# Protocolos de Enrutamiento Para la Capa de Red en Arquitecturas de Redes de Datos

Andres Gomez Marquez

Dpto. Informática. Universidad Cooperativa de Colombia.
E-mail: rangoma@hotmail.com

**Resumen.** Los protocolos de enrutamiento para la capa de red son usados para resolver peticiones de servicios de envío de paquetes de datos a través de diferentes redes de datos. El punto más importante de este estudio es mostrar los diferentes algoritmos de enrutamiento que resuelven esta cuestión, y a su vez compararlos en forma cualitativa para conocer cuáles son sus fortalezas y cuáles son sus puntos débiles. También se analiza brevemente qué pasa en la capa de red en la Internet.

Palabras Claves: Protocolo de Enrutamiento – Algoritmos Estáticos – Algoritmos Dinámicos

## 1 Introducción

Teniendo en cuenta las necesidades y los avances producidos en una sociedad sumamente compleja, resulta de gran importancia destacar tanto la transmisión de información, como la necesidad de que ésta llegue a destino en el momento preciso mediante el uso de las redes.

Es a través de la Internet que queda probado, y todos los días se muestra con mejor y mayor detalle, que ésta ha sido y será revolucionaria en las áreas de los servicios financieros, de entretenimiento, salud, educación y gobierno (Mondel y Hof, 2001).

El proceso de digitalización de todas las técnicas de comunicación, transmisión (cable, satélite) y recepción, producen nuevas convergencias entre diferentes sectores (cultura, comunicación, lengua, educación, telecomunicaciones, etc.), pero muy especialmente lo que producen es *la transformación de los "espacios de comunicación"* los límites y las fronteras y, como consecuencia, *la transformación de los espacios de intercambios culturales.* De hecho, todas las sociedades, por definición, han sido y serán "sociedades de la comunicación".

Los principales cambios estructurales de la sociedad se producen ahora entorno del tratamiento y de la transmisión de la información.

La capa de Red, dentro de una arquitectura de rede de datos, es la que se encarga de llevar los paquetes de datos desde el origen (estación transmisora) hasta el destino (estación receptora). Llegar a destino, en tiempo y forma, puede requerir que el

algoritmo de ruteo, que es el encargado de escoger las rutas y las estructuras de datos, cumpla con ciertas propiedades que aseguren la eficiencia de su trabajo.
Estas propiedades son: *corrección, estabilidad, robustez, equitatividad, sencillez y optimalidad*.
La corrección y la sencillez casi no requieren comentarios; no así la necesidad de robustez, la cual se refiere a que el algoritmo debe ser diseñado para que funcione dentro de la red por años, sin fallas generales. El algoritmo deberá estar preparado para manejar cambios de topología y tráfico sin requerir el aborto de las actividades o el rearranque de la red.
La equitatividad y la optimalidad resultan con frecuencia contradictorias, ya que muchas veces se requiere una concesión entre la eficacia global (optimización) y la equitatividad; es decir, antes de intentar encontrar un justo medio entre estas dos, se debe decidir qué es lo que se busca optimizar.
Minimizar el retardo de los paquetes (disminuyendo escalas y ancho de banda) y maximizar el rendimiento total de la red sería la combinación más apropiada para un algoritmo de ruteo.

## 2 Algoritmos de Ruteo

La capa de Red proporciona la dirección lógica que permite que dos sistemas dispares que se encuentran en redes lógicas diferentes determinen una posible ruta para comunicarse.
En la capa de red es donde residen los algoritmos que implementan los protocolos de enrutamiento.
En la mayoría de las subredes, los paquetes requerirán varias escalas para completar el viaje. La excepción serían las redes de difusión, pero aún aquí es importante el enrutamiento, ya que el origen y el destino pueden no estar en la misma red.

El algoritmo de enrutamiento es la parte del software de la capa de red encargada de decidir la línea de salida por la que se transmitirá un paquete de entrada.
Si la subred usa datagramas entonces esta decisión debe hacerse cada vez que llega un paquete de datos de entrada, debido a que la mejor ruta podría haber cambiado desde la última vez.
Si la subred utiliza circuitos virtuales internamente, las decisiones de enrutamiento se tomarán sólo al establecerse el circuito y los paquetes seguirán la ruta previamente establecida.

## 3 Clasificación de los Algoritmos de Enrutamiento

*Algoritmos no adaptables:* No basan sus decisiones de enrutamiento en mediciones o estimaciones del tráfico ni en la topología. La decisión de qué ruta tomar de *I* a *J* se calcula por adelantado, fuera de línea y se cargan en los routers al iniciar la red. Éste procedimiento se llama *enrutamiento estáticos.* La desventaja de este tipo de algoritmos es que no es posible responder a situaciones cambiantes como por ejemplo saturación, exceso de tráfico o fallo en una línea.
En un conjunto de redes complejas, se necesita cierto grado de cooperación "dinámica" entre los dispositivos de encaminamiento. En particular se deben evitar aquellas porciones de red que sufren congestión, entendiéndose esto como aquella situación donde hay demasiados paquetes en alguna parte de la subred, y como consecuencia el rendimiento de ésta baja.
Para poder tomar estas decisiones de encaminamiento dinámicas, los dispositivos involucrados en el ruteo deben intercambiar información usando algoritmos de encaminamiento especiales para este propósito. La información que se necesita sobre el estado del conjunto de redes tiene que venir expresada en términos de qué redes son accesibles a través de qué dispositivos y en términos de las características de retardo de varias rutas.
*Algoritmos adaptables:* En contraste con los algoritmos no adaptables, éstos cambian sus decisiones de enrutamiento para reflejar los cambios de topología y de tráfico. Difieren de los algoritmos estáticos en el lugar de obtención de su información (ej. localmente, en los routers adyacentes o de todos), el momento del cambio de sus rutas (ej. cada $\Delta t$ seg., o cuando cambia la carga) y la métrica usada para la optimalidad (ej. distancia, nº de escalas, tiempo estimado del tránsito). Este tipo de algoritmos no pueden ser demasiado complejos ya que son implementados en los routers y deben ejecutarse en tiempo real con recursos de CPU y la memoria con que el router dispone.

## 4 Principio de Optimización

Este postulado establece que, si el enrutador *J* está en la trayectoria óptima del enrutador *I* al enrutador *K*, entonces la trayectoria óptima de *J* a *K* también está en la misma ruta. Haciendo referencia a la figura 1, llamemos $r_1$ a la parte de la ruta de I a *J*, y $r_2$ al resto de la ruta. Si existiera una ruta mejor que $r_2$ entre *J* y *K*, podría concatenarse con $r_1$ para mejorar la ruta entre *I* y *K*, contradiciendo nuestra aseveración de que $r_1$ y $r_2$ es *óptima.*

Como consecuencia directa del principio de optimalidad, podemos ver que el grupo de trayectorias óptimas de todas las de orígenes a un destino dado forma un árbol con raíz en el destino. Ese árbol que se forma, se llama *árbol de descenso,* donde la métrica de distancia es el número de escalas. El árbol de descenso puede no ser único, pueden existir otros árboles con las mismas longitudes de trayectoria.
La meta de todos los algoritmos de enrutamiento es descubrir y usar los árboles de descenso para todos los enrutadores.

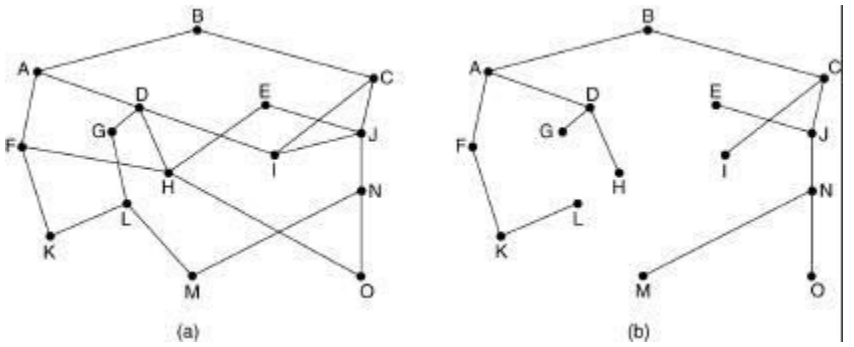


Figura 1. (a) Subred. (b) Árbol descendente para el ruteador B.

Dado que un árbol de descenso ciertamente es un árbol, no contiene ciclos, por lo que cada paquete será entregado con un número de escalas finito y limitado.
En la práctica, no siempre sucede esto, los enlaces y los enrutadores pueden caerse y reactivarse durante la operación, por lo que diferentes enrutadores pueden tener ideas distintas sobre la topología actual de la subred. El fin último de los algoritmos de ruteo es descubrir y usar los árboles de descenso de todos los enrutadores.

## 5 Algoritmos Estáticos

### Enrutamiento por trayectoria más corta

Esta es una técnica de amplio uso en muchas formas, ya que es sencilla y fácil de entender. La idea es armar un grafo de la subred en el que cada nodo representa un enrutador y cada arco del grafo una línea de comunicación (enlace). Para seleccionar la ruta entre un par dado de enrutadores, el algoritmo simplemente encuentra en el grafo la trayectoria más corta entre ellos.
El concepto de *trayectoria más corta* se debe a que la forma de medir la longitud de la ruta es usando alguna métrica, entendiéndose por métrica al peso relativo que se da a cada uno de los factores que intervienen en el cálculo de la distancia en una red, los cuales podrían ser el número de saltos, la distancia física, el retraso de transmisión por un paquete de prueba, el ancho de banda, el tráfico promedio, el costo de comunicación, etc.
Se conocen varios algoritmos de cálculo de la trayectoria más corta entre dos nodos de un grafo. Cada nodo se etiqueta (entre paréntesis) con su distancia al nodo de origen a través de la mejor trayectoria conocida. Inicialmente no se conocen trayectorias, por lo que todos los nodos tienen la etiqueta infinito. A medida que avanza el algoritmo y se encuentran trayectorias, pueden cambiar las etiquetas, reflejando mejores trayectorias. Una etiqueta puede ser tentativa o permanente. Inicialmente todas las etiquetas son tentativas. Al descubrirse que una etiqueta representa la trayectoria más corta posible del origen a ese nodo, se vuelve permanente y no cambia más.

**Inundación**

Otro algoritmo estático es la inundación, en la que cada paquete de entrada se envía por cada una de las líneas de salida, excepto aquella por la que llegó. La inundación evidentemente genera grandes cantidades de paquetes duplicados, de hecho, una cantidad infinita a menos que se tomen algunas medidas para limitar ese proceso. Una de tales medidas puede ser un contador de escalas contenido en la cabecera de cada paquete, el cual disminuye en cada escala, descartándose al llegar el contador a cero. Idealmente el contador debe inicializarse a la longitud de la trayectoria; puede inicializar el contador en el peor de los casos, es decir, el diámetro de la subred.
Una variación de la inundación, un poco más práctica es la *inundación selectiva.* En este algoritmo, los enrutadores no envían cada paquete de entrada por todas las líneas, sino sólo por aquellas que van aproximadamente en la dirección correcta.
La inundación no es práctica en la mayoría de las aplicaciones, pero tiene algunos usos. Por ejemplo, en aplicaciones militares y en las aplicaciones de bases de datos distribuidas a veces es necesario actualizar concurrentemente todas las bases de datos, en cuyo caso puede ser útil la inundación.

**Enrutamiento basado en flujo**

Los algoritmos vistos hasta ahora sólo toman en cuenta la topología; no consideran la carga. Si por ejemplo, siempre hay una gran cantidad de tráfico entre un nodo *A* y un nodo *B*, ambos adyacentes, podría ser mejor enrutar el tráfico de ambos por caminos alternativos un poco más largos tal vez. Seguidamente veremos un algoritmo estático; el enrutamiento basado en flujo usa tanto la topología como la carga para el enrutamiento.
La idea en que se basa el análisis es que, para una línea dada, si se conocen la capacidad y el flujo promedio, es posible calcular el retardo promedio de los paquetes en esa línea a partir de la teoría de colas. De los retardos promedio de todas las líneas, es directo el cálculo de un promedio ponderado por el flujo para obtener el retardo de paquete medio de la subred completa. El problema de enrutamiento se reduce entonces a encontrar el algoritmo de enrutamiento que produzca el retardo promedio mínimo para la subred.
Para usar esta técnica, debe conocerse por adelantado cierta información: primero, la topología de la subred, segundo debe estar dada la matriz de tráfico y tercero debe estar disponible la matriz de capacidad, donde se especifica la capacidad de cada línea en bps. Por último, debe escogerse algún algoritmo tentativo de enrutamiento.

# 6 Algoritmos Dinámicos

## Enrutamiento vector de distancia

Los algoritmos de enrutamiento por vector de distancia operan haciendo que cada enrutador mantenga una tabla (por ejemplo, un vector) que da la mejor distancia conocida a cada destino y la línea a usar para llegar ahí. Estas tablas se actualizan intercambiando información con vecinos.

Este algoritmo recibe otros nombres como: algoritmo de enrutamiento *Bellman-Ford* distribuido y el algoritmo *Ford-Fullkerson*, en reconocimiento a los investigadores que lo desarrollaron.

En el enrutamiento por vector de distancia, cada enrutador mantiene una tabla de enrutamiento indizada por, y conteniendo un registro de, cada enrutador de la subred. Esta entrada comprende dos partes: la línea preferida de salida hacia ese destino y una estimación del tiempo o distancia a ese destino. La métrica usada podría ser la cantidad de escalas, el retardo de tiempo en milisegundos, el número total de paquetes encolados por la trayectoria, o algo parecido.

Se supone que cada enrutador conoce la "distancia" a cada uno de sus vecinos. Si la métrica es de escalas, la distancia simplemente es una escala. Si la métrica es la longitud de la cola, el enrutador simplemente examina cada cola. Si la métrica es el retardo, el enrutador puede medirlo directamente con paquetes especiales de ECO que el receptor simplemente marca con la hora y envía de regreso tan rápido como puede.

Supóngase que se usa como métrica el retardo y que el enrutador conoce el retardo a cada uno de sus vecinos. Cada $T$ mseg, cada enrutador envía a todos sus vecinos una lista de los retardos estimados a cada uno de los destinos. También recibe una lista parecida de cada vecino. Imagine que una de estas tablas acaba de llegar del vecino $X$, siendo $X_i$ la estimación de $X$ respecto al tiempo que le toma llegar al enrutador $i$ a través de $X$ en $X_i + m$ mseg vía $X$ . Efectuando este cálculo para cada vecino, un enrutador puede encontrar la estimación que parezca ser la mejor y usar esa estimación y la línea correspondiente en su nueva tabla de enrutamiento.

Este proceso de actualización se ilustra en la figura 2. En la parte (a) se muestra una subred. En las primeras cuatro columnas de la parte (b) aparecen los vectores de retardo recibidos de los vecinos del enrutador $J$. $A$ indica tener un retardo de 12 mseg a $B$, un retardo de 25 mseg a $C$, un retado de 40 mseg a $D$, etc. Suponiendo que $J$ ha medido o estimado el retardo de sus miembros, $A$, $I$, $H$ y $K$ en 8, 10, 12 y 16 mseg, respectivamente.

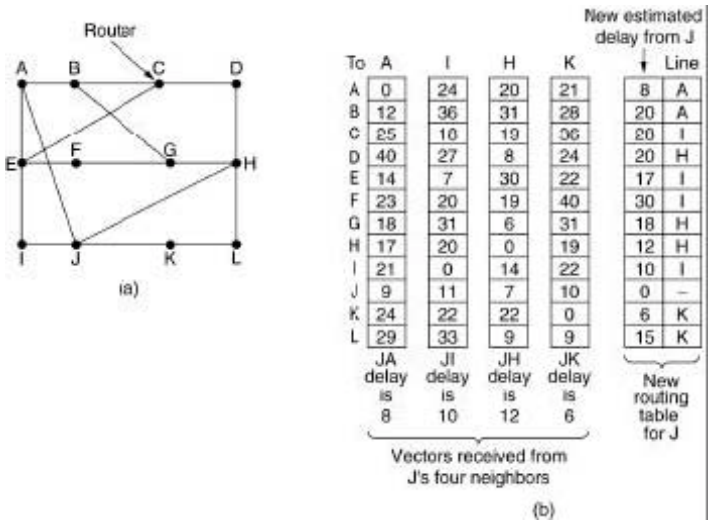


Figura 2. (a) Subred. (b) Entrada de *A, I, H, K*, y la nueva tabla de enrutamiento de *J*.

Considere la manera en que *J* calcula su nueva ruta al enrutador *G*. Sabe que puede llegar a *A* en 8 mseg, y *A* indica ser capaz de llegar a *G* en 18 mseg, por lo que *J* sabe que puede contar con un retador de 26 mseg a *G* si enviaría a *A* los paquetes destinados a *G*. Del mismo modo, *J* calcula el retardo a *G* a través de *I, H* y *K* en 41 (31+10), 18 (6+12) y 37 (31+6 ) mseg, respectivamente. El mejor de estos valores es 18, por lo que escribe una entrada en su tabla de enrutamiento indicando que el retardo a *G* es de 18 mseg, y que la ruta a usar es vía *H*. Se lleva a cabo el mismo cálculo para los demás destinos, y la nueva tabla de enrutamiento se muestra en la última columna de la figura.

**El problema del conteo a infinito**

Este algoritmo funciona bien en teoría, pero tiene un problema serio en la práctica: aunque converge en la respuesta correcta, puede hacerlo lentamente. En particular reacciona con rapidez a las buenas noticias, pero con lentitud ante las malas. Considere un enrutador cuya mejor ruta al destino *X* es larga. Si en el siguiente intercambio el vecino *A* informa repentinamente un retardo corto a *X*, el enrutador simplemente se conmuta a modo de usar la línea a *A* para enviar tráfico hasta *X*. En el intercambio de vectores, se procesan las nuevas noticias.
Para ver la rapidez de propagación de las buenas noticias, considere la subred de 5 nodos (lineal) de la figura 3.

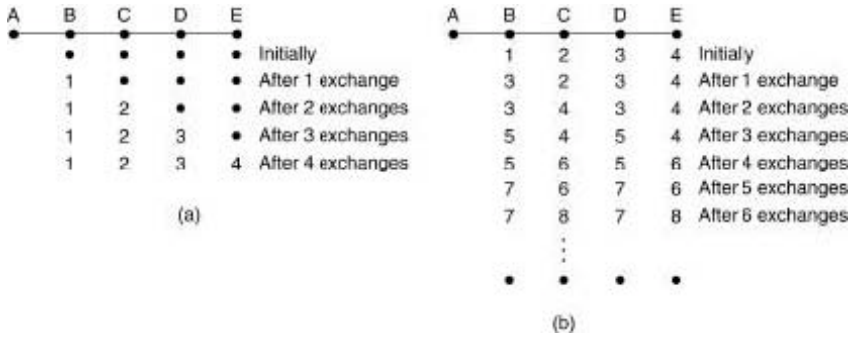


Figura 3. El problema de conteo a infinito.

Al activarse *A*, los enrutadores saben de él gracias a los intercambios de vectores. En el momento del primer intercambio, *B* se entera de que su vecino de la izquierda tiene un retardo de 0 hacia *A*. *B* crea una entrada en su tabla, indicando que *A* está usando una escala de distancia hacia la izquierda. El resto de los enrutadores aún piensan que *A* está desactivado. Las entradas de la tabla de *A* en este punto se muestran en la segunda fila de la figura 3 (a). Durante el siguiente intercambio *C* se entera de que *B* tiene una trayectoria a *A* de longitud 1, por lo que actualiza su tabla de enrutamiento para indicar una trayectoria de longitud 2, pero *D* y *E* no se enteran de las buenas nuevas sino hasta después. Como es evidente, las buenas noticias se difunden a razón de una escala por intercambio. En una subred cuya trayectoria mayor tiene una longitud de *N* escalas, en un lapso de *N* intercambios todo el mundo sabrá las líneas y enrutadores recientemente revividos.

Considerando ahora la situación de la figura 3 (b), en que todas las líneas y enrutadores están activos inicialmente. Los enrutadores *B, C* y *E* tienen distancias a *A* de 1, 2, 3 y 4, respectivamente. De pronto *A* se desactiva, o bien se corta la línea entre *A* y *B*, que de hecho es la misma cosa desde el punto de vista de *B*. En el primer intercambio de paquetes *B* no escucha nada de *A*. *C* sabe que tiene una trayectoria a *A* de longitud 2 y avisa a *B*, sin saber este último que la trayectoria de *C* pasa a través de *B* mismo. Como resultado, *B* ahora piensa que puede llegar a *A* por medio de *C*. *D* y *E* no actualizan sus entradas para *A* en el primer intercambio.

En el segundo intercambio, se nota que cada uno de sus vecinos indica tener una trayectoria a *A* de longitud 3. *C* escoge una de ellas al azar y hace que su nueva distancia sea 4. Los intercambios subsecuentes se muestran en el resto de la figura 3 (b).

A partir de la figura 3 queda clara la razón porqué las malas noticias viajan con tanta lentitud.

**Recorte por horizonte dividido (solución)**

Hay muchas soluciones a este problema, pero ninguna lo soluciona completamente. El algoritmo de horizonte dividido funciona de la misma manera que el enrutamiento por vector a distancia, excepto que la distancia a *X* no se informa en la línea por la que se envían paquetes para *X*. Usando el horizonte dividido, las malas noticias se propagan

a razón de una escala por intercambio. Esta velocidad es mucho mejor que sin este algoritmo.
La verdadera mala noticia es que horizonte dividido, aunque se usa ampliamente, a veces falla.

**Enrutamiento por estado de enlace**

El concepto de este algoritmo es sencillo y puede describirse en cinco partes. Cada enrutador debe:

*1. Descubrir a sus vecinos y conocer sus direcciones de red.*

Al ponerse en operación un enrutador, su primera tarea es averiguar quiénes son sus vecinos; esto se logra enviando un paquete especial de HOLA (*HELLO*) por cada línea punto a punto. Se espera que el enrutador del otro extremo envíe de regreso su dirección única.

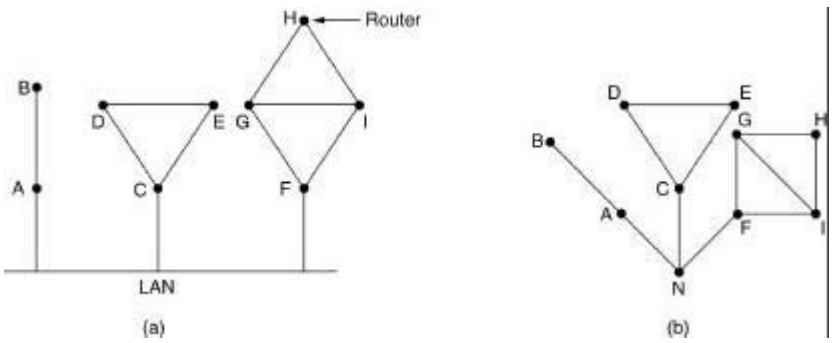


Figura 4. (a) Nueve enrutadores y una LAN. (b) Modelo de grafo de (a).

Al conectarse dos o más enrutadores mediante una LAN, la situación es ligeramente más complicada. En la figura 4 se ilustra una LAN a la que están conectados directamente tres enrutadores, *A*, *B*, y *F*. Cada uno de estos enrutadores está conectado a uno o más enrutadores adicionales.
Una manera de modelar la LAN es considerarla como otro nodo, como se muestra en la figura 4 parte (b). Aquí se ha introducido un nodo artificial nuevo, *N*, al que están conectados *A*, *C* y *F*. El hecho de que sea posible ir de *A* a *C* a través de la LAN se representa aquí mediante la trayectoria *ANC.*

*2. Medición del costo de la línea.*

El algoritmo de enrutamiento por estado de enlace requiere que cada enrutador sepa, o cuanto menos tenga una idea razonable del estado de cada uno de sus vecinos. La manera más directa de determinar este retardo es enviar un paquete especial ECO (*ECHO*) a través de la línea, el cual debe enviar de regreso inmediatamente el otro lado. Si mide el tiempo de ida y vuelta y lo divide entre dos, el enrutador transmisor puede tener una idea razonable del retardo. Para obtener mejores resultados aún la prueba puede llevarse a cabo varias veces y usarse el promedio.

*3. Construcción de los paquetes de estado de enlace.*

Una vez que se ha recabado la información necesaria para el intercambio, el siguiente paso es que cada enrutador construya un paquete con todos los datos. Este paquete comienza con la identidad del transmisor, seguida de un número de secuencia, una edad y una lista de vecinos. Para cada vecino, se coloca el retardo a ese vecino. En la figura 5 (a) se muestra un ejemplo de una subred, con los retardos en las líneas. Los paquetes de estado de enlace de los seis enrutadores se muestran en la figura 5 (b).

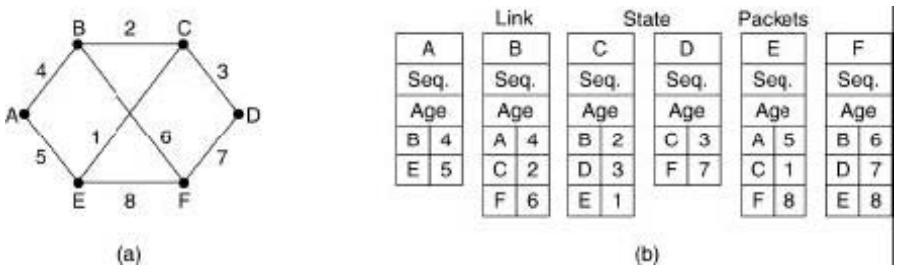


Figura 5. (a) Subred. (b) Paquetes de estado de enlace para esta subred.

Es fácil construir los paquetes de estado de enlace. La parte difícil es determinar cuándo construirlos. Una posibilidad es construirlos periódicamente, es decir, a intervalos regulares.

Otra posibilidad es al ocurrir un evento significativo, como la caída o reactivación de una línea o de un vecino, o el cambio apreciable de sus propiedades.

*4. Distribución de los paquetes de estado de enlace.*

La parte mas complicada del algoritmo es la distribución confiable de los paquetes de estado de enlace. A medida que se distribuyen e instalan los paquetes los enrutadores que reciban los primeros cambiarán sus rutas. En consecuencia, los distintos enrutadores podrían estar usando versiones diferentes de la topología, lo que puede conducir a inconsistencias, ciclos, máquinas inalcanzables, y otros problemas.

El algoritmo que se utiliza para la distribución de los paquetes de estado de enlace seria inundación.

*5. Cálculo de nuevas rutas.*

Una vez que un enrutador ha acumulado un grupo completo de paquetes, puede construir el grafo de la subred completa porque todos los enlaces están representados. De hecho, cada enlace se representa dos veces, para cada dirección. Los dos valores pueden promediarse o usarse por separado. Ahora puede ejecutarse localmente el algoritmo de la trayectoria más corta posible a todos los destinos. Los resultados de este algoritmo pueden instalarse en las tablas de enrutamiento, y reiniciarse la operación normal.

Para una subred con *n* enrutadores, cada uno de los cuales tiene *k* vecinos, la memoria requerida para almacenar los datos de entrada es proporcional a *nk*. En las subredes grandes este puede ser un problema. También puede serlo el tiempo de cómputo. Sin embargo en muchas situaciones prácticas, el enrutamiento por estado de

enlace funciona bien. Se usa ampliamente en redes actuales, algunos protocolos que lo usan son: el protocolo OSPF, que se emplea cada vez con mayor frecuencia en Internet, el IS-IS (sistema intermedio - sistema intermedio), diseñado por DECnet y el NetWare de Novell usa una variante menor del IS-IS (NLSP) para el enrutamiento de paquetes IPX.

**Enrutamiento jerárquico**

A medida que crece el tamaño de las redes, crecen proporcionalmente las tablas de enrutamiento del enrutador. Las tablas que siempre crecen no solo consumen memoria del enrutador, sino que también necesitan más tiempo CPU para examinarlas y más ancho de banda para enviar informes de estado entre enrutadores. En cierto momento, la red puede crecer hasta el punto en que ya no es factible que cada enrutador tenga una entrada para cada uno de los demás enrutadores, por lo que el enrutamiento tendrá que hacerse jerárquicamente, como ocurre en la red telefónica.
Al usarse el enrutamiento jerárquico, los enrutadores se dividen en lo que llamamos *regiones,* en donde cada enrutador conoce todos los detalles de la manera de enrutar paquetes a destinos dentro de su propia región, pero no sabe nada de la estructura interna de las otras regiones. Al interconectar diferentes redes, es natural considerar cada una como región independiente, a fin de liberar a los enrutadores de una red de la necesidad de conocer la estructura topológica de las demás.

**Enrutamiento por difusión**

En algunas aplicaciones, los *host* necesitan enviar mensajes a varios otros *host* o a todos los demás. Por ejemplo, el servicio de distribución de informes ambientales, la actualización de los precios de la bolsa o los programas de radio en vivo podrían funcionar mejor difundiéndolos a todas las máquinas y dejando que aquellas interesadas lean los datos. El envío simultáneo de un paquete a todos los destinos se llama *difusión.*

Hay varios métodos para llevarlo a cabo.

Un método de difusión que no requiere características especiales de la red es que el origen simplemente envíe copias del paquete a todos los destinos. El método no sólo desperdicia ancho de banda, sino que también requiere que el origen tenga una lista completa de todos los destinos. En la práctica, este es el método menos deseable.
La inundación es otro candidato pero el problema de éste como técnica de difusión es el mismo que tiene como algoritmo de enrutamiento punto a punto: genera demasiados paquetes y consume demasiado ancho de banda.
Un tercer algoritmo es el *enrutamiento multidestino.* Con este método cada paquete contiene una lista de destinos que indican los destinos deseados. El enrutador genera una copia nueva del paquete para que cada línea de salida a usar, e incluye en cada paquete sólo aquellos destinos que usan la línea. En efecto, el grupo de destinos se divide entre las líneas de salida. Este enrutamiento es idéntico al de los paquetes con direccionamiento individual, excepto que, cuando varios paquetes deben seguir la misma ruta, uno de ellos paga la tarifa completa y los demás viajan gratis.
El último algoritmo de difusión es un intento de aproximar el comportamiento del algoritmo el árbol de extensión, aún cuando los enrutadores no saben nada en lo

absoluto sobre árboles de extensión de los demás enrutadores. La idea es excepcionalmente sencilla una vez planteada. Cuando llega un paquete difundido a un enrutador, éste lo revisa para ver si llegó por la línea normalmente usada para enviar paquetes al origen de la difusión. De ser así, hay excelentes posibilidades de que el paquete difundido haya seguido la mejor ruta desde el enrutador y, por lo tanto, sea la primera copia en llegar al mismo. Siendo este el caso reenvía copias del paquete por todas las líneas, excepto por aquella por la que llegó. Sin embargo, si el paquete difundido llegó por otra línea diferente de la preferida, se descarta el paquete como probable duplicado.

## 7 Sistemas Autónomos

Un *sistema autónomo* o AS será la subred que es administrada por una autoridad común, que tiene un protocolo de ruteo homogéneo mediante el cual intercambia información en toda la subred y que posee una política común para el intercambio de tráfico con otras redes o sistemas autónomos. En Internet se dan, al menos, dos niveles jerárquicos de ruteo, el que realiza dentro de un sistema autónomo y el que se efectúa entre sistemas autónomos.
El primero es denominado *ruteo interno o intraáres*, al segundo se lo denomina *ruteo externo o interáreas.* Dado que los requerimientos en unos y en otros son muy diferentes, se utilizan protocolos de ruteo muy distintos.

## 8 La Capa de Red en Internet

En la capa de red, la Internet puede verse como un conjunto de subredes, o *sistemas autónomos (AS)* interconectados. No hay una estructura real, pero existen varios *backbone* principales. Estos se construyen a partir de líneas de alto ancho de banda y enrutadores rápidos. Conectadas a los *backbone* hay redes regionales (de nivel medio), y conectadas a estas redes están las LAN de muchas universidades, compañías y proveedores de servicios de Internet. La siguiente figura muestra un dibujo de esta organización:

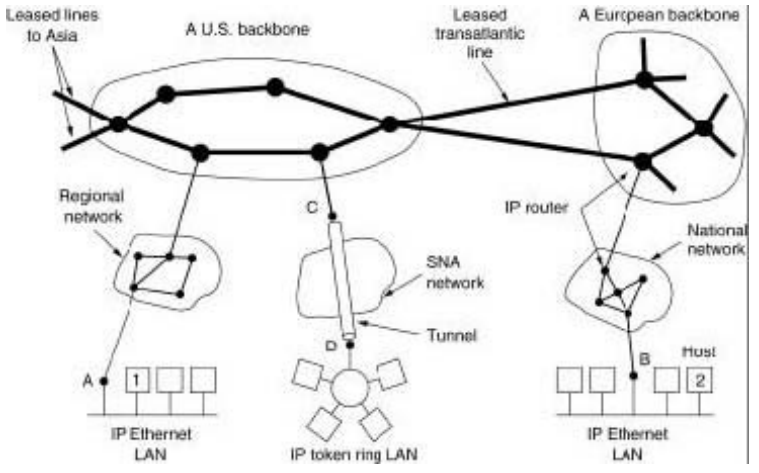


Figura 6. La Internet como conjunto interconectado de muchas redes.

El "pegamento" que mantiene unida la Internet es el protocolo de la capa de red, *IP* (*Internet Protocol*, protocolo de Internet). Este protocolo se diseñó desde sus principios con la interconexión de redes en mente. Su trabajo es proporcionar un medio de mejor esfuerzo para el transporte del datagrama del origen hacia el destino, sin importar si estas máquinas están en la misma red o si hay otras redes entre ellas.

**Direcciones IP**

Para comenzar el estudio de la capa de red de Internet es necesario citar el formato de los datagramas de IP mismos que son los paquetes de datos que viajan por la red. Un datagrama IP consiste en una parte de cabecera y una parte de texto. La cabecera tiene una parte fija de 20 bytes y una parte opcional de longitud variable.
Cada *host* y enrutador de Internet tiene una dirección de IP, que codifica su número de red y su número de *host*. La combinación es única: no hay dos máquinas que tengan la misma dirección de IP. Todas las direcciones de IP son de 32 bits de longitud y se ocupan en los campos de dirección de origen y de dirección de destino de los paquetes. Aquellas máquinas conectadas a varias redes tienen direcciones de IP diferentes en cada red.
Los formatos de clase A, B, C y D permiten hasta 126 redes con 16 millones de *host* cada una, 16.382 redes con hasta 64k *hosts*, 2 millones de redes (tipo LAN) de hasta 254 *host*s cada una, y multitransmisión. Los números de redes los asigna el *NIC* (*Network Information Center*, Centro de Información de Redes) para evitar conflictos. La dirección de IP menor es 0.0.0.0 y la mayor es 255.255.255.255.
Los valores 0 y -1 tienen significado especial, como se muestra en la figura 7. El valor 0 significa esta red o este *host*. El valor -1 se usa como dirección de difusión para indicar todos los *host* de la red indicada.

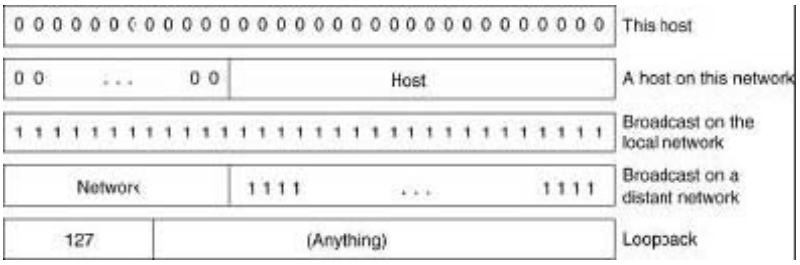


Figura 7. Direcciones especiales de IP.

La dirección de IP 0.0.0.0 es usada por los *hosts* cuando están siendo arrancados, pero no se usa después, las direcciones de IP con 0 como número de red se refieren a la red actual. Estas direcciones permiten que las máquinas se refieran a su propia red sin saber su número (pero tienen que saber su clase para saber cuántos 0 hay que incluir). La dirección que consiste solamente en unos permite la difusión en la red local, por lo común una LAN. Las direcciones con un número de red propio y solamente unos en el campo de *host* permiten que las máquinas envíen paquetes de difusión a LAN distantes desde cualquier parte de Internet. Por último, todas las direcciones de la forma 127.xx.yy.zz se reservan para pruebas de realimentación.

**Subredes**

Todos los host de una red deben tener el mismo número de red. Esta propiedad del direccionamiento IP puede causar problemas a medida que crecen las redes.
Al crecer la cantidad de redes locales distintas, su administración puede volverse un conflicto. Cada vez que se instala una nueva red, el administrador del sistema tiene que comunicarse con el NIC para obtener un número de red nuevo. Después, este número debe anunciarse mundialmente. Además, mover una máquina de una LAN a otra requiere un cambio de su dirección de IP, lo que a su vez puede significar la modificación de sus archivos de configuración y también el anuncio de la nueva dirección de IP a todo el mundo.
La solución a este problema es permitir la división de una red en varias partes para su uso interno, pero aún actuar como una sola red ante el mundo exterior. En la literatura de Internet, a estas partes se les llama *subredes*.
Para ver el funcionamiento de las subredes, es necesario explicar la manera en que se procesan los paquetes IP en un enrutador. Cada enrutador tiene una tabla en la que se lista cierto número de direcciones IP (red, 0) y cierto número de direcciones IP (esta red, *host*). El primer tipo indica cómo llegar a redes distantes. El segundo tipo indica cómo llegar a redes locales. Asociada a cada tabla está la interfase de red a usar para llegar al destino y cierta información adicional.
Al llegar un paquete de IP se busca su dirección de destino en la tabla de enrutamiento. Si el paquete es para una red distante, se reenvía al siguiente enrutador de la interfase dada en la tabla; si es para un *host* local (por ejemplo, en la LAN del enrutador), se envía directamente al destino. Si la red no está en la tabla, el paquete se reenvía a un enrutador predeterminado con las tablas más extensas. Este algoritmo

significa que cada enrutador sólo tiene que llevar el registro de otras redes y *hosts* locales, no de pares de red – *host*, reduciendo en gran medida el tamaño de la tabla de enrutamiento.
Al introducirse subredes, se cambian las tablas de enrutamiento, agregando entradas con forma de (esta subred, subred, 0) y (esta red, esta subred, *host*). Por lo tanto, un enrutador de la subred *k* sabe cómo llegar a todas las demás subredes y también cómo llegar a todos los *hosts* de la subred *k*. De hecho, todo lo que se necesita es hacer que cada enrutador haga un AND booleano con la *máscara de subred* para deshacerse del nuevo *host* y buscar la dirección resultante en sus tablas (después de haber determinado la clase de red de la cual se trataba).

**Protocolo de enrutamiento de pasarela interior: OSPF**

Como se ha dicho antes, la Internet se compone de una gran cantidad de sistemas autónomos (AS). Cada AS es operado por una organización diferente y puede usar internamente su propio algoritmo de enrutamiento. Por ejemplo, las redes internas de las compañías X, Y, y Z generalmente se verían como tres AS si las tres estuvieran en Internet. Las tres pueden usar algoritmos de enrutamiento diferentes internamente. No obstante, la existencia de estándares aún para enrutadores internos, simplifica la implementación en las líneas divisorias entre los AS y permite la reutilización de código. El algoritmo de enrutamiento interno de un AS se llama *protocolo de pasarela interior;* al algoritmo de enrutamiento entre varias AS se le llama *protocolo de pasarela exterior.*
El protocolo de pasarela interior original de Internet fue un protocolo de vector de distancia (RIP) basado en el algoritmo Bellman-Ford. Este protocolo funcionó bien en pequeños sistemas, pero menos bien a medida que los AS se volvieron más grandes. En 1988, la *Internet Engineering Task Force* (grupo de trabajo de ingeniería de Internet) comenzó a trabajar en su sucesor. Ese sucesor llamado OSPF (*Open Shortest Path First,* abrir primero la trayectoria más corta), se convirtió en estándar en 1990.
Dada la amplia experiencia con otros protocolos de enrutamiento, el grupo que diseñó el nuevo protocolo tenía una larga lista de requisitos que cumplir. Primero, el algoritmo tenía que publicarse como literatura abierta, de ahí "O" (de *Open*) en OSPF. Segundo, el nuevo protocolo tenía que reconocer una variedad de métricas de distancia, incluidas distancia físicas, retardo y otras. Tercero, tenía que ser un algoritmo dinámico, uno que se adaptara a los cambios de topología rápida y automáticamente.

Cuarto, algo nuevo para el OSPF, tenía que reconocer el enrutamiento basado en el tipo de servicio. El nuevo protocolo tenía que ser capaz de enrutar el tráfico de tiempo real de una manera y otros tipos de tráfico de otra manera.
Quinto, y relacionado con el anterior, el nuevo protocolo tenía que efectuar equilibrio de cargas, dividiendo la carga entre varias líneas. La mayoría de los protocolos previos enviaban todos los paquetes a través de la mejor ruta. La segunda mejor ruta no se usaba en lo absoluto. En muchos casos, la división de la carga a través de varias líneas produce un mejor desempeño. Sexto se requería el reconocimiento de sistemas jerárquicos.

Por último, se requería un mecanismo para manejar los enrutadores que se conectaban a Internet a través de un túnel. Los protocolos previos no manejaban bien esta cuestión.

El OSPF reconoce tres tipos de conexiones y redes:
1. Líneas punto a punto entre dos enrutadores (exactamente).
2. Redes multiacceso con difusión (por ejemplo, la mayoría de las LAN).
3. Redes multiacceso sin difusión (por ejemplo, la mayoría de las WAN de conmutación de paquetes). El OSPF funciona haciendo una abstracción del conjunto de redes, enrutadores y líneas en un grafo dirigido en el que a cada arco se le asigna un costo (distancia, retardo, etc.). Entonces se calcula la trayectoria más corta con base en los pesos de los arcos.

En la figura 8 (a) se muestra la representación gráfica de la red de la figura 8 (b). Lo fundamental que hace el OSPF es representar la red como un grafo de este tipo y luego calcular la trayectoria más corta de un enrutador a todos los demás.

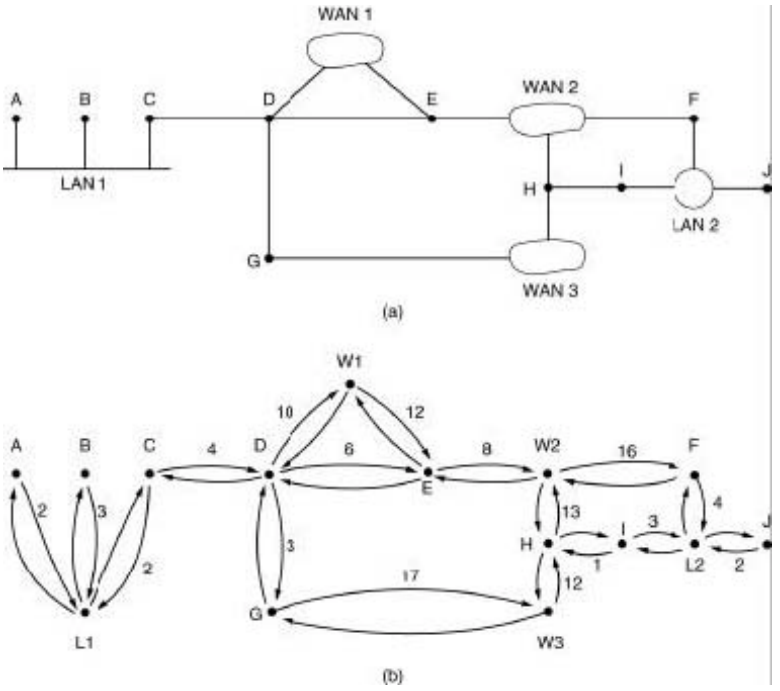


Figura 8. (a) Sistema Autónomo. (b) Representación con grafos de (a).

Muchas de las AS de Internet son grandes y nada fáciles de manejar. El OSPF permite su división en *áreas* numeradas, donde un área es una red o un grupo de redes contiguas. Un área es una generalización de una subred. Fuera de un área, su topología y detalles no son visibles.

Cada AS tiene un área de *backbone*, llamada área 0. Todas las áreas se conectan al *backbone*, posiblemente mediante túneles, por lo que hay posibilidad de ir de cualquier área del AS a cualquier otra a través del *backbone*. Un túnel se representa en el grafo como un arco y tiene un costo. Cada enrutador conectado a dos o más áreas es parte del *backbone*, siendo la topología del mismo no visible desde fuera del *backbone*.

Dentro de un área, cada enrutador tiene la misma base de datos de estado de enlace y ejecuta el mismo algoritmo de trayectoria más corta; su tarea principal es calcular la trayectoria más corta de sí mismo a todos los demás enrutadores del área, incluido el enrutador que está conectado al *backbone*.
La manera en que el OSPF maneja el enrutamiento de tipo de servicio es teniendo varios grafos, uno etiquetado con los costos cuando la métrica es el retardo, otro etiquetado con los costos cuando la métrica es el rendimiento, y uno más etiquetado con los costos cuando la métrica es la confiabilidad. Aunque esto triplica el cálculo, permite rutas separadas para optimizar el retardo, el rendimiento y la confiabilidad.
Durante una operación normal pueden necesitarse tres tipos de rutas: intraárea, interárea e interAS. Las rutas intraáreas son las más fáciles, dado que el enrutador de origen ya conoce la trayectoria más corta al enrutador de destino. El enrutamiento interárea siempre procede en tres pasos: va del origen al *backbone*, pasa a través del *backbone*, al área de destino y va al destino. Este algoritmo obliga a una configuración en estrella en el OSPF, siendo el *backbone* el centro y las demás áreas los rayos.

En la figura 9 se muestra parte de la Internet con AS y áreas.

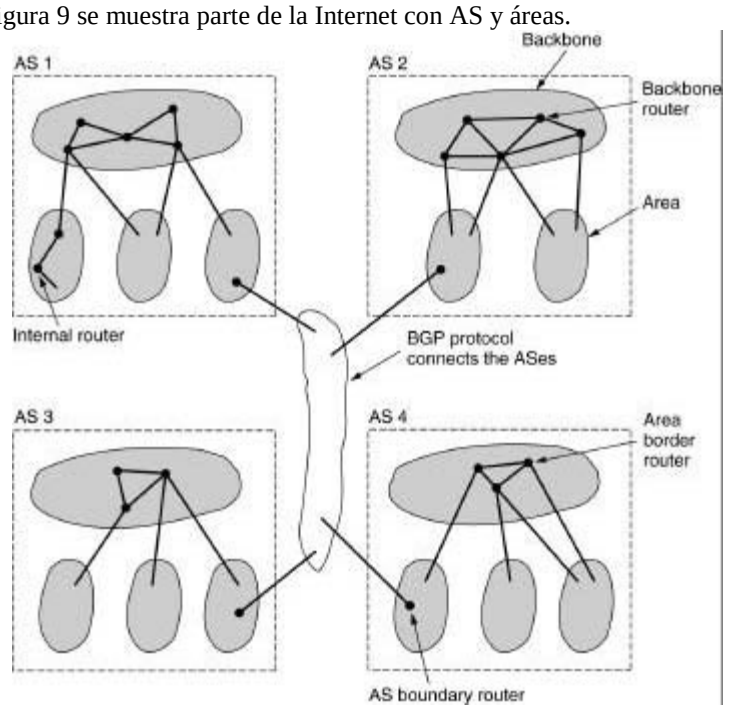


Figura 9. Relación entre los AS, los *backbone* y las áreas en el OSPF.

El OSPF distingue cuatro clases de enrutadores:

1 Enrutadores internos que están contenidos en una sola área.
2 Enrutadores de borde de área que conectan dos o más áreas.
3 Enrutadores de *backbone* que están en el *backbone.*
4 Enrutadores de frontera de AS que hablan con los enrutadores de otras áreas AS.

El OSPF funciona intercambiando información entre enrutadores *adyacentes*, que no es lo mismo que entre enrutadores vecinos. En particular es ineficiente hacer que todos los enrutadores de una LAN hablen con todos los enrutadores de otra LAN. Para evitar esta situación, se elige un enrutador como *enrutador designado*, el cual se dice que es adyacente a todos los demás enrutadores, e intercambia información con ellos. Los enrutadores que no son vecinos no se intercambian información entre ellos, se mantienen actualizados designando un enrutador de respaldo para facilitar la transición en el caso que el enrutador designado primario se caiga.

**Protocolo de enrutamiento de pasarela exterior: BGP**

Dentro de un solo AS, el protocolo de enrutamiento recomendado en Internet es el OSPF (aunque no es el único en uso). Entre los AS se usa un protocolo diferente, el *BGP* (*Border Gateway Protocol*, protocolo de pasarela exterior). Se requiere un protocolo diferente entre las AS, ya que las metas de un protocolo de pasarela interior y un protocolo de pasarela exterior no son iguales. Un protocolo de pasarela interior sólo tiene que mover paquetes con la mayor eficiencia posible, desde el origen hasta el destino; no necesita preocuparse por la política.

Los enrutadores de protocolo de pasarela exterior tienen que preocuparse por los asuntos políticos. En particular el BGP, se ha diseñado para permitir muchos tipos de políticas de enrutamiento aplicables al tráfico interAS.

Las políticas típicas comprenden consideraciones políticas, de seguridad o económicas. Éstas se configuran manualmente en cada enrutador BGP. No son parte del protocolo mismo.

Desde el punto de vista de un enrutador BGP, el mundo consiste en otros enrutadores BGP y en las líneas que los conectan. Se consideran conectados dos enrutadores BGP si comparten una red en común. Dada la importancia del BGP en el tránsito, las redes se agrupan en una de tres categorías. La primera categoría es la de las *redes de punta*, que sólo tienen una conexión al grafo BGP; no se pueden usar para tráfico en tránsito porque no hay nadie del otro lado. La segunda categoría son las *redes multiconectadas.* Por último están las *redes de tránsito*, como los *backbone*, que están dispuestas a manejar los paquetes de terceros, posiblemente con algunas restricciones.

Los pares de enrutadores BGP se comunican entre ellos estableciendo conexiones TCP. Este tipo de operación proporciona comunicación confiable y esconde todos los detalles de la red por la que pasa.

El BGP fundamentalmente es un protocolo de vector de distancia, pero muy diferente de casi todos los demás, como el RIP. En lugar de mantener sólo el costo a cada destino, cada enrutador BGP lleva el registro de la trayectoria seguida. Del mismo modo, en lugar de dar periódicamente sus costos estimados a todos los

destinos posibles, cada enrutador BGP le dice a sus vecinos la trayectoria exacta que se está usando.

## 9 Conclusiones

Para lograr sus objetivos la capa de red debe conocer la topología de la subred de comunicación y escoger aquella ruta más adecuada para cumplir con su cometido.
El protocolo que reside en cada uno de los routers dentro de una red, mediante distintas evaluaciones, calcula la mejor trayectoria a utilizar.
Después de realizarse un análisis cualitativo de todos los algoritmos de enrutamiento se puede ver que no existe aquél que ante cualquier circunstancia sea el que mejor resuelva siempre el problema del encaminamiento. Sólo dependerá de qué recurso o criterio se elija como prioritario para el envío de los paquetes de datos.
Es el caso del enrutamiento por vector de distancias, que se preocupa prioritariamente por el nº de saltos (routers), mientras que el enrutamiento por estado de enlace se preocupa principalmente del estado de las interfaces que el router soporta; y es de ahí su nombre "estado de enlace".
Otra característica que ha sido descripta y que es de suma importancia de los protocolos de enrutamiento, es si deben rutear dentro o fuera de la subred donde se encuentran. Los protocolos de enrutamiento internos se utilizan para actualizar routers bajo el control de un sistema autónomo; mientras que los exteriores se emplean para permitir que dos redes con distintos sistemas autónomos se comuniquen; el ejemplo más actual es el de Internet: OSPF para ruteo interno, BGP para externo.
El estudio de los protocolos de la capa de red está en permanente evolución, siendo un tema de gran interés y expectativa de futuros desarrollos teniendo presente la continua evolución de las redes de comunicaciones de datos, cada vez sometidas a mayores requerimientos en cuanto a sus prestaciones, las que están directamente relacionadas con el desempeño de los protocolos de red.

## Bibliografía

1 Tanenbaum, A. S.: Redes de Computadoras – 3ª Edición. Prentice Hall Hispanoamericana S.A. (1997).
2 Stallings, W.: Comunicaciones y Redes de Computadores – 5ª Edición. Prentice Hall. (1997).
3 Coulouris, G.; Dollimore, J.; Kindberg, T.: Sistemas Distribuidos – 3ª Edición. Addison Wesley. (2001).