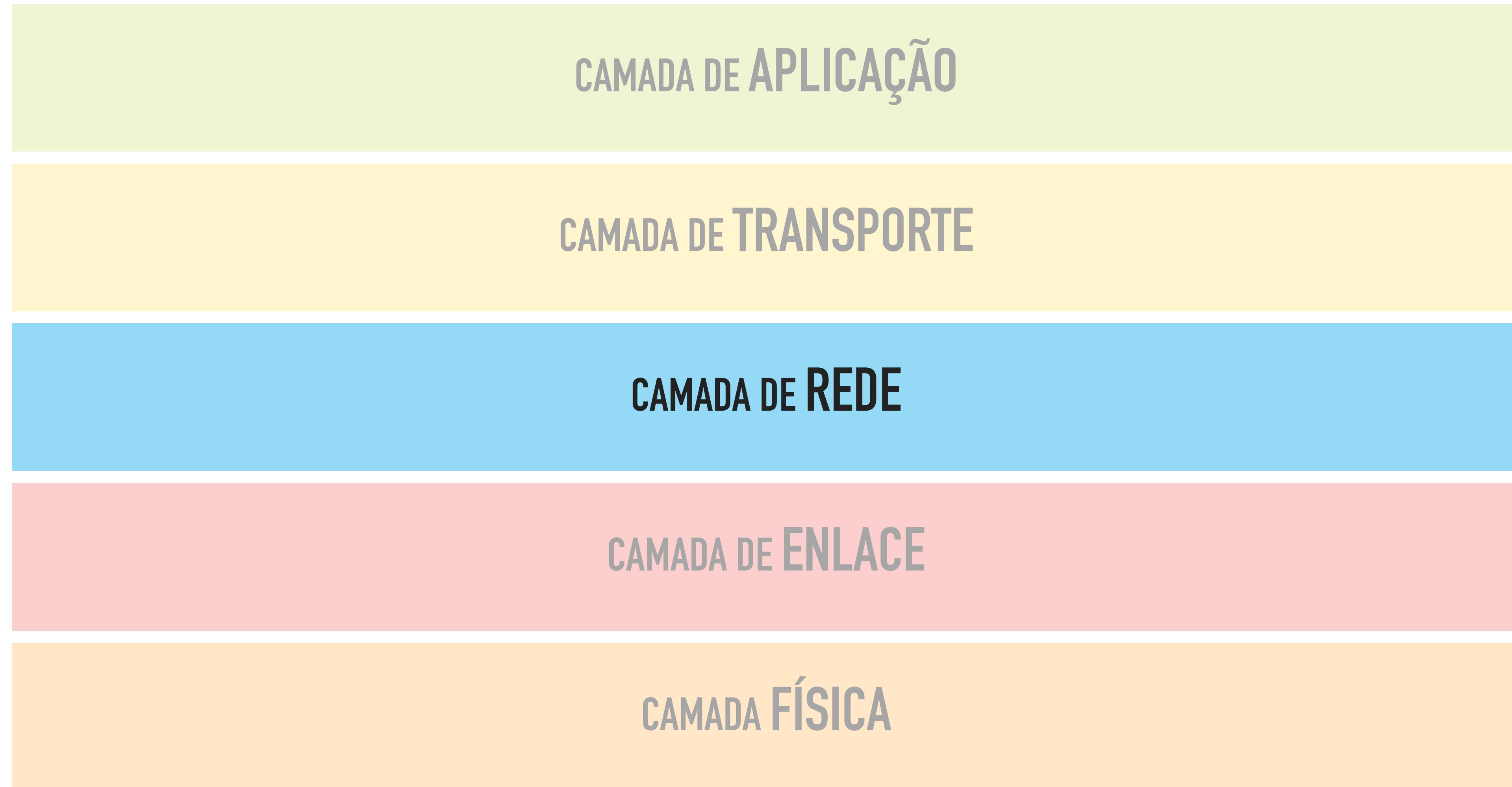


IMD0043

ROTEAMENTO POR ESTADO DE ENLACE



TRÊS CATEGORIAS DE PROTOCOLOS

Criar árvore, rotear na árvore

- ▶ Ex: *spanning tree protocol* (STP) para Ethernet comutada (lembre das camadas!)
 - ▶ Vantagens: fácil, sem *loops*, sem *dead ends*
 - ▶ Desvantagens: processamento desnecessário, alta latência, largura de banda ineficiente

Obter uma visão global

- ▶ Ex: estado de enlace (*link state*)

Computação distribuída de roteamento

- ▶ Ex: vetor de distância (*distance vector*)
- ▶ Ex: Protocolo BGP

MÉTRICAS DE ROTEAMENTO

- ▶ Objetivos do roteamento: calcular caminhos/rotas com **custo mínimo** X
 - ✓ X = número de saltos (*hops*)
 - ✓ X = latência
 - ✓ X = peso
 - ✓ X = probabilidade de falha
 - ✓ ...
- ▶ Geralmente assumimos que todos os *links* tem um custo associado
- ▶ Precisamos **minimizar o custo do caminho completo**
 - ▶ *custo da rota* = soma dos custos individuais dos enlaces do caminho

#1 CRIAR ÁRVORE A PARTIR DA TOPOLOGIA

- ▶ Remover *links* suficientes para que se obtenha uma árvore com todos os nós
 - ▶ *Spanning trees!*
- ▶ Porém:
 - ▶ Uso desnecessário de recursos para processar pacotes
 - ▶ Alta latência
 - ▶ Desperdício de largura de banda

#2 VISÃO GLOBAL

- ▶ Protocolo:
 - ▶ Onde criar uma visão global
 - ▶ Como criar uma visão global
 - ▶ Disseminar os cálculos de roteamento (se necessário)
 - ▶ Quando executar os cálculos de roteamento
- ▶ Algoritmo:
 - ▶ Calcular as rotas com menores custos (ex: Dijkstra)

ONDE CRIAR A VISÃO GLOBAL?

- ▶ Opção #1: servidor central
 - ▶ Coleta uma visão global
 - ▶ Calcula a tabela de roteamento para cada nó
 - ▶ "Instala" as tabelas de roteamento em cada nó
 - ▶ *Software-defined networks (SDN)*: veremos no final do curso...
- ▶ Opção #2: em cada roteador
 - ▶ Cada roteador coleta uma visão global
 - ▶ Calcula sua própria tabela de roteamento usando um protocolo de *link-state*
 - ▶ Protocolo de roteamento de estado de enlace. ex: OSPF

ROTEAMENTO LINK-STATE (ESTADO DE ENLACE)

Todo roteador sabe o seu estado de enlace

- ▶ Conhece o estado dos *links* para seus vizinhos
- ▶ *up/down* e o custo associado

O roteador *flooda* o seu estado de enlace para todos os outros roteadores

- ▶ Utiliza um pacote especial - *Link State Announcements* (LSA)
- ▶ Esse anúncio é entregue a todos os nós
- ▶ Dessa forma, todos os roteadores aprendem o grafo de toda a rede

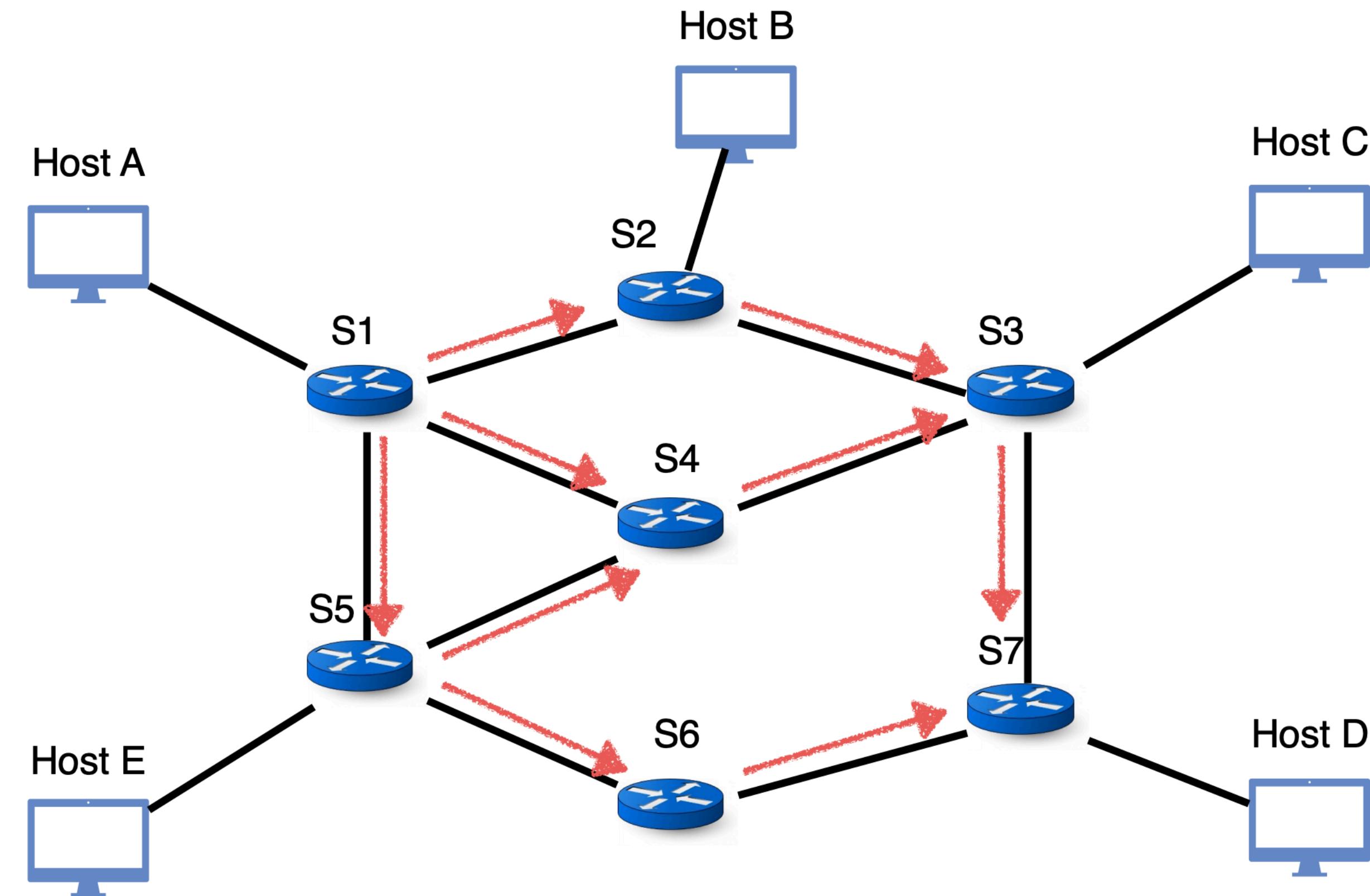
Executa o cálculo de roteamento localmente

- ▶ O custo mínimo das rotas partindo dele para todos os outros

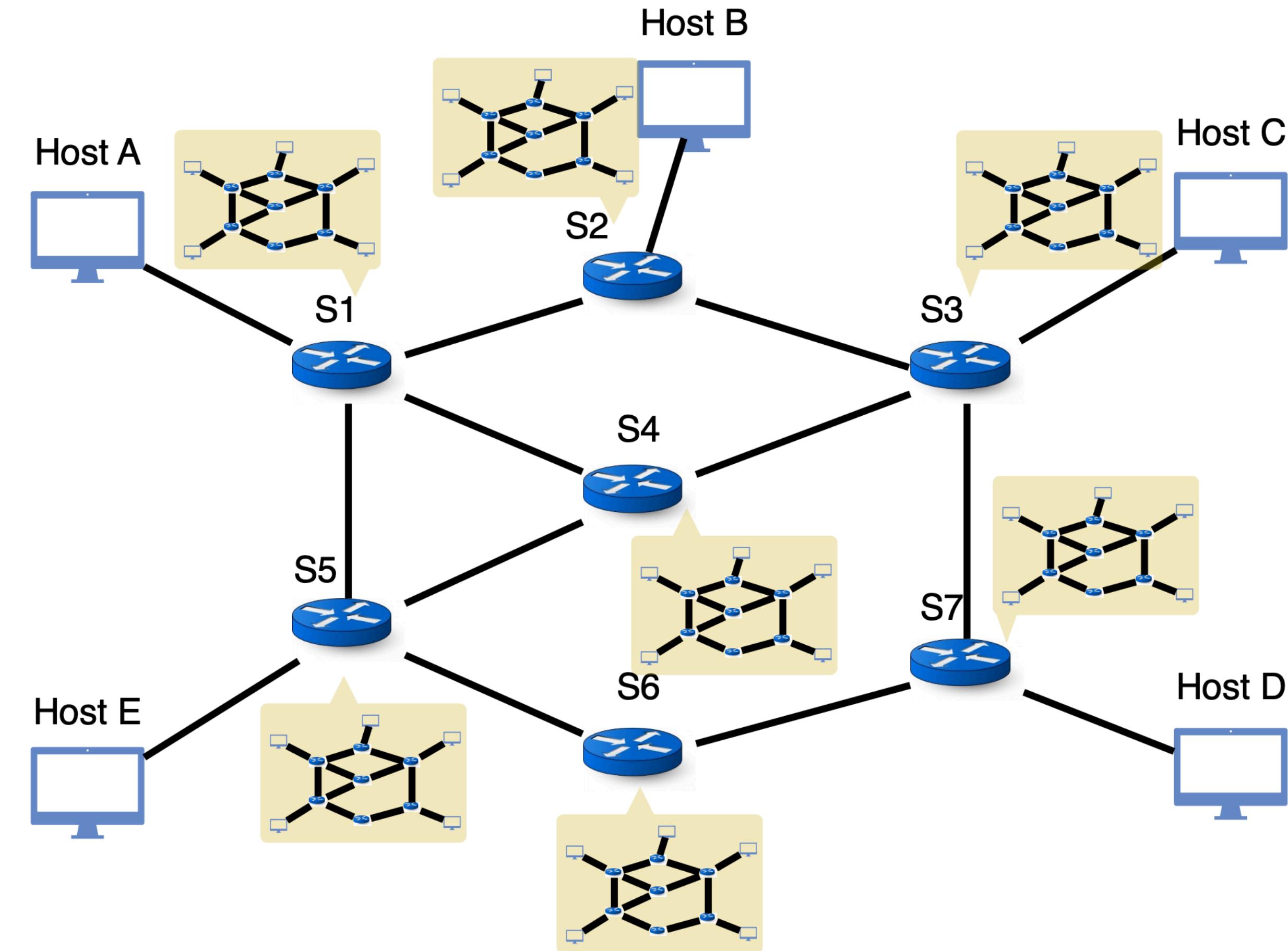
COMO O FLOODING FUNCIONA?

- ▶ LSA é recebido em um *link* do roteador
- ▶ Esse roteador:
 - ▶ Persiste/armazena o pacote
 - ▶ Encaminha o pacote para todos os seus outros *links*
 - ▶ Não envia para o *link* de entrada. Por que?
- ▶ Se um LSA já recebido chega novamente em um roteador
 - ▶ Roteador *drops* o pacote (não é necessário reencaminha-lo)

ROTEAMENTO POR ESTADO DE ENLACE



CADA NÓ TEM UMA VISÃO GLOBAL



QUANDO INICIAR O FLOOD DE ANÚNCIOS?

Mudança de topologia

- ▶ Falhas em *links*
- ▶ Recuperação de *links*

Mudança de configuração

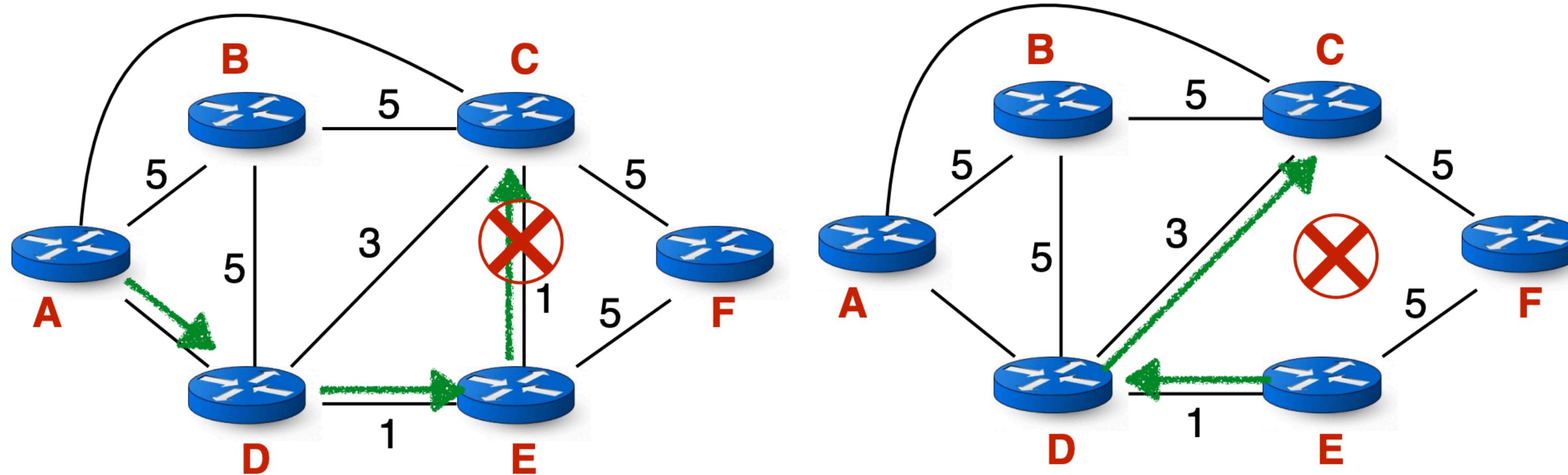
- ▶ Mudança no custo dos *links*

Periodicamente

- ▶ Atualiza a informação do estado de enlace
- ▶ Tipicamente, 30min (exemplo)
- ▶ Corrige eventuais dados corrompidos

LOOPS AINDA SÃO POSSÍVEIS?

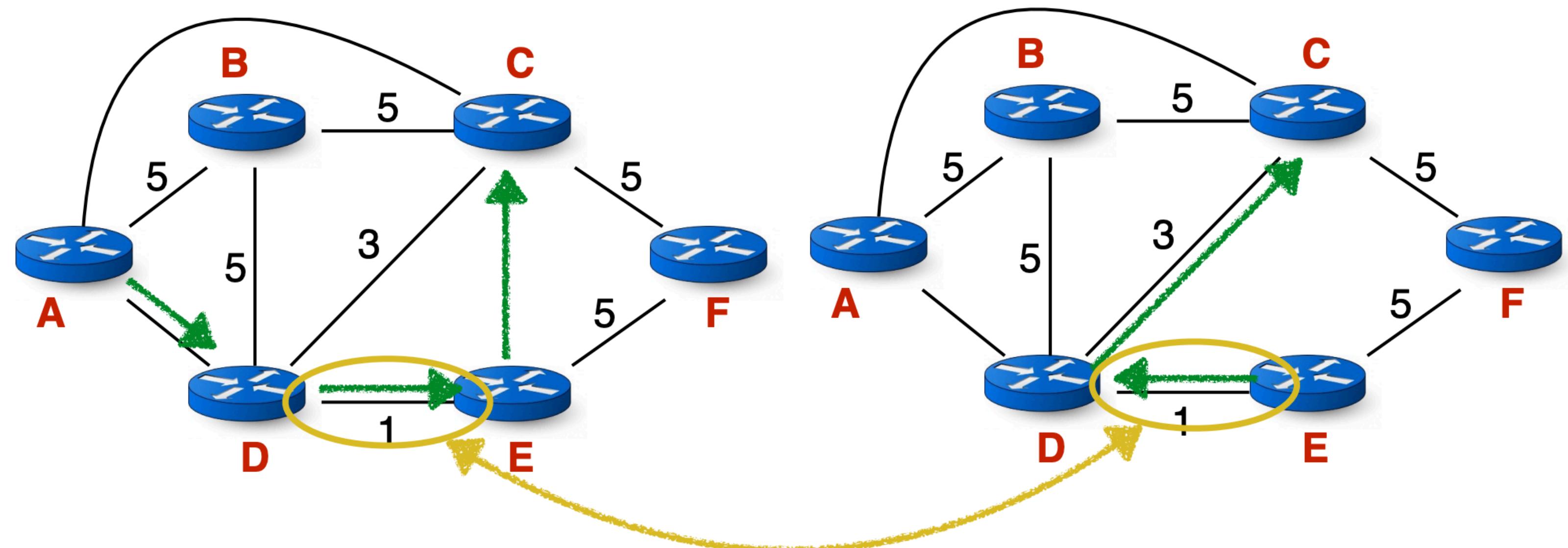
- ▶ Podemos ter *loops* utilizando roteamento por estado de enlace?



A e D sabem esse caminho para C
Link E-C falha, mas D ainda não sabe

E sabe esse caminho para C
E chega em C por D, D chega em C via E
LOOP!

LOOPS TRANSIENTES



- ▶ Visões de estados de enlace inconsistentes
- ▶ Alguns roteadores ficam sabendo de falhas antes de outros
- ▶ Os menores caminhos podem não ser mais consistentes
- ▶ Pode causar *loops* transitentes de encaminhamento

CONVERGÊNCIA EM ROTEAMENTO POR ESTADO DE ENLACE

- ▶ **Eventualmente**, todos os roteadores terão informação de roteamento consistente
 - ▶ ex: todos os nós contendo o mesmo banco de dados de estado de enlace
- ▶ Eventualmente na verdade quer dizer: “*se nada mudar por um tempo*”
- ▶ O encaminhamento é consistente após a convergência:
 - ▶ Todos os nós tem o mesmo banco de estado de enlaces
 - ▶ Todos os nós encaminham pacotes pelos mesmos caminhos
- ▶ Porém, enquanto ainda está convergindo, **problemas podem ocorrer!**

TEMPO DE CONVERGÊNCIA

Fontes de atraso da convergência?

- ▶ Tempo para detectar uma falha
- ▶ Tempo para *floodar* a informação do estado de enlace (\sim o maior RTT)
- ▶ Tempo para recalcular as tabelas de roteamento

Problemas de desempenho durante o período de convergência?

- ▶ *Dead ends*
- ▶ Pacotes em *loop*
- ▶ e outros problemas que veremos mais para a frente!

ESTADO DE ENLACE É CONCEITUALMENTE SIMPLES

- ▶ Todo mundo *flooda* a informação dos *links*
- ▶ Todo mundo sabe o grafo da rede
- ▶ Todo mundo calcula independentemente os caminhos no grafo

A complexidade está nos detalhes

PRÓXIMA SEMANA

Algoritmos por vetor de distâncias