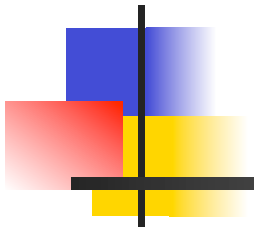


Cap. 4 – O nível rede

Nota prévia

A estrutura da apresentação é semelhante e utiliza algumas das figuras, textos e outros materiais do livro de base do curso

James F. Kurose and Keith W. Ross, "Computer Networking - A Top-Down Approach Featuring the Internet," Addison Wesley Longman, Inc., 3rd Edition, 2005

Objectivos do capítulo

- Perceber os princípios base do nível rede:
  - Encaminhamento (*routing*)
  - Escala
  - Endereçamento
- Estudar mais em detalhe o nível rede na Internet

Organização do capítulo

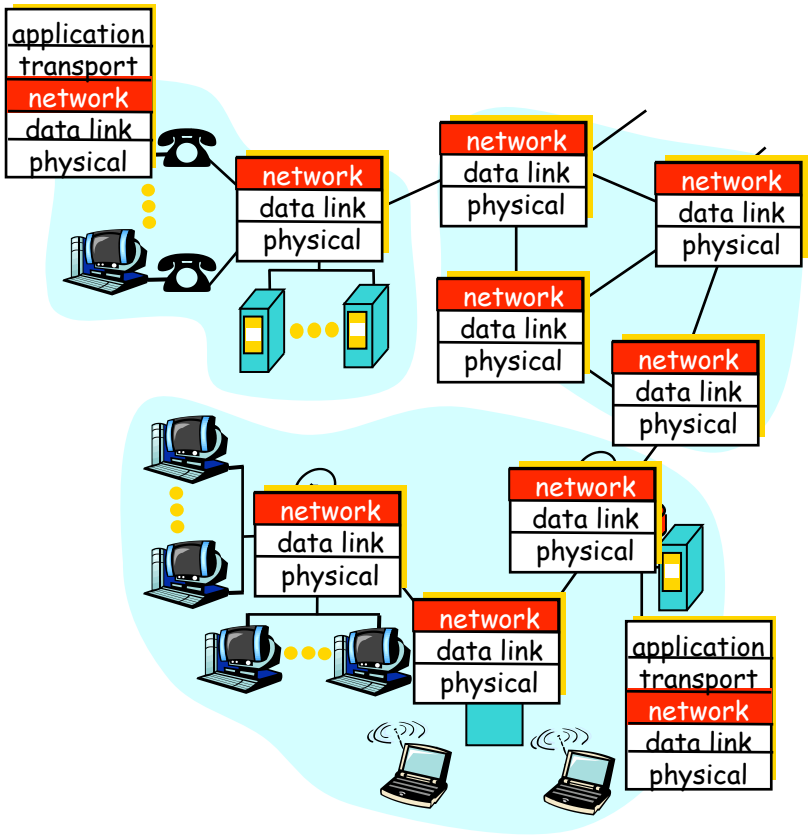
- Objectivos do nível rede
- Estudo do protocolo IP e de alguns aspectos do funcionamento das redes IP
- Algoritmos de encaminhamento

Objectivos do nível rede

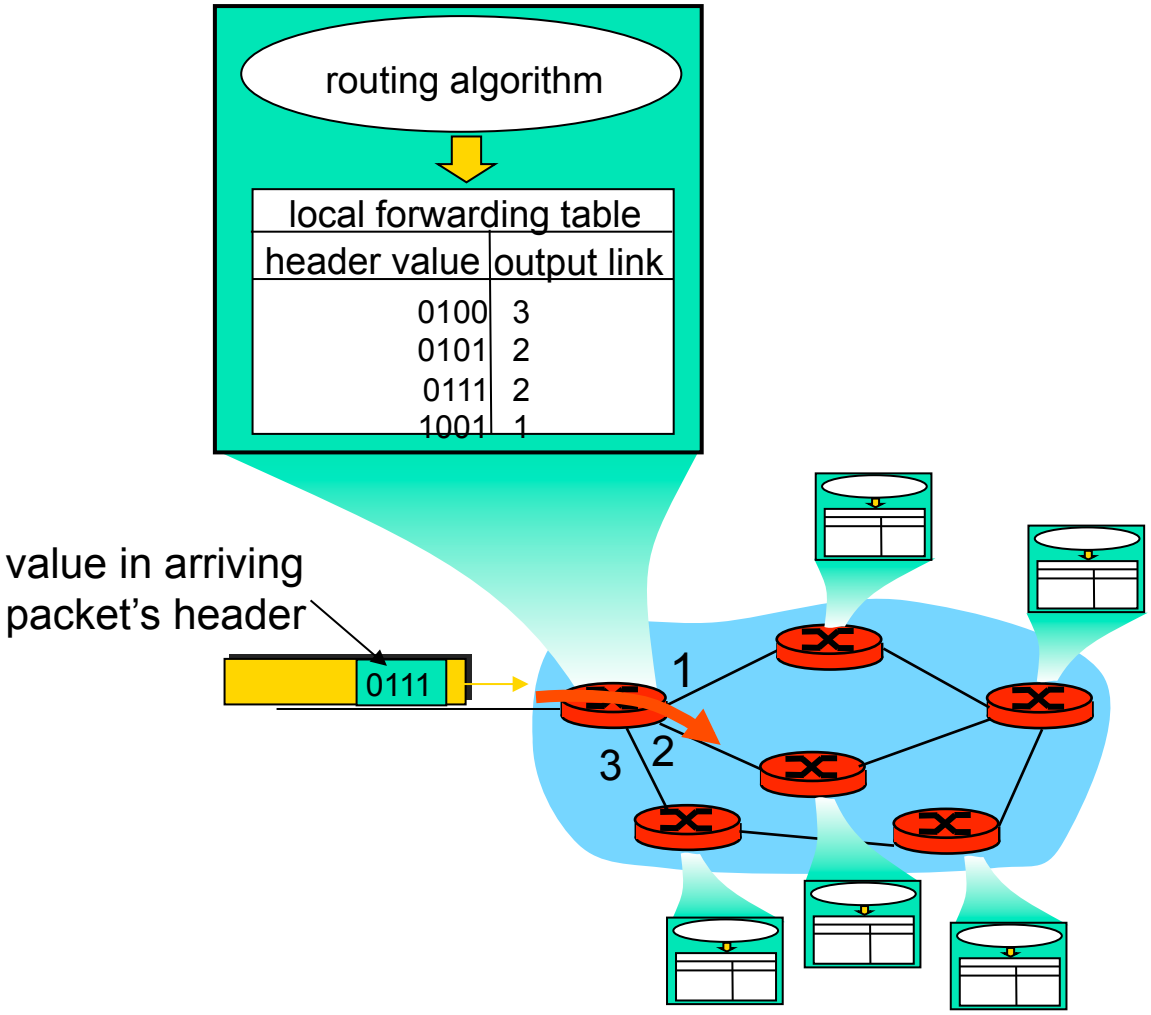
Dados dois ou mais computadores, seja qual for a sua ligação à rede, o nível rede assegura que os mesmos podem enviar pacotes uns aos outros. A noção de nó comutador de pacotes (*router*, nó ou "encaminhador") é central.

Funções executadas ao nível rede:

- **Escolha do caminho** tomado pelos pacotes da origem até ao destino (algoritmos de encaminhamento)
- **Switching** – (comutação) comutar os pacotes da interface de entrada para a de saída
- Call setup – (estabelecimento do circuito) algumas arquitecturas de rede requerem que os nós de comutação (*routers*) estabeleçam um circuito virtual antes de os *hosts* poderem comunicar



Algoritmo de encaminhamento e comutação

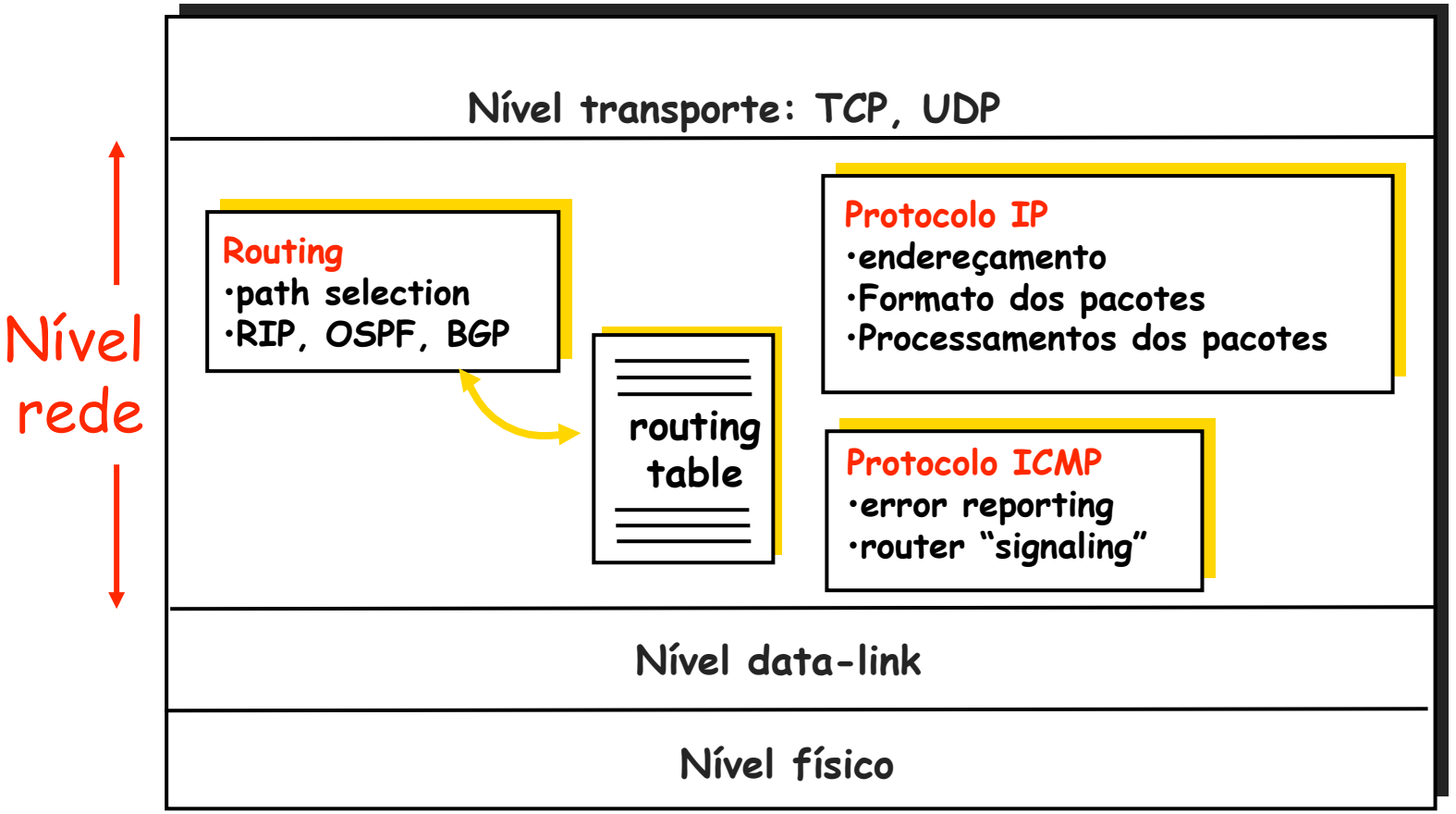


Organização do capítulo

- Objectivos do nível rede
- Estudo do protocolo IP e de alguns aspectos do funcionamento das redes IP
- Algoritmos de encaminhamento

O Nível rede na Internet

Funções do nível rede num computador ou num *router*



Endereçamento

Numa rede é necessário afectar endereços aos computadores de forma a poder designá-los ao nível rede. Esta forma de designação é concebida em função do desempenho e facilidade de implementação do encaminhamento.

Na Internet, cada computador dispõe geralmente de um endereço único e global. Exemplo: 192.86.45.15

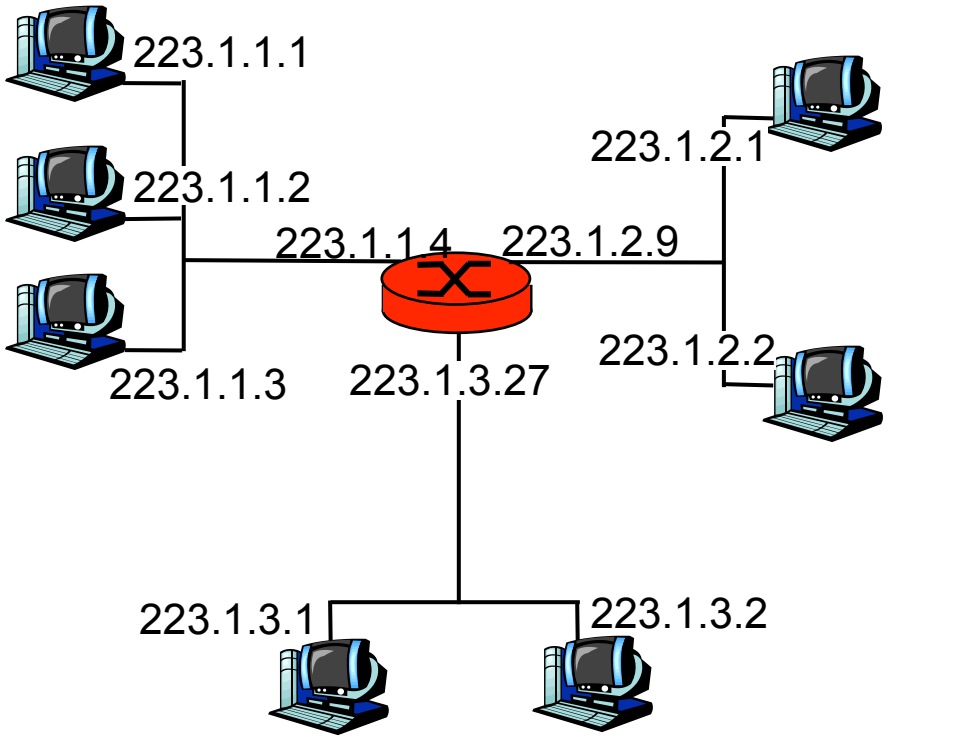
Esse endereço tem 32 bits e está dividido em duas parte: o número de rede do computador (o número da rede é único a nível mundial) e o número do computador dentro da sua rede (este número só é único dentro da rede do computador).

Para além do endereço, um computador tem também um nome (exemplo: "mail.di.fct.unl.pt"), esse nome é para ser usado pelos humanos, por oposição aos endereços que são formas de designação baixo nível.

Os endereços designam interfaces

Interface: **conexão entre um computador ou um *router* e um canal**

- **Os *routers* têm várias interfaces**
- **Em geral, um computador só tem uma interface**
- **Os endereços IP estão associados às interfaces e não aos computadores. Quando um computador só tem uma interface, o endereço do computador confunde-se com o da sua interface**

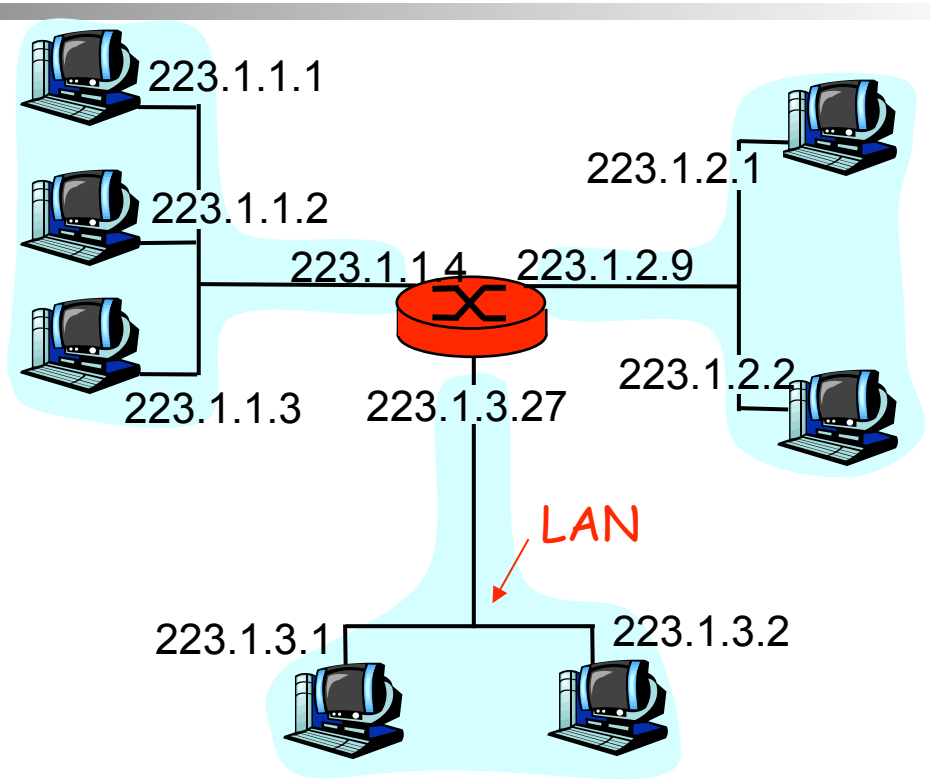


223.1.1.1 = 11011111 00000001 00000001 00000001

223 1 1 1

Estrutura dos endereços IP

- Os endereços IP têm duas partes:
  - Endereço rede (*high order bits* – bits à esquerda)
  - Endereço computador dentro da rede (*low order bits*)
- O que é uma rede IP do ponto de vista do endereçamento?
  - Conjunto de interfaces que partilham um canal e cujos endereços IP têm o mesmo prefixo
  - Dentro da mesma rede enviam-se pacotes sem intervenção de um *router*

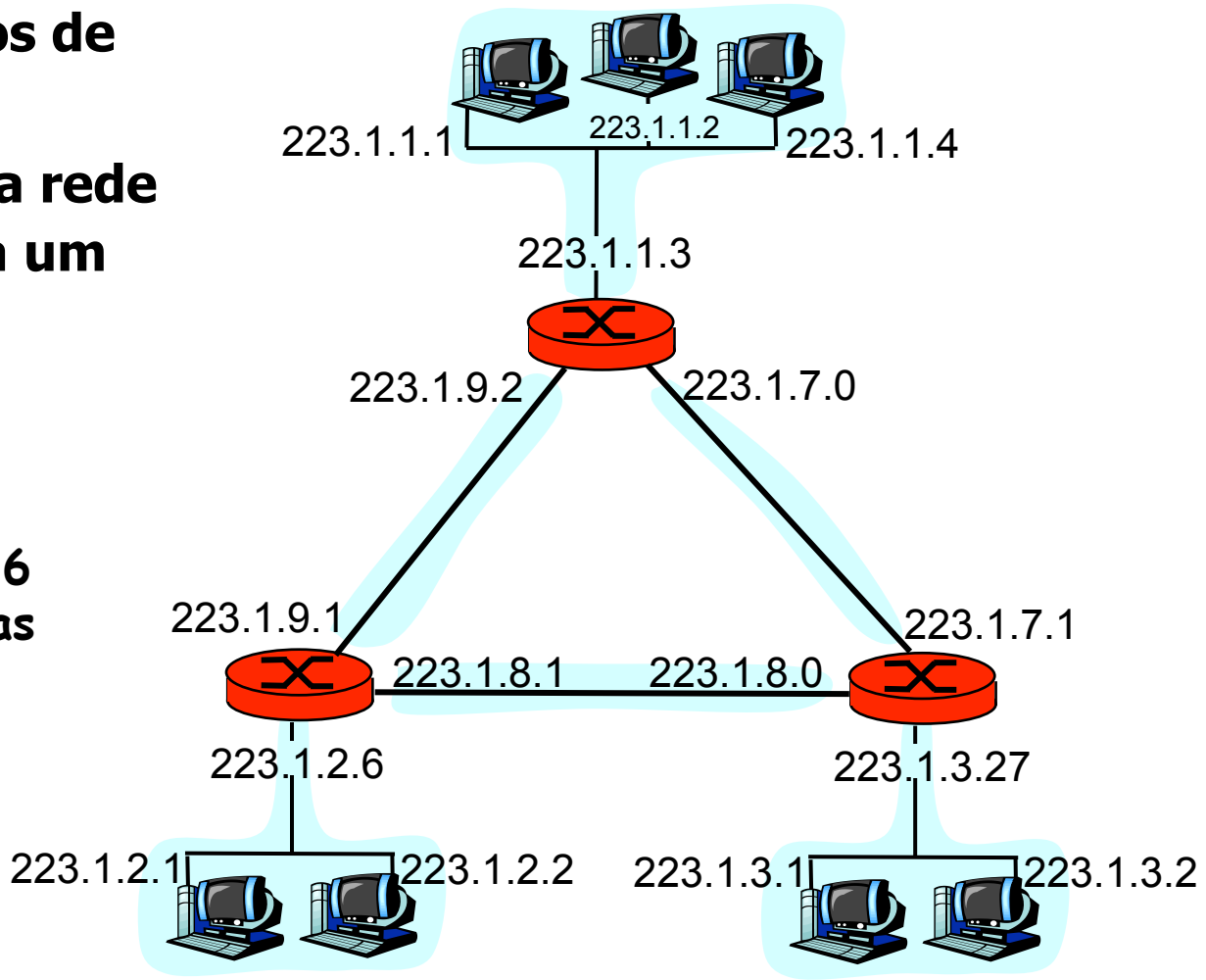


Esta rede tem três redes IP interligadas (neste exemplo, a parte rede do endereço tem sempre 24 bits, os primeiros 3 bytes)

Como encontrar as redes IP ?

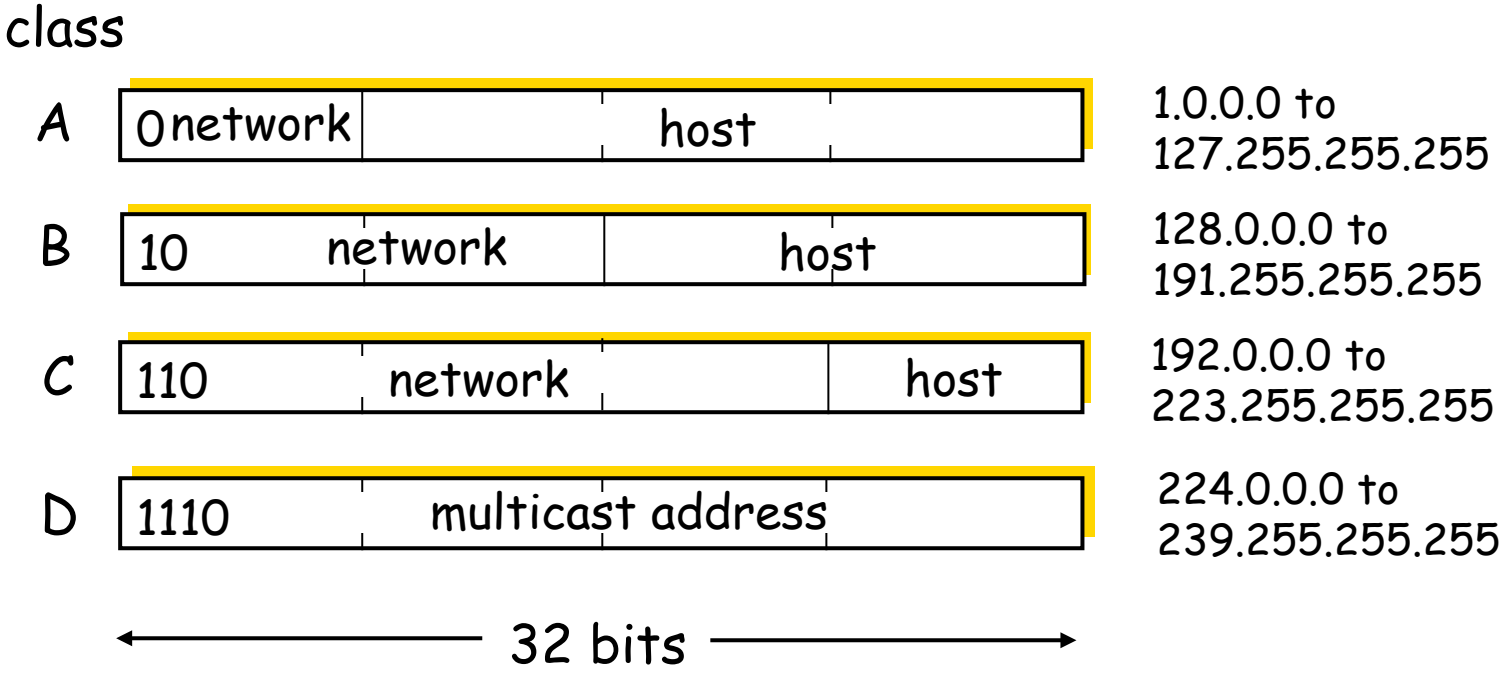
- **Isolar os prefixos de rede presentes**
- **Geralmente cada rede está associada a um canal**

Esta rede (internetwork) tem 6 redes IP interligadas



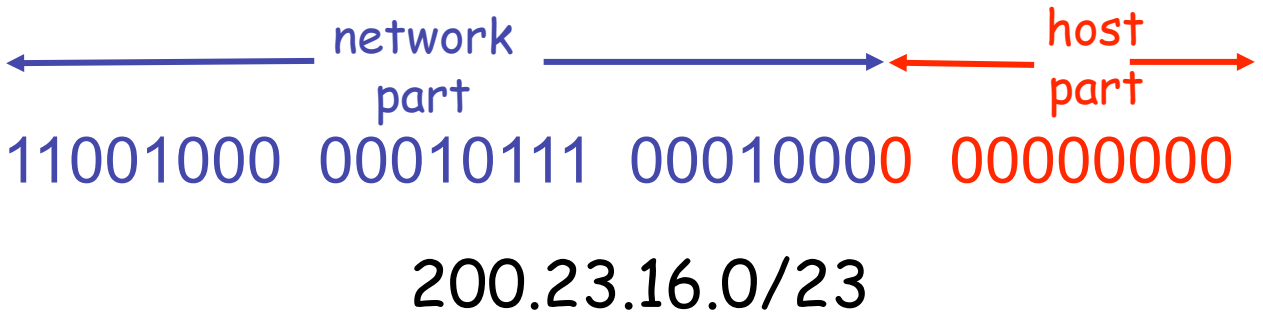
Gamas de endereços IP

Endereçamento baseado em classes ("class-full"):



Endereçamento IP: CIDR

- **Endereçamento com classes:**
  - **Conduzia a uma utilização ineficiente dos endereços, os quais constituem hoje em dia, um recurso escasso**
  - **Por exemplo, uma classe B permite cerca de 65000 endereços mas seria necessária sempre que uma classe C não chegasse**
- **CIDR: Classless InterDomain Routing**
  - **O prefixo rede tem uma dimensão arbitrária**
  - **Formato do endereço: a.b.c.d/x, onde x indica a dimensão do prefixo rede (número de bits deste prefixo)**



Máscaras dos endereços

- Quando se está a endereçar um computador não é necessário conhecer a sua rede pois o endereço de 32 bits é único
- No entanto, quando se faz encaminhamento, é necessário encontrar o endereço de rede, ou seja o número de bits do prefixo rede
- Para esse efeito, o endereço tem de ser conhecido pelo par a.b.c.d/X
- Outra forma de indicar X é indicar uma máscara, ou seja, uma palavra de 32 bits cujo AND lógico com o endereço IP extrai a parte rede
- Exemplo: se X = 24, a máscara é 255.255.255.0

Como são afectados os endereços IP ?

Parte computador, pode ser de duas formas:

- **Manualmente, isto é, codificado num ficheiro de parametrização do computador**

- **DHCP: Dynamic Host Configuration Protocol: dinamicamente através de um protocolo (tipo "plug and play"):**
  - **O computador faz um broadcast: mensagem "DHCP discover"**
  - **O servidor DHCP responde com uma mensagem: "DHCP offer"**
  - **O host solicita um endereço IP: mensagem "DHCP request"**
  - **O servidor DHCP responde com o endereço: "DHCP ack"**

Como são afectados os endereços IP ?

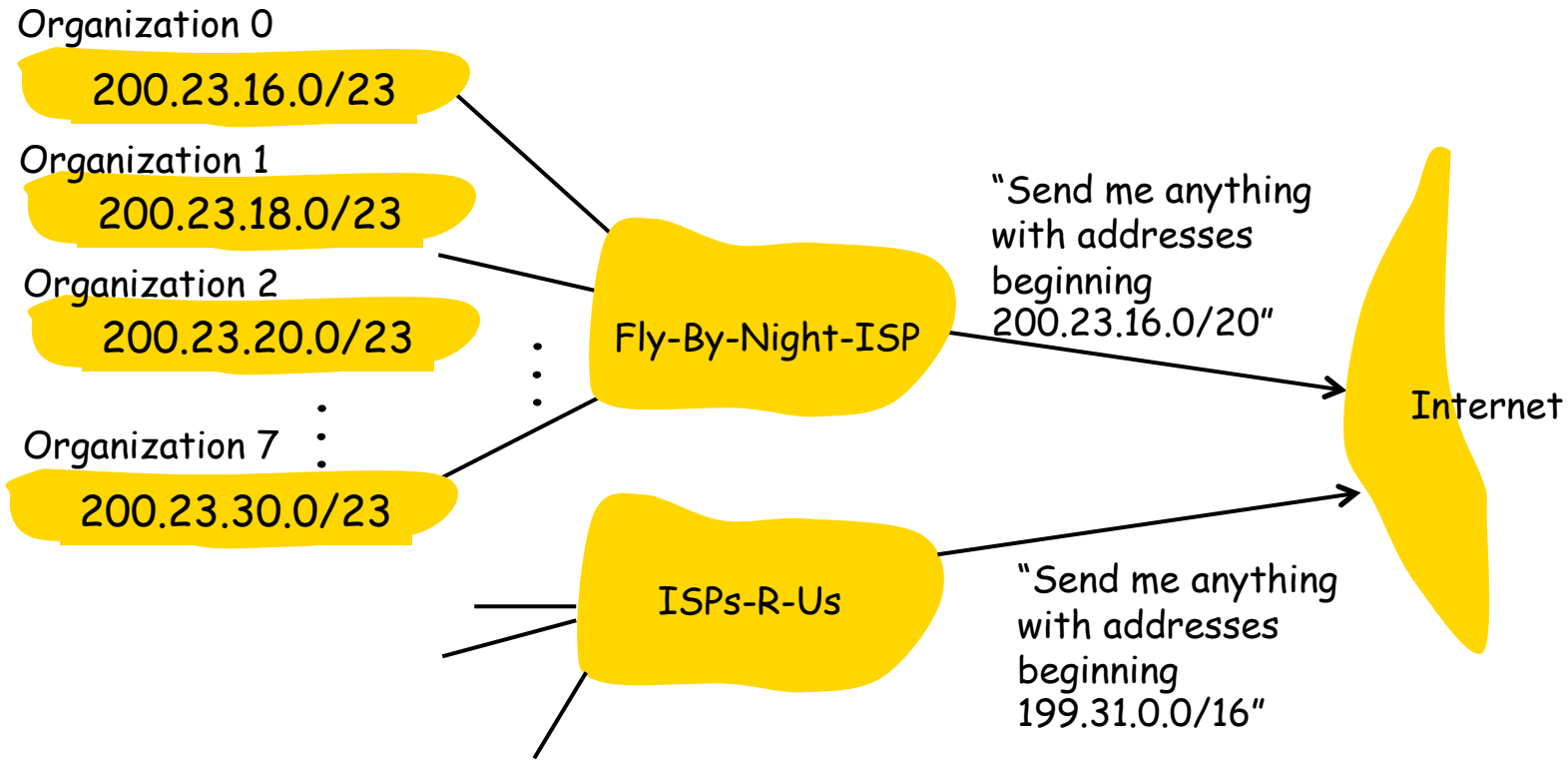
Parte rede:

- É geralmente afectada pelo ISP aos seus clientes

ISP's block	11001000 00010111 00010000	00000000	200.23.16.0/20
Organization 0	11001000 00010111 00010000	00000000	200.23.16.0/23
Organization 1	11001000 00010111 00010010	00000000	200.23.18.0/23
Organization 2	11001000 00010111 00010100	00000000	200.23.20.0/23
...	…..		….
….			
Organization 7	11001000 00010111 00011110	00000000	200.23.30.0/23

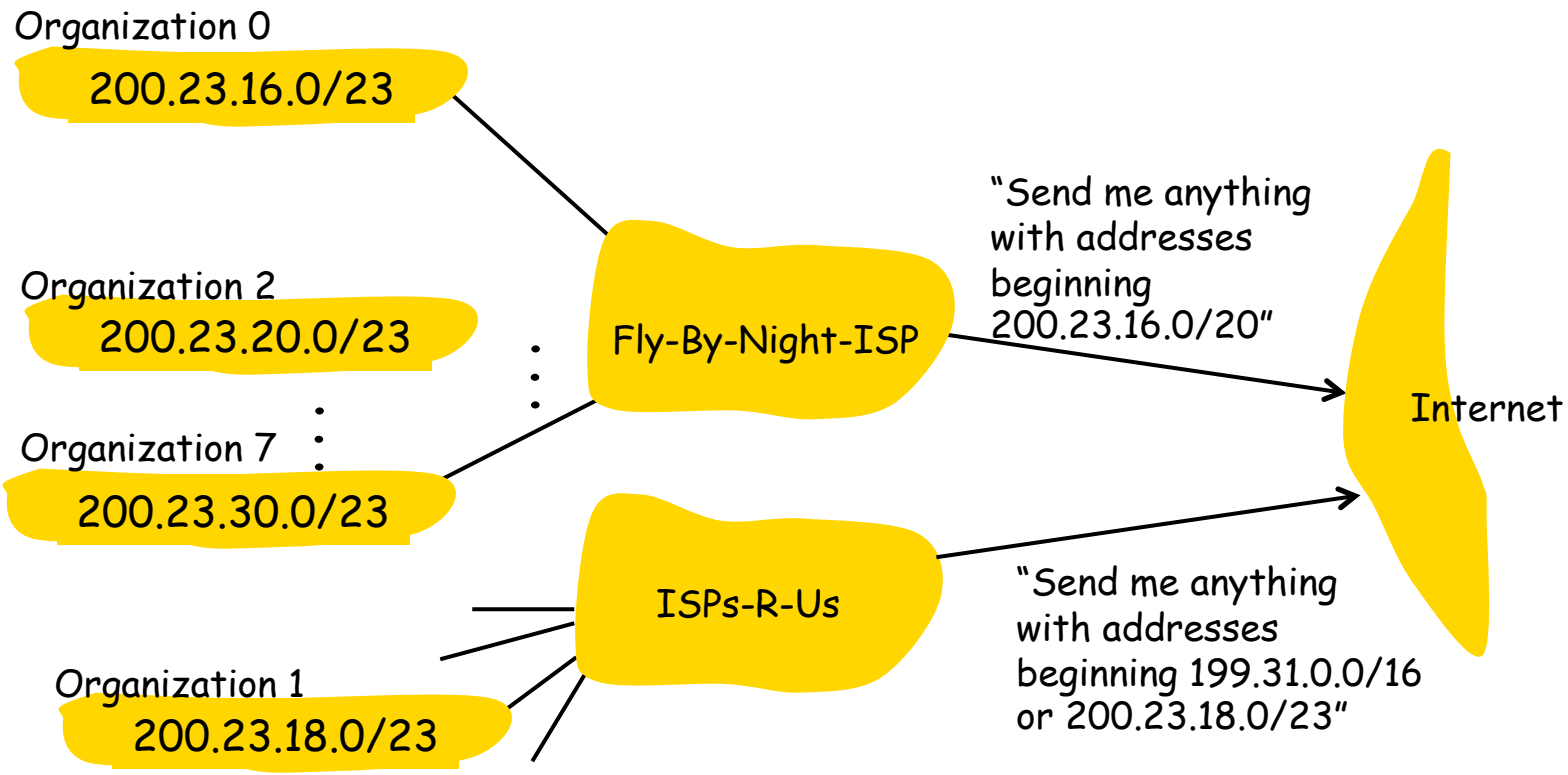
Endereçamento hierárquico: agregação de rotas

Ao nível Inter-AS os ISPs não anunciam separadamente as redes dos seus clientes. Anunciam apenas os seus blocos de agregados de redes o que optimiza os protocolos de encaminhamento



The longest prefix is the best

O ISPs-R-Us tem uma rota mais específica para a Organization 1. Esta regra diminui a eficácia do routing mas permite maior flexibilidade.



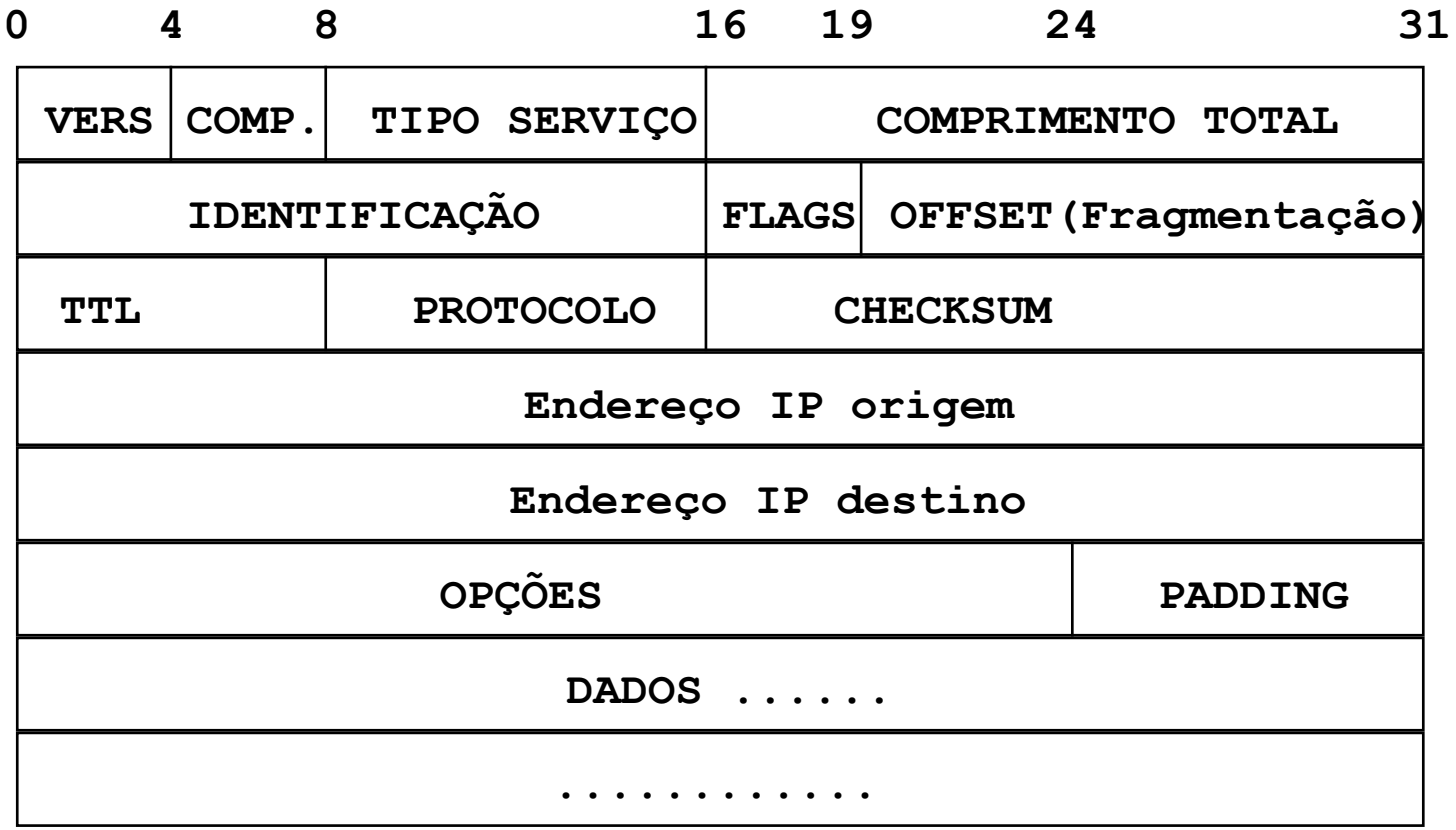
Como é que um ISP obtém um bloco de endereços ?

ICANN: Internet Corporation for Assigned Names and Numbers

- **Afecta endereços IP**
- **Gere o topo da hierarquia DNS**
- **Resolve as disputas sobre nomes de domínios**

- **Na Europa é o RIPE (Réseaux IP Europeans) que gere a afectação dos endereços IP**

Datagrama IP



Significado de alguns campos

Versão - vamos na versão 4, com a 6 a ser cada vez mais presente

COMP - Comprimento do cabeçalho (em palavras de 32 bits)

O tipo de serviço (só mais recentemente começa a ser usado)

O comprimento total é em bytes (datagrama máximo = 64 Kbytes)

TTL - Time to live (segurança contra erros de encaminhamento)

PROTOCOLO - Suporta a *desmultiplexagem* por protocolos

CHECKSUM – Protecção contra pacotes corrompidos

Opções: “source routing”, “record routing”, ...

Encaminhamento

Cada computador que necessita de fazer encaminhamento, tem que possuir tabelas de encaminhamento ou routing

As tabelas de encaminhamento relacionam endereços de rede IP destino com interfaces ou endereços IP de routers

Quando se faz encaminhamento, só entra em jogo a parte do endereço IP que diz respeito à rede, não o endereço completo

O encaminhamento difere de acordo com as tabelas de encaminhamento. Assim, existem três tipos de encaminhamento:

- Encaminhamento Directo
- Encaminhamento Indirecto
- Encaminhamento por Defeito

Formas de encaminhamento

Encaminhamento Directo :

- Ocorre quando tanto o endereço IP do host como o endereço IP destino, estão na mesma rede física. Por isso, os pacotes IP vão directamente da origem ao destino pois o router ou host tem uma interface nessa rede

Encaminhamento Indirecto

- Ocorre quando tanto o endereço IP origem como o destino, não se encontram na mesma rede física. A única forma de encaminhar os pacotes é entregá-los a um ou mais routers. O endereço IP do primeiro router (next hop) deve pois ser conhecido.

Encaminhamento por defeito :

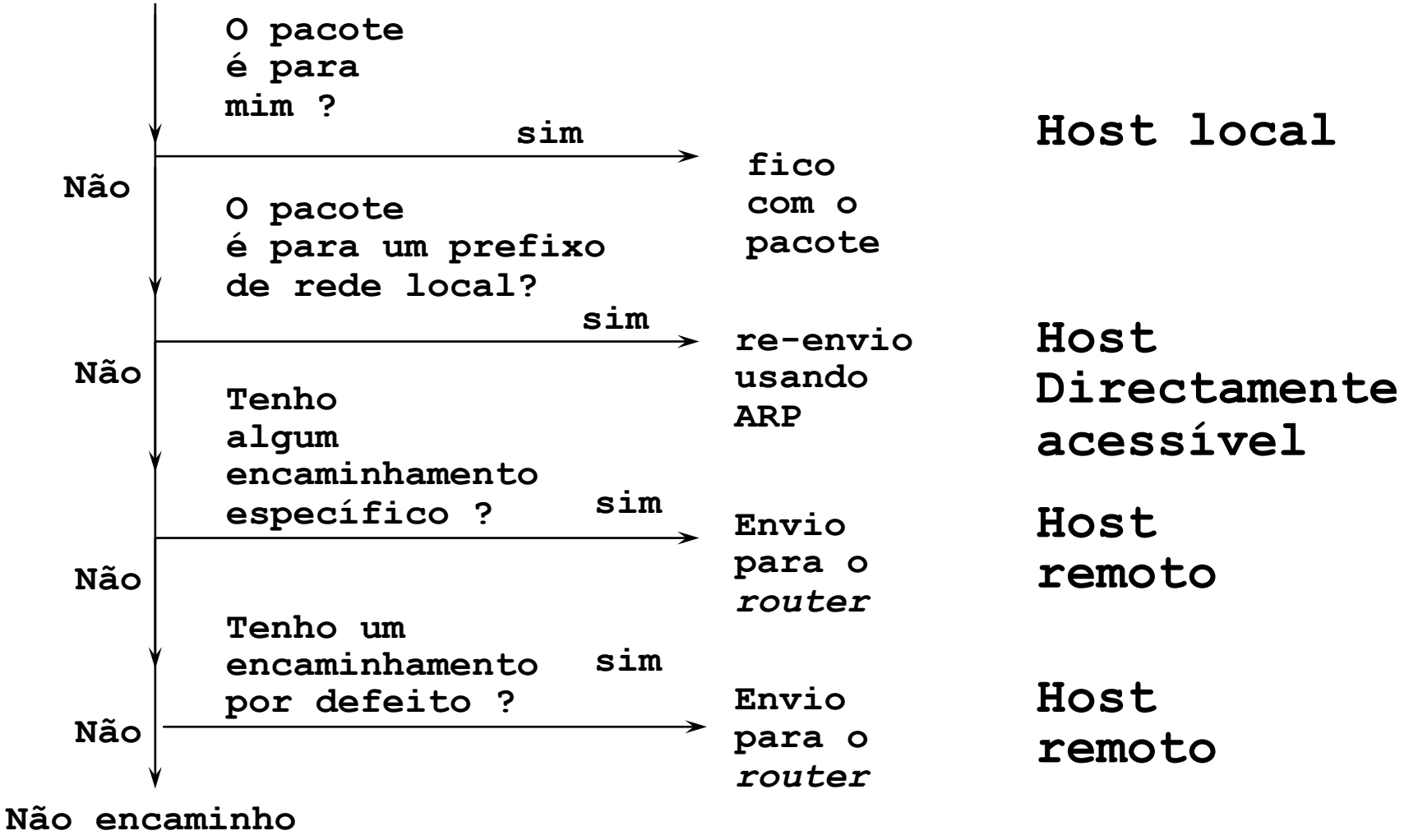
- Ocorre quando na tabela de routing não existe nenhuma informação sobre encaminhamento directo ou indirecto e um caminho por defeito é conhecido.

Tabelas de encaminhamento

Acessível num computador através dos comandos:
netstat -r ou route por exemplo

Endereço destino	Máscara	Next hop
34.1.0.0	255.255.0.0 ou /16	54.34.23.12
38.3.0.0	255.255.0.0 ou /16	54.34.23.12
192.182.1.0	255.255.255.0 ou /24	54.34.12.65
54.34.23.0	255.255.255.0 ou /24	directo (e.g. Eth0)
54.34.12.0	255.255.255.0 ou /24	directo (e.g. Eth1)
0.0.0.0	0.0.0.0 ou /0	54.34.12.65

Tratamento do pacote



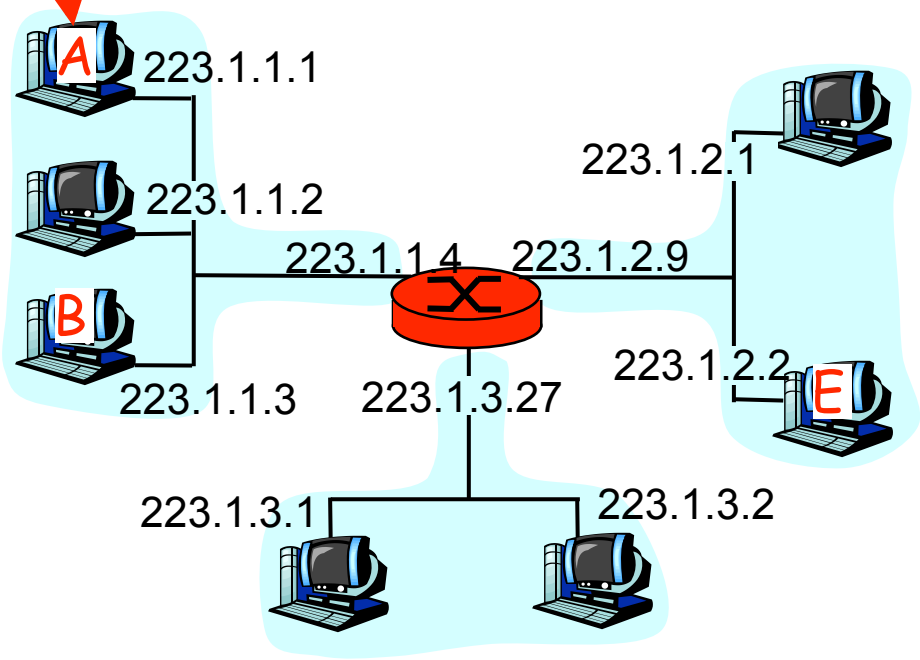
Rede exemplo

Datagrama IP:

misc fields	source IP addr	dest IP addr	data

routing table in A

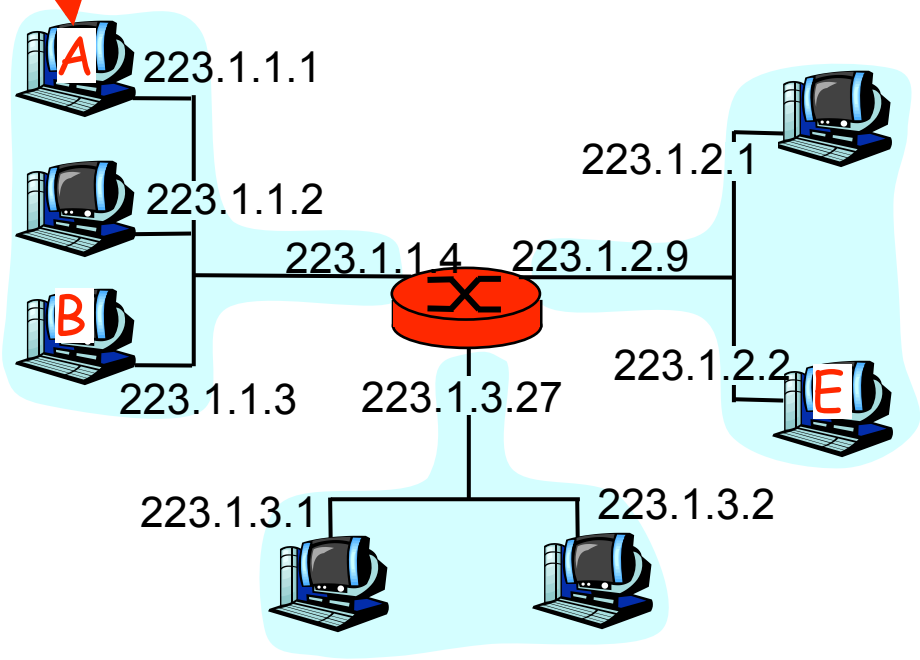
Dest. Net.	next router	Nhops
223.1.1	eth0	1
223.1.2	223.1.1.4	2
223.1.3	223.1.1.4	2



Encaminhamento directo

misc fields	223.1.1.1	223.1.1.3	data

Dest. Net.	next router	Nhops
223.1.1	eth0	1
223.1.2	223.1.1.4	2
223.1.3	223.1.1.4	2



Datagrama com origem em A e destinado a B:

- **Extrair o endereço de rede: 223.1.1**
- **B está na mesma rede**
- **O nível data-link envia directamente o datagrama de A para B**

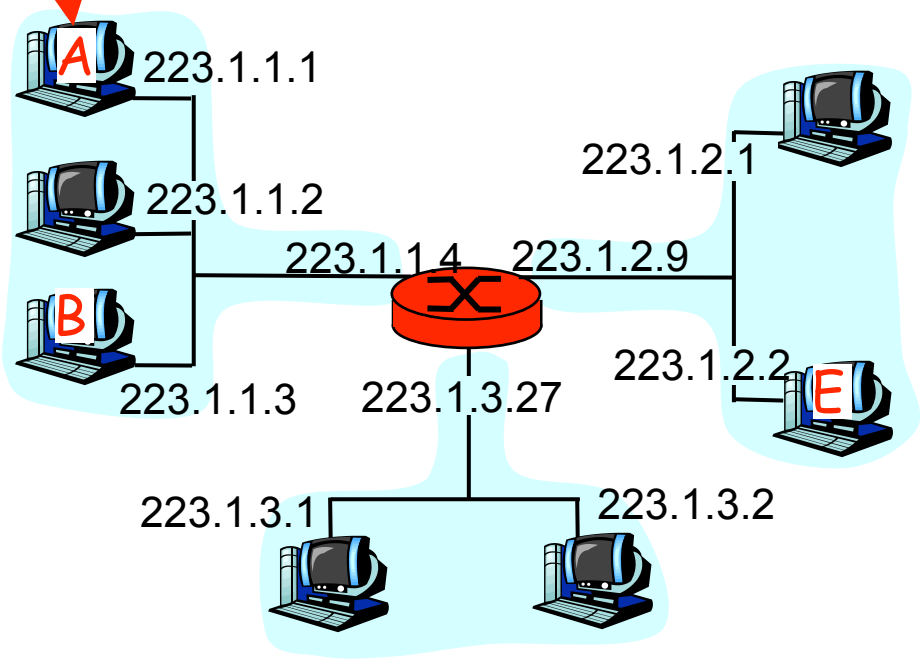
Encaminhamento indirecto

misc fields	223.1.1.1	223.1.2.2	data

Datagrama com origem em A e destinado a E:

- **Extrair endereço de rede – 223.1.2**
- **Está noutra rede - indirecto**
- **A tabela de routing diz para enviar via 223.1.1.4**
- **O nível data-link sabe enviar para esse endereço visto que o mesmo está na mesma rede de A**
- **O datagrama chega ao router e o processo continua**

Dest. Net.	next router	Nhops
223.1.1	eth0	1
223.1.2	223.1.1.4	2
223.1.3	223.1.1.4	2



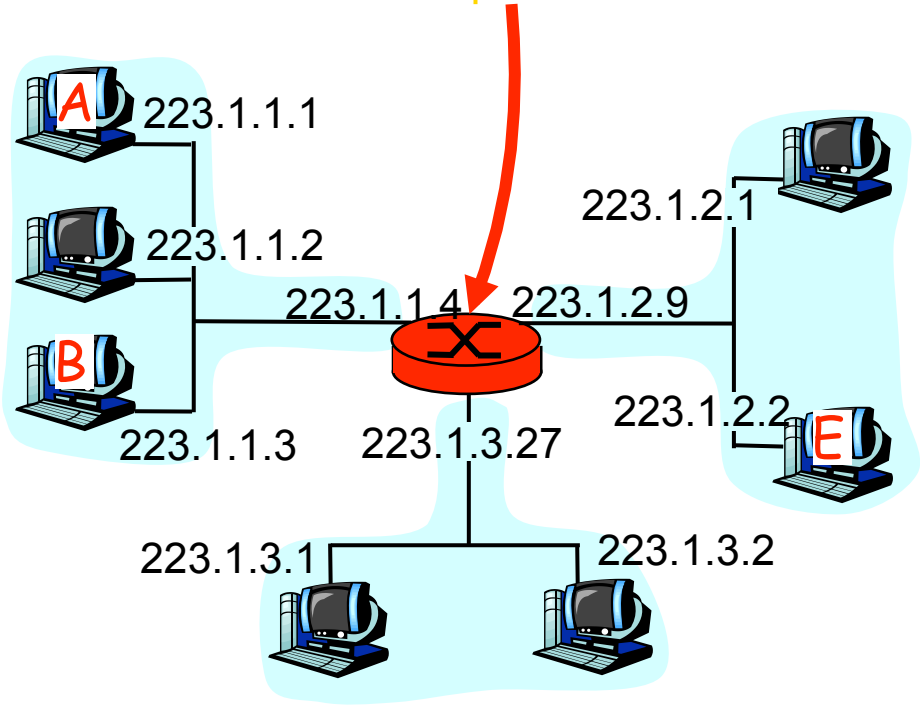
Continuação

misc fields	223.1.1.1	223.1.2.3	data

O datagrama chegou a 223.1.1.4 com destino a 223.1.2.3

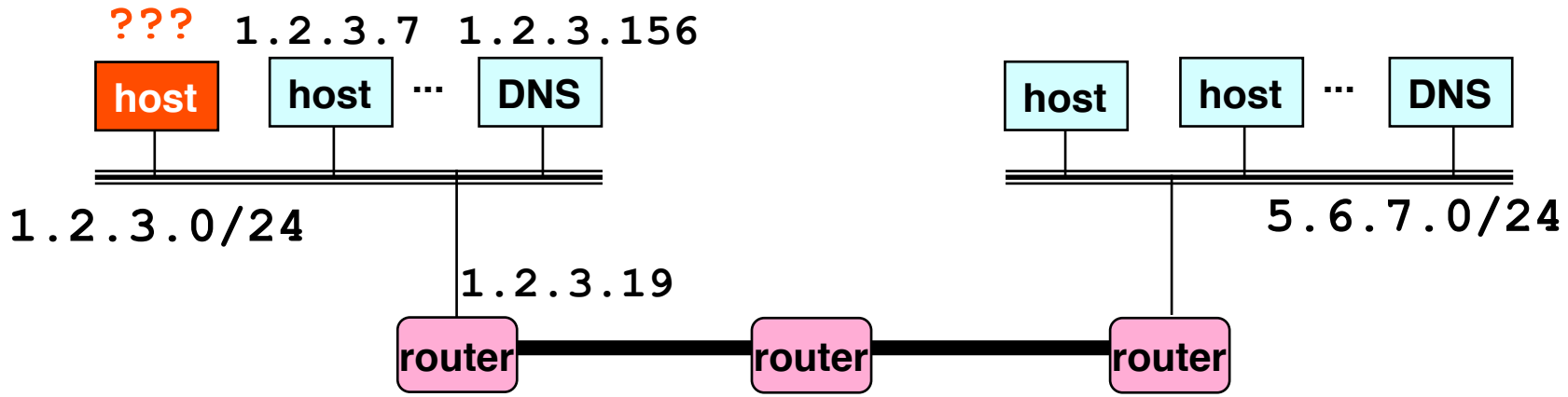
- **Extrair o endereço de rede**
- **Está numa rede directamente acessível via a interface 223.1.2.9**
- **O nível data-link dessa interface sabe como enviar directamente para o destino: 223.1.2.9**
- **O datagrama chega ao destino - 223.1.2.2**

Dest. network	next router	Nhops	interface
223.1.1	eth0	1	223.1.1.4
223.1.2	eth1	1	223.1.2.9
223.1.3	eth2	1	223.1.3.27



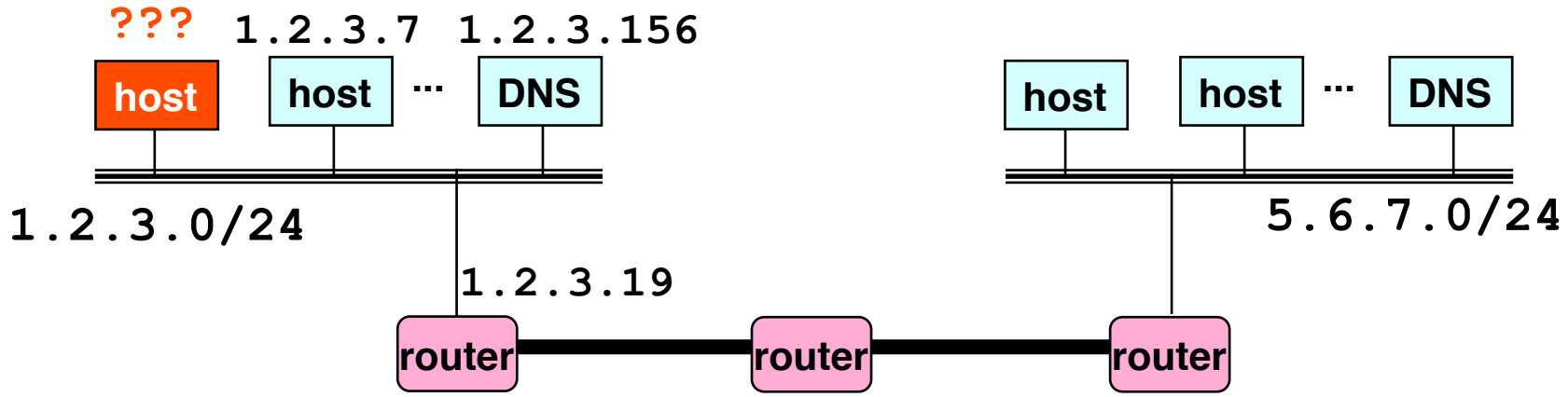
Como fazer "Bootstrap" a um End Host - DHCP and ARP

- Que servidor DNS local usar?
- Qual o meu endereço IP?
- Como enviar pacotes para destinos remotos?
- Como assegurar que os pacotes remotos me chegam ?



Tem de se evitar a configuração manual

- **Dynamic Host Configuration Protocol (DHCP)**
  - Os hosts aprendem como enviar pacotes para outros hosts remotos
  - Isto é, adquirem o IP address, DNS servers, e router por defeito
- **Address Resolution Protocol (ARP)**
  - Os outros hosts passam a saber como enviar pacotes para mim
  - Isto é, passamos a ter uma relação entre endereços IP e endereços aas placas



Ideias Base de Ambos os Protocolos

- ***Broadcasting:* quando tiver dúvidas pergunte alto!**
  - Envie um Broadcast para todos os hosts da rede local
  - … mas só quando não sabe já o que pretende
- ***Caching:* guarde o que aprendeu por algum tempo**
  - Guarde o que aprendeu para não repetir o processo
  - Lembre-se do seu endereço e do dos outros hosts
- ***Soft state:* … mas mais tarde ou mais cedo esqueça-se do passado**
  - Associe um time-to-live (TTL) à informação
  - … e refresque ou suprima a informação
  - Isto é fundamental para se adaptar a modificações inesperadas

Precisamos de outro tipo de endereços

- As LANs foram desenhadas para protocolos arbitrários
  - Não só para IP e a Internet
- Usar endereços IP exigiria reconfigurações constantes
  - Usar endereços IP exigiria reconfigurações sempre que a placa mudasse de computador ou este arrancasse de novo
- Broadcasting constante para todas as placas é "caro"
  - Pois exigiria que cada host analisasse cada pacote

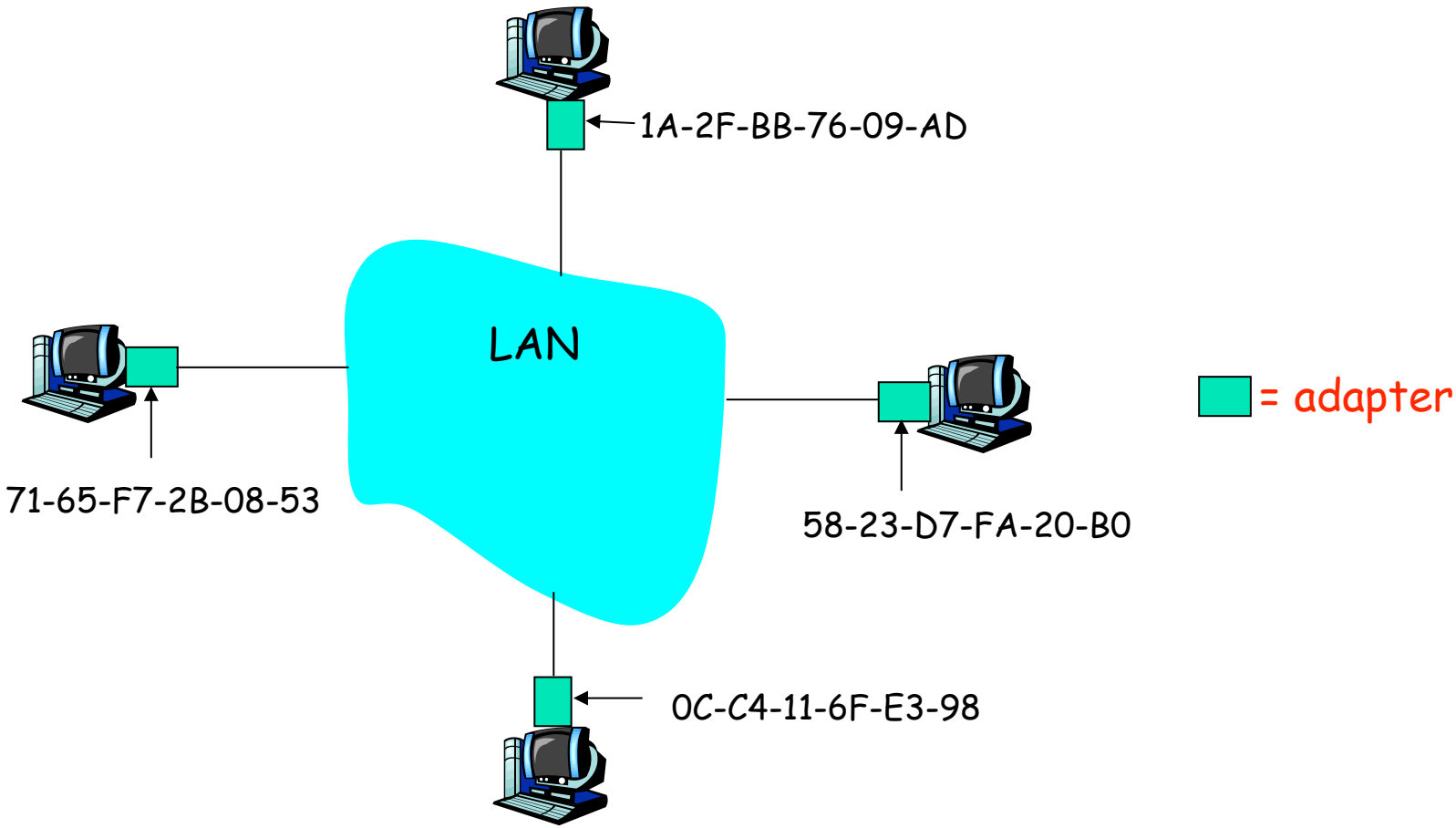
Assim temos os chamados

Medium Access Control (MAC) addresses

MAC Address vs. IP Address

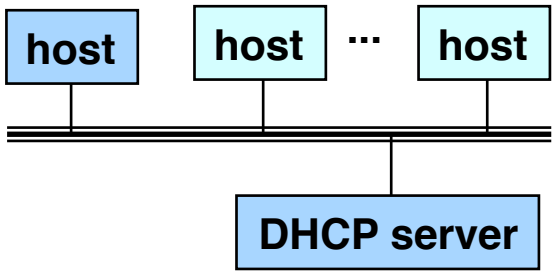
- **Endereços MA — MAC addresses**
  - Gravados numa memória quando a placa é construída
  - Como um número de bilhete de identidade
  - Espaço de nomes horizontal (*flat name space*) 48 bits (e.g., 00-0E-9B-6E-49-76)
  - Portáteis pois podem ficar com o host mesmo que este se mude
  - Usados para enviar pacotes na mesma rede local
- **Endereços IP — IP addresses**
  - Configurados ou aprendidos dinamicamente
  - Como o endereço postal
  - Espaço hierárquico de nomes de 32 bits (e.g., 12.178.66.9)
  - Não é portátil pois depende do local onde o host está
  - Usado para encaminhamento na rede IP

MAC Addresses numa LAN



O Problema de Bootstrapping

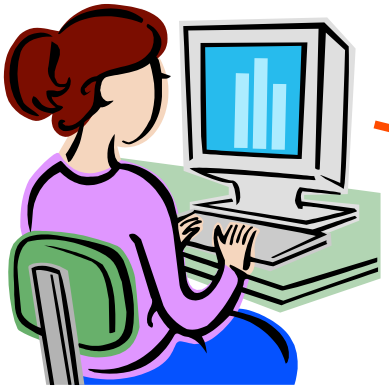
- **O Host ainda não tem endereço IP**
  - Como enviar pacotes se não pode indicar o endereço origem?
- **O host não sabe a quem pedir um endereço IP nem que outros endereços "andam por aí"**
- **Solução: descobrir um servidor que possa dar uma ajuda**
  - Broadcast a server-discovery message
  - Server sends a reply offering an address



Resposta de um servidor DHCP

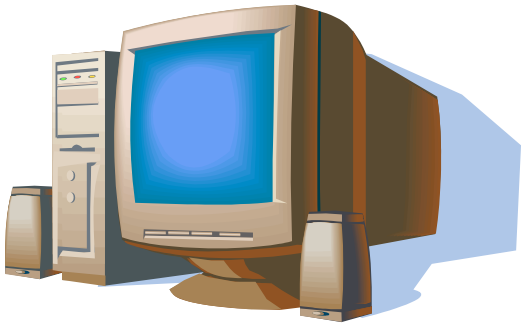
- DHCP "offer message"
  - Parâmetros de configuração (proposed IP address, mask, gateway router, DNS server, ...)
  - Lease time (o tempo durante o qual esta informação é válida)
- A resposta pode vir de mas do que um servidor
  - Protege contra um crash de um servidor único
  - Os vários servidores respondem com uma oferta
  - O cliente decide qual deve aceitar
- Aceitação de uma das ofertas
  - O cliente envia uma mensagem DHCP com os parâmetros aceites
  - O servidor confirma com um ACK
  - ... e os outros servidores verificam que não foram escolhidos

DHCP - Dynamic Host Configuration Protocol



arriving client

DHCP discover (broadcast)



DHCP server 233.1.2.5

DHCP offer

DHCP request (broadcast)

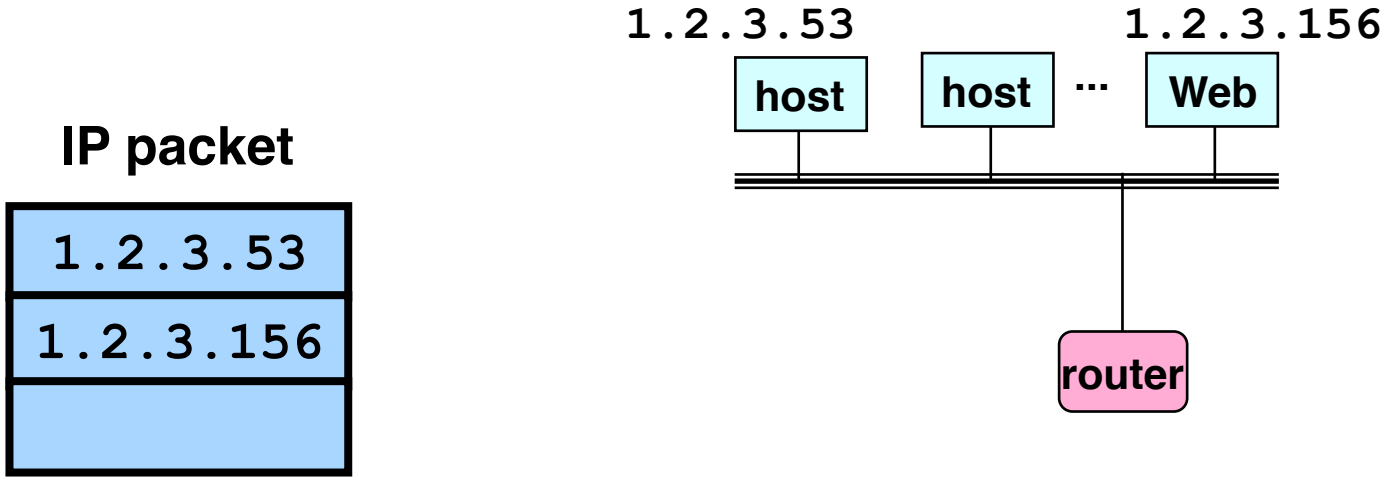
DHCP ACK

Soft State: Refresh or Forget

- **Porque é necessário o "lease time"?**
  - O cliente pode libertar o endereço (DHCP RELEASE)
    - Por exemplo "ipconfig /release" no DOS SHELL prompt
    - Trata-se um shutdown ordeiro do computador
  - Mas nem sempre isso acontece
    - O host tem um crash
    - ou o software do cliente tem bugs
  - E o endereço ficaria afectado para sempre

- **Performance trade-offs**
  - Pequeno "lease time": os endereços inactivos são rapidamente devolvidos
  - Longo "lease time": evita frequentes renovações

Como se enviam pacotes por um link?



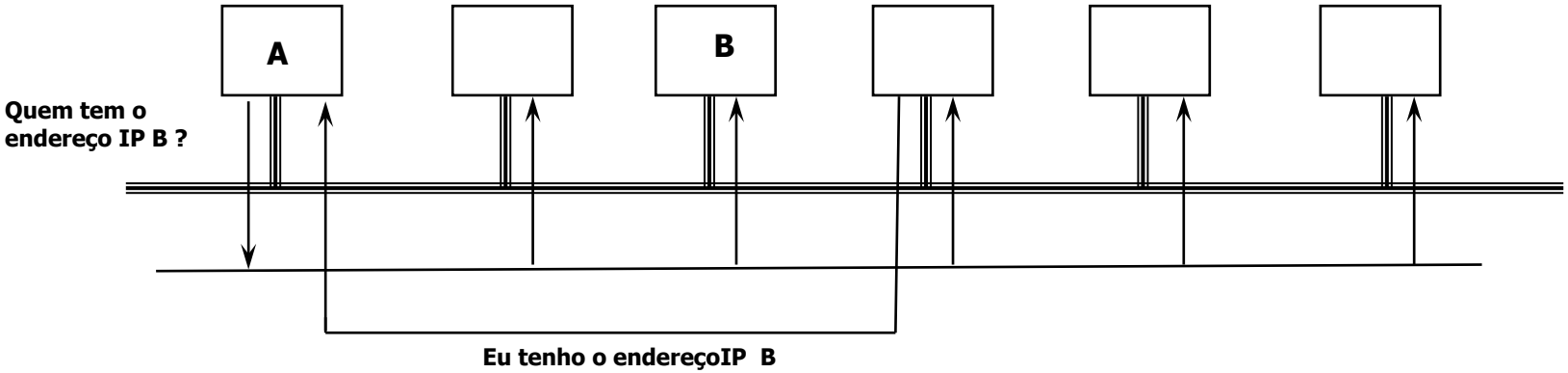
- As placas Ethernet só conhecem endereços MAC
  - É necessário traduzir o endereço IP de destino num endereço MAC
  - E encapsular o pacote IP num link-level frame

Address Resolution Protocol (ARP) Table

- Cada nó tem uma tabela ARP
  - Com pares (IP address, MAC address)
- E consulta a tabela antes de enviar um pacote
  - Se encontrar o endereço MAC correspondente ao endereço IP de destino
  - Encapsula o pacote IP num frame e envia-o
- Mas, se o endereço IP não está na tabela ?
  - O emissor envia um broadcast: "Who has IP address 1.2.3.156?"
  - O receptor responde: "MAC address 58-23-D7-FA-20-B0"
  - O emissor coloca esses dados na tabela ARP
  - E procede como acima
- Tudo se passa automaticamente

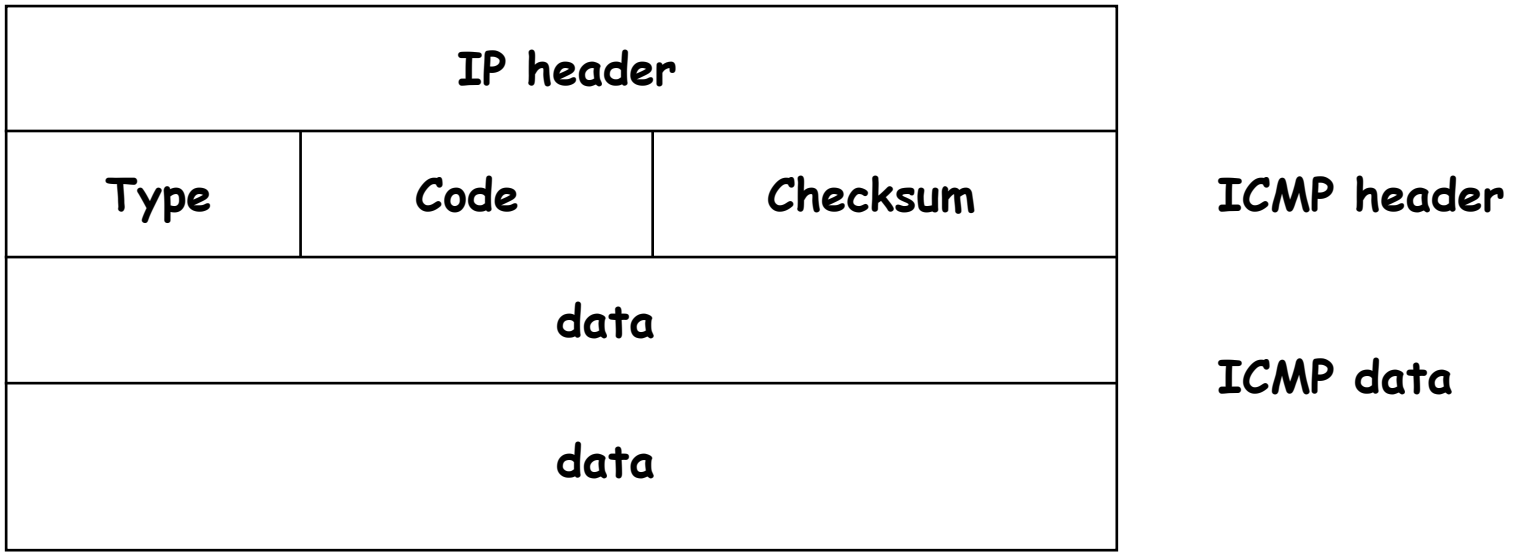
Protocolo ARP

Numa rede com suporte de broadcast, uma única interface dá acesso a todos os hosts da mesma rede. Para determinar o endereço físico do host de destino o software da interface ao nível data-link usa o protocolo ARP (Address Resolution Protocol) que é suportado directamente através de frames especiais (por exemplo nas redes ethernet)



Existe um protocolo simétrico (RARP - reverse address resolution protocol) que permite a um host determinar o seu endereço IP (na inicialização) a partir do seu endereço ethernet – hoje em dia foi substituído pelo DHCP

ICMP - Internet Control Message Protocol



- Utilizado para transmitir informação do nível rede
  - erros: *unreachable host, network, port, protocol, ...*
  - *echo request/reply* (utilizado pelo ping)
- *ICMP message*: tipo, código e os primeiros 8 bytes do datagrama IP que provocou o envio do ICMP

Exemplos de mensagens ICMP

Type	Code	description
0	0	echo reply (ping)
3	0	dest. network unreachable
3	1	dest host unreachable
3	2	dest protocol unreachable
3	3	dest port unreachable
3	6	dest network unknown
3	7	dest host unknown
4	0	source quench (congestion control - not used)
8	0	echo request (ping)
9	0	route advertisement
10	0	router discovery
11	0	TTL expired
12	0	bad IP header

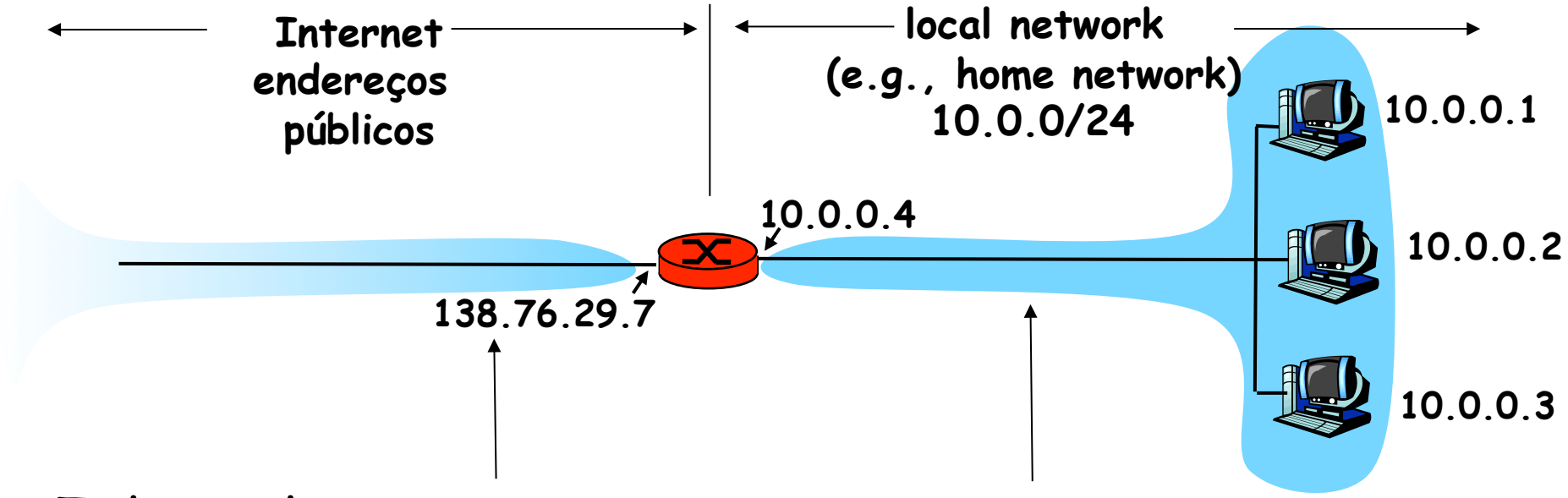
Traceroute e ICMP

- A fonte envia uma sequência de segmentos UDP para o destino:
  - Primeiro com TTL =1
  - Segundo com TTL=2, etc.
  - Número de porta arbitrário
- Quando o enésimo segmento chega ao enésimo router:
  - O router suprime o segmento
  - e envia para a origem uma mensagem ICMP (type 11, code 0)
  - As mensagens têm por endereço IP origem o enésimo router
- Quando a mensagem ICMP chega à origem pode-se calcular o RTT
- Em cada passo enviam-se 3 segmentos

Terminação

- O segmento UDP chega ao destino
- O destino envia uma mesnagem ICMP "host unreachable" (type 3, code 3)
- O número máximo de testes é alcançado.

NAT: Network Address Translation



Todos os datagramas que saem da rede local têm o mesmo endereço público: 138.76.29.7, mas diferentes números de porta

Os datagramas com origem ou destino nesta rede têm o endereço 10.0.0/24
Estes endereços dizem-se privados

Continuação

- **Motivação:** toda a rede interna só usa um único endereço público
  - Assim, o número de endereços afectado pelo ISP é menor
  - A configuração e a alteração da rede interna (privada) não é visível do exterior
  - Pode-se mudar de ISP sem modificar a configuração da rede interna
  - A rede interna não é visível no exterior o que constituí uma barreira de segurança suplementar

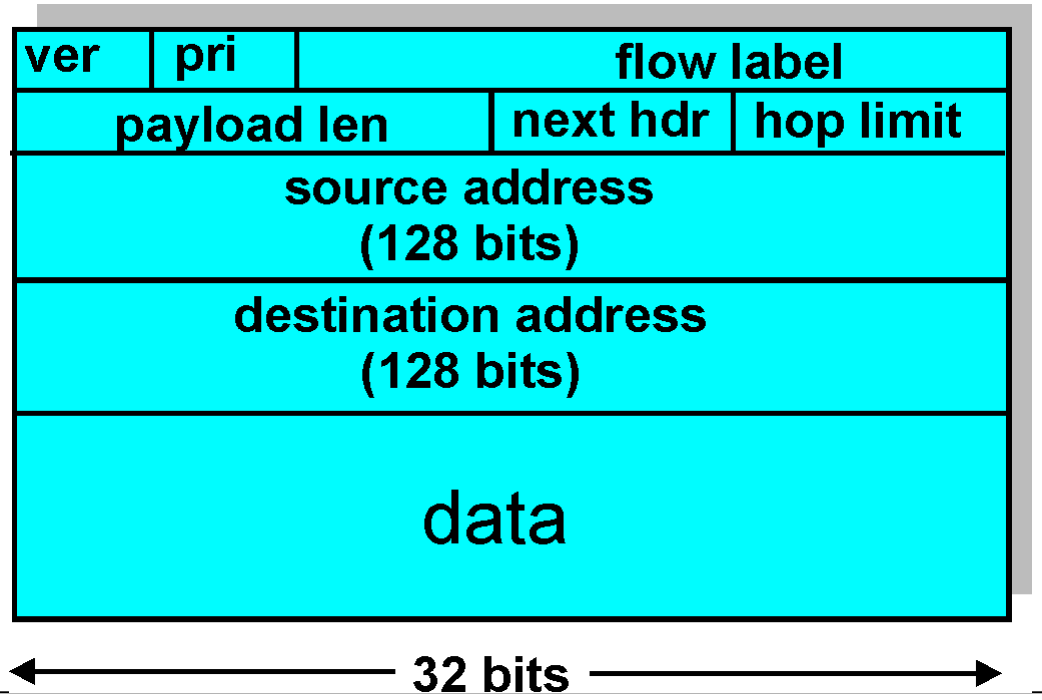
- Mas criam-se dificuldades suplementares a certas aplicações e não é fácil fornecer serviços na Internet pública

IP versão 6 - IPv6

- **Motivação inicial:** os endereços de 32 bits estão "em vias" de se esgotarem
- **Motivações suplementares:**
  - Melhorar o tempo de processamento do cabeçalho
  - Introduzir modificações para facilitar o processamento da qualidade de serviço - QoS
  - Novo endereço "anycast": encaminhar para o "melhor" de vários servidores replicados
  - Tornar obrigatório o suporte da mobilidade e da segurança
- **Formato do datagrama IPv6:**
  - Cabeçalho de tamanho fixo de 40 bytes
  - A fragmentação pelos *routers* foi suprimida

Cabeçalho do pacote IPv6

Prioridade prioridade relativa dos datagramas do mesmo flow
Flow Label: identifica datagramas do mesmo fluxo ("flow")
(o conceito de "fluxo" é flexível).
Next header: identifica um cabeçalho suplementar de um nível superior



Outras modificações do IPv4

- *Checksum*: foi removido para facilitar o processamento
- *Options:* continuam a ser permitidas mas através de um cabeçalho suplementar indicado pelo "Next Header field"
- *ICMPv6:* nova versão de ICMP
  - Tipos de mensagens adicionais, e.g. "Packet Too Big"
  - Funções para facilitar a parametrização das máquinas, etc…

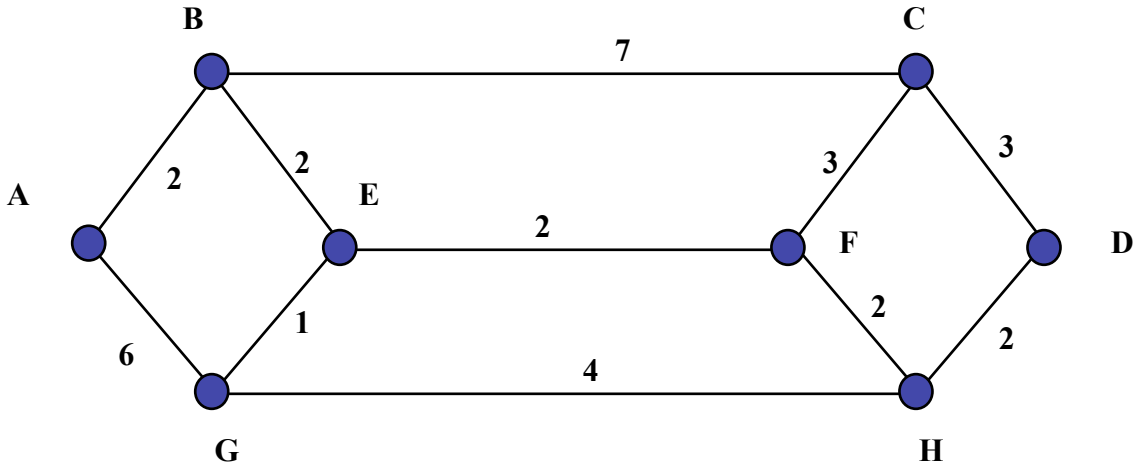
Organização do capítulo

- Objectivos do nível rede
- Estudo do protocolo IP e de alguns aspectos do funcionamento das redes IP
- Algoritmos de encaminhamento

Caracterização de um processo de encaminhamento

- **Correcção** - os pacotes chegam ao destino
- **Resiste à evolução da rede** - acompanha a escala da rede
- **Adapta-se dinamicamente** às variações da topologia, do estado dos canais, etc.
- **Equitativo** ("*fair*") para os diferentes computadores (origem e destino dos pacotes)
- **Óptimo** - minimiza o tempo de trânsito para a maioria dos percursos ou o rendimento global da rede
- **Simples** - não tem complexidade inútil, para minimizar o seu custo e maximizar a fiabilidade e a escala de aplicação

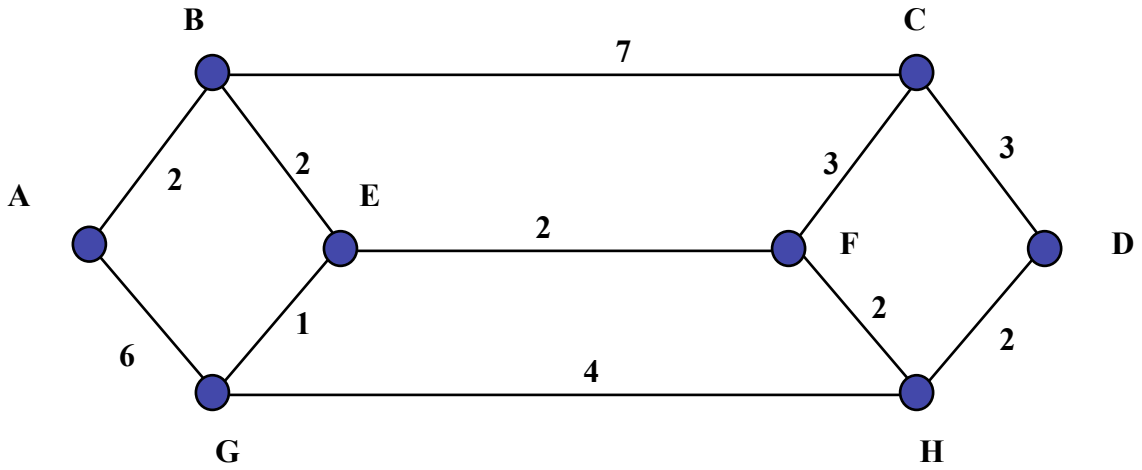
Enunciado do problema



- A rede pode ser vista como um grafo:
  - Os nós do grafo são os *routers*, os arcos são os canais
  - Custo de um arco, um valor inteiro ou real não nulo
  - Em geral usa-se uma função genérica, dita função custo ou métrica de encaminhamento (inverso da capacidade, tempo de encaminhamento, ...)
- Caminho "bom":
  - Tipicamente é o que minimiza o custo total da origem até ao destino

Determinação centralizada do caminho mais curto

Tal consiste em avaliar o caminho mais curto (*SP - shortest path*) até ao destino. Trata-se de um problema clássico de optimização, em teoria dos grafos.



Determinação centralizada do caminho mais curto

Tal consiste em avaliar o caminho mais curto (*SP - shortest path*) até ao destino. Trata-se de um problema clássico de optimização, em teoria dos grafos.

Algoritmo de Dijkstra

- Este algoritmo centralizado calcula o caminho mais curto de cada nó até cada um dos outros; o conjunto dos caminhos determinados é uma árvore com raiz no nó origem

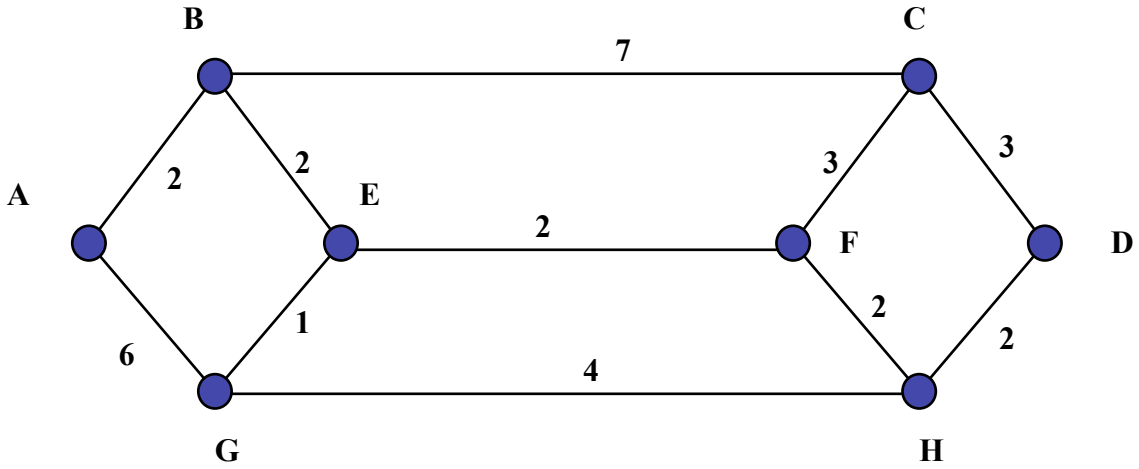
Notação

- **c(x,y):** custo do canal de x para y; = ∞ se não existe nenhum canal que liga x a y
- **D(v):** custo total do nó origem até ao nó v
- **p(v):** predecessor pelo caminho escolhido até ao nó v
- **N':** conjunto de nós para os quais já se conhece o caminho mais curto a partir do nó origem

Algoritmo de Dijkstra

```
1  Início:
2    N' = {u}
3    Para todo os nós v
4      if v é adjacente de u
5          then D(v) = c(u,v)
6      else D(v) = ∞
7
8  Loop
9     Encontrar w not in N' tal que D(w) é um mínimo
10    juntar w a N'
11   actualizar D(v) para todos os nós v adjacentes a w que não
           pertencem a N' :
12       D(v) = min( D(v), D(w) + c(w,v) )
13    /* o novo custo para v é o velho D(v) ou o já conhecido
14     custo mínimo até w D(w) mais o custo de w para v c(w,v) */
15 until todos os nós em N'
```

Quais os melhores caminhos ?

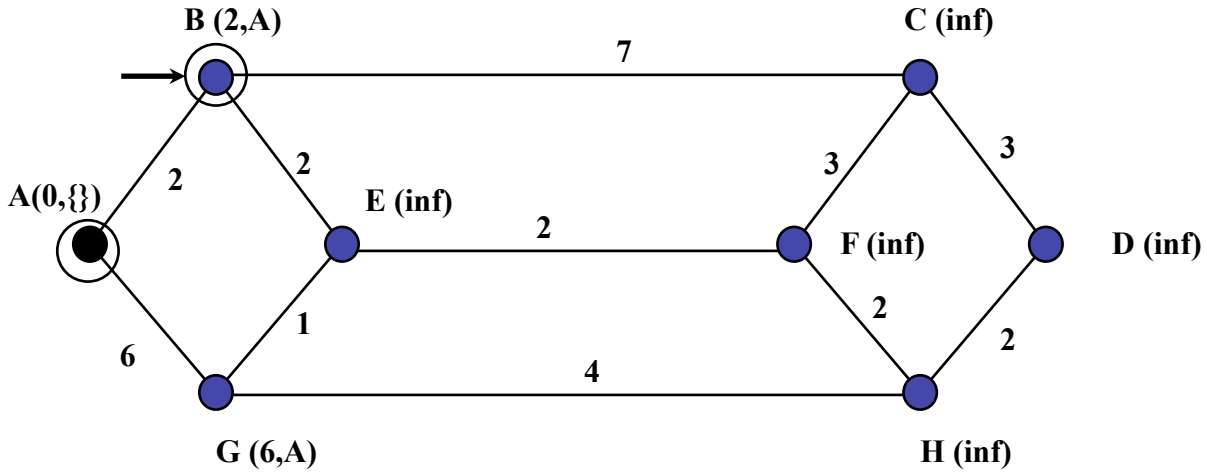


Legenda:

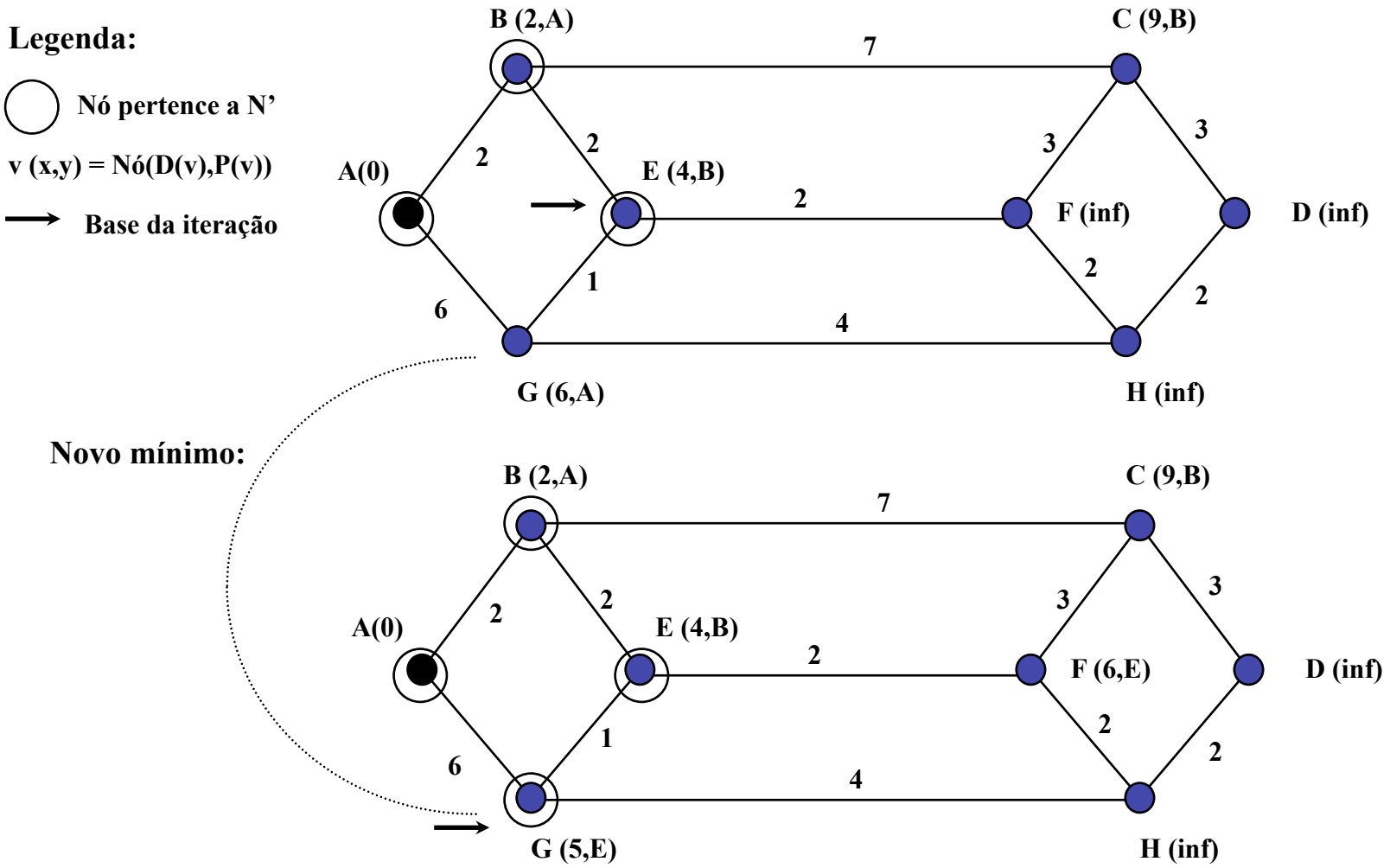
Nó pertence a N'

v (x,y) = Nó(D(v),P(v))

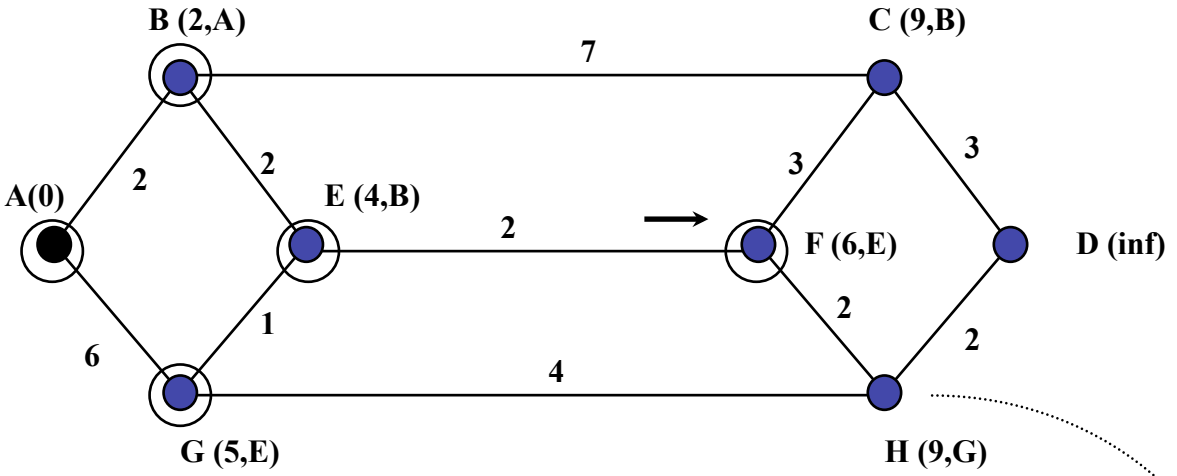
Base da iteração



Continuação (1)



Continuação (2)



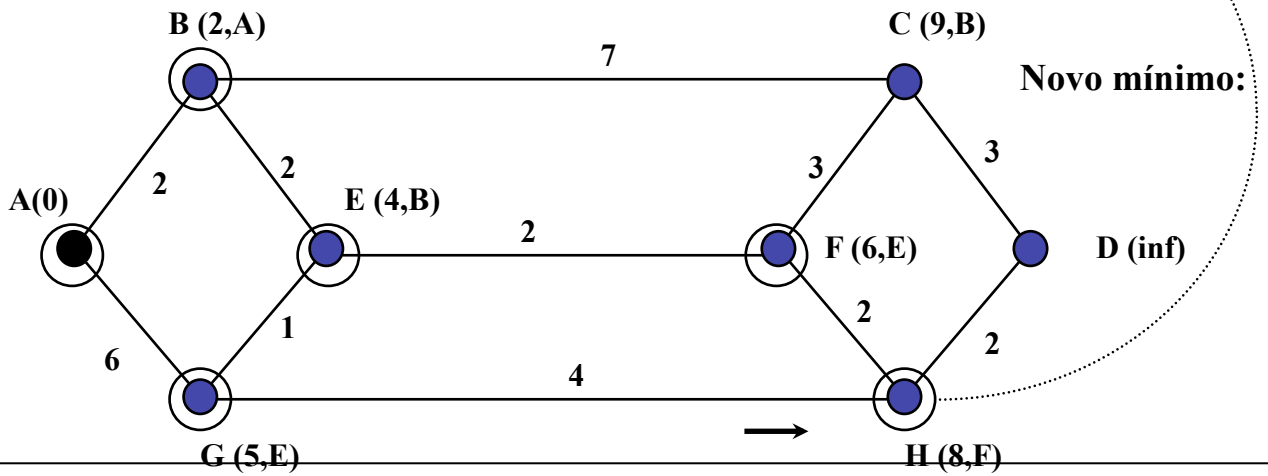
Legenda:

Nó pertence a N'

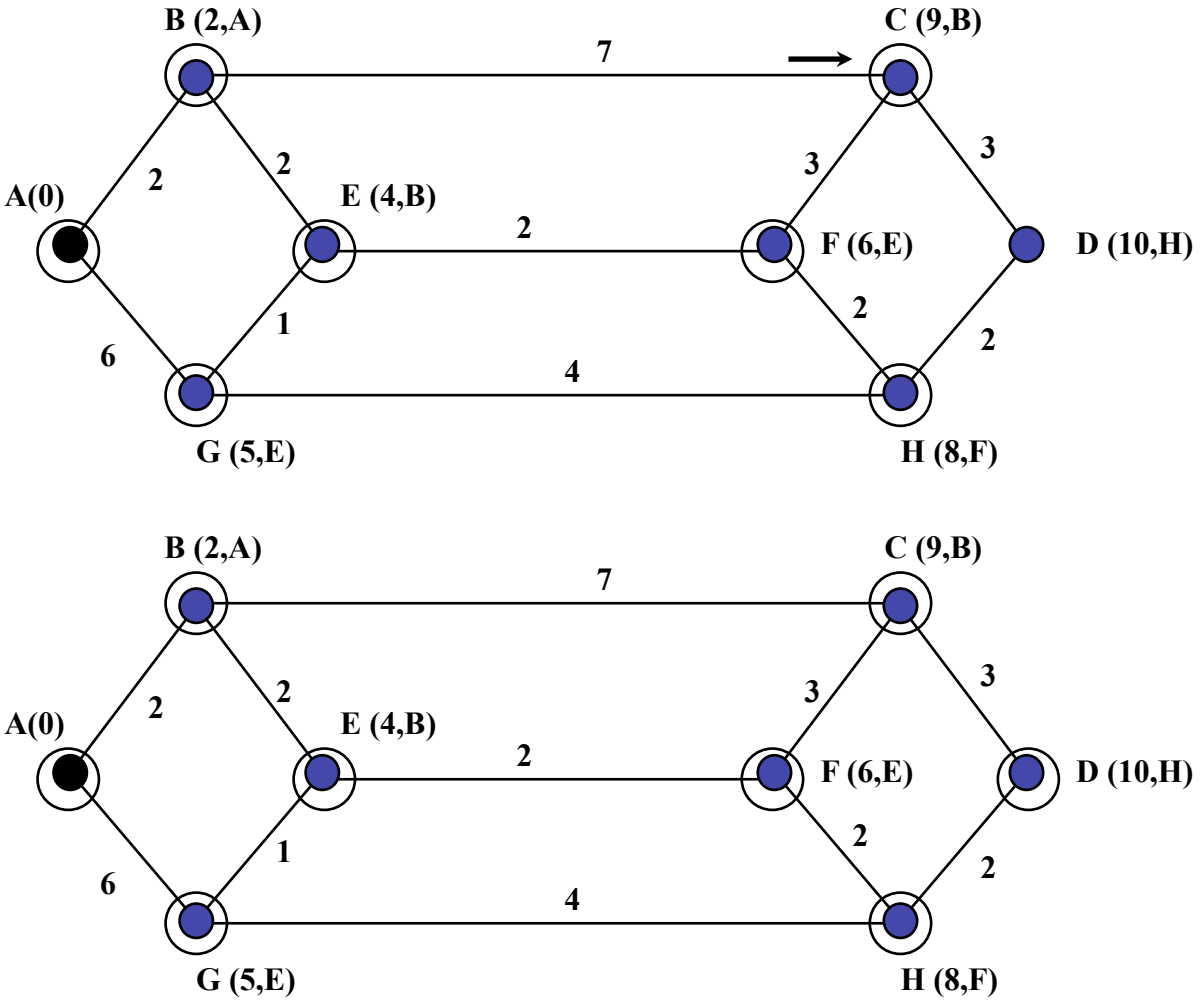
v (x,y) = Nó(D(v),P(v))

Base da iteração

Novo mínimo:



Continuação (3)



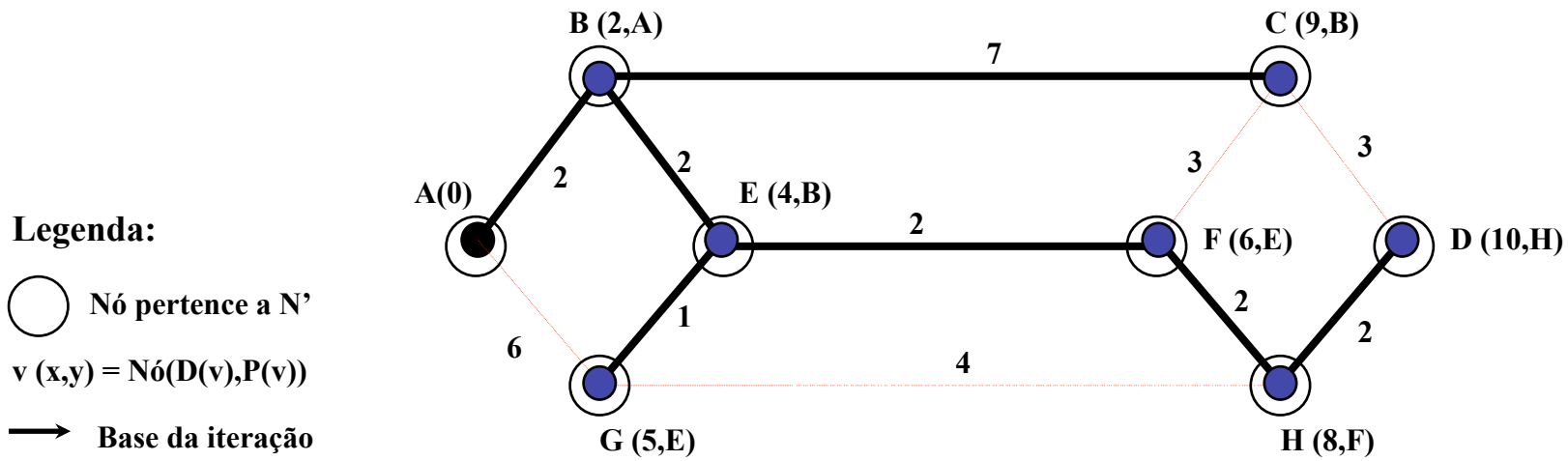
Legenda:

Nó pertence a N'

v (x,y) = Nó(D(v),P(v))

Base da iteração

Arvore de caminhos mínimos



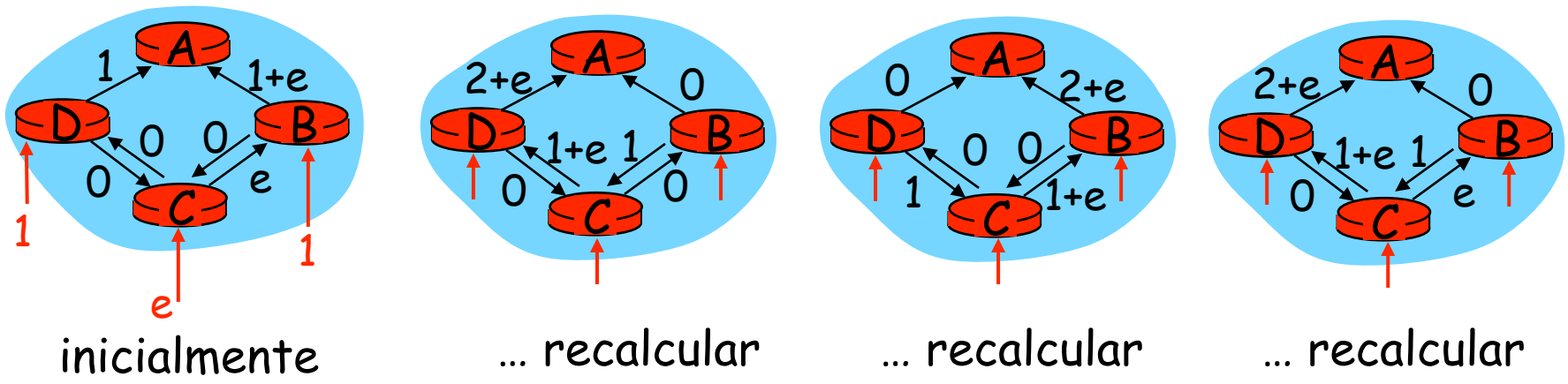
Discussão

Complexidade: n nós

- **Em cada iteração: é preciso testar todos os nós, w, que não em N'**
- **n(n+1)/2 comparações: $O(n^2)$**
- **há realizações mais eficientes: $O(n \log n)$**

O Algoritmo pode apresentar oscilações:

- **Por exemplo, quando o custo de um canal = quantidade de tráfego que o atravessa**



Encaminhamento distribuído

Numa rede são possíveis várias estratégias de algoritmos distribuídos de encaminhamento. Um critério essencial é a informação necessária para preencher a tabela de encaminhamento:

Mínima. *O router* conhece apenas os vizinhos e não tem tabela de encaminhamento: **Inundação ("Flooding")**

Descentralizada. *O router* conhece apenas os vizinhos e o custo para lá chegar. Um processo iterativo de computação com troca de informação com os vizinhos permite construir uma tabela de encaminhamento. **Algoritmos "distancia vectorial"**

Global. Todos os *routers* conhecem a totalidade da topologia da rede e usam essa informação para construir uma tabela de encaminhamento: **Algoritmos "Estado dos canais" ou "link state"**

Encaminhamento por inundação

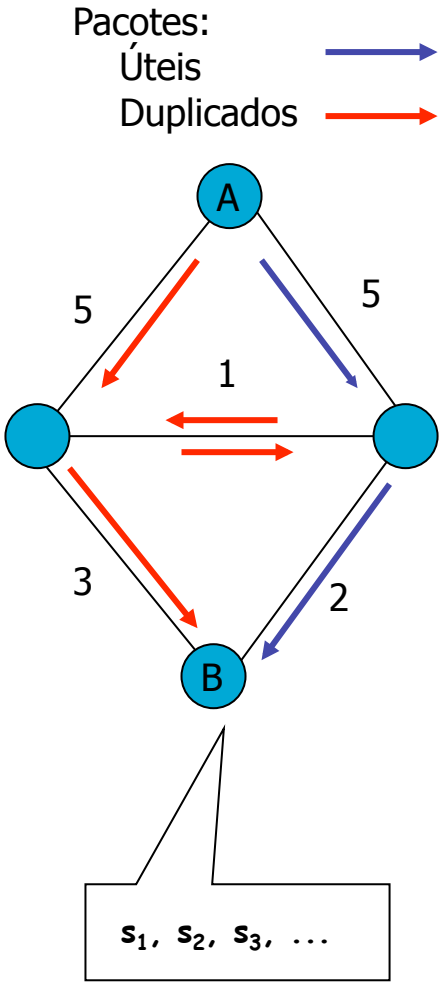
Quando um *router* recebe um pacote, se este não é o destino, reemite o pacote por todos os canais, com excepção daquele por que recebeu o pacote.

Como encaminha muitos duplicados, exige um método para eliminar os pacotes redundantes:

- *hop count* no cabeçalho de cada pacote
- número de sequência por origem e tabela de pacotes vistos ultimamente

Provavelmente, o primeiro pacote chega ao destino pelo caminho óptimo, seguido eventualmente por alguns duplicados. Este aspecto garante fiabilidade devido à redundância.

Trata-se de uma algoritmo muito robusto mas geralmente irrealista devido ao elevado número de duplicados.



Algoritmo "Vector-distance" ou Bellman-Ford

Cada *router* tem uma tabela de encaminhamento com entradas com:

- destino
- *next hop* (caminho a tomar para lá chegar)
- custo (desse caminho)

Cada *router* mede periodicamente o custo para chegar a cada vizinho (se a métrica for o número de saltos - *hops* - esse custo é sempre 1 ou infinito)

Periodicamente cada *router* passa ao seu vizinho a indicação dos destinos que conhece e com que custo lá chega (vector de distâncias ou anúncio de visibilidade ou de "*reachability*")

Quando recebe um anúncio de visibilidade ("*reachability*") de um vizinho, cada *router* modifica a sua tabela para reflectir os (melhores) caminhos que vai aprendendo

Exemplo de uma das iterações periódicas

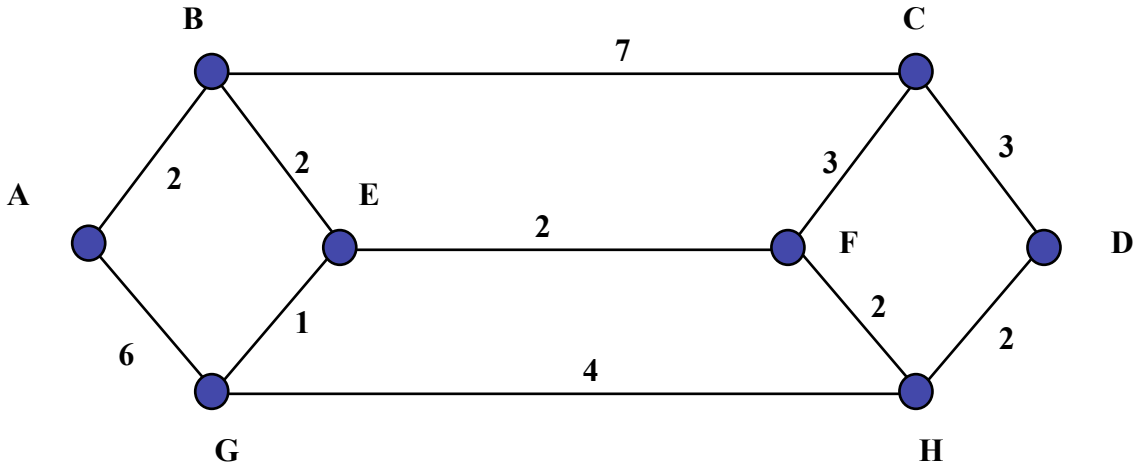


Tabela de A

from A to	next hop	cost
A	local	0
G	G	6
B	B	2
C	B	9
E	B	4
H	G	10

+

Anúncio de B

from B to	cost
B	0
A	2
E	2
C	7
G	3
F	4
H	6
D	8

=

Nova tabela de A

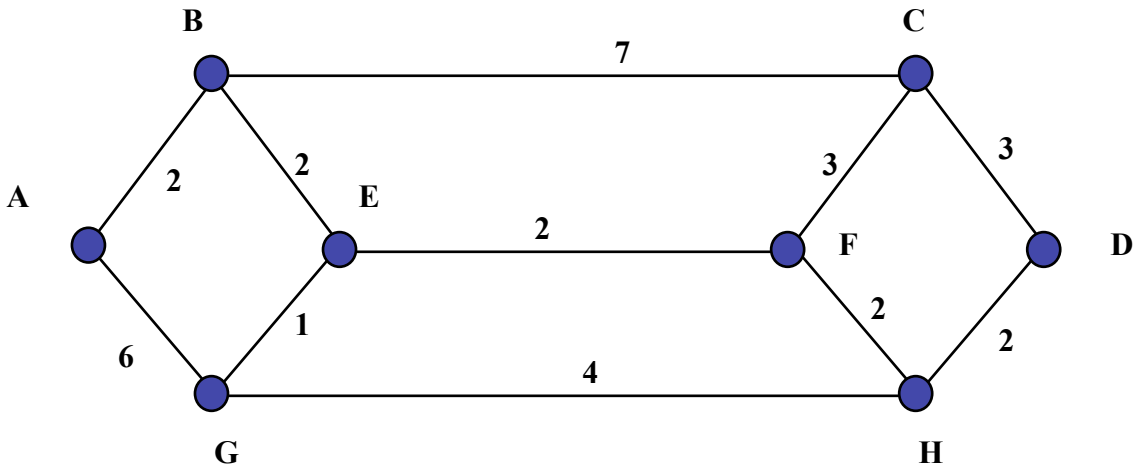
from A to	next hop	cost
A	local	0
G	B	5
B	B	2
C	B	9
E	B	4
H	B	8
F	B	6
D	B	10

Exemplo: evolução da tabela de encaminhamento de A

(1)

from A to	next hop	cost
A	local	0

from A to	next hop	cost
A	local	0
G	G	6
B	B	2



(2)

from A to	next hop	cost
A	local	0
G	G	6
B	B	2
C	B	9
E	B	4
H	G	10

(3)

from A to	next hop	cost
A	local	0
G	B	5
B	B	2
C	B	9
E	B	4
H	G	10
F	B	6
D	B	12

(4)

from A to	next hop	cost
A	local	0
G	B	5
B	B	2
C	B	9
E	B	4
H	B	8
F	B	6
D	B	10

As boas notícias andam depressa

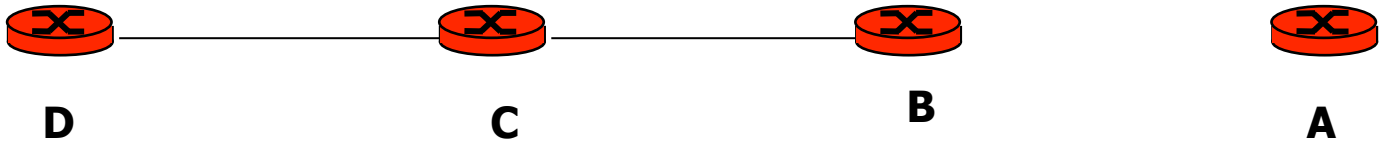
O Tempo de convergência de um algoritmo de encaminhamento é o tempo que medeia entre dar-se uma alteração no estado da rede e a mesma ser reflectida em todos os *routers*. O canal BA vem acima (custo passa de infinito a 1 por exemplo):

- O nó B detecta a alteração e propaga-a para os vizinhos que acabam por ficar a conhecer o novo destino
- O processo converge rapidamente e é conhecido pelo processo "Good news travel fast" nos algoritmos vector de distâncias

As más notícias andam devagar

O Canal BA deixa de funcionar, o seu custo passa a infinito:

- Caso o canal BA vá abaixo, o seu custo passa a infinito, no entanto se B receber um anúncio de C onde figura a presença de A, dá-se um ciclo associado a um fenómeno designado por "bad news travel slowly"
- É o problema da contagem para o infinito ("count to infinity").
- Esta instabilidade passa-se sempre que B fique convencido que há um caminho para A via C



Os anúncios entre *routers* podem ser enviados só periodicamente ou sempre que existirem alterações das tabelas ou do estado dos *routers* ("trigered updates"). Mesmo neste caso o ciclo de anúncios continuará até que o valor do custo seja tão elevado que seja considerado "infinito".

Tentativas de solução

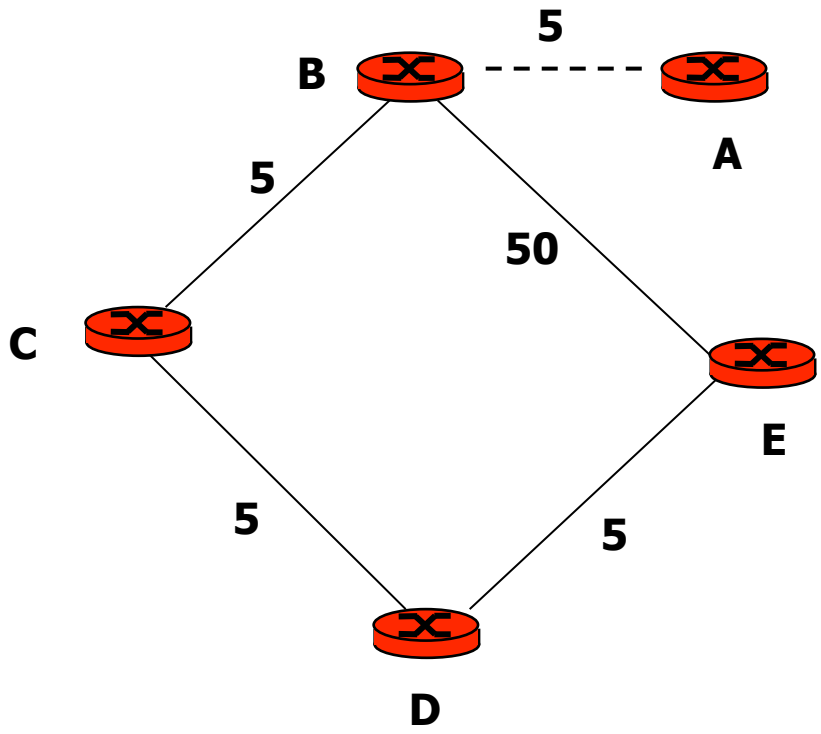
"Split horizon with poisoned reverse" - um *router* R1 anuncia ao *router* R2 à distância infinito todos os destinos para os quais R1 usa R2 como melhor caminho (R1 anuncia a R2 com distância infinito, tudo aquilo que R1 acha que está por detrás de R2)

Infinito é modelizado por um número relativamente baixo para aumentar a velocidade de convergência (tal solução implica que este algoritmo não pode ser aplicado a situações em que o diâmetro da rede é muito grande). No protocolo RIP, por exemplo, infinito = 16, pelo que mesmo que aparecesse o ciclo, este terminaria rapidamente

"Triggered updates" - sempre que há uma alteração nos canais ou nas tabelas de encaminhamento envia-se imediatamente anúncios.

Garantia total de ausência de ciclos

Estes mecanismos não garantem total ausência de problemas pois podem existir casos particulares em que os mesmos poderiam voltar a aparecer.



Aproximações possíveis

- O protocolo RIP faz anúncios periódicos e anúncios na sequência de uma alteração

- Para resolver este problema, quando detecta um anúncio de que o custo de um destino passou a ser infinito, bloqueia durante algum tempo todos os novos anúncios para esse destino

- No protocolo BGP cada anúncio contém igualmente o caminho completo pelo que o receptor pode detectar ciclos

Em resumo

O algoritmo "vector de distâncias" calcula de forma distribuída o caminho óptimo para cada destino. É um algoritmo muito simples do ponto de vista computacional, convergindo rapidamente perante as "boas notícias", mas pior e com eventuais instabilidade perante as "más notícias".

Trata-se de uma solução certamente adequada se o diâmetro da rede, a sua complexidade e o número de destinos diferentes é relativamente baixo, pois nestas condições este algoritmo é muito simples e eficaz. Tal pode compensar o seu principal defeito que é o tempo de convergência por causa das más notícias.

Algoritmo estado dos canais (*link-state*)

Os protocolos estado dos canais (*link state*) baseiam-se na seguinte aproximação:

1. Cada *router* tem uma base de dados com o estado de cada canal da rede e uma tabela de encaminhamento

2. Sempre que há uma alteração, são difundidos por inundação pacotes indicando o novo estado dos canais; desta forma todos os routers convergem para uma cópia idêntica da base de dados de canais (*link state database*)

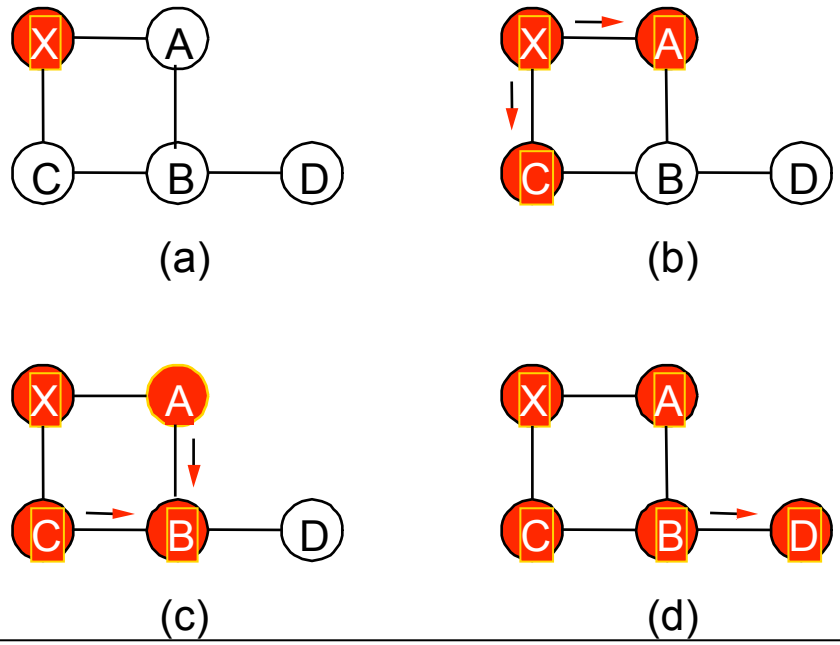
3. Quando há alterações da base de dados dos canais, são calculados os melhores caminhos para cada destino através do algoritmo "SP" de Dijkstra; a partir deste calculo é produzida uma nova versão da tabela de encaminhamento.

Link-State Routing

- **Cada router sabe quais são os seus canais e o respectivo estado**
  - O canal está up ou down
  - O custo do canal
- **Cada router executa um broadcast do link state**
  - Para qe cada router fique com uma visão completa do grafo
- **Cada router executa o algoritmo de Dijkstra**
  - Calcula os shortest paths
  - … e reconstrói a sua tabela de encaminhamento
- **Protocolos concretos que usam este método**
  - Open Shortest Path First (OSPF)
  - Intermediate System – Intermediate System (IS-IS)

Difusão dos Link-States

- **Flooding**
  - Cada nó envia a informação pelos seus links
  - E o nó a seguir também envia a infromação pelos seus links
  - … excepto aquele pela qual a informação lhe chegou



Difusão dos Link-States

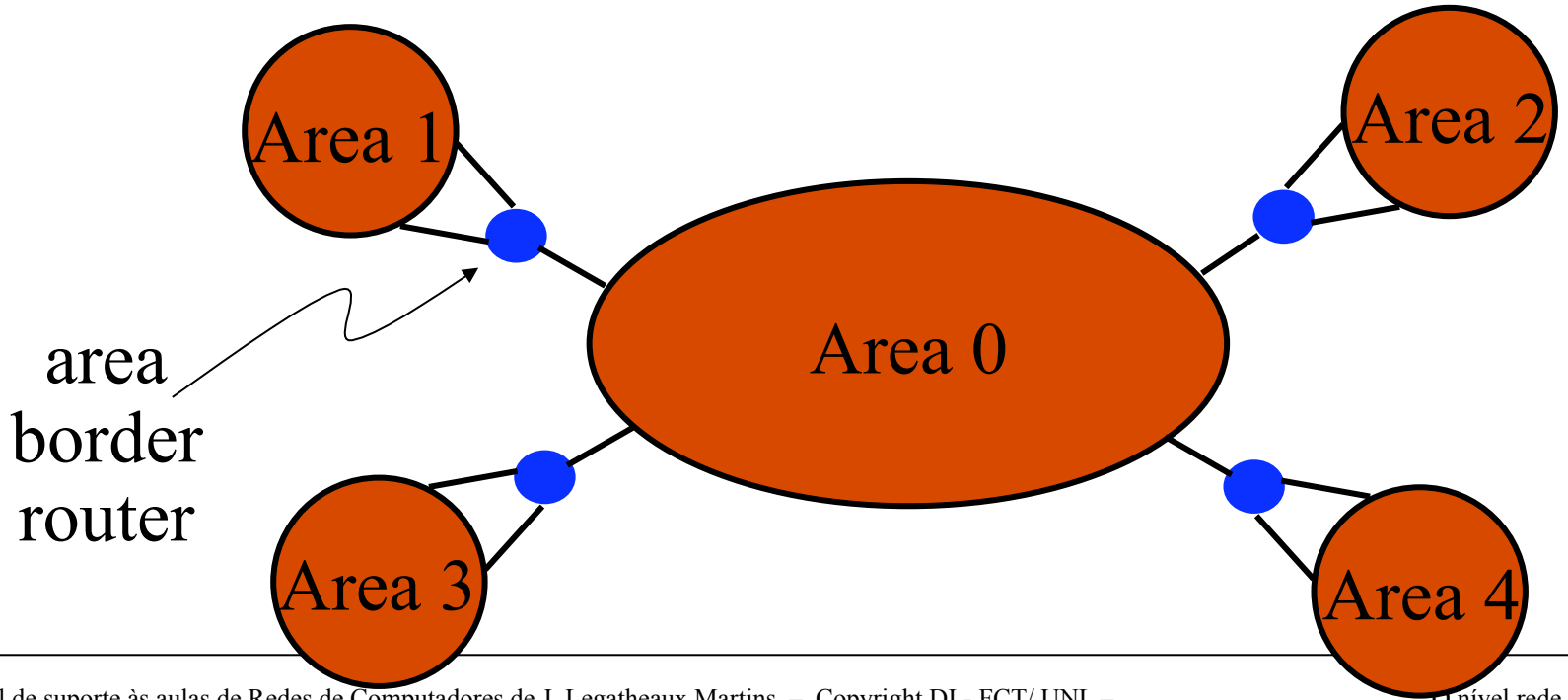
- **Difusão fiável**
  - Assegura que a informação chega a todos os nós
  - … e que cada um destes usa a última versão
- **Desafios**
  - Perca de pacotes
  - Chegada fora de ordem
- **Soluções**
  - Acknowledgments e retransmissões
  - Números de sequência
  - Time-to-live em cada pacote

Quando é que um nó desencadeia a difusão ?

- **Alteração da topologia**
  - Um link ou um nó ficam indisponíveis
  - Um link ou um nó ficam de novo disponíveis
- **Alteração da configuração**
  - Um link muda de custo
- **Periodicamente**
  - A informação de link-state é refrescada periodicamente
  - Tipicamente a cada 30 minutos
  - Permite corrigir alguma eventual corrupção dos dados

A problemática da escala

- **Overheads do algoritmo link-state**
  - Usa flooding dos link-state packets por toda a rede
  - Utiliza o algoritmo Dijkstra's shortest-path
- **Introduz-se hierarquia através de "areas"**



Visão sintética

Algoritmo estado dos canais (*link state*) - os *routers* fazem *difusão fiável* para todos os outros do custo associado a cada um dos seus canais. Perante alterações, cada *router* recalcula o custo de todos os caminhos com origem nele e actualiza a tabela de encaminhamento (o algoritmo usado pode ser o SPF de Dijkstra)

Vector distâncias (*distance vector*) - os *routers* passam aos seus vizinhos parte do o estado das suas tabelas de encaminhamento. Estes anúncios permitem a actualização imediata das tabelas de encaminhamento se for caso disso.

Com o algoritmo estado dos canais, a rede é inundada de pequenos anúncios de estado dos canais quando há alterações. Compete aos *routers*, consumindo CPU e memória significativas, recalcular as tabelas de encaminhamento.

Com o algoritmo vector de distâncias, cada *router* passa muita informação, mas apenas aos seus vizinhos. O calculo para actualização das tabelas é trivial. No entanto este algoritmo tem problemas de instabilidade (ciclos introduzidos pelo fenómeno das "más notícias") que são resolvidos aumentando o tempo de convergência.

Comparação entre LS e VD

Um protocolo LS faz uma difusão fiável de um estado que é replicado por cada *router*. Cada *router* recalcula a tabela de encaminhamento de forma centralizada. Os protocolos VD correspondem a um algoritmo distribuído que vai calculando a tabela de encaminhamento de forma incremental.

Os protocolos VD são mais simples. Ocupam menos memória, têm menos tipos de mensagens, são mais fáceis de administrar.

Os protocolos LS implicam anúncios mais curtos mas que se espalham por toda a rede.

Os protocolos LS não têm o problema "*contagem para o infinito*" pelo que tratam de forma mais segura as situações em que há circuitos que são interrompidos.