Índice

Acrónimos

AAL ATM Adaptation Layer.

ACON All Class in One Network.

ARP Address Resolution Protocol.

ASDR Adaptive State Dependent Routing.

ATM Asynchronous Transfer Mode.

BGP Border Gateway Protocol.

CAC Connection Admission Control.

CBF Constrained Belman-Ford.

CBR Constant Bit Rate.

DAR Dynamic Alternative Routing.

DCR Dynamic Controlled Routing.

DNHR Dynamic NonHierarchical Routing.

DNS Domain Name System.

EASDR Extended Adaptive State Dependent Routing.

EGP Exterior Gateway Protocol.

ES-IS End System to Intermediate System.

GCAC Generalized Connection Admission Control.

GOS Grade of Service.

IDRP IS-IS inter-Domain Routing Protocol.

ISDN Integrated Service Digital Network.

IETF Internet Engineering Task Force.

IGP Interior Gateway Protocol.

IP Internet Protocol.

IPX Internetwork Packet eXchange.

IS-IS Intermediate System to Intermediate System.

ISP Internet Service Provider.

LLR Least Loaded Routing.

MDP Markov Decision Problem.

MFA Mean-Field Annealing.

MPLS Multi Protocol Label Switching.

NP Non Polynomial.

OPSF Open Shortest Path First.

OCON One Class in One Network.

OSI Open System Interconnection.

PNNI Protocol Network to Network Interface.

QoS Quality of Service.

RDSI Red Digital de Servicios Integrados.

RFC Request For Comment.

RIP Routing Information Protocol.

RNA Red Neuronal Artificial.

RSVP Resource reSerVation Protocol.

RTC Red Telefónica Conmutada.

RTNR Real-Time Network Routing.

SAAL Signaling ATM Adaptation Layer.

SNA System Network Architecture

SSCF Service Specific Coordination Function.

SSCOP Service Specific Connection Oriented Protocol.

STR State and Time-Dependent Routing.

QoS Quality of Service.

VBR Variable Bit Rate.

VC Virtual Channel.

VP Virtual Path.

WAN Wide Area Network (redes de area extensa).

WFQ Weighted Fair Queuing.

Lista de símbolos

a_k **:** Número de fuentes activas (enviando paquetes), de las x_k de la clase k presentes en el enlace.

$A_k(M\Delta T)$ **:** Tráfico de clase k medido al final del intervalo $M\Delta T$.

$\hat{A}_k((M+1)\Delta T)$ **:** Tráfico predicho para el final del intervalo $(M+1)\Delta T$.

$AVG(i)$ **:** Valor medio de la conexión *i-ésima*.

AVG_k **:** Valor medio de las conexiones de la clase k.

b_k **:** Ancho de banda requerido por una conexión de la clase k.

b : Vector $[b_1, \ldots, b_K]$.

b^* **:** Relación entre momentos v/m

$B(N, \rho)$ **:** Función de *Erlang-B*.

B_{ij} **:** Probabilidad de bloqueo de llamadas en el enlace e_{ij}.

$B_{k,ij}$ **:** Probabilidad de rechazar una llamada de la clase k en el enlace e_{ij}.

B_{med} **:** Probabilidad media de bloqueo de llamada de las K clases.

B_{max} **:** Probabilidad máxima de bloqueo individual de las K clases.

$\widetilde{B}_{ij}$ **:** Porcentaje de tiempo que el enlace está "bloqueado", es decir, que no hay recursos suficientes para aceptar una llamada.

$\widetilde{B}_{k,ij}$ **:** Porcentaje de tiempo que el enlace está "bloqueado" para las llamadas de tipo k, es decir, el porcentaje de tiempo en que se cumple la condición $C_{ij} - c_{ij} < b_k$.

BW **:** Ancho de banda residual.

BW_{ij} **:** Ancho de banda residual del enlace e_{ij}.

BW_{of} **:** Ancho de banda ofrecido al sistema.

BW_{rec} **:** Ancho de banda rechazado por el sistema.

c_{ij} **:** Ancho de banda ocupado en el enlace e_{ij}.

C **:** Capacidad de un enlace.

C_{ij} **:** Capacidad en unidades de ancho de banda del enlace e_{ij}

$coste_{ij}$ **:** Coste asociado a la utilización del enlace e_{ij}

$coste_P$ **:** Coste total del camino P.

$coste_{k,ij}$ **:** Coste de llevar una llamada de la clase k por el enlace e_{ij}.

$coste_{k,P}$ **:** Coste de transportar una llamada de la clase k por el camino P.

$D(P, b_k, \theta_k)$ **:** Retardo máximo en el camino P.

e_{ij} **:** Enlace que conecta a los nodos i, j.

E **:** Número total de enlaces en el sistema.

$Jitter(P, b_k, \theta_k)$ **:** *Jitter* a lo largo del camino P.

K **:** Número de clases de tráfico en el sistema.

l **:** Ancho de banda ocupado normalizado de un enlace ($l \leq 1$).

l_k **:** Número de unidades de ancho de banda normalizadas de las conexiones de la clase k.

$l_k^{[virtual]}$ **:** Exceso de tráfico de la clase k.

L **:** número máximo de unidades de ancho de banda normalizadas del enlace $L = \lfloor C/\Delta_c \rfloor$

m **:** Media resultante de la superposición de K clases distintas.

$MAX(i)$ **:** Velocidad de pico de la conexión *i-ésima*.

MAX_k **:** Velocidad de pico de las conexiones de la clase k.

n **:** Número de enlaces de los que consta un camino.

N **:** Número máximo de conexiones en un enlace.

$o_j^{[s]}$ **:** Valor de salida de la neurona *j-ésima* de la capa s.

$p\ (l)$ **:** Probabilidad normalizada de que estén ocupados l unidades de ancho de banda normalizada.

$p_k(a_k)$ **:** Probabilidad de que haya a_k fuentes activas de clase k.

$\tilde{p}\ (l)$ **:** Probabilidad no normalizada de que estén ocupados l unidades de ancho de banda normalizadas.

$prop_{ij}$ **:** Retardo de propagación del enlace e_{ij}.

Pr **:** Probabilidad de pérdida de ancho de banda.

$q_k^{[virtual]}$ **:** Tasa de pérdidas virtual.

$q_k^{[max]}$ **:** Tasa de pérdidas máxima de paquete permitida para la clase k.

r_{s-1} **:** Número de neuronas de la capa $s-1$.

s **:** Identificador de la capa de la RNA.

S_{ij} **:** Zona válida del control de admisión del enlace e_{ij}.

$S_{k,ij}$ **:** Subconjunto de estados de S_{ij} en el cual el control de admisión admitirá una llamada de la clase k.

t_{fin}^n **:** Intervalo de tiempo en que finalizó la estimación de tráfico de la ventana enésima.

$T_{intervalo}$ **:** Intervalo de observación usado por la actualización por umbral adaptativo.

Th **:** Umbral de actualización empleado para determinar cuándo se debe de enviar un nuevo mensaje de actualización de estado.

To_{max} **:** Tamaño máximo que puede tener un paquete en la red.

$u_j^{[s]}$ **:** Potencial interno de la neurona *j-ésima* de la capa s.

$u(e_{ij})$ **:** Fracción normalizada del ancho de banda ocupado ($\frac{c_{ij}}{C_{ij}}$) en el enlace e_{ij}.

v **:** Varianza resultante de la superposición de K clases distintas.

w_{ij}^s **:** Valor del peso sináptico que une la neurona j de la capa s con la neurona i de la capa $s-1$.

x **:** Número de conexiones presentes en un enlace.

x_k **:** Número de conexiones de la clase k presentes en un enlace.

$x_{k,ij}$ **:** Número de conexiones de la clase k que hay en el enlace e_{ij}.

$\mathbf{X}$ **:** Matriz de orden KxE que modela el estado del sistema.

z **:** Restricción *z-ésima*.

Z **:** Número de restricciones simultáneas que se deben cumplir.

γ_k **:** Carga media soportada por un enlace.

$\Gamma_k(n)$ **:** Restricción de coste impuesta a las llamadas de la clase k.

Δ_c **:** Máximo común divisor de los anchos de banda b_k.

θ_k **:** Valor máximo que puede tomar el contador del *Leaky Bucket*) para la conexión k.

λ **:** Tasa de llamadas ofrecida al sistema.

λ_{ij} **:** Tasa de llamadas ofrecidas al enlace e_{ij}

$\lambda_{k,ij}$ **:** Tasa de llamadas de la clase k ofrecida al enlace e_{ij}.

$\widetilde{\lambda}_{ij}$ **:** Tasa de llamadas cursadas por el enlace e_{ij}.

$\widetilde{\lambda}_{k,ij}$ **:** Tasa de llamadas de la clase k cursadas por el enlace e_{ij}.

$1/\mu$ **:** Duración media de las llamadas del sistema.

$1/\mu_k$ **:** Duración media de las llamadas de la clase k.

ρ **:** Tráfico ofrecido al sistema, siendo $\rho = \lambda/\mu$.

$\rho_{k,ij}$ **:** Tráfico de la clase k ofrecido al enlace e_{ij}, siendo $\rho_{k,ij} = \lambda_{k,ij}/\mu_k$

$\sigma^2(i)$ **:** Varianza de la velocidad binaria de la conexión *i-ésima*.

σ_k^2 **:** Varianza de la velocidad binaria de las conexiones de la clase k.

ς_k **:** Velocidad de decremento del *Leaky Bucket* para la conexión k.

$\pi_{ij}(\mathbf{x})$ **:** Probabilidad de que el enlace e_{ij} se encuentre en el estado $\mathbf{x}$.

ϕ **:** Tasa de actualización objetivo de la actualización por umbral adaptativo.

$\psi_j^{[s]}$ **:** Función de activación de las neuronas.

ω_{ij}^d **:** Valor del parámetro de calidad de servicio *d-ésimo* en el enlace e_{ij}.

Índice de Tablas

Índice de Figuras

Capítulo 1

Introducción

Uno de los grandes desafíos en las modernas redes de comunicaciones de servicios integrados es transportar información entre los usuarios, asegurando una cierta calidad de servicio o *QoS* (*Quality of Service*) y garantizando su propia naturaleza multiservicio. Es decir, estas redes deberán ser diseñadas para poder transportar una multiplicidad de servicios distintos con requisitos heterogéneos y soportar futuros servicios con requisitos todavía desconocidos. Este escenario diverge enormemente de la situación planteada hasta hace poco, donde para cada servicio existía una red dedicada. Estas circunstancias exigen nuevas maneras de plantear el control y gestión del tráfico cursado.

La necesidad de satisfacer una cierta calidad de servicio demanda, a diferencia de las redes basadas en datagramas como la Internet clásica, la necesidad de trabajar con tráfico orientado a conexión, de forma similar a como han trabajado las redes telefónicas tradicionales. Sin embargo, a diferencia de estas últimas, donde sólo se transporta una única clase de tráfico, las nuevas redes deben ser capaces de transportar un tráfico heterogéneo, donde se superponen tráfico con requisitos estrictos de retardo y *Jitter* (como es el caso de la voz o el vídeo), con tráficos de datos donde los parámetros de retardo son secundarios y el principal requisito a satisfacer es entregar los datos de forma correcta y sin errores.

Está claro que las nuevas necesidades precisan de un nuevo tipo de redes capaces de trabajar con tráfico heterogéneo y, simultáneamente, cumplir con los requisitos de calidad exigidos. Con este fin surgió la red *ATM* (*Asynchronous Transfer Mode*), con el objetivo de ser una red de propósito general orientada a conexión capaz de transportar

diferentes tipos de tráfico con diversos requerimientos de calidad. En las redes multiservicio como *ATM*, la calidad se garantiza mediante la sinergia de un conjunto de controles que actúan a distinto nivel (control de admisión, vigilancia, controles reactivos...). En este sentido, uno de los más importantes mecanismos para garantizar la calidad de servicio es el encaminamiento. El encaminamiento, a diferencia de otros controles existentes en las modernas redes multiservicio, es un control informativo, ya que, por sí solo, no toma decisiones de aceptación o rechazo de llamadas. Su misión es sugerir cuál es el mejor camino posible para realizar la conexión e indicar si, con la información de que se dispone, existen suficientes recursos para que la conexión pueda ser completada con éxito. Las redes *ATM*, al igual que la Red Telefónica Conmutada (RTC), están diseñadas para trabajar con circuitos (virtuales en el caso de la red *ATM* y circuitos físicos en el caso de la red telefónica), por lo que los primeros algoritmos de encaminamiento diseñados para trabajar en este tipo de redes fueron, en principio, adaptaciones de los desarrollados para la RTC, los cuales se modificaron para poder trabajar en un entorno multiservicio. Las redes telefónicas se caracterizan por presentar una tasa de interconexión muy densa y, además, por tener en el centro de decisión de encaminamiento un alto grado de conocimiento del estado de la red. Los algoritmos y protocolos de encaminamiento desarrollados para las redes *ATM* también parten del hecho de que éstas redes exhiben una muy alta tasa de interconexión, gracias a los enlaces virtuales, y de que los mecanismos de toma de decisión disponen de una información lo suficientemente precisa del estado de la red. Los algoritmos de encaminamiento desarrollados para la RTC, así como para la red *ATM*, están diseñados para explotar estas características a fin de obtener un alto rendimiento.

Por otro lado, cuando se comenzó el desarrollo de la red *ATM* se pensó que esta sería la única red que existiría, desplazando a todas las demás. No obstante, en la actualidad el escenario ha cambiado de forma sustancial debido al auge de *Internet*. Si bien inicialmente el protocolo *IP* en el que se basa *Internet* no soporta comunicaciones orientadas a conexión y calidad de servicio, los nuevos protocolos (*IPv6* [Hui97] y *MPLS* [CDF+99] [RVC99]) permiten establecer circuitos virtuales y garantizar calidad de servicio. Sin embargo, esta red *Internet* evolucionada presenta diferencias notables frente a la red *ATM*. En primer lugar la red de redes presenta una baja densidad de mallado, aunque con enlaces de muy alta capacidad. En segundo lugar, los centros de decisión están dispersos y la información de que disponen éstos para realizar el encaminamiento es escasa y muchas veces obsoleta y, además, a diferencia del la red *ATM*, el tráfico cursado no está perfectamente caracterizado ni dividido en clases. Todas estas circunstancias

hacen que los algoritmos de encaminamiento inicialmente diseñados para redes *ATM* presenten un bajo rendimiento en redes *IP*. Estos hechos han sido los que han forzado la estructuración de esta tesis al estudiar tanto el problema del encaminamiento en redes *ATM* (con una alta tasa de conexión, un conocimiento preciso de la red y un tráfico perfectamente caracterizado y dividido en clases) como el problema del encaminamiento en redes con baja tasa de conexión y un conocimiento poco preciso del estado de la red.

1.1 Concepto y tipos de encaminamiento

1.1.1 Encaminamiento y controles de gestión de recursos

Antes de entrar en el desarrollo de esta tesis, es necesario definir en qué consiste el encaminamiento y dónde y cómo actúa. Ash [Ash97] define como encaminamiento "una función indispensable en las redes de telecomunicaciones que conecta una llamada desde un origen a un destino y es el corazón de la arquitectura, diseño y operación de cualquier red". Girard [Gir90] realiza la siguiente introducción al concepto de encaminamiento: "Considere una red que trabaja en tiempo real. Cuando una nueva llamada llega a la red debemos determinar si existe un camino a través del cual la llamada pueda ser conectada a su destino. Si se encuentra ese camino, debemos decidir si se emplea o no", y la misión del encaminamiento es encontrar ese camino. Estas definiciones recogen la importancia del encaminamiento dentro de las redes de comunicaciones como parte esencial para cumplir la labor para las que se diseñaron y construyeron, es decir, transportar información de una fuente o conjunto de fuentes a un destino o conjunto de destinos. Por otro lado, la declaración aportada por Girard ya sugiere la relación evidente que existe entre el encaminamiento y los sistemas de control de flujo (como los controles de admisión).

Como se ha apuntado, entre los requisitos más importantes que han de satisfacer las modernas redes de comunicaciones se encuentran los requerimientos de calidad de servicio solicitados por los usuarios. Desde el punto de vista del encaminamiento, este hecho implica que, amén de establecer el camino entre el origen y el destino, éste tiene que cumplir estos requisitos, por lo que al buscar y establecer el camino se deben de tener en cuenta los criterios de calidad de servicio. El *IETF* (*Internet Engineering Task Force*) define en la petición de comentarios RFC (*Request For Comment*) 2386 [CNRS98] el concepto de encaminamiento basado en calidad de servicio como "un mecanismo de

encaminamiento de acuerdo con el cual los caminos son determinados en base al conocimiento de la disponibilidad de recursos en la red así como de los requerimientos de calidad de servicio solicitados". Basándose en esta definición el mismo RFC establece los objetivos del encaminamiento con calidad de servicio que son:

1. Determinación dinámica de los caminos posibles: El encaminamiento con calidad de servicio debe determinar el camino, entre todo los posibles, que sea la mejor elección para acomodar los requerimientos de *QoS* solicitados.
2. Optimización de los recursos: El encaminamiento dinámico y con calidad de servicio debe ayudar al uso eficiente de los recursos de la red.
3. Moderar la degradación del rendimiento: El encaminamiento dinámico debe ofrecer una degradación del rendimiento de la red menor que la degradación que se obtendría si no se emplean estos mecanismos.

Para definir y comprender el término encaminamiento dinámico es necesario conocer que, históricamente, en la evolución de las técnicas de encaminamiento, han existido dos épocas muy diferenciadas. Durante una primera época el encaminamiento fue fijo y jerárquico, es decir, los caminos entre el origen y el destino siempre eran los mismos con independencia del estado de la red. Es con la aparición de los ordenadores, los programas de control y el intercambio de señalización mediante canal común cuando se inició la segunda época en la que se produjo el tránsito del encaminamiento fijo jerárquico a un encaminamiento dinámico.

El término dinámico hace referencia a los métodos utilizados por el encaminamiento que son sensibles a las variaciones temporales de las condiciones de la red. Por lo tanto, para que el encaminamiento sea dinámico, el algoritmo ha de ser capaz de recoger información del estado de la red. Este hecho exige que el encaminamiento coopere con los diversos sistemas de control de recursos existentes en la red.

En esta línea, Fabregat [Fab98] presenta la relación existente entre los distintos controles que afectan al encaminamiento. Esta relación se ilustra en la figura 1.1, y consta de los siguientes controles:

Planificación de recursos. Conjunto de acciones de control a largo plazo que determina las cantidades físicas de recursos que ha de poseer la red física, entendida como el conjunto de nodos y enlaces que los conectan.

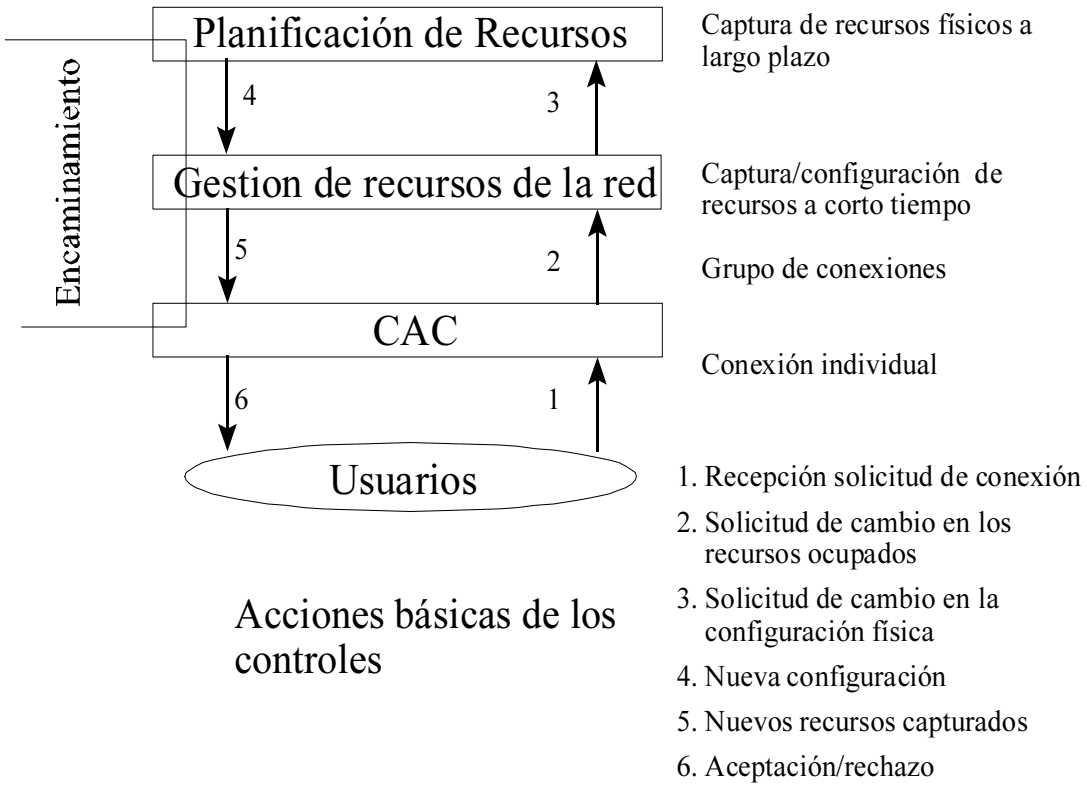


Figura 1.1: Relación entre los elementos de gestión de recursos, recursos y control de admisión de llamadas (*CAC*)

Gestión de recursos. Conjunto de acciones relativas al establecimiento de enlaces virtuales y reserva de los recursos asignados a éstos con el objetivo de maximizar el rendimiento. Entendemos como red virtual la red lógica generada mediante el empleo de enlaces virtuales (*Virtual Path* o *VP*). Esta es la red que emplea tanto el control de admisión como los mecanismos de encaminamiento para llevar a cabo la conexión. En las redes tradicionales (como la RTC) la red virtual y la red física coincidían. Pero, al aparecer el concepto de enlace virtual en *ATM*, esta asociación desaparece, pudiendo ser la red virtual distinta a la red física. En las redes *IP* aparece un fenómeno similar mediante la aplicación del protocolo *Multi Protocol Label Switching* (*MPLS*) [CDF+99] [RVC99] en combinación con la reserva de recursos mediante protocolos como el *Resource reSerVation Protocol* (*RSVP*) [BZB+97].

Control de admisión de llamadas o CAC (*Connection Admission Control*).

Conjunto de acciones ejecutadas en la fase de establecimiento de la conexión. Estas acciones tienen como objetivo decidir si se acepta (o no) la conexión. Esta decisión es tomada en función de las características del tráfico que solicita la conexión, de los requisitos de calidad de servicio solicitados y del estado de la red.

Basándose en esta clasificación de los controles, el encaminamiento forma parte fundamental de los mecanismos de **gestión de recursos**, y se relaciona directamente con los mecanismos de **planificación de recursos** y con el **control de admisión**. De hecho, la planificación afectará estrechamente al rendimiento de los algoritmos de encaminamiento mientras que, para poder encaminar dinámicamente, es necesario que el sistema de encaminamiento reciba información del control de admisión de llamadas.

1.1.2 Clasificación de los algoritmos de encaminamiento

En este apartado recogemos algunos de los criterios más importantes para clasificar los algoritmos de encaminamiento. Es importante conocer estos criterios, así como las ventajas e inconvenientes de cada uno de los tipos establecidos por los distintos criterios de clasificación, ya que la elección de un determinado tipo de algoritmo de encaminamiento condicionará su funcionamiento y rendimiento dentro del escenario que se pretende tratar.

Uno de los criterios más empleados a la hora de esta clasificación es el lugar en que se localiza el mecanismo de decisión. En función de éste se suele distinguir entre los siguientes grupos básicos.

Centralizado. Un sistema de decisión central recibe información del estado global de la red y realiza las funciones de decisión para todas las llamadas. Este mecanismo ha sido el tradicionalmente empleado en las redes telefónicas.

Distribuido. El sistema de decisión está distribuido por toda la red y cada nodo toma sus propias decisiones en función del conocimiento (siempre parcial) que tenga del estado de la red. Con el objeto de conocer este estado los distintos elementos intercambian información entre ellos. En la versión más clásica de este tipo de algoritmos cada nodo encamina la información independientemente del resto; esto puede dar lugar a que toda la información no siga un mismo camino. Este mecanismo es el que implementan en la actualidad las redes *IP*.

Aislado. Es un caso especial de encaminamiento distribuido donde los nodos no conocen nada del estado de la red y encaminan únicamente en función de su estado local. Este tipo de algoritmos se ha empleado en redes de deflexión de paquetes.

Es evidente que cada uno de estos métodos presentan ventajas e inconvenientes. Por ejemplo, el encaminamiento centralizado permite un mejor aprovechamiento de los recursos al conocer todo el estado de la red, lo que en principio lo hace adecuado para los algoritmos de encaminamiento con *QoS*. Pero presenta como desventaja la vulnerabilidad ante fallos, ya que un fallo en el sistema central dejará a toda la red inoperante. Además, también presenta la necesidad de enviar la información de estado de la red al sistema de decisión central. Si la red es muy grande este puede ser un grave problema y dar lugar, bien a una sobrecarga de mensajes con información de estado, o bien a inconsistencias entre el conocimiento registrado sobre el estado de la red y el estado real de misma.

El encaminamiento distribuido posee como ventaja la redundancia ante fallos pero presenta el inconveniente de que las rutas son menos eficientes y pueden dar lugar a bucles o lazos cerrados con relativa facilidad. Al igual que el encaminamiento centralizado, aunque usualmente en menor medida, introduce la necesidad de obtener información del estado de la red.

Por el contrario, el encaminamiento aislado no presenta ningún problema para actualizar la información, ya que no intercambia información con el entorno, pero las rutas generadas son muy poco eficientes y sólo tiene sentido para redes de datos con topologías muy específicas.

Existen soluciones mixtas que intentan recoger lo mejor de los distintos grupos. Una de estas soluciones es el encaminamiento en origen, en la cual cada uno de los nodos funciona, en cierta manera, como el nodo central, aunque sólo para las llamadas generadas en él. Para que cada nodo funcione adecuadamente, ha de conocer el estado de toda la red con el objetivo de construir la ruta completa entre el origen y el destino. El encaminamiento en origen consigue evitar la vulnerabilidad del encaminamiento centralizado, pero conservando sus ventajas con respecto a la calidad del camino encontrado. Por contra incrementa la complejidad de los controles a implementar en los nodos finales dado que les traslada la función de encaminamiento que antes residía en el nodo central. Por otro lado tampoco resuelve la exigencia de distribuir información actualizada del estado de la red.

Otro de los criterios empleados para clasificar los algoritmos de encaminamiento es

la manera en que se encuentra organizada la información del estado de la red que posee el mecanismo de decisión. Basándose en este criterio se suele distinguir entre:

Algoritmos basados en vector de estados. En este tipo de algoritmos los nodos conocen el estado de sus enlaces e intercambian con sus nodos vecinos los costes estimados de establecer una llamada desde él a los diferentes nodos destino. A partir del intercambio de esta información con los nodos vecinos, más el coste de sus enlaces locales, cada nodo es capaz, a su vez, de estimar el coste total de un enlace desde él a todos los destinos. La información que guarda el nodo en su tabla de rutas es el coste estimado a cada destino y el enlace de salida que debe emplear para llegar a ese destino. Un ejemplo de este tipo de algoritmos lo podemos encontrar el en protocolo *RIP* (*Routing Interior Protocol*) [Hed88]. Este tipo de algoritmos tiene como principal aval su sencillez y como mayor inconveniente la necesidad de un largo tiempo de convergencia cuando se produce un cambio en la red. Este largo tiempo puede dar lugar a pérdidas de datos y a graves degradaciones de rendimiento. Normalmente este esquema sigue, por naturaleza, una estrategia distribuida y emplea el algoritmo Bellman-Ford [Bel57] distribuido para el cálculo de las rutas.

Algoritmos basados en estado de los enlaces. De acuerdo con este tipo de estrategias, el nodo conoce el estado de todos los enlaces y a partir de esta información es capaz de construir la ruta completa hasta los diferentes destinos. Su ventaja más destacable es su rápida convergencia ante cambios en el estado de la red, en tanto que basta con conocer el cambio (por ejemplo un enlace que falla) para que todos los nodos puedan reconstruir las tablas de encaminamiento a todos los destinos. Esta familia de algoritmos, en la que se incluye el protocolo *OSPF* (*Open Shortest Path First*) [Moy97] [Moy98], resulta mucho más adecuada para un encaminamiento centralizado o encaminamiento en origen que el basado en un vector de estados. Emplean normalmente el algoritmo de Dijkstra [Dij59] para el cálculo de las rutas. A pesar de la complejidad que conlleva su implementación, este tipo es la opción más adecuada si se busca diseñar protocolos que cumplan requerimientos de calidad de servicio.

Otro criterio de clasificación es el propuesto por Dziong [Dzi97], el cual considera la información de estado que se emplea para la selección del camino. Así, se distinguen cuatro grandes grupos:

Fijo. El camino entre un nodo origen y un nodo destino es siempre el mismo con independencia del estado de la red. Por lo tanto, este tipo de algoritmos y protocolos de encaminamiento se caracteriza porque la elección del camino se realiza sin ninguna información acerca de la situación actual de la red. Usualmente, el rendimiento de esta estrategia es inferior al resto y depende en gran medida de los algoritmos de dimensionamiento de la red.

Dependiente de Macro-estado. Esta estrategia, que no contempla el estado de los enlaces en el proceso de elección de camino, toma en consideración todos los posibles caminos (que permite el mallado) entre el origen y el destino deseados. Cada uno de estos caminos se clasifica, de acuerdo con el *CAC*, como macro-estado admisible si es capaz de transportar la conexión, y macro-estado inadmisible si no lo es. De este modo, la selección de un camino se realiza en función del macro-estado de los caminos. Existen dos conocidos tipos de algoritmos dentro de este grupo. El primero es el encaminamiento secuencial. En este caso los distintos caminos son ordenados secuencialmente siguiendo algún criterio. Tras esto, el primer camino dentro de la secuencia es escogido para llevar la información. Si este camino falla, por tener un estado inadmisible (por rechazo del *CAC* de algún nodo incluido en el camino), se selecciona el siguiente camino dentro de la secuencia, si este vuelve a fallar la operación sigue de acuerdo con la secuencia hasta encontrar un camino válido o hasta rechazar la llamada. Un ejemplo de este tipo de este tipo de algoritmos se puede encontrar en [YSSM95] y [IMYS91]. En el segundo tipo, los caminos no están ordenados, la decisión de escoger un determinado camino frente a otro se realiza de forma aleatoria. Un ejemplo de este tipo de algoritmos es el *DAR* (*Dynamic Alternative Routing*) [GKK88]. En este algoritmo primeramente se ofrece la conexión, en caso de existir, al camino directo (enlace directo entre el origen y el destino). Si éste falla se escoge otro camino de forma aleatoria entre todos los posibles.

Adaptativo. De acuerdo con esta estrategia, cada nodo toma y emplea estadísticas de tráfico obtenidas de sus enlaces (por ejemplo, llamadas perdidas, llamadas ofrecidas,...) relacionadas tanto con el flujo de tráfico, como con la distribución de las métricas de *QoS*, y en función de estas estadísticas modifica su comportamiento. Un ejemplo de este tipo de algoritmos lo podemos encontrar en [NWM77] donde se propone un autómata finito que establece las probabilidades de escoger un determinado camino en función de la percepción local del estado de la red.

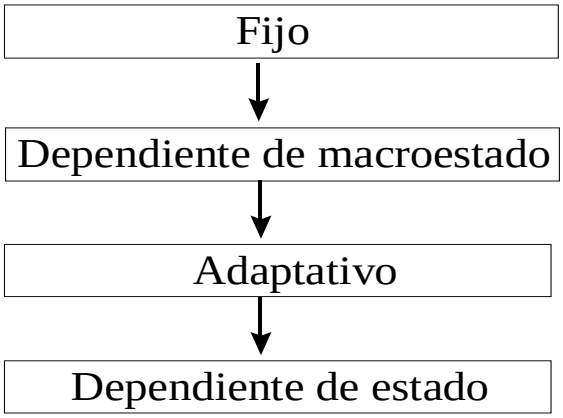


Figura 1.2: Criterio de clasificación de Dziong [Dzi97] y grado de inteligencia de los algoritmos de menor a mayor

Dependiente de estado. En este caso, el algoritmo toma la decisión de encaminamiento en función del conocimiento del estado de los enlaces de toda o parte de la red. En la literatura existen abundantes ejemplos de este tipo de algoritmos como el *RTNR* [ACFH91] o el *Widest Shortest* [Ma98].

En la figura 1.2 se resume de forma esquemática este criterio de clasificación. En ella las flechas indican el grado de inteligencia de los algoritmos de encaminamiento ordenados de menor a mayor. En la actualidad también existen algoritmos híbridos que son mezcla de algunas de estas categorías.

Basándose en todos estos criterios de clasificación se puede afirmar que las características apropiadas para los algoritmos de encaminamiento con calidad de servicio deben incluirse dentro del grupo en encaminamiento en origen o, en su defecto, encaminamiento centralizado y basarse en el estado de los enlaces. Estas dos características son obvias, en la medida en que, para poder definir un camino que cumpla los requisitos de calidad de servicio es necesario conocer el estado de toda la red y establecer el camino de una vez, en lugar de crearlo nodo a nodo, circunstancia que puede dar lugar a que la información llegue al destino por caminos distintos (y, por lo tanto, con parámetros de calidad no homogéneos). Por la misma razón, los algoritmos de encaminamiento que satisfagan cierta calidad de servicio han de depender del estado de la red. Por lo tanto, todos los algoritmos estudiados en esta tesis se encuadran dentro del encaminamiento en

origen, conocen el estado de los enlaces y son dependientes del estado.

1.2 Reseña histórica de los algoritmos de encaminamiento

En esta sección se estudiarán brevemente algunas de las diversas propuestas presentadas con el fin de solucionar el problema del encaminamiento en redes de comunicaciones.

Dentro de la evolución de las redes de comunicaciones con encaminamiento dinámico, podemos distinguir entre dos períodos muy diferentes. El primero de estos periodos se caracteriza por la existencia de redes dedicadas, es decir, un tipo de red para cada tipo de servicio. La separación de estos dos periodos se produce con la aparición de las redes de servicios integrados donde se emplea un mismo recurso de trasmisión para transportar todo tipo de información. Hasta ese mismo momento las soluciones existentes se podían clasificar en dos grupos: redes telefónicas y redes de datos. Cada uno de estos grupos se define por unas características propias muy marcadas.

Las redes telefónicas se caracterizan por:

- Un único tipo de tráfico provocado por un único tipo de servicio perfectamente conocido (la voz) que requiere *jitter* y retardo extremo a extremo limitado. Estas redes son orientadas a conexión y reservan un recurso fijo de ancho de banda por llamada.
- La topología de la red se diseña para transportar una carga prevista de antemano provocada por un único servicio (la voz) cuya carga y naturaleza no cambian muy a menudo.

Las redes de datos, por el contrario, utilizan soluciones distintas basadas en condiciones completamente diferentes:

- Sólo es tomado en consideración el tráfico *best-effort*, es decir, aquel que no exige ningún requerimiento concreto de retraso y *jitter* y solicita de la red simplemente una transmisión sin errores de un bloque de información. En el caso de ocurrir

situaciones de sobrecarga, todo el tráfico, que es transportado en paquetes, es retrasado.

- La conectividad de la red es dispersa y cambia a menudo.

La red de servicios integrados surge de la necesidad de soportar sobre un recurso único estos dos tipos de servicio.

A continuación se pasará a estudiar brevemente cada uno de los tres tipos de redes: redes telefónicas, de datos y, por último, las redes integradas.

1.2.1 Redes Telefónicas

Históricamente, los primeros intentos de resolver el problema del encaminamiento estuvieron basados en el empleo de operadores humanos que se encargaban de resolver el camino entre los distintos usuarios, creando de forma manual los circuitos que los conectaban a través las primeras redes telefónicas. Al aumentar la complejidad de las redes el empleo de personal humano deja de ser una opción válida y surge la necesidad de sustituir este sistema por un sistema automatizado. Es en este momento cuando aparecen las primeras centrales automáticas y el encaminamiento fijo jerárquico [Ash97]. Este encaminamiento está diseñado para que los algoritmos y protocolos de encaminamiento sean lo más simples posible. En la figura 1.3 se ilustra un ejemplo típico de esta topología. El algoritmo de encaminamiento escoge siempre, como enlace de salida del nodo, el que le lleva al nodo de nivel superior que más cerca esté del destino. Así, si la central a la que está conectado el nodo origen no tiene conexión directa con el nodo destino ésta pasa la solicitud de conexión a la central de nivel inmediatamente superior. Este esquema se caracteriza por el hecho de que el camino entre cualquier nodo origen y cualquier nodo destino es fijo. Consecuentemente, el rendimiento del encaminamiento jerárquico depende básicamente de los algoritmos de dimensionamiento de la red antes que del propio algoritmo de encaminamiento en sí. No es hasta la aparición de los ordenadores y su posterior difusión a principios de los años 70, cuando aparece la posibilidad de almacenar programas de control en los conmutadores. A partir de este momento, gracias a la posibilidad de emplear dichos programas, se produce una rápida evolución en los algoritmos y protocolos de encaminamiento, surgiendo los primeros sistemas de encaminamiento dinámico con aplicación práctica.

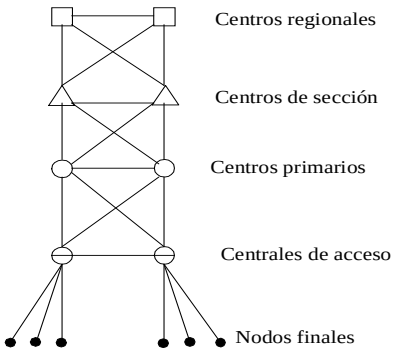


Figura 1.3: Red jerárquica de 5 niveles

Estos primeros sistemas se basaron en rutas precalculadas mediante complejos algoritmos de optimización lineal y posteriormente almacenadas en tablas en los nodos. En función de la hora del día el sistema poseía un conjunto de tablas que indicaban cuáles debían ser las rutas seleccionadas con el objetivo genérico de reducir la probabilidad de bloqueo total de llamadas. Los algoritmos de optimización tomaban como datos la topología de la red y la evolución de la distribución del tráfico ofrecido a la red a lo largo del día, así como los orígenes y destinos de este tráfico, para calcular estas tablas. Este tipo de algoritmos se empleó en los primeras implementaciones de encaminamiento dinámico para redes telefónicas, ya que el precio y la escasa potencia de los ordenadores en aquellos momentos impedía realizar los cálculos en tiempo real. Puesto que en estos algoritmos los cálculos se realizan *off-line*, no importaba el tiempo empleado para el cálculo. Una vez que se obtenía la solución, esta se almacenaba en una tabla que podía ser consultada rápidamente. Un ejemplo de este tipo de algoritmos es el *DNHR* (*Dynamic No Hierarchical Routing*) [Ash85]; para una información más amplia se puede consultar [Ash97] donde se presenta un detallado análisis de este tipo de algoritmos.

Más adelante, a finales de los 70, al aumentar la capacidad computacional de los ordenadores, aparecen los primeros algoritmos capaces de reaccionar de forma totalmente dinámica y de calcular en tiempo real los caminos entre los nodos origen y destino. Aquí es cuando se puede empezar a hablar con propiedad de algoritmos y protocolos de encaminamiento dinámico. En este ámbito, Ash [Ash97] discute varias formas de implementar un sistema de encaminamiento en tiempo real. Entre las formas que distingue podemos mencionar:

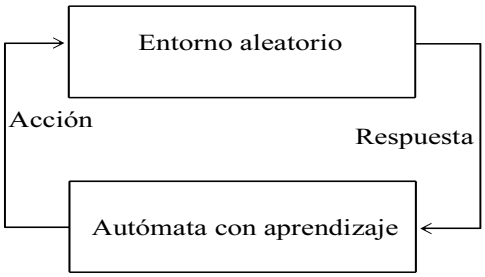


Figura 1.4: Interacción del autómata con el entorno

Encaminamiento automático *out-of-chain*. Este tipo de encaminamiento propone una expansión al encaminamiento jerárquico fijo. En este caso, cuando la ruta normal se encuentra bloqueada se ofrece la llamada a otro camino para intentar completarla.

Encaminamiento con aprendizaje. Este tipo de encaminamiento sigue una estrategia distribuida y emplea el uso de autómatas con aprendizaje. Estos autómatas están continuamente modificando sus decisiones a través de la realimentación que reciben del entorno [ASGV98][NWM77]. Así, si reciben una realimentación positiva (se ha podido establecer la llamada) aumentan la probabilidad de escoger dicha ruta para la siguiente llamada con el mismo origen-destino. Si la realimentación es negativa disminuyen la probabilidad de escoger dicha ruta. El esquema general de esta realimentación del autómata se puede ver en la figura 1.4. Otras técnicas, que también están basadas en el aprendizaje son el *DAR* [GKK88] empleado por *British Telecom* y el algoritmo *State and Time Dependent Routing* [IMYS91] [YSSM95] [KI95] empleado por NTT.

Encaminamiento centralizado en tiempo real. Este tipo de encaminamiento se basa en la presencia de un nodo central que recoge la información del estado de la red y cada cierto tiempo recalcula la rutas. Un ejemplo de este tipo de algoritmos es el *DCR* (*Dynamic Controlled Routing*) [RBCC95].

Encaminamiento distribuido en tiempo real. En este tipo los distintos nodos son capaces de tomar decisiones en función del estado de la red. Un ejemplo de este tipo de encaminamiento es *RTNR* (*Real Time Network Routing*) [ACFH91].

Combinación de encaminamiento preplanificado y en tiempo real. Este tipo de algoritmos utiliza una etapa preplanificada para reducir el espacio de estados y, por lo tanto, la información a manejar en el momento de tomar la decisión de encaminamiento. El funcionamiento de la etapa preplanificada es similar al funcionamiento del algoritmo *DNHR*, donde empleando un algoritmo de optimización lineal y teniendo en cuenta la distribución del tráfico en función de las horas del día se construye una tabla de posibles rutas para cada par de nodos origen-destino. Con esto se consigue reducir el número de posibles caminos que el algoritmo de tiempo real tiene que comprobar a fin de buscar el mejor camino posible. Este algoritmo presenta como inconveniente la posibilidad de que el mejor camino posible no se encuentre dentro de la tabla, con lo que el algoritmo de tiempo real escogerá un camino pseudo-óptimo.

1.2.2 Redes de datos

Al hablar de la historia de las redes de datos hay que tener en cuenta que tanto su aparición como su evolución han ido en paralelo a la aparición y evolución de los ordenadores. Por esta razón, la etapa inicial de encaminamiento exclusivamente fijo que se produjo en las redes telefónicas, antes de aparecer los ordenadores y, con ellos, los algoritmos de encaminamiento dinámico, no ha existido. En esta sección se presentan las soluciones más populares utilizadas en las redes de datos empezando por el modelo *OSI* y siguiendo por el modelo Internet (*IP*). No vamos a entrar a explicar en profundidad ninguno de estos modelos, sobre los que existe disponible una abundante literatura (véase, por ejemplo, [Tan97] o [Sta94]).

1.2.2.1 Modelo OSI

Al hablar del modelo *OSI* es necesario hacer hincapié en que no se ha de confundir este modelo con un sistema real. El sistema *OSI* es un modelo de referencia que muestra cómo debería estructurarse un sistema de comunicaciones. La principal novedad que el modelo *OSI* presenta, frente a los sistemas precedentes, es la división del sistema en una serie de capas superpuestas (nivel físico, de enlace, red, transporte, sesión, presentación y aplicación), cada una con funcionalidades exclusivas y muy diferenciadas. En la figura 1.5 se puede observar el modelo de capas *OSI*, así como la manera en la que se establece

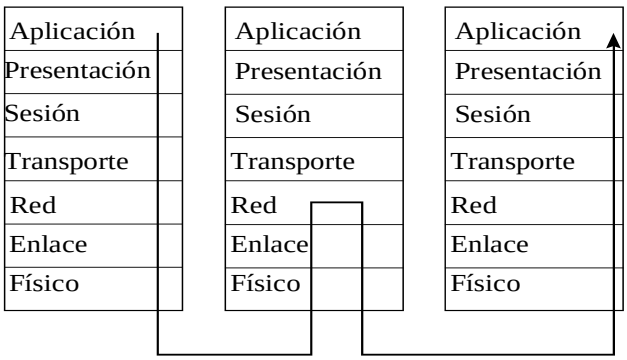


Figura 1.5: Modelo de capas de *OSI*: Establecimiento de la conexión

la comunicación entre dos aplicaciones finales y el comportamiento de los sistemas intermedios o de tránsito. Como se deduce de la figura, los nodos de tránsito no utilizan todas las capas, de hecho, si los nodos intermedios únicamente van a realizar tareas de red, éstos no precisan de las capas superiores a la de red.

La capa de red es la capa encargada de las funciones de encaminamiento. En esta capa el modelo *OSI* establece tres funciones:

Identificación. Es una función que esta capa presta a la capa de aplicación y consiste en asignar un nombre que identifique de forma unívoca a un equipo terminal. El modelo *OSI* proporciona el servicio de directorio X.500 para este objetivo.

Direccionamiento. El modelo *OSI* no especifica cómo deben ser las direcciones, sino que establece una serie de recomendaciones para que su formato sea independiente de los dominios administrativos. Entre estas recomendaciones se establece que la dirección debería contener un campo denominado *Initial Domain Part*, que actúa como identificador entre los diferentes tipos de direcciones.

Encaminamiento. Su misión es la recopilación de información y, empleando dicha información, la determinación de los caminos entre los nodos orígenes y los nodos destinos de cada llamada. Dentro de este apartado el modelos *OSI* define tres protocolos básicos para el encaminamiento (*ES-IS*, *IS-IS* y *IDRP*) que pasamos a describir a continuación.

El protocolo *ES-IS* (*End System to Intermediate System*) es utilizado tanto por los

sistemas finales como por los nodos intermedios para averiguar cómo está conectada la red (topología) y poder determinar a cuál de los nodos intermedios se han de enviar los paquetes dirigidos a un determinado sistema final. Entendemos como sistema final el conjunto de los nodos donde se ejecutan las aplicaciones, los cuales, por tanto, son generadores y receptores de tráfico. Por el contrario, los sistemas intermedios son aquellos a los que están conectados los sistemas finales y cuya misión es transportar la información de un sistema final a otro. Estos nodos no generan ningún tipo de tráfico ni son destinatarios finales con excepción del tráfico de señalización. Este protocolo permite a los sistemas finales conocer a qué sistemas intermedios están conectados y a los sistemas intermedios qué sistemas finales tiene conectados. Así mismo, posibilita el redireccionamiento de los paquetes enviados por los sistemas finales. Este redireccionamiento ocurre cuando un sistema final que desea enviar información a otro no conectado a su red local, escoge uno de los sistemas intermedios a los que está conectado y éste no es adecuado para alcanzar el destino. En estas circunstancias el sistema intermedio devuelve el paquete al sistema final origen indicando que lo envíe a través de otro sistema intermedio.

El protocolo *IS-IS* (*Intermediate System to Intermediate System*) es utilizado dentro de un dominio y permite tanto el intercambio de información entre sistemas intermedios como la computación de rutas, posibilitando una rápida adaptación ante cambios topológicos. Se puede definir **dominio** como una porción del espacio de direcciones que depende de una misma entidad administrativa y, por ende, presenta un comportamiento homogéneo desde el punto de vista del encaminamiento. Este protocolo está basado en el conocimiento del estado de los enlaces por parte de los nodos intermedios y aprovecha este conocimiento para computar el camino más corto. Las rutas calculadas se almacenan en tablas de forma que, cuando llega un paquete, se busca su dirección destino en la tabla y se determina por cuál de los enlaces de salida del sistema intermedio se debe enviar. Esta filosofía de trabajo es la utilizada por el algoritmo *OSPF* [Moy97].

Por último, el protocolo *IDRP* (*IS-IS inter-Domain Routing Protocol*) es usado en las fronteras o límites de los sistemas intermedios y permite conectar unos dominios con otros. Este protocolo presenta unos tiempos de respuesta ante cambios de topología mayores que el protocolo *IS-IS*.

1.2.2.2 La red Internet

La red Internet ha evolucionado de forma radical desde su aparición a principios de los 70. De hecho, las últimas propuestas llevan camino de convertir las redes basadas en tecnología *IP* en unas redes de servicios integrados donde no sólo se enviarán datos y la información circulará en forma de datagramas, sino que, además, permitirán establecer circuitos virtuales y enviar información con requisitos de *QoS* como pueda ser la telefonía o el vídeo.

El modelo de capas propuesto por Internet es más sencillo que el modelo *OSI* ya que los tres niveles superiores se han fundido en un único nivel de aplicación (véase la figura 1.6). La capa de red sigue siendo, al igual que el modelo *OSI*, la encargada de proporcionar los servicios de identificación, direccionamiento y encaminamiento.

El servicio de identificación es muy similar al proporcionado por *OSI* y puede ser considerado una versión reducida del servicio de directorios X.500. El protocolo empleado por las redes *IP* para prestar este servicio es el protocolo *DNS* (*Domain Name System*).

Mayores diferencias existen en el servicio de direccionamiento, sobre todo en la forma de asignar direcciones. En las redes *IP* las direcciones constan de una secuencia de 32 bits para la versión 4 y 128 bits para la versión 6, careciendo del *Initial Domain Part*, por lo que en principio existe un único formato de dirección. Aunque en este momento están conviviendo los dos métodos de asignación, en un futuro debería ser, por problemas de agotamiento en la numeración, la versión 6 la única que permanezca con la posible excepción de la existencia de redes privadas sin conexión exterior. La forma de asignar direcciones en este tipo de redes es mucho más rígida que en el caso *OSI*, aunque posteriores RFC [FLYV93] proporcionan mecanismos para flexibilizar la asignación de direcciones. [1]

Internet emplea para el encaminamiento dos clases de protocolos: La primera clase, para la cual se utilizan la abreviación *IGP* (*Interior Gateway Protocol*) se encarga del encaminamiento dentro del mismo sistema autónomo (sistema que depende de una misma autoridad administrativa), y es equivalente al protocolo *IS-IS* del modelo *OSI*, mientras que la segunda clase, que se encarga de comunicar distintos sistemas autónomos, se suele denominar mediante las siglas *EGP* (*Exterior Gateway Protocol*) o *BGP*(*Border*

[1] En referencias como [Sal00] están recogidos los distintos RFC que indican cómo implementar el protocolo IPv6 y dónde es posible obtener más información de cómo se asignan las direcciones en IPv6.

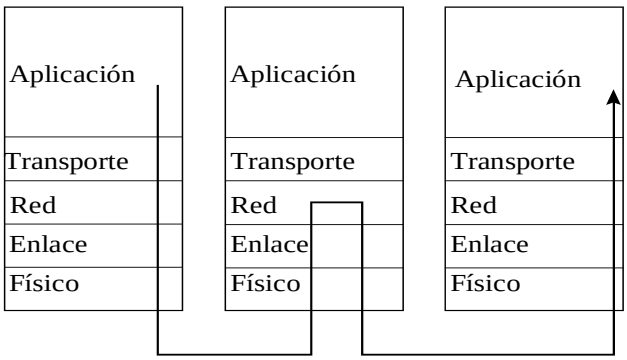


Figura 1.6: Modelo de capas de Internet

Gateway Protocol), siendo equivalente al protocolo *IDRP* del modelo *OSI*. El protocolo *ES-IS* empleado en el modelo *OSI* no tiene equivalencia dentro de las redes Internet, siendo el protocolo de resolución de direcciones *ARP* (*Address Resolution Protocol*) el protocolo con un funcionamiento más cercano al protocolo *ES-IS*.

El protocolo recomendado por el *IETF* (*Internet Engineering Task Force*) para el encaminamiento *IGP* en las redes basadas en la tecnología *IP* es el protocolo *OSPF* [Moy97] [Moy98]. Este protocolo sustituye al antiguo protocolo *RIP* [Hed88], si bien este último todavía está en uso en numerosas redes. El protocolo *OSPF* emplea una tabla o base de datos donde están recogidos los estados de todos los enlaces del dominio. Esta base de datos se encuentra replicada en todos los nodos, los cuales utilizan (en paralelo) un algoritmo de búsqueda de camino para determinar, en función de su base de datos local, el camino más corto a los distintos nodos de la red. El principal problema de este algoritmo es conseguir que las distintas bases de datos se mantengan sincronizadas a fin de que los distintos nodos tengan exactamente la misma tabla de rutas y, de esta manera, no se den problemas como los lazos cerrados. El problema del lazo cerrado se puede ver en la figura 1.7, donde el nodo A pretende enviar un mensaje al nodo G. El nodo A comprueba en su tabla que el camino más corto pasa por el nodo B y le envía el mensaje. Éste, a su vez, envía el mensaje al nodo D y éste al mirar en su tabla comprueba que debe de enviar el mensaje al nodo E, de donde pasa al nodo C (por una mala actualización del estado del enlace e_{DG}). Por último, el nodo C al mirar en su tabla, determina que el camino más corto al nodo G pasa por el nodo B con lo que se produce un lazo cerrado y el mensaje estaría continuamente circulando por la red sin ser

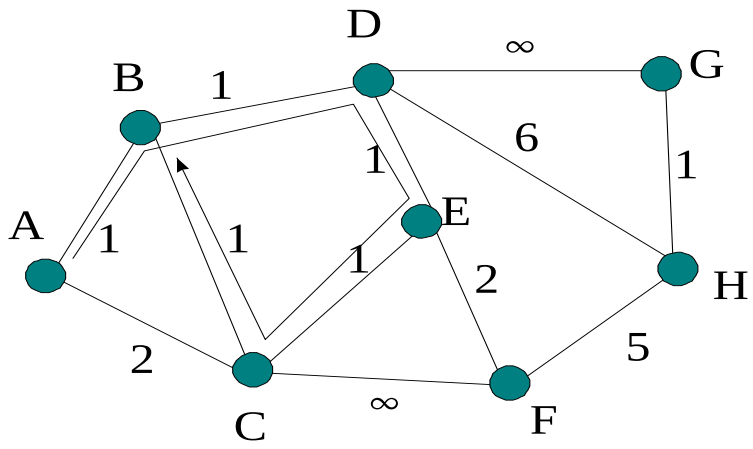


Figura 1.7: Esquema de una red con un bucle cerrado en el encaminamiento del nodo *A* al nodo *G*. El dato de cada enlace informa del coste de su uso.

entregado.

Una característica novedosa que el protocolo *OSPF* presenta frente a protocolos previos (como el *RIP*, que envía la señalización sobre los mismos paquetes de datos *IP*), es que los mensajes de señalización utilizan paquetes con un código distinto al código empleado por los paquetes de datos *IP* a fin de priorizar su procesamiento, lo que equivale, salvando las diferencias, a una señalización por canal común, en la que toda la señalización se envía por un canal distinto al utilizado para enviar la información.

El protocolo *OSPF* también es capaz de trabajar con varios niveles jerárquicos y, de esta manera, dividir la red en subredes lógicas con el objetivo de ocultar información y disminuir la cantidad de datos a tratar e intercambiar. Sin embargo, su forma de funcionamiento está fuertemente condicionada por la forma de direccionar de las redes *IP*, por lo que utiliza el esquema de subredes proporcionado por el protocolo *IP*.

Para el encaminamiento entre sistemas autónomos (dominios), las redes *IP* emplean el protocolo *BGP-4* [RL95]. Este protocolo está basado en vector de estados y su funcionamiento es similar al protocolo *IDPR*. Cuando se ideó este protocolo no se buscó que tuviera una rápida respuesta ante cambios en la red, sino al contrario del *OSPF*, el protocolo *BGP-4* está pensado para que sea fácil añadir políticas de encaminamiento, las cuales dependen del administrador de cada sistema autónomo. Las políticas permiten especificar directivas tales como evitar que un determinado mensaje con destino a un nodo que está en otro sistema autónomo pase por una determinada red (por problemas de seguridad en esa red, por ejemplo) aunque dicha red sea el camino más corto.

Existen otros tipos de redes de datos como *SNA* (*System Network Architecture*) de IBM o la red *IPX* de Novell, aunque hoy en día estas redes están en franco retroceso y tienden a desaparecer ante el empuje de las redes *IP*. La red *SNA* originalmente no incluía encaminamiento dinámico (de hecho requería la configuración manual), aunque, posteriormente, se incluyo el protocolo *Advanced Peer to Peer Protocol* que permitió la posibilidad de encaminamiento dinámico. Este protocolo utiliza una base de datos del estado de los enlaces distribuida entre todos los nodos, al igual que el protocolo *OSPF*, pero, a diferencia de este último, no admite múltiples niveles de topología. El protocolo de encaminamiento empleado por las redes *IPX* se denomina, al igual que en la redes *IP*, *Routing Information Protocol* (*RIP*), y presenta muchas similitudes con su homólogo del mundo *IP*.

1.2.3 Redes integradas

Dentro de las actuales redes integradas se pueden distinguir tres tipos. La primera de ellas es la *RDSI* (Red Digital de Servicios Integrados) o, en su denominación inglesa, *ISDN* (*Integrated Service Digital Network*) que puede ser considerada como una adaptación de las redes telefónicas para trasmitir otros servicios más allá de la voz. Ésta surgió al adoptarse la tecnología digital dentro de las redes telefónicas, la cual abrió la posibilidad de transmitir datos además de transmitir la voz digitalmente. Sin embargo, aunque es ahora cuando las operadoras de telefonía la están publicitando masivamente en el mercado de la pequeña empresa e incluso del hogar, esta red está obsoleta, ya que su escaso ancho de banda (sobre todo en las conexiones básicas) no la faculta para la transmisión de muchos de los futuros servicios de comunicaciones. La segunda de las redes integradas es la que se denominó *RDSI* (o *ISDN*) de Banda Ancha, también conocida por red *ATM*, al ser esta la tecnología sobre la que se apoya. La tercera de las redes integradas se puede ver, en cierta manera, como una red híbrida, ya que surge de la adaptación a las redes *IP* de los mecanismos de control de la calidad de servicio inicialmente desarrollados para las redes *ATM*. De hecho esta red se parece, en algunos aspectos, a las redes *ATM* aunque con paquetes de longitud variable. A continuación estudiaremos la red *ATM* con un poco más de detalle.

1.2.3.1 Las redes ATM

La aparición de las redes *ATM* como redes digitales multiservicio con capacidad de satisfacer requisitos de *QoS* significó un brusco cambio con las tecnologías precedentes. Una de las aportaciones más importantes que añadió la tecnología *ATM* fue la definición de redes virtuales que aparecen con el concepto de enlace virtual (*virtual path*).

Las redes *ATM* surgen con la idea de sustituir las redes dedicadas existentes (telefónicas y de datos), aunando la capacidad de transmitir tanto datos como voz o vídeo. Sin embargo, en las etapas iniciales de diseño se consideró que la mayor parte de la información que iba a transportar este tipo de redes lo constituiría el tráfico de audio. Esta consideración implica que los requisitos de *QoS* que se han de satisfacer son, fundamentalmente, retardo y *jitter*, considerando como secundarios otros requisitos de calidad de servicio como la probabilidad de pérdida de paquete y estableciéndose que el propio diseño de la red permitiría unas bajas probabilidades de pérdida. A fin de facilitar el diseño con vistas a conseguir estos objetivos, la red *ATM* se basó en conmutación de paquetes, en la creación de canales virtuales (*VC, Virtual Channel*) y en el empleo de paquetes de pequeño tamaño y de longitud fija a fin de facilitar el *hardware* de los conmutadores y acelerar en éstos el proceso de los paquetes. El concepto de canal virtual se puede definir como un "camino" que está formado por una secuencia de enlaces y nodos de tal forma que todos los paquetes que pertenecen a una misma conexión siguen el mismo camino y son tratados de igual forma por los distintos elementos que forman parte de éste (desde la perspectiva del encaminamiento, el *VC* ha tenido poca importancia, ya que desde el punto de vista operacional del algoritmo de encaminamiento no existe diferencia entre un circuito físico y uno virtual). Así pues, la información es dividida en paquetes de igual longitud (denominados celdas o células) y todas la celdas que corresponden a una misma comunicación siguen el mismo camino.

Como se ha comentado, las redes *ATM* son redes multiservicio, donde existe la posibilidad de transportar muy distintos tipos de tráfico sobre una misma red. En las redes *ATM* existen dos maneras de conseguir esto:

- La primera es crear redes lógicas, mediante enlaces virtuales, de forma que exista una red lógica independiente para cada tipo de tráfico. De esta manera la red funciona en realidad como un cierto número de redes lógicas (una para cada tipo de servicio ofrecido por la red) entre las que se reparten los recursos (ancho de banda) de la red global. El inconveniente que presenta esta estrategia es que se

desaprovechan recursos, puesto que una de las redes puede estar colapsada mientras que otra de las redes lógicas se encuentra infrautilizada.

- La segunda estrategia consiste en que todas las distintas clases compitan por recursos dentro de la misma red lógica. Con esto se consigue una mejor utilización de los recursos disponibles. Esta última estrategia sí ha influido en los algoritmos de encaminamiento puesto que exige trabajar de forma simultánea con diferentes tipos de tráfico.

Los primeros algoritmos presentados para resolver el problema del encaminamiento en redes multiservicio son variaciones de conocidos algoritmos empleados en tráfico telefónico [Hwa93] [Ash97] [RH00], adaptados para trabajar con múltiples clases de llamada. Plotkin [Plo95] introduce algoritmos de tipo exponencial especialmente diseñados para redes multiservicio. Posteriormente, Gawlick [Gaw95] propone una adaptación práctica de este tipo de algoritmos exponenciales a fin de permitir una fácil implementación en redes reales.

Sin embargo, todos estos algoritmos presenta como principal inconveniente la necesidad de conocer perfectamente y en cada momento tanto el tipo de tráfico (a fin de poder ajustar los parámetros usados por la función de coste) como el estado de la red.

Ya se ha apuntado que una de las principales novedades que aparece en las redes *ATM* ha sido el concepto de enlace virtual y con él, la aparición de las redes virtuales. La utilización de enlaces virtuales impone un nuevo y superior nivel de encaminamiento. Así pues, las redes de comunicaciones que emplean estos enlaces virtuales o *Virtual Path* (*VP*) tienen dos niveles de encaminamiento. El nivel superior, que trabaja a una escala de tiempo mucho mayor que la duración de una llamada, se encarga de la definición de la red lógica, donde se deciden los enlaces lógicos entre los nodos y la capacidad asignada a estos, mientras que el segundo nivel de encaminamiento trabaja a corto plazo, y se encarga de encontrar la ruta para una determinada llamada dentro de la red lógica.

A continuación se desarrolla con algo más de profundidad este concepto de red virtual que aparece con las redes *ATM*, el cual se está exportando a otros tipos de redes como las redes *IP*, mediante la tecnología *MPLS*.

1.2.3.1.1 El concepto de Enlace Virtual. Como se ha apuntado, la red *ATM* distingue entre enlaces (*VP*) y canales (*VC*) virtuales. El *VC* consiste en una sucesión

de enlaces físicos y/o lógicos que forman el camino que conecta dos nodos. A fin de establecer el camino se ha realizado previamente una reserva de recursos en los enlaces para poder llevar la información correspondiente a una conexión, es decir, el canal virtual lleva, entre los nodos origen y destino, la información correspondiente a una conexión. El enlace virtual también permite conectar dos nodos mediante una sucesión de enlaces físicos. Sin embargo, aquí termina la similitud con el canal virtual, ya que la información que se transporte sobre el *VP* no corresponde a una única conexión sino a un conjunto de ellas y, además, los nodos que conecta el enlace virtual no tienen por que ser los nodos de origen y destino de la información, pudiendo ser nodos de tránsito.

La idea que subyace bajo el concepto de enlace virtual es conseguir que los nodos intermedios, por donde pasa el enlace virtual, traten de igual manera a todos los paquetes que pertenezcan al mismo *VP*, con independencia del canal al que pertenezcan. Teniendo en cuenta que el enlace virtual se identifica por sus nodos terminales (y no por toda la secuencia de nodos que los componen), se consigue disminuir el tamaño de las tablas de conmutación en los nodos intermedios y, por consiguiente simplificar el *hardware*, obteniendo nodos más rápidos y baratos. En [Fab98] se definen las siguientes características del *VP*:

- Un enlace lógico consta de dos o más enlaces físicos consecutivos.
- Cada *VP* puede acomodar varios canales virtuales.
- Cada *VP* determina un camino predefinido dentro de la red física, de forma que todos los paquetes entregados al nodo origen del *VP* serán enviados al nodo final del *VP*.
- Cada *VP* tiene asignado un ancho de banda, es decir, una capacidad, que determina el número de enlaces virtuales que pueden circular por él.

Las ventajas proporcionadas por el empleo de los enlaces virtuales se pueden resumir en los siguiente puntos:

- Se reduce la señalización de control de los canales virtuales.
- En el proceso de establecimiento de llamada se eliminan los nodos de tránsito que existen dentro del *VP*, ya que estos no tienen constancia de los canales virtuales que están establecidos dentro de los *VP* de los que él forma parte como nodo

intermedio. Esto reduce las funciones de proceso del nodo y simplifica los equipos, disminuyendo el tiempo necesario para establecer el canal, así como el tiempo de proceso de los paquetes en los nodos.

- Permite cambiar de forma dinámica el ancho de banda asignado a los enlaces virtuales.

- Proporciona separación lógica entre los servicios, si se asignan enlaces lógicos distintos para los distintos servicios.

- Aumenta la robustez ante caídas de los canales virtuales, facilitando una rápida recuperación en caso de fallo.

Sin embargo, los enlaces virtuales presentan como principal inconveniente que reducen la ganancia estadística que se obtiene al multiplexar los distintos canales. En resumen, la aparición del enlace virtual ha cambiado el concepto de encaminamiento, obligando a efectuar encaminamiento a dos niveles y a relacionar de forma estrecha dos aspectos de la gestión de redes (el encaminamiento propiamente dicho y el dimensionamiento y planificación de los enlaces) que hasta ese momento habían estado separados.

1.2.3.1.2 El protocolo PNNI. El protocolo de encaminamiento básico empleado por las redes *ATM* es el protocolo *PNNI* (*Protocol Network to Network Interface*) [For96]. Este protocolo determina tanto el formato de la dirección de los distintos nodos como los mecanismos para realizar la tarea de encaminamiento. Uno de los grandes avances de este protocolo, frente a los anteriores, es que contempla la posibilidad de encaminamiento con calidad de servicio y con múltiples niveles jerárquicos. De hecho, establece unas normas para la agregación de enlaces y nodos con el objeto de construir modelos jerarquizados. Estas redes jerarquizadas constan de diferentes niveles de observación. Para explicar esto en detalle prestemos atención a la figura 1.8 donde se puede ver una red formada por dos niveles de jerarquía. Los nodos de la capa superior son nodos lógicos que están formados por subredes y la información de estado de estos nodos lógicos se crea a partir de la agregación de la información del estado, tanto de los enlaces como de los nodos, de la capa inferior. A su vez, el nodo "líder" de cada subred posee una información resumida sobre la organización del nivel superior, así como información de estado de este nivel. Esta información permite a los nodos de un grupo comunicarse con otros nodos de la red que se encuentran en su mismo grupo y, mediante esta información resumida, construir

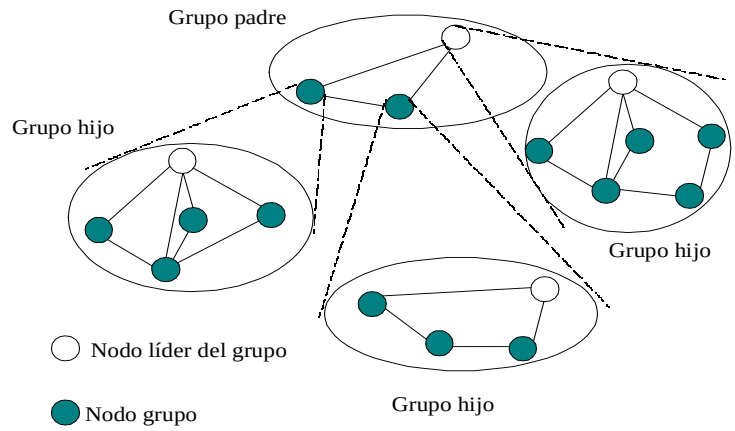


Figura 1.8: Esquema jerárquico del protocolo *PNNI*

un camino a través de los niveles superiores para conectarse a nodos pertenecientes a otras subredes.

De forma resumida, en la siguiente lista se exponen las características de este protocolo:

- Permite tanto la conexión punto a punto (*unicast*), punto multipunto (*multicast*) como *anycast* (cualquiera dentro de un conjunto de posibles destinatarios).
- Permite tratar conjuntos de elementos (nodos y enlaces) como un único enlace lógico.
- Define múltiples niveles de jerarquía.
- Posibilita el reencaminamiento de las conexiones establecidas en caso de caídas en los elementos de la red.
- Utiliza el descubrimiento automático de la topología.
- Las conexiones siguen la misma ruta que los mensajes de establecimiento (señalización asociada).
- Admite el empleo de multiples métricas (coste, capacidad, retardo, *jitter*,...) para decidir el camino. Este protocolo permite, además, que los distintos elementos dispongan de atributos que permiten restringir su uso en determinadas circunstancias (por ejemplo, se puede imponer que un determinado tipo de llamada no utilice nunca un determinado nodo o enlace).

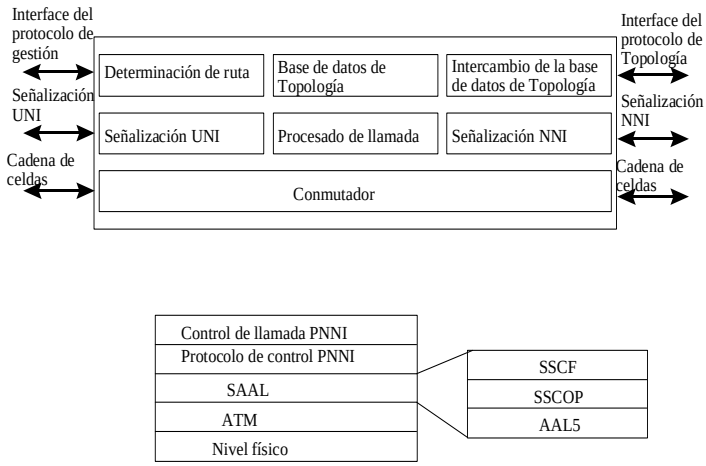


Figura 1.9: Modelo de referencia del protocolo PNNI y su capa de control

- Posibilita el encaminamiento en origen basado en el estado de los enlaces y con requisitos de calidad de servicio.

En la figura 1.9 se puede observar el modelo de referencia definido para el protocolo *PNNI*, así como su capa de control. El modelo de referencia está dividido en tres áreas: la cadena de celdas, las señalizaciones *UNI* (*User to Network Interface*) y *NNI* (*Network to Network Interface*) y el protocolo de gestión de interface. Estas áreas, a su vez, están divididas en las partes mostradas en la figura 1.9. Los módulos de procesado de llamada y de señalización están basados en la serie de recomendaciones Q de la *ITU*, mientras que los módulos encargados del intercambio de topología, gestión de la base de datos de topología y determinación de ruta tienen como objetivo la búsqueda del camino más corto.

Así mismo, en la figura 1.9 se puede observar la pila empleada por el protocolo en su capa de control, siendo esta capa la encargada de procesar los paquetes de señalización. Veamos a continuación con un poco más de detalle la capa de adaptación de señalización (*SAAL, Signaling ATM Adaptation Layer*) existente dentro del plano de control. Esta capa es la responsable de la correcta transmisión de la información en toda conexión *ATM* punto a punto y es, por lo tanto, la responsable de todo lo concerniente a errores en los datos, pérdidas o duplicaciones que puedan ocurrir en el enlace. Esta capa está a

su vez subdividida en:

1. *SSCF (Service Specific Coordination Function)*: su misión es establecer la correspondencia entre los servicios ofrecidos por la capa *SSCOP* (*Service Specific Connection Oriented Protocol*) y las necesidades de los procedimientos de señalización de la capa superior, por ejemplo, el protocolo de control de *PNNI* (aunque esta capa permite utilizar los niveles inferiores con cualquier otro tipo de protocolo de control). Las responsabilidades de esta capa se pueden resumir en:

   - Control de flujo. Esta capa debe notificar al usuario acerca de los niveles de congestión a fin de prevenir la pérdida de paquetes.
   - Estado del enlace. Debe mantener información acerca del estado del enlace como, por ejemplo, enlace en servicio, enlace fuera de servicio, etc. Usando esta información generará primitivas hacia la capa SSCOP a fin de ayudar a gestionar el enlace.
   - Procedimientos de alineación. Posee y mantiene toda la información relativa a los procedimientos de reconfiguración que se encuentran activos cuando se produce un alta (el enlace pasa de no disponible a disponible) o baja (pasa de disponible a indisponible) en el estado de un enlace.

2. *SSCOP (Service Specific Connection Oriented Protocol)*, nivel que debe proporcionar una manera libre de errores de transmitir la información de señalización del protocolo *PNNI*. Entre los servicios proporcionados por esta capa destacan:

   - Transferencia de datos del usuario con integridad de secuencia.
   - Corrección de errores con posibilidad de retransmisión.
   - Control de flujo.
   - Control de conexión.
   - Comunicación de los errores que se hayan producido a la capa de gestión.
   - Mantenimiento de la conexión en periodos largos en los cuales no se produzca transmisión de datos.
   - Comunicación de información de estado.

3. *AAL (ATM Adaptation Layer)*, que proporciona los servicios de segmentación y reensamblado de la información de señalización.

Sin embargo, el gran avance del protocolo *PNNI* frente a los anteriores protocolos de encaminamiento existentes es que está pensado para trabajar con diversas clases de servicio, soportando multiples métricas y proporcionando un camino que soporta calidad de servicio aunque dicho camino sea pseudo-óptimo (es un camino válido pero no el mejor camino de todos los posibles). Pero, sobre todo, *PNNI* proporciona un mecanismo para trabajar con multiples niveles de jerarquía estableciendo un método sistemático de ocultamiento/agregación de la información para poder trabajar en estas condiciones.

1.3 Encaminamiento con *QoS*

Aunque existen propuestas concretas para soportar calidad de servicio sobre algunas de las redes anteriormente citadas, como la red *ATM* a través del protocolo *PNNI*, o en el caso de Internet con extensiones del protocolo *OSPF* [AKW+99], los conceptos básicos que se manejan al tratar de encaminamiento con calidad de servicio son genéricos y aplicables a todas las redes integradas. Por ello, es posible estudiar el encaminamiento abstrayéndose del nivel de red y de sus implementaciones particulares.

Como se ha venido apuntando, el encaminamiento con calidad de servicio surge por la necesidad de garantizar la calidad de servicio extremo a extremo solicitada por los usuarios. Los primeros intentos se basaron en el empleo de complejas funciones de coste que resultaban de una combinación de los distintos parámetros requeridos [WC96b]. No obstante, esta técnica tuvo que abandonarse debido a que el camino que encuentra es un camino promedio que no garantiza el cumplimiento de los requisitos de *QoS* para todos los parámetros. Para conseguir encontrar dicho camino se buscó una solución desde otro punto de vista. En concreto se planteó la modificación de los algoritmos de búsqueda de forma que fueran capaces de encontrar el camino sujeto a múltiples restricciones simultáneas. Como contrapartida de este planteamiento, surge un problema NP-Completo, esto es, un problema cuya solución no está acotada en el tiempo mediante una función polinómica [GJ79], de difícil tratabilidad. Para paliar este problema se han propuesto diversas aproximaciones que permiten encontrar el camino sujeto a múltiples restricciones en un tiempo polinómico.

Uno de los primeros trabajos que permiten resolver este problema es el presentado por Widyono [Wid94], que utiliza una variante del algoritmo Bellman-Ford con la que

consigue encontrar un conjunto de caminos de coste mínimo para un conjunto de restricciones de retardo crecientes. Dentro de este conjunto de caminos se escoge el camino de coste mínimo que no supere la restricción de retardo impuesta. Przygienda [Prz95] presenta un algoritmo parecido a este último que permite resolver el problema de la calidad de servicio, aunque en lugar de emplear una única métrica emplea un conjunto de elementos pareados $\binom{métrica_1}{métrica_2}$, de forma que es capaz de encontar el camino de coste mínimo con respecto a varias métricas. Se dice que un camino P con valores en sus métricas $\binom{métrica_1}{métrica_2}$ tiene un coste menor que un camino P' con valores en sus métricas $\binom{métrica'_1}{métrica'_2}$ sólo si se cumple la expresión:

$$\binom{métrica_1}{métrica_2} \preceq \binom{métrica'_1}{métrica'_2} \text{ si y sólo si } métrica_1 \leq métrica'_1 \text{ y } métrica_2 \leq métrica'_2 \tag{1.1}$$

Con ello se consigue que las dos métricas influyan en la elección del camino. No obstante, este algoritmo consume excesivos recursos de tiempo cuando el número de nodos de la red es elevado, lo que hace poco práctica su implementación. Por otro lado, Wang y Crowcroft [WC96b] [WC96a] proporcionan un algoritmo útil capaz de encontrar un camino que satisface restricciones de retardo y ancho de banda. La limitación que presenta este algoritmo es que una de las métricas ha de ser siempre el ancho de banda, a diferencia de los anteriores que, en principio, pueden trabajar con cualquier parámetro de calidad de servicio. Posteriormente, han aparecido diversos métodos heurísticos que permiten encontrar el camino sujeto a múltiples restricciones [Ma98][CN98a][Che99][Nb00]. Estos algoritmos se estudiarán con más detalle en el capítulo 3.

Un protocolo de encaminamiento capaz de proporcionar calidad de servicio, además de emplear algún algoritmo de búsqueda con restricciones, ha de considerar una serie de características que facilitan la implementación de estos protocolos. Recopilando estas características se recomienda que el protocolo:

- Se base en el estado de los enlaces. Esta condición no es estrictamente necesaria, pero su empleo permite que el sistema de decisión tenga una información más precisa del estado de la red, resultando, por lo tanto, más fácil que la ruta encontrada cumpla los requisitos.

- Sea centralizado o basado en origen. El hecho de construir la ruta completa en un solo punto evita riesgos como la posibilidad de lazos cerrados.

- Se efectúe la búsqueda bajo demanda. No es imprescindible, pero los caminos encontrados suelen ser mejores cuando su búsqueda se realiza bajo demanda (en el momento en que se solicita) empleando la última información disponible del estado de la red. Por contra, si se emplean caminos precomputados, estos deberían actualizarse cuando se produce un cambio importante en el estado de la red.

1.4 Organización de la tesis

La organización de los capítulos de esta tesis se debe a la la evolución de las redes comentada en este capítulo. Así pues, en el capítulo 2 se presenta una extensión de un algoritmo usado en redes telefónicas para poder trabajar en redes *ATM* con múltiples clases de servicio, proponiéndose una solución neuronal para el cálculo del algoritmo que permite su cómputo en tiempo real. Complementariamente, se proponen extensiones del algoritmo mediante la utilización de predicción de la carga de tráfico ofrecido a los enlaces a fin de mejorar el rendimiento del algoritmo básico.

Cuando se diseñó este algoritmo se pensaba que las redes *ATM* tendrían un altísimo grado de interconexionado, gracias a esto lo nodos dispondrían de un alto grado de conocimiento del estado de la red debido que todo nodo conoce el estado de los enlaces a los que está conectado y que cualquier camino que conecte un par de nodos cualesquiera consta, a lo sumo, de la concatenación de dos enlaces. Por todo esto, el algoritmo presentado en este segundo capítulo está diseñado para trabajar en redes muy malladas, suponiéndose que los nodos dispondrían de un conocimiento casi-perfecto del estado de la red. Sin embargo, la evolución de las redes ha demostrado que el escenario predicho no se ha cumplido. El hecho es que las redes actuales de comunicaciones presentan una baja tasa de interconexionado con enlaces de muy alta capacidad. La utilización de esta baja tasa de interconexionado ha dado lugar a que los caminos estén formados por la concatenación de múltiples enlaces y a un bajo grado de conocimiento del estado de la red. Este hecho ha determinado que estos algoritmos hiperespecializados para el entorno *ATM* presenten un bajo rendimiento en este nuevo entorno y surja la necesidad de trabajar con algoritmos menos especializados que sean capaces de adaptarse al cambiante entorno actual.

En el capítulo 3 se estudia el problema del encaminamiento en este nuevo escenario

con redes de bajo grado de interconexión. También se estudia en este capítulo el problema de la búsqueda de caminos que satisfagan de forma simultánea múltiples restricciones de calidad de servicio. Además, también se estudia el problema de que la información que se posee para tomar la decisión de encaminamiento se encuentre desfasada y se investigan posibles métodos para conseguir un buen rendimiento de los algoritmos de encaminamiento con una baja tasa de actualización del estado de la red. Finalmente, en este mismo capítulo se analiza la influencia de la función de distribución de probabilidad empleada para modelar la duración de las llamadas en el encaminamiento. La importancia de esto radica en que hasta hace relativamente poco se había considerado que las llamadas seguían una distribución exponencial. Recientes estudios, por el contrario, demuestran que esto no es cierto, por lo que parece preciso estudiar la influencia que estas distribuciones pueden tener en los algoritmos de encaminamiento.

Por último, en el capítulo 4 se relacionan las conclusiones fundamentales de nuestro trabajo, y las que consideramos son las líneas futuras de investigación que podrían partir de nuestros resultados.

Capítulo 2

Encaminamiento basado en procesos de decisión de Markov

2.1 Introducción

Debido a la similitud existente entre las redes de conmutación de circuitos y las redes de servicios integrados (basadas en circuitos virtuales), es lógico examinar y adaptar, a estas últimas, los algoritmos de encaminamiento dinámico inicialmente desarrollados para la RTC. Es posible clasificar a los actuales algoritmos de encaminamiento, utilizados en la RTC, dentro de uno de dos grandes grupos [Hwa93], en función del tipo de métrica en la que se apoya: el primer grupo sigue una métrica cóncava (esto es, asigna como coste de los caminos los puntos que imponen más restricciones o "cuellos de botella") y lo constituyen los algoritmos de búsqueda del camino menos cargado (*Least Loaded Routing* o *LLR*). La forma de trabajo de este grupo consiste en buscar los puntos críticos de los distintos caminos posibles y, una vez localizados, se comparan para escoger el camino cuyo punto crítico disponga de la mayor cantidad de recursos libres. Un ejemplo de este tipo de algoritmos se puede ver aplicado a la red de la figura 2.1, donde el ancho de banda disponible en el enlace es el recurso que determina el enlace crítico. En esta red existe la posibilidad de establecer una conexión entre los nodos 1 y 4 por dos posibles caminos. Al aplicar un algoritmo de tipo *LLR*, se busca cuál de los dos enlaces que forma cada camino dispone del menor ancho de banda disponible y este valor será el coste total del camino. Seguidamente, se comparan los costes de los distintos caminos y se selecciona el camino con mayor ancho de banda disponible (mayor coste) o, lo que es lo mismo, el camino menos cargado entre los dos posibles. El otro grupo de algoritmos sigue una

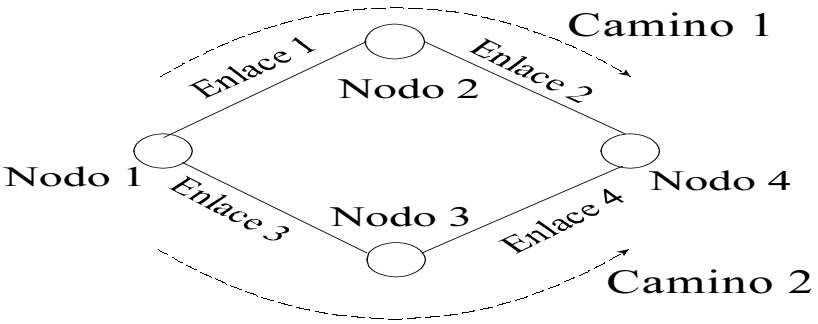


Figura 2.1: Red básica con dos posibles caminos.

métrica aditiva o multiplicativa (o lo que es lo mismo, el coste de un camino se obtiene sumando o multiplicando los costes de los enlaces que lo componen), y están basados en los procesos de decisión de Markov (*Markov Decision Process* o *MDP*). Este tipo de algoritmos asigna un coste a cada camino en función del estado de los diversos enlaces de los que se compone. El objetivo es buscar el camino de coste mínimo (o máximo), sin embargo, a diferencia de los algoritmos *LLR* el coste del camino no depende del estado de un enlace individual, si no que en él influyen, conjuntamente, los estados de todos los enlaces de los que consta. Debido a que los algoritmos *MDP* utilizan el estado de todos los enlaces que atraviesa, éstos presentan un mejor rendimiento que los algoritmos *LLR* [OK85][Hwa93]. Por esta razón, en este capítulo se estudiará la posibilidad de extender los algoritmos *MDP* desarrollados para la RTC a fin de utilizarlos en una red de servicios integrados como *ATM*.

No obstante, muchos de los algoritmos *MDP* desarrollados para la RTC presentan dos inconvenientes para su aplicación en las redes de alta velocidad. El primero de ellos es el elevado coste computacional que presentan cuando deben soportar más de una clase de tráfico. Un ejemplo de este problema se puede ver en el estudio de Dziong *et al.*[DM89] donde extienden su trabajo previo desarrollado para la RTC a redes con múltiples clases de llamada. Este estudio constata que la complejidad del problema crece velozmente conforme aumenta el número de clases de llamadas, de forma que rápidamente se hace imposible su empleo en una red real donde el cálculo del coste debe realizarse en tiempo real. En este capítulo proponemos la utilización de redes neuronales a fin de solucionar esta problemática. El segundo de los problemas que presentan los algoritmos *MDP* es la supuesta independencia entre estados de los enlaces. En realidad, el estado en el que se encuentra un enlace influye en el estado del resto de los enlaces de la red. Considerando

estas circunstancias de correlación la complejidad del problema es tal que es imposible resolverlo. Por esta razón, es necesario trabajar bajo la suposición de la existencia de independencia entre enlaces. Este hecho implica que el estado en el cual se encuentra un enlace no influye en el estado del resto de los enlaces de la red. La importancia de este problema ya ha sido estudiada por Hwang [Hwa93] que demuestra que el rendimiento de los algoritmos se ve escasamente influenciado por esta suposición, por lo que no será objeto de estudio en esta tesis.

2.2 Procesos de decisión de Markov

Existe en la actualidad una abundante bibliografía acerca de los procesos de decisión de *Markov* [Put94][Ros95], por esta razón nos centraremos en el modelado, mediante este tipo de procesos, del comportamiento de una red de comunicaciones. Como ya se ha comentado y puesto que el espacio de estados de los procesos *MDP* necesario para modelar con fidelidad una red de tamaño medio es enorme, asumiremos la existencia de independencia entre los estados de los distintos enlaces de la red, a fin de reducir el tamaño del espacio de estados. Esta suposición ya ha sido empleada en trabajos como [OK85][Kri91][Hwa93] y [MSS00].

2.2.1 Modelo de red

Consideremos una red formada por un conjunto V de nodos unidos entre sí por un conjunto de enlaces E, donde cada enlace $e_{ij} \in E$ de este conjunto es identificado por medio de los nodos que conecta y dispone de una capacidad de C_{ij} unidades de ancho de banda. Así mismo, existe un conjunto W de todas las posibles solicitudes de conexión entre dos nodos origen-destino. Al ser una red multiservicio, existen K clases distintas de tráfico, cada una de ellas identificada mediante un índice $k = (1, 2, \ldots, K)$ y por las b_k unidades de ancho de banda que requiere (asumimos que $b_1 < b_2 < \ldots < b_{K-1} < b_K$). Suponemos que la llegada a la red de las llamadas de la clase k que solicitan una conexión entre un par de nodos origen-destino w, sigue una distribución de *Poisson* de tasa λ_w^k. La tasa total de llamadas de la clase k que solicitan una nueva conexión será $\lambda_k = \sum_{w \in W} \lambda_w^k$. Por otro lado, la duración de las llamadas para cada clase vendrá dada por una distribución exponencial de media $1/\mu_k$.

La asunción de que las llamadas que ingresan en la red siguen una distribución de

Poisson es adecuada si se cumplen una serie de requisitos [Bou88]: Por un lado que el número de usuarios sea elevado (suposición que se cumple en las redes de area extensa (*WAN*)) y, además, que la tasa de pérdida de llamadas en la red sea baja. Esta condición es necesaria (siempre que se permitan reintentos de llamada) para evitar la presencia de correlación en la generación de tráfico. Sin embargo, si se tiene en cuenta que las redes reales se dimensionan para que la probabilidad de bloqueo de llamada sea menor de un 2% en el peor de los casos, el empleo de la distribución de *Poisson* para la generación de llamadas, aun cuando se permitan reintentos de llamada, sigue siendo una aproximación razonable. Como se verá en el apartado 3.7, la utilización de la función de distribución exponencial para modelar la duración de llamada es un modelo generalista válido frente a otras funciones de distribución.

Asumiendo la independencia entre enlaces podemos modelar el estado del sistema mediante una matriz $\mathbf{X}$ de KxE elementos, donde el elemento $x_{k,ij} \in \mathbf{X}$ indica el número de conexiones de la clase k presentes en el enlace e_{ij}. El estado en el que se encuentra el enlace e_{ij} debe satisfacer la condición de no saturación impuesta por la siguiente ecuación:

$$\sum_{k=1}^{K} b_k \; x_{k,ij} \;\leq\; C_{ij} \tag{2.1}$$

Es posible modelar [DM89][Hwa93][KH95][Dzi97] el comportamiento de todo el sistema mediante un proceso de nacimiento y muerte (véase la figura 2.2), donde $\lambda_{k,ij}$ simboliza el tráfico ofrecido al enlace e_{ij} por el tráfico de clase k y $1/\mu_k$ es la duración media de las llamadas de la clase k. Es evidente que el tráfico ofrecido al enlace depende, entre otras causas, del algoritmo de encaminamiento empleado. Por esta razón, la elección de un determinado algoritmo debe realizarse en función del objetivo concreto buscado. En nuestro caso, el objetivo buscado es maximizar el aprovechamiento de los recursos de la red (*throughput*) repartiendo el tráfico ofrecido a la red entre los diversos enlaces a fin de minimizar el tráfico rechazado por ésta, satisfaciendo, simultáneamente, los requisitos de calidad de servicio exigidos por cada tipo de tráfico.

2.3 Algoritmos de encaminamiento basados en procesos de decisión de Markov

Como se ha visto, se puede modelar el comportamiento de una red mediante cadenas de Markov. Por esto se dice que están basados en procesos de decisión de *Markov* todos

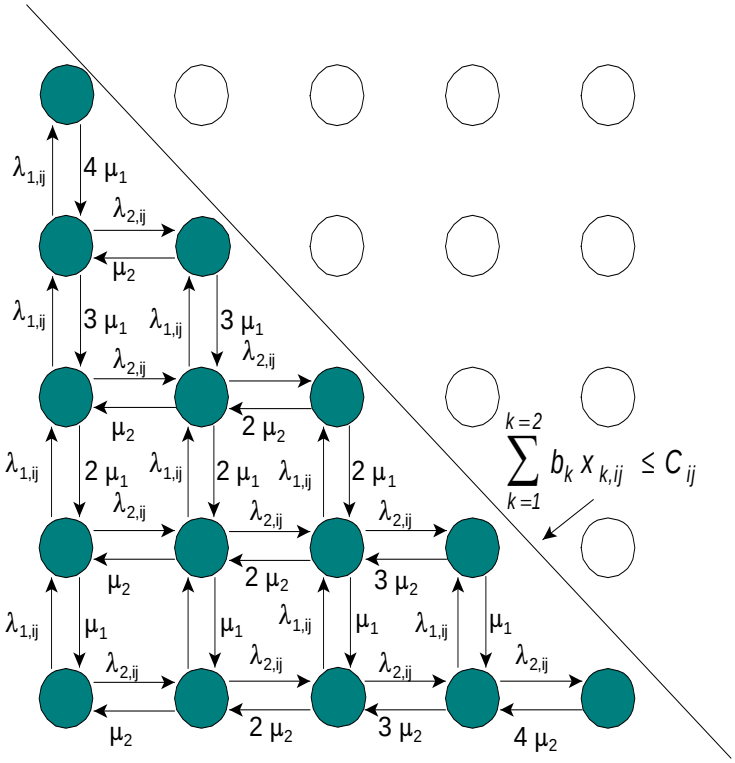


Figura 2.2: Ejemplo de la cadena de estados para el enlace e_{ij} con dos clases de tráfico (K=2).

los algoritmos de encaminamiento que emplean, como criterio de decisión para elegir un camino, el estado de los enlaces de los que constan los distintos caminos existentes entre un nodo origen y un nodo destino [DM89] [Hwa93] [KH95].

Además de lograr maximizar el *throughput* es necesario conseguir, de forma simultánea, que se cumplen los requisitos de calidad de servicio. Para alcanzar estos objetivos es necesario apoyarse en los distintos controles existentes en la red. Puesto que la red *ATM* emplea un control de admisión para garantizar estos requisitos de *QoS* solicitados por las llamadas, es razonable pensar que los distintos algoritmos de encaminamiento se deberían apoyar en éstos a fin de conseguir que los caminos seleccionados para las distintas conexiones cumplen los requisitos de *QoS* exigidos. El coste asignado al enlace debería depender, por lo tanto, no sólo del estado del enlace, sino además del control de admisión empleado. En una primera aproximación es lógico asignar a cada enlace un coste proporcional a la probabilidad de rechazo de llamada en el enlace. En

los apartados siguientes se expone cómo se obtiene dicha probabilidad y se presenta un algoritmo que asigna un coste en función de ésta.

2.3.1 La función de Erlang

Inicialmente, estudiaremos el comportamiento de un enlace individual para un único tipo de tráfico. Puesto que hemos supuesto la independencia entre enlaces, este comportamiento es extensible a todos los enlaces de la red en presencia de un único tipo de tráfico. Con posterioridad extenderemos el sistema para múltiples clases de tráfico.

El comportamiento de un enlace genérico se puede modelar mediante una cola de tipo $G/G/N/N$. Suponiendo que las llamadas llegan al enlace siguiendo una distribución de *Poisson* y que su duración se rige por una distribución exponencial, el comportamiento de un enlace individual es igual al comportamiento de una cola $M/M/N/N$ donde N es la capacidad del enlace, es decir, el número de llamadas que el enlace es capaz de atender simultáneamente. El comportamiento de este sistema de colas fue estudiado por el matemático Erlang y la probabilidad de pérdida de llamada viene dada por la conocida fórmula de *Erlang-B,* cuya expresión es:

$$B(N,\rho) \;=\; \frac{\frac{\rho^N}{N!}}{\sum_{i=0}^{N} \frac{\rho^i}{i!}} \tag{2.2}$$

donde $\rho = \lambda/\mu$ es el tráfico ofrecido al enlace y N es la capacidad del enlace.

Esta expresión es numéricamente difícil de estimar, ya que su cálculo implica la utilización de grandes números que rápidamente exceden la capacidad de representación interna de los ordenadores actuales, por lo que es conveniente sustituir la expresión anterior por la siguiente expresión iterativa que permite un cálculo numérico mucho más sencillo:

$$B(N,\rho) \;=\; \begin{cases} 1 & si \quad N = 0 \\ \frac{\rho\ B(N-1,\rho)}{\rho\ B(N-1,\rho)+N} & si \quad N = 1,2,... \end{cases} \tag{2.3}$$

Sin embargo, la función de *Erlang-B* únicamente permite obtener la probabilidad de pérdida de llamada para una única clase de llamada. Es necesario modificar esta expresión a fin de poder extenderla a sistemas con múltiples clases de llamadas. Para ello, es necesario tener en cuenta la política empleada por el control de admisión.

2.3.1.1 La función de Erlang-B generalizada

El cálculo de la función de *Erlang-B* generalizada depende de la región del espacio de estados que el control de admisión de cada enlace determina como región válida. De este modo, será el comportamiento del *CAC* quien determine la probabilidad de rechazar una llamada.

Para calcular la probabilidad de bloqueo que un determinado control de admisión provoca en un enlace e_{ij}, de capacidad C_{ij} unidades de ancho de banda y en presencia de K clases distintas de tráfico, con requisitos de ancho de banda efectivo de b_k cada una de ellas, emplearemos un vector $\mathbf{x}_{ij} = \{x_{1,ij}, \cdots, x_{l,ij}, \cdots, x_{K,ij}\}$ que indica cuántas llamadas de cada clase hay en un momento determinado en el enlace y, por lo tanto, identifica el estado en el que se encuentra. Por otro lado, las variables $\lambda_{k,ij}$ y μ_k representan, respectivamente, la tasa de llamadas de la clase k que llegan al enlace y la tasa de llamadas de la clase k que finalizan. Definimos $\rho_{k,ij} = \lambda_{k,ij}/\mu_k$ como el tráfico de la clase k ofrecido al enlace e_{ij}.

Lógicamente, el conjunto de estados válidos del enlace debe cumplir la condición $S_{ij} = \{\mathbf{x}_{ij} : \mathbf{b} \cdot \mathbf{x}_{ij} \leq C_{ij}\}$ donde $\mathbf{b}$ es el vector $\mathbf{b} = [b_1, \ldots, b_K]$. La política de admisión determinará si una llamada entrante de clase k debe ser aceptada o rechazada. En nuestro caso, una nueva llamada entrante de clase k será aceptada sólo si existe suficiente ancho de banda, es decir, si se verifica la condición:

$$\mathbf{x}_{ij}\mathbf{b} + b_k \leq C_{ij} \tag{2.4}$$

En caso contrario la llamada será rechazada. Teniendo en cuenta todo lo anterior, la probabilidad de que el enlace e_{ij} se encuentre en un estado $\mathbf{x}$ viene dada por la expresión [Ros95]:

$$\pi_{ij}(\mathbf{x}) = \frac{\prod_{k=1}^{K} \rho_{k,ij}^{x_{k,ij}}/x_{k,ij}!}{\sum_{\mathbf{x}\in S_{ij}} \prod_{k=1}^{K} \rho_{k,ij}^{x_{k,ij}}/x_{k,ij}!} \qquad \mathbf{x} \in S_{ij} \tag{2.5}$$

donde

$$S_{ij} = \{\mathbf{x} : \sum_{k=1}^{K} x_{k,ij}\ b_k \leq C_{ij}\} \tag{2.6}$$

y la probabilidad de rechazar una llamada de la clase k vendrá dada por la expresión:

$$B_{k,ij} = 1 - \sum_{\mathbf{x}\ \in\ S_{k,ij}} \pi(\mathbf{x}) =$$
$$1 - \frac{\sum_{\mathbf{x}\in S_{k,ij}} \prod_{k=1}^{K} \rho_{k,ij}^{x_{k,ij}}/x_{k,ij}!}{\sum_{\mathbf{x}\in S_{ij}} \prod_{k=1}^{K} \rho_{k,ij}^{x_{k,ij}}/x_{k,ij}!} \tag{2.7}$$

donde $S_{k,ij}$ es el subconjunto de estados de S_{ij} en el cual el control de admisión admitirá una llamada de la clase k y viene dada por la expresión:

$$S_{k,ij} = \{\mathbf{x} \in S_{ij} : \sum_{k=1}^{K} x_{k,ij} b_k \leq C_{ij} - b_k\} \tag{2.8}$$

Una vez que se conoce la expresión que determina la probabilidad de rechazar una llamada en un enlace es posible estimar la probabilidad de rechazar una llamada en un determinado camino. Puesto que hemos supuesto la independencia entre enlaces, podemos expresar la probabilidad de rechazo de una llamada de la clase k a través de un camino P formado por los enlace $e_{al}, \ldots, e_{ij}, \ldots, e_{mb}$ y que une a los nodos a y b mediante la siguiente ecuación:

$$B_k = 1 - \prod_{e_{ij} \in P} (1 - B_{k,ij}) \tag{2.9}$$

Si el objetivo es encontrar el camino que reduce la probabilidad de rechazar la llamada se debe minimizar la ecuación (2.9), lo que, tras una transformación logarítmica, es equivalente a minimizar la siguiente expresión [GO97]:

$$\sum_{e_{ij} \in P} -\log(B_{k,ij}) \tag{2.10}$$

Esto permite aplicar el algoritmo de Dijkstra para buscar el camino de menor probabilidad de pérdida [Dij59], ya que éste está diseñado para trabajar con métricas aditivas.

Como se puede apreciar, el empleo de la probabilidad de bloqueo como función de coste hace posible que cada clase de llamada tenga un coste distinto, posibilitando un tratamiento diferenciado a cada una de las distintas clases de tráfico existentes en la red. Así, se podría decir que existen K clases distintas de redes virtuales, cada una para una clase de tráfico. Es evidente que el camino encontrado aplicando este método debe cumplir, al menos, los requisitos de calidad de servicio enlace a enlace, puesto que se apoya en el CAC de los enlaces.

Este algoritmo presenta, sin embargo, algunas deficiencias. La más grave de ellas es la respuesta en condiciones de picos puntuales en la ocupación del enlace producidos por rápidas oscilaciones en el número de llamadas ofrecidas. En esta circunstancia, aun con un tráfico medio bajo (y por lo tanto, un coste bajo), se perderá un elevado número de llamadas debido a que se ofrecería al enlace un número considerable de llamadas en un tiempo muy breve (no hay que olvidar que el tráfico es un promedio de ocupación y

no refleja las cargas puntuales). La segunda de las deficiencias es el enorme coste computacional del cálculo de la función *Erlang-B*, el cual, en principio, impediría emplear el algoritmo en implementaciones reales. Por último, se debería también considerar la dificultad de medir el tráfico ofrecido al enlace, ya que lo que es posible medir directamente es el tráfico cursado por el enlace, siendo necesario algún mecanismo capaz de estimar el tráfico ofrecido a partir de la medida del tráfico cursado.

A continuación vamos a ir viendo las distintas soluciones utilizadas para resolver estos problemas.

2.3.2 El algoritmo EASDR

Krishnan y Ott [KO88] desarrollaron un algoritmo para el encaminamiento de llamadas en la RTC basado en la aplicación de la teoría de los procesos de decisión de *Markov*. En este caso el coste del enlace refleja la ganancia obtenida al aceptar (o rechazar) una llamada frente al hecho de aceptar (o rechazar) las futuras llamadas que se estiman que puedan llegar. El coste asignado al enlace viene dado por la expresión:

$$coste_{ij} = \frac{B_{ij}(\rho_{ij}, C_{ij})}{B_{ij}(\rho_{ij}, c_{ij})} \tag{2.11}$$

donde B_{ij}, que se obtiene a partir de la ecuación (2.2), es la probabilidad de bloqueo de llamadas en el enlace e_{ij}, ρ_{ij} es el tráfico ofrecido al enlace, C_{ij} es el número de unidades de ancho de banda del enlace y c_{ij} representa el número de unidades de ancho de banda ocupadas en un determinado momento en el enlace, cumpliéndose que $c_{ij} \leq C_{ij}$. El total del un determinado camino P será:

$$coste_P = \sum_{e_{ij} \in P} \frac{B_{ij}(\rho_{ij}, C_{ij})}{B_{ij}(\rho_{ij}, c_{ij})} \tag{2.12}$$

Este algoritmo impone una **restricción de coste**, de forma que la llamada únicamente será aceptada si el coste total del camino es menor que 1.

Como ya se apuntó, uno de los problemas que presenta esta expresión viene dado por la dificultad de conocer el tráfico ρ_{ij} ofrecido al enlace, ya que lo que realmente es posible medir es el tráfico cursado en los enlaces y el tráfico total ofrecido a la red. El valor del tráfico ofrecido a cada enlace solo se obtiene fácilmente en el caso de tráfico uniformemente distribuido (todos los nodos generan tráfico con la misma tasa y existe la misma probabilidad de escoger un determinado destino). En caso contrario, es necesario

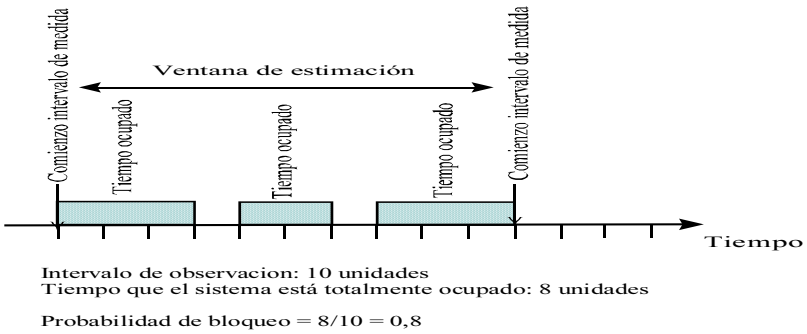


Figura 2.3: Ejemplo del cálculo del porcentaje de tiempo $\widetilde{B}_{ij}$.

resolver un complejo sistema de ecuaciones diferenciales [KO88]. A fin de solucionar el problema de la obtención del tráfico ofrecido al enlace, Krishnan propone en [Kri91] un algoritmo al que denomina *ASDR* (*Adaptive State−Dependent Routing*) el cual permite, mediante medidas del tráfico cursado por el enlace realizadas a intervalos periódicos, estimar el tráfico ofrecido a éste. Para poder realizar esta medida es necesario distinguir entre:

- λ_{ij} número medio de llamadas ofrecidas al enlace por unidad de tiempo (valor que deseamos conocer)
- $\widetilde{\lambda}_{ij}$ número medio de llamadas soportadas por el enlace e_{ij} por unidad de tiempo (medida que se realiza en el enlace).
- $\widetilde{B}_{ij}$ porcentaje de tiempo que el enlace está "bloqueado", es decir, que no hay recursos suficientes para aceptar una llamada.

A partir de esto datos de $\widetilde{\lambda}_{ij}$ y $\widetilde{B}_{ij}$, se puede aproximar el tráfico ofrecido al enlace mediante la expresión:

$$\lambda_{ij} = \frac{\widetilde{\lambda}_{ij}}{1 - \widetilde{B}_{ij}} \tag{2.13}$$

En la figura 2.3 se muestra un ejemplo gráfico de una medida del valor de $\widetilde{B}_{ij}$.

Para poder aplicar el algoritmo *ASDR* a las redes *ATM* es necesario realizar algunas modificaciones al algoritmo básico, ya que ahora existen múltiples clases de llamada.

2.3.2.1 Algoritmo EASDR o *Extended ASDR*

Para extender el algoritmo *ASDR* a redes multiservicio es necesario sustituir la expresión de la ecuación (2.2), para el cálculo de la probabilidad de bloqueo por la expresión obtenida en la ecuación (2.7) [ALGS96][ALGS98], con lo cual, el coste del enlace e_{ij} para la clase de llamada k queda:

$$coste_{k,ij} = \begin{cases} \frac{B_{k,ij}\ (\rho_{1,ij},\cdots,\rho_{k,ij},\cdots,\rho_{K,ij},C_{ij}\ )}{B_{k,ij}\ (\rho_{1,ij},\cdots,\rho_{k,ij},\cdots,\rho_{K,ij},c_{ij}\ )} & si\ \ c_{ij}+b_k \leq C_{ij} \\ 1 & si\ \ c_{ij}+b_k > C_{ij} \end{cases} \tag{2.14}$$

donde c_{ij} es el ancho de banda ocupado en el enlace y se calcula :

$$c_{ij} = \sum_{k=1}^{K} b_k\ x_{k,ij} \tag{2.15}$$

siendo $x_{k,ij}$ el número de llamadas de la clase k que transporta actualmente el enlace e_{ij}. También es necesario reescribir la ecuación (2.13), ya que ahora hay K clases distintas de tráfico, con lo que el tráfico ofrecido por cada clase queda:

$$\lambda_{k,ij} = \frac{\widetilde{\lambda}_{k,ij}}{1-\widetilde{B}_{k,ij}} \tag{2.16}$$

donde:

- $\lambda_{k,ij}$ es el número medio de llamadas del tipo k ofrecidas al enlace por unidad de tiempo (valor que deseamos conocer)
- $\widetilde{\lambda}_{k,ij}$ es el número medio de llamadas de la clase k soportadas (y medidas) en el enlace e_{ij} por unidad de tiempo.
- $\widetilde{B}_{k,ij}$ es el porcentaje de tiempo que el enlace está "bloqueado" para las llamadas de tipo k, es decir, el porcentaje de tiempo que se cumple la condición $C_{ij} - c_{ij} < b_k$.

Igualmente, es necesario modificar la **restricción de coste** aplicada en el algoritmo *ASDR*, según la cual la llamada es rechazada si el coste total del camino es mayor o igual a uno. Esta condición sólo es aplicable a las redes donde siempre exista un enlace directo entre dos nodos cualesquiera. Sin embargo, esta condición no tiene por qué ser cierta en las redes *ATM* (no hay que olvidar que este tipo de redes ha sido diseñado para poder trabajar en todo tipo de entornos), lo que obliga a realizar una modificación en la

condición **restricción de coste** a fin de poder aplicar este algoritmo. La **restricción de coste** aplicada a un camino P queda entonces como indica la siguiente expresión:

$$coste_{k,P} = \sum_{e_{ij} \in P} coste_{k,ij} < \Gamma_k(n) * n \qquad \Gamma_k(n) \leq 1 \tag{2.17}$$

donde n es el número de enlaces de que consta el camino P y $\Gamma_k(n)$ es una variable de control que depende de la clase de tráfico y del número de enlaces que posee el camino e indica la penalización que se aplica a un determinado tipo de tráfico al escoger un determinado camino. Esta variable de control permite priorizar o castigar una determinada clase de tráfico frente a otra y limitar la elección de caminos con multitud de enlaces, a menos que haya una abundante disponibilidad de recursos en el camino. La penalización de un determinado tipo de tráfico tiene como objetivo limitar la aceptación de una determinada clase a fin de beneficiar la aceptación, por parte de la red, de las restantes clases. Esto posibilita, por ejemplo, favorecer las llamadas de corta duración y que consumen poco ancho de banda (como las llamadas telefónicas), frente a llamadas que demandan gran cantidad de recursos (como podría ser la transmisión de vídeo). No hay ningún método que determine el valor óptimo de esta variable, y su definición depende, básicamente, de la política que desee aplicar el operador.

Este algoritmo de encaminamiento sólo es capaz de garantizar que se cumplen los requisitos de calidad de servicio solicitados por la llamada enlace a enlace (en el siguiente capítulo se mostrarán diversos algoritmos que permiten cumplir los requisitos de *QoS* extremo a extremo empleando versiones modificadas de los algoritmos de Dijkstra [Dij59] o Bellman-Ford [Bel57]), ya que se apoya en el control de admisión para establecer el coste del enlace. Sin embargo, el empleo de valores restrictivos de la variable $\Gamma_k(n)$ permite garantizar una alta probabilidad de que las conexiones establecidas cumplan los requisitos de calidad de servicio extremo a extremo requeridos.

Como se puede observar en la expresión (2.14), si no hay suficientes recursos para admitir una llamada de una determinada clase en el enlace, el coste asignado a éste es uno. Esto permite identificar los enlaces que no satisfacen los requisitos solicitados por la llamada entrante, de manera que podemos eliminarlos de la topología a emplear para buscar el camino entre el par de nodos origen-destino. El organigrama completo del algoritmo se ha representado en la figura 2.4, donde es posible apreciar que antes de realizar la búsqueda del camino se eliminan los enlaces con un coste igual a 1. La acción de eliminar estos enlaces permite reducir el tiempo que emplea el algoritmo de búsqueda (Dijkstra o Bellman-Ford) para encontrar el camino de coste mínimo.

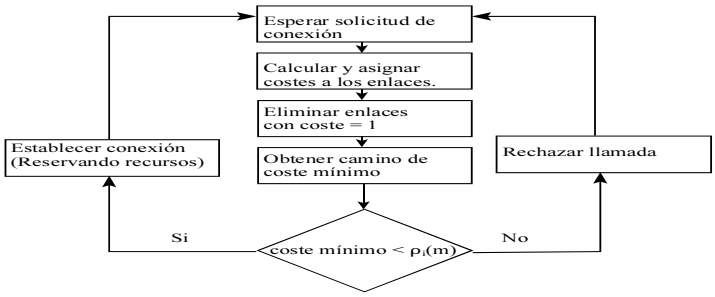


Figura 2.4: Organigrama del algoritmo.

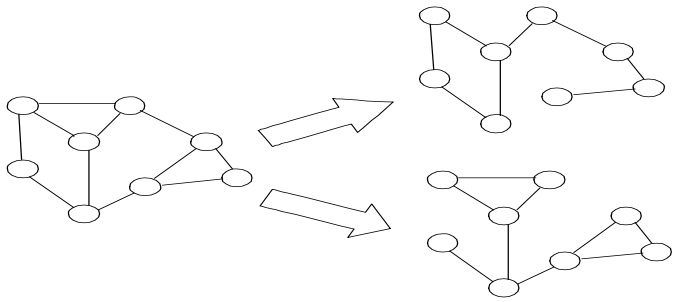


Figura 2.5: Descomposición funcional de una red en dos subredes lógicas para dos tipos de tráfico.

El algoritmo *EASDR* presenta dos claras virtudes: por un lado, el propio algoritmo funciona como un control de admisión y, por otro, permite tratar cada tipo de tráfico de forma independiente, como si tuviéramos K redes virtuales independientes (figura 2.5). A continuación se expondrán cada uno de estos puntos con más detalle.

1. *ATM* está basado en un encaminamiento en origen [For96], es decir, el nodo generador de la llamada determina cuál será el camino hasta el nodo destino. Una vez determinado el camino se procede a reservar los recursos en los enlaces del camino, existiendo la posibilidad de que alguno de los controles de admisión intermedios rehúse la llamada, con lo cual la llamada será rechazada. Este rechazo implica una pérdida de tiempo y ancho de banda en la reserva y posterior liberación de los recursos que la llamada había reservado y que no van a ser usados. A fin de minimizar la posibilidad del rechazo de la llamada en la fase de establecimiento,

el ATM Forum [For96] propone el empleo de un control de admisión genérico o *GCAC* (*Generalized Call Admission Control*), presente en el nodo origen, que permite evaluar si existe una alta probabilidad de que la llamada sea aceptada antes de empezar a reservar los recursos [For96]. De esta manera, si no se supera la prueba del *GCAC* se estima que no existen suficientes recursos a lo largo del camino para establecer la llamada, con lo cual, o bien se rechaza la llamada o se busca un camino alternativo. El propio algoritmo *ASDR* extendido (*EASDR*) funciona como un control de admisión, ya que si el coste de un enlace es 1 se sabe positivamente que dicho enlace no pasaría la prueba del control de admisión genérico. En cambio, los caminos encontrados y que cumplen el criterio establecido por la **restricción de coste** cumplirán las restricciones impuestas por el control de admisión genérico.

2. La otra ventaja que presenta este algoritmo, como se ha comentado anteriormente, es la capacidad de dividir la red real, compartida por las distintas clases de llamada, en K redes virtuales, cada una de ellas correspondiente a cada una de las clases de llamada existentes en la red. Esto se produce debido al hecho de que el algoritmo *EASDR* asigna un coste distinto a las distintas clases, lo que posibilita tratar cada red de forma independiente pero sin perder las ventajas de la ganancia estadística obtenida al combinar varias clases de tráfico sobre los mismos enlaces virtuales.

2.3.3 Algoritmos que reducen el coste computacional

Todas las variantes de algoritmos basados en *MDP* que emplean la probabilidad de bloqueo de llamada como parte de su métrica (como son los algoritmos propuestos por Dziong [DM89], Hwang [Hwa93] o como el propio algoritmo *ASDR*) presentan un elevado coste computacional. Este hecho restringe su aplicabilidad en sistemas reales que han de manejar un gran volumen de datos. Por lo tanto, para poder emplear estos algoritmos en sistemas reales es necesario encontrar métodos que reduzcan su tiempo de cómputo. Existen varias alternativas a fin de conseguir reducir estos tiempos, por ejemplo Hwang [Hwa93] basándose en el criterio de maximización del beneficio y empleando la estrategia denominada *link shadow cost* [Kel88] [Dzi97], propone una modificación que permite reducir el espacio de estados del enlace y, por ende, el coste computacional. Sin embargo, esta simplificación sólo es aplicable si el tráfico ofrecido al enlace llega siguiendo una distribución de *Poisson* y tiene una duración de tipo exponencial; además, supone que las clases de tráfico están perfectamente delimitadas y el ancho de banda que precisa cada clase está cuantificado y se conoce a priori, es decir, el ancho de banda requerido

por cada clase de llamada es determinista y es posible cuantificarlo mediante un número entero. En cualquier caso, esta mejora sigue sin resolver el problema de la estimación del tráfico ofrecido al enlace.

Se han propuesto diversas alternativas que permiten reducir el coste computacional de obtener la probabilidad de bloqueo mediante la aplicación de un algoritmo recursivo [Rob81][Ros95][RME96]. A continuación se expondrá el algoritmo propuesto por Roberts [Rob81] para el cálculo de la probabilidad de bloqueo.

Se define como Δ_c el máximo común divisor de los anchos de banda b_k, con $k = \{1, \ldots, K\}$, requeridos por las distintas clases de llamada y expresados en unidades de ancho de banda. Sea un enlace de capacidad C, definimos como $L = \lfloor C/\Delta_c \rfloor$ el número máximo de unidades de ancho de banda normalizadas del enlace y $l_k = \lfloor b_k/\Delta_c \rfloor$ el número de unidades de ancho de banda normalizadas que exige cada clase de llamada.

La probabilidad $\tilde{p}\ (l)$ no normalizada de que estén ocupados l unidades de ancho de banda normalizadas viene dada por la siguiente ecuación recursiva:

$$\tilde{p}\ (l)\ = \begin{cases} 0 & l < 0 \\ 1 & l = 0 \\ \frac{1}{l}\ \sum_{k=1}^{K} l_k\ \rho_k\ \tilde{p}(l - l_k) & l = 1, \ldots, L \\ 0 & l > L \end{cases} \tag{2.18}$$

Este valor de probabilidad no está normalizado, por lo que continuación es necesario su normalización con lo que queda:

$$p\ (l)\ = \frac{\tilde{p}(l)}{\sum_{l=0}^{L} \tilde{p}(l)} \tag{2.19}$$

A partir de estos valores es fácil calcular la probabilidad B_k de rechazar una llamada de la clase k mediante la expresión:

$$B_k = \sum_{l=L-l_k+1}^{L} p(l) \tag{2.20}$$

La complejidad computacional del algoritmo, aunque menor que la función de *Erlang-B* generalizada, todavía es elevada; sin embargo, se han propuesto diversas aproximaciones que permiten calcular de manera rápida un valor aproximado de la probabilidad de bloqueo de llamada.

A continuación se expondrá el método propuesto por Lindberger [Lin94]. Este método caracteriza la media m y la varianza v resultantes de la superposición de K clases distintas de llamada como:

$$\begin{aligned} m &= \sum_{k=1}^{K} \frac{\lambda_k}{\mu_k} b_k \\ v &= \sum_{k=1}^{K} \frac{\lambda_k}{\mu_k} b_k^2 \end{aligned} \tag{2.21}$$

donde λ_k, como ya se apuntó, es la tasa de llegada de las llamadas de la clase k y μ_k es la duración media de las llamadas de la clase k. Si se definen b_{min} y b_{max} como los valores mínimo y máximo respectivamente que toma $b_k \forall k \in K$, la probabilidad media de bloqueo de llamada B_{med} y la mayor probabilidad de bloqueo individual B_{max} se pueden estimar por la siguiente expresión:

$$\begin{aligned} B_{med} &= B\left(\tfrac{C-b^*+b_{min}}{b^*}, \tfrac{m}{b^*}\right) \\ B_{max} &= B_{med} \cdot \tfrac{b_{max}}{b^*} \cdot \left(\tfrac{C}{m}\right)^{(b_{max}-b^*)/2b^*} \end{aligned} \tag{2.22}$$

donde b^* es la relación entre momentos v/m y $B()$ es la fórmula de *Erlang-B*. A continuación se define el valor de ϵ como:

$$\epsilon = (C/m)^{1/2b^*} - 1 \approx (C/m - 1)/2b^* \tag{2.23}$$

A partir de las expresiones enumeradas la probabilidad de pérdida individual de cada clase puede ser aproximada mediante la siguiente expresión:

$$\frac{B_k}{B_{med}} \simeq \frac{b_k(1 + b_k\epsilon)}{b^* + \epsilon \sum_{k=1}^{K} \rho_k \; b_k^3/m} \tag{2.24}$$

Sin embargo, el cálculo de la probabilidad de bloqueo mediante esta aproximación presenta varias limitaciones, la primera de ellas es que no es aplicable a sistemas que utilizan reserva de recursos (*Trunk Reservation*), la segunda limitación es debida a la suposición de que tanto la duración como el tiempo entre llamadas siguen una distribución exponencial, por último, esta aproximación presenta un error que puede ser de hasta el 10%.

A continuación presentaremos un modo alternativo de resolver estos problemas mediante el empleo de redes neuronales introducidas para estimar la probabilidad de bloqueo de llamada.

2.4 Aplicaciones neuronales al encaminamiento

2.4.1 Introducción

La aplicación de redes neuronales a la resolución de problemas de comunicaciones no es novedosa, de hecho, es relativamente fácil encontrar diversas aplicaciones neuronales en los modernos sistemas de comunicaciones. Un ejemplo de esto lo podemos ver en los filtros adaptativos empleados en los moduladores-demoduladores o módems (que son, en el fondo, redes neuronales sencillas), o el autómata con aprendizaje [NWM77] propuesto para resolver el problema del encaminamiento de forma distribuida y cuya estructura es idéntica a la red neuronal *adaline* [CU92]. Sin embargo, la idea de intentar resolver el problema de la búsqueda del camino de coste mínimo surge con fuerza a partir de la aplicación de la red de *Hopfield* [Hop82][Hop84] [HT85] para la resolución de problemas **NP-completos** como, por ejemplo, el problema del viajante [FS91]. Puesto que el problema de la búsqueda del camino de coste mínimo es similar al problema del viajante, se propuso la utilización de la red de *Hopfield* para su resolución [ZT89] [MAK93] [CSM93]. La red de *Hopfield* tiene asociada una función de energía. Así, al permitir a la red evolucionar en el tiempo la red busca una situación de equilibrio donde esta función sea mínima. Para poder aplicar la red de *Hopfield* para resolver un determinado problema se debe buscar una función de energía objetivo que estará directamente relacionada con el objetivo buscado. En el caso del encaminamiento, el objetivo es encontrar el camino de coste mínimo. La principal ventaja de la aplicación de la red de *Hopfield* es su posibilidad de una implementación *hardware*, lo que facilita un masivo paralelismo, y, por lo tanto, una muy alta velocidad de cómputo. Sin embargo, presenta como principal inconveniente que la función de energía posee un alto número de mínimos locales, lo que implica que exista una alta probabilidad de que la red converja hacia soluciones no óptimas. A fin de reducir la probabilidad de que la red neuronal caiga en un mínimo local es posible emplear la técnica del *mean-field annealing* (*MFA*) [CU92] según la cual, la función de transferencia de la neurona se modifica de forma dinámica en cada paso de computación. La aplicación de esta técnica permite encontrar la solución óptima de la red con una mayor probabilidad. Pero la aplicación del *MFA* presenta, a su vez, un importante inconveniente, y es la dificultad de su implementación *hardware*, mientras que la implementación *software* del algoritmo precisa de un elevado tiempo de cómputo. Por otro lado existen algoritmos convencionales de encaminamiento [Bel57][Dij59], de los

que además se pueden encontrar implementaciones computacionales cada vez más eficientes [GS95] [CGR96] [CGS96], que permiten obtener la solución óptima en un menor tiempo de cómputo que la implementación *software* de la red de *Hopfield* con *MFA*, con el agravante de que esta última no siempre ofrece dicha solución óptima.

En este ámbito, existen otras propuestas de uso de redes neuronales. Entre ellas, es especialmente interesante la red *JEB* (*Jensen, Eshera and Barash*) [WW95] para la aplicación en entornos distribuidos basados en vector de distancia. No obstante, los algoritmos basados en estos vectores presentan una serie de inconvenientes, sobre todo si se pretende que éstos soporten calidad de servicio. En consecuencia, no resulta fácil aplicar la red neuronal *JEB* a algunas de las implementaciones actuales basadas en vector de distancia como el algoritmo *BGP* [RL95]. Por otro lado, ninguna de las aplicaciones de las redes neuronales enfocadas a la búsqueda del camino de coste mínimo, propuestas hasta ahora, permite resolver el problema del camino mínimo sujeto a múltiples restricciones, siendo la resolución de este problema indispensable, como se verá en el siguiente capítulo, para poder garantizar que se cumplen múltiples requisitos de calidad de servicio extremo a extremo.

2.4.2 El perceptrón multicapa

Existe otra alternativa para la aplicación de RNA (Red Neuronal Artificial) con el fin de resolver el problema del encaminamiento. En lugar de intentar resolver el problema del encaminamiento directamente, se propone utilizar los algoritmos neuronales para estimar las funciones de coste [ALGS96][ALGS98]. Por ejemplo, es posible emplear redes neuronales, como el perceptrón multicapa, para obtener la función de probabilidad de bloqueo de llamada en un sistema con múltiples clases de llamada y con distribuciones arbitrarias tanto para la duración de las llamadas como del tiempo entre llamadas. También es posible utilizar una red neuronal para el cálculo de la función de coste, sobre todo en los casos donde el cálculo de ésta no sea posible por métodos analíticos o el coste computacional sea demasiado alto para ser posible su uso práctico.

El perceptrón multicapa es un tipo de red neuronal *feedforward*, donde no existe realimentación hacia atrás sino que la información fluye, únicamente, hacia adelante, a través de una serie de capas formadas por neuronas. Todas las neuronas de una capa están conectadas a todas las neuronas de las capas adyacentes mediante enlaces unidireccionales. Estos enlaces están representados por los pesos sinápticos que actúan como

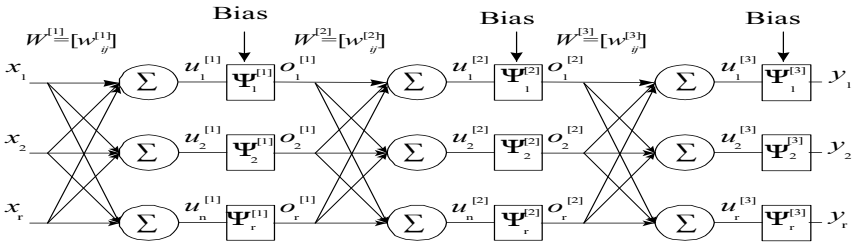


Figura 2.6: Esquema de un perceptrón con tres capas de neuronas.

multiplicadores en sus correspondientes enlaces. En la figura 2.6 se muestra un perceptrón de 3 capas, en el que se observa a las neuronas agrupadas en capas secuencialmente conectadas. A cada capa se le asigna un número identificativo (1, 2 o 3 para un perceptrón de tres capas). La capa 3 es la capa de salida y proporciona la respuesta de la red. Las capas de neuronas existentes entre las capas de entrada y salida se denominan capas ocultas. No existe límite teórico al número de capas ocultas que un perceptrón puede poseer, sin embargo, normalmente, los perceptrones poseen, a lo sumo, dos capas de neuronas ocultas. Está demostrado que para resolver una función de complejidad arbitraria es suficiente con que el perceptrón posea dos capas de neuronas ocultas [RE86]. Como se ha comentado, cada neurona está conectada a todas las neuronas de las capas adyacentes, no existiendo conexión entre las neuronas de la misma capa, de modo que la información únicamente fluye hacia adelante.

Definimos como potencial interno de la neurona *j-ésima* de la capa s a la expresión $u_j^{[s]}$, la cual se corresponde al valor que toma internamente la neurona para posteriormente utilizarlo con la función de activación. El valor del potencial interno se obtiene aplicado la siguiente expresión:

$$u_j^{[s]} \;=\; \sum_{i=0}^{r_{s-1}} w_{ij}^s o_i^{[s-1]} \qquad (s = 1, 2, 3; j = 1, 2, \ldots, r_s) \tag{2.25}$$

donde w_{ij}^s es el valor del peso sináptico que une la neurona j de la capa s con la neurona i de la capa $s-1$ y r_{s-1} es el número de neuronas de la capa $s-1$. La salida de la neurona j de la capa s es computada pasando el valor $u_j^{[s]}$ a través de la función de activación no lineal $\psi_j^{[s]}$:

$$o_j^{[s]} \;=\; \psi_j^{[s]}(u_j^{[s]}) \;=\; \psi_j^{[s]}\left(\sum_{i=0}^{n_s} w_{ij}^s o_i^{[s-1]}\right) \tag{2.26}$$

Una de las funciones de activación más utilizadas es sigmoide. Según esta función de activación el valor de la salida de la neuronal $o_j^{[s]}$ resulta ser:

$$o_j^{[s]} = \frac{1}{1 + e^{u_j^{[s]}}} \tag{2.27}$$

Además de la función sigmoide, otro tipo de función de activación que se ha empleado en este trabajo ha sido la función lineal, utilizadas en las neuronas de la capa de salida. Según esta función, el valor de salida de la neurona $o_j^{[s]}$ valdrá:

$$o_j^{[s]} = u_j^{[s]} \tag{2.28}$$

El aprendizaje de un perceptrón queda definido por la determinación de los pesos. Este tipo de red neuronal emplea, para la obtención de los pesos sinápticos adecuados, un aprendizaje supervisado. En esta clase de aprendizaje se proporciona a la red un conjunto de patrones de aprendizaje de la función que se desea aprender formados por unos valores de entrada y el valor de salida correspondiente a dicha entrada. De esta manera es posible calcular la diferencia entre el valor de salida que debería proporcionar la red neuronal y el valor que realmente se obtiene. En función de esta diferencia se modifican los pesos sinápticos con el objetivo de que la diferencia converja a cero. El algoritmo de minimización de este error normalmente empleado se denomina *backpropagation* o también regla delta generalizada. Si bien el perceptrón multicapa se conocía hacía tiempo, éste no era muy empleado debido a que se desconocía cómo modificar los pesos de las capas ocultas. La aparición del algoritmo *backpropagation* [Par82] [Wer90] permitió su utilización para resolver cualquier problema de aproximación de funciones. Desde entonces este tipo de red neuronal ha gozado de una amplia popularidad. A continuación, vamos a exponer el sistema empleado para la estimación de la probabilidad de bloqueo mediante el perceptrón multicapa.

2.4.3 Estimación de la probabilidad de bloqueo mediante técnicas adaptativas

Como hemos comentado anteriormente, es posible emplear una red neuronal para estimar la probabilidad de bloqueo de llamada en un enlace en un sistema con múltiples clases de llamada. Las ventajas de emplear un perceptrón para esta labor son básicamente tres:

-*Velocidad.* La arquitectura masivamente paralela permite obtener el resultado con una mayor velocidad que los métodos recursivos.

-*Independencia del tipo de tráfico.* El problema de los métodos recursivos o de las aproximaciones es la suposición de que tanto el tiempo entre llamadas como la duración de éstas siguen una distribución exponencial. La red neuronal, puesto que aprende a través del comportamiento del sistema, no requiere hacer estas aproximaciones.

-*Independencia del tipo de control de admisión.* Un problema a la hora de estimar la probabilidad de pérdida mediante métodos analíticos es la política aplicada en el control de admisión, ya que si existe ganancia estadística y reserva de recursos (*Trunk Reservation*) la obtención de la probabilidad de bloqueo se complica enormemente, debido a que el método recursivo presentado anteriormente deja de ser válido y las aproximaciones presentan un elevado error numérico, pudiendo llegar a ser imposible su estimación o, en última instancia, la obtención de la función de coste del enlace cuando ésta depende, a su vez, de la probabilidad de bloqueo.

Frente a este conjunto de ventajas es necesario indicar que el perceptrón no carece de problemas, siendo el mayor de ellos la dificultad de obtener los patrones de aprendizaje que representen todo el espacio de estados o, al menos, la zona de trabajo habitual de la red.

Es posible emplear otros tipos de redes neuronales como las funciones de base radial [CU92]. Estas presentan un problema que desaconseja su uso para resolver el problema que nos ocupa: su carencia de capacidad extrapoladora (a diferencia del perceptrón), lo que ofrecería un comportamiento muy degradado del sistema cuando éste se encuentre en una zona del espacio de estado para el que no se ha proporcionado patrones de aprendizaje a la red neuronal. Esta circunstancia aconseja emplear el perceptrón. La arquitectura utilizada está basada en el esquema *OCON* (*One Class in One Network*, Una Clase en Una Red) representado en la figura 2.7, frente al esquema *ACON* (*All Classes in One Network*, Todas las Clases en Una Red), incluido en la figura 2.8. Como se deduce de la figura 2.7, el esquema *OCON* emplea un perceptrón para cada clase de tráfico, frente a la posibilidad de emplear una única red para la obtención de todos los valores.

La decisión de emplear el modelo *OCON* se debe a que el aprendizaje de cada una de las redes es independiente de las demás, pudiéndose interrumpir el entrenamiento de cada red individualmente en el momento que esta alcanza el error deseado, con lo que

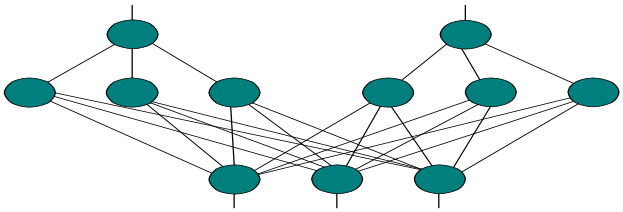


Figura 2.7: Esquema neuronal OCON.

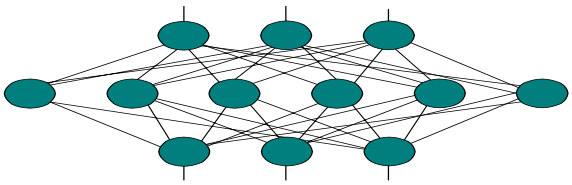


Figura 2.8: Esquema neuronal ACON.

se acelera el tiempo necesario para el aprendizaje. Con el esquema *ACON* es necesario seguir entrenando hasta que todas las salidas alcanzan dicho error. Esta circunstancia, además de alargar el tiempo necesario para el entrenamiento de la red, puede dar lugar a situaciones de sobrentrenamiento o *overtraining*, con lo que se degrada el propio ajuste del sistema. Adicionalmente, el esquema *OCON* posee una tasa de interconexión de las neuronas menor, circunstancia que, además de acelerar el aprendizaje, permite una mejor generalización y, por lo tanto, una mejor extrapolación e interpolación de la función a imitar.

El perceptrón empleado presenta tres capas de neuronas, con lo que se garantiza la posibilidad del aprendizaje dado que la función de probabilidad de pérdida de llamada es continua. El esquema empleado se ha representado en la figura 2.9, particularizado para dos clases de tráfico.

Aunque la técnica básica de aprendizaje del algoritmo de entrenamiento *backpropagation* se basa en el gradiente descendente, existen diversas propuestas alternativas [CU92] a fin de acelerar la velocidad de aprendizaje. Entre estas técnicas destacan las que están basadas en el método de Newton. A fin de conseguir una convergencia más rápida hemos empleado un aprendizaje aplicando técnicas de cuasi-Newton [CU92] que

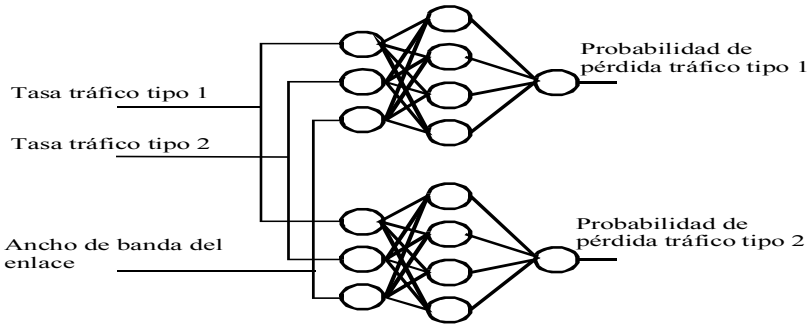


Figura 2.9: Esquema neuronal empleado para la obtención de la probabilidad de pérdida de llamada con dos clases de tráfico.

reducen notablemente el tiempo de aprendizaje de la red.

2.4.3.1 Pruebas

El cálculo de la probabilidad de bloqueo depende directamente de los controles de admisión utilizados. Si definimos como ancho de banda efectivo a la cantidad mínima de ancho de banda que se debe de asignar a una determinada conexión para satisfacer sus requerimiento de *QoS*, cada *CAC* determina de forma distinta este ancho de banda efectivo, en función de los criterios de ganancia estadística que dicho control aplique. Esta circunstancia dará lugar a que dos controles de admisión distintos sobre un mismo enlace y con las mismas clases de tráfico asignen anchos de banda efectivos distintos y, por lo tanto, permitan un número distinto de conexiones simultáneas.

En este trabajo, a fin de probar la versatilidad de la red neuronal, se han utilizado tres controles de admisión distintos sobre un mismo modelo neuronal, dos controles de admisión que admiten ganancia estadística y un control de admisión sin ganancia estadística.

Los controles de admisión que hemos utilizado han sido: el control de admisión propuesto por E. Wallmeier y C. M. Hauber o regla sigma [WH91], el propuesto por T. Murase *et al.* [MSST91] y el control de admisión basado en la velocidad de pico.

La regla sigma divide la región de admisión en dos subregiones. La primera subregión (cuyas conexiones enumeraremos con los subíndices $\{1 \ldots f\}$), sí admite multiplexación estadística, en tanto que la segunda no admite dicha multiplexación, reservándose para

conexiones críticas a las que se les asigna los subíndices $\{f+1 \ldots x\}$, siendo x el número total de conexiones presentes en el enlace.

Así, se admite una nueva solicitud de conexión si, incluyendo a la nueva conexión, se cumple la condición:

$$\sum_{i=1}^{x} MAX(i) \leq C \tag{2.29}$$

donde $MAX(i)$ es la velocidad de pico de la conexión i. En caso de que no se cumpla dicha condición, la nueva conexión es aceptada sólo si se cumplen las dos siguientes condiciones.

$$\begin{aligned} c &= C - \sum_{i=f+1}^{x} MAX(i) \geq C_{II} \\ g(c) &* \left[\sum_{i=1}^{f} \sigma^2(i) \right]^{1/2} + \sum_{i=1}^{f} AVG(i) + \max(MAX(i)) \leq c \end{aligned} \tag{2.30}$$

donde C_{II} es el ancho de banda mínimo necesario para poder multiplexar estadísticamente, $AVG(i)$ es el ancho de banda medio precisado por la la conexión i, $\sigma^2(i)$ es la varianza de la velocidad binaria de la conexión i y $g(c)$ es una constante, que depende de c, calculada de forma empírica. Para estimar esta constante, Wallmeier utiliza el $\alpha - cuantil$ de la distribución normal ya que éste actúa como límite inferior del valor de $g(c)$. En caso de no disponer de un ancho de banda suficiente para poder multiplexar estadísticamente se aceptarán llamadas sólo si se cumple la condición (2.29). Este control de admisión presenta como desventajas la dificultad de estimar la constante $g(c)$, ya que ésta posee una fuerte dependencia no lineal con el ancho de banda, así como la suposición de que todas las fuentes presentan los mismos requerimientos de *QoS*. En las pruebas realizadas se ha utilizado un valor de $g(c)$ dado por la expresión[WH91]:

$$q(c) = 8 + \frac{40}{c} \tag{2.31}$$

El método propuesto por Murase *et al.* [MSST91] calcula la probabilidad de pérdida de paquete para una determinada combinación de fuentes *ON-OFF*. La condición de aceptación de una conexión de la clase k es:

$$q_k^{[virtual]} = \frac{l_k^{[virtual]}}{\gamma_k} < q_k^{[max]} \tag{2.32}$$

donde $q_k^{[virtual]}$ es la tasa de pérdidas virtual, $q_k^{[max]}$ es la tasa de pérdidas máxima de paquete permitida para la clase k y $l_k^{[virtual]}$ el exceso de tráfico definido como:

$$l_k^{[virtual]} = \sum_{a_1 \in \left(\sum_{i=1}^{K} a_i MAX_i - C\right) \geq 0}^{a_1 = x_1} \cdots \sum_{a_K \in \left(\sum_{i=1}^{K} a_i MAX_i - C\right) \geq 0}^{a_K = x_K} \left[\prod_{i=1}^{K} p_i(a_i) \frac{\left(\left(\sum_{j=1}^{K} a_j MAX_j \right) - C \right) MAX_k x_k}{\sum_{j=1}^{K} a_j MAX_j} \right] \tag{2.33}$$

donde a_k es el número de fuentes activas (que están enviado paquetes), de las x_k de la clase k presentes en el enlace. La probabilidad $p_k(a_k)$ de que a_k fuentes de clase k estén activas viene definida por la relación binomial:

$$p_k(a_k) = \begin{pmatrix} x_k \\ a_k \end{pmatrix} \left(\frac{AVG_k}{MAX_k} \right)^{a_k} \left(1 - \frac{AVG_k}{MAX_k} \right)^{x_k - a_k} \tag{2.34}$$

Por último, en la expresión (2.32), la carga de tráfico γ_k se define como:

$$\gamma_k = x_k AVG_k \tag{2.35}$$

donde x_k es el número máximo de fuentes de la clase k que podemos tener activas. En nuestros experimentos se ha fijado la probabilidad de pérdida de paquete objetivo en 10^{-9} para todos los tipos de conexión.

El último tipo de control de admisión utilizado es el control de admisión basado en la velocidad de pico. De acuerdo con este control, una nueva conexión de clase i es admitida si y sólo si se cumple la condición:

$$\sum_{k=1}^{K} x_k \; MAX_k + MAX_i \; \leq \; C \tag{2.36}$$

Para poder verificar las prestaciones del método se han utilizado, con una misma RNA, distintos tipos de tráfico y tipos de control de admisión, obteniéndose los patrones de aprendizaje mediante la función de *Erlang-B* generalizada. Para el control de admisión basado en la regla sigma se ha utilizado un ancho de banda constante de 150 Mbit/s [WH91] con tres clases de tráfico distintas y cuyas características se han tabulado en la tabla 2.1. Se ha supuesto que las conexiones son de tipo *ON-OFF* con lo que la varianza de la conexión *i-ésima* viene dada por la expresión [WH91]:

$$\sigma^2(i) = AVG(i)(MAX(i) - AVG(i)) \tag{2.37}$$

Clase	Pico (MAX)	Media (AVG)	Tipo
1	64 Kbit/s	64 Kbit/s	Alta prioridad
2	2,048 Mbit/s	1,024 Mbit/s	Baja prioridad
3	10 Mbit/s	2,048 Mbit/s	Baja prioridad

Tabla 2.1: Características del tráfico empleado en las pruebas con el *CAC* regla sigma.

Clase	Pico (MAX)	Media (AVG)
1	6,6 Mbit/s	0,66 Mbit/s
2	10 Mbit/s	1 Mbit/s

Tabla 2.2: Características del tráfico empleado en las pruebas con el *CAC* propuesto por Murase *et al.* [MSST91] y el *CAC* de velocidad de pico.

Para el control de admisión basado en la velocidad de pico y el control propuesto por Murase *et al.* [MSST91] se ha utilizado un ancho de banda variable con dos clases de tráfico cuyas características podemos ver en la tabla 2.2.

En las figuras que a continuación se comentan se representa la probabilidad de bloqueo de los distintos controles de admisión en función del tráfico ofrecido a la red. La medida del tráfico empleada en las figuras se ofrece en Erlang Mb/s [WH91], que se define como:

$$\sum_{k=1}^{K} \rho_k \; MAX_k \tag{2.38}$$

donde ρ_k es el tráfico total ofrecido por la clase k y MAX_k la velocidad de pico de la clase k.

Debido a que estamos trabajando en un espacio multidimensional, es necesario realizar proyecciones sobre los planos a fin de reducir el número de dimensiones y poder representar de forma gráfica los resultados de una manera comprensible. Para realizar estas proyecciones se fijó el ancho de banda del enlace y la proporción entre las distintas clases de tráfico, obteniéndose, en la proyección, la probabilidad de pérdida de llamada en función del tráfico total ofrecido. De esta forma, se ha generado una proyección del sistema sobre un espacio bidimensional fácilmente representable y donde es más sencillo interpretar los resultados obtenidos. Las figuras 2.10, 2.11, 2.12 y 2.13 representan la proyección sobre dos planos distintos, tanto del tráfico de la clase 1 como del tráfico de la clase 2, y su comparación con la probabilidad de pérdida obtenida aplicando el cálculo analítico ofrecido por la expresión (2.7). En concreto, las figuras 2.10 y 2.11 representan la proyección sobre un plano con un ancho de banda de 87 Mbit/s y con una combinación

de tráfico del 40% para el tráfico de la clase 1 y del 60% para el tráfico de la clase 2. Se ha escogido un ancho de banda de 87 Mbit/s para comprobar la capacidad interpoladora de la RNA, ya que no se han proporcionado patrones de aprendizaje a la RNA con dicho ancho de banda. Las figuras 2.12 y 2.13 representan la proyección sobre el plano con combinaciones de tráfico de 60% para el tráfico de la clase 1 y del 40% para el tráfico de la clase 2.

A continuación presentamos los resultados obtenidos al aplicar el CAC basado en la regla sigma. Este control de admisión presenta problemas cuando se trabaja con enlaces de capacidad variable, por lo que se han limitado las pruebas a un ancho de banda fijo en el enlace. En la figura 2.14 se muestran los resultados obtenidos cuando se aplica el CAC de la regla sigma en el cálculo de la probabilidad de bloqueo de llamada, tanto por medios analíticos como por técnicas neuronales, para el tráfico de alta prioridad (tráfico de velocidad constante), con una combinación de tráfico de 60% para la clase 1, 30% para la clase 2 y 10% para la clase 3.

Como se puede apreciar en la figura 2.14, la red neuronal es capaz de estimar con un alto grado de precisión la función de probabilidad de pérdida de llamada.

En el siguiente experimento se estudiará la capacidad de una red neuronal para aprender la función de coste del algoritmo $EASDR$ (propuesto anteriormente). En este caso, a fin de simplificar el sistema y sin perder generalidad, utilizaremos un control de admisión de velocidad de pico. Las características de los tráficos empleados para esta prueba están indicadas en la tabla 2.2, siendo la combinación de tráfico utilizada de un 40% del tráfico total para el tráfico de la clase 1 y de un 60% para el tráfico de la clase 2, con un tráfico total de 20 *Erlangs*. Los resultados de este experimento se muestran en la figura 2.15, donde se presentan los costes obtenidos mediante la expresión (2.14) y los conseguidos por medios neuronales. En esta prueba, el cálculo neuronal se ha realizado de dos formas distintas: por un lado se ha empleado una red neuronal con una entrada para el tráfico ofrecido al enlace para cada clase de tráfico, más dos entradas adicionales, una para el ancho de banda del enlace y la otra para el ancho de banda ocupado en el enlace, de forma que la red neuronal obtiene directamente el resultado de la expresión (2.14). La otra forma de obtener la función de coste se ha llevado a cabo estimando de forma separada el valor del numerador y del denominador de (2.14), realizándose a continuación la división. Para operar de esta forma se pueden emplear dos redes neuronales idénticas, o bien una única red donde primero se obtiene el numerador y a continuación se obtiene el denominador. Los resultados de este experimento se ilustran

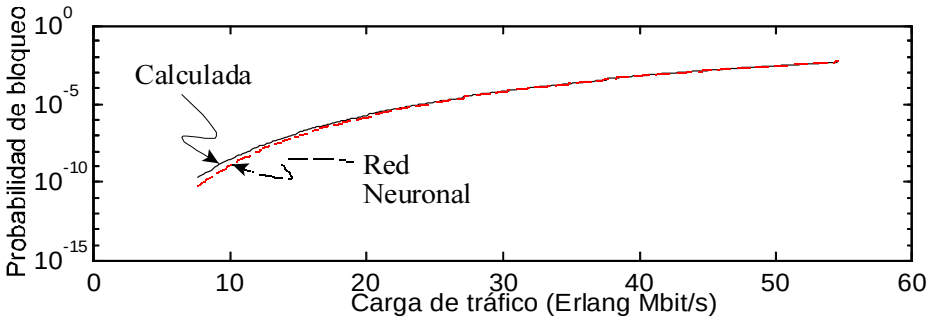


Figura 2.10: Probabilidad de bloqueo para el tráfico de la clase 1. Combinación de tráfico: 40% tráfico de la clase 1, 60% tráfico de la clase 2.

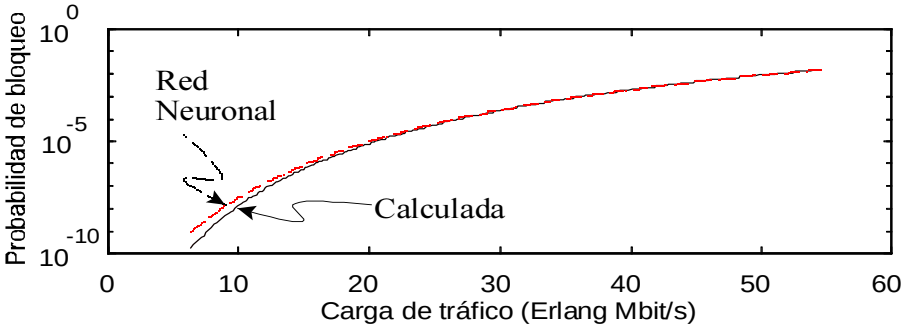


Figura 2.11: Probabilidad de bloqueo para el tráfico de la clase 2. Combinación de tráfico: 40% tráfico de la clase 1, 60% tráfico de la clase 2.

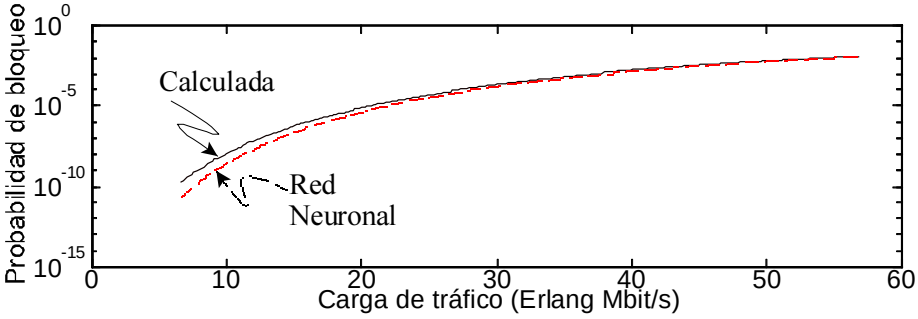


Figura 2.12: Probabilidad de bloqueo para el tráfico de la clase 1. Combinación de tráfico: 60% tráfico de la clase 1, 40% tráfico de la clase 2.

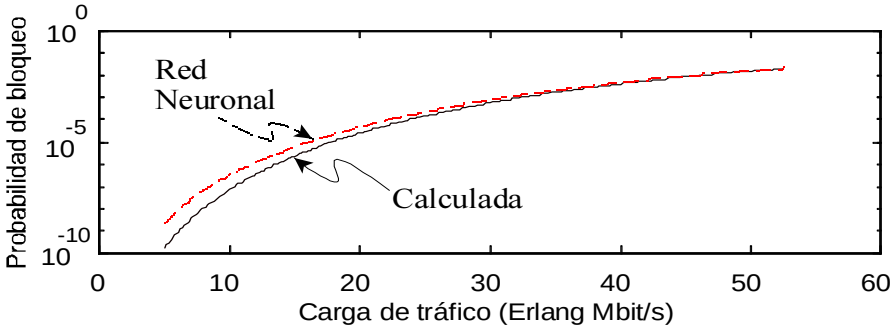


Figura 2.13: Probabilidad de bloqueo para el tráfico de la clase 2. Combinación de tráfico: 60% tráfico de la clase 1, 40% tráfico de la clase 2.

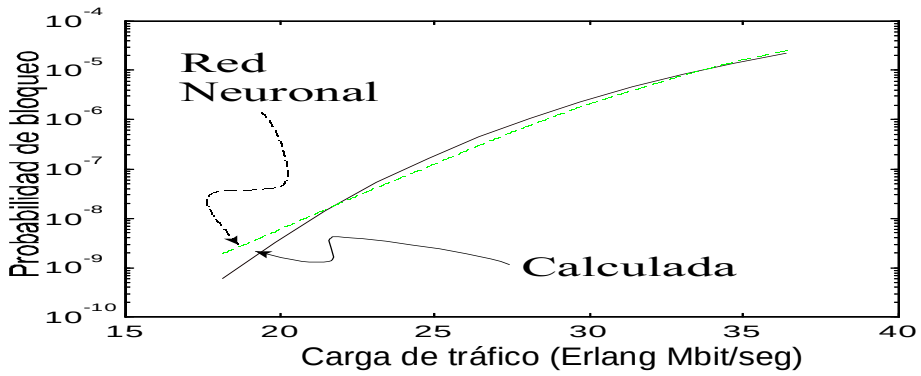


Figura 2.14: Probabilidad del bloqueo del tráfico de la clase 1 con control de admisión de regla sigma.

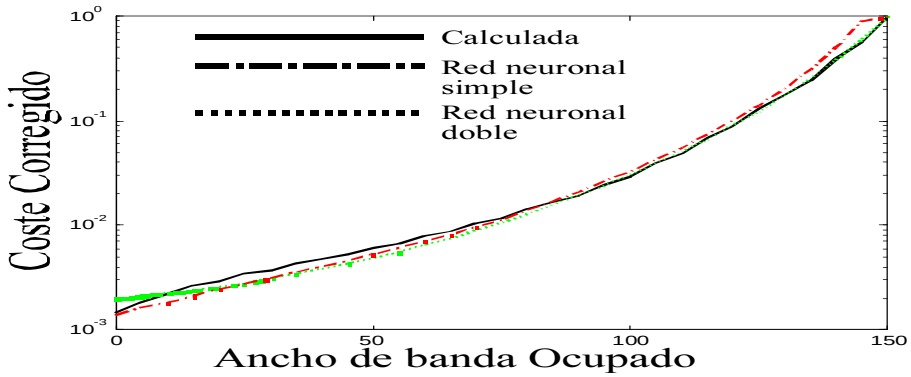


Figura 2.15: Comparación del valor de la función de coste obtenida mediante métodos analíticos y técnicas neuronales.

en la figura 2.15, donde es posible comparar el resultado obtenido por ambos métodos neuronales y mediante el método analítico para el tráfico de clase 1. Como es posible apreciar, la red neuronal es capaz de aproximar con precisión tanto las probabilidades de pérdida de llamada como la función de coste del algoritmo de encaminamiento *EASDR*, propuesto anteriormente para redes multiservicio.

2.5 Comportamiento del algoritmo EASDR

2.5.1 Introducción

Como se ha podido comprobar en el apartado anterior, las técnicas neuronales permiten estimar la probabilidad de pérdida de llamada y la función de coste del algoritmo

EASDR. A continuación vamos a estudiar la posible validez de este algoritmo para el encaminamiento de llamadas. Para esto es necesario crear un entorno de pruebas y comparar su rendimiento con alguno de los algoritmos propuestos en la literatura para el encaminamiento en redes multiservicio.

Para la realización de las pruebas se ha desarrollado un entorno de simulación orientado a llamadas en **OPNET**^{TM}. Es decir, sobre una herramienta para la simulación de eventos discretos se ha desarrollado un entorno de pruebas donde sólo se generan llamadas o, dicho de otra manera, sólo se generan los eventos de inicio y fin de llamada obviándose los niveles inferiores (ráfagas, cálculo de paquetes, etc.). El simulador cuenta con un módulo gestor que recoge la información de los eventos de generación y fin de llamada. Esta información es empleada por el sistema gestor para determinar la ocupación de los recursos de red. La decisión de emplear un simulador orientado a llamada se debe a la necesidad de poder simular largos intervalos de tiempo, que han de ser varios órdenes de magnitud superiores a la duración media de las llamadas, a fin de que los resultados sean significativos. Si en lugar de realizar una simulación orientada a llamada se hubiera realizado una simulación orientada a celda, habría sido imposible realizar una simulación realista del comportamiento de la red, puesto que cada llamada puede generar varios millones de paquetes y, a su vez, durante la simulación, se producen en la red varios cientos de miles de llamadas, lo que implicaría que el tiempo necesario para completar una simulación con resultados fiables sería excesivo, del orden de meses o años en los ordenadores actuales.

Con el propósito de forzar una carga no homogénea en la red se ha empleado una matriz de tráfico. El objetivo perseguido al emplear esta matriz es que los enlaces de la red no presenten una carga homogénea y, de esta manera, poder analizar el rendimiento de los diversos algoritmos al distribuir el tráfico entre los distintos enlaces de la red, sobre todo en situaciones de sobrecarga.

La red empleada en las pruebas, que se muestra en la figura 2.16, es la topología de red que emplea Gawlick [Gaw95] en su tesis y que corresponde, según este autor, a una red real de *AT&T*. Para comparar el rendimiento se han empleado los algoritmos *EXP-MC* (exponencial), *EXP* (exponencial simple) [Gaw95], *LLR* y el clásico *MIN-HOP* (menor número de enlaces en el camino) frente al algoritmo propuesto, *EASDR*, tanto en su versión con cálculo analítico como con cálculo neuronal.

Los algoritmos *EXP-MC* y *EXP* [Gaw95] están basados en una función de coste

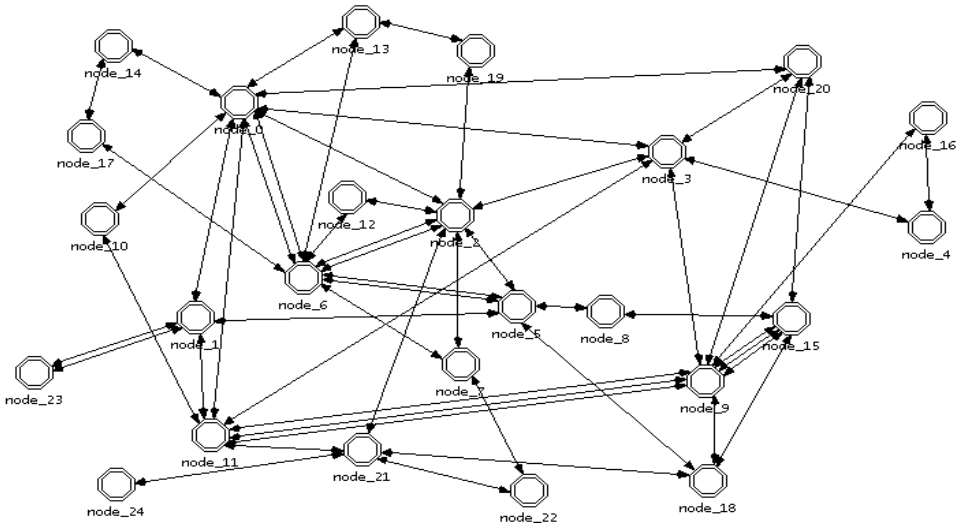


Figura 2.16: Red de pruebas.

exponencial. Plotkin demostró [Plo95] que es posible obtener un algoritmo de encaminamiento óptimo basado en funciones de coste exponenciales. Sin embargo, este tipo de algoritmos precisan de una serie de constantes, no existiendo métodos que permitan obtener analíticamente el valor de dichas constates, por lo que han de ajustarse de forma empírica. El número de constantes que presenta el algoritmo exponencial básico lo hace poco útil para su aplicación en redes reales. A fin de poder aplicar los algoritmos de tipo exponencial en sistemas reales, Gawlick [Gaw95] expone dos simplificaciones del algoritmo exponencial básico. Estas simplificaciones, denominadas *EXP* y *EXP-MC*, ofrecen dos algoritmos sencillos de implementar y con un buen rendimiento.

El esquema del algoritmo *EXP* se incluye en la la figura 2.17 y, como podemos apreciar, consta de dos fases: en la primera fase busca el camino con menor número de saltos y asigna un coste a cada enlace del camino. En la segunda fase comprueba que el coste total del camino no supere un determinado valor y que haya suficiente ancho de banda para encaminar la llamada k que requiere un ancho de banda b_k.

El algoritmo *EXP-MC*, expuesto en la figura 2.18, es similar al algoritmo *EXP* con la salvedad de que ya no escoge el camino con menor número de saltos, sino el camino de coste mínimo. El problema que presenta este algoritmo es la elección del valor de ϑ (base utilizada en la función exponencial que define el coste). El valor de esta constante depende de las características del tráfico que se pretende cursar en la red y se ha de fijar empíricamente. En este sentido, en [KZA+97] se comprueba la variación del

buscar camino P de menor número de saltos entre nodos origen - destino

si $\sum_{e_{ij} \in P} \vartheta^{u(e_{ij})} \leq \vartheta \quad y \quad u(e_{ij}) + \frac{b_k}{C(e_{ij})} < 1$
entonces
    encaminar la llamada por el camino P
    $\forall e_{ij} \in P \; u(e_{ij}) = u(e_{ij}) + \frac{b_k}{C(e_{ij})}$
si no
    rechazar la llamada
donde $u(e_{ij})$ es la fracción de ancho de banda ocupado, definido como $u(e_{ij}) = \frac{c_{ij}}{C_{ij}}$ y ϑ es una constante que se obtiene empíricamente.

Figura 2.17: Algoritmo *EXP* (exponencial simple).

$\forall e_{ij} \in E \quad coste_{ij} = \sum_{e_{ij} \in P} \vartheta^{u(e_{ij})}$
buscar camino P de coste mínimo entre nodos origen - destino

si $\sum_{e_{ij} \in P} \vartheta^{u(e_{ij})} \leq \vartheta \quad y \quad u(e_{ij}) + \frac{b_k}{C(e_{ij})} < 1$
entonces
    encaminar la llamada por el camino P
    $\forall e_{ij} \in P \; u(e_{ij}) = u(e_{ij}) + \frac{b_k}{C(e_{ij})}$
si no
    rechazar la llamada

Figura 2.18: Algoritmo *EXP-MC* (Exponencial con camino de coste mínimo).

comportamiento del algoritmo *EXP* en función del valor del parámetro ϑ.

Para las primeras pruebas se ha establecido el ancho de banda de todos los enlaces de la red a 140 Mbit/s y se ha empleado un único tipo de tráfico con un ancho de banda de 1 Mbit/s y una duración media de llamada de 1800 segundos. El objetivo de esta prueba es comparar el rendimiento de los distintos algoritmos y estudiar qué efecto tiene sobre el algoritmo *EASDR* el intervalo de tiempo escogido para la medida del tráfico ($\lambda_{k,ij}$) ofrecido a los enlaces.

Como se puede apreciar en la figura 2.19 (parcialmente ampliada en la figura 2.20), donde se presenta el comportamiento del algoritmo EASDR con intervalos de actualización de tráfico de 60, 1800 y 3000 segundos, la elección del tamaño de este intervalo tiene una gran importancia en el comportamiento del sistema. Así, si el tiempo de observación

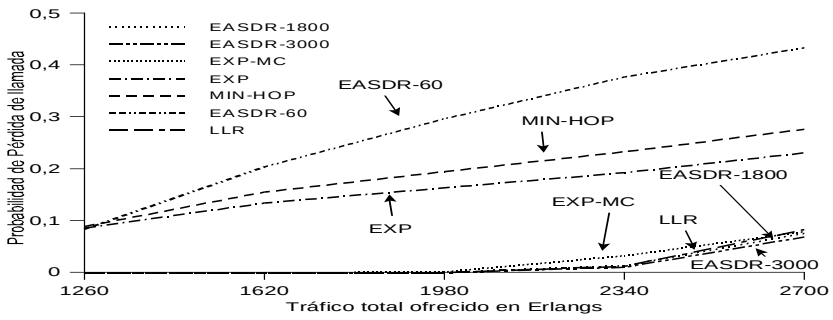


Figura 2.19: Comparación de los algoritmos *EXP-MC*, *LLR*, *MIN-HOP* y *EASDR* con distintos intervalos de actualización: 60, 1800 y 3000 segundos

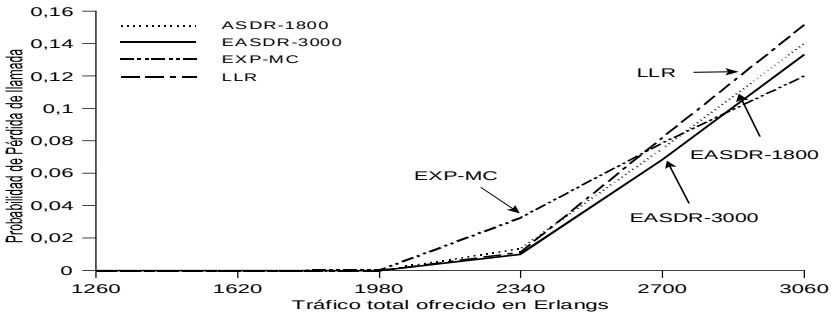


Figura 2.20: Comparación de los algoritmos *EXP-MC*, *LLR* y *EASDR* con intervalos de estimación de 1800 y 3000 segundos.

es demasiado pequeño (interpretando el tamaño en relación a la duración media de la llamada) puede ocurrir que, durante ese intervalo, todo el ancho de banda del enlace permanezca ocupado y ninguna llamada finalice durante el intervalo de observación. En este caso, dado que el tráfico ofrecido se calcula empleando la expresión (2.13) y el porcentaje de tiempo que en el intervalo de observación el enlace ha estado completamente ocupado ha sido del 100% ($\tilde{B}_{ij} = 1$), se asumiría que el tráfico ofrecido en dicho intervalo es infinito ($\lambda_{ij} = \infty$); esta situación se refleja claramente en la curva del método *EASDR* con un intervalo de 60 s. A fin de evitar esta situación, estos intervalos han de elegirse, al menos, con valores similares a la duración media de las llamadas.

En las sucesivas pruebas, y a efectos de claridad en las figuras, limitamos la comparativa a los algoritmos *EASDR* y *EXP-MC* ya que en las diversas pruebas realizadas son estos algoritmos los que mejor comportamiento presentan.

2.5.2 EASDR neuronal

En este apartado se analiza el comportamiento del algoritmo *EASDR* cuando se aplican técnicas neuronales, evaluando su rendimiento en comparación con el algoritmo *EXP-MC*.

Para estas pruebas se han empleado dos clases de tráfico, con una duración de 1800 segundos para ambas y con un ancho de banda de 1 Mb/s para el tráfico de la clase 1 y de 10 Mb/s para el tráfico de la clase 2.

Un problema que presentan los algoritmos neuronales radica en la obtención de patrones de entrenamiento que cubran con suficiencia todo el espacio de estado a caracterizar. Estos patrones han de ser representativos del espacio pero al mismo tiempo finitos en el mismo. El entrenamiento *on-line* obtiene los patrones cuando la red ya está en funcionamiento pero presenta el inconveniente de que se limita a los casos más frecuentes, careciendo de muestras de situaciones no habituales que, caso de acaecer, tienden a ser desplazadas rápidamente por las muestras que se obtienen con mayor frecuencia. El resultado de esta situación es que la red neuronal no identifica bien la función a imitar en esos rangos de valores infrecuentes. Para evitar este problema se emplea el método propuesto por A. Díaz *et al.* [EJS94] en el que el espacio de estados se divide en zonas, a cada una de las cuales se le asigna una tabla de patrones. De esta manera se logra conservar información de todo el espacio de estados, impidiendo que los patrones que pertenecen a las zonas más comunes desplacen a los de las zonas menos frecuentes.

Un segundo problema derivado de la aplicación de técnicas neuronales es la asignación de los valores iniciales a los pesos sinápticos de la red neuronal, por cuanto que estos valores iniciales influyen en el comportamiento final de ésta. Este efecto es posible apreciarlo en la figura 2.21, donde se han empleado dos redes con igual número de neuronas y conjunto de patrones de entrenamiento, permitiéndoles evolucionar hasta que el error total, con respecto a los patrones de entrenamiento, caiga por debajo de 10^{-5}, siendo la única diferencia entre estas redes el conjunto de pesos sinápticos iniciales. Como es posible observar, los valores sinápticos iniciales tienen un gran influencia en el comportamiento final del sistema. A fin de disminuir este efecto, se propone no limitar la red neuronal a un entrenamiento *on-line* exclusivamente. Así, se recomienda introducir una etapa previa con entrenamiento *off-line* que puede emplear patrones de entrenamiento obtenidos de forma analítica, para finalizar el entrenamiento de la red de forma *on-line* tomando los valores directamente del sistema. Añadidamente, la etapa

	2 clases	3 clases
Método analítico	2493,1427 s	2701,04384 s
Método neuronal	0,164835 s	0,210780 s
Relación	15125,0808	12814,51674

Tabla 2.3: Relación de tiempo entre el método neuronal y el método analítico.

de entrenamiento *off-line* evita el intervalo de tiempo inicial durante el cual el sistema ofrece un funcionamiento deficiente por falta de suficientes patrones de entrenamiento.

La siguiente cuestión a plantearse es la necesidad de introducir una red neuronal como método alternativo del cálculo de la probabilidad de bloqueo. La aplicación de la red neuronal tiene sentido cuando, o bien el tráfico existente en la red no tiene tratabilidad analítica (y por lo tanto no se podría calcular la función de coste analíticamente), o bien la potencia computacional disponible es escasa. La ventaja de emplear una RNA para resolver el problema del coste computacional se puede apreciar en la tabla 2.3, donde se han calculado 500 valores de coste escogiendo al azar el tráfico ofrecido y el ancho de banda ocupado para cada una de las clases, primero de forma analítica (sin emplear aproximaciones) y, posteriormente, estimándose de forma neuronal. La prueba se realizó en un PC con procesador Pentium a 200 MHz. Como es posible apreciar, la solución neuronal, a pesar de que la red está implementada mediante *software*, es mucho más rápida. Una implementación *hardware* de la red neuronal que aprovechara sus posibilidades de paralelismo masivo aumentaría de forma sustancial estas diferencias.

A continuación, se compara el funcionamiento del algoritmo *EASDR* con resolución neuronal con el ofrecido por el algoritmo *EXP-MC*. El resultado de este experimento se

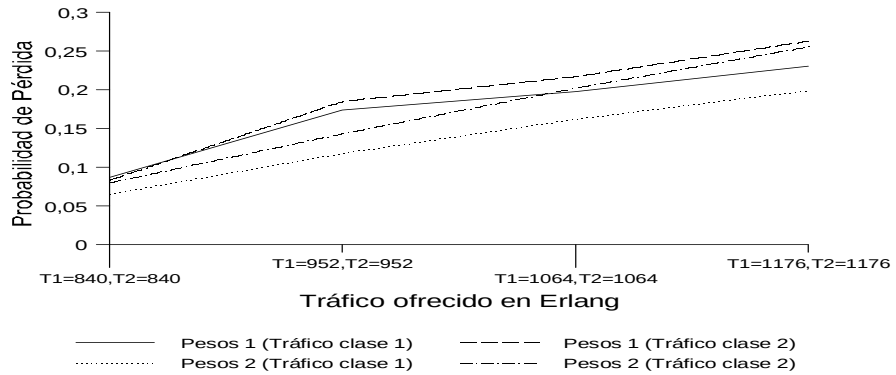


Figura 2.21: Efecto de los pesos iniciales en el resultado final del entrenamiento.

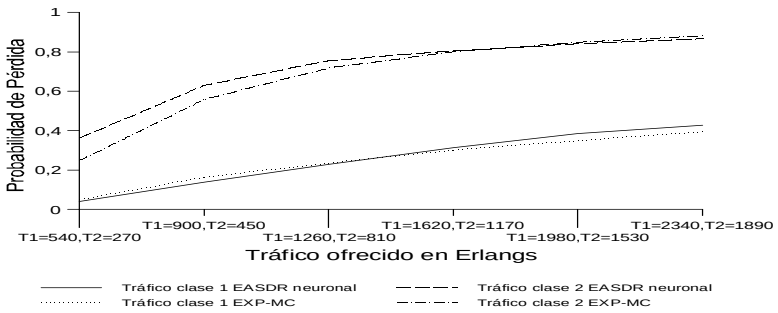


Figura 2.22: Comparación de rendimiento de los algoritmo *EASDR* y *EXP-MC*.

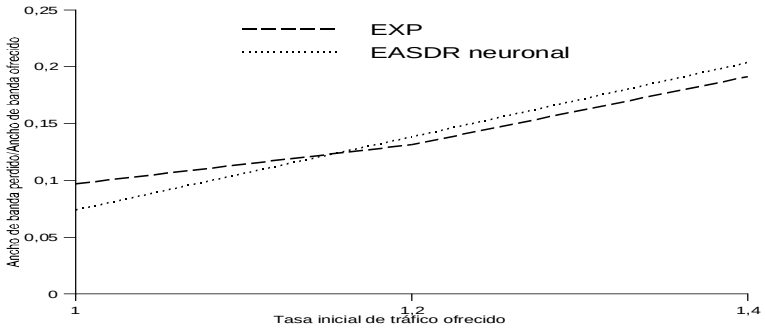


Figura 2.23: Comparación del rendimiento de los algoritmos *EASDR* y *EXP-MC* para tres clases de tráfico.

ha representado en la figura 2.22, probándose que ambos algoritmos ofrecen un comportamiento parecido.

A fin de extender estas pruebas se ha efectuado otro experimento. En este caso se han empleado tres clases de tráfico con ancho de banda 0,5, 2 y 4 Mbits/s, y una duración media de las llamadas de 500 segundos para las tres clases. Los porcentajes de tráfico relativos con respecto al tráfico de clase 1, han sido de un 50% para el tráfico de la clase 2 y un 25% para la clase 3. La figura 2.23 ilustra, para los dos algoritmos, la proporción total de ancho de banda encaminado con éxito con respecto al porcentaje total de ancho de banda ofrecido.

En la figura 2.23 se aprecia que, para cargas bajas y medias de tráfico, el algoritmo *EASDR* presenta un mejor comportamiento puesto que permite distribuir mejor el tráfico al tenerse en cuenta el flujo ofrecido a los enlaces en el momento de establecer un camino

y favoreciendo, de esta manera, que se empleen enlaces con bajas cargas de tráfico. Sin embargo, para altas cargas es más favorable el rendimiento del algoritmo *EXP-MC*. La razón de este fenómeno se podría encontrar en el inconveniente que supone la necesidad de conocer el tráfico ofrecido a los enlaces en un entorno altamente dinámico, como es una red en sobrecarga.

Una alternativa, que combina las distintas soluciones, pasaría por emplear una superposición de algoritmos. Así, para bajas cargas se pensaría en *LLR* por su sencillez, para carga medias en *EASDR*, mientras que para altas cargas se utilizaría el *EXP-MC*.

2.6 Estimación de carga de tráfico en condiciones de variabilidad periódica

2.6.1 Introducción

Como se ha comentado anteriormente, y es posible deducir de la expresión (2.13) y observar en la figura 2.19, la estimación del tráfico ofrecido depende de los intervalos de tiempo de observación, en los cuales se mide tanto el tráfico cursado por el enlace como el porcentaje de tiempo que el enlace está bloqueado para un determinada clase de tráfico. Se ha comprobado que cuanto mayores sean estos intervalos mejor tiende a ser la estimación. Sin embargo, emplear tamaños de estimación de tráfico demasiado grandes también conlleva inconvenientes. En una red real el tráfico ofrecido no sigue una distribución constante, sino que tiende hacia procesos estacionales (oscilaciones). Esto implica, especialmente si se emplean intervalos de estimación grandes, que cerca del final del intervalo las medidas que se tienen del tráfico puede estar obsoletas y no representar correctamente el comportamiento del tráfico actual, lo que llevará a tomar decisiones de encaminamiento incorrectas.

A fin de disminuir este problema en la estimación de carga de tráfico, proponemos dos métodos que permiten disminuir el efecto de la obsolescencia de los datos. El primero de ellos es el empleo de múltiples ventanas parcialmente solapadas en el tiempo y el segundo está basado en la predicción de carga de tráfico a corto plazo.

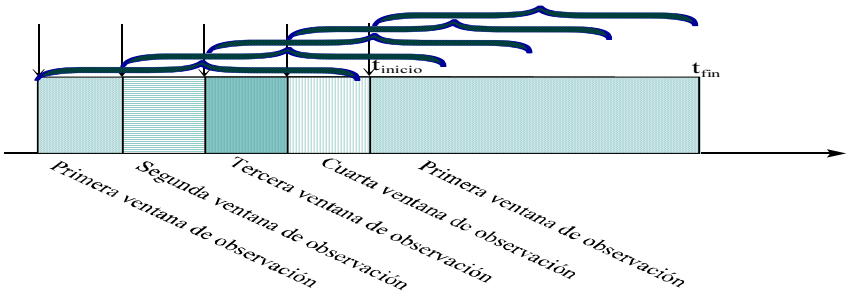


Figura 2.24: Sistema con cuatro ventanas solapadas en el tiempo.

2.6.2 Modelo de ventanas solapadas

El uso de las ventanas solapadas [AOS99] para la estimación de tráfico es una idea sencilla que permite realizar la discretización de una ventana deslizante aplicada a la estimación de tráfico. La principal desventaja de una ventana deslizante es la necesidad que lleva aparejada de almacenar una gran cantidad de información (de hecho su utilización implicaría guardar información acerca de todos los eventos de establecimiento/liberación de llamadas). Como alternativa, se propone aprovechar un sistema de ventanas parcialmente solapadas en el tiempo (véase la figura 2.24), donde se dispone de un número n de intervalos de estimación, cada uno de ellos desfasado con respecto al anterior $\frac{T}{n}$ segundos, donde T es la duración del intervalo de estimación en segundos. Este método permite emplear, en el cálculo del coste del enlace, medidas de tráfico no demasiado desfasadas en el tiempo. Así, para el cálculo del coste en el instante t_i se emplea la estimación realizada en la última ventana finalizada en t^k_{fin}, del modo:

$$t^k_{fin} = \max(t^1_{fin}, \cdots, t^n_{fin}) \;\; \forall t^j_{fin} \leq t_i \tag{2.39}$$

donde t^n_{fin} es el intervalo de tiempo en que finalizó la estimación de tráfico de la ventana enésima (nótese que con $n = \infty$ estaríamos en el caso de una deslizante). Resulta más interesante, desde el punto de vista de implementación, trabajar con un número de ventanas solapadas no demasiado grande, ya que acota la información a almacenar.

A fin de comprobar la bondad del método de las ventanas solapadas para la estimación de carga de tráfico y su posterior aplicación al encaminamiento, realizaremos una prueba con el algoritmo *ASDR* (con un único tipo de tráfico), donde el tráfico ofrecido a la red sigue una distribución de *Poisson* y una duración exponencialmente distribuida, de

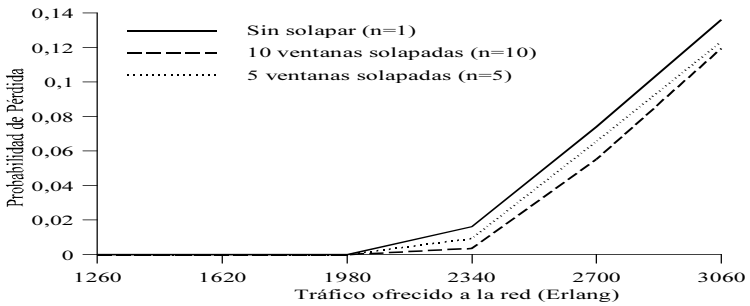


Figura 2.25: Comparación basada en la probabilidad de bloqueo de llamada para el algoritmo *ASDR* con distintos tipos de solapamiento en la ventana de estimación.

valor medio $1/\mu$ de 1.800 segundos, teniendo las llamadas un ancho de banda equivalente de 1 Mbits/s. Los enlaces, al igual que en las pruebas anteriores, se programaron con un ancho de banda de 140 Mbits/s, mientras que el tamaño del intervalo de estimación de tráfico se ha fijado en 3000 segundos. Se han realizado las pruebas con distinto número n de ventanas solapadas, constatándose que a partir de 10 ventanas la ganancia obtenida al aumentar n es demasiado pequeña para compensar la mayor necesidad de almacenamiento. Los resultados de los experimentos sin solapamiento y con solapamiento de 5 y 10 ventanas se muestran en la figura 2.25. Como es posible comprobar, el empleo de ventanas solapadas permite mejorar el rendimiento de aquellos algoritmos que precisan conocer la carga de tráfico por enlace.

2.6.3 Estimación de tráfico mediante predicción

Este método persigue predecir la futura carga de tráfico en función de la carga que ha habido en los intervalos anteriores[AOS99]. Presenta como principal ventaja la posibilidad de diseñar controles anticipativos, que procuran evitar situaciones de sobrecarga incluso antes de que esta tenga lugar, frente a los sistemas clásicos, que emplean controles reactivos y que desarrollan su acción con posterioridad a la aparición de la sobrecarga.

Realizar controles anticipativos en los mecanismos de encaminamiento es difícil puesto que es imposible predecir cuándo se va a producir una llamada determinada y cuál va a ser su duración, dado que ambos sucesos son completamente aleatorios. Desde esta perspectiva, no sería posible aplicar predicción al encaminamiento de llamadas ya que no podemos predecir el comportamiento individual de una única llamada. No obstante,

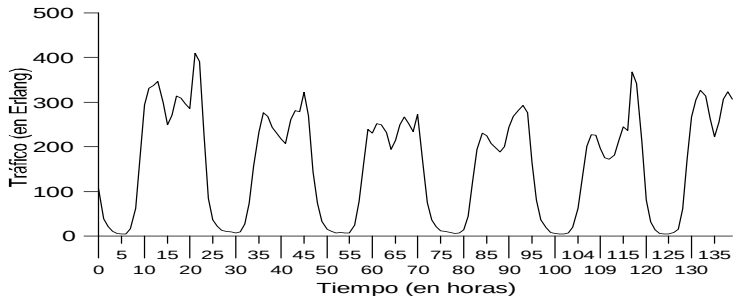


Figura 2.26: Tráfico real ofrecido a una central telefónica.

en una red real, sí es factible predecir el comportamiento estadístico de las llamadas, es decir, la tasa media de llegada y su duración media o, en otras palabras, el tráfico ofrecido a la red, en tanto que presentan una evolución quasi periódica en el tiempo.

A modo de ejemplo de la periodicidad de este comportamiento, se ha ilustrado en la figura 2.26 el tráfico real solicitado en una central telefónica a lo largo de una semana. Como se puede observar, presenta un comportamiento periódico con un período de 24 horas. Este comportamiento ha sido ampliamente estudiado [Ash97] y ha servido de base para el diseño de algoritmos como el *STR* (*State and Time-Dependent Routing*) [IMYS91] o el *DNHR* (*Dynamic NonHierarchical Routing*) [Ash85].

Con el objeto de utilizar la predicción de tráfico para resolver problemas de encaminamiento, es necesario emplear algoritmos que incorporen, como parte del algoritmo de cálculo del coste del enlace, el tráfico ofrecido al enlace.

Para realizar la predicción empleamos un filtro *FIR* [RS78], ya que éste permite realizar la predicción de la carga de tráfico a partir de unas pocas muestras históricas. Basándose en el empleo de este filtro, el tráfico es estimado en intervalos de tiempo de ΔT segundos. Así, si la última medida de tráfico se ha realizado en el instante de tiempo $M\Delta T$ segundos, el tráfico predicho para las llamadas de clase k en el instante de tiempo $(M+1)\Delta T$ vendrá dado por la expresión:

$$\hat{A}_k((M+1)\Delta T) = \sum_{l=0}^{P} \alpha_{lk} A_k((M-l)\Delta T) \tag{2.40}$$

donde $A_k(M\Delta T)$ es el tráfico de clase k medido al final del intervalo $M\Delta T$, $\hat{A}_k((M+1)\Delta T)$ el tráfico predicho para el final del intervalo $(M+1)\Delta T$, y $P+1$ el orden del

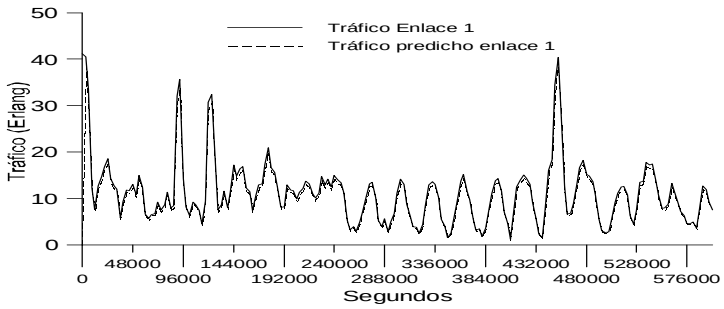


Figura 2.27: Evolución del tráfico ofrecido a un enlace y su predicción.

filtro. Los coeficientes del filtro son calculados a partir de la autocorrelación mediante el método de Durbin [RS78].

Una vez obtenida la predicción del tráfico para el instante de tiempo $(M+1)\Delta T$, es posible, mediante una sencilla interpolación lineal, obtener los valores estimados de tráfico para los instantes de tiempo comprendidos entre $M\Delta T < t < (M+1)\Delta T$.

A fin de probar las ventajas de la predicción de la carga de tráfico en los enlaces se ha forzado que el tráfico tenga una oscilación periódica dada por la siguiente expresión:

$$\lambda_g^n(t) = \lambda^n \left(\frac{1}{2} + \frac{1}{2} \cos \left(\frac{2\pi t}{30000} + \phi(n) \right) \right) \tag{2.41}$$

donde $\lambda_g^n(t)$ es el tráfico generado por el nodo n en el instante de tiempo t, siendo λ_n la tasa nominal de dicho tráfico, la cual depende de la matriz de tráfico empleada. Por su parte $\phi(n)$, es una variable aleatoria uniformemente distribuida entre 0 y 2π que permite que los picos y los valles en el tráfico generado por los distintos nodos no coincidan en el tiempo. El empleo de esta técnica para generar el tráfico permite producir oscilaciones del tráfico ofrecido a los enlaces como se puede apreciar en la figura 2.27. En este caso se ha utilizado un filtro de orden $p = 6$.

Puesto que el tráfico ofrecido a los enlaces presenta una variación quasi-periódica, con una fuerte correlación entre sus muestras, es fácilmente predecible su comportamiento. El empleo de mecanismos de predicción permite mejorar el algoritmo de encaminamiento básico, como se deduce de la la figura 2.28. En este experimento se ha ofrecido a la red un tráfico con oscilación periódica con un valor medio de 3.240 *Erlangs* y una duración media de llamada de 1.800 segundos y distribución exponencial. En dicha figura se presenta el resultado obtenido al ofrecer la misma secuencia de llamadas a los

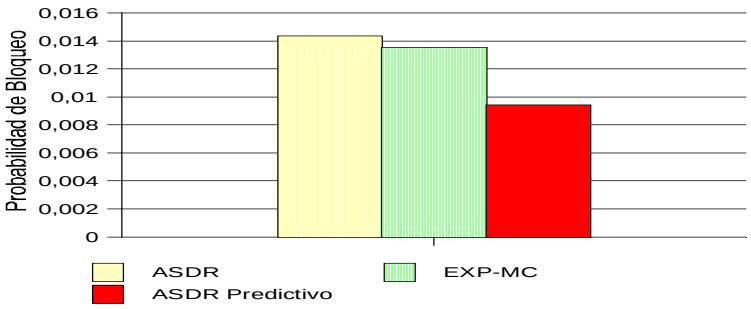


Figura 2.28: Probabilidades de pérdida de los algoritmos *EXP-MC* y *ASDR* con predicción y sin predicción.

distintos algoritmos. Se observa que la predicción permite mejorar el algoritmo básico.

A pesar de las ventajas que aporta la predicción, también presenta inconvenientes. En primer, lugar se ha supuesto que el tráfico varía de forma quasi-periódica. Sin embargo, es posible que al aumentar el tráfico de ordenadores, sin supervisión humana, este comportamiento de tipo periódico desaparezca, ya que éstos no se ven afectados por el ciclo diario de los seres humanos. El previsible incremento en la complejidad en este tipo de entornos daría lugar a un tráfico mucho más difícil de predecir. Por otro lado, se deben considerar los efectos provocados por la propia predicción, ya que ésta afecta a los resultados del encaminamiento, lo que provoca divergencias entre los valores que realmente se miden en la red y los valores predichos, haciendo la predicción inestable. Un ejemplo de este fenómeno sería la situación donde se predice una fuerte sobrecarga en un enlace, en dichas circunstancias el algoritmo debe desviar el tráfico a fin de prevenir esta situación no deseable. Esta actuación dará lugar a que el valor de la muestra final sea muy distinto al valor que se había predicho.

2.7 Conclusiones

En este capítulo se ha presentado una extensión del algoritmo *ASDR* para su utilización en redes *ATM* con múltiples clases de llamada. Al estudiar el rendimiento de éste se ha podido constatar que dicho algoritmo presenta un excelente rendimiento para cargas de tráficos bajas y medias. Por el contrario, el algoritmo presenta diversas limitaciones que dificultan su implementación en sistemas reales. Entre estas limitaciones

destacan especialmente dos: la primera es la alta carga computacional que precisa y, la segunda, la dificultad que conlleva la estimación del tráfico ofrecido a los enlaces, estimación que es necesaria para el cálculo del algoritmo. A fin de resolver estos problemas se han presentado una serie de soluciones con el objetivo de poder usar este algoritmo en situaciones reales.

La solución propuesta en este trabajo para resolver el problema del coste computacional se basa en la aplicación de técnicas neuronales que permiten estimar el valor de la función de coste. La utilización de este tipo de técnicas, además de aliviar notablemente el coste computacional, posibilita adaptar el funcionamiento del algoritmo de encaminamiento, de forma dinámica, a los distintos controles de admisión utilizados en la red, permitiendo trabajar al algoritmo en redes heterogéneas con distintos controles de admisión y sin un conocimiento previo de su funcionamiento.

Para resolver el problema de la estimación de tráfico ofrecido a los enlaces se han propuesto dos alternativas. La primera de estas alternativas está basada en el uso de una serie de ventanas solapadas en el tiempo, permitiendo de esta sencilla manera trabajar con valores de estimación no excesivamente desfasados en el tiempo y consiguiendo mejorar el rendimiento del sistema. La segunda estrategia, mucho más agresiva, se basa en la creación de controles anticipativos mediante la predicción de carga de tráfico. Se ha podido constatar que, si bien esta alternativa mejora el rendimiento del sistema, al intentar optimizar la predicción se produce un efecto de realimentación que la hace inestable.

En cualquier caso, debemos cuestionarnos la validez de este tipo de algoritmos en el entorno actual, ya que fueron especialmente diseñados para redes *ATM* donde las diferentes clases de tráfico están perfectamente caracterizadas. Por contra, el entorno actual es muy diferente, debido, por un lado, al éxito de las redes IP y su posterior adaptación para transportar tráfico con requisitos de *QoS* y, por otro lado, al avance de los algoritmos de compresión de datos. Estas circunstancias han dado lugar a un entorno en el cual no hay una separación clara entre las distintas clases de tráfico y, en consecuencia, donde antes aparecían unas clases perfectamente caracterizadas, ahora surge una serie de vagos descriptores que solapan las distintas clases. Este hecho da lugar a que los algoritmos anteriormente comentados presenten un bajo rendimiento.

Con el objetivo de estudiar y resolver este problema, surge la necesidad de trabajar con algoritmos que no precisen una perfecta caracterización del tráfico presente en la red,

sino que sean capaces de trabajar con una información que no dependa de los descriptores de tráfico de las clases. Este problema se estudia con detalle en el siguiente capítulo, en el cual se abandona este tipo de algoritmos diseñados para trabajar con clases por otro tipo de algoritmos, más sencillos, capaces de trabajar en estas condiciones de desconocimiento e imprecisión de los datos.

Capítulo 3

Encaminamiento con calidad de servicio

3.1 Introducción

Uno de los más importantes desafíos que las modernas redes de comunicaciones deben afrontar es poder garantizar que se cumplan los requisitos de *QoS* extremo a extremo que solicita cada conexión. Desde el punto de vista del encaminamiento, esto significa encontrar un camino que no sólo minimice el coste asociado a una determinada métrica (tradicionalmente el número de saltos en Internet o coste económico en RTC), sino que, además, garantice una serie de requisitos de calidad extremo a extremo.

3.1.1 Métricas de calidad de servicio

Antes de continuar, es necesario definir el concepto de calidad de servicio. Crawley [CNRS98] define como calidad de servicio "un conjunto de requisitos de servicio que deben ser proporcionados por la red para transportar un flujo de información", entendiéndose como flujo [CNRS98] "una secuencia de paquetes desde un origen a un destino con una determinada calidad de servicio asociada". Así pues, la calidad de servicio es un acuerdo entre el usuario y el proveedor de servicios en el que este último se compromete a proporcionar al usuario una conexión que cumpla una serie de requisitos mínimos. Los distintos requisitos constituyen el conjunto de parámetros de *QoS* solicitados por el usuario.

A fin de poder realizar encaminamiento con calidad de servicio es necesario modificar el modelo de red presentado en el apartado 2.2.1. Ahora, los enlaces disponen de una serie de propiedades además del ancho de banda. Estas propiedades adicionales están relacionadas con la capacidad de la red para soportar conexiones con los requisitos de calidad solicitados. En la figura 3.1 se puede ver, a modo de ejemplo, el esquema de una red donde se muestran las distintas métricas de *QoS* asociadas a cada enlace e_{ij}, que en este caso consisten en el ancho de banda residual del enlace, el retardo, el coste económico que implica el uso del enlace y la probabilidad de pérdida de paquete $(BW_{ij}, D_{ij}, coste_{ij}, P_{ij})$. Las restricciones impuestas por los nodos individuales se incluyen al describir el estado de los enlaces adyacentes. Así, el ancho de banda disponible en el enlace será el mínimo disponible de entre el propio ancho de banda del enlace y el ancho de banda que es capaz de conmutar el nodo. De hecho, en la actualidad es el nodo quien suele limitar el ancho de banda, circunstancia que tiende a aumentar, ya que la velocidad de los enlaces se duplica en media cada 12 meses mientras que la velocidad de las CPU lo hace cada 18 meses (ley de Moore). Por otra parte, el retardo del enlace será la suma de los retardos de trasmisión más los retardos producidos en las colas del nodo, en tanto que la probabilidad de pérdida dependerá de la tasa de error binaria de trasmisión combinada con la probabilidad de perder un paquete en las colas del nodo debido a un posible *overflow* o desbordamiento.

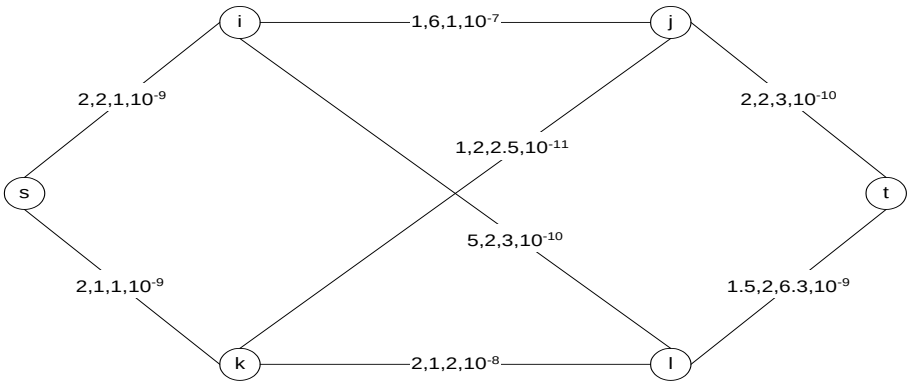


Figura 3.1: Red de ejemplo con el estado de sus enlaces.

3.1.2 Búsqueda del camino con QoS

El tipo de métrica utilizado influye en la implementación de los algoritmos de búsqueda de caminos.

Un camino P formado por los enlaces $e_{ij} \in P$, en donde existe un coste $coste_{ij}$ asociado a cada enlace e_{ij}, sigue una métrica cóncava si y sólo si se cumple:

$$coste_P = min(coste_{ij}) \ \ \forall e_{ij} \in P \tag{3.1}$$

siendo $coste_P$ el coste total del camino P. Un ejemplo de este tipo de métrica es el ancho de banda disponible en un camino. En cambio, diremos que el coste sigue una métrica aditiva si:

$$coste_P = \sum_{e_{ij} \in P} coste_{ij} \tag{3.2}$$

Este tipo de métrica se utiliza tanto para el retardo como para el *jitter* o para el coste económico de los enlaces. Por último, se asume que el coste se rige una métrica multiplicativa si cumple:

$$coste_P = \prod_{e_{ij} \in P} coste_{ij} \tag{3.3}$$

Un ejemplo de este tipo de métrica lo constituye la probabilidad de pérdida. Sin embargo, hay que tener en cuenta que una métrica multiplicativa puede reducirse en última instancia a una de tipo aditivo [GO99].

El problema de la búsqueda de un camino que cumpla los requisitos de *QoS* es la minimización de una función en la que se combinan métricas de muy diversa naturaleza (cóncavas, aditivas y multiplicativas).

La complejidad de trabajar con varias métricas queda ejemplificada si regresamos a la red de la figura 3.1. En ella existen varias posibles alternativas para establecer un camino entre el nodo s y el nodo t en función de la métrica empleada como objetivo a minimizar. Así, si se pretende emplear el ancho de banda como métrica objetivo, el camino escogido será $P = s \rightarrow i \rightarrow l \rightarrow t$ y el coste total del camino será de 1'5 unidades de ancho de banda. Si, por el contrario, se escoge el retardo como métrica objetivo el camino será $P = s \rightarrow k \rightarrow l \rightarrow t$. En el caso de considerar el coste económico, el camino sería $P = s \rightarrow i \rightarrow j \rightarrow t$ y, por último, si el objetivo hubiera sido minimizar la probabilidad de pérdida de paquete el camino habría resultado $P = s \rightarrow k \rightarrow j \rightarrow t$.

En el peor de los casos asumibles, buscar un camino que cumpla todos los requisitos de calidad de servicio no sólo consiste en encontrar el camino de coste mínimo con respecto a una determinada métrica, sino que, además, dicho camino debe cumplir simultáneamente todas las restricciones impuestas por los parámetros de *QoS*. Lee *et al.* [LHH95] establecen que estas restricciones pueden ser de tres tipos:

1. Restricciones debidas al rendimiento (retardo, ancho de banda, *jitter*, etc.). Dentro de este grupo podemos distinguir, a su vez, entre tres clases de restricciones:

   (a) Restricciones de enlace, que hacen referencia a los recursos que ha de poseer un enlace para que éste pueda ser usado (por ejemplo, el ancho de banda disponible en los enlaces). Este tipo de restricciones sigue habitualmente una métrica cóncava.

   (b) Restricciones de camino, que hacen referencia a las limitaciones extremo a extremo que se deben cumplir. Este tipo de restricciones, que se indican mediante un valor máximo que la conexión no debe superar, normalmente siguen una métrica aditiva o multiplicativa.

   (c) Por último, restricciones de árbol utilizadas en las conexiones *multicast*. Esta clase de acotaciones impone límites a los recursos que son utilizados en la construcción del árbol que conecta los distintos nodos de una conexión multipunto o *multicast*. La aplicación de éstas pueden dar lugar a que determinados nodos no puedan conectarse o que sea necesario rehacer el árbol completamente.

2. Restricciones debidas a los recursos. Su misión es limitar la utilización de un determinado recurso (o grupo de recursos) para un determinada clase de conexiones (por ejemplo, evitar el empleo de líneas de transmisión radio para comunicaciones de alta seguridad). Es el usuario o el administrador quien determina cuáles son las restricciones que se aplicarán dentro de este grupo.

3. Restricciones de prioridad. Este tipo implanta una serie de prioridades a la hora de establecer una conexión, de forma que puede rechazar solicitudes de conexión de llamadas de baja prioridad, aun cuando existen suficientes recursos para su establecimiento, a fin de poder establecer futuras conexiones de alta prioridad cuando estas sean solicitadas. Un ejemplo de este tipo es el formado por los canales reservados para las comunicaciones militares, ya que, aunque habitualmente estos canales no estén siendo utilizados, no pueden ser empleados por otras conexiones.

En [GJ79] se demuestra que minimizar una función multivariable que consta de más de una métrica sólo se resuelve en un tiempo no acotado polinómicamente. Así pues, la búsqueda de un camino de coste mínimo con respecto a una serie de métricas es un problema NP-Completo, puesto que no es posible encontrar una solución a éste en un tiempo polinómico, lo que quiere decir que el tiempo necesario para encontrar la

solución no es proporcional a la complejidad del problema. Sin embargo, en determinadas circunstancias, es posible encontrar una solución aproximada al problema en un tiempo polinómico.

Una de las soluciones propuestas para evitar este problema NP-Completo consiste en construir una función de coste que sea combinación de los distintos parámetros de *QoS* requeridos y, a continuación, buscar el camino de coste mínimo de acuerdo a una variable única, combinación de las demás [WC96b]. De esta manera desaparece el problema NP-completo ya que las distintas métricas han sido reducidas a una sola. Sin embargo, esta solución no garantiza que el camino encontrado satisfaga todas las restricciones de calidad de servicio, puesto que esta nueva métrica resulta de un compromiso entre las demás, pudiendo ocurrir que valores elevados en un determinado parámetro sean compensados por valores pequeños de algún otro de los parámetros. Esta circunstancia puede dar lugar a caminos de coste combinado mínimo que no cumplan alguna de las restricciones.

Con el objetivo de solucionar el problema de búsqueda de camino de coste mínimo sujeto a múltiples restricciones se han propuesto en la literatura diversos algoritmos seudo-óptimos que resuelven el problema en un tiempo polinómico. A continuación se presentan algunos de ellos.

3.2 Algoritmos de búsqueda de camino con calidad de servicio

Los algoritmos de búsqueda de camino con *QoS* pueden dividirse en dos grandes grupos:

1. Encaminamiento *unicast*, donde existe un origen s, un destino t y una serie de restricciones de calidad de servicio. Su objetivo es encontrar un camino que satisfaga el conjunto de restricciones de calidad de servicio deseadas.

2. Encaminamiento *multicast* donde existe un nodo origen s, un conjunto de nodos destino R y un conjunto de restricciones de calidad de servicio. En este caso el fin es encontrar el "árbol" que conecta el nodo origen con el conjunto de los nodos destino cumpliendo los requisitos de calidad de servicio.

Este trabajo está centrado en el problema del encaminamiento *unicast*. Por otro lado, los distintos algoritmos aquí presentados están diseñados para trabajar con encaminamiento en origen. No obstante, un estudio extendido a algoritmos *multicast* y con procesamiento distribuido es posible encontrarlo en [CN98b].

La decisión de centrarse en los algoritmos con encaminamiento en origen se debe a que esta estrategia ha sido la empleada tradicionalmente en las redes orientadas a conexión. Para poder garantizar calidad de servicio a los usuarios es necesario que la información emplee circuitos, ya sean físicos o virtuales, y por lo tanto, utilizar redes orientadas a conexión. El encaminamiento en origen presenta una serie de ventajas [Wid94]:

1. Favorece la construcción del árbol de rutas. Una de las primeras misiones del algoritmo de encaminamiento es construir los caminos a los distintos destinos. Las restricciones obligan a replantearse y revisar constantemente los caminos, y esto último es más fácil si se adopta un esquema centralizado que uno distribuido.

2. Evita en mayor medida la posibilidad de formar lazos cerrados. El encaminamiento en origen permite conocer los nodos que componen el camino y, en consecuencia, detectar y eliminar los lazos cerrados de una manera más sencilla que los algoritmos distribuidos, donde no se conoce la totalidad de los nodos por los que pasará el flujo.

3. Favorece la gestión de políticas de decisión. Éstas permiten al administrador establecer una serie de reglas de decisión como, por ejemplo, excluir la utilización de un enlace o conjunto de enlaces para un determinado tipo de tráfico.

4. Permite gestionar con mayor facilidad los recursos compartidos, optimizando la gestión de los sistemas con ganancia estadística.

Es cierto que esta estrategia presenta varias desventajas, la mayor de las cuales es la exigencia de conocer el estado completo de la red en cada nodo para poder determinar un camino válido.

A continuación presentaremos varios algoritmos *unicast* de búsqueda de camino con restricciones de *QoS*, proponiendo seguidamente un algoritmo capaz de encontrar el camino mínimo con respecto a una determinada métrica cumpliendo de forma simultánea múltiples restricciones. El algoritmo propuesto está basado en el algoritmo *K-shortest*

path, lo que permite encontrar no uno, sino varios caminos ordenados en forma ascendente con respecto a la métrica objetivo y que cumplen múltiples restricciones de calidad de servicio.

3.2.1 Algoritmos de búsqueda de camino con QoS y encaminamiento en origen

3.2.1.1 Algoritmo *Constrained Belman-Ford* (CBF) (o Belman-Ford con restricciones).

Este método, propuesto en [Wid94], es capaz de encontrar el camino de coste mínimo entre un nodo origen y un nodo destino, que, simultáneamente, cumpla las restricciones de retardo impuestas. Dicho algoritmo realiza la búsqueda del camino mediante un bucle en el cual se incrementa de forma monótona creciente el retardo hasta alcanzar el valor límite impuesto. Dentro del bucle se busca, para cada valor que toma el retardo, el camino de coste mínimo. De esta forma se tiene una tabla donde aparece el camino de coste mínimo para cada valor de retardo. Así, si se desea encontrar el camino de coste mínimo que satisface cierta restricción de retardo, basta con buscar en esta tabla el camino con menor coste de todos los existentes. En varios de los algoritmos expuestos a continuación se emplean técnicas parecidas basadas en el algoritmo *Bellman-Ford*. En la figura 3.2 se presenta el pseudo-código de este algoritmo.

3.2.1.2 Algoritmo presentado por A. Przygienda.

En su tesis [Prz95], A. Przygienda presenta un algoritmo basado en un diagrama de Hasse que utiliza una estructura regular de "celosías" o celdas. Estos diagramas están formados por conjuntos de elementos ordenados jerárquicamente, en donde el criterio de ordenación utilizado es el siguiente: supongamos que disponemos de un conjunto en el cual cada elemento esta compuesto por dos componentes distintos, diremos que un elemento que toma como valores a, b para cada uno de sus componentes antecede a un segundo elemento que toma como valores x, y si se cumple la condición:

$$\begin{pmatrix} a \\ b \end{pmatrix} \preceq \begin{pmatrix} x \\ y \end{pmatrix} \Leftrightarrow a \leq x \;\; y \;\; b \leq y \tag{3.4}$$

Przygienda establece una correspondencia entre cada componente de los elementos del diagrama de *Hasse*, con un parámetro de calidad de servicio y, a continuación, propone

```
CBF(origen, destinos[], restricciones[]){
//camino_coste_minimo[] Contiene la lista de caminos a cada nodo ordenados en
//forma de menor a mayor coste (y de menor a mayor retardo) para encontrar un
//camino se toma el primer elemento de la lista que cumple las restricciones
//destinos[] contiene el conjunto de nodos a los que se quiere construir el
//camino.
// La estructura enlace consta de los siguiente campos:
// origen_enlace nodo origen de los dos nodos que conecta el enlace
// destino_enlace nodo destino de los dos nodos que conecta el enlace
// coste_acumulado coste acumulado desde el origen hasta el nodo destino_enlace
// coste Coste asignado al enlace
// retardo_acumulado retardo acumulado desde el origen hasta el nodo destino_enlace
// retardo Retardo de ese enlace
 retardo_maximo = MAX(restricciones[]) //Encuentra la limitación máxima
 Retardo_acumulado = 0 //iniciliaza la variable que lleva el retardo del camino
 Para todo enlace desde el origen{
     enlace.coste_acumulado = enlace.coste //Función de coste
     enlace.retardo_acumulado = enlace.retardo //Retardo
     insertar_pila_con_priorida(enlace) //esta pila está ordenada de forma que
//el enlace de menor valor en el campo coste_aumulado es el primero de la pila
 }
 repetir mientras que retardo_acumulado <= retardo_maximo y
 la pila con prioridad no esta vacía{
        enlace = sacar_elemento_pila_con_prioridad()
        Comprobar si hay cambio_coste(camino_coste_minimo,
                            enlace.destino_enlace,
                            enlace.origen_enlace,
                            enlace.coste_aumulado,
                            enlace.retardo_acumulado)
        Si hay cambio{
               Para todo enlace desde el nodo enlace.destino_enlace {
               nuevo_enlace.coste_aumulado = enlace.coste_acumulado +
                                                      nuevo_enlace.coste
               nuevo_enlace.retardo_aumulado = enlace.retardo_acumulado +
                                                     nuevo_enlace.retardo
               insertar_pila_con_priorida(nuevo_enlace)
           }
        }
        retardo_acumulado = enlace.retardo_acumulado;
  }
}

función boolean cambio_coste(camino_coste_minimo, nodo_actual, nodo_anterior,
nuevo_coste, nuevo_retardo){
//Añade un nuevo camino a la lista de caminos del nodo actual
//si éste tiene menor coste (y mayor retardo) que el camino
//actualmente a la cabeza de la lista
  camino_actual = camino_coste_minimo(nodo_actual);
  Si existen un camino con igual número de nodos en camino_actual {
    Si coste de camino_actual <= nuevo_coste
      return false //El nuevo coste es mayor que el almacenado y no es necesario reemplazar
    En caso contrario
         si retardo camino_actual = nuevo_retardo {
//Eliminar el camino de la cabeza de la lista pues este puede ser
//reemplazado por un camino con idéntico retardo pero menor coste
       borrar camino_coste_minimo(nodo_actual)
    }
  }
  reconstruir  camino_coste_minimo(nodo_actual) con la
  nueva información  para que el nodo predecesor al
  nodo_actual sea nodo_anterior
  Actualizar coste y retardos de camino_coste_minimo(nodo_actual)
  a nuevo_coste y nuevo_retardo
  return true
}
```

Figura 3.2: Algoritmo *CBF*

un algoritmo capaz de ordenar los caminos en función de este diagrama. De esta manera es posible obtener un camino de compromiso entre los diversos parámetros de calidad de servicio sin necesidad de construir una función cuyo valor sea una combinación de los mismos. Este algoritmo es fácil de extender a múltiples requisitos de calidad de servicio, sin embargo presenta un inconveniente debido a que el camino escogido es el que presenta la mejor relación entre las distintas métricas. En cambio, normalmente es más interesante escoger un camino con una peor relación entre métricas, pero que sea mínimo con respecto a una de ellas y al mismo tiempo cumpla todos los requisitos exigidos.

La solución aportada es interesante pues evita el problema que presentaban propuestas anteriores, donde se diseñaba una función de coste que era combinación lineal de los distintos parámetros de calidad de servicio.

3.2.1.3 Algoritmo presentado por Z. Wang y J. Crowcroft.

En [WC96b] Wang y Crowcroft presentan una variación del algoritmo de *Dijkstra* que permite encontrar un camino que cumple simultáneamente los requisitos de ancho de banda y de retardo. Este algoritmo trabaja en dos etapas, en la primera se eliminan los enlaces en los cuales el ancho de banda disponible es menor que el ancho de banda requerido. A continuación, en la red residual se aplica el algoritmo de *Dijkstra* empleando como métrica objetivo el retardo de los enlaces, de forma que se encuentra el camino con menor retardo. El camino con requisitos de calidad de servicio será posible siempre que el camino encontrado tenga un retardo menor que el límite impuesto.

3.2.1.4 Algoritmo presentado por Q. Ma y P. Steenkiste.

Este algoritmo, aplicado en [MS97] y [Ma98], se basa en la suposición de que la red emplea algoritmos de planificación tipo *Weighted Fair Queuing (WFQ)* [DKS89] [Gol94] [BZ97][BZ96] [Kes97], ya que mediante la utilización de estos algoritmos en combinación con el uso de un mecanismo de conformado tipo *Leaky Bucket*, es posible acotar y relacionar entre sí los distintos parámetros de *QoS* (como pueden ser el retardo o el *jitter*). Así pues, si a una conexión k se le asigna un *Leaky Bucket* de parámetros $\langle \varsigma_k, \theta_k \rangle$ (donde ς_k es la velocidad de decremento del *Leaky Bucket* y θ_k es el valor máximo que puede tomar el contador del *Leaky Bucket*) el retardo máximo de la conexión k a

través del camino P vendrá dado mediante la expresión [Zha95]:

$$D(P, b_k, \theta_k) = \frac{\theta_k}{b_k} + \frac{n\, To_{max}}{b_k} + \sum_{e_{ij} \in P} \frac{To_{max}}{C_{ij}} + \sum_{e_{ij} \in P} prop_{ij} \tag{3.5}$$

donde C_{ij} es la capacidad del enlace e_{ij}, n es el número de enlaces que componen el camino P, To_{max} es el tamaño máximo que puede tener un paquete en la red, $prop_{ij}$ es el retardo de propagación del enlace y $b_k \geq \varsigma_k$ es el ancho de banda reservado para la conexión. Igualmente, el *jitter* o variación maxima del retardo vendrá acotado por la expresión:

$$Jitter(P, b_k, \theta_k) = \frac{\theta_k}{b_k} + \frac{n\, To_{max}}{b_k} \tag{3.6}$$

Basándose en estas expresiones, Ma y Steenkiste [MS97] proponen una modificación del algoritmo *Constrained Bellman-Ford*, añadiendo un nuevo bucle que itera sobre el ancho de banda residual de todos los enlaces, incrementando éste en cada ciclo (con lo que se consiguen menores retardos y *jitter*) y eliminando los enlaces cuyo ancho de banda sea menor que el nuevo límite impuesto en cada ciclo del bucle. De esta manera, y gracias a la relación existente entre el ancho de banda, retardo y *Jitter*, el nuevo algoritmo es capaz de encontrar, a cambio de un alto coste computacional, un camino que cumpla los requisitos de calidad de servicio extremo a extremo.

Por otro lado, Orda [Ord98] realiza un estudio detallado de este tipo de algoritmos, que se apoyan en la utilización de mecanismos de control de flujo como *Leaky Bucket* y *WFQ*, para soportar conexiones con calidad de servicio. En este trabajo se estudia la complejidad máxima del algoritmo así como la influencia en la calidad de la solución obtenida de discretizar los parámetros de calidad de servicio con el objetivo de reducir el coste computacional.

3.2.1.5 Algoritmo propuesto por R. Guerin y A. Orda.

En los trabajos [GO97] [GO99], Guerin y Orda estudian el problema del encaminamiento en redes con mecanismos de control de flujo (como el *Leaky Bucket* y *WFQ*) bajo las condiciones de un conocimiento impreciso del estado de la red. En concreto, investigan la factibilidad de resolver el problema cuando es posible modelar esta imprecisión mediante un conjunto de funciones de distribución de probabilidad. Así, proporcionan algoritmos que son capaces de resolver el problema si se suponen ciertas distribuciones de

la probabilidad de encontrar ancho de banda disponible en los enlaces y de sufrir cierto retardo. De esta manera, para el caso de restricciones en el ancho de banda cada uno de los nodos, conoce para cada enlace e_{ij} de la red la función distribución de probabilidad $p_{ij}(w)$ de que existan al menos $w \in [0 \dots C_{ij}]$ unidades disponibles de ancho de banda. Para una conexión k el objetivo es encontrar el camino que maximiza la probabilidad de encontrar b_k unidades de ancho de banda disponibles. En el trabajo citado, los autores demuestran que este problema puede ser resuelto mediante el algoritmo de *Dijkstra* asignando un coste a los enlaces igual a $-\log(p_{ij}(b_k))$.

Cuando la métrica empleada es el retardo, el objetivo es encontrar el camino con la mayor probabilidad de cumplir los requerimientos de retardo exigidos. Al igual que en el caso anterior, los nodos conocen la función de distribución de probabilidad $p_{ij}(d)$ del enlace e_{ij} y, por ende, la probabilidad de que el retardo sea igual o menor que d. Una vez más, el objetivo es encontrar el camino que maximiza una probabilidad, en este caso la de que el retardo total sea menor o igual que el retardo límite D_{max}. Aunque se demuestra que éste es un problema NP-Completo, bajo determinadas condiciones (por ejemplo, que la función de distribución que gobierna la probabilidad de ancho de banda residual siga una distribución exponencial) es posible resolver el problema en un tiempo polinómico. Para ello transforman los requisitos globales (por ejemplo que el retardo total sea menor que D_{max}) en una serie de requisitos locales mediante un mecanismo conocido como particionado de red. En el caso del retardo implica que se define un retardo límite D_{ij} por cada enlace, de modo que el retardo d_{ij} en cada enlace debe ser menor o igual que el retardo D_{ij}, cumpliéndose que $\sum D_{ij} = D_{max}$. A continuación, aplicando la transformación $-\log(p_{ij}(d_{ij}))$, donde $p_{ij}(d_{ij})$ es la probabilidad de que el enlace e_{ij} tenga un retardo de $d_{ij} \leq D_{ij}$ unidades de tiempo, se busca el camino de menor coste. Posteriormente, Lorenz y Orda [LO98] extienden este trabajo permitiendo su empleo en redes jerárquicas.

3.2.1.6 Algoritmo propuesto por Chen y Nahrstedt.

Este algoritmo [CN98a] permite obtener un camino que es mínimo con respecto a una determinada métrica y que de forma simultánea cumple un número Z de restricciones de calidad de servicio, con un valor límite c^z para cada una de ellas. Presenta como limitación el hecho de que las distintas restricciones deben seguir una métrica aditiva. Este algoritmo esta basado en el algoritmo CBF, aunque para reducir el coste computacional efectúa una transformación del problema. En este caso, transforma un conjunto

de costes y restricciones de valor arbitrario en otro conjunto en el que tanto los costes como las restricciones toman un conjunto restringido de valores enteros.

Así, si se tiene un conjunto de enlaces e_{ij}, donde cada uno de ellos tiene $Z \geq 2$ parámetros de calidad de servicio con valores ω_{ij}^z para cada uno de ellos, el objetivo es encontrar un camino que cumpla los distintos requisitos de calidad de servicio extremo a extremo, es decir, que dicho camino cumpla las condiciones $\omega^d \leq y^z$ *para cada* $z \in [2..Z]$. Para ello, el algoritmo transforma los valores ω_{ij}^z *para cada* $z \in [2..Z]$ en otro conjunto de valores γ_{ij}^z mediante la función:

$$\gamma_{ij}^z = \left\lceil \frac{\omega_{ij}^z \, h_z}{y^z} \right\rceil \tag{3.7}$$

siendo $\lceil \rceil$ una función no lineal que redondea al valor superior más cercano y h_z un número entero positivo. El problema ahora puede resolverse mediante un algoritmo iterativo basado en una versión ligeramente modificada del algoritmo CBF, en el cual cada bucle se realiza sobre el conjunto de valores de h_z y la condición limite para finalizar éste es $\gamma^z \leq h_z$. De esta manera se ha conseguido limitar el número de ciclos que debe de realizar cada bucle a fin de encontrar la solución, reduciendo de manera significativa el tiempo de cómputo. Este algoritmo no garantiza que se va a encontrar el camino, sino que proporciona una probabilidad de encontrar el camino de coste mínimo que cumple las distintas restricciones. Dicha probabilidad aumenta al aumentar los valores empleados de h_z a costa de incrementar los tiempos de computación, ya que crece el número de valores de γ_{ij}^z sobre los cuales debe iterar el algoritmo.

3.2.1.7 Algoritmo propuesto por Roginsky, Christesen y Srinivasan.

Este algoritmo [RCS99] está diseñado para buscar un camino que cumpla simultáneamente dos restricciones de tipo aditivo, como por ejemplo el retardo y el *jitter*. Para ello, los autores realizan una modificación del algoritmo CBF a fin de aumentar el espacio de búsqueda de camino. En este caso se implementa un algoritmo de dos etapas. En la primera etapa busca y almacena en una matriz los datos (retardo, *jitter*,...) de los mejores caminos a los distintos nodos en función del número de nodos intermedios de que consta el camino. Así, en esta matriz existe una entrada para cada número de saltos y cada nodo destino. Un valor será almacenado en la matriz si para un nodo concreto y un determinado número de saltos no hay ninguna otra entrada que tenga valores menores, tanto del retardo como del *jitter*. Cada vez que se añade una nueva entrada es necesario

explorar la matriz a fin de eliminar las entradas cuyos valores sean mayores que la nueva (en el caso de que coincidan el número de saltos y el nodo destino). En la segunda fase del algoritmo, se explora la matriz obtenida a fin de extraer el camino óptimo que cumple las dos restricciones. Lógicamente, la principal desventaja de este algoritmo es que exige una búsqueda intensiva en el espacio de estados.

3.2.1.8 Algoritmo propuesto por De Neve y Van Mieghem.

Este algoritmo [Nb00], al igual que el que proponemos en este trabajo, está basado en una variación del algoritmo *K-Shortest Path*. Además de cumplir las distintas restricciones, este algoritmo busca el camino cuyo coste es mínimo con respecto a una función que es combinación de los distintos parámetros de calidad de servicio, a fin de encontrar una solución no dominante, es decir, no busca el camino de menor coste que cumpla la restricción de retardo, sino uno que resulte de un compromiso entre los costes de los dos parámetros, de tal forma que la aportación de los distintos elementos en la solución final encontrada sea de igual importancia. Sin embargo, creemos que el objetivo final no es buscar una solución no dominante, sino todo lo contrario, buscar el camino que cumpla los requisitos de calidad de servicio y cuyo coste esté relacionado, de forma dominante, con el *throughput* de la red a fin de mejorar el aprovechamiento del sistema.

3.2.2 Algoritmo mejorado de búsqueda de camino óptimo con múltiples restricciones

3.2.2.1 Introducción

El algoritmo aquí presentado está diseñado para trabajar con encaminamiento en origen y permite encontrar, con una alta probabilidad de éxito, el camino de coste mínimo sujeto a multiples restricciones de *QoS*. Es decir, el algoritmo no garantiza que encontrará, con toda seguridad, el camino de coste mínimo sujeto a restricciones (siempre que éste exista), sino lo que proporciona es una alta probabilidad de encontrar dicho camino. Esta probabilidad es ajustable mediante una variable de control, que se relaciona con los tiempos de cómputo del algoritmo. A diferencia de los algoritmos presentados anteriormente, el tiempo de cómputo no depende directamente del número de restricciones simultáneas que se desean cumplir, sino de forma indirecta a través de esta variable de control que sí depende del número de restricciones. Sin embargo, como

en la práctica los distintos parámetros de *QoS* están relacionados entre sí, el crecimiento de la variable de control no es lineal con el crecimiento del número de parámetros. Dependiendo del tamaño de la red y su complejidad (grado de interconexión), se puede llegar rápidamente a un valor a partir del cual se tiene una probabilidad de encontrar el camino de coste mínimo muy cercana al 100%.

El algoritmo que se propone es una variación de la solución de tipo *K-shortest path* presentada por Chong *et al.* en [CMM95]. El problema *K-shortest path* se puede ver como la generalización del algoritmo de *Dijkstra*, donde no sólo se pretende encontrar el camino de coste mínimo entre un nodo origen y un destino, sino que, añadidamente, se persigue encontrar los K mejores caminos ordenados de mejor a peor.

A este algoritmo básico se le ha añadido un mecanismo de poda que elimina los caminos que no cumplen los requisitos de calidad de servicio. El hecho de que este algoritmo esté basado en el *K-shortest path* permite llegar a un nodo por diversos caminos alternativos, lo que posibilita que si no es posible progresar en la búsqueda desde un determinado nodo, por producirse una violación de los requisitos de calidad de servicio, este nodo no sea eliminado del árbol de búsqueda (como ocurre con el algoritmo de *Dijkstra* [Dij59]), y sea posible alcanzar el destino por otro camino alternativo que no viole los requerimientos de calidad de servicio exigidos. Esta limitación del algoritmo de *Dijkstra* se muestra claramente en la red de la figura 3.3, en donde se pretende establecer una conexión entre los nodos A y E cuyo retardo no supere las 7 unidades. El algoritmo de *Dijkstra* básico escoge como camino el formado por los nodos $A \rightarrow B \rightarrow D \rightarrow E$ ya que es el camino de coste mínimo, sin embargo, puesto que el retardo total del camino es 9 unidades, la llamada será rechazada. Para poder establecer con éxito la conexión es necesario escoger el camino $A \rightarrow C \rightarrow D \rightarrow E$, pero el algoritmo básico solo permite acceder a un nodo por un único camino y en el caso del nodo D este ya ha sido accedido por el camino $A \rightarrow B \rightarrow D$, con lo cual la llamada será rechazada. En cambio, el algoritmo *K-shortest path* permite visitar al nodo D por varios caminos alternativos, de forma que además del camino $A \rightarrow B \rightarrow D$, también estará disponible el camino $A \rightarrow C \rightarrow D$ y a través de este último se podrá acceder al nodo E. Un ejemplo más completo se muestra en la figura 3.4 aplicado a la red de la figura 3.1, donde se pretende encontrar el camino de coste mínimo entre los nodos s y t que simultáneamente cumpla los requisitos de ancho de banda $BW \geq 1.5\ unidades$ y de retardo $D_{limite} \leq 4$. El algoritmo aquí presentado, busca varios caminos en paralelo, progresando siempre por la solución de menor coste hasta que ésta viola una de las restricciones impuestas o llega al nodo destino. Así pues, el camino $P = s \rightarrow i \rightarrow j \rightarrow t$, que es el camino de coste mínimo,

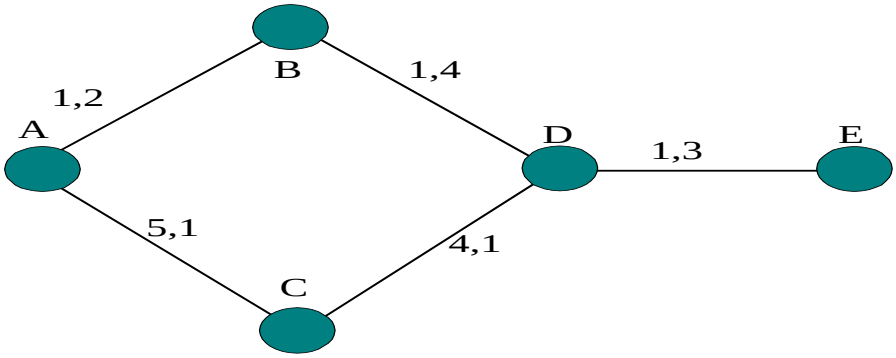


Figura 3.3: Ejemplo de las limitaciones del algoritmo de *Dijkstra*. Cada dupleta de números indica coste y retardo.

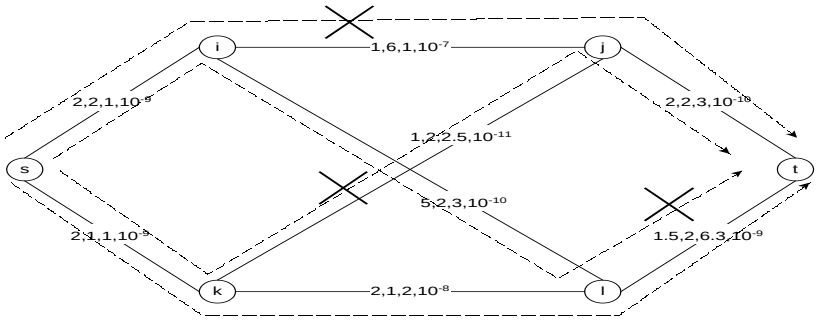


Figura 3.4: Ejemplo de búsqueda en paralelo con poda de los caminos no válidos. Para cada enlace el vector de números indica: Ancho de banda, retardo, coste y probabilidad de pérdida de paquete.

no puede elegirse ya que el ancho de banda del enlace e_{ij} (1 unidad) es menor que el límite. El camino $P = s \rightarrow k \rightarrow j \rightarrow t$ también es eliminado por no disponer suficiente ancho de banda en el enlace e_{kj} (1 unidad). Por otro lado, el camino $P = s \rightarrow i \rightarrow l \rightarrow t$ sí dispone de suficiente ancho de banda, sin embargo, conforme se construye el camino se produce una violación de la acotación impuesta al retardo, por lo que también es eliminado. Finalmente, el único camino disponible es $P = s \rightarrow k \rightarrow l \rightarrow t$ y esta será la solución proporcionada por el algoritmo.

3.2.2.2 Estructuras de datos utilizadas

Antes de empezar a describir con detalle el funcionamiento del algoritmo hay que especificar las estructuras de datos utilizadas por este. En concreto, para cada nodo se utiliza un vector de K elementos (en el cual se almacenará la información relativa a los K caminos por los que se podría llegar al nodo), donde cada elemento consta, además de los campos utilizados para los parámetros de *QoS*, los campos *etiqueta*, *coste*, *predecesor* y *prev_idk*. Los campos *coste* y *etiqueta* son utilizados de igual forma que en el algoritmo de *Dijkstra* básico y permiten almacenar el coste total para llegar al nodo por un determinado camino, así como decidir si este nodo forma parte (*etiqueta=permanente*) o está en proceso de formar parte de un camino (*etiqueta=tentativo*). El campo *predecesor* identifica el nodo que antecede al actual en uno de los caminos que se están construyendo y *prev_idk* identifica un elemento dentro del vector del nodo anterior. Estos campos permiten reconstruir el camino más corto hasta el origen. Inicialmente a todos los campos relativos a los parámetros de *QoS* se les asigna el valor pésimo: infinito (caso del coste, retardo y *jitter*), cero (ancho de banda) o uno (probabilidad de pérdida), el campo *etiqueta* se pone a tentativo (indica que todavía no forma parte de ningún camino) y los campos *prev_idk* y *predecesor* se inicializan a un valor no válido. Así mismo, se utiliza una estructura de almacenamiento temporal en forma de pila donde se guarda información relativa a los cambios que se han producido en cada ciclo del algoritmo.

3.2.2.3 Funcionamiento

En el apéndice A se muestra el código correspondiente a la implementación básica del algoritmo, el cual emplea una lista simplemente enlazada para la gestión de la pila. Este algoritmo comienza tomando el nodo inicial y señalando en todas sus etiquetas que dicho nodo es permanente (forma parte de un camino). En un segundo paso se procede a calcular los parámetros de *QoS* y el coste a todos los nodos a los que éste está directamente conectado. Si el coste calculado es menor que el coste que estos nodos tienen almacenados en sus vectores, y no se viola ninguna de las restricciones de *QoS*, se escoge un elemento del vector cuyo coste sea mayor que el nuevo y se sustituyen los valores contenidos por los nuevos valores, almacenando a continuación la información relativa a este cambio en la pila. En un tercer paso, se extrae de la pila el elemento de menor coste y se repite el algoritmo desde el segundo paso (sustituyendo el nodo de inicio por un nuevo nodo cuya información ha sido extraída de la pila). Estos pasos se

repiten hasta alcanzar el nodo destino o hasta que no quede ningún elemento en la pila (situación que indica que el algoritmo no ha podido encontrar un camino válido).

Empleando métodos de búsqueda heurística [PTVF92] para el manejo de la pila, el orden de la complejidad del algoritmo es, en el peor de los casos, $O(K|E|\log(K|V|) + K^2|E|)$, donde $|V|$ es el número de nodos de la red, $|E|$ es el número de enlaces de la red y K el número máximo de posibles caminos por los que se intenta llegar a un determinado nodo. Esta variable K es la variable de control del algoritmo. Cuanto mayor sea el valor de K mayor será la probabilidad de obtener como solución el camino de coste mínimo que cumple los límites impuestos. Las pruebas han demostrado que el mecanismo de poda permite una convergencia mucho más rápida que la indicada por la complejidad máxima del algoritmo, mientras que los tiempos de cómputo medios aumentan sólo ligeramente al aumentar el valor de K.

La pila funciona como una FIFO con prioridad, donde esta última viene dada por el coste (menor coste, mayor prioridad). Así pues, los elementos en la pila se ordenan de manera que el elemento de menor coste de todos los presentes en la pila es el primero en ser extraído. Adicionalmente al mecanismo de poda, hemos introducido otra modificación al algoritmo básico cambiando el criterio de ordenación de la pila de forma que si dos elementos tienen igual valor, entonces se toma como segundo criterio de ordenación el ancho de banda, con lo que se consigue que el elemento con mayor ancho de banda disponible será el primero en ser sacado de la pila. A igualdad de coste y ancho de banda tienen mayor prioridad los elementos más antiguos en la pila, hecho que permite escoger el camino con menor número de enlaces. Cuando se introduce un elemento en la pila se ha de comprobar si dicho elemento está ya presente. En el caso de que el elemento a introducir en la pila (identificado por el nodo y un parámetro k que indica cuál de los K caminos que el algoritmo prueba en paralelo es) estuviera ya al haberse accedido a él por un camino alternativo, se compara el antiguo coste con el nuevo y se deja en la pila el elemento con menor coste. Como se puede observar, otra de de las ventajas de este algoritmo es la facilidad para implementar diversos criterios de elección de camino en caso de igualdad de coste, lo que se efectúa cambiando únicamente los mecanismos de ordenación de la pila. El mecanismo de ordenación de la pila admite múltiples e interesantes variaciones. Por ejemplo, se puede cuantificar el coste de forma similar al esquema presentando por Orda [Ord98], que utiliza valores discretizados con lo que se consigue una mayor posibilidad de que los distintos caminos tengan igual coste, permitiendo de esta forma que el segundo o tercer criterio de ordenación adquiera más peso, provocando una mejor distribución del tráfico.

3.2.2.4 Pruebas del algoritmo

Para la validación del algoritmo se ha creado un programa que genera topologías de forma arbitraria, en tanto que como algoritmo de referencia se utilizó el algoritmo de *Dijkstra* con la misma codificación presentada en el texto de Tanenbaum [Tan97]. Las pruebas se han realizado sobre un total de 30 redes distintas generadas aleatoriamente, con un total de 50 nodos cada una. Tanto los costes como los retardos de los enlaces se han asignado de forma aleatoria. Dentro de cada red se han seleccionado al azar 50 pares de nodos origen-destino y se han buscado los caminos que los conectan. Los resultados de estas pruebas se presentan en la figura 3.5 en la que se muestran los porcentajes de acierto de cada uno de los algoritmos para distintos valores de K. En la tabla 3.1 se han incluido los tiempos de cómputo empleados para la obtención de los caminos que conectan a los 1500 pares origen-destino. Los resultados de tiempo se obtuvieron en un ordenador *Pentium-II* a 350 MHz con *Windows NT* como sistema operativo. Como se puede apreciar, el mecanismo de poda permite obtener, con una alta probabilidad, la solución óptima incluso para valores de K relativamente pequeños. De hecho, a partir de valores de $K \geq 15$ siempre se ha obtenido un 100% de aciertos. Otra ventaja añadida de nuestra propuesta es que los tiempos de cómputo crecen lentamente al aumentar el valor de K, al contrario del algoritmo *K-Shortest* básico. La razón de esto es que el mecanismo de poda elimina las soluciones que no cumplen los requisitos permitiendo avanzar rápidamente por las soluciones más prometedoras, de tal forma que la primera de las K posibles soluciones obtenidas es el camino mínimo que cumple las restricciones. En cambio, el algoritmo *K-Shortest* necesita obtener todas las K soluciones antes de determinar con cuál de ellas se queda.

El algoritmo que hemos propuesto aquí es una interesante alternativa a las distintas variaciones del algoritmo *Bellman-Ford* presentadas para buscar un camino que cumpla los requisitos de calidad de servicio en redes con tasa garantizada (*granted rate*), es decir, redes que utilicen un mecanismo de planificación tipo *WFQ* [Zha95] [BZ96] [Kes97] o *Deadline First* [GGPS96] [Ram95], ya que dichos mecanismos relacionan entre sí las distintas métricas de calidad de servicio y hacen posible que con valores pequeños de K se obtenga una alta probabilidad de obtener el camino de coste mínimo que cumpla los requisitos de calidad de servicio. Además, el empleo de estos mecanismos de gestión introducen otra importante característica, ya que, gracias a su uso, la reserva del pertinente ancho de banda necesario para la conexión permite garantizar el resto de las métricas. Este hecho es especialmente interesante si consideramos un coste que depende

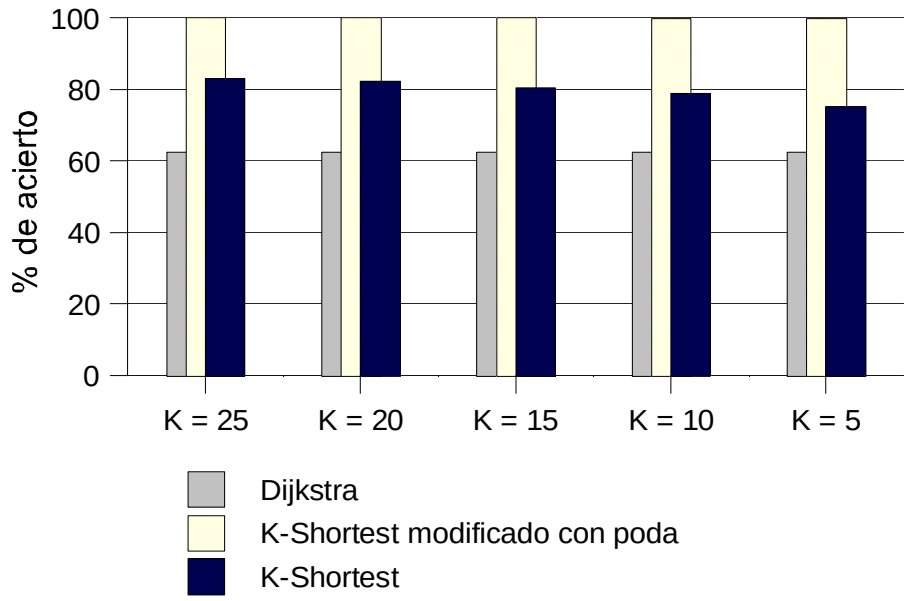


Figura 3.5: Comparación de los porcentajes de aciertos de distintos algoritmos de búsqueda de camino.

exclusivamente del ancho de banda empleado en los enlaces y buscamos maximizar el *throughput* de la red. Bajo estas circunstancias este algoritmo permitirá encontrar un camino de coste mínimo (lo que implica maximizar la utilización de la red) que además cumple los requisitos de calidad de servicio.

En la tabla 3.2 podemos ver un resumen comparativo de las características del algoritmo presentado frente a los anteriormente comentados.

	Dijkstra	K-Shortest Modificado	K-Shortest
$K = 25$	0,3 s.	4,05 s.	171,57 s.
$K = 20$	0,3 s.	3,856 s.	110,48 s.
$K = 15$	0,3 s.	3,63 s.	62,34 s.
$K = 10$	0,3 s.	2,92 s.	26,5 s.
$K = 5$	0,3 s.	2,54 s.	6,21 s.

Tabla 3.1: Resultados de los tiempos de cómputo para distintos valores de K

Algoritmo	Problema que resuelve	Información utilizada	Ventajas	Inconvenientes
CBF	Coste con restricciones de retardo	Global	Es capaz de encontrar siempre la solución	Tiempo de cómputo; una única restricción
Przygienda [Prz95]	Multiples restricciones	Global	Permite trabajar con múltiples restricciones	No garantiza camino óptimo
Wang y Crowcroft [WC96b]	Retardo con restricciones de retardo o retardo con restricciones de ancho de banda	Global	Sencillo de implementar	Sólo trabaja con ancho de banda y retardo; no admite coste
Ma y P. Steenkiste [MS97]	Multiples restricciones	Global	Permite trabajar con múltiples restricciones	Para poder trabajar con múltiples restricciones precisa métricas relacionadas; tiempos de cómputo
Guerin y Orda [GO97]	Retardo con restricciones de retardo o retardo con restricciones de ancho de banda	Imprecisa global	Diseñado para trabajar con datos imprecisos	Necesidad de asumir ciertas aproximaciones no necesariamente ciertas (distribución de los parámetros en los nodos)
Chen y Nahrstedt [CN98a]	Múltiples restricciones	Global	Capaz de trabajar con múltiples restricciones	No garantiza que se encontrará el camino; tiempos de cómputo aumentan con el número de restricciones utilizadas
Roginsky, Christesen y Srinivasan [RCS99]	Coste mínimo con restricciones con dos restricciones aditivas	Global	Capaz de trabajar con dos restricciones de tipo aditivo	Tiempos de cómputo
De Neve y Van Mieghem [Nb00]	Coste con restricciones de retardo (solución no dominante)	Global	Fácil de extender a multiples restricciones	No garantiza que se encontrará el camino
Algoritmo propuesto	Múltiples restricciones	Global	Tiempos de cómputo poco dependientes del número de restricciones a satisfacer, es fácil añadir nuevas restricciones	No garantiza que se encontrará el camino óptimo

Tabla 3.2: Comparativa entre los distintos algoritmos

3.3 Actualización no inmediata del estado de los enlaces

3.3.1 Introducción

Como ya se apuntó, el encaminamiento con *QoS* en origen exige conocer en cada nodo el estado de los enlaces que componen la red. En este sentido, un aspecto del encaminamiento, que recientemente está siendo estudiado en detalle, es el problema del encaminamiento cuando no disponemos en los nodos de información precisa del estado actual de la red [LO98] [GO99] [AGKT99]. Esta dificultad puede ser debida a muy diversas causas. En [LO98] se identifican cuatro fuentes de imprecisión:

1. Dinámica de la red: Algunas de las métricas (ancho de banda, retardo, etc.) asociadas a los enlaces varían con el tiempo. Es difícil (si no imposible) disponer de información actualizada en cada momento y en todos los nodos del estado de todos los enlaces de la red.

2. Agregación de la información: Esta permite agrupar y reducir la información transmitida sobre el estado de la red y disminuir, por lo tanto, la cantidad de información que es necesario tener disponible. Esto es importante, sobre todo en grandes redes, para minimizar la sobrecarga producida por el intercambio de información de actualización de estado, sin embargo, produce imprecisión en los datos que los nodos disponen sobre enlaces particulares.

3. Ocultación de la información: Por motivos de seguridad u otras razones, parte de la información del estado de la red puede estar oculta y, en consecuencia, resultar desconocida para los nodos.

4. Cálculo aproximado: Ciertos valores de los parámetros de la red no pueden ser estimados de forma precisa. Así, la información de la que se dispone no incluye sino valores aproximados de su valor real.

Además, existe una limitación física que impide el envío continuo de mensajes de actualización. Incluso aunque éstos se transmitan cada vez que se produce un cambio en la red, la sobrecarga puede llegar a ser muy elevada, colapsando los enlaces. En cualquier caso, siempre existiría una fuente de imprecisión debida a los propios retardos en la transmisión y el tiempo necesario para procesar la información de estado. Por lo

tanto, es imposible que los nodos posean una información completamente actualizada del estado de la red cuando se disponen a encaminar una llamada. Por ello, la decisión de escoger un determinado camino frente a otro estará sujeta a imprecisiones y siempre existirá la posibilidad de escoger un camino que realmente no posea suficientes recursos para acomodar la conexión cuando quizás existan otros que sí podrían llevarla a cabo con éxito. A fin de disminuir este efecto, se han propuesto diversas soluciones que presuponen que las métricas de *QoS* siguen una cierta distribución de probabilidad. Supuesto el conocimiento de estas funciones de distribución, Lorenz [LO98] proporciona un algoritmo capaz de encontrar el camino con la mayor probabilidad de éxito de completar la conexión. No obstante, esta solución conlleva también sus propias dificultades, ya que conocer a priori la forma de las funciones puede ser difícil (a veces imposible por el número de factores que influyen sobre ellas). Por ejemplo, la función de distribución para el ancho de banda disponible en un enlace (y esto es directamente extensible al retardo si se emplean mecanismos de gestión de colas tipo *WFQ* [Zha95]) depende del tipo de tráfico, de la matriz de tráfico (que a su vez depende de la hora del día), del ancho de banda del enlace y del algoritmo de encaminamiento, lo que dificulta establecer una aproximación para la función de distribución de ancho de banda.

Existirá, pues, una multitud de situaciones en las que resulta imposible estimar estas funciones de distribución, por lo que es necesario buscar mecanismos alternativos que permitan encaminar con información no actualizada del estado de la red. En este sentido, una manera sencilla de trabajar con imprecisión es acotar la variación máxima que dichos datos pueden presentar. Un ejemplo donde es fácil de implementar esta política se encuentra cuando la variación del ancho de banda disponible en el enlace e_{ij} ($BW_{ij} = C_{ij} - c_{ij}$, donde BW_{ij} es en ancho de banda disponible, C_{ij} es la capacidad del enlace y c_{ij} es en ancho de banda ocupado en el enlace) se actualiza mediante umbrales. Esto es, cada vez que BW_{ij} supera un determinado umbral de variación máxima se notifica al resto de los nodos de la red que se ha producido un cambio en el ancho de banda disponible. En este caso podemos acotar fácilmente la variación máxima de BW_{ij}. De hecho, existen varios trabajos donde se estudia el problema de la imprecisión en el conocimiento del ancho de banda disponible en los enlaces, que, aprovechando la acotación de la variación máxima de este parámetro, buscan el camino con mayor probabilidad de éxito de completar la conexión [CN99][Che99][AGKT99][AKW+99][Apo99].

Como se ha comentado, cuando se utiliza la actualización mediante umbrales es posible acotar dentro de qué rango se encuentra el ancho de banda disponible. Si éste se actualiza mediante un umbral proporcional, es decir, cuando el ancho de banda disponible

en el enlace e_{ij} varía más de un determinado porcentaje $Th * 100$ desde el último valor notificado BW_{ij}^{new}, siempre se garantiza que (despreciando el tiempo de transmisión y procesado de los mensajes de actualización) el ancho de banda realmente disponible en el enlace (BW_{ij}) estará acotado por la siguiente expresión:

$$BW_{ij}^{new} * (1 - Th) \leq BW_{ij} \leq BW_{ij}^{new} * (1 + Th) \tag{3.8}$$

con $Th \in [0, 1]$.

Aunque la actualización mediante umbrales proporcionales sea la política más eficiente (como se ha podido constatar en las pruebas y se ilustrará más tarde) no siempre se utiliza esta estrategia de actualización (de hecho, la estrategia de actualización por tiempos se ha utilizado ampliamente en muy diversos protocolos).

El problema principal de disparar el proceso de actualización basándose en las variaciones de actualización de un sólo parámetro es que, en principio, resulta desconocido el margen de evolución de los demás. Una solución a este problema consiste en estimar, a través de los sucesivos mensajes de actualización de estado, el intervalo de oscilación de los parámetros. Para esto es posible emplear el método propuesto por Chen y Nahrstedt en [CN99]. De acuerdo con esta solución, si el retardo del enlace se actualiza cada vez que llega un mensaje de actualización del ancho de banda, el intervalo de variación del retardo se podría estimar por la siguiente expresión:

$$\Delta D^{new} = \alpha * \Delta D^{old} + (1 - \alpha) * \left| D^{new} - D^{old} \right| \tag{3.9}$$

donde ΔD^{new} y ΔD^{old} son, respectivamente, la nueva y anterior estimaciones del intervalo de variación del retardo y $\alpha < 1$ es una constante que pondera el peso de las medidas anteriores. A fin de garantizar una alta probabilidad de que el valor esté dentro de dicho intervalo, la expresión (3.9) se modifica del modo:

$$\Delta D^{new} = \alpha * \Delta D^{old} + (1 - \alpha) * \beta * \left| D^{new} - D^{old} \right| \tag{3.10}$$

donde $\beta > 1$ es una constante que amplía la importancia de la última notificación aumentando la probabilidad de que el retardo esté dentro del intervalo $D^{new} - \Delta D^{new} \leq D \leq D^{new} + \Delta D^{new}$, a costa de una menor precisión en la acotación. Las cotas del resto de parámetros de calidad de servicio necesarios se pueden pueden aproximar de manera similar.

3.3.2 Algoritmos de asignación de costes

En este apartado se estudia el comportamiento de diversos algoritmos de asignación de costes en condiciones de imprecisión en los datos, buscando, como objetivo último, maximizar el aprovechamiento de la red con la menor sobrecarga posible de información de actualización. Así pues, el objetivo buscado es maximizar el *throughput* de la red (definido como la relación entre el ancho de banda ofrecido a la red y el que es capaz de procesar).

En el caso del tráfico telefónico, la amplia experiencia en el diseño de redes y la uniformidad del tráfico soportado, han permitido construir redes capaces de trabajar con una baja probabilidad de pérdida de llamada en condiciones normales. Una vez diseñadas las redes se añaden los algoritmos de encaminamiento que reaccionarán ante situaciones de sobrecarga de la red o ante posibles anomalías como fallos puntuales en los enlaces. En cambio, en las presentes y futuras redes multimedia la heterogeneidad del tráfico hace difícil este tipo de diseño. Añadidamente, debido a esta diversidad del tráfico, parece lógico pensar que las futuras redes presentarán escenarios muy dinámicos, con multitud de cambios de estado en períodos de tiempo muy breve y, por lo tanto, existirá un elevado grado de imprecisión por la imposibilidad de actualizar de forma inmediata todas las variaciones que se produzcan.

Debido al alto grado de imprecisión de los datos que cabe esperar encontrar en este tipo de redes, creemos que es necesario emplear algoritmos sencillos de asignación de coste en lugar de complejos algoritmos con multitud de parámetros, cada uno de los cuales sujeto a imprecisión, circunstancia que aumentaría el grado de desconocimiento con el que trabajaría el algoritmo de encaminamiento. En concreto, los algoritmos que hemos empleado en este estudio han sido:

Algoritmo *Shortest-Widest* (*SW*) [WC96b]: De acuerdo con esta política se selecciona el camino "mas ancho" (*Widest*) o con mayor ancho de banda libre; en caso de existir dos caminos con igual ancho de banda disponible se selecciona el de menor número de saltos (*Shortest*). En este caso, el coste del enlace viene dado por la siguiente expresión cóncava: $Coste_{ij} = BW_{ij}$. El coste de un camino p entre los nodos $o - d$ (formado por los enlaces $e_{oa}, \cdots, e_{ij}, \cdots, e_{ld}$) vendrá dado por la expresión $Coste_p = \min(Coste_{oa}, \cdots, Coste_{ij}, \cdots, Coste_{ld})$. Formalmente, de todos los posible caminos se escoge el camino de coste máximo. Si dos caminos tienen el mismo coste se selecciona el camino con menor número de nodos intermedios. Este

algoritmo (o sus variantes) ha sido ampliamente utilizado en las redes de comunicaciones bajo el nombre de camino menos cargado (*Least Loaded Routing*). Un ejemplo comercial de este tipo de algoritmos es el algoritmo *Real Time Network Routing*[ACFH91] utilizado por AT&T en su red de larga distancia.

Algoritmo *Widest-shortest* (*WS*)[Ma98]: Supondría la versión complementaria del anterior. Así, se selecciona el camino con menor número de nodos capaz de transportar la conexión (es decir, el coste de todos los enlaces es el mismo e igual a 1, buscándose el camino con coste mínimo). En el caso de existir dos caminos con igual número de nodos se selecciona el de mayor ancho de banda libre.

Algoritmo *Exp-MC* (Exponencial) [Gaw95]: El coste $Coste_{ij}$ asignado a cada enlace e_{ij} viene dado por una función exponencial:

$$Coste_{ij} = a^{\frac{c_{ij}}{C_{ij}}} \tag{3.11}$$

donde a es una constante, c_{ij} es el ancho de banda utilizado en el enlace e_{ij} y C_{ij} es la capacidad del enlace e_{ij}. El coste total del camino p es la suma de los costes parciales $Coste_p = \sum_{e_{ij} \in p} Coste_{ij}$. Para poder elegir un camino, además, el coste total debe satisfacer la siguiente condición $Coste_p \leq a$. Este algoritmo, ya descrito anteriormente, es una implementación práctica de los algoritmos exponenciales [Plo95]. La importancia de estos algoritmos estriba en que bajo determinadas circunstancias tienen un comportamiento muy cercano al óptimo. Sin embargo, presentan como inconveniente la necesidad de establecer heurísticamente la constante a.

Algoritmo Lineal: El coste $Coste_{ij}$ de cada enlace e_{ij} viene dado por una función lineal:

$$Coste_{ij} = \frac{c_{ij}}{C_{ij}} \tag{3.12}$$

El coste total del camino viene dado por la suma de los costes de todos los enlaces de que consta el camino. Este algoritmo presenta como principal ventaja su extrema sencillez. Al ser una función lineal, es posible enviar directamente el valor del coste dentro de los mensajes de actualización sin perder ninguna precisión.

Algoritmo Hiperbólico [Ma98] [Ste95]: El coste $Coste_{ij}$ de cada enlace e_{ij} viene dado por la expresión:

$$Coste_{ij} = \frac{1}{BW_{ij}} \tag{3.13}$$

donde $BW_{ij} = C_{ij} - c_{ij}$ es el ancho de banda residual. Algoritmos de este tipo ya han sido probados extensamente y, dada su sencillez, es interesante estudiar su comportamiento cuando existe imprecisión en los datos.

Algoritmo *Shortest Safest* (*SS*)[AGKT99][Apo99]: Este algoritmo estima la probabilidad de que el enlace tenga suficiente ancho de banda libre para portar la conexión, suponiendo que la función de distribución del ancho de banda es uniforme en el intervalo $BW_{ij}^{new} * (1 - Th) \leq BW_{ij} \leq BW_{ij}^{new} * (1 + Th)$. De hecho Apostolopoulos realiza un estudio acerca del rendimiento del algoritmo cuando la función de distribución de probabilidad es uniforme y cuando ésta se obtiene mediante el histograma del ancho de banda residual medido en los enlaces, no obtiendo diferencias significativas. Por ello y por motivos de simplicidad, en este trabajo se ha supuesto que la función de distribución es uniforme. Una vez estimada la probabilidad de que los enlaces tengan suficiente ancho de banda libre para acomodar la conexión, se procede a buscar el camino con mayor probabilidad de éxito. Para asignar el coste, se aplica la trasformación $Coste_{ij} = -\log(P_{ij})$, donde P_{ij} es la probabilidad de que el enlace e_{ij} tenga suficiente ancho de banda disponible para acomodar la conexión. El camino con mayor seguridad (y, por lo tanto, con mayor probabilidad de éxito) será el camino con menor coste una vez aplicada la transformación. En el caso de que existan dos caminos con igual coste, se escoge el camino de menor número de saltos, de forma similar al algoritmo *SW*. Para nuestro estudio vamos a probar el algoritmo *SS* con la denominada seguridad $s = 0$, ya que como demuestra el trabajo [AGKT99], es la variante del algoritmo que mejor rendimiento presenta. La seguridad $s = 0$ hace referencia a que se eliminan de la búsqueda aquellos enlaces en los que se sabe con total seguridad que no existe suficiente ancho de banda disponible, es decir, $BW_{ij}^{new} * (1 + Th) < b_k$, donde b_k es el ancho de banda solicitado por la conexión. La inclusión de este algoritmo en la comparativa permite cotejar el resultado de los algoritmos anteriores con otro basado en cálculo de probabilidades.

Ciertamente, aunque es posible escoger otros algoritmos, se han seleccionado éstos en parte por su sencillez, que los hace indicados para el cálculo del coste en grandes redes donde la velocidad de cálculo es importante. En todos los casos el cumplimiento de los parámetros *QoS* de la conexión se ha obtenido mediante el empleo de algoritmos de búsqueda de camino con restricciones (*K-Shortest path* modificado), tal y como se definieron en el apartado anterior.

En los siguientes apartados, se estudia el comportamiento de estos algoritmos bajo distintas circunstancias.

3.3.3 Algoritmos de búsquedas multicamino

Una solución para mejorar el rendimiento del encaminamiento cuando se trabaja en condiciones de imprecisión, y presuponiendo que el tiempo de establecimiento de conexión es despreciable frente al tiempo de variación del estado de la red, es intentar establecer la conexión por varios caminos de forma simultánea, a fin de aumentar la probabilidad de establecer con éxito la llamada. En la literatura se han propuesto diversas implementaciones de este tipo de búsqueda de camino (también denominado encaminamiento multicamino o *multipath routing*) [Sha96][CN99] [TPBO99]. Este tipo de solución presenta, como principal inconveniente, la sobrecarga introducida en la red por reservar recursos que posteriormente no van a ser usados.

El consumo de recursos producido por estas técnicas se manifiesta de tres maneras: La primera, mediante el envío de la señalización para reservar los recursos en los distintos caminos; dicha señalización consume ancho de banda y tiempo de proceso. La segunda manera es la que resulta del ancho de banda que ya ha sido reservado en un camino que posteriormente no va a ser usado. Hasta que no se liberan los recursos reservados a lo largo de dicho camino ninguna otra conexión puede solicitarlos. Y por último, se ha de considerar el ancho de banda y el tiempo de proceso requerido por la señalización necesaria para liberar los recursos reservados en los caminos no usados. Si bien puede llegar a conformar una herramienta poderosa, todas estas circunstancias obligan a limitar el empleo de la búsqueda multicamino.

En este trabajo se estudian varias alternativas de búsqueda multicamino, intentando restringir el número de caminos que se buscan de forma simultánea. Para ello vamos a distinguir tres tipos de criterios posibles a la hora de definir la evolución de los parámetros de coste [ACS00a][ACS00b][ACS00c]:

Optimista: Con esta asunción, se supone que el ancho de banda ha evolucionado hacia la mejor situación posible. Es decir, si en el último mensaje de actualización se notificó que el ancho de banda disponible en el enlace es BW_{ij}^{new}, a efectos de cálculo se supone que el ancho de banda disponible en el enlace es $BW_{ij} = BW_{ij}^{new} * (1+Th)$. Este criterio se caracteriza por presentar un mayor número de intentos fallidos

(llamadas que se intentan y que posteriormente son rechazadas por los controles de admisión) que los siguientes.

Pesimista: Conforme a este criterio, se supone que el ancho de banda ha evolucionado hacia la peor situación posible. Así, para el algoritmo de encaminamiento se asume que el ancho de banda disponible en el enlace es $BW_{ij} = BW_{ij}^{new} * (1 - Th)$. Su principal virtud es que su tasa de rechazos es nula, ya que los caminos encontrados por el algoritmo de búsqueda con este criterio siempre tienen suficiente ancho de banda disponible.

Medio: El camino medio supone que no ha habido ningún cambio en el recurso disponible desde la última notificación ($BW_{ij} = BW_{ij}^{new}$).

Las estrategias multicamino empleadas se basan en la utilización de los distintos tipos de camino identificados al aplicar los criterios optimista, pesimista y medio. Además, dentro de la búsqueda multicamino vamos a distinguir dos variantes que son:

Búsqueda secuencial: Inicialmente se prueba el enfoque optimista. Si al tratar de establecer la conexión sobre el camino encontrado éste resulta no ser válido, se prueba con el camino obtenido mediante el criterio medio, y si la conexión vuelve a ser rechazada, se emplea como último recurso el pesimista.

Búsqueda en paralelo: Se prueban simultáneamente los caminos obtenidos por las tres alternativas. Durante la fase de exploración del camino, se recolecta información del estado real de cada uno de los caminos y el nodo destino escoge la solución de menor coste de todas las que culminaron con éxito.

Es evidente que la estrategia multicamino paralelo presenta un mayor coste, a la hora de establecer una ruta, que la búsqueda secuencial.

3.4 Pruebas y resultados

3.4.1 Entorno de pruebas y medidas de rendimiento

En las pruebas se ha empleado la red utilizada por un proveedor de Internet o *ISP* (*Internet Service Provider*). En concreto, se consideró la red de la empresa MCI,

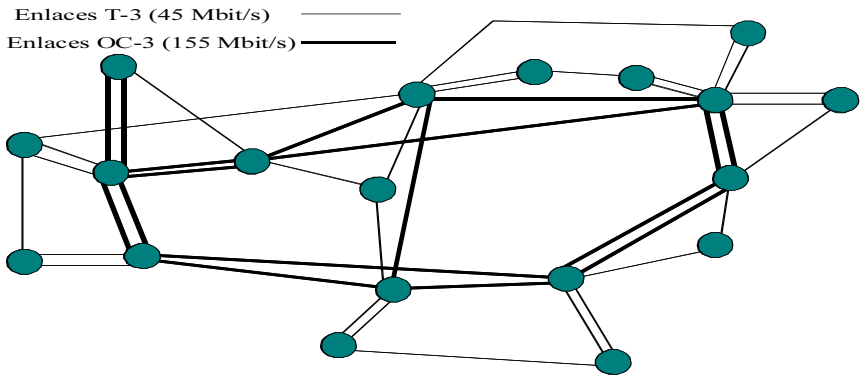


Figura 3.6: Topología de la red MCI empleada en las pruebas.

cuya topología, con diversas variantes, ya ha sido empleada en varios estudios anteriores [Ma98][Che99][Apo99]. La estructura de esta red se ha ilustrado en la figura 3.6. A fin de comprobar cómo afecta el cambio de topología al rendimiento final de los algoritmos, también emplearemos una variante de la red anterior donde se han eliminado los enlaces en paralelo (entre dos nodos cualesquiera no puede haber más que un enlace que los conecte), limitándose además la capacidad de todos los enlaces a un ancho de banda de 50 Mbits/s.

Uno de los problemas que han surgido en la simulación en condiciones de imprecisión, es que el rendimiento final presenta una fuerte dependencia con respecto de la matriz de tráfico empleada. Esta circunstancia puede dar lugar a cientos de resultados parciales, en los que se demuestra, sólo aparentemente, el mejor rendimiento de una función de coste frente al resto. Por ello, se ha optado por una matriz de tráfico uniforme donde la probabilidad de seleccionar un destino es igual para todos los nodos, los cuales generan tráfico con la misma probabilidad y las mismas características. Esto, unido a una velocidad común de los enlaces, pretende reducir el efecto de la topología sobre los resultados para tratar de aislar los efectos de los algoritmos.

Para estudiar el rendimiento de los algoritmos de encaminamiento, en las redes telefónicas tradicionalmente se ha empleado la probabilidad de bloqueo de llamada, es decir, la probabilidad, bajo una determinada tasa de tráfico, de que una llamada sea rechazada y no pueda completar la conexión. Esta medida es adecuada en este tipo de redes en las que todas las llamadas son similares y solicitan los mismos recursos. Sin embargo, no es conveniente para redes donde el tráfico es heterogéneo y solicitan recursos de la red muy dispares. Por ejemplo, si se utilizara como medida del rendimiento

de la red esta probabilidad de bloqueo, interesaría diseñar algoritmos que aceptasen únicamente llamadas que precisan pocos recursos, rechazando las llamadas que consumen demasiados.

Para evaluar el comportamiento de la red cuando hay distintos tipos de tráfico con requerimientos distintos, Ma [Ma98] propone utilizar como medida del rendimiento la probabilidad de bloqueo del ancho de banda que se calcula mediante la expresión (3.14)

$$Pr = \frac{BW_{rec}}{BW_{of}} \tag{3.14}$$

donde Pr es la citada probabilidad de pérdida de ancho de banda, BW_{rec} el ancho de banda rechazado y BW_{of} el ancho de banda total solicitado a la red. Cuando una llamada que precisa un ancho de banda b_k, solicita recursos a la red, el valor BW_{of} se incrementa en b_k unidades. Por contra, sólo si esta llamada es rechazada, BW_{rec} se incrementa en b_k unidades.

Así pues, uno de los datos utilizados para comparar el rendimiento de los diversos algoritmos es la capacidad para encaminar el mayor ancho de banda posible con respecto al ancho de banda ofrecido.

La sobrecarga introducida por el encaminamiento se barema mediante lo que podemos denominar tasa de actualizaciones. Esta tasa refleja la cantidad de mensajes de actualización de estado de los enlaces que se han generado en el sistema por unidad de tiempo. Interesa que esta tasa sea lo más baja posible por dos razones: la primera es que, cuanto menor sea el número de actualizaciones, menor será el ancho de banda consumido por los mensajes de actualización y mayor el disponible para las llamadas (*overload*). La segunda de las razones se debe a que cuando llega un mensaje de actualización a un nodo este debe recalcular la tabla de rutas para todos los nodos, lo que implica una fuerte carga computacional (*overprocessing*).

En este sentido, en una red donde el parámetro a minimizar es simplemente el número de saltos, se cumple que si el camino más corto para ir de el nodo s al t es $s \rightarrow i \rightarrow j \rightarrow t$ entonces el camino más corto para ir del nodo s al j será $s \rightarrow i \rightarrow j$. Basándose en esta circunstancia, el algoritmo de *Dijkstra* obtiene rápidamente todo el árbol de rutas. Sin embargo, esta condición no es cierta si se emplea encaminamiento con calidad de servicio, lo que obliga a obtener la tabla de rutas nodo a nodo con un coste computacionalmente mayor que el algoritmo de *Dijkstra* básico. Esta es otra de las razones que hacen deseable que los algoritmos de encaminamiento generen el menor

número posible de actualizaciones.

Por lo que se refiere a los parámetros de tráfico utilizados en las pruebas, se considera una duración de las llamadas dada por una distribución exponencial de valor medio 1200 segundos, mientras que el tráfico ofrecido a la red sigue una distribución de *Poisson*. El ancho de banda solicitado por las llamadas presenta una distribución uniforme entre 1 Mbits/s y 5 Mbits/s. Emplear un simulador orientado a llamada implica suponer que el tráfico es de tipo *CBR* (*Constant Bit Rate*) o, en su defecto, despreciar la variación de la tasa durante la conexión. La suposición de que el tráfico sea *CBR* se acerca mucho a la situación actual, en la que la mayor parte del tráfico que precisa calidad de servicio utiliza una tasa binaria constante. En cualquier caso, el tráfico *VBR* (*Variable Bit Rate*) se puede aproximar como caso peor por un *CBR* con una tasa igual a la máxima. La duración de las simulaciones se ha limitado a 10^6 segundos para cada caso.

Las redes telefónicas trabajan con tasas de pérdida de llamada de hasta un 2%, gracias a la amplia experiencia que existe en el diseño de este tipo de redes y a la homogeneidad del tráfico cursado. Sin embargo, es de esperar que las redes multimedia (al menos en sus comienzos) trabajen con mayores probabilidades de pérdida. Por ello, se ha estudiado el comportamiento de los algoritmos hasta un rango de pérdida cercano al 10%. Por otro lado, el número máximo de enlaces que puede tener un camino se ha limitado a 15.

3.4.2 Comparación de estrategias de actualización

En este apartado se realiza un estudio sobre el rendimiento de las distintas políticas de actualización comúnmente utilizadas [ACS00d]. Para este experimento se ha utilizado la red simplificada, ya comentada, donde todos los enlaces tienen la misma capacidad de 50 Mbits/s.

La metodología de trabajo empleada se basa en comparar el rendimiento obtenido cuando actualizamos el estado de los enlaces mediante diversas alternativas. Los métodos usados son la actualización mediante umbrales proporcionales, mediante temporizadores y mediante umbrales fijos.

- Con la actualización por umbral proporcional, un nodo genera un mensaje de actualización del estado del enlace si la variación del ancho de banda del enlace e_{ij}

cumple la siguiente relación:

$$\left| \frac{BW_{ij} - BW_{ij}^{new}}{BW_{ij}^{new}} \right| > Th \tag{3.15}$$

donde BW_{ij} es el ancho de banda que realmente tiene el enlace, BW_{ij}^{new} es la última notificación que se realizó del ancho de banda disponible y Th es el valor del umbral en tanto por uno.

- La actualización mediante temporizador genera un mensaje de actualización cuando ha transcurrido un determinado intervalo de tiempo desde el último mensaje de actualización. Así, en el instante T^{actual} se envía un mensaje de actualización si:

$$T^{actual} = T^{anterior} + \Delta T \tag{3.16}$$

donde T^{actual} es el tiempo en el instante actual, $T^{anterior}$ es instante de tiempo en el cual se generó el último mensaje de actualización y ΔT es el valor del intervalo de tiempo entre mensajes de actualización.

- Por último, la actualización mediante umbrales fijos divide el ancho de banda total del enlace en una serie de franjas de igual tamaño con un ancho de banda ΔBW cada una de ellas. Se producirá un mensaje de actualización cada vez que el ancho de banda disponible en el enlace pasa de una franja a otra, del modo:

$$\left| BW_{ij} - BW_{ij}^{new} \right| > \Delta BW \tag{3.17}$$

Como función de coste para estos experimentos se ha utilizado la función lineal ($Coste_{ij} = \frac{c_{ij}}{C_{ij}}$).

En este experimento comparamos los rendimientos obtenidos al actualizar mediante umbrales de 40% y 80% ($Th = 0,4$ y $0,8$), con respecto a los que resultan de la actualización mediante temporizadores de 60 y 240 segundos, así como a los obtenidos mediante la actualización por umbrales fijos de 5 y 10 Mbits/s.

Para evaluar el rendimiento se tienen en cuenta tanto la probabilidad de bloqueo de llamada como la tasa de actualizaciones requerida para alcanzar dicha probabilidad de bloqueo. En las figuras 3.7 y 3.8 podemos ver, respectivamente, la probabilidad de bloqueo y la tasa de actualizaciones obtenidas para cada uno de los distintos métodos de actualización. Las figuras evidencian que la estrategia más interesante es la actualización mediante umbral proporcional, ya que para una probabilidad de bloqueo similar presenta una menor tasa de actualización. Este método ofrece un crecimiento lineal de la tasa de

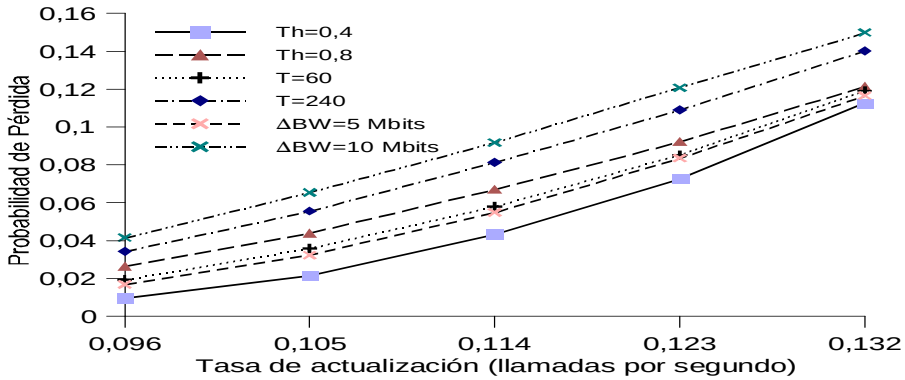


Figura 3.7: Probabilidad de pérdida de ancho de banda para los distintos métodos de actualización. Enlaces limitados a 50 Mbits/s.

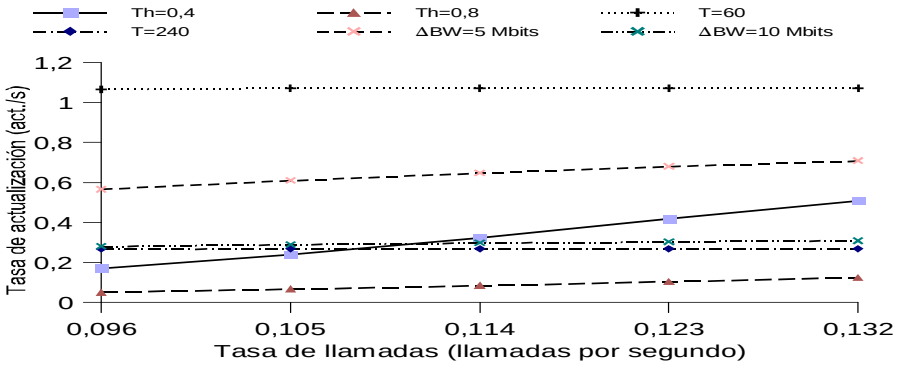


Figura 3.8: Tasa de actualización de los distintos métodos de actualización. Enlaces limitados a 50 Mbits/s.

actualización con respecto a la tasa de llamadas, al contrario de las otras técnicas, que presentan un comportamiento constante o casi constante. No obstante, para la región normal de funcionamiento, la actualización mediante umbral proporcional presenta un mejor rendimiento en la relación *probabilidad de pérdida/tasa de actualización*. Este fenómeno es claramente visible si se observa el comportamiento cuando el valor del umbral de actualización es del 40% y lo comparamos con el resto de los resultados. Para bajas tasas de tráfico, en las que el sistema ofrece pérdidas menores de un 5%, la actualización mediante umbral del 40% presenta la probabilidad de pérdida más baja con una muy baja tasa de actualización, solo superada por el umbral del 80%.

Este mejor comportamiento de la actualización mediante umbral proporcional frente a la actualización mediante temporizador o mediante umbral fijo es fácil de explicar. La actualización mediante temporizador renueva el estado de los enlaces a intervalos fijos

de tiempo, se hayan producido o no cambios importantes en el estado de los enlaces, a diferencia de la actualización mediante umbrales. Por otro lado, el empleo de umbrales fijos penaliza por igual a todos los rangos de ocupación de los enlaces, en contraste con la actualización por umbral proporcional, que aumenta la tasa de actualización cuanto mayor es la ocupación del enlace y, por lo tanto, más crítica es la disponibilidad de recursos.

Existe la posibilidad de actualizar el estado de los enlaces en función del ancho de banda ocupado en lugar del ancho de banda residual, es decir, en lugar de emplear la condición dada por la ecuación (3.15) se emplea la siguiente expresión:

$$\left| \frac{c_{ij} - c_{ij}^{old}}{c_{ij}^{old}} \right| > Th \tag{3.18}$$

donde c_{ij} es el ancho de banda ocupado en el enlace y c_{ij}^{old} es el ancho de banda ocupado que había en el último mensaje de actualización, cumpliéndose que $c_{ij} = C_{ij} - BW_{ij}$.

En este caso, conforme aumenta la ocupación del enlace disminuye la tasa de actualización. Los resultados obtenidos al comparar los métodos de actualización en función del ancho de banda disponible y ocupado se incluyen en las figuras 3.9 y 3.10. Como es posible observar, la actualización en función del ancho de banda disponible en el enlace proporciona un mejor rendimiento. La razón de esto radica en que cuanto menor sea el ancho de banda disponible en los enlaces más actualizaciones se reciben del mismo. A esto hay que añadir que la actualización mediante la variación del ancho de banda ocupado provoca que, conforme aumenta la saturación de los enlaces (y por lo tanto la probabilidad de perder llamadas) disminuya la tasa de actualización, causa por la que aparecen fuertes pérdidas a altas tasas de tráfico ofrecido. Por otro lado, cuando la red está poco cargada y, por ende, la probabilidad de perder llamadas es pequeña, la actualización basada en el ancho de banda ocupado genera una gran cantidad de mensajes de actualización que, por otro lado, no introducen mejoras en las pérdidas.

Estos resultados justifican que el resto de los experimentos se realicen empleando un umbral proporcional referido al ancho de banda residual, puesto que es el mecanismo de actualización de la información que mejor relación *probabilidad de pérdida/tasa de actualización* presenta en cualquier rango de funcionamiento.

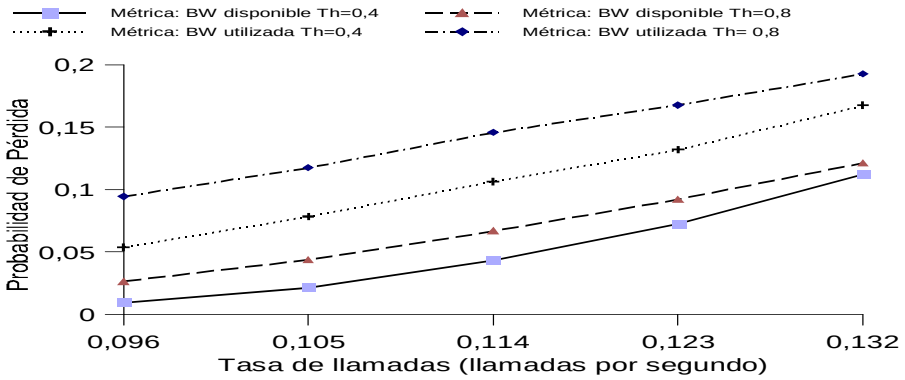


Figura 3.9: Comparación de la probabilidad de pérdida cuando se actualiza el estado de la red mediante un umbral proporcional relativo al ancho de banda disponible y al ancho de banda ocupado.

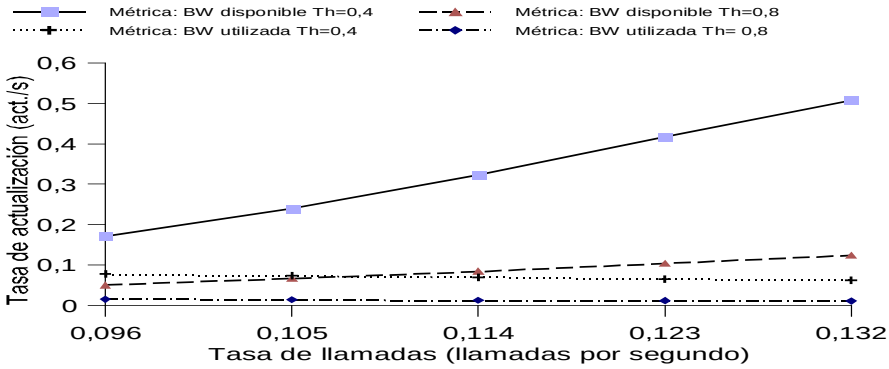


Figura 3.10: Comparación de la tasa de mensajes de actualización cuando se actualiza el estado de la red mediante umbral proporcional relativo al ancho de banda disponible y al ancho de banda ocupado.

3.4.3 Estudio del rendimiento de los algoritmos de asignación de coste

En este apartado se estudia y compara el rendimiento de los distintos algoritmos de asignación de costes en presencia de imprecisión.

En las figuras 3.11 y 3.12 podemos ver, respectivamente, el comportamiento de los distintos algoritmos cuando se emplean como banco de pruebas la red MCI simplificada y la red MCI real. Como se puede apreciar, los algoritmos que presentan un mejor comportamiento son el ***WS***, el **Hiperbólico** y el **Lineal**, siendo el ***SW*** es el algoritmo que peor rendimiento exhibe. También es posible apreciar que el comportamiento del

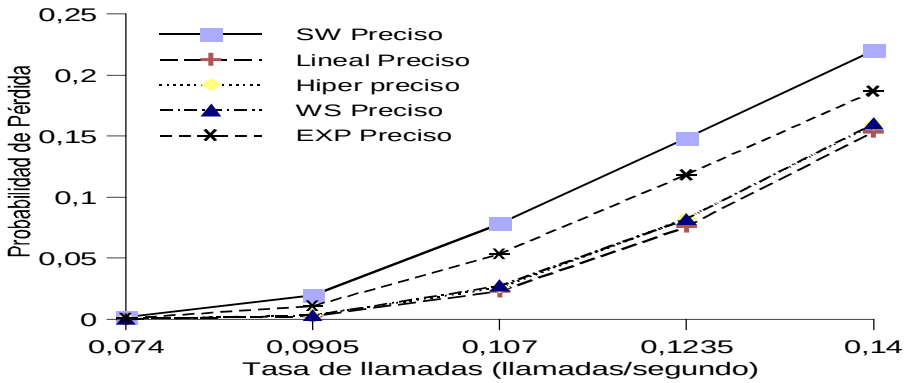


Figura 3.11: Probabilidades de pérdida de ancho de banda para todos los algoritmos con un conocimiento preciso de la red. Enlaces limitados a 50 Mbits/s.

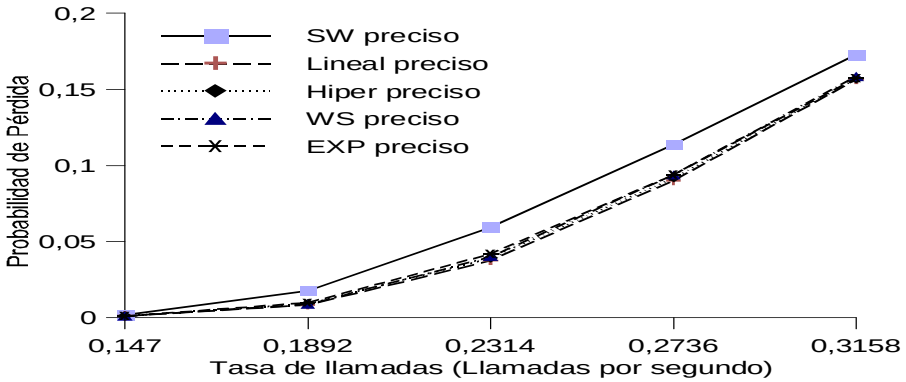


Figura 3.12: Probabilidades de pérdida de ancho de banda para todos los algoritmos con un conocimiento preciso de la red. Red MCI real

algoritmo **Exponencial** presenta una fuerte dependencia de la topología, con lo que es posible ajustar el comportamiento de esta función modificando el valor de su parámetro a. El valor escogido para las pruebas ha sido $a = 9,4\ 10^5$, que es el mismo que utiliza Gawlick [Gaw95]. El algoritmo ***SS*** no se ha incluido en este primer estudio comparativo puesto que su utilización sólo tiene sentido si existe imprecisión en los datos. De las figuras se deduce que la homogeneización de la velocidad de los enlaces permite distinguir más claramente el comportamiento de las distintas funciones de coste.

A fin de comprobar el rendimiento de los distintos algoritmos en el caso de que exista imprecisión en los datos se han repetido los experimentos anteriores con presencia de imprecisión. Para ello, se eligió un umbral de actualización del ancho de banda del 80% ($Th = 0,8$), es decir, se actualiza el estado de un enlace cuando la variación de su ancho de banda residual excede el 80% del último valor notificado. Los resultados

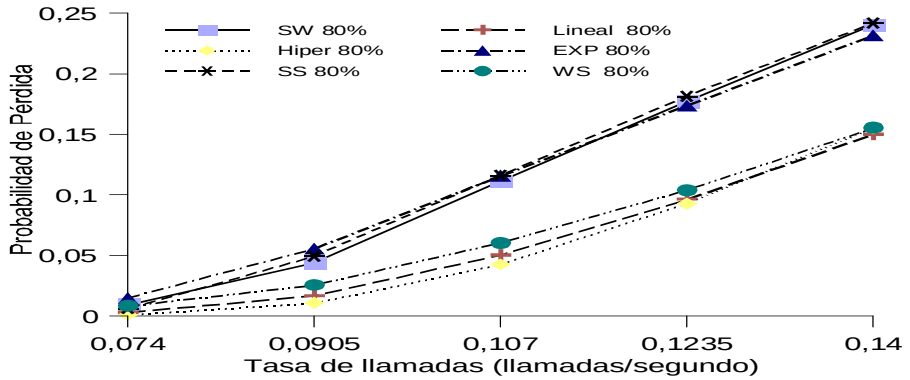


Figura 3.13: Probabilidades de pérdida de ancho de banda para todos los algoritmos con un conocimiento impreciso de la red y un umbral de actualización del 80%. Enlaces limitados a 50 Mbit/s.

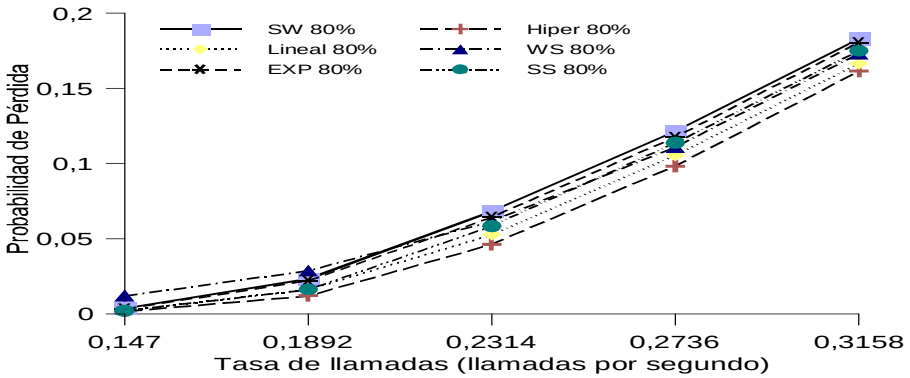


Figura 3.14: Probabilidades de pérdida de ancho de banda para todos los algoritmos con un conocimiento impreciso de la red y un umbral de actualización del 80%. Red MCI real.

de este experimento se muestran en las figuras 3.13 y 3.14. Como es posible apreciar, los mejores resultados se obtienen con las funciones de coste ***WS***, **Hiperbólico**, ***SS*** y **Lineal**. El algoritmo ***SS*** presenta un excelente rendimiento para bajas cargas de tráfico. Sin embargo, su comportamiento para altas cargas se degrada.

Como se ha señalado anteriormente, el segundo parámetro a analizar en este estudio es la tasa de actualizaciones. En función de esta tasa, puede llegar a ser preferible escoger un determinado algoritmo que si bien tenga una probabilidad de pérdida de ancho de banda mayor, presente una menor tasa de actualización. El comportamiento de dicha tasa (que repite a grandes rasgos el de las pérdidas), se puede observar para cada algoritmo en las figuras 3.15 (para la red simplificada) y 3.16 (para la red MCI).

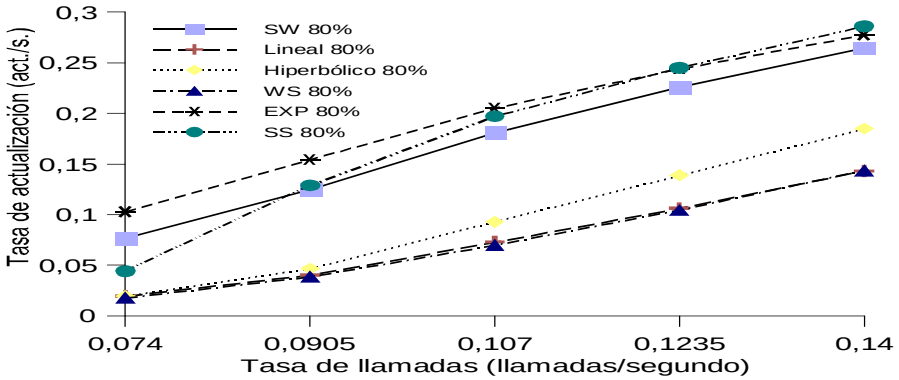


Figura 3.15: Tasa de actualización para todos los algoritmos con un conocimiento impreciso de la red y un umbral de actualización del 80%. Enlaces limitados a 50 Mbit/s.

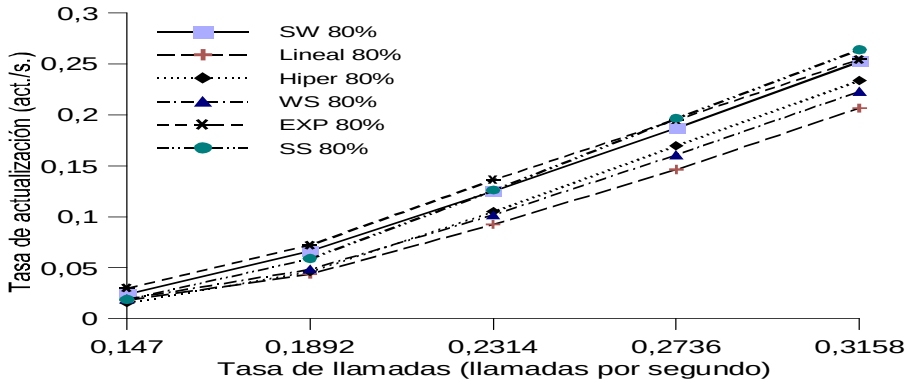


Figura 3.16: Tasa de actualización para todos los algoritmos con un conocimiento impreciso de la red y un umbral de actualización del 80%. Red MCI real.

A la vista de estos resultados es lógico centrar de momento nuestro estudio en los algoritmos **Hiperbólico**, **Lineal**, ***WS*** y ***SS***, los tres primeros debido a su rendimiento tanto en términos de probabilidad de pérdida de ancho de banda como de la tasa de actualización, y el cuarto por su excelente rendimiento a bajas tasas y a modo de comparación, en la medida en que representa una implementación práctica y realista de los trabajos realizados anteriormente por Guerin y Orda [GO97] [GO99] cuando existe imprecisión en los datos.

3.4.3.1 Efecto del umbral de actualización

El siguiente aspecto a estudiar es la repercusión del valor del umbral en el rendimiento de los distintos algoritmos; con este objeto se estudiará el comportamiento de

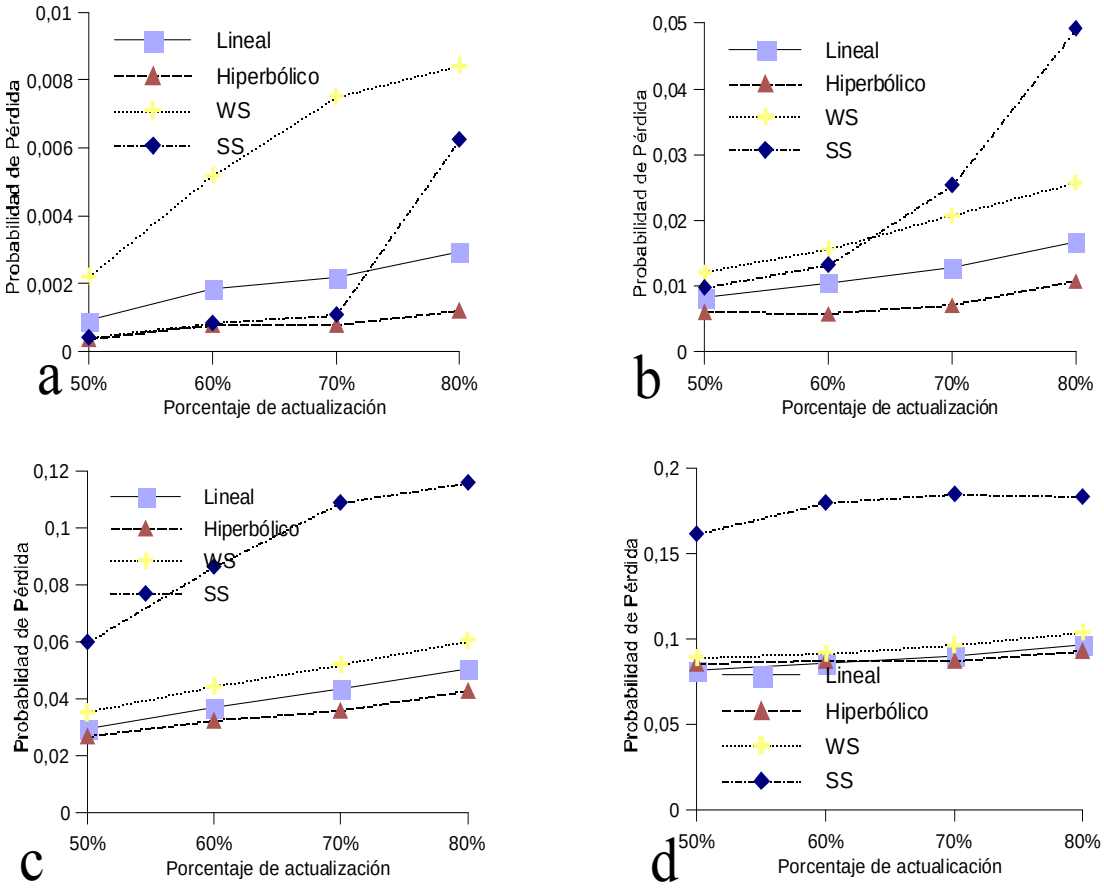


Figura 3.17: Probabilidad de pérdida de ancho de banda en función del porcentaje para: tasa de llamadas 0,074 (a), tasa de llamadas 0,0905 (b), tasa de llamadas 0,107 (c) y tasa de llamadas 0,1235 (d). Enlaces limitados a 50 Mbit/s.

éstos en función del umbral de actualización.

De la figura 3.17 se puede colegir que el comportamiento de los distintos algoritmos se degrada al aumentar el umbral de actualización. Sin embargo, esta degradación depende tanto del algoritmo como de la tasa de tráfico. Así pues, tanto el algoritmo **Lineal** como el **Hiperbólico** presentan un comportamiento poco dependiente del umbral en todas las tasas mientras que los resultados del algoritmo ***SS*** dependen fuertemente del umbral. Este comportamiento es debido a que el ***SS*** trabaja con probabilidades de éxito, lo que provoca que al aumentar el umbral se incremente, de forma no lineal (no hay que olvidar que la probabilidad es multiplicativa) la incertidumbre y, por lo tanto, las pérdidas. El algoritmo ***SS***, al igual que el ***SW***, busca balancear el tráfico ofrecido a la red, lo que hace que su rendimiento disminuya al aumentar la tasa de tráfico, en

tanto que, en una red cargada, los algoritmos que priman la distribución equitativa del tráfico entre los distintos caminos presentan un peor comportamiento que aquellos que buscan conservar recursos. Además, al aumentar el umbral de incertidumbre, ***SS*** y ***SW*** tienden a provocar bruscas oscilaciones del tráfico enviado por los distintos caminos cada vez que que llega un mensaje de actualización, siendo tanto mayor la oscilación cuanto mayor sea el umbral.

El pobre rendimiento del algoritmo ***WS*** frente al resto para bajas cargas se justifica en su modo de funcionamiento, puesto que el primer objetivo de este algoritmo es la conservación de recursos (por ello escoge siempre el camino con menor número de enlaces como primera alternativa), mientras que la distribución de la carga entre los distintos caminos posibles aparece como un objetivo secundario. Para tasas bajas, resulta de mayor interés balancear o distribuir paritariamente el tráfico entre los distintos caminos, ya que existe abundancia de medios. Intentar conservar libres los recursos hará que algunos enlaces se saturen mientras que otros se encuentren aún completamente libres. Conforme aumenta la carga de tráfico ofrecida a la red esta situación cambia, y conservar recursos es una política adecuada, ya que rechazar una llamada que consuma muchos medios podría permitir en un futuro próximo conectar dos llamadas que consuman menos recursos. Los resultados obtenidos aconsejan usar la función **Hiperbólica** para tráficos bajos-medios y la **Lineal** para tráficos medios-altos, ya que estas funciones proponen un equilibrio entre balanceo del tráfico y conservación de recursos.

Hasta aquí se ha estudiado el comportamiento de estos algoritmos con respecto de la probabilidad de pérdida de ancho de banda. Ahora es necesario analizar su rendimiento en relación a la tasa de actualizaciones. En la figura 3.18 se ha ilustrado el comportamiento de los distintos algoritmos en función del porcentaje de actualización en la red simplificada. Se observa que los algoritmos **Lineal** y ***WS*** son los que mejor comportamiento presentan. Esto es debido, entre otras causas, a la tendencia a emplear los caminos más cortos frente a los más largos. Al tener estos caminos un menor número de enlaces y al ser menos propensos a distribuir el tráfico entre los distintos caminos, se genera un menor número de actualizaciones. Curiosamente, el comportamiento del algoritmo ***SS*** difiere para cargas bajas del resto de los algoritmos estudiados, puesto que mientras que, en éstos al aumentar el porcentaje disminuye la tasa de actualización, en aquel se puede dar el fenómeno contrario. Esto es debido, fundamentalmente, a bruscas redistribuciones del tráfico que provocan inestabilidades que obligan a nuevas actualizaciones.

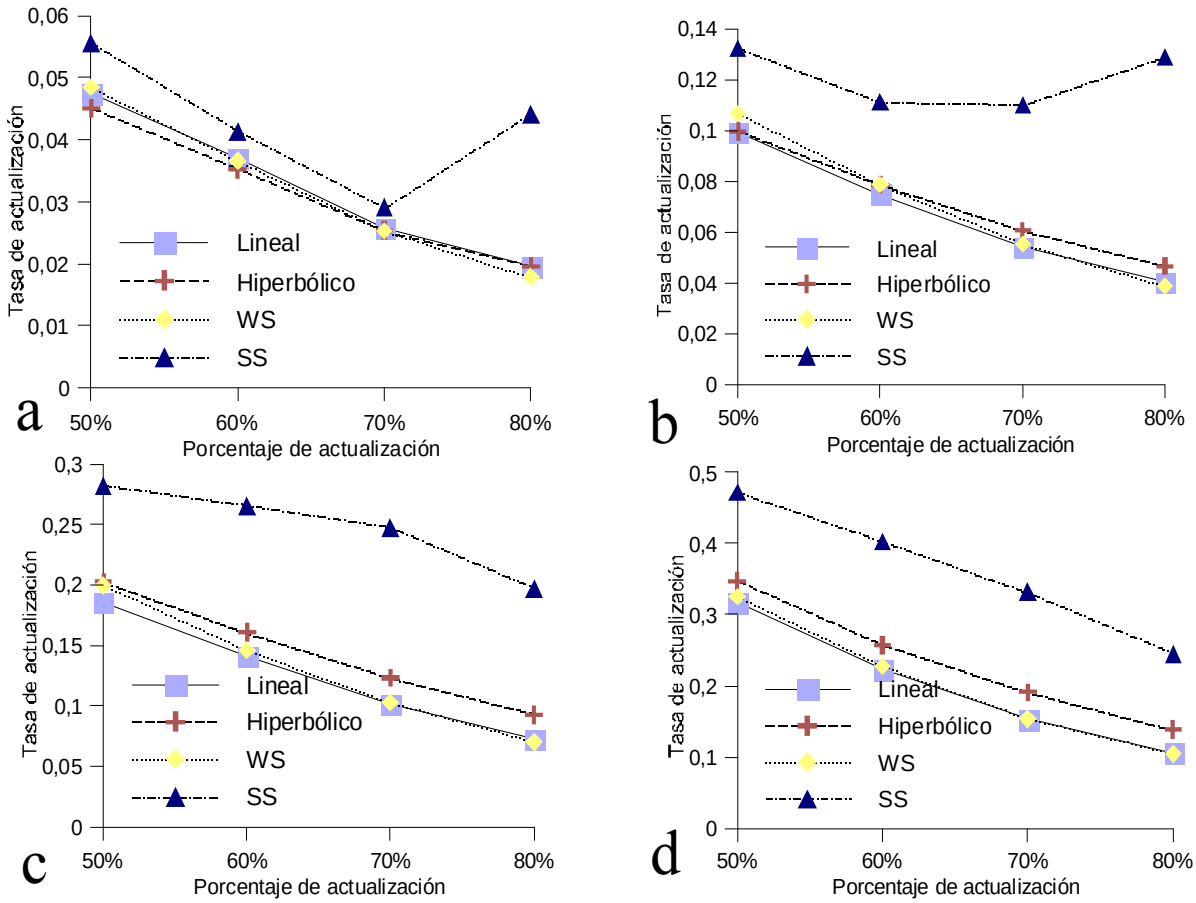


Figura 3.18: Tasa de actualización en función del porcentaje para: tasa de llamadas 0,074 (a), tasa de llamadas 0,0905 (b), tasa de llamadas 0,107 (c) y tasa de llamadas 0,1235 (d). Enlaces limitados a 50 Mbit/s.

En vista de todos los resultados anteriores, parece obvio que los algoritmos más interesantes son el **Lineal**, **Hiperbólico** y ***WS***, por lo que nos centraremos en dichos algoritmos.

3.4.3.2 Estrategias Optimista, Pesimista y Multicamino

En el estudio realizado hasta este momento se ha supuesto que, tanto para los cálculos de la función de coste, como para la búsqueda del camino de coste mínimo, el ancho de banda permanece constante hasta que llega el siguiente mensaje de actualización. Es decir, se considera que el ancho de banda de un enlace es igual al último mensaje de actualización recibido. A continuación, estudiaremos el comportamiento de

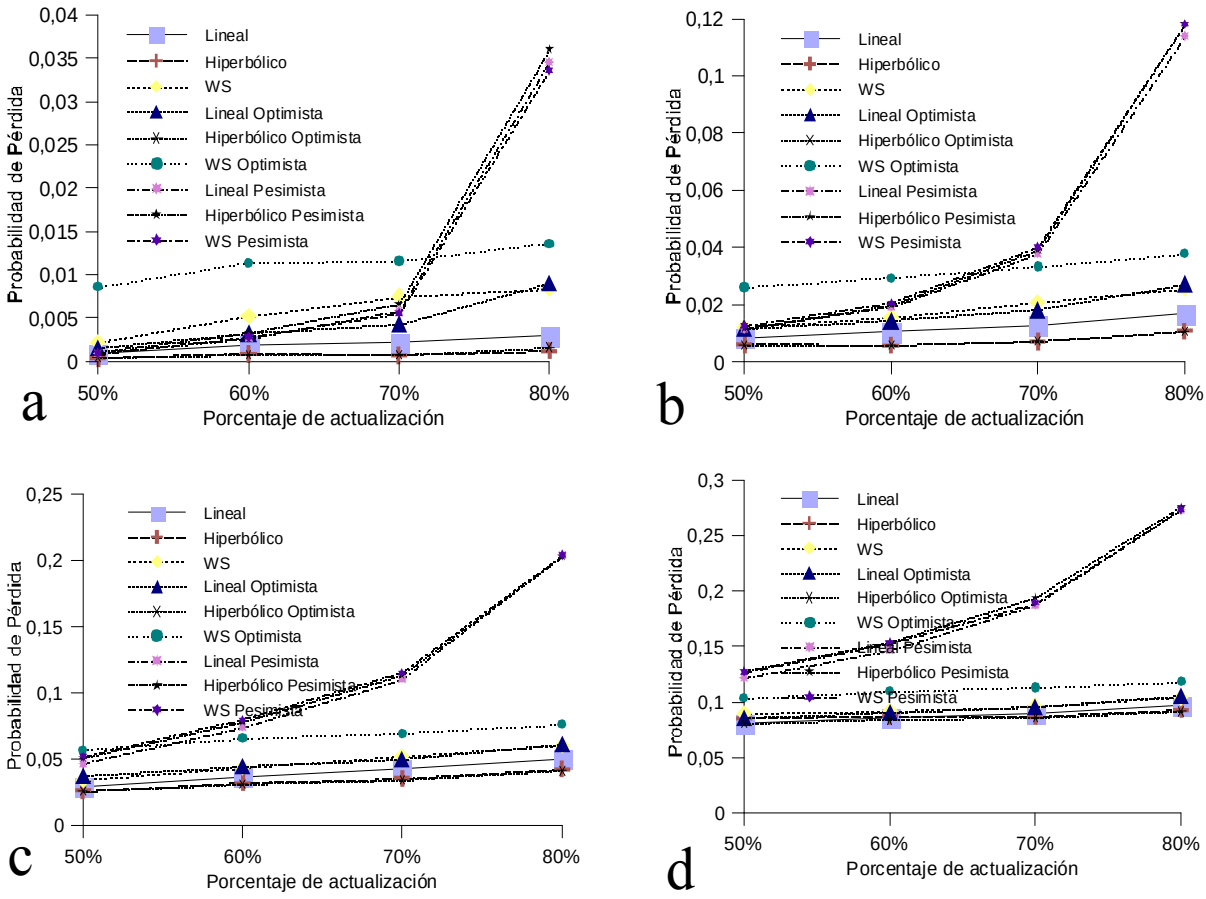


Figura 3.19: Comparación de la probabilidad de pérdida de los casos optimista y pesimista en función del porcentaje para: 0,074 llamadas/s (a), 0,0905 llamadas/s (b), 0,107 llamadas/s (c) y 0,1235 (d) llamadas/s. Enlaces limitados a 50 Mbit/s.

las diversas alternativas que se comentaron en el apartado 3.3.3.

Inicialmente estudiamos el comportamiento de los casos optimista y pesimista frente al caso medio. Las pruebas demuestran que la efectividad del uso de los distintos criterios depende tanto de la tasa de tráfico como del umbral del actualización, presentando comportamientos muy dispares, tal como se ilustra en la figura 3.19, (donde se dibuja el comportamiento en función del umbral para distintas tasas) y en las figuras 3.20 y 3.21, en las que se puede apreciar el comportamiento de los distintos criterios en función de la tasa con un umbral de actualización del 80% ($Th = 0,8$).

De estas figuras parecen extraerse una serie de conclusiones:

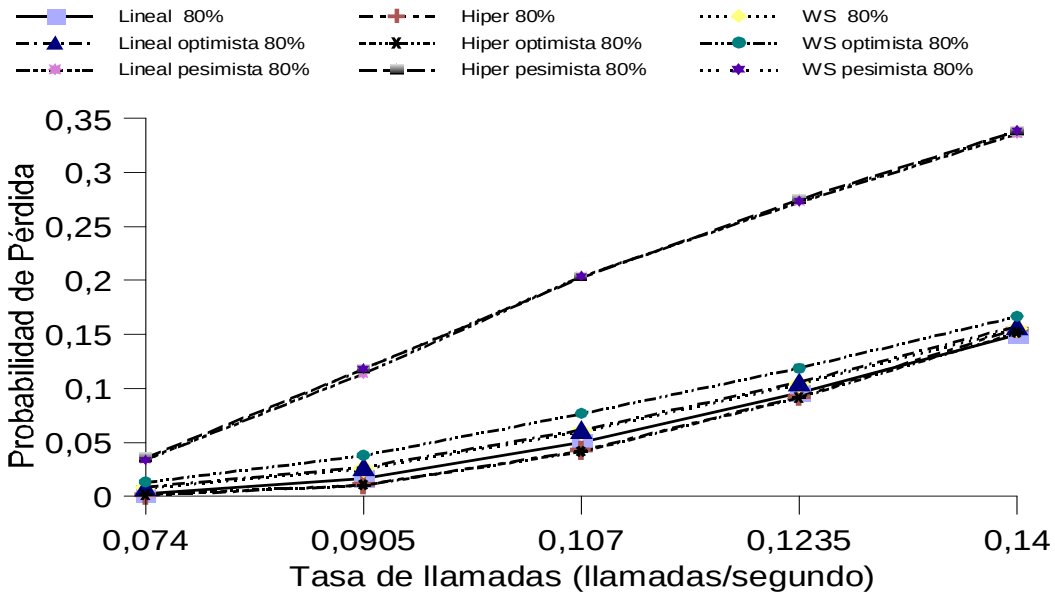


Figura 3.20: Probabilidad de pérdida de ancho de banda de las estrategias optimista, pesimista, y medio para las funciones de coste **Lineal**, **Hiperbólico** y ***WS***. Enlaces limitados a 50 Mbit/s.

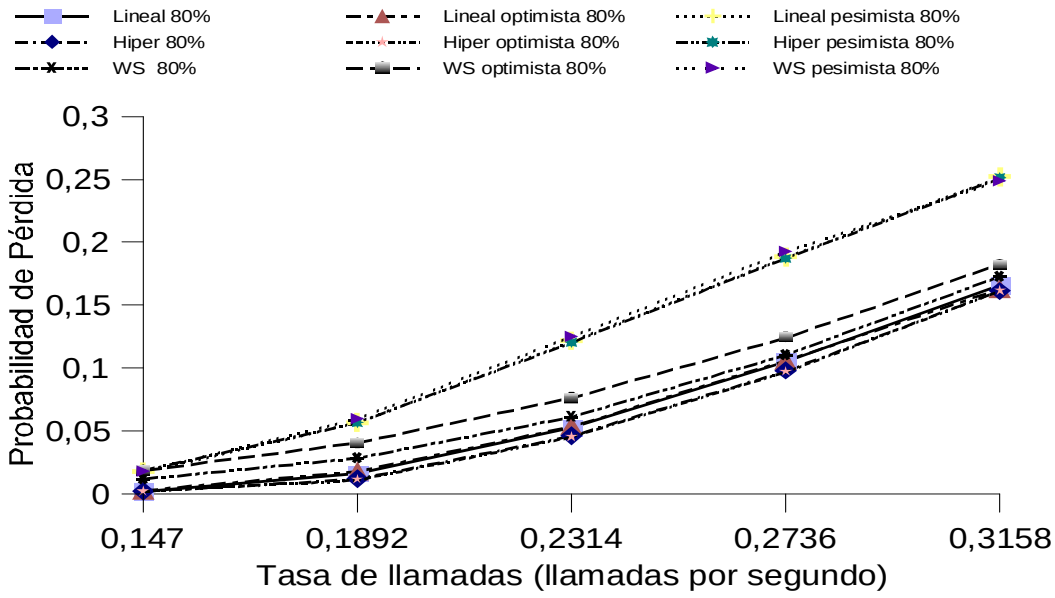


Figura 3.21: Probabilidad de pérdida de ancho de banda de las estrategias optimista, pesimista, y medio para las funciones de coste **Lineal**, **Hiperbólico** y ***WS***. Red MCI real.

- El criterio pesimista funciona mal con intervalos de actualización elevados. Esto se debe a que este criterio presupone que el ancho de banda disponible en los enlaces es $BW_{ij} = BW_{ij}^{new} * (1 - Th)$, por lo que al emplear elevados valores de Th este criterio contempla como no válidos la mayor parte de los enlaces y no es capaz de encontrar rutas viables. Este efecto se agrava al aumentar la tasa de tráfico, ya que se incrementa la probabilidad de que se cumpla $BW_{ij} < b_k$, con lo que la llamada es rechazada. Por contra, para tasas pequeñas de tráfico y umbrales de actualización pequeños, este criterio resulta interesante, pues evita que muchas de las conexiones enviadas por caminos asumidos como válidos sean rechazadas durante la etapa de establecimiento, por falta de recursos reales. Por tanto, sólo debe ser empleado para tasas de ocupación bajas con umbrales de actualización pequeños.

- El criterio optimista carece de utilidad salvo para tasas elevadas. Este comportamiento, complementario con el anterior, se explica por que a tasas elevadas los caminos se encuentran colapsados con una alta probabilidad, por lo que ya en la fase de búsqueda de camino muchas llamadas son rechazadas sin ni siquiera intentar establecer el camino. Con el criterio optimista se supone que los enlaces disponen de mayor ancho de banda del que realmente poseen, pero al menos se obtienen caminos posibles en la fase de búsqueda pudiéndose pasar a la fase de establecimiento. Aunque, posteriormente, la llamada puede ser rechazada en esta fase, el intento podría aprovechar la posibilidad que se haya liberado suficiente ancho de banda para establecer el camino. Para tasas bajas, el criterio optimista presenta un mal comportamiento ya que intenta caminos en los que no existe ancho de banda suficiente pero que, al ser más cortos, se asumen como válidos.

- El caso medio presenta el comportamiento más uniforme, tanto para tasas de tráfico, como para diversos porcentajes de actualización, por lo que salvo en casos muy específicos, donde se encuentre perfectamente definido y acotado alguno de estos parámetros, será la elección lógica. Por otro lado, el algoritmo de coste más interesante resulta ser el **Hiperbólico** desde el punto de vista de la probabilidad de bloqueo, y el **Lineal** en cuanto a la tasa de actualizaciones.

Una vez completado el estudio del comportamiento de la red bajo los criterios básicos, pasamos a repetir el experimento con los criterios de búsqueda multicamino, tanto en la modalidad secuencial como en paralelo. Debido a la variedad de comparaciones posibles, se analizan los resultados de cada uno de los algoritmos por separado y finalmente se compararan aquellos casos que mejor resultado han presentado para los

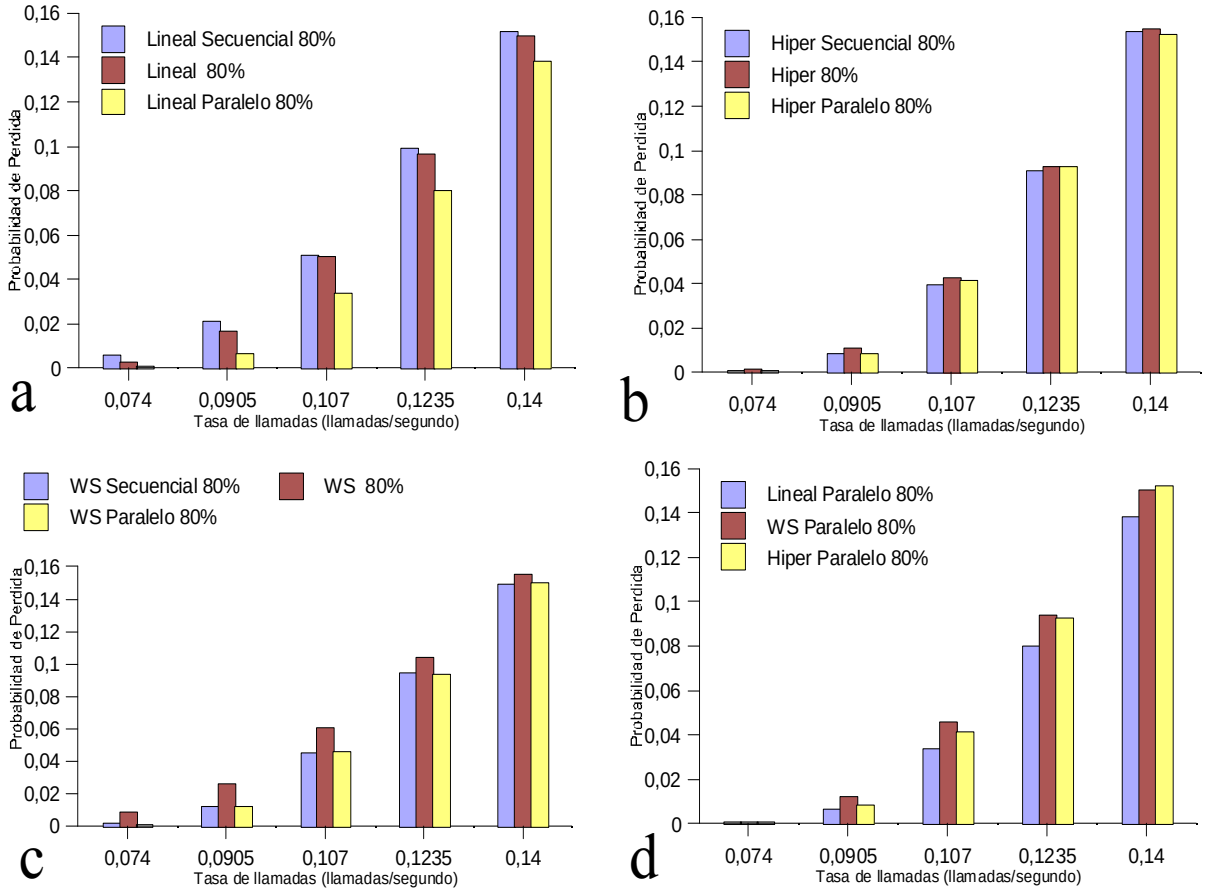


Figura 3.22: Probabilidad de pérdida de ancho de banda de las estrategias multicamino para la función de coste **Lineal** (a), **Hiperbólico** (b) y ***WS*** (c); En d se comparan los mejores resultados obtenidos para **Lineal**, **Hiperbólico** y ***WS***. Enlaces limitados a 50 Mbit/s.

algoritmos **Lineal**, **Hiperbólico** y ***WS*** (figuras 3.22 y 3.23).

Una característica interesante de las técnicas de búsqueda en paralelo es su menor sensibilidad frente a cambios en el umbral de actualización en comparación con el resto de técnicas. De hecho, las técnicas de búsqueda en paralelo pueden llegar a presentar un mejor comportamiento conforme aumenta el umbral (figuras 3.25 y 3.26). Esto puede justificarse por el hecho de que conforme aumenta el umbral, divergen en mayor medida los caminos obtenidos por cada uno de los criterios optimista, medio y pesimista, consiguiéndose una mejor distribución del tráfico entre los distintos caminos y mejorando el rendimiento final del sistema.

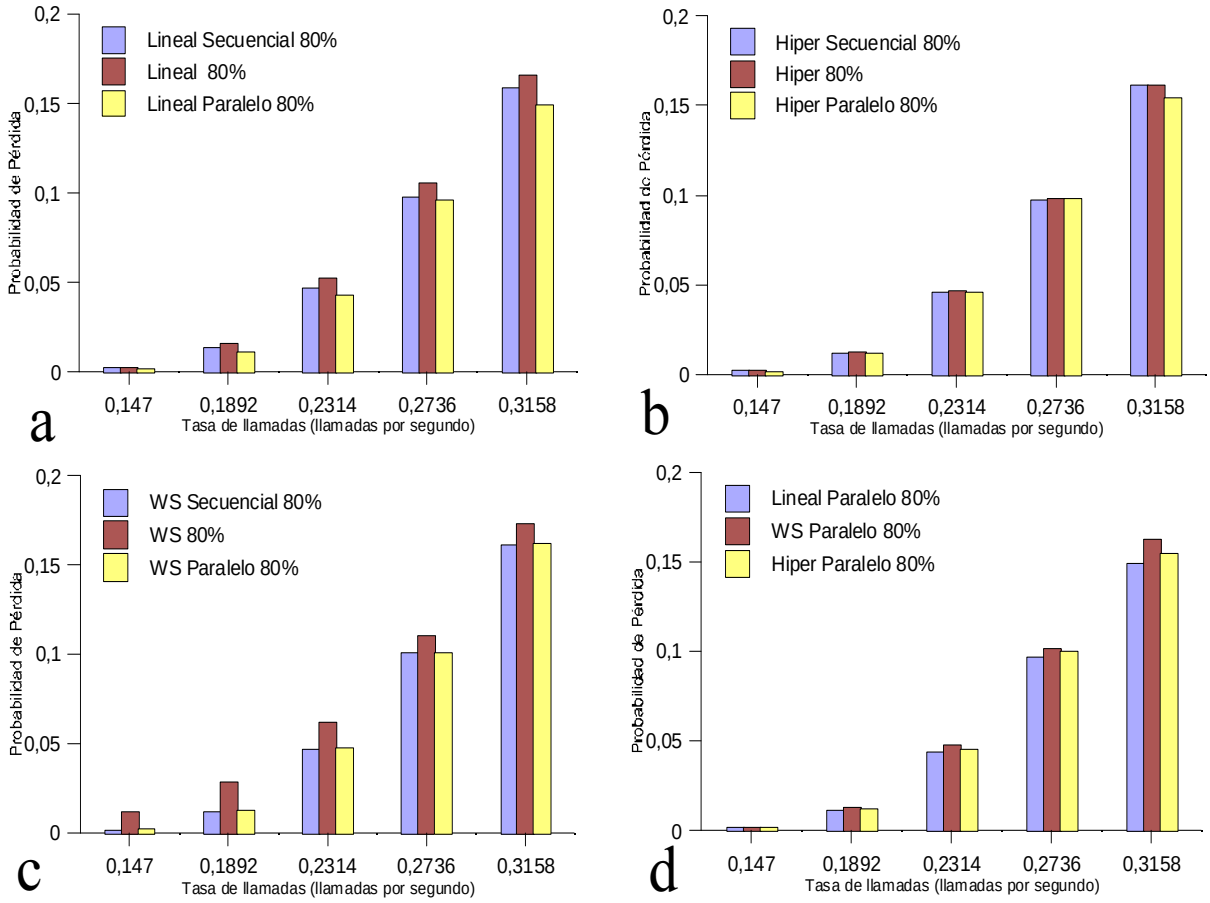


Figura 3.23: Probabilidad de pérdida de ancho de banda de las estrategias multicamino para la función de coste **Lineal** (a), **Hiperbólico** (b) y ***WS*** (c); en d se comparan los mejores resultados obtenidos para **Lineal**, **Hiperbólico** y ***WS***. Red MCI real

Es evidente que el empleo de técnicas multicamino mejora el rendimiento de los diferentes algoritmos, aunque de forma distinta para cada uno de ellos. Por ejemplo, para el algoritmo **Hiperbólico** la mejora frente al caso medio es demasiado pequeña para justificar su empleo, mientras que para los algoritmos **Lineal** y ***WS***, el empleo de este tipo de técnicas multicamino mejora el comportamiento en todos los rangos, especialmente para bajas tasas de tráfico donde estos algoritmos presentaban un peor comportamiento frente al **Hiperbólico**. La escasa diferencia observada en el caso del algoritmo **Hiperbólico** se debe a la práctica coincidencia de los caminos encontrados por el criterio optimista y el caso medio, situación que provoca una escasa ganancia.

Debido al mal comportamiento del criterio pesimista cabría plantearse si realmente debería considerarse a la hora de buscar los caminos, ya que eliminándolo se reduce la

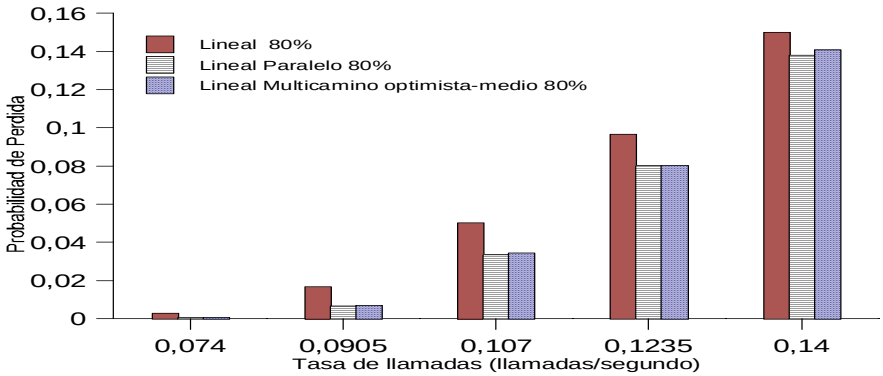


Figura 3.24: Comparación de la probabilidad de pérdida de los casos: medio, multicamino paralelo y multicamino limitado a los criterios optimista y medio para el algoritmo de asignación de costes Lineal. Enlaces limitado a 50 Mbits/s.

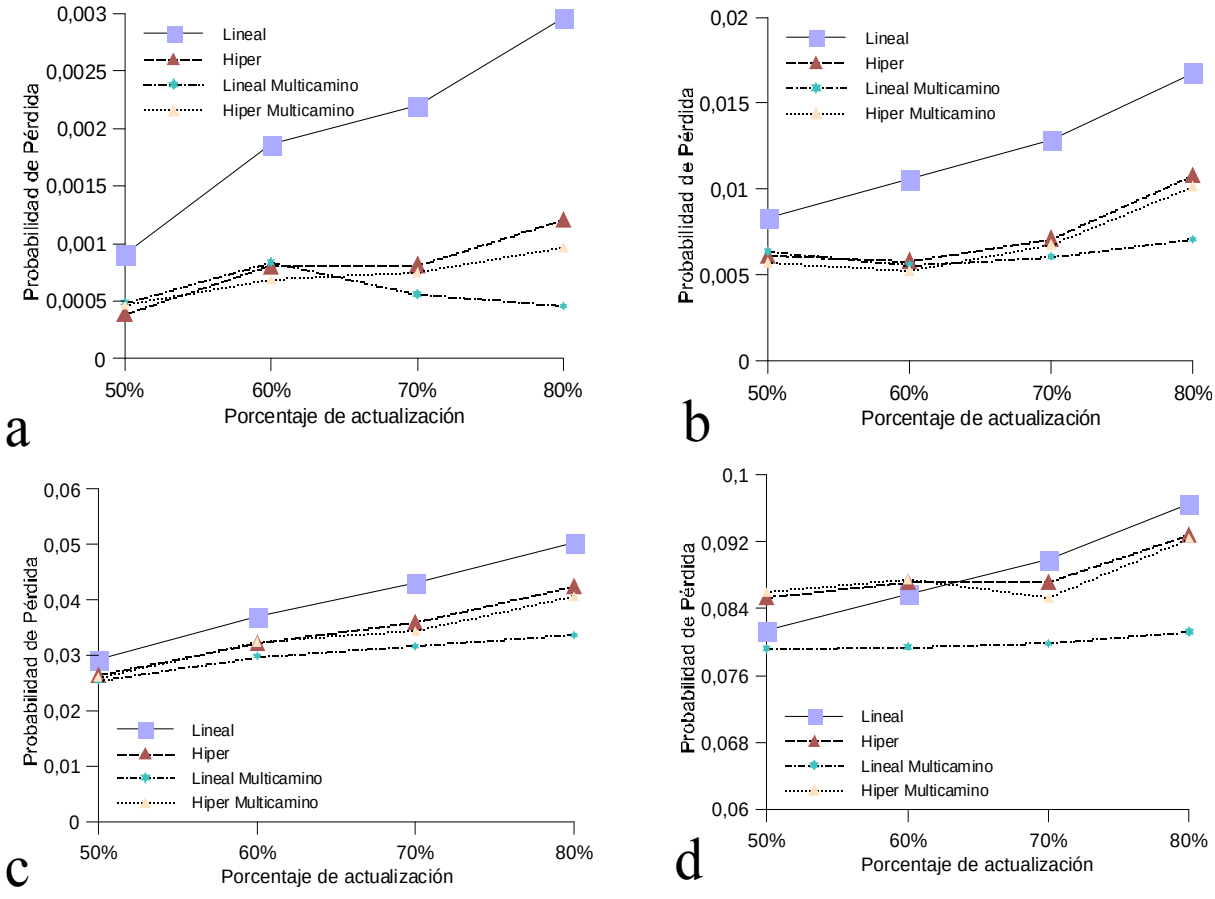


Figura 3.25: Comparación de la probabilidad de pérdida de los casos multicamino y medio en función del porcentaje para: 0,074 llamadas/s (a), 0,0905 llamadas/s (b), 0,107 llamadas/s (c) y 0,1235 llamadas/s (d). Enlaces limitados a 50 Mbits/s.

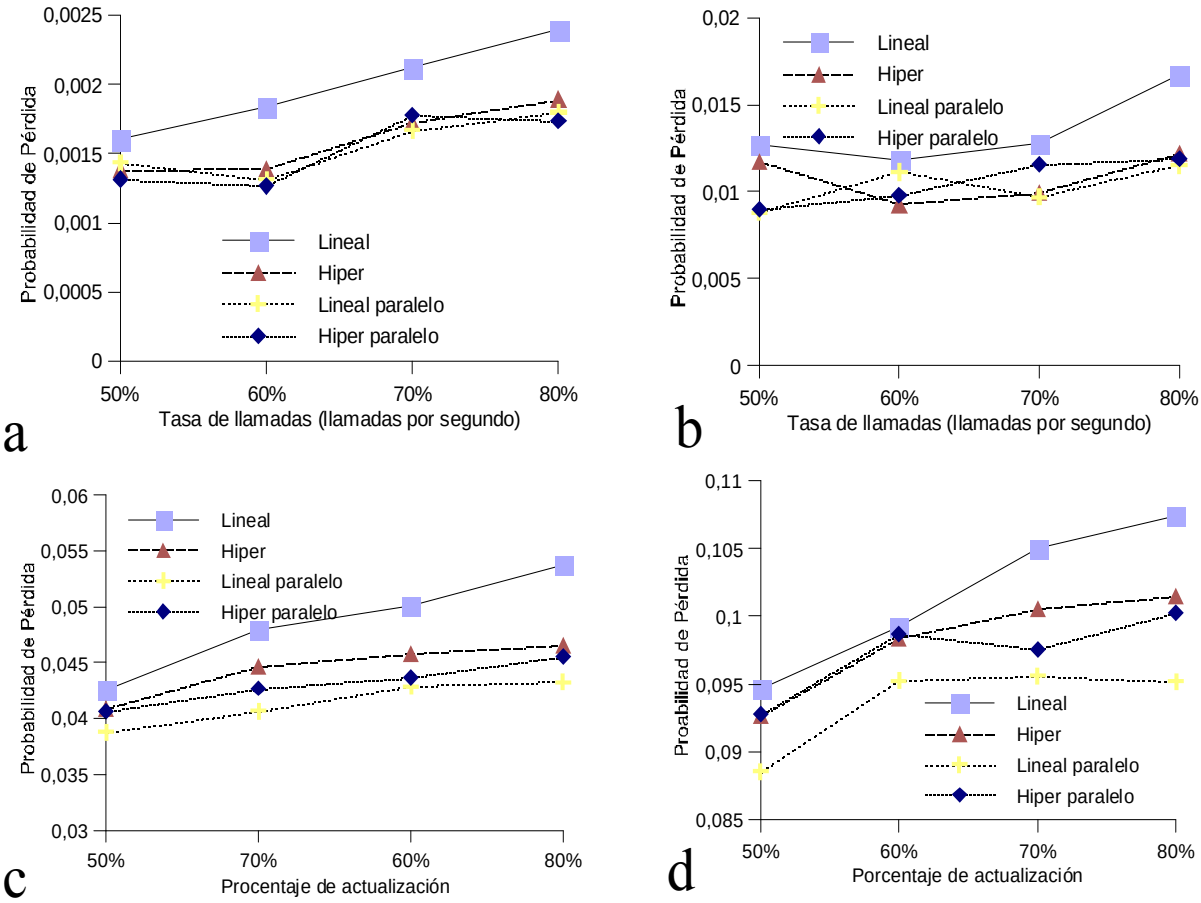


Figura 3.26: Comparación de la probabilidad de pérdida de los casos multicamino y medio en función del porcentaje para: tasa de llamadas 0,147 (a), tasa de llamadas 0,1892 (b), tasa de llamadas 0,2314 (c) y tasa de llamadas 0,3158 (d). Red MCI real.

complejidad final del sistema. Para contestar esta pregunta es necesario comparar los resultados obtenidos en el caso de probar los tres caminos con búsqueda en paralelo frente al caso en que sólo se prueban en paralelo los caminos obtenidos con los criterios optimista y medio. En la figura 3.24 podemos ver el resultado de esta comparación. Como se puede apreciar, el método de las tres opciones en paralelo es sólo ligeramente superior, lo que cuestiona su utilidad frente a la búsqueda reducida.

3.4.3.3 Efecto del reintento de llamadas perdidas

Uno de los fenómenos que con mayor fuerza degrada el funcionamiento de cualquier algoritmo de encaminamiento lo constituye la existencia de reintentos de llamadas que

han sido previamente rechazadas. Este efecto es particularmente pernicioso pues provoca una correlación en la elección de los destinos, con lo que las llamadas tienden a emplear enlaces que ya están bloqueados, incrementando la probabilidad de que la llamada sea rechazada de nuevo. Así pues, es necesario analizar cómo afecta a los esquemas de encaminamiento el hecho de que exista cierta posibilidad de reintento de llamada.

A fin de forzar la correlación en el destino, cuando una llamada es rechazada fijamos una probabilidad de reintento dentro de un determinado intervalo de tiempo. Esta probabilidad es decreciente con el número de reintentos mientras que el tiempo de espera se modela mediante una distribución exponencial de media 20 s. Para nuestros experimentos la probabilidad utilizada de reintento de llamadas se puede ver en la tabla 3.3.

Se han comparado los resultados de emplear tráfico con rellamadas con el caso anterior de tráfico sin rellamadas y los resultados de estas pruebas se presentan en las figuras 3.27 y 3.28 con un umbral de actualización del 80% para los dos escenarios empleados. Se puede contemplar que el comportamiento se degrada más cuanto mayor es el tráfico ofrecido a la red. Este fenómeno es fácil de explicar, ya que al aumentar la carga crece de una manera no lineal la probabilidad de que una de estas nuevas llamadas entrantes sea rechazada y, por lo tanto, la tasa de llamadas que reintentan establecer una conexión.

En la figura 3.29 podemos observar la probabilidad de pérdida de los algoritmos **Lineal** e **Hiperbólico**, tanto para el caso medio como para el caso multicamino. Se han escogido estos dos criterios por ser los que mejor comportamiento han presentado hasta el momento. Por último, en la figura 3.30 podemos ver la tasa de actualización. Como se deduce de las figuras, el comportamiento de los distintos algoritmos y criterios es similar al caso en que no hay reintentos de llamada, excepto en que la probabilidad de pérdida de llamada y la tasa de actualización son mayores y aumentan con la tasa de tráfico de una manera mucho más brusca.

Número de reintentos	probabilidad
1	0,8
2	0,6
3	0,3
4	0,1
5	0

Tabla 3.3: Probabilidades de rellamada

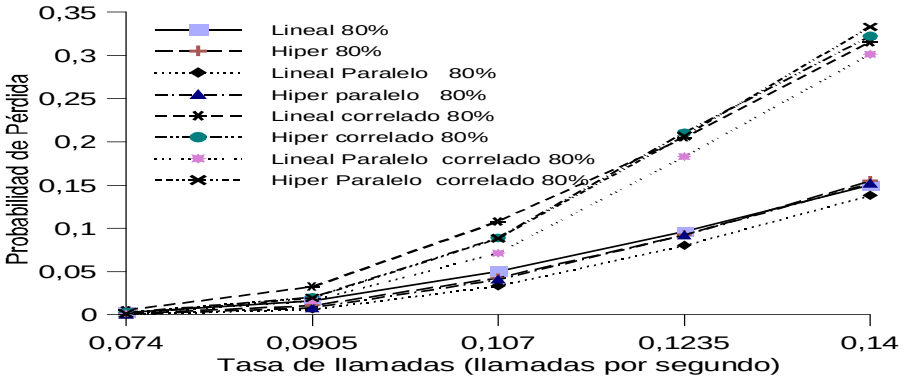


Figura 3.27: Comparación de la probabilidad de pérdida de ancho de banda para un umbral de actualización del 80% de los algoritmos **Hiperbólico** y **Lineal** con y sin reintentos. Enlaces limitados 50 Mbits/s.

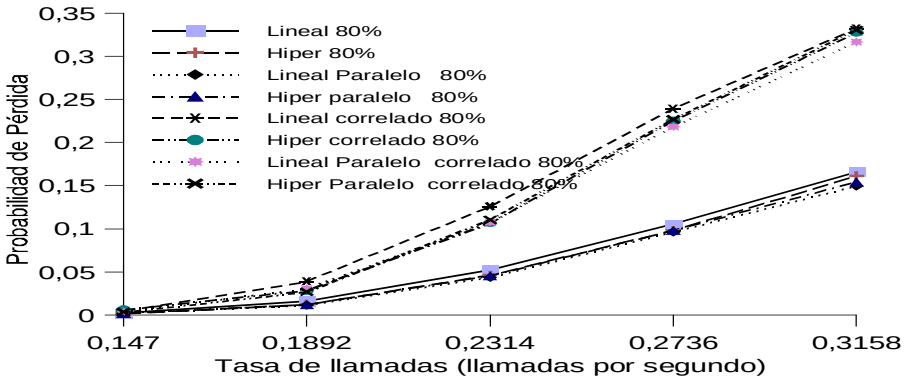


Figura 3.28: Comparación de la probabilidad de pérdida de ancho de banda para un umbral de actualización del 80% de los algoritmos **Hiperbólico** y **Lineal** con y sin reintentos. Red MCI real.

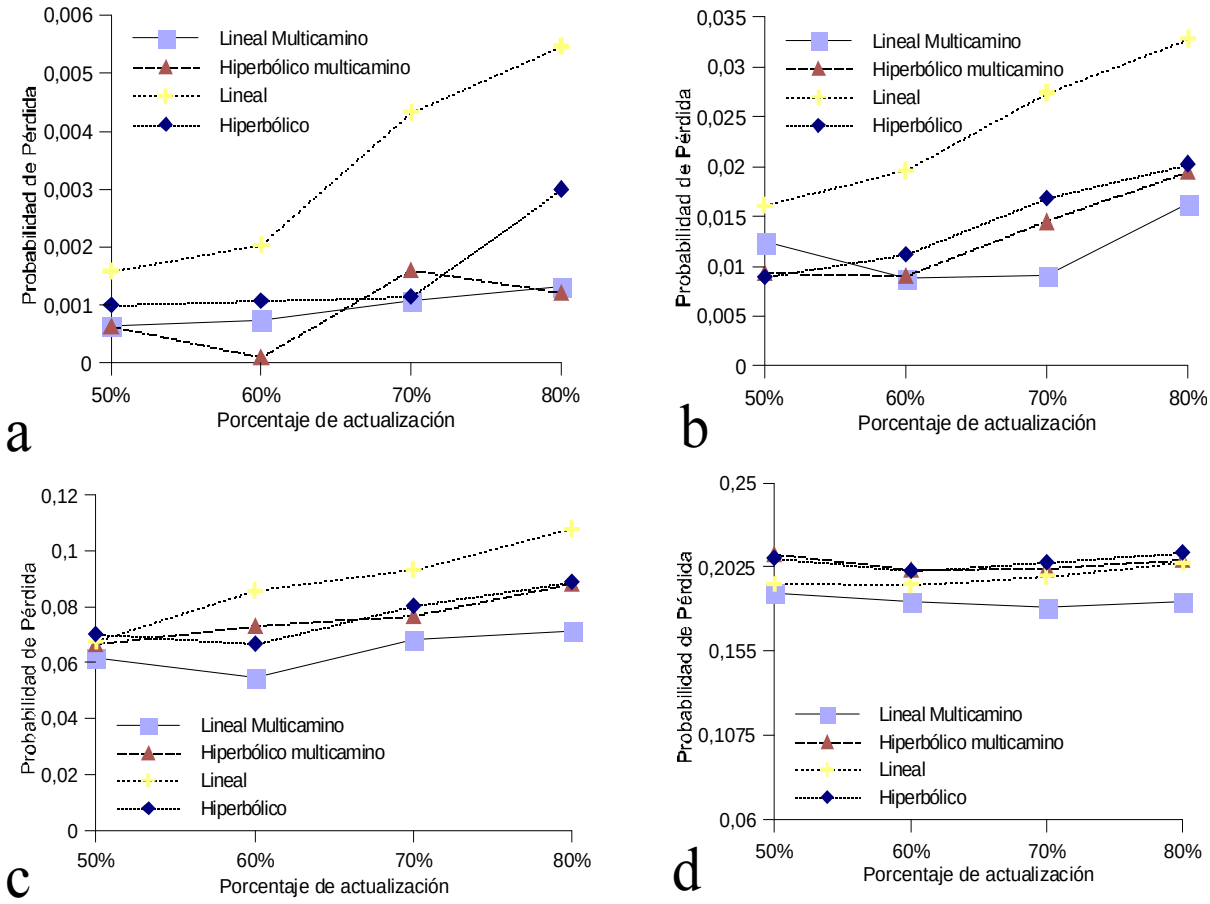


Figura 3.29: Comparación de la probabilidad de pérdida con reintentos de llamada de los casos multicamino y medio en función del porcentaje para: tasa de llamadas 0,074 (a), tasa de llamadas 0,0905 (b), tasa de llamadas 0,107 (c) y tasa de llamadas 0,1235 (d).Enlaces limitados 50 Mbits/s.

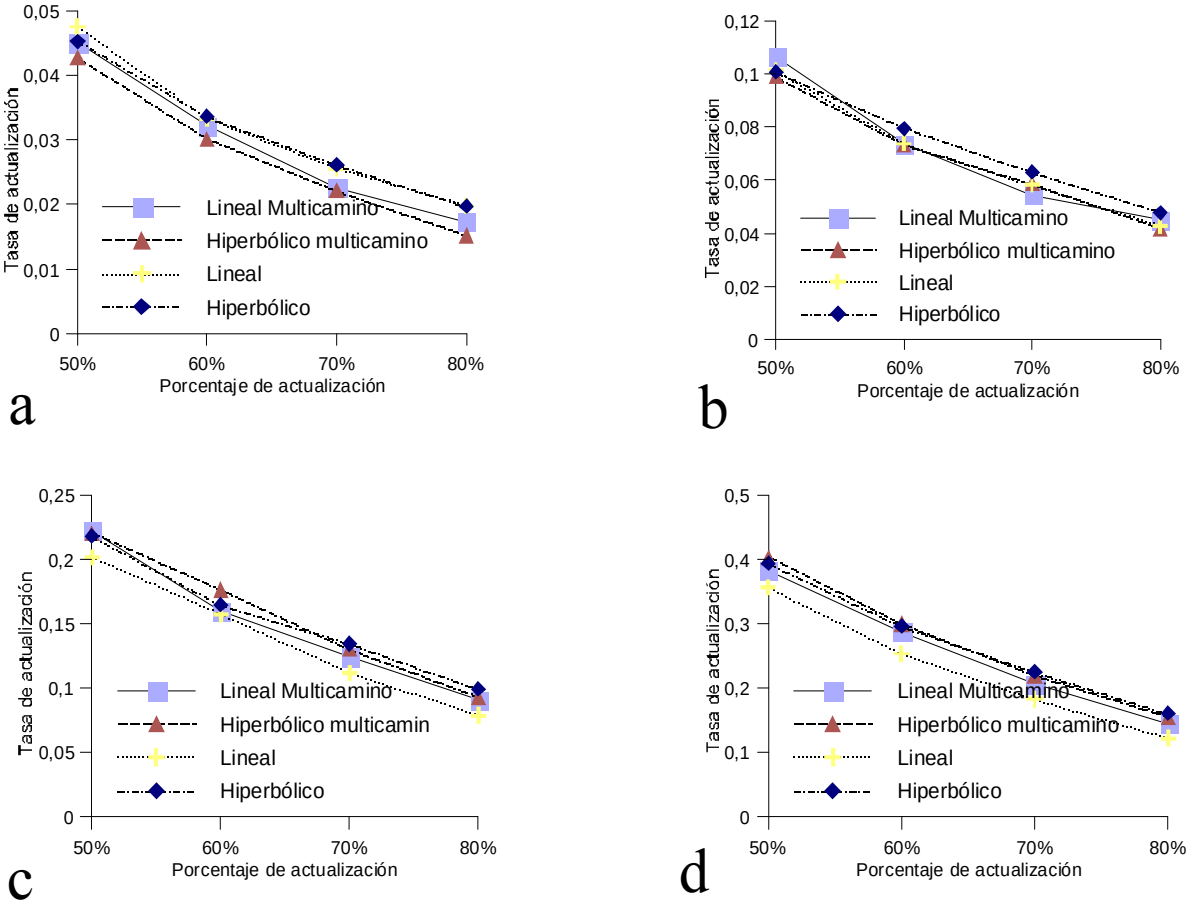


Figura 3.30: Comparación de la tasa de actualización con reintentos de llamada de los casos multicamino y medio en función del porcentaje para: tasa de llamadas 0,074 (a), tasa de llamadas 0,0905 (b), tasa de llamadas 0,107 (c) y tasa de llamadas 0,1235 (d).Enlaces limitados a 50 Mbits/s.

3.5 Efecto del grado de agregación de llamadas sobre la elección del umbral

En esta sección estudiaremos la influencia de la agregación de llamadas en los algoritmos de encaminamiento con imprecisión en los datos. En las pruebas que se realizan a continuación se tendrán en cuenta dos posibles escenarios. En el primer escenario, de alta capacidad de agregación, el ancho de banda de cada llamada es pequeño frente a la capacidad de los enlaces y, por lo tanto, se permite un gran número de llamadas simultáneas sobre cada enlace. En el segundo escenario, de baja agregación, los elevados recursos que exigen las llamadas sólo posibilitan un reducido número de llamadas simultáneas sobre cada enlace. En concreto, para las pruebas se ha utilizado la red MCI con enlaces limitados a 50 Mbits/s y, en cuanto a las características del tráfico simulado, estas son las mismas que en las pruebas anteriores con la salvedad del ancho de banda, que para las pruebas de alta agregación viene dado mediante una distribución uniforme que toma valores entre 0,25 y 0,75 Mbits/s, mientras que en las de baja agregación la distribución del ancho de banda requerido toma valores entre 16 y 20 Mbits/s. En cuanto a los umbrales de actualización escogidos para el estudio, han sido 40%, 60% y 80% ($Th = 0,4$, $0,6$ y $0,8$, respectivamente), aparte del caso de conocimiento preciso del estado de los enlaces. Por otro lado, la función de coste utilizada para estas pruebas ha sido la **Lineal**. En la figura 3.31 se ilustran los resultados con una alta agregación de tráfico, mientras que los resultados con baja agregación se muestran en la figura 3.32.

De estas figuras se constata una serie de tendencias, algunas de éstas tan evidentes como que la importancia de la elección del umbral aumenta al bajar la agregación. Este

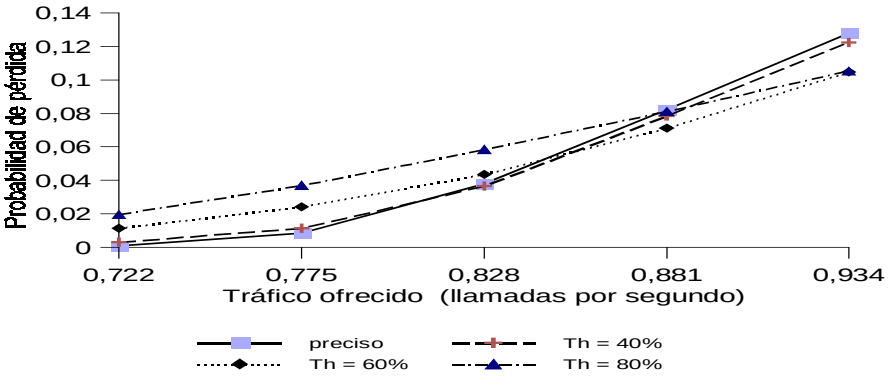


Figura 3.31: Evolución de la probabilidad de pérdida de ancho de banda para varios porcentajes de disparo de actualización cuando existe una alta agregación de llamadas

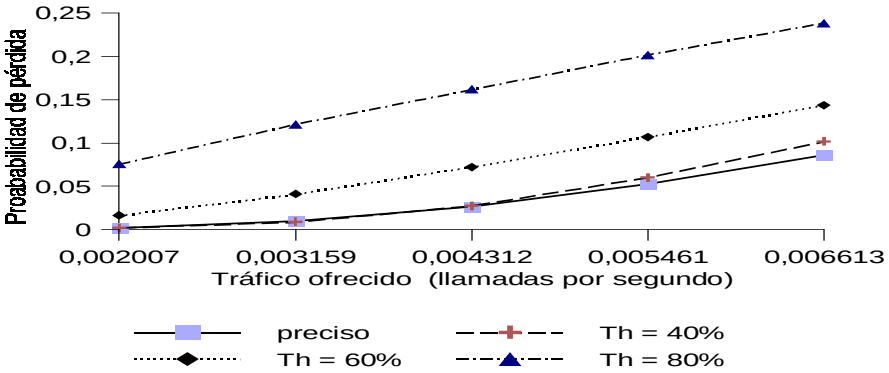


Figura 3.32: Evolución de la probabilidad de pérdida de ancho de banda para varios porcentajes de disparo de actualización cuando existe una baja agregación de llamadas

fenómeno es fácilmente explicable: con baja agregación cada llamada consume una gran cantidad de los recursos de los enlaces. Intentar en esas condiciones enviar una llamada sobre un enlace sobre el que se acaba de establecer otra llamada (de la que no se tiene noticia por no haber generado mensaje de actualización) implica una alta probabilidad de rechazo por falta de recursos. También en condiciones de baja agregación se aprecia que el umbral del 40% se encuentra cercano al comportamiento con conocimiento preciso. Esto se justifica por que la conexión o desconexión de una única llamada ya produce variaciones de ancho de banda cercanas o superiores al 40%, con lo que se está muy cerca del caso preciso en donde cualquier cambio de estado en la red dispara el proceso de actualización.

Sin embargo, los resultados más interesantes se encuentran al estudiar el comportamiento de la red en condiciones de alta agregación de llamadas, sobre todo si tenemos en cuenta que este es el entorno actual de funcionamiento, ya que, en este momento, a nivel comercial ya se están implantando redes con capacidades cercanas al terabits/s por fibra óptica, como es el caso de la red *EuroRings* [KPN01] de KPN. En estas condiciones el comportamiento es bien distinto. Como se puede apreciar en la figura 3.31, cuando la carga ofrecida es pequeña y menor se diseña el umbral de actualización (y por lo tanto, mayor resulta la tasa de actualización), mejor será el rendimiento. En este aspecto se comporta de forma similar al caso de baja agregación. Las diferencias aparecen al aumentar la carga de tráfico ofrecido. En este caso, como queda ilustrado en la figura 3.31, para altas cargas de tráfico ofrecido el comportamiento mejora cuanto mayor es el umbral. Esto se justifica gracias a la forma en que se distribuye el tráfico entre los distintos caminos. En el caso de utilizar un umbral demasiado pequeño se provocan grandes

oscilaciones en el tráfico ofrecido entre los distintos caminos posibles. Si bien esta estrategia es interesante cuando se trabaja con cargas bajas y hay disponibles caminos con abundantes recursos, no es adecuada para cargas altas. En este caso es preferible utilizar caminos más cortos aunque más cargados, lo que redunda en que se consume un menor número de recursos de red que podrán ser utilizados para conectar futuras llamadas. Sólo se producirá una elección de otro camino cuando este se encuentre sobrecargado. Este tipo de condición se verifica cuando se utiliza un umbral de actualización elevado. Así pues, cuando existe un elevado índice de agregación de llamadas interesa umbrales pequeños para cargas bajas y umbrales elevados para cargas altas.

Basándonos en este comportamiento vamos a presentar a continuación un nuevo mecanismo de actualización que muestra un elevado índice de actualización para cargas pequeñas, mientras que aumentará de forma dinámica su umbral de actualización para adaptarse a cargas elevadas de tráfico.

Otra posible solución, no implementada en esta tesis, pasaría por definir una función de costes que primara el balanceo o la economía de recursos en función de la carga de la red.

3.6 Actualización mediante umbral adaptativo

3.6.1 Descripción del algoritmo

En este apartado presentamos un nuevo mecanismo de actualización que intenta adaptar el umbral de disparo al tráfico ofrecido al enlace [ACS01c]. Dicho mecanismo es fácil de implementar en las redes actuales y presenta, con respecto al mecanismo de umbral proporcional usado hasta el momento, un mejor aprovechamiento del ancho de banda reservado para la señalización, así como un mejora en el rendimiento en términos de probabilidad de pérdida de ancho de banda.

La idea básica de este método de actualización se fundamenta en modificar el umbral de actualización de forma dinámica, con el objetivo de que la tasa de actualización permanezca constante con independencia del tráfico ofrecido a la red.

El algoritmo básico de modificación del umbral se muestra en la siguiente expresión:

$$Th = \begin{cases} Th + \Delta Th & si \quad \frac{Actualizaciones}{T_{intervalo}} > \phi \\ Th - \Delta Th & si \quad \frac{Actualizaciones}{T_{intervalo}} < \phi \end{cases} \tag{3.19}$$

donde $T_{intervalo}$ es un intervalo de observación en el cual se mide el número de mensajes de actualización que llegan al nodo, ϕ es la tasa de actualización objetivo, *Actualizaciones* es el número de mensajes de actualización que han llegado al nodo en el intervalo $T_{intervalo}$, Th es el umbral de actualización empleado para determinar cuándo se debe de enviar un nuevo mensaje de actualización de estado de un enlace y ΔTh es el incremento (o decremento) del umbral que se realiza en cada intervalo de estimación. El mensaje de actualización del estado del enlace se envía, al igual que en el caso de umbral proporcional, cuando se verifica la condición $\frac{|BW_{old} - BW_{new}|}{BW_{old}} > Th$, donde BW_{old} es el ancho de banda disponible en el enlace en la última actualización, BW_{new} es el ancho de banda actualmente disponible y Th el último valor modificado del umbral.

Como se puede observar, el algoritmo propuesto es realmente simple y consigue el efecto que en el apartado anterior se determinó como deseable, es decir, un umbral de actualización pequeño para tasas bajas y umbral elevado para tasa altas. Así pues, gracias a este sencillo algoritmo, se consigue que al bajar el tráfico ofrecido y disminuir el número de mensajes de actualización se reduzca el tamaño del umbral de actualización a fin de aumentar la tasa. Por el contrario, cuando sube la tasa de tráfico, crece el número de actualizaciones y, por lo tanto, el algoritmo aumenta el umbral a fin de intentar que el número de actualizaciones permanezca constante. Otra ventaja derivada del algoritmo es que el ancho de banda consumido por las actualizaciones pasa a ser constante e independiente de la tasa de tráfico, al contrario de la técnica de umbral proporcional que tiende a consumir mayor ancho de banda al aumentar dicha tasa, con lo que agrava el problema, pues deja un menor ancho de banda útil.

3.6.2 Pruebas y resultados

Una vez descrito el funcionamiento del algoritmo, pasaremos a la fase de pruebas a fin de validar su funcionamiento. En estas pruebas hemos utilizado la red MCI limitada y un tráfico con las mismas características que el usado para el caso de alta agregación del apartado anterior. En cuanto a los umbrales escogidos para este estudio han sido 40%, 60% y 80% para el caso de umbral proporcional. Las tasas objetivo para el umbral adaptativo fueron $0,1$, $0,5$ y 1 actualizaciones/s. El intervalo de tiempo para estimar

la tasa de actualizaciones se fija en 300 segundos. En cuanto al tráfico ofrecido se han considerado dos posibles escenarios. El primero utiliza un tráfico fijo durante todo el intervalo de prueba (como el descrito para todas las pruebas anteriores). En el segundo se ha utilizado un tráfico que varía de forma periódica (con un periodo de 86400 segundos) siguiendo la siguiente ecuación:

$$\lambda(t) = \widetilde{\lambda}\left(1 + \cos\left(\frac{2\pi t}{86400} + \psi\right)\right) \tag{3.20}$$

donde $\widetilde{\lambda}$ es la tasa de tráfico media en llamadas por segundo, t es el tiempo en segundos desde que comenzó la simulación y ψ es un valor escogido mediante una distribución uniforme entre $[0, 2\pi]$ con el objetivo de conseguir que la variación de la tasa en los distintos nodos esté desfasada.

En la figura 3.33 se muestran los resultados cuando el tráfico ofrecido es constante. Se observa cómo el umbral proporcional aumenta la tasa de actualizaciones conforme sube la tasa de tráfico ofrecido, sin mejorar por ello la probabilidad de pérdida de llamada. En cambio, el algoritmo propuesto limita el número de actualizaciones impidiendo que aumenten al crecer el tráfico, pero al mismo tiempo, manteniendo una buena probabilidad de pérdida. Es cierto que este algoritmo presenta una mayor tasa de actualización que el algoritmo de umbral proporcional para bajas tasas, pero también ofrece una mejor probabilidad de pérdida, teniendo en cuenta que a estas tasas no es tan importante una tasa de actualización relativamente alta porque hay suficientes recursos de ancho de banda y de cómputo disponibles.

El caso de tráfico variable se ilustra en las figuras 3.34 y 3.35. En la primera de estas figuras se ha repetido el experimento de tráfico fijo con la única excepción de que el tráfico ofrecido a la red varía siguiendo la ecuación (3.20). En la segunda figura se ha introducido unos límites a la variación máxima del umbral, de forma que este no puede tomar valores menores de $0,2$ ni superiores a $0,8$. Esta acotación evita que la tasa pueda hacerse demasiado pequeña, circunstancia que ocurriría si el tráfico ofrecido es muy bajo, puesto que en esta situación el sistema tendería a actualizar el estado con cada llamada. Por contra, la cota superior intenta evitar que en el caso de una alta tasa de tráfico el umbral pueda llegar a hacerse muy grande, cercano al 100%, en cuyo caso dejaría de actualizar el estado de los enlaces y el sistema dejaría de tener encaminamiento dinámico.

Lo que sí se puede observar en estas figuras es que el algoritmo presenta dificultades para estabilizar la tasa de actualización debido a las oscilaciones del tráfico ofrecido a la

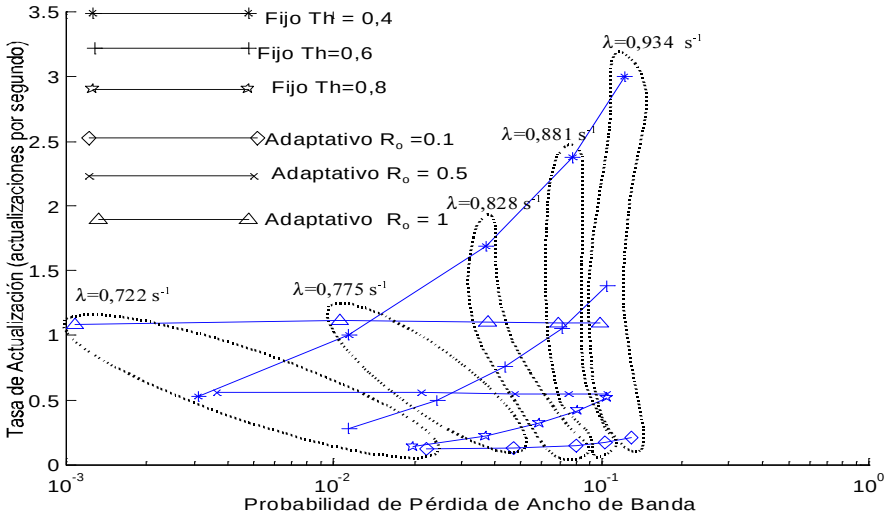


Figura 3.33: Probabilidad de pérdida frente a tasa de actualización para umbral adaptativo y umbral fijo con tasas de tráfico constantes

red, circunstancia que provoca unas fuertes variaciones en el número de actualizaciones generadas por intervalo. El problema estriba en la utilización de un ΔTh constante, ya que si este es muy bajo y se produce una fuerte oscilación, al sistema tiene dificultades para volver a la situación de equilibrio. Si además se considera que el sistema es dinámico, lo probable será que nunca la alcance. Si, por el contrario, este valor es demasiado elevado, el sistema se convierte en inestable por las fuertes oscilaciones que se producen. Una alternativa es que el valor ΔTh cambie a su vez de forma dinámica, de forma que cuanto más alejado esté de la situación de equilibrio mayor sea su valor y viceversa, cuanto más cerca este de esa situación de equilibrio menor sea el valor de ΔTh.

Como se puede constatar, el algoritmo propuesto presenta un excelente comportamiento para tasas de tráfico elevadas, manteniendo controlada la tasa de actualización para que no suba de forma súbita y presentando, simultáneamente, una buena probabilidad de pérdida. Así mismo, permite trabajar correctamente cuando el tráfico ofrecido es bajo con una buena tasa de actualización, distribuyendo el tráfico entre los distintos caminos de forma eficiente.

Resumiendo, en este apartado hemos presentado una alternativa a la actualización de umbral proporcional. Esta alternativa permite adaptar el umbral con el objetivo de lograr un mejor aprovechamiento de la red, consiguiendo umbrales de actualización pequeños cuando el tráfico ofrecido a la red es bajo (y por lo tanto, interesa una alta tasa

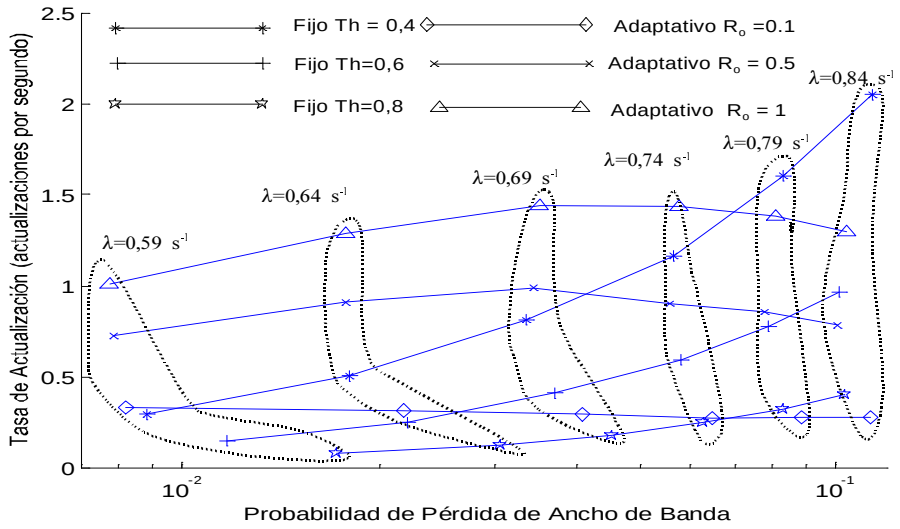


Figura 3.34: Probabilidad de pérdida frente a tasa de actualización para umbral adaptativo y umbral fijo con tasas de tráfico variables en el tiempo.

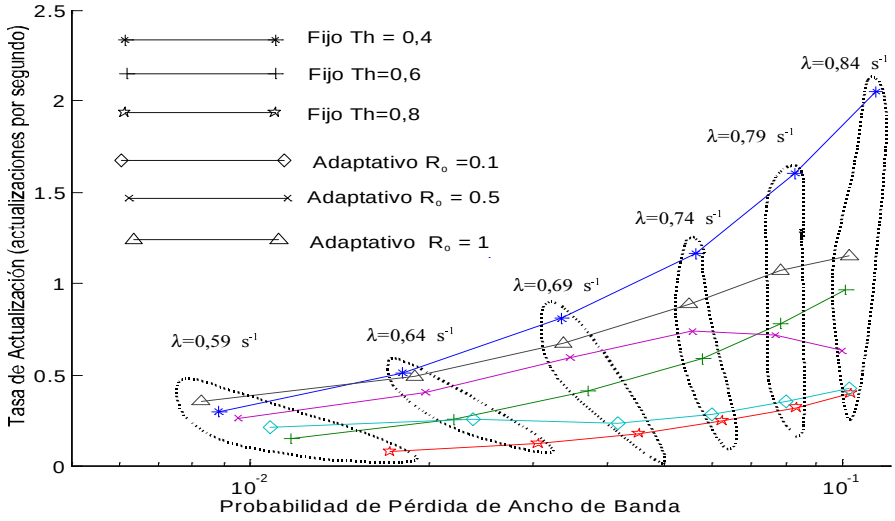


Figura 3.35: Probabilidad de pérdida frente a tasa de actualización para umbral adaptativo acotado y umbral fijo con tasas de tráfico variables en el tiempo ($Th_{max} = 0,8$, $Th_{min} = 0,2$).

de actualización para conseguir una buena distribución del tráfico), y umbrales elevados cuando la tasa de tráfico es alta (a fin de evitar que una excesiva distribución del tráfico penalice el aprovechamiento de los recursos de la red).

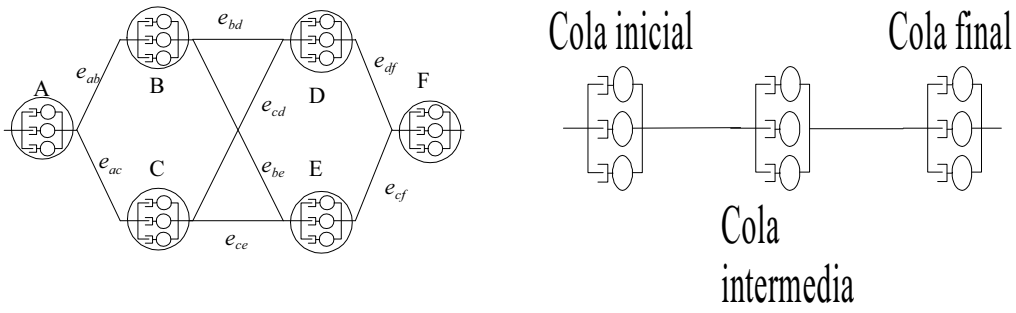
3.7 Modelos de tráfico

Tradicionalmente, el modelo de tráfico empleado para el estudio y simulación de los algoritmos y protocolos de encaminamiento establece que la función de distribución que modela las llamadas entrantes en la red sigue un modelo de *Poisson* y que la duración de las llamadas viene modelada mediante una distribución de tipo exponencial. Sin embargo, estudios de la última década [BJ99][Bol94][GGV98] rebaten éste modelo y determinan que la distribución que modela la duración de las llamadas sigue un modelo distinto de la distribución exponencial.

Cabría pensar que este hecho puede afectar en gran medida al rendimiento de las redes de comunicaciones, puesto que gran parte de los algoritmos de encaminamiento y prácticamente, todos los algoritmos de dimensionamiento empleados en la actualidad, han sido diseñados y optimizados bajo la suposición de que la duración de las llamadas sigue una distribución estrictamente exponencial, entre otras razones, por tratabilidad analítica. Dada esta circunstancia, surge la necesidad de estudiar la dependencia del rendimiento final de la red con respecto al hecho de que la duración de las conexiones no siga el modelo exponencial.

En este trabajo se ha realizado un estudio del rendimiento de una red de comunicaciones orientada a conexión modelando la duración de las llamadas con distintas funciones de distribución de probabilidad. Este estudio se ha llevado a cabo mediante simulación, empleando como banco de pruebas la misma red de area extensa (MCI), de una complejidad tal que imposibilita el estudio analítico. De hecho, los modelos analíticos existentes en la actualidad solo son válidos para redes sencillas formadas por un número muy reducido de nodos.

Para comprender este problema basta con mirar en detalle el comportamiento de los enlaces de una red, los cuales actúan como un sistema de colas conectadas unas con otras (figura 3.36(a)). Dado que las llamadas llegan a la red siguiendo una distribución de *Poisson* y, suponiendo independencia entre enlaces, el comportamiento de cada enlace sigue el de una cola $M/G/N/N$, donde N indica el número de fuentes que se pueden



(a) Esquema del modelo de colas de una red de comunicaciones

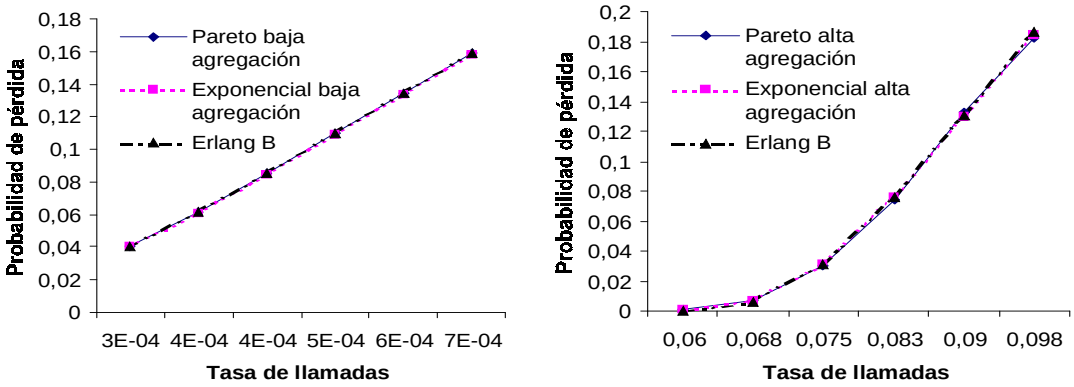
(b) Detalle del modelo de colas

Figura 3.36: Esquema del una red orientada a conexión y detalle de su sistema de colas

servir simultáneamente. Así pues, cualquier red de comunicaciones orientada a conexión no es más que un conjunto de colas conectadas entre sí, por lo que todo camino que conecta un par de nodos está formado por una sucesión de colas donde cada conexión compite por una serie de recursos en cada una de las colas que atraviesa.

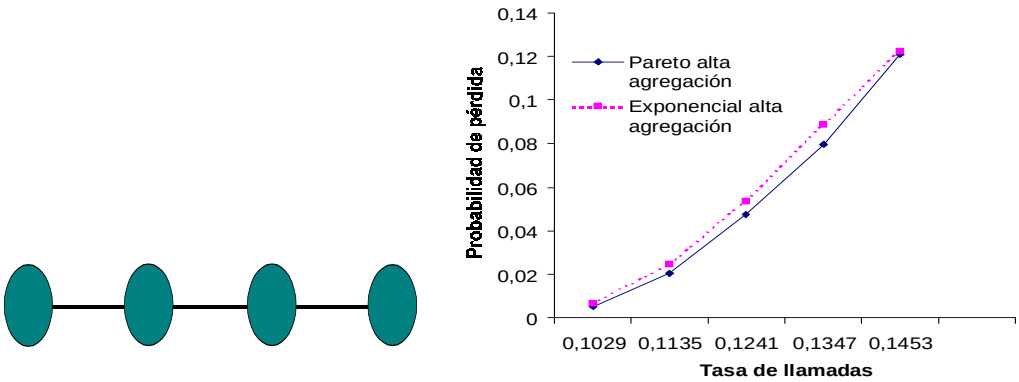
El comportamiento de una cola individual es independiente del tipo de distribución empleada [Aal00] como se muestra en la figura 3.37 para dos valores de N, donde se observa que la probabilidad de pérdida de llamada es independiente del número N y de la distribución empleada. En este caso, empleando el mismo valor medio, se comparan la distribución exponencial y una de Pareto con varianza infinita, la cual se definirá más adelante. La figura prueba, lógicamente, la coincidencia con el valor teórico ofrecido por la fórmula de *Erlang-B*. Sin embargo, en un sistema compuesto por una serie de colas conexionadas sí puede tener influencia el tipo de distribución usada. A fin de verificar esta influencia realizamos una sencilla prueba en la cual, sobre una red muy simple, se obtiene la probabilidad de pérdida de llamada. Los resultados de esta prueba se muestran en la figura 3.38.

En este caso se aprecian, ciertas diferencias en el comportamiento en función de la distribución empleada. Aunque estas diferencias sean pequeñas, es necesario estudiar con más detalle el efecto de las distribuciones en el comportamiento de las redes de comunicaciones [ACS01a][ACS01b], ya que estas diferencias pueden verse aumentadas al incrementarse la complejidad de las redes.



(a) Baja agregación $N = 3$ (b) Alta agregación $N = 100$

Figura 3.37: Probabilidad de pérdida de llamada de un sistema $M/G/N/N$ para dos valores de N y dos distribuciones de la duración de llamada.



(a) Red simple (b) Pérdidas red simple $N = 100$

Figura 3.38: Ejemplo del efecto de la distribución sobre una red simple.

La dificultad del estudio radica en que parte de las conexiones servidas por las colas son finales (es decir, es el enlace final de la conexión) y parte son derivadas a otros enlaces, con lo que aparece un sistema de colas en tándem (figura 3.36(b)). Para complicar el modelo, el número de colas en tándem es variable y depende del camino. Además, una cola puede ser simultáneamente cola final, cola inicial y cola intermedia al formar parte de distintos caminos.

Actualmente no existen modelos analíticos capaces de modelar el comportamiento de un sistema de esta complejidad. Por esta razón, este estudio se ha basado en la simulación del rendimiento de un algoritmo de encaminamiento cuando el tráfico ofrecido a la red sigue distintas funciones de distribución para modelar la duración de las llamadas.

3.7.1 Pruebas y resultados

La red empleada en la pruebas es la versión modificada de la red MCI utilizada en los apartados anteriores con los enlaces limitados a una capacidad de 50 Mbits/s. En cuanto al algoritmo de asignación de costes usado en estas pruebas se ha optado por el **Lineal** utilizado en apartados anteriores.

Para las simulaciones de las llamadas se han considerado los dos escenarios planteados en la sección 3.5 con los mismos tipos de tráfico.

Se han usado distintas funciones de distribución para modelar el tiempo t de duración considerando varios casos límites en cuanto a su variabilidad. No obstante, todas fueron diseñadas para un mismo valor de duración medio ($1/\mu_t$). En concreto, las funciones utilizadas han sido:

Función exponencial En este caso la función de densidad de probabilidad viene dada por:

$$f_{exp}(t) = \begin{cases} \mu_t \, e^{-t\,\mu_t} & si \;\; t \geq 0 \\ 0 & en \;\; otro \;\; caso \end{cases} \tag{3.21}$$

donde $1/\mu_t$ es la media de la distribución. Esta distribución se caracteriza porque su media y su varianza son iguales.

Pareto En este caso la función de densidad de probabilidad viene dada por la siguiente

expresión:

$$f_{par}(t) = \begin{cases} \frac{\alpha\beta^{\alpha}}{(\beta+t)^{\alpha+1}} & si\ t \geq 0 \\ 0 & en\ otro\ caso \end{cases} \tag{3.22}$$

donde β está relacionada con la media $1/\mu_t$ mediante la expresión $1/\mu_t = \frac{\beta}{\alpha-1}$ y α indica la tasa de la caída de la función de densidad de probabilidad, o dicho de otro modo el grado de subexponenciabilidad. Si la caída es muy lenta, en concreto, si $\alpha < 2$, la distribución tendrá una varianza infinita. Esta subexponencialidad se relaciona con las características fractales o autosemejantes detectadas en distintos tipos de tráfico (Ethernet, video,...) durante los últimos 10 años y que han abierto toda una línea de investigación en el campo de teletráfico. En las pruebas consideramos dos casos de distribución de Pareto con $\alpha = 1,5$ y $\alpha = 1,8$.

Multimodal La duración de las llamadas sólo puede tomar un número limitado y finito de valores. En este trabajo se han empleado dos distribuciones multimodales distintas con dos modas cada una (la distribución solo puede tomar dos valores). Las modas se eligieron en los valores $\frac{1}{Q}$ y Q veces el valor medio $(1/\mu_t)$, donde Q es una constante que determina la separación de las dos modas. En las pruebas se ha trabajado con dos valores de Q, que han sido 16 y 4 (multimodal 16 y 4). La probabilidad de generar una llamada de duración $\frac{1}{Q*\mu_t}$ viene dada por la expresión:

$$p_1 = \frac{Q-1}{Q-\frac{1}{Q}} \tag{3.23}$$

mientras que la probabilidad de generar una llamada de duración $Q * \frac{1}{\mu_t}$ viene dada por la siguiente expresión:

$$p_2 = \frac{1-\frac{1}{Q}}{Q-\frac{1}{Q}} \tag{3.24}$$

donde $1/\mu_t$ es el valor medio de la duración de las llamadas.

Determinista (Var 0) Todas las llamadas tienen exactamente la misma duración, que lógicamente coincidirá con el valor medio $1/\mu_t$. Este tipo de distribución se caracteriza por tener varianza cero, con lo que, junto con la distribución de Pareto, constituyen los dos casos extremos.

En todos los casos se ha supuesto una duración media de llamada $(1/\mu_t)$ de 1200 segundos.

En la figura 3.39 se pueden observar las funciones de distribución correspondientes a cada una de las funciones de distribución de probabilidad utilizadas en las pruebas. Los distintos resultados obtenidos se han representado con un margen de confianza del 95% calculado mediante la distribución de *t-student* con cuatro grados de libertad. En la figura 3.40 se incluyen las probabilidades de pérdida normalizadas con respecto a las pérdidas de la función exponencial, para los casos preciso e impreciso con alta agregación con sus correspondientes márgenes de confianza.

El primer objetivo es el estudiar el rendimiento en el caso ideal en que existe en los nodos un conocimiento preciso del estado de la red. Esto lo hacemos tanto para una baja como para una alta agregación. En las figuras 3.41 y 3.42 se pueden ver los resultados en términos absolutos y relativos de este experimento. De estas figuras se pueden extraer una serie de observaciones. La primera de éstas y la más importante es que, en principio, y al contrario de lo esperado y del comportamiento demostrado en otros sistemas de colas, la distribución empleada para modelar la duración no influye de forma determinante en el rendimiento del sistema. La segunda observación que se extrae es que el rendimiento empeora al disminuir la varianza de la distribución empleada, en este caso, la función de distribución con varianza cero se comporta como una cota superior, mientras que se puede observar que el comportamiento de las funciones subexponenciales (Pareto) mejora al disminuir su grado de subexponencialidad.

Sin embargo, uno de los resultados más interesantes aparece al comparar la figura 3.41(a) con la figura 3.41(b) y la figura 3.42(a) con la figura 3.42(b). En estas figuras es posible observar (al contrario que cabría esperar por la ley de los grandes números) que el rendimiento de las distintas funciones de distribución está mucho más cercano entre sí en el caso de existir una baja agregación de conexiones en los enlaces.

En las figuras 3.43 y 3.44 se puede observar el rendimiento del sistema cuando existe imprecisión en el conocimiento del estado de los enlaces (para un valor fijo de $Th = 0,8$). Al estudiar los resultados obtenidos se aprecia que en el caso de imprecisión disminuye la importancia de la distribución empleada para modelar la duración media de las llamadas. Sin embargo, todavía se mantiene la tendencia de un menor rendimiento cuanto menor es la varianza, de forma que en el caso en que todas las llamadas tienen igual duración, el rendimiento final es menor.

Al estudiar el efecto de la agregación de tráfico en las distintas distribuciones (figuras 3.41 y 3.43) se observa que, al igual que lo observado en el apartado 3.5 para la

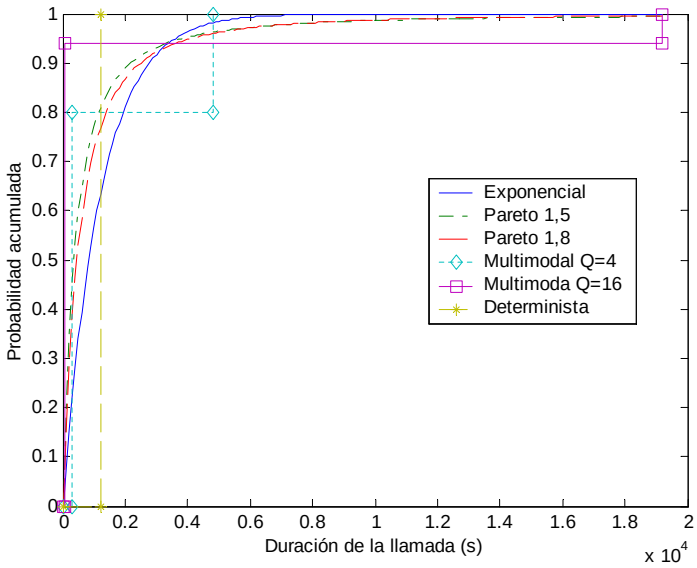
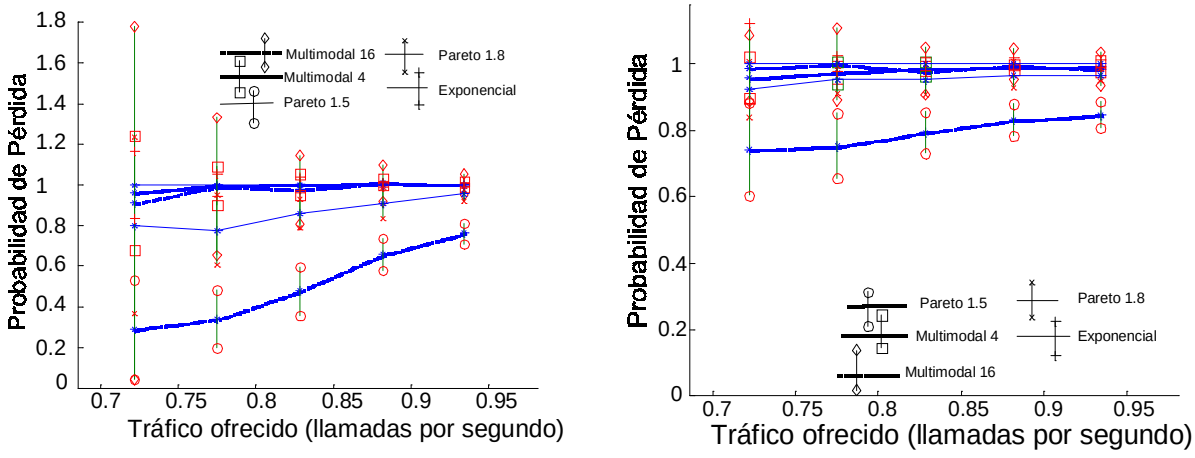
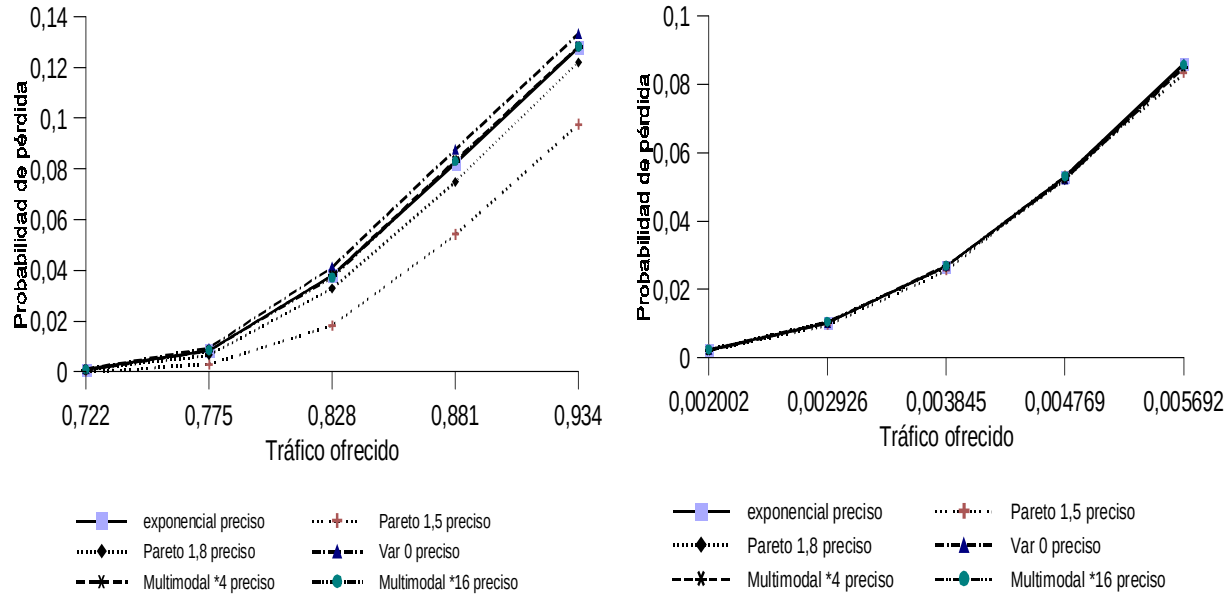


Figura 3.39: Funciones CDF de las distintas funciones de probabilidad utilizadas en las pruebas



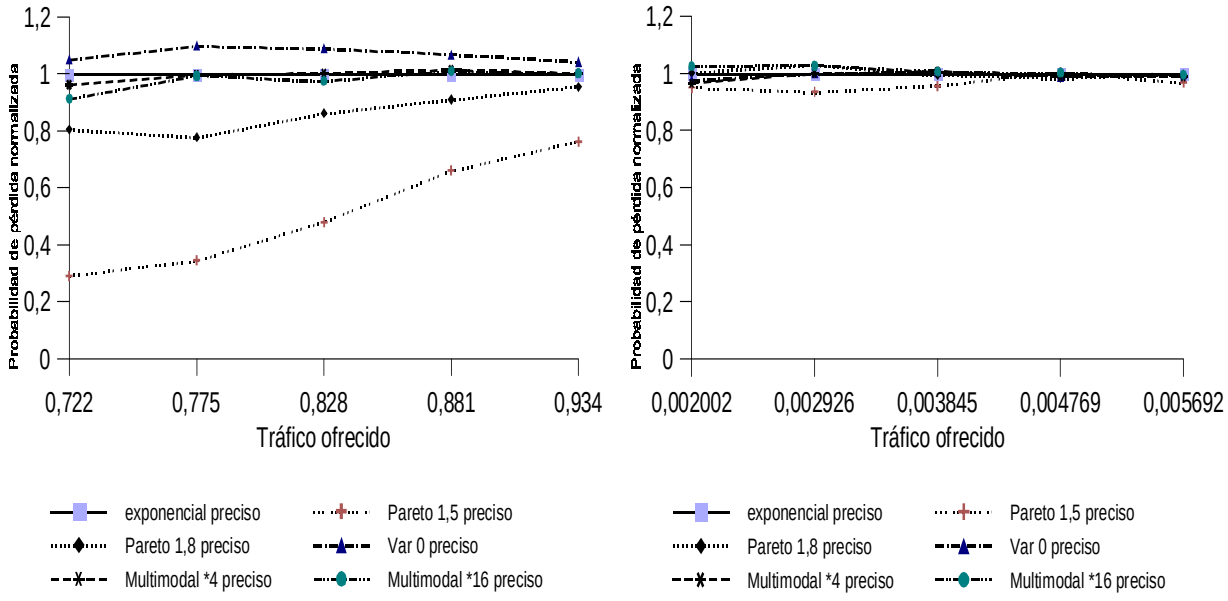
(a) Preciso (b) Impreciso

Figura 3.40: Comparación de las probabilidades de pérdida normalizadas así como sus márgenes de confianza para los casos preciso e impreciso con alta agregación



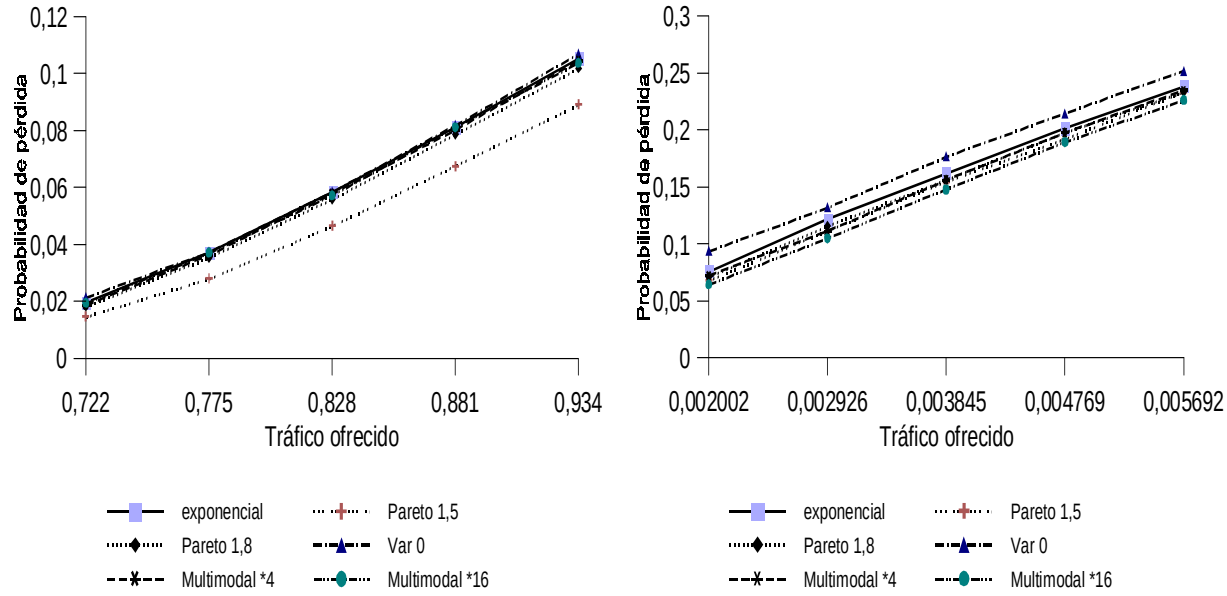
(a) Alta agregación (b) Baja agregación

Figura 3.41: Comparación de las probabilidades de pérdida con un conocimiento preciso del estado de la red



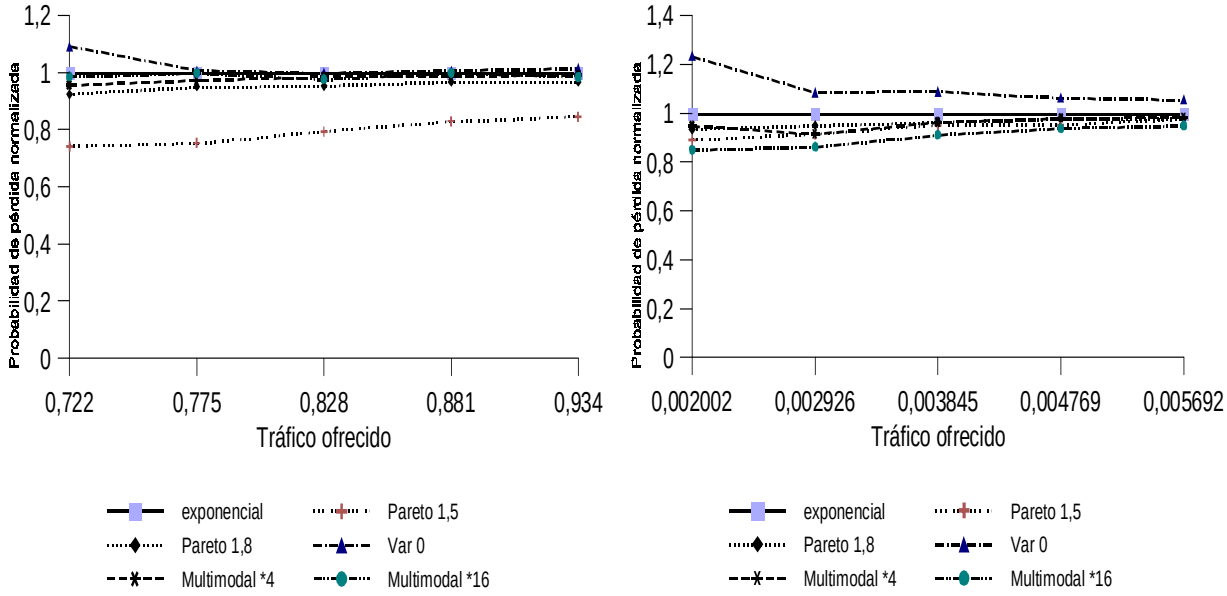
(a) Alta agregación (b) Baja agregación

Figura 3.42: Comparación de las probabilidades de pérdida normalizadas por la probabilidad de pérdida de la distribución exponencial con conocimiento preciso del estado de la red



(a) Alta agregación (b) Baja agregación

Figura 3.43: Comparación de las probabilidades de pérdida con un conocimiento impreciso del estado de la red, tasa de actualización del 80%



(a) Alta agregación (b) Baja agregación

Figura 3.44: Comparación de las probabilidades de pérdida normalizadas por la probabilidad de pérdida de la distribución exponencial con conocimiento impreciso del estado de la red, tasa de actualización 80%

función exponencial, mejora el rendimiento conforme aumenta el tráfico cuando existe imprecisión y una alta agregación de tráfico.

3.8 Conclusiones

De todos los resultados anteriores podemos entresacar y resumir las siguiente conclusiones:

- Aunque el problema de búsqueda de camino sujeto a múltiples restricciones es un problema **NP-Completo**, es posible diseñar algoritmos de búsqueda que resuelven el problema en un tiempo polinómico. Sin embargo, al tratarse de aproximaciones, todas las alternativas presentan limitaciones, ya sea bien por presentar un tiempo de cómputo excesivo para resultar útil su implementación o por no garantizar que encontrará el camino en el 100% de los casos si se pretende acotar el tiempo de cómputo a un valor razonable.

- La función de coste que mejor comportamiento presenta desde el punto de vista de probabilidad de bloqueo es la **Hiperbólica**, aunque ofrece una mayor tasa de actualizaciones que puede desembocar en inestabilidades.

- La función **Lineal** es una solución de compromiso entre *throughput* o aprovechamiento de la red y tasa de actualización.

- Las técnicas predictivas básicas no incrementan el rendimiento de los algoritmos y su funcionamiento es limitado. El algoritmo ***Security Shortest*** presenta un buen comportamiento para valores de tráfico pequeños. El método pesimista sólo funciona bien para umbrales de actualización pequeños y tasas bajas mientras que el método optimista únicamente presenta un rendimiento aceptable para tasas altas de tráfico.

- Las técnicas multicamino permiten mejorar el rendimiento de los algoritmos básicos, especialmente el **Lineal**. A coste de aumentar la complejidad en la fase de establecimiento se consigue un mayor *throughput*. Para el algoritmo **Lineal** con multicamino la tasa de actualizaciones de llamada aumenta aunque permanece por debajo de la tasa del algoritmo **Hiperbólico**. El rendimiento del algoritmo ***WS*** mejora aunque su rendimiento final es inferior al algoritmo **Lineal**. Las pruebas

han demostrado que se pueden obtener excelentes resultados limitando el número de caminos que se prueban en paralelo a dos (optimista y medio). El método multicamino secuencial no aporta grandes ventajas y ninguno de los métodos multicamino aplicado al algoritmo **Hiperbólico** consigue aumentar en gran medida su rendimiento excepto para bajas tasas de tráfico.

- La agregación de tráfico tiene una gran influencia en el rendimiento del sistema cuando existe imprecisión en los datos, presentando peor comportamiento en caso de imprecisión cuanto menor sea la capacidad de agregación.

- En caso de una elevada capacidad de agregación el sistema presenta un mejor comportamiento para elevadas cargas cuanto mayor sea el umbral de actualización debido a que el sistema tiende a ser más conservativo con los recursos cuanto mayor sea este. Esto hace que en condiciones de alta carga y para las funciones de coste analizadas pueda llegar a ser más interesante tener un conocimiento parcial del estado de la red.

- El algoritmo presentado para umbral adaptativo consigue mejorar el rendimiento de los sistemas con una elevada capacidad de agregación gracias a su capacidad de adaptar el umbral en función del tráfico ofrecido a la red. Este algoritmo es fácil de implementar en su versión básica. Por el contrario, en redes con variación dinámica de la tasa de tráfico ofrecida a la red, es necesario modificar el algoritmo básico permitiendo que la modificación del umbral evolucione de forma dinámica a fin de poder adaptarse a variaciones bruscas del tráfico ofrecido a los enlaces.

- La función que modela la duración de las llamadas es poco significativa para el rendimiento de la red. Teniendo en cuenta esta circunstancia, es perfectamente razonable emplear, por tratabilidad analítica, bien una función exponencial o una función determinista para modelar la duración de las llamadas, con la seguridad de que el resultado con los modelos reales va a ser similar o incluso mejor que el esperado (por ejemplo, cuando la duración de las llamadas sigue una distribución de *Pareto* los resultados finales suelen ser mejores que los obtenidos con la distribución exponencial). En cualquier caso, los resultados parecen indicar que si la capacidad de los enlaces es elevada, el rendimiento del sistema empeora cuanto menor es la varianza.

Capítulo 4

Conclusiones y futuros trabajos

4.1 Conclusiones finales

En esta tesis se ha estudiado el comportamiento de los algoritmos de encaminamiento en una red realista en diversas condiciones de carga, así como el efecto de la imprecisión en los datos y de las distribuciones que modelan la duración de llamada en el rendimiento final. Así mismo, se ha prestado atención a los mecanismos de actualización de los datos con el objetivo de maximizar el rendimiento de los algoritmos. A continuación expondremos un resumen de las conclusiones extraídas a lo largo de este trabajo:

1. Es posible diseñar algoritmos de encaminamiento de altas prestaciones adaptados a entornos particulares, como puede ser la red *ATM*. Si se puede caracterizar perfectamente el tráfico y el entorno de trabajo, es factible diseñar un algoritmo de alto rendimiento para ese sistema concreto. Sin embargo, estos algoritmos presentan una fuerte dependencia con el entorno para el que fueron diseñados, circunstancia que los hace poco adecuados para su implementación en sistemas como son las redes actuales, donde la variedad de topologías, tipos de enlaces o la diversidad de tráfico presente en la red prácticamente imposibilitan el caracterizar de forma precisa el entorno de trabajo. En esta situación, es preferible utilizar algoritmos generalistas, a pesar de que éstos presenten un peor rendimiento que los especializados en un entorno particular.

2. La aplicación de los algoritmos neuronales a los problemas de encaminamiento solo se justifica en situaciones concretas, como pueda ser el cálculo de funciones

sin tratabilidad analítica o aquellas donde el tiempo de cómputo es crítico. Sin embargo, si se dan las circunstancias de tratabilidad analítica y los tiempos de cómputo están acotados dentro de las necesidades de la aplicación, no es aconsejable el uso de redes neuronales debido a que la complejidad añadida al sistema es superior a todas las posibles ventajas aportadas.

3. Bajo determinadas situaciones es recomendable implementar mecanismos predictivos que permitan a los algoritmos de encaminamiento adelantarse a situaciones de sobrecarga de la red, evitando en algunas situaciones que esta sobrecarga llegue a producirse y mejorando el rendimiento de los algoritmos básicos. Sin embargo, la utilización de la predicción no está exenta de problemas, siendo el mayor de éstos la influencia que la propia predicción introduce en la variación temporal de los datos, circunstancia que hace que la predicción pueda llegar a ser completamente ineficaz.

4. Al estudiar el comportamiento de los sistemas en presencia de imprecisión en los datos se llega a la conclusión de que es conveniente la utilización de algoritmos de encaminamiento sencillos, ya que éstos tienden a ser más robustos en situaciones de desconocimiento del estado de la red y de las características del tráfico. Se ha podido constatar que, en estas condiciones, las funciones de coste **Lineal** e **Hiperbólica** son las que mejor rendimiento presentan por ofrecer un equilibrio entre rendimiento y conservación de recursos al limitar el número de actualizaciones generadas.

5. El grado de agregación en condiciones de imprecisión afecta de forma importante al rendimiento de los algoritmos, siendo el sistema tanto más vulnerable a la imprecisión en los datos cuanto menor es el grado de agregación de las llamadas.

6. El método de actualización de estado de los enlaces más eficiente es la actualización por variación porcentual del ancho de banda residual, método que hemos denominado actualización por umbral proporcional. Mediante este método se consigue un mejor rendimiento, en términos de ancho de banda perdido por mensajes de actualización enviados, que el logrado por otros métodos comúnmente utilizados, como es la actualización por intervalos fijos de tiempo.

7. Al mismo tiempo, hemos presentado una variación del método anteriormente comentado, que denominamos actualización por umbral adaptativo, y que logra un

mejor rendimiento que el método básico, ya que consigue un mejor aprovechamiento de los recursos variando el umbral en función del número de mensajes de actualización generados. En este sentido, se prueba que en condiciones de baja carga se debe primar la distribución paritaria del ancho de banda entre los caminos mientras que para cargas altas la búsqueda del camino más corto resulta prioritaria. Esta mayor importancia de la topología reduce la necesidad de actualizaciones que provocan sobrecarga en la red sin introducir mejoras en términos de pérdidas.

8. El rendimiento de los algoritmos en condiciones de imprecisión es mejorable mediante la utilización de técnicas multicamino. Sin embargo, estos métodos son computacionalmente costosos y la mejora en el rendimiento es pequeña para justificar su implementación, excepto en situaciones particulares donde sea necesario por cualquier medio obtener el mayor rendimiento posible.

9. Es posible solucionar el problema del encaminamiento con calidad de servicio extremo a extremo mediante el uso de algoritmos de búsqueda de camino con restricciones. Estos algoritmos son capaces de encontrar un camino que cumpla simultáneamente múltiples restricciones de calidad de servicio. De hecho, hemos presentado un algoritmo de encaminamiento capaz de resolver el problema con múltiples cotas en un tiempo acotado polinómicamente. El algoritmo presentado no garantiza que encontrará el camino de coste mínimo sujeto a restricciones, sino que garantiza que será capaz de encontrar dicho camino con una cierta probabilidad.

10. La función de distribución empleada para modelar la duración de las llamadas no influye en el rendimiento final del sistema, siendo el valor medio el que determina éste. Esta situación permite optar por las distribuciones exponenciales, las cuales facilitan la tratabilidad analítica.

4.2 Futuros trabajos

Hay dos aspectos de este trabajo que es necesario seguir estudiando con más detalle, el primero es el efecto de las distribuciones empleadas para modelar la duración de las llamadas en el rendimiento final del sistema. Se ha podido comprobar que estas distribuciones no presentan una influencia significativa en dicho rendimiento, aunque en las pruebas ha aparecido una cierta tendencia a que la evolución del ancho de banda

perdido mejore conforme aumenta la varianza de la distribución usada. Sin embargo, este no es un hecho concluyente, lo que hace necesario estudiar en detalle el porqué de este comportamiento.

El segundo aspecto que es necesario desarrollar con más detalle es el algoritmo de umbral adaptativo. Hasta este momento se ha empleado un mecanismo de incremento/decremento fijo, pero es necesario desarrollar un mecanismo de incremento dinámico que permita una rápida convergencia del algoritmo en condiciones de variación de tráfico, circunstancias en las que se producen rápidas oscilaciones del tráfico ofrecido a los enlaces y, por lo tanto, variaciones en el número de mensajes de actualización que se envían dentro de cada intervalo de tiempo de estimación. Mediante el empleo de un mecanismo adaptativo se podría conseguir una rápida convergencia, evitando las fuertes oscilaciones que se producen con el algoritmo básico, en el número de mensajes de actualización.

Dejando a un lado estos efectos puntuales, entendemos que las futuras investigaciones se deben centrar en el desarrollo de algoritmos de encaminamiento *multicast* (de varios a varios) y *anycast* (a cualquiera dentro de un grupo) garantizando calidad de servicio.

En especial, es necesario resolver el problema del encaminamiento *multicast* en entornos dinámicos donde el número de nodos conectados al grupo varía con el tiempo. Para comprender este problema pongamos como ejemplo un sistema de videoconferencia diseñado para soportar un grupo de discusión. En este sistema el número de usuarios conectados cambia con el tiempo, de forma que en cualquier momento un nuevo usuario puede conectarse al grupo o alguien conectado desconectarse. Esto implica que el árbol que conecta a los distintos usuarios entre sí es dinámico, ya que en cualquier momento pueden surgir nuevas ramas o pueden ser podadas algunas de las existentes. Esta circunstancia obliga a estar continuamente replanteando el árbol de conexión, problema que se agrava si, además, se deben de satisfacer requisitos de *QoS* heterogéneos, en un escenario en donde los diversos usuarios precisan requerimientos distintos. Aunque existen varias propuestas en este sentido, este problema no está de ninguna manera cerrado y es en este momento foco de una intensa actividad investigadora.

Por último, es necesario desarrollar mecanismos de encaminamiento que permitan trabajar de forma simultánea con tráfico orientado a conexión, con requisitos de *QoS* y con tráfico *best-effort* en formato datagrama [Ma98], ya que el entorno más probable para las futuras redes de comunicaciones es que estén basadas en la tecnología *IP* y

soporten simultáneamente datagramas y tráfico orientado a conexión con requisitos de *QoS*.

Bibliografía

[Aal00] S. Aalto. "Introduction to Teletraffic Theory" Laboratory of Telecommunications Technology. Publicado en Internet, http://keskus.hut.fi/opetus/s38145/ (2000).

[ACFH91] G. Ash, J. Chen, A. Frey y B. Huang. "Real-Time Network Routing in a Dynamic Class of Service Network". En *Proceedings of the 13th International Teletraffic Congress (ITC'13)*, Copenague (Junio 1991). North Holland Elservier Science Publisher.

[ACS00a] A. Ariza, E. Casilari y F. Sandoval. "Encaminamiento en Redes Orientadas a Flujo con Imprecisión." En *Actas de URSI 2000*, Zaragoza (Septiembre 2000).

[ACS00b] A. Ariza, E. Casilari y F. Sandoval. "QoS Routing under outdate knowledge of the network." En *Proceedings of Eighth IFIP Workshop on Performance Modelling and Evaluation of ATM&IP Networks*, Ilkley (Julio 2000).

[ACS00c] A. Ariza, E. Casilari y F. Sandoval. "QoS routing with outdated network knowledge". *Electronics Letters*, 36(15), 1332–1334 (Julio 2000).

[ACS00d] A. Ariza, E. Casilari y F. Sandoval. "Strategies for Updating Link States in QoS Routers". *Electronics Letters*, 36(20), 1749–1750 (Septiembre 2000).

[ACS01a] A. Ariza, E. Casilari y F. Sandoval. "Efecto de la distribución de la duración de las llamadas sobre los algoritmos de encaminamiento". En *Actas de JITEL 01*, Barcelona (Septiembre 2001). Aceptado, pendiente de publicación.

[ACS01b] A. Ariza, E. Casilari y F. Sandoval. "Impact of Call Holding Time Ditribution on QoS Routing". En *Proceedings of IASTED International Conference on Advances in Communication (AIC2001)*, Rodas (Julio 2001). Aceptado, pendiente de publicación.

[ACS01c] A. Ariza, E. Casilari y F. Sandoval. "QoS Routing with adaptive updating of link states". *Electronics Letters*, 37(9), 604–606 (Abril 2001).

[AGKT99] G. Apostolopoulos, R. Guerin, S. Kamat y S. Tripathi. "Improving QoS Routing Performance Under Inaccurate Link State Information". En *Proceedings of the 16th International Teletraffic Congress (ITC'16)*, páginas 7–11, Edimburgo (Junio 1999). North Holland Elservier Science Publisher.

[AKW+99] G. Apostolopoulos, S. Kamat, D. Williams, R. Guerin, A. Orda y T. Przygienda. "QoS Routing Mechanisms and OSPF Extensions". RFC 2676, IETF (1999).

[ALGS96] A. Ariza, A. García Lopera, E. Gonzalez y F. Sandoval. "Encaminamiento en redes M.T.A. mediante redes neuronales artificiales". En *Actas de Telecom I+D 96*, Madrid (1996).

[ALGS98] A. Ariza, A. García Lopera, E. Gonzalez y F. Sandoval. "Encaminamiento para múltiples clases de llamada en redes MTA." En *Actas de URSI 98*, Pamplona (1998).

[AOS99] A. Ariza, M.C. Oria y F. Sandoval. "Estimación de carga de tráfico aplicada al encaminamiento de llamadas." En *Actas de Jitel 99*, Madrid (1999).

[Apo99] G. Apostolopoulos. "Cost and Performance Trade-Offs of Quality of Service Routing". Tesis Doctoral, University of Maryland (Agosto 1999).

[ASGV98] A. F. Atlasis, M. P. Saltouros, G.I.Stassinopoulos y A. V. Vasilakos. "An Adaptive Routing Algorithm for ATM Networks Using a Learning Automaton". En *Proceedings of the IEEE Globecom'98*, Sidney (Noviembre 1998).

[Ash85] G. Ash. "Use of Trunk Status Map for Realtime DNHR". En *Proceedings of the 11th International Teletraffic Congress (ITC'11)*, Tokio (Septiembre 1985). North Holland Elservier Science Publisher.

[Ash97] G. Ash. *Dynamic Routing in Telecommunications Networks*. McGraw-Hill (1997).

[Bel57] R. E. Bellman. *Dynamic Programming*. Princeton University Press, Princeton, N.J. (1957).

[BJ99] F. Barceló y J. Jordán. "Channel Holding Time Distribution in Public Cellular Telephony". En *Proceedings of the 16th International Teletraffic Congress (ITC'16)*, páginas 107–116, Edimburgo (Junio 1999). North Holland Elservier Science Publisher.

[Bol94] V. A. Bolotin. "Modeling Call Holding Time Distributions for CCS Network Design and Performance Analysis". *IEEE JSAC*, 12(3), 433–438 (Abril 1994).

[Bou88] J. Boucher. *Voice Teletraffic Systems Engineering*. Artech House (1988).

[BZ96] J. Bennett y H. Zhang. "WF2Q: Worst-case Fair Weighted Fair Queueing". En *Proceedings of the IEEE Infocom'96*, San Francisco (Marzo 1996).

[BZ97] J. Bennett y H. Zhang. "Hierarchical Packet Fair Queueing Algorithms". *IEEE/ACM Transactions on Networking*, 5(5), 675–689 (Octubre 1997).

[BZB+97] R. Braden, L. Zhang, S. Berson, S. Herzog y S. Jamin. "Resource ReSerVation Protocol (RSVP)". RFC 2205, IETF (Septiembre 1997).

[CDF+99] R. Callon, P. Doolan, N. Feldman, A. Fredette, G. Swallow y A. Viswanathan. "A Framework for MPLS". Informe técnico, *Draft* (Borrador) del IETF (Septiembre 1999).

[CGR96] B. V. Cherkassky, A. Goldberg y T. Radzik. "Shortest Paths Algorithms: Theory and Experimental Evaluation". *Mathematical Programming*, 73(2), 129–174 (1996).

[CGS96] B.V. Cherkassky, A.V. Goldberg y C. Silverstein. "Buckets, Heaps, Lists, and Monotone Priority Queues". Informe técnico, NEC Research Institute Inc. (1996).

[Che99] S. Chen. "Routing Support for Providing Guaranteed End-to-End Quality of Service". Tesis Doctoral, University of Illinois at Urbana-Champaign (1999).

[CMM95] E.I. Chong, S. Maddila y S. Morley. "On Finding Single-Source Single-Destination K Shortest Paths". En *Proceedings of the International Conference on Computing and Information (ICCI'95)*, páginas 40–47, Ontario (Julio 1995).

[CN98a] S. Chen y K. Nahrstedt. "On finding Multi-Constrained path". En *Proceedings of the IEEE ICC'98*, Atlanta (Junio 1998).

[CN98b] S. Chen y K. Nahrstedt. "An Overview of Quality-of-Service Routing for the Next Generation High-Speed Networks: Problems and Solutions". *IEEE Networks*, 12(6), 64–79 (Noviembre-Diciembre 1998).

[CN99] S. Chen y K. Nahrstedt. "Distributed Quality-of-Service Routing in Ad-Hoc Networks". *IEEE JSAC*, 17(8), 1–18 (Agosto 1999).

[CNRS98] E. Crawley, R. Nair, B. Rajagopalan y H. Sandick. "A Framework for QoS-based Routing in the Internet". RFC 2386, IETF (Agosto 1998).

[CSM93] S. Cavalieri, A. Di Stefano y O. Mirabella. "Neural Strategies to Handle Routing in Computer Networks". *International Journal of Neural Systems*, 4(3), 269–289 (Septiembre 1993).

[CU92] A. Cichocki y R. Unbehauen. *Neural Networks for Optimization and Signal Processing*. John Wiley (1992).

[Dij59] E. Dijkstra. "A note on Two Problems in Connection with Graphs". *Numerische Mathematik*, 1, 269–271 (1959).

[DKS89] A. Demers, S. Keshav y S. Shenker. "Analysis and Simulation of A Fair Queueing Algorithm". En *Proceedings of the ACM SIGCOMM'89*, Austin (1989).

[DM89] Z. Dziong y L. Manson. "Control of Multi-Service Loss Networks". En *Proceedings of the 28th Conference on Decision and Control*, Tampa (Diciembre 1989).

[Dzi97] Z. Dziong. *Atm Network Resource Management*. Mc GrawHill (1997).

[EJS94] A. Díaz Estrella, A. Jurado y F. Sandoval. "New training patterns selection method for ATM call admission neural control". *Electronics Letters*, 30(7), 577–579 (1994).

[Fab98] R. Fabregat. "Congestion Probability Routing in Virtual Path ATM Networks". Tesis Doctoral, Universitat De Girona (1998).

[FLYV93] V. Fuller, T. Li, J. Yu y K. Varadhan. "Classless Inter-Domain Routing (CIDR): An Address Assignment and Aggregation Strategy". RFC 1519, IETF (Septiembre 1993).

[For96] ATM Forum. "Private Network-Network Interface Specification Version 1.0 (PNNI 1.0)". Informe técnico, ATM Forum (1996).

[FS91] J. A. Freeman y D. M. Skapura. *Neural Networks, Agorithms, Applications, and Programming Techniques*. Addinson- Wesley (1991).

[Gaw95] R. Gawlick. "Admission Control and Routing: Theory and Practice". Tesis Doctoral, MIT (1995).

[GGPS96] L. Georgiadis, R. Guerin, V. Peris y K. Sivarajan. "Efficient Network QoS Provisioning Based on per Node Traffic Shaping". *IEEE/ACM Transactions on Networking*, 4(4), 482–501 (Agosto 1996).

[GGV98] R. G. Garroppo, S. Giordano y A. Vaccaro. "A Teletraffic Analysis of Dial-up Connections over PSTN". En *Proceedings of the IEEE Globecom'98*, Sidney (Noviembre 1998).

[Gir90] A. Girard. *Routing and Dimensioning in Circuit-Switched Networks*. Addison-Wesley (1990).

[GJ79] M. Garey y D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman and Co., New York (1979).

[GKK88] R. J. Gibbens, F. P. Kelly y P. B. Key. "Dynamic Alternative Routing Modeling and Behaviour". En *Proceedings of the 12th International Teletraffic Congress (ITC'12)*, Turin (Junio 1988). North Holland Elservier Science Publisher.

[GO97] R. Guerin y A. Orda. "Qos-Based Routing in Networks with Inaccurate Information: Theory and Algorithms". En *Proceedings of the IEEE Infocom'97*, Kobe (Abril 1997).

[GO99] R. Guerin y A. Orda. "QoS-based Routing in Networks with Inaccurate Information: Theory and Algorithms". *IEEE/ACM Transaction on Networking*, 7(3), 350–364 (Junio 1999).

[Gol94] S. Golestani. "A Self-Clocked Fair Queueing Scheme for Broadband Applications". En *Proceedings of the IEEE Infocom'94*, Toronto (Junio 1994).

[GS95] A.V. Goldberg y C. Silverstein. "Implementations of Dijkstra's Algorithm Based on Multi-Level Buckets". Informe técnico, NEC Research Institute Inc. (1995).

[Hed88] C. Hedrick. "Routing Information Protocol". RFC 1058, IETF (1988).

[Hop82] J. Hopfield. "Neural Networks and Physical Systems with Emergent Collective Computational Properties". *Proceedings of the National Academy of Sciences of the USA*, 79, 2554–2588 (1982).

[Hop84] J. Hopfield. "Neurons with Graded Response Have Collective Computational Properties Like Those of two-state Neurons". *Proceedings of the National Academy of Sciences of the USA*, 81, 3088 – 3092 (1984).

[HT85] J. Hopfield y D. Tank. "Neural Computation of Decisions in Optimization Problems". *Biological Cybernetics*, 52, 141–152 (1985).

[Hui97] C. Huitema. *IPv6 The New Internet Protocol*. Prentice Hall, 2ª edición (1997).

[Hwa93] R. Hwang. "Routing in High-Speed Networks". Tesis Doctoral, University of Massachusetts (1993).

[IMYS91] A. Inoue, K. Mase, H. Yamamoto y M Suyama. "A State and Time-Dependent Dynamic Routing Scheme for Telephone Networks". En *Proceedings of the 13th International Teletraffic Congress (ITC'13)*, Copenague (Junio 1991). North Holland Elservier Science Publisher.

[Kel88] F. P. Kelly. "Routing in circuit-switched networks: optimization, shadow prices and decentralization". *Advances in Applied Probability*, 20, 112–144 (1988).

[Kes97] S. Keshav. *An Engineering Approach to Computer Networking*. Professional Computing. Addison-Wesley (1997).

[KH95] A. Koralov y J. Huy. "Dynamic Routing in Circuit-Switched Broadband Networks". Informe técnico, Rutgers University (1995).

[KI95] K. Kawashima y A. Inoue. "State- and Time-Dependent Routing in the NTT Network". *IEEE Communications Magazine*, 33(7), 40–47 (Julio 1995).

[KO88] K. R. Krishnan y T. J. Ott. "Forward-Looking Routing: A New State-Dependent Routing Scheme". En *Proceedings of the 12th International Teletraffic Congress (ITC'12)*, Turin (Junio 1988). North Holland Elservier Science Publisher.

[KPN01] KPNQwest. "La Red". Publicado en Internet, http://www.kpnqwest.es/company/red.htm (2001).

[Kri91] K. R. Krishnan. "Adaptive State-Dependent Routing Using On-Line Trunk-Group Measurement". En *Proceedings of the 13th International Teletraffic Congress (ITC'13)*, Copenague (Junio 1991). North Holland Elservier Science Publisher.

[KZA+97] K. R. Krishnan, L. Zhang, M. Andrews, W. Aiello y S. Bhatt. "A Performance Comparison of Competitive On-Line Routing and State-Dependent Routing". En *Proceedings of the IEEE Globecom'97*, Fenix (Noviembre 1997).

[LHH95] W.C. Lee, M. G. Hluchyi y P. A. Humblet. "Routing Subject to Quality of Service Constrains in Integrated Communications Networks". *IEEE Networks*, 9(4), 14–16 (Julio/Agosto 1995).

[Lin94] K. Lindberger. "Dimensioning and Design Methods for Integrated ATM Networks". En *Proceedings of the 14th International Teletraffic Congress (ITC'14)*, páginas 897–906, Antibes (1994). North Holland Elservier Science Publisher.

[LO98] D. Lorenz y A. Orda. "QoS Routing in Networks with Uncertain Parameters". *IEEE Transactions on Networking*, 6(6), 768–778 (Diciembre 1998).

[Ma98] Q. Ma. "Quality-of-Service Routing in Integrated Service Networks". Tesis Doctoral, Carnegie Mellon University, Pittsburgh (January 1998).

[MAK93] M. K. Mehmet-Ali y F. Kamoun. "Neural Networks for Shortest Path Computation and Routing in Computer Networks". *IEEE Trans. on Neural Networks*, 4(6), 941–954 (1993).

[Moy97] J. Moy. "OSPF Version 2". RFC 2178, IETF (1997).

[Moy98] J. Moy. *OSPF Anatomy of an Internet Routing Protocol*. Addison-Wesley (1998).

[MS97] Q. Ma y P. Steenkiste. "Quality-of-Service Routing with Performance Guarantees". En *Proceedings of the 4th International IFIP Workshop on Quality of Service*, Paris (Mayo 1997).

[MSS00] A. Magi, A. Szentesi y Szviatovszki. "Analysis of link cost functions for PNNI routing". *Computer Networks*, 34(20), 181–197 (2000).

[MSST91] T. Murase, S. Suzuki, S. Sato y T. Takeuchi. "A Call Admisión Control Scheme for ATM Networks Using a Simple Quality Estimate". *IEEE JSAC*, 9(9), 1461–1470 (1991).

[Nb00] H. De Neve y P. Van Mieghem b. "TAMCRA: a Tunable Accuracy Multiple Constraints Routing Algorithm". *Computer Communications*, 23(7), 667–669 (2000).

[NWM77] K. Narendra, E. A. Wright y L. G. Mason. "Application of Learning Automata to Telephone Traffic Routing and Control". *IEEE, Trans. On Systems, Man, and Cybernetics*, SMC-7(11), 785–792 (1977).

[OK85] T. J. Ott y K. R. Krishnan. "State-Dependent Routing of Telephone Traffic and the Use of Separable Routing Scheme". En *Proceedings of the 11th International Teletraffic Congress (ITC'11)*, Tokio (Septiembre 1985). North Holland Elservier Science Publisher.

[Ord98] A. Orda. "Routing with End to End QoS Garantess in Broadband Networks". En *Proceedings of the IEEE Infocom'98*, San Francisco (Marzo 1998).

[Par82] D. B. Parker. Learning-Logic. Invention Report s81-64, File 1, Office of Technology Licensing, Stanford University (1982).

[Plo95] S. Plotkin. "Competitive Routing of Virtual Circuits in ATM Networks". *IEEE JSAC*, 13(8), 1128–1136 (Agosto 1995).

[Prz95] A. Przygienda. "Link State Routing with QoS in ATM LANs". Tesis Doctoral, Swiss Federal Institute of Technology, Thesis Nr. 11051 (Abril 1995).

[PTVF92] W. Press, S. Teukolsky, W. Vetterling y B. Flannery. *Numerical Recipes in C*. Cambridge University Press, 2ª edición (1992).

[Put94] M. L. Puterman. *Markov Decision Processes*. Series in probability and mathematical. John Whiley (1994).

[Ram95] S. Rampal. "Routing and End-To-End Quality of Service in Multimedia Networks". Tesis Doctoral, North Carolina State University (Agosto 1995).

[RBCC95] J. Régnier, F. Bédard, J. Choquete y A. Caron. "Dynamically Controlled Routing in Networks with Non-DCR-Compliant Switches". *IEEE Communications Magazine*, 33(7), 48–57 (Julio 1995).

[RCS99] A.L. Roginsky, K.J. Christensen y V. Srinivasan. "New Methods for Shortest Path Selection for Multimedia Traffic with Two Delay Constraints". *Computer Communications*, 22(17), 1531–1539 (Octubre 1999).

[RE86] D. E. Rumelhart y J. L. McClelland (Eds). *Parallel Distributed Processing*. The MIT Press (1986).

[RH00] D. Towsley R. Hwang, J. Kurose. "MDP Routing for Multi-Rate Loss Network". *Computer Network*, 34(2), 241–261 (Agosto 2000).

[RL95] Y. Rekhter y T. Li. "A Border Gateway Protocol (BGP-4)". RFC 1771, IETF (Marzo 1995).

[RME96] J. Roberts, U. Mocci y J.T. Virtamo (Eds.). *Broadband Network Teletraffic. Final Report of Action COST 242*. Lecture Notes in Computer Science, Vol. 1155. Springer-Verlag (1996).

[Rob81] J.W. Roberts. "A Service System with Heterogeneous user Requirements - Application to multi-service telecommunications systems". En *Proceedings*

of the Performance of Data Communication Systems and their Applications, páginas 423–431. North Holland Elservier Science Publiser (1981).

[Ros95] K. W. Ross. *Multiservice Loss Models For Broadband Telecommunications Networks*. Springer-Verlag (1995).

[RS78] L. R. Rabiner y R. W. Schafer. *Digital Processing of Speech Signal*. Prentice Hall (1978).

[RVC99] E. Rosen, A. Viswanathan y R. Callon. "Multiprotocol Label Switching Architecture". Informe técnico, *Draft* (Borrador) del IETF (Septiembre 1999).

[Sal00] P. Salus. *Big Book of IPv6 Addresing RFCs*. Morgan Kaufman (2000).

[Sha96] Y. Shavitt. "Burst Control in High-Speed Networks". Tesis Doctoral, Technion, Haifa, Israel (1996).

[Sta94] W. Stallings. *Data and Computer Communications*. Prentice-Hall, 4ª edición (1994).

[Ste95] M. Steenstrup. *Routing in Communications Networks*. Prentice Hall (1995).

[Tan97] A. Tanenbaum. *Redes de Computadoras*. Prentice-Hall, 3ª edición (1997).

[TPBO99] N. Taft-Plotkin, B. Bellur y R. Ogier. "Quality-of-Service Using Maximally Disjoint Paths". En *Proceedings of IFIP Seventh International Workshop on Quality-of-Service (IWQoS)*, Londres (Junio 1999).

[WC96a] Z. Wang y J. Crowcroft. "QoS Routing for Supporting Resource Reservation". *IEEE JSAC*, 14(7), 1228–1234 (Septiembre 1996).

[WC96b] Z. Wang y J. Crowcroft. "Quality-of-Service Routing for Supporting Multimedia Applications". *IEEE JSAC*, 14(7), 1288–1294 (Septiembre 1996).

[Wer90] P. Werbos. "Backpropagation Through Time what it Does and How Do It". *Proc. IEEE*, 78(10), 1550–1560 (1990).

[WH91] E. Wallmeier y C. M. Hauber. "Bloking Probabilities in ATM Pipes Controlled by Acceptance Algorithm Based on Mean and Peak Bit Rates". En *Proceedings of the 13th International Teletraffic Congress (ITC´13)*, Copenague (Junio 1991). North Holland Elservier Science Publisher.

[Wid94] R. Widyono. "The Design and Evaluation of Routing Algorithms for Real-Time Channels". Informe técnico TR94024, University of California at Berkeley International Computer Science Institute (Junio 1994).

[WW95] C. Wang y P. N. Weissler. "The Use of Artificial Neural Networks for Optimal Message Routing". *IEEE Network*, 9(2), 16–24 (Marzo/Abril 1995).

[YSSM95] H. Yokoe, S. Shioda, H. Saito y J. Matsuda. "Performance Evaluation of Routing Scheme in B-ISDN". *IEICE Transactions in Communications*, E78-B(4), 514–521 (Abril 1995).

[Zha95] H. Zhang. "Service Disciplines For Guaranteed Performance Service in Packet-Switching Networks". *Proceedings of the IEEE*, 83(10), 1374–96 (Octubre 1995).

[ZT89] L. Zhang y S. C. A. Thormapoulos. "Neural Network Implementation of the Shortest Path Algorithm for Traffic Routing in Communications Networks". En Wizard V. Oz y Mihalis Yannakakis, editores, *Proceedings of the IEEE International Joint Conference of Neural Networks*, Washington (Junio 1989).

Apéndice A

Implementación del algoritmo *K-Shortest* con poda

En las siguientes figuras se muestra el seudocódigo correspondiente a la implementación básica del algoritmo *K-Shortest* con mecanismo de poda utilizado en esta tesis para la búsqueda de caminos con requisitos de *QoS*. En la figura A.1 se presenta el algoritmo básico y en las figuras A.2 y A.3 se muestran el mecanismo de poda y el manejo de la pila, respectivamente.

La implementación del manejo de la pila se ha realizado empleando una lista simplemente enlazada, que si bien es fácil de implementar, ralentiza la ejecución del algoritmo.

```
Dijkstra_modificado(origen,destino,camino,coste_camino,
                    Retardo_limite,probabilidad_limite,
                    saltos_limite,banda_limite){
 Inicilizar_estructuras();
 num_nodos=0;
 Introducir_pila (origen,0,0,0,0,INFINITO,0);
 while(Sacar_pila(&nodo,&coste,&retardo,&perdida,
       &saltos,&banda,&id_camino)){
    estados[nodo][id_camino].etiqueta=permanente;
    if (nodo==destino)  {
      if (estados[nodo][id_camino].coste<coste_maximo){
        coste_camino=estados[nodo][id_camino].coste;
        num_nodos=1;
        i=1;
        camino_aux[0]=destino;
        nodo_aux = nodo;
        idk_aux = id_camino;
        do{
          camino_aux[i]=estados[nodo_aux][idk_aux].predecesor;
          nodo_aux = estados[camino_aux[i]][idk_aux].predecesor;
          idk_aux = estados[camino_aux[i]][idk_aux].prev_idk;
          i++;
          num_nodos++
        }while (nodo_aux!=origen)
        for (i=0;i<num_nodos;i++)
          camino[i]=camino_aux[num_nodos-i-1];
          return(num_nodos);
      }
    }
    for (para todo vertice del nodo)  {
      coste = podar_camino();
      max_coste = Extraer_maximo (i,&max_id_camino);
      if (max_coste>coste){
       Actualizar_estados();
       if(esta_en_pila(i,max_id_camino))
         reemplazar_pila(i,coste,retardo,perdida,
                                 saltos,banda,max_id_camino);
       else
         introducir_pila(i,coste,retardo,perdida,
                                 saltos,banda,max_id_camino);
      }
    }
 }
 return (num_nodos);
}
```

Figura A.1: Algoritmo Básico

```
podar_camino(){
 retardo = estados[nodo][id_camino].retardo +
                              Topologia[nodo][i].retardo;
 saltos = estados[nodo][id_camino].saltos +1;
 perdida = estados[nodo][id_camino].perdida*
                              (1-Topologia[nodo][i].perdida);
 banda = min(estados[nodo][id_camino].banda,
                               Topologia[nodo][i].banda);
 if (Retardo>Retardo_limite ||
    (1-perdida)> probabilidad_limite||
     saltos>saltos_limite || banda<banda_limite)
     coste = estados[nodo][id_camino].coste+INFINITO;
 else
     coste = estados[nodo][id_camino].coste+Topologia[nodo][i].coste;
     return (coste);
}
```

Figura A.2: Mecanismo de poda.

```
introducir_pila (nodo,coste,retardo,perdida,saltos,banda,idk){
 Pila *aux1,*aux2,*Elemento;
 Inicializar_elemento(Elemento,nodo,coste,retardo,
                                idk,perdida,saltos,banda);
/* Inicialización de la pila, Puntero_a_pila señala al primer elemento de la
pila */
 if (Puntero_a_pila==NULL)
    {Puntero_a_pila=Elemento; Puntero_a_pila->siguiente=NULL; return;}
/* Comprobar si se debe colocar como primer elemento de la pila */
 if (Puntero_a_pila->coste>Elemento->coste)
    {Elemento->siguiente = Puntero_a_pila; Puntero_a_pila=Elemento;  return;}
/* En caso contrario se busca en la pila donde ponerlo */
 aux1=Puntero_a_pila;
 do{
    if((Elemento->coste>aux1->coste)&&(aux1->siguiente!=NULL))
         {aux2=aux1;  aux1=aux1->siguiente;  continue;}
    else if ((aux1->siguiente==NULL) && (Elemento->coste>aux1->coste))
          {aux1->siguiente=Elemento; return;}
/* Poner primero el de mayor ancho de banda disponible */
    else if (Elemento->coste==aux1->coste){
         if((Elemento->banda<=aux1->banda)&&(aux1->siguiente!=NULL))
                 {aux2=aux1;  aux1=aux1->siguiente; continue;}
         else if ((aux1->siguiente==NULL) && (Elemento->banda<=aux1->banda))
              {aux1->siguiente=Elemento;  nodo_anterior = nodo; return;}
         else {
              if (aux1!=Puntero_a_pila)
                  {Elemento->siguiente=aux1;  aux2->siguiente=Elemento;}
              else{Elemento->siguiente = Puntero_a_pila; Puntero_a_pila=Elemento;}
              nodo_anterior = nodo;  return;
         }
     }
     else {
         if (aux1!=Puntero_a_pila)
          {Elemento->siguiente=aux1; aux2->siguiente=Elemento;}
         else
          {Elemento->siguiente = Puntero_a_pila; Puntero_a_pila=Elemento;}
         return;
     }
 }while (aux1!=NULL);
 exit (0);
}
```

Figura A.3: Manejo y ordenación de la pila del algoritmo *K-shortest path* modificado.