

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

Clasificación de los algoritmos de ruteo

Según información global o descentralizada?

Global:

- ❑ Todos los routers tienen la topología completa y costos de enlaces
- ❑ Algoritmo “estado de enlace”
- ❑ Descentralizada:
- ❑ El router conoce vecinos conectados físicamente y su costo del enlace a ellos.
- ❑ Proceso iterativo de cómputo e intercambio de información con sus vecinos
- ❑ Algoritmo “vector de distancia”

Según si es estático o dinámico?

Estático:

- ❑ routes cambian lentamente en el tiempo

Dinámico:

- ❑ routes cambias 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 de 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
 - Entrega la **tabla de re-envío** para ese nodo
- ❑ iterativo: después de k iteraciones, conoce camino de menor costo a k destinos

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

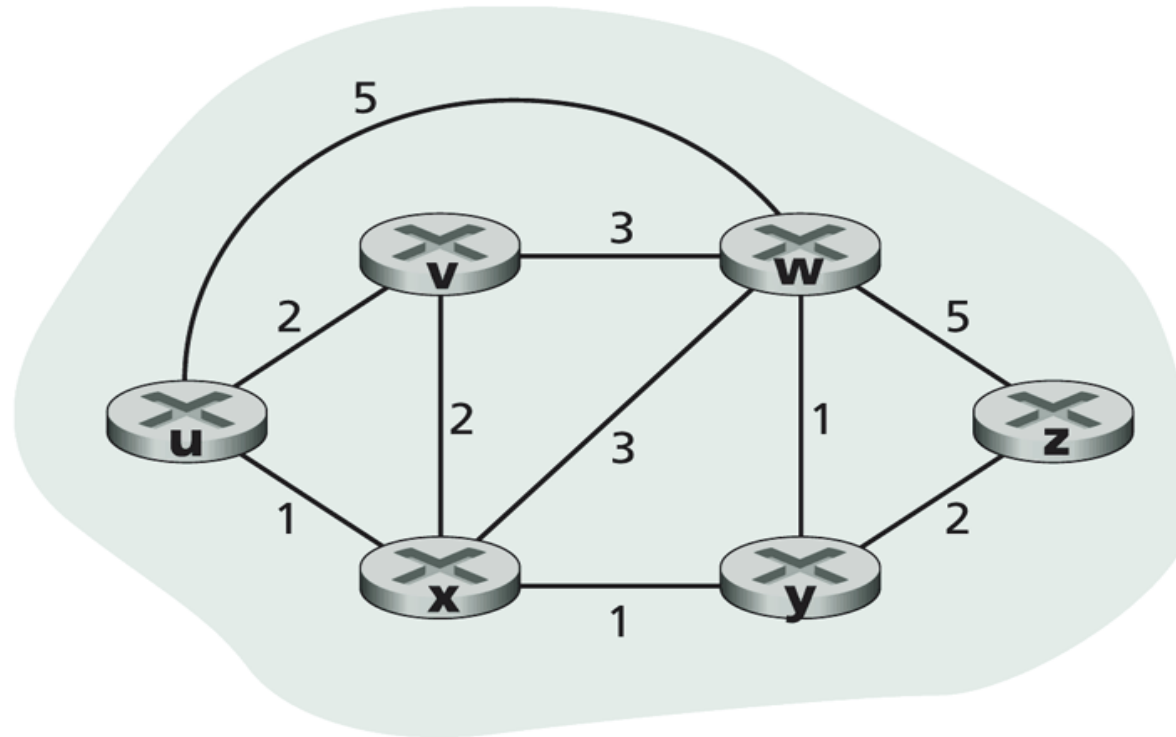


Figure 4.25 ♦ Abstract graph model of a computer network

Algoritmo de Dijkstra

1 **Inicialización:**

2 $N' = \{u\}$

3 for all nodes v

4 if v adjacent to u

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

6 else $D(v) = \infty$

7

8 **Loop**

9 find w not in N' such that $D(w)$ is a minimum

10 add w to N'

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

12 $D(v) = \min(D(v), D(w) + c(w,v))$

13 /* nuevo costo a v es ya sea el costo del camino actual a v o

14 el costo del camino más corto conocido a w más el costo de w a v^* /

15 **until all nodes in N'**

Notación:

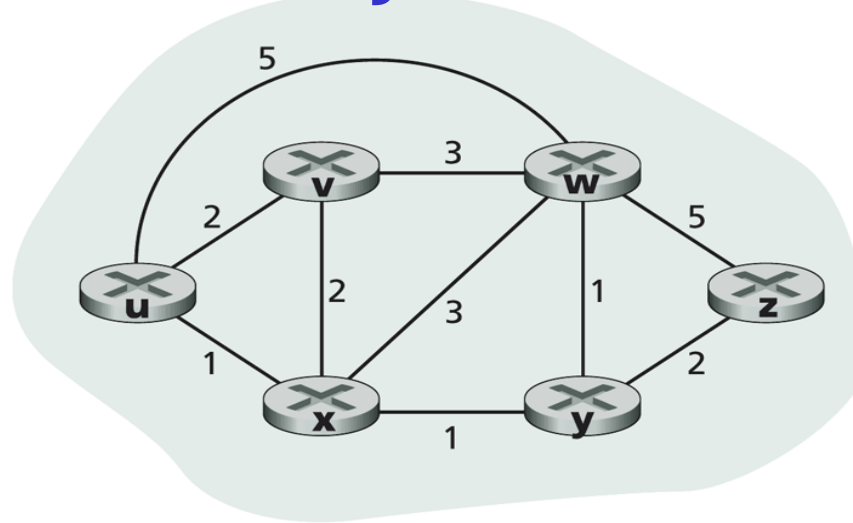
$c(x,y)$: costo del enlace desde nodo x a y ; $= \infty$ si no es vecino directo

$D(v)$: 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 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>	∞	∞
1	ux	2,u	4,x		<u>2,x</u>	∞
2	uxy	<u>2,u</u>	3,y			4,y
3	uxyv		<u>3,y</u>			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

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 (programación dinámica)

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 estimado a sus vecinos
- Cuando el nodo x recibe un nuevo DV estimado desde un vecino, éste actualiza su propio DV usando la ecuación de B-F:

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

- Bajo condiciones naturales, el valor estimado de $D_x(y)$ converge al menor costo real $d_x(y)$

Algoritmo Vector de Distancia (4)

Iterativo y asincrónico:

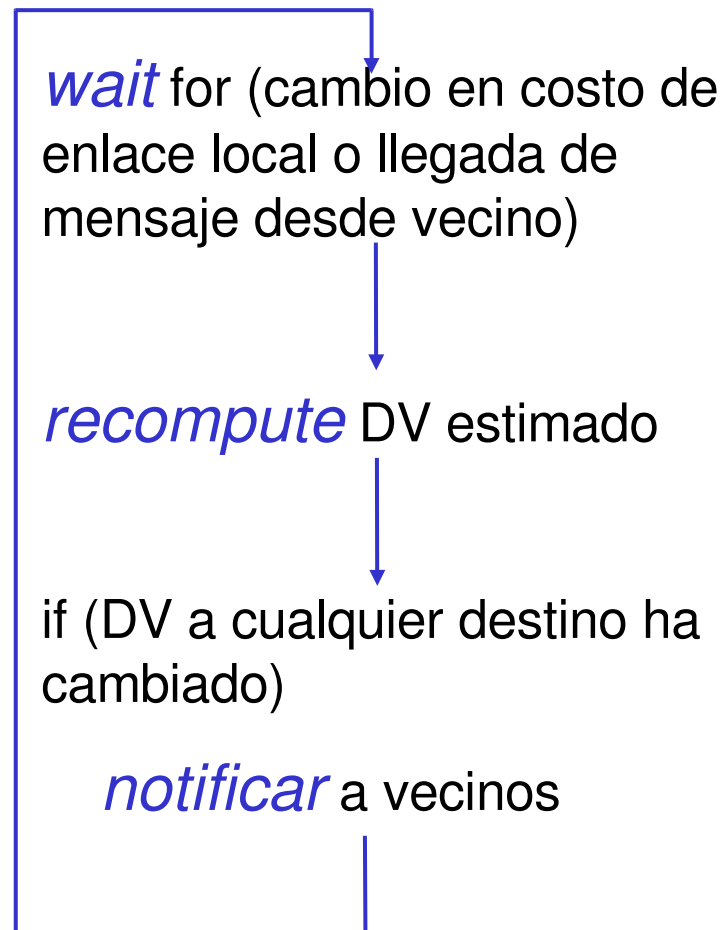
cada iteración local es causada por:

- ❑ Cambio en costo de enlace local
- ❑ Actualización de DV por mensaje de vecino

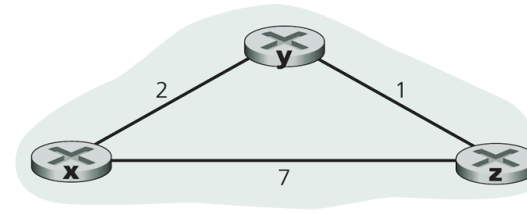
Distribuido:

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

Cada nodo:



Ejemplo: Vector de distancia



Node x table

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

		cost to		
from		x	y	z
	x	0	2	3
	y	2	0	1
	z	7	1	0

		cost to		
from		x	y	z
	x	0	2	3
	y	2	0	1
	z	3	1	0

Node y table

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

		cost to		
from		x	y	z
	x	0	2	7
	y	2	0	1
	z	7	1	0

		cost to		
from		x	y	z
	x	0	2	3
	y	2	0	1
	z	3	1	0

Node z table

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

		cost to		
from		x	y	z
	x	0	2	7
	y	2	0	1
	z	3	1	0

		cost to		
from		x	y	z
	x	0	2	3
	y	2	0	1
	z	3	1	0

.....> Time

Figure 4.27 ♦ Distance vector (DV) algorithm

Casos en algoritmo DV

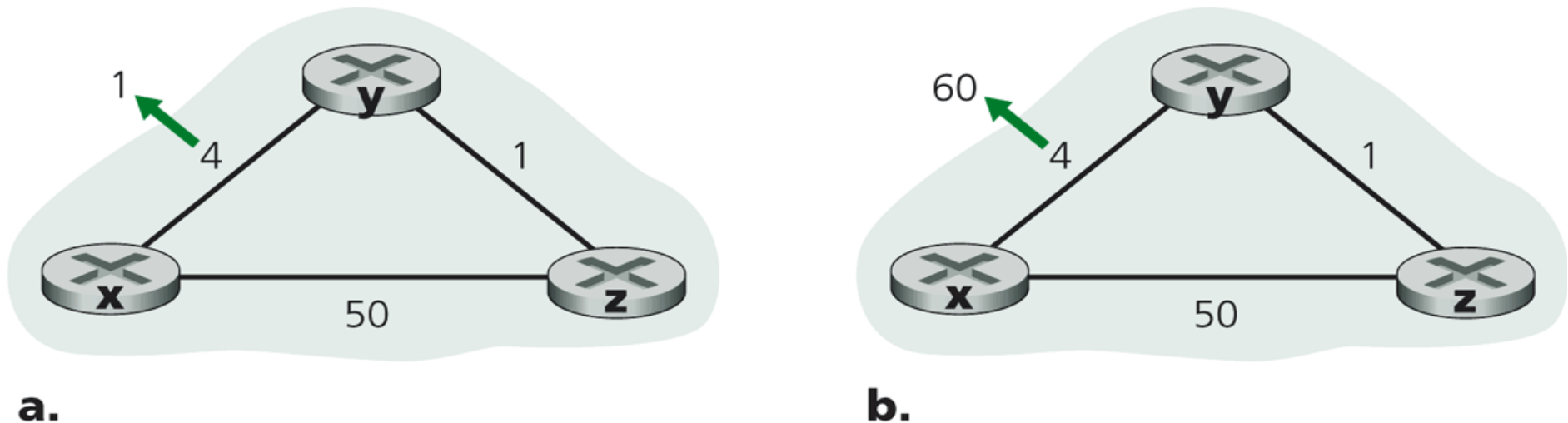


Figure 4.28 ♦ Changes in link cost

- ❑ Considerar que **y** detecta el cambio:
- ❑ El caso **a** conduce a una situación estable en dos iteraciones.
- ❑ Caso **b** conduce a un loop
- ❑ Caso **b** la actualización toma mucho tiempo, se conoce como problema de cuenta infinita.
- ❑ ¿Solución?: Que nodo **z** informe a su vecino **y** que su ruta a **x** es infinita cuando **z** llega a **x** vía **y**
- ❑ El problema sigue cuando el loop involucra más de dos nodos

Comparación de Algoritmos de estado (LS) de enlace 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 incorrecto costo *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 inundarías 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, AS)
- ❑ Routers en el mismo AS corren 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

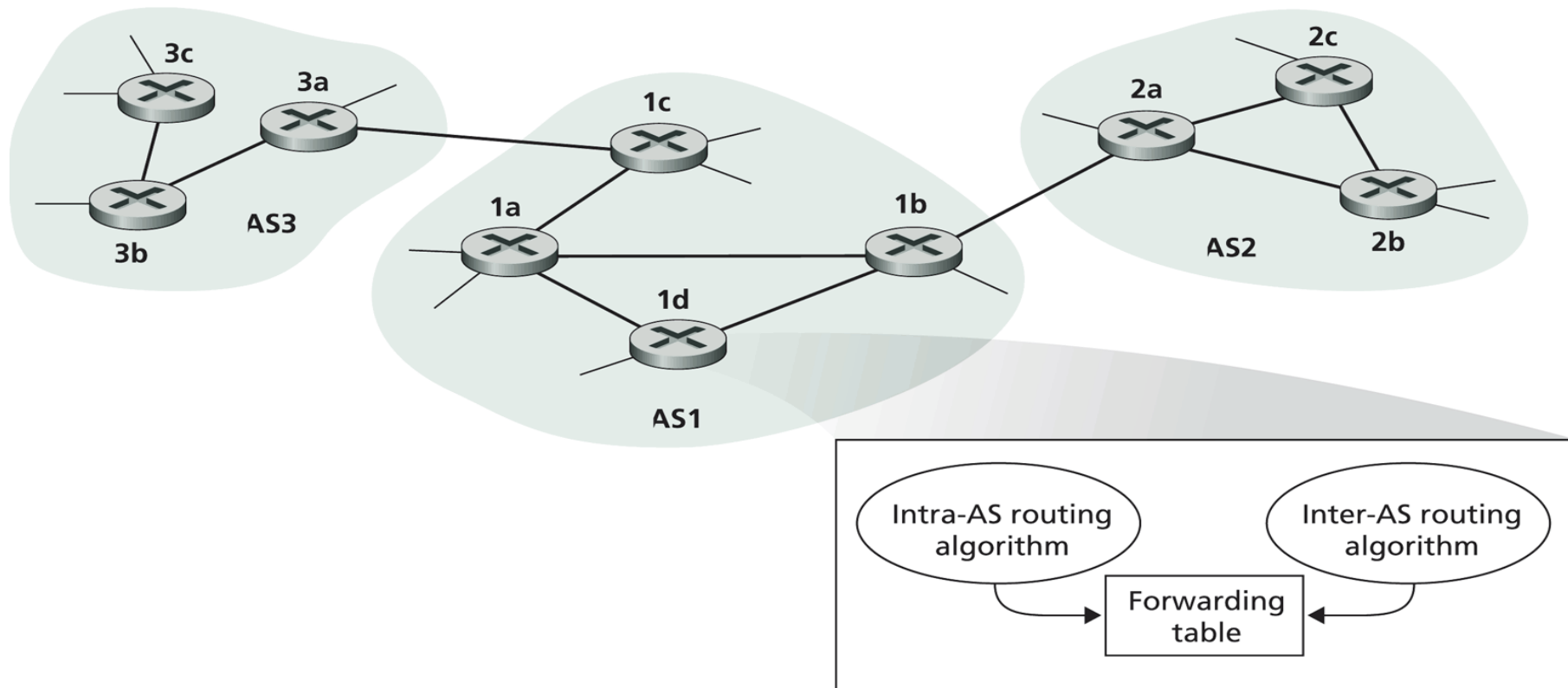


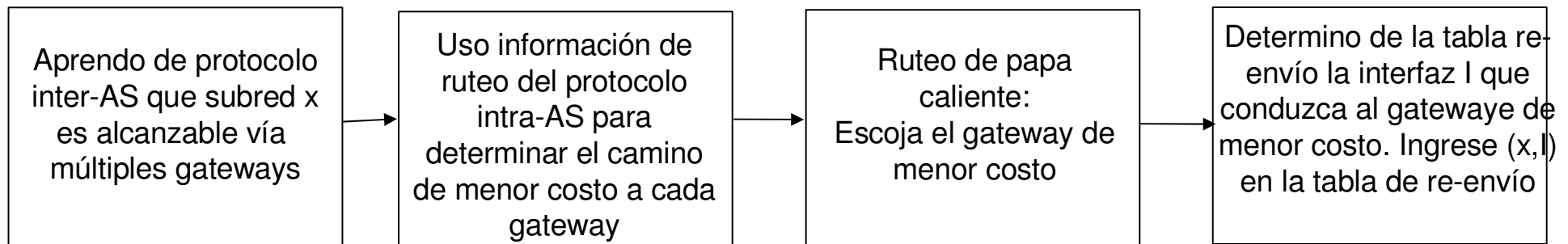
Figure 4.29 ♦ An example of interconnected autonomous systems

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

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

Ejemplo: Elección entre múltiples ASes

- Ahora supongamos que AS1 aprende del 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