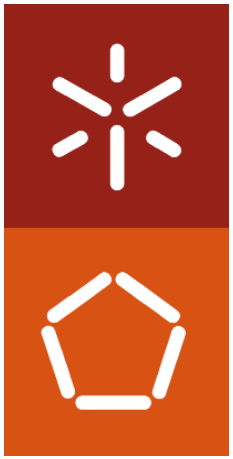
Conceitos, Algoritmos e Protocolos de Encaminhamento

Comunicações por Computador

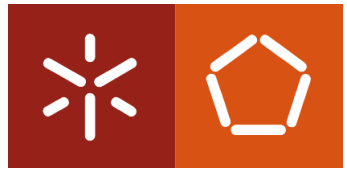
Mestrado Integrado em Engenharia Informática

3º ano/2º semestre

2015/2016

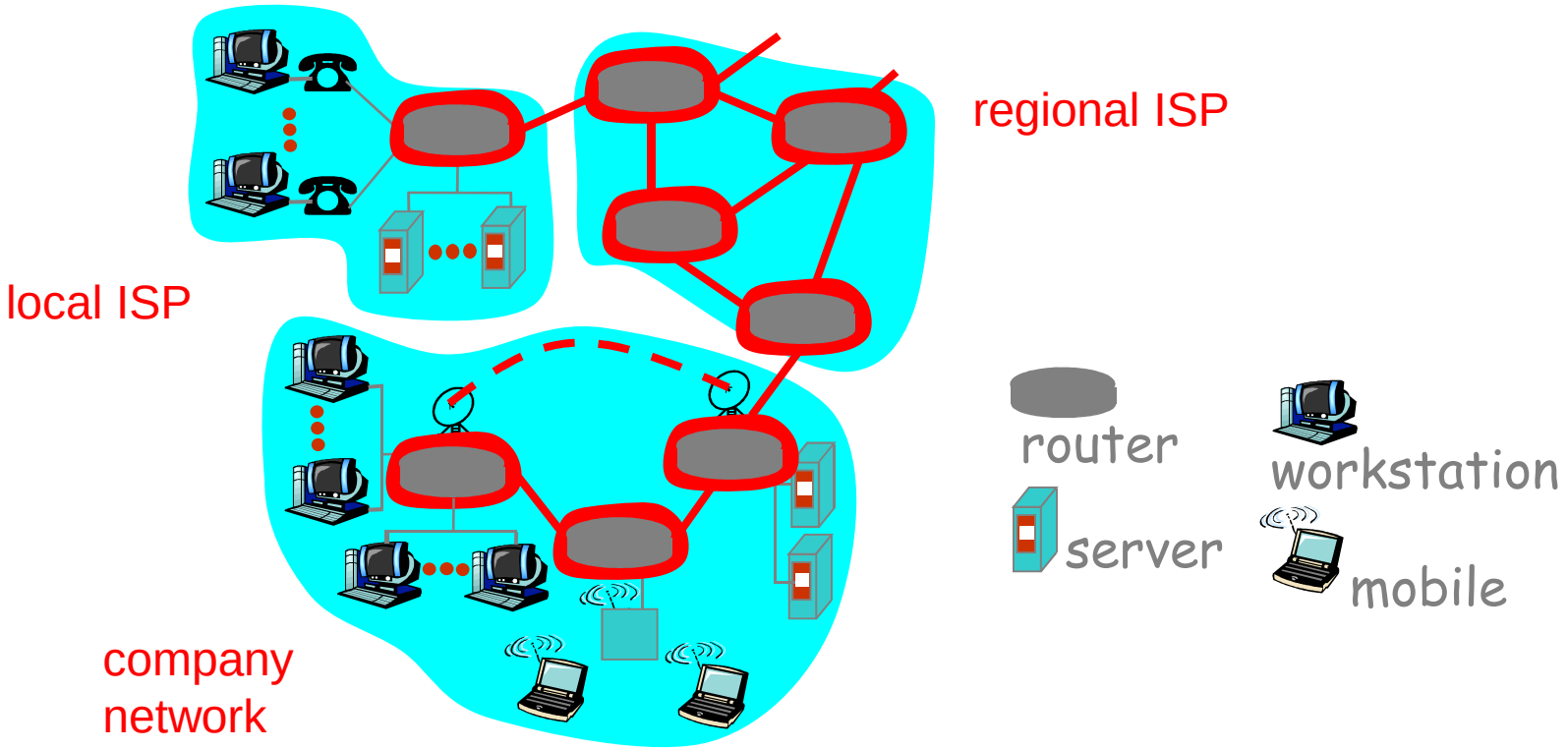


Sumário



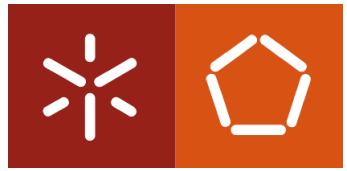
- **Encaminhamento**
  - Processo de Forwarding
  - Processo de Routing
- **Algoritmos de encaminhamento dinâmico**
  - Estado de Ligação (LS)
  - Vectores de Distância (DV)
  - Comparação entre DV e LS
- **Protocolos de encaminhamento**
  - Protocolos de encaminhamento interno (IGP)
  - Protocolos de encaminhamento externo (EGP)

Encaminhamento na Internet



Fonte: Computer Networking: A Top-Down Approach
Featuring the Internet, J. Kurose, Addison-Wesley

Exercício



- **Sobre os seguintes endereços IPv4 podemos afirmar que**

  193.136.21.134

  193.136.20.129

  193.136.20.65

  a) Estão todos na mesma rede...

  b) Estão todos em redes distintas...

  c) Apenas o 2º e o 3º estão na mesma rede...

  d) O que diabo é um endereço IPv4?

  e) Outra resposta.... (especificar)

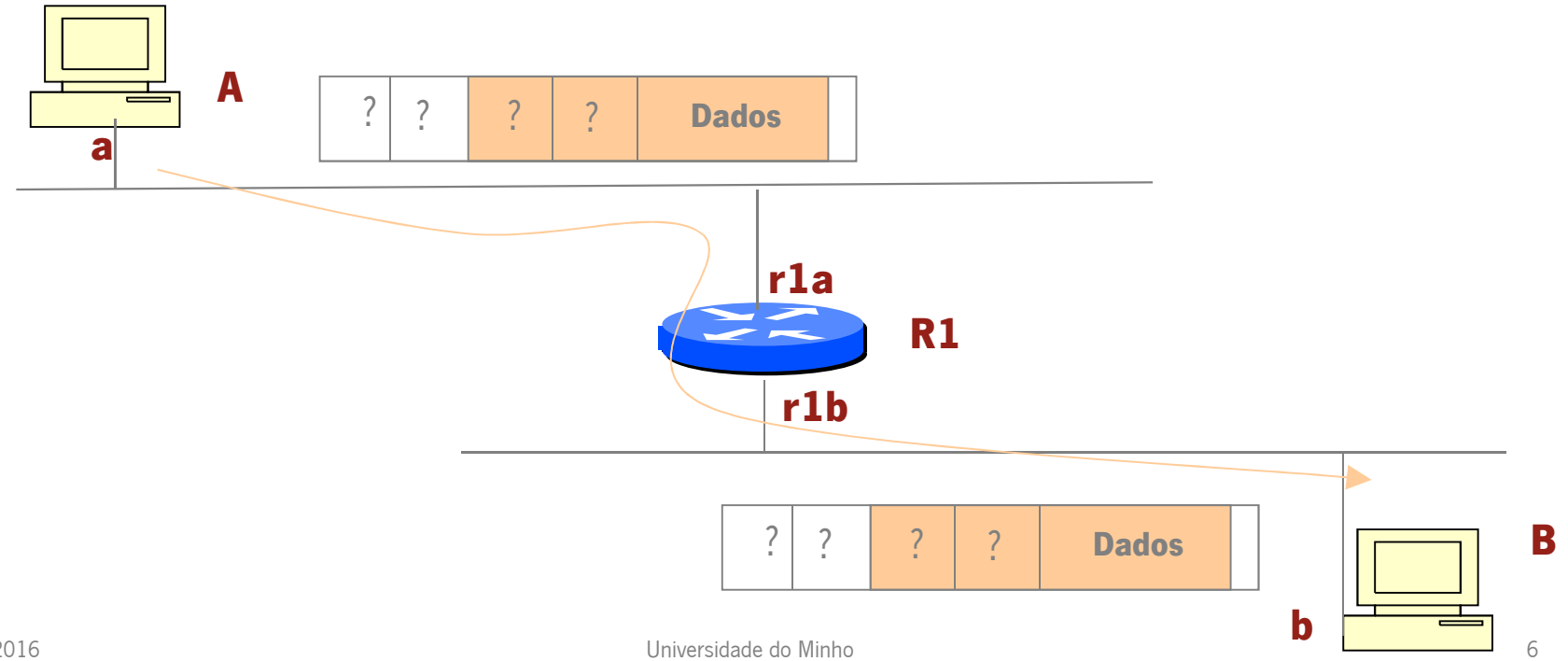
Equipamentos de Interligação

Exercício: Comparação

	hubs	***bridges***	***Switches (L2, L3)***	***routers***
plug & play				
Isolamento de tráfego				
Encaminhamento optimizado				
Store-and-forward (armazena-e-reenvia)				

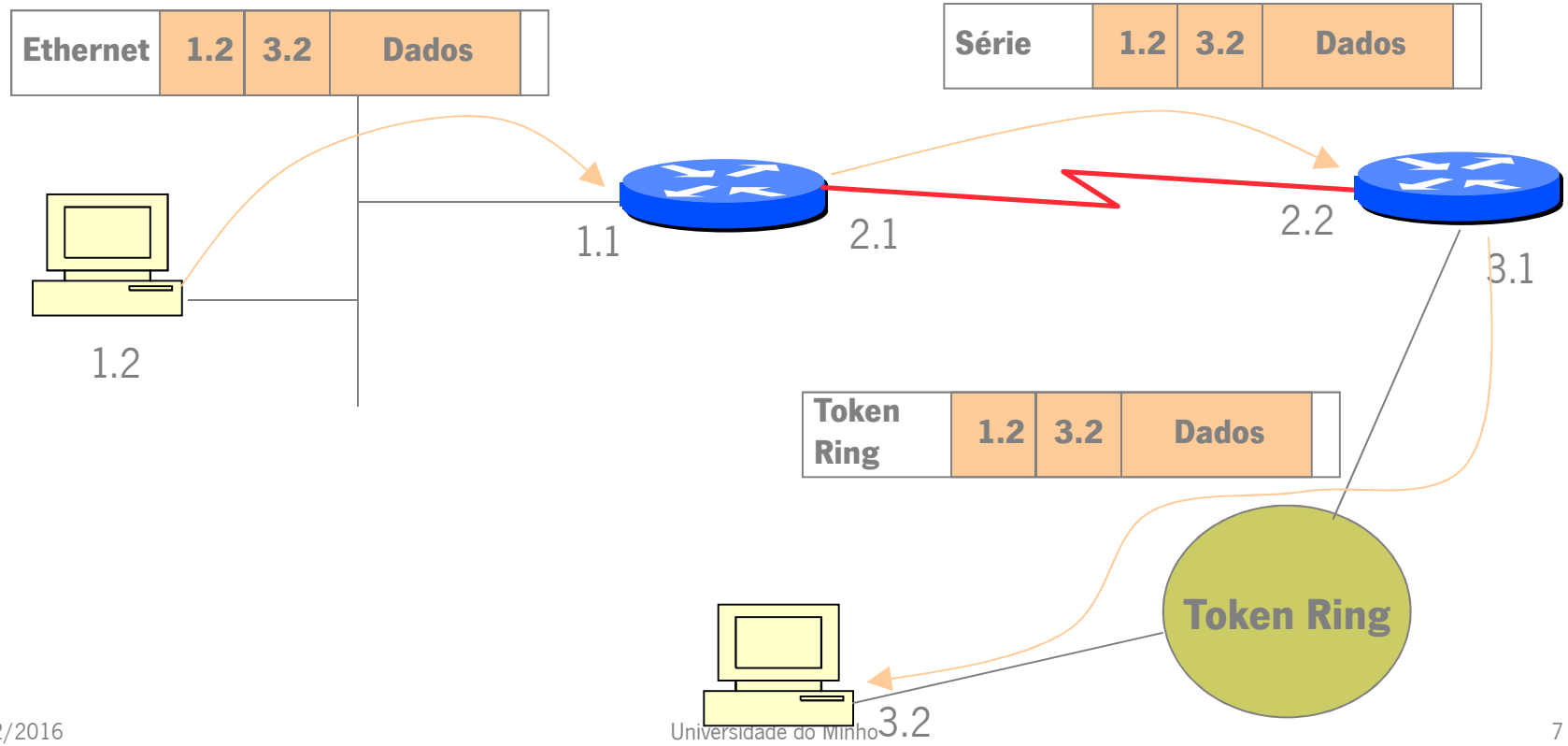
Relações entre nível 3 e nível 2

- ***Exercício de revisão:***
  - *Quais os endereços MAC e IP da trama no percurso de A para B? (Use a terminologia MAC(a) e IP(a) para referir os endereços da interface **a**)*
  - *Como são obtidos?*

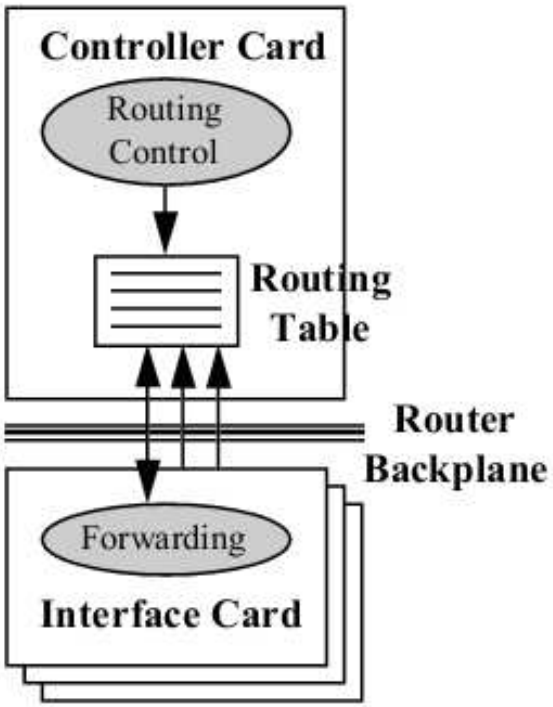


Encaminhamento na Internet

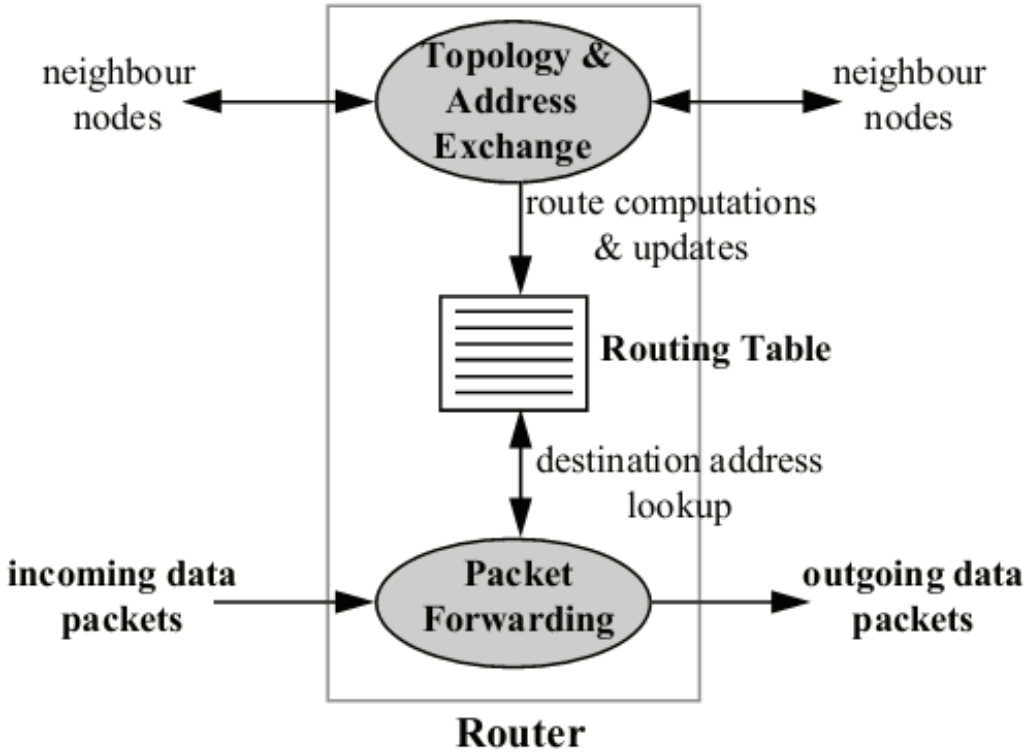
- ***Routers*** **armazenam e reenviam datagramas IP**
  - Desencapsula no interface de entrada, encapsula no interface de saída, de acordo com o tipo de interface...



Routers – Arquitectura Genérica



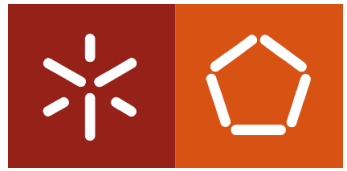
a) Arquitectura básica



b) Componentes de routing

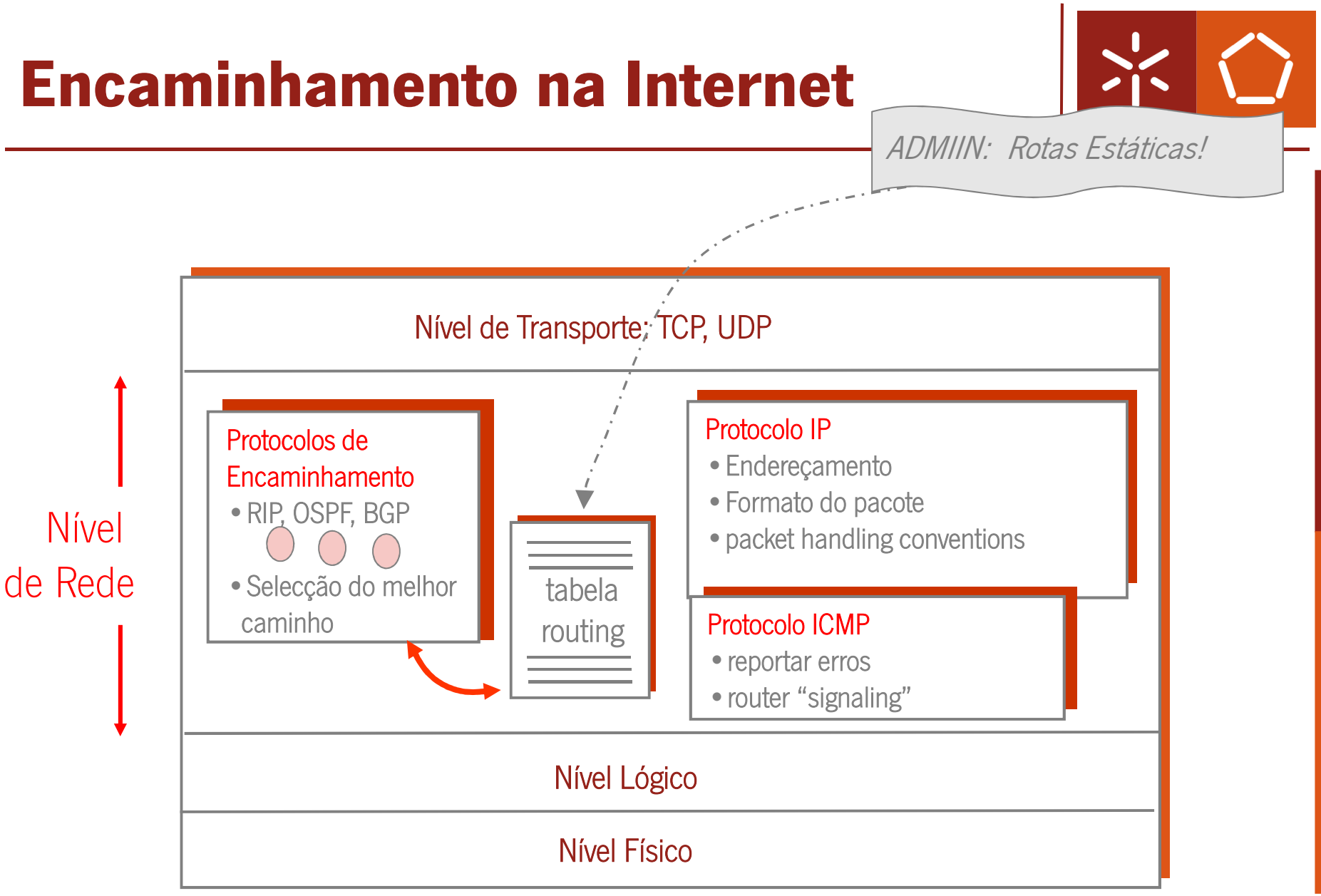
Fonte: James Aweya, IP router architectures: an overview

Encaminhamento na Internet



- **Reenvio (forwarding) num encaminhador:**
  - Utiliza a tabela de reenvio previamente preenchida pelos protocolos de encaminhamento ou pelo administrador
  - Procura na tabela, para um dado "destino", o "próximo salto" e o "interface de saída"
  - Comuta o pacote pelo interface respectivo, encapsulando-o numa trama de acordo com o tipo de interface
- **Encaminhamento (routing):**
  - Preenche a tabela de encaminhamento com a(s) melhor(es) rotas para as redes de destino (*classfull*) ou para um conjunto de prefixos de endereços (*classless*)
  - Pode ser um processo manual, feito pelo administrador – ***encaminhamento estático***
  - Ou um processo automático resultante da operação de um ***protocolo de encaminhamento***, no caso mais comum de ***encaminhamento dinâmico***

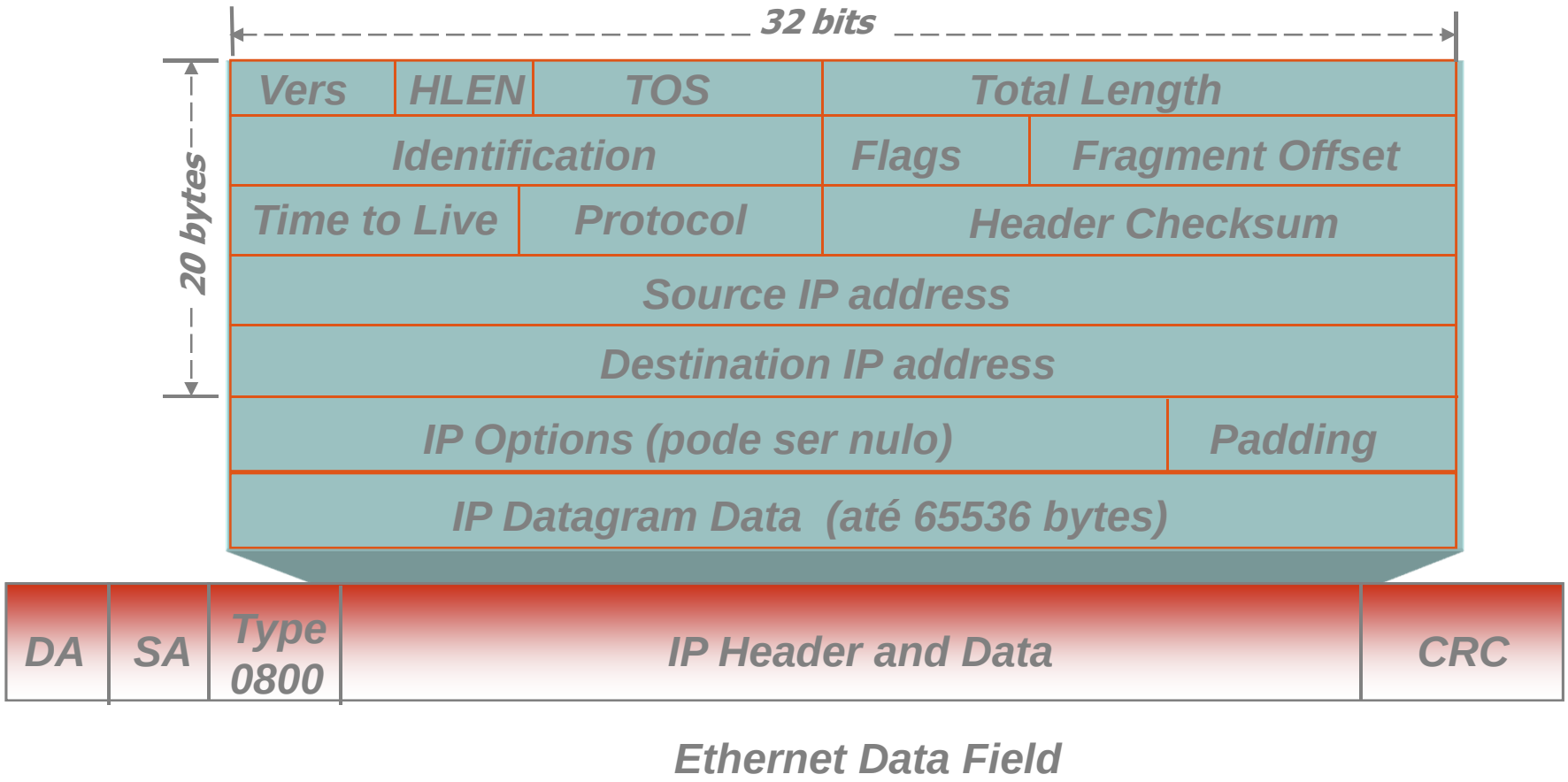
Encaminhamento na Internet



Fonte: Computer Networking: A Top-Down Approach Featuring the Internet, J. Kurose

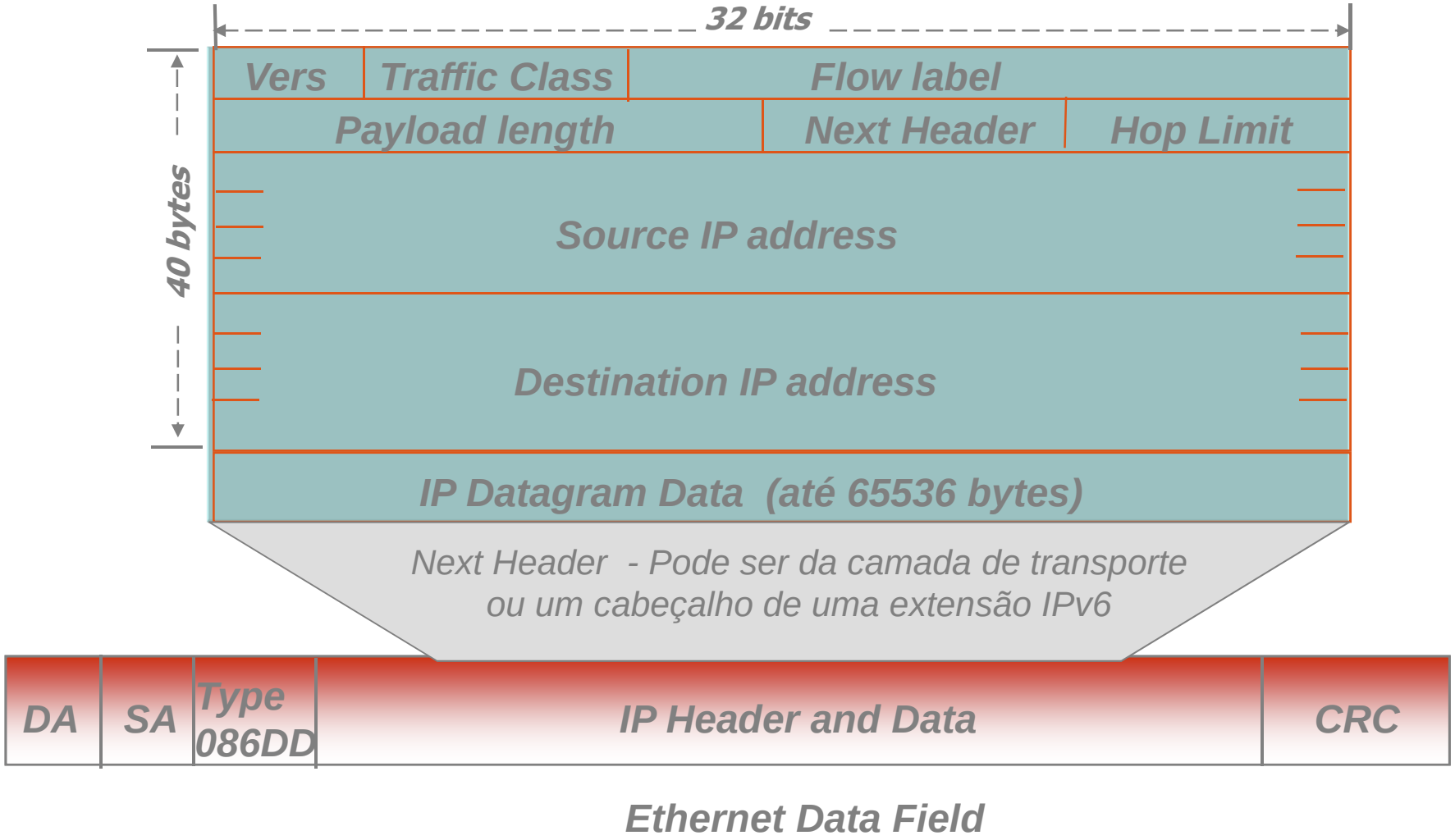
IPv4 – Internet Protocol version 4

Formato dos datagramas

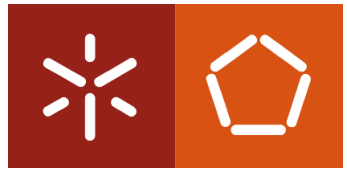


IPv6 – Internet Protocol version 6

Formato dos datagramas



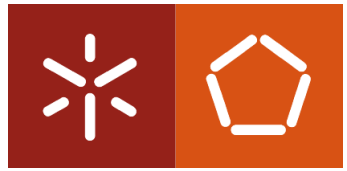
Encaminhamento na Internet



- **O procedimento de reenvio (forwarding):**

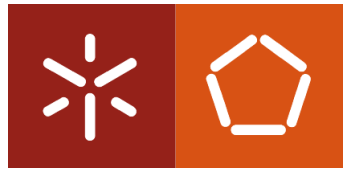
1. Chegada do pacote IP ao *router* numa das suas interfaces (ex: *Ethernet*), encapsulado numa trama; É processado de imediato o cabeçalho de nível 2; O pacote pode ser destruído se não for dirigido ao interface (ou *broadcast)*;
2. Analisa-se de seguida o campo tipo do cabeçalho 2. O valor 0x800 indica que lá dentro vem um datagrama IP. Descarta-se o cabeçalho Ethernet e passa-se ao nível IP;
3. Processa-se cabeçalho IP, verificando versão, tamanhos e *checksum* (só no IPv4); se alguma verificação falhar, descarta-se silenciosamente e termina;
4. Se o endereço de destino é um endereço do *router* ou o endereço de *broadcast* IP então passa-se ao nível acima (transporte) e termina;
5. Senão, verifica se o campo TTL é superior a 1; Decrementa TTL e ajusta checksum (só IPv4) do cabeçalho antes de reenviar; Envia *ICMP TTL Exceeded* se TTL=0;

Encaminhamento na Internet



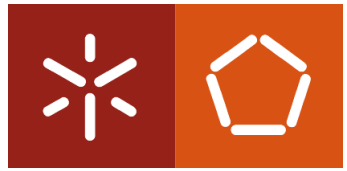
- **<u>O procedimento de reenvio</u> (forwarding):**
  6. Pesquisa o endereço de destino IP na tabela de encaminhamento se endereço *unicast (multicast* e *broadcast* são tratados de maneira diferente); Não se processam pacotes IP *unicast* recebidos em *broadcast de* nível 2; O endereço de destino é usado como chave de pesquisa na tabela de encaminhamento e obtém-se a rota mais adequada (melhor *match*);
  7. Se não existir rota, o pacote é descartado e devolvido um *ICMP Destination unreachable*; Se existir, extrai-se da rota a interface de saída e o endereço do próximo router;
  8. Se o pacote IP for maior que o MTU da interface de saída, fragmenta-se (só IPv4 e se o bit *DF - Don't Fragment* o permitir); Envia *ICMP Destination unreachable* se não for possível ou permitido;
  9. O *router* acrescenta então um novo cabeçalho de nível 2; Pode ter de usar o ARP (ou equivalente) para obter o endereço de destino nível 2 a partir do endereço IP do próximo salto;
  10. Envia pelo interface definido para o próximo salto;

Encaminhamento na Internet



- **Modificações ao processo de reenvio *normal*:**
  - No caso IPv6 existem algumas diferenças (checksum, etc) que veremos depois!
  - ***Multi-path routing*** ou ***Load Balancing***
    - Habitualmente só existe uma rota possível – a melhor! – para cada destino; Mas pode existir mais que uma rota de igual custo para o mesmo destino e o router pode distribuir os pacotes por cada uma dessas rotas, distribuindo o tráfego equitativamente!
      - Vantagem: mais largura de banda disponível para tráfego para esse destino
    - A pesquisa na tabela devolve várias rotas; a escolha de qual usar depende da implementação;
      - por vezes tenta-se manter os fluxos TCP na mesma rota evitando reordenações; escolha da rota é feita usando uma função *hash* dos endereços origem e destino juntamente com o cabeçalho TCP;

Encaminhamento na Internet

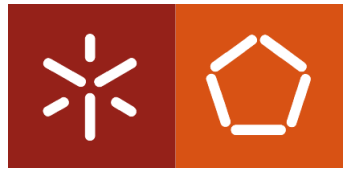


- ***Multi-path routing*** ou ***Load Balancing***

  Usando genericamente o termo *custo* para referir a *métrica* associada a cada rota, a melhor rota é a de menor custo. A distribuição de carga por múltiplas rotas, pode ser feita de duas formas:

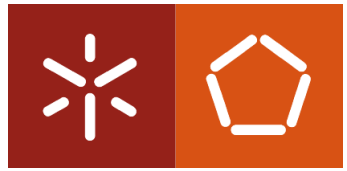
  - **Distribuição de carga entre rotas de igual custo**
    - O tráfego é distribuído equitativamente por todas as rotas;
  - ***Distribuição de carga entre rotas de custo diferente***
    - Os pacotes são divididos pelas rotas disponíveis, sendo a distribuição de tráfego feita na proporção inversa do custo da rota. Mais tráfego para as rotas de menor custo e vice-versa. Nem todos os protocolos suportam esta opção.

Encaminhamento na Internet



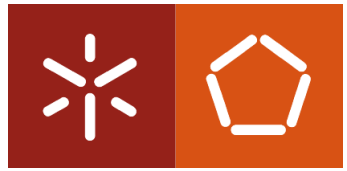
- **Modificações ao processo de reenvio *normal*:**
  - ***Encaminhamento com base no campo TOS***
    - O campo TOS definia originalmente 5 tipos de serviço (normal, minimizar custo, maximizar fiabilidade, maximizar débito, minimizar o atraso) tem agora novos usos nos novos modelos de Qualidade de Serviço; Teoricamente o router pode escolher rotas diferentes para diferentes valores do campo TOS;
    - A pesquisa na tabela é feita usando o par (endereço destino, campo TOS) do pacote como chave de pesquisa;
      - A melhor rota para um tipo de serviço pode não ser a melhor para outro;
      - Implica um rota por tipo de serviço, na tabela de encaminhamento;
    - Pouco usado na prática, é ainda um alvo nos novos modelos de Qualidade de Serviço: **QoS routing**

Encaminhamento na Internet



- **Encaminhamento determinado pela origem**
  - Alternativa ao normal que é determinado nó-a-nó (hop-by-hop)
  - Comportamento alterado pelas opções IP: ***Record-route, Strict-Source Routing e Loose-Source Routing***
    - A opção ***record-route*** obriga cada router a acrescentar o seu endereço à lista de endereços do percurso que vai sendo construída no cabeçalho;
    - ***Strict-Source Routing*** : A aplicação que origina o pacote pode determinar o caminho que ele deve seguir de forma rígida, salto a salto, incluindo-o no cabeçalho (opção)! O pacote tem de percorrer apenas e exactamente todos os routers enumerados no cabeçalho, pela ordem respectiva!
    - ***Loose-Source Routing:*** A aplicação indica um conjunto de routers por onde o pacote deve passar no seu percurso; pode ser usados outros;
  - Opções pouco usadas porque atrasam processamento; routers optimizados para desempenho podem ignorar estas opções; considerado também como ameaça á segurança por alguns ISPs; a menos que seja interno: ao ISP apenas;

Encaminhamento na Internet

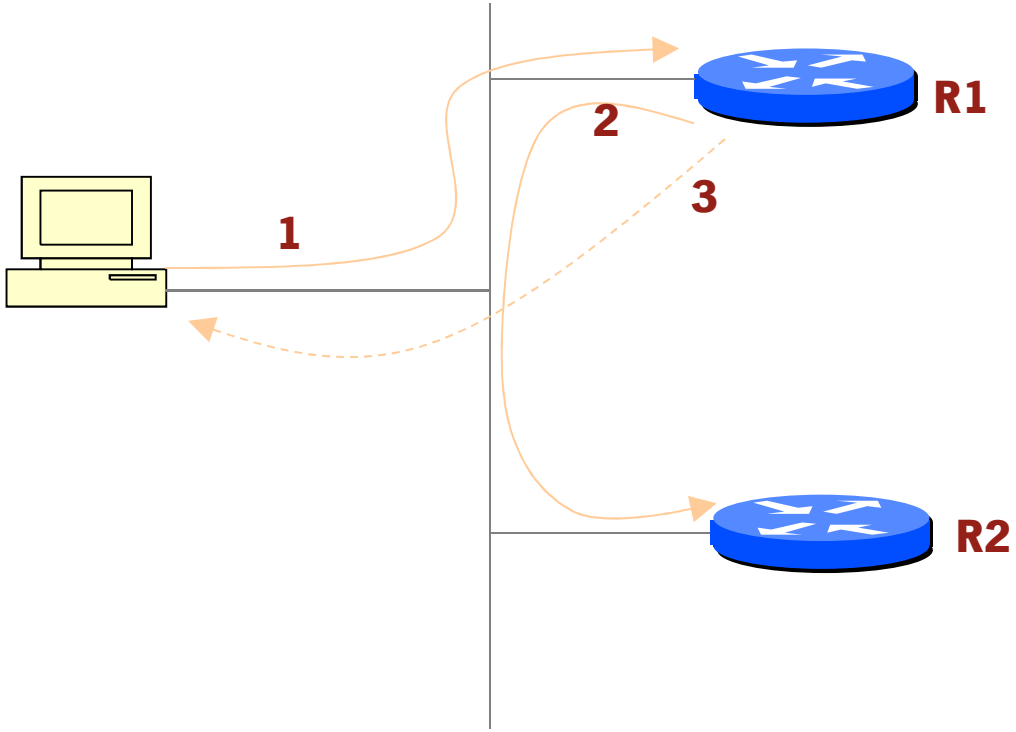


- **O uso do ICMP:**
  1. Reportar erros de encaminhamento:
     - *Destination Unreachable, Parameter Problem, Fragmentation Needed, TTL Exceeded*
  2. Descobrir encaminhadores:
     - *Router Discovery, Redirect*
  3. Testar a acessibilidade:
     - *Echo Request, Echo Reply*

- **Routers devem ter cuidado no envio de pacotes ICMP**
  - Não enviar ICMP na resposta a broadcast, multicast ou ICMP!!
  - Para evitar sobrecarga, podem ser suprimidos (ex: Echo reply)
  - Costumam ser usados para diagnóstico: traceroute

Encaminhamento na Internet

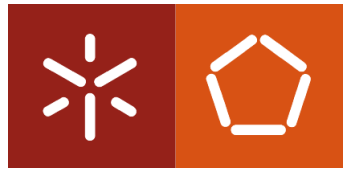
- **Exemplo de uso: primeiro router, numa rede multi-acesso**



1. **Host envia pacote para router R1 que ele pensa ser o melhor router de saída...**
2. **Router R1 reenvia pacote para R2 que é de facto o próximo salto**
3. **R1 notifica o host com uma mensagem ICMP Redirect para que corrija a rota de futuro... (temporizado)**

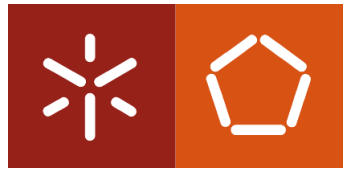
Baseado numa figura em "OSPF – Anatomy of an Internet Routing Protocol", John T. Moy, Addison-Wesley, 2002

Encaminhamento na Internet



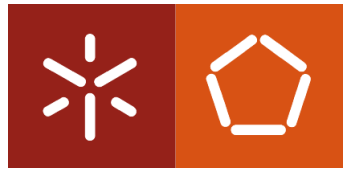
- **A tabela de encaminhamento (routing table)**
  - Campos na tabela:
    - destino, interface saída, próximo salto
  - Podem ter milhares de entradas...
    - pesquisa tem de ser muito eficiente! É o processo mais crítico!
- **Como se processa a pesquisa?**
  - A chave de pesquisa são endereços IP...
  - Ou melhor, a porção do endereço que identifica a rede....
  - Que pode ser obtida usando a *máscara de rede*
  - Portanto, o processo é diferente com *endereçamento* por classes (***classfull***) ou sem classes (***classess***)

Encaminhamento na Internet



- **Endereçamento por classes (ou *Classful*)**
  - esquema original, baseado na RFC 791
  - usa os primeiros bits como identificadores de classe
  - Revelou-se **completamente inadequado**... **e está obsoleto!!!!**
- **Endereçamento sem classes (ou *Classless*)**
  - não considera os bits de classe utilizando uma máscara de 32 bits para determinar o endereço de rede
  - permite encaminhamento mais eficiente por agregação de rotas, designado por CIDR (*Classless Internet Domain Routing*)
  - tabelas de encaminhamento mais pequenas
    - as rotas são agregadas por grupos de endereços adjacentes
  - usado pelas tabelas de encaminhamento de ISPs

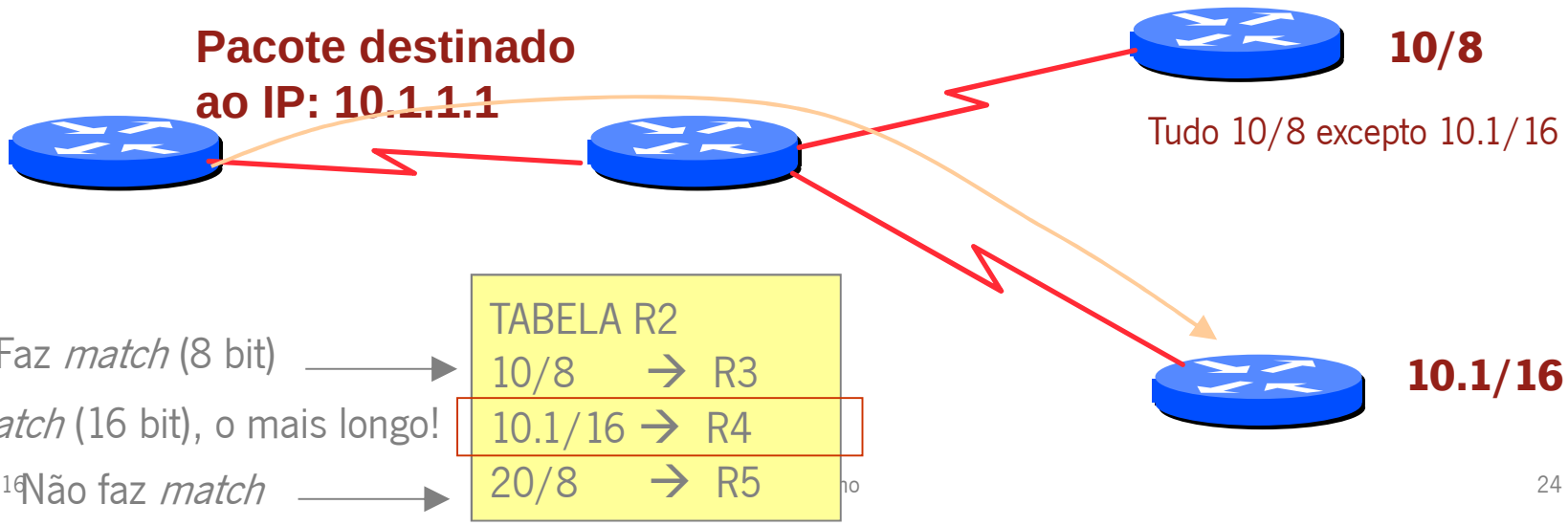
Encaminhamento na Internet



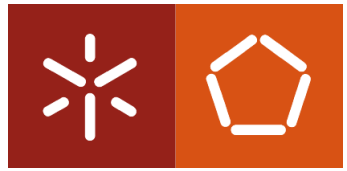
- **Endereçamento por classes *(classfull):***
  - Uso ineficiente do espaço de endereçamento, exaustão de espaço
  - Ex: uma classe B aloca 65K hosts mesmo que existam apenas 2K hosts!
- **Enderecamento sem classes (*classless*):**
  - **Parte de rede** (do endereço) com comprimento arbitrário
  - Formato: **a.b.c.d/x**, em que x é o nº de bits correspondente à parte de rede

Encaminhamento na Internet

- ***Classless routing*** **– pesquisa de rotas:**
  - A classe não é relevante...
  - Melhor match bit a bit
    - Procurar o prefixo mais longo (mais bits), existente na tabela de encaminhamento que faz *match* com o endereço
    - Rota por defeito 0.0.0.0/0 (prefixo mais curto – 0 bit – que faz *match* com tudo)



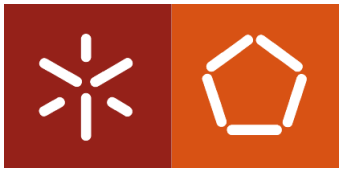
Encaminhamento na Internet



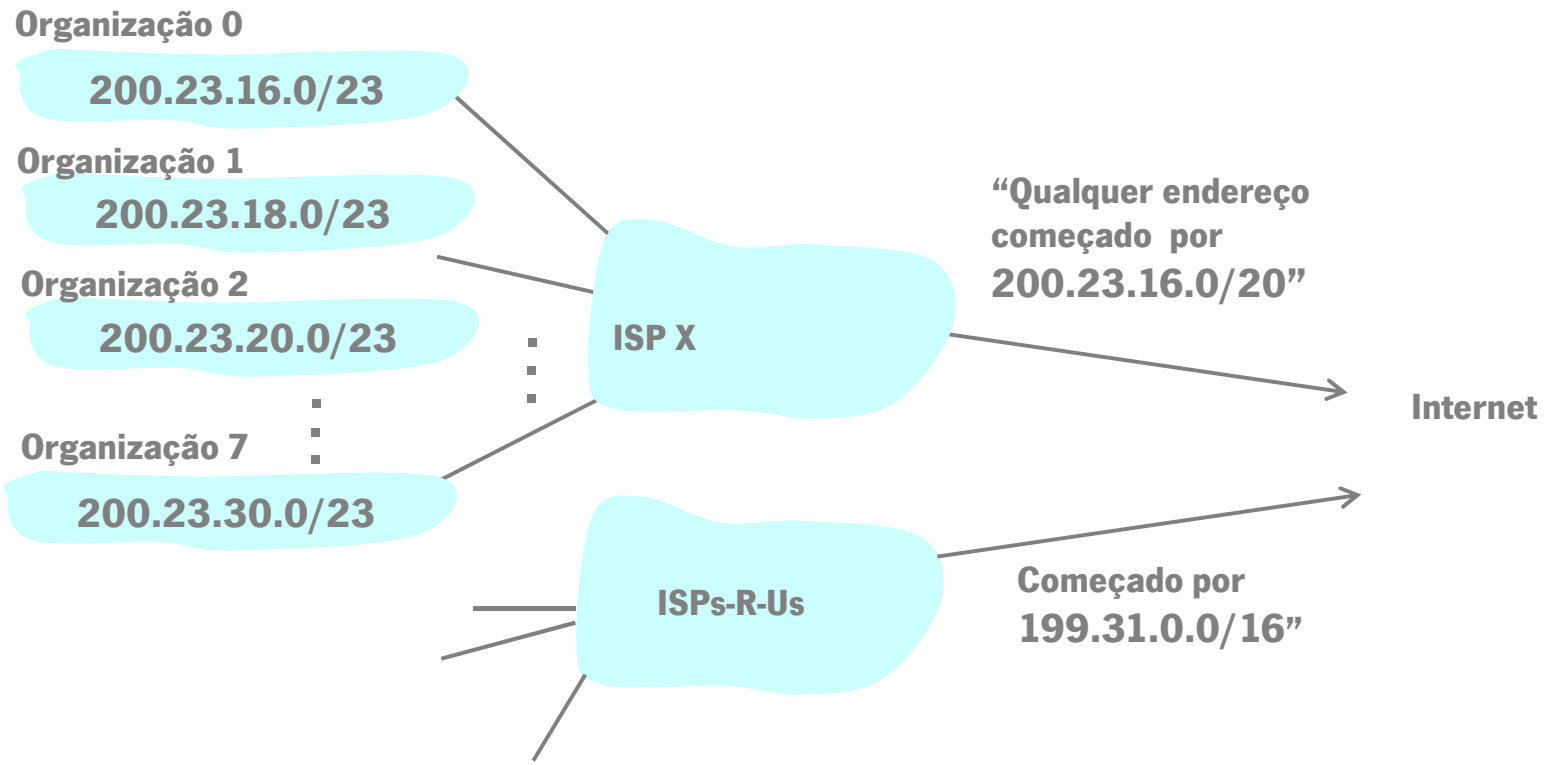
- **Porquê procurar o prefixo mais longo da tabela?**
  - Endereços são alocados via espaço de endereçamento do ISP
  - Atribuição hierárquica permite agregação...

Bloco do ISP	**11001000 00010111 00010000**	**00000000**	**200.23.16.0/20**
Organização 0	**11001000 00010111 0001000**0	**00000000**	**200.23.16.0/23**
Organização 1	**11001000 00010111 0001001**0	**00000000**	**200.23.18.0/23**
Organização 2	**11001000 00010111 0001010**0	**00000000**	**200.23.20.0/23**
...	**.....**	**....**	**....**
Organização 7	**11001000 00010111 0001111**0	**00000000**	**200.23.30.0/23**

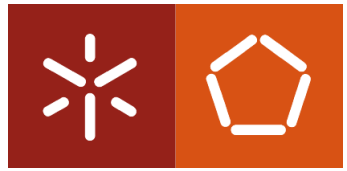
Encaminhamento na Internet



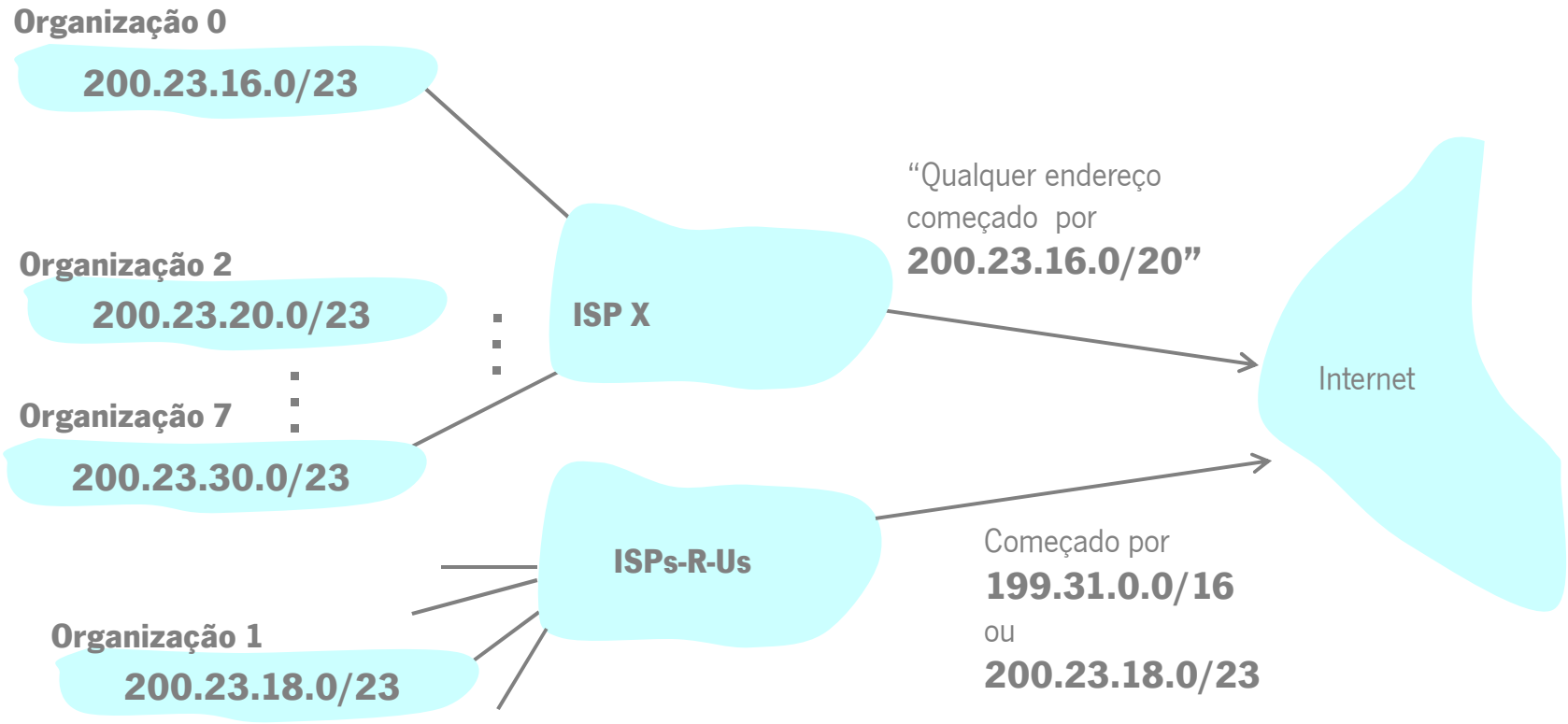
Rotas agregadas facilitam o encaminahmento



Encaminhamento na Internet

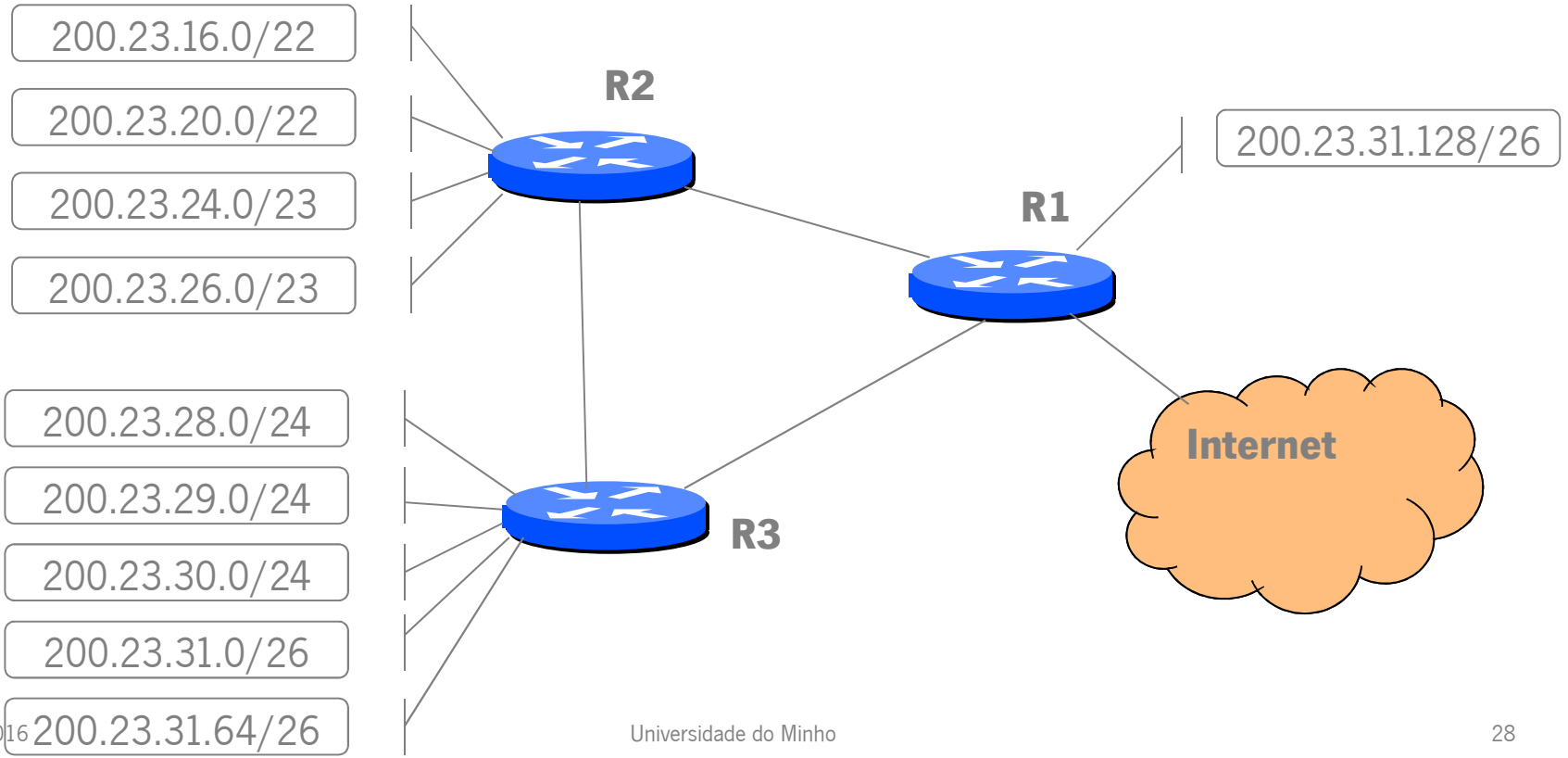


Mas se organização 1 mudar de ISP...
... ISPs-R-Us tem uma rota mais específica para a Organização 1



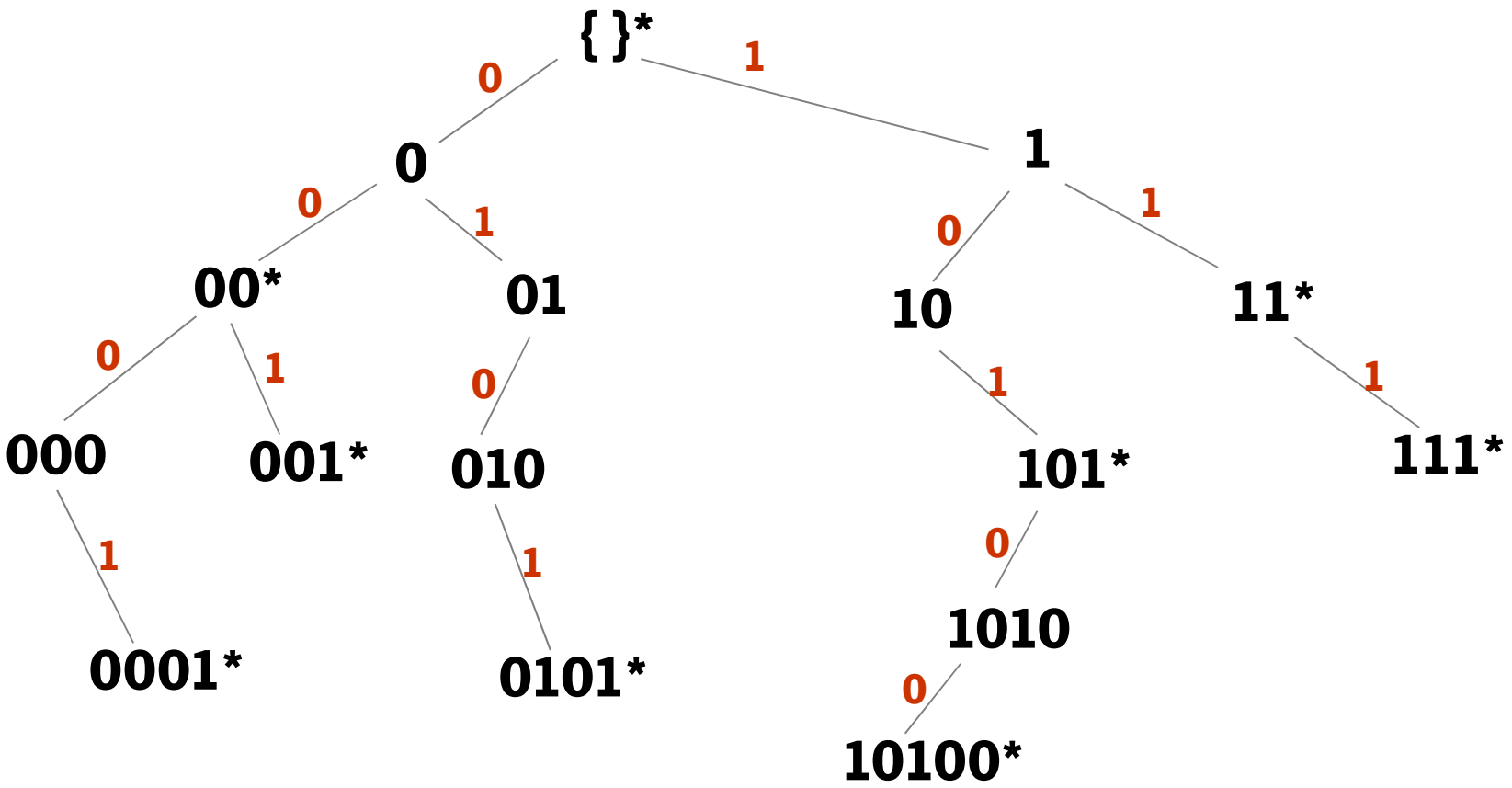
Exercício

- Usando a gama de endereços privada 192.168.0.0/16 para endereços de interligação, apresente a tabela de encaminhamento do router R1, usando sempre que possível, rotas agregadas.



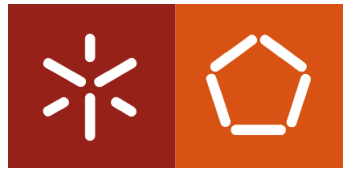
Reenvio (*forwarding*)

- Pesquisa do prefixo mais longo: estrutura *Trie*



Baseado num exemplo de "Interconnections – Second Edition", Radia Perlman, Addison-Wesley, 2002

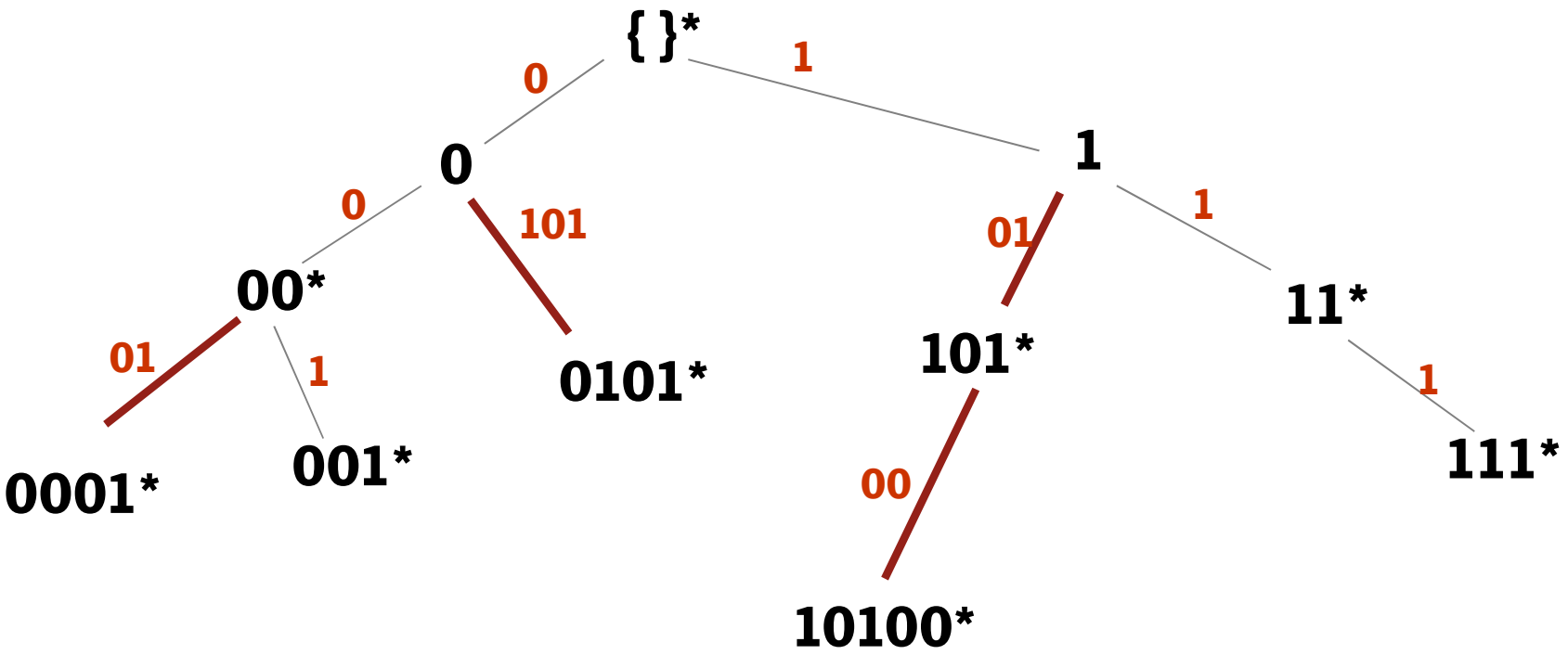
Reenvio (*forwarding*)



- **Estrutura de dados: árvore binária**
  - Cada nó tem dois apontadores: para vértice "0" e para o vértice 1
  - Cada vértice representa um prefixo possível; pode não ter ramos ou apontar para vértice de falha...
- **A raíz da árvore {}* consiste num prefixo de tamanho zero**
  - Faz *match* com qualquer endereço - *rota por defeito*
- **Cada vértice tem uma *flag* associada (asterisco "*") que indica se o prefixo é válido (tem rota associada)**
  - Na travessia a partir da raíz, guarda-se sempre o último vértice visitado que tinha um asterisco.
- **Os bits do endereço são analisados um a um, da esquerda para a direita, seguindo o ramo associado ao seu valor;**
  - Na figura, o melhor *match* para o prefixo 10101101 é o nó 101*
- **Tempo de pesquisa proporcional ao tamanho médio dos prefixos armazenados... pode ser optimizado**

Reenvio (*forwarding*)

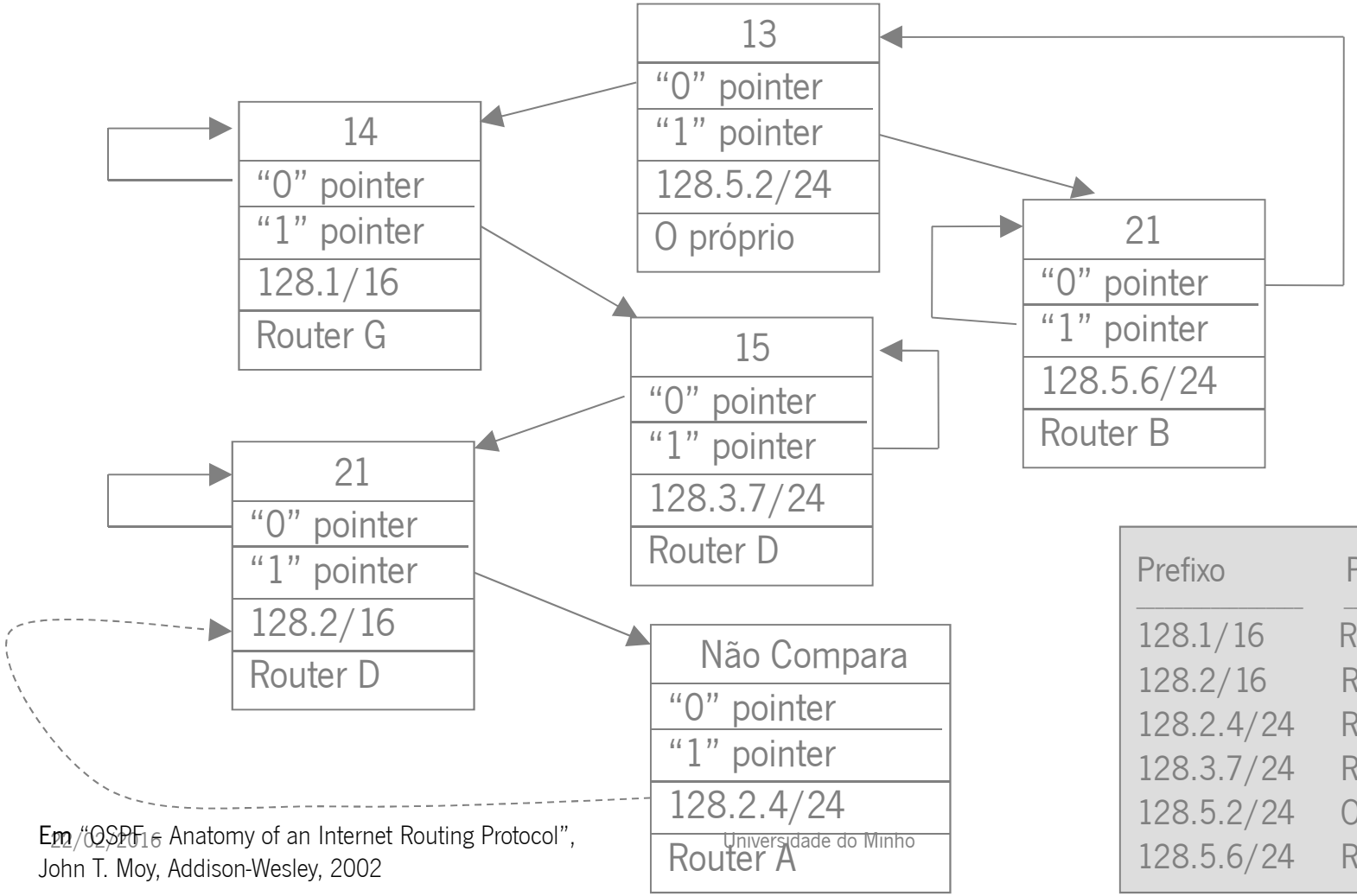
- *Trie* (juntado vértices sem ramificações e sem asteriscos)



Baseado num exemplo de "Interconnections – Second Edition", Radia Perlman, Addison-Wesley, 2002

Reenvio (*forwarding*)

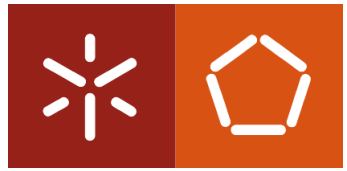
- **Pesquisa do prefixo mais longo: *Patricia Tree*** (variante da Trie)



Prefixo	Próximo salto
128.1/16	Router G
128.2/16	Router D
128.2.4/24	Router A
128.3.7/24	Router D
128.5.2/24	O próprio
128.5.6/24	Router B

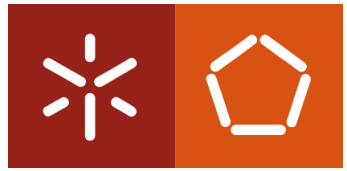
Em "OSPF - Anatomy of an Internet Routing Protocol", John T. Moy, Addison-Wesley, 2002

Reenvio (*forwarding*)



- ***Patricia*: árvore binárias**
  - travessia feita com base nalguns bits do endereço
  - Travessia pára num nó com nº de bit menor ou igual ao último testado, ou sem indicação de bit a testar;
- **A pesquisa de 128.2.5.6 no exemplo anterior:**
  - Inicio no bit 13 (estando numerados de 0 a 31);
  - O bit 13 é zero, portanto travessia para nó com o bit 14;
  - Testar bit 14, que é 1; avançar para bit 15;
  - Bit 15 é 0, travessia para nó com bit 21;
  - Testar bit 21, que é 1; Vai para num nó sem bit para comparar e a travessia pára;
  - Como não faz match com o valor do prefixo armazenado nesse nó, é o nó anterior que tem o melhor *match*!

Reenvio (*forwarding*)



- **Existem outros algoritmos... estes são apenas um exemplo**
- **Nos routers com ligações GBit, o tempo de pesquisa na tabela é crítico..**
- **Uma forma de pesquisar rapidamente é evitar a pesquisa:**
  - Alguns *routers* usam uma *cache* de rotas;
  - A pesquisa na *cache* é mais fácil: *hash* do endereço destino (match exacto)
  - Pacotes para novos destinos implicam pesquisa na tabela e demoram mais tempo a ser expedidos...
  - Funciona bem dentro das organizações, mas não parece muito útil no núcleo da Internet, pela enorme variedade de destinos possíveis;

Modelo Abstracto de Rede

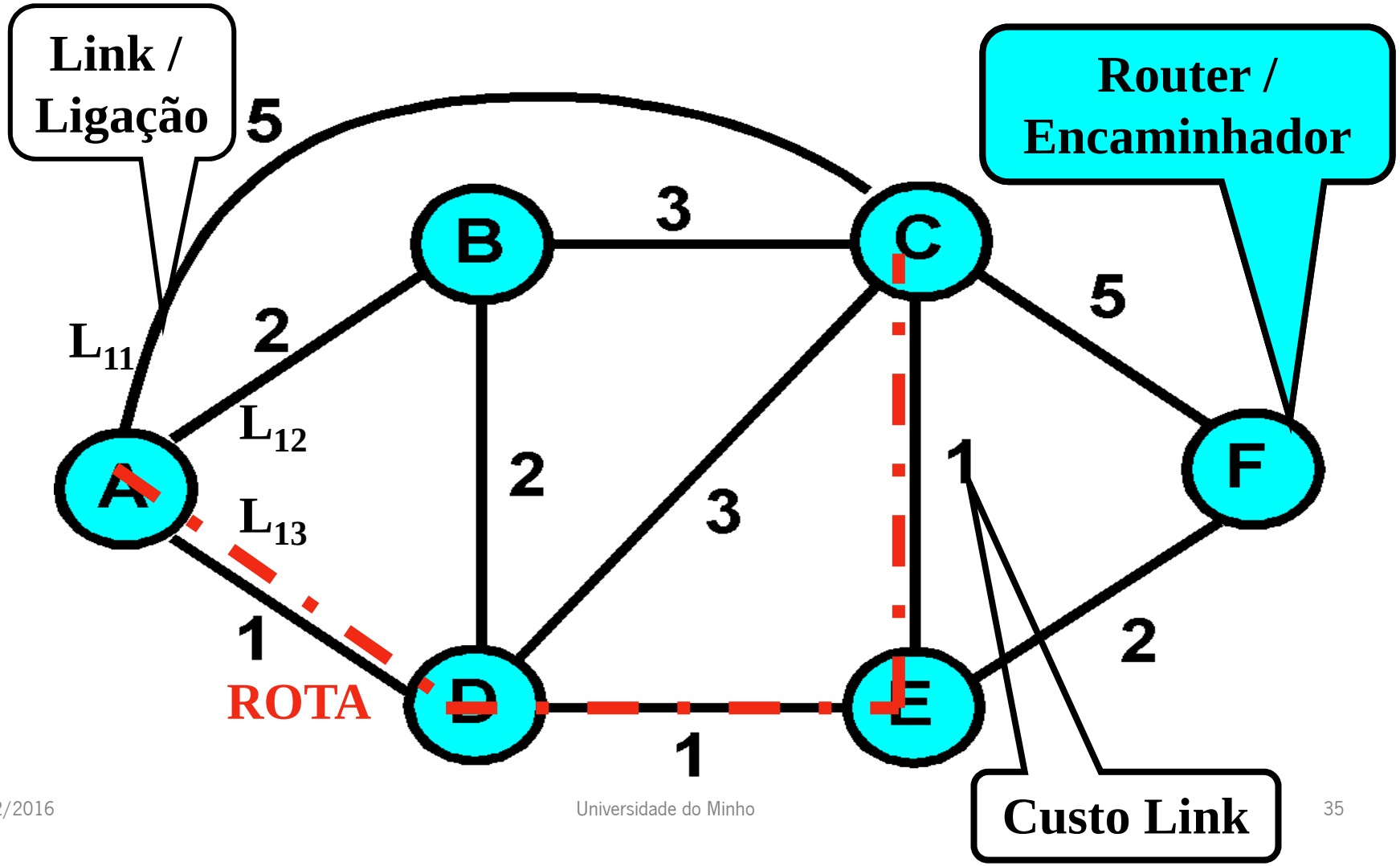
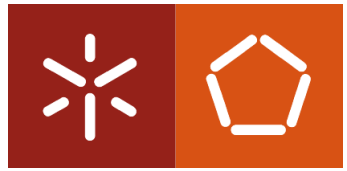


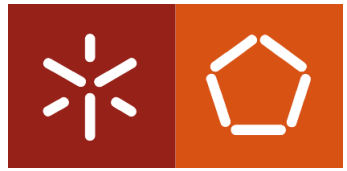
Tabela de Encaminhamento



NÓ A

Destino	Próximo Nó	Link	Custo
A	A		0
...	...	...	...
C	D	L_{13}	3
...	...	...	...
E	D	L_{13}	2

Encaminhamento em redes IP



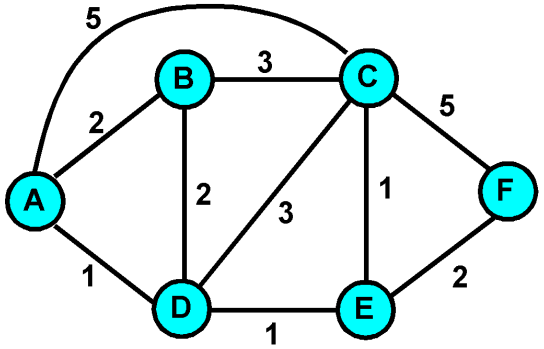
Algoritmo Encaminhamento

Objectivo: dado um conjunto de encaminhadores com ligações de rede a interligá-los o objectivo do algoritmo de encaminhamento é determinar um "bom" caminho desde a fonte até ao destino

Abstracção:

- **A topologia de rede é um Grafo:**
  - Os nós do grafo são os encaminhadores
  - Os arcos do grafo são as ligações da rede
  - O custo das ligações pode ser estabelecido em função do atraso, da capacidade, do nível congestão, do custo, da distância, etc
  - Um "bom" caminho significa tipicamente o caminho de "custo mínimo", mas há outras possibilidades...

Abstracção: a rede como um grafo



Grafo: G = (N,E)

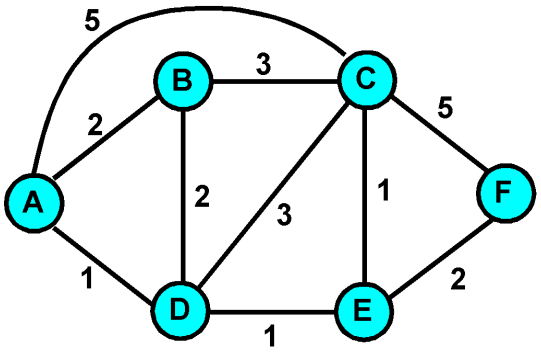
N = conjunto de routers = { a, b, c, d, e, f }

E = conjunto de links ={ (a,b), (a,d), (b,c), (b,d), (d,c), (d,e), (e,c), (c,f), (e,f) }

Nota: Esta abstracção é útil noutros contextos de rede

Exemplo: P2P, onde N é o conjunto de peers e E o conjunto de conexões TCP

Abstracção: a rede como um grafo



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

  - exemplo c(a,c) = 5

- Custo poderia ser sempre 1, inversamente proporcional à largura de banda ou inversamente proporcional à congestão

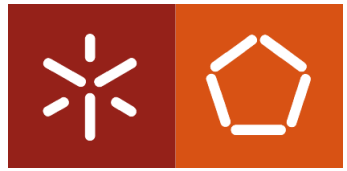
Custo de um caminho $(x_1, x_2, x_3, \ldots, x_p) = c(x_1,x_2) + c(x_2,x_3) + \ldots + c(x_{p-1},x_p)$

Qual o caminho de custo mínimo entre a e f?

Algoritmos de encaminhamentos: procuram caminhos de custo mínimo

Encaminhamento em redes IP

Classificação dos Algoritmos de Encaminhamento



Informação Global ou Descentralizada?

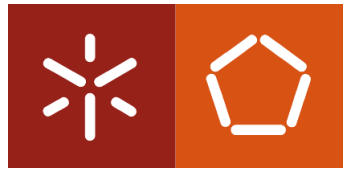
Global:

- Todos os encaminhadores têm um conhecimento completo da topologia e custo das ligações
- Algoritmos de estado das ligações (LS-"link state")

Descentralizada:

- Os encaminhadores só conhecem os vizinhos (fisicamente ligados) e o custo das ligações respectivas
- processo de computação é iterativo, troca de informação com os vizinhos
- Algoritmos de vector de distância (DV-"distance vector")

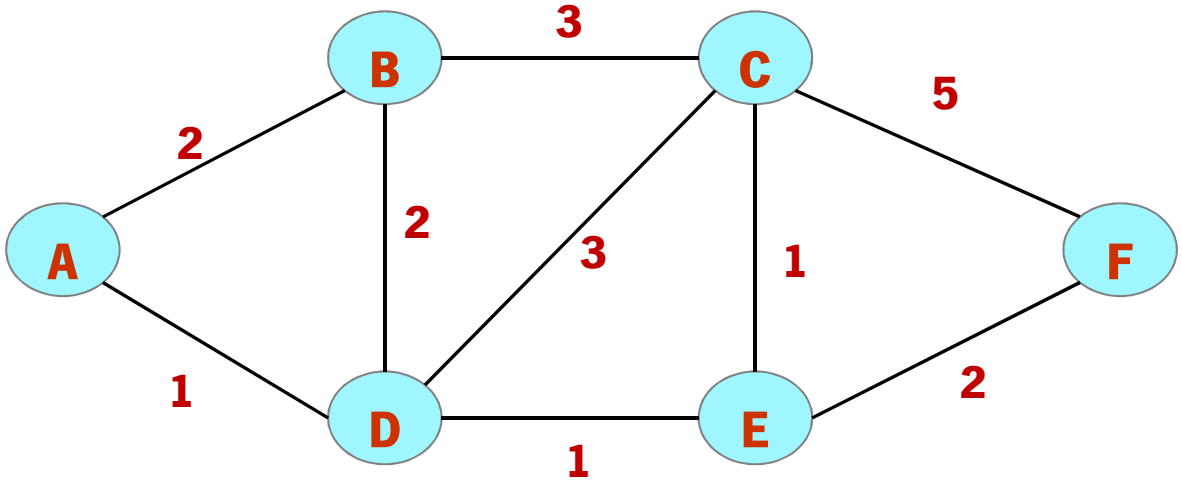
Algoritmos do Estado das Ligações (LSA)



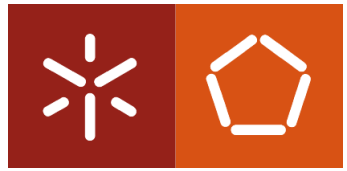
- **Todos os nós da topologia espalham pela rede o "estado das suas ligações" de forma a construírem a "base de dados topológica"**
  - Inicialmente necessitam de conhecer apenas os seus vizinhos directos, para quem enviam a identificação de todos os seus vizinhos bem como o custo das ligações que os separam deles
  - Um encaminhador ao receber esta informação actualiza a sua base de dados topológica e re-envia a informação para todos os seus vizinhos
  - Ao fim de algum tempo todos os nós possuem um conhecimento completo da topologia e dos custos de todas as ligações
- **Sobre esta informação, em cada encaminhador, é utilizado um algoritmo de descoberta dos caminhos de custo mínimo, tipicamente o algoritmo de Dijkstra.**
- **Com o resultado obtido da aplicação do algoritmo de Dijkstra, é preenchida a tabela de encaminhamento.**

Exercício

- Usando um algoritmo de estado da ligação, qual a visão topológica da rede que tem o nó A no final da primeira iteração de troca de LSAs?
- Seria correcto calcular as rotas nesta fase? E no final da segunda iteração?

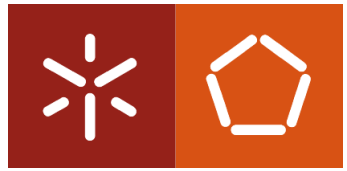


Algoritmo de Dijkstra



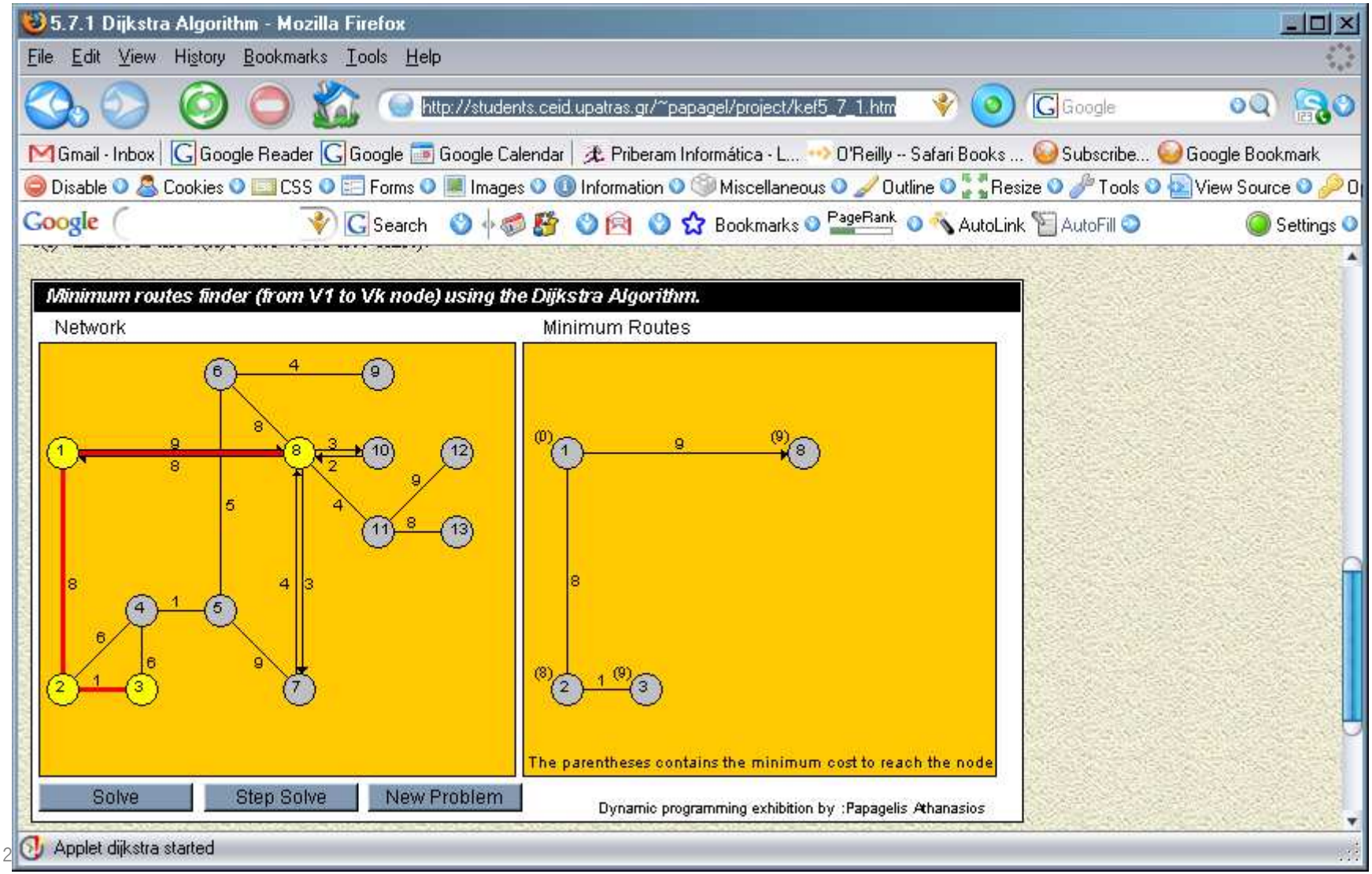
- **Algoritmo Iterativo que ao fim de k iterações consegue descobrir os caminhos de custo mínimo de um determinado nó para k destinos**
  - Seja c(i,j) o custo da ligação do nó i para o nó j. Se o nó i e o nó j não estão directamente ligados c(i,j)=∞;
  - Seja D(v) o custo do caminho desde o nó origem até ao nó v
  - Seja p(v) o nó que antecede v no caminho desde o nó origem até ao nó v
  - Seja N o conjunto de todos os nós para os quais já se conhece o caminho de custo mínimo

Algoritmo de Dijkstra

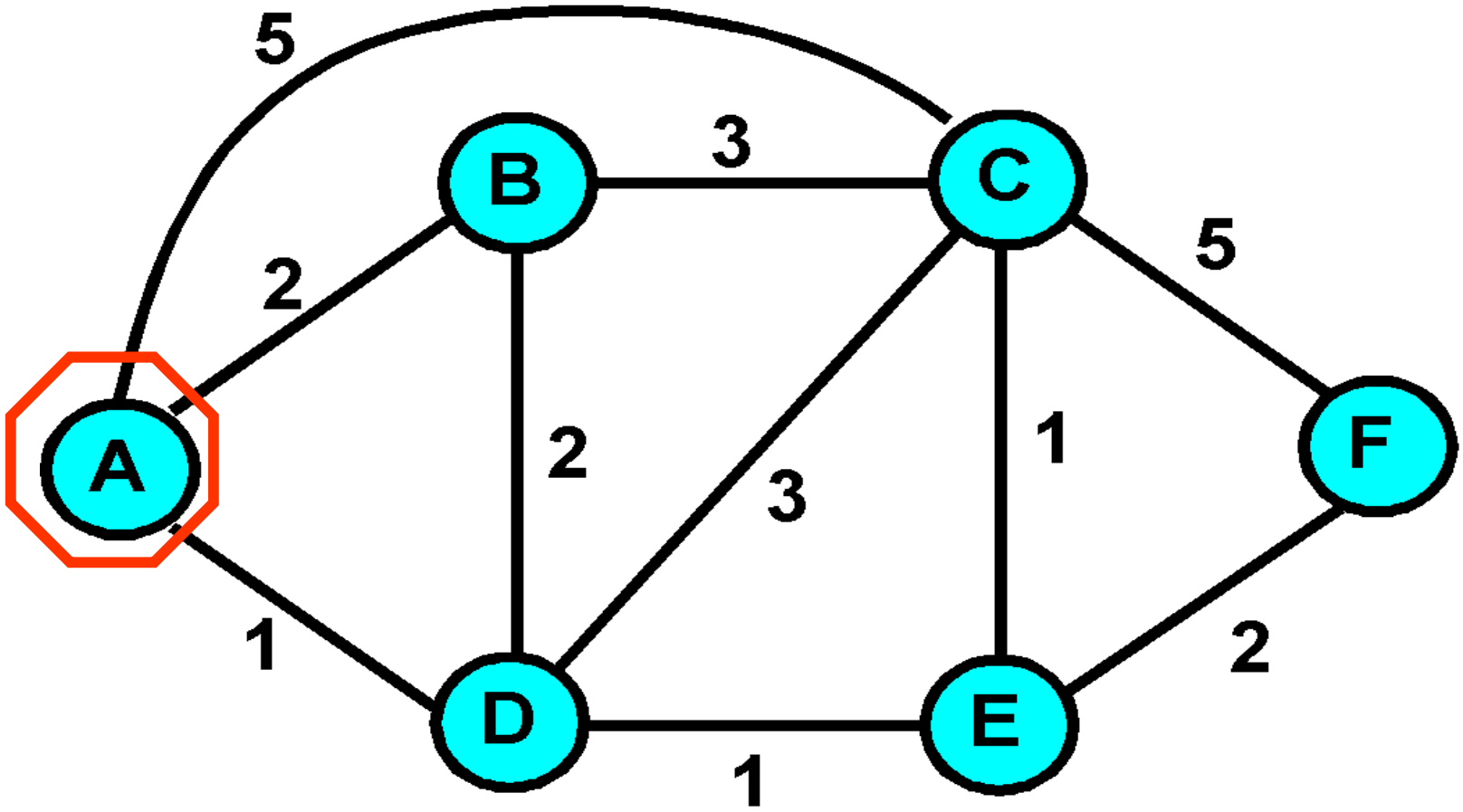


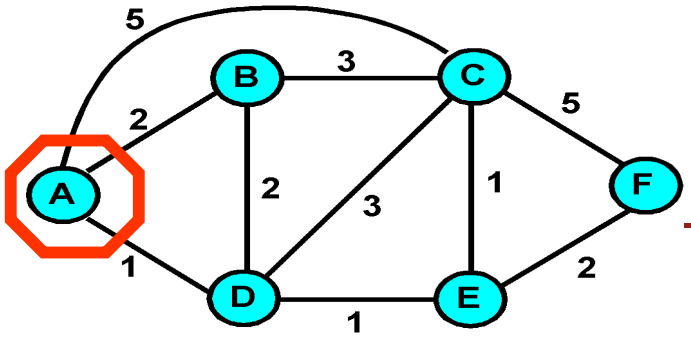
```
1  Initialization:
2    N = {A}
3    for all nodes v
4      if v adjacent to A
5        then D(v) = c(A,v); P(v)=A;
6        else D(v) = infty;
7
8  Loop
9      find w not in N such that D(w) is a minimum
10     add w to N
11     update D(v) for all v adjacent to w and not in N:
12           if  ( D(v) >  (D(w) + c(w,v)) )
13                D(v) =  D(w)+c(w,v); p(v) = w;
14           /* new cost to v is either old cost to v or known
15           shortest path cost to w plus cost from w to v */
16 until all nodes in N
```

Dijkstra


5.7.1 Dijkstra Algorithm - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://students.ceid.upatras.gr/~papagel/project/kef5_7_1.htm
Minimum routes finder (from V1 to Vk node) using the Dijkstra Algorithm.
Network
Minimum Routes
The parentheses contains the minimum cost to reach the node
Solve
Step Solve
New Problem
Dynamic programming exhibition by :Papagelis Athanasios
Applet dijkstra started

LSA – Operação



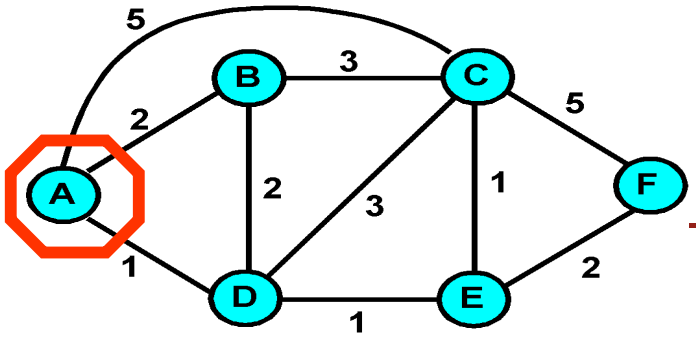


LSA (Passo 0)

step	N	D(B),p(B)	D(C),p(C)	D(D),p(D)	D(E),p(E)	D(F),p(F)
0	A	2,A	5,A	1,A	infty	infty

Steps in running the link state algorithm (LSA)

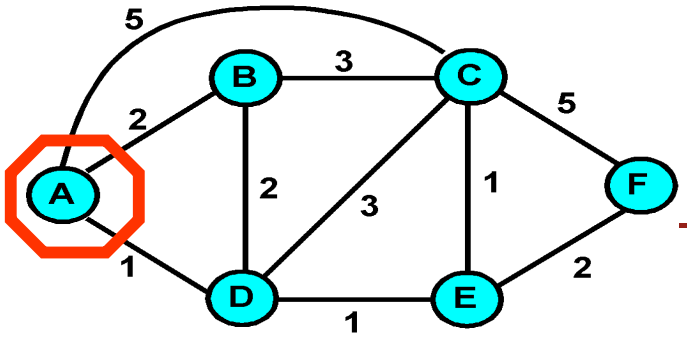
LSA (Passo 1)



step	N	D(B),p(B)	D(C),p(C)	D(D),p(D)	D(E),p(E)	D(F),p(F)
0	A	2,A	5,A	1,A	infty	infty
1	AD	2,A	4,D		2,D	infty

Steps in running the link state algorithm (LSA)

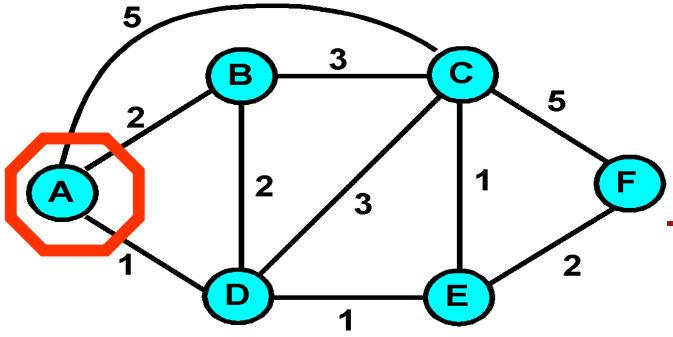
LSA (Passo 2)



step	N	D(B),p(B)	D(C),p(C)	D(D),p(D)	D(E),p(E)	D(F),p(F)
0	A	2,A	5,A	1,A	infty	infty
1	AD	2,A	4,D		2,D	infty
2	ADE	2,A	3,E			4,E

Steps in running the link state algorithm (LSA)

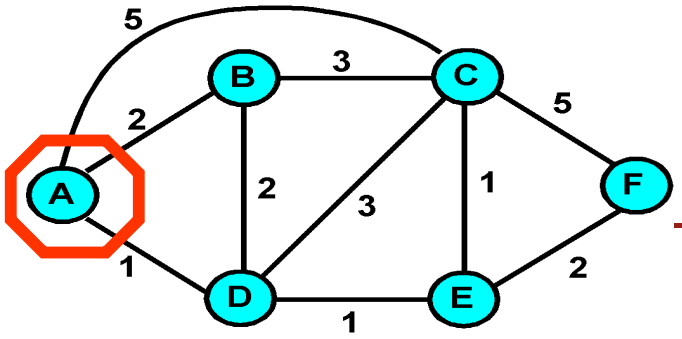
LSA (Passo 3)



step	N	D(B),p(B)	D(C),p(C)	D(D),p(D)	D(E),p(E)	D(F),p(F)
0	A	2,A	5,A	1,A	infty	infty
1	AD	2,A	4,D		2,D	infty
2	ADE	2,A	3,E			4,E
3	ADEB		3,E			4,E

Steps in running the link state algorithm (LSA)

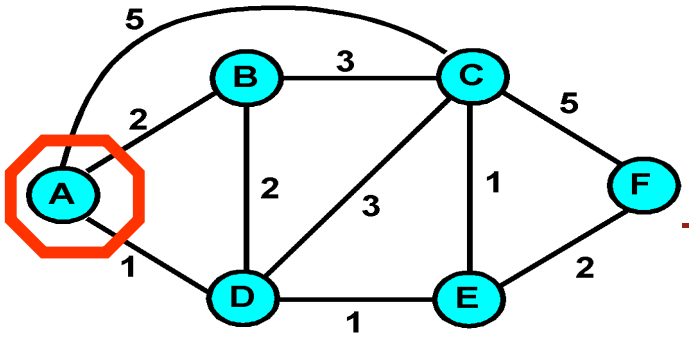
LSA (Passo 4)



step	N	D(B),p(B)	D(C),p(C)	D(D),p(D)	D(E),p(E)	D(F),p(F)
0	A	2,A	5,A	1,A	infty	infty
1	AD	2,A	4,D		2,D	infty
2	ADE	2,A	3,E			4,E
3	ADEB		3,E			4,E
4	ADEBC					4,E

Steps in running the link state algorithm (LSA)

LSA (Passo 5)



step	N	D(B),p(B)	D(C),p(C)	D(D),p(D)	D(E),p(E)	D(F),p(F)
0	A	2,A	5,A	1,A	infty	infty
1	AD	2,A	4,D		2,D	infty
2	ADE	2,A	3,E			4,E
3	ADEB		3,E			4,E
4	ADEBC					4,E
5	ADEBCF					

Steps in running the link state algorithm (LSA)

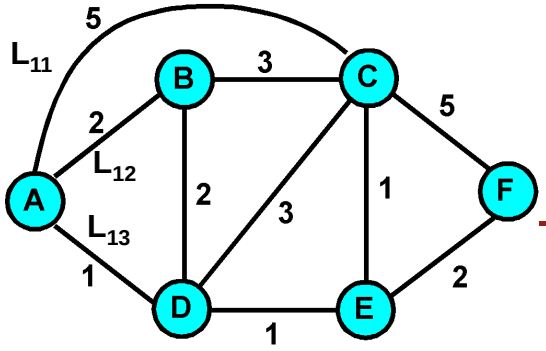


Tabela de Encaminhamento Resultante

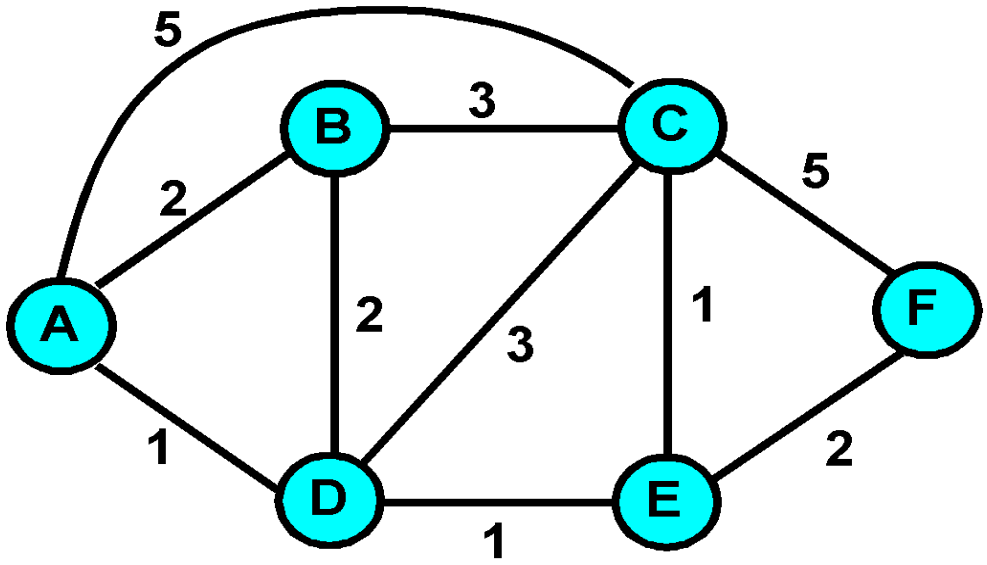
NÓ A

Destino	Próximo Nó	Link	Custo
A	A	—	0
B	B	L_{12}	2
C	D	L_{13}	3
D	D	L_{13}	1
E	D	L_{13}	2

Algoritmos do Estado das Ligações (LSA)

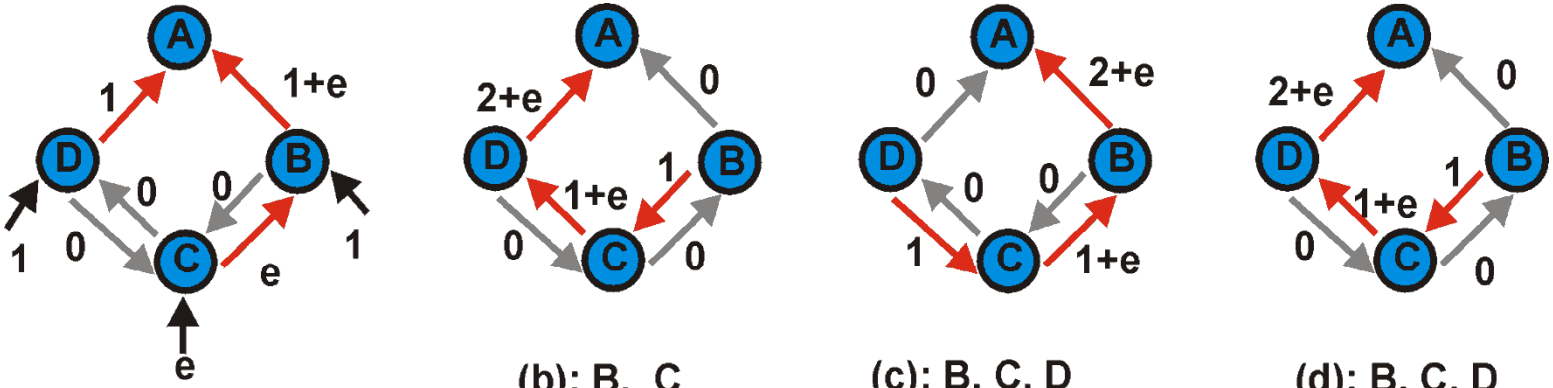
- ***Exercício:***

  Utilizando o algoritmo de Dijkstra determine a tabela de encaminhamento do nó F, partindo do princípio que este nó tem um conhecimento completo da topologia da rede.



LSA – Oscilações

- **Este algoritmo pode, na presença de métricas que dependem do estado da rede, apresentar alguns problemas**
  - Por exemplo, se a métrica reflectir a carga nas ligações, sendo por isso uma métrica assimétrica
  - No exemplo, B e D enviam uma unidade de tráfego para A e C envia *e* unidades de tráfego também para A



(a): initial routing

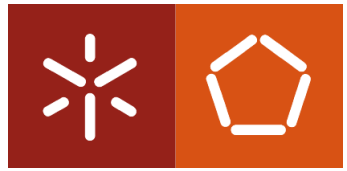
(b): B, C detect better path to A, clockwise

(c): B, C, D detect better path to A, counterclockwise

(d): B, C, D detect better path to A, clockwise

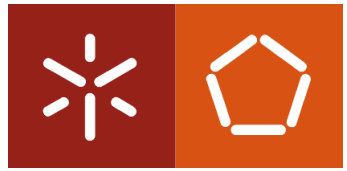
Fonte: Computer Networking: A Top-Down Approach Featuring the Internet, J. Kurose, Addison-Wesley, 2001

Algoritmos de Vectores de Distância (DVA)



- Ao contrário dos algoritmos de estado de ligação, os algoritmos de vectores de distância não usam informação global;
- São distribuídos, iterativos e assíncronos
  - Cada nó recebe informação de encaminhamento de algum dos seus vizinhos directos, recalcula a tabela de encaminhamento e envia essa informação de volta para os vizinhos;
  - O processo continua até que não haja informação de encaminhamento a ser trocada entre nós vizinhos
  - Não exige que os nós estejam sincronizados uns com os outros

Algoritmos de Vectores de Distância (DVA)



Equação de Bellman-Ford

Seja

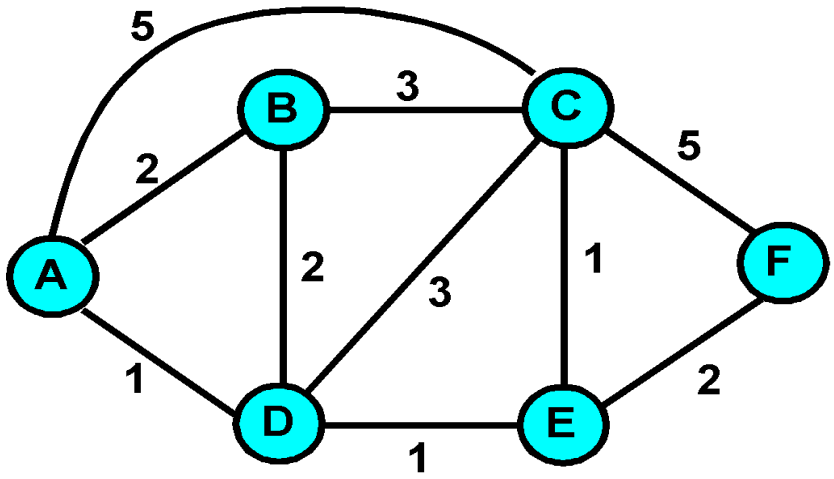
$d_x(y)$:= custo do caminho de custo mínimo de x para y

Então

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

Sendo o mínimo obtido de todos os vizinhos v de y

Exemplo



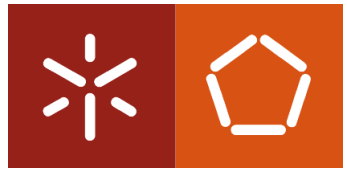
$d_B(F) = 5,\ d_C(F) = 3,\ d_D(F) = 3$

Bellman-Ford diz que:

$$d_A(F) = \min \{ c(A,B) + d_B(F),\ c(A,C) + d_C(F),\ c(A,D) + d_D(F) \}$$
$$= \min \{2 + 5,\ 1 + 3,\ 5 + 3\} = 4$$

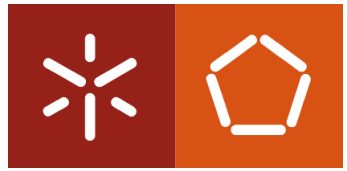
O nó vizinho que conseguir o valor mínimo
Será o próximo salto → tabela de reenvio

Algoritmos de Vectores de Distância (DVA)



- **$D_x(y)$ = estimativa do custo mínimo de x para y**
- **Nó x conhece o custo para todos os seus vizinhos v: c(x,v)**
- **Nó x mantém um vector de distâncias D_x = $[D_x(y): y \in N]$**
- **Nó x também mantém os vectores de distâncias dos seus vizinhos:**
  - Para cada vizinho v, x guarda
    D_v = $[D_v(y): y \in N]$

Algoritmos de Vectores de Distância (DVA)



A ideia básica:

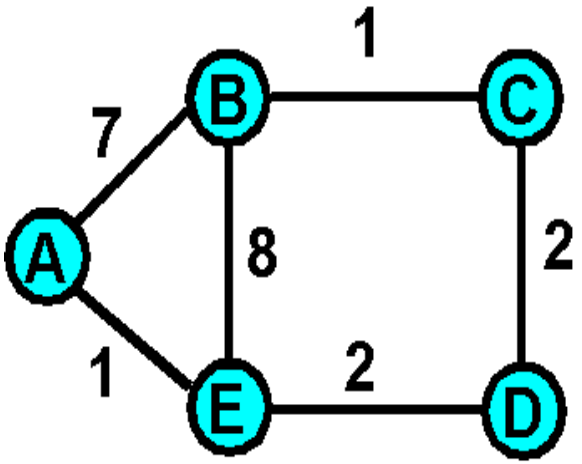
- Cada nó envia periodicamente a sua estimativa do vector de distâncias a todos os seus vizinhos
- Quando um nó x recebe um novo vector VD de um dos seus vizinhos, actualiza o seu próprio vector de distâncias usando a equação de Bellman-Ford:

$$D_x(y) \leftarrow \min_v\{c(x,v) + D_v(y)\} \quad \text{Para todos os nós } y \in N$$

- Em condições normais, a estimativa do vector distâncias $D_x(y)$ *converge para o vector de custo mínimo* $d_x(y)$

Tabela de distâncias

DVA

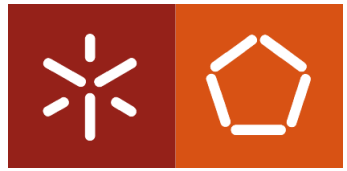


cost to destination via

$D^E()$ destinat.	A	B	D
A	(1)	14	5
B	7	8	(5)
C	6	9	(4)
D	4	11	(2)

Universidade do MinhoFonte: Computer Networking: A Top-Down Approach Featuring the Internet, J. Kurose, Addison-Wesley, 2001

Da Tabela de Distâncias à Tabela de Encaminhamento



Custo p/ o destino **via**

$D^E()$	A	B	D
A	1	14	5
B	7	8	5
C	6	9	4
D	4	11	2

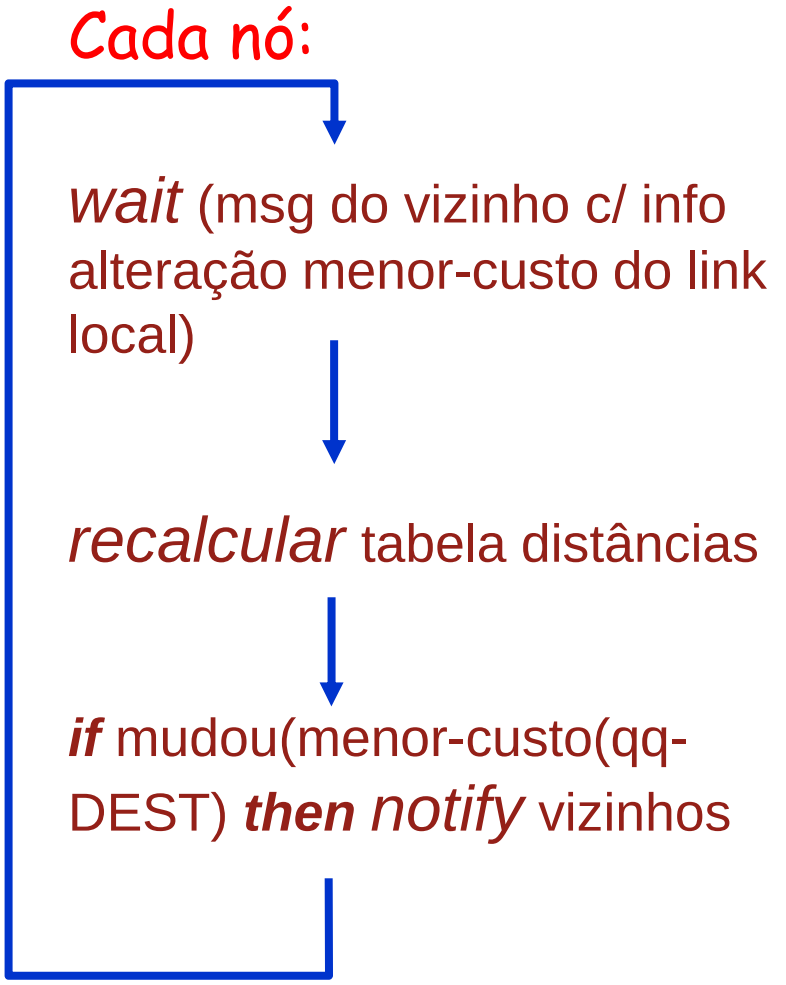
destination

Link Saída, custo

destination		
A	A,	1
B	D,	5
C	D,	4
D	D,	2

Tabela Distâncias → Tabela de Routing

Algoritmos de Vector de Distâncias (DVA)



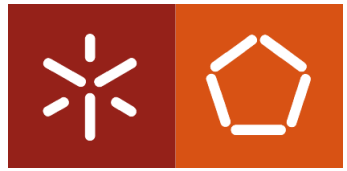
Iterativo, assíncrono: cada iteração local é causada por:

- mudança custo link local
- mensagem do vizinho: vizinho anuncia novo custo

Distribuído:

- cada nó notifica vizinhos *só* quando muda o menor custo p/ qq destino
  - vizinhos notificam vizinhos (se necessário!)

Algoritmo Belman-Ford



$$D^X(Y,Z) = \text{distance } \textit{from } X \textit{ to } Y, \textit{ via } Z \text{ as next hop}$$

$$= c(X,Z) + \min_w \{D^Z(Y,w)\}$$

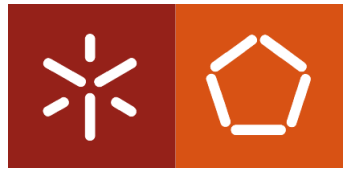
At each node, X:

```
1  Initialization:
2   for all adjacent nodes v:
3     DX(*,v) = infty   /* the * operator means "for all rows" */
4     DX(v,v) = c(X,v)
5   for all destinations, y
6     send minwD(y,w) to each neighbor  /* w over all X's neighbors */

7
```

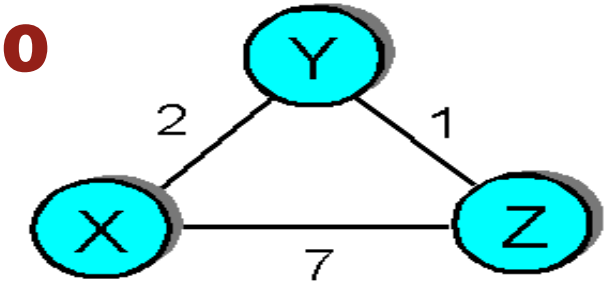
Fonte: Computer Networking: A Top-Down Approach Featuring the Internet, J. Kurose, Addison-Wesley, 2001

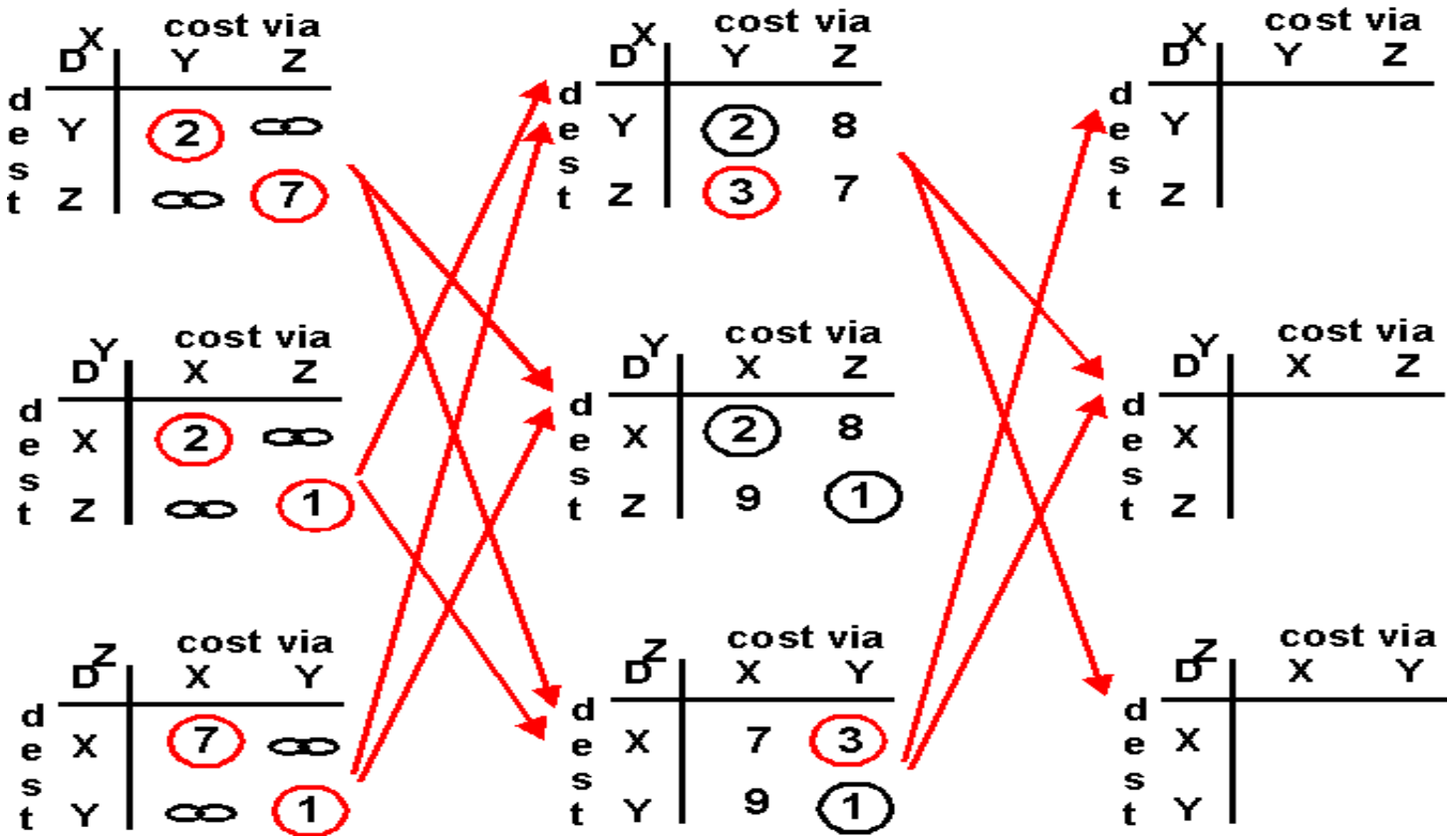
Algoritmo Belman-Ford



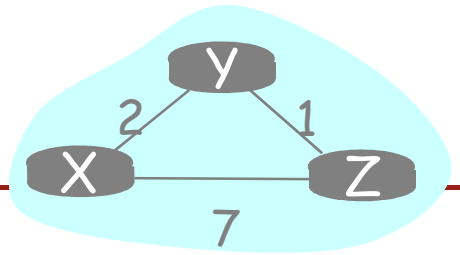
8 ***loop forever***
9 **wait** (until I see a link cost change to neighbor V
10 or until I receive update from neighbor V)
12 **if (c(X,V) changes by d)**
13 */* change cost to all dest's via neighbor v by d */*
14 */* note: d could be positive or negative */*
15 ***for*** all destinations y: $D^X(y,V) = D^X(y,V) + d$
16
17 **else if (update received from V wrt destination Y)**
18 */* shortest path from V to some Y has changed */*
19 */* V has sent a new value for its* $min_w D^V(Y,w)$ **/*
20 */* call this received new value is "newval" */*
21 ***for*** the single destination y: $D^X(Y,V) = c(X,V) + newval$
22
23 **if we have a new** $\mathbf{min}_w$ $D^X(Y,w)$ for any destination Y
24 send new value of min_w $D^X(Y,w)$ to all neighbors

DVA – Operação





DVA - exemplo



D^X	cost via Y	cost via Z
dest Y	2	∞
dest Z	∞	7

D^X	cost via Y	cost via Z
dest Y	2	8
dest Z	3	7

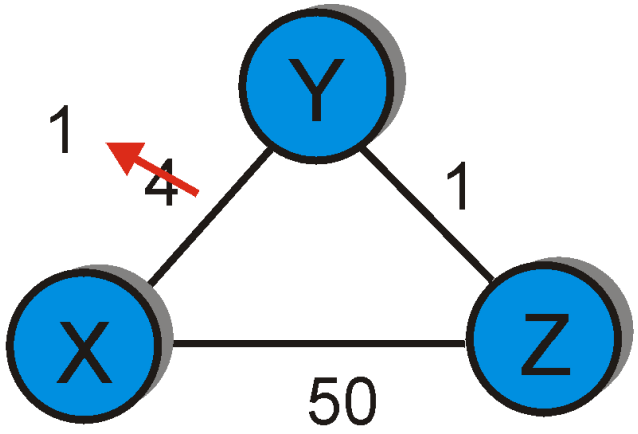
D^Y	cost via X	cost via Z
dest X	2	∞
dest Z	∞	1

D^Z	cost via X	cost via Y
dest X	7	∞
dest Y	∞	1

$$D^X(Y,Z) = c(X,Z) + \min_w\{D^Z(Y,w)\}$$
$$= 7+1 = 8$$

$$D^X(Z,Y) = c(X,Y) + \min_w\{D^Y(Z,w)\}$$
$$= 2+1 = 3$$

DVA: Good News...



D Y	via X	via Z
X	(4)	6

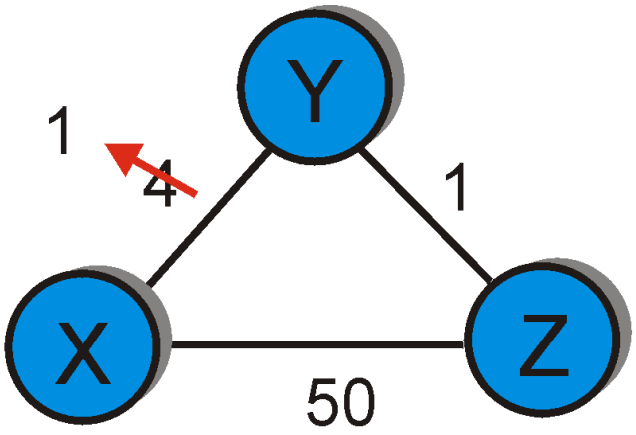
D Z	via X	via Y
X	50	(5)

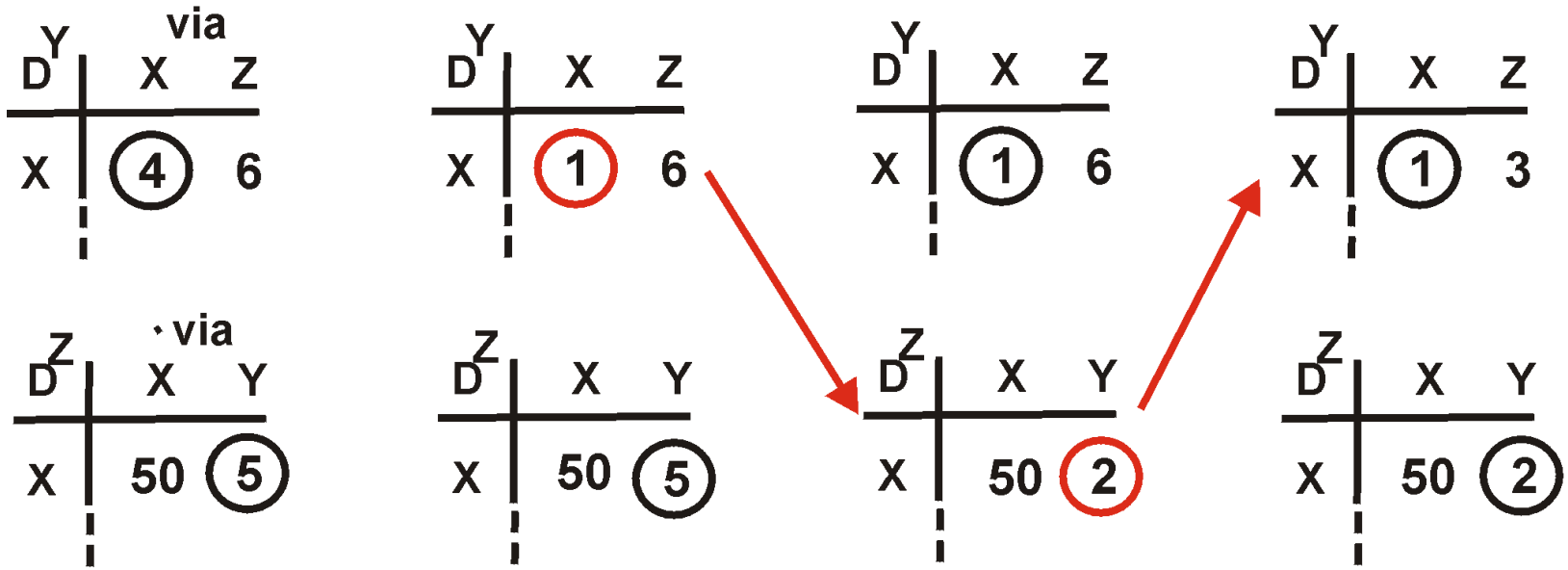
Fonte: Computer Networking: A Top-Down Approach Featuring the Internet, J. Kurose, Addison-Wesley, 2001

c(X,Y) change

time → t_0 t_1 t_2

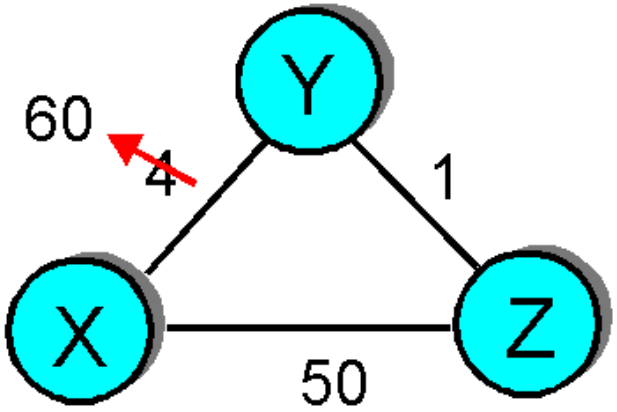
DVA: Good News... Travel Fast





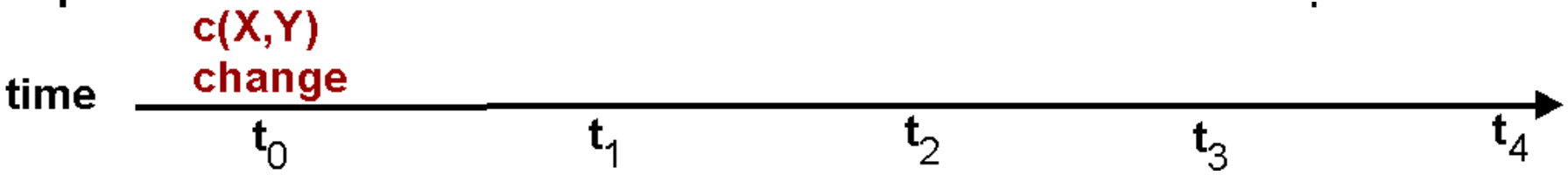
DVA: Bad News...

Fonte: Computer Networking: A Top-Down Approach Featuring the Internet, J. Kurose, Addison-Wesley, 2001

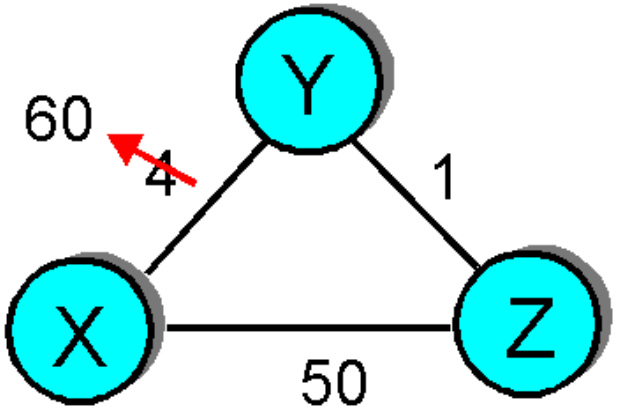


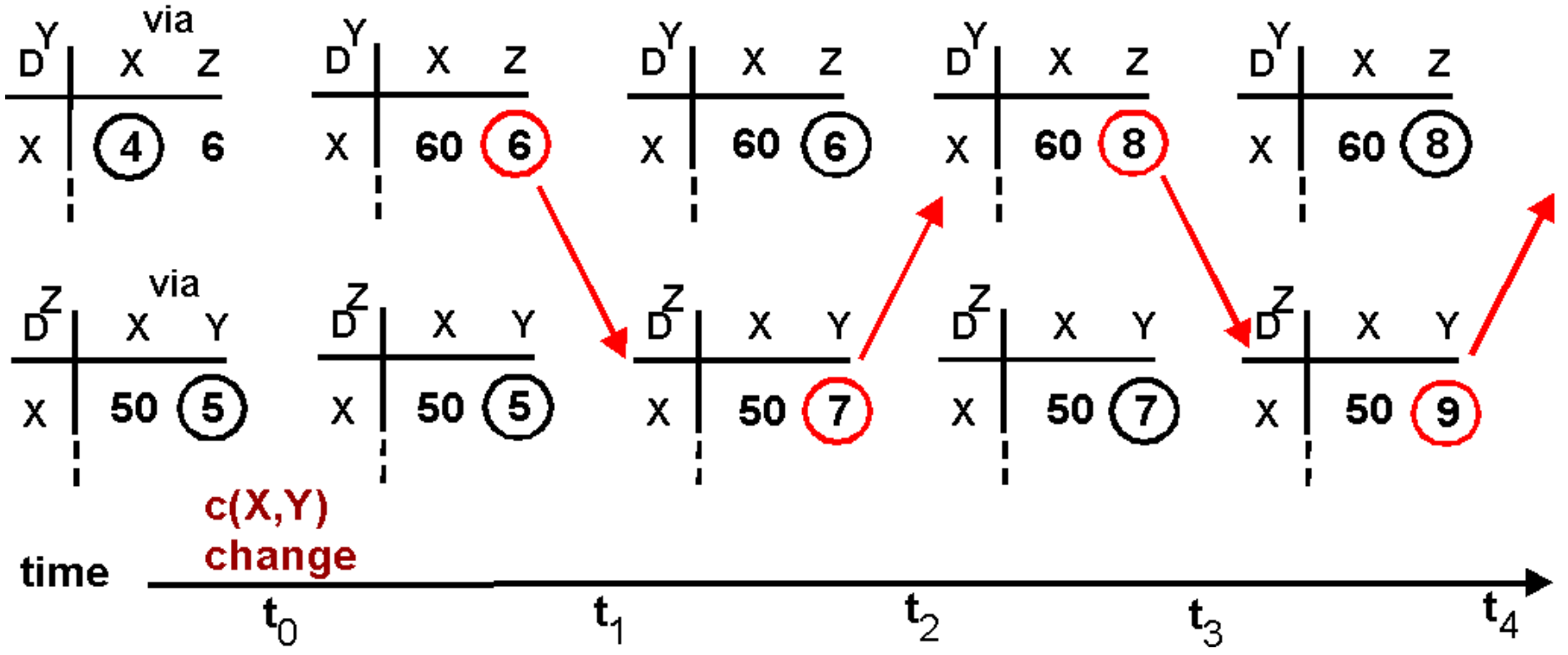
D(Y)	via X	via Z
X	(4)	6

D(Z)	via X	via Y
X	50	(5)



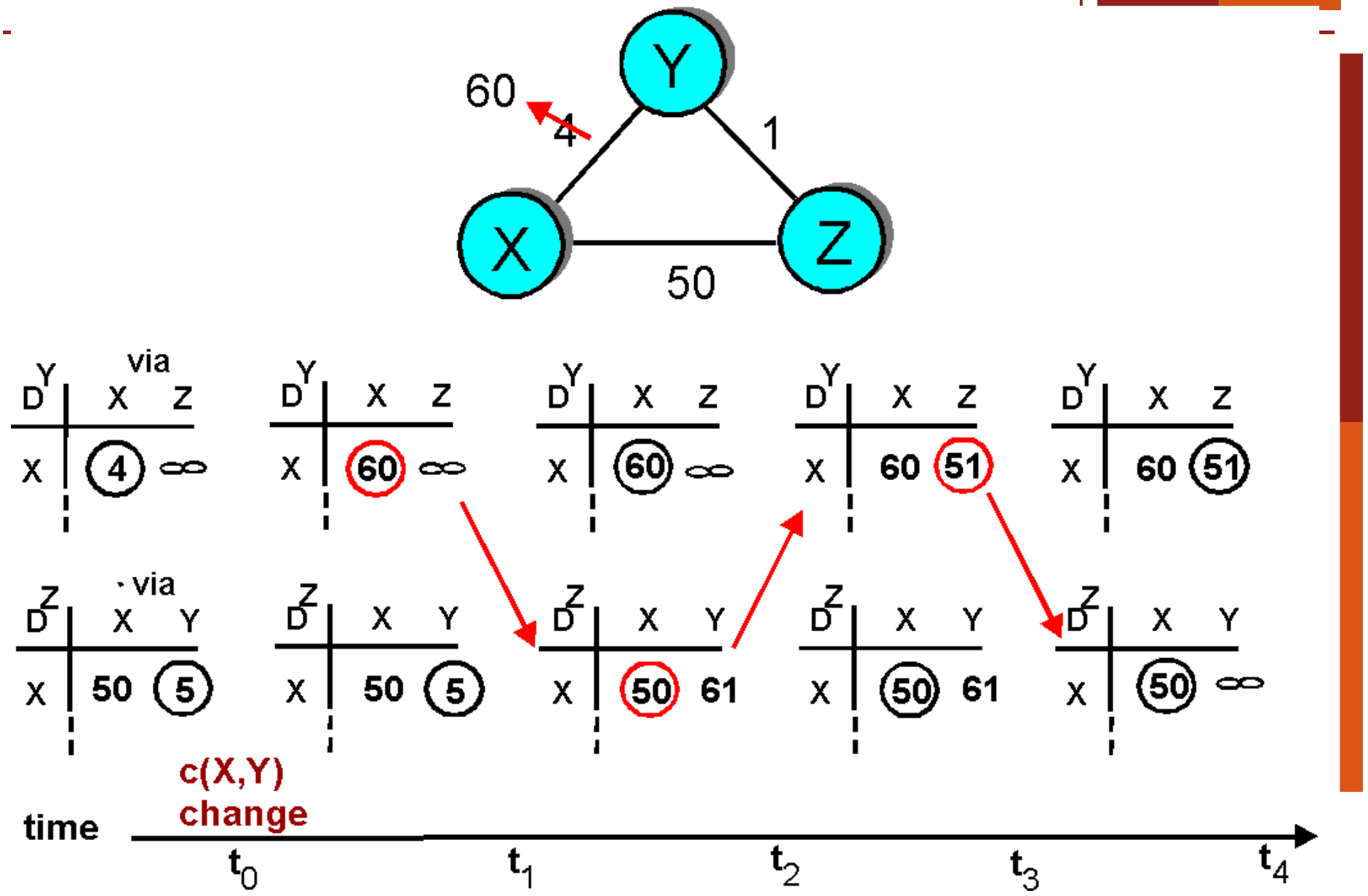
DVA: Bad News... Count to Infinity!



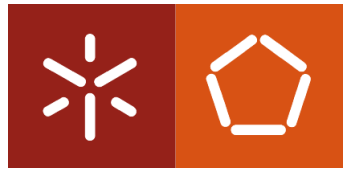


DVA: Bad News... Count to Infinity!

Fonte: Computer Networking: A Top-Down Approach Featuring the Internet, J. Kurose, Addison-Wesley, 2001



DV



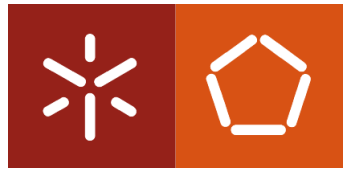
- **Soluções**
  - Divisão do horizonte

Se x aprendeu rota para z com y, nunca ensina essa rota a y!

  - Envenenamento do percurso inverso (***poison reverse***)

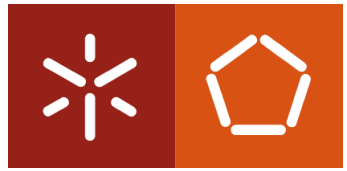
Se x aprendeu rota para z com y, então "mente" a y dizendo-lhe que o custo da sua rota para z é infinito! Dx(z) = Infinito

Estado de Ligação *versus* Vector de Distância



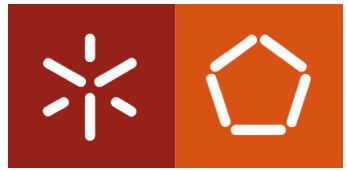
- **Sobrecarga introduzida pela mensagens de controle**
  - nos algoritmos de estado de ligação todos os nós necessitam de conhecer o custo de todas as ligações, por isso sempre que o custo de uma ligação muda, uma mensagem com o novo custo tem que ser enviada para todos os nós
  - nos algoritmos de vector de distâncias a mudança do custo de uma ligação só provoca o envio de mensagens se resultar na mudança da tabela de encaminhamento
- **Convergência**
  - os algoritmos de estado de ligação convergem mais depressa mas, com algumas métricas estão sujeitos a oscilações;
  - em contrapartida os algoritmos de vector de distâncias convergem lentamente, podem apresentar ciclos enquanto não convergem e sofrem do problema da contagem até ao infinito.

Estado de Ligação *versus* Vector de Distância

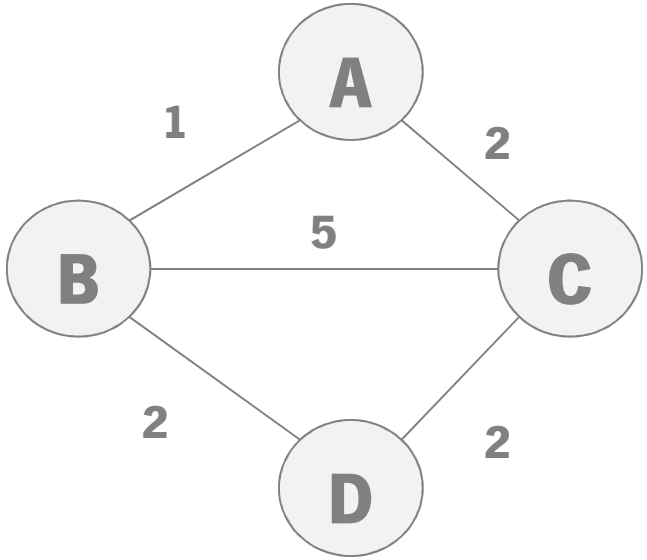


- **Robustez**
  - nos algoritmos de estado de ligação cada encaminhador calcula a sua tabela de encaminhamento usando a base de dados topológica, de forma independente dos outros encaminhadores. Isso confere a este tipo de algoritmos robustez
  - nos algoritmos de vector de distâncias se algum encaminhador estiver a calcular mal a sua tabela de encaminhamento, os erros cometidos vão-se propagar aos outros encaminhadores da topologia
- **Recursos computacionais**
  - os algoritmos de estado de ligação são mais exigentes do que os algoritmos de vector de distâncias, quer em termos de memória (base de dados topológica versus tabela de distâncias) , quer em termos de capacidade de processamento

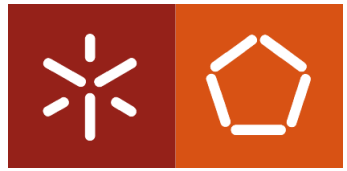
Exercício



- Qual a tabela de distâncias final do nó A?
- Nessas circunstâncias, que anúncios faz/faria A para os seus vizinhos, usando divisão do horizonte e envenenamento do percurso inverso?

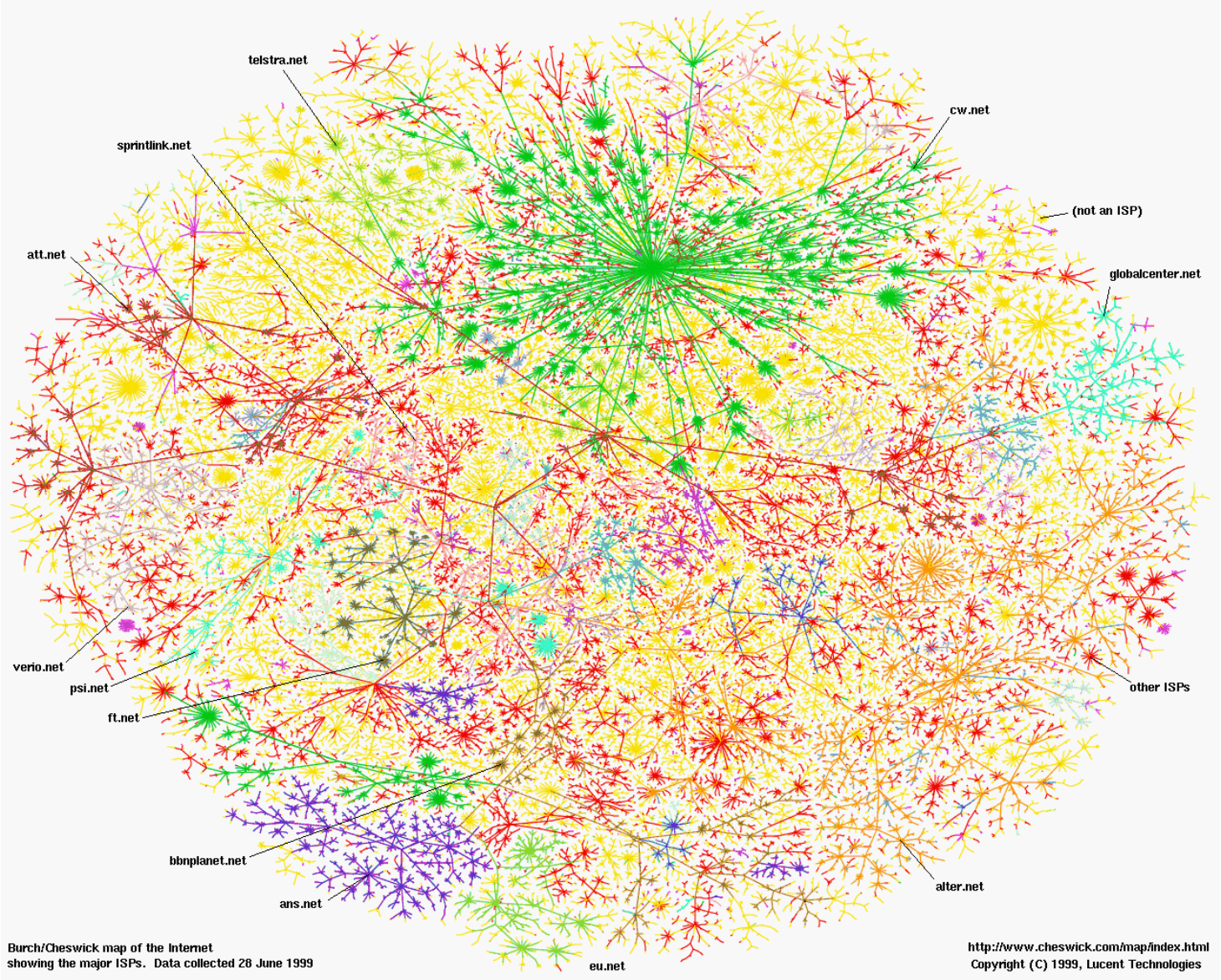


Protocolos

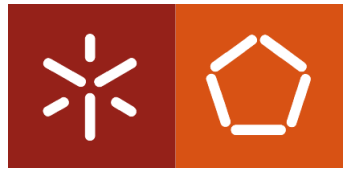


- **Baseados Estado da Ligação:**
  - OSPF – Open Shortest Path First
  - OSI ISIS – OSI Intermediate System to Intermediate System Routing Protocol
- **Baseados no Vector Distância**
  - RIP – Routing Information Protocol
    - Existe em todos os sistemas operativos (routed)
  - IGRP – Interior Gateway Routing Protocol (CISCO)
  - EIGRP – Extended IGRP (CISCO)
- **Implementações Open Source**
  - Zebra → Quagga (para Unix e Unix like systems)

Encaminhamento na Internet

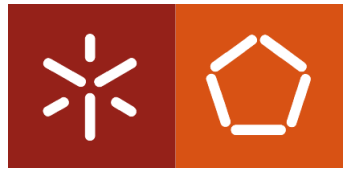


Encaminhamento na Internet



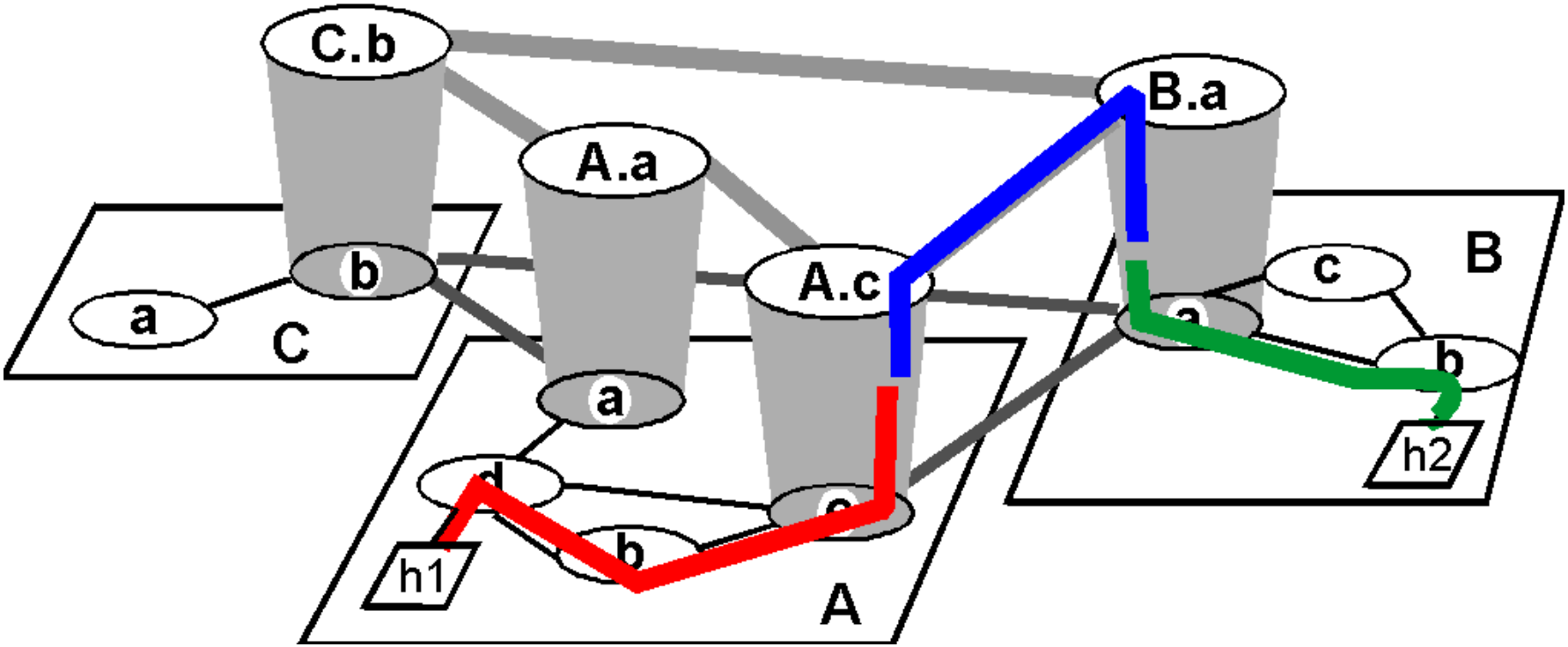
- **Por razões de escala e de autonomia administrativa, a internet não pode ser encarada como uma topologia de rede onde todos os encaminhadores executam o mesmo algoritmo de encaminhamento para encontrar os melhores caminhos para todos os destinos possíveis**
  - O número de encaminhadores é demasiado grande o que torna a sobrecarga necessária ao cálculo, armazenamento e comunicação da informação de encaminhamento demasiado grande;
  - Idealmente uma organização deveria poder escolher o algoritmo de encaminhamento que deseja utilizar nas suas redes;

Encaminhamento na Internet



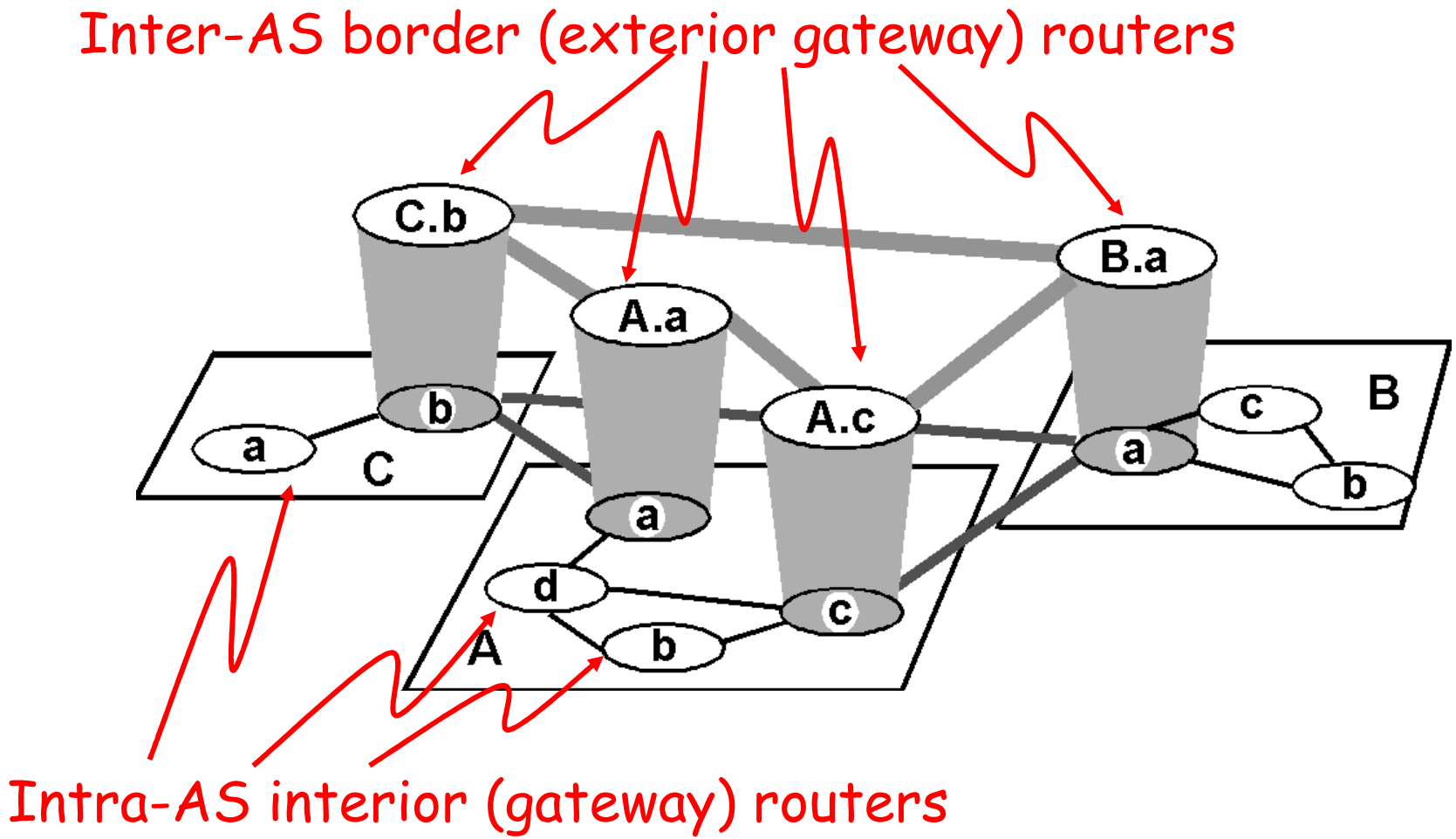
- **Estes problemas são resolvidos agregando os encaminhadores em Sistemas Autónomos (AS – Autonomous Systems).**
  - Os encaminhadores dentro de um mesmo sistema autónomo utilizam todos o mesmo algoritmo de encaminhamento (LS ou DV) e possuem informação acerca de todos os encaminhadores que fazem parte do sistema autónomo.
  - Os protocolos de encaminhamento que se utilizam no interior de um sistema autónomo designam-se por protocolos intra-domínio (***intradomain routing protocols***) ou internos (***IGP - Interior Gateway Protocol***)
- **Para interligar os diferentes Sistemas Autónomos entre si é necessário utilizar pelo menos um encaminhador por Sistema Autónomo e com eles constituir uma rede de "nível superior".**
  - Esses encaminhadores além de executarem o protocolo intra-domínio, utilizam um protocolo de encaminhamento inter-domínio (***interdomain routing protocol***) ou externos (***EGP – Exterior Gateway Protocol***)

Encaminhamento na Internet



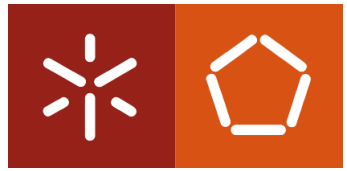
Fonte: Computer Networking: A Top-Down Approach Featuring the Internet, J. Kurose, Addison-Wesley, 2001

Encaminhamento na Internet



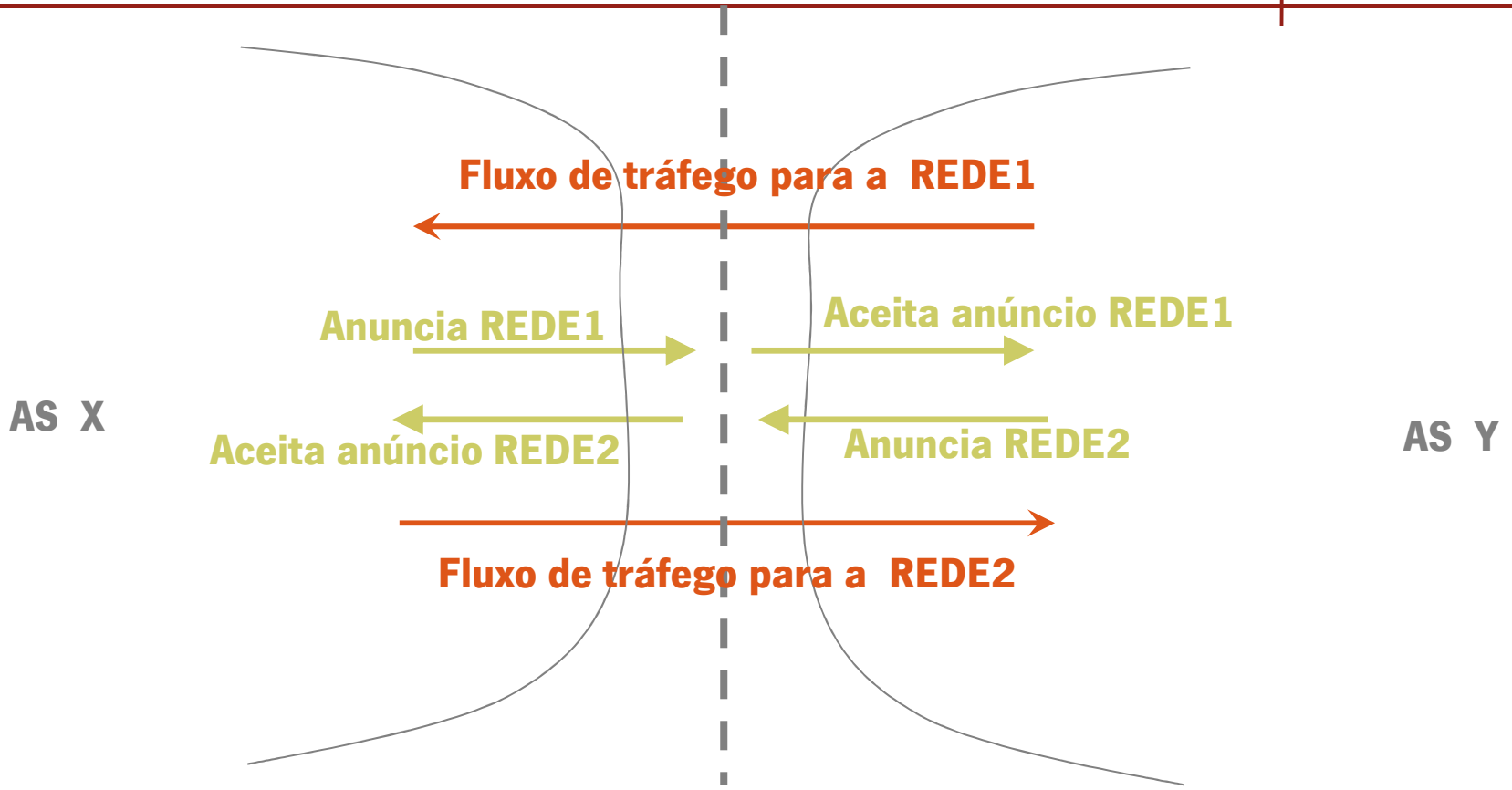
Fonte: Computer Networking: A Top-Down Approach Featuring the Internet, J. Kurose, Addison-Wesley, 2001

Encaminhamento na Internet



- **Número de Sistema Autónomo**
  - Os números de sistema autónomo podem ser privados (AS 64512 até ao AS 65535) ou públicos (atribuídos pelo IANA, ou autoridades regionais, como o RIPE na Europa)
  - Usado nas trocas de informação de encaminhamento com os sistemas autónomos vizinhos
- **Sistemas autónomos que fazem negócio com a conectividade também se designam por ISP (*Internet Service Providers*)**
  - Estabelecem acordos de parceria entre si (*peering agreements*)
    - São apenas de troca de rotas se estão ao mesmo nível
  - Se não estão ao mesmo nível o de nível mais baixo (*downstream*) é *cliente* e do nível acima (*upstream*) é *fornecedor*

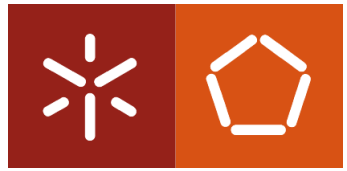
Encaminhamento na Internet



- **A comunicação entre a REDE1 e a REDE2 é possível se e só se:**
  (1) REDE1 anunciada por ASX, (2) anúncio aceite por ASY,
  (3) REDE2 anunciada por ASY, (4) anúncio aceite por ASX

Fonte: RFC1930

Encaminhamento na Internet



- **Protocolos IGP**
  - Usam processos automáticos de descoberta e troca de informação
  - Todos os encaminhadores são de confiança, sujeitos à mesma administração e às mesmas regras
  - As rotas e outra informação de encaminhamento pode ser difundida livremente entre todos os encaminhadores (todos têm a mesma visão da rede)

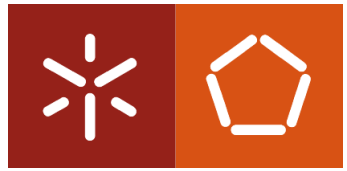
OSPF e ISIS → Estado da Ligação
RIP e EIGRP → Vector Distância

- **Protocolos EGP**
  - As relações com os pares são previamente definidas e configuradas manualmente
  - A conectividade com redes externas é definida por políticas (divulgação e aceitação de rotas, preferências, etc)
  - Define limites administrativos

BGP – Border Gateway Protocol

→Vector Distância que associa a cada prefixo o caminho AS completo

Encaminhamento Intra-Domínio *versus* Encaminhamento Inter-domínio



Políticas:

- No encaminhamento inter-domínio é fundamental ter o controle sobre a forma como o encaminhamento é efectuado. Por exemplo, a decisão de não encaminhar determinado tipo de tráfego através de um AS, tem de ser possível.
- No encaminhamento intra-domínio as decisões "políticas" de encaminhamento assumem pouca importância, uma vez que todos os nós estão sob a mesma autoridade administrativa.

Escala:

- O encaminhamento hierárquico reduz o tamanho das tabelas de encaminhamento e a quantidade e tamanho das mensagens de actualização da informação de encaminhamento.

Desempenho:

- No encaminhamento intra-domínio o desempenho é a preocupação principal, ao passo que no encaminhamento inter-domínio tem um papel secundário, sendo ultrapassado pelas políticas.