Algoritmos de Ruteo

31 Mayo 2005

Enrutamiento por Vector de Distancia

- Llamado tambien:
  - Bellman-Ford
  - Ford-Fulkerson
  - RIP

Usado Inicialmente en ARPANET

Componentes Principales

- Cada enrutador posee una tabla con
- una entrada por cada destino conteniendo la distancia a el mismo
- cara enrutador comparte la tabla con sus vecinos.

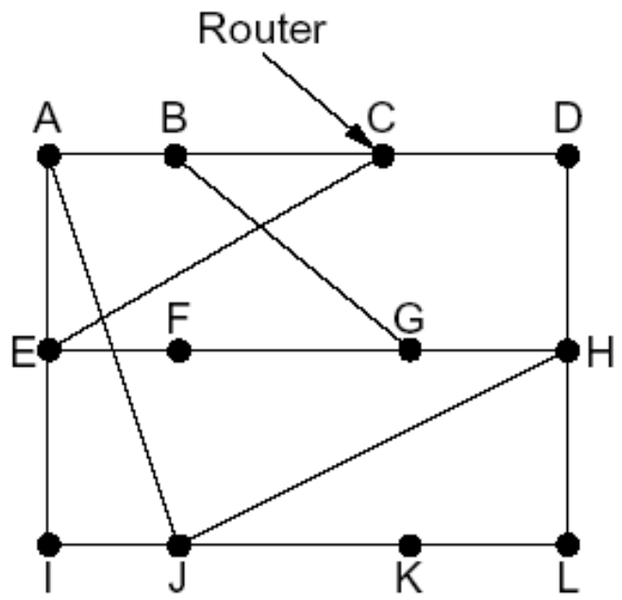
Metricas

- cantidad de saltos
- retardo
- numero total de paquetes encolados

Algoritmo

- Cada enrutador comparte su tabla con sus vecinos
- cada enrutador recibe de sus vecinos sus respectivas tablas
- cada enrutador calcula una nueva tabla utilizando las tablas de sus vecinos y la distancia los vecinos

Ejemplo



(a)

To	A	I	H	K	New estimated delay from J	Line
A	0	24	20	21	8	A
B	12	36	31	28	20	A
C	25	18	19	36	28	I
D	40	27	8	24	20	H
E	14	7	30	22	17	I
F	23	20	19	40	30	I
G	18	31	6	31	18	H
H	17	20	0	19	12	H
I	21	0	14	22	10	I
J	9	11	7	10	0	–
K	24	22	22	0	6	K
L	29	33	9	9	15	K
	JA delay is 8	JI delay is 10	JH delay is 12	JK delay is 6	New routing table for J	

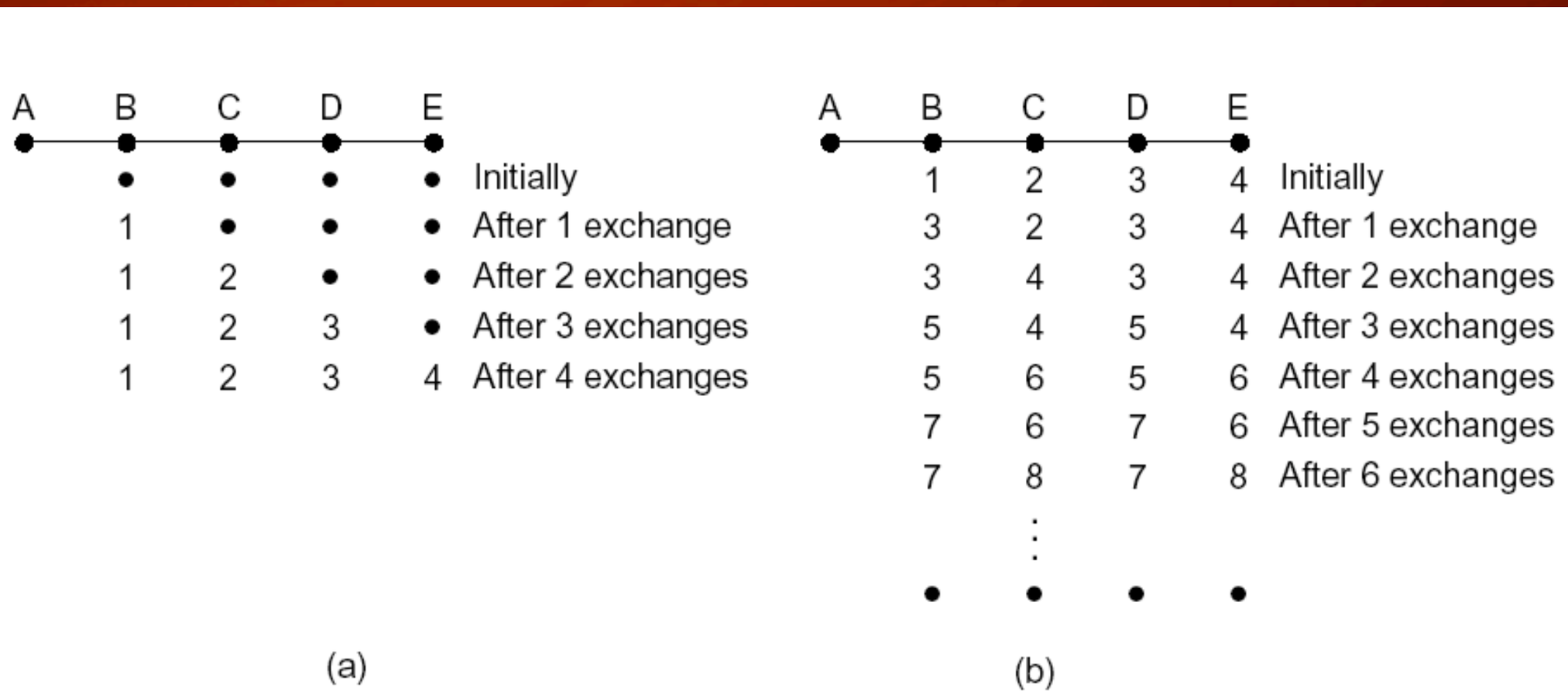
Vectors received from J's four neighbors

(b)

Calcular la Distancia de J a G

Defectos y Virtudes

- las buenas noticias viajan rapido
- las malas no tanto...



Enrutamiento por estado de enlace

- Descubrir a sus vecinos y conocer sus direcciones de red
- medir el retardo a sus vecinos
- contruir un paquete que indique esto
- enviar este paquete a todos los demas
- calcular la ruta mas corta a todos

conociendo vecinos

- Paquete HELLO por cada linea punto a punto
- respuesta con identificadores
- Identificadores de enrutadores únicos!

midiendo la linea

- enviar paquetes ECHO a los vecinos
- calcular el tiempo de ida y vuelta
- enviar varias veces para optner valores mas útiles
- tener o no en cuenta la carga: pros y contras.

Construcción de paquetes

- una edad,
- una lista de vecinos
- identificador del emisor,
- y un numero de secuencia,
- una edad,
- una lista de vecinos

Cuando Construir?

- Faciles de construir, pero cuando?
- Periodica: a intervalos regulares
- cuando ocurra un evento (caida de un enlace, nuevo vecino, etc)

distribucion de paquetes

- Inundación: como contralarla?
  - Cada enrutador guarda (número de secuencia, origen)
  - Se descartan viejos y ya vistos
  - una edad se utiliza para evitar problemas con los numeros de sequencia

calculo de rutas

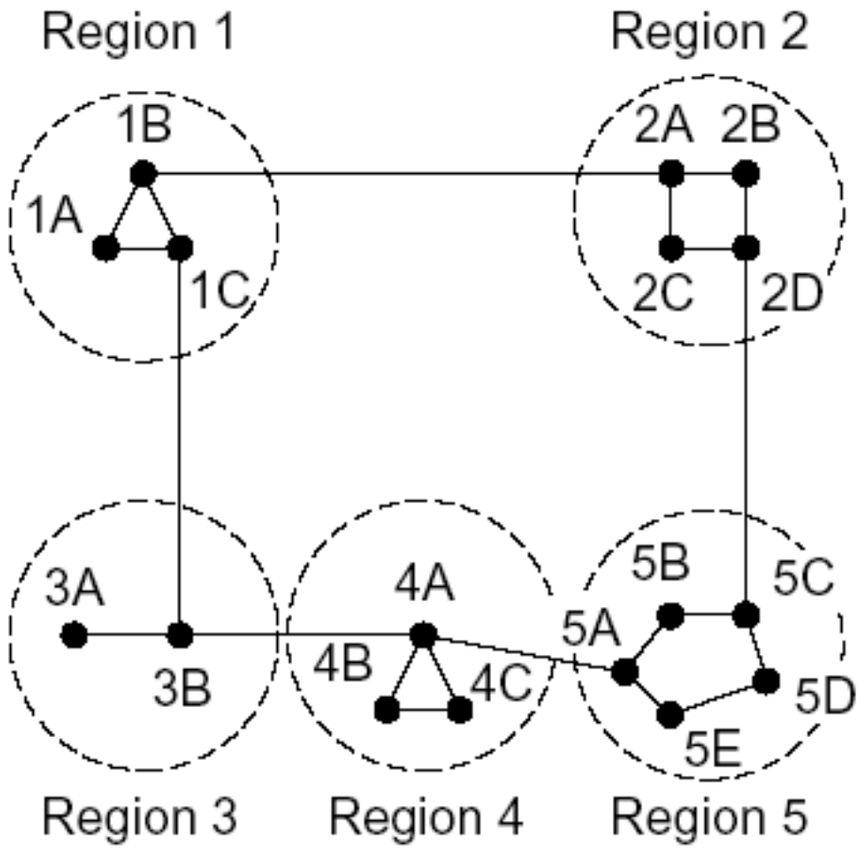
- se puede correr el algoritmo de Dijkstra localmente
- en grandes redes puede ser un problema: necesita demasiada, memoria, el grafo puede ser erroneo.

Multicast Jerarquico

Ninguno de los algoritmos es escalable: todos requieren que cada enrutador sepa de todos los demas => demasido exigente.

Solución: Buscar lineas subobtimas definiendo regiones y separar algoritmos intra-regiones e inter-regiones

Ejemplo



Full table for 1A

Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2A	1B	2
2B	1B	3
2C	1B	3
2D	1B	4
3A	1C	3
3B	1C	2
4A	1C	3
4B	1C	4
4C	1C	4
5A	1C	4
5B	1C	5
5C	1B	5
5D	1C	6
5E	1C	5

(b)

Hierarchical table for 1A

Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2	1B	2
3	1C	2
4	1C	3
5	1C	4

Enrutamiento Broadcast

Problema: Queremos enviar un mensaje a casi todas las máquinas

1) Enviar un mensaje a cada maquina individualmente

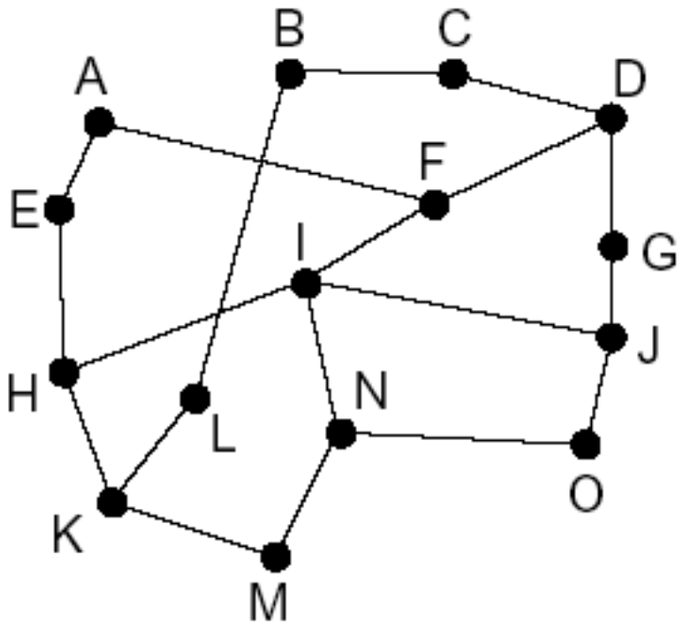
2) Usar inundación: Funciona si se puede controlarla

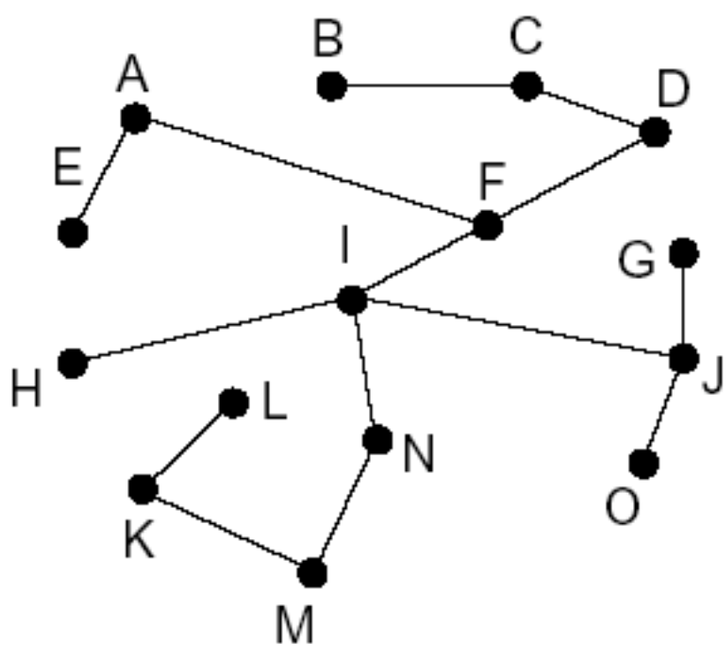
3) Usar mapas de bits: Un mapa va con cada paquete

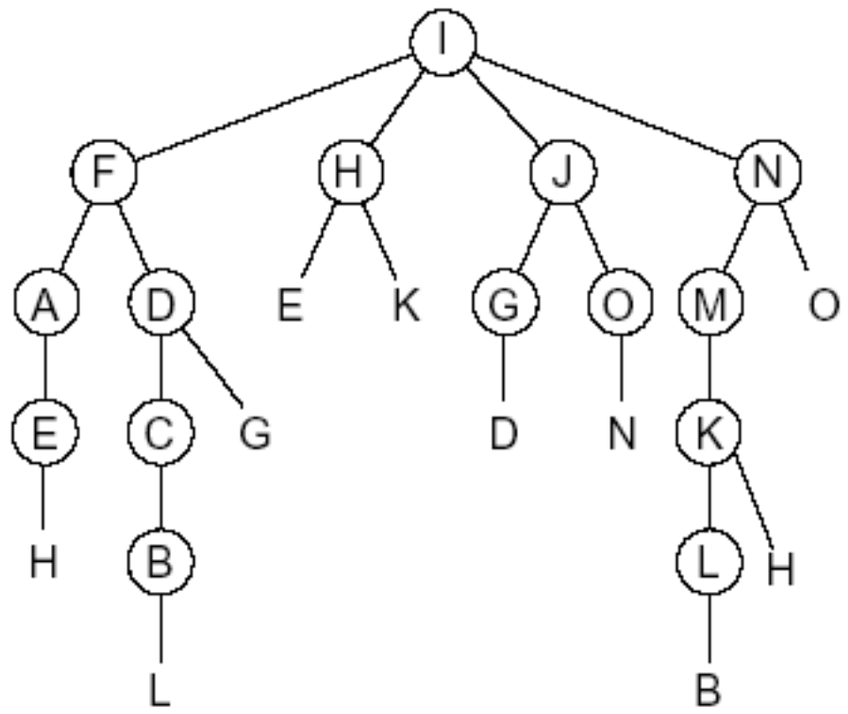
4) Construir un arbol ‘sink’, al estilo de Dijkstra

construyendo el arbol

Como construir un arbol de ‘sink’ de **i**







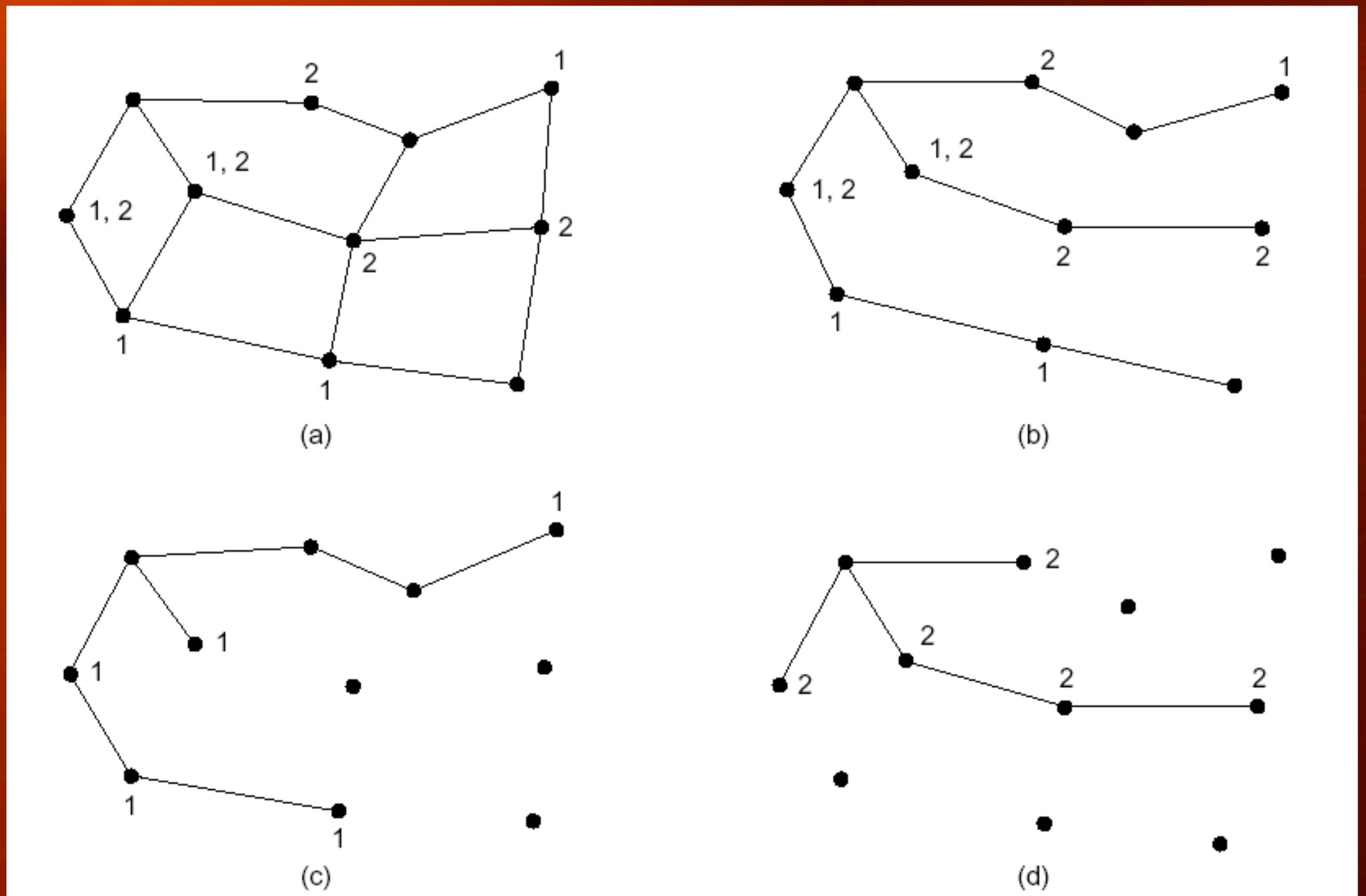
Reenviar un paquete si viene por la ruta preferida

Enrutamiento por multidifusión

Problema: Queremos mandar un mensaje solo a un subconjunto de nodos. Necesitamos saber cuando entramos y salimos de un nodo en la subred

Solución: Construir un árbol de expansión (en cada enrutador) para toda la red. Usar el id-grupo para cortar caminos del árbol.

Ejemplo

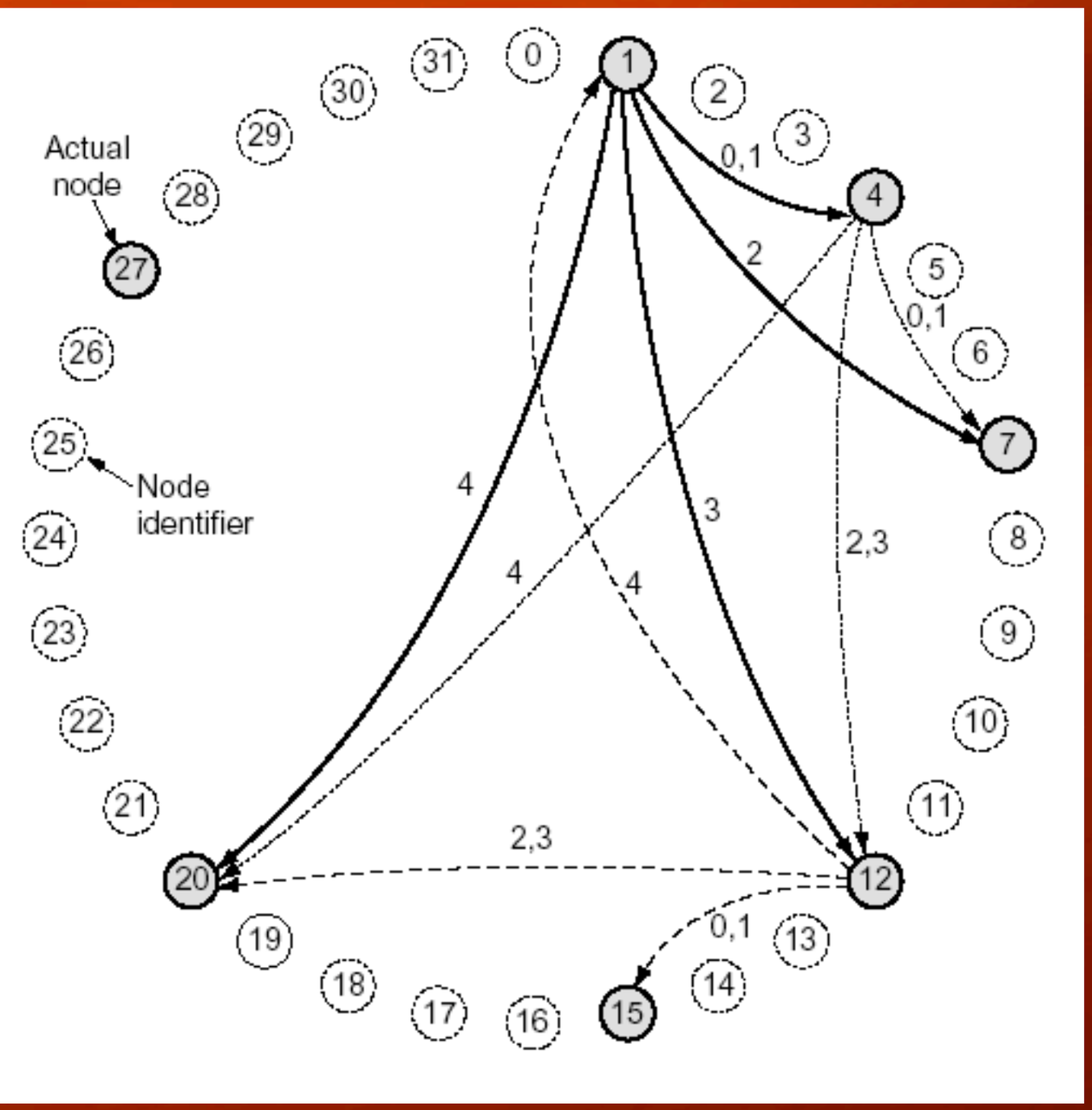


Redes peer-to-peer

Preliminares: Consiremos una colección grande de nodos, cada uno con una red IP. Cada nodo tiene un identificador de 160 bit. Logicamente todos los 2^160 nodos se organizan en un anillo.

Importante: Cada maquina almacena archivos. Cada archivo tiene una llave de 160 bits. Se anume que el nodo con el ID mas bajo mayor que *llave* almacena el archivo (identificado con *succ(llave)*).

Busquedas en redes peer-to-peer



los nodos conocen a su sucesor

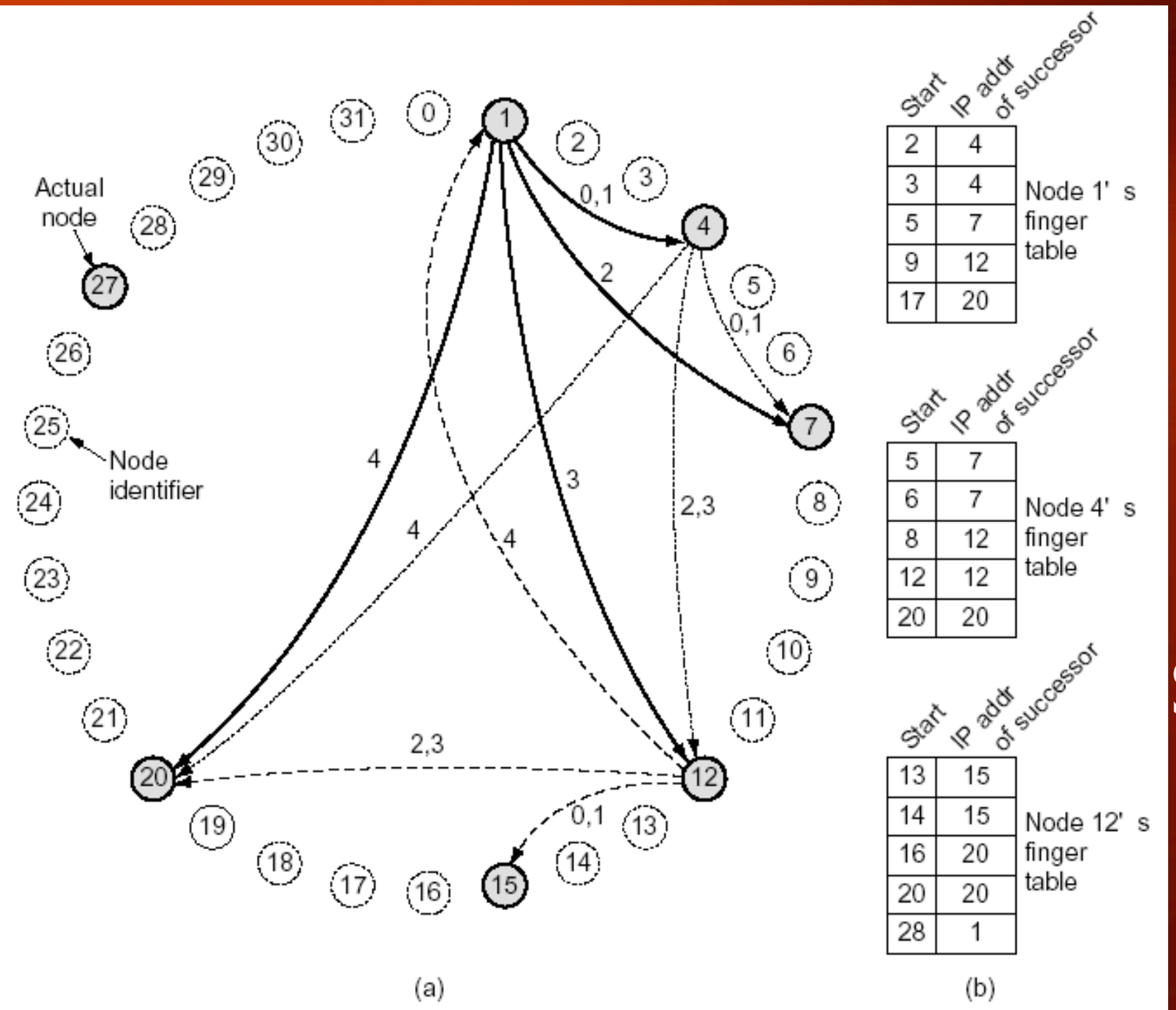
se envia IP del solicitante y clave

Los paquetes de busqueda se propagan en el anillo hasta que se encuentra el buscado

Optimización

- Tablas Finger:
  - m entradas, indexadas de 0 a m-1
  - dos campos por entrada: inicio y succ(inicio)
  - inicio=k+2^i(modulo 2^m)

Ejemplo



Start	IP addr of successor
2	4
3	4
5	7
9	12
17	20

Node 1' s finger table

Start	IP addr of successor
5	7
6	7
8	12
12	12
20	20

Node 4' s finger table

Start	IP addr of successor
13	15
14	15
16	20
20	20
28	1

Node 12' s finger table

(b)

Como buscar?

Si clave *llave* esta entre el nodo k y succ(k) la busqueda se manda a succ(k)

Sino, se busca en la tabla de finger el nodo mas cercano inferior a *clave*

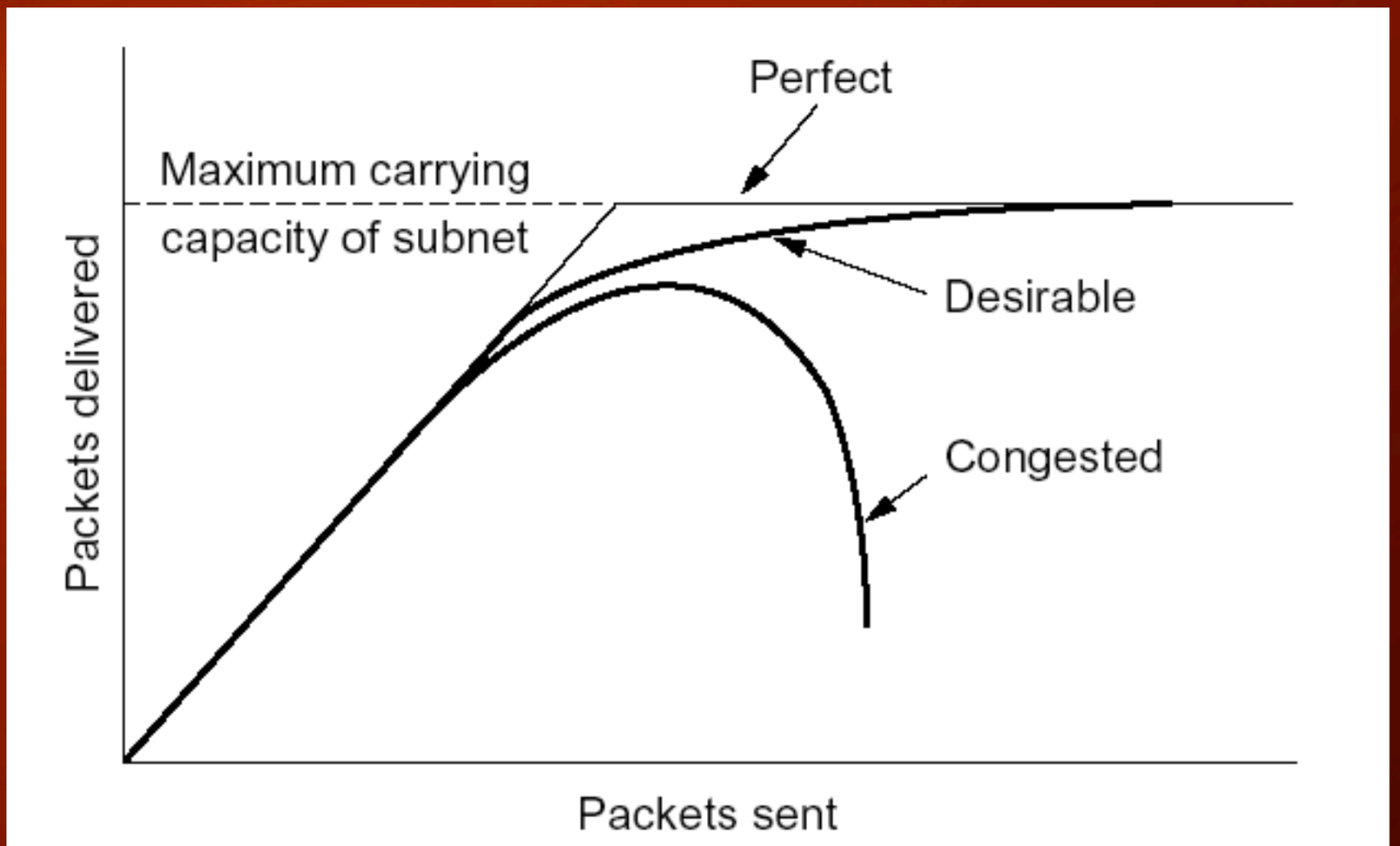
Ejemplos: Buscar 3, 14, 16 desde el nodo 1

Insertando Nodos

- nodo r desea unirse
- contacta algun nodo y pide succ(r)
- r solicita a succ(r) su predecesor
- r informa a estos dos que quiere entrar y asi entra
- el nuevo predecesor de r entrega las claves que correspondan

Congestión

Cuando hay demasiados paquetes hay una degradación del desempeño



Razones para la congestión

- procesadores lentos,
- poco ancho de banda,
- desequilibrio en el sistema,
- llegan paquetes por varias lineas que deben salir por la misma
  - paquetes se encolan
  - si no hay suficiente memoria se descartan
  - la memoria infinita no resuelve el problema!!

Control de flujo vs. de la congestión

- Flujo: Se refiere a lineas punto a punto entre un emisor y un receptor. **Perpectiva**: las maquinas
- Congestión: A toda la red, de manera que la red transmita a su maxima capacidad. **Perspectiva**: la red

Tipos Soluciones

- **Ciclo Abierto**: Buen diseño, una vez implemetado no se toca. Ejemplos: Cuando crear nuevos ciclos, que paquetes descartar, etc.
- **Ciclo Cerrado**: Retroalimentación: Monitorear, tomar decisiones y modificar el comportamiento.

CG en circuitos virtuales

- Control de Admisión: Una vez detectada la congestión no admitir nuevos circuitos.
- Control de Ruteo: Aceptar nuevos circuitos pero rutearlos cuidadosamente.
- Control de Calidad: negociar la calidad de la conexión antes, de manera de anticipar las necesidades

CG en datagramas

- Bit de advertencia:
  - Un ruteador marca al paquete con un bit de advertencia.
  - Este bit se copia la paquete de ACK que retorna al emisor
  - El emisor regula su caudal de datos

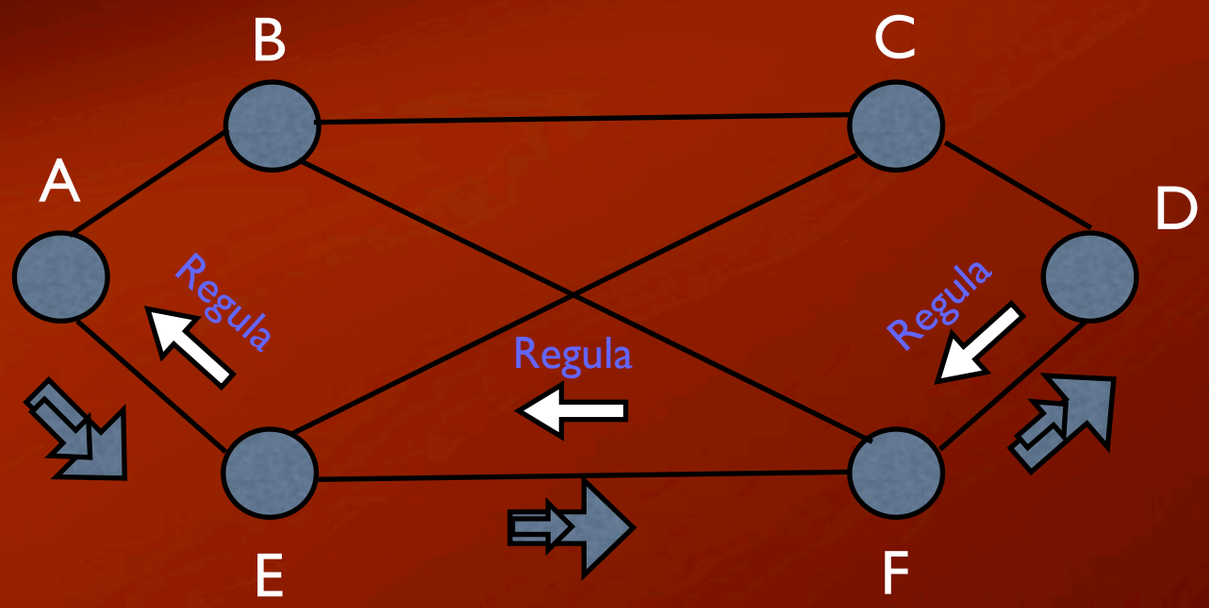
CG Datagramas

- El anterior es un método indirecto: Esto es, no genera paquetes ad hoc.
- Métodos directos con paquetes reguladores.

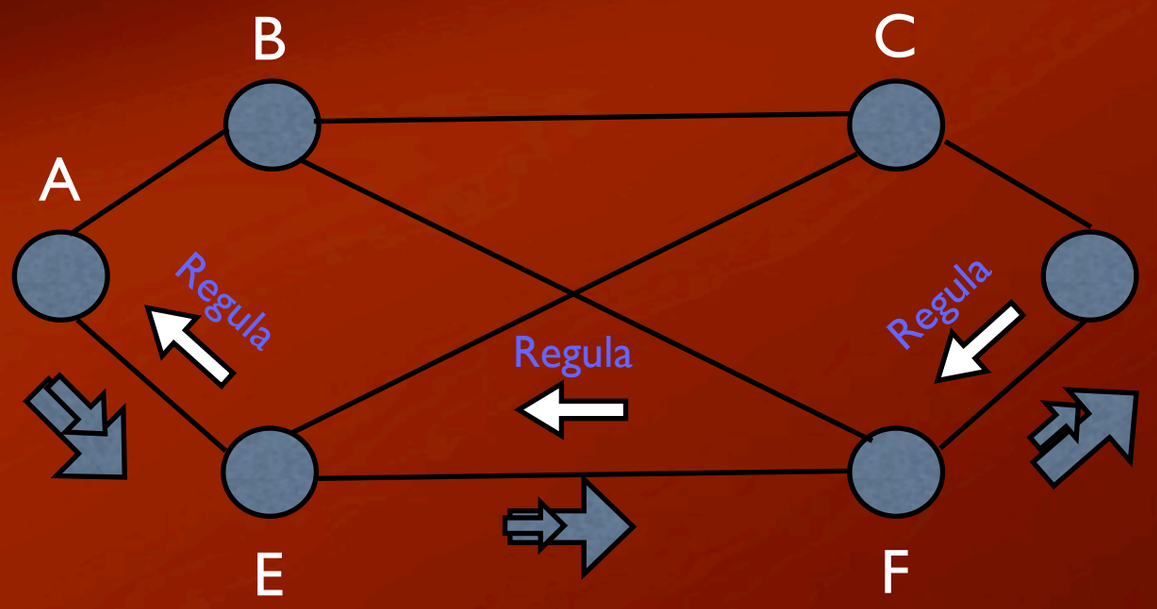
Paquete Regulador

- El enrutador regresa un paquete regulador al destino.
- Los paquetes de tráfico son marcados indicando que ya se genero un regulador
- el orgien disminuye su trafico al recibir un regulador
- el origen espera cierto teimpo para volver a transmitir

Ejemplo



Salto a Salto



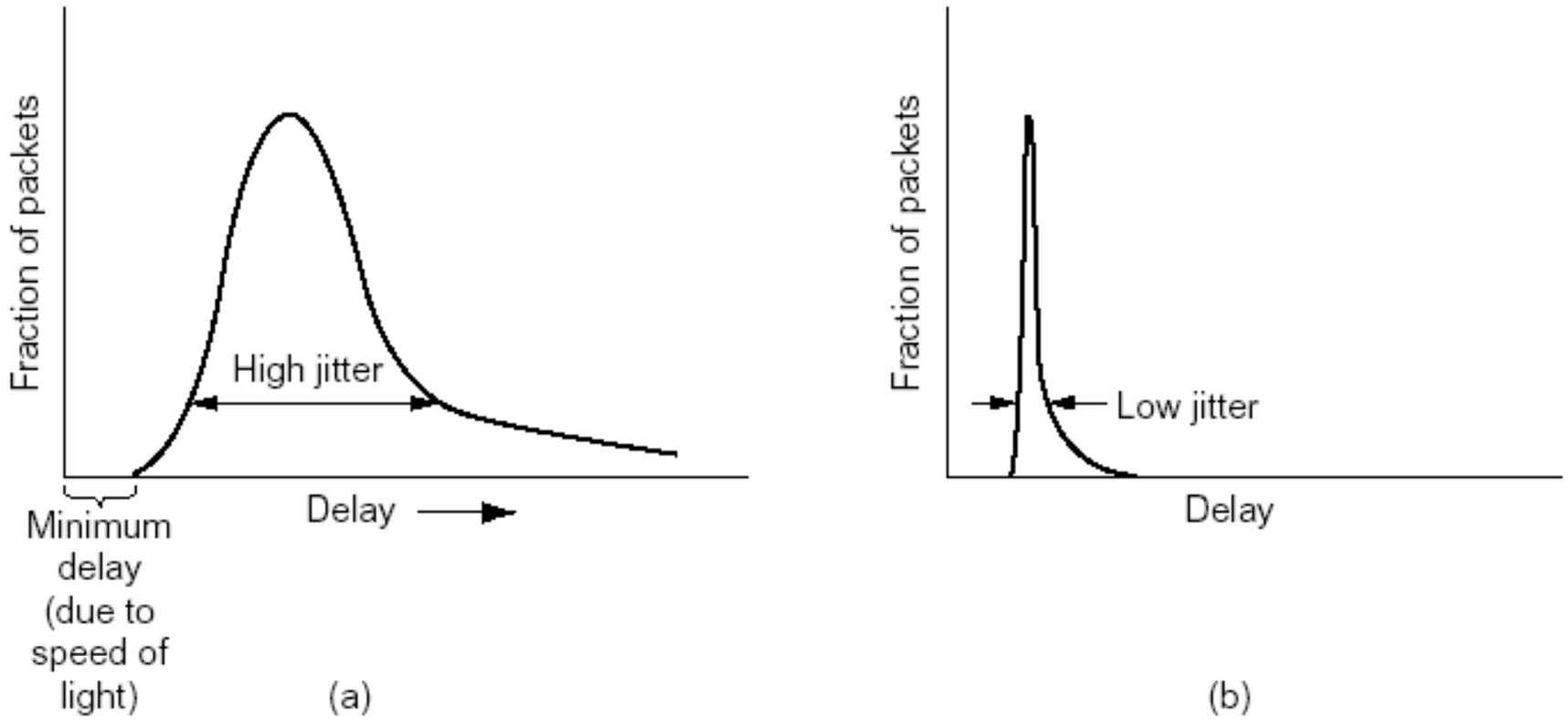
Desprendimiento de Carga

- Cuando el enrutador no puede con todos los paquetes los tira
- como seleccionar los paquetes a tirar?
  - transmision de archios: los mas nuevos.
  - transmision de video: lo mas viejos

Fluctuación (Jitter)

- Fluctuación es la desviación en el retardo de los paquetes
- Retardo: Es el tiempo de transito de los paquetes.
- Fluctuacion es un concepto global: cuanto nos separamos de la media.

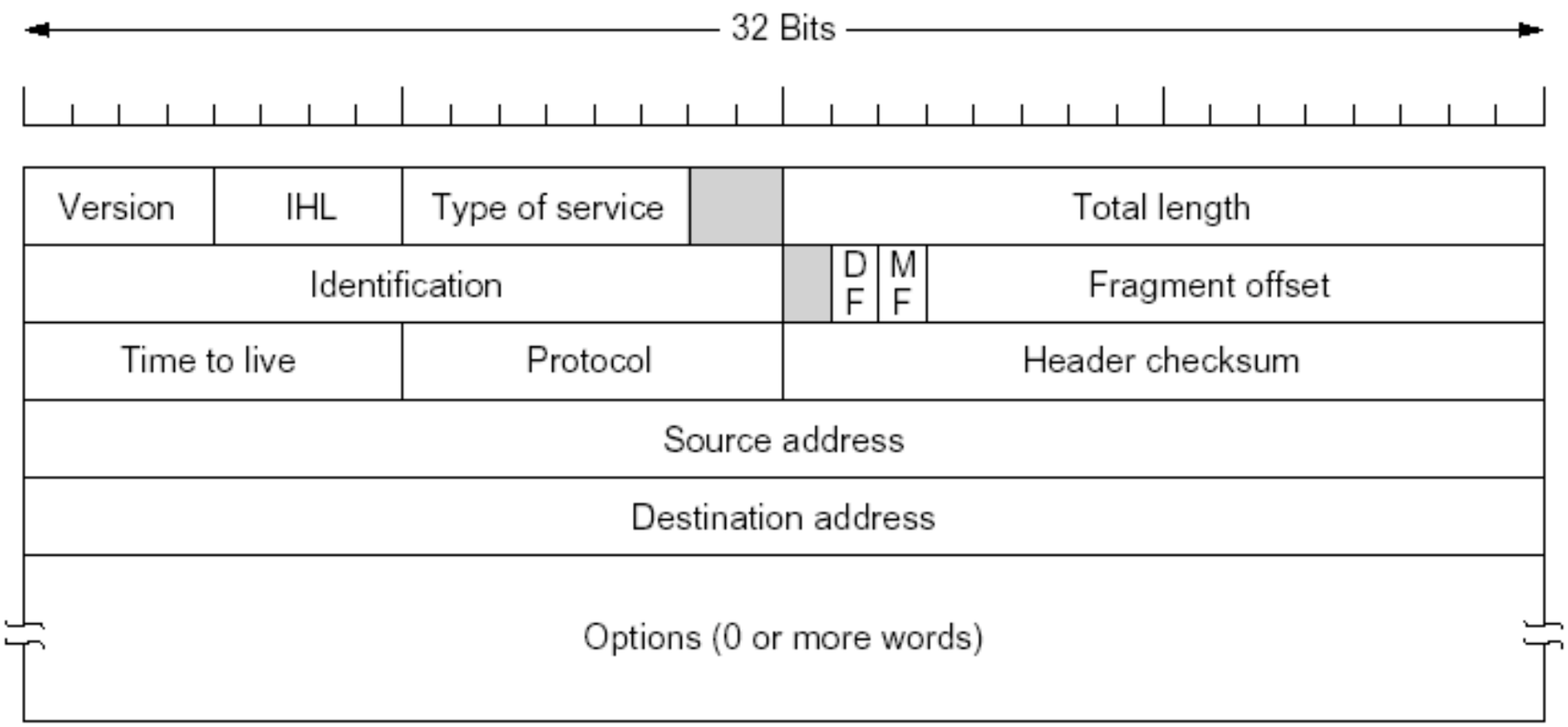
Fluctuación



Calidad de Servicio

Entra y no lo doy, hay preguntas para guiarlos.

IP



IHL: Longitud del encabezado	TOS: 3 bit priority mas 3 bit flag
TLen: Longitud del encabezado mas datos	ID: id datagrama
DF: No fragmentar	MF: Hay mas fragmentos
FOff: Offset en datagrama original	TTL: Maximo numero de hops
Prot: Identificador de protocolo	HCS: Chequea el encabezado