FACULDADE DE CIÊNCIAS UNIVERSIDADE DO PORTO

U.PORTO

U.PORTO FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

Network Routing

Redes de Computadores

2021/22

Pedro Brandão

References

- These slides are from "Computer Networking: A Top Down Approach 5th edition. Jim Kurose, Keith Ross Addison-Wesley, April 2009"
- And "Computer Networks, 5th edition. Andrew S. Tanenbaum, David J. Wetherall, Prentice Hall, 2011"
- With adaptations/additions by Manuel Ricardo and Pedro Brandão

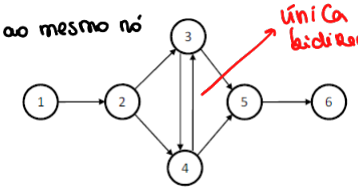
Driving questions…

- *What is a graph?*
- *What is a spanning tree?*
- *What is a shortest path tree?*
- *How are paths defined in a network?*
- *How does the Dijkstra algorithm work?*
- *How does a link state routing protocol work?*
- *How does a node learn about neighbours?*
- *How does the Bellman-Ford algorithm work?*
- *How does a distance vector work?*
- *What are the limitations of the layer 2 network of switches?*
- *How does the IEEE spanning tree protocol work?*
- *What is the maximum capacity of a flow network?*

Graph – Directed and Undirected

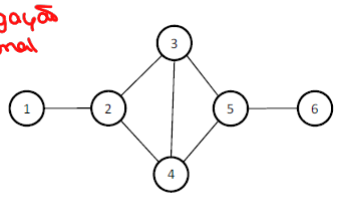
→ + do que 1 maneira de chegar ao mesmo nó

única ligação bidirecional



a) Directed graph

↳ Direção explícita



b) Undirected graph → podem ser bidirecionais

vértices

$G = (V, E)$
$V = \{v_1, v_2, v_3, v_4, v_5, v_6\}, \quad |V| = 6$
$E = \{(v_1, v_2), (v_2, v_3), (v_2, v_4), (v_3, v_4), (v_4, v_3), (v_3, v_5), (v_4, v_5), (v_5, v_6)\}, \quad |E| = 8$

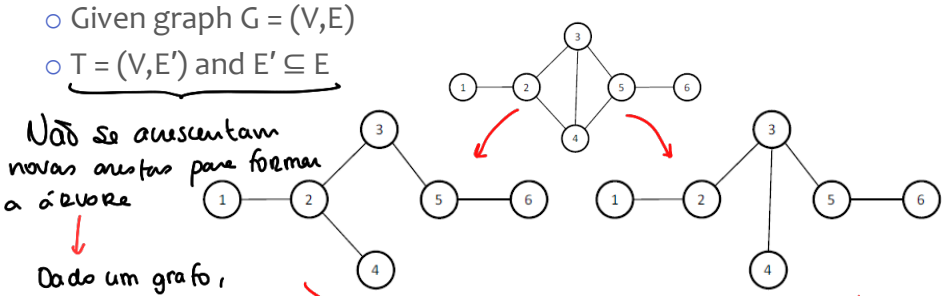
edges

$G = (V, E)$
$V = \{v_1, v_2, v_3, v_4, v_5, v_6\}, \quad |V| = 6$
$E = \{(v_1, v_2), (v_2, v_3), (v_2, v_4), (v_3, v_4), (v_3, v_5), (v_4, v_5), (v_5, v_6)\}, \quad |E| = 7$

Aparece +1 devido à bidirecionalidade de 3 e 4

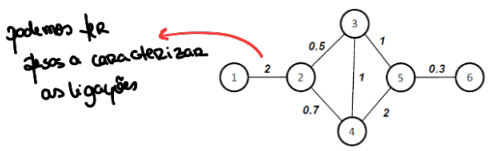
Tree

- Trees T = (V,E)
  - graph with **no cycles**
  - number of edges |E| = |V |−1
  - any two vertices of the tree are connected by exactly **one** path
- A tree T is said to span a graph (spanning tree) if:
  - Given graph G = (V,E)
  - T = (V,E′) and E′ ⊆ E



Shortest Path Trees

- Graphs and Trees can be weighted
  - $G = (V, E, w)$
  - $T = (V, E', w)$
  - $w: E \rightarrow \mathbb{R}$ — para cada aresta/ligação — temos um valor associado



- Total cost of a tree T ➔ $C_{total}(T) = \sum_{i=i}^{|E|} w(e_i)$
- Minimum Spanning Tree T^* ➔ $C_{total}(T^*) = \min(C_{total}(T))$ — árvore com um custo igual ao menor custo total
  - algorithms used to compute MST: Prism, Kruskal
- Shortest Path Tree (SPT) Rooted at Vertex s — a partir de s, temos uma árvore composta pelo caminho + curto entre s e todos os outros vértices do grafo
  - tree composed by the union of the shortest paths between s and each of other vertices of G
  - Algorithms used to compute SPT: Dijkstra, Bellman-Ford
- Computer networks use Shortest Path Trees

FACULDADE DE CIÊNCIAS UNIVERSIDADE DO PORTO

U.PORTO

U.PORTO FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

Routing in Layer 3 Networks

(Routing)

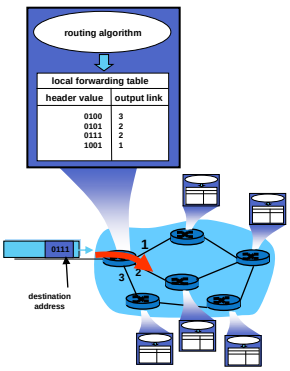
Forwarding, Routing

Pacote destinado X
Verificar tabela e Reencaminhar para outra seguinte

Construção de tabelas de forwarding. Pode implicar saber qual o nó seguinte + favorável ou conhecer todos os caminhos e, portanto, saber qual o + favorável

- Forwarding ➔ data plane
  - directing packet from input to output link
  - using a forwarding table
- Routing ➔ control plane
  - computing paths the packets will follow
  - routers exchange messages
  - each router creates its forwarding table

Vai sendo adaptada à medida que se vão recebendo novas informações sobre o que são as ligações na rede onde se encontram os routers



RCom 21/22 - Routing

U.PORTO

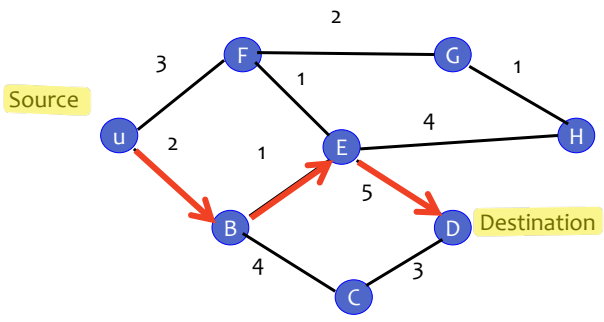
8

Importance of Routing

- End-to-end performance
  - path affects quality of service → escolha do caminho tem influência na qualidade relativamente à transmissão dos dados
  - delay, throughput, packet loss
- Use of network resources
  - balance traffic over routers and links → vai, indiretamente, implicar um melhoramento dos parâmetros a cima
  - avoiding congestion by directing traffic to less-loaded links
- Transient disruptions
  - failures, maintenance
  - limiting packet loss and delay during changes → caso ocorram falhas é necessário existir uma rápida adaptação das rotas para outra rota redundante

Shortest-Path Routing

- Path-selection model
  - Destination-based
  - Load-insensitive (e.g., static link weights) → Vão tem adaptação ao que ocorre a largura de banda disponível, delay associado, perdas a acontecer, etc.
  - Minimum hop count or minimum sum of link weights



os outros têm todos um custo maior

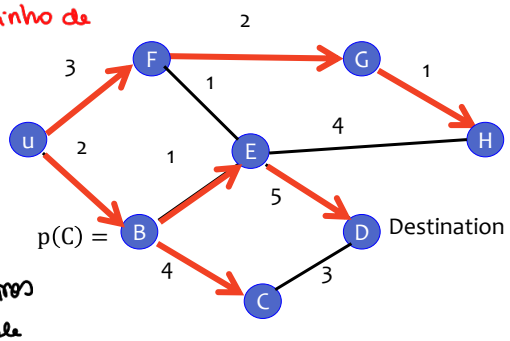
Shortest path: 2+1+5=8

Shortest-Path Routing

- Given a network topology with link costs
  - c(x, y) - link cost from node x to node y
  - Infinity (∞) if x and y are not direct neighbours

- Compute the least-cost paths from source **u** to all nodes
  - p(v) - node predecessor of node v in the path to u

predecessor no caminho de menor custo



Inicialmente, o nó u tem 2 ligações para os vizinhos que custam 3 e 2, para todos os outros tem um custo ∞, pois ele não sabe nenhuma rota ainda

p(C) = B
p(E) = B
p(G) = F
p(H) = G
p(E) = B
p(D) = E

RCom 21/22 - Routing

U.PORTO

11

Dijkstra's Shortest-Path Algorithm

- Iterative algorithm
  - After k iterations ➔ known least-cost paths to k nodes

- **S** ➔ set of nodes for which least-cost path is known
  - Initially, **S={u}**, where **u** is the source node
  - Add one node to **S** in each iteration

- **D(v)** ➔ current cost of path from source to node **v**
  - Initially
    - **D(v)=c(u, v)** for all nodes **v** adjacent to **u**
    - **D(v)= ∞** for all other nodes **v**
  - Continually update **D(v)** when shorter paths are learned

inicialmente só conhecemos D dos nós vizinhos

independentemente de estar em S ou ser só uma estimativa

são as ligações que o nó tem e conhece

Conforme se vão adicionando nós a S

RCom 21/22 - Routing

U.PORTO

12

Dijsktra's Algorithm

```
1  Initialization:
2   S = {u}
3   for all nodes v
4    if v adjacent to u {
5      D(v) = c(u,v)
6    }
7    else D(v) = ∞

8   Loop
9    find node w not in S with the smallest D(w)
10     add w to S
11     update D(v) for all v adjacent to w and not in S:
12        D(v) = min{D(v), D(w) + c(w,v)}
13  until all nodes in S
```

Inicialmente vamos usar o vizinho com menos peso na ligação

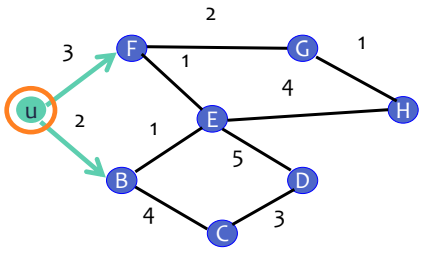
mínimo entre
. o valor que já lá está
. Caminho + curto de u a w + c(w,v)

Dijkstra's Algorithm – Example – 1

S={u} → nó u adicionado a S

	D(S)
u	0
B	2
C	∞
D	∞
E	∞
F	3
G	∞
H	∞

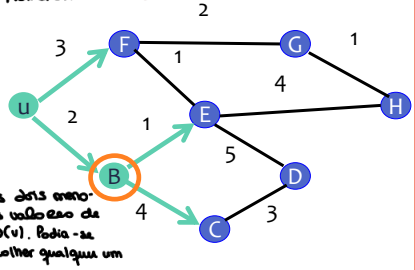
Não são vizinhos de u



C(u,B) < C(u,F)

S={u,B} → Adiciona-se B a S

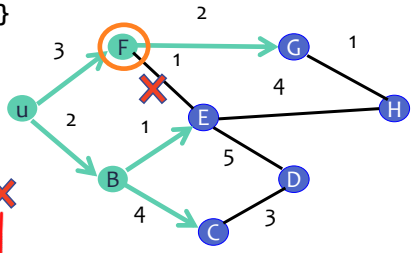
	D(S)
u	0
B	2
C	6
D	∞
E	3
F	3
G	∞
H	∞



dos dois menores valores de D(v). Podia-se escolher qualquer um

S={u,B,F}

	D(S)
u	0
B	2
C	6
D	∞
E	3
F	3
G	5
H	∞



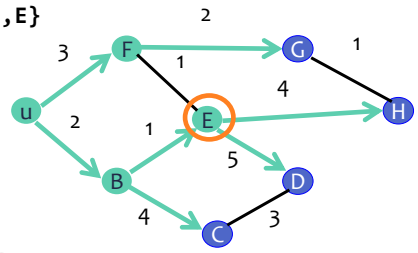
não há nenhuma melhoria associada, o mínimo era o que já conhecíamos

S={u,B,F,E}

	D(S)
u	0
B	2
C	6
D	8
E	3
F	3
G	5
H	7

Próximo

vai ser atualizado ao iterar sobre G

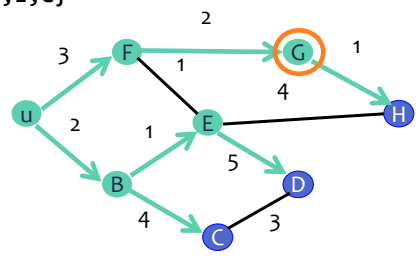


Dijkstra's Algorithm – Example – 2

não introduz nenhuma entrada

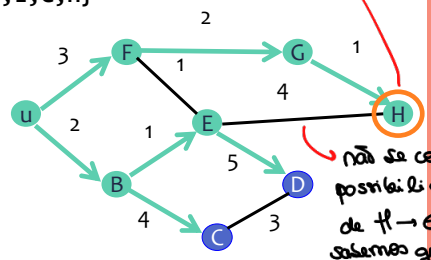
S={u,B,F,E,G}

	D(S)
u	0
B	2
C	6
D	8
E	3
F	3
G	5
H	6



S={u,B,F,E,G,H}

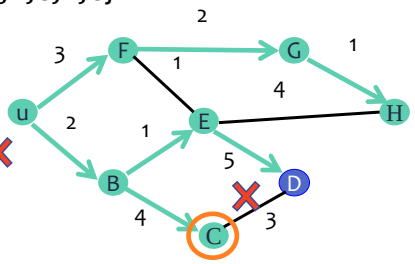
	D(S)
u	0
B	2
C	6
D	8
E	3
F	3
G	5
H	6



não se considera a possibilidade da ligação de H→E, porque já sabemos qual o shortest path até E, e só vamos iterar sobre os que não sabemos (não estão em S)

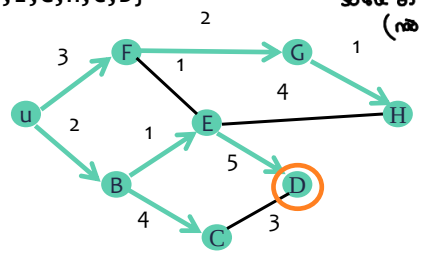
S={u,B,F,E,G,H,C}

	D(S)
u	0
B	2
C	6
D	8
E	3
F	3
G	5
H	6



S={u,B,F,E,G,H,C,D}

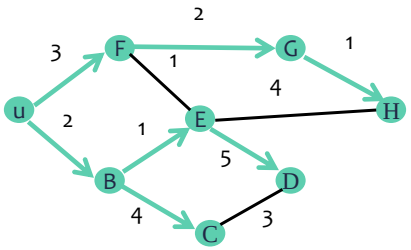
	D(S)
u	0
B	2
C	6
D	8
E	3
F	3
G	5
H	6



Dijkstra's Algorithm – Example – 3

	D(S)	Next Hop
u	0	---
B	2	B
C	6	B
D	8	B
E	3	B
F	3	F
G	5	F
H	6	G

Forwarding table at u

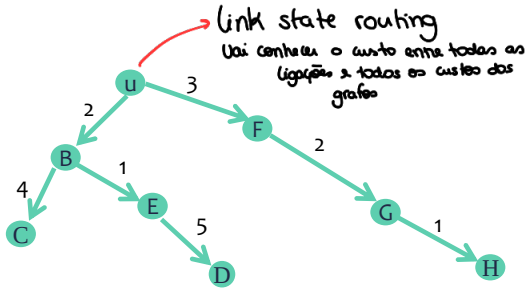


Shortest-path tree from u

Dijkstra's Algorithm – Example – 4

	D(S)	Next Hop
u	0	---
B	2	B
C	6	B
D	8	B
E	3	B
F	3	F
G	5	F
H	6	G

Forwarding table at u



Shortest-path tree from u

Link-State Routing

- Each router keeps track of its incident links
  - link up, link down
  - cost on the link → custo dinâmico em vez de estático
- Each router broadcasts link state
  - every router gets a complete view of the graph
- Each router runs Dijkstra's algorithm, to
  - compute the shortest paths
  - construct the forwarding table
- Example protocols
  - Open Shortest Path First (OSPF), STD 54, RFC 2328, OSPF Version 2
  - Intermediate System – Intermediate System (IS-IS)
    - ISO/IEC 10589:2002, Information technology — Telecommunications and information exchange between systems — Intermediate System to Intermediate System intra-domain routeing information exchange protocol for use in conjunction with the protocol for providing the connectionless-mode network service (ISO 8473)
    - RFC 1195, Use of OSI IS-IS for routing in TCP/IP and dual environments

- Quando ocorre algum evento que possa ter influência nas rotas das tabelas de reencaminhamento, é preciso avisar todos os nós da rede (avisam os vizinhos que avisam os vizinhos, etc) para que sejam conhecidas as alterações em toda a rede

Detection of Topology Changes

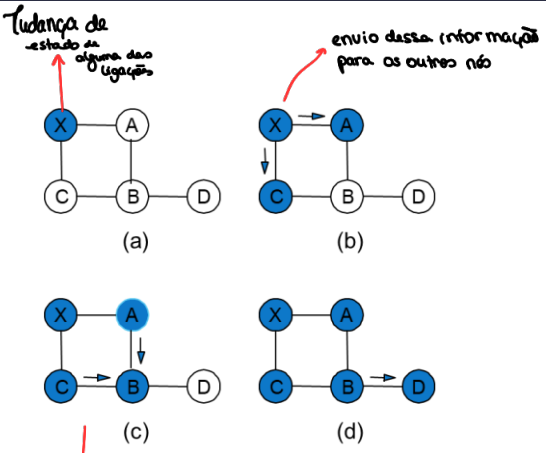
- Beacons generated by routers on links
  - Periodic "hello" messages in both directions
  - few missed "hellos" ➔ link failure

- São enviadas mensagens periodicamente, para que se consiga detetar falhas em determinadas ligações

Broadcasting the Link State

- How to Flood the link state?
  - every node sends link-state information through adjacent links
  - next nodes forward that info to all links
    - except the one where the information arrived
- When to initiate flooding?
  - Topology change
    - link or node failure/recovery
    - link cost change
  - Periodically
    - refresh link-state information
    - typically 30 minutes

→ mesmo que não hajam mudanças



envio controlado para não ser devolvida a informação para quem enviou aquele nó:
nó recebe de A e C, mas só envia para D

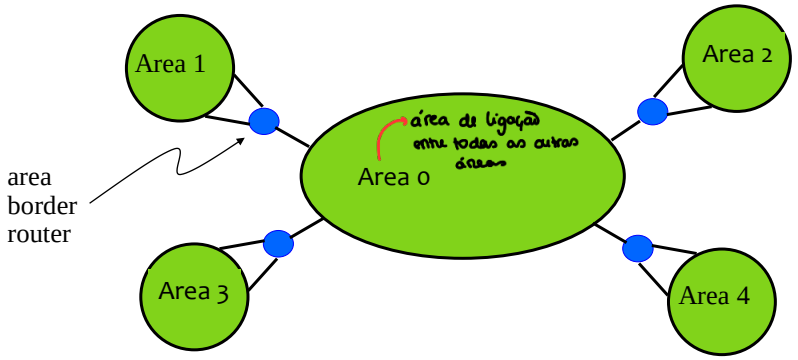
→ À medida que aumentamos o tamanho da rede, vai haver muita informação a ser transmitida periodicamente ou em caso de mudança, por toda a nossa rede

RCom 21/22 - Routing

U.PORTO

Scaling Link-State Routing

- Overhead of link-state routing
  - flooding link-state packets throughout the network
  - running Dijkstra's shortest-path algorithm
- Introducing hierarchy through "areas"



21

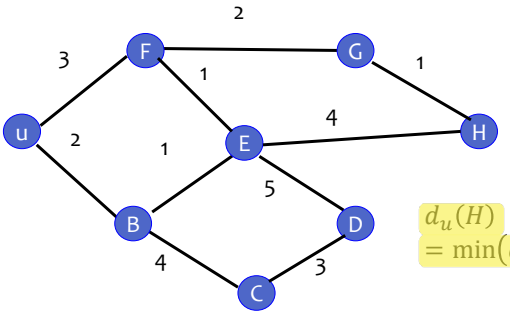
Criam-se áreas, onde os routers estão nas fronteiras e acumulam o conhecimento de um lado e do outro, mas só transmitem resumos, ou a possibilidade de chegar a determinadas redes a partir deles, e não todo o estado associado a internamente a cada uma das áreas (como os custos associados a todos os links dessas áreas)

RCom 21/22 - Routing

U.PORTO

Bellman-Ford Algorithm

- Define distances at each node x
  - $d_x(y)$ = cost of least-cost path from x to y
- Update distances based on neighbours
  - $d_x(y) = \min\big(c(x,n) + d_n(y)\big)$ over all neighbours n of x



- não precisamos de saber o percurso, apenas precisamos de saber que há uma distância entre o nosso vizinho e o valor que queremos colocar na nossa tabela.
- Distância até esse ponto:
  Dist até ao vizinho + Dist vizinho até esse nó

$d_u(H) = \min\big(c(u,F) + d_F(H), c(u,B) + d_B(H)\big)$

3 3 2 5

6 7

22

Distance Vector Algorithm

- c(x,n) = cost for direct link from x to n
  - node x maintains costs of direct links c(x,n)
  - n are x's neighbours
- $D_x(y)$ = estimate of least cost from x to y
  - node x maintains **distance vector** $\mathbf{D}_x = [D_x(y): y \in V]$
  - V is all nodes

↳ Vetor que o nó x mantém sobre os custos para chegar a todos os outros nós

- Node x maintains also its neighbours' distance vectors
  - for each neighbour n, x maintains $\mathbf{D}_n = [D_n(y): y \in V]$

↳ para poder isto atualizar à medida que os vizinhos vão anunciando os custos para chegar aos outros nós

- Each node x periodically sends D_x to its neighbours
  - and neighbours update their own distance vectors
  - $D_z(y) \leftarrow \min_x\{c(z,x) + D_x(y)\}$ for each node $y \in V$
    - z is a neighbour of x

↳ Eu consigo chegar a estes nós com este custo

- Over time, the distance vector D_x converges

Distance Vector Algorithm

A cada iteração podemos alterar vetor D

- Iterative, asynchronous
  - each local iteration caused by:
    - local link cost change
    - distance vector update message from neighbour

Cada nó calcula custo para cada outro nó

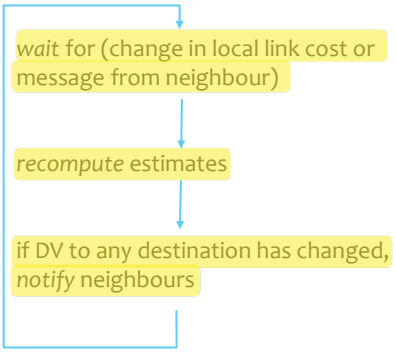
↳ Não importa o caminho, só o custo final e para onde tem de reencaminhar a seguir

- Distributed
  - node notifies neighbours
    - only when its DV changes

- Neighbours then
  - notify their neighbours,
  - if necessary

↳ se utilizavam aquela ligação para chegar ao nó destino

- Each node:



Distance Vector Example - Step 0

* Ainda não há valores, pois não conhecemos

Os que não são vizinhos têm ∞

Distância para todos os outros nós

Table for A		
Dst	Cst	Hop
A	0	A
B	4	B
C	∞	–
D	∞	–
E	2	E
F	6	F

Table for B		
Dst	Cst	Hop
A	4	A
B	0	B
C	∞	–
D	3	D
E	∞	–
F	1	F

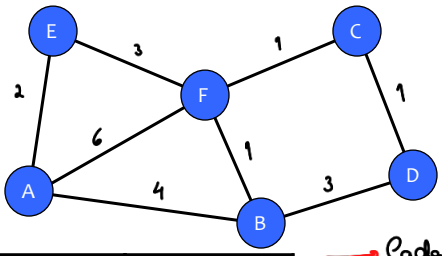


Table for C		
Dst	Cst	Hop
A	∞	–
B	∞	–
C	0	C
D	1	D
E	∞	–
F	1	F

Table for D		
Dst	Cst	Hop
A	∞	–
B	3	B
C	1	C
D	0	D
E	∞	–
F	∞	–

Table for E		
Dst	Cst	Hop
A	2	A
B	∞	–
C	∞	–
D	∞	–
E	0	E
F	3	F

Table for F		
Dst	Cst	Hop
A	6	A
B	1	B
C	1	C
D	∞	–
E	3	E
F	0	F

→ Cada um dos nós envia informações para os vizinhos até onde é que consegue chegar e com que custos associados

D(A,E) = D(E,A) no caso de um grafo não direcionado

Um grafo direcionado pode ter custos diferentes

Distance Vector Example - Step 1

D(A,F) = 5 ⇝ A → E → F

Table for A		
Dst	Cst	Hop
A	0	A
B	4	B
C	7	F
D	7	B
E	2	E
F	5	E

Table for B		
Dst	Cst	Hop
A	4	A
B	0	B
C	2	F
D	3	D
E	4	F
F	1	F

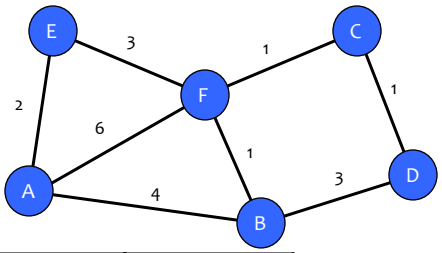


Table for C		
Dst	Cst	Hop
A	7	F
B	2	F
C	0	C
D	1	D
E	4	F
F	1	F

Table for D		
Dst	Cst	Hop
A	7	B
B	3	B
C	1	C
D	0	D
E	∞	–
F	2	C

Table for E		
Dst	Cst	Hop
A	2	A
B	4	F
C	4	F
D	∞	–
E	0	E
F	3	F

Table for F		
Dst	Cst	Hop
A	5	B
B	1	B
C	1	C
D	2	C
E	3	E
F	0	F

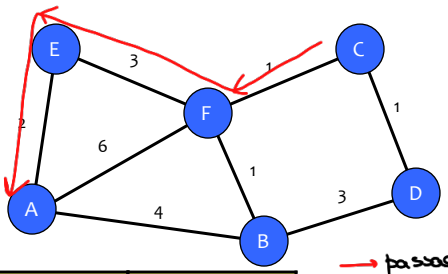
↳ Alterações que cada nó fez ao seu vetor de distâncias

→ A alterou a sua rota para chegar a F, mas para chegar até C o valor que utilizou é o valor antigo que tinha para chegar a F

Distance Vector Example - Step 2

Table for A		
Dst	Cst	Hop
A	0	A
B	4	B
C	6	E
D	7	B
E	2	E
F	5	E

Table for B		
Dst	Cst	Hop
A	4	A
B	0	B
C	2	F
D	3	D
E	4	F
F	1	F



→ passadas algumas iterações os valores estabilizam até que ocorram mudanças novamente

Table for C		
Dst	Cst	Hop
A	6	F
B	2	F
C	0	C
D	1	D
E	4	F
F	1	F

Table for D		
Dst	Cst	Hop
A	7	B
B	3	B
C	1	C
D	0	D
E	5	C
F	2	C

Table for E		
Dst	Cst	Hop
A	2	A
B	4	F
C	4	F
D	5	F
E	0	E
F	3	F

Table for F		
Dst	Cst	Hop
A	5	B
B	1	B
C	1	C
D	2	C
E	3	E
F	0	F

é na mesma através de F, mas o caminho depois de F é alterado

Altera custos → Custo para chegar a C vai ser recalculado

Propaga a informação

Routing Information Protocol (RIP)

- STD 56, RFC 2453, RIP Version 2
- Distance vector protocol
  - nodes send distance vectors every 30 seconds
  - or, when an update causes a change in routing
- RIP is limited to small networks

limitado a pequenas redes Visto que há uma atualização um pouco mais constante

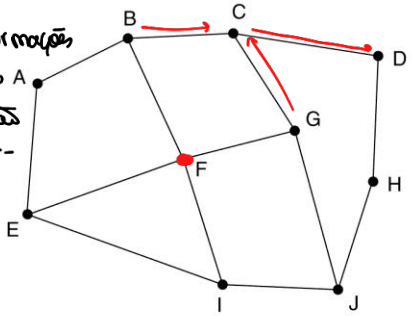
RCom 21/22 - Routing

U.PORTO

BGP – The Exterior Gateway Routing Protocol

↳ Numa Rede interna vamos ter interior routing protocols

· Vão agregar as informações que possuem das redes interras, portanto não passa toda a informação do que se está a passar. Só um sumário.



(a)

A set of BGP routers.

Information F receives from its neighbors about D

From B: "I use BCD"
From G: "I use GCD"
From I: "I use IFGCD"
From E: "I use EFGCD"

Vai receber a informação de como é que os outros chegam a D

(b)

Information sent to F

* B diz que passa pela Rede B → rede C → rede D

→ Em muitos dos casos não é só o preço que vai ser tido em consideração, mas também em contas políticas de propagação.

→ Só G consegue chegar a D através de J ou C. Anuncia só 1. Pode ter algum acordo com C que lhe fique + barato chegar até D. Ele não anuncia que chega a J. Assim os outros nós não o vão utilizar como rede de passagem até J.
↳ Depende dos acordos que existam

29

FACULDADE DE CIÊNCIAS UNIVERSIDADE DO PORTO · U.PORTO · U.PORTO FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

Unique Spanning Tree in Ethernet Networks

Routing

TTL → Impede que pacote fique perdido na rede. Ao chegar a 0 é descartado.
↳ Vai sempre diminuindo cada vez que o router encaminha o pacote

L2 Networking - Single Tree Required

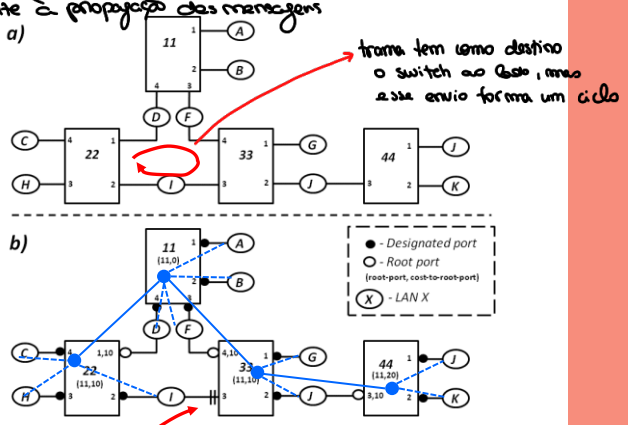
havendo ciclos

- Ethernet frame
  - No hop-count
  - Could loop forever
  - broadcast frame, mis-configuration

No caso da ethernet ao passar pelos switches, como não há nenhum controlo podia acontecer de uma trama ficar eternamente na rede
↳ relativamente à propagação das mensagens

- Layer 2 network
  - Required to have tree topology
  - Single path between every pair of stations

- Spanning Tree Protocol (STP)
  - Running in bridges
  - Helps building the spanning tree
  - Blocks ports



trama tem como destino o switch ao lado, mas esse envio forma um ciclo

Possíveis portas que levariam a loops

Relativamente ao envio de determinadas tramas

RCom 21/22 - Routing

U.PORTO

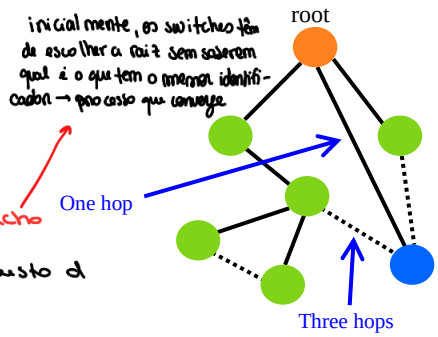
31

Constructing a Spanning Tree

- Distributed algorithm
  - switches need to elect a "root"
    - the switch with the smallest identifier
  - each switch identifies if its interface is on the shortest path from the root
  - messages (Y, d, X)
    - from node X
    - claiming Y is the root
    - and the distance is d

Há um envio de mensagens:
- consigo chegar à raiz que acho que é Y desde de X com o custo d

inicialmente, os switches têm de escolher a raiz sem saberem qual é o que tem o menor identificador → processo que converge



RCom 21/22 - Routing

U.PORTO

32

Steps in Spanning Tree Algorithm

- Initially, each switch thinks it is the root
  - switch sends a message out every interface
  - identifying itself as the root with distance 0
  - example: switch X announces (X, 0, X)

X é um valor numérico: root ID

consegue chegar a X até X com ds 0

- Other switches update their view of the root
  - upon receiving a message, check the root id
  - if the new id is smaller, start viewing that switch as the root

Atualizam raiz e distância

Apercebem-se de quem é a Raiz

- Switches compute their distance from the root
  - add 1 to the distance received from a neighbour
  - identify interfaces not on a shortest path to the root and exclude them from the spanning tree

à medida que vão recebendo informações vão atualizando

Example - Switch #4's Viewpoint

- Switch #4 thinks it is the root
  - sends (4, 0, 4) message to 2 and 7

Suas interfaces

- Then, switch #4 hears from #2
  - receives (2, 0, 2) message from 2
  - … and thinks that #2 is the root
  - and realizes it is just one hop away

2 < 4

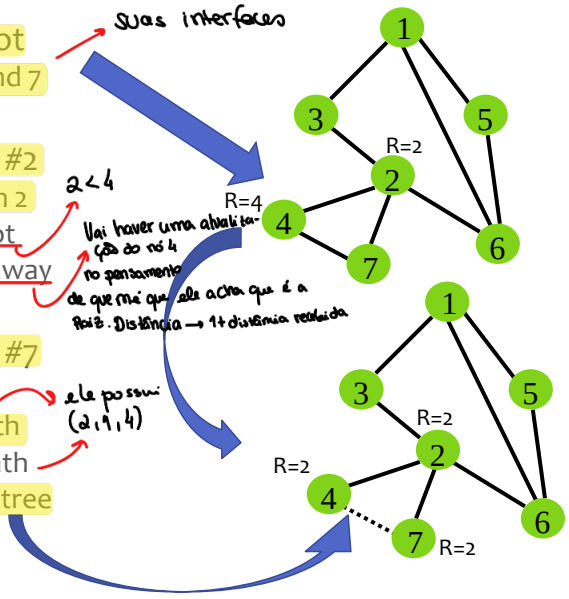
Vai haver uma atualização do nó 4 no pensamento de que ele acha que é a Raiz. Distância → 1 + distância recebida

Após atualização do nó 4

- Then, switch #4 hears from #7
  - receives (2, 1, 7) from 7
  - and realizes this is a longer path
  - so, prefers its own one-hop path
  - and removes 4-7 link from the tree

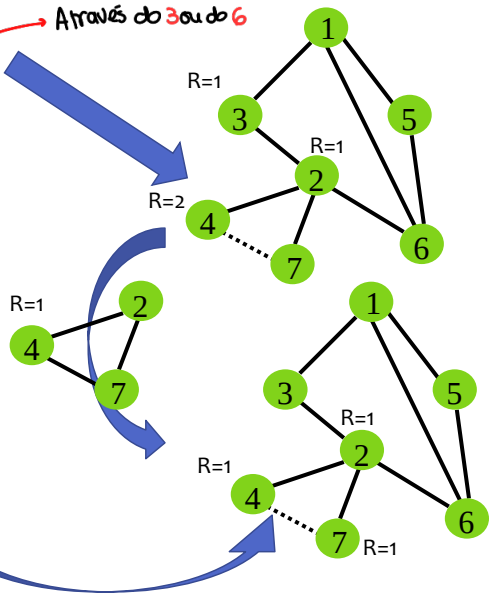
ele possui (2, 1, 4)

Não necessita desse caminho para chegar a 2



Example - Switch #4's Viewpoint

- Switch #2 hears about switch #1
  - switch 2 hears (1, 1, 3) from 3
  - switch 2 starts treating 1 as root
  - and sends (1, 2, 2) to neighbours
- Switch #4 hears from switch #2
  - switch 4 starts treating 1 as root
  - and sends (1, 3, 4) to neighbours
- Switch #4 hears from switch #7
  - switch 4 receives (1, 3, 7) from 7
  - and realizes this is a longer path
  - so, prefers its own three-hop path
  - and removes 4-7 link from the tree



FACULDADE DE CIÊNCIAS UNIVERSIDADE DO PORTO
U.PORTO
U.PORTO FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

Maximum Flow of a Network

Routing

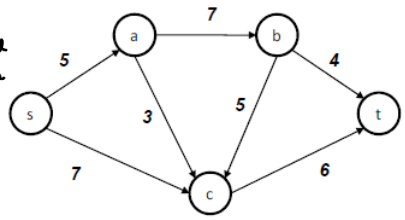
RCom 21/22 - Routing

U.PORTO

Flow Network Model

- Flow network
  - source s
  - sink t
  - nodes a, b and c
- Edges are labelled with capacities
  - (e.g. bit/s)

largura de banda



- Communication networks are not flow networks
  - they are queue networks
  - flow networks enable to determine limit capacity values

37

RCom 21/22 - Routing

U.PORTO

Maximum Capacity of a Flow Network

- Max-flow min-cut theorem
  - maximum amount of flow transferable through a network
  - equals minimum value among all simple cuts of the network
- Cut ➔ split of the nodes V into two disjoint sets S and T
  - $S \cup T = V$ → no momento da divisão não podemos excluir nós
  - there are $2^{|V|-2}$ possible cuts
- Capacity of cut (S, T):

$$c(S,T) = \sum_{(u,v)|u\in S, v\in T, (u,v)\in E} c(u,v)$$

tem de existir uma aresta a ligar os nós

u num lado da divisão
v no outro lado da divisão

→ Os nós do mesmo cut têm de ter ligação entre si

38

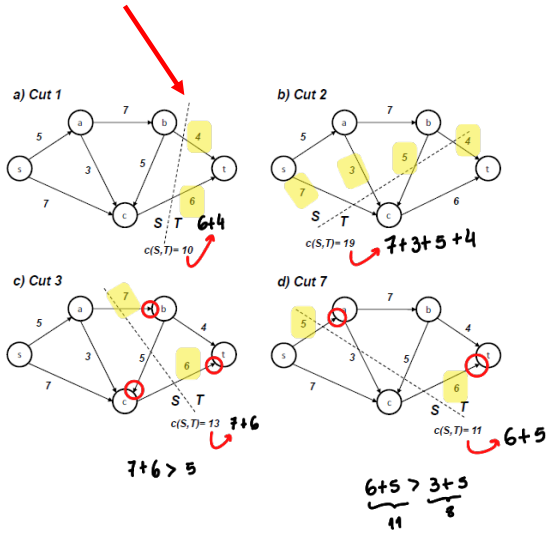
Max-flow Min-cut - Example

todas as possibilidades

$2^{|5|-2} = 8$ possible cuts

Cut	Vertices s	a	b	c	t	$c(S,T)$	Feasability
1	S	S	S	S	T	**10**	✓
2	S	S	S	T	T	19	✓
3	S	S	T	S	T	13	✓
4	S	S	T	T	T	17	✓
5	S	T	S	S	T	-	×
6	S	T	S	T	T	-	×
7	S	T	T	S	T	11	✓
8	S	T	T	T	T	12	✓

Maximum flow = 10



Cut 5

se b não tem ligações

Impossível

Cut 6

se b não tem ligações

Homework

1. Review slides

2. Read from Tanenbaum
   - Section 5.2 – Routing algorithms
   - Section 4.8.3 Spanning Tree Bridges

3. Answer questions at Moodle

FACULDADE DE CIÊNCIAS UNIVERSIDADE DO PORTO

U.PORTO

U.PORTO FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

End of Routing