

# Capítulo 4: Capa Red - III

ELO322: Redes de Computadores  
Agustín J. González

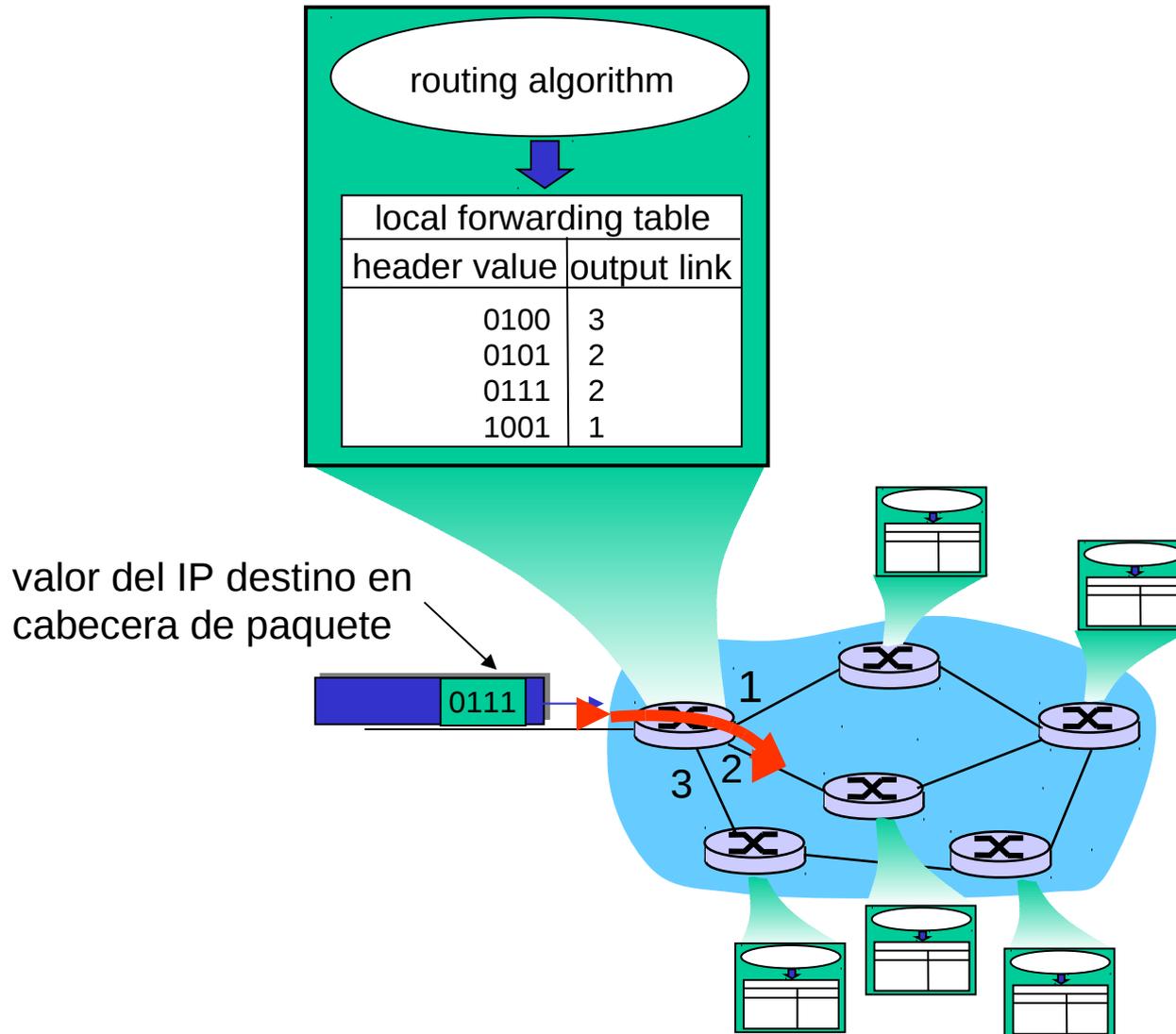
Este material está basado en:

- Material de apoyo al texto *Computer Networking: A Top Down Approach Featuring the Internet 3rd* edition. Jim Kurose, Keith Ross Addison-Wesley, 2004.

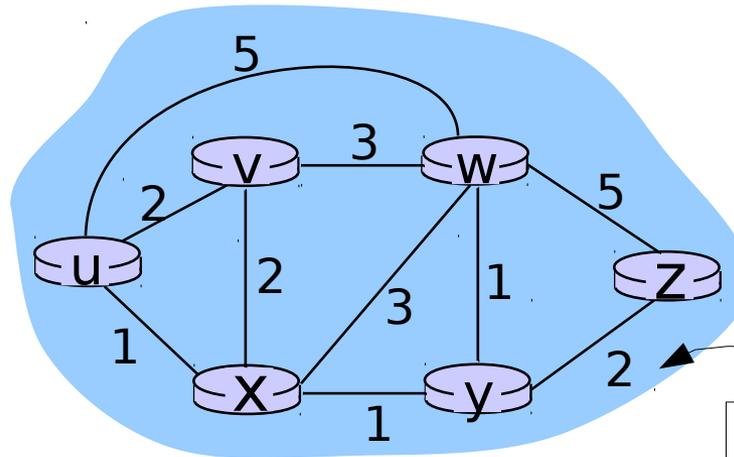
# Capítulo 4: Capa de Red

- 4.1 Introducción
- 4.2 Circuitos virtuales y redes de datagramas
- 4.3 ¿Qué hay dentro de un router?
- 4.4 IP: Internet Protocol
  - Formato de Datagrama
  - Direccionamiento IPv4
  - ICMP
  - IPv6
- 4.5 **Algoritmos de ruteo**
  - Estado de enlace
  - Vector de Distancias
  - Ruteo Jerárquico
- 4.6 Ruteo en la Internet
  - RIP
  - OSPF
  - BGP
- 4.7 Ruteo Broadcast y multicast

# Interacción de ruteo y re-envío



# Abstracción de Grafos



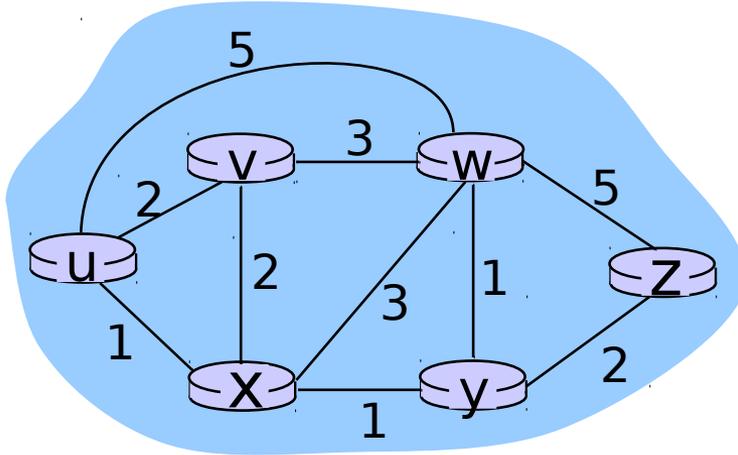
Costo del enlace:  
BW, congestión, \$

Grafo:  $G = (N,E)$

$N =$  conjunto de routers =  $\{ u, v, w, x, y, z \}$

$E =$  conjunto de enlaces =  $\{ (u,v), (u,x), (v,x), (v,w), (x,w), (x,y), (w,y), (w,z), (y,z) \}$

# Abstracción de Grafos : costos



- $c(x, y)$  = costo de enlace  $(x, y)$ 
  - e.g.,  $c(w,z) = 5$
- costo siempre puede ser 1, o inversamente relacionado al ancho de banda o directamente relacionado a la congestión

Costo de la ruta  $(x_1, x_2, x_3, \dots, x_p) = c(x_1, x_2) + c(x_2, x_3) + \dots + c(x_{p-1}, x_p)$

Pregunta: ¿Cuál es la ruta de mínimo costo entre u y z ?

Algoritmo de ruteo: algoritmo que encuentra el costo mínimo

# Clasificación de los algoritmos de ruteo

Según si usa información global o descentralizada?

Global:

- ▣ Todos los routers conocen la topología completa y costos de enlaces
- ▣ Algoritmos de “estado de enlace” (link state)

Descentralizada:

- ▣ El router conoce vecinos conectados físicamente y el costo del enlace a ellos.
- ▣ Proceso iterativo de cómputo e intercambio de información con sus vecinos
- ▣ Algoritmos de “vector de distancia”

Según si es estático o dinámico?

Estático:

- ▣ Rutas cambian poco en el tiempo

Dinámico:

- ▣ Rutas cambian más rápidamente
  - ▣ Actualizaciones periódicas
  - ▣ En respuesta a cambios de costos de enlaces

# Capítulo 4: Capa de Red

- ▣ 4.1 Introducción
- ▣ 4.2 Circuitos virtuales y redes de datagramas
- ▣ 4.3 ¿Qué hay dentro de un router?
- ▣ 4.4 IP: Internet Protocol
  - ▣ Formato de Datagrama
  - ▣ Direccionamiento IPv4
  - ▣ ICMP
  - ▣ IPv6
- ▣ 4.5 **Algoritmos de ruteo**
  - ▣ **Estado de enlace**
  - ▣ Vector de Distancias
  - ▣ Ruteo Jerárquico
- ▣ 4.6 Ruteo en la Internet
  - ▣ RIP
  - ▣ OSPF
  - ▣ BGP
- ▣ 4.7 Ruteo Broadcast y multicast

# Un Algoritmo de ruteo “estado de enlace”

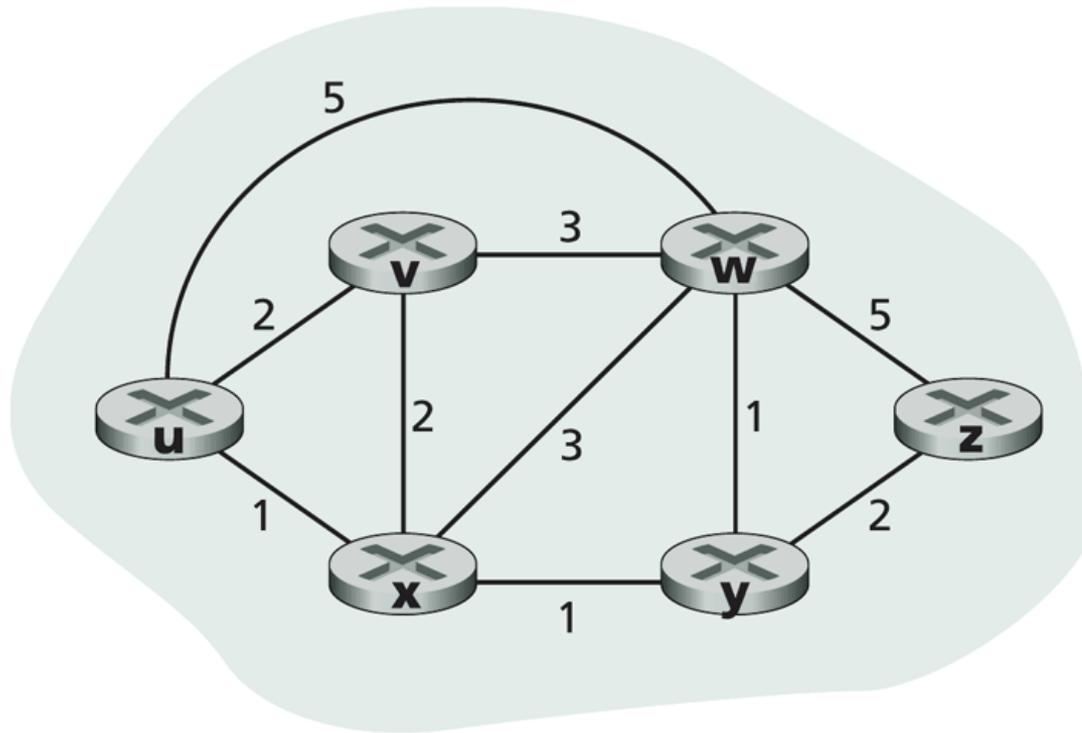
## Algoritmo de Dijkstra

- Supone topología de red y costos de enlaces conocidos a todos los nodos
  - Se logra vía “difusión de estado de enlace”
  - Todos los nodos tienen la misma información
- Se calcula el camino de costo menor desde un nodo (fuente) a todos los otros
  - Determina **tabla de re-envío** para ese nodo
- Iterativo: después de  $k$  iteraciones, se conoce el camino de menor costo a  $k$  destinos (ver los valores de  $p(v)$  en el camino resultante)

## Notación:

- $c(x,y)$ : costo del enlace desde nodo  $x$  a  $y$ ;  $= \infty$  si no es vecino directo
- $D(v)$ : valor actual del costo del camino desde fuente a destino  $v$ .
- $p(v)$ : nodo predecesor a  $v$  en el camino de fuente a  $v$ .
- $N'$ : conjunto de nodos cuyo camino de costo mínimo ya se conoce

# Modelo abstracto para la red



**Figure 4.25** ♦ Abstract graph model of a computer network

# Algoritmo de Dijkstra

## **Inicialización:**

$N' = \{u\}$

for todos los nodos  $v$

if  $v$  es vecino de  $u$

then  $D(v) = c(u,v)$

else  $D(v) = \infty$

## **Loop**

find  $w$  not in  $N'$  tal que  $D(w)$  es un mínimo

agregue  $w$  a  $N'$

actualiza  $D(v)$  para todo  $v$  adyacente a  $w$  que no está en  $N'$  usando:

$$D(v) = \min( D(v), D(w) + c(w,v) )$$

/\* nuevo costo a  $v$  es el costo del camino actual a  $v$  o el costo del camino más corto conocido a  $w$  más el costo de  $w$  a  $v$ \*/

**until todos los nodos están en  $N'$**

## **Notación:**

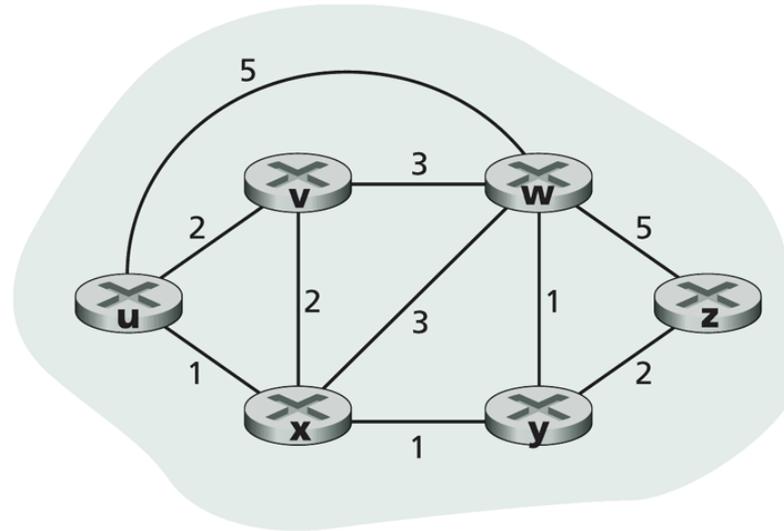
$c(x,y)$ : costo del enlace desde nodo  $x$  a  $y$ ;  $= \infty$  si no es vecino directo

$D(v)$ : valor actual del costo del camino desde fuente a destino  $v$ .

$p(v)$ : nodo predecesor a  $v$  en el camino de fuente a  $v$ .

$N'$ : conjunto de nodos cuyo camino de costo mínimo (desde origen) ya se conoce

# Algoritmo de Dijkstra



step	$N'$	$D(v), p(v)$	$D(w), p(w)$	$D(x), p(x)$	$D(y), p(y)$	$D(z), p(z)$
0	u	2,u	5,u	<u>1,u</u>	$\infty$	$\infty$
1	ux	2,u	4,x		<u>2,x</u>	$\infty$
2	uxy	<u>2,u</u>	3,y			4,y
3	uxyv		3,y			4,y
4	uxyvw					<u>4,y</u>
5	uxyvwz					

**Table 4.3** ♦ Running the link-state algorithm on the network in Figure 4.25

# Algoritmo de Dijkstra, discusión

## Complejidad para $n$ nodos

- Cada iteración: ve todos los nodos,  $w$ , no en  $N$
- $n(n+1)/2$  comparaciones:  $O(n^2)$
- Otras implementaciones son posibles:  $O(n \log n)$

## Oscilaciones en cálculos son posibles:

- e.g., si costo enlace = cantidad de tráfico enviado por enlace

# Capítulo 4: Capa de Red

- ▣ 4.1 Introducción
- ▣ 4.2 Circuitos virtuales y redes de datagramas
- ▣ 4.3 ¿Qué hay dentro de un router?
- ▣ 4.4 IP: Internet Protocol
  - ▣ Formato de Datagrama
  - ▣ Direccionamiento IPv4
  - ▣ ICMP
  - ▣ IPv6
- ▣ 4.5 **Algoritmos de ruteo**
  - ▣ Estado de enlace
  - ▣ **Vector de Distancias**
  - ▣ Ruteo Jerárquico
- ▣ 4.6 Ruteo en la Internet
  - ▣ RIP
  - ▣ OSPF
  - ▣ BGP
- ▣ 4.7 Ruteo Broadcast y multicast

# Algoritmo Vector de Distancia (1)

## Ecuación de Bellman-Ford

Define

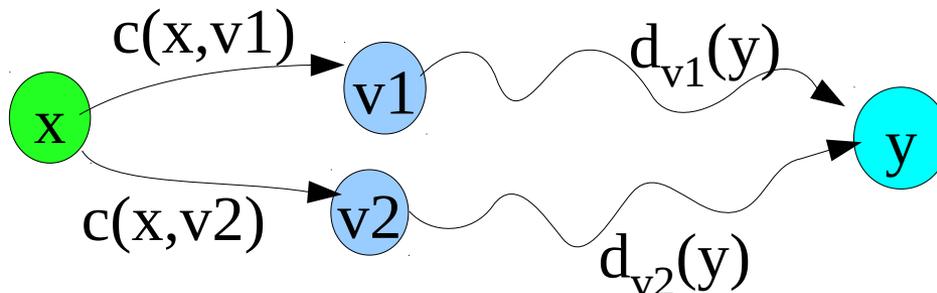
$d_x(y) :=$  costo del camino de menor costo de  $x$  a  $y$

Entonces:

$$d_x(y) = \min \{c(x,v) + d_v(y)\}$$

$v$  es vecino de  $x$

Donde min es tomado sobre todos los vecinos  $v$  de  $x$



## Algoritmo Vector de Distancia (2)

- $D_x(y)$  = costo mínimo estimado de x a y
- Vector de distancia:  $\mathbf{D}_x = [D_x(y): y \in N]$
- Nodo x conoce el costo a cada vecino v:  $c(x,v)$
- Nodo x mantiene  $\mathbf{D}_x = [D_x(y): y \in N]$
- Nodo x también mantiene los vectores de distancia de sus vecinos
  - Para cada vecino v, x mantiene  $\mathbf{D}_v = [D_v(y): y \in N]$

# Algoritmo Vector de distancia (3)

## Idea básica:

- Cada nodo envía periódicamente su vector de distancia estimada a sus vecinos
- Cuando el nodo  $x$  recibe un nuevo vector de dist. estimada desde un vecino, éste actualiza su propio vector de dist. usando la ecuación de B-F:

$$D_x(y) \leftarrow \min_v \{c(x,v) + D_v(y)\} \quad \text{para cada nodo } y \text{ en } N$$

- Si el vector de dist. cambia entonces el nodo  $x$  envía su nuevo vector a sus vecinos, y ellos a su vez pueden actualizar sus vectores de distancia
- Bajo condiciones normales, el valor estimado de  $D_x(y)$  converge al menor costo real  $d_x(y)$

# Algoritmo Vector de Distancia (4)

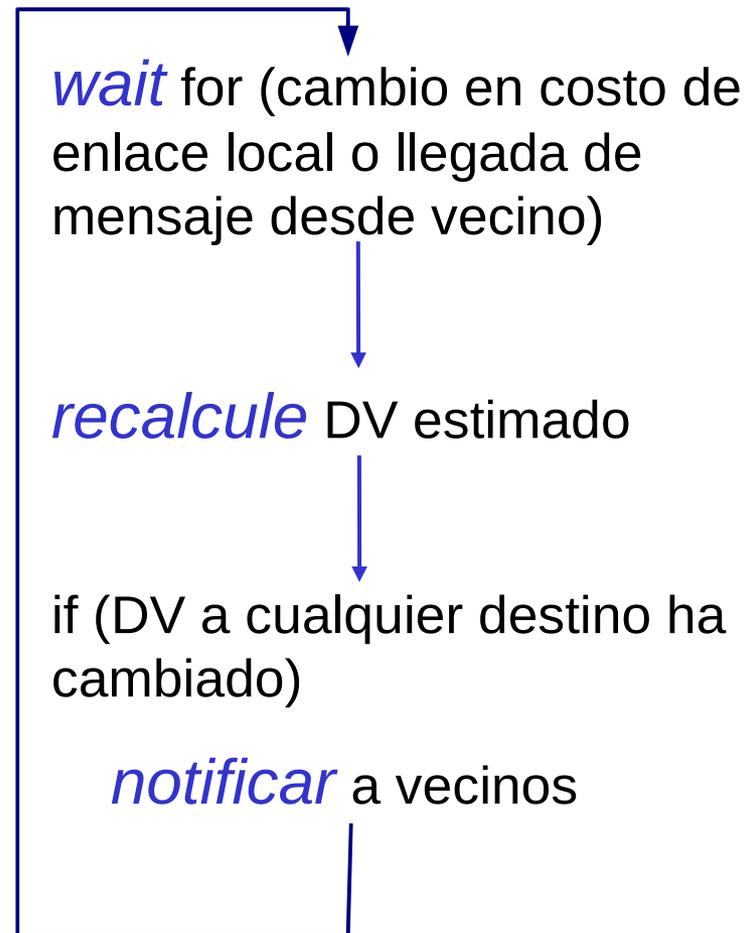
**Iterativo y asincrónico:** cada iteración local es causada por:

- Cambio en costo de enlace local
- Actualización de vector por mensaje de vecino

**Distribuido:**

- Cada nodo notifica a sus vecinos **sólo** cuando su vector cambia
  - Vecinos entonces notifican a sus vecinos si es necesario

**Cada nodo:**



$$D_x(y) = \min\{c(x,y) + D_y(y), c(x,z) + D_z(y)\}$$

$$= \min\{ 2+0 , 7+1 \} = 2$$

$$D_x(z) = \min\{c(x,y) + D_y(z), c(x,z) + D_z(z)\}$$

$$= \min\{ 2+1 , 7+0 \} = 3$$

**node x table**

		cost to		
		x	y	z
from	x	0	2	7
	y	∞	∞	∞
	z	∞	∞	∞

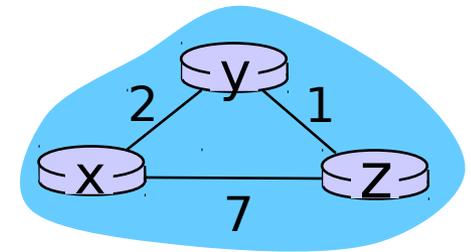
**node y table**

		cost to		
		x	y	z
from	x	∞	∞	∞
	y	2	0	1
	z	∞	∞	∞

**node z table**

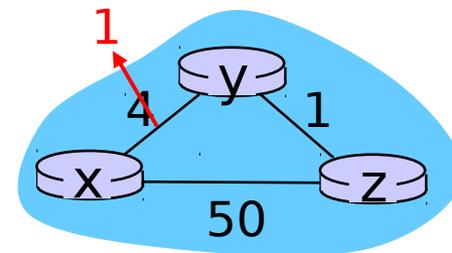
		cost to		
		x	y	z
from	x	∞	∞	∞
	y	∞	∞	∞
	z	7	1	0

Ejemplo:  
Vector de  
distancia



# Vector de distancia: cambios en costos de enlaces

- ❑ **Cambios en costos de enlaces:**
- ❑ nodo detecta un cambio de costo en uno de sus enlaces
- ❑ actualiza información de ruteo, recalcula vector de distancia
- ❑ si hay cambio en DV notifica a sus vecinos



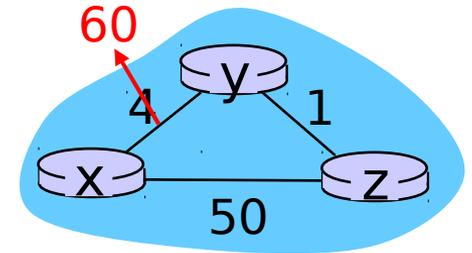
En el tiempo  $t_0$ , y detecta un cambio en costo de enlace, actualiza su DV e informa a sus vecinos.

En el tiempo  $t_1$ , z recibe la información de y, también actualiza su tabla. Calcula un nuevo costo para x y le envía su DV a sus vecinos.

En el tiempo  $t_2$ , y recibe la actualización de z y actualiza su tabla de distancia. Los costos mínimos de y no cambian, y no envía ningún nuevo mensaje a z.

# Vector de distancia: cambio en costo de enlaces

- **Cambio en costos de enlaces:**
- buenas noticias viajan rápido
- noticias malas viajan lento
- ¿Cómo pasa esto?



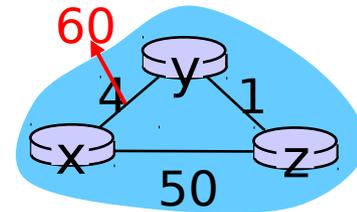
## Vector de distancia: cambio en costo de enlaces (e.g. incremento de costo)

- Inicialmente:  $D_y(x) = 4$ ,  $D_y(z) = 1$ ,  $D_z(x) = 5$ ,  $D_z(y) = 1$
- En el tiempo  $t_0$  y detecta el cambio de costo y calcula:

$$D_y(x) = \min \{c(y,x) + D_x(x), c(y,z) + D_z(x)\} = \\ = \min \{60 + 0, 1 + 5\} = 6$$

### **node y table**

		x	y	z
from	x	0	4	5
	y	4	0	1
	z	5	1	0



### **node y table**

		x	y	z
from	x	0	4	5
	y	6	0	1
	z	5	1	0

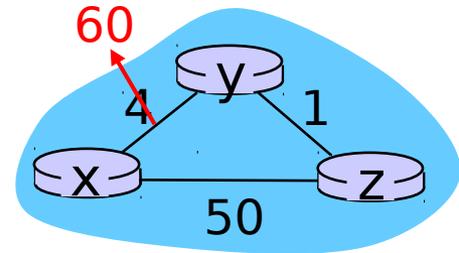
# Vector de distancia: cambio en costo de enlaces

□ ¿Qué pasa si el enlace se cae?

Su costo es  $\infty$ ,  $\Rightarrow$

Reversa envenenada:

- Si Z routea a través de Y para llegar a X:
  - Z le dice a Y que su distancia a X es infinita (para que Y no rutee a X vía Z)
- ¿Resuelve completamente el problema de contar hasta el infinito? No, ¿por qué?



# Comparación de algoritmos de estado de enlace (LS) y vector de distancia (DV)

## Complejidad de mensajes

- ▢ LS: con  $n$  nodos,  $E$  enlaces,  $O(nE)$  mensajes son enviados
- ▢ DV: sólo intercambios entre vecinos
  - ▢ Tiempo de convergencia varía

## Rapidez de convergencia

- ▢ LS:  $O(n^2)$ , algoritmo requiere  $O(nE)$  mensajes
  - ▢ Puede tener oscilaciones
- ▢ DV: tiempo de convergencia varía
  - ▢ Podría estar en loops
  - ▢ Problema de cuenta infinita

**Robustez:** ¿qué pasa si un router funciona mal?

### LS:

- ▢ Nodos pueden comunicar costo incorrecto del *link*
- ▢ Cada nodo computa sólo su propia tabla

### DV:

- ▢ DV nodo puede comunicar costo de *camino* incorrecto
- ▢ La tabla de cada nodo es usada por otros
  - error se propaga a través de la red

# Capítulo 4: Capa de Red

- 4.1 Introducción
- 4.2 Circuitos virtuales y redes de datagramas
- 4.3 ¿Qué hay dentro de un router?
- 4.4 IP: Internet Protocol
  - Formato de Datagrama
  - Direccionamiento IPv4
  - ICMP
  - IPv6
- 4.5 Algoritmos de ruteo
  - Estado de enlace
  - Vector de Distancias
  - Ruteo Jerárquico
- 4.6 Ruteo en la Internet
  - RIP
  - OSPF
  - BGP
- 4.7 Ruteo Broadcast y multicast

# Ruteo Jerárquico

- Nuestro estudio del ruteo hasta ahora es idealizado. Suponemos que:
- Todos los routers son idénticos
- La red es “plana”
- ... esto *no es* verdad en la práctica

**Escala:** con 200 millones de destinos:

- No podemos almacenar todos los destinos en tablas de ruteo!
- Los intercambios de tablas de ruteo inundarían los enlaces!

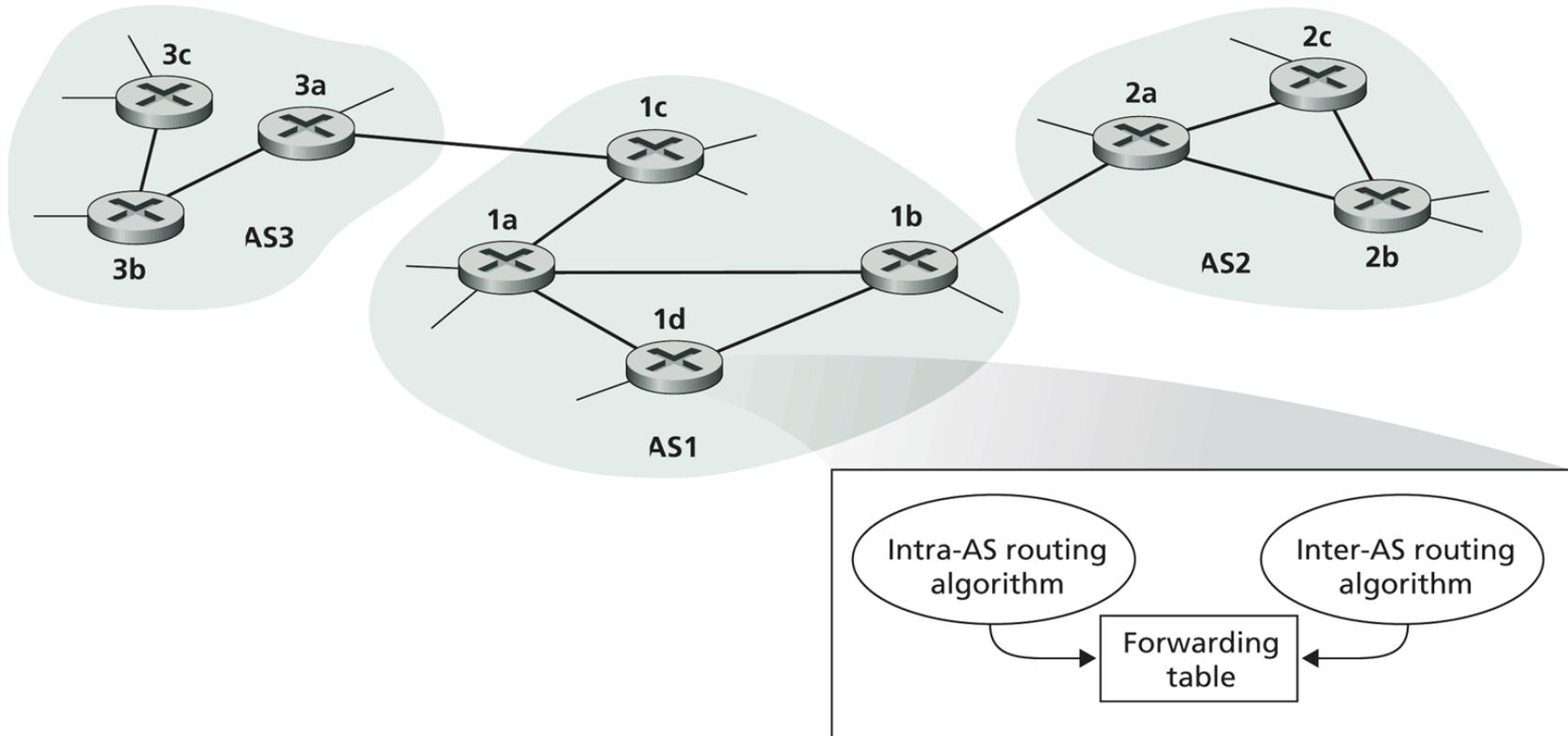
**Autonomía administrativa**

- Internet = red de redes
- Cada administrador de red puede querer controlar el ruteo en su propia red

# Ruteo Jerárquico

- ▣ Agrupar router en regiones, “sistemas autónomos” (autonomous systems o AS)
- ▣ Routers en el mismo AS usan el mismo protocolo de ruteo
  - ▣ Protocolo de ruteo “intra-AS”
  - ▣ Routers en diferentes AS pueden correr diferentes protocolos intra-AS
- ▣ Router de borde (Gateway router)
- ▣ Tienen enlace directo a router en otros sistemas autónomos

# Ruteo Jerárquico



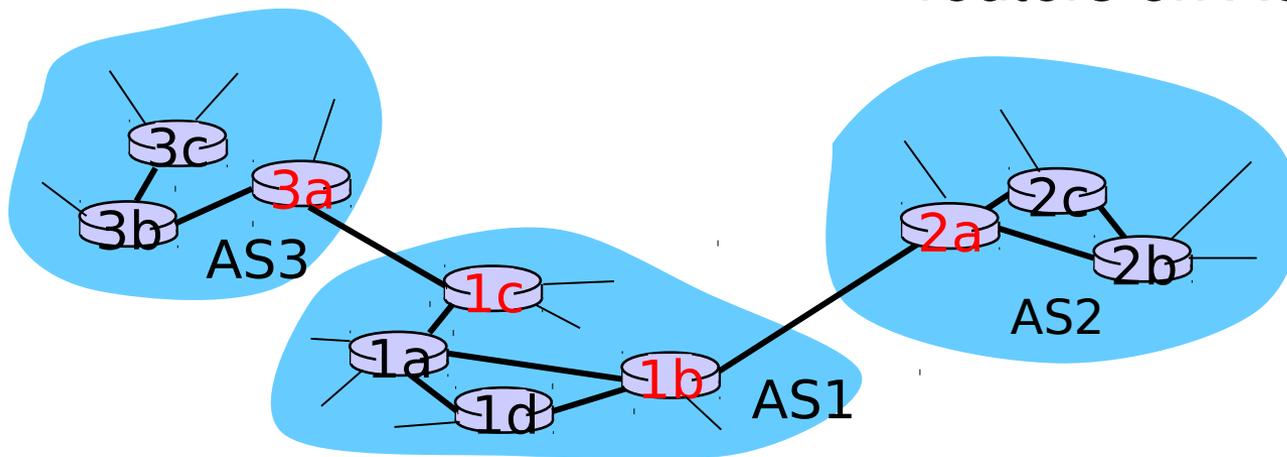
**Figure 4.29** ♦ An example of interconnected autonomous systems

# Ruteo Inter-AS

- Router en AS1 recibe un datagrama para un destino afuera de AS1
  - A cuál Router debería enviar el paquete?

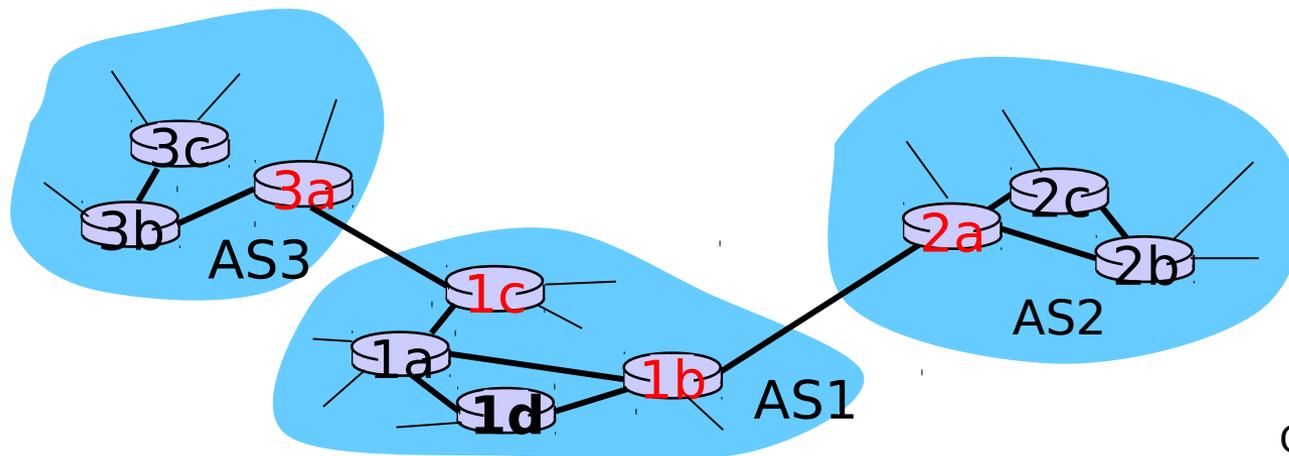
## AS1 necesita:

1. aprender qué destinos son alcanzables a través de AS2 y cuáles a través de AS3
2. propagar esta información a todos los routers en AS1



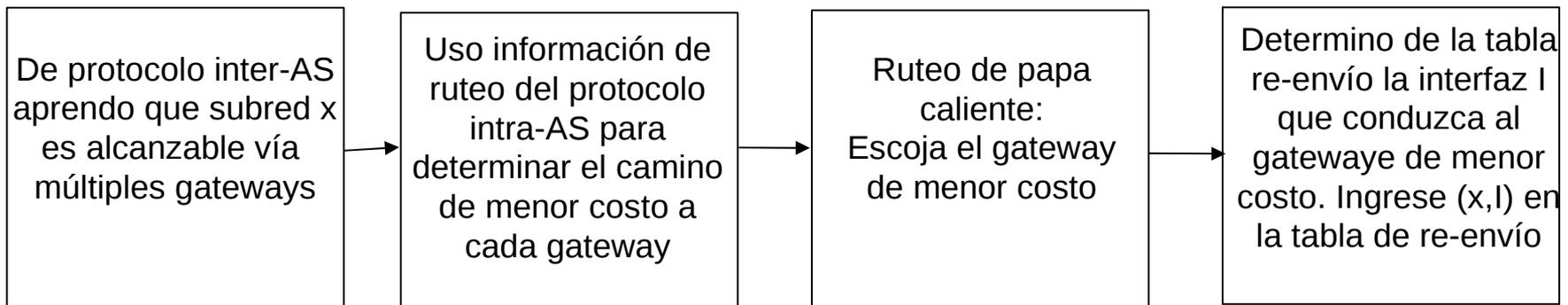
# Ejemplo: definición de la tabla de re- envío en router 1d

- Supongamos que AS1 sabe por el protocolo inter-AS que la subred **x** es alcanzable desde AS3 (gateway 1c) pero no desde AS2.
- El protocolo intra-AS propaga la información de alcance a todos los routers internos.
- Router **1d** determina de la información de ruteo intra-AS que su interfaz **1** está en el camino de costo mínimo a 1c.
- Luego éste pone en su tabla de re-envío: **(x, 1)**.



# Ejemplo: Elección entre múltiples AS

- Ahora supongamos que AS1 sabe por el protocolo inter-AS que la subred **x** es alcanzable desde AS3 y desde AS2.
- Para configurar la tabla de re-envío, router 1d debe determinar hacia qué gateway éste debería re-enviar los paquetes destinados a **x**.
- Ésta es también una tarea del protocolo de ruteo inter-AS
- **Ruteo de la papa caliente (Hot potato routing)**: enviar el paquete hacia el router más cercano de los dos.



# Capítulo 4: Capa de Red

- 4.1 Introducción
- 4.2 Circuitos virtuales y redes de datagramas
- 4.3 ¿Qué hay dentro de un router?
- 4.4 IP: Internet Protocol
  - Formato de Datagrama
  - Direccionamiento IPv4
  - ICMP
  - IPv6
- 4.5 Algoritmos de ruteo
  - Estado de enlace
  - Vector de Distancias
  - Ruteo Jerárquico
- 4.6 **Ruteo en la Internet**
  - RIP
  - OSPF
  - BGP
- 4.7 Ruteo Broadcast y multicast