# SSC0540
# Redes de Computadores

## Capítulo 4 - Camada de Rede

Prof. Jó Ueyama

Maio/2012

# Capítulo 4: Camada de rede

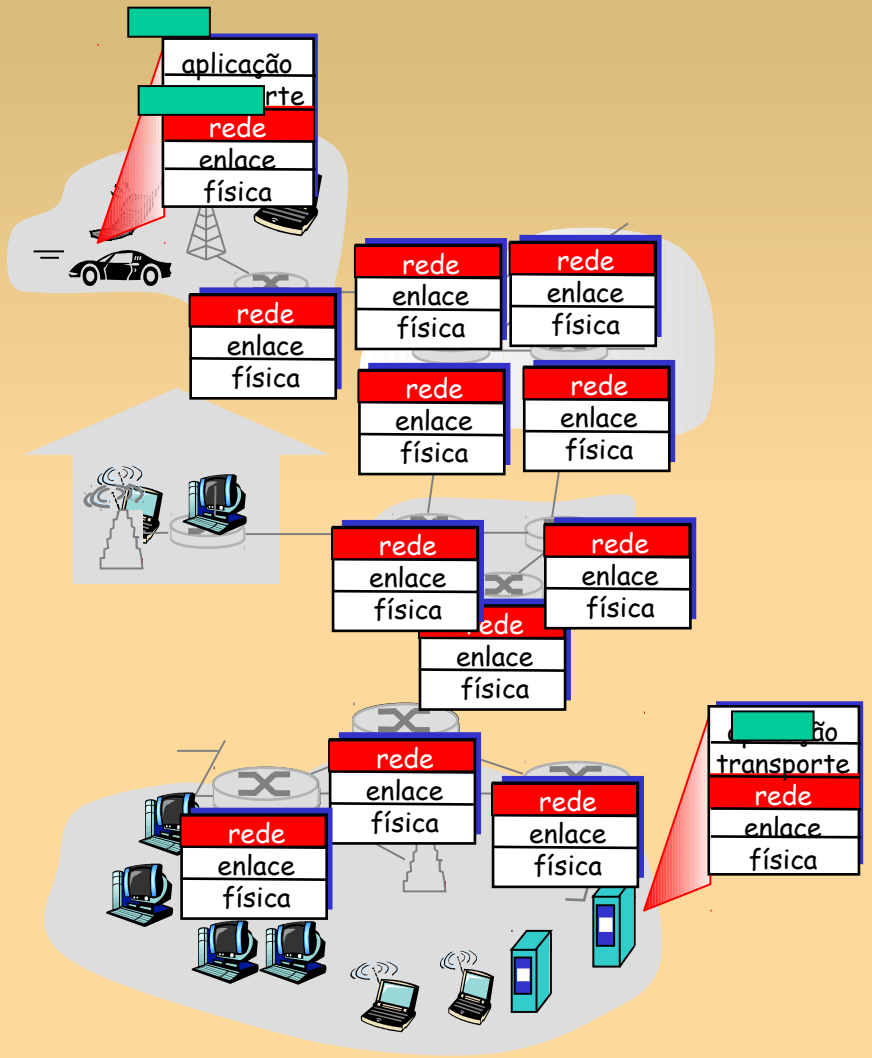
## Objetivos do capítulo:

- entender os princípios por trás dos serviços da camada de rede:
  - modelos de serviço da camada de rede
  - repasse *versus* roteamento
  - como funciona um roteador
  - roteamento (seleção de caminho)
  - lidando com escala
  - tópicos avançados: IPv6, mobilidade
- instanciação, implementação na Internet

- 4. 1 Introdução
- 4.2 Redes de circuitos virtuais e de datagramas
- 4.3 O que há dentro de um roteador?
- 4.4 IP: Internet Protocol
  - formato do datagrama
  - endereçamento IPv4
  - ICMP
  - IPv6
- 4.5 Algoritmos de roteamento
  - estado de enlace
  - vetor de distâncias
  - roteamento hierárquico
- 4.6 Roteamento na Internet
  - RIP
  - OSPF
  - BGP
- 4.7 Roteamento broadcast e multicast

# Camada de rede

- segmento de transporte do hosp. emissor ao receptor
- o lado emissor encapsula segmentos em datagramas
- o lado receptor entrega segmentos à camada de transporte
- protocolos da camada de rede em *cada* hosp., roteador
- roteador examina campos de cabeçalho em todos os datagramas IP que passam por ele



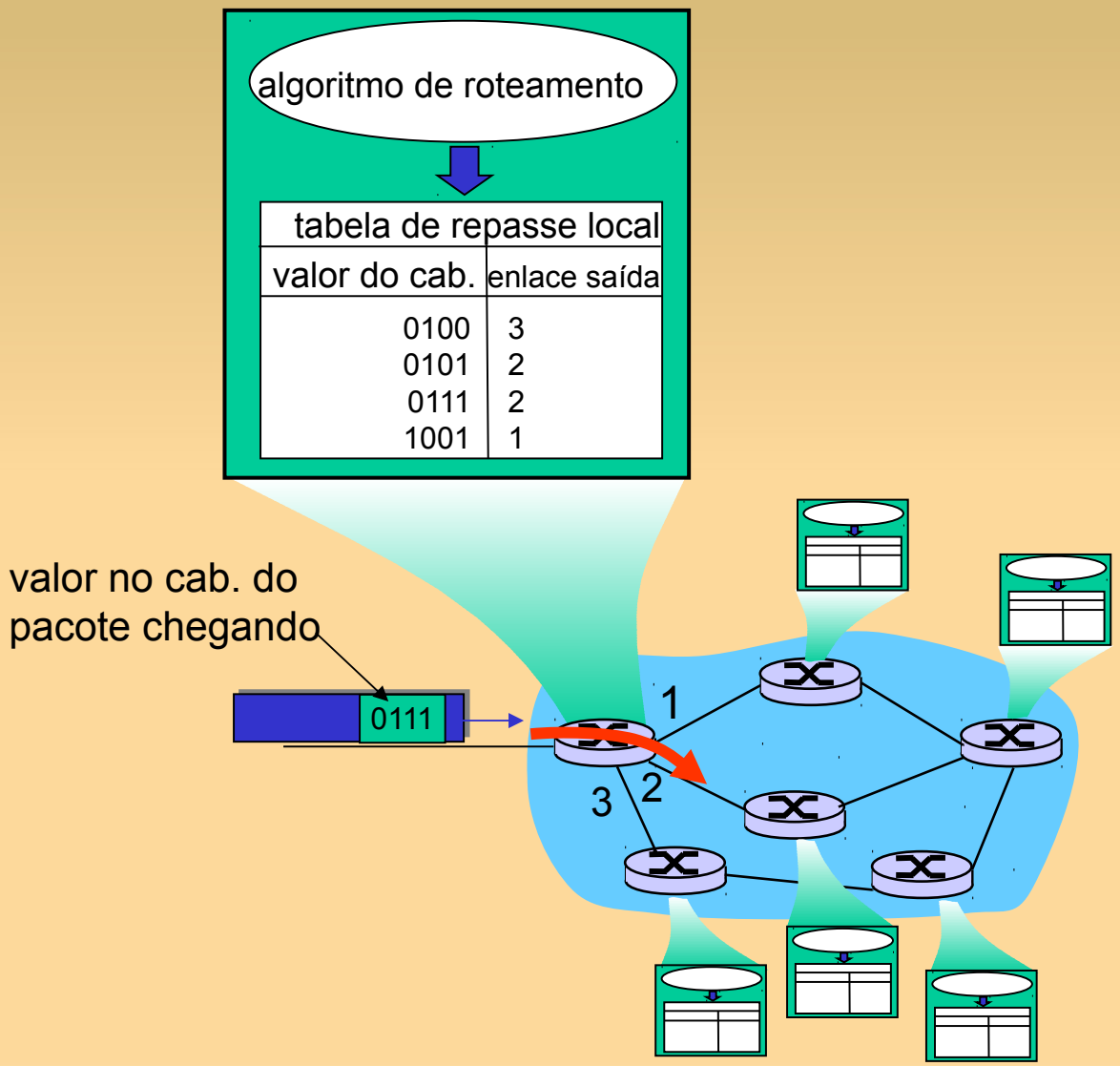
# Duas importantes funções da camada de rede

- *repasse:* mover pacotes da entrada do roteador para a saída apropriada do roteador
- *roteamento:* determinar rota seguida pelos pacotes da origem ao destino
  - *algoritmos de roteamento*

analogia:

- roteamento: processo de planejamento da viagem da origem ao destino
- repasse: processo de passar por um único cruzamento

# Interação entre roteamento e repasse



# Estabelecimento de conexão

- 3ª função importante em *algumas* arquiteturas de rede:
  - ATM, frame relay, X.25
- antes que os datagramas fluam, dois hospedeiros finais *e* roteadores entre eles estabelecem conexão virtual
  - roteadores são envolvidos
- serviço de conexão da camada de rede *versus* transporte:
  - rede: entre dois hospedeiros (também pode envolver roteadores entre eles, no caso de VCs)
  - transporte: entre dois processos

# Modelo de serviço de rede

P: Que *modelo de serviço* é o melhor para o "canal" que transporta datagramas do remetente ao destinatário?

exemplo de serviços para datagramas individuais:

- entrega garantida
- entrega garantida com atraso limitado

exemplo de serviços para fluxo de datagramas:

- entrega de datagrama na ordem
- largura de banda mínima garantida
- restrições sobre mudanças no espaçamento entre pacotes

# Modelos de serviço da camada de rede:

| Arquitetura da rede | Modelo de serviço | Garantia de largura de banda | Garantia contra perda | Ordenamento | Temporização | Indicação de congestionamento |
|---|---|---|---|---|---|---|
| Internet | Melhor esforço | Nenhuma | Nenhuma | Qualquer ordem possível | Não mantida | Nenhuma |
| ATM | CBR | Taxa constante garantida | Sim | Na ordem | Mantida | Não ocorrerá congestionamento |
| ATM | ABR | Mínima garantida | Nenhuma | Na ordem | Não mantida | Indicação de congestionamento |

# Capítulo 4: Camada de rede

- 4. 1 Introdução
- 4.2 Redes de circuitos virtuais e de datagramas
- 4.3 O que há dentro de um roteador?
- 4.4 IP: Internet Protocol
  - formato do datagrama
  - endereçamento IPv4
  - ICMP
  - IPv6
- 4.5 Algoritmos de roteamento
  - estado de enlace
  - vetor de distâncias
  - roteamento hierárquico
- 4.6 Roteamento na Internet
  - RIP
  - OSPF
  - BGP
- 4.7 Roteamento broadcast e multicast

# Serviço com e sem conexão da camada de rede

- rede de datagrama fornece serviço sem conexão da camada de rede
- rede VC fornece serviço com conexão da camada de rede
- análogo aos serviços da camada de transporte, mas:
  - serviço: hospedeiro a hospedeiro
  - sem escolha: a rede oferece um ou outro
  - implementação: no núcleo da rede

# Circuitos virtuais

"Caminho da origem ao destino comporta-se como um circuito telefônico"

- com respeito ao desempenho
- ações da rede ao longo do caminho da origem ao destino

- estabelecimento e término para cada chamada *antes* que os dados possam fluir
- cada pacote carrega identificador VC (não endereço do hospedeiro de destino)
- *cada* roteador no caminho origem-destino mantém "estado" para cada conexão que estiver passando
- recursos do enlace e roteador (largura de banda, buffers) podem ser *alocados* ao VC (recursos dedicados = serviço previsível)

# Implementação do VC

Um VC consiste em:

1. caminho da origem ao destino
2. números de VC, um número para cada enlace ao longo do caminho
3. entradas em tabelas de repasse nos roteadores ao longo do caminho

- pacote pertencente ao VC carrega número do VC (em vez do endereço de destino)
- número do VC pode ser alterado em cada enlace
  - novo número de VC vem da tabela de repasse

# Tabela de repasse

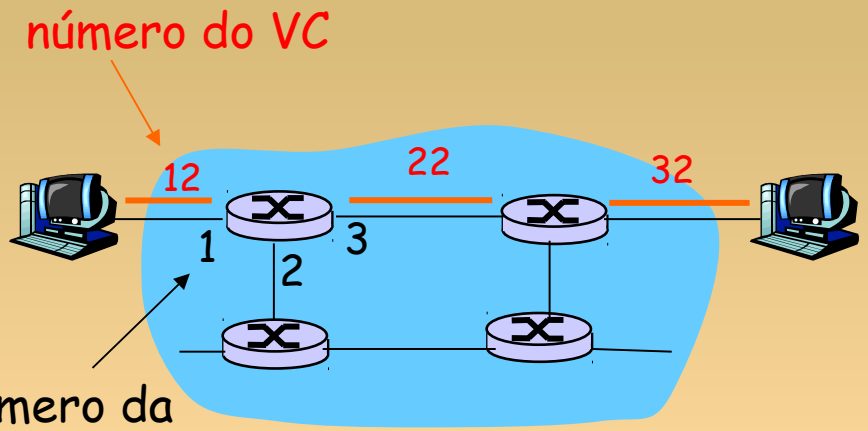


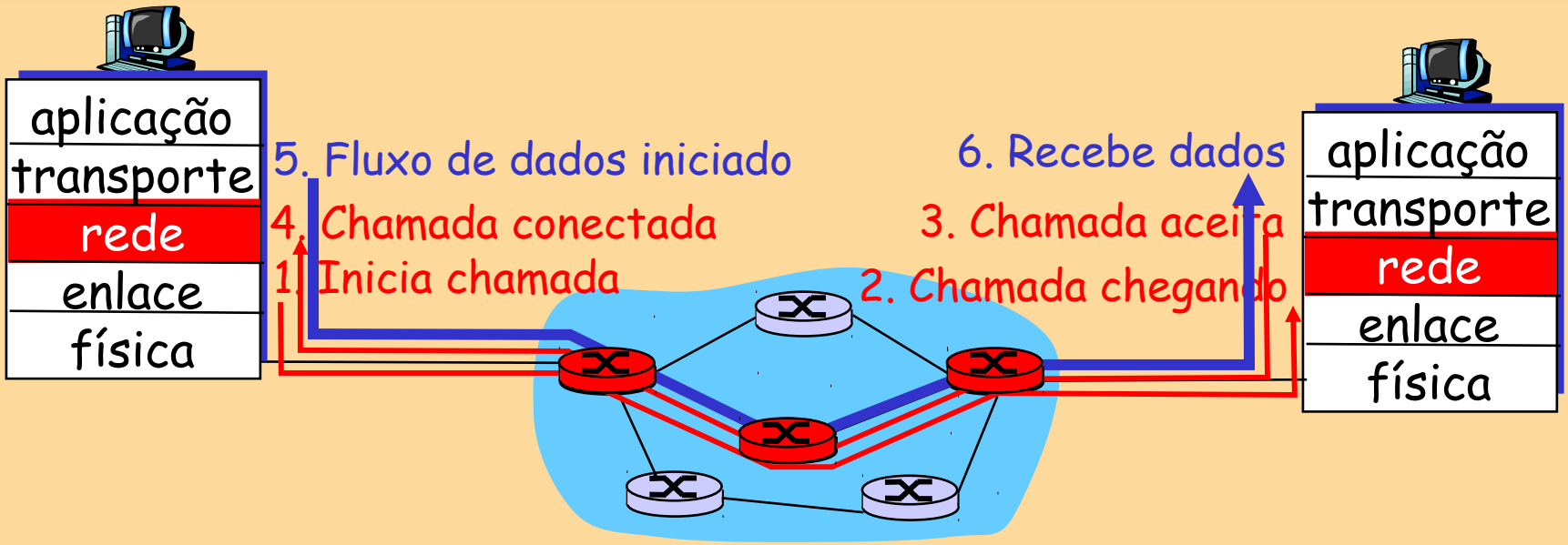

tabela de repasse no roteador noroeste:

| Interface de entrada | Nº do CV de entrada | Interface de saída | Nº do CV de saída |
|---|---|---|---|
| 1 | 12 | 2 | 22 |
| 2 | 63 | 1 | 18 |
| 3 | 7 | 2 | 17 |
| 1 | 97 | 3 | 87 |
| ... | ... | ... | ... |

Roteadores mantêm informação de estado da conexão!

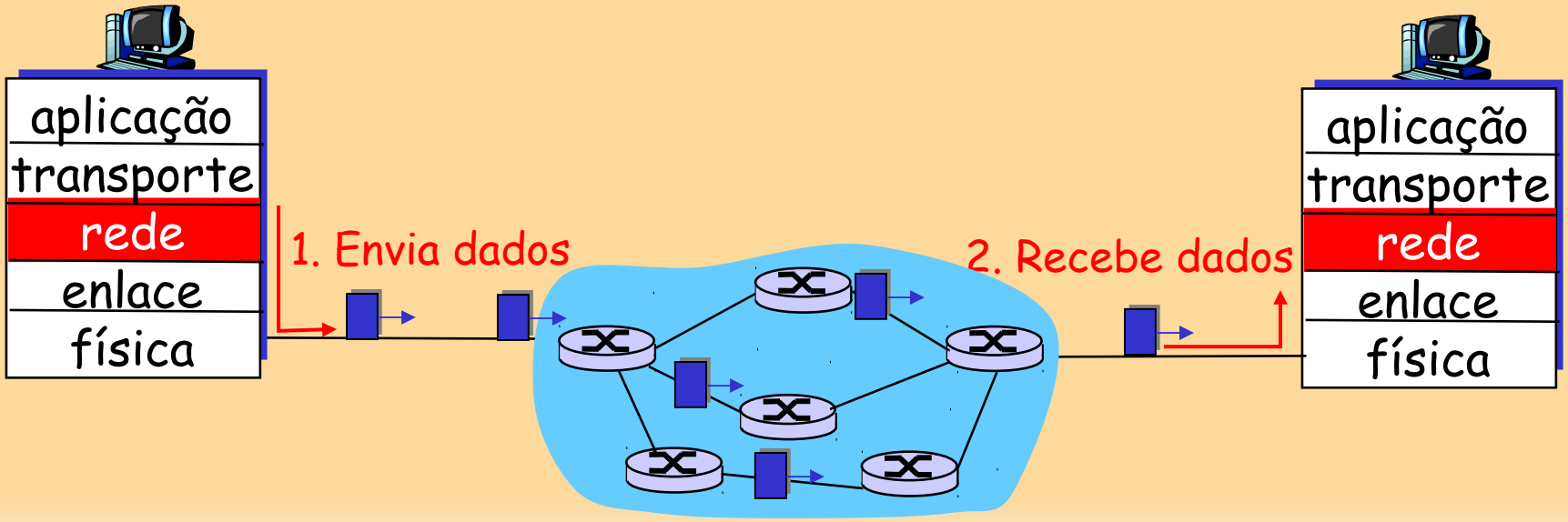
# Circuitos virtuais: protocolos de sinalização

- usados para estabelecer, manter e terminar VC
- usados em ATM, frame-relay, X.25
- não usados na Internet de hoje



# Redes de datagrama

- sem estabelecimento de chamada na camada de rede
- roteadores: sem estado sobre conexões fim a fim
  - sem conceito em nível de rede da “conexão”
- pacotes repassados usando endereço do hospedeiro de destino
  - pacotes entre mesmo par origem-destino podem tomar caminhos diferentes



# Tabela de repasse

4 bilhões de entradas possíveis

| Faixa de endereços de destino | Interface de enlace |
|---|---|
| 11001000 00010111 00010000 00000000<br>até<br>11001000 00010111 00010111 11111111 | 0 |
| 11001000 00010111 00011000 00000000<br>até<br>11001000 00010111 00011000 11111111 | 1 |
| 11001000 00010111 00011001 00000000<br>até<br>11001000 00010111 00011111 11111111 | 2 |
| senão | 3 |

# Concordância do prefixo mais longo

| Concordância do pref xo | Interface do enlace |
|---|---|
| 11001000 00010111 00010 | 0 |
| 11001000 00010111 00011000 | 1 |
| 11001000 00010111 00011 | 2 |
| senão | 3 |

Exemplos: 0-1

DA: 11001000 00010111 00010110 10100001 Qual interface?

DA: 11001000 00010111 00011000 10101010 Qual interface?

# Rede de datagramas ou VC: por quê?

## Internet (datagrama)

- troca de dados entre computadores
  - serviço “elástico”, sem requisitos de temporização estritos
- sistemas finais “inteligentes” (computadores)
  - pode adaptar, realizar controle, recup. de erros
  - simples dentro da rede, complexidade na “borda”
- muitos tipos de enlace
  - diferentes características
  - serviço uniforme difícil

## ATM (VC)

- evoluída da telefonia
- conversação humana:
  - requisitos de temporização estritos, confiabilidade
  - necessário para serviço garantido
- sistemas finais “burros”
  - telefones
  - complexidade dentro da rede

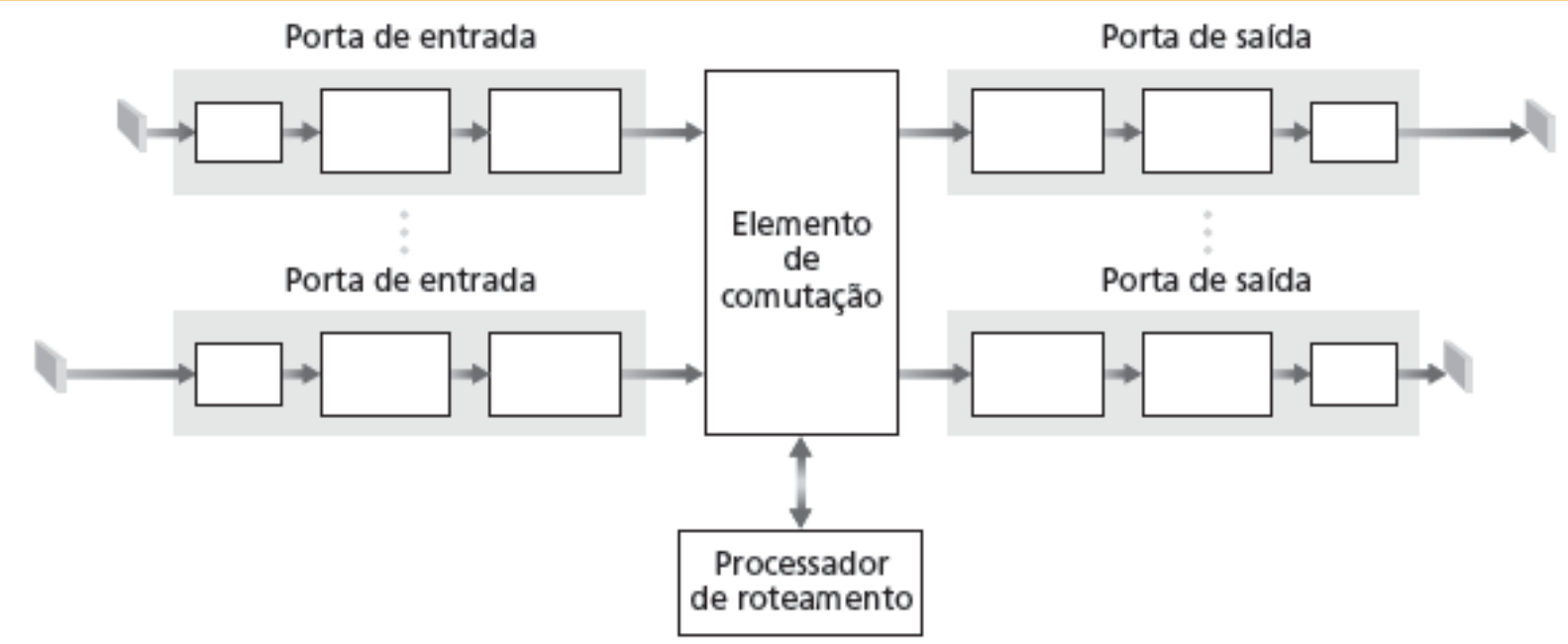
# Capítulo 4: Camada de rede

- 4. 1 Introdução
- 4.2 Redes de circuitos virtuais e de datagramas
- 4.3 O que há dentro de um roteador?
- 4.4 IP: Internet Protocol
  - formato do datagrama
  - endereçamento IPv4
  - ICMP
  - IPv6
- 4.5 Algoritmos de roteamento
  - estado de enlace
  - vetor de distâncias
  - roteamento hierárquico
- 4.6 Roteamento na Internet
  - RIP
  - OSPF
  - BGP
- 4.7 Roteamento broadcast e multicast

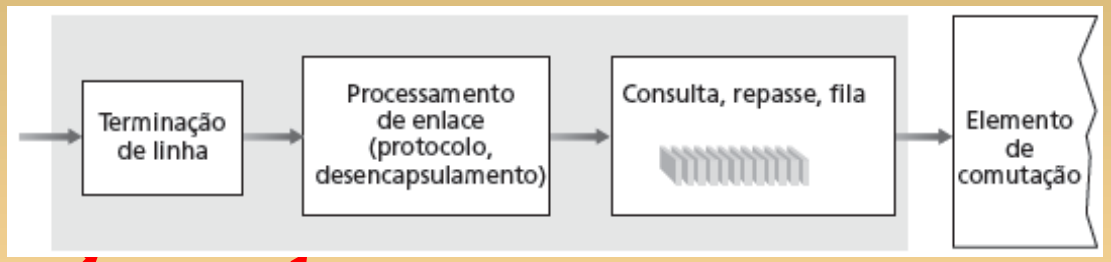
# Visão geral da arquitetura do roteador

Duas funções principais do roteador:

- executar algoritmos/protocolo de roteamento (RIP, OSPF, BGP)
- *repassar* datagramas do enlace de entrada para saída



# Funções da porta de entrada



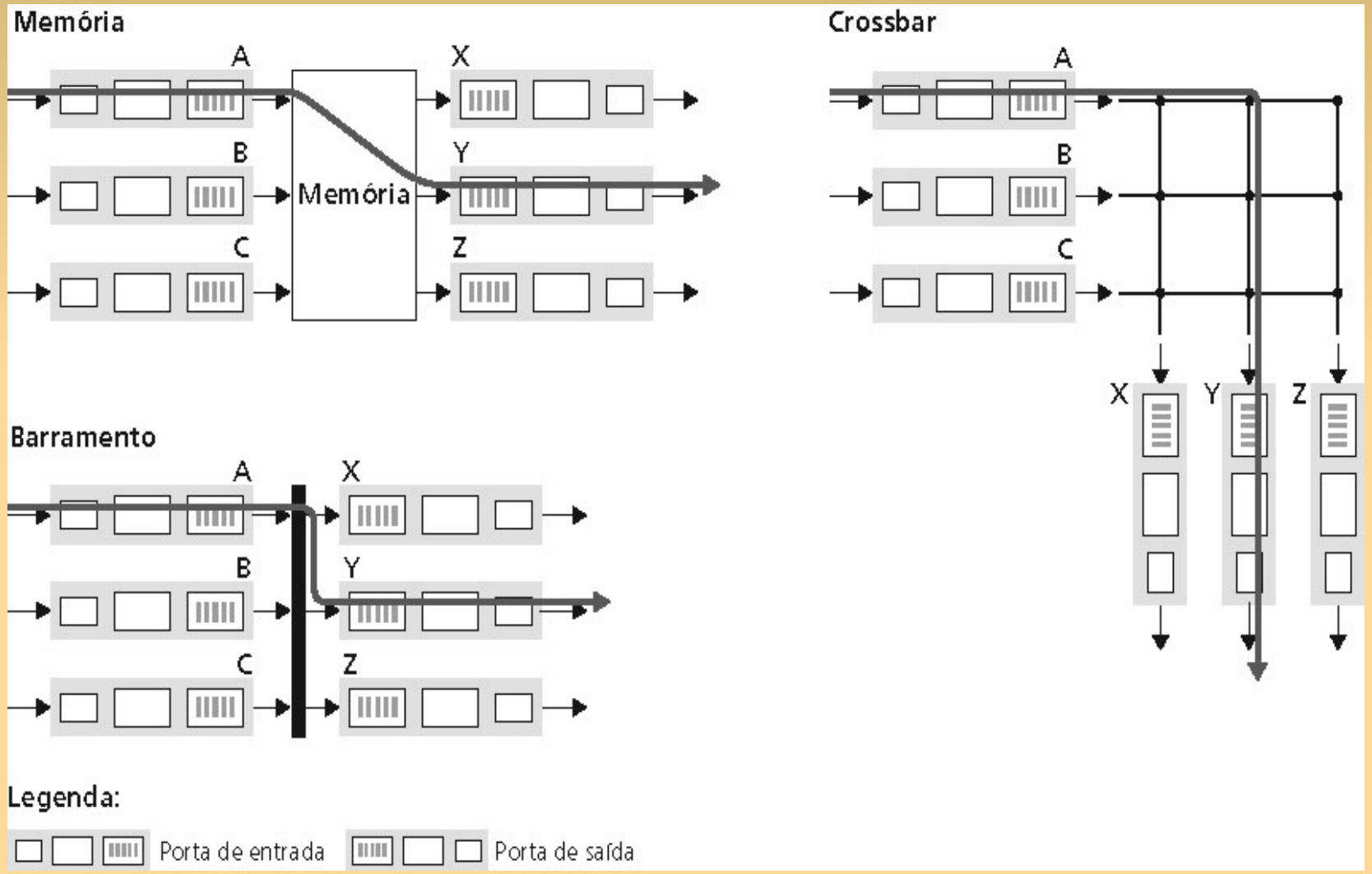

Camada física:
recepção por bit

Camada de enlace de dados:
p. e., Ethernet
ver Capítulo 5

**Comutação descentralizada**:

- dado destino do datagrama, porta de saída é pesquisada usando tabela de repasse
- objetivo: processamento completo da porta de entrada na 'velocidade de linha'
- fila: se datagramas chegarem mais rápido que taxa de repasse no elemento de comutação

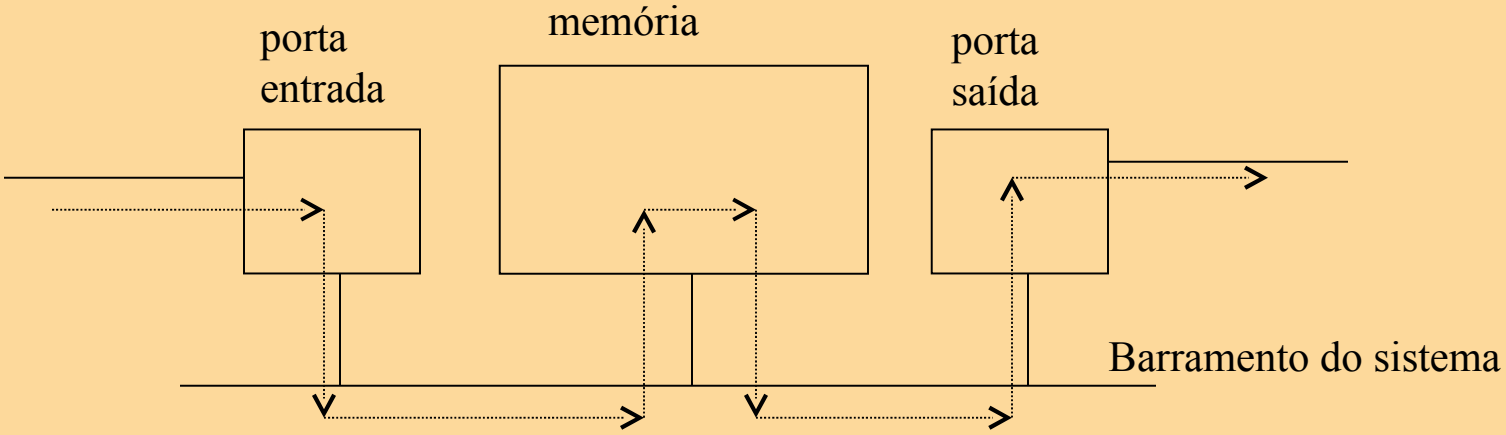
# Três tipos de Estrutura de Comutação



# Comutação por memória

Roteadores de primeira geração:

- computadores tradicionais com a comutação via controle direto da CPU
- pacote copiado para a memória do sistema
- velocidade limitada pela largura de banda da memória (2 travessias de barramento por datagrama)



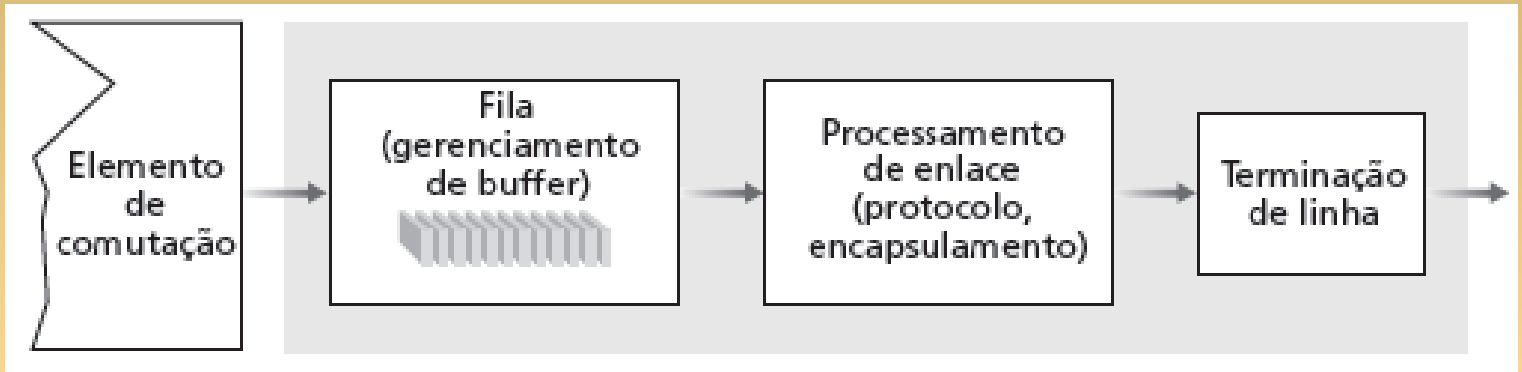
# Comutação por um barramento

- datagrama da memória da porta de entrada à memória da porta de saída por um barramento compartilhado
- disputa pelo barramento: velocidade da comutação limitada pela largura de banda do barramento
- barramento Cisco 5600 de 32 Gbps: velocidade suficiente para roteadores de acesso e corporativos

# Comutação por uma rede de interconexão

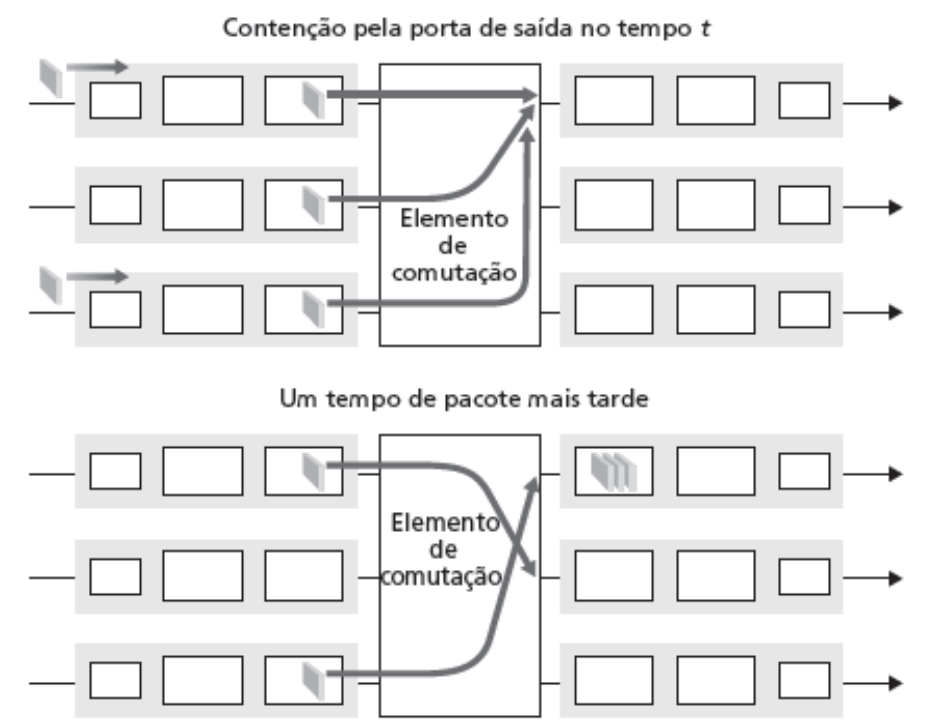
- contorna limitações de largura de banda do barramento
- Outras redes de interconexão desenvolvidas inicialmente para conectar processadores no multiprocessador
- projeto avançado: fragmenta datagrama em células de tamanho fixo, comuta células através do elemento de comutação
- Cisco 12000: comuta 60 Gbps através da rede de interconexão

# Portas de saída




- *Buffering* exigido quando os datagramas chegam do elemento de comutação mais rápido que a taxa de transmissão
- *Disciplina de escalonamento* escolhe entre os datagramas enfileirados para transmissão

# Enfileiramento na porta de saída



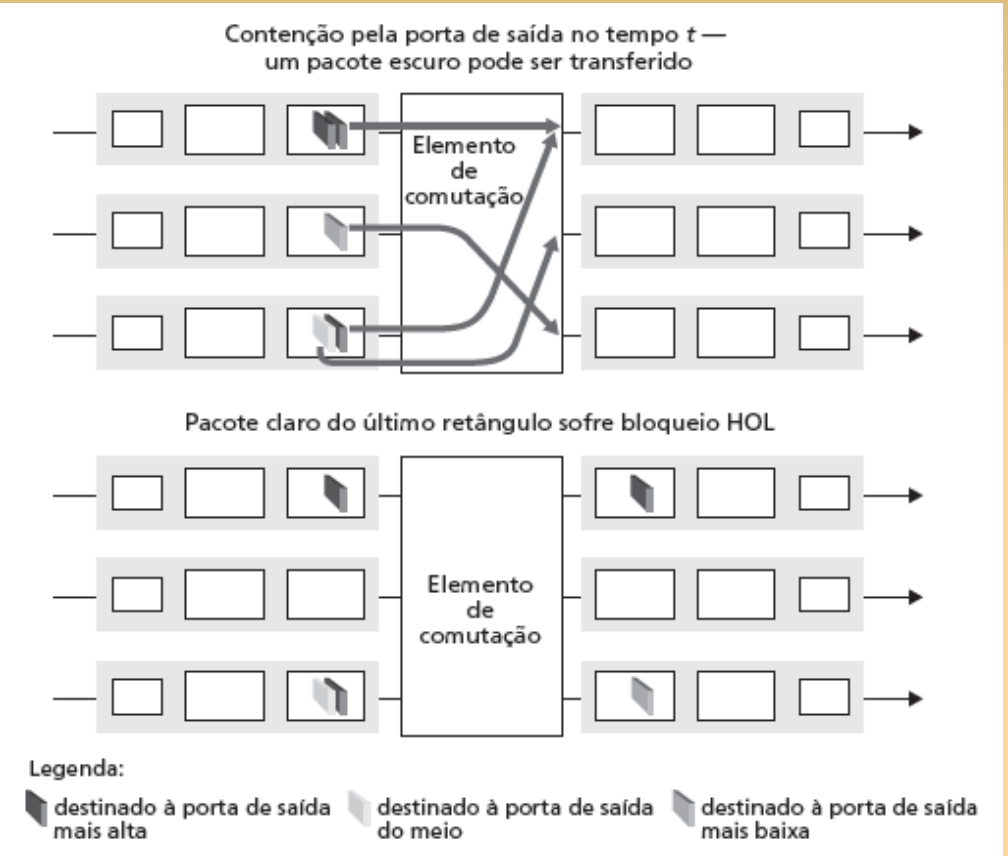

- buffering quando a taxa de chegada via comutador excede a velocidade da linha de saída
- *enfileiramento (atraso) e perda devidos a estouro de buffer na porta de saída!*

# Quanto armazenamento em buffer?

- regra prática da RFC 3439: armazenamento médio em buffer igual à RTT “típica” (digamos, 250 ms) vezes capacidade do enlace C
  - p. e., C = enlace de 10 Gps: buffer de 2,5 Gbit
- recomendação recente: com *N* fluxos, armazenamento deve ser igual a $\frac{RTT \cdot C}{\sqrt{N}}$

# Enfileiramento da porta de entrada

- elemento de comutação mais lento que portas de entrada combinadas -> enfileiramento possível nas filas de entrada
- bloqueio de cabeça de fila (HOL) : datagrama enfileirado na frente da fila impede que outros na fila sigam adiante
- *atraso de enfileiramento e perda devidos a estouro no buffer de entrada*

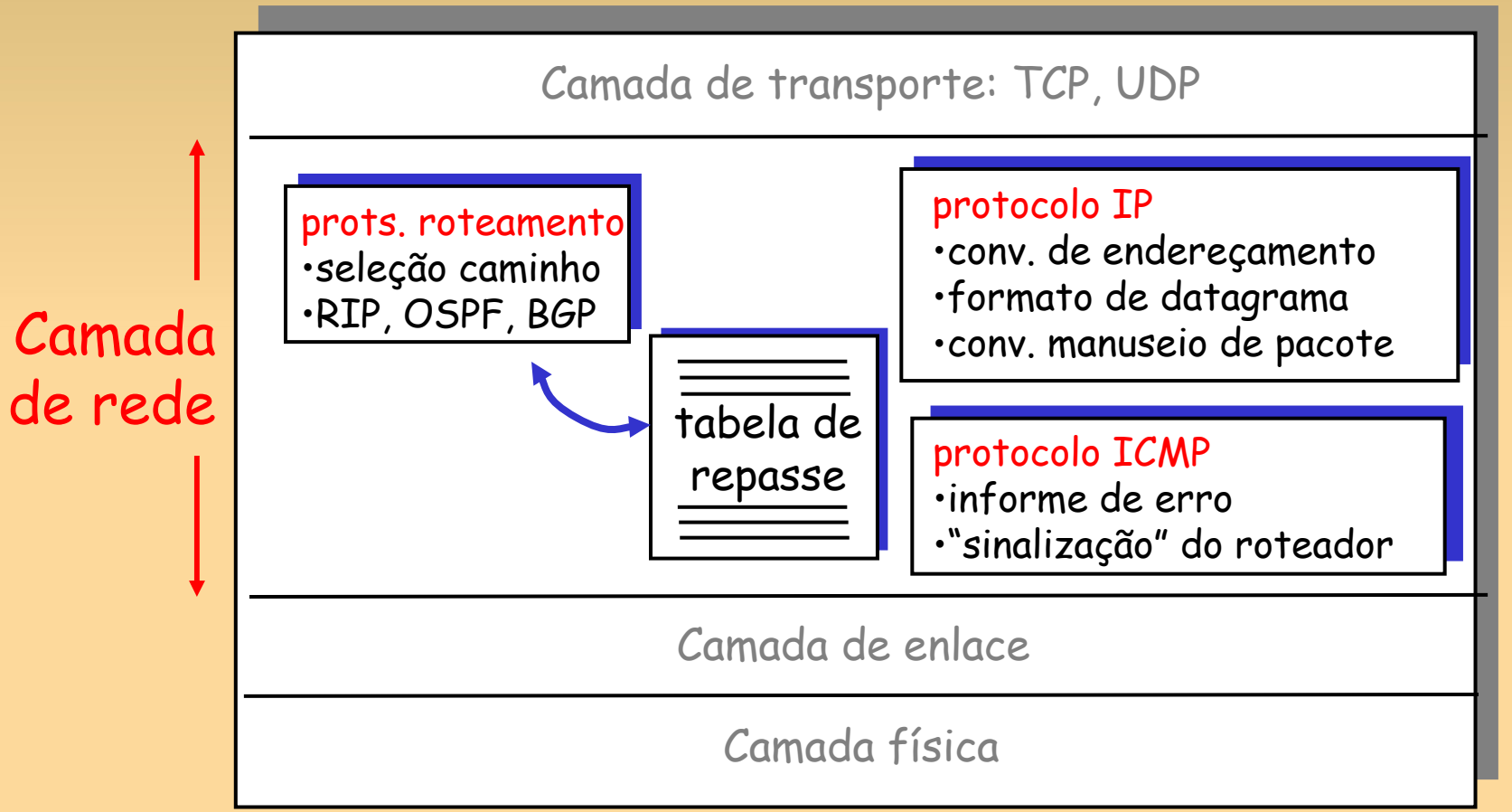


# Capítulo 4: Camada de rede

- 4.1 Introdução
- 4.2 Redes de circuitos virtuais e de datagramas
- 4.3 O que há dentro de um roteador?
- 4.4 IP: Internet Protocol
  - formato do datagrama
  - endereçamento IPv4
  - ICMP
  - IPv6
- 4.5 Algoritmos de roteamento
  - estado de enlace
  - vetor de distâncias
  - roteamento hierárquico
- 4.6 Roteamento na Internet
  - RIP
  - OSPF
  - BGP
- 4.7 Roteamento broadcast e multicast

# A camada de rede da Internet

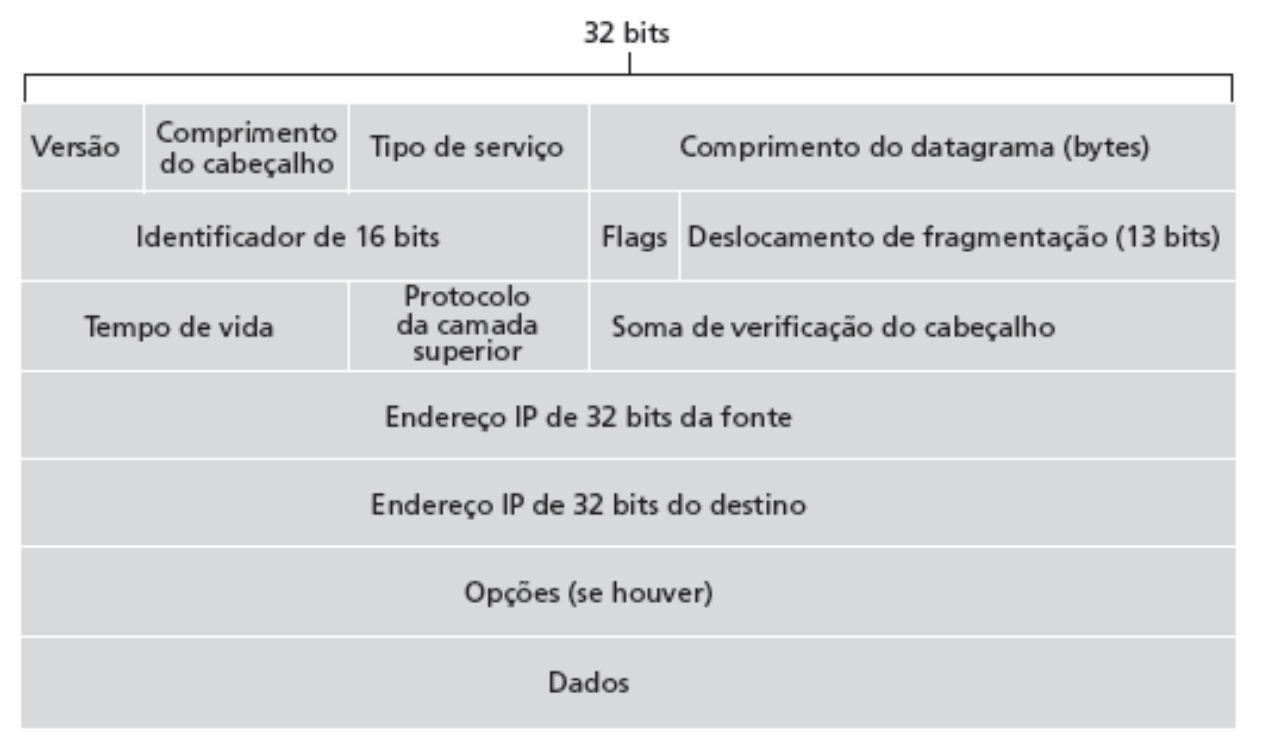
Funções na camada de rede do hospedeiro e roteador:



# Capítulo 4: Camada de rede

- 4. 1 Introdução
- 4.2 Redes de circuitos virtuais e de datagramas
- 4.3 O que há dentro de um roteador?
- 4.4 IP: Internet Protocol
  - formato do datagrama
  - endereçamento IPv4
  - ICMP
  - IPv6
- 4.5 Algoritmos de roteamento
  - estado de enlace
  - vetor de distâncias
  - roteamento hierárquico
- 4.6 Roteamento na Internet
  - RIP
  - OSPF
  - BGP
- 4.7 Roteamento broadcast e multicast

# Formato do datagrama IP

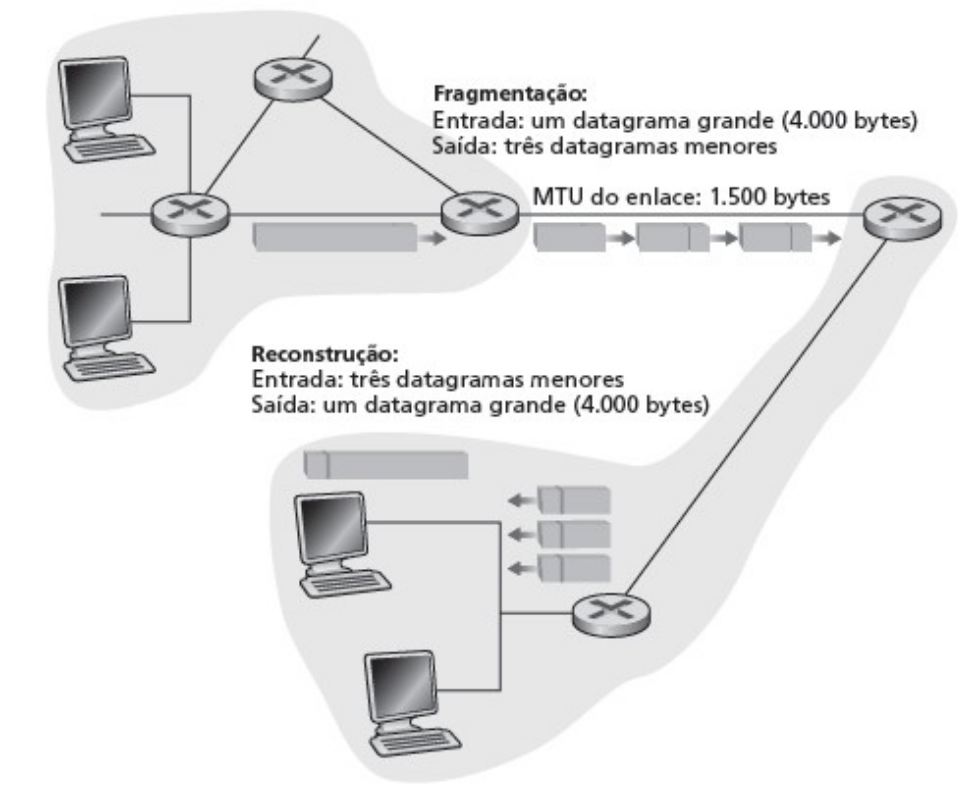



Quanto overhead com TCP?

- 20 bytes de TCP
- 20 bytes de IP
- = 40 bytes + overhead da camada de aplicação

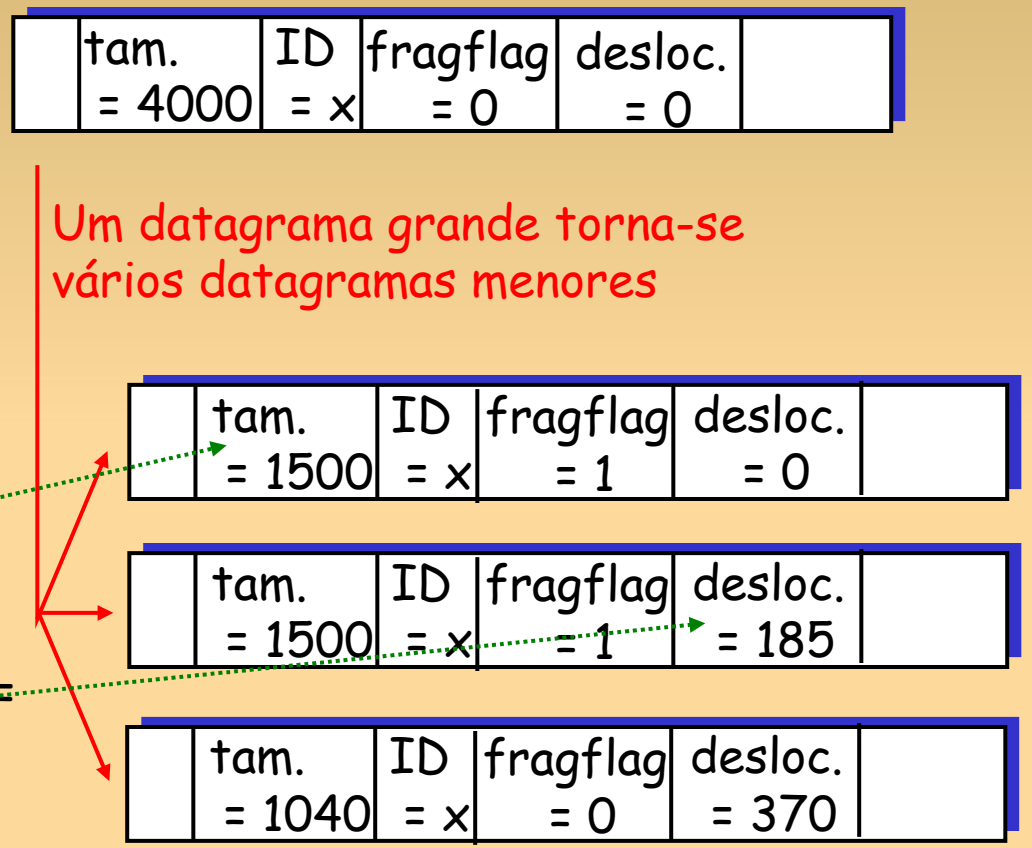
# Fragmentação e reconstrução do IP

- enlaces de rede têm MTU (tamanho máx. transferência) – maior quadro em nível de enlace possível.
  - diferentes tipos de enlace, diferentes MTUs
- grande datagrama IP dividido ("fragmentado") dentro da rede
  - um datagrama torna-se vários datagramas
  - "reconstruído" somente no destino final
  - bits de cabeçalho IP usados para identificar, ordenar fragmentos relacionados



## Exemplo

- datagrama de 4000 bytes
- MTU = 1500 bytes

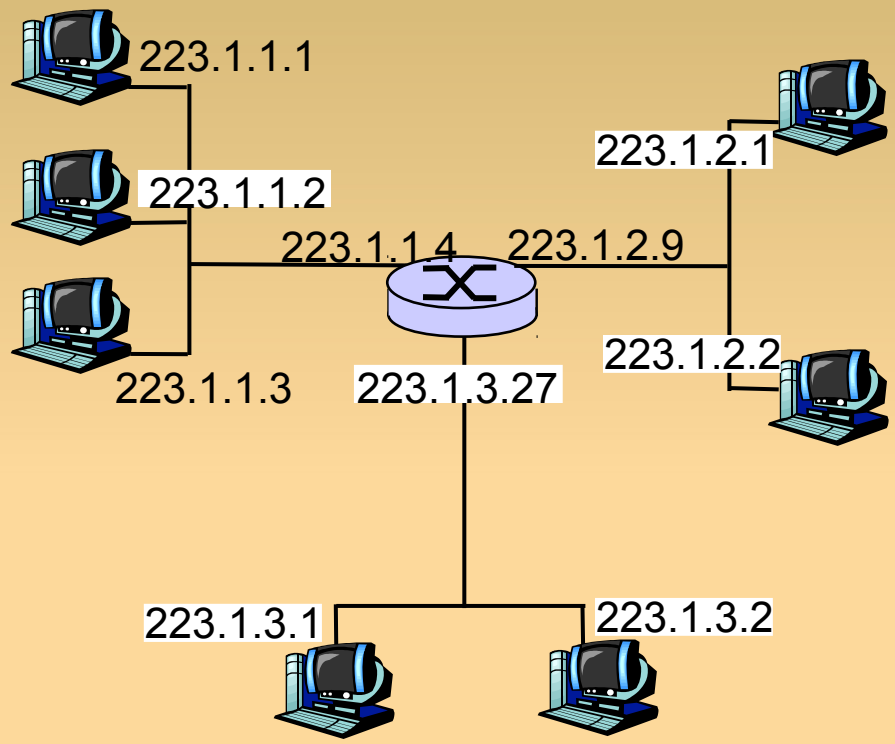


# Capítulo 4: Camada de rede

- 4. 1 Introdução
- 4.2 Redes de circuitos virtuais e de datagramas
- 4.3 O que há dentro de um roteador?
- 4.4 IP: Internet Protocol
  - formato do datagrama
  - endereçamento IPv4
  - ICMP
  - IPv6
- 4.5 Algoritmos de roteamento
  - estado de enlace
  - vetor de distâncias
  - roteamento hierárquico
- 4.6 Roteamento na Internet
  - RIP
  - OSPF
  - BGP
- 4.7 Roteamento broadcast e multicast

# Endereçamento IP: introdução

- endereço IP: identificador de 32 bits para *interface de* hospedeiro e roteador
- *interface:* conexão entre hospedeiro/ roteador e enlace físico
  - roteadores normalmente têm várias interfaces
  - hospedeiro normalmente tem uma interface
  - endereços IP associados a cada interface

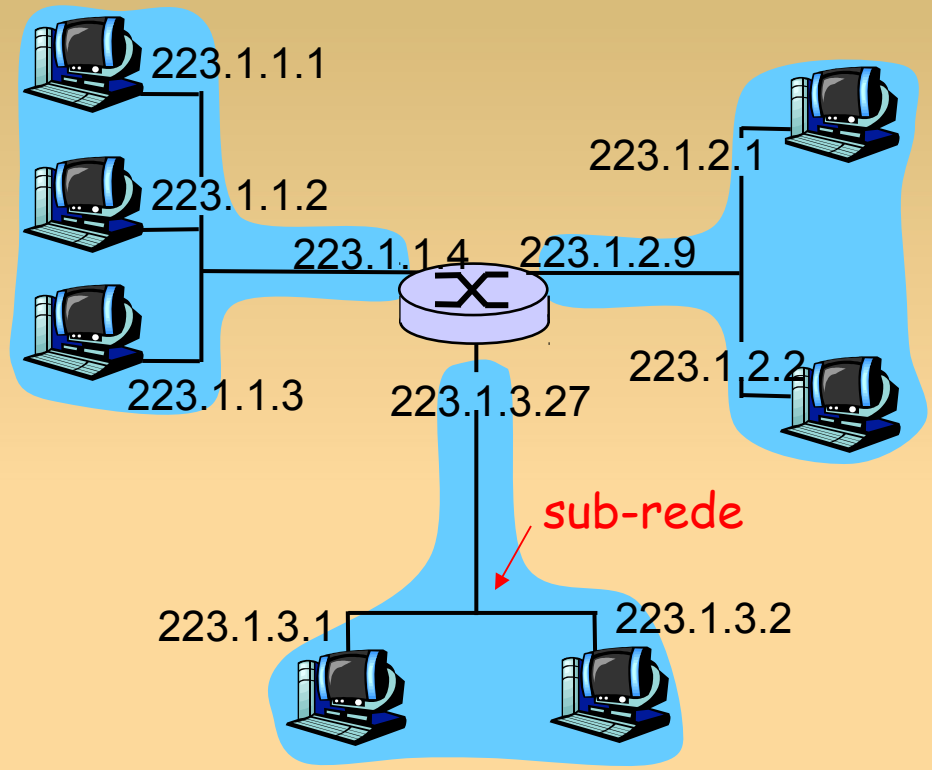



223.1.1.1 = 11011111 00000001 00000001 00000001

223 1 1 1

# Sub-redes

- endereço IP:
  - parte da sub-rede (bits de alta ordem)
  - parte do host (bits de baixa ordem)
- *O que é uma sub-rede?*
  - dispositivo se conecta à mesma parte da sub--rede do endereço IP
  - pode alcançar um ao outro fisicamente sem roteador intermediário

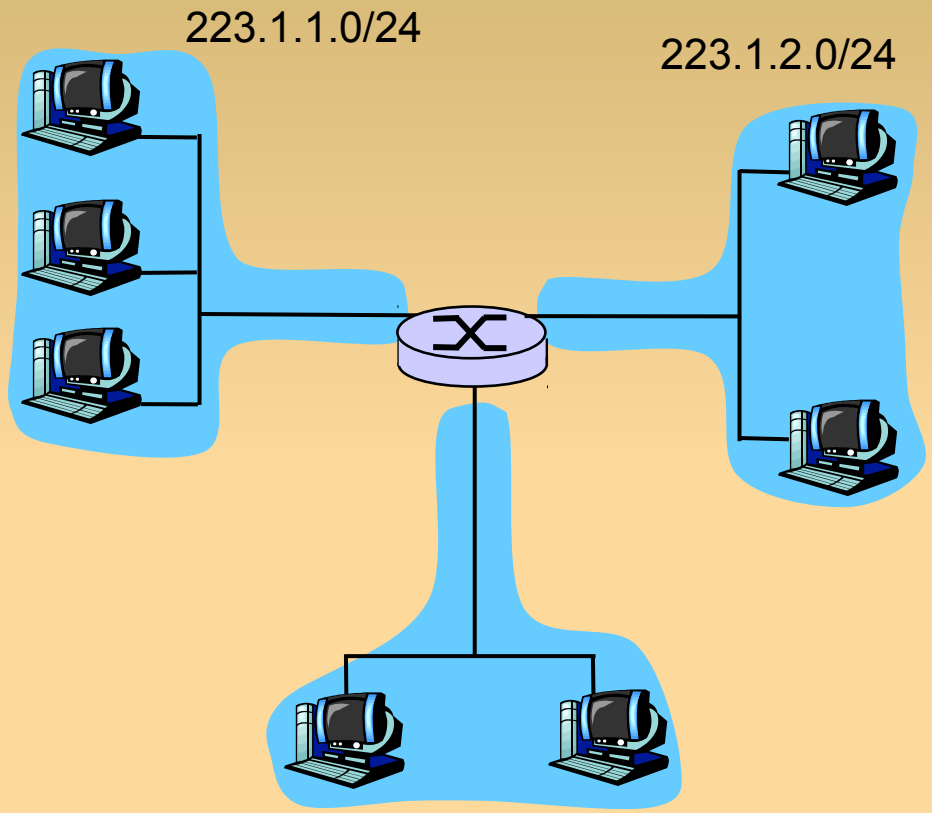



rede consistindo em 3 sub-redes

# Sub-redes

Receita

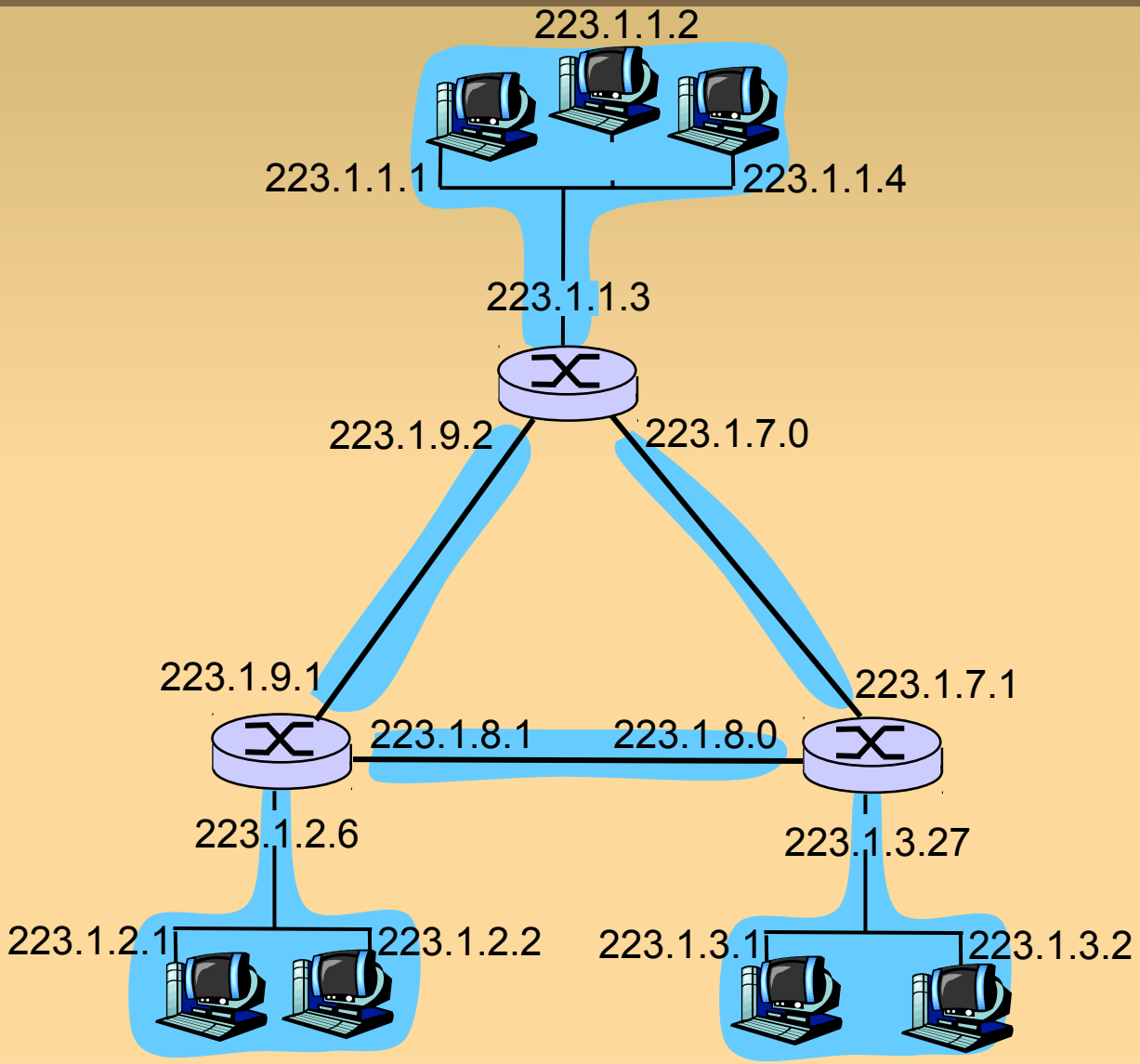
- para determinar as sub-redes, destaque cada interface de seu hospedeiro ou roteador, criando ilhas de redes isoladas. Cada rede isolada é denominada sub-red




Máscara de sub-rede: /24

# Sub-redes

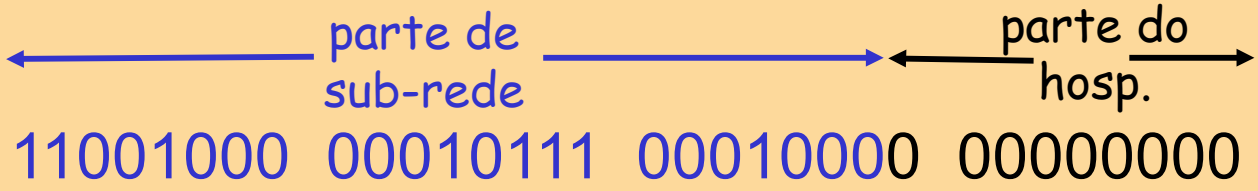
Quantas?



# Endereçamento IP: CIDR

CIDR: Classless InterDomain Routing (roteamento interdomínio sem classes)

- parte de sub-rede do endereço de tamanho arbitrário
- formato do endereço: a.b.c.d/x, onde x é # bits na parte de sub-rede do endereço




200.23.16.0/23

# Endereços IP: como obter um?

P: Como um *hospedeiro* obtém endereço IP?

- fornecido pelo administrador do sistema em um arquivo
  - Windows: painel de controle->rede ->configuração->tcp/ip->propriedades
  - UNIX: /etc/rc.config
- DHCP: Dynamic Host Configuration Protocol: recebe endereço dinamicamente do servidor
  - "plug-and-play"

# DHCP: Dynamic Host Configuration Protocol

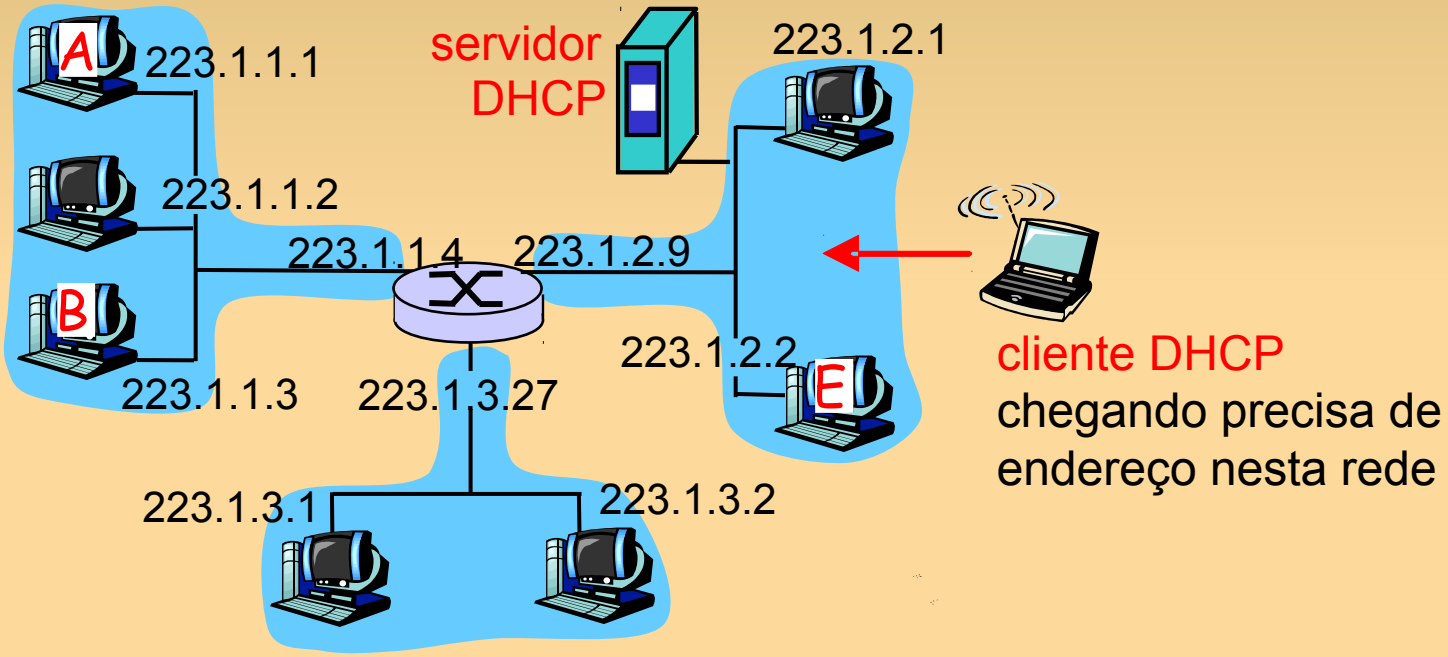
Objetivo: permitir que o hospedeiro obtenha *dinamicamente* seu endereço IP do servidor de rede quando se conectar à rede

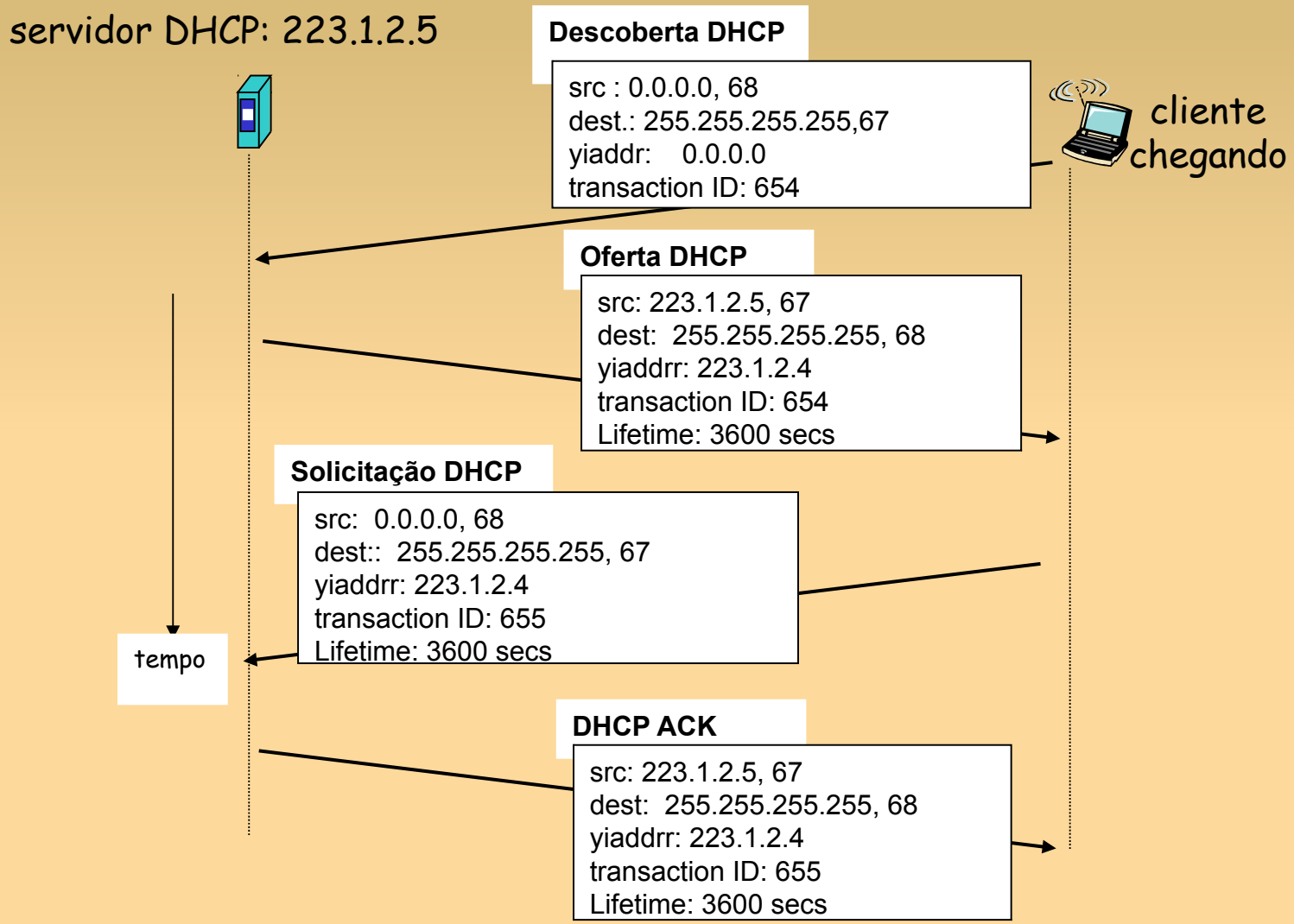
- pode renovar seu prazo no endereço utilizado
- permite reutilização de endereços (só mantém endereço enquanto conectado e "ligado")
- aceita usuários móveis que queiram se juntar à rede (mais adiante)

Visão geral do DHCP:

- host broadcasts "DHCP discover" msg
- servidor DHCP responde com msg "DHCP offer"
- hospedeiro requer endereço IP: msg "DHCP request"
- servidor DHCP envia endereço: msg "DHCP ack"

# DHCP – cenário cliente/servidor



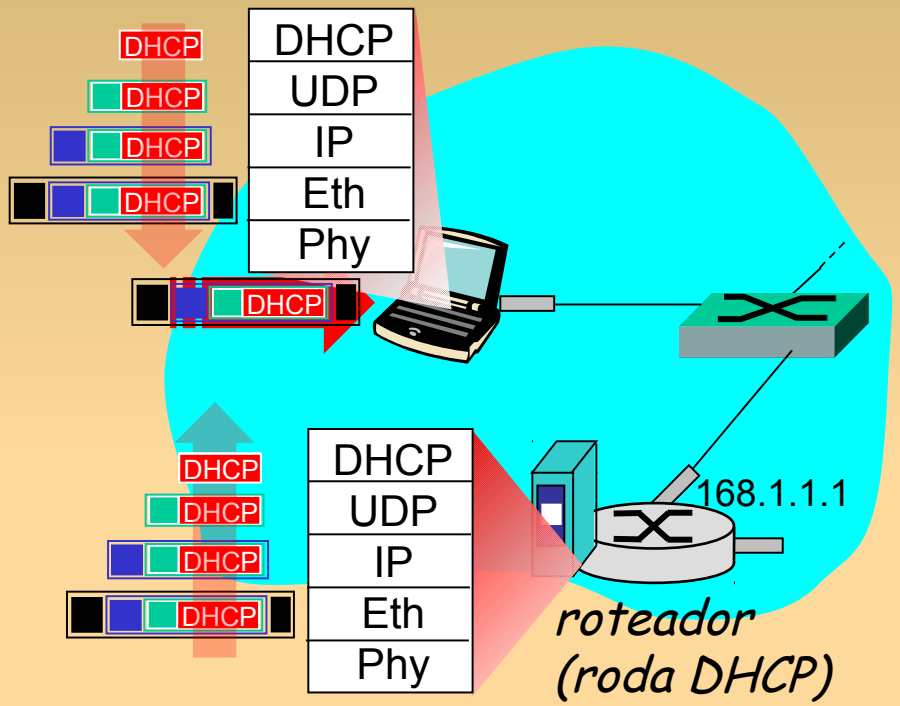


# DHCP: mais do que endereço IP

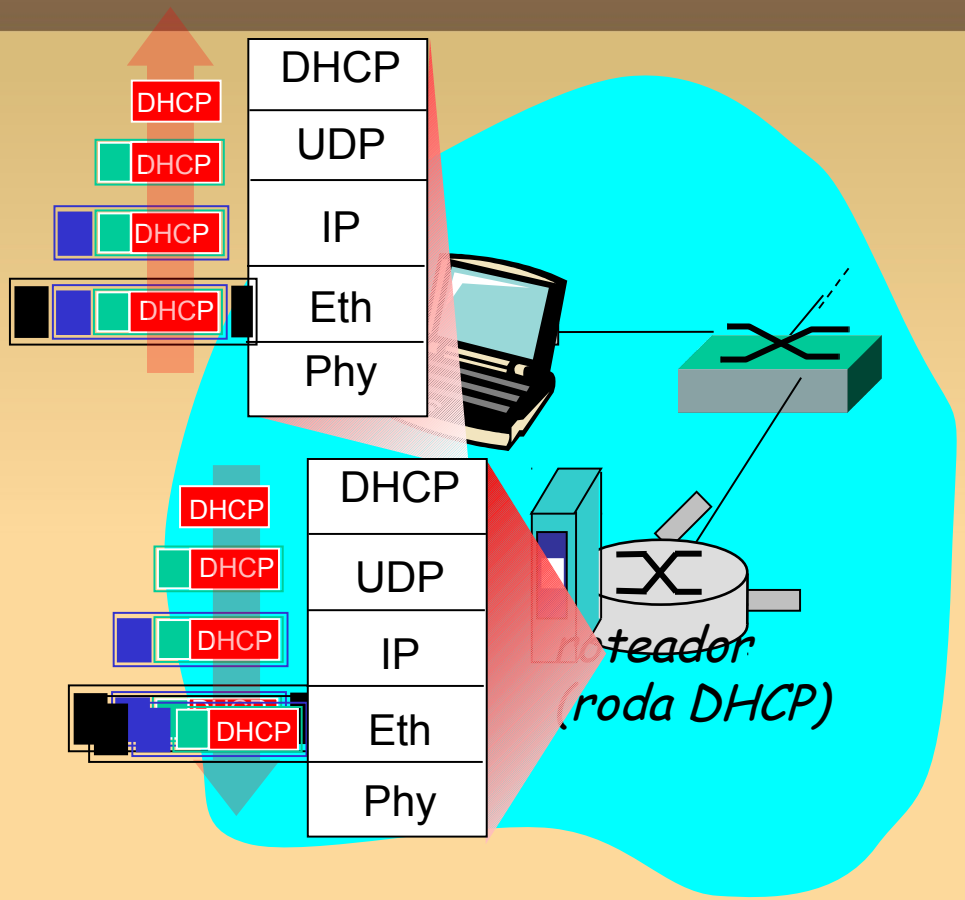
DHCP pode retornar mais do que apenas o endereço IP alocado na sub-rede:

- endereço do roteador do primeiro salto para o cliente
- nome e endereço IP do servidor DNS
- máscara de rede (indicando parte de rede *versus* hospedeiro do endereço)

# DHCP: exemplo




- conexão de laptop precisa do seu endereço IP, endereço do roteador do primeiro salto, endereço do servidor DNS: use DHCP
- solicitação DHCP encapsulada no UDP, encapsulada no IP, encapsulado no Ethernet 802.1
- broadcast de quadro Ethernet (dest: FFFFFFFFFFFF) na LAN, recebido no roteador rodando DHCP
- Ethernet demultiplexado para IP demultiplexado, UDP demultiplexado para DHCP




- servidor DCP formula DHCP ACK contendo endereço IP do cliente, endereço IP do roteador do primeiro salto para cliente, nome & endereço IP do servidor DNS

- encapsulamento do servidor DHCP, quadro repassado ao cliente, demultiplexando para DHCP no cliente
- cliente agora sabe seu endereço IP, nome e endereço IP do servidor DSN, endereço IP do seu roteador do primeiro salto

# Endereços IP: como obter um?

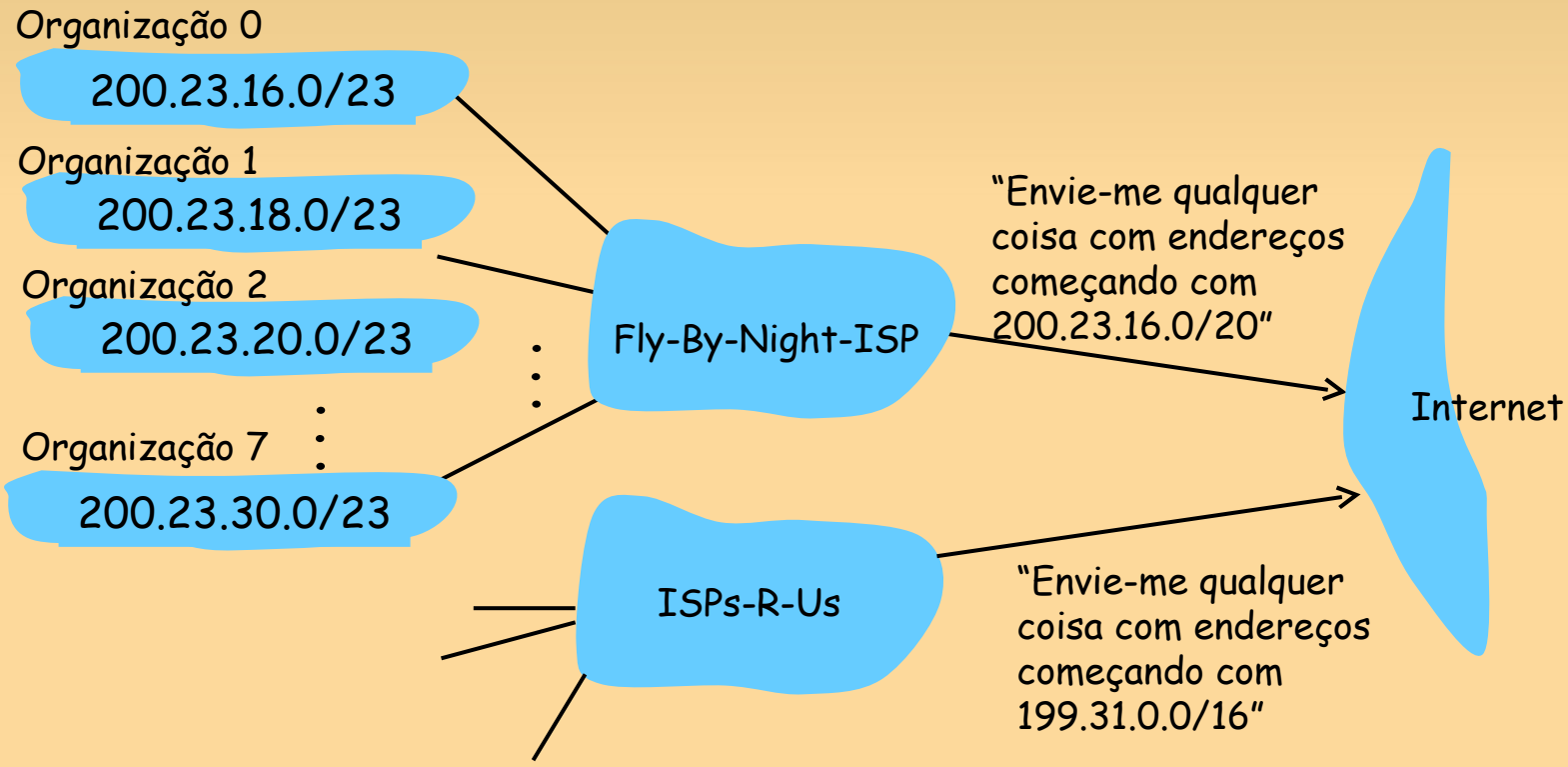
P: Como a *rede* obtém a parte de sub-rede do endereço IP?

R: Recebe parte alocada do espaço de endereços do seu ISP

| | | | |
|---|---|---|---|
| Bloco do ISP | 11001000 00010111 00010000 | 00000000 | 200.23.16.0/20 |
| Organização 0 | 11001000 00010111 00010000 | 00000000 | 200.23.16.0/23 |
| Organização 1 | 11001000 00010111 00010010 | 00000000 | 200.23.18.0/23 |
| Organização 2 | 11001000 00010111 00010100 | 00000000 | 200.23.20.0/23 |
| ... | ..... | .... | .... |
| Organização 7 | 11001000 00010111 00011110 | 00000000 | 200.23.30.0/23 |

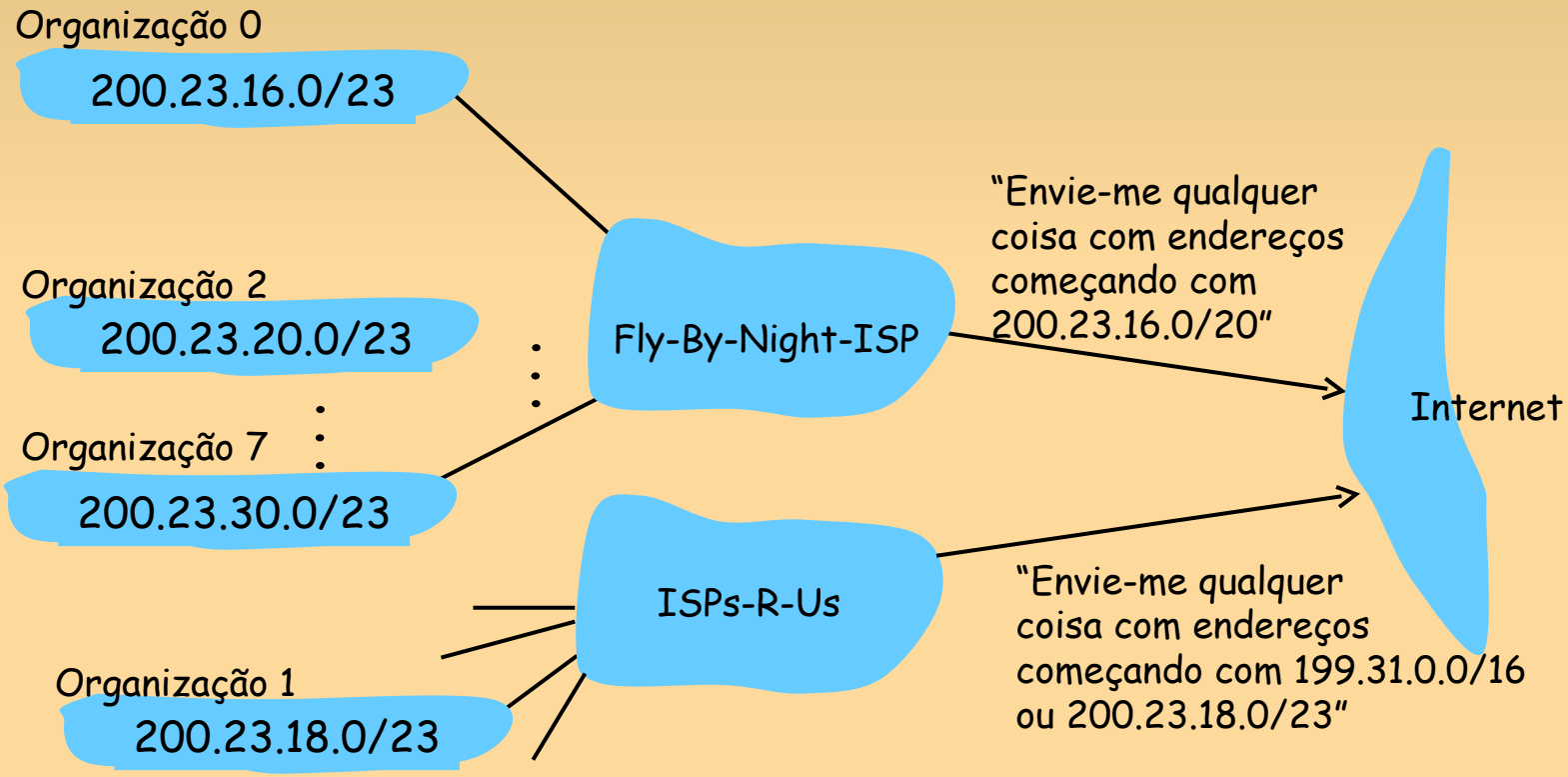
# Endereçamento hierárquico: agregação de rota

Endereçamento hierárquico permite anúncio eficiente da informação de roteamento:



# Endereçamento hierárquico: rotas mais específicas

ISPs-R-Us tem uma rota mais específica para Organização 1



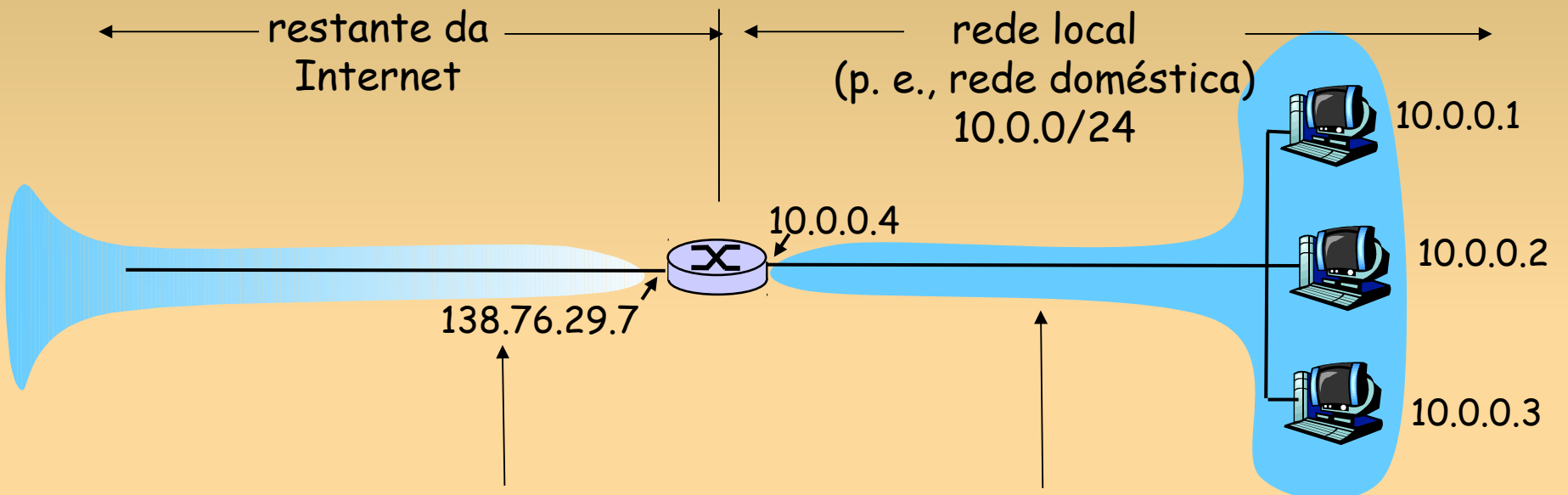
# Endereçamento IP: a última palavra...

P: Como um ISP recebe bloco de endereços?

R: ICANN: Internet Corporation for Assigned Names and Numbers

- aloca endereços
- administra o DNS
- atribui nomes de domínio e resolve disputas

# NAT: Network Address Translation




*todos* os datagramas *saindo* da rede local têm mesmo endereço IP NAT de origem: 138.76.29.7, mas diferentes números de porta de origem

datagramas com origem ou destino nesta rede têm endereço 10.0.0/24 para origem/destino (como sempre)

# NAT: Network Address Translation

- motivação: rede local usa apenas um endereço IP no que se refere ao mundo exterior:
  - intervalo de endereços não necessário pelo ISP: apenas um endereço IP para todos os dispositivos
  - pode mudar os endereços dos dispositivos na rede local sem notificar o mundo exterior
  - pode mudar de ISP sem alterar os endereços dos dispositivos na rede local
  - dispositivos dentro da rede local não precisam ser explicitamente endereçáveis ou visíveis pelo mundo exterior (uma questão de segurança).

# NAT: Network Address Translation

Implementação: roteador NAT deve:

- *Enviando datagramas: substituir* (endereço IP de origem, # porta) de cada datagrama saindo por (endereço IP da NAT, novo # porta)

  . . . clientes/servidores remotos responderão usando (endereço IP da NAT, novo # porta) como endereço de destino

- *lembrar (na tabela de tradução NAT)* de cada par de tradução (endereço IP de origem, # porta) para (endereço IP da NAT, novo # porta)

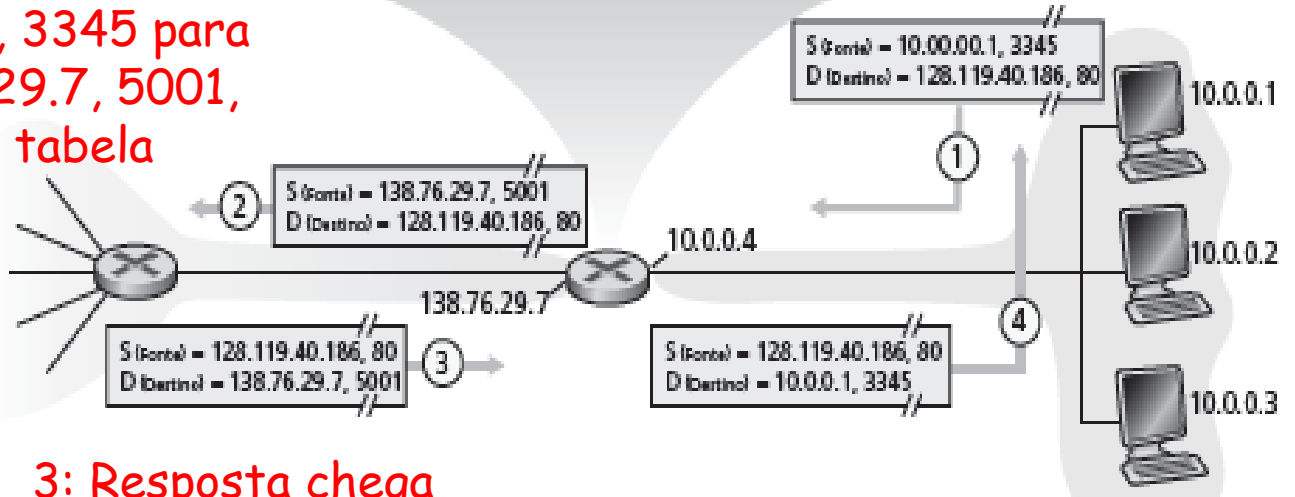
- *recebendo datagramas: substituir* (endereço IP da NAT, novo # porta) nos campos de destino de cada datagrama chegando por (endereço IP origem, # porta) correspondente, armazenado na tabela NAT

# NAT: Network Address Translation

2: roteador NAT muda endereço de origem do datagrama de 10.0.0.1, 3345 para 138.76.29.7, 5001, atualiza tabela

| Tabela de tradução NAT | |
|---|---|
| Lado da WAN | Lado da LAN |
| 138.76.29.7, 5001 | 10.0.0.1, 3345 |
| ... | ... |

1: hospedeiro 10.0.0.1 envia datagrama para 128.119.40.186, 80




3: Resposta chega endereço destino: 138.76.29.7, 5001

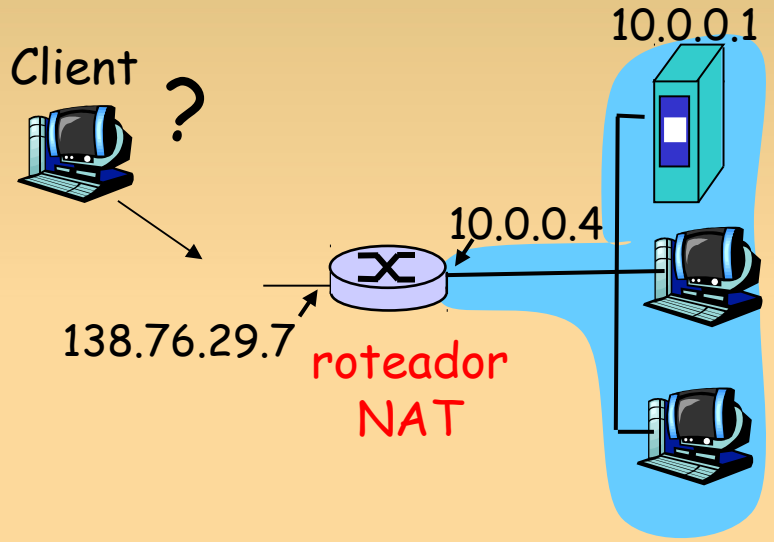
4: roteador NAT muda endereço de destino do datagrama de 138.76.29.7, 5001 para 10.0.0.1, 3345

# NAT: Network Address Translation

- campo de número de porta de 16 bits:
  - 60.000 conexões simultâneas com um único endereço no lado da LAN!
- NAT é controvertido:
  - roteadores só devem processar até a camada 3
  - viola argumento de fim a fim
    - a possibilidade de NAT deve ser levada em conta pelos projetistas da aplicação, p. e., aplicações P2P
  - a falta de endereços deverá ser resolvida pelo IPv6

# Problema da travessia da NAT

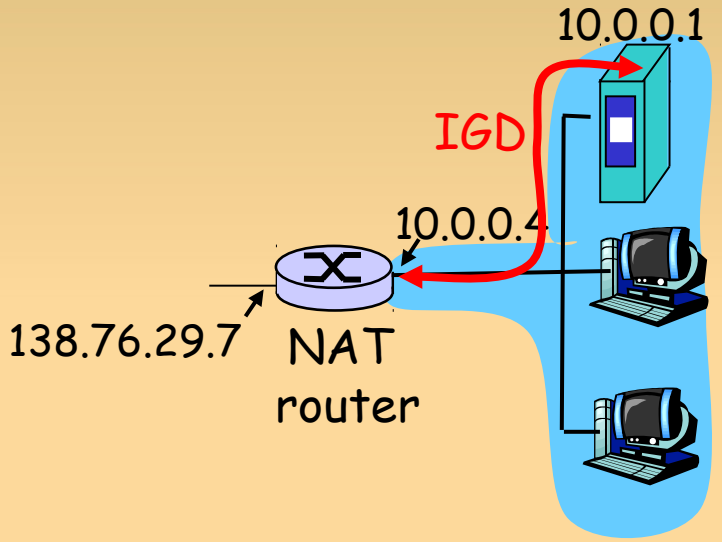
- cliente quer se conectar ao servidor com endereço 10.0.0.1
  - endereço do servidor 10.0.0.1 local à LAN (cliente não pode usá-lo como endereço destino)
  - apenas um endereço NAT visível externamente: 138.76.29.7
- solução 1: configure a NAT estaticamente para repassar as solicitações de conexão que chegam a determinada porta ao servidor
  - p. e., (138.76.29.7, porta 2500) sempre repassado para 10.0.0.1 porta 25000



# Problema da travessia da NAT

- solução 2: Universal Plug and Play (UPnP) Internet Gateway Device (IGD) Protocol. Permite que o hospedeiro com NAT:
  - descubra endereço IP público (138.76.29.7)
  - inclua/remova mapeamentos de porta (com tempos de posse)

  ou seja, automatizar configuração estática do mapa de porta NAT



# Capítulo 4: Camada de rede

- 4. 1 Introdução
- 4.2 Redes de circuitos virtuais e de datagramas
- 4.3 O que há dentro de um roteador?
- 4.4 IP: Internet Protocol
  - formato do datagrama
  - endereçamento IPv4
  - ICMP
  - IPv6
- 4.5 Algoritmos de roteamento
  - estado de enlace
  - vetor de distâncias
  - roteamento hierárquico
- 4.6 Roteamento na Internet
  - RIP
  - OSPF
  - BGP
- 4.7 Roteamento broadcast e multicast

# ICMP: Internet Control Message Protocol

- usado por hospedeiros & roteadores para comunicar informações em nível de rede
  - relato de erro: hospedeiro, rede, porta, protocolo inalcançável
  - eco de solicitação/ resposta (usado por ping)
- camada de rede "acima" do IP:
  - msgs ICMP transportadas em datagramas IP
- mensagem ICMP: tipo, código mais primeiros 8 bytes do datagrama IP causando erro

| Tipo | Cód, | Descrição |
|---|---|---|
| 0 | 0 | resposta de eco (ping) |
| 3 | 0 | rede de destino inalcançável |
| 3 | 1 | hosp. de destino inalcançável |
| 3 | 2 | protocolo de destino inalcançável |
| 3 | 3 | porta de destino inalcançável |
| 3 | 6 | rede de destino desconhecida |
| 3 | 7 | hosp. de destino desconhecido |
| 4 | 0 | redução da fonte (controle de congestionamento – não usado) |
| 8 | 0 | solicitação de eco (ping) |
| 9 | 0 | anúncio de rota |
| 10 | 0 | descoberta do roteador |
| 11 | 0 | TTL expirado |
| 12 | 0 | cabeçalho IP inválido |

# Traceroute e ICMP

- origem envia série de segmentos UDP ao destino
  - primeiro tem TTL = 1
  - segundo tem TTL = 2 etc.
  - número de porta improvável
- quando nº datagrama chegar no nº roteador:
  - roteador descarta datagrama
  - e envia à origem uma msg ICMP (tipo 11, código 0)
  - mensagem inclui nome do roteador & endereço IP

- quando a mensagem ICMP chega, origem calcula RTT
- traceroute faz isso 3 vezes

Critério de término

- segmento UDP por fim chega no hospedeiro de destino
- destino retorna pacote ICMP "host inalcançável" (tipo 3, código 3)
- quando origem recebe esse ICMP, termina.

# Capítulo 4: Camada de rede

- 4. 1 Introdução
- 4.2 Redes de circuitos virtuais e de datagramas
- 4.3 O que há dentro de um roteador?
- 4.4 IP: Internet Protocol
  - formato do datagrama
  - endereçamento IPv4
  - ICMP
  - IPv6
- 4.5 Algoritmos de roteamento
  - estado de enlace
  - vetor de distâncias
  - roteamento hierárquico
- 4.6 Roteamento na Internet
  - RIP
  - OSPF
  - BGP
- 4.7 Roteamento broadcast e multicast

# IPv6

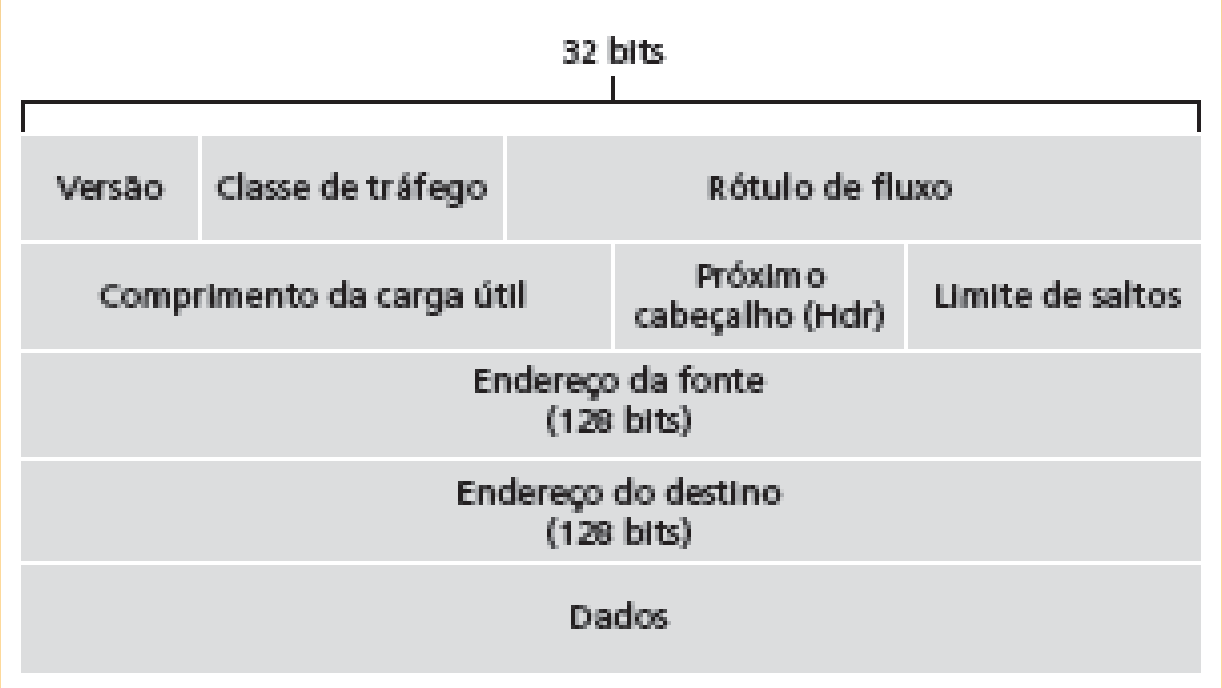
- motivação inicial: espaço de endereço de 32 bits logo estará completamente alocado
- motivação adicional:
  - formato de cabeçalho ajuda a agilizar processamento e repasse
  - mudanças no capítulo para facilitar QoS

formato de datagrama IPv6:

- cabeçalho de 40 bytes de tamanho fixo
- fragmentação não permitida

# Cabeçalho IPv6

*prioridade:* identificar prioridade entre datagramas no fluxo
*rótulo de fluxo:* identificar datagramas no mesmo "fluxo."
(conceito de "fluxo" não bem definido)
*próximo cabeçalho:* identificar protocolo da camada superior para dados



# Outras mudanças do IPv4

- *soma de verificação*: removida inteiramente para reduzir tempo de processamento em cada salto
- *opões:* permitidas, mas fora do cabeçalho, indicadas pelo campo de “Próximo Cabeçalho”
- *ICMPv6:* nova versão do ICMP
  - tipos de mensagem adicionais, p. e. “Pacote Muito Grande”
  - funções de gerenciamento de grupo multicast

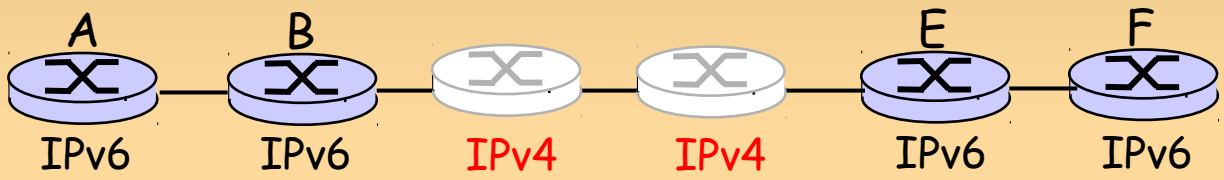
# Transição de IPv4 para IPv6

- nem todos os roteadores podem ser atualizados simultaneamente
  - sem “dia de conversão”
  - como a rede operará com roteadores IPv4 e IPv6 misturados?
- *implantação de túnel:* IPv6 transportado como carga útil no datagrama IPv4 entre roteadores IPv4

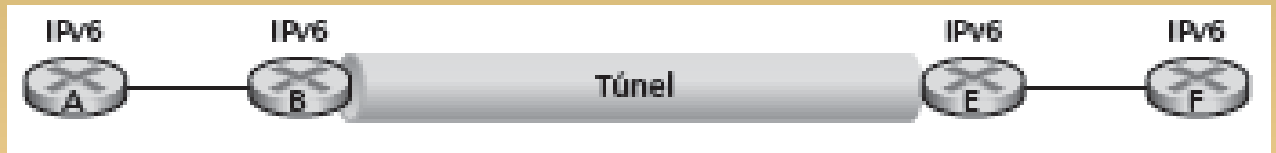
# Implantação de túnel

Visão lógica:

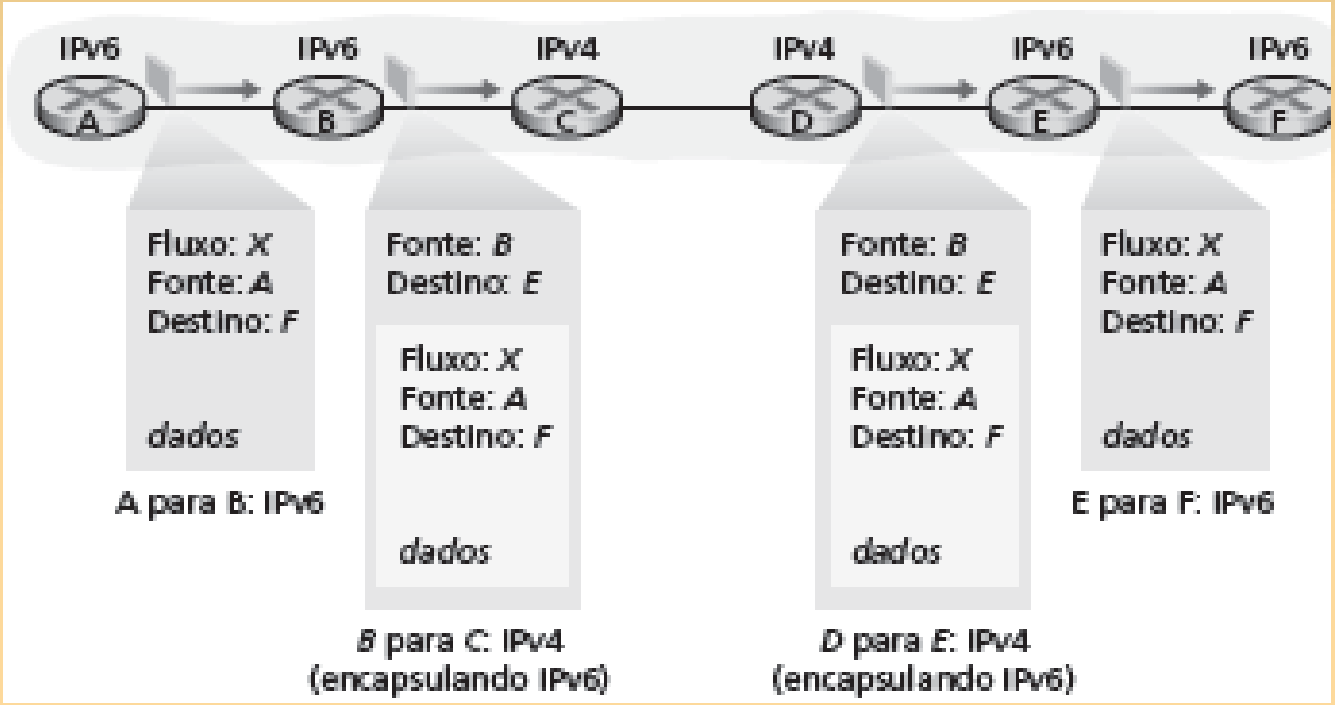

Visão física:



Visão lógica:



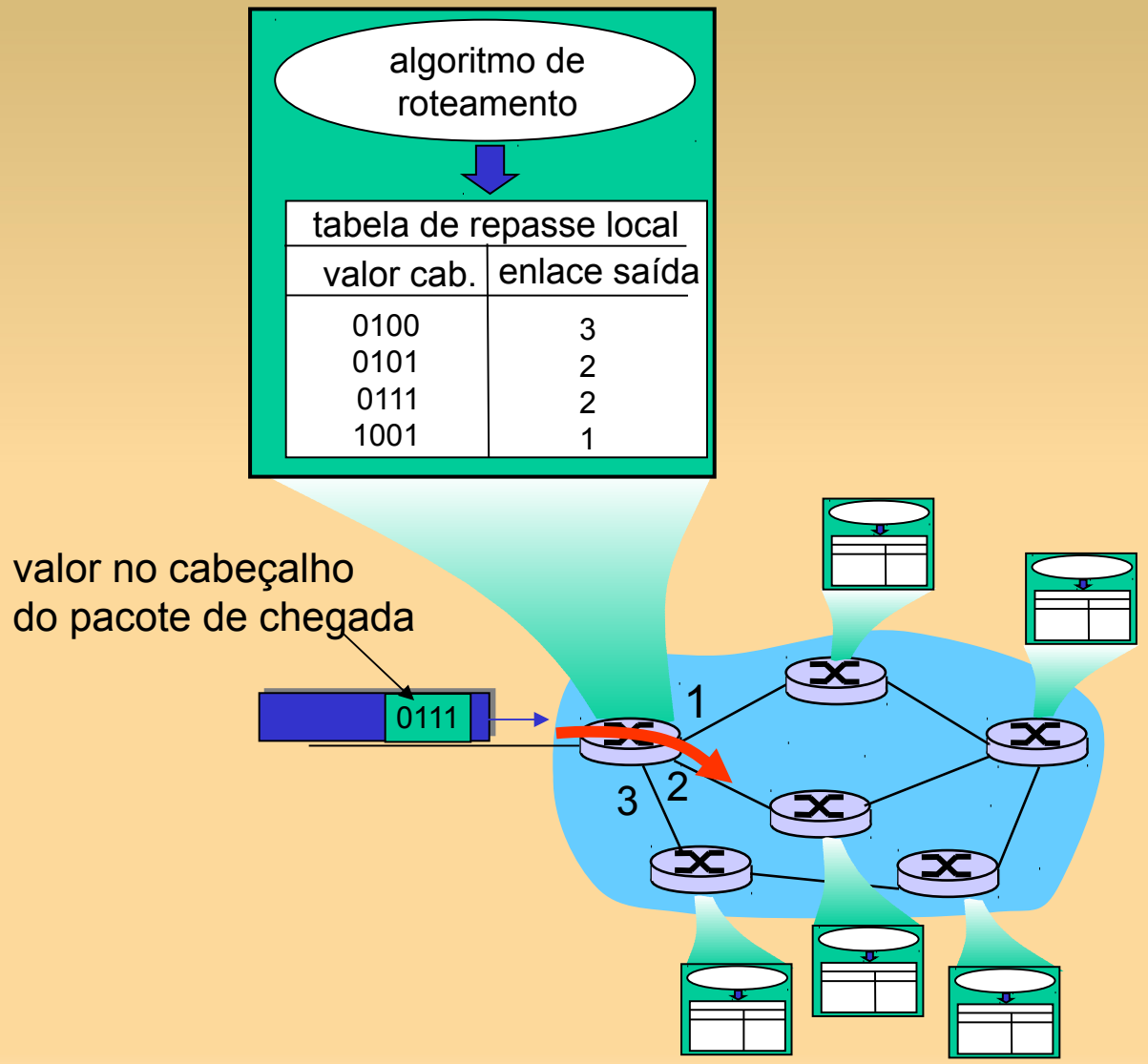

Visão física:



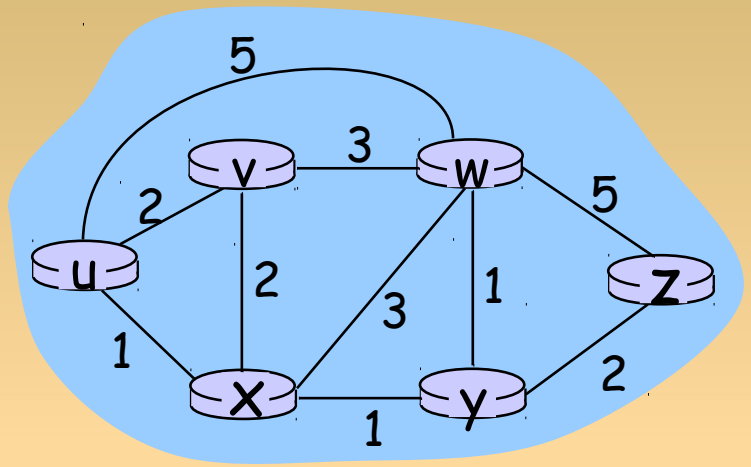
# Capítulo 4: Camada de rede

- 4. 1 Introdução
- 4.2 Redes de circuitos virtuais e de datagramas
- 4.3 O que há dentro de um roteador?
- 4.4 IP: Internet Protocol
  - formato do datagrama
  - endereçamento IPv4
  - ICMP
  - IPv6
- 4.5 Algoritmos de roteamento
  - estado de enlace
  - vetor de distâncias
  - roteamento hierárquico
- 4.6 Roteamento na Internet
  - RIP
  - OSPF
  - BGP
- 4.7 Roteamento broadcast e multicast

# Interação entre roteamento e repasse



# Abstração de grafo




Grafo: G = (N,E)

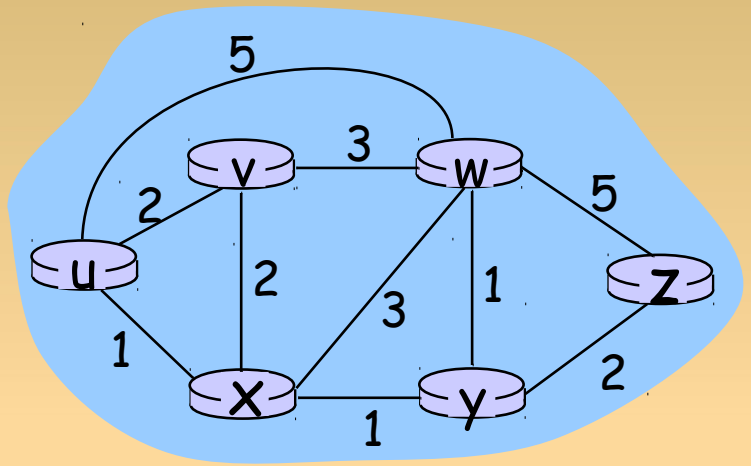
N = conjunto de roteadores = { u, v, w, x, y, z }

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

Comentário: Abstração de grafo é útil em outros contextos de rede

Exemplo: P2P, onde N é conj. de pares e E é conj. de conexões TCP

# Abstração de grafo: custos




- c(x,x') = custo do enlace (x,x')

  - p. e., c(w,z) = 5

- custo poderia ser sempre 1, ou inversamente relacionado à largura ou inversamente relacionado ao congestionamento

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

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

algoritmo de roteamento: algoritmo que encontra o caminho de menor custo

# Classificação do algoritmo de roteamento

## informação global ou descentralizada?

global:

- todos os roteadores têm topologia completa, informação de custo do enlace
- algoritmos de “estado do enlace”

descentralizada:

- roteador conhece vizinhos conectados fisicamente, custos de enlace para vizinhos
- processo de computação iterativo, troca de informações com vizinhos
- algoritmos de “vetor de distância”

## Estático ou dinâmico?

estático:

- rotas mudam lentamente com o tempo

dinâmico:

- rotas mudam mais rapidamente
  - atualização periódica
  - em resposta a mudanças no custo do enlace

# Capítulo 4: Camada de rede

- 4. 1 Introdução
- 4.2 Redes de circuitos virtuais e de datagramas
- 4.3 O que há dentro de um roteador?
- 4.4 IP: Internet Protocol
  - formato do datagrama
  - endereçamento IPv4
  - ICMP
  - IPv6
- 4.5 Algoritmos de roteamento
  - estado de enlace
  - vetor de distâncias
  - roteamento hierárquico
- 4.6 Roteamento na Internet
  - RIP
  - OSPF
  - BGP
- 4.7 Roteamento broadcast e multicast

# Algoritmo de roteamento de estado do enlace

## algoritmo de Dijkstra

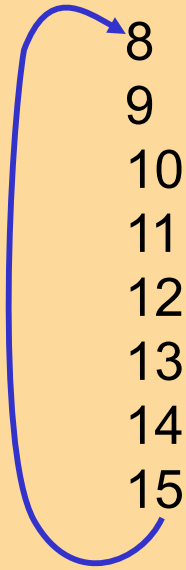
- nova topologia, custos de enlace conhecidos de todos os nós
  - realizado por “broadcast de estado do enlace”
  - todos os nós têm a mesma informação
- calcula caminhos de menor custo de um nó (“origem”) para todos os outros nós
  - da tabela de repasse para esse nó
- iterativo: após k iterações, sabe caminho de menor custo para k destinos

## notação:

- c(x,y): custo do enlace do nó x até y; = ∞ se não forem vizinhos diretos
- D(v): valor atual do custo do caminho da origem ao destino v
- p(v): nó predecessor ao longo do caminho da origem até v
- N': conjunto de nós cujo caminho de menor custo é definitivamente conhecido

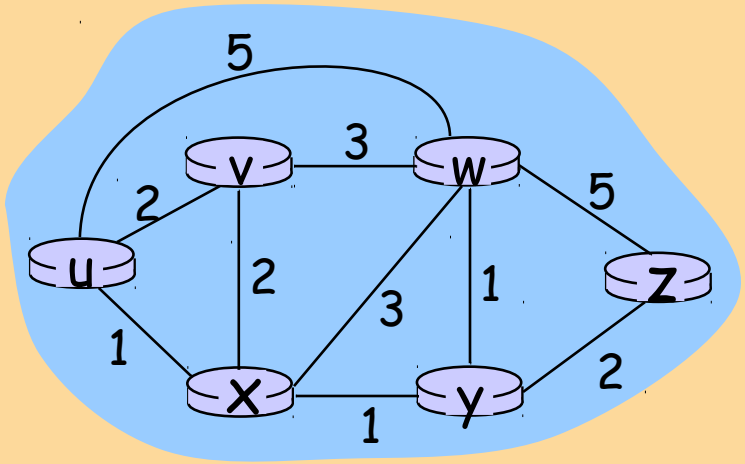
# Algoritmo de Dijkstra

```
1  Inicialização:
2    N' = {u}
3    para todos os nós v
4      se v adjacente a u
5          então D(v) = c(u,v)
6      senão D(v) = ∞
7
8   Loop
9     acha w não em N' tal que D(w) é mínimo
10    acrescenta w a N'
11    atualiza D(v) para todo v adjacente a w e não em N' :
12       D(v) = min( D(v), D(w) + c(w,v) )
13    /* novo custo para v é custo antigo para v ou custo conhecido
14     do caminho mais curto para w + custo de w para v */
15  até todos os nós 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 | | | | | |



# Algoritmo de Dijkstra: exemplo (2)

árvore resultante do caminho mais curto a partir de u:

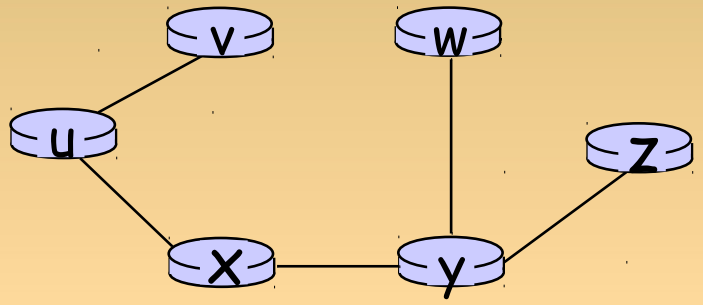



tabela de repasse resultante em u:

| destino | enlace |
|---|---|
| v | (u,v) |
| x | (u,x) |
| y | (u,x) |
| w | (u,x) |
| z | (u,x) |

# Capítulo 4: Camada de rede

- 4. 1 Introdução
- 4.2 Redes de circuitos virtuais e de datagramas
- 4.3 O que há dentro de um roteador?
- 4.4 IP: Internet Protocol
  - formato do datagrama
  - endereçamento IPv4
  - ICMP
  - IPv6
- 4.5 Algoritmos de roteamento
  - estado de enlace
  - vetor de distâncias
  - roteamento hierárquico
- 4.6 Roteamento na Internet
  - RIP
  - OSPF
  - BGP
- 4.7 Roteamento broadcast e multicast

# Algoritmo de vetor de distância

Equação de Bellman-Ford (programação dinâmica)

defina

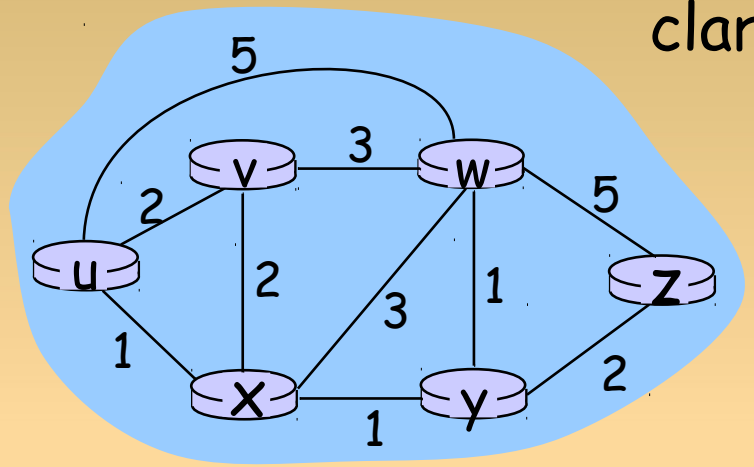
$d_x(y)$ : = custo do caminho de menor custo de x para y

depois

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

onde min assume todos os vizinhos v de x

# Exemplo de Bellman-Ford




claramente, $d_v(z) = 5$, $d_x(z) = 3$, $d_w(z) = 3$

equação B-F diz:

$$d_u(z) = \min \{ c(u,v) + d_v(z), \; c(u,x) + d_x(z), \; c(u,w) + d_w(z) \}$$

$$= \min \{2 + 5, \; 1 + 3, \; 5 + 3\} = 4$$

nó que alcança mínimo é o próximo salto no caminho mais curto ➔ tabela de repasse

# Algoritmo de vetor de distância

- $D_x(y)$ = estimativa do menor custo de x para y
- nó x sabe custo de cada vizinho v: $c(x,v)$
- nó x mantém vetor de distância $\mathbf{D}_x = [D_x(y): y \in N ]$
- nó x também mantém vetor de distância de seus vizinhos
  - para cada vizinho v, x mantém $\mathbf{D}_v = [D_v(y): y \in N ]$

# Algoritmo de vetor de distância (4)

ideia básica:

- de tempos em tempos, cada nó envia sua própria estimativa de vetor de distância aos vizinhos
- assíncrono
- quando um nó x recebe nova estimativa DV do vizinho, ele atualiza seu próprio DV usando a equação de B-F:

$D_x(y) \leftarrow \min_v\{c(x,v) + D_v(y)\}$ *para cada nó y* $N$

- sob condições modestas, naturais, a estimativa $D_x(y)$ *converge para o menor custo real* $d_x(y)$

# Algoritmo de vetor de distância (5)

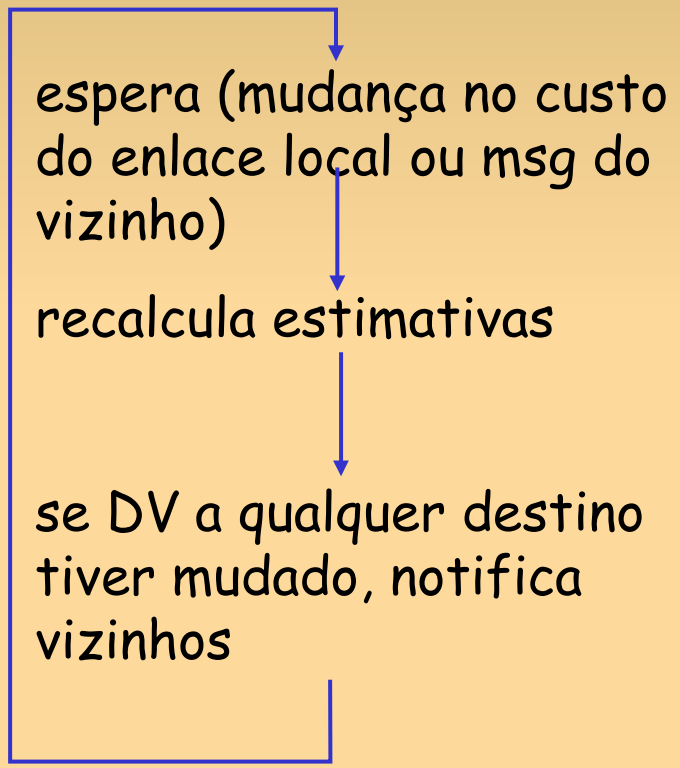
iterativo, assíncrono: cada iteração local causada por:

- mudança de custo do enlace local
- mensagem de atualização do DV do vizinho

distribuído:

- cada nó notifica os vizinhos *apenas* quando seu DV muda
  - vivinhos, então, notificam seus vizinhos, se necessário

Cada nó:



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

**tabela nó x**

| de | custo para x | custo para y | custo para z |
|---|---|---|---|
| x | 0 | 2 | 7 |
| y | ∞ | ∞ | ∞ |
| z | ∞ | ∞ | ∞ |

| de | custo para x | custo para y | custo para z |
|---|---|---|---|
| x | 0 | 2 | 3 |
| y | 2 | 0 | 1 |
| z | 7 | 1 | 0 |

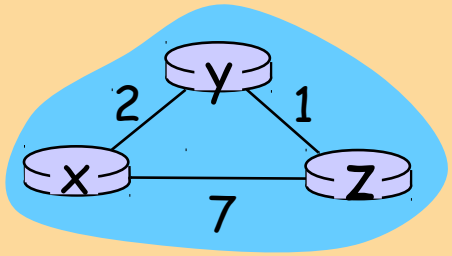
$$D_x(z) = \min\{c(x,y) + D_y(z),\ c(x,z) + D_z(z)\}$$
$$= \min\{2+1\ ,\ 7+0\} = 3$$

**tabela nó y**

| de | custo para x | custo para y | custo para z |
|---|---|---|---|
| x | ∞ | ∞ | ∞ |
| y | 2 | 0 | 1 |
| z | ∞ | ∞ | ∞ |

**tabela nó z**

| de | custo para x | custo para y | custo para z |
|---|---|---|---|
| x | ∞ | ∞ | ∞ |
| y | ∞ | ∞ | ∞ |
| z | 7 | 1 | 0 |




tempo

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

**tabela nó x**

custo para

| de | x | y | z |
|---|---|---|---|
| x | 0 | 2 | 7 |
| y | ∞ | ∞ | ∞ |
| z | ∞ | ∞ | ∞ |

custo para

| de | x | y | z |
|---|---|---|---|
| x | 0 | 2 | 3 |
| y | 2 | 0 | 1 |
| z | 7 | 1 | 0 |

custo para

| de | x | y | z |
|---|---|---|---|
| x | 0 | 2 | 3 |
| y | 2 | 0 | 1 |
| z | 3 | 1 | 0 |

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

**tabela nó y**

custo para

| de | x | y | z |
|---|---|---|---|
| x | ∞ | ∞ | ∞ |
| y | 2 | 0 | 1 |
| z | ∞ | ∞ | ∞ |

custo para

| de | x | y | z |
|---|---|---|---|
| x | 0 | 2 | 7 |
| y | 2 | 0 | 1 |
| z | 7 | 1 | 0 |

custo para

| de | x | y | z |
|---|---|---|---|
| x | 0 | 2 | 3 |
| y | 2 | 0 | 1 |
| z | 3 | 1 | 0 |

**tabela nó z**

custo para

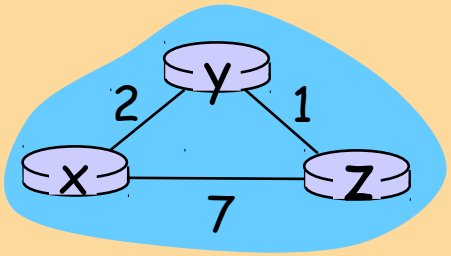
| de | x | y | z |
|---|---|---|---|
| x | ∞ | ∞ | ∞ |
| y | ∞ | ∞ | ∞ |
| z | 7 | 1 | 0 |

custo para

| de | x | y | z |
|---|---|---|---|
| x | 0 | 2 | 7 |
| y | 2 | 0 | 1 |
| z | 3 | 1 | 0 |

custo para

| de | x | y | z |
|---|---|---|---|
| x | 0 | 2 | 3 |
| y | 2 | 0 | 1 |
| z | 3 | 1 | 0 |

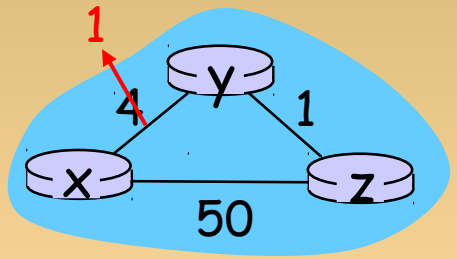



tempo

# Vetor de distância: mudanças de custo do enlace

mudanças de custo do enlace:

- nó detecta mudança de custo no enlace local
- atualiza informação de roteamento, recalcula vetor de distância
- se DV mudar, notifica vizinhos




"boas notícias correm rápido"

no tempo $t_0$, $y$ detecta a mudança do custo do enlace, atualiza seu DV e informa aos seus vizinhos.

no tempo $t_1$, $z$ recebe a atualização de $y$ e atualiza sua tabela. Calcula um novo custo mínimo para $x$ e envia seu DV aos vizinhos.

no tempo $t_2$, $y$ recebe a atualização de $z$ e atualiza sua tabela de distância. Menores custos de $y$ não mudam, e daí $y$ *não* envia qualquer mensagem a $z$.

## mudanças de custo do enlace:

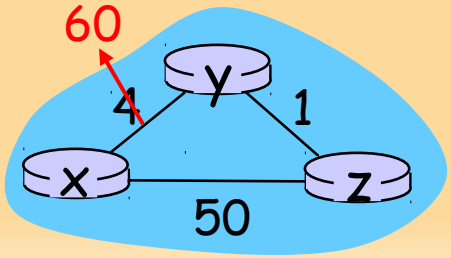
$D_y(x) = \min\{c(y,x) + D_x(x), c(y,z) + D_z(x)\} = \min(60 + 0 , 1 + 5) = 6$

$D_z(x) = \min\{c(z,x) + D_x(x), c(z,y) + D_y(x)\} = \min(50 + 0 , 1 + 6) = 7$

- boas notícias correm rápido
- más notícias correm lento – problema da "contagem até o infinito"!
- 44 iterações antes que o algoritmo estabilize: ver texto

## reverso envenenado:

- se Z passa por Y para chegar a X:
  - Z diz a Y que sua distância (de Z) até X é infinita (de modo que Y não roteará para X passando por Z)
- isso solucionará completamente o problema da contagem até o infinito?



# Comparação dos algoritmos LS e DV

- Distance vector (e.g. RIP, BGP)
  - Cada roteador envia a tabela (vetor) para o seu vizinho; apenas enviam o vetor para os seus vizinhos
  - Os vizinhos não conhecem a topologia da rede (i.e. como outros roteadores estão interconectados)
- Link state (e.g. OSPF)
  - Roteadores trocam informações sobre as suas conexões dentro de uma rede e constrói a topologia
  - Cada nó possui conhecimento da topologia de rede
  - Cada roteador utiliza o algoritmo de Dijkstra para calcular a melhor rota

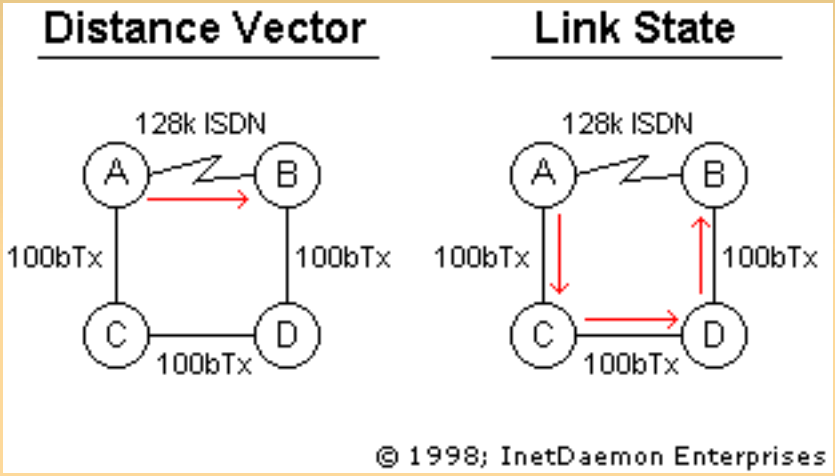
# Comparação dos algoritmos LS e DV

- Distance vector
  - Envia updates periódicos a cada 30 a 90 segundos
  - Envia apenas para os seus vizinhos o custo do enlace baseado no # de hops
  - Pode carregar informações erradas (confiam nos seus vizinhos)
- Link state
  - Updates são enviados para toda a rede (broadcast) e não apenas para os vizinhos
  - Mas, enviam apenas os custos dos enlaces ligados diretamente a eles

# Comparação dos algoritmos LS e DV

- Distance vector
  - Mais simples e eficiente em redes menores e com pouca variação
  - Envia updates periódicos
  - Vire a direita = 200Km São Paulo; Vire a esquerda = 150Km Jundiaí; siga reto = Ribeirão
- Link state
  - Utiliza outros critérios como largura de banda, atraso, confiabilidade e balanceamento de carga
  - Envia updates apenas quando há mudança no link
  - Fornece o mapa; você pode ver uma visão geral do percurso

# Comparação dos algoritmos LS e DV



# Capítulo 4: Camada de rede

- 4. 1 Introdução
- 4.2 Redes de circuitos virtuais e de datagramas
- 4.3 O que há dentro de um roteador?
- 4.4 IP: Internet Protocol
  - formato do datagrama
  - endereçamento IPv4
  - ICMP
  - IPv6
- 4.5 Algoritmos de roteamento
  - estado de enlace
  - vetor de distâncias
  - roteamento hierárquico
- 4.6 Roteamento na Internet
  - RIP
  - OSPF
  - BGP
- 4.7 Roteamento broadcast e multicast

# Roteamento hierárquico

nosso estudo de roteamento até aqui – o ideal:

- todos os roteadores idênticos
- rede "achatada"

… *não* acontece na prática

escala: com 200 milhões de destinos:

- não pode armazenar todos os destinos nas tabelas de roteamento!
- troca de tabela de roteamento atolaria os enlaces!

autonomia administrativa

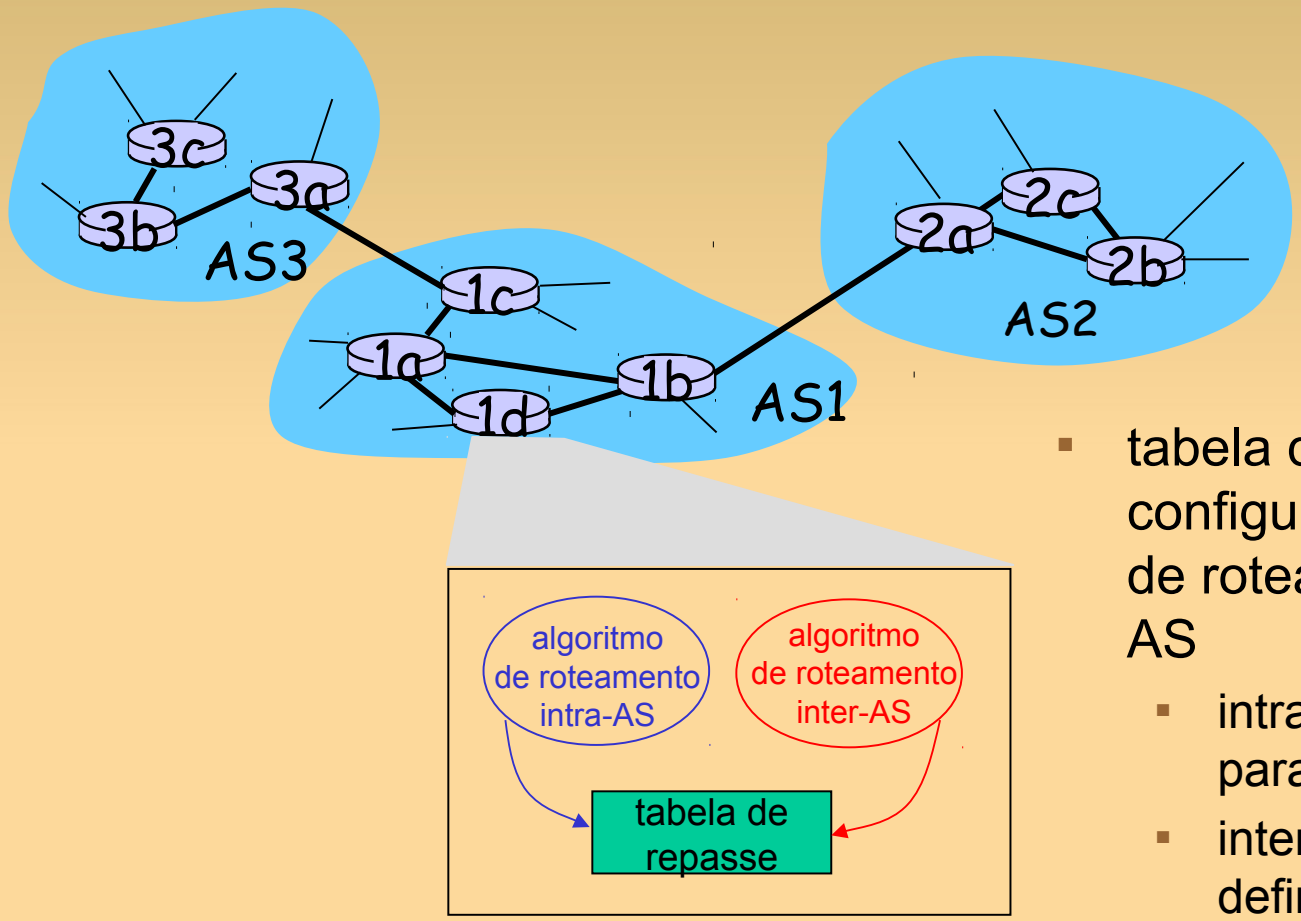
- Internet = rede de redes
- cada administrador de rede pode querer controlar o roteamento em sua própria rede

- roteadores agregados em regiões, “sistemas autônomos” (AS)
- roteadores no mesmo AS rodam o mesmo protocolo de roteamento
  - protocolo de roteamento “intra-AS”
  - roteadores em ASes diferentes podem executar protocolo de roteamento intra-AS diferente

roteador de borda

- Enlace direto com roteador em outro AS

# ASes interconectados




- tabela de repasse configurada por algoritmo de roteamento intra e inter-AS
  - intra-AS define entradas para destinos internos
  - inter-AS & intra-AS definem entradas para destinos externos

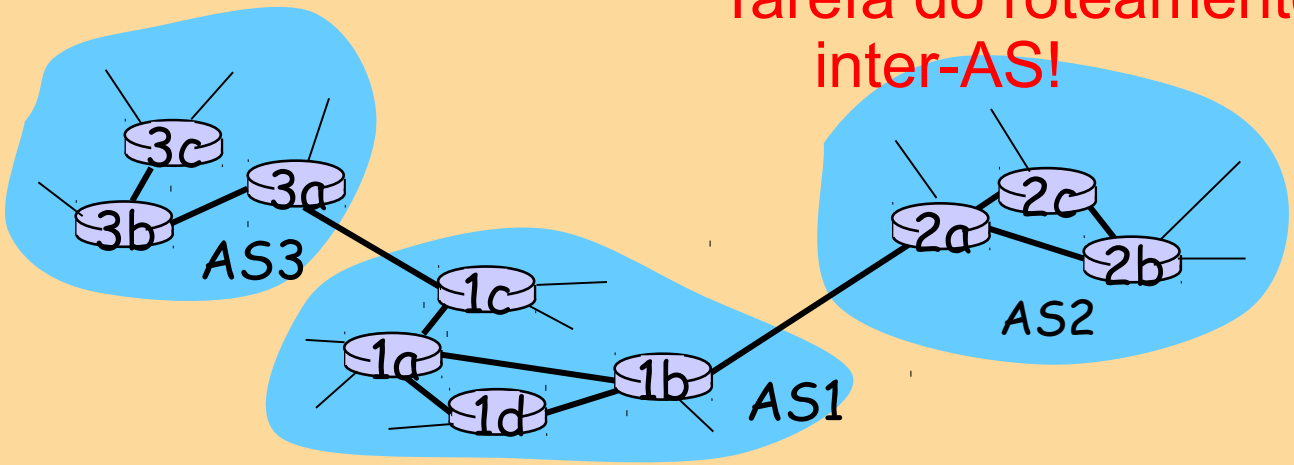
# Tarefas inter-AS

- suponha que roteador no AS1 recebe datagrama destinado para fora do AS1:
  - roteador deve encaminhar pacote ao roteador de borda, mas qual?

AS1 deve:

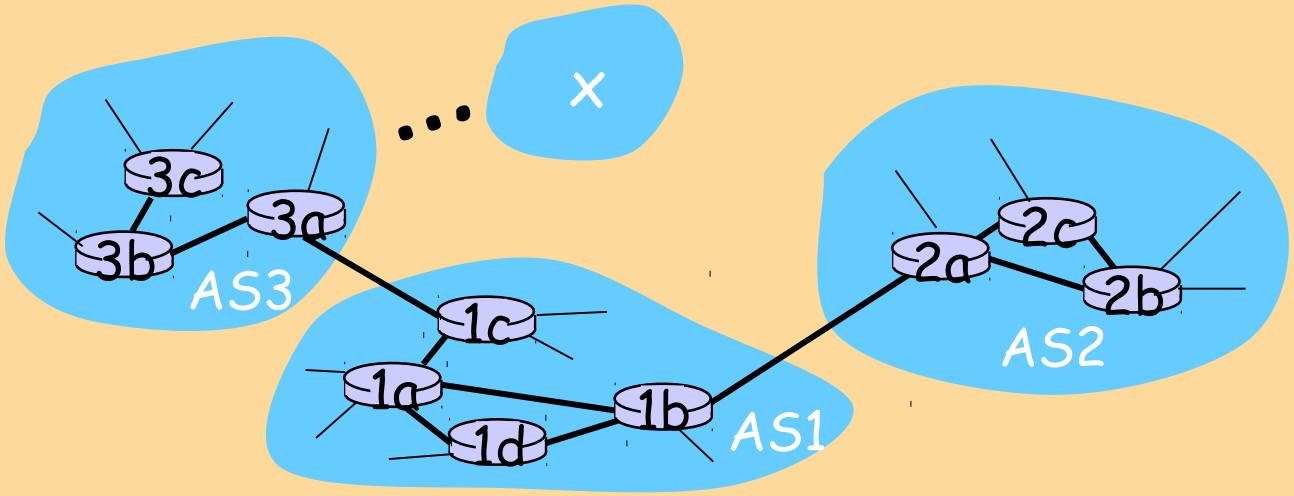
1. descobrir quais destinos são alcançáveis por AS2 e quais por AS3
2. propagar essa informação de acessibilidade a todos os roteadores no AS1

Tarefa do roteamento inter-AS!



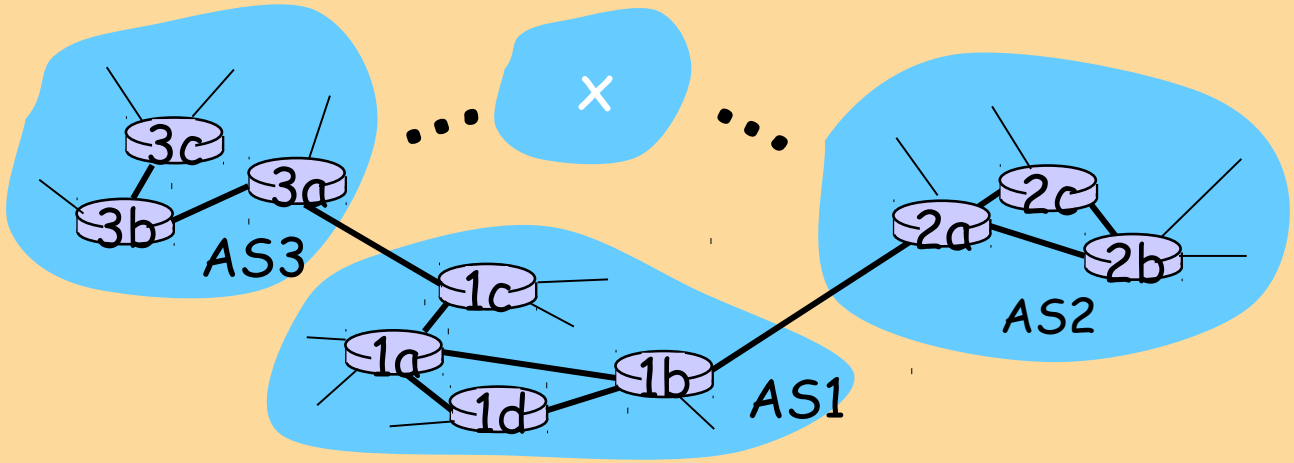
# Exemplo: definindo tabela de repasse no roteador 1d

- suponha que AS1 descubra (pelo protocolo inter-AS) que a sub-rede *x* é alcançável via AS3 (gateway 1c), mas não via AS2.
- protocolo inter-AS propaga informação de acessibilidade a todos os roteadores internos.
- roteador 1d determina pelo roteamento intra-AS informação de que sua interface *I* está no caminho de menor custo para 1c.
  - instala entrada da tabela de repasse *(x,I)*



# Exemplo: escolhendo entre múltiplos ASes

- agora suponha que o AS1 descubra pelo protocolo inter-AS que a sub-rede *x* pode ser alcançada por AS3 *e* por AS2.
- para configurar a tabela de repasse, roteador 1d deve determinar para que gateway ele deve repassar os pacotes para o destino x.
  - isso também é tarefa do protocolo de roteamento inter-AS!



- agora suponha que AS1 descubra pelo protocolo inter-AS que sub-rede *x* pode ser alcançada por AS3 *e* por AS2.
- para configurar a tabela de repasse, o roteador 1d deve determinar para qual gateway deve repassar pacotes para destino x.
  - isso também é tarefa do protocolo de roteamento inter-AS!
- roteamento da batata quente: envia pacote para o mais próximo dos dois roteadores.