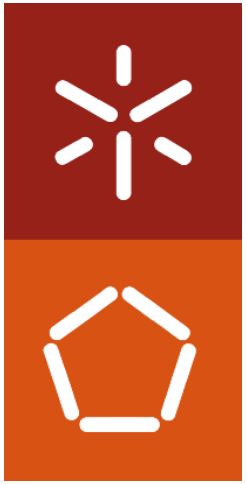
Conceitos, Algoritmos e Protocolos de Encaminhamento

Comunicações por Computador

Licenciatura em Engenharia Informática

Universidade do Minho



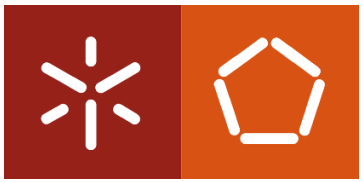
Adaptado dos capítulos 4 e 5 do livro

Computer Networking: A Top Down Approach

J.Kurose & K.Ross, Addison-Wesley, 2021.

Encaminhamento IP

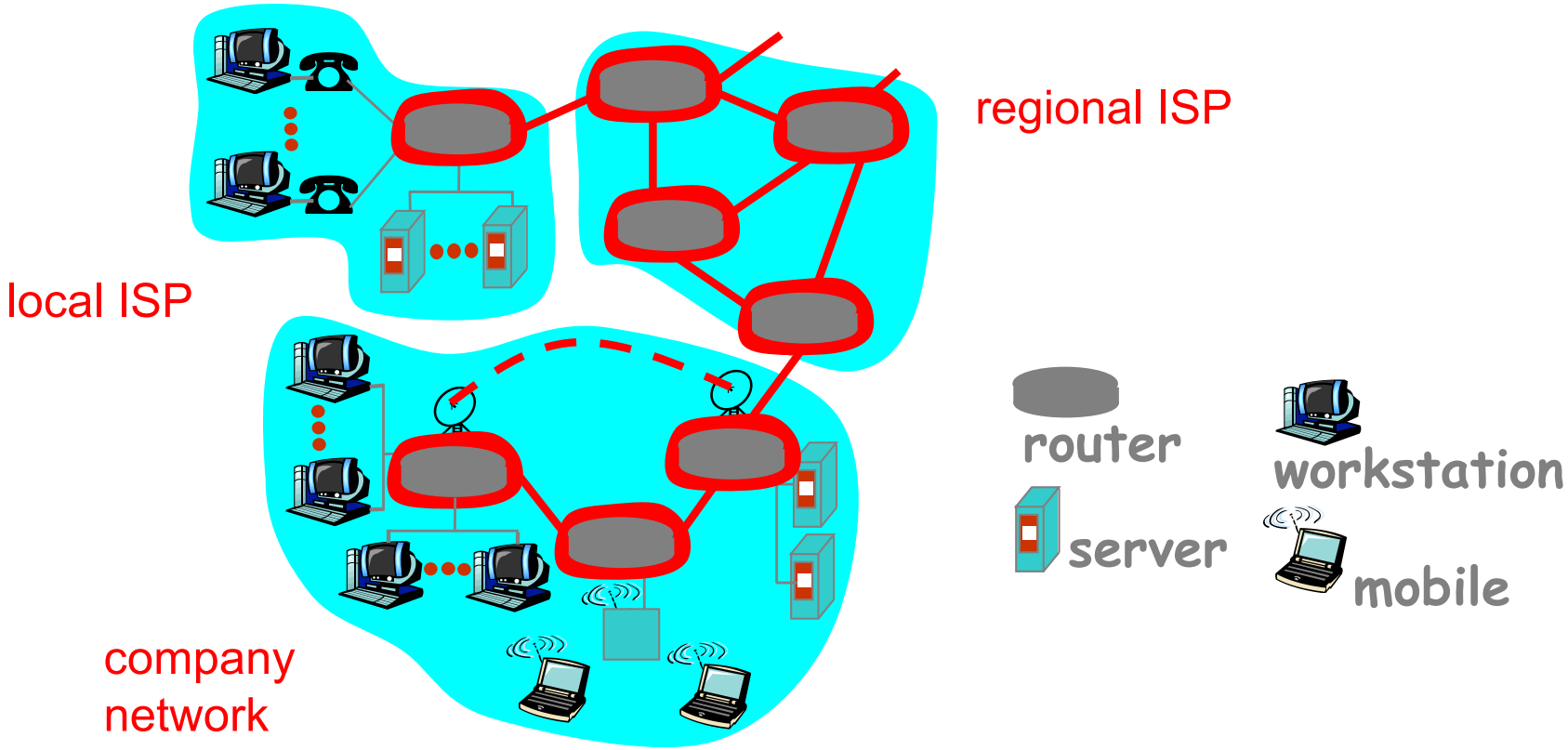
Sumário



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

Encaminhamento IP

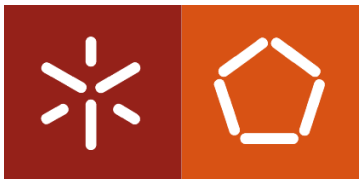
Introdução



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

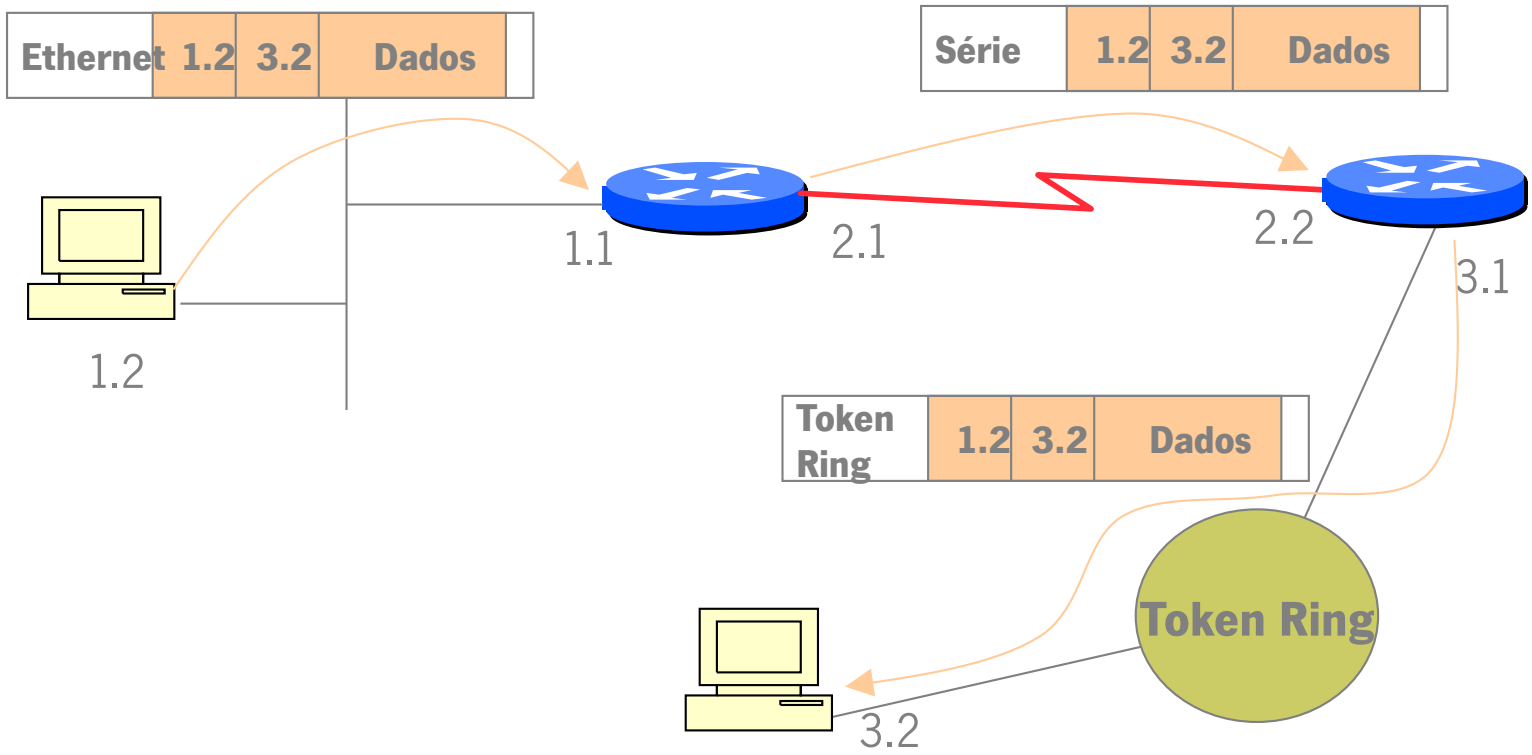
Encaminhamento IP

Introdução



Routers **armazenam e reenviam datagramas IP**

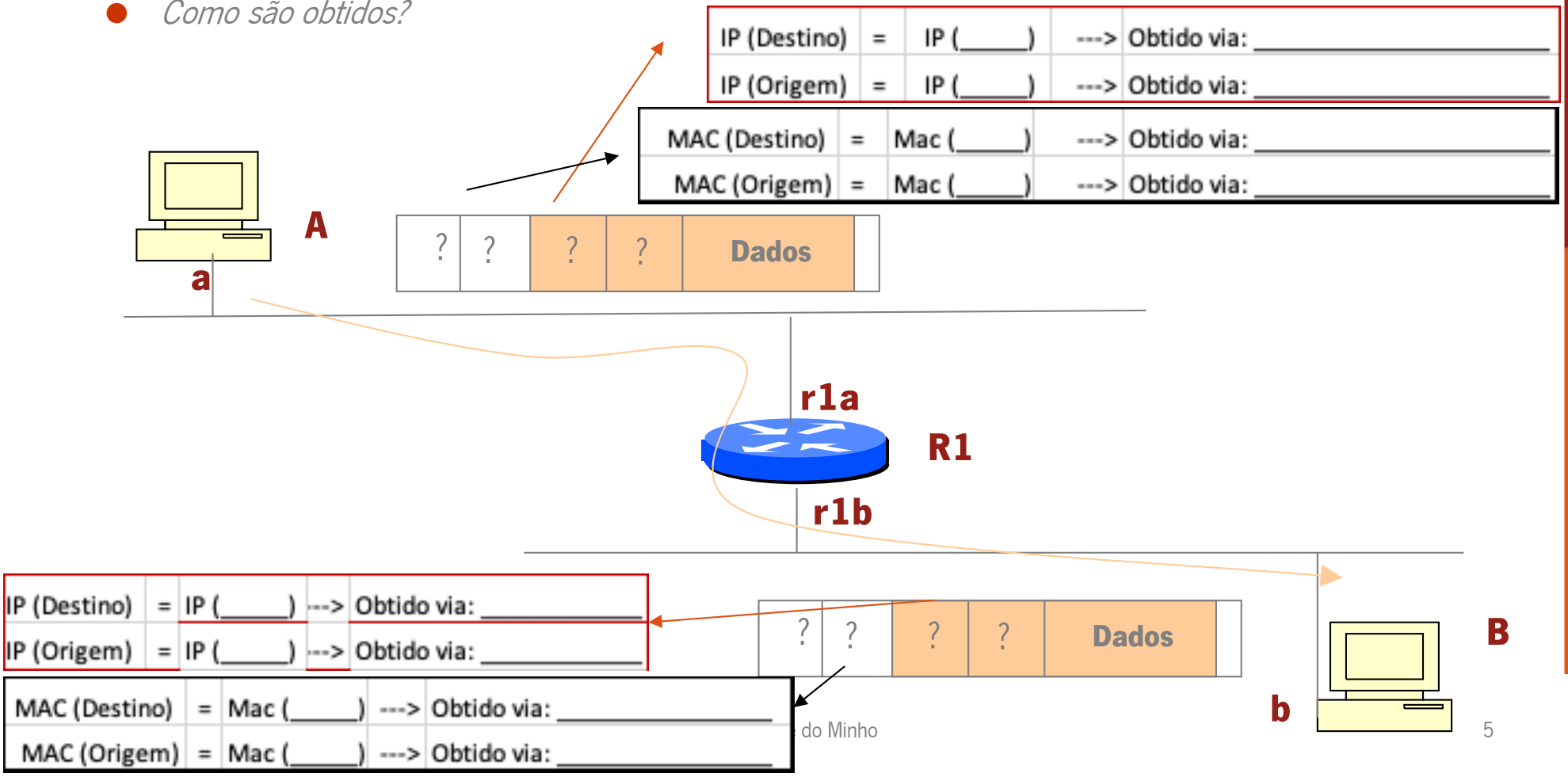
"Desencapsula" no interface de entrada, encapsula no interface de saída, de acordo com o tipo de interface...



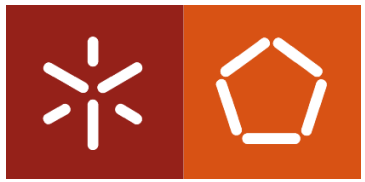
Encaminhamento IP

Introdução

- **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?*



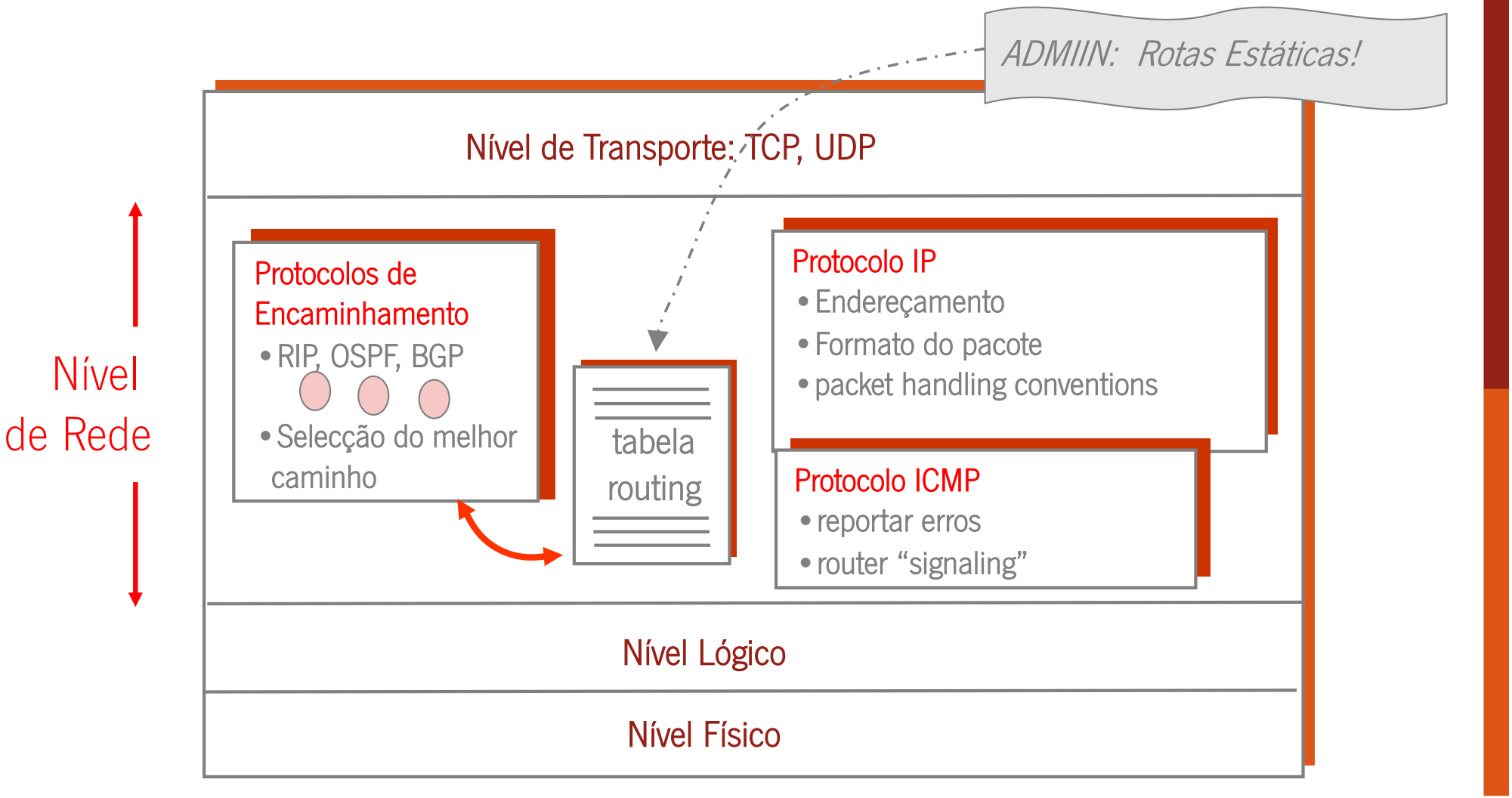
Encaminhamento IP
Introdução



- **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 respetivo, 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, no caso mais comum, um processo automático resultante da operação de um protocolo de encaminhamento dinâmico.

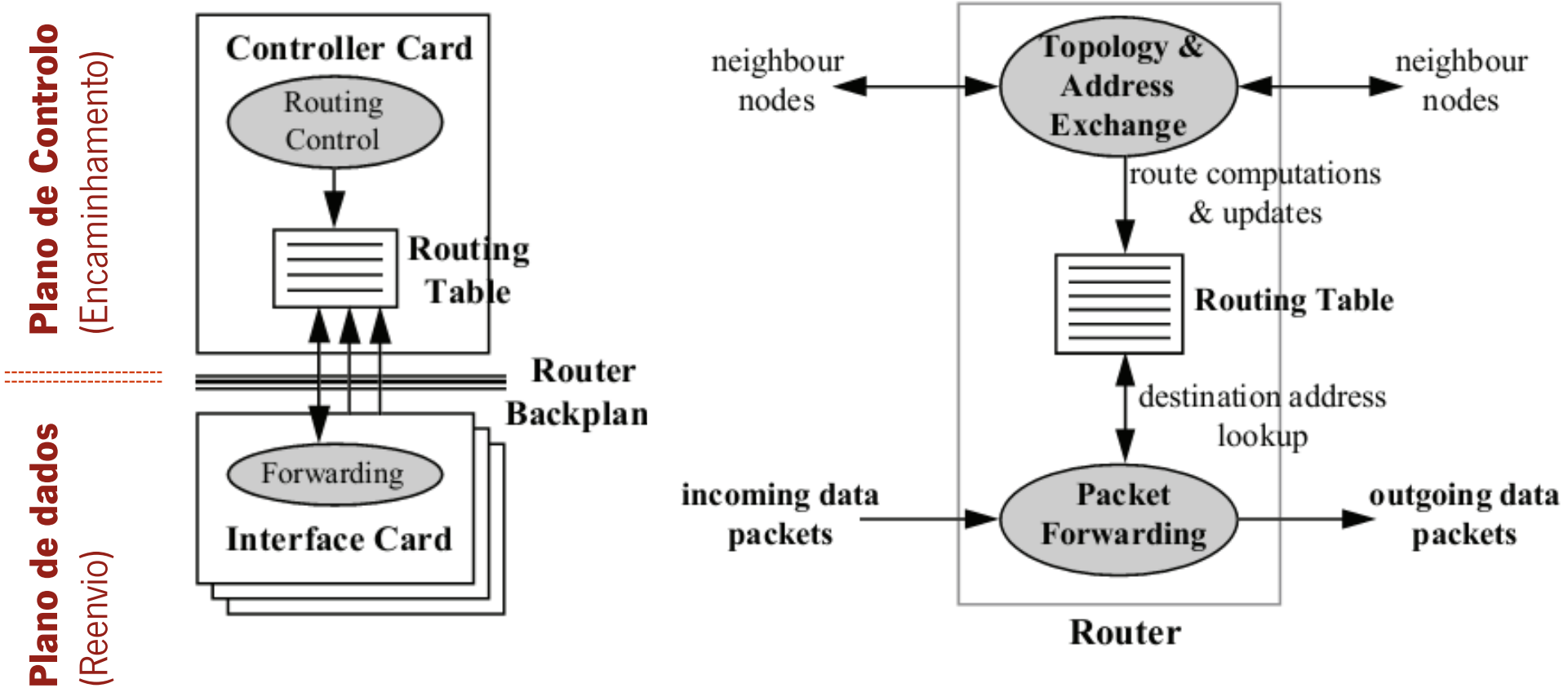
Encaminhamento IP

Introdução



Encaminhamento IP

Routers – Arquitetura

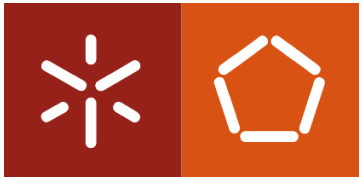


a) Arquitetura básica *b) Componentes de routing*

Fonte: James Aweya, IP router architectures: an overview

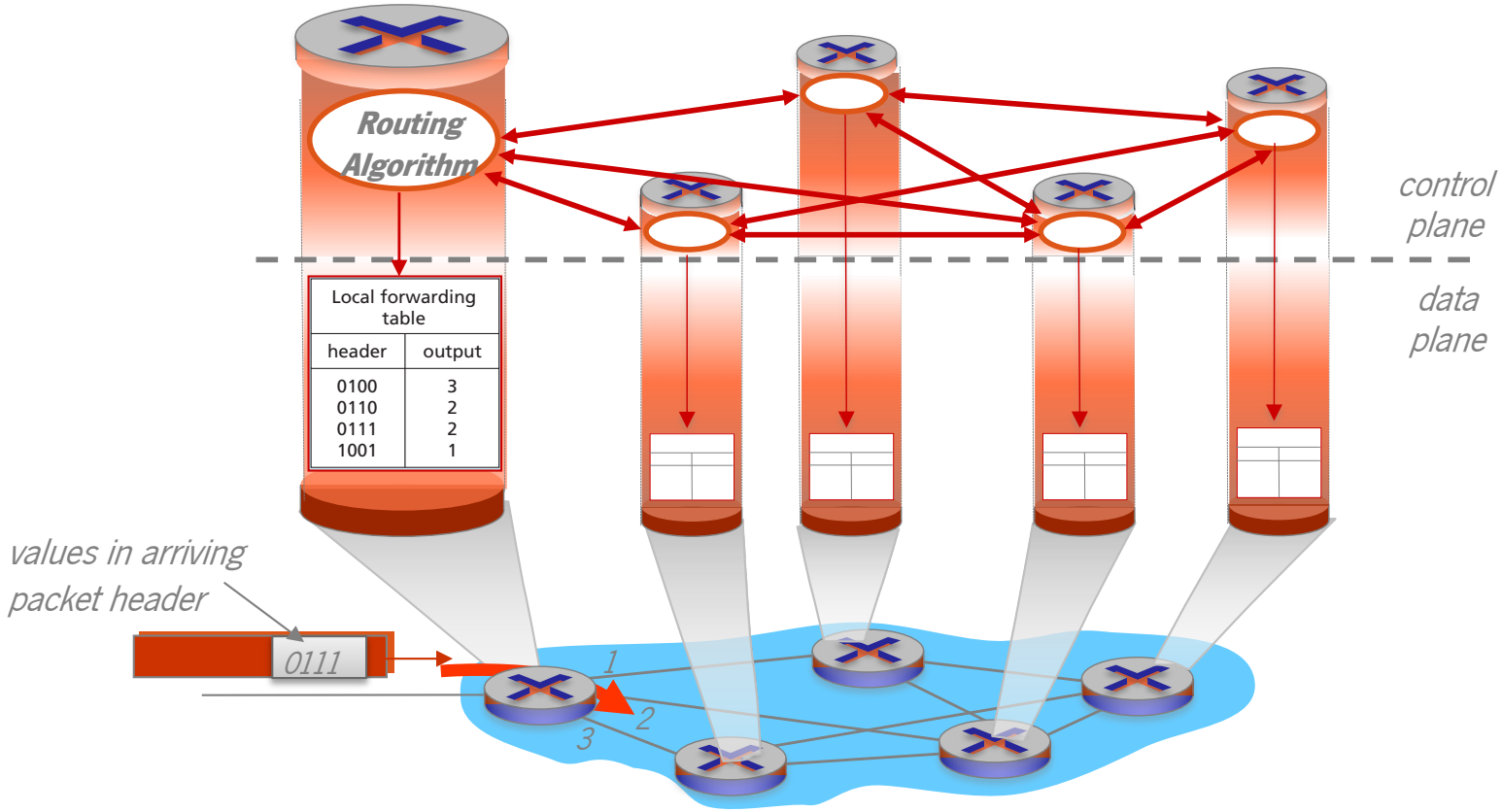
Encaminhamento IP

Routers – Plano de Controlo Distribuído



Existe uma componente de controlo em todos os *routers* !

(algoritmos de *routing* são distribuídos)

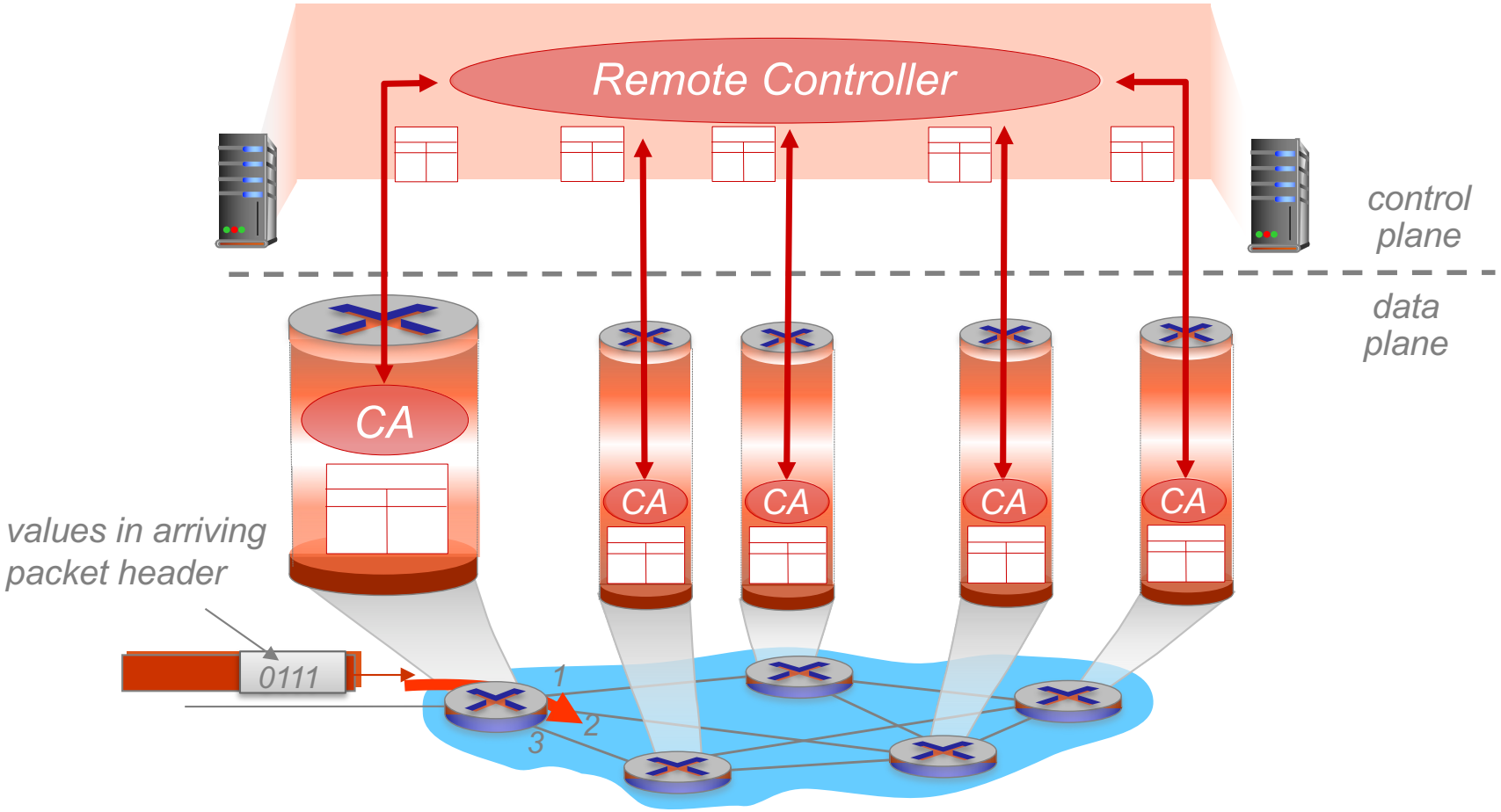


Encaminhamento IP

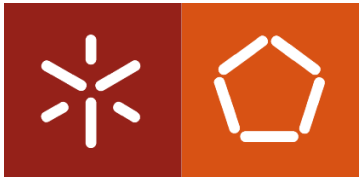
Routers – Plano de Controlo Centralizado

Um controlador remoto que interage com agentes locais (CA)!

(algoritmos de *routing* centralizados)



Encaminhamento IP
Conceitos – Algoritmo de *Routing*...



Algoritmo de Encaminhamento/*Routing*

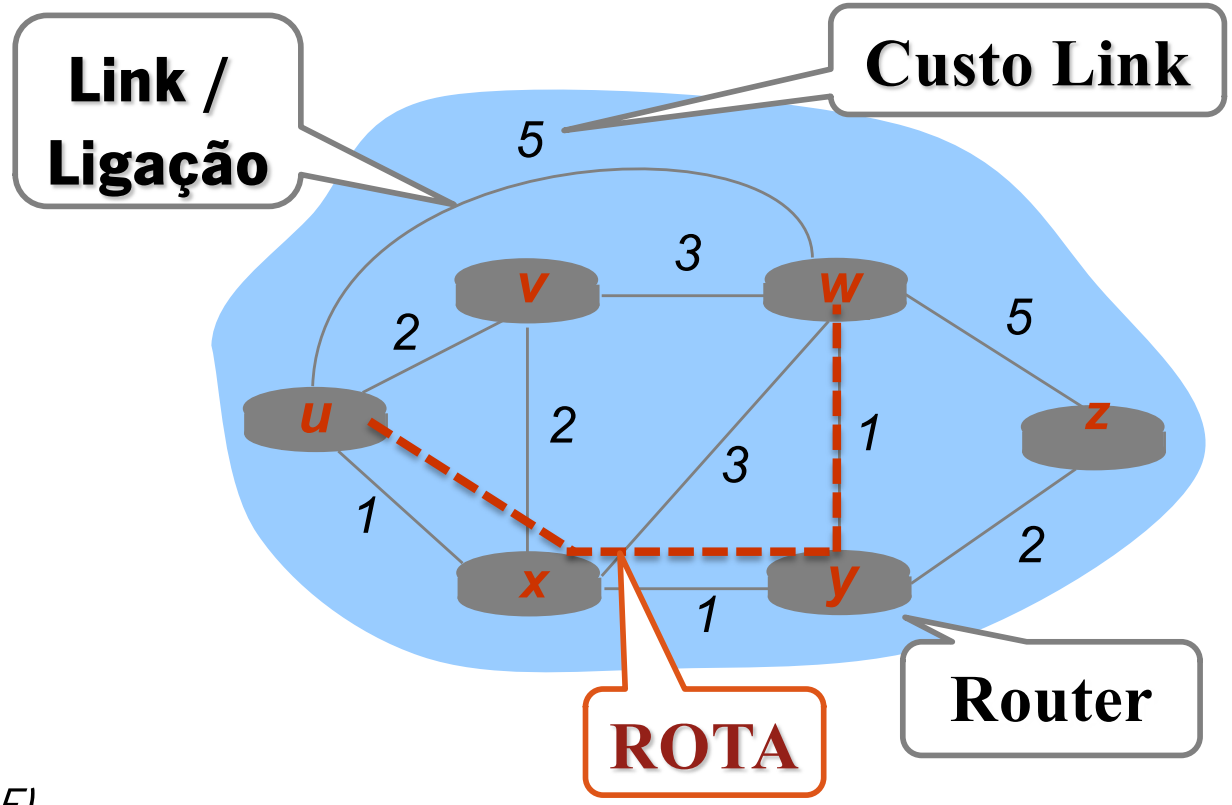
Dada uma topologia de rede (um conjunto de encaminhadores com ligações de rede a interligá-los), representável como um grafo com pesos nos arcos/ligações dos seus nós, o objetivo do algoritmo de encaminhamento é determinar um "bom" caminho desde o nó fonte/origem até ao nó destino.

A topologia de rede é um grafo em que:

- Os nós do grafo são os encaminhadores/*routers* ;
- Os arcos do grafo são as ligações/*links* da rede;
- O custo das ligações pode ser estabelecido em função de vários critérios (individualmente ou em conjunto), como por exemplo, o atraso, a capacidade/ritmo nominal, do nível de congestão, do custo operacional, da distância, etc.;
- Um "bom" caminho geralmente significa o caminho que minimiza ou maximiza o seu custo total...

Encaminhamento IP

Conceitos – A rede como um grafo...



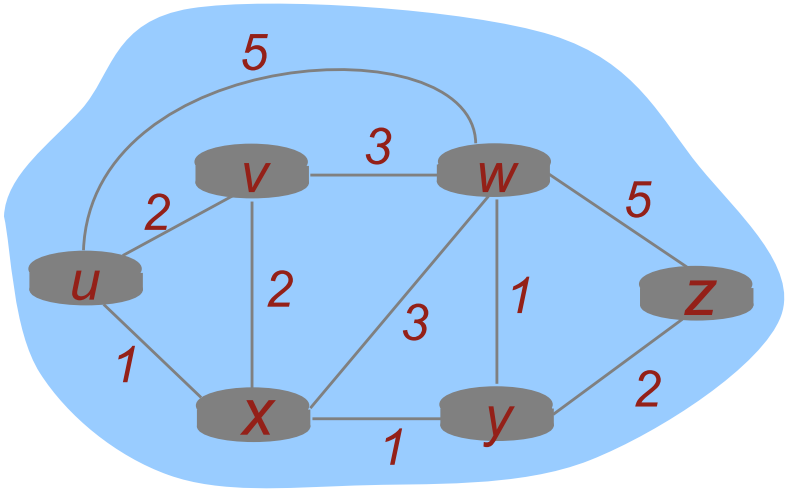
graph: G = (N,E)

N = set of routers = { u, v, w, x, y, z }

E = set of links ={ (u,v), (u,x), (v,x), (v,w), (x,w), (x,y), (w,y), (w,z), (y,z) }

Encaminhamento IP

Conceitos – A rede como um grafo...



c(S,D) = custo do link (S,D)

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

Os algoritmos de encaminhamento geralmente procuram caminhos de custo mínimo!

Encaminhamento IP
Tabelas de Encaminhamento

Nó *u*

Destino	Próximo Nó	Link	Custo
u	*u*	—	*0*
...	...	...	...
x	*x*	L_{ux}	*1*
...	...	...	...
w	*x*	L_{ux}	*3*

Encaminhamento IP
Conceitos – Tipos de Algoritmos

Os algoritmos de encaminhamento podem gerir a informação de duas formas distintas:

Global:

- Todos os encaminhadores têm um conhecimento completo da topologia e custo das ligações;
- Algoritmos de estado das ligações (***Link State*** **– LS**).

Descentralizada:

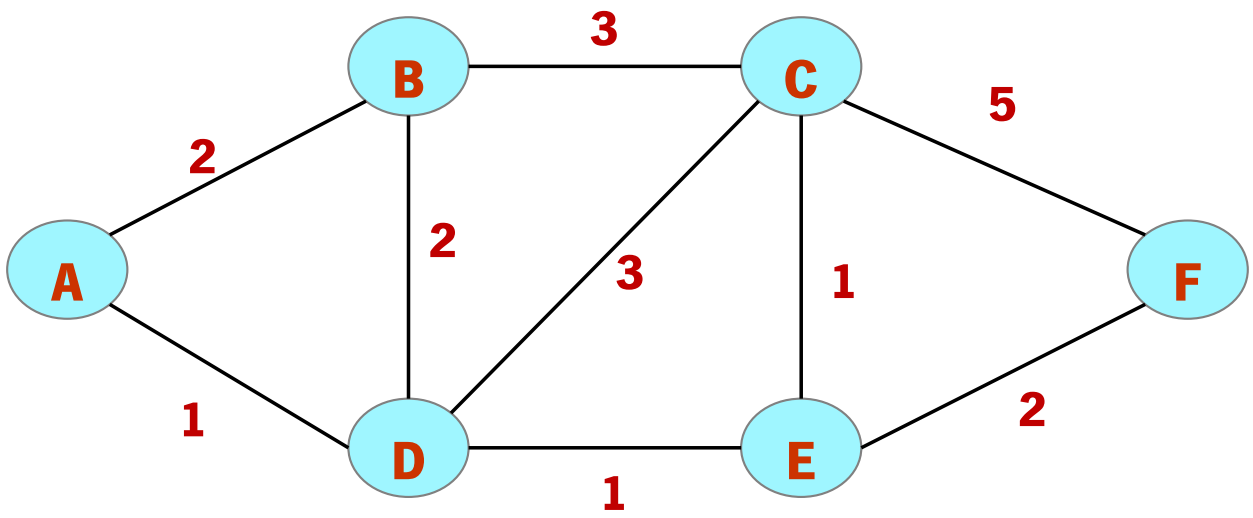
- Os encaminhadores só conhecem os vizinhos a que estão fisicamente/logicamente ligados e o custo das ligações respetivas;
- O processo de computação é iterativo, havendo troca de informação entre vizinhos;
- Algoritmos de vetor de distância (***Distance Vector*** **– DV**).

Encaminhamento IP
Algoritmos *Link State* (LS)

- Todos os nós da topologia espalham pela rede informação sobre o estado das suas ligações de forma a construírem a base de dados topológica (usando um método de inundação fiável – ***Reliable Flooding***):
  - Inicialmente necessitam de conhecer apenas os seus vizinhos diretos, para quem enviam a identificação de todos os outros seus vizinhos, bem como o custo das ligações respetivas;
  - Um nó/encaminhador ao receber esta informação atualiza a sua base de dados topológica e reenvia 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.
- Com esta informação, em cada nó/encaminhador, é executado um algoritmo de descoberta dos caminhos de custo mínimo, normalmente o algoritmo de **Dijkstra**.
- Com o resultado obtido da aplicação do algoritmo anterior é preenchida a tabela de encaminhamento do nó.

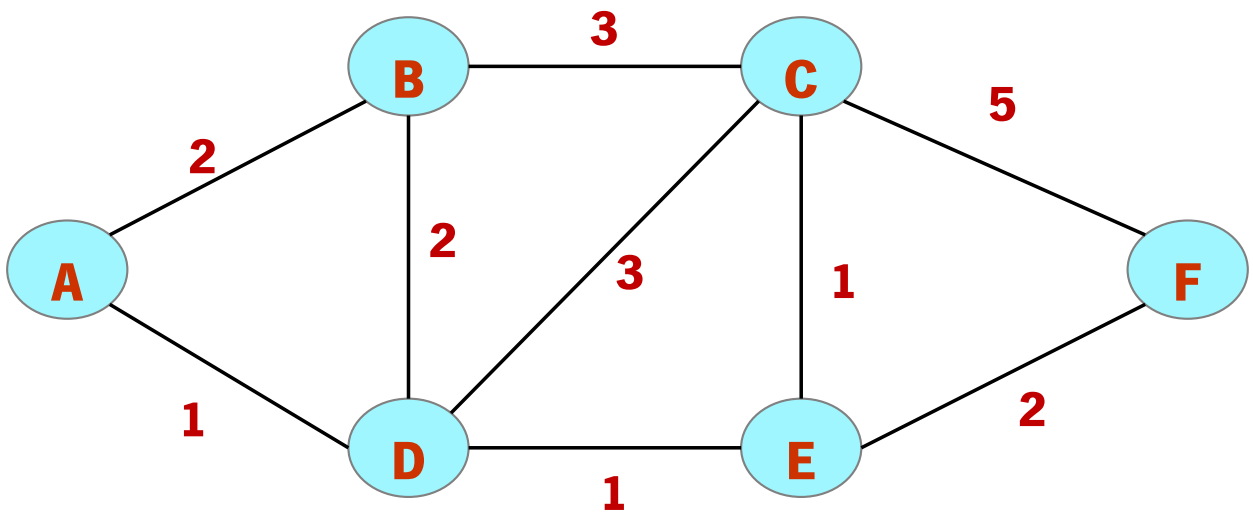
Encaminhamento IP
Algoritmos LS – Exercício

- 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 informação (*Link State Announcement* – LSA)?
- Seria correto calcular as rotas nesta fase? Quando seria?



Exercício

1. Cada nó gera uma mensagem LSA com o estado dos seus links

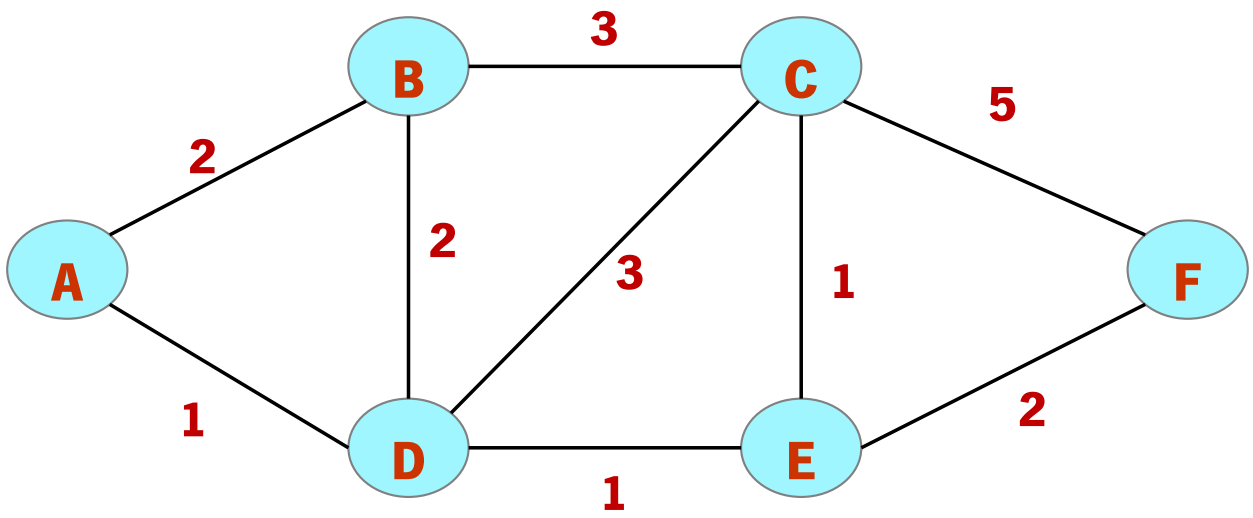


Exercício

1. Cada nó gera uma mensagem LSA com o estado dos seus links

   LSA(A) = { (A→B, 2); (A→D, 1) }, ... , LSA(F) = { (F→C, 5); (F→E, 2) }

   Grafo da rede = { todos os LSAs de todos os nós }

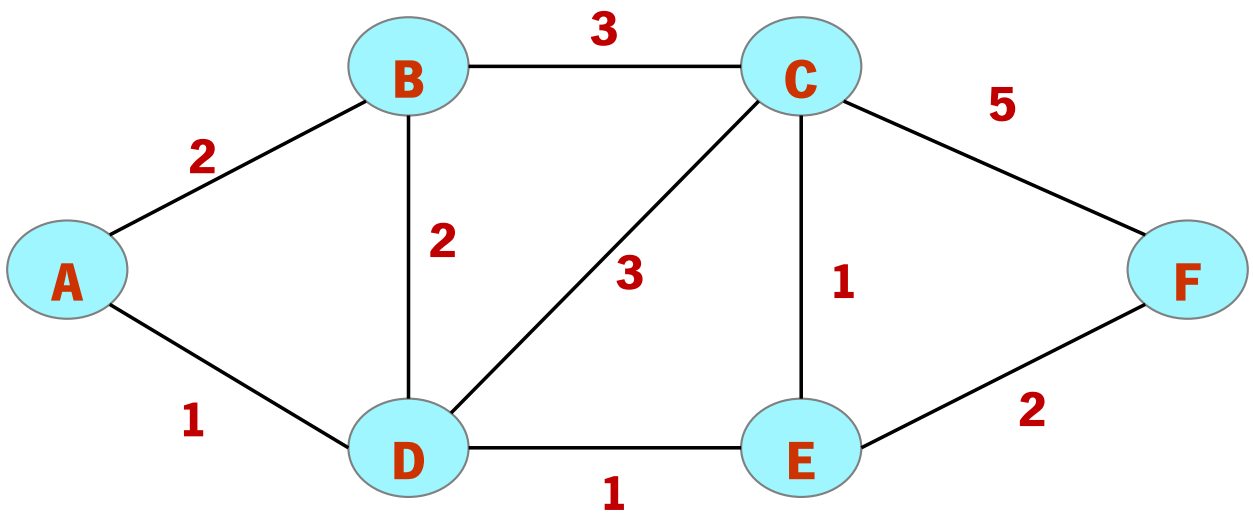


Exercício

1. Cada nó gera uma mensagem LSA com o estado dos seus links

   LSA(A) = { (A→B, 2); (A→D, 1) }, ... , LSA(F) = { (F→C, 5); (F→E, 2) }

2. Cada nó tem de enviar a sua msg LSA a todos os outros (Flooding)



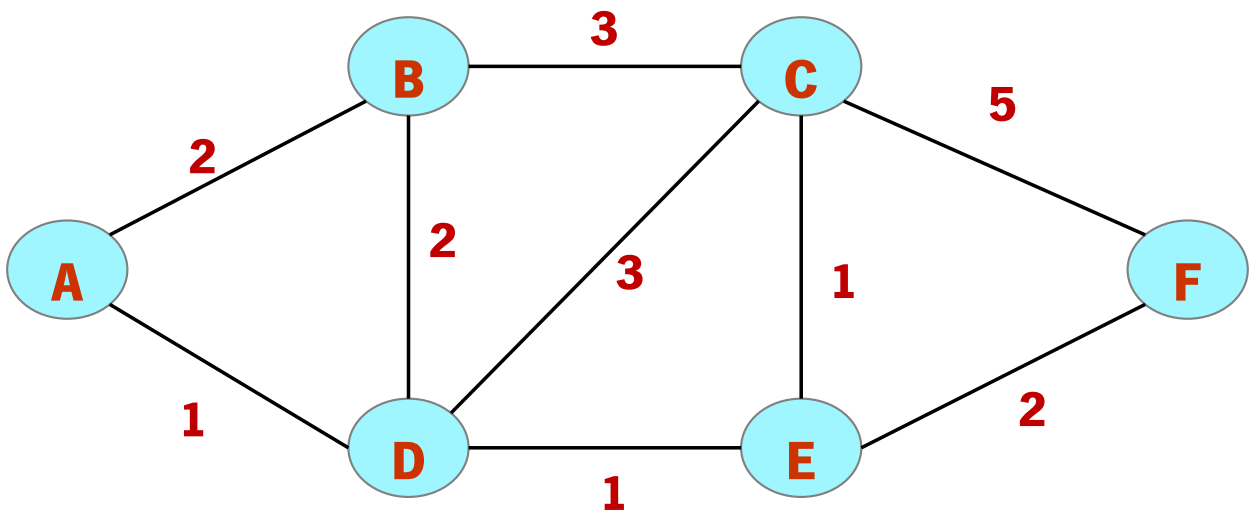
Exercício

1. Cada nó gera uma mensagem LSA com o estado dos seus links

   LSA(A) = { (A→B, 2); (A→D, 1) }, ... , LSA(F) = { (F→C, 5); (F→E, 2) }

2. Cada nó tem de enviar a sua msg LSA a todos os outros (Flooding)

   Exemplo só para as mensagens enviadas pelo A



Exercício

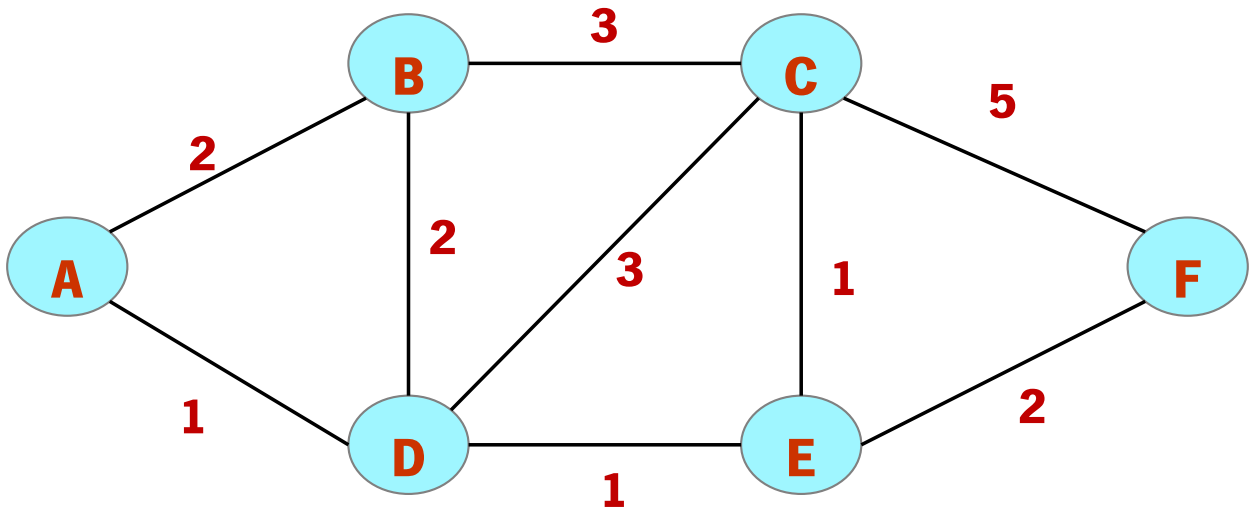
1. Cada nó gera uma mensagem LSA com o estado dos seus links

   LSA(A) = { (A→B, 2); (A→D, 1) }, ... , LSA(F) = { (F→C, 5); (F→E, 2) }

2. Cada nó tem de enviar a sua msg LSA a todos os outros (Flooding)

   Exemplo só para as mensagens enviadas pelo A

< LSA >
(iteração)



Exercício

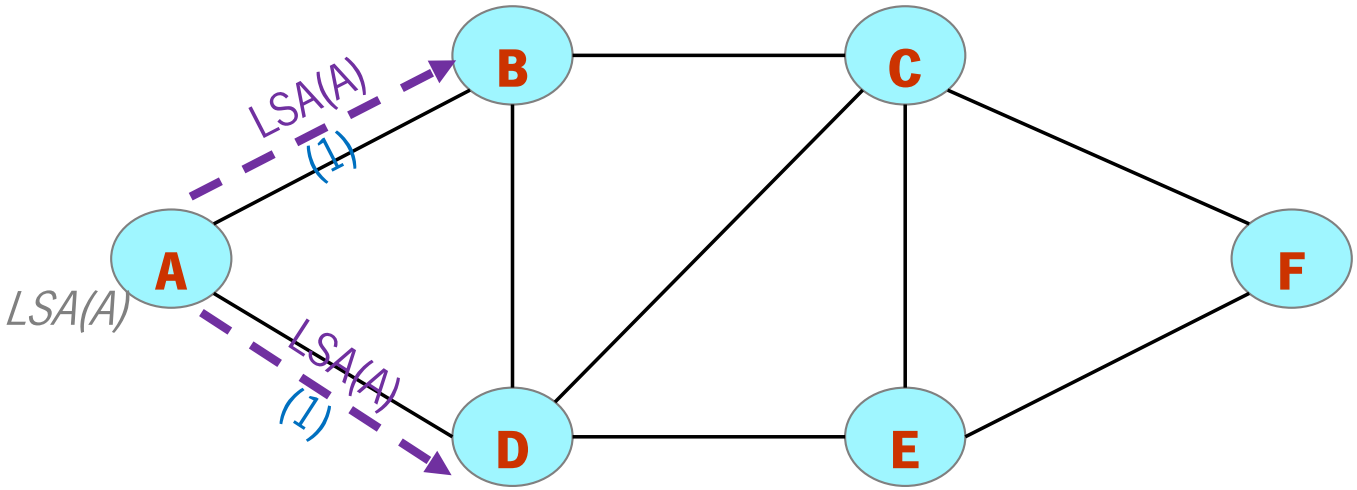
1. Cada nó gera uma mensagem LSA com o estado dos seus links

   LSA(A) = { (A→B, 2); (A→D, 1) }, ... , LSA(F) = { (F→C, 5); (F→E, 2) }

2. Cada nó tem de enviar a sua msg LSA a todos os outros (Flooding)

   Exemplo só para as mensagens enviadas pelo A

< LSA >
(iteração)



-→ O nó A vai enviar a todos os vizinhos...

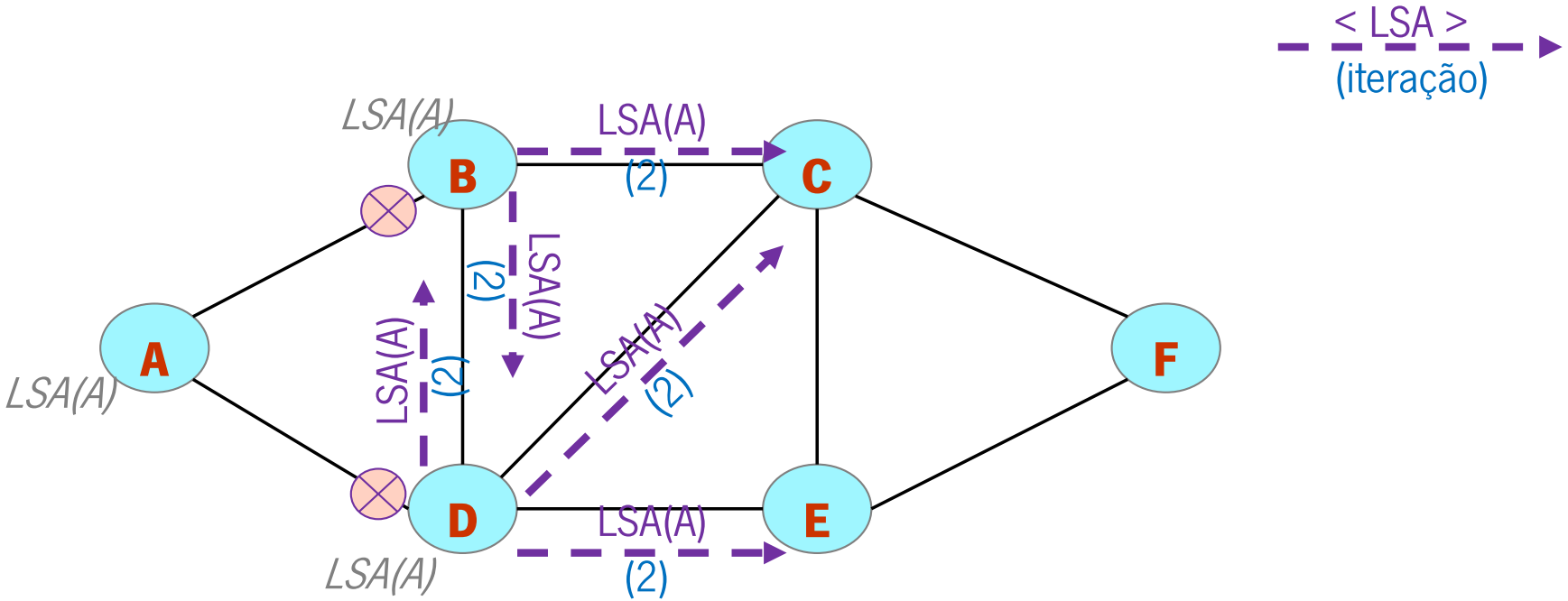
Exercício

1. Cada nó gera uma mensagem LSA com o estado dos seus links

   LSA(A) = { (A→B, 2); (A→D, 1) }, ... , LSA(F) = { (F→C, 5); (F→E, 2) }

2. Cada nó tem de enviar a sua msg LSA a todos os outros (Flooding)

   Exemplo só para as mensagens enviadas pelo A



Os vizinhos reenviam também a todos, **exceto** de onde receberam! (RPF)

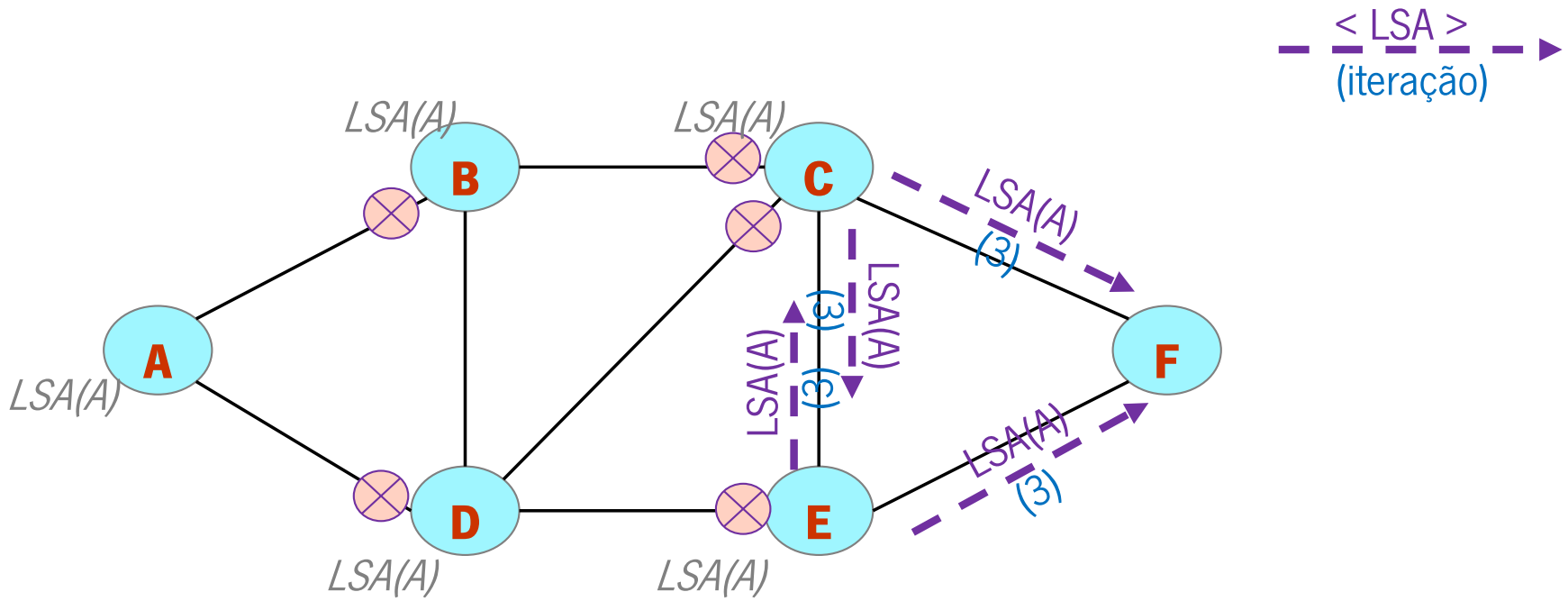
Exercício

1. Cada nó gera uma mensagem LSA com o estado dos seus links

   LSA(A) = { (A→B, 2); (A→D, 1) }, ... , LSA(F) = { (F→C, 5); (F→E, 2) }

2. Cada nó tem de enviar a sua msg LSA a todos os outros (Flooding)

   Exemplo só para as mensagens enviadas pelo A



Os vizinhos reenviam também a todos, **exceto** de onde receberam! (RPF)

Exercício

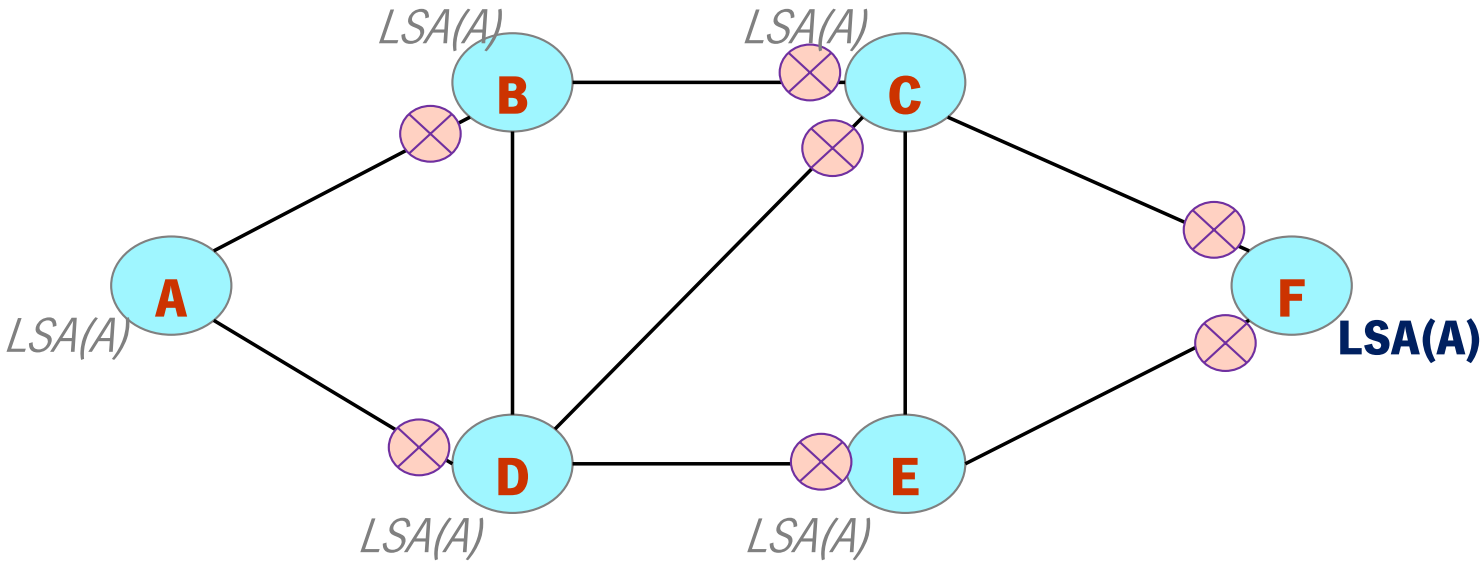
1. Cada nó gera uma mensagem LSA com o estado dos seus links

   LSA(A) = { (A→B, 2); (A→D, 1) }, ... , LSA(F) = { (F→C, 5); (F→E, 2) }

2. Cada nó tem de enviar a sua msg LSA a todos os outros (Flooding)

   Exemplo só para as mensagens enviadas pelo A

<ACK>
(iteração)



3. A receção tem de ser confirmada de volta até ao A (certeza de que chegou)

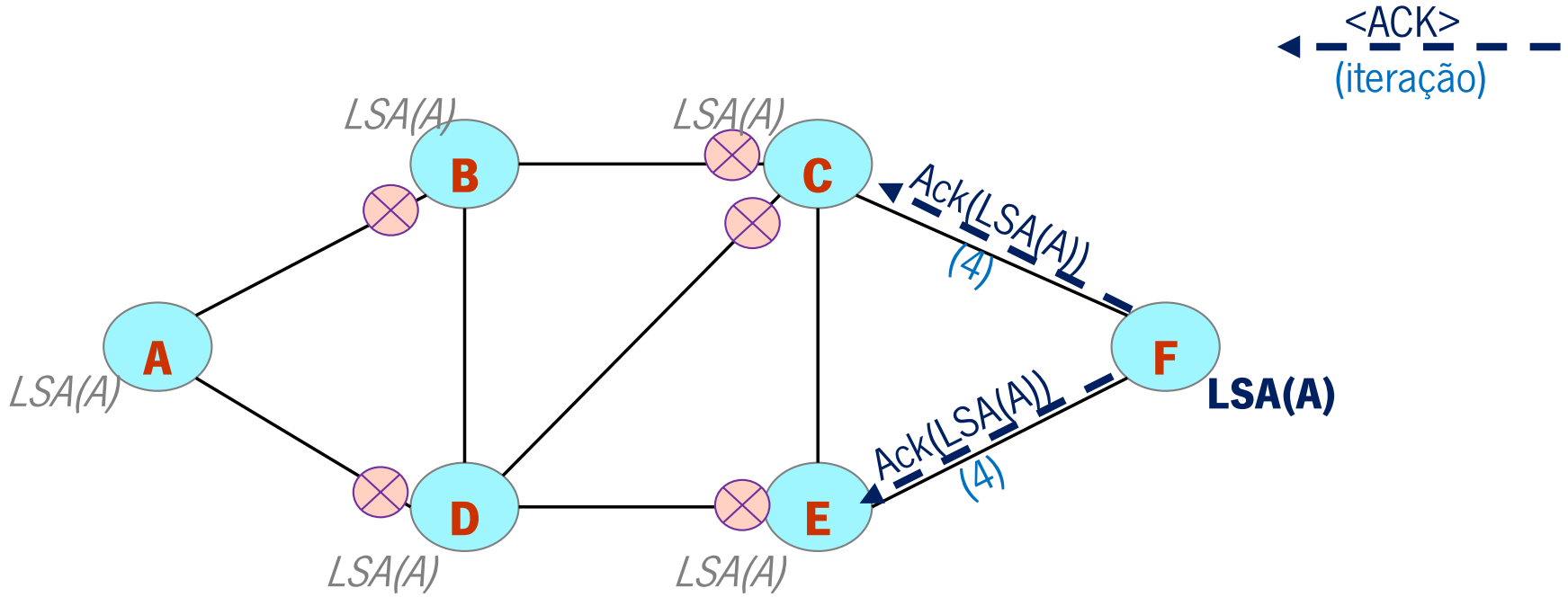
Exercício

1. Cada nó gera uma mensagem LSA com o estado dos seus links

   LSA(A) = { (A→B, 2); (A→D, 1) }, ... , LSA(F) = { (F→C, 5); (F→E, 2) }

2. Cada nó tem de enviar a sua msg LSA a todos os outros (Flooding)

   Exemplo só para as mensagens enviadas pelo A



3. A receção tem de ser confirmada de volta até ao A (certeza de que chegou)

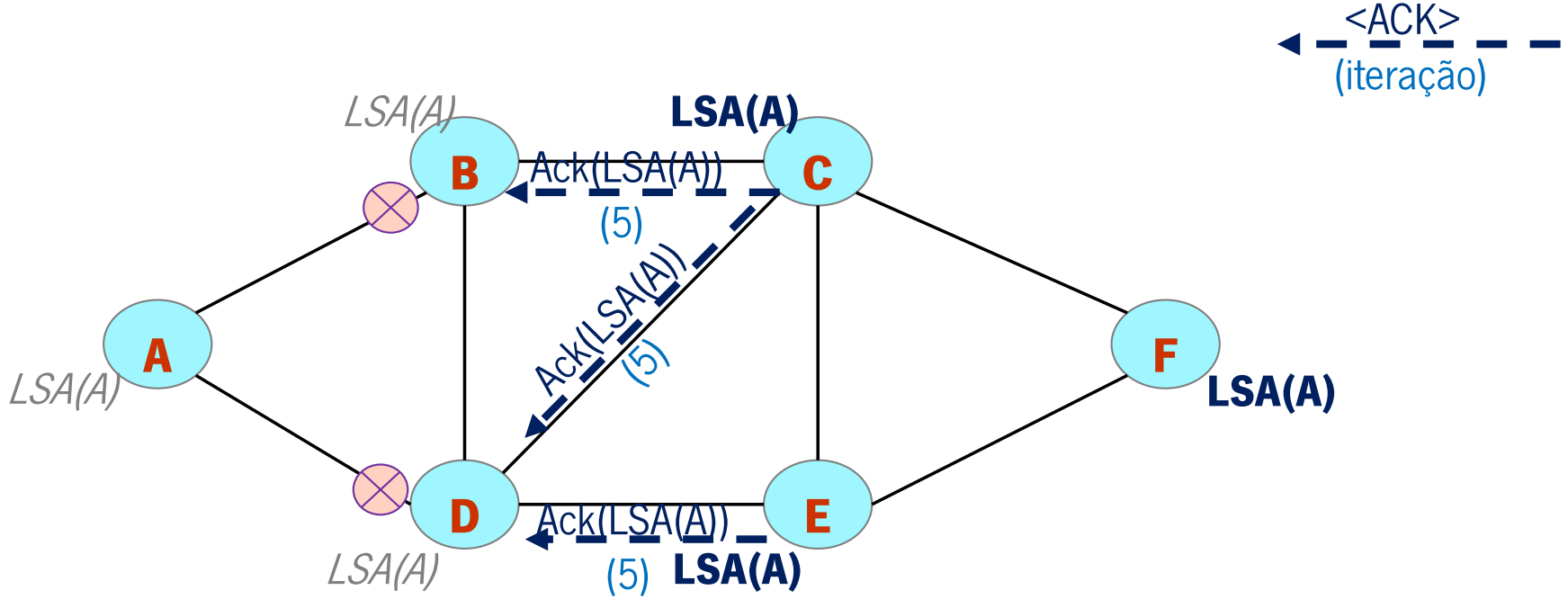
Exercício

1. Cada nó gera uma mensagem LSA com o estado dos seus links

   LSA(A) = { (A→B, 2); (A→D, 1) }, ... , LSA(F) = { (F→C, 5); (F→E, 2) }

2. Cada nó tem de enviar a sua msg LSA a todos os outros (Flooding)

   Exemplo só para as mensagens enviadas pelo A



3. A receção tem de ser confirmada de volta até ao A (certeza de que chegou)

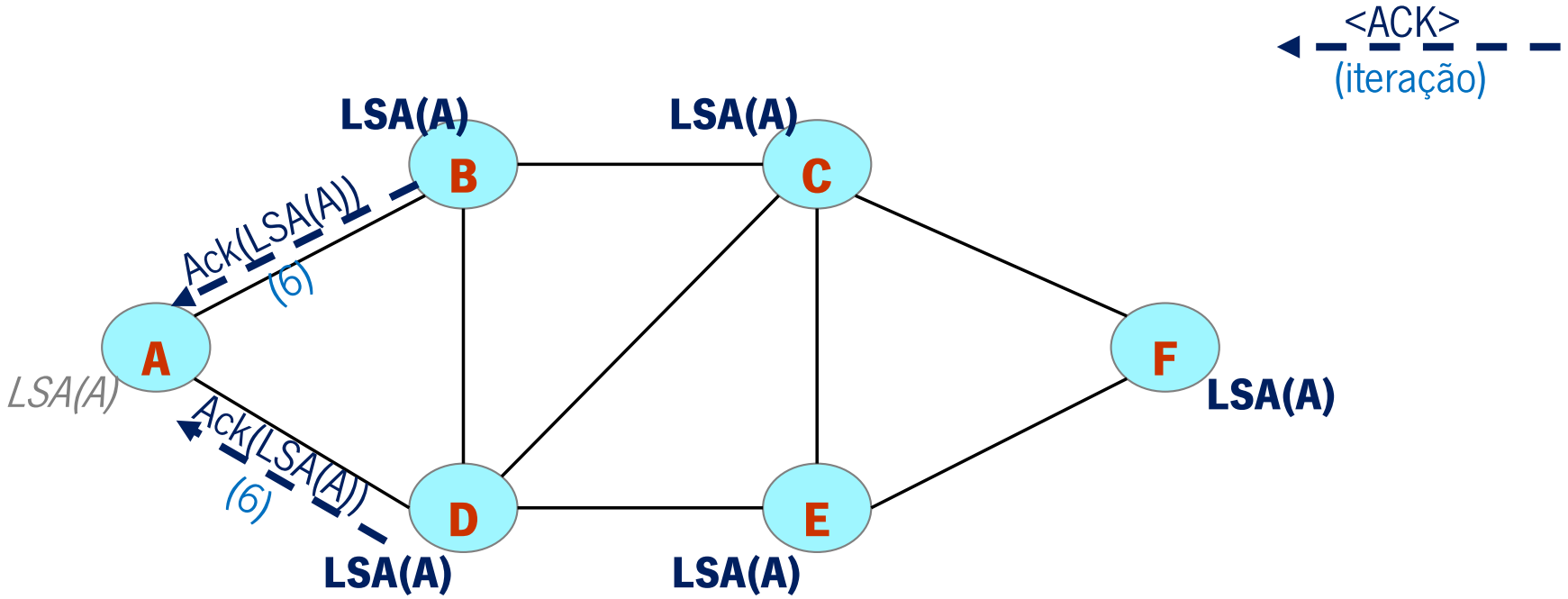
Exercício

1. Cada nó gera uma mensagem LSA com o estado dos seus links

   LSA(A) = { (A→B, 2); (A→D, 1) }, ... , LSA(F) = { (F→C, 5); (F→E, 2) }

2. Cada nó tem de enviar a sua msg LSA a todos os outros (Flooding)

   Exemplo só para as mensagens enviadas pelo A



3. A receção tem de ser confirmada de volta até ao A (certeza de que chegou)

Exercício

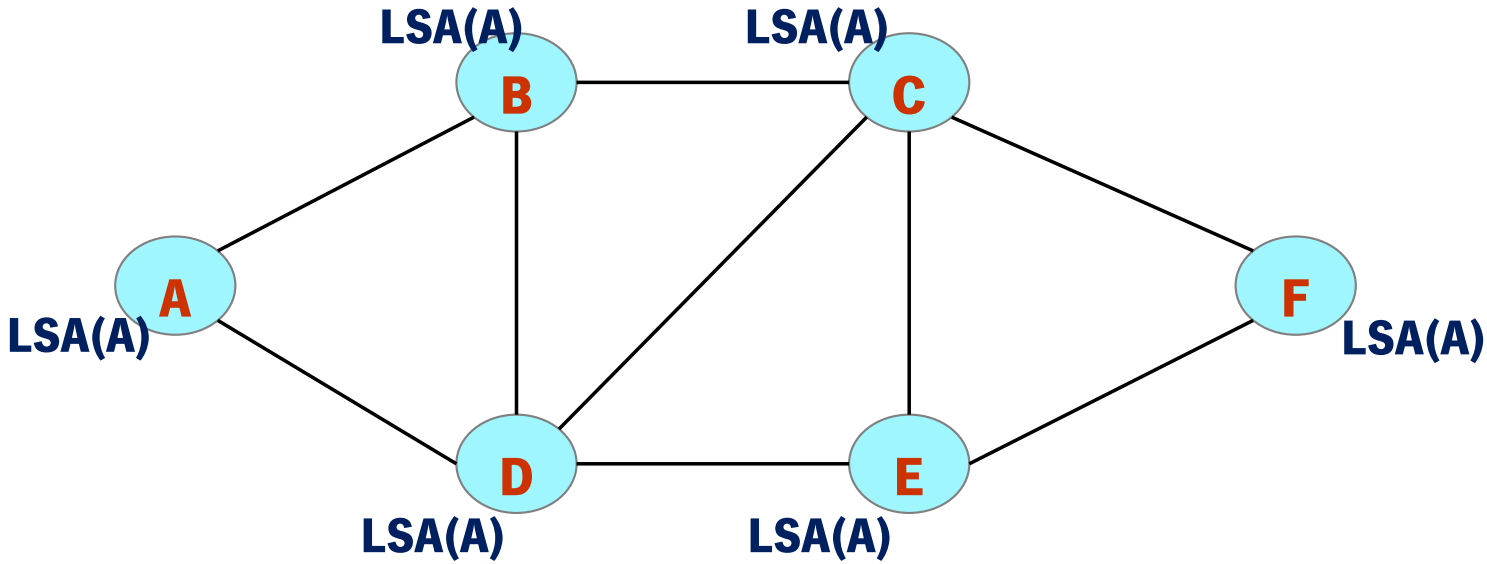
1. Cada nó gera uma mensagem LSA com o estado dos seus links

   LSA(A) = { (A→B, 2); (A→D, 1) }, ... , LSA(F) = { (F→C, 5); (F→E, 2) }

2. Cada nó tem de enviar a sua msg LSA a todos os outros (Flooding)

   Exemplo só para as mensagens enviadas pelo A

<ACK>
(iteração)

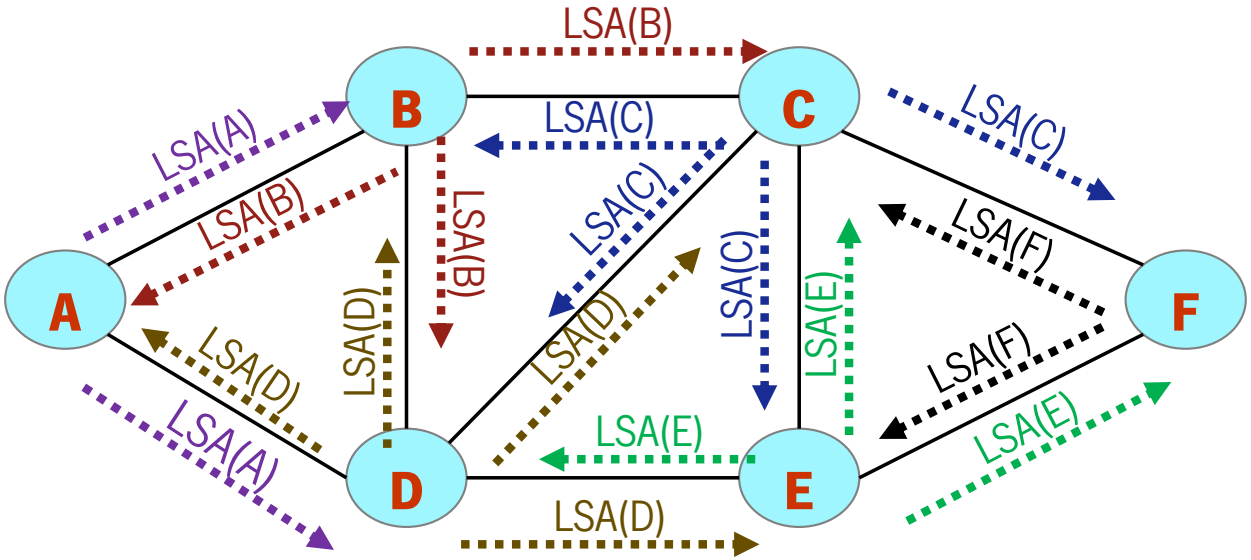


3. A receção tem de ser confirmada de volta até ao A (certeza de que chegou)

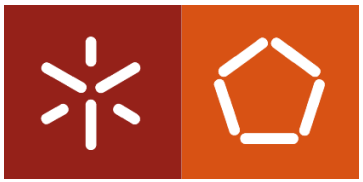
Exercício

Conseguem imaginar todos a enviar ao mesmo tempo???

Sincronizado? (pouco provável!) Só na primeira iteração:



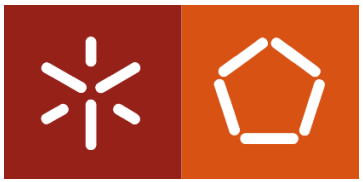
Encaminhamento IP
Algoritmos LS – Dijkstra



Algoritmo iterativo que ao fim de K iterações consegue descobrir os caminhos de custo mínimo de um determinado nó para K destinos:

- Seja $c(i,j) \neq \infty$ o custo da ligação do nó i para o nó adjacente j;
- Se o nó i e o nó j não estão diretamente ligados, então $c(i,j)=\infty$;
- 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 de origem até ao nó v;
- Seja N' o conjunto de todos os nós para os quais já se conhece o caminho de custo mínimo...
- Então o algoritmo Dijkstra para cálculo num nó origem dos valores mínimos D(v) para todos os nós da rede é dado por...

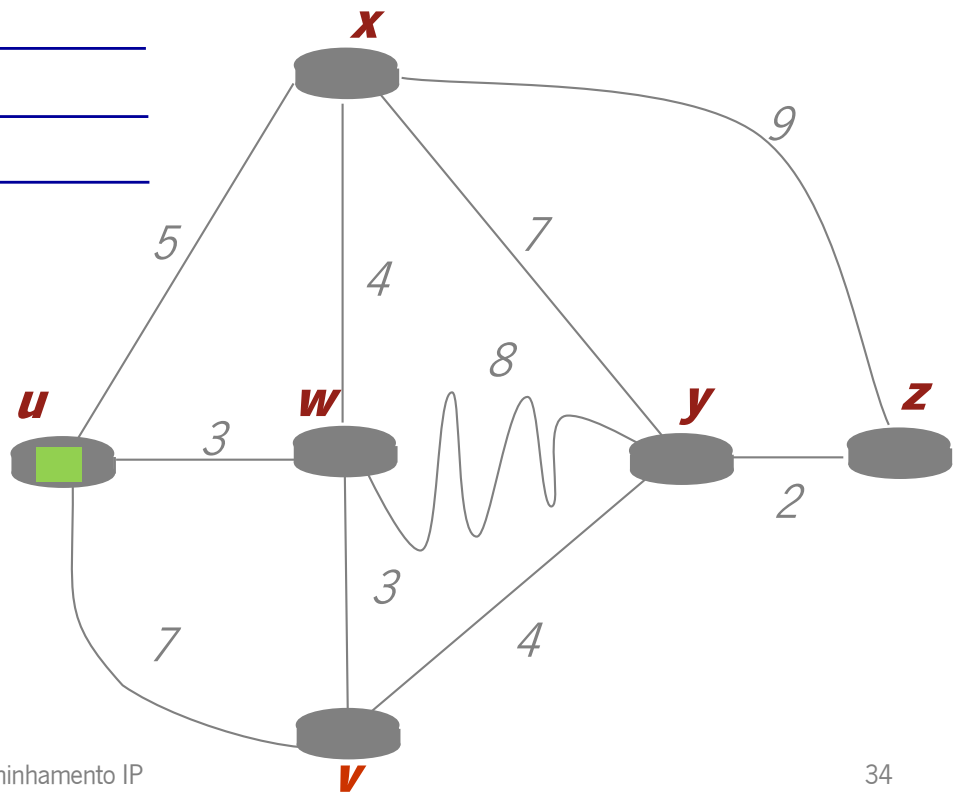
Encaminhamento IP
Algoritmos LS – Dijkstra



```
0   ** Algorithm for node A
1   Initialization:
2     N' = {A}
3     for all nodes B
4       if B adjacent to A
5           then D(B) = c(A,B)
6       else D(B) = ∞
7
8    Loop
9       find C not in N' such that D(C) is a minimum
10      add C to N'
11      for all B adjacent to C and not in N'
12         D(B) = min( D(B), D(C) + c(C,B) )
13      ** new cost to B is either old cost to B or known
14      ** shortest path cost to C plus cost from C to B
15  until all nodes in N'
```

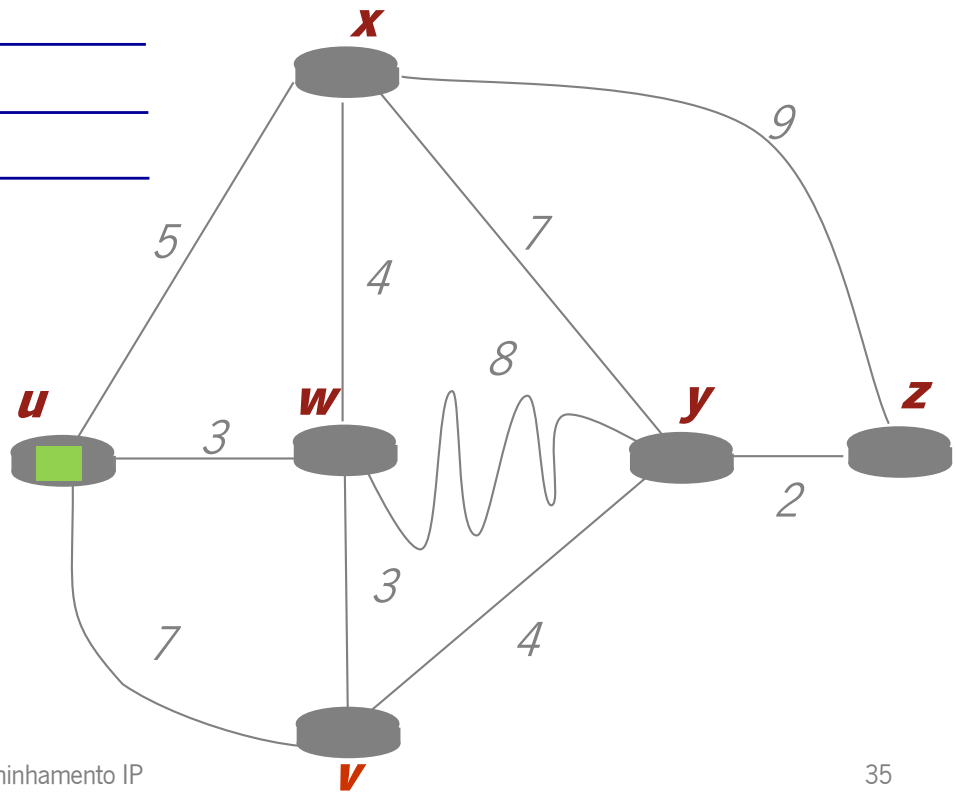
Encaminhamento IP
Algoritmo Dijkstra – Exemplo

Step	N'	D(v) p(v)	D(w) p(w)	D(x) p(x)	D(y) p(y)	D(z) p(z)
0	u					



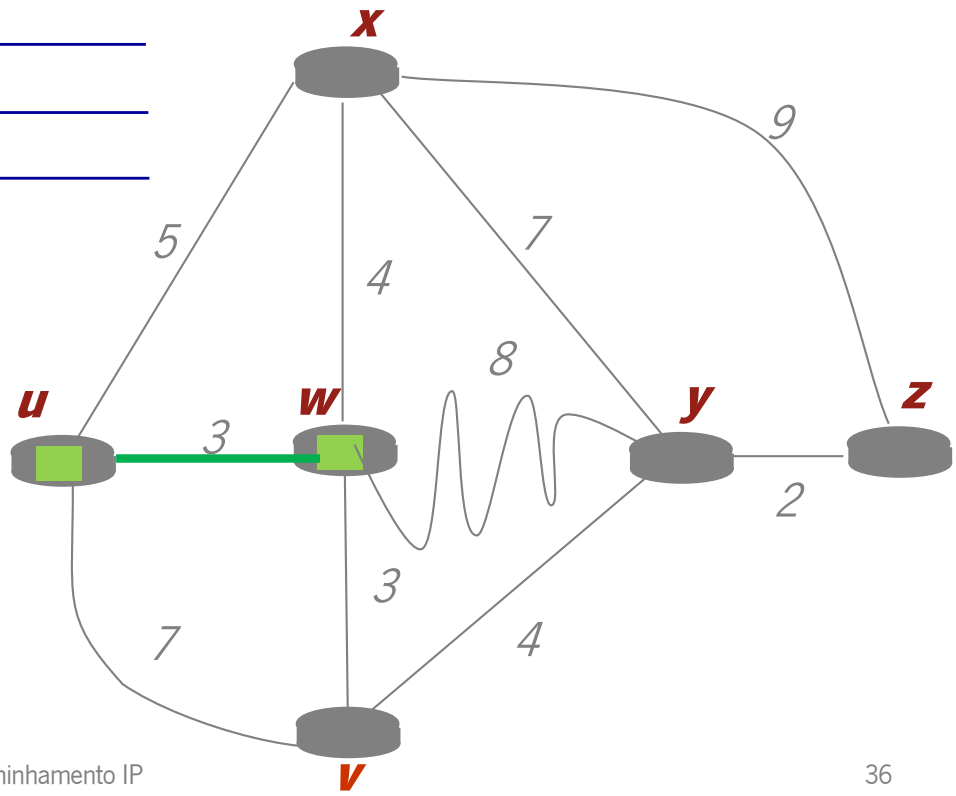
Encaminhamento IP
Algoritmo Dijkstra – Exemplo

Step	N'	D(v) p(v)	D(w) p(w)	D(x) p(x)	D(y) p(y)	D(z) p(z)
0	u	7,u	3,u	5,u	∞	∞



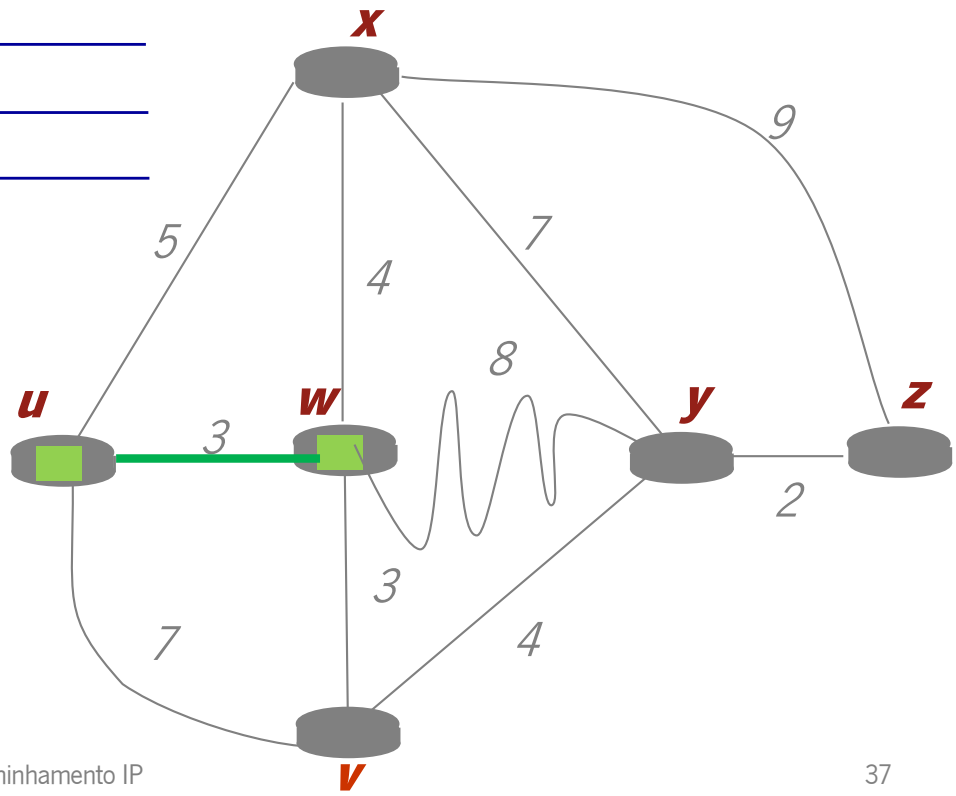
Encaminhamento IP
Algoritmo Dijkstra – Exemplo

Step	N'	D(v) p(v)	D(w) p(w)	D(x) p(x)	D(y) p(y)	D(z) p(z)
0	u	7,u	3,u	5,u	∞	∞
1	uw					



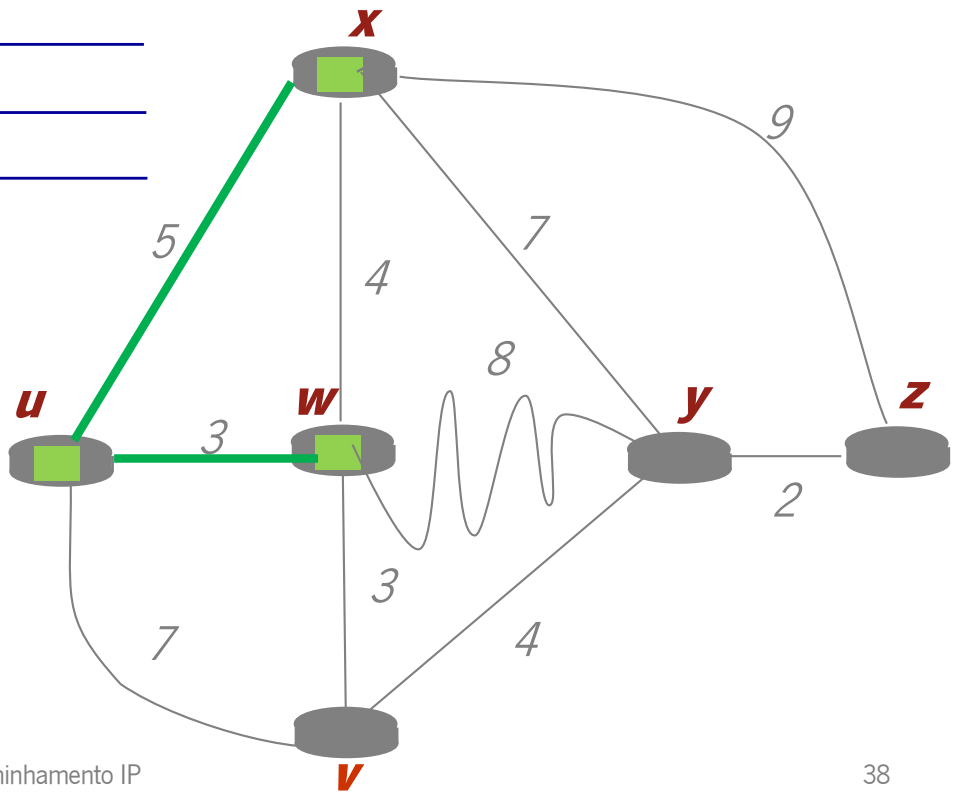
Encaminhamento IP
Algoritmo Dijkstra – Exemplo

Step	N'	D(v) p(v)	D(w) p(w)	D(x) p(x)	D(y) p(y)	D(z) p(z)
0	u	7,u	3,u	5,u	∞	∞
1	uw	6,w		5,u	11,w	∞



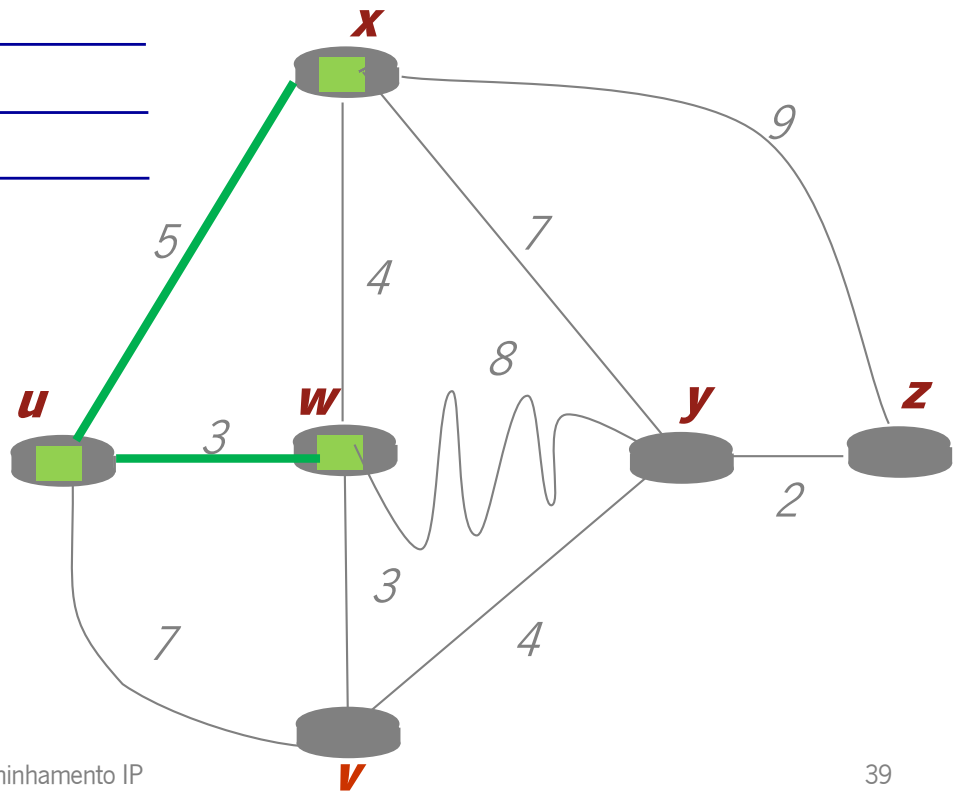
Encaminhamento IP
Algoritmo Dijkstra – Exemplo

Step	N'	D(v) p(v)	D(w) p(w)	D(x) p(x)	D(y) p(y)	D(z) p(z)
0	u	7,u	3,u	5,u	∞	∞
1	uw	6,w		5,u	11,w	∞
2	uwx					



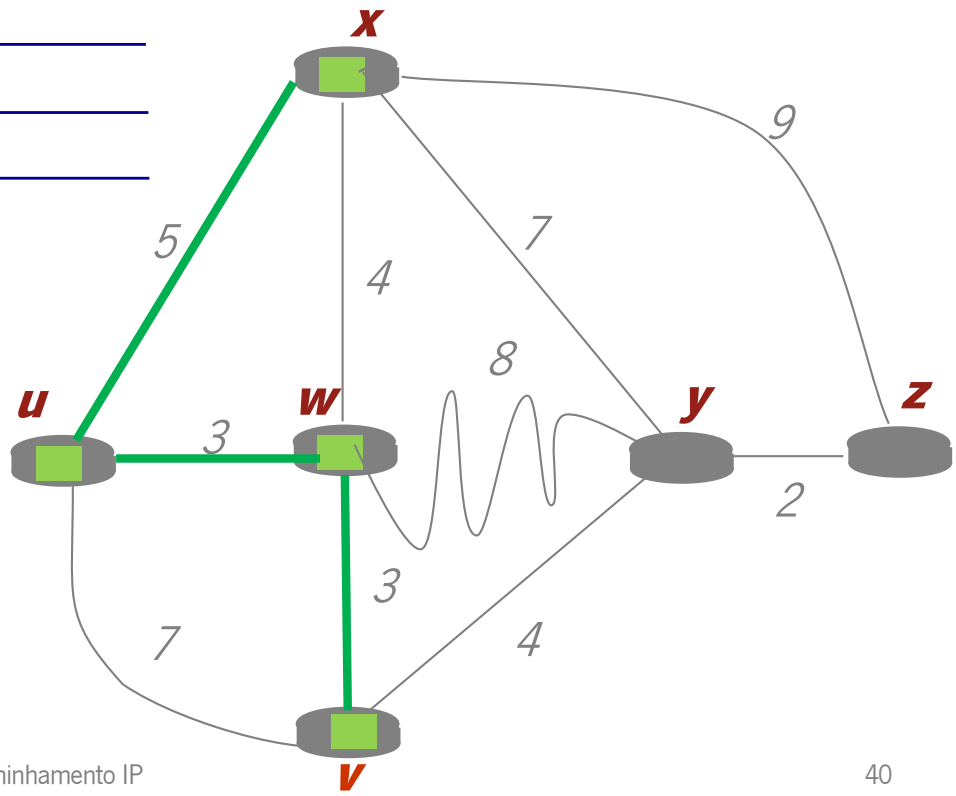
Encaminhamento IP
Algoritmo Dijkstra – Exemplo

Step	N'	D(v) p(v)	D(w) p(w)	D(x) p(x)	D(y) p(y)	D(z) p(z)
0	u	7,u	3,u	5,u	∞	∞
1	uw	6,w		5,u	11,w	∞
2	uwx	6,w			11,w	14,x



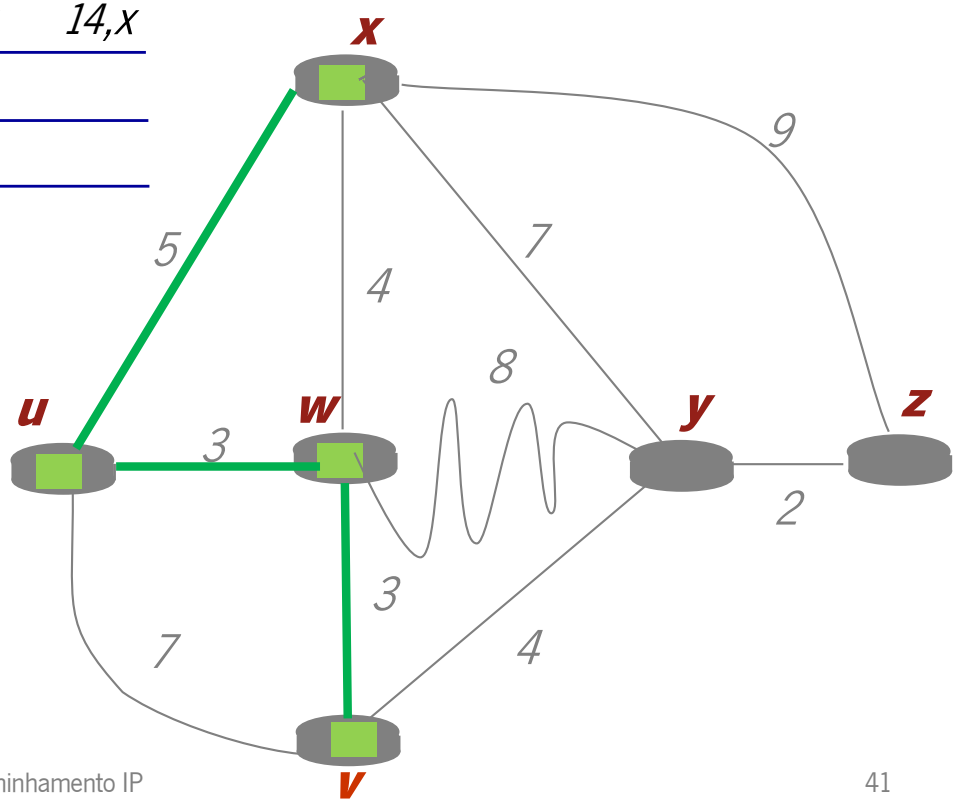
Encaminhamento IP
Algoritmo Dijkstra – Exemplo

Step	N'	D(v) p(v)	D(w) p(w)	D(x) p(x)	D(y) p(y)	D(z) p(z)
0	u	7,u	3,u	5,u	∞	∞
1	uw	6,w		5,u	11,w	∞
2	uwx	6,w			11,w	14,x
3	uwxv					



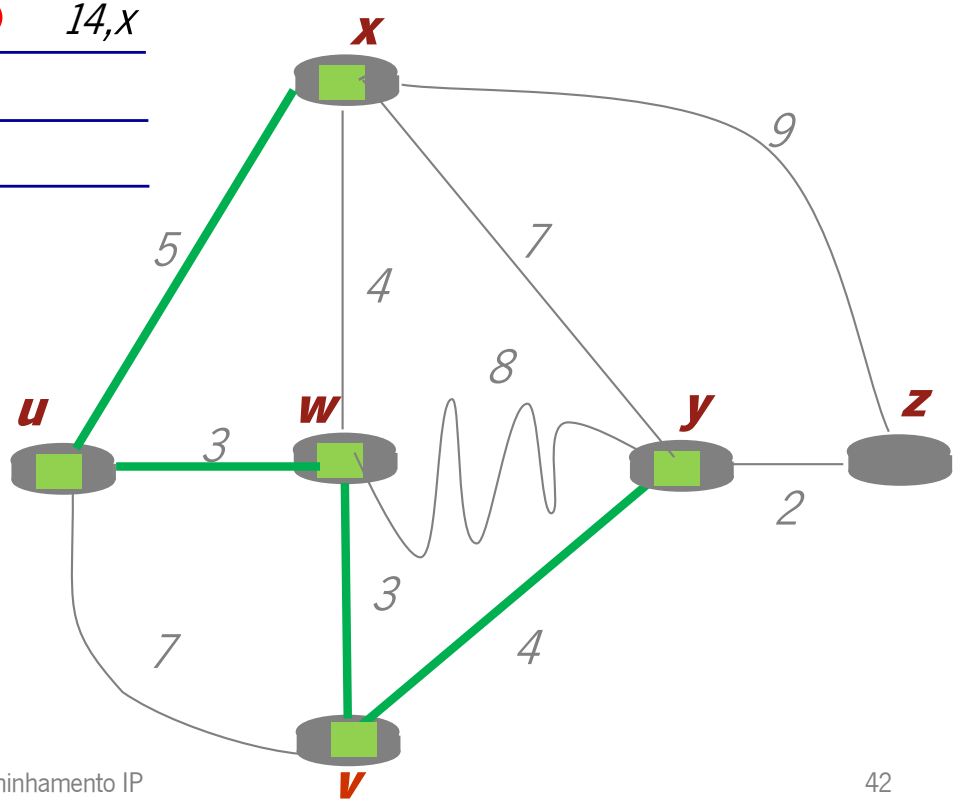
Encaminhamento IP
Algoritmo Dijkstra – Exemplo

Step	N'	D(v) p(v)	D(w) p(w)	D(x) p(x)	D(y) p(y)	D(z) p(z)
0	u	7,u	3,u	5,u	∞	∞
1	uw	6,w		5,u	11,w	∞
2	uwx	6,w			11,w	14,x
3	uwxv				10,v	14,x



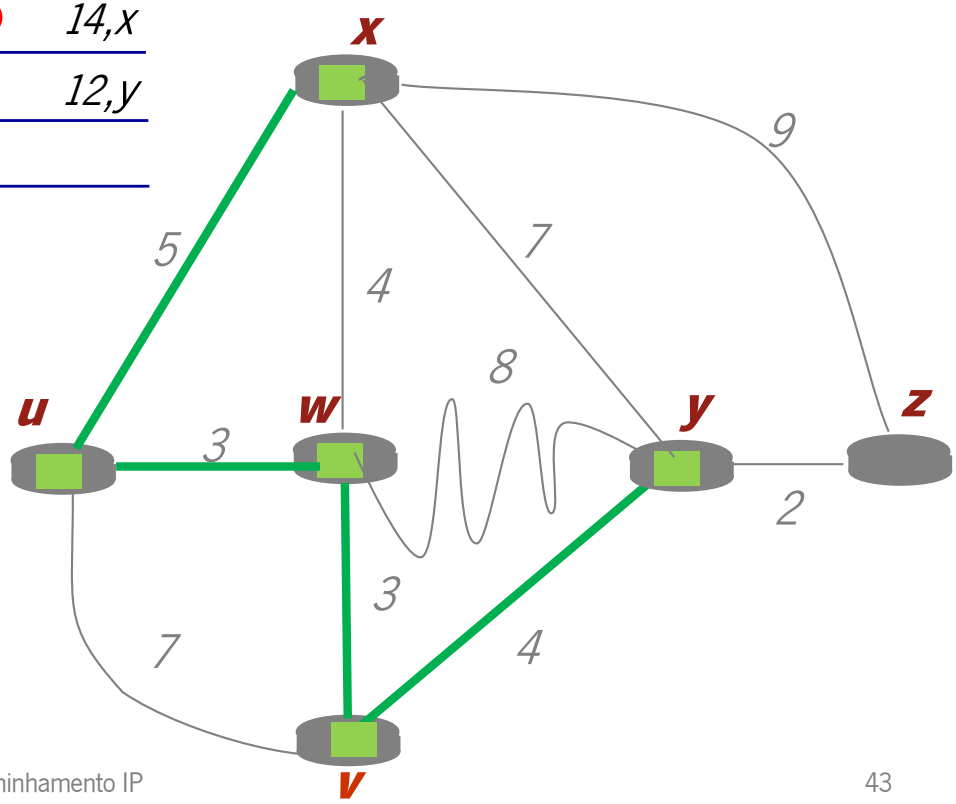
Encaminhamento IP
Algoritmo Dijkstra – Exemplo

Step	N'	D(v) p(v)	D(w) p(w)	D(x) p(x)	D(y) p(y)	D(z) p(z)
0	u	7,u	3,u	5,u	∞	∞
1	uw	6,w		5,u	11,w	∞
2	uwx	6,w			11,w	14,x
3	uwxv				10,v	14,x
4	uwxvy					



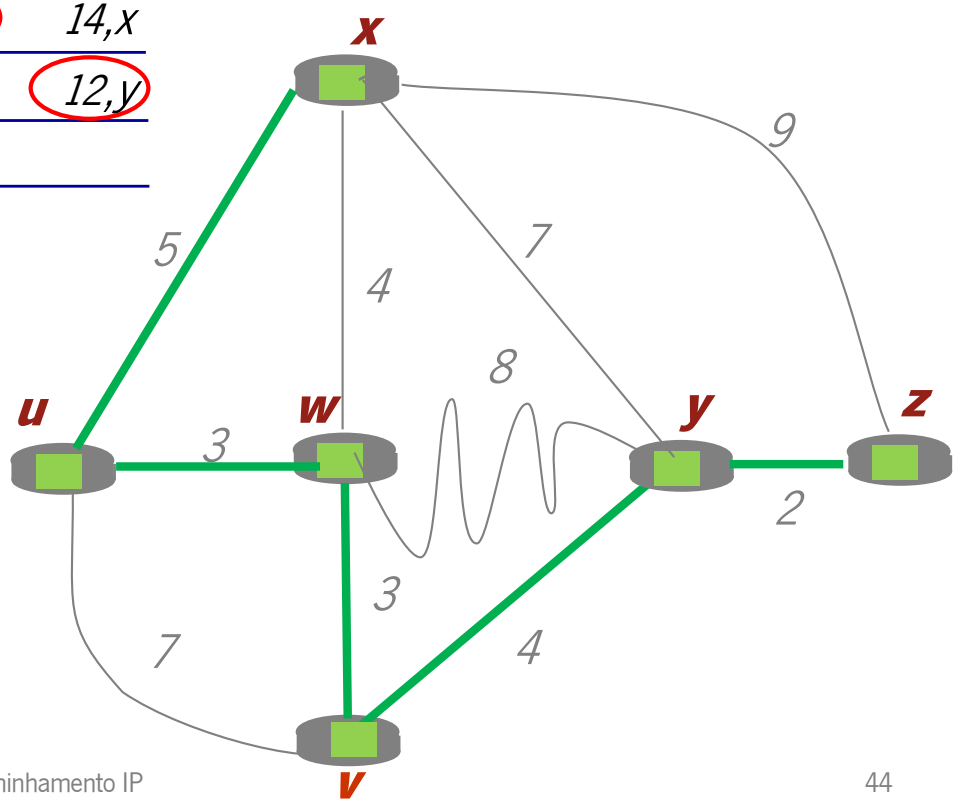
Encaminhamento IP
Algoritmo Dijkstra – Exemplo

Step	N'	D(v) p(v)	D(w) p(w)	D(x) p(x)	D(y) p(y)	D(z) p(z)
0	u	7,u	3,u	5,u	∞	∞
1	uw	6,w		5,u	11,w	∞
2	uwx	6,w			11,w	14,x
3	uwxv				10,v	14,x
4	uwxvy					12,y



Encaminhamento IP
Algoritmo Dijkstra – Exemplo

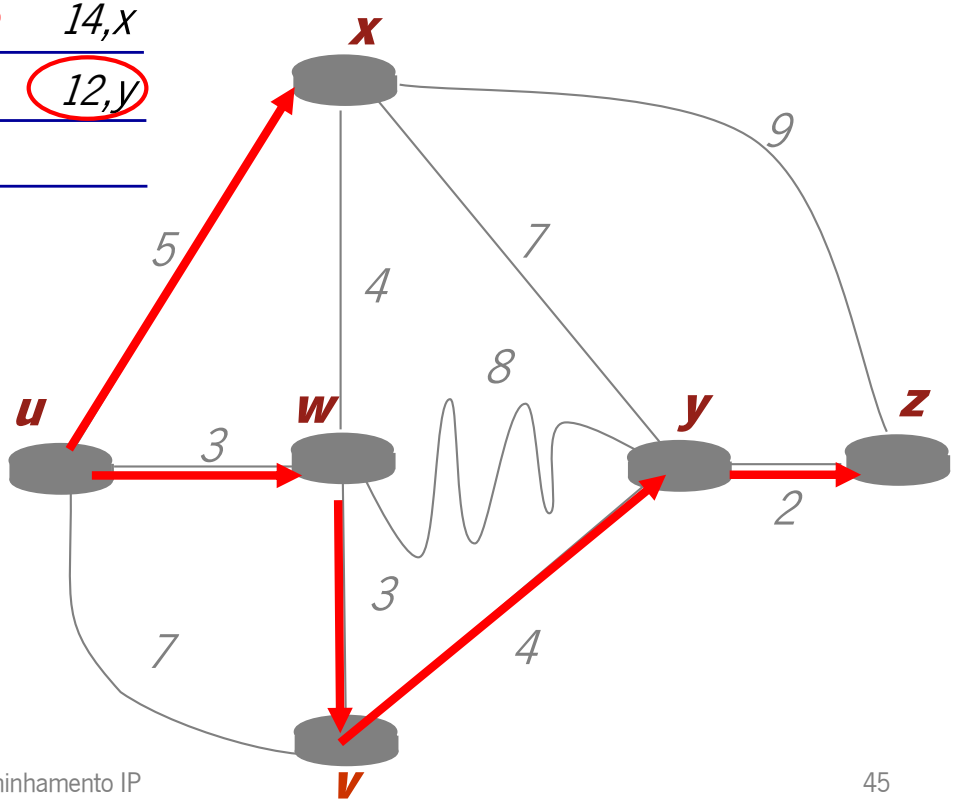
Step	N'	D(v) p(v)	D(w) p(w)	D(x) p(x)	D(y) p(y)	D(z) p(z)
0	u	7,u	3,u	5,u	∞	∞
1	uw	6,w		5,u	11,w	∞
2	uwx	6,w			11,w	14,x
3	uwxv				10,v	14,x
4	uwxvy					12,y
5	uwxvyz					



Encaminhamento IP
Algoritmo Dijkstra – Exemplo

Step	N'	D(v) p(v)	D(w) p(w)	D(x) p(x)	D(y) p(y)	D(z) p(z)
0	u	7,u	3,u	5,u	∞	∞
1	uw	6,w		5,u	11,w	∞
2	uwx	6,w			11,w	14,x
3	uwxv				10,v	14,x
4	uwxvy					12,y
5	uwxvyz					

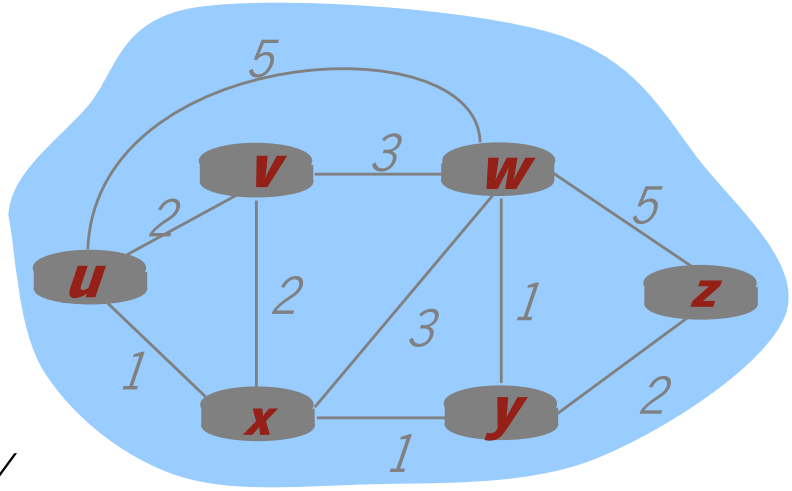
Nota: Construir a árvore de caminhos mais curtos, registando os predecessores... Quando existirem empates podem ser resolvidos arbitrariamente/aleatoriamente.



Encaminhamento IP

Algoritmo Dijkstra – Outro exemplo

Step	N'	D(v),p(v)	D(w),p(w)	D(x),p(x)	D(y),p(y)	D(z),p(z)
0	u	2,u	5,u	1,u	∞	∞
1	ux	2,u	4,x		2,x	∞
2	uxy	2,u	3,y			4,y
3	uxyv		3,y			4,y
4	uxyvw					4,y
5	uxyvwz					



Mais exercícios em:

http://gaia.cs.umass.edu/kurose_ross/interactive/

Encaminhamento IP
Algoritmo Dijkstra – Outro exemplo

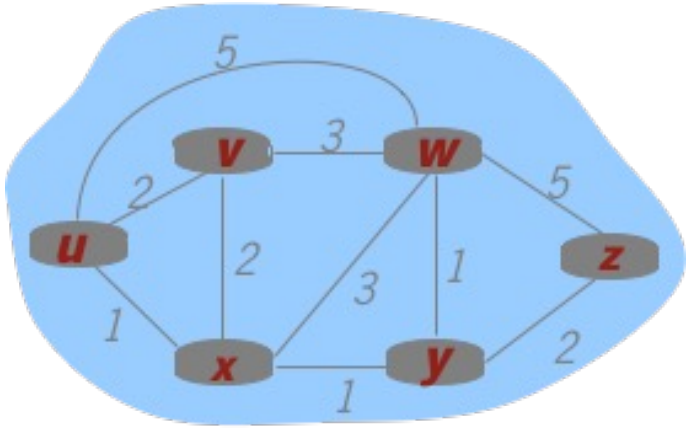
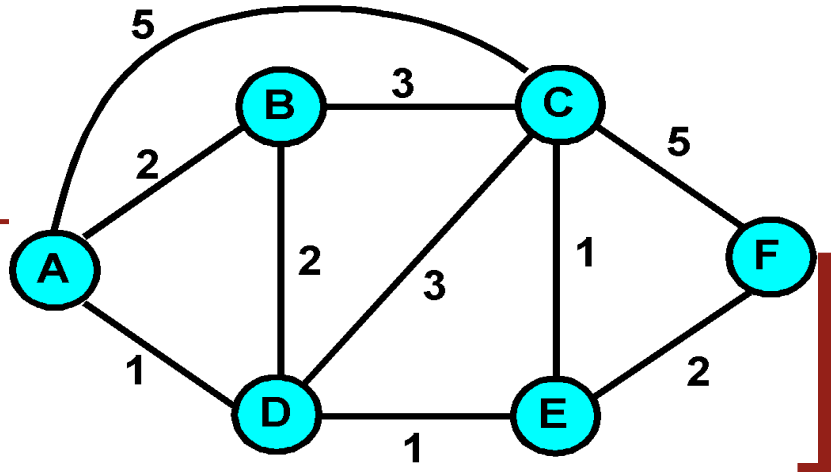


Tabela de encaminhamento do nó **u**

Destino	Próximo Nó	Link	Custo
u	u	——	0
v	v	(u,v)	2
w	x	(u,x)	3
x	x	(u,x)	1
y	x	(u,x)	2
z	x	(u,x)	4

Exercício LSA (Router F)



Cálculo (algoritmo Dijkstra, router F)

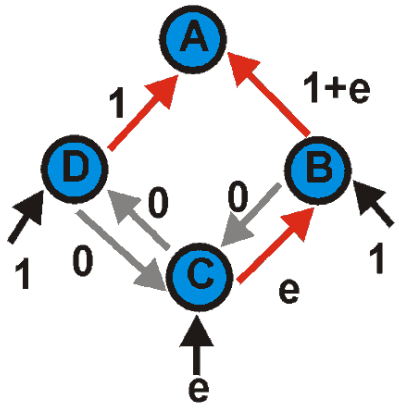
Cálculo (árvore caminhos mais curtos, router F)

Tabela Routing resultante (Router F)

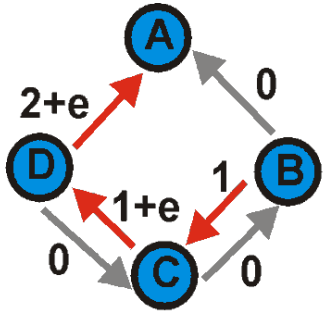
Destino	Próximo Salto	Link Saída	Custo
A			
B			
C			
D			
E			
F			

Encaminhamento IP
Algoritmos LS – Escalabilidade & Oscilações

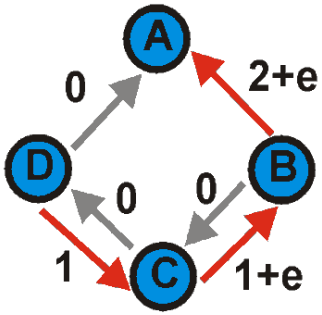
- Esforço/Complexidade do algoritmo para N nós da rede é $O(N^2)$, portanto, a implementação dos algoritmos LS têm alguns problemas de escalabilidade.
- Na presença de métricas assimétricas que espelham o estado da rede (por exemplo, se a métrica refletir a carga nas ligações) o cálculo da melhor rota sofre oscilações (ver exemplo abaixo em que a métrica reflete a quantidade de dados transmitidos).



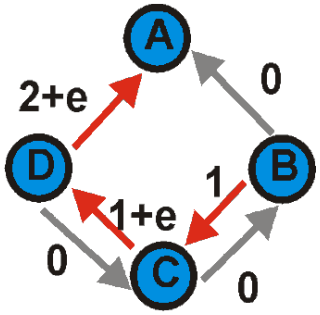
(a): initial routing



(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

Encaminhamento IP
Conceitos – Tipos de Algoritmos

Os algoritmos de encaminhamento podem gerir a informação de duas formas distintas:

Global:

- Todos os encaminhadores têm um conhecimento completo da topologia e custo das ligações;
- Algoritmos de estado das ligações (***Link State*** **– LS**).

Descentralizada:

- Os encaminhadores só conhecem os vizinhos a que estão fisicamente/logicamente ligados e o custo das ligações respetivas;
- O processo de computação é iterativo, havendo troca de informação entre vizinhos;
- Algoritmos de vetor de distância (***Distance Vector*** **– DV**).

Encaminhamento IP
Algoritmos *Distance Vector* (DV)

Ao contrário dos algoritmos LS, os algoritmos DV não usam informação global, são distribuídos, iterativos e assíncronos.

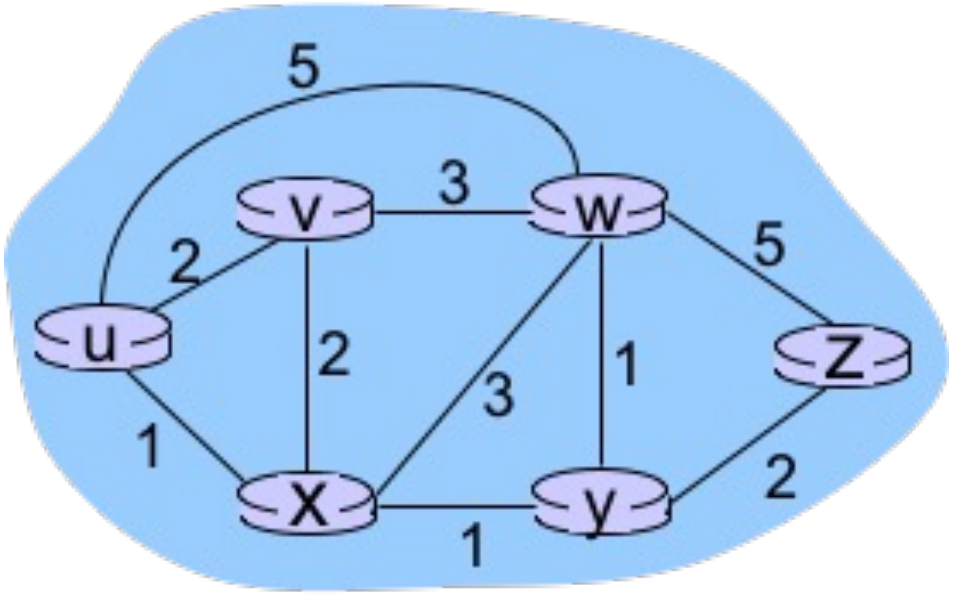
- Cada nó recebe informação de encaminhamento de algum dos seus vizinhos diretos, recalcula a tabela de encaminhamento e envia essa informação de encaminhamento de volta;
- O processo continua até que não haja informação de encaminhamento a ser trocada entre nós vizinhos, i.e., até que a informação de encaminhamento convirja;
- Não exige que os nós estejam sincronizados uns com os outros em relação à topologia completa da rede.

Encaminhamento IP
Algoritmos DV – Equação de Bellman-Ford

Seja **c(x,v)** o custo do caminho entre **x** e **v** adjacentes e $\mathbf{V_x}$ o grupo de todos os nós vizinhos/adjacentes a **x**, então o custo do melhor caminho de **x** para **y** (ou a rota de custo mínimo entre o nó **x** e o nó **y**) é dado por:

$$\mathbf{d_x(y) = min \{ \ c(x,v) + d_v(y) \ \}},$$ para todos os **v** em $\mathbf{V_x}$

Encaminhamento IP
Algoritmos DV – Equação de Bellman-Ford



Exemplo:

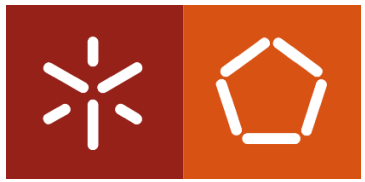
Se, em certo momento,
já se souber que:

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

então,

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

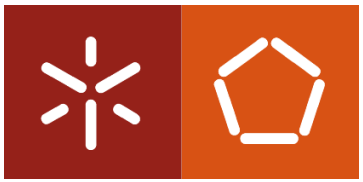
Encaminhamento IP
Algoritmos DV



Sabendo que:

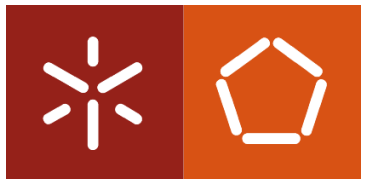
- **N'** é o grupo de todos os nós da rede e que **x ϵ N'**
- $\mathbf{V_x}$ é o grupo de nós adjacentes/vizinhos de **x**
- O nó **x** conhece o custo para todos os seus vizinhos
  $\mathbf{C_x = \{ c(x,v) \}, v \epsilon V_x}$
- O custo do melhor caminho de **x** para **y** é dado por
  $\mathbf{d_x(y) = min \{ \ c(x,v) + d_v(y) \ \}, y \epsilon N', v \epsilon V_x}$
- Seja o custo do melhor caminho de **x** para **y** expresso por
  $\mathbf{D_x(y), y \epsilon N'}$
- Então...

Encaminhamento IP
Algoritmos DV

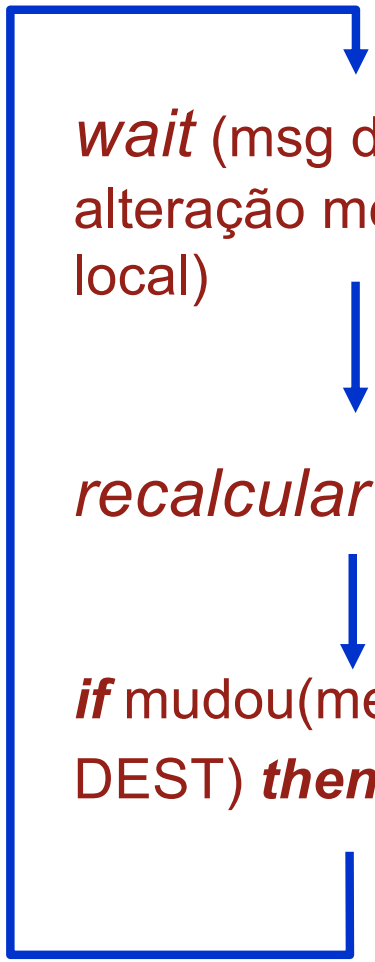


- O nó **x** mantém um vetor de distâncias próprio expresso por $\mathbf{DV_x = \{ D_x(y) \}, y \in N'}$
- O nó **x** também mantém os vetores de distâncias dos seus vizinhos $\mathbf{DV_v = \{ D_v(y) \}, y \in N', v \in V_x}$
- Cada nó **x** envia periodicamente a sua *estimativa* $\mathbf{DV_x}$ a todos os seus vizinhos
- Quando um nó **x** recebe um novo $\mathbf{DV_v}$ de um dos seus vizinhos atualiza o seu próprio vetor $\mathbf{DV_x}$
- Em condições normais, a estimativa de $\mathbf{D_x(y)}$ converge para o valor de $\mathbf{d_x(y)}$ ao fim de algum tempo
- A troca contínua dos DV mantém a convergência

Encaminhamento IP
Algoritmos DV



Cada nó:

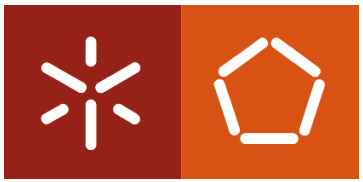


O processo é iterativo e assíncrono.
Cada iteração local é causada por:

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

O processo é distribuído.
Cada nó notifica vizinhos apenas quando muda o menor custo para qualquer destino.

Encaminhamento IP
Algoritmo Bellman-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:

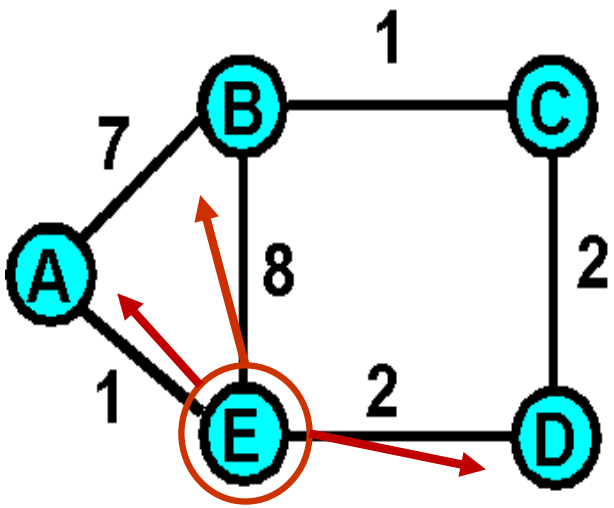
Fonte: Computer Networking: A Top-Down Approach Featuring the Internet, J. Kurose, Addison-Wesley

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

Encaminhamento IP
Algoritmo Bellman-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 for 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 as "newval" */
21      for the single destination y: D^X(Y,V) = c(X,V) + newval
22
23    if we have a new min_w D^X(Y,w) for any destination Y
24       send new value of min_w D^X(Y,w) to all neighbors
```

Encaminhamento IP
Algoritmo Bellman-Ford – Exemplo

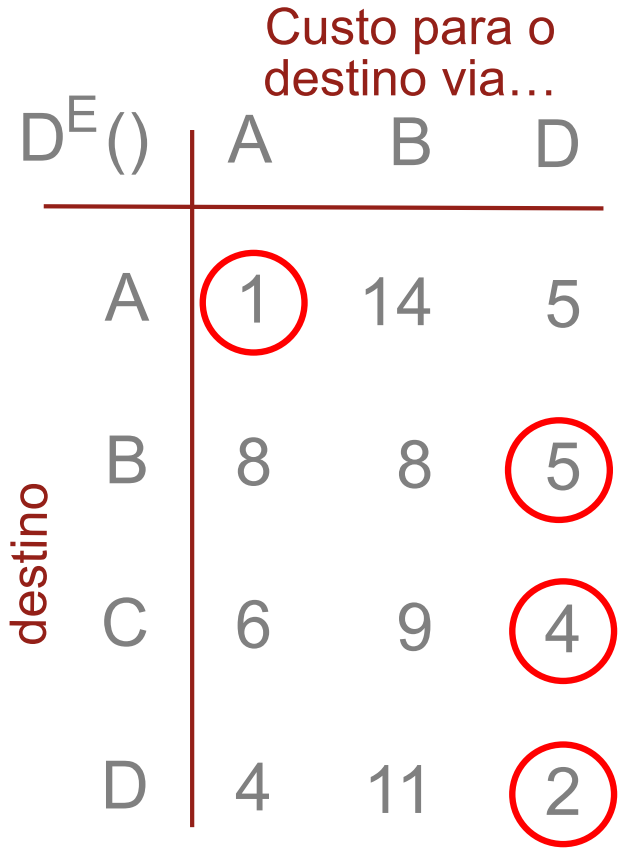


E

dest.	D( ) via A	D( ) via B	D( ) via D
A	(1)	14	5
B	7	8	(5)
C	6	9	(4)
D	4	11	(2)

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

Encaminhamento IP
Algoritmo Bellman-Ford – Exemplo



destino	Saída	Custo
A	A,	1
B	D,	5
C	D,	4
D	D,	2

Tabela DV → Tabela Encaminhamento

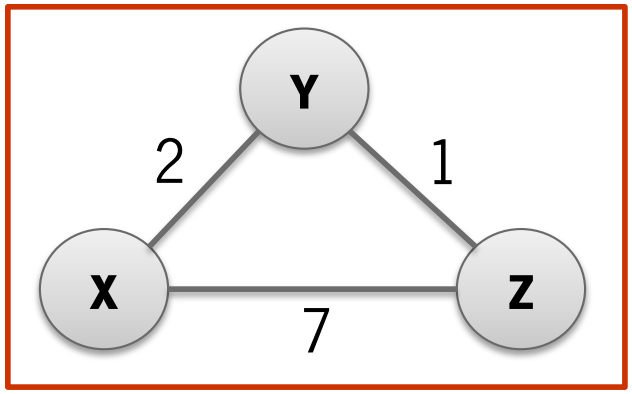
Encaminhamento IP
Algoritmo Bellman-Ford – Exemplo

Tabelas DV

Destinos	D^X	Vizinhos Y	Vizinhos Z
	Y	**2**	∞
	Z	∞	**7**

Destinos	D^Y	Vizinhos X	Vizinhos Z
	X	**2**	∞
	Z	∞	**1**

Destinos	D^Z	Vizinhos X	Vizinhos Y
	X	**7**	∞
	Y	∞	**1**



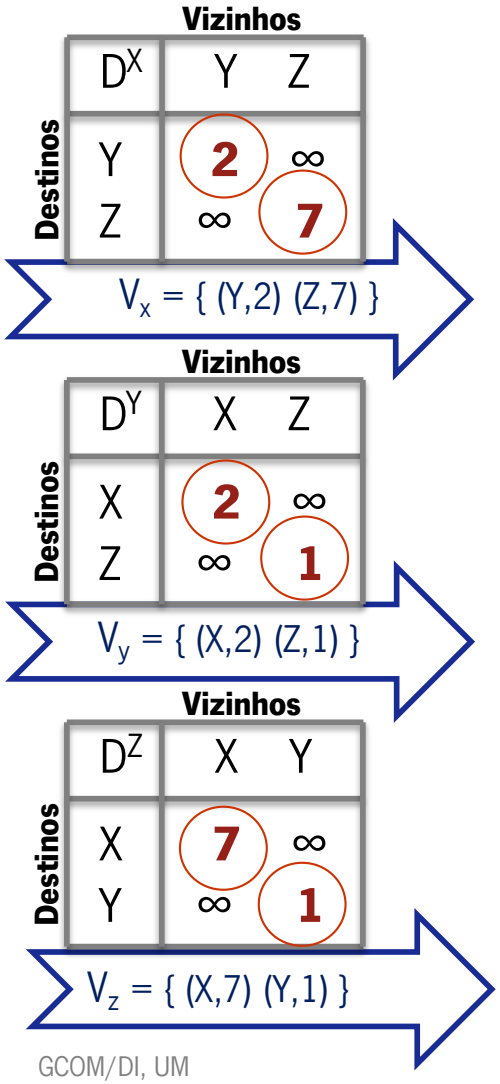
Inicialização:

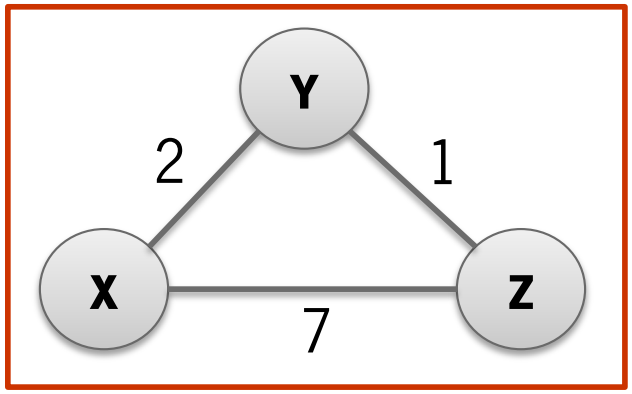
...Custo para os vizinhos é o custo do link direto

...Todos os outros a infinito

Encaminhamento IP
Algoritmo Bellman-Ford – Exemplo

Tabelas DV





...Preparar DV com as melhores distâncias

....Enviar DV a todos os vizinhos

Encaminhamento IP
Algoritmo Bellman-Ford – Exemplo

Tabelas DV

Destinos	Vizinhos Y	Vizinhos Z
Y	**2**	∞
Z	∞	**7**

$V_x = \{ (Y,2) (Z,7) \}$

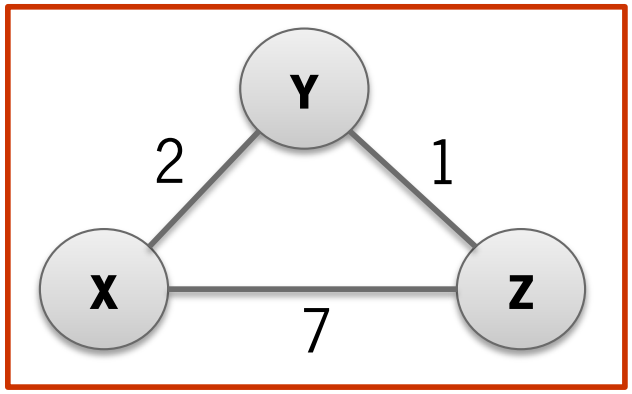
Destinos	Vizinhos X	Vizinhos Z
X	**2**	∞
Z	∞	**1**

$V_y = \{ (X,2) (Z,1) \}$

Destinos	Vizinhos X	Vizinhos Y
X	**7**	∞
Y	∞	**1**

$V_z = \{ (X,7) (Y,1) \}$

Destinos	Vizinhos Y	Vizinhos Z
Y	**2**	8
Z	**3**	7



O Nó X Recebe Vetores de Y e Z e atualiza tabela:

→ Y diz que chega a Z com custo 1

$D^X(Z,Y) = c(X,Y) + D^Y(Z,Z) = 2 + 1 = 3$

→ Z diz que chega a Y com custo de 1

$D^X(Y,Z) = c(X,Z) + D^Z(Y,Y) = 7 + 1 = 8$

aplicando a equação Bellman-Ford!

Encaminhamento IP
Algoritmo Bellman-Ford – Exemplo

Tabelas DV

Destinos	Vizinhos Y	Vizinhos Z
Y	**2**	∞
Z	∞	**7**

V_x = { (Y,2) (Z,7) }

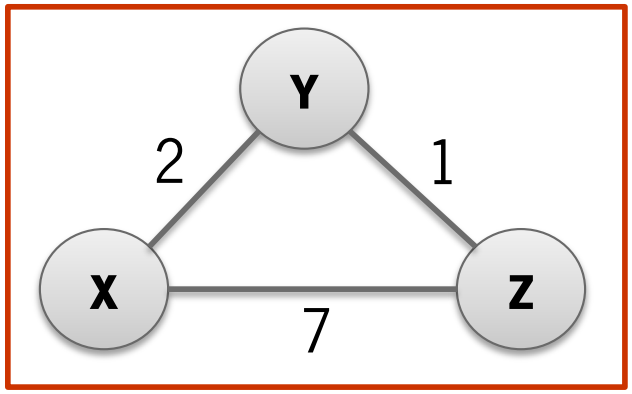
Destinos	Vizinhos Y	Vizinhos Z
Y	**2**	8
Z	**3**	7

Destinos	Vizinhos X	Vizinhos Z
X	**2**	∞
Z	∞	**1**

V_y = { (X,2) (Z,1) }

Destinos	Vizinhos X	Vizinhos Y
X	**7**	∞
Y	∞	**1**

V_z = { (X,7) (Y,1) }



Escolhe as melhores distâncias...

... se houver alterações envia aos vizinhos!

Encaminhamento IP
Algoritmo Bellman-Ford – Exemplo

Tabelas DV

Destinos	Vizinhos Y	Vizinhos Z
Y	**2**	∞
Z	∞	**7**

$V_x = \{ (Y,2) (Z,7) \}$

Destinos	Vizinhos X	Vizinhos Z
X	**2**	∞
Z	∞	**1**

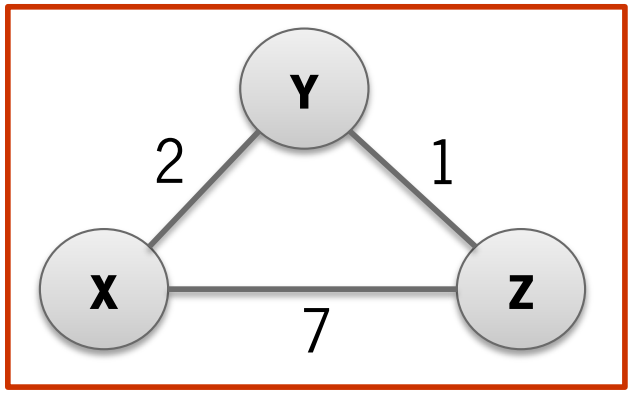
$V_y = \{ (X,2) (Z,1) \}$

Destinos	Vizinhos X	Vizinhos Y
X	**7**	∞
Y	∞	**1**

$V_z = \{ (X,7) (Y,1) \}$

Destinos	Vizinhos Y	Vizinhos Z
Y	**2**	8
Z	**3**	7

$V_x = \{ (Y,2) (Z,3) \}$



... na segunda iteração

Encaminhamento IP
Algoritmo Bellman-Ford – Exemplo

Tabelas DV

Vizinhos			
Destinos	D^X	Y	Z
	Y	**2**	∞
	Z	∞	**7**

$V_x = \{ (Y,2) (Z,7) \}$

Vizinhos			
Destinos	D^Y	X	Z
	X	**2**	∞
	Z	∞	**1**

$V_y = \{ (X,2) (Z,1) \}$

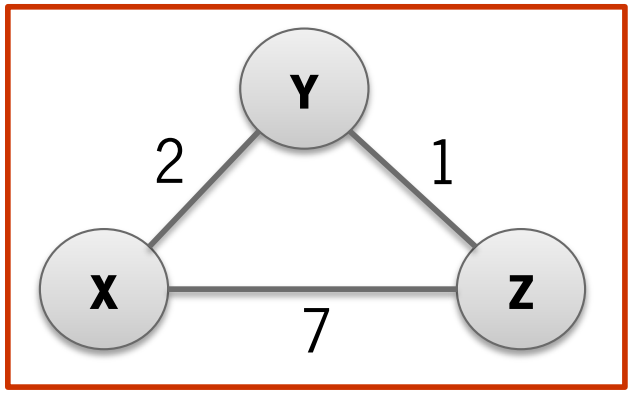
Vizinhos			
Destinos	D^Z	X	Y
	X	**7**	∞
	Y	∞	**1**

$V_z = \{ (X,7) (Y,1) \}$

Vizinhos			
Destinos	D^X	Y	Z
	Y	**2**	8
	Z	3	**7**

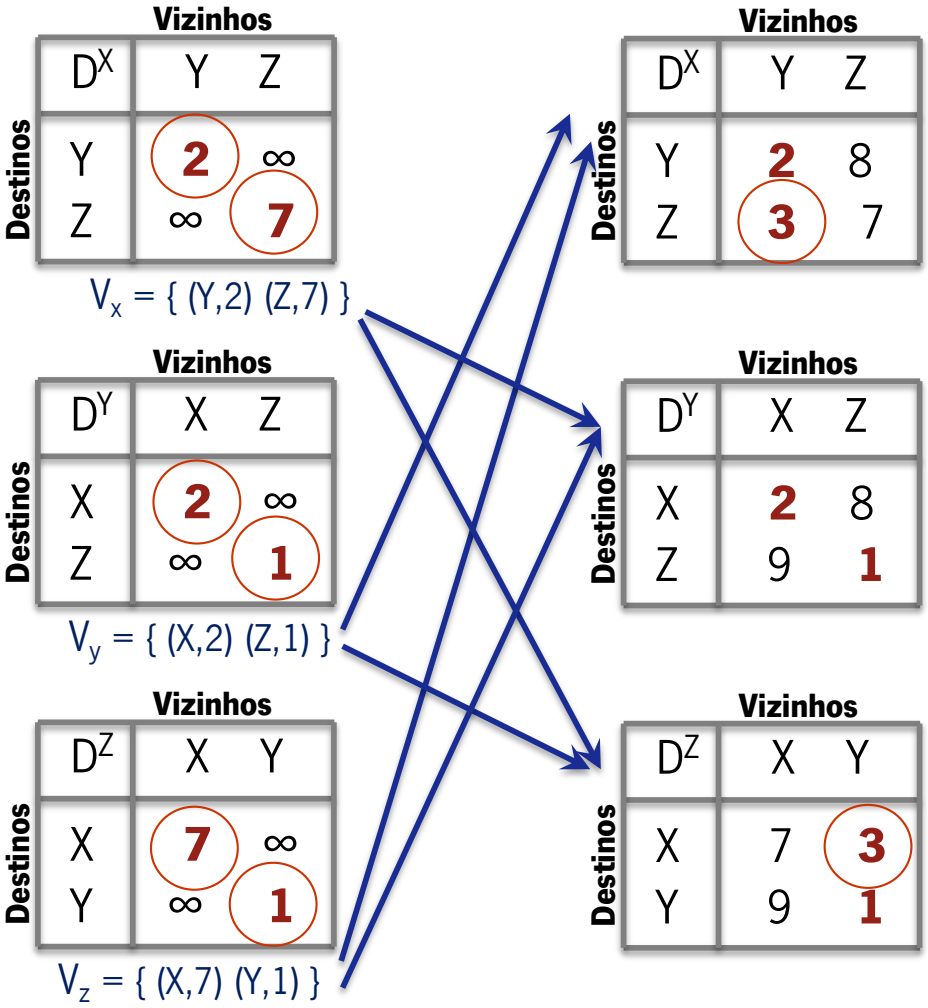
Vizinhos			
Destinos	D^Y	X	Z
	X	**2**	8
	Z	9	**1**

Vizinhos			
Destinos	D^Z	X	Y
	X	**7**	3
	Y	9	**1**



... o mesmo se passa nos nós Y e Z

Encaminhamento IP
Algoritmo Bellman-Ford – Exemplo

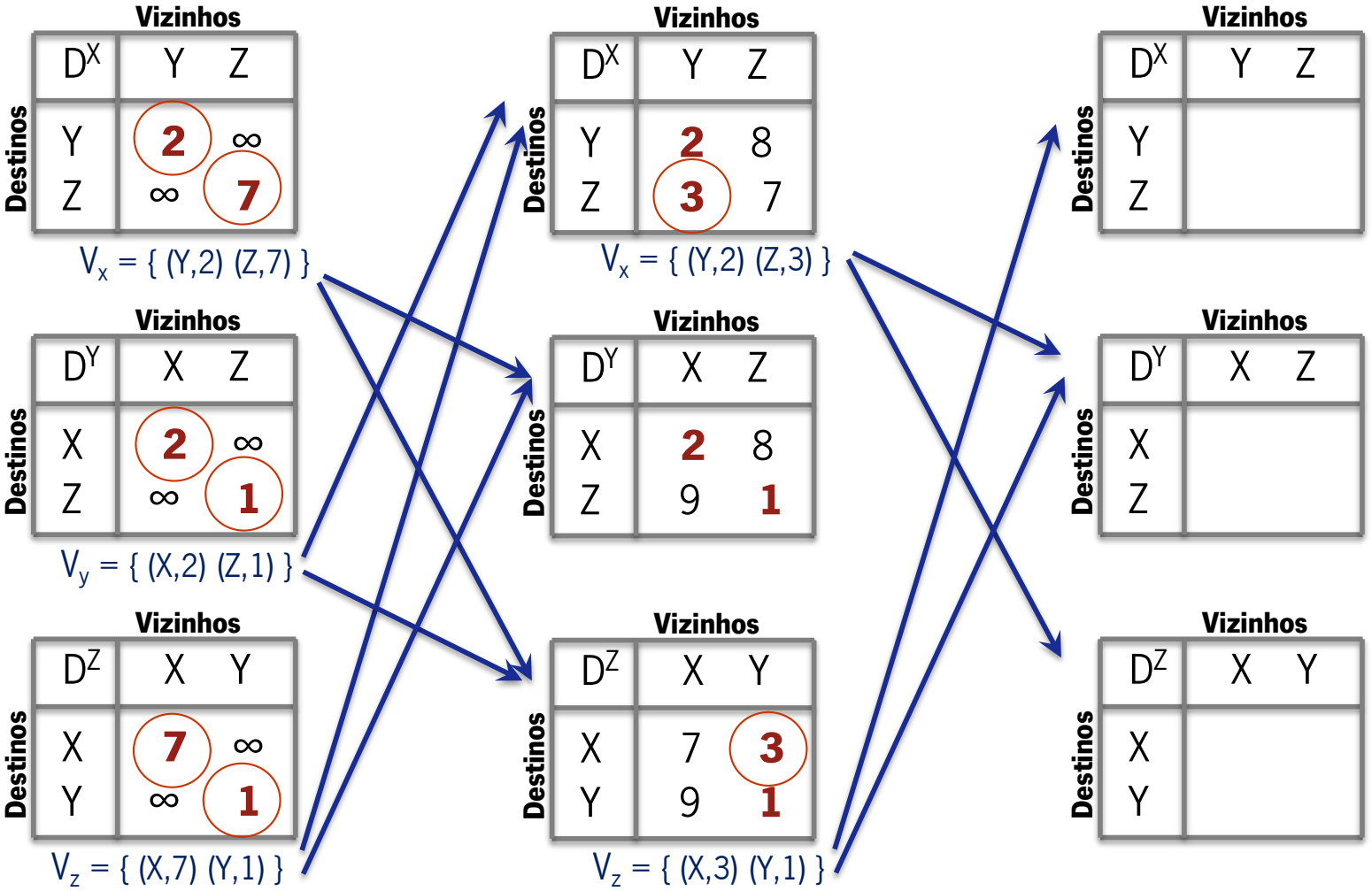


Nó X: Aprendeu uma nova rota para Z por Y

Nó Y... não melhorou nenhuma rota.
Convergiu. Não envia nada.

Nó Z: Aprendeu uma nova rota para X por Y

Encaminhamento IP
Algoritmo Bellman-Ford – Exemplo

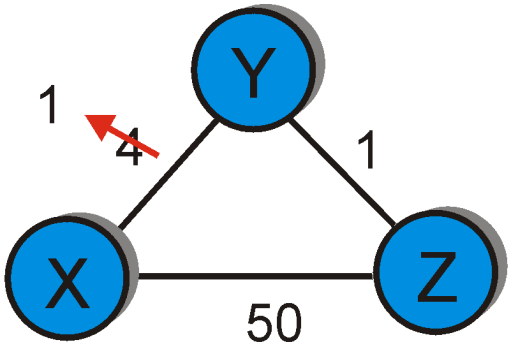


Que acontece na segunda iteração?

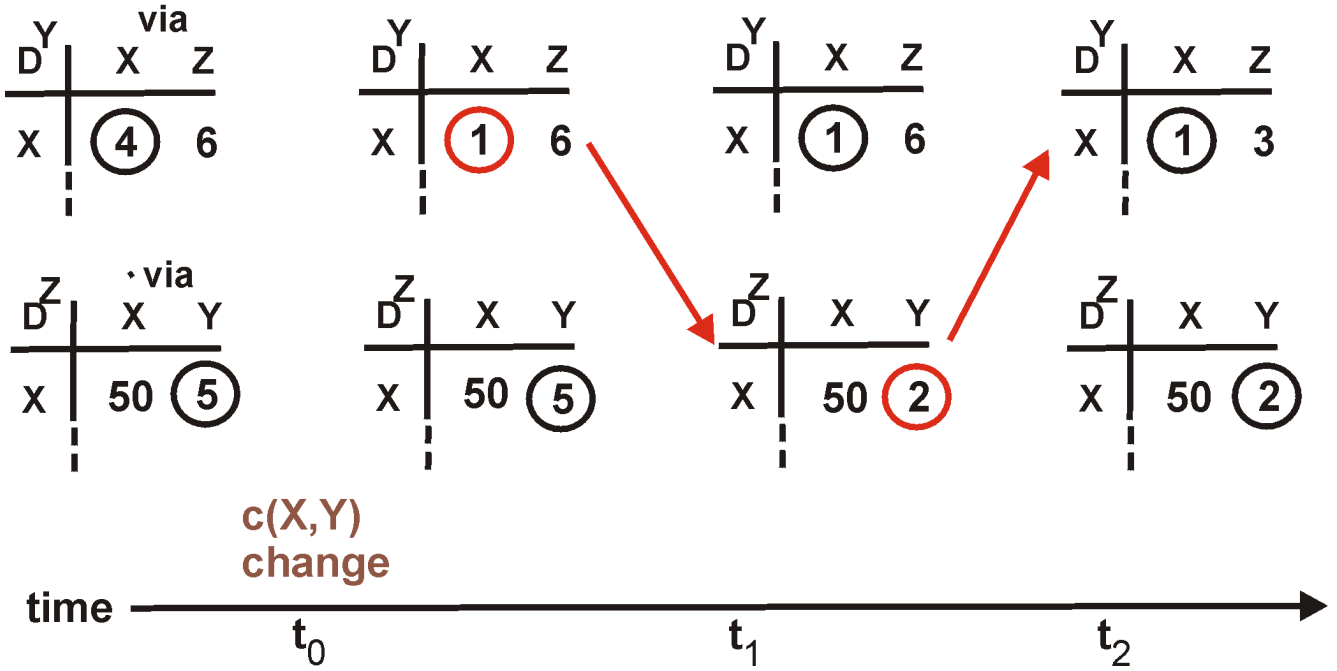
Já convergiu?

Encaminhamento IP
Algoritmos DV – Problemas

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

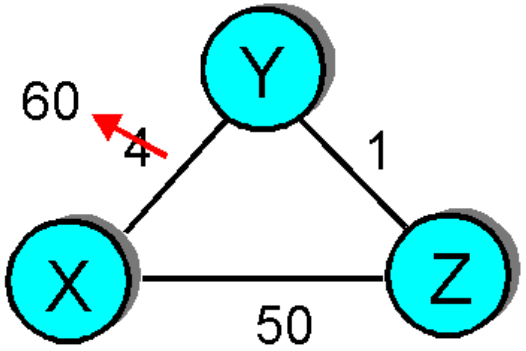


DV: Good news...
...travel fast

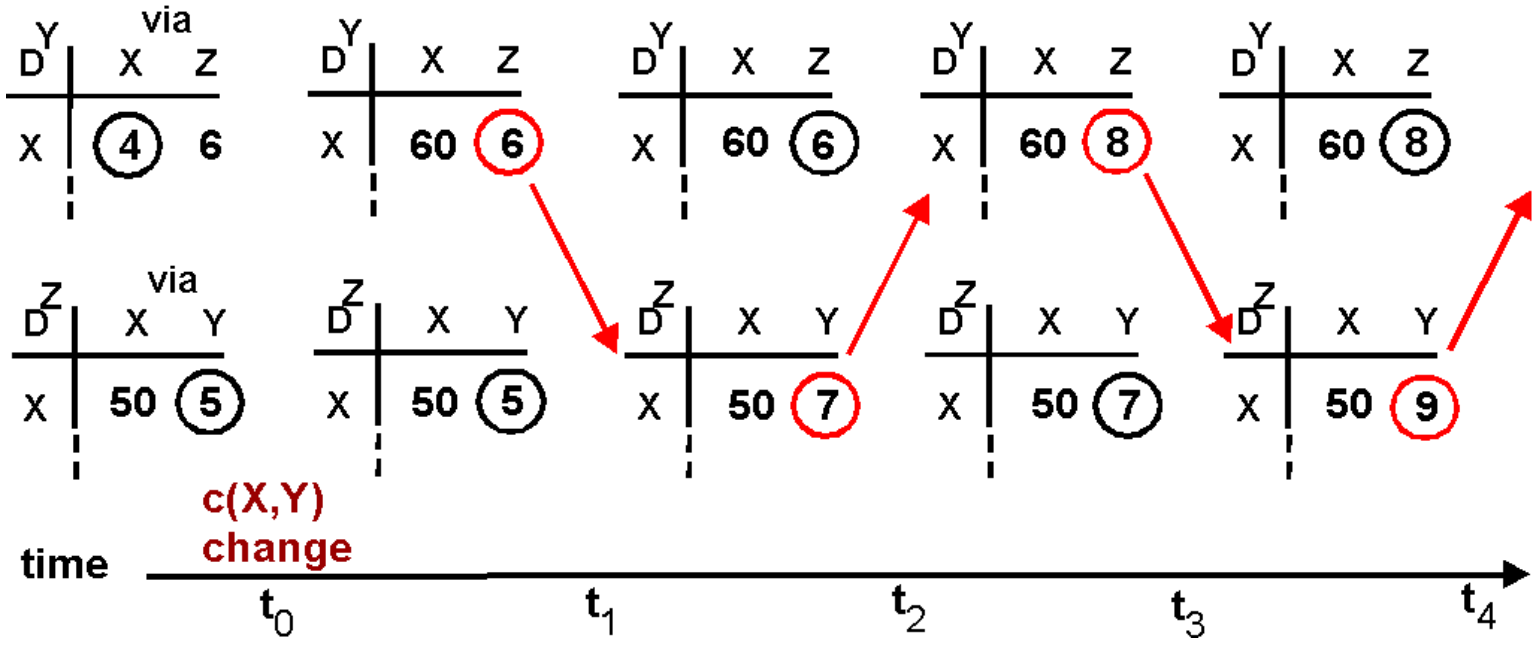


Encaminhamento IP
Algoritmos DV – Problemas

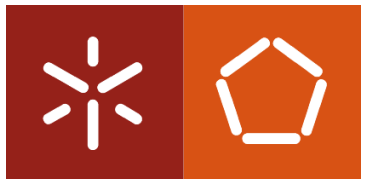
Fonte: *Computer Networking: A Top-Down Approach Featuring the Internet, J. Kurose, Addison-Wesley, 2001*



DV: Bad news...
…travel slow…
and may count to infinity…



Encaminhamento IP
Algoritmos DV – Soluções para Loops...



- **Divisão do horizonte (*Split Horizon*)**

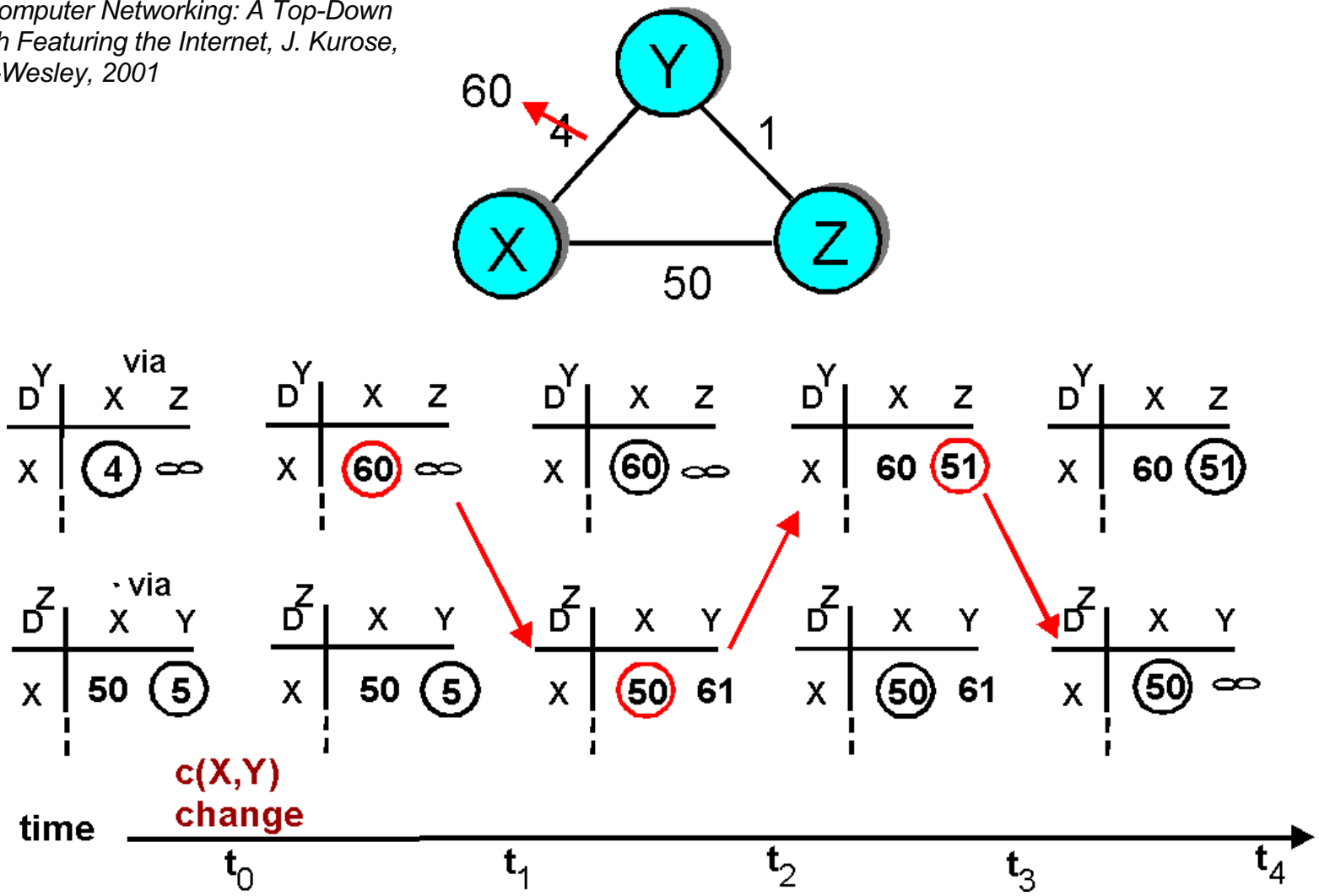
Se Y aprendeu rota para X com Z, nunca ensina essa rota a Z!

- **Envenenamento do percurso inverso (*Poison Reverse*)**

Se Y aprendeu rota para X com Z, então "mente" a Z anunciando que o custo da sua rota para X é igual a infinito!

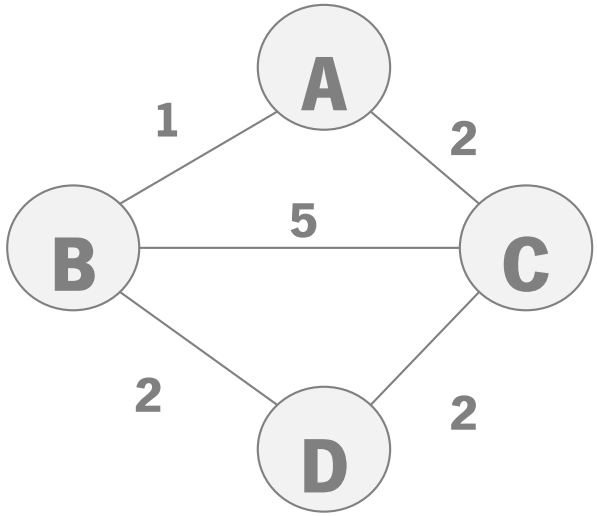
Encaminhamento IP
Algoritmos DV – Envenenamento Inverso

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



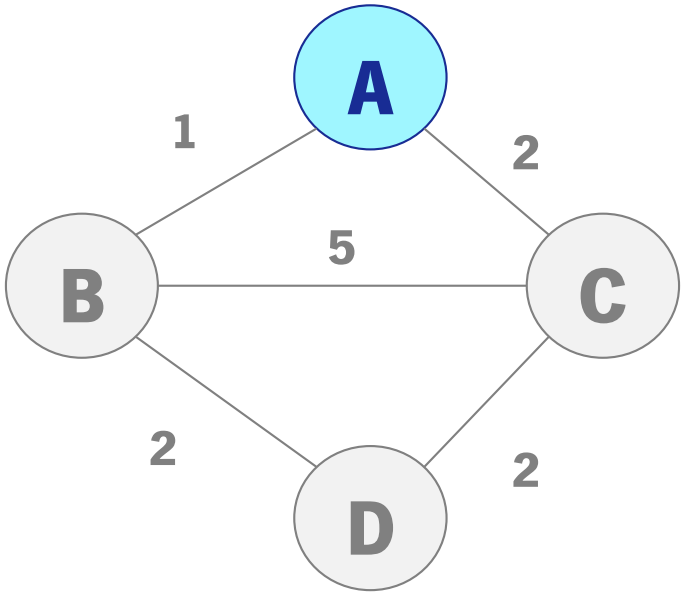
Encaminhamento IP
Exercícios

1. Calcule as tabelas DV para todos os nós seguindo o algoritmo Bellman-Ford até o encaminhamento em todos os nós convergir, tendo em consideração que usava o mecanismo de envenenamento da rota inversa para evitar ciclos/loops.



Exercício

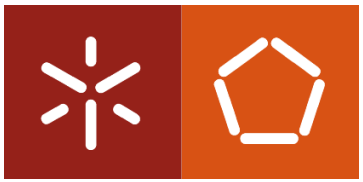
- Qual a tabela de distâncias final do nó A?
- Nessas circunstâncias, que anúncios faz/faria A para os seus vizinhos, usando divisão do horizonte e envenenamento do percurso inverso?



D^A	B	C
B		
C		
D		

Encaminhamento IP

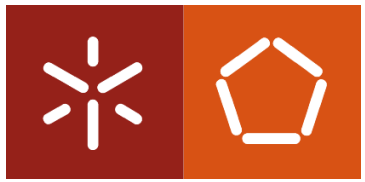
Reflexões finais comparativas – LS vs DV



- **Sobrecarga introduzida pela mensagens de controlo:**
  - Nos algoritmos LS 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 para que todos conheçam a nova topologia;
  - Nos algoritmos DV a mudança do custo de uma ligação só provoca o envio de mensagens se resultar na mudança da tabela de encaminhamento.
- **Convergência:**
  - Os algoritmos LS convergem mais depressa mas, com alguns tipos de métricas dinâmicas estão sujeitos a oscilações;
  - Os algoritmos DV convergem lentamente, podem apresentar ciclos enquanto não convergem, e é necessário incluir mecanismos para resolver o problema dos *loops* de eventuais contagens até ao infinito.

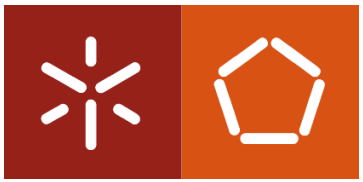
Encaminhamento IP

Reflexões finais comparativas – LS vs DV



- **Robustez:**
  - Nos algoritmos LS, 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 uma robustez maior.
  - Nos algoritmos DV, 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 LS são mais exigentes do que os algoritmos DV, quer em termos de memória (base de dados topológica vs tabela de distâncias), quer em termos de capacidade de processamento.

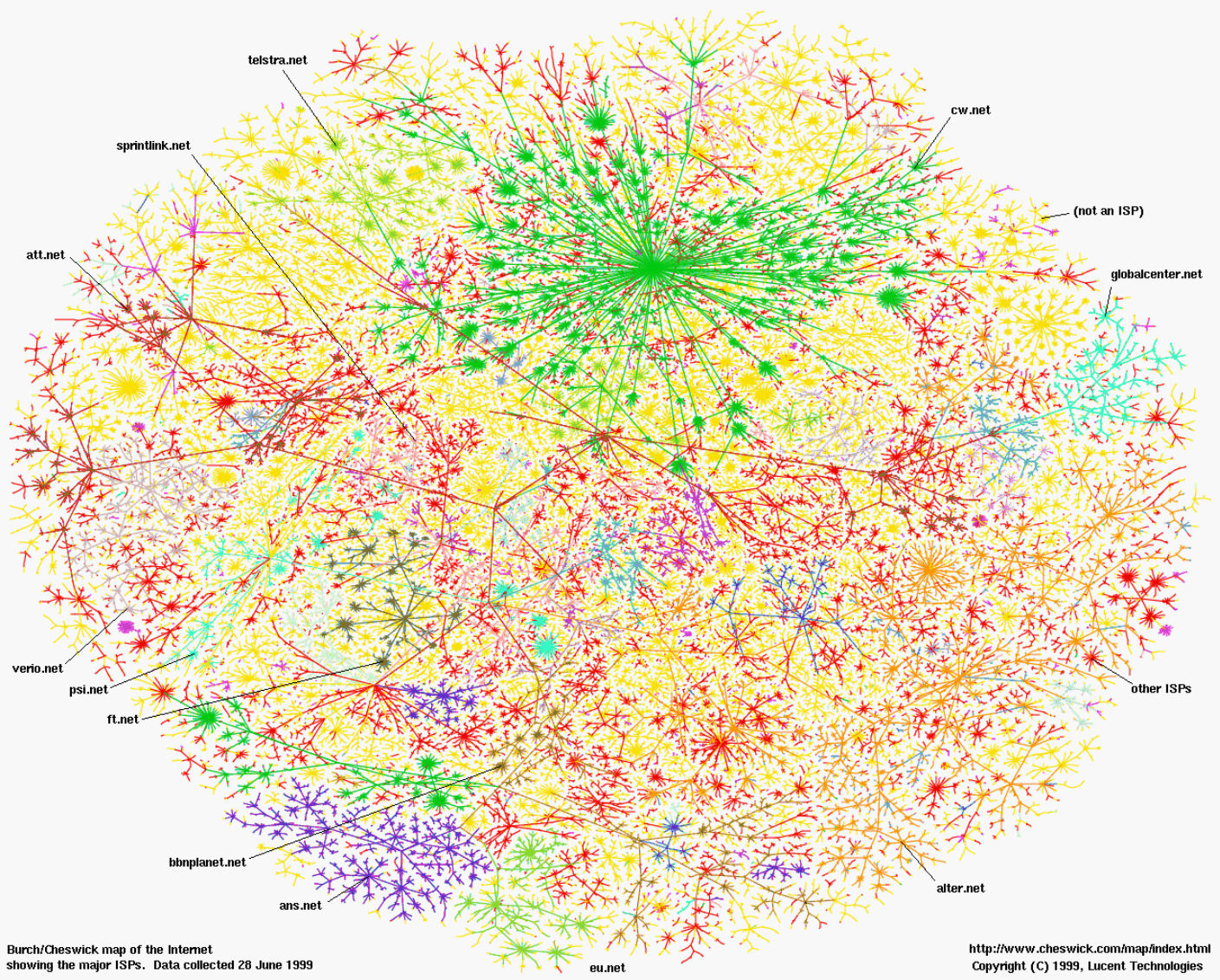
Encaminhamento IP
PROTOCOLOS



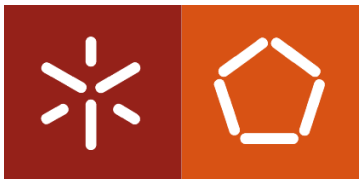
- **Baseados em algoritmos LS:**
  - OSPF – *Open Shortest Path First*
  - OSI ISIS – *OSI Intermediate System to Intermediate System Routing*
- **Baseados em algoritmos DV:**
  - RIP – *Routing Information Protocol*
    (Existe em todos os sistemas operativos)
  - IGRP – *Interior Gateway Routing Protocol* (CISCO)
    (proprietário da CISCO)
  - EIGRP – *Extended IGRP* (CISCO)
    (durante muitos anos protocolo proprietário da CISCO, aberto em 2013, publicado como RFC7868 em maio de 2016)

Encaminhamento IP

A Internet...



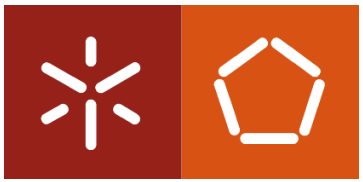
Encaminhamento IP
A Internet...



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!

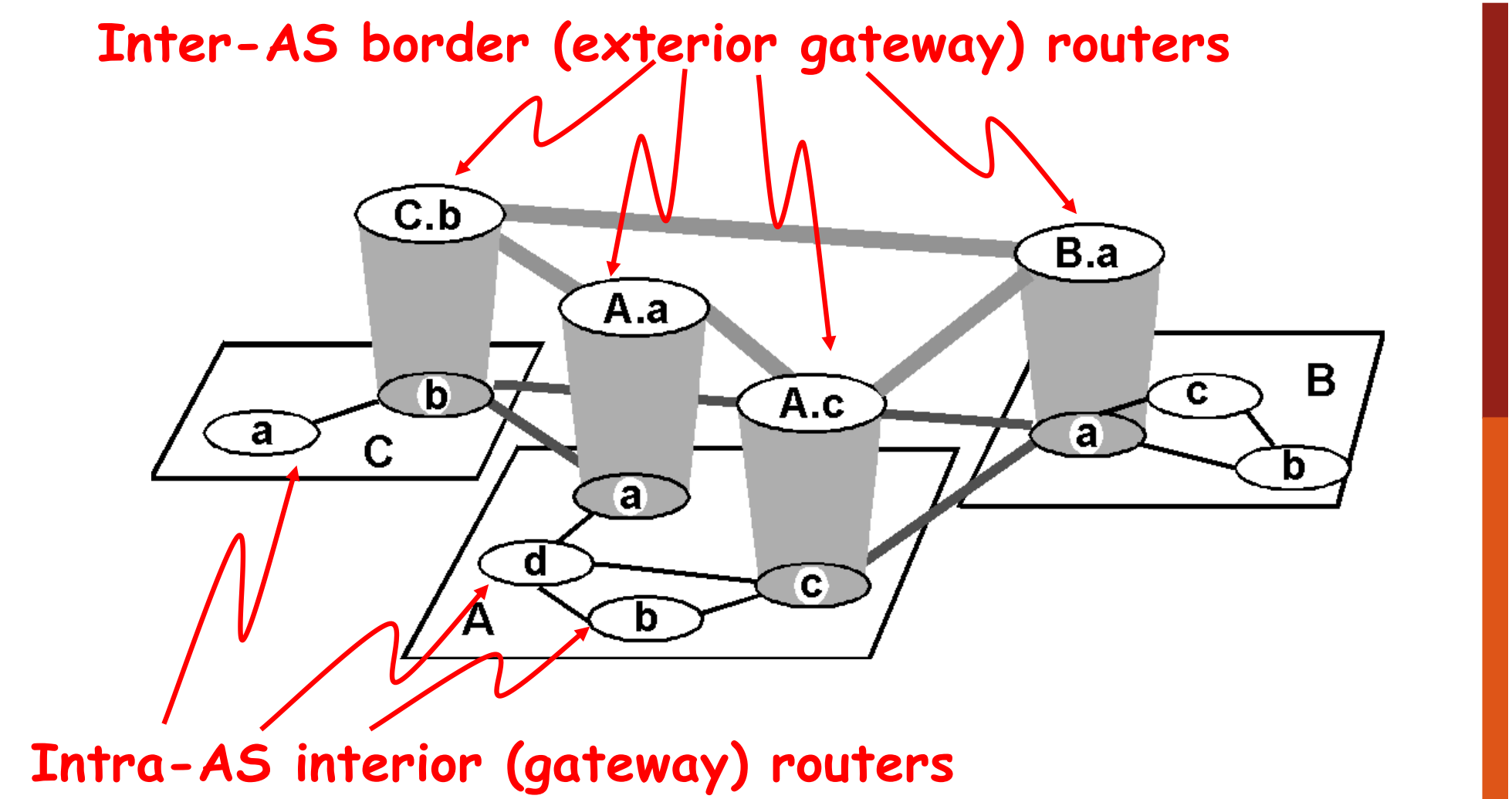
- O número de encaminhadores é demasiado grande pelo que a sobrecarga necessária (para o cálculo, armazenamento e comunicação da informação de encaminhamento) demasiado exigente;
- Idealmente, uma organização deveria poder escolher o algoritmo de encaminhamento que deseja utilizar nas suas redes.

Encaminhamento IP
Sistemas Autónomos



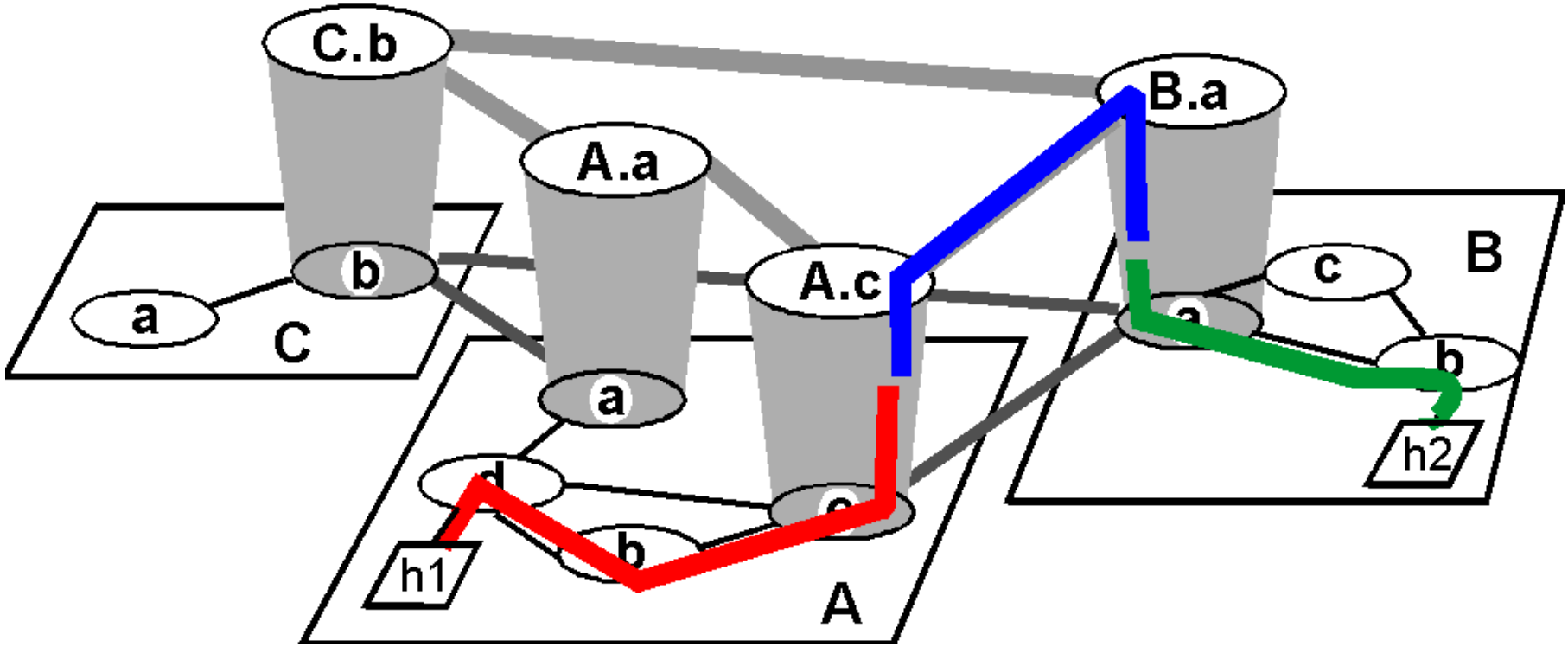
- Estes problemas são resolvidos agregando os encaminhadores em **Sistemas Autónomos** (*Autonomous Systems* – AS):
  - Os encaminhadores dentro de um mesmo AS utilizam todos o mesmo algoritmo de encaminhamento (LS ou DV) e possuem informação acerca de todos os encaminhadores que fazem parte do sistema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

Encaminhamento IP
Sistemas Autónomos



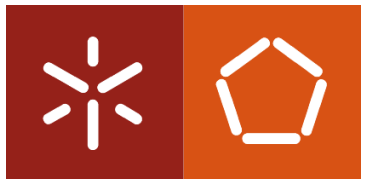
Fonte: *Computer Networking: A Top-Down Approach Featuring the Internet, J. Kurose, Addison-Wesley, 2001*

Encaminhamento IP
Sistemas Autónomos



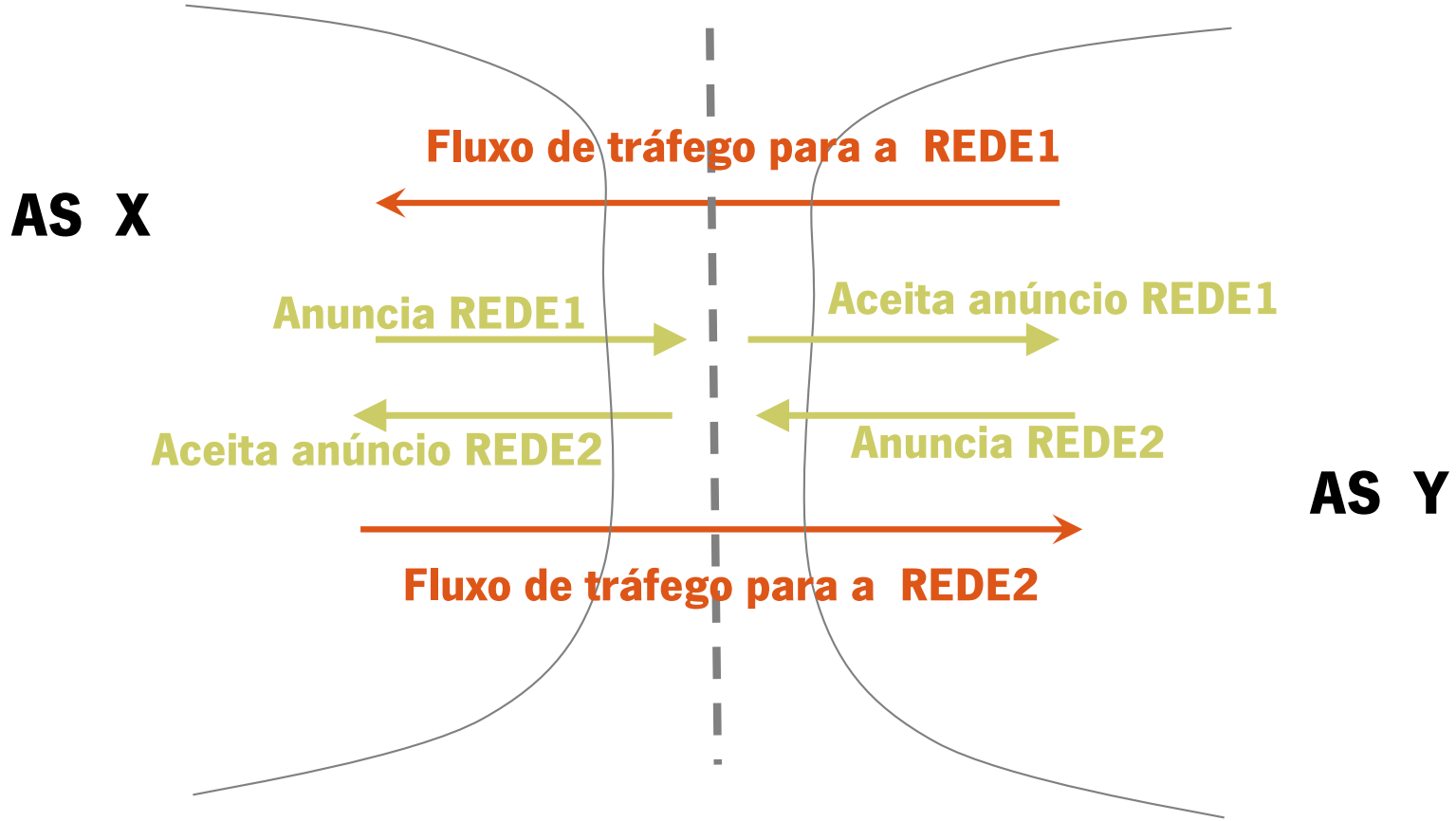
Fonte: *Computer Networking: A Top-Down Approach Featuring the Internet, J. Kurose, Addison-Wesley, 2001*

Encaminhamento IP
Sistemas Autónomos



- Identificação dos AS:
  - Os números de AS podem ser privados (AS 64512 até ao AS 65535) ou públicos (atribuídos pelo IANA, ou autoridades regionais, como o RIPE na Europa);
  - Essa identificação é usada nas trocas de informação de encaminhamento com os sistemas autónomos vizinhos.
- Os AS que fazem negócio com a conetividade também se designam por Provedores de Serviço Internet (*Internet Service Providers* – ISP):
  - Estabelecem acordos de parceria entre si (*peering agreements*), restringindo-se a troca de rotas se estão ao mesmo nível (em termos de fluxo de informação);
  - Se não estão ao mesmo nível, o de nível inferior (*downstream*) é cliente do serviço e o de nível superior (*upstream*) é o fornecedor do serviço.

Encaminhamento IP
Sistemas Autónomos



A comunicação entre a REDE1 e a REDE2 é possível se e só se:

(1) REDE1 anunciada por AS X, (2) anúncio aceite por AS Y,
(3) REDE2 anunciada por AS Y, (4) anúncio aceite por AS X

Encaminhamento IP
Tipos de Protocolos – IGP vs EGP

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).

LS: OSPF & ISIS
DV: RIP & EIGRP

Protocolos EGP

- As relações com os pares são previamente definidas e configuradas manualmente;
- A conetividade com redes externas é definida por políticas (divulgação e aceitação de rotas, preferências, etc.);
- Definem-se limites administrativos.

DV: BGP – *Border Gateway Protocol*

Encaminhamento IP
Tipos de Protocolos – IGP vs EGP

- Políticas:
  - No encaminhamento inter-domínio é fundamental ter o controle sobre a forma como o encaminhamento é efetuado. Por exemplo, a decisão de não encaminhar determinado tipo de tráfego através de um AS, tem de ser possível de espelhar na configuração operacional do encaminhamento.
  - 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 nestes dois níveis reduz o tamanho das tabelas de encaminhamento e a quantidade e tamanho das mensagens de atualização 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 pela importância da definição das políticas de encaminhamento.