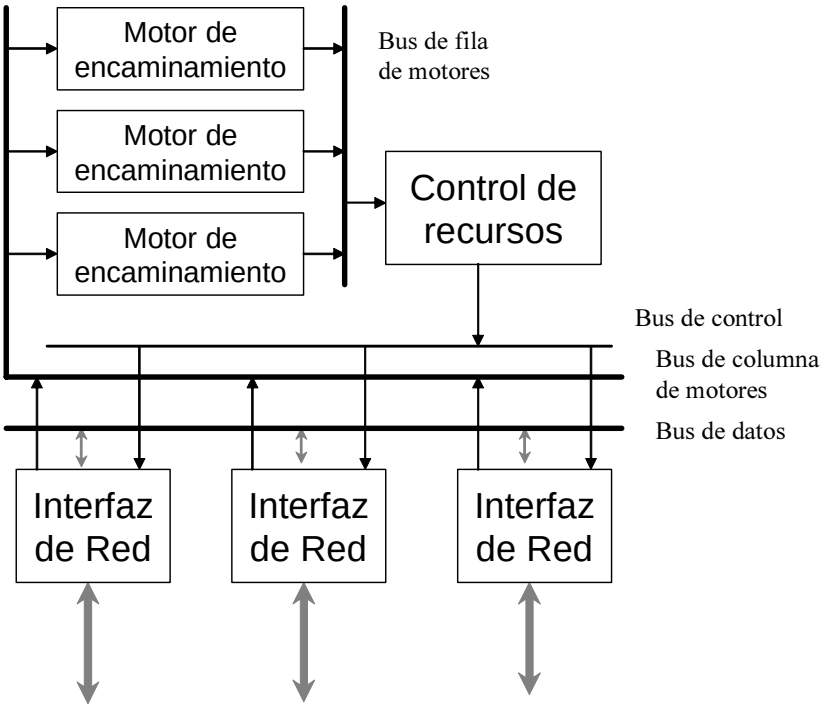


Figura 2.4: Varios motores de encaminamiento paralelos.

Cuando un paquete llega a una interfaz de entrada se separa la cabecera IP del resto del paquete y se envía a un motor de encaminamiento. Mientras el motor encamina la cabecera, el resto del paquete se guarda en un buffer en la interfaz de entrada. Cuando se determina el enlace de salida, se envían a la interfaz apropiada tanto la cabecera como los datos, y se almacenan en un buffer de salida.

Los motores de encaminamiento trabajan en paralelo sobre distintas cabeceras, pero el orden de los paquetes se preserva aplicando los algoritmos adecuados. Las ventajas de esta arquitectura son el incremento de la productividad al usar varios motores en paralelo y la reducción de transmisiones innecesarias al enviar sólo la cabecera a los motores. Un encaminador ejemplo es el que propuso C. Partridge [10] en 1998.

2.4.3. Arquitecturas basadas en conmutador con varios procesadores

En una tercera generación de encaminadores IP se sustituyó el bus por un elemento de conmutación o switch fabric, eliminando el cuello de botella de la conexión entre las interfaces (Figura 2.5).

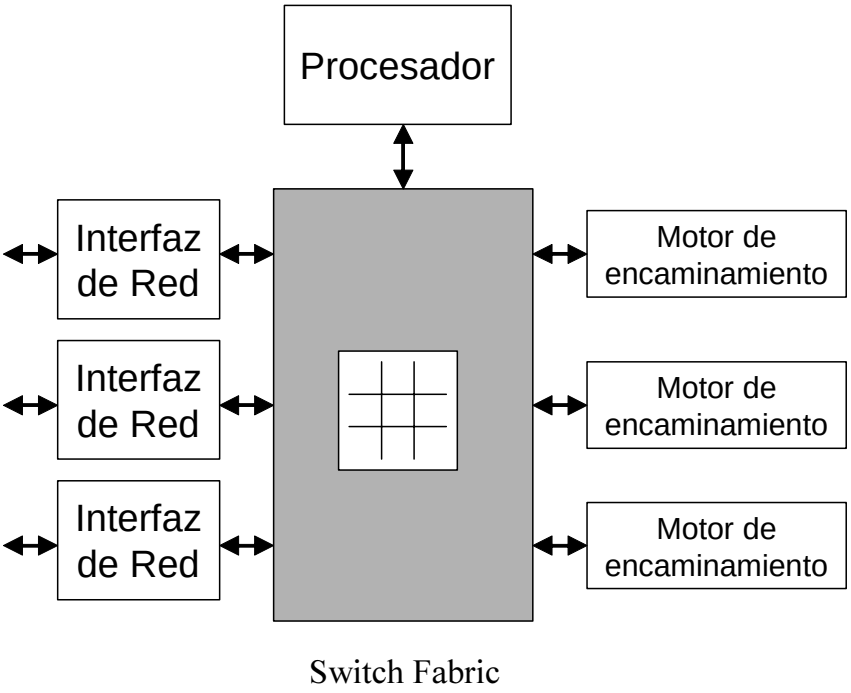


Figura 2.5: Arquitectura de encaminador basada en conmutador con varios procesadores.

Cuando un paquete llega a una interfaz, se separa su cabecera y se envía a un motor de encaminamiento. El resto queda en un buffer de la interfaz de entrada. El motor determina la dirección de salida y devuelve la cabecera actualizada a la interfaz de entrada, que recompone el paquete y lo envía a la interfaz de salida.

Además de estos elementos, también hay un procesador de control para la gestión de las tablas de encaminamiento y los enlaces. Los motores de encaminamiento tienen caches de la tabla general, en caso de fallo deben hacer una consulta a la tabla maestra.

2.4.4. Arquitecturas basadas en conmutador con procesadores totalmente distribuidos

Una forma adecuada de reducir los cuellos de botella de la memoria y el procesador es añadir capacidad de procesamiento a cada interfaz de red, aumentando el rendimiento global. Una arquitectura del encaminador modular de este tipo se muestra en la Figura 2.6

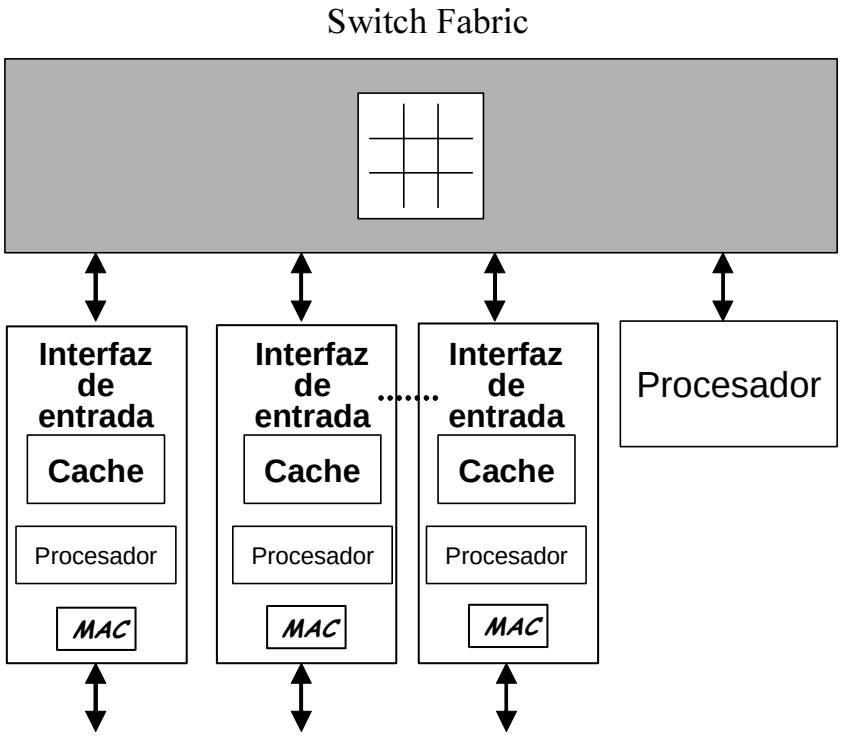


Figura 2.6: Arquitectura de encaminador basada en conmutador con procesadores totalmente distribuidos.

Cada interfaz de red tiene suficiente capacidad de procesamiento y espacio de buffer para procesar todos los paquetes que le llegan. Para proporcionar garantías de servicio, un puerto puede tener que clasificar los paquetes entrantes en clases de servicio. Además, es posible que el puerto también tenga que ejecutar protocolos de enlace de datos o de red. Las interfaces de red se conectan mediante un conmutador de altas prestaciones. Por otra parte, se usa una CPU para realizar algunas tareas centralizadas. Como resultado de esto, la capacidad total de procesamiento y de buffer está distribuida entre las interfaces y la CPU. El Avici TSR [6] utiliza esta arquitectura.

2.5. TECNOLOGÍA DE ENCAMINADORES IP BASADOS EN CONMUTADOR

En esta sección se incluye una descripción más detallada del encaminador basado en conmutador, identificando los elementos que lo forman y las tecnologías principales que se han propuesto para ellos.

2.5.1. Input Adapter

Los puertos de entrada (Input Adapters o IAs) realizan la función de la capa 1 del modelo OSI interconectando los enlaces físicos de distintos tipos, por lo que deben ser conformes a dichos estándares. Además, realizan la desencapsulación del nivel de enlace de datos. Los enlaces de entrada también contribuyen a mantener la información de encaminamiento enviando directamente al procesador los paquetes de control, como RIP, OSPF o IGMP.

En algunos encaminadores, los puertos de entrada mantienen una copia de un subconjunto de la tabla de encaminamiento para permitir la conmutación descentralizada y evitar que las búsquedas en la tabla sean un cuello de botella. Sin embargo, los puertos de entrada con capacidades limitadas simplemente envían todos los paquetes al procesador central para que los encamine.

2.5.2. Elemento de conmutación

El elemento de conmutación (o switch fabric) es el responsable de transferir paquetes entre otros elementos del encaminador, normalmente paquetes de usuario desde los puertos de entrada hacia los de salida. En el diseño del elemento de conmutación hay que tener en cuenta cuestiones como la multidifusión, la tolerancia a fallos, y las prioridades al retrasar o descartar paquetes [2]. El elemento de conmutación trata de maximizar la productividad y minimizar la latencia y la tasa de fallos de un tráfico de entrada dado. Se pueden distinguir varios tipos de switch fabric, como se verá en los siguientes puntos.

SF de medio compartido

Un medio compartido permite la transferencia de paquetes, ya sea éste un bus sencillo, un anillo o un bus doble. La opción más simple es el bus, un único medio por el que todo el tráfico entre los módulos debe pasar. Los datos se transmiten usando la multiplexación por división del tiempo o TDM, donde a cada módulo se le proporciona un slot de tiempo. Sin embargo, los buses están muy limitados en capacidad y por la sobrecarga del arbitraje necesario. Esto hace que esta arquitectura sea incapaz de llegar a grandes velocidades.

En la Figura 2.7 se muestra un ejemplo del uso de un bus compartido mediante TDM. Los paquetes entrantes son puestos en el bus siguiendo un orden, por ejemplo round robin. En cada salida se comprueba si los paquetes son destinados a ella mirando algún tipo de etiqueta. Los paquetes pasan a los buffers de salida de los puertos.

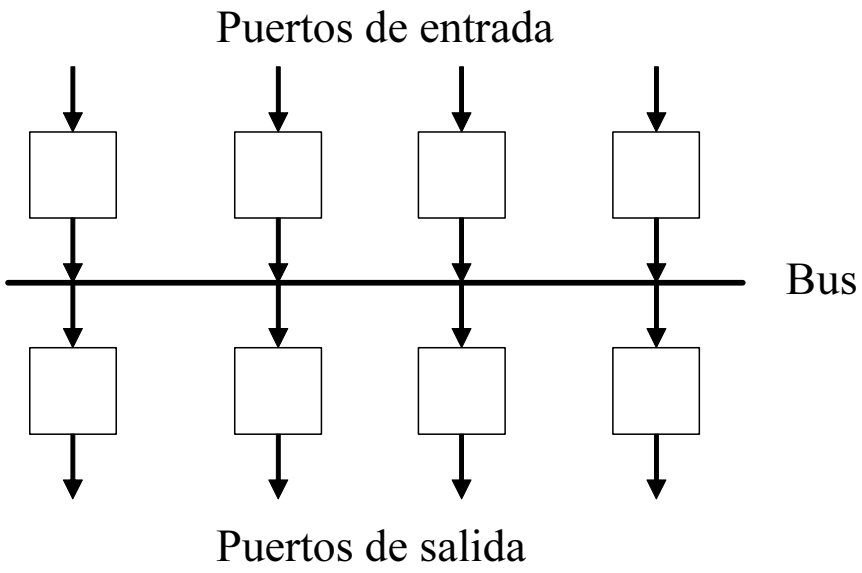


Figura 2.7: Bus de uso compartido.

En un bus que soporte N puertos de entrada y N puertos de salida, cada uno a una velocidad de V paquetes por segundo, el bus debe operar a $N \times V$ paquetes por segundo para no generar conflictos por el ancho de banda. Si va más despacio, harán falta colas a la entrada para absorber los retrasos.

La principal ventaja de usar un bus como elemento de conmutación es que el diseño de los distintos módulos del encaminador se simplifica. El bus proporciona un medio adecuado para la comunicación, donde la multidifusión es algo natural. Por estos motivos este tipo de arquitectura se ha usado mucho en distintas implementaciones.

El gran problema del bus es que debe operar a $N \times V$ paquetes por segundo. Además, también los buffers a la salida deben tener esa velocidad. Hay límites físicos que impiden hacer estos elementos tan rápidos como se desearía, imponiendo límites a la escalabilidad de esta aproximación a grandes tamaños y velocidades. Se puede hacer N o V grandes, pero ambos a la vez no. Además, el uso de buffers a la salida, como se verá más adelante, resulta en una baja utilización de los mismos.

SF de memoria compartida

En este tipo de elemento de conmutación se recogen los paquetes entrantes y se ponen en una memoria de acceso aleatorio con dos o más puertos. Los paquetes salientes se leen de la memoria y se ponen en el enlace. Este modo de almacenar los paquetes tiene la misma ventaja que los buffers a la salida, evita totalmente el HOL Blocking y permite maximizar la productividad. Además, la principal ventaja respecto a los buffers a la salida es que con esta aproximación se aprovecha mucho más el espacio de buffer. Estas cuestiones serán mejor tratadas en puntos sucesivos. Estas ventajas hacen que ésta sea una aproximación popular. En la Figura 2.8 se puede ver esta arquitectura.

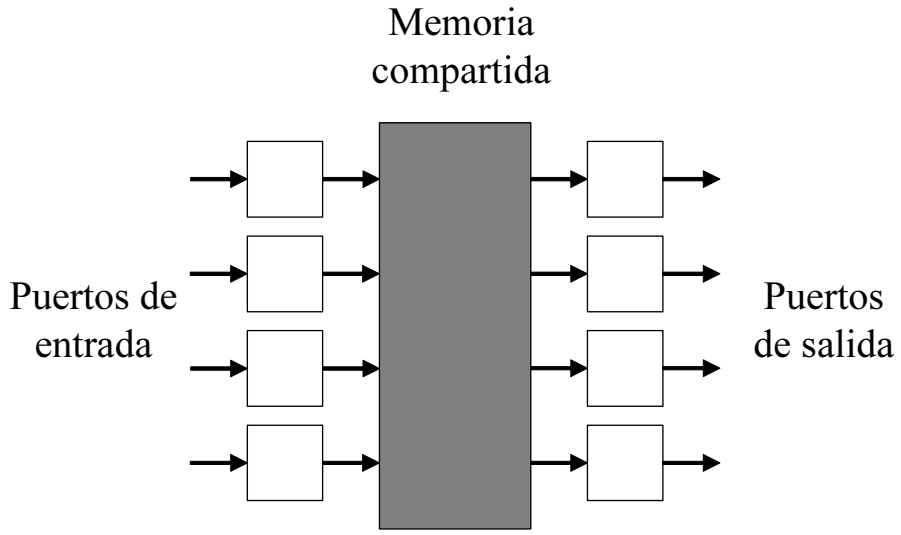


Figura 2.8: SF de memoria compartida.

A pesar de sus ventajas, esta aproximación no es ideal. Como los paquetes se leen y escriben de uno en uno, la memoria debe trabajar al ritmo de la productividad total del encaminador. De esta forma, la memoria debe ser capaz de escribir un paquete en $\frac{1}{N \times V}$ segundos, N veces más rápido que los puertos de entrada. Como las memorias de acceso aleatorio están limitadas en su velocidad, el factor $N \times V$ está acotado, con lo que el encaminador puede tener muchas entradas o puertos rápidos, pero no todo a la

vez. Además, este medio de conmutación requiere un controlador centralizado que opere tan rápido como la memoria. Esto puede ser difícil si tiene que manejar varias clases de prioridad, complicados algoritmos de planificación o multidifusión.

En este tipo de encaminadores la memoria es un punto de fallo único, muy difícil de evitar con una segunda memoria. Esto, unido a todo lo anterior, hace que este medio de conmutación sea más apropiado para sistemas de baja capacidad.

SF distribuido con buffers a la salida

En esta aproximación existen caminos independientes entre todas las N^2 posibles parejas de entradas y salidas. Cuando llega un paquete se difunde por buses separados hacia todos los puertos (Figura 2.9). Filtros de direcciones en cada salida determinan si los paquetes son de esa salida. En ese caso los paquetes pasan a las colas de salida.

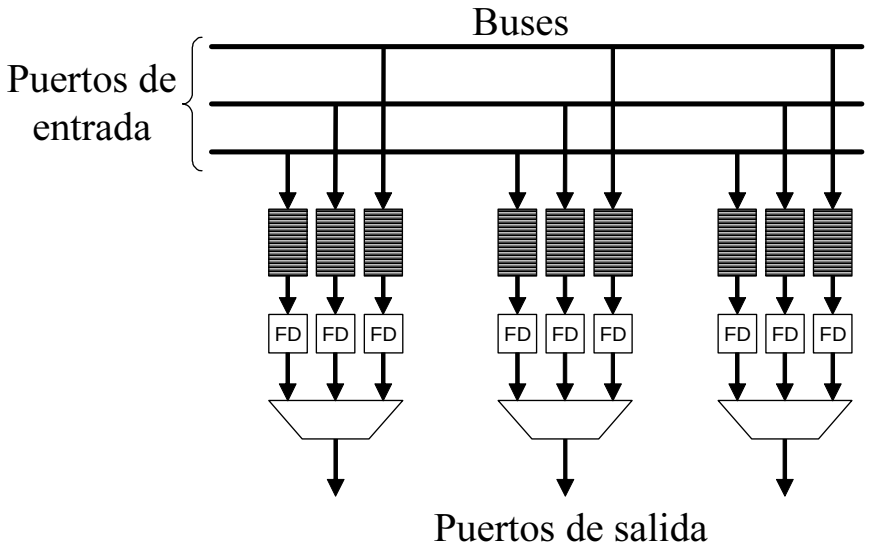


Figura 2.9: SF distribuido con buffers a la salida.

Esta aproximación tiene muchas características interesantes. En primer lugar, no hay conflicto entre los N^2 caminos independientes entre entradas y salidas, así que sólo hay colas a la salida. Al igual que en el caso del medio compartido, este SF es de difusión y selección, lo que simplifica la multidifusión. También como el SF de medio compartido, es fácil de diseñar pero esta vez los filtros de direcciones y los buffers sólo tienen que trabajar a la velocidad del puerto. Además, no hay límite de velocidad que frene la escalabilidad. Por este motivo hay diseños comerciales de encaminadores con este SF [24].

Sin embargo no todo es perfecto. El crecimiento cuadrático del número de buffers

(N^2) impone un límite práctico al número de puertos que puede implementar el encaminador. La velocidad V se puede incrementar hasta los límites de los filtros de direcciones y los buffers. Se han hecho propuestas que reducen el número de buffers a la salida. En lugar de tener uno por entrada, se ponen L. Este límite viene de observar que la probabilidad de que lleguen simultáneamente L paquetes a la misma salida es pequeña. De este modo se gana en escalabilidad a costa de incrementar la probabilidad de pérdida de paquetes.

SF de crossbar

La colocación de los buffers a la salida ofrece grandes ventajas pero también tiene serios inconvenientes, como queda reflejado en la Sección 2.5.4. Por este motivo muchas veces se elige la opción de buffers a la entrada. Un SF muy popular para este tipo de arquitecturas es el crossbar, por tres motivos: su bajo coste, buena escalabilidad y el hecho de que no es bloqueante. En la Figura 2.10 se detalla este tipo de SF.

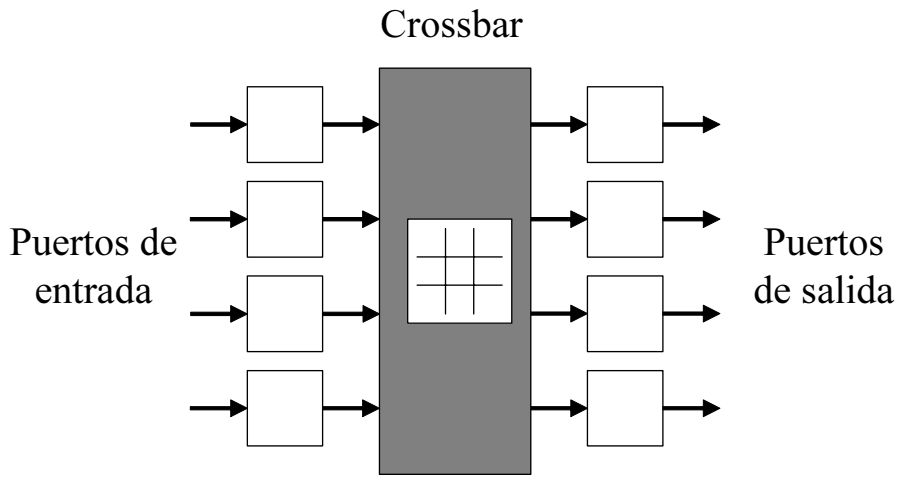


Figura 2.10: Encaminador con SF crossbar.

El crossbar es capaz de conectar cualquiera de las N entradas con cualquiera de las N salidas. La velocidad a la que necesita operar es la misma que la de los enlaces. En la situación actual de la tecnología, se considera que esta arquitectura es más escalable que las de buffers a la salida o memoria compartida. A pesar de sus ventajas, los crossbars presentan varios retos que hay que superar para ofrecer valores razonables de latencia y productividad.

Es un problema bien conocido que si sólo hay una cola FIFO en cada buffer de entrada, el HOL Blocking hace que la productividad máxima se reduzca mucho, alrededor de un 60 % con tráfico uniforme [16]. Para resolver este problema se han propuesto distintas alternativas que se incluyen en el punto de organización de los buffers.

SF de grandes dimensiones

Con el crecimiento que está teniendo Internet en los últimos años, los encaminadores IP deben estar preparados para soportar un gran número de puertos. Sin embargo, tener que implementar un switch fabric de grandes dimensiones plantea nuevos retos a sus diseñadores. En [2] el autor afirma que:

> "Es generalmente aceptado que grandes elementos de conmutación para los encaminadores con productividad de 1 Terabit por segundo o más, no pueden ser realizados simplemente aumentando el tamaño y la velocidad del elemento de conmutación. En su lugar, se deben construir interconectando módulos de conmutación de productividad limitada. Los módulos pequeños pueden diseñarse siguiendo cualquier aproximación, y hay varios modos para interconectarlos."

Aunque el crossbar funciona bien, cuando se necesita incrementar mucho su tamaño, su complejidad crece excesivamente y el rendimiento se degrada por el tiempo necesario para el arbitraje y la compleja gestión de los buffers de entrada. Estas cuestiones limitan el tamaño máximo del conmutador. Para resolver esto es necesario introducir sistemas más complejos, como una red multietapa.

Una multietapa es una red que establece las conexiones entre los diferentes terminales de entrada y salida por medio de un conjunto de dispositivos de conmutación dispuestos u organizados en una serie de etapas. El número de estos conmutadores, la latencia de los mensajes, la longitud del camino que deben seguir los mensajes o el tipo de canales (unidireccionales o bidireccionales) son varios de los aspectos que se tienen en cuenta a la hora de estudiar estas redes [18].

Otro parámetro importante de la red es el número de permutaciones permitidas. Una permutación indica las conexiones entre los terminales de entrada y los de salida, estando cada origen conectado a un único destino. Una permutación de la forma $(p_0, q_0), (p_1, q_1), \ldots, (p_n, q_n)$ indica que el origen p_0 se conecta al destino q_0, p_1 a q_1, etc. La facilidad para establecer una determinada conexión entre un terminal de entrada y uno de salida es una buena medida de la capacidad de conexión de este tipo de redes.

Una clasificación ampliamente aceptada de estas redes es la que agrupa a la gran cantidad de modelos propuestos según, precisamente, su capacidad de conexión [9]. De acuerdo con este criterio las redes multietapa pueden ser:

1. *Bloqueantes*. El establecimiento de una conexión entre un terminal de entrada y otro de salida, ambos libres, no siempre es posible, debido a las conexiones ya existentes. Esto se traduce en que no todas las permutaciones posibles serán realizables. A pesar de ello, dada su sencillez de diseño y control, son ampliamente utilizadas. La red Omega es un ejemplo de este tipo de redes.

2. *No bloqueantes*. Cualquier terminal de entrada puede ser conectado a cualquier terminal de salida libre sin que ello afecte a las conexiones ya existentes. Todas las permutaciones, por tanto, serán realizables, y de ahí que alcance la misma funcionalidad que el crossbar. El gran inconveniente de estas redes es que son caras y tienen un tipo de control más complejo. Un ejemplo de redes no bloqueantes es la red Clos.

3. *Reordenables*. Al igual que en las no bloqueantes, se puede establecer cualquier conexión entre un terminal de entrada y uno de salida libre. Sin embargo, esta conexión puede necesitar la reordenación de las conexiones ya existentes. El número de etapas es menor y las características de los conmutadores menos exigentes. La red de Benĕs es el ejemplo más característico de redes multietapa reordenables.

Otro aspecto que diferencia unas multietapas de otras es el que se refiere al número de etapas que las forman y la manera de conectar dichas etapas. El número de etapas dependerá, para un patrón de conexión dado, del número de puertos de los conmutadores. Las conexiones se establecen de acuerdo con uno, o varios, de los múltiples modelos posibles, como por ejemplo perfect shuffle, bit reversal, butterfly o n-cube.

Sin embargo, también muchas de ellas tienen características muy similares, y que las hacen de alguna manera y en ciertos aspectos topológica y funcionalmente equivalentes. La equivalencia topológica se manifiesta en que el grafo que se obtiene para dos redes cualesquiera es el mismo tras renumerar adecuadamente los terminales de entrada y salida en una de ellas. Dos redes multietapa son funcionalmente equivalentes si realizan las mismas permutaciones.

2.5.3. Output Adapter

Los adaptadores de salida realizan funciones muy similares a los de entrada. De hecho, en muchas implementaciones ambas funciones las realiza el mismo dispositivo.

2.5.4. Organización de los buffers

En las secciones anteriores se ha puesto de manifiesto la importancia de los buffers y su organización en la arquitectura del encaminador. En los puntos siguientes profundizaremos más en este tema.

Buffers a la salida

En esta organización los paquetes entrantes cruzan inmediatamente el switch fabric y se colocan en el buffer correspondiente a la salida por la que deben ir. Puede darse el caso de que varios paquetes lleguen simultáneamente por varios enlaces, lo que obliga al switch fabric a atenderlos a la vez. Además, varios paquetes podrían ir a la misma salida simultáneamente, con lo que el buffer de salida también debe ser más rápido que los enlaces. En el peor de los casos, tanto el switch fabric como los buffers deben ser N veces más rápidos que los enlaces, donde N es el número de enlaces del encaminador. Esto supone un serio freno a la escalabilidad del encaminador.

Sin embargo, esta organización tiene la ventaja de evitar el HOL Blocking completamente. También facilita el arbitraje distribuido en las salidas.

Buffers centrales compartidos

Los paquetes que entran al encaminador van todos a un único buffer central, compartido por todos los puertos. Esta organización aprovecha al máximo el espacio del buffer, aunque el tráfico no se distribuya uniformemente. Por otro lado, en un momento dado podrían llegar N paquetes simultáneamente y salir otros N paquetes. Esto obliga a implementar N puertos de escritura y otros N puertos de lectura o implementar menos puertos, pero más rápidos que los enlaces. De nuevo, esto es una gran limitación.

Buffers a la entrada

La organización de buffers a la entrada no tiene problemas de escalabilidad, porque a cada buffer sólo puede llegar un paquete en un momento dado, con lo que opera a la velocidad del enlace. Sin embargo, al utilizarse colas FIFO, sólo el primer paquete de cada cola es seleccionable para entrar en el switch fabric, lo que produce el efecto de HOL Blocking. Este efecto consiste en que cuando el paquete en cabeza se bloquea, todos

los que están detrás de él se bloquean también, aunque pudieran salir. De esta forma, la productividad máxima de cada entrada se limita a alrededor del 60 % [16].

Dado que esta organización de los buffers escala tan bien, se ha hecho un gran esfuerzo para resolver el problema del HOL Blocking. Una solución [1] [20] que se ha propuesto es mejorar la estructura de la cola de entrada. En lugar de una única cola FIFO, se mantiene una cola por cada salida, de modo que los paquetes que iban a una salida libre, puedan hacerlo. A esta solución se la conoce como VOQ (Virtual Output Queuing).

La organización de colas a la entrada [21] [15] puede dar lugar a que varios paquetes compitan por ir a la misma salida, lo que obliga a implementar un algoritmo de arbitraje que programe el emparejamiento entre entradas y salidas. El emparejamiento máximo, que conecta el mayor número de entradas con salidas es el que mejor rendimiento da, pero es muy complejo de calcular. En su lugar se utilizan algoritmos que calculan emparejamientos maximales, en los que no se puede establecer otra pareja de entrada y salida sin romper otra. No es la mejor solución, pero es rápida de calcular.

Buffers a la entrada y a la salida

En esta organización el switch fabric opera a una velocidad mayor que la de los enlaces, pero sin llegar a ser N veces más rápido. También facilita el arbitraje distribuido en las salidas. Se trata de una solución de compromiso. Al operar el switch fabric tan rápido, son necesarios buffers en las salidas además de en las entradas. Se ha comprobado que un switch fabric unas cuatro veces más rápido que los enlaces mitiga en gran parte el efecto del HOL Blocking.

2.5.5. Control de flujo

El control de flujo consiste en controlar el avance de los paquetes en la red. Para el control de flujo se han propuesto distintas soluciones [9]. Aquí se revisarán las tres más populares.

Control de flujo basado en créditos

Se trata de un control de flujo salto a salto, en el que cada elemento de la red mantiene una cuenta del espacio disponible en los buffers del siguiente elemento. Esta

cuenta son los créditos. Cada vez que se envía una unidad de control de flujo, generalmente un flit, se decrementa la cuenta de créditos. Cuando los datos abandonan el buffer en la siguiente etapa, el encaminador se lo notifica a su antecesor enviándole el número adecuado de créditos.

Este tipo de control de flujo evita el descarte de paquetes, porque siempre se está seguro de que habrá sitio en el buffer de la siguiente etapa. El problema de esta técnica es que puede ser más compleja de implementar que otras.

Control de flujo Stop and Go

Esta técnica también es salto a salto. En los buffers de cada elemento de la red se marcan dos niveles: Stop y Go. Un receptor acepta paquetes hasta que su buffer llega a la marca de Stop. En ese momento envía al emisor el mensaje Stop, indicándole que deje de mandar paquetes o se producirá un desbordamiento. A medida que el buffer del receptor se vacía alcanzará la marca Go. En ese instante se envía el mensaje Go al emisor indicándole que puede reanudar el envío de paquetes.

La eficacia de este tipo de control de flujo depende directamente de lo bien que se sitúen las marcas Stop y Go. Este lugar dependerá directamente del tiempo de vuelo del enlace. Generalmente esta técnica no aprovecha tan bien los buffers como otras.

Control de flujo por tasa de inyección

El control de flujo por tasa de inyección (rate-based flow control) es el estándar de ATM y lo incorpora también el nuevo estándar Infiniband [12]. En ATM los elementos de la red ajustan dinámicamente el ritmo al que se lanzan las celdas mediante mensajes o bits especiales en las celdas. Se trata de un control de flujo extremo a extremo.

Cuando un elemento detecta que sus buffers se están llenando, toma dos medidas: primera, envía un mensaje a la fuente para que reduzca su tasa de inyección. Segunda, descartar las celdas excedentes, generalmente empezando por las de baja prioridad.

Descartar celdas tiene sus ventajas y sus inconvenientes. Cuando el tráfico es multimedia, como la transmisión de un vídeo, generalmente se prefiere perder una pequeña parte de la transmisión, que probablemente pase inadvertida, a no hacer descarte pero retrasar toda la transferencia, lo que podría ser nefasto en aplicaciones de tiempo real. Este fue el motivo de que ATM implemente el control de flujo por tasa de inyección. Por otro

lado, en aplicaciones que no son de tiempo real, como una transferencia de fichero, las celdas que se descarten deben retransmitirse porque las pérdidas son inadmisibles. Estas retransmisiones pueden suponer una gran merma en la productividad de la transferencia.

2.5.6. Técnicas de conmutación

La técnica de conmutación determina cómo se establecen las conexiones entre las entradas y las salidas del conmutador, aunque afectan a muchos otros aspectos de la red. Se comenta a continuación, de forma breve, las más usadas [9].

Conmutación de circuitos

Esta técnica ha sido ampliamente utilizada en telefonía, y en los primeros tiempos de la implantación de redes. Antes de comenzar la transmisión, el emisor envía un paquete sonda al receptor, reservando todos los recursos necesarios. Cuando se han obtenido los recursos, el emisor empieza a transmitir los datos, sin que haya riesgo de contención de ninguna clase. Al terminar la transmisión se liberan los recursos.

La ventaja de esta técnica es que una vez establecida una conexión tiene garantizados sus recursos, pero produce una baja utilización de la red.

Store and Forward

En este caso los mensajes se dividen en paquetes, que constituyen la unidad de control de flujo. Un paquete se envía hacia un conmutador, donde entra en un buffer hasta que se recibe completamente. En ese momento se reenvía hacia el destinatario.

Con esta técnica los recursos de la red se aprovechan mucho más, porque no hay reserva de gran número de recursos. Se han propuesto varias mejoras para este esquema básico, que dan lugar a otras técnicas, como Virtual Cut-through o Wormhole.

Virtual Cut-through

Se trata de una mejora de Store and Forward. En lugar de que los conmutadores esperen a la recepción completa del paquete para encaminarlos hacia la siguiente etapa, el

paquete sale del conmutador en cuanto el enlace está disponible, reduciendo en un orden de magnitud la latencia. Todavía es necesario reservar espacio para todo el paquete en el siguiente buffer y, por tanto, la unidad de control de flujo sigue siendo el paquete.

Wormhole

Se propuso como mejora de Virtual Cut-through para reducir el tamaño de los buffers. Ahora ya no es necesario reservar en los buffers espacio para todo el paquete. De hecho, lo habitual cuando se implementa esta técnica es que los buffers sean más pequeños que un paquete. Cuando la cabecera del paquete se detiene, todos los flits deben detenerse también manteniendo los recursos que estuvieran usando en ese momento. De este modo, un paquete puede quedar parado y repartido en varios conmutadores. Por este motivo se incrementa la probabilidad de interbloqueo. La unidad de control de flujo es el flit.

2.6. QoS

Como quedó de manifiesto en el punto de evolución de Internet (Sección 2.1) las aplicaciones actuales demandan algo más que ancho de banda. Hoy en día hay muchos tipos de aplicaciones, como las denominadas aplicaciones multimedia que demandan QoS. En esta sección se revisará con detalle de dónde vienen esas necesidades y cómo impactan en los encaminadores.

2.6.1. Necesidad de QoS

En la introducción se expone que en los últimos años el uso de Internet ha cambiado. Se ha hecho más interactivo y contiene más multimedia. Además de los cambios en las antiguas aplicaciones, han surgido otras nuevas. Todo esto es diferente del uso que se había previsto de Internet, y la tecnología que se propuso en sus inicios ya no es suficiente. En la Internet que viene es necesaria la calidad de servicio (QoS).

Algunos ejemplos de estas nuevas aplicaciones, que requieren QoS, podrían ser:

Videoconferencia. La videoconferencia consiste en una comunicación en la que intervienen la voz y la imagen. Se trata de una aplicación interactiva en tiempo real, y

por tanto sus requisitos en términos de retardo son muy exigentes, menos de 150 ms [11]. Por otro lado, dado que no hay mucho movimiento, los actuales algoritmos de compresión de vídeo, como H.263, reducen los requisitos de ancho de banda a alrededor de 64 Kbps [13].

Difusión de televisión digital. Se trata de una aplicación asimétrica en la que un operador emite una señal de audio y vídeo para muchos clientes. En este caso el ancho de banda sí es una gran necesidad, del orden de Mbps. La latencia ya no es tan importante, aunque conviene que su variación (*jitter*) no cambie mucho. Este tipo de aplicaciones (uno a muchos) puede beneficiarse mucho de un inteligente uso de la multidifusión.

Vídeo bajo demanda. Es una aplicación similar a la anterior, pero esta vez es el usuario quien inicia la transmisión. Por este motivo ya no se puede beneficiar de la multidifusión. Mediante técnicas de buffering se pueden reducir los requisitos de latencia o *jitter*, pero no los de ancho de banda.

Telemedicina. En esta aplicación un médico puede tratar a un paciente de forma remota. La gran carga de interactividad requiere una baja latencia y poco jitter. Además, se necesita transmitir imágenes de alta resolución, demandando mucho ancho de banda. Por último, estas aplicaciones son muy sensibles a errores. De hecho, suele ser normal que la compresión de este tipo de imágenes deba ser sin pérdidas y la transmisión sin errores.

Información a manejar. Aunque no se trata de una aplicación, es claro que las redes deben soportar más cantidad de información. En el año 1993 había 140 sitios Web, en el año 1997 se alcanzó el millón y en el año 2000 los más de 20 millones [8]. Además de haber cada vez más sitios Web, éstos son cada vez más grandes, plantean más opciones, más interactividad, adaptabilidad al usuario, etc.

Por otro lado, durante los últimos años la tecnología de redes ha experimentado grandes avances. Estos avances se pueden centrar en:

Velocidades de transmisión. Se ha pasado de los tradicionales 10 Mbps de la popular Ethernet, al orden de los Gigabits por segundo gracias a la fibra óptica, como por ejemplo en GigabitEthernet.

Tecnologías de red. Desde X.25 o Frame Relay, pasando por ATM, hasta Infiniband, la tecnología de redes ha ido evolucionando, aprovechando cada vez mejor los recursos de la red.

Abaratamiento de costes. Las mejoras en las redes han venido acompañadas de una bajada en los precios. El número de usuarios de Internet, por ejemplo, crece exponencialmente y cada vez es más popular el acceso mediante ADSL o cable módem.

Sin embargo, aunque las mejoras en el campo de las redes han sido notables, no bastan para cubrir todas las necesidades de las nuevas aplicaciones. Sobredimensionar la red no es rentable, por lo que hay que buscar soluciones que aprovechen mejor los recursos de los que se disponen.

2.6.2. Tipos de tráfico

Un encaminador acepta tráfico de muy distintos tipos. Algunos tipos de tráfico tienen requisitos en cuanto a latencia máxima, otros no son tan sensibles al retardo, pero necesitan un ancho de banda constante. A otros simplemente les basta con llegar. A los encaminadores que son conscientes de estas diferencias e intentan proporcionar un trato distinto a cada tipo de tráfico según sus necesidades se les denomina encaminadores con soporte de QoS.

Pelissier [25] propone una clasificación de los flujos de tráfico en función de sus requisitos. Se distinguen cuatro categorías:

DBTS (Dedicated Bandwidth Time Sensitive) Tráfico que requiere un ancho de banda mínimo. Además, también necesita una latencia máxima acotada. Un ejemplo podría ser la videoconferencia.

DB (Dedicated Bandwidth) Es similar al tipo anterior, en cuanto a que también requiere un ancho de banda mínimo, pero no es demasiado sensible a la latencia. Éste es el caso de la transmisión de vídeo bajo demanda sin interactividad.

BE (Best Effort) Este era el tráfico dominante en Internet hasta ahora: insensible al ancho de banda y la latencia, dentro de unos márgenes que cubrían los elementos de la red haciendo su trabajo lo mejor posible (best effort). Ejemplos: navegar por la Web, enviar un e-mail, etc.

CH (Challenged) Tráfico cuyas prestaciones se degradan de forma intencionada, para evitar que afecte al best effort. Se trata de tareas administrativas, que no deben interferir con el uso normal, como la transferencia de copias de seguridad.

2.6.3. Servicios Diferenciados y Servicios Integrados

Los dos principales enfoques propuestos para que Internet afronte el reto de la QoS son: los Servicios Diferenciados [3] y los Servicios Integrados [4]. Muchos son los parecidos y diferencias entre ambos enfoques. A continuación se ven brevemente las principales ideas de cada uno de ellos.

Servicios Diferenciados

El modelo de Servicios Diferenciados se basa en un esquema de prioridades. Los encaminadores hacen un tratamiento paquete a paquete en el que sólo tienen en cuenta el nivel de servicio que éstos llevan incorporado en sus cabeceras. En IPv4 era en el campo ToS, pero nunca se llegó a usar. En IPv6 es el campo Traffic Class.

Para marcar cada paquete con el nivel de servicio adecuado hace falta un acuerdo de los usuarios con sus ISPs y un tratamiento adecuado en los encaminadores fronterizos.

Como los encaminadores sólo deben mantener información acerca de un número limitado de clases de servicio, este modelo tiene buenas propiedades de escalabilidad. Además, las operaciones más complejas, como las de identificación y marcado del tráfico, sólo deben realizarse en los encaminadores fronterizos, de modo que se pueden utilizar encaminadores más sencillos en el interior, abaratando costes.

Servicios Integrados

En el modelo de Servicios Integrados se persigue un marco global para proporcionar QoS en Internet. En este marco se hace un tratamiento a nivel de flujos de paquetes, donde todos los paquetes de un mismo flujo deben tener los mismos requisitos de QoS.

El modelo de Servicios Integrados requiere las siguientes funciones:

- Control de admisión: Antes de comenzar a transmitir, las aplicaciones deben hacer una reserva previa de recursos. Si no es posible satisfacer las demandas realizadas por las aplicaciones puede haber un proceso de negociación, pudiendo suceder que la conexión no llegue a establecerse.
- Algoritmo de enrutamiento: Debe tener en cuenta las necesidades de QoS, buscando no sólo la ruta más corta, sino la que mejores prestaciones puede dar.

- Arbitraje: Para resolver adecuadamente los conflictos que se puedan presentar en el interior de los encaminadores.
- Descarte de paquetes: Si se permite el descarte de paquetes para tratar la congestión, habrá que tener en cuenta las necesidades de QoS. Esto también es aplicable a Servicios Diferenciados.

2.6.4. Soporte de QoS en el encaminador

Desde su posición privilegiada para controlar el tráfico de la red, los encaminadores son elementos clave para proporcionar QoS. Más allá del servicio best effort, los encaminadores empiezan a soportar un número de clases de servicio o niveles de QoS. Este esquema de prioridades se utiliza para indicar el tratamiento diferente que debe recibir una clase de tráfico sobre otras. Los elementos de conmutación deben manejar estas clases de tráfico de un modo diferente, de acuerdo a sus requisitos de QoS. En la organización con buffers a la salida, por ejemplo, es habitual implementar un buffer para cada nivel de servicio, en cada salida. Estos buffers pueden estar separados físicamente, o un único buffer dividirse de forma lógica.

Esta separación de los buffers se hace para poder aplicarles una gestión diferenciada, como por ejemplo, la aplicación de distintas políticas de descarte de paquetes (por ejemplo, Drop Tail, Drop-From-Front, Random Early Detection (RED), etc.), o las políticas de planificación de los paquetes salientes (por ejemplo, prioridades estrictas, round-robin, weighted round-robin, etc.). Por estos motivos, es claro que el soporte de QoS debe ser una parte importante del diseño del switch fabric.

Capítulo 3

OBJETIVOS Y METODOLOGÍA

En este capítulo se indican los objetivos del proyecto, señalando, además, cuales son las etapas a seguir para conseguir alcanzarlos. Se comenzará viendo de forma detallada todos los objetivos que se pretendían cubrir al inicio de este proyecto. A continuación, se verá la metodología usada y las distintas etapas seguidas a lo largo de este proyecto.

3.1. OBJETIVOS

Como se ha puesto de manifiesto en los capítulos anteriores, las exigencias demandadas a los encaminadores son cada vez mayores. Para conseguir cumplir con ellas se han planteado diversas líneas de actuación. Así, por ejemplo, se ha propuesto incrementar el número de canales virtuales. Con más canales virtuales es posible evitar el HOL Blocking [20]. También se han planteado complejas políticas de calidad de servicio, como asignar un canal virtual a cada flujo [19]. Sin embargo, incrementar el número de canales virtuales implica o bien reducir el espacio de buffer asignado a cada canal virtual con colas a la entrada o a la salida, o bien usar un buffer central, que optimiza el espacio pero debe funcionar a frecuencias de reloj muy altas.

En cualquier caso, lo que parece claro es que es posible encontrar soluciones a cualquier problema planteado en el diseño de los encaminadores, a costa, eso sí, en la mayoría de los casos, de un aumento en su complejidad.

El objetivo principal de este proyecto es analizar si una reducción en la complejidad del diseño permite seguir alcanzando niveles de prestaciones similares. Evidentemente reducir complejidad no debe suponer eliminar funcionalidades, y por tanto esta

reducción debe estar basada en la eliminación de redundancias. Un análisis detenido del modelo de encaminador que se presentó en el capítulo anterior permite ver que hay muchas posibles redundancias. En los buffers del IA se absorbe el tráfico entrante en el orden en que llega. Sin embargo, en su interior tiene lugar cierto procesamiento que permite que a la salida los paquetes más prioritarios adelanten a los menos urgentes. Este flujo ordenado de paquetes entra a continuación en la primera etapa del SF, en forma de varios canales virtuales. En el primer conmutador entran los flujos provenientes de varios IAs. En su interior se realiza un nuevo procesamiento de modo que a la salida se vuelve a tener un flujo ordenado, compuesto otra vez de varios canales virtuales.

En el conmutador ya se puede apreciar que quizá se esté haciendo trabajo de más. Como se ha visto, a la salida de los IAs ya se tienen los paquetes ordenados ¿Es realmente necesario volver a separar el flujo en canales virtuales? ¿No sería más fácil tratar de conservar, en la medida de lo posible, el trabajo realizado en el IA?

A estas y otras preguntas se pretende dar respuesta con el estudio que aquí se presenta. La idea básica es llevar a un simulador una serie de propuestas orientadas a simplificar el diseño de un encaminador y comprobar hasta qué punto la reducción en la complejidad del diseño, perseguida por aquellas, se refleja o no en una disminución de las prestaciones.

Este objetivo global se puede desglosar en un conjunto de subobjetivos parciales, los cuales se enumeran a continuación:

- **Estudio y comprensión del modelo de encaminador corporativo**

  Se trata de revisar las características de los actuales encaminadores corporativos y buscar aspectos de su diseño sobre los cuales sea posible algún tipo de simplificación que no suponga una reducción significativa de prestaciones.

- **Elaboración de diversas propuestas de diseño**

  Localización de los puntos de diseño sobre los que, de acuerdo con la idea perseguida, sea razonable abordar algún tipo de simplificación, se trata de elaborar propuestas que incidiendo sobre dichos aspectos del diseño permitan alcanzar esa simplificación.

- **Estudio de las propuestas realizadas y de la idoneidad de las mismas**

  Se obtendrán, a través de una adecuada batería de pruebas, una serie de resultados que deben servir para evaluar las diferentes propuestas realizadas.

3.2. METODOLOGÍA

Para lograr estos objetivos, se han seguido una serie de pasos habituales en este tipo de estudios y que responden a una metodología debidamente programada. A continuación se resumen las diferentes etapas seguidas, algunas de las cuales se han solapado en el tiempo:

- **Estudio del estado del arte del tema central del proyecto**

  Para alcanzar los conocimientos necesarios para realizar un estudio como el aquí presentado, se hace necesaria la consulta de bibliografía especializada, en cualquiera de sus formatos: libros, artículos de revistas, ponencias de congresos, páginas Web, etc.

- **Estudio de las redundancias en los encaminadores IP**

  Una vez se tienen los conocimientos sobre el funcionamiento de los encaminadores, sus tipos y características, se ha abordado el tema de simplificar su diseño. Para ello se han detectado las posibles redundancias y se han propuesto implementaciones alternativas.

- **Aprendizaje y manejo de un entorno de trabajo**

  Las características del estudio realizado han determinado que el análisis de las propuestas presentadas se haya efectuado mediante simulación. Se ha utilizado un simulador de un encaminador IP cuyas características se indican en el Capítulo 5. Esto ha obligado a un profundo aprendizaje de su funcionamiento y una exhaustiva revisión de su estructura e implementación para poder usarlo adecuadamente, una vez incorporadas las modificaciones que reflejan las diversas propuestas aquí presentadas.

- **Diseño e implementación de las propuestas**

  Una vez obtenido el grado de comprensión requerido del simulador, se han de identificar los puntos del código donde las modificaciones sean necesarias. El simulador no tenía soporte para QoS, por lo que hubo que realizar las modificaciones oportunas. La implementación se realizó en forma de nuevas opciones. De este modo es posible simular distintas configuraciones de encaminador, proporcionando la entrada adecuada.

- **Obtención y análisis de resultados**

  Esta última fase ha consistido en preparar una amplia batería de pruebas, desarrollar las correspondientes simulaciones y analizar los resultados obtenidos. Esta batería de pruebas estaba orientada fundamentalmente a evaluar una serie de configuraciones del encaminador, explotando en distinto grado las redundancias identificadas. Para que el proceso de evaluación sea correcto y preciso se han considerado todos aquellos aspectos relevantes en este tipo de estudios.

Capítulo 4

MODIFICACIONES EN EL DISEÑO DE ENCAMINADORES IP

Como se ha indicado en el capítulo anterior, el objetivo de este trabajo es analizar la posibilidad de reducir la complejidad de los encaminadores sin que ello signifique pérdida de rendimiento. Como también se ha señalado, esa reducción estará basada en la eliminación de ciertas redundancias detectadas en el diseño de estos encaminadores.

En este capítulo se pone al descubierto alguna de esas redundancias y se proponen modificaciones en el encaminador que las eliminan, de cara a simplificar el diseño. Previamente se presenta el modelo de encaminador concreto sobre el que se ha centrado el estudio, describiendo cada uno de los componentes que lo integran.

4.1. MODELO DE ENCAMINADOR

El encaminador en el que se centra este proyecto es de tipo corporativo. Como se vio en el Capítulo 2, debido a las características de las redes a las que está destinado, este tipo de encaminador necesita conectar un gran número de sistemas, por lo que implementa muchos puertos. También debe ofrecer un gran ancho de banda, pero teniendo en cuenta las necesidades de QoS. Por último, el coste de este tipo de encaminadores no debe ser excesivo. Estos requisitos hacen que los encaminadores corporativos sean un reto para sus diseñadores y el dispositivo ideal para probar los objetivos descritos en el Capítulo 3.

El encaminador concreto en el que se inspira en buena medida este proyecto es el Avici TSR [26]. Se trata de un encaminador de altas prestaciones (TSR viene de Terabit

Switch Router), con un gran número de puertos, hasta 640.

Las propuestas que aquí se realizan no sólo sirven para el mencionado Avici TSR. Cualquier otro encaminador o conmutador con características similares, es decir, con un tratamiento del tráfico en varias etapas, podría ser susceptible de padecer las mismas redundancias. Por lo tanto, también podrían aplicársele las mismas mejoras.

4.1.1. Arquitectura del encaminador

La arquitectura de este tipo de encaminador es la basada en conmutador con procesadores totalmente distribuidos (Sección 2.4.4). En ella, las interfaces de red o IAs tienen suficiente capacidad de procesamiento y buffer como para tratar todo el tráfico que les llega.

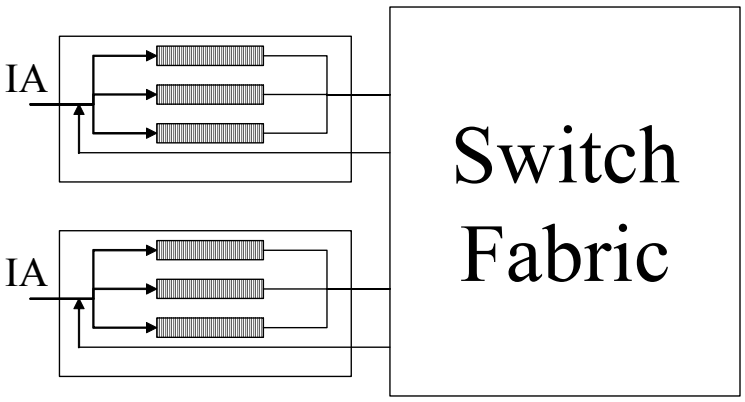


Figura 4.1: Arquitectura básica del encaminador.

Una descripción gráfica del encaminador se muestra en la Figura 4.1. En ella se resaltan los dos componentes principales del encaminador: el switch fabric y los Input Adapter.

Input Adapter

La misión de los IA consiste en almacenar y clasificar el tráfico entrante. Para ello dispone de un buffer dividido en canales virtuales para los distintos niveles de servicio. El tamaño del buffer es lo bastante grande para absorber las posibles ráfagas que puedan llegar. Además, implementa algún tipo de mecanismo de control de flujo, como, por ejemplo, el de créditos. En este proyecto no es de interés entrar en estas cuestiones, por lo

que supondremos que el tamaño del buffer no es un problema, dentro de unos límites razonables. Se considera también VOQ pues es la solución al HOL Blocking habitualmente incorporada en este tipo de encaminadores.

Dado que estos dispositivos reciben tráfico de muy diferentes características, se considera que se disponen de facilidades para su clasificación.Los IAs también actuarán como adaptadores de salida. Esto quiere decir que cada IA gestiona un enlace bidireccional.

Otra de las funciones del IA es dividir los paquetes entrantes, denominados mensajes desde ahora, en porciones más pequeñas de tamaño fijo. Es decir, un mensaje sería un datagrama IP y un paquete una porción de ese datagrama. Cuando los paquetes cruzan el switch fabric llegan al IA de salida donde son reensamblados en el mensaje original.

Switch Fabric

De entre los medios de conmutación que se presentaron en la Sección 2.5.2 el encaminador utilizará una red multietapa. El Avici TSR, modelo de referencia, tiene el procesamiento distribuido entre los IAs y utiliza una red de interconexión toro 3d. La red multietapa que se modela en este proyecto proporciona peores prestaciones, según [6]. A pesar de ello, para el desarrollo de este proyecto se ha preferido la red multietapa por dos motivos:

- El objetivo del proyecto no es medir el rendimiento del switch fabric, sino el de las mejoras que se proponen en este capítulo. Para ese fin la multietapa es una opción tan válida como el toro 3d.
- El simulador de partida ya implementaba las redes multietapa. Modificar ese simulador, o hacer otro nuevo, habría sido demasiado costoso para el proyecto y tampoco habría aportado demasiado.

Las redes multietapa proporcionan la escalabilidad que requiere un encaminador corporativo, aunque quizá no den las mismas prestaciones que otros tipos de switch fabric. Sin embargo, la red multietapa es ideal para poner en práctica las ideas centrales del proyecto, al presentar redundancias entre sus etapas. Esto no quiere decir que las mejoras que aquí se discuten sólo sirvan en redes multietapa. Llevarlas a los nodos del toro 3d del Avici TSR, por ejemplo, sería algo directo, y está dentro de lo que se plantea como trabajo futuro.

En la Figura 4.2 se muestra la multietapa considerada para este modelo. Se trata de una shuffle-exchange, una multietapa de tipo bloqueante. Sin embargo, esto no debe afectar demasiado al rendimiento si se diseña adecuadamente.

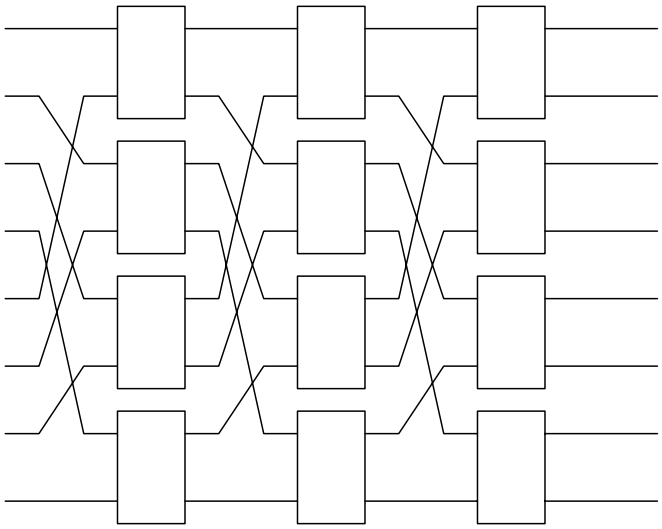


Figura 4.2: Estructura del SF mediante red multietapa del tipo Omega.

Cada uno de los conmutadores tiene la estructura mostrada en la Figura 4.3. Como se puede observar, se compone de los siguientes elementos:

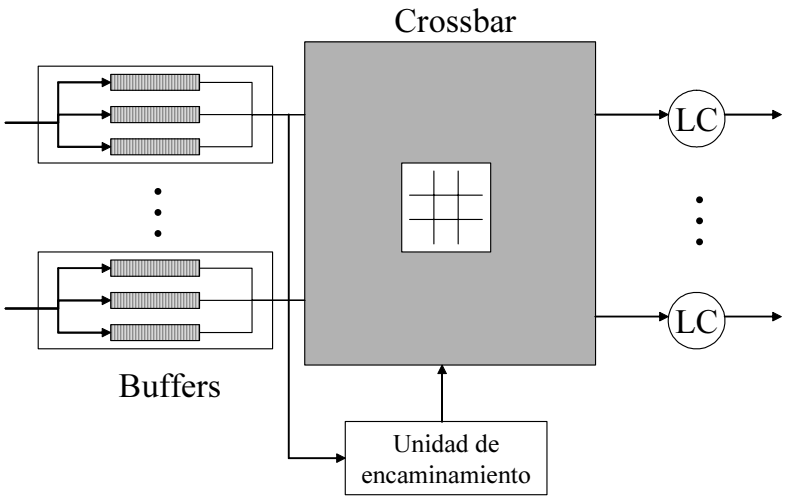


Figura 4.3: Un conmutador de la multietapa.

Crossbar: Sirve de switch fabric a nivel de conmutador, conectando los enlaces de entrada con los de salida.

Buffer: Espacio del almacenamiento para los mensajes que esperan a que quede libre su enlace. Sobre los buffers se implementa, mediante canales virtuales, el VOQ y el tratamiento de los niveles de servicio. Estos canales virtuales son en realidad colas lógicas, que se reparten el espacio del buffer según sea necesario, de un modo parecido a los buffers centrales.

Unidad de encaminamiento: Ejecuta la función de encaminamiento, que para el modelo considerado se trata de encaminamiento basado en la dirección destino. El puerto de salida en cada etapa viene dado por el dígito del identificador del puerto de salida del encaminador asociado a esa etapa.

Link Controller: Resuelve los conflictos sobre un enlace, decidiendo qué paquete debe continuar. El arbitraje del conmutador es distribuido.

Los paquetes que forman un mensaje nunca se adelantan unos a otros, preservan el orden. Además, no se intercalan paquetes de dos mensajes distintos. El primer paquete del mensaje va reservando recursos según avanza, mientras que el último los va liberando.

Entre los conmutadores de la multietapa se implementa como técnica de conmutación virtual cut-through, de modo que la cabecera de un paquete no necesita esperar a que llegue el resto para salir por el enlace. Sin embargo, a diferencia de wormhole, siempre debe haber espacio suficiente para todo el paquete en el buffer del siguiente salto.

El control de flujo entre conmutadores está basado en créditos, de modo que cada conmutador conoce cuánto espacio le quedan a los buffers de la siguiente etapa. Este tipo de control de flujo elimina la necesidad de descartar paquetes.

4.1.2. Modelo del tráfico

Aunque el encaminador modelado para este proyecto es de tipo IP, sin embargo, el tráfico se abstrae mediante paquetes de tamaño fijo, con una distribución de destinos uniforme.

4.2. REDUNDANCIA EN LA CLASIFICACIÓN DEL TRÁFICO

Como se introdujo en el Capítulo 3, el tráfico que sale de los IAs está ordenado de acuerdo a los mecanismos internos que éstos implementan. Este flujo ordenado llega al switch fabric, donde los conmutadores pueden plantearse separar este flujo en canales virtuales o no.

El problema que se plantea en el primer conmutador es muy similar a la ordenación de un vector de números mediante el método divide y vencerás. El conocido algoritmo consiste en dividir el vector en varias partes, ordenarlas por separado y combinar las soluciones. Para hacer esto último se crea el nuevo vector solución tomando en cada paso el elemento de cabeza menor de los vectores componentes, es decir, sólo se comparan las cabezas de los vectores, respetando la ordenación que se ha hecho en las etapas anteriores.

En el caso del conmutador el problema es similar. En lugar de deshacer el flujo que le llega en canales virtuales, desbaratando el orden en que venían los paquetes, podría tratar de conservar ese orden ahorrando trabajo y simplificando la lógica. Esta mayor simplicidad se traducirá en elementos más baratos y con ciclos de reloj más cortos.

Por supuesto, estos razonamientos también son aplicables a los conmutadores de las siguientes etapas, de hecho a todos los conmutadores, los cuales respetarían la ordenación de sus predecesores.

4.2.1. Proporcionar Calidad de Servicio

Si sólo hubiera un único criterio para ordenar los paquetes, como el deadline o el nivel de servicio, el esquema planteado en la sección anterior tendría un rendimiento perfecto. A la salida de la última etapa los paquetes estarían ordenados del mejor modo posible. Sin embargo, lo habitual es que haya que tener en cuenta otros criterios a la hora de que un paquete adelante a otro. En concreto, criterios del tipo de ancho de banda.

Para proporcionar un cierto ancho de banda el algoritmo de planificación suele dividir el tiempo en slots y repartirlos proporcionalmente a la demanda de cada flujo o nivel de servicio. Un algoritmo común para esta tarea es el weighted round robin [17]. A la salida del IA, por ejemplo, tendremos una mezcla de paquetes tal que la cantidad de cada tipo será proporcional al ancho de banda otorgado.

Cuando el flujo proporcionado llega al conmutador de la primera etapa, éste no necesita separar el flujo en canales virtuales porque las proporciones de paquetes de cada tipo son las correctas. Le basta con repartir el ancho de banda de forma adecuada entre los enlaces de entrada, de modo que se preserve la proporción entre paquetes.

Como se ha visto, cuando sólo hay un criterio para ordenar los paquetes, entonces todo va muy bien. Pero, ¿qué pasa si tenemos que aplicar varios criterios a la vez? En la clasificación del tráfico de la Sección 2.6.2 se vio que hay tráfico sensible al retraso, sensible al ancho de banda y sensible a ambos. Para resolver este problema podemos plantearnos varias soluciones: podríamos demultiplexar el tráfico en dos categorías, sensible a retraso y sensible a ancho de banda, y aplicar el arbitraje sobre ellas. También podríamos confiar en que la ordenación del IA sea lo bastante buena como para que no haya que separarla en canales virtuales. O quizá este problema haga que la idea de respetar la ordenación anterior no sea práctica. Estas son algunas de las cuestiones que se tratarán de resolver en este proyecto.

Por otro lado se tiene el tráfico best effort. En general, el IA se limitará a rellenar los huecos de tiempo que queden tras enviar el tráfico más prioritario, quizá garantizándole un mínimo de ancho de banda. En esta situación se estropea el orden de los paquetes, porque si el conmutador de la siguiente etapa se limita a observar el paquete de la cabeza del flujo, como se ha defendido antes, podría darse el caso de que un paquete más prioritario quedase bloqueado detrás de un paquete best effort. Sin embargo, la solución es tan sencilla como demultiplexar el tráfico best effort en su propio canal virtual.

En resumen, la propuesta consiste en tratar de reducir el número de canales virtuales que se necesitan en las etapas posteriores al IA. En lugar de tener un CV por flujo o nivel de servicio, tendríamos tres canales virtuales: el de paquetes sensibles al retardo, el de paquetes sensibles al ancho de banda y el de paquetes best effort.

4.3. MODIFICACIONES EN EL DISEÑO DEL ENCAMINADOR

El modelo que se ha presentado en la Sección 4.1 no tiene ningún soporte para QoS. Por ese motivo, es ahí donde se centran todas las mejoras que se verán a continuación. Estas mejoras constituirán la base para la definición de las distintas configuraciones que se considerarán en el Capítulo 5, y que serán evaluadas en el Capítulo 6 para comprobar si se puede sacar provecho de las redundancias que se han puesto de manifiesto.

4.3.1. Mejoras del encaminador en el IA

En esta sección se tratan las mejoras que se proponen para el IA de un encaminador IP básico. Fundamentalmente se añade soporte para varios niveles de servicio, en forma de canales virtuales, y se implementan varias políticas de gestión de las nuevas colas. Estas alternativas serán evaluadas en el Capítulo 6.

Niveles de Servicio

El primer paso para poder aplicar QoS es distinguir el tráfico que entra en el encaminador. Para ello se pueden utilizar campos explícitamente pensados para este fin, como ToS en IPv4 o el campo Traffic Class de IPv6. Sin embargo, los encaminadores IP, a la espera de que IPv6 se imponga, pueden utilizar los números bien conocidos de puertos TCP y UDP para distinguir qué tipo de tráfico lleva cada paquete. Por otro lado, en un escenario del tipo de servicios integrados, se podrían utilizar los servicios de RSVP para reservar y después poder identificar a qué flujo pertenece cada paquete.

Una vez que se conoce el tipo de tráfico, se puede proceder a un tratamiento adecuado. Para conseguir esto, sobre todo en entornos de servicios diferenciados, se pueden utilizar canales virtuales, uno por clase de servicio.

Teniendo en cuenta lo anterior, al modelo que hemos planteado se le añaden los módulos necesarios para identificar el tipo de tráfico que llevan los paquetes y el soporte para canales virtuales, de modo que los paquetes se pongan en la cola adecuada.

Este modelo de encaminador está orientado a los servicios diferenciados, por lo que habrá un número reducido de posibles tipos de paquetes, pudiendo implementar un canal virtual por tipo.

Asignación de CV por Nivel de Servicio

En relación con la mejora anterior, es necesario un módulo que decida qué canal virtual es el adecuado para cada paquete. Para ello se pueden seguir criterios de antigüedad, destino, ocupación de las colas o, especialmente, tipo de tráfico. Por supuesto, estos criterios pueden combinarse para formar políticas más complejas.

También puede resultar de interés, sobre todo para aplicar las ideas que se presentan en la Sección 4.2.1, poner tráfico de varios tipos sobre un único canal virtual. Esta

multiplexación, que podría parecer contraproducente, puede producir beneficios en forma de simplificación del diseño de los encaminadores, como trataremos de comprobar en el Capítulo 6.

Selección de CV por Nivel de Servicio

Para proporcionar QoS no basta con tener los paquetes clasificados en colas o canales virtuales. Cada vez que el enlace del IA queda libre, hay que decidir qué paquete debe ser el siguiente en salir. La implementación más habitual de los buffers, por su sencillez es la de colas FIFO. Esa es la implementación en este modelo de encaminador. Cuando las colas son de este tipo, el problema de elegir paquete se reduce a mirar sólo los que están en cabeza de las colas.

La forma más básica para proporcionar calidad de servicio es establecer un orden estricto entre las clases de servicio y tratar de atender siempre a los paquetes de mayor prioridad antes que otros menos prioritarios. Este esquema tan sencillo es fácil de implementar y puede dar buenas prestaciones, aunque puede hacer que ciertas clases de tráfico (las menos prioritarias) lleguen a sufrir de inanición. Esta situación se produce cuando un paquete de baja prioridad espera a ser atendido, pero esto nunca sucede porque paquetes de mayor prioridad siempre mantienen los recursos.

Por otra parte, este esquema de prioridades estrictas también sirve para simular una cola con reordenación de paquetes. Si en una cola con capacidad de reordenarse tenemos, por ejemplo, tres tipos de paquetes, donde unos son más prioritarios que otros, esto es equivalente a tener tres colas FIFO, una para cada nivel, con un multiplexor que toma siempre el paquete más prioritario disponible. Por lo tanto, necesitamos una cola FIFO por tipo de servicio. En un esquema de servicios integrados, donde el número de flujos distintos que atraviesan un encaminador es potencialmente muy alto, esta idea sería inviable.

Selección de CV por WRR

Como se ha señalado anteriormente, el esquema de prioridades estrictas puede dar lugar a inanición. Por este motivo esta política es llamada “injusta”. Una política “justa” sería round robin. Este algoritmo establece un turno rotatorio entre las colas que compiten por el enlace, de modo que en cada ciclo va consultándolas de un modo circular. Con este esquema la inanición queda totalmente desechada, pues se tiene la seguridad de que todas

las colas serán atendidas en un tiempo finito.

Un defecto que tiene esta política es que se trata a todos los canales por igual. Aunque esto parece lo más justo, generalmente no es lo más adecuado. Si, por ejemplo, en una cola se tiene tráfico e-mail y en otra una videoconferencia, parece lógico que los paquetes de la videoconferencia obtengan más ancho de banda que los del e-mail.

Para resolver este problema se ha propuesto el algoritmo weigthed round robin [17]. Este algoritmo consiste en asignar unos pesos a los canales virtuales que compiten, de modo que cada canal tenga el turno un número de veces proporcional a su peso antes de cederlo al siguiente canal. De este modo se mantiene la propiedad de no producir inanición de round robin, pero dando un trato más adecuado a los tipos de tráfico que lo necesitan.

Para el modelo de encaminador aquí estudiado se propone un algoritmo basado en el weighted round robin. Consiste en llevar una cuenta de créditos para cada canal virtual. Un paquete puede ocupar el enlace si su canal virtual tiene créditos. Al pasar ese paquete, se descuenta un crédito al canal. Cuando el número de créditos es cero, se mira si otro canal tiene créditos. Podría darse el caso de que los canales con créditos no tengan paquetes para transmitir mientras que paquetes listos para salir estén en colas sin créditos. En este caso se incrementan los contadores de créditos de todas las colas en un número proporcional a su peso de créditos. De este modo se pretende que ráfagas de paquetes del mismo tipo no desequilibren la distribución del tráfico que se define con los pesos de las colas.

4.3.2. Mejoras del encaminador en el SF

Además de en los IAs, para proporcionar calidad de servicio también son necesarios cambios en los conmutadores del switch fabric. En este apartado se describen varias propuestas que pueden considerarse para el modelo de encaminador que se ha propuesto en este capítulo. Estas alternativas se evaluarán también en el Capítulo 6.

Asignación de CV por Nivel de Servicio

En los conmutadores, al igual que en los IAs, también son necesarios mecanismos para identificar el tipo de los paquetes y asignarlos a la cola o canal virtual adecuado. En este caso la identificación podría viajar en una cabecera interna al encaminador, puesta

por el IA tras identificar cada paquete.

Selección de CV por Nivel de Servicio, RR de enlaces

El encaminador que se está considerando implementa VOQ en los buffers de entrada de los conmutadores para evitar el efecto del HOL Blocking. Por este motivo, las colas o canales virtuales de entrada estarán organizados de acuerdo a dos criterios, el enlace de salida y el nivel de servicio.

Como en el IA, las colas son de tipo FIFO. A la hora de elegir qué paquete debe salir por un enlace de salida, el controlador del enlace (link controller) debe decidir qué cola mirar primero de acuerdo a los dos criterios antes expuestos, enlace de salida y nivel de servicio. Una primera aproximación podría consistir en aplicar un round robin entre las colas de los distintos enlaces y después, una vez elegido un enlace de entrada, atender al paquete más prioritario de ese enlace. Después de todo, todos los enlaces son igual de importantes en este modelo.

Selección de CV por Nivel de Servicio, RR en el mismo NS

La política anterior tiene el problema de que un paquete de baja prioridad podría ser atendido antes que otro de mayor prioridad. Esto podría suceder como consecuencia del round robin de enlaces: si en las colas de un enlace sólo hay paquetes de baja prioridad y tiene el turno rotatorio, saldrá un paquete de este tipo aunque en los siguientes enlaces haya paquetes de alta prioridad esperando.

Para resolver esto, el link controller podría cambiar el orden en que mira las colas. Primero miraría todas las de alta prioridad y en caso de empate aplicaría el round robin de enlaces. De este modo no se produciría ningún adelantamiento de paquetes de baja prioridad.

Selección de CV por WRR

Los algoritmos anteriores están basados en esquemas de prioridades estrictas. Por este motivo podría haber paquetes que sufrieran inanición. Para resolver esto se puede utilizar un algoritmo basado en weighted round robin como el planteado para los IAs.

A lo largo de la Sección 4.3 se han propuesto modificaciones en el diseño del encaminador que se ha modelado. En concreto, en la Sección 4.3.1 se han apuntado las posibles mejoras respecto al IA, y en la Sección 4.3.2 se han visto las posibles mejoras en el switch fabric. En los capítulos siguientes se van a evaluar estas opciones y, sobre todo, sus combinaciones para estudiar sus prestaciones.

Capítulo 5

METODOLOGÍA DE EVALUACIÓN

Este capítulo describe el método seguido para evaluar las ideas presentadas en este proyecto. Se indican las características del simulador utilizado y se detallan los índices de prestaciones que se han considerado para evaluar las propuestas. Por último, se indican cuáles han sido finalmente las configuraciones modeladas.

5.1. MÉTODO DE EVALUACIÓN

En la evaluación de prestaciones de cualquier sistema se pueden seguir varios métodos [14], siendo los más comunes:

Mediciones reales Consiste en tomar las medidas directamente sobre un sistema real. Si el sistema no existe, las medidas se toman sobre prototipos. Este tipo de evaluación, cuando es posible, resulta el método más fiable. Sin embargo, a veces resulta demasiado caro, peligroso o poco práctico. Por ejemplo, no resultaría adecuado para pruebas de bombas nucleares o medir la evolución de las galaxias. En el caso de este proyecto habría que construir prototipos del encaminador para ver su funcionamiento, lo que es totalmente impensable.

Modelos analíticos En este caso se construye un modelo matemático del sistema y se intenta obtener conclusiones a partir de él. Este método es rápido y barato, aunque no suele proporcionar demasiado nivel de detalle pues hay que abstraer mucho el sistema modelado, perdiendo precisión en el proceso. Para realizar la evaluación de este proyecto

con el método del modelo analítico se debería crear un modelo del encaminador basado en la teoría de colas. Sin embargo, esta alternativa no da el nivel de detalle que se busca.

Simulaciones Este método consiste en crear un programa de ordenador, el simulador, que abstrae el comportamiento del sistema al nivel de detalle que resulte interesante. La construcción del simulador puede resultar tediosa, pero es una alternativa mucho más barata que los prototipos y da resultados más precisos que un modelo analítico. Por estos motivos se ha optado por este método para evaluar las propuestas de este proyecto.

5.2. SIMULADOR

El simulador usado en este proyecto permite modelar el comportamiento de un único encaminador de tipo corporativo con cierto nivel de detalle. El simulador es dirigido por eventos y simula:

- **el estado** del encaminador en un momento dado, y
- **los cambios** que se producen en cualquiera de los elementos considerados en el encaminador.

Como se ha comentado, el simulador está dirigido por eventos. Estos eventos están asociados a su instante temporal de ocurrencia. Este tiempo permite mantener los eventos ordenados en una cola y que vayan activándose cuando les corresponda. En el estado se considera la ocupación de los buffers, la posición de cada paquete, los créditos, etc. Los cambios pueden ser, por ejemplo, la llegada de un mensaje, la liberación de un enlace o la determinación de la ruta de un paquete.

El modelo de encaminador es el indicado en el Capítulo 4, aunque sin excesivas modificaciones podrían obtenerse otros modelos distintos.

El simulador admite un numeroso conjunto de parámetros de entrada que acaban caracterizando una determinada configuración. Tras la ejecución del programa se obtienen una serie de resultados que permiten evaluar el rendimiento de dicha configuración. En las secciones siguientes se detallan todos estos parámetros de entrada y salida.

Hay que indicar que el núcleo del simulador no ha sido desarrollado por el autor de este proyecto. El simulador en su etapa inicial modelaba un encaminador IP básico,

sin soporte de QoS. El autor de este proyecto ha implementado todas aquellas partes relevantes para las propuestas realizadas.

5.2.1. Arquitectura simulada

El simulador construido recoge las características del modelo de la Sección 4.1, aunque habría que hacer algunas consideraciones.

El simulador modela el comportamiento de un único encaminador, por lo que no se pueden ver las interacciones entre varios encaminadores. Queda como trabajo futuro el implementar esto.

El tamaño de los buffers de los IA se supone siempre lo bastante grande como para absorber el tráfico entrante, sin que esto suponga una merma del realismo de los resultados. En primer lugar, no es de interés tratar de desbordar este buffer porque se estaría en congestión y esa es una situación que no se quiere que alcance el encaminador. En segundo lugar, el mecanismo de control de flujo, que no se simula, se encargaría de impedir que los buffers se llegaran a desbordar. A pesar de suponer buffers grandes, si el número de mensajes alcanza cierto límite el simulador muestra un mensaje de alerta, lo que obliga a descartar los resultados ya que los tiempos de espera en el IA dejan de ser realistas.

Los mensajes no se reordenan en las colas de los canales virtuales, es decir, las colas son FIFO. Esto es así porque tampoco se considera el deadline de los paquetes. En lugar de ello se utilizan tres niveles de servicio, para distinguir tres niveles distintos de demanda en cuanto a latencia máxima. Esto se detalla con más amplitud en la Sección 5.2.4.

Cuando el IA actúa como salida, se supone un comportamiento de sumidero perfecto. Una vez que los paquetes salen del switch fabric, lo que suceda con ellos ya no es de interés para las propuestas de este proyecto. Por otro lado, el sumidero perfecto simula unos buffers de salida bien dimensionados. Tampoco se recomponen los mensajes, tarea que no tiene ninguna utilidad porque no se simulan las etapas posteriores.

5.2.2. Entradas del simulador

Las entradas del simulador se pueden clasificar en varios grupos:

1. Parámetros de la topología de la multietapa

   - El simulador permite simular una multietapa de tipo bloqueante, con diferentes patrones de interconexión entre etapas (perfect-shuffle, bit-reversal, butterfly, etc.).
   - Varios son los algoritmos de encaminamiento incorporados, todos ellos basados en un recorrido bidireccional o de ida y vuelta.
   - El número de IAs, el número de conmutadores, el número de etapas y el número de entradas por conmutador.

2. Parámetros de diseño del encaminador

   - Respecto al input adapter
     - Retardos. Puede indicarse el tiempo de encaminamiento en ciclos de reloj y el tiempo necesario para asignar un enlace de salida, también en ciclos.
     - Número de canales virtuales.
     - Política de asignación de canal virtual. Criterio para decidir a qué canal virtual debe asignarse cada paquete. Las opciones posibles son aleatoria, según destino y según nivel de servicio, como se describe en la Sección 4.3.1.
     - Política de selección de canal virtual. Criterio para decidir qué canal virtual es el más prioritario en cada momento. Las opciones son round robin, mensaje más viejo, mensaje más priotario o weighted round robin, como se muestra en la Sección 4.3.1.
     - Pesos de los niveles de servicio para WRR. Indican la importancia relativa de los distintos niveles de servicio.
   - Respecto al enlace
     - Tiempo de vuelo. El número de flits que puede haber en el enlace en un momento dado. Dependerá de la longitud del cable, del retardo de propagación y del ancho de banda.
     - Tiempo de transferencia. Tiempo necesario para poner un flit en el enlace.
   - Respecto al conmutador
     - Retardos. En ciclos de reloj puede indicarse el tiempo de encaminamiento, el tiempo de transferencia por el conmutador y el tiempo necesario para asignar un enlace de salida.
     - Tamaño de los buffers de entrada. Dado en créditos, donde cada crédito equivale a un paquete.
     - Número de canales virtuales.

- Política de asignación de canal virtual. Criterio para decidir a qué canal virtual debe asignarse cada paquete. Las opciones posibles son según destino y según nivel de servicio, como se describe en la Sección 4.3.2.
- Política de selección de canal virtual. Criterio para decidir qué canal virtual es el más prioritario en cada momento. Las opciones son round robin, mensaje más priotario con round robin de enlaces, prioridades estrictas o weighted round robin, como se muestra en la Sección 4.3.2.
- Pesos de los niveles de servicio para WRR.

3. Parámetros de carga de la red

   - Distribución de los destinos de los mensajes. El simulador implementa algunas de las distribuciones de destinos más comunes: uniforme, uniforme con radio de localidad, bit-reversal y hot-spot.
   - Longitud de los mensajes. Hay dos tipos de mensajes, largos y cortos, pudiendo especificarse la distribución de cada tipo. La longitud se distribuye uniformemente entre dos valores.
   - Tasa de generación de los mensajes. Se especifica un tiempo mínimo y un tiempo máximo. Los tiempos obtenidos se distribuyen uniformemente entre ellos.

   Estos parámetros relativos a la carga de la red podrían ser proporcionados al simulador de diversas formas:

   - *Carga sintética*. Consiste en la generación aleatoria del tráfico (direcciones destino, tamaño de los mensajes, tiempo entre ellos), según distintos patrones.
   - *Trazas*. Las trazas son ficheros con información sobre el tráfico que atraviesa un sistema real. El simulador lee el fichero de traza y genera los eventos adecuados.
   - *Carga real de aplicaciones*. Consiste en ejecutar aplicaciones reales sobre el simulador. Desde el punto de vista de las aplicaciones no hay diferencia entre el simulador y un auténtico encaminador, así que el tráfico generado es el más realista.

   Teniendo en consideración las ventajas e inconvenientes de cada método se optó por la carga sintética, aunque en un futuro se pretende evaluar las propuestas realizadas con tráfico más realista.

4. Parámetros de QoS

- Número de niveles de servicio. Se numeran desde *1* hasta *n*, donde el *0* es el más prioritario y el *n* el que menos prioridad tiene.
- Distribución de los niveles de servicio. Porcentaje de tráfico generado de cada nivel de servicio.

5. Parámetros específicos del simulador

   - Mensajes durante la simulación. La condición de parada de la simulación no es un número de ciclos, sino la recepción de un número de mensajes.
   - Mensajes en el periodo transitorio. Se trata del número de mensajes que se ignoran para la toma de resultados y para la condición de parada. Con este parámetro se intenta evitar el efecto de una red descargada antes de estabilizarse.
   - Semilla para el generador de números aleatorios.

De las entradas anteriores, se incorporaron como parte de este proyecto las referidas a calidad de servicio, incluyendo los pesos de weighted round robin. También se añadieron diversas políticas de asignación y selección de canal virtual, tanto en el IA como en el switch fabric.

5.2.3. Salidas del simulador

El simulador modela la evolución del encaminador y de sus distintos componentes, así como del tráfico que se va tratando. La simulación termina cuando se ha alcanzado un determinado número de paquetes. En ese momento el simulador ofrece, entre otros, los siguientes resultados:

- Número de ciclos simulados.
- Mensajes recibidos. Debe coincidir con la condición de parada.
- Latencia media de la cabecera y desviación típica. Este tiempo es el transcurrido desde que la cabecera de un mensaje entra en el switch fabric hasta que se recibe en el IA destino.
- Latencia media y desviación típica. Es el tiempo transcurrido desde que un paquete entra en el switch fabric hasta que se recibe en el IA destino.

- Latencia media desde la generación y desviación típica. Tiempo transcurrido desde que un mensaje entra en el IA origen hasta que su último paquete llega al IA destino.
- Productividad. Viene dada en flits/ciclo y flits/ciclo/nodo.

Además, para cada nivel de servicio se ofrecen los siguientes resultados:

- Número de mensajes generados y porcentaje respecto del total.
- Número de flits inyectados.
- Latencia media de cruce del switch fabric.
- Espera media en los buffers de los IAs.
- Productividad. Viene dada en tanto por 1 y se calcula dividiendo los flits salientes entre los flits entrantes.

5.2.4. Abstracción de los tipos de tráfico

Como se mostró en el Capítulo 2, las aplicaciones pueden demandar garantías en cuanto al ancho de banda o la latencia. Los tipos de tráfico que se vieron en la Sección 2.6.2 se simularon mediante el uso de 5 niveles de servicio, tal y como se refleja en la Tabla 5.1.

Tabla 5.1: Niveles de servicio

Nivel	**Tráfico**
1 2 3	Deadline
4	Ancho de banda
5	Best Effort y Challenged

El tráfico sensible al deadline se modeló mediante los niveles 1, 2 y 3, donde se supone que el tráfico del nivel de servicio 1 necesita salir antes que el tráfico del nivel de servicio 2 y el tráfico del nivel de servicio 2 antes que el tráfico del nivel de servicio 3. Los paquetes no llevan un deadline como tal, así que no es posible hacer una ordenación efectiva de los paquetes. Sin embargo, esto se abstrae mediante los tres niveles de servicio. En los resultados de la simulación habrá que comprobar que la latencia de cada nivel es igual o mejor que la del nivel inmediatamente inferior.

El tráfico sensible a ancho de banda se modela con el nivel de servicio 4. En este caso el parámetro más relevante es la productividad. Los mensajes tampoco llevan el requisito de ancho de banda específico, ni hay una reserva previa de recursos. De este modo el encaminador no puede darle a cada flujo el ancho de banda justo. Se limita a proporcionar el mejor servicio posible en función de la prioridad de cada paquete.

El tráfico best effort y el challenged se modela con el nivel de servicio 5. Aunque para este tipo de tráfico no hay que optimizar nada, generalmente se considera que debe tener una productividad mínima para no deteriorar demasiado la percepción de los usuarios. Este tipo de tráfico hace necesaria y posible la calidad de servicio. Necesaria porque, si no estuviera, la red iría muy descargada y todos los mensajes irían a las máximas prestaciones. Posible porque todas las mejoras de los otros niveles de servicio se hacen a costa del tráfico best effort.

5.3. ÍNDICES DE PRESTACIONES

En esta sección se destacan los índices de prestaciones que se han considerado. Estas medidas se obtendrán poniendo el encaminador en una zona próxima a la congestión, para, como se explica más adelante, evaluar la provisión de QoS en situaciones con elevada carga. Los índices de prestaciones considerados son:

Latencia de cruce El tiempo que tarda un paquete en cruzar el switch fabric. Aquí no se considera el tiempo que se esperó en el IA.

La latencia es importante para el tráfico de tiempo real, como videoconferencias o telemedicina. Además, es importante que no varíe demasiado, *jitter* pequeño.

Espera en el IA Tiempo que cada paquete estuvo esperando en los buffers del IA. Sumado a la latencia de cruce nos proporciona el tiempo total que estuvo un paquete en el encaminador, la latencia desde la generación, que es la que realmente importa.

Productividad Viene expresada en tanto por uno y se obtiene de dividir las palabras salientes entre las palabras entrantes.

La productividad es importante para aplicaciones que deben mover una gran cantidad de datos, como transferencias de ficheros o vídeo.

Tasa de inyección Se trata del parámetro que se utilizará como base en la mayoría de las comparaciones y mide el número de palabras por ciclo y nodo que se ponen en el IA.

La tasa de inyección se reparte entre los distintos niveles de servicio según se indique. Los resultados interesantes se obtienen con una tasa de inyección próxima al punto de saturación. Si la red va descargada, entonces los resultados carecen de interés porque todo va bien. Si la red está demasiado cargada, entonces todo va mal y eso tampoco es interesante.

5.4. MODELOS SIMULADOS

En esta sección se detallan las características de los modelos que se han simulado. Con las diversas modificaciones incorporadas al simulador, se ha seleccionado un conjunto de configuraciones que se diferencian en algunos aspectos del IA y de los conmutadores. A cada una se le ha asignado un identificador que permitirá diferenciarla posteriormente cuando en una misma gráfica se presenten los resultados obtenidos de cada una de ellas.

5.4.1. Configuraciones de encaminadores

Las propuestas del Capítulo 4 se concretan en distintos modelos de encaminadores cuyo rendimiento queremos comparar. Estos encaminadores se simulan proporcionando distintas configuraciones al simulador. Todas ellas tienen la misma arquitectura y el mismo switch fabric, las diferencias se centran en aprovechar las redundancias que se ponían de manifiesto en el Capítulo 4.

Se han considerado 8 configuraciones de encaminadores para el estudio que quedan reflejadas en la Tabla 5.2.

Las abreviaturas o códigos utilizados en la Tabla 5.2 tiene el siguiente significado:

- **Conf**: Nombre de la configuración.
- **CVs**: Número de canales virtuales para implementar QoS, sin contar los necesarios para implementar VOQ. El número total de canales virtuales sería el producto de éstos y los usados para atenuar o eliminar el HOL Blocking.

Tabla 5.2: Configuraciones del encaminador.

	IA				**Switch Fabric**			
Conf	CVs	A. CV	S. CV	VOQ	CVs	A. CV	S. CV	VOQ
A	5	NS	WRR	Sí	5	NS	WRR	Sí
B	5	NS	WRR	Sí	5	NS	NS-G	Sí
C	5	NS	WRR	Sí	3	NS-M	NS-G	Sí
D	5	NS	WRR	2	3	NS-M	NS-R	Sí
E	5	NS	WRR	Sí	1	-	RR	Sí
F	1	-	RR	Sí	5	NS	WRR	Sí
G	5	NS	NS	Sí	5	NS	NS-G	Sí
H	1	-	RR	Sí	1	-	RR	Sí

- **A. CV**: Política de asignación de canal virtual a un paquete que llega. NS indica que se asigna de acuerdo al nivel de servicio, NS-M es como la anterior, pero multiplexando varios niveles en un solo canal virtual y - indica que se asigna en función del destino, sin tener en cuenta las necesidades de QoS.
- **S. CV**: Política de selección de canal virtual. WRR indica Weighted Round-Robin, NS significa esquema de prioridades estricto, NS-G significa esquema de prioridades estricto y global, NS-R significa esquema de prioridades estricto con round robin de enlaces y RR indica round robin.
- **VOQ**: Virtual Output Queuing. Si aparece un número, indica el número de canales virtuales que se usan para atenuar el efecto del HOL Blocking, no se trata de VOQ estrictamente, porque todavía podría producirse bloqueo de cabeza de línea.

El ciclo de reloj es de la misma duración en todas las configuraciones, pero esto no es del todo realista. Las configuraciones más simples probablemente permitirían acortar el ciclo de reloj, dando lugar a latencias más bajas. Sin embargo, para conocer con exactitud cuanto baja ese ciclo de reloj habría que realizar complejos estudios que escapan totalmente a los objetivos de este proyecto.

A continuación se detallan más cada una de las configuraciones de la Tabla 5.2.

Configuración A

Simula los encaminadores de altas prestaciones más complejos. Debe proporcionar los resultados de referencia para el resto de configuraciones, pues es el diseño más complejo y a partir del cual se tratarán de hacer todas las simplificaciones.

En el IA se consideran los 5 niveles de servicio en forma de 5 canales virtuales. Los paquetes se asignan al canal virtual que corresponde a su nivel de servicio. La selección del siguiente paquete a sacar se hace mediante weighted round robin. Se implementa VOQ para evitar el efecto del HOL Blocking.

También se consideran dentro del switch fabric los 5 niveles de servicio en 5 canales virtuales. Cada paquete se asigna de acuerdo a su nivel de servicio y se selecciona el siguiente a salir mediante weighted round robin. Se implementa VOQ para evitar los efectos del HOL Blocking.

Configuración B

Esta configuración trata de aprovechar una primera redundancia. En lugar de aplicar weighted round robin en el IA y en el switch fabric, probablemente baste con hacerlo en el IA. En lugar de weighted round robin, los conmutadores del switch fabric, sus link controllers en realidad, seleccionan siempre el paquete más prioritario de los que esperan para salir por ese enlace.

Configuración C

Esta vez se trata de explotar otra redundancia, si los paquetes sensibles a retraso ya vienen ordenados, volver a separarlos en canales virtuales no debería aportar mucha mejora. Para comprobarlo se usan sólo tres canales virtuales en los conmutadores del switch fabric. La política de asignación de canal virtual es, en este caso, por nivel de servicio multiplexado.

Configuración D

Implementar VOQ en el IA es costoso, hace que se multiplique el número de canales virtuales necesarios. Sin embargo, el hecho de clasificar el tráfico en niveles de servicio también contribuye a reducir el efecto del HOL Blocking. Por este motivo se crean sólo dos grupos de colas, haciendo la asignación mediante la operación de módulo del identificador de destino.

Por otro lado, en el switch fabric el link controller aplica un round robin de enlaces de entrada y es dentro de cada enlace donde se aplica un esquema de prioridades absoluto.

Se trata de un algoritmo más sencillo de implementar si los buffers de cada enlace están separados.

Esta configuración explota todas las redundancias detectadas. Su diseño sería mucho más sencillo y barato de implementar que las anteriores y sin embargo, si las hipótesis del Capítulo 3 se cumplen, no debería experimentar una gran reducción de prestaciones.

Configuración E

Con esta configuración se pretende comprobar si es necesario separar en otro canal virtual el tráfico best effort del resto, tal como se sugiere en la Sección 4.2.1. Para ello se utiliza un switch fabric lo más sencillo posible, que no distingue entre niveles de servicio. La única mejora que incorpora es el VOQ.

Configuración F

La configuración F pretende evaluar la importancia del tratamiento de los paquetes que se hace en el IA. Para ello define un IA que se limita a pasar paquetes al SF y cuya única mejora es el VOQ para evitar el efecto del HOL Blocking. De este modo, si los resultados empeoran, quedará claro que no se debe descuidar esta etapa.

Configuración G

Esta configuración pretende valorar el efecto del algoritmo weighted round robin en los índices de prestaciones. Por ese motivo se utiliza un criterio de prioridades absoluto en los conmutadores del switch fabric y en los IAs.

Con esta configuración se verá si el ahorro de usar prioridades estrictas, en forma de un diseño más simple, vale la pena.

Configuración H

Esta configuración no implementa ningún tipo de mejora de QoS. Se define para proporcionar la referencia inferior al resto de configuraciones.

5.4.2. Parámetros específicos del simulador

A la hora de realizar las simulaciones, se han utilizado valores habituales para este tipo de estudios. En concreto, los valores utilizados para los parámetros de entrada del simulador son:

- Mensajes durante el periodo transitorio: 100000
- Número de mensajes para sacar resultados: 200000
- Tiempo de transferencia de una palabra por un canal físico: 1
- Tiempo de transferencia de una palabra por el conmutador: 1
- Tiempo de encaminamiento: 1
- Tiempo de vuelo por el canal: 8
- Tiempo necesario para asignar el canal de salida: 1
- Longitud media de los mensajes = 16.0
- Distribución de destinos de los mensajes: Uniforme
- Capacidad (en paquetes = créditos) de los buffers: 11
- Mensajes generados por nivel:
  - Nivel 1: 2.50 %
  - Nivel 2: 2.50 %
  - Nivel 3: 5.00 %
  - Nivel 4: 30.00 %
  - Nivel 5: 60.00 %
- Peso de los mensajes para WRR en los IAs, por nivel:
  - Nivel 1: 1
  - Nivel 2: 1
  - Nivel 3: 2
  - Nivel 4: 8
  - Nivel 5: 14
- Peso de los mensajes para WRR en el SF, por nivel:

- Nivel 1: 5
- Nivel 2: 5
- Nivel 3: 3
- Nivel 4: 6
- Nivel 5: 10

Capítulo 6

EVALUACIÓN DE PRESTACIONES

En este capítulo se verá una primera sección de evaluación de configuraciones individuales. En ella se dan los índices de prestaciones de cada configuración en forma gráfica y se comentan los resultados más significativos. Después de esto, en la siguiente sección se comparan las configuraciones A a E, para cada uno de los índices considerados. Estas configuraciones, de la A a la E, son las que tratan de explotar las redundancias expuestas en el Capítulo 4.

6.1. EVALUACIÓN DE CONFIGURACIONES

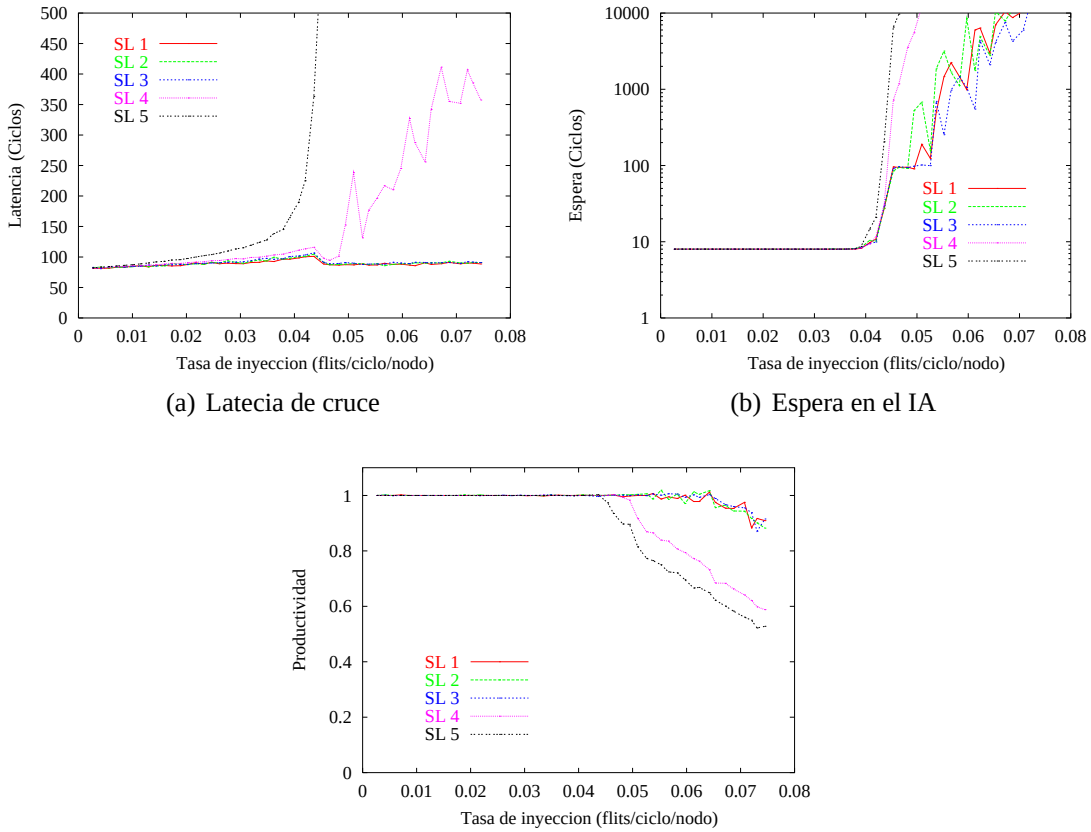
Esta sección se centra en la evaluación de las prestaciones de las distintas configuraciones presentadas en el capítulo anterior. A modo de referencia rápida, se reproduce de nuevo la tabla de configuraciones.

Tabla 6.1: Configuraciones del encaminador.

	IA				**Switch Fabric**			
Conf	CVs	A. CV	S. CV	VOQ	CVs	A. CV	S. CV	VOQ
A	5	NS	WRR	Sí	5	NS	WRR	Sí
B	5	NS	WRR	Sí	5	NS	NS-G	Sí
C	5	NS	WRR	Sí	3	NS-M	NS-G	Sí
D	5	NS	WRR	2	3	NS-M	NS-R	Sí
E	5	NS	WRR	Sí	1	-	RR	Sí
F	1	-	RR	Sí	5	NS	WRR	Sí
G	5	NS	NS	Sí	5	NS	NS-G	Sí
H	1	-	RR	Sí	1	-	RR	Sí

6.1.1. Configuración A

La configuración A es la más parecida al modelo de referencia porque trata de proporcionar calidad de servicio sin aprovechar ninguna de las redundancias. Se trata de la marca a batir por el resto de configuraciones.



(a) Latecia de cruce

(b) Espera en el IA

(c) Productividad

Figura 6.1: Resultados obtenidos por la configuración A.

En la Figura 6.1 (a) puede observarse la latencia de la configuración A para los niveles de servicio considerados. Se puede ver que el sistema entra en congestión alrededor de una tasa de inyección de 0.04 flits/nodo/ciclo. Antes de llegar a ese punto la latencia del nivel 5 (el menos prioritario) ya era claramente mayor. También crece la latencia de cruce del nivel 4. La latencia de cruce de los niveles 1 a 3 se mantiene estable.

Como se puede observar, la latencia de los niveles 1 a 3 hace un amago de subir para una tasa de inyección de 0.043 flits/nodo/ciclo, pero después baja y se mantiene en niveles estables. Esto sucede porque en el límite de la saturación, paquetes de todos los niveles de servicio ocupan los buffers de los conmutadores, lo que aumenta la latencia de los niveles de mayor prioridad. Sin embargo, a una tasa de inyección algo mayor, hay tantos paquetes de alta prioridad, que los de baja prioridad apenas tienen ocasión de entrar en el switch fabric, de modo que los buffers se reparten prácticamente entre los tres niveles

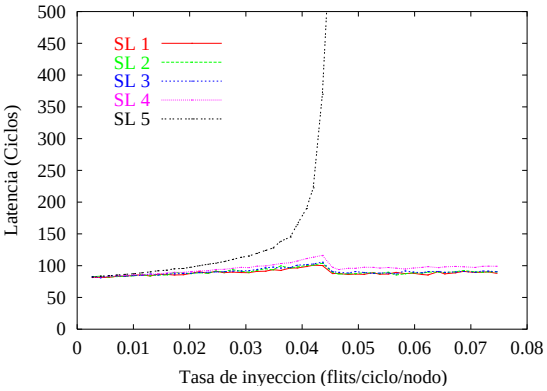
más altos (recordemos que los canales virtuales son colas lógicas, tratando de aprovechar al máximo el espacio).

La espera en el IA (Figura 6.1 (b)) crece para todos los niveles al llegar el punto de saturación. Sin embargo, puede apreciarse que los niveles de prioridad más altos suben más lentamente. De todos modos, llegados a una tasa de inyección lo bastante alta, la espera es demasiado grande para todos los niveles. Puede apreciarse una forma de "dientes de sierra" entre los niveles más prioritarios. Esto es debido a que cuando un nivel de servicio obtiene unos resultados más bajos es a costa del tráfico de los otros niveles de servicio.

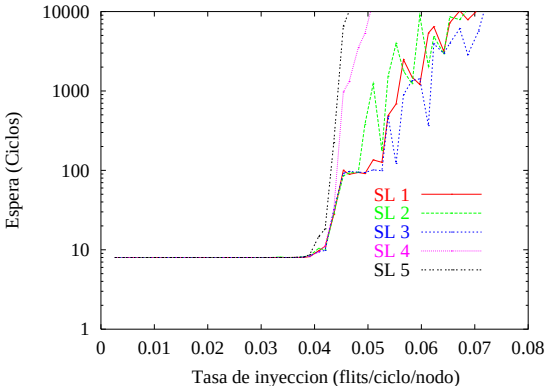
En cuanto a la productividad, en la Figura 6.1 (c) se aprecia como el tráfico best effort es el primero en sentir la congestión, después lo hace el nivel cuatro y por último empiezan a bajar los niveles más prioritarios.

6.1.2. Configuración B

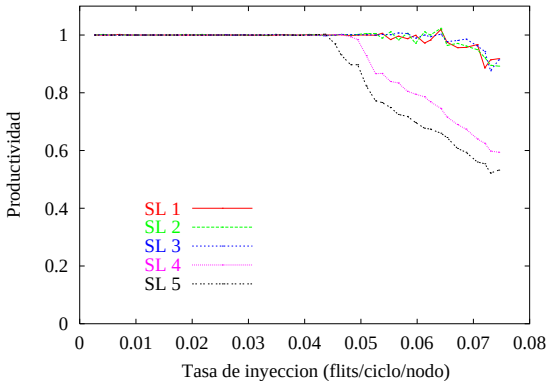
La configuración B trata de aprovechar una primera redundancia. En lugar de aplicar WRR en el IA y en el switch fabric, trata de ver si basta con hacerlo en el IA. Los conmutadores del switch fabric, sus link controller en realidad, seleccionan siempre el paquete más prioritario de los que esperan para salir por ese enlace, en lugar de usar weighted round robin.



(a) Latecia de cruce



(b) Espera en el IA



(c) Productividad

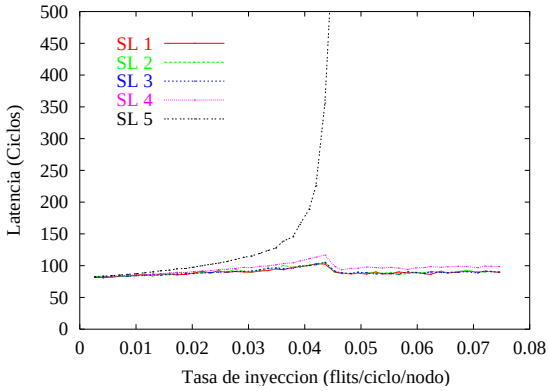
Figura 6.2: Resultados obtenidos por la configuración B.

Los resultados de esta configuración se muestran en la Figura 6.2 y se diferencian de la anterior fundamentalmente en la latencia de cruce. La latencia de cruce mejora en los niveles 1 a 4 a costa del 5, que empeora con el esquema de prioridades estricto. Esto no es un problema, mientras los paquetes sigan saliendo (buena productividad), ya que el nivel 5 es best effort y no es sensible a retrasos.

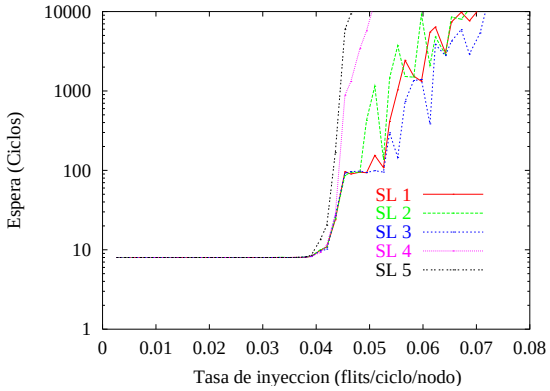
En cuanto a la espera en el IA y la productividad, parece claro que el comportamiento es muy similar, aunque esto se verá mejor en la Sección 6.2.

6.1.3. Configuración C

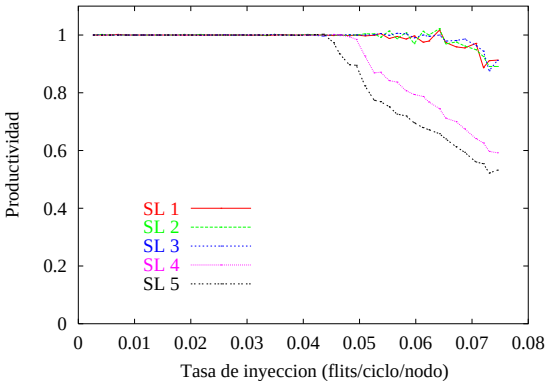
Esta configuración trata de explotar otra redundancia, si los paquetes sensibles a retraso ya vienen ordenados, volver a separarlos en canales virtuales no debería aportar mucha mejora. Para comprobarlo, en la configuración C se usan sólo tres canales virtuales para implementar niveles de servicio en los conmutadores del switch fabric. La política de asignación de canal virtual es, en este caso, por nivel de servicio multiplexado.



(a) Latecia de cruce



(b) Espera en el IA



(c) Productividad

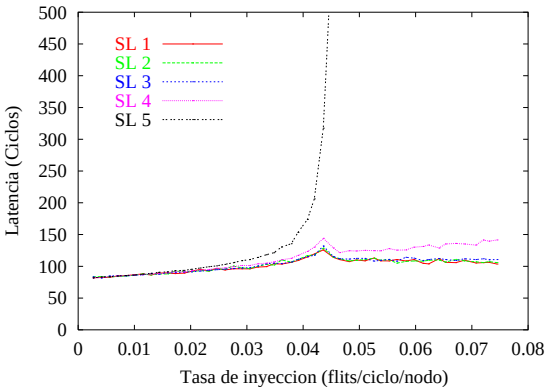
Figura 6.3: Resultados obtenidos por la configuración C.

Los resultados de esta configuración se muestran en la Figura 6.3 y son casi idénticos a los de la configuración B, lo que son buenas noticias, porque ahora se tienen dos canales virtuales menos en los conmutadores.

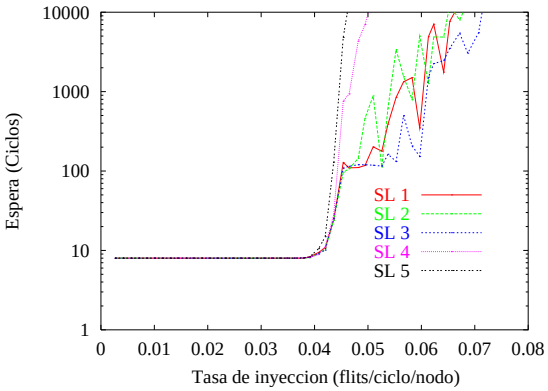
6.1.4. Configuración D

La configuración D presenta varias simplificaciones. En primer lugar, dado que clasificar el tráfico en niveles de servicio contribuye a reducir el efecto del HOL Blocking, quizás no sea necesario implementar un VOQ estricto. Por eso se prueba a usar sólo dos canales virtuales para reducir el efecto del HOL Blocking.

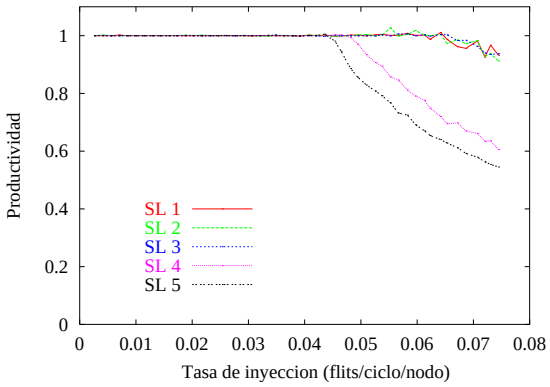
En segundo lugar, en el switch fabric el link controller aplica un round robin de enlaces de entrada y es dentro de cada enlace donde se aplica un esquema de prioridades absoluto. Se trata de un algoritmo más sencillo de implementar si los buffers de cada enlace están separados.



(a) Latecia de cruce



(b) Espera en el IA



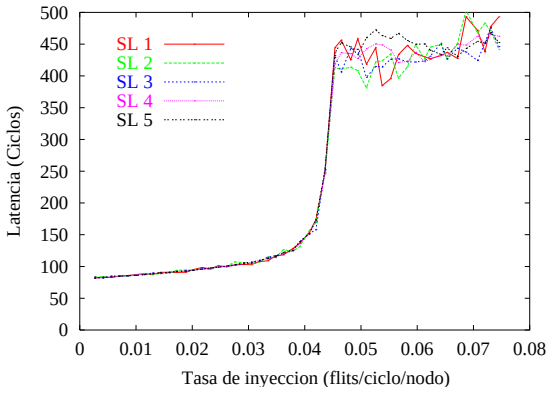
(c) Productividad

Figura 6.4: Resultados obtenidos por la configuración D.

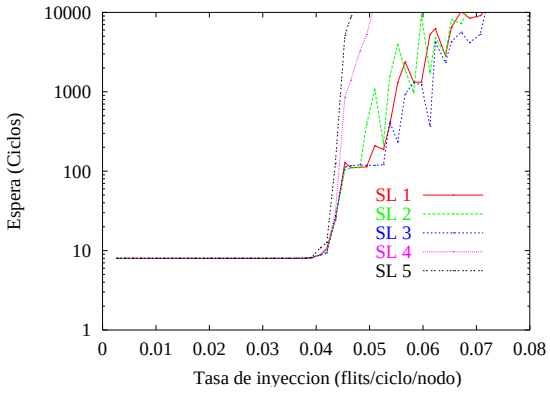
Los resultados de esta configuración se muestran en la Figura 6.4 y en las gráficas se aprecia que los resultados son muy parecidos a los de las configuraciones anteriores. Esto es bueno, porque se trata de la configuración más sencilla hasta ahora.

6.1.5. Configuración E

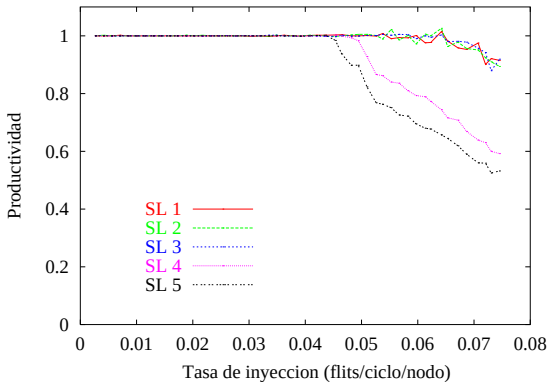
Con esta configuración se pretendía comprobar si era necesario separar en otro canal virtual el tráfico best effort del resto, tal como se sugería en la Sección 4.2.1. Para ello se utilizó un switch fabric lo más sencillo posible, que no distinguía entre niveles de servicio, con la única mejora del VOQ.



(a) Latecia de cruce



(b) Espera en el IA



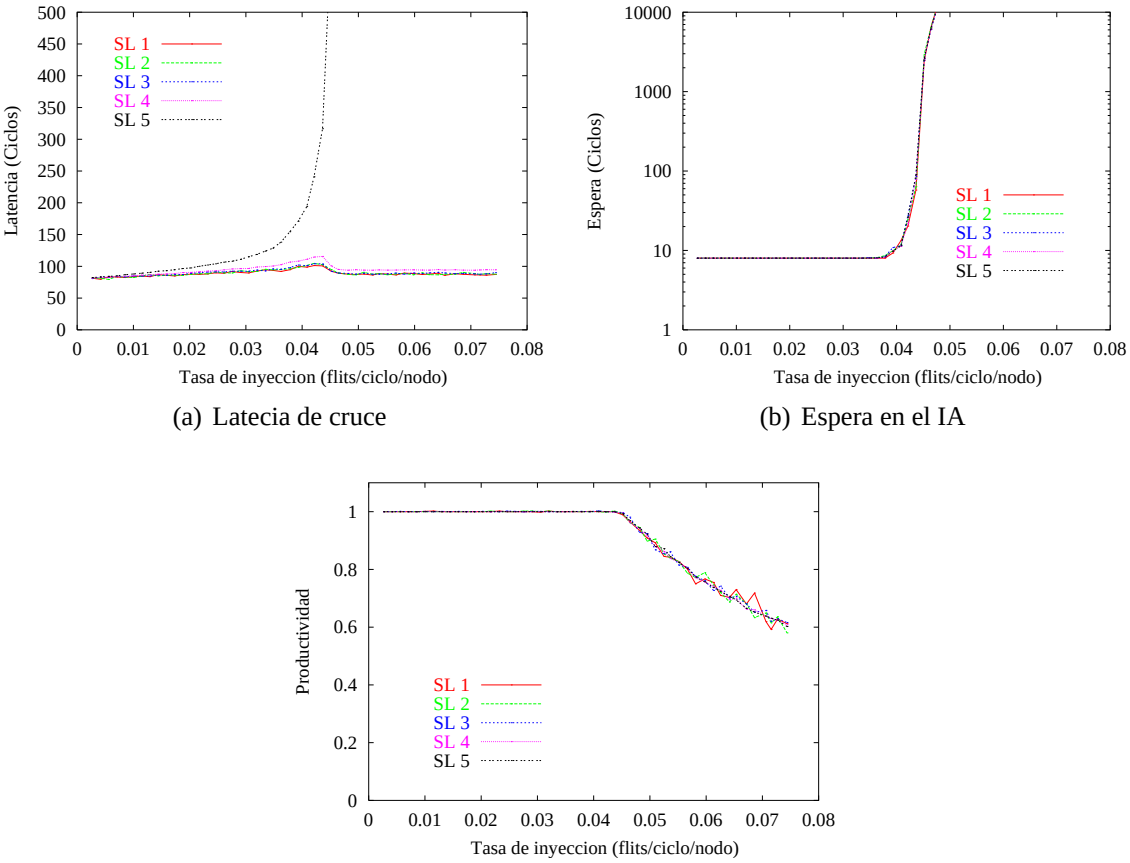
(c) Productividad

Figura 6.5: Resultados obtenidos por la configuración E.

Los resultados de esta configuración se muestran en la Figura 6.5 y, como cabía espererar, no hay diferencias entre las latencias de cruce de los distintos niveles (Figura 6.5 (a)). Esto es muy malo, ya que no se proporciona QoS en este aspecto. Sin embargo, el valor importante es la latencia total, que incluiría la espera en el IA. En la espera en el IA (Figura 6.5 (b)) sí que hay diferencias. De hecho se trata de una gráfica muy similar a las anteriores, al igual que la de productividad (Figura 6.5 (c)).

6.1.6. Configuración F

La configuración F pretendía evaluar la importancia del tratamiento de los paquetes que se hace en el IA. Para ello definía un IA que se limitaba a pasar paquetes al SF y cuya única mejora era el VOQ para evitar el efecto del HOL Blocking.



(a) Latecia de cruce

(b) Espera en el IA

(c) Productividad

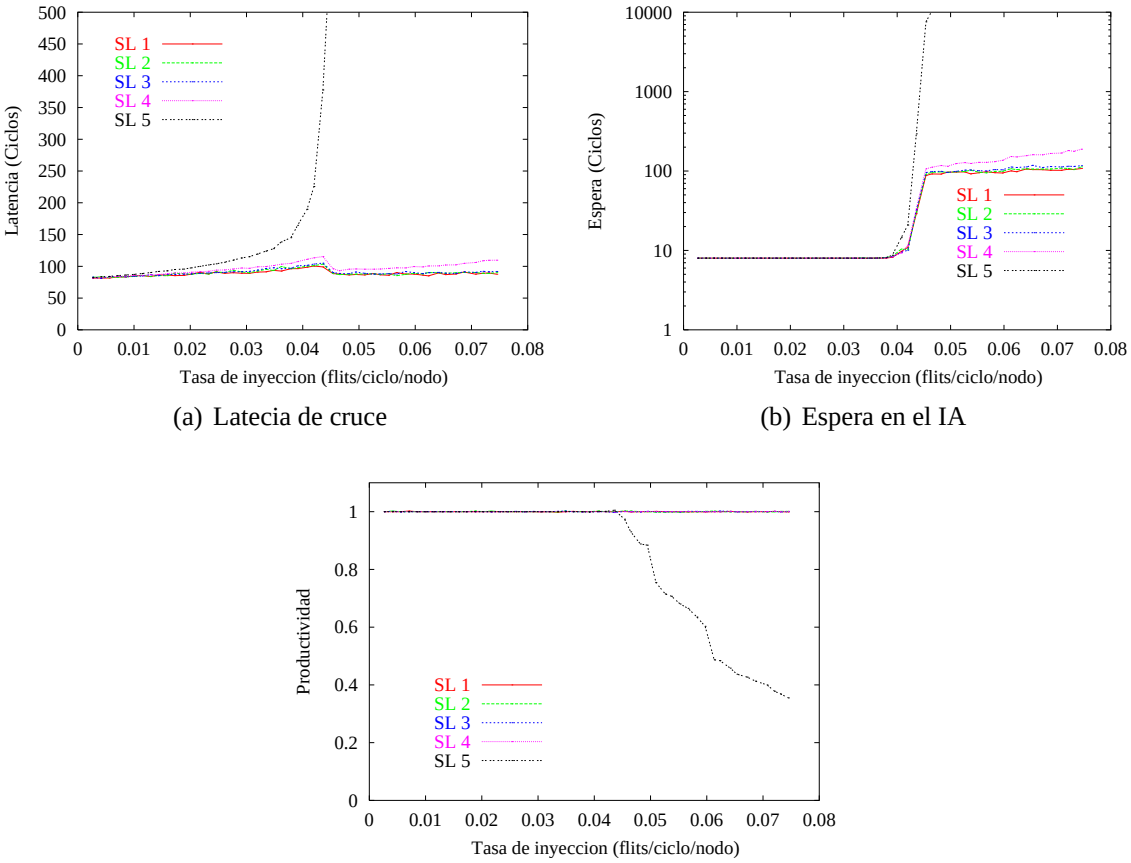
Figura 6.6: Resultados obtenidos por la configuración F.

Los resultados se muestran en la Figura 6.6 y en esta configuración sólo hay verdaderas diferencias entre niveles de servicio en la latencia de cruce (Figura 6.6 (a)). La espera en el IA (Figura 6.6 (b)) es la misma para todos los niveles, mientras que las diferencias en la productividad (Figura 6.6 (c)) son mínimas, comparadas con las que había en las configuraciones A a D. Todo ello a pesar de que el switch fabric sí implementa todas las mejoras que se plantearon en la configuración A.

Esta configuración pone de manifiesto que para simplificar el diseño de los encaminadores no basta con ir quitando cosas, sino que hay que hacer un estudio previo como el de los Capítulos 3 y 4 para detectar donde hay redundancias y cómo tratarlas.

6.1.7. Configuración G

La configuración G pretendía valorar el efecto del algoritmo weighted round robin en los índices de prestaciones. Por ese motivo se utilizaba un criterio de prioridades absoluto en los conmutadores del switch fabric y en los IAs.



(a) Latecia de cruce

(b) Espera en el IA

(c) Productividad

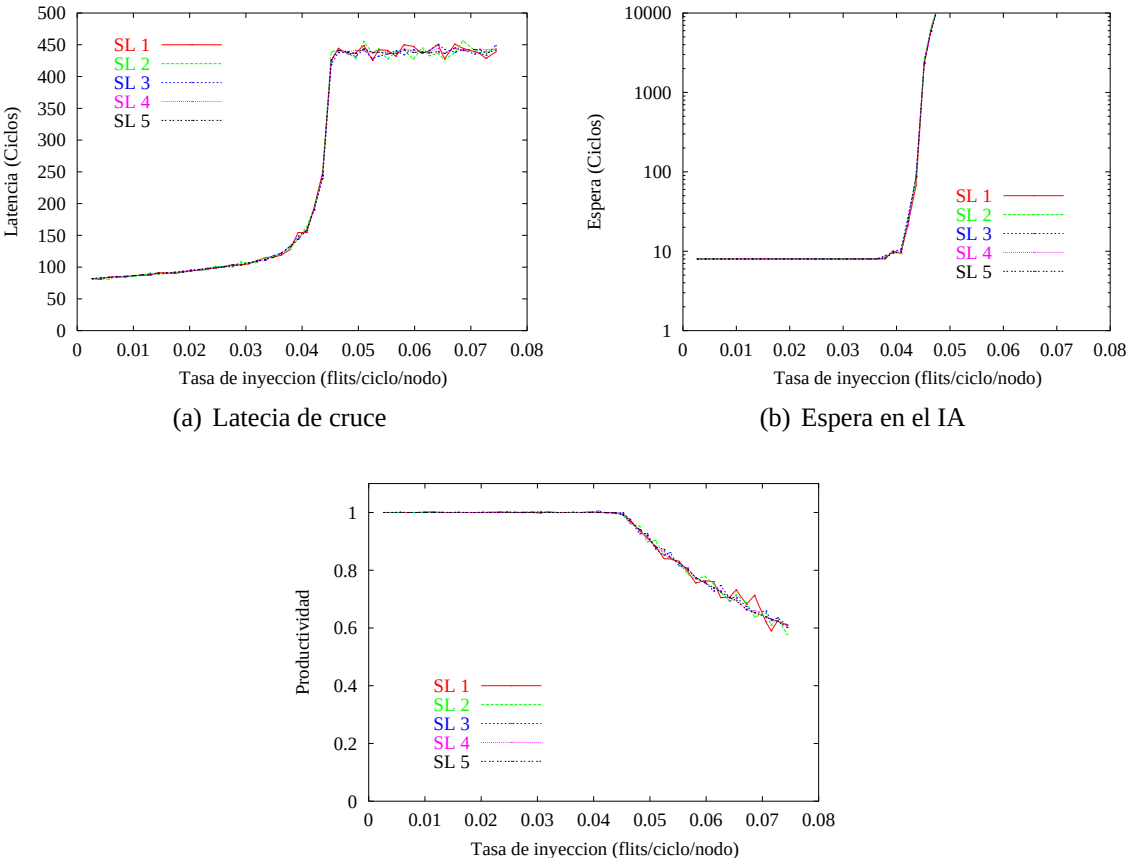
Figura 6.7: Resultados obtenidos por la configuración G.

Los resultados de esta configuración se muestran en la Figura 6.7 y en esta ocasión el esquema de prioridades estricto hace que todos los niveles vayan bien a costa del 5, lo que se aprecia bien en la Figura 6.7 (c). Sólo cuando la tasa de inyección se hace muy grande empieza a empeorar el 4. Este esquema penaliza demasiado al tráfico best effort, sobre todo en lo que respecta a la productividad.

Con esta configuración se tiene otro ejemplo de simplificación excesiva. A la hora de simplificar el diseño hay que tratar de no perder funcionalidad, como sucede en este caso.

6.1.8. Configuración H

La configuración H no implementaba ningún tipo de política de QoS. Se trataba de señalar la referencia inferior para el resto de configuraciones.



(a) Latecia de cruce

(b) Espera en el IA

(c) Productividad

Figura 6.8: Resultados obtenidos por la configuración H.

Los resultados se muestran en la Figura 6.7 y en esta configuración todos los niveles tienen el mismo comportamiento al no haber tratamiento de QoS.

6.2. COMPARATIVA DE LAS CONFIGURACIONES A, B, C, D Y E

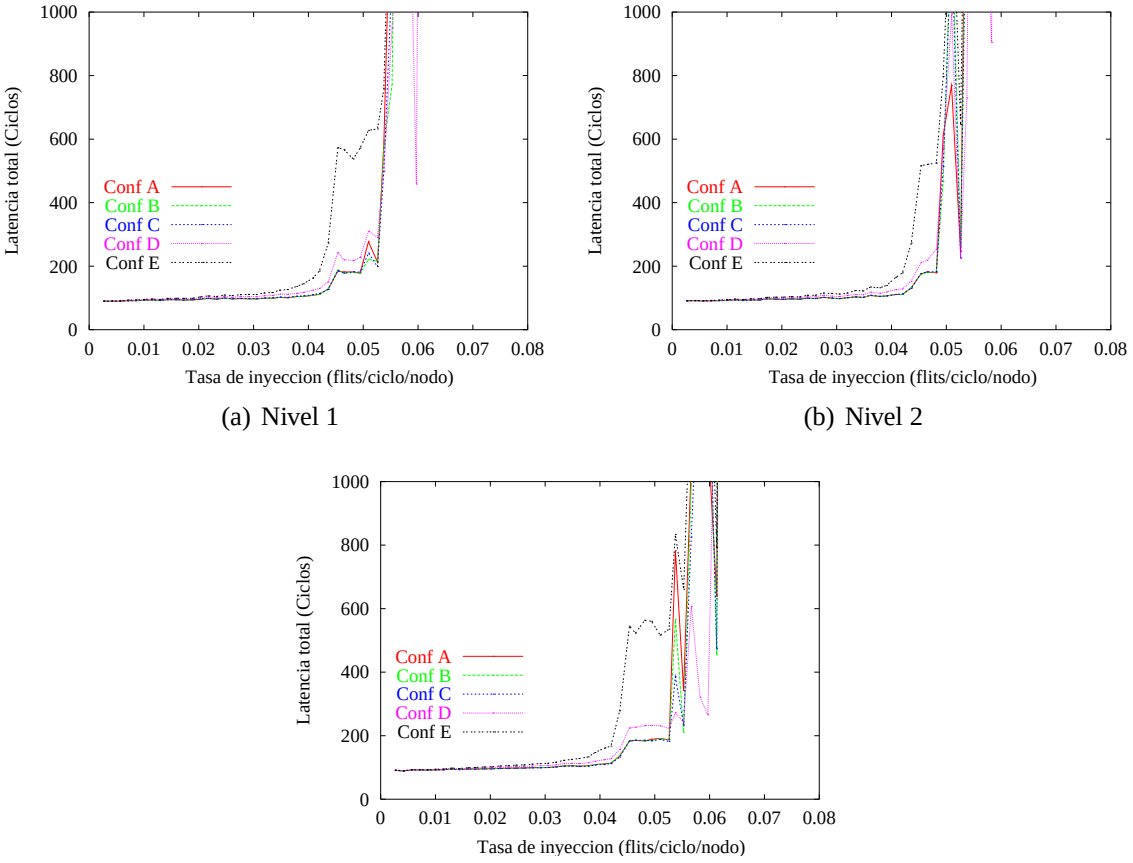
Las configuraciones de la A a la E son las más interesantes para los objetivos del proyecto porque son las que intentan aprovechar las redundancias expuestas en el Capítulo 4. Para ver claramente si se mantienen las prestaciones eliminando redundancias, se comparará a continuación las prestaciones de cada nivel de servicio en cada configura-

ción.

Las gráficas de latencia de esta sección muestran la latencia total, que incluye el tiempo de espera en el IA y el tiempo de cruce del switch fabric. De este modo, se refleja mejor cómo afecta al tráfico el paso por el encaminador.

6.2.1. Tráfico sensible a retraso

Es esta sección se ha estudiado el comportamiento del tráfico sensible al retraso (niveles de servicio del 1 al 3) para las configuraciones más importantes y los índices de prestaciones estudiados.



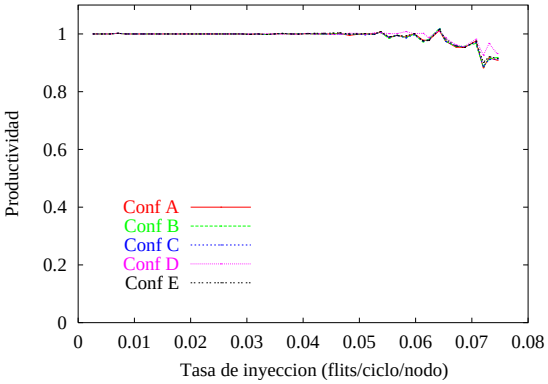
(a) Nivel 1

(b) Nivel 2

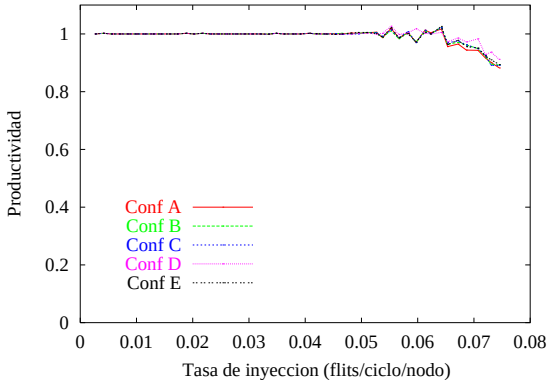
(c) Nivel 3

Figura 6.9: Latencia total del tráfico sensible a retraso.

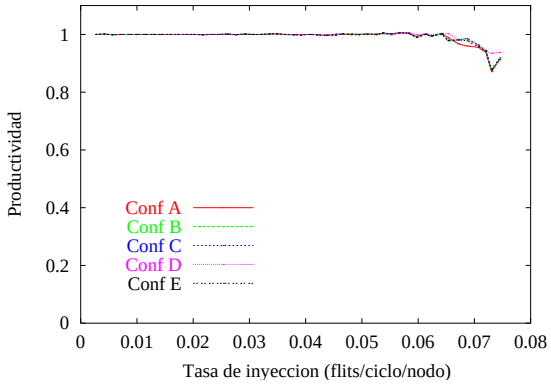
En la Figura 6.9 se puede ver el tiempo total empleado en el encaminador por el tráfico de los niveles de servicio del 1 al 3. Recordemos que este tráfico presenta restricciones en cuanto a latencia máxima. Las configuraciones A, B y C tienen un rendimiento muy similar. En la configuración D, al usar round robin de enlaces antes del esquema de prioridades estrictas en los conmutadores, las latencias son un poco mayores en la franja



(a) Nivel 1



(b) Nivel 2



(c) Nivel 3

Figura 6.10: Productividad del tráfico sensible a retraso.

desde que comienza la saturación hasta que el tráfico sensible a retraso se ve seriamente afectado.

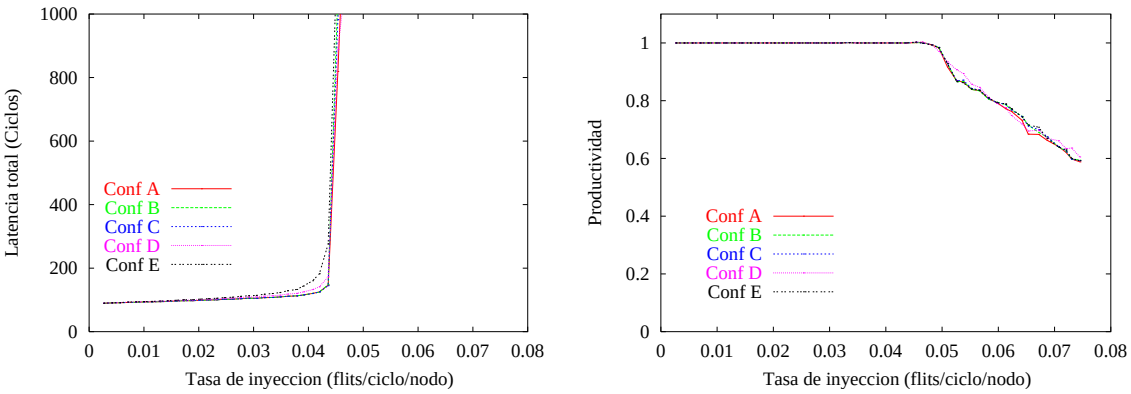
La configuración E es claramente inferior, porque en los conmutadores no aplica prioridad ninguna. Esto confirma que los razonamientos realizados en la Sección 4.2.1 eran correctos, porque no basta con ordenar el tráfico sólo en los IAs, sino que es necesario algún procesamiento posterior, aunque sea poco, como en la configuración D.

En cuanto a la productividad, en la Figura 6.10 queda claro que es prácticamente la misma en las cinco configuraciones para los tres niveles de servicio. De este modo queda claro que el punto que más influencia tiene en la productividad del encaminador es el IA. Esto probablemente es debido a que sus buffers son mucho mayores.

En lo que respecta al tráfico sensible a retraso, la configuración D, con un diseño más sencillo que el resto de configuraciones, mantiene un nivel de prestaciones similar, tal y como se quería.

6.2.2. Tráfico sensible a ancho de banda

Esta sección se centra en evaluar los resultados del tráfico sensible a ancho de banda en las configuraciones más importantes.



(a) Latencia total (b) Productividad

Figura 6.11: Tráfico sensible a ancho de banda.

En la Figura 6.11 (a) se aprecia que apenas hay diferencias entre las distintas configuraciones en el trato al tráfico sensible a ancho de banda. La configuración E es un poco peor, otra vez debido a que no distingue niveles de servicio en los conmutadores del switch fabric.

La Figura 6.11 (b) muestra que la productividad es casi la misma en todas las configuraciones. Todas aplican la misma política en el IA.

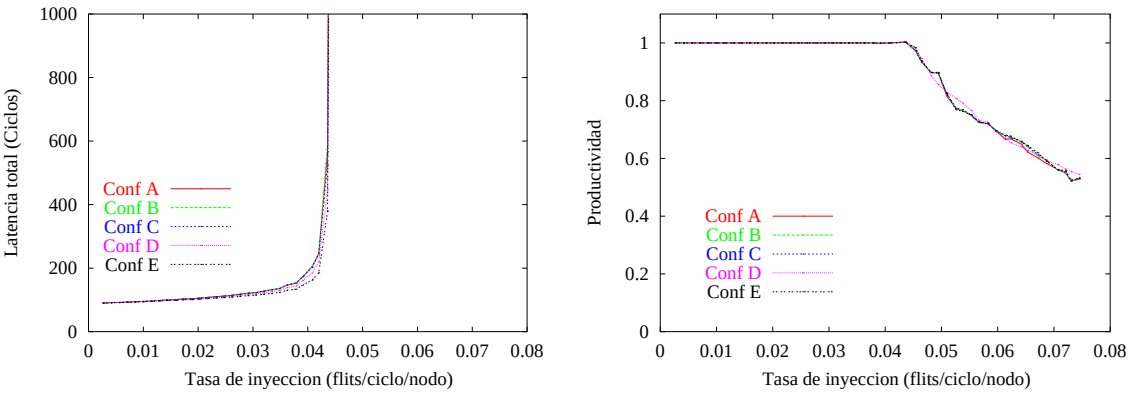
De nuevo la configuración D sigue a la altura de las demás, a pesar de su diseño más sencillo.

6.2.3. Tráfico best effort

El último tipo de tráfico, por lo que respecta a este estudio, es el best effort. En esta sección se verá como se le trata en las configuraciones que se vienen considerando.

El tráfico best effort se comporta de forma casi idéntica en todas las configuraciones (Figura 6.12 (a)). Como cabría esperar, la configuración E es un poco más benigna que el resto, porque no le penaliza en los conmutadores del switch fabric.

La productividad del tráfico best effort es prácticamente la misma en las cinco configuraciones (Figura 6.12 (b)).



(a) Latencia total (b) Productividad

Figura 6.12: Tráfico best effort.

De nuevo la configuración D se mantiene a la altura. Se puede concluir en este punto que las propuestas del Capítulo 4 han llevado a un diseño más sencillo del encaminador que, sin embargo, mantiene la funcionalidad y las prestaciones.

Capítulo 7

CONCLUSIONES

Para finalizar, en este capítulo se incluyen las principales conclusiones del estudio aquí presentado así como algunas de las líneas y tareas que podrían dar continuidad al mismo.

7.1. APORTACIONES

En numerosas ocasiones, a lo largo de este documento, se ha insistido en el hecho de que los diseños actuales de encaminadores, y/o conmutadores como elementos integrantes de éstos, se caracterizan por una creciente complejidad. Esta complejidad se debe, fundamentalmente, a que se están incorporando continuas actualizaciones encaminadas a resolver problemas determinados, sin tener en cuenta la interacción entre todas esas propuestas, es decir sin considerar el sistema en su conjunto.

En opinión de algunos investigadores con una amplia experiencia en estos temas, en el diseño de los conmutadores, y de los encaminadores que los usan, se está alcanzando una situación similar a la que se llegó hace ya más de 20 años con los procesadores CISC y que motivó la propuesta de los procesadores RISC.

Según estos mismos expertos, y aprendiendo de aquella experiencia, sería aconsejable tender a una simplificación del diseño de los conmutadores, considerando el sistema (encaminador) en su conjunto y la interacción entre sus diferentes componentes, en lugar de abordar un diseño considerando cada uno de ellos de forma totalmente independiente.

Inspirados en este planteamiento, se ha realizado el estudio que recoge esta memo-

ria, centrado en encaminadores IP, y que básicamente ha pretendido analizar la interacción entre la planificación que se lleva a cabo en los inputs adapters y la planificación que se realiza en el switch fabric, de la misma manera en la que los diseñadores de procesadores podrían considerar la interacción entre procesadores y compiladores.

Para realizar este estudio, en los capítulos anteriores se ha abordado un estudio del diseño actual de los encaminadores. En él, se han buscado posibles redundancias en el tratamiento que se hace de los paquetes, especialmente el tratamiento de QoS.

El estudio teórico llevó a pensar que quizás los conmutadores estaban repitiendo parte del trabajo que ya había sido hecho por el input adapter. Esto llevó al planteamiento de nuevos modelos de encaminador que, sin reducir prestaciones, tuviesen un diseño más simple.

La evaluación realizada en el Capítulo 6 mostró que, tal como se preveía, las nuevas configuraciones, a pesar de ser más simples, mantenían unas prestaciones similares. En concreto, se ha observado que la denominada configuración D, con un diseño más sencillo que otras de las configuraciones evaluadas, mantiene un nivel de prestaciones similar. Esta configuración D incorpora sólo dos grupos de colas en el input adapter haciendo la asignación de los paquetes mediante la operación módulo del identificador del destino. Por otro lado, en el switch fabric se plantea un algoritmo mucho más sencillo de implementar para seleccionar el siguiente paquete. Básicamente, consiste en aplicar round robin entre los enlaces de entrada, y para cada uno de ellos un esquema de prioridades.

Es decir, aprovechando la planificación realizada en los input adapters es posible reducir el número de canales virtuales en el switch fabric, sin introducir por ello HOL Blocking, ni disminuir las prestaciones o alterar el comportamiento global de la planificación. Esta reducción del número de canales virtuales, así como la simplificación de los algoritmos utilizados, se traduce en una disminución del espacio requerido para su implementación, y por tanto una simplificación en el diseño.

Hay que tener en cuenta que una simplificación en el diseño de los encaminadores es importante porque, aunque no aporta ninguna funcionalidad nueva, puede contribuir a abaratar costes y a obtener arquitecturas más simples. Estas nuevas arquitecturas probablemente permitan ciclos de reloj más cortos, mejorando el rendimiento del encaminador.

Así pues, a partir de los resultados obtenidos, es razonable dar continuidad a este estudio y seguir profundizando en la simplificación del diseño de los encaminadores IP de altas prestaciones manteniendo su funcionalidad y rendimiento. De esta forma, en encaminadores como el Avici TSR habría que estudiar a fondo si existen redundancias que

pudieran permitir dichas simplificaciones.

Por último, cabe resaltar que como consecuencia del trabajo realizado para este proyecto se ha escrito un artículo que será presentado próximamente:

A. Martínez, F. J. Alfaro, J. L. Sánchez, J. Duato: *Una Propuesta para Simplificar el Diseño de Encaminadores IP con QoS.* XIV Jornadas de Paralelismo. Madrid (España). Septiembre 2003.

7.2. TRABAJO FUTURO

El trabajo aquí realizado ha aportado buenos resultados, pero que no deben entenderse como un punto y final. Así pues, este estudio abre la puerta a otras posibles líneas de investigación, como las siguientes:

- Utilizar un esquema de reserva de recursos (Servicios Integrados). Esto podría permitir un dimensionado más adecuado de los recursos, además de poder utilizar políticas más complejas para ofrecer QoS.
- Comprobar las interacciones y posibles redundancias entre varios encaminadores. Sería interesante simular una red completa y buscar redundancias entre todos sus elementos.
- Variar la proporción de los mensajes generados para cada uno de los niveles de servicio.
- Utilizar trazas de tráfico real. Quizás el uso de tráfico sacado de encaminadores de Internet ofrezca una perspectiva interesante para este estudio.
- Considerar otros parámetros del diseño del encaminador, como el número de input adapters, el tamaño o la organización de los buffers, etc.
- Ampliar este estudio más allá del entorno de Internet, buscando redundancias en los encaminadores o conmutadores de otros tipos de redes, como por ejemplo, redes de área local.
- Realizar un estudio con VHDL u otro lenguaje de definición de hardware, para ver cuál es el impacto real de las mejoras, en cuanto a área de silicio y disminución del ciclo de reloj.

Bibliografía

[1] T. Anderson, S. Owicki, J. Saxe, and C. Thacker. High-speed switch scheduling for local area networks, 1993.

[2] J. Aweya. On the design of ip routers part 1: Router architectures. *Journal of Systems Architecture*, 46:483–511, July 2000.

[3] S. Blake, D. Back, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An Architecture for Differentiated Services. Internet Request for Comment RFC 2475, Internet Engineering Task Force, December 1998.

[4] R. Braden, D. Clark, and S. Shenker. Integrated Services in the Internet Architecture: an Overview. Internet Request for Comment RFC 1633, Internet Engineering Task Force, June 1994.

[5] S. F. Bryant and D. L. A. Brash. The decnis 500/600 multiprotocol bridge/router and gateway. *Digital Technical Journal*, 5(1), 1993.

[6] William J. Dally. Scalable switching fabrics for internet routers. Technical report, Computer Systems Laboratory, Stanford University and Avici Systems, Inc., 2003.

[7] DARPA. RFC 791: Internet protocol. Internet Request for Comment RFC 791, Internet Engineering Task Force, September 1981.

[8] Asociación de Usuarios de Internet. Historia de internet. `http://www.aui.es/historia/ihistoria.htm`.

[9] J. Duato, S. Yalamanchili, and L. Ni. *Interconnection networks*. Morgan Kaufman, 2002.

[10] C. Partridge et al. A 50Gb/s IP Router. *IEEE/ACM Trans. on Networking*, 6(3):237 – 248, June 1998.

[11] M. W. Garret. *Contributions Toward Real-Time Services on Packet-Switched Networks*. PhD thesis, CU/CTR/TR 340-93-20, Universidad de Columbia, Mayo 1993.

[12] InfiniBand Trade Association. *InfiniBand Architecture Specification Volume 1. Release 1.0*, October 2000.

[13] ITU-T. Video Coding for Low Bitrate Communication, ITU-T Recommendation H.263. Technical report, International Telecommunication Union, 1996.

[14] R. Jain. *The art of computer system performance analysis: techniques for experimental design, measurement, simulation and modeling*. John Wiley and Sons, Inc., 1991.

[15] M. Karol and M. Hluchyj. Queuing in high-performance packet-switching. *IEEE Select. Areas. Commun.*, 6:1587–1597, 1998.

[16] M. Karol, M. Hluchyj, and S. Morgan. Input versus output queueing on space-division packet switching. *IEEE Transactions on Communications*, 35(12):1347–1356, December 1987.

[17] M. Katevenis, S. Sidiropoulos, and C. Corcoubetis. Weighted round-robin cell multiplexing in a general-purpose ATM switch. *IEEE Journal on Selected Areas in Communications*, Oct. 1991.

[18] C. Kruskal and M. Snir. The performance of multistage interconnection networks for multiprocessors. *IEEE Transactions on Computers*, C-32(12):1091–1098, December 1983.

[19] D. Love, S. Yalamanchili, J. Duato, M.B. Caminero, and F.J. Quiles. "Switch scheduling in the Multimedia Router (MMR)". In *Proceedings de International Parallel and Distributed Processing Symposium (IPDPS)*. IEEE Computer Society, Mayo 2000.

[20] N. McKeown. *Scheduling Algorithms for input-queued cell switches*. PhD thesis, University of California at Berkeley, 1995.

[21] Nick McKeown, Martin Izzard, Adisak Mekkittikul, William Ellersick, and Mark Horowitz. Tiny Tera: A packet switch core — using new scheduling algorithms to build a 1-terabits packet switch with a central hub no larger than a can of soda. *IEEE Micro*, 17(1):26–33, 1997.

[22] NUA. How many online? http://www.nua.ie.

[23] D. Packer Ali. Gigabit, terabit and optical routers: A design & architectural survey. Technical report, Kent State University, May 2001.

[24] White Paper. The integrated network services switch architecture and technology. Technical report, Berkeley Networks, 1997.

[25] Joe Pelissier. Providing Quality of Service over Infiniband Architecture Fabrics. In *Proceedings of the 8th Symposium on Hot Interconnects*, August 2000.

[26] Avici Systems. Avici Terabit Switch Router. http://www.avici.com. http://www.avici.com.