



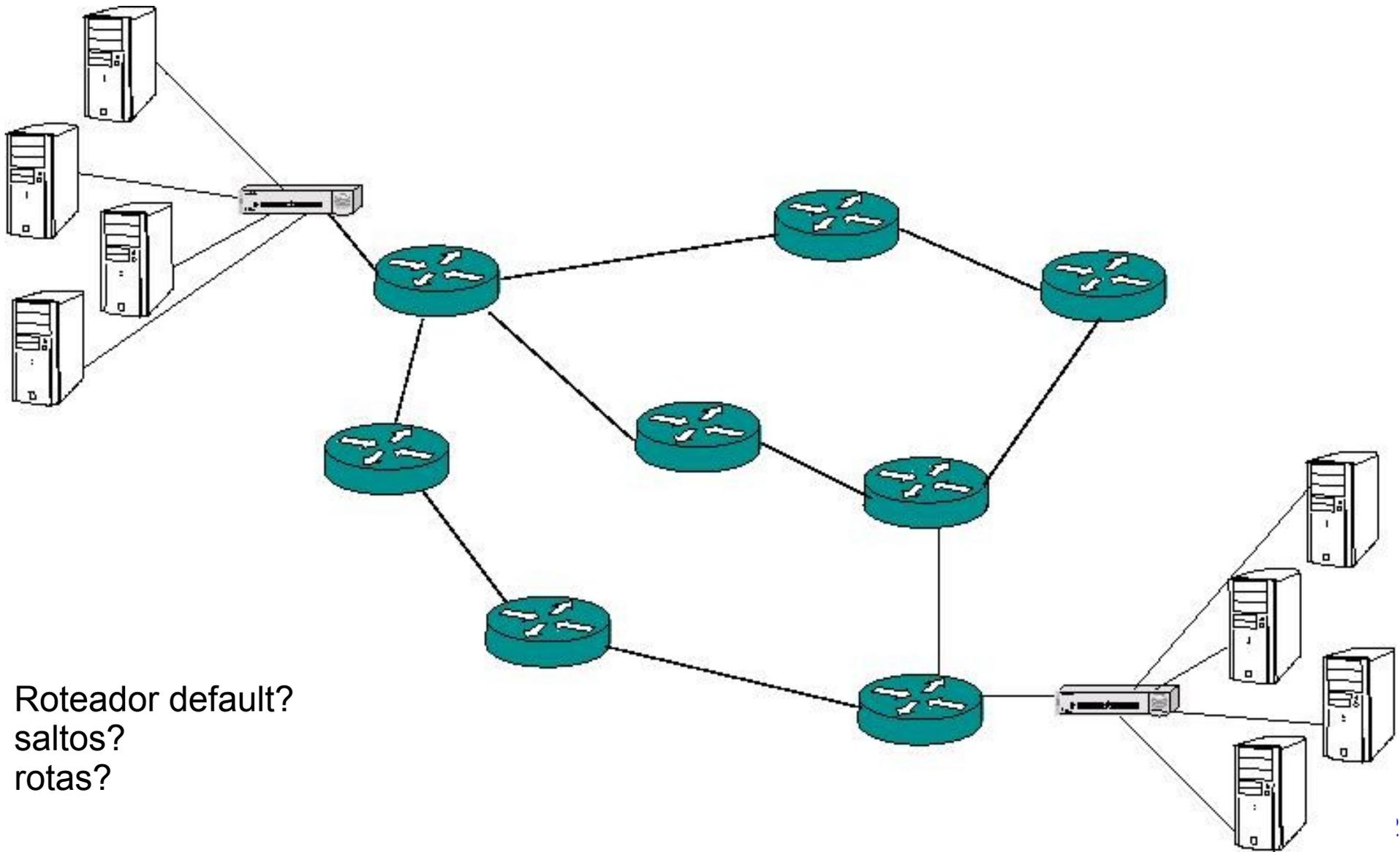
Redes de Computadores

Capítulo 4.5 – Algoritmos de Roteamento

Capítulo 4.6 – Roteamento na Internet

Prof. Jó Ueyama
Abril/2011

Rede



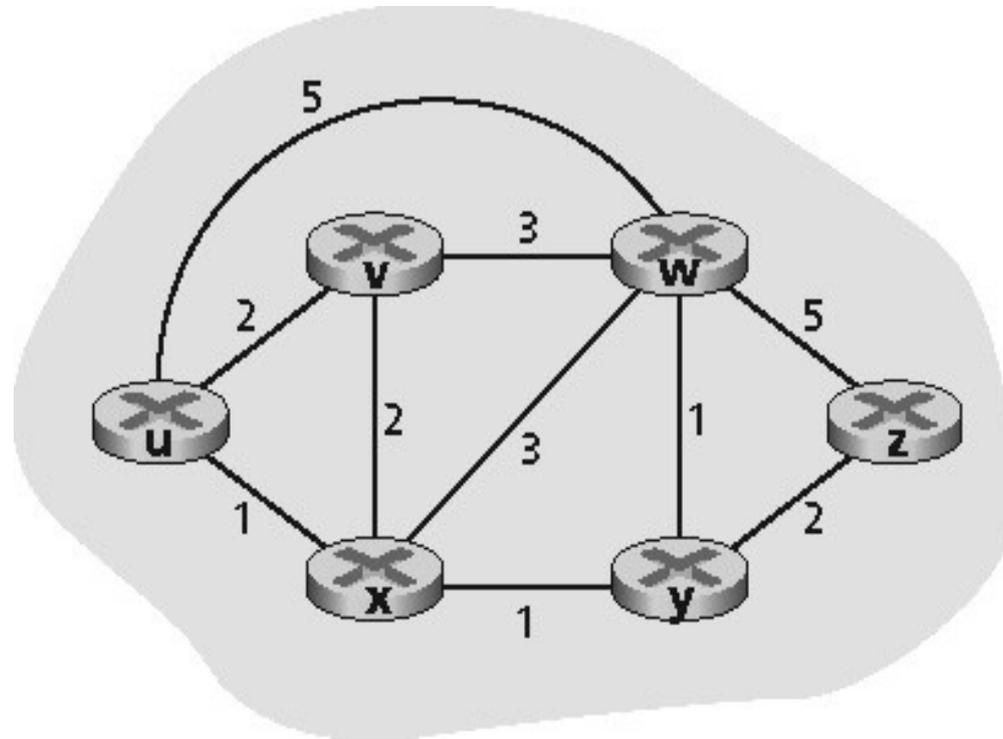
Roteador default?
saltos?
rotas?



Roteamento

- Como as rotas são determinadas?
 - estaticamente;
 - através de algoritmos de roteamento.
- Como escolher um bom caminho?
 - aquele de menor “custo”.

Grafo

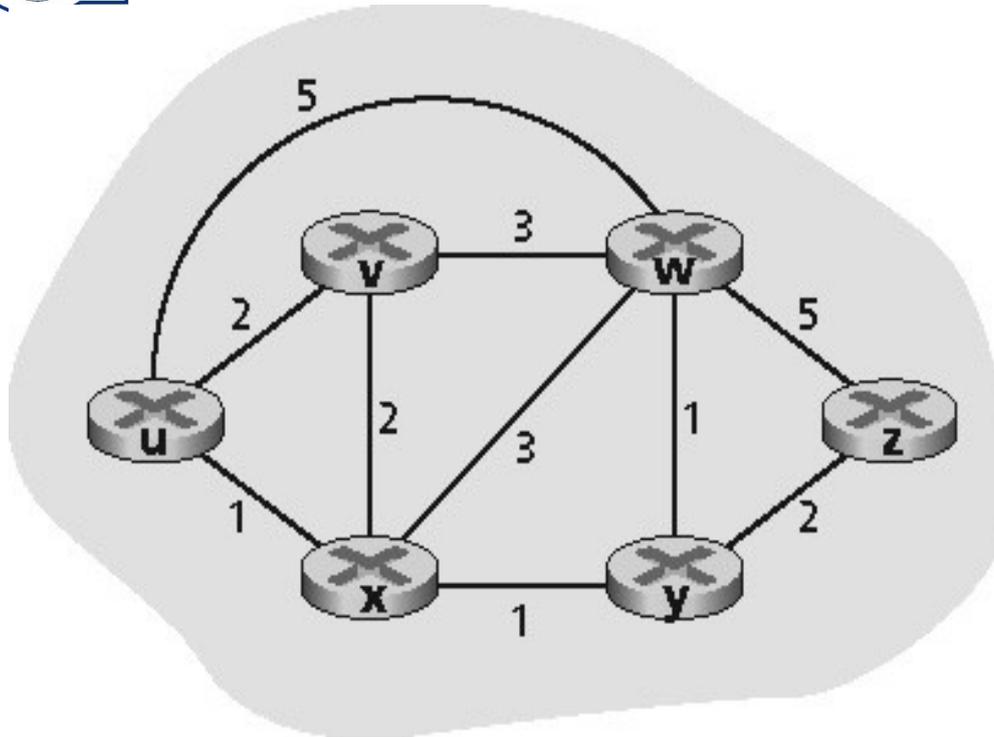


Grafo: $G = (N, E)$

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

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

Grafos: custo dos enlaces



- $c(x, x')$ = custo do link (x, x')
 - ex., $c(w, z) = 5$
- Custo poderia ser sempre 1, ou relacionado à largura de banda ou ao congestionamento.

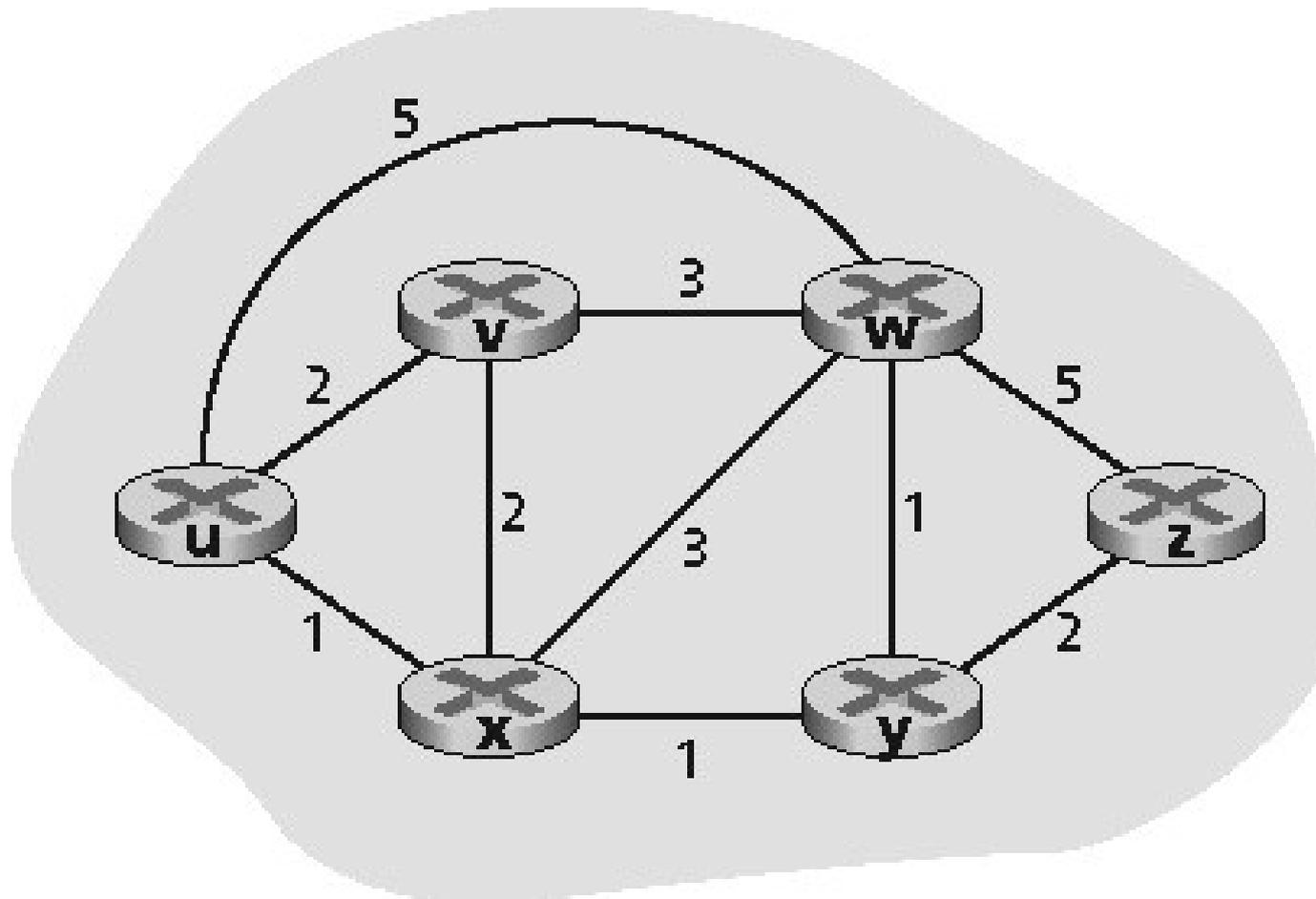
Custo do caminho $(x_1, x_2, \dots, x_p) = c(x_1, x_2) + c(x_2, x_3) + \dots + c(x_{p-1}, x_p)$

Questão: Qual é o caminho de menor custo entre u e z ?

Algoritmo de roteamento encontra o caminho de menor custo.

Exercício

- Qual o caminho de menor custo entre u e z ?





Classificação quanto a informação

- ▽ Global:
 - todos os roteadores têm informações completas da topologia e dos custos dos enlaces.
 - algoritmos de estado do enlace (*link state*).
- ▽ Descentralizada:
 - roteadores só conhecem informações sobre seus vizinhos e os enlaces para eles;
 - processo de computação interativo, troca de informações com os vizinhos;
 - algoritmos de vetor de distâncias (*distance vector*).



Classificação quanto ao tipo

- ▽ **Estático:**
 - as rotas não mudam ou mudam lentamente ao longo do tempo;
 - normalmente são criadas manualmente.

- ▽ **Dinâmico:**
 - as rotas mudam mais rapidamente;
 - podem responder a mudanças no custo dos enlaces;
 - atualizações periódicas;
 - suscetíveis a problemas como *loops* de roteamento e oscilação em rotas.



Roteamento de vetor de distâncias (DV)



Roteamento de vetor de distâncias (DV)

Iterativo, assíncrono: cada iteração local é causada por:

- mudança no custo do enlace local;

- mensagem de atualização DV do vizinho.

Assíncrono: todos os nós não precisam atualizar simultaneamente.

Distribuído: cada nó notifica os vizinhos apenas quando seu DV mudar:

- os vizinhos então notificam seus vizinhos, se necessário.



Roteamento de vetor de distâncias (DV)

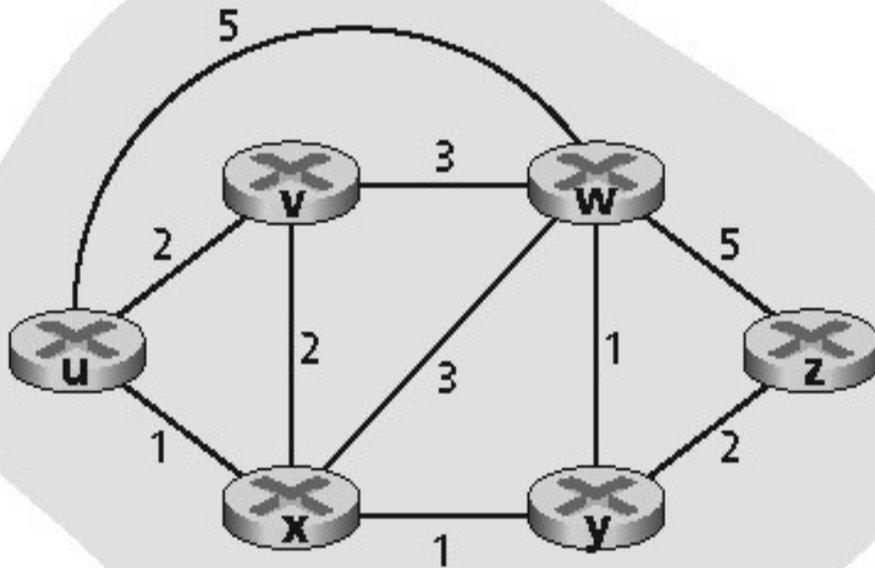
Baseado na Equação de Bellman-Ford (programação dinâmica).

Define $d_x(y)$ = custo do caminho de menor custo de x para y .

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

onde \min é calculado sobre todos os vizinhos de x .

Exemplo da Equação de Bellman-Ford



Para obter a distância entre u e z, temos:

$$\begin{aligned}
 d_u(z) &= \min \{ c(u,v) + d_v(z), \\
 &\quad c(u,x) + d_x(z), \\
 &\quad c(u,w) + d_w(z) \} \\
 &= \min \{ 2 + 5, \\
 &\quad 1 + 3, \\
 &\quad 5 + 3 \} = 4
 \end{aligned}$$

O nó que atinge o mínimo é o próximo salto no caminho mais curto → tabela de roteamento



Algoritmo de vetor de distâncias

∇ Idéia básica:

Cada nó envia periodicamente sua própria estimativa de vetor de distância aos vizinhos.

Quando o nó x recebe nova estimativa de DV do vizinho, ele atualiza seu próprio DV usando a equação B-F:

$$D_x(y) = \min_v \{c(x,v) + D_v(y)\}$$

para cada nó $y \in N$

∇ Em condições naturais, a estimativa $D_x(y)$ converge para o menor custo atual $d_x(y)$.

Tabelas de roteamento

$$D_x(y) = \min\{c(x,y) + D_y(y), c(x,z) + D_z(y)\} = \min\{2+0, 7+1\} = 2$$

$$D_x(z) = \min\{c(x,y) + D_y(z), c(x,z) + D_z(z)\} = \min\{2+1, 7+0\} = 3$$

		Custo até		
		x	y	z
De	x	0	2	7
	y	∞	∞	∞
	z	∞	∞	∞

		Custo até		
		x	y	z
De	x	0	2	3
	y	2	0	1
	z	7	1	0

		Custo até		
		x	y	z
De	x	0	2	3
	y	2	0	1
	z	3	1	0

Tabela do nó y

		Custo até		
		x	y	z
De	x	∞	∞	∞
	y	2	0	1
	z	∞	∞	∞

		Custo até		
		x	y	z
De	x	0	2	7
	y	2	0	1
	z	7	1	0

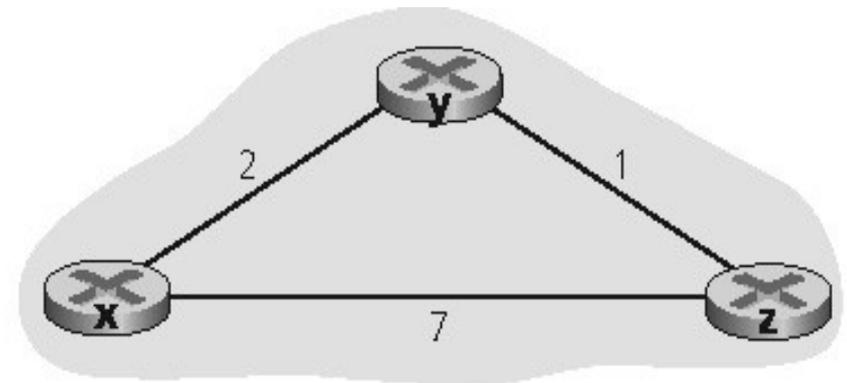
		Custo até		
		x	y	z
De	x	0	2	3
	y	2	0	1
	z	3	1	0

Tabela do nó z

		Custo até		
		x	y	z
De	x	∞	∞	∞
	y	∞	∞	∞
	z	7	1	0

		Custo até		
		x	y	z
De	x	0	2	7
	y	2	0	1
	z	3	1	0

		Custo até		
		x	y	z
De	x	0	2	3
	y	2	0	1
	z	3	1	0

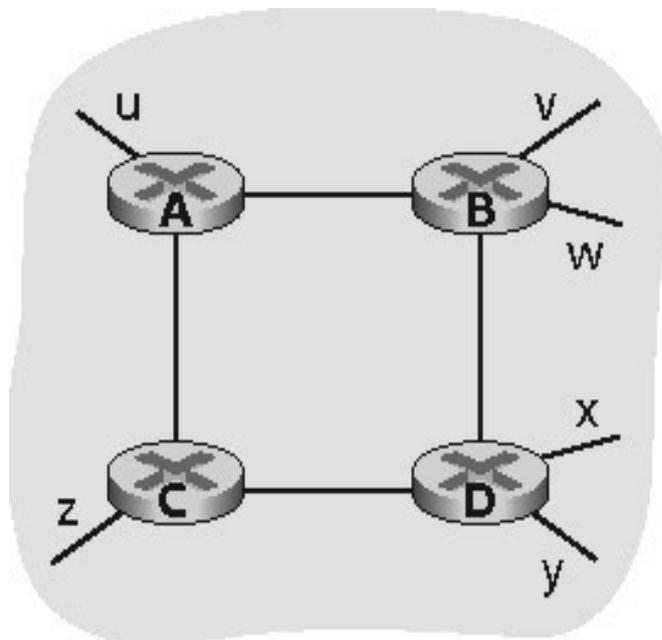




RIP (Routing Information Protocol)

RIP (Routing Information Protocol)

- ∇ Protocolo de vetor de distâncias.
- ∇ Distribuído no BSD-UNIX em 1982.
- ∇ Métrica de distância: # de saltos
 - versão 1: máx. = 15 saltos [RFC1058]
 - versão 2: otimizações [RFC1723]



Destino	Saltos
u	1
v	2
w	2
x	3
y	3
z	2



RIP: Anúncios

- ▽ Vetores de distância:
trocados a cada 30 seg via **Anúncio RIP**.
- ▽ Anúncio contém:
lista de até 25 redes de destino dentro do AS;
distâncias entre remetente e rede destino.
- ▽ Utiliza segmentos UDP para troca de mensagens.
porta 520.

RIP: Exemplo

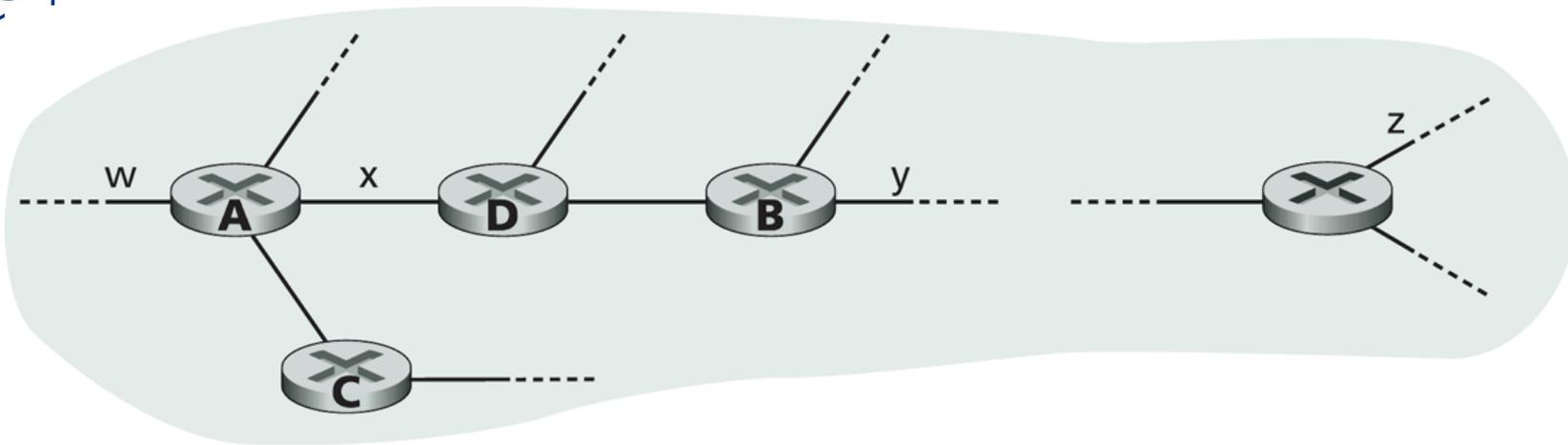


Figure 4.32 ♦ A portion of an autonomous system

Destination Subnet	Next Router	Number of Hops to Destination
w	A	2
y	B	2
z	B	7
x	—	1
....

Figure 4.33 ♦ Routing table in router *D* before receiving advertisement from router *A*



RIP: Exemplo (continuação)

Destination Subnet	Next Router	Number of Hops to Destination
z	C	4
w	—	1
x	—	1
....

Figure 4.34 ♦ Advertisement from router A

Destination Subnet	Next Router	Number of Hops to Destination
w	A	2
y	B	2
z	A	5
....

Figure 4.35 ♦ Routing table in router D after receiving advertisement from router A



RIP: Falhas de enlace e recuperação

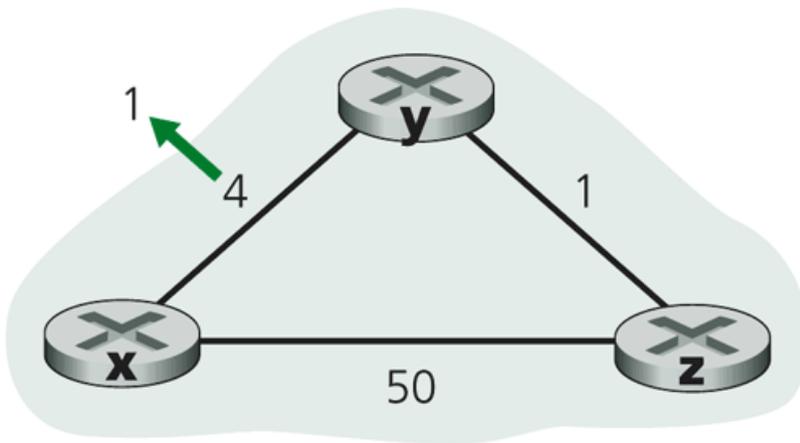
▽ Se não há um aviso depois de 180s, então o vizinho e o enlace são declarados mortos.

Rotas através do vizinho são anuladas;
novos anúncios são enviados aos vizinhos;
se necessário, os vizinhos por sua vez devem enviar novos anúncios;

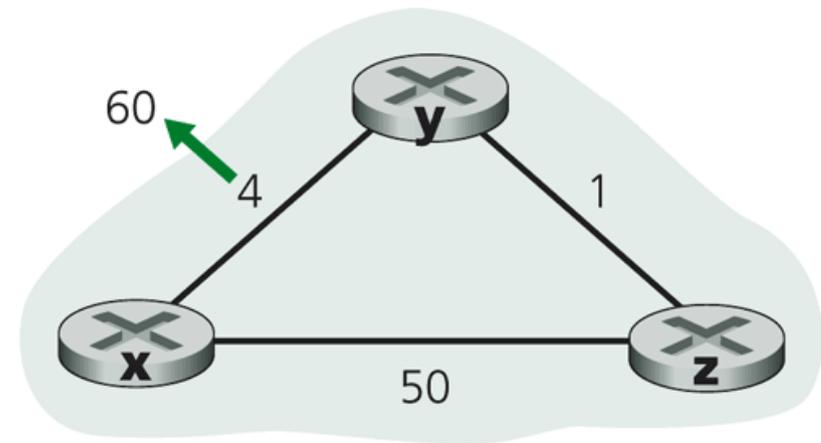
Assim, a falha de um enlace se propaga rapidamente para a rede inteira.

RIP: loops

- ∇ Roteador x utiliza rota via y para chegar a roteador z ($D=5$).
- ∇ O que acontece se o custo do enlace entre x e y aumenta?



a.



b.

Figure 4.28 ♦ Changes in link cost



RIP: loops (cont.)

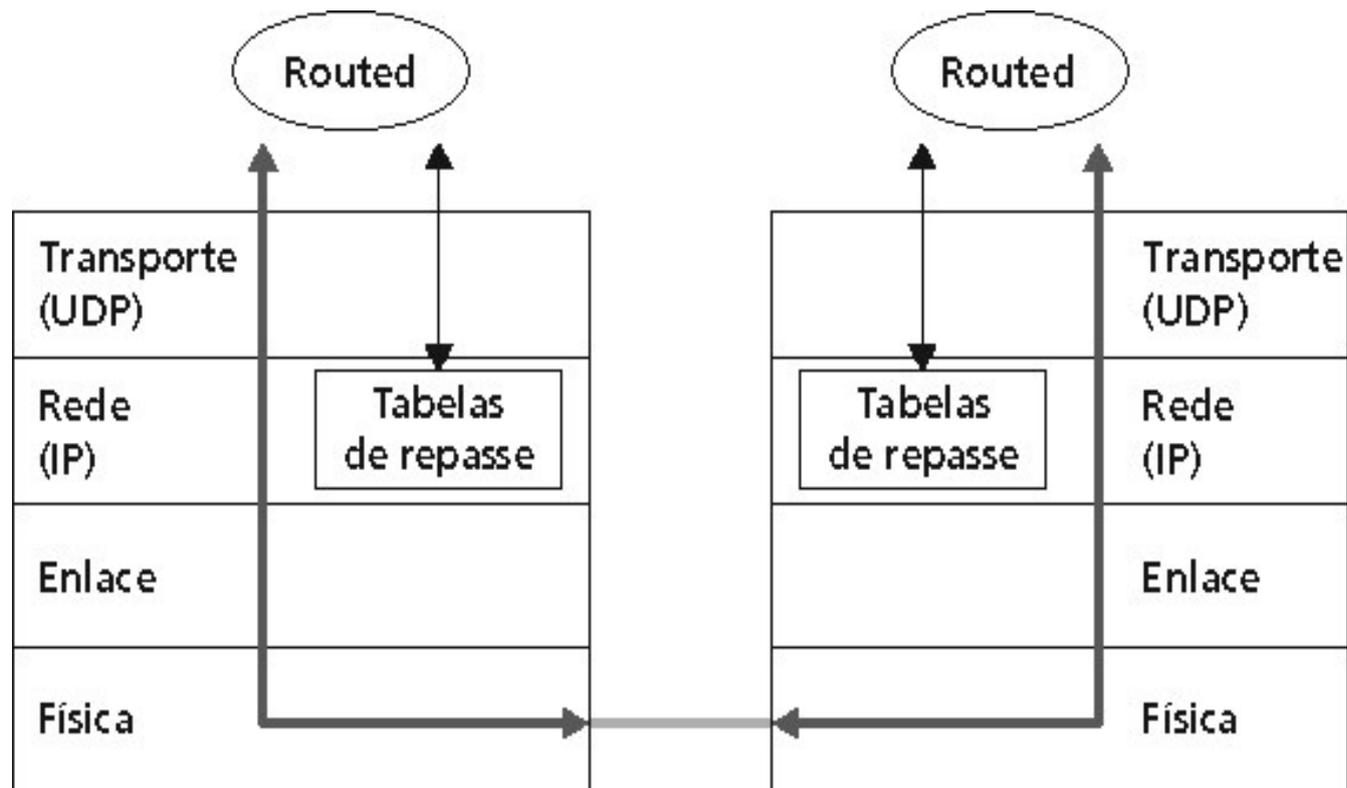
- ∇ Cria loop de roteamento entre y e z... até que o custo desta rota seja equivalente ao custo da rota alternativa (mais “barata” agora).
- ∇ Solução: “reversão envenenada” (*poisoned reverse*):
 - se a rota de z para x passa por y, então z anuncia distância infinita para y (16 saltos na versão 1);
 - porém não resolve todos os problemas de loops.



RIP: implementação em um gateway linux

Tabelas de roteamento: manipuladas por um *daemon* chamado route-d.

- Anúncios são enviados em pacotes UDP com repetição periódica.





Roteamento de estado de enlace (LS)



Roteamento de estado de enlace

- ∇ Algoritmo de Dijkstra
- ∇ Topologia de rede e custo dos enlaces são conhecidos por todos os nós:
 - implementado via “*link state broadcast*”;
 - todos os nós têm a mesma informação.
- ∇ Computa caminhos de menor custo de um nó (fonte) para todos os outros nós:
 - cria uma tabela de roteamento para o nó onde o algoritmo foi aplicado.
- ∇ Convergência: após k iterações, conhece o caminho de menor custo para k destinos.



Notação

- ∇ **$C(i,j)$** : custo do enlace do nó i ao nó j . Custo é infinito se não houver ligação entre i e j .
- ∇ **$D(v)$** : valor atual do custo do caminho da fonte ao destino v .
- ∇ **$p(v)$** : nó predecessor ao longo do caminho da fonte ao nó v .
- ∇ **N'** : conjunto de nós cujo caminho de menor custo é definitivamente conhecido.
- ∇ **u** : nó onde está sendo executado o algoritmo.



Algoritmo de Dijkstra

1 **Inicialização:**

2 $N' = \{u\}$

3 para todos os nós v

4 se v for vizinho de u

5 então $D(v) = c(u,v)$

6 senão $D(v) = \infty$

7

8 **Loop**

9 encontre w não em N' , tal que $D(w)$ é um mínimo

10 adicione w a N'

11 atualize $D(v)$ para cada vizinho v de w e não em N' :

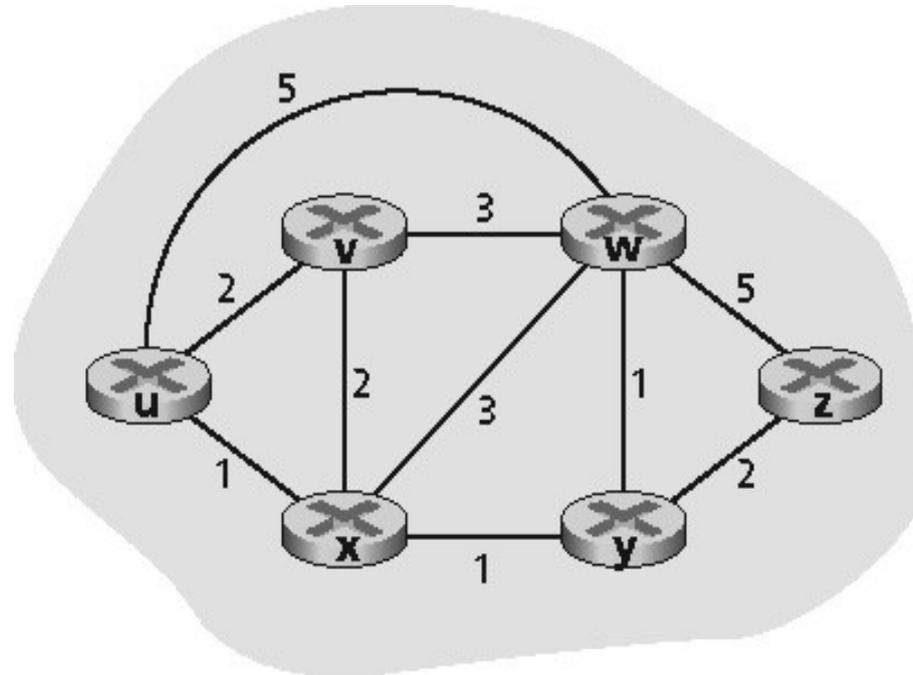
12 $D(v) = \min(D(v), D(w) + c(w,v))$

13 /* novo custo para v o custo anterior para v ou o menor

14 custo de caminho conhecido para w mais o custo de w a v */

15 **até que todos os nós estejam em N'**

Algoritmo de Dijkstra: Exemplo



Etapa	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	1,u	∞	∞
1	ux	2,u	4,x		2,x	∞
2	uxy	2,u	3,y			4,y
3	uxyv		3,y			4,y
4	uxyvw					4,y
5	uxyvwz					4,y



Complexidade do algoritmo

- ∇ Considere n nós.
- ∇ Cada iteração: precisa verificar todos os nós w , que não estão em N .
 $n(n+1)/2$ comparações: $O(n^2)$.



OSPF (Open Shortest Path First)



OSPF (Open Shortest Path First)

- ∇ Open: publicamente disponível[RFC2178]
- ∇ Protocolo de estado de enlace:
 - disseminação de pacotes LS;
 - mapa topológico em cada nó;
 - algoritmo de Dijkstra para cálculo de rotas.
- ∇ Anúncios do OSPF transportam um registro para cada roteador vizinho.



OSPF (cont.)

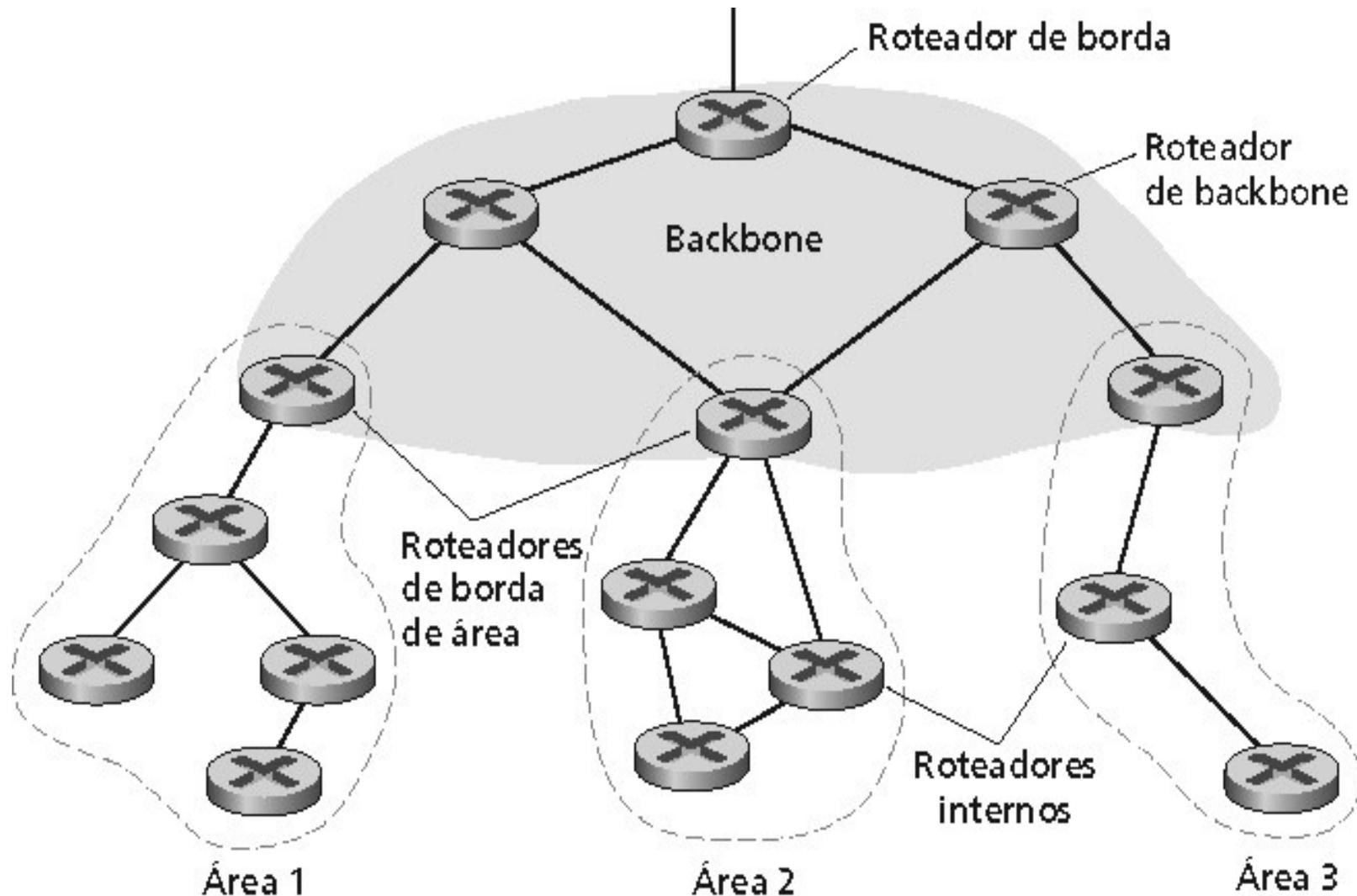
- ∇ Anúncios são distribuídos para **todo** o AS (via flooding):
 - mensagens OSPF diretamente sobre IP (protocolo de camada superior 89).
- ∇ Atualização periódica a cada 30 min (adiciona robustez).
- ∇ Mensagem de HELLO para verificar se enlaces estão ativos.
- ∇ Custo do enlace é definido pelo administrador do roteador.



OSPF: avanços

- ▽ Segurança: todas as mensagens do OSPF são autenticadas.
- ▽ Múltiplos caminhos de mesmo custo são permitidos (o RIP só permite um caminho).
- ▽ Integra tráfego uni- e multicast:
multicast OSPF (MOSPF) usa a mesma base de dados de topologia do OSPF.
- ▽ OSPF hierárquico: OSPF para grandes domínios.

OSPF: hierarquia em domínio roteamento





Comparação entre LS e DV

Robustez: o que acontece se um roteador funciona mal?

LS:

- ∇ nós podem informar custos de **link** incorretos;
- ∇ cada nó calcula sua própria tabela de roteamento.

DV:

- ∇ nó DV pode informar custo de **caminho** incorreto
- ∇ tabela de cada nó é usada por outros:
 - propagação de erros pela rede.



Roteamento Hierárquico



Roteamento Hierárquico

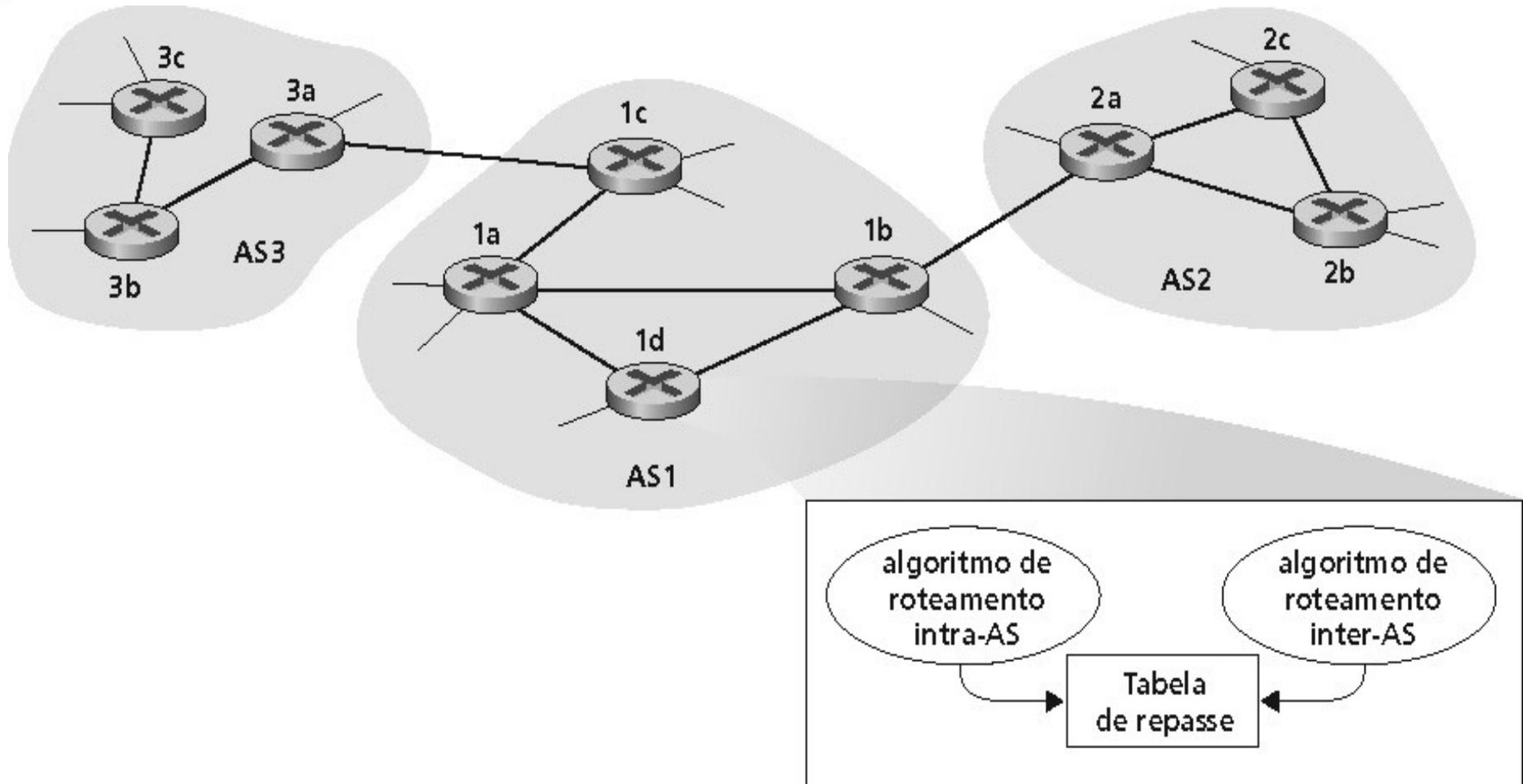
- Até agora, situação ideal e simplista:
 - roteadores são todos idênticos;
 - rede plana (*flat*).
- Não funciona porque:
 - **Escala:** com 200 milhões de destinos:
 - tamanho da tabela de rotas é intratável!
 - mudanças na tabela -> congestionamento!
 - **Autonomia administrativa:**
 - Internet = rede de redes.
 - Cada administração de rede pode querer controlar o roteamento na sua própria rede.



Roteamento Hierárquico

- ∇ Agrega roteadores em regiões: “**sistemas autônomos**” (AS).
- ∇ Roteadores no mesmo AS rodam o mesmo protocolo de roteamento:
 - protocolo de roteamento “**intra-AS**”
 - roteadores em diferentes AS podem rodar diferentes protocolos de roteamento.
- ∇ **Roteador de borda (*gateway router*):**
 - enlace direto para um roteador em outro AS.

Sistemas Autônomos Interconectados



- Tabela de roteamento é configurada por ambos os algoritmos, intra e inter-AS:

Intra-AS estabelece entradas para destinos internos;

Inter-AS e intra-As estabelecem entradas para destinos externos.



Roteamento Intra-AS

- Também conhecido como **Interior Gateway Protocols (IGP)**
- Protocolos mais comuns:
 - RIP: Routing Information Protocol;
 - OSPF: Open Shortest Path First;
 - IGRP: Interior Gateway Routing Protocol (proprietário da Cisco, baseado no DUAL).



Roteamento Inter-AS: BGP

- ∇ Roteamento inter-AS ou externo a AS.
- ∇ Padrão de fato para uso na Internet: BGP (Border Gateway Protocol).
- ∇ Versão 4: RFCs 1771, 1772 e 1773.



BGP (Border Gateway Protocol)

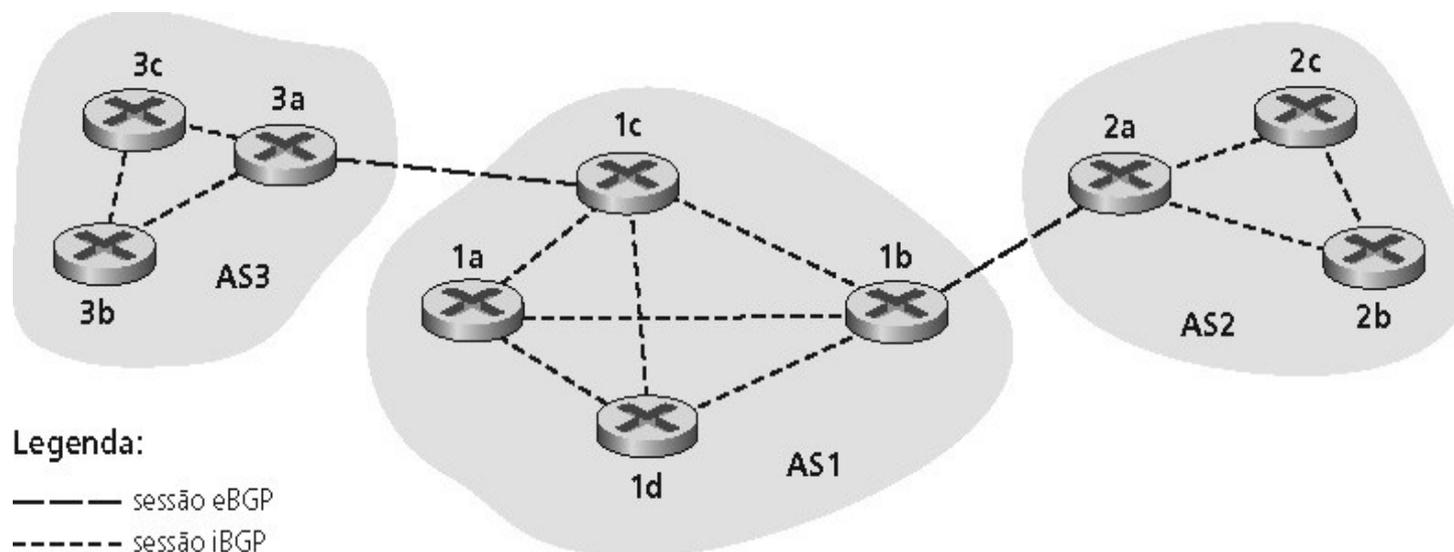


BGP

- ∇ Provê a cada AS meios para:
 1. obter informações de alcance de sub-rede dos ASs vizinhos;
 2. propagar informações de alcance para todos os roteadores internos ao AS;
 3. determinar rotas “boas” para as sub-redes baseado em informações de alcance e política.
- ∇ Permite que uma subnet comunique sua existência para o resto da Internet: **“Eu existo e estou aqui”**.

BGP: Conceitos Básicos

- ▽ Pares de roteadores trocam informações de roteamento por conexões TCP semipermanentes na porta 179.
- ▽ Sessões BGP não correspondem aos enlaces físicos!



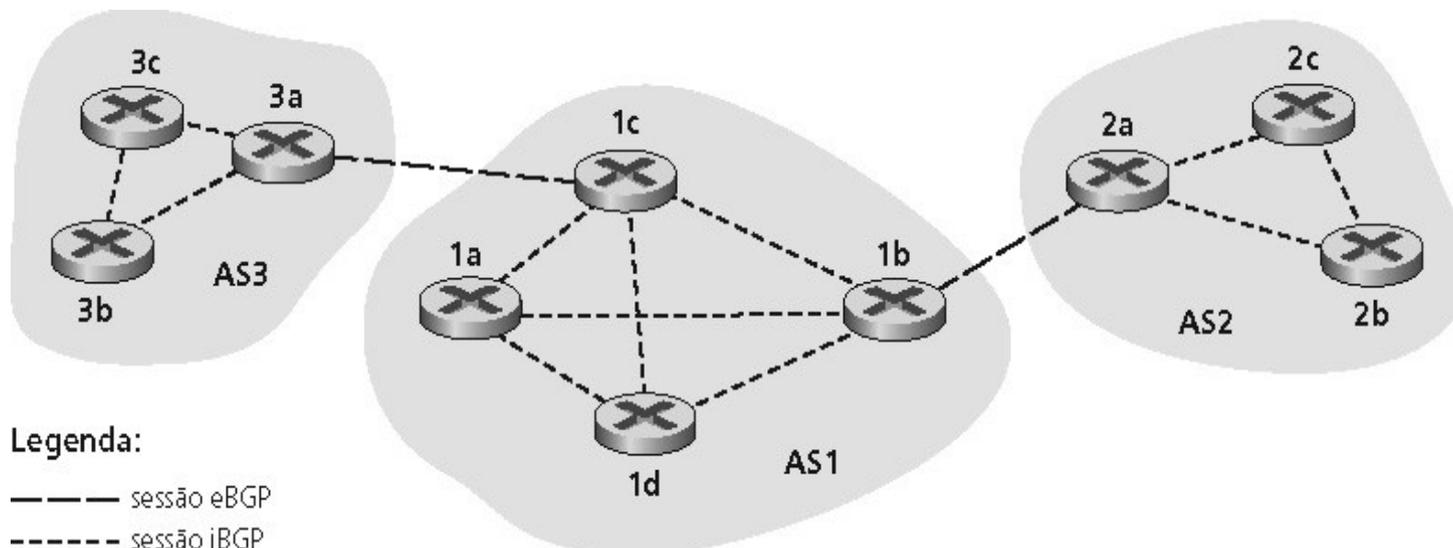


BGP: Conceitos Básicos

- ∇ No BGP, anúncios não tratam de hospedeiros, mas sim de prefixos.
- ∇ Quando AS2 comunica um prefixo ao AS1, AS2 está **prometendo** que encaminhará todos os datagramas destinados a esse prefixo em direção ao prefixo.
- ∇ AS2 pode agregar prefixos em seu comunicado.

Informações de alcance

- ∇ Em cada sessão eBGP entre 3a e 1c, AS3 envia informações de alcance de prefixo para AS1.
- ∇ 1c pode então usar iBGP para distribuir essa nova informação de alcance de prefixo para todos os roteadores em AS1.
- ∇ 1b pode recomunicar essa nova informação para AS2 por meio da sessão eBGP 1b-para-2a.
- ∇ Quando um roteador aprende um novo prefixo, ele cria uma entrada para o prefixo em sua tabela de roteamento.





Atributos de caminhos e roteadores BGP

- ASN: número do AS (registrado na ICANN)
- Anúncio inclui os atributos do BGP:
Prefixo + atributos = “rota”
- Dois atributos importantes:
AS-PATH: contém os ASs pelos quais o anúncio para o prefixo passou: AS 67 AS 17.
NEXT-HOP: indica o endereço IP da interface que leva ao roteador de borda (next-hop).
- Quando um roteador gateway recebe um comunicado de rota, ele usa política de importação para aceitar/rejeitar.



BGP: Seleção de Rota

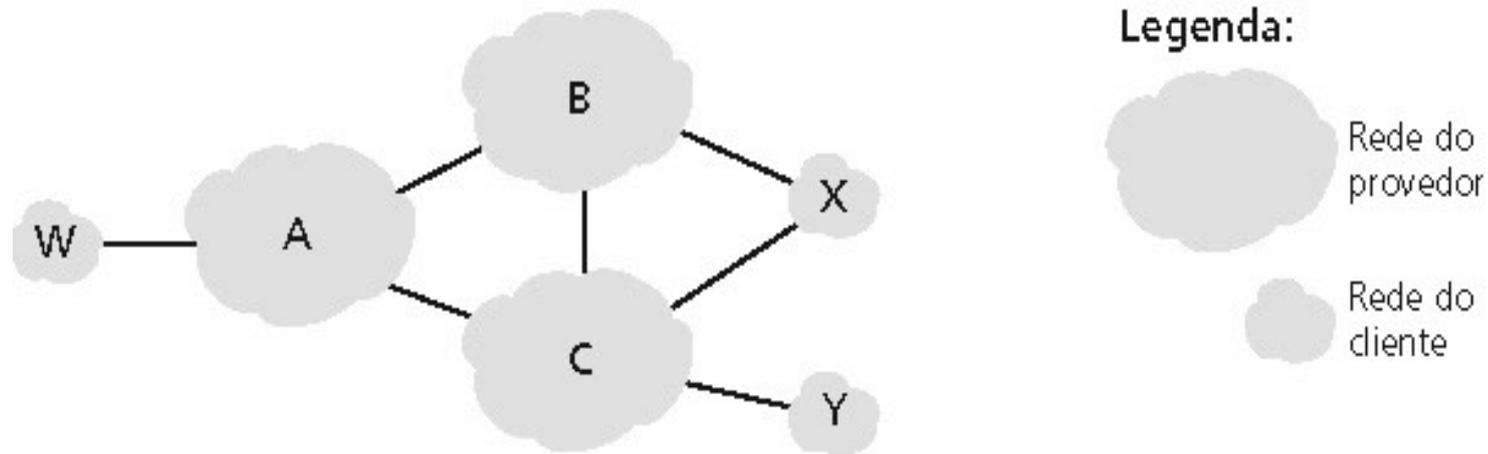
- ∇ Um roteador pode aprender mais do que 1 rota para o mesmo prefixo, e então deve selecionar uma rota.
- ∇ Regras de eliminação (em ordem):
 - atributo de valor de preferência local: decisão de política;
 - AS-PATH (caminho) mais curto;
 - roteador do NEXT-HOP (próximo salto) mais próximo: roteamento da “batata quente”;
 - identificadores BGP.



Mensagens BGP

- ∇ Utilizam TCP.
- ∇ Mensagens:
 - **OPEN**: abre conexão TCP para o par e autentica o transmissor;
 - **UPDATE**: comunica novo caminho (ou retira um antigo);
 - **KEEPALIVE** mantém a conexão ativa na ausência de atualizações (updates); e confirma requisição OPEN.
 - **NOTIFICATION**: reporta erros em mensagens anteriores; usado para fechar a conexão

BGP: Política de roteamento



- ∇ Rede stub: tráfego é destinado a ela e oriundo dela.
- ∇ X é uma rede stub com múltiplas interconexões:
 - logo X não deve rotear tráfego de B para C.
 - então X não anuncia a B uma rota para C!



Por que diferenças entre inter e intra-AS?

▽ Políticas:

Inter-AS: a administração quer ter controle sobre como seu tráfego é roteado e sobre quem roteia através da sua rede.

▽ Escalabilidade:

roteamento hierárquico poupa espaço da tabela de rotas e reduz o tráfego de atualização.

▽ Desempenho:

preocupação maior é desempenho em intra-AS.



Dúvidas?