Camada de Rede

Roteamento

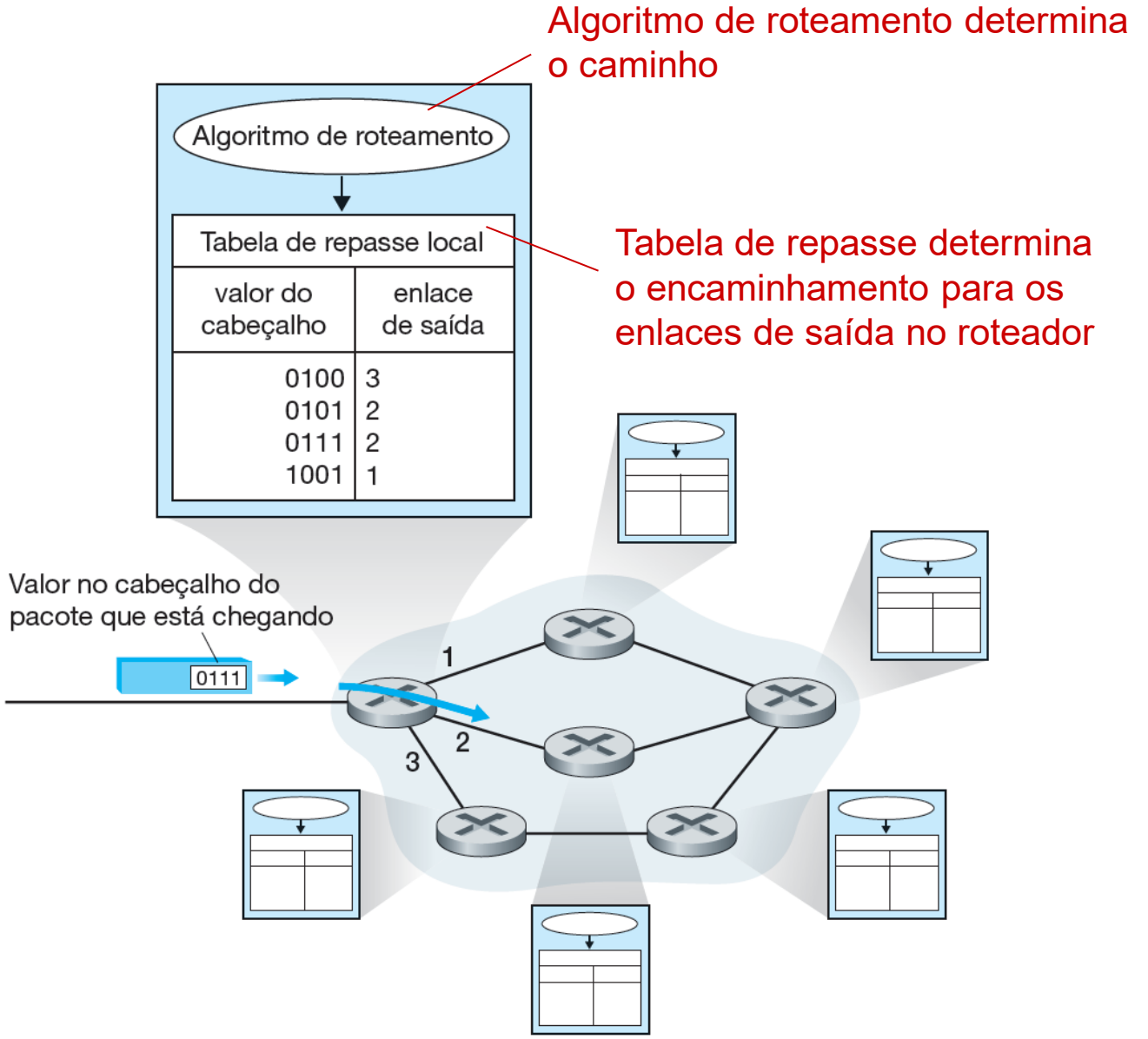
DCA0130 – Redes de Computadores

Prof. Carlos M. D. Viegas

Departamento de Engenharia de Computação e Automação
Universidade Federal do Rio Grande do Norte

Camada de Rede

- Principais funções do roteador
  - Encaminhamento/repasse:
    - Move os pacotes de um enlace de entrada para um enlace de saída no roteador
  - Roteamento:
    - Determina a rota/caminho a ser seguida/o pelos pacotes
      - Algoritmos de roteamento
      - Mantém informações de roteamento para outras redes
  - Não guardam estado sobre conexões fim a fim
    - Não existe o conceito de "conexão" na camada de rede



Introdução aos Algoritmos de Roteamento

- Os algoritmos de roteamento são executados nos roteadores
  - Determinam qual a interface de saída deve ser utilizada para a transmissão de um pacote
  - Existem diversas métricas para determinar o melhor caminho:
    - Menor número de hops
    - Menor atraso
    - Menor taxa de perda de pacotes
    - Largura de banda disponível
  - O processo de roteamento consiste no preenchimento e atualização das tabelas que contêm as informações de custo para cada enlace de saída
    - Os algoritmos de roteamento são responsáveis por obter essas informações
    - É importante destacar que um roteador não armazena o caminho completo em suas tabelas
  - O encaminhamento/repasse é o processo que trata a chegada dos pacotes, consultando a tabela de roteamento e direcionando-os para a interface de saída (definida previamente pelo algoritmo roteamento)

Introdução aos Algoritmos de Roteamento

- As rotas podem ser:
  - Estáticas
    - As rotas são definidas manualmente (fixas)
    - Útil quando a escolha de rotas fixas é desejável
    - Não responde bem em caso de falhas
  - Dinâmicas
    - Alteram as decisões de roteamento automaticamente para refletir mudanças de topologia ou de tráfego na rede
      - Baseadas em métricas, por exemplo: menor atraso, menor taxa de perda de pacotes, etc.
    - Os algoritmos de roteamento podem ser:
      - Baseados em tabelas (proativos): o algoritmo de roteamento constrói uma tabela que é atualizada periodicamente
      - Sob demanda (reativos): o algoritmo de roteamento só toma a decisão quando requisitado (ou seja, sob demanda)

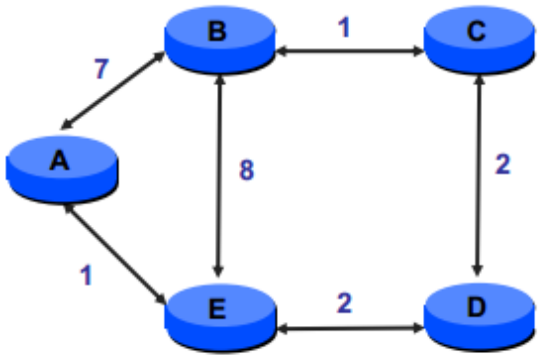
Algoritmos de Roteamento

- Roteamento por Vetor de Distâncias (DV)
  - Conhecido como algoritmo de Bellman-Ford
  - Os algoritmos de roteamento DV são sob-demanda (reativos)
  - Cada roteador mantém uma tabela contendo a menor distância até cada destino e qual interface de saída deve ser utilizada (distribuído)
  - As tabelas são atualizadas com base na troca de informações com seus vizinhos (iterativo)
  - Não é necessário que todos os roteadores executem o roteamento simultaneamente (assíncrono)
  - O algoritmo RIP (*Routing Information Protocol*) é um exemplo de DV

  - Ideia básica do vetor de distâncias:
    - Cada nó envia periodicamente sua estimativa de vetor de distância (DV) aos vizinhos
    - Quando um nó X recebe uma nova estimativa DV do vizinho, ele atualiza seu próprio DV usando a equação de Bellman-Ford
    - Os DV's são enviados para todos os vizinhos até a convergência!

Algoritmos de Roteamento

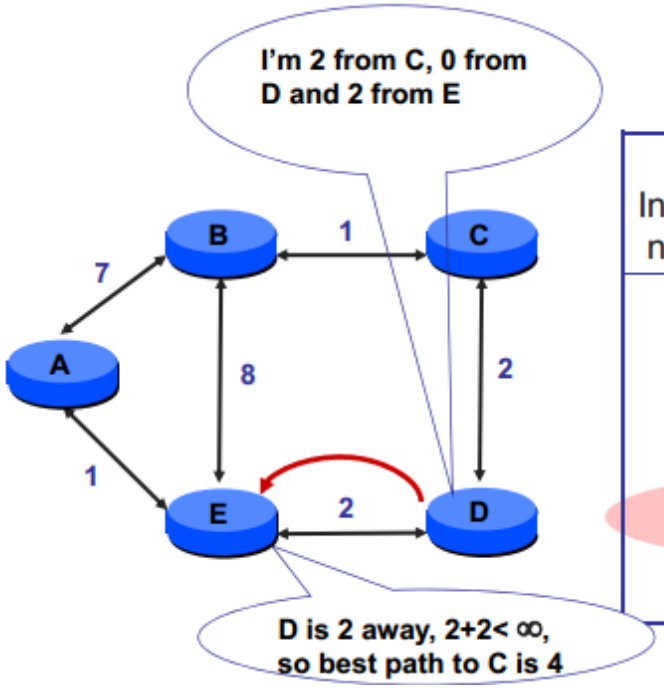
- Roteamento por Vetor de Distâncias - Exemplo
  - Estado inicial



Info at node	Distance to Node				
	A	B	C	D	E
A	0	7	∞	∞	1
B	7	0	1	∞	8
C	∞	1	0	2	∞
D	∞	∞	2	0	2
E	1	8	∞	2	0

Algoritmos de Roteamento

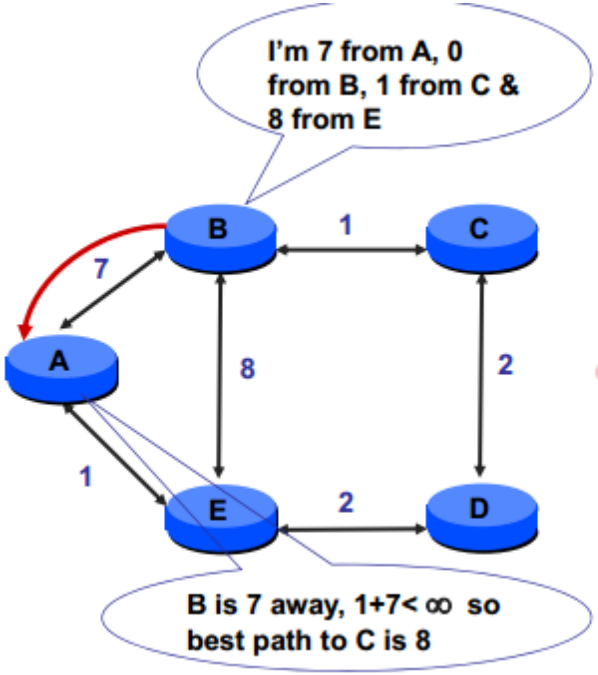
- Roteamento por Vetor de Distâncias - Exemplo
  - D envia vetor para E



Info at node	Distance to Node A	B	C	D	E
A	0	7	∞	∞	1
B	7	0	1	∞	8
C	∞	1	0	2	∞
D	∞	∞	2	0	2
E	1	8	**4**	2	0

Algoritmos de Roteamento

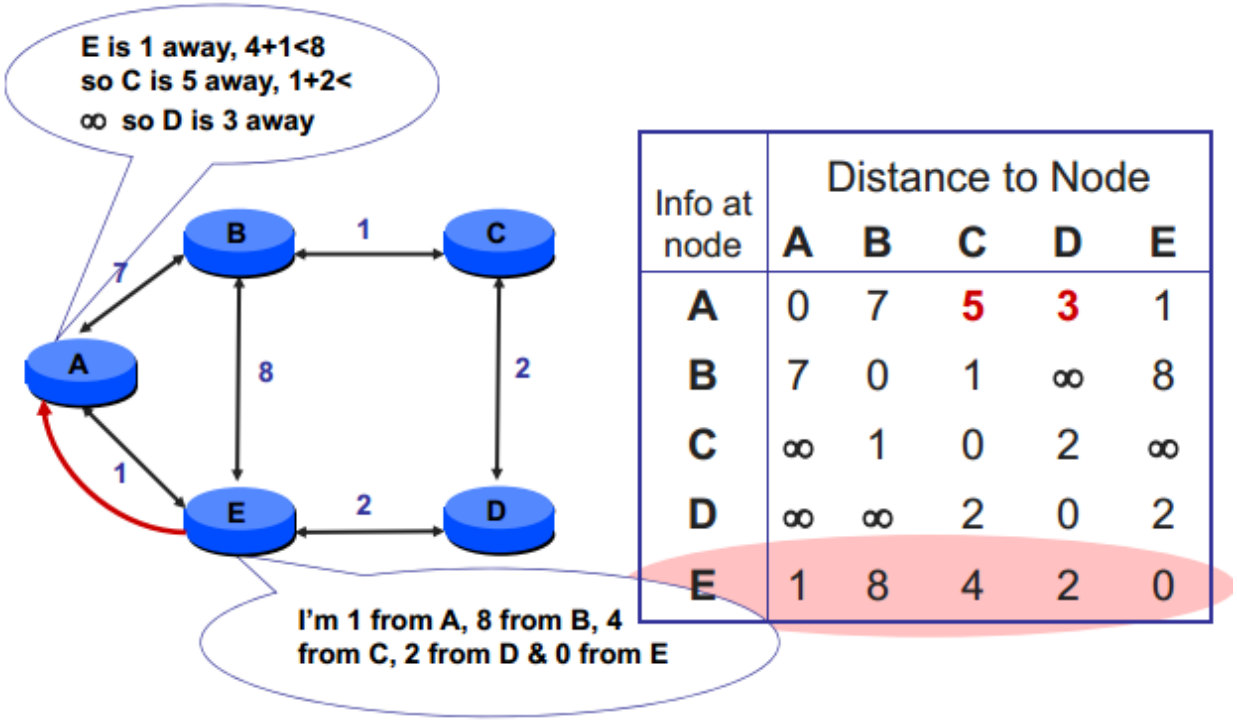
- Roteamento por Vetor de Distâncias - Exemplo
  - B envia vetor para A



Info at node	Distance to Node A	B	C	D	E
A	0	7	8	∞	1
B	7	0	1	∞	8
C	∞	1	0	2	∞
D	∞	∞	2	0	2
E	1	8	4	2	0

Algoritmos de Roteamento

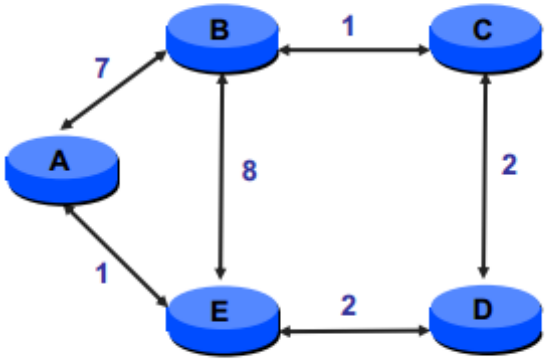
- Roteamento por Vetor de Distâncias - Exemplo
  - E envia vetor para A



Info at node	Distance to Node A	B	C	D	E
A	0	7	5	3	1
B	7	0	1	∞	8
C	∞	1	0	2	∞
D	∞	∞	2	0	2
E	1	8	4	2	0

Algoritmos de Roteamento

- Roteamento por Vetor de Distâncias - Exemplo
  - Até a convergência...



Info at node	Distance to Node A	B	C	D	E
A	0	6	5	3	1
B	6	0	1	3	5
C	5	1	0	2	4
D	3	3	2	0	2
E	1	5	4	2	0

Vetor de distância de A

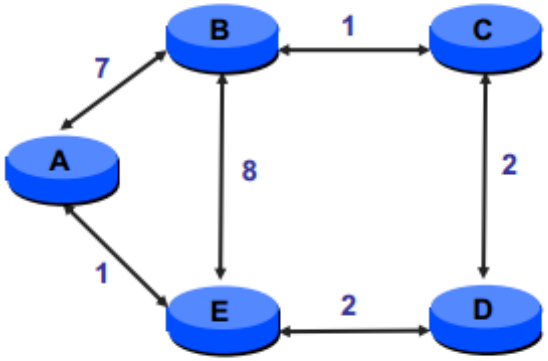
Dest	Custo	Next Hop
B	6	E
C	5	E
D	3	E
E	1	E

Vetor de distância de B

Dest	Custo	Next Hop
A	6	C
C	1	C
D	3	C
E	5	C

Algoritmos de Roteamento

- Roteamento por Vetor de Distâncias - Exemplo
  - Até a convergência...



Info at node	Distance to Node				
	A	B	C	D	E
A	0	6	5	3	1
B	6	0	1	3	5
C	5	1	0	2	4
D	3	3	2	0	2
E	1	5	4	2	0

Vetor de distância de C

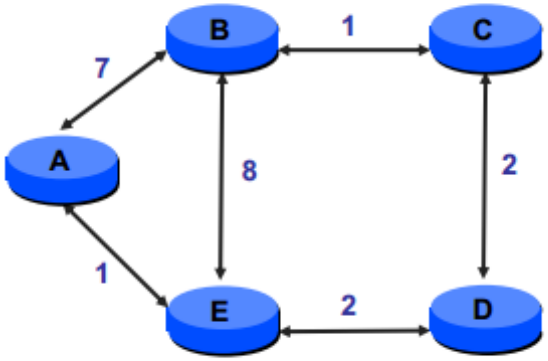
Dest	Custo	Next Hop
A	5	D
B	1	B
D	2	D
E	4	D

Vetor de distância de D

Dest	Custo	Next Hop
A	3	E
B	3	C
C	2	C
E	2	E

Algoritmos de Roteamento

- Roteamento por Vetor de Distâncias - Exemplo
  - Até a convergência...



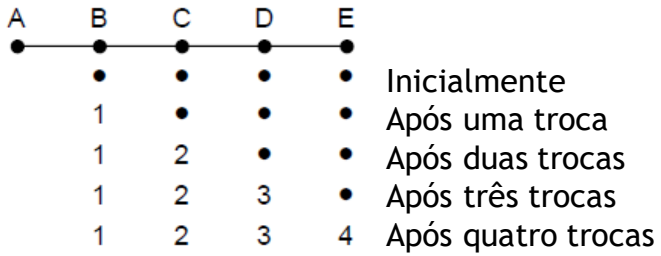
Info at node	Distance to Node A	B	C	D	E
A	0	6	5	3	1
B	6	0	1	3	5
C	5	1	0	2	4
D	3	3	2	0	2
E	1	5	4	2	0

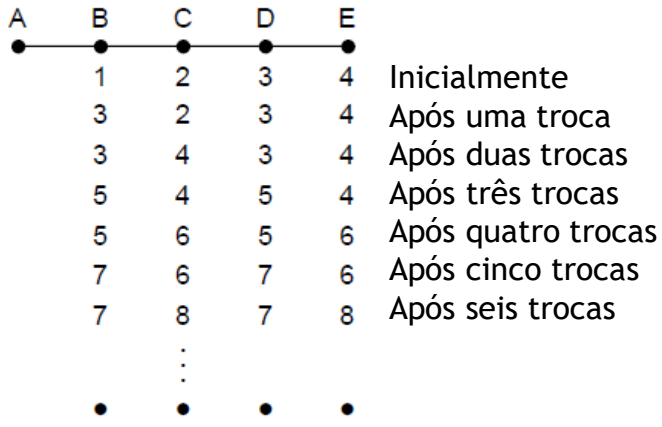
Vetor de distância de E

Dest	Custo	Next Hop
A	1	A
B	5	D
C	4	D
D	2	D

Algoritmos de Roteamento

- Roteamento por Vetor de Distâncias
  - Problema da contagem ao infinito
    - O estabelecimento das rotas pela rede é chamado de convergência
    - Apesar de convergir para a resposta correta, o roteamento por vetor de distâncias pode ser feito de forma lenta
    - Quando um nó falha, a atualização das tabelas pode demorar, uma vez que os vizinhos anunciarão aos demais que conseguem chegar até o nó que falhou (mas na realidade não conseguem)





Algoritmos de Roteamento

- Roteamento por Vetor de Distâncias (na prática) – RIP
  - RIPv1 (*Routing Information Protocol*)
    - Definido pela RFC 1058 para IPv4
    - A métrica utilizada é o número de máquinas intermediárias (número de *hops*)
    - Não suporta máscaras de sub-rede (*classful*)
    - Cada roteador divulga sua tabela periodicamente a cada 30 segundos
    - As mensagens divulgadas levam *n tuplas* contendo: <redes destino, métrica>
    - A divulgação para os vizinhos é realizada por *broadcast* (endereço IP 255.255.255.255)
    - No procedimento normal, se a rota não for atualizada em 180 segundos é considerada inatingível
    - A informação de rota inatingível é repassada aos roteadores “vizinhos” (diretamente alcançáveis)
  - RIPv2
    - Definido pelas RFCs 1721 e 1722 para IPv4
    - Similar ao RIPv1, mas com suporte a máscaras de sub-rede (*classless*)
    - A divulgação para os vizinhos é realizada por *multicast* (endereço IP 224.0.0.9)

Algoritmos de Roteamento

- Roteamento por Vetor de Distâncias (na prática) – RIP

  - Comandos para habilitar o RIP no Cisco Packet Tracer (na CLI)
    - RIPv1

      (digitar antes `enable`, e em seguida `configure terminal`)

      ```
      router rip
          network [rede_a_anunciar]
      ```

    - RIPv2

      (digitar antes `enable`, e em seguida `configure terminal`)

      ```
      router rip
          version 2
          no auto-summary
          network [rede_a_anunciar]
      ```

Algoritmos de Roteamento

- Roteamento por Vetor de Distâncias (na prática) – RIP
  - Problemas/desvantagens:
    - RIPv1 não possui mecanismos de segurança/autenticação
      - É suscetível a *IP spoofing* ("falsificação" de IP)
      - RIPv2 implementa hash MD5 para autenticação
    - Não permite balanceamento de carga
    - Não tem controle da “idade” das mensagens
      - Mensagens “velhas” podem ser processadas após mensagens “novas”
        - Causa inconsistências nas tabelas de roteamento
    - Limitação de número de roteadores intermediários (máximo 15 hops)
      - Métrica = 16, indica rota inalcançável
    - RIPv1 não suporta máscaras de sub-rede
      - RIPv2 suporta
    - Não é possível formar hierarquia de roteadores, isto é, a rede é plana
      - Não escalável

Algoritmos de Roteamento

- Roteamento por Estado de Enlace (LS)
  - O roteamento por vetor de distância demora para convergir devido ao problema da contagem ao infinito
  - Os algoritmos de roteamento LS são baseados em tabela (proativos)
  - O roteamento por estado de enlace foi proposto para resolver esse problema
  - Todos os nós precisam conhecer o mapa da rede para aplicar o algoritmo de caminho mínimo
    - Conhecido como algoritmo de Dijkstra
  - Cada roteador realiza os seguintes passos:
    1. Descobrir seus vizinhos e aprender seus endereços de rede
    2. Medir o atraso (ou o custo) até cada um de seus vizinhos
    3. Criar um pacote que informe tudo o que acabou de aprender
    4. Enviar este pacote e receber pacotes de todos os outros roteadores
    5. Calcular o caminho mais curto até cada um dos outros roteadores
  - O algoritmo OSPF (*Open Shortest Path First*) é um exemplo de LS

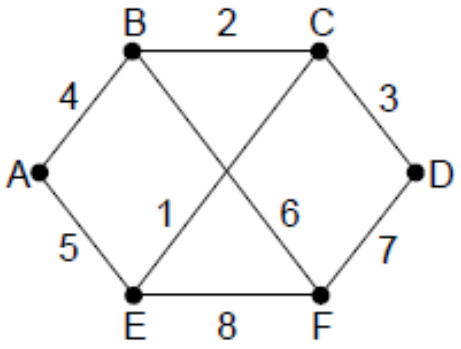
Algoritmos de Roteamento

- Roteamento por Estado de Enlace
  1. Descobrir seus vizinhos e aprender seus endereços de rede
     - Um pacote ‘hello’ é enviado em cada enlace do roteador
     - Cada vizinho que receber, deve responder informando quem é (seu endereço de rede)

  2. Medir o atraso (ou o custo) até cada um de seus vizinhos
     - O algoritmo de estado de enlace exige que cada roteador conheça o atraso para cada um de seus vizinhos
     - Esse atraso é medido de forma automática:
       - Um pacote especial ‘echo’ é enviado e devolvido imediatamente pelo receptor
       - Com o tempo de ida e volta (RTT) desse pacote é possível estimar o atraso

Algoritmos de Roteamento

- Roteamento por Estado de Enlace
  3. Criar um pacote contendo tudo que acabou de aprender (pacote de estado de enlace)
     - Os pacotes devem conter: identidade do transmissor, número de sequência, idade (TTL), lista de vizinhos e os custos do enlace
     - Quando criar estes pacotes?
       - Periodicamente em intervalos regulares
       - Na ocorrência de eventos significativos (quebra de enlace gerada pela saída de algum vizinho)

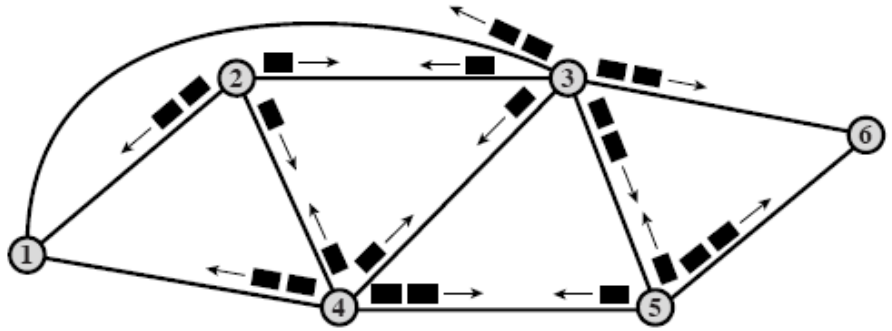


Link State Packets

A		B		C		D		E		F	
Seq.		Seq.		Seq.		Seq.		Seq.		Seq.	
Age		Age		Age		Age		Age		Age	
B	4	A	4	B	2	C	3	A	5	B	6
E	5	C	2	D	3	F	7	C	1	D	7
		F	6	E	1			F	8	E	8

Algoritmos de Roteamento

- Roteamento por Estado de Enlace
  4. Enviar este pacote e também receber os pacotes dos outros roteadores
     - Todos os roteadores precisam receber os pacotes de estado de enlace de forma rápida e confiável
     - Para isso é utilizada a técnica de inundação (*flooding*) e cada pacote deverá ter um número de sequência único e um tempo de vida
       - Para tornar o algoritmo mais robusto, os pacotes de estado de enlace são retidos antes de serem transmitidos
         - Aguardam um determinado tempo em uma área de retenção caso mais pacotes de estado de enlace cheguem
         - Caso chegue um pacote de estado de enlace, proveniente da mesma origem, mas com número de sequência diferente, é descartado o mais antigo



Algoritmos de Roteamento

- Roteamento por Estado de Enlace
  5. Calcular o caminho mais curto até cada um dos outros roteadores
     - Uma vez que uma rota tenha “acumulado” um conjunto completo de pacotes de estado de enlance, é possível criar um grafo de sub-rede
     - Executa-se então o algoritmo de Dijkstra localmente para obter o caminho mais curto até todos os destinos possíveis
       - O resultado desse algoritmo é armazenado na tabela de roteamento e a operação de repasse se baseará na mesma

Algoritmos de Roteamento

- Roteamento por Estado de Enlace (na prática) - OSPF
  - OSPF (*Open Shortest Path First)*
    - Definido pela RFC 2328 para IPv4 e RFC 5340 para IPv6
    - A métrica utilizada é o custo da interface (baseado a largura de banda)
    - Cada roteador mantém uma base de dados descrevendo a topologia
      - Constrói uma árvore de menores caminhos alcançáveis
      - Calcula rotas com base no algoritmo de Dijkstra
      - Rápida convergência e rápida recuperação de enlaces quebrados
    - Oferece balanceamento de carga
      - Em caso de rotas com mesmo “custo”, a carga é dividida por igual para cada uma
    - Permite particionar a rede em múltiplas zonas/áreas
    - Suporta máscaras de sub-rede (*classless*)

Algoritmos de Roteamento

- Roteamento por Estado de Enlace (na prática) - OSPF

  - Comandos para habilitar o OSPFv2 no Cisco Packet Tracer (na CLI)

    (digitar antes `enable`, e em seguida `configure terminal`)

```
router ospf [id_processo]
   network [rede_a_anunciar] [mascara_wildcard] area [numero_da_zona]
```