Aula 19 - Algoritmos de Vetor de Distâncias, Roteamento Hierárquico

Diego Passos

Universidade Federal Fluminense

Redes de Computadores I

Material adaptado a partir dos slides originais de J.F Kurose and K.W. Ross.

Revisão da Última Aula...

- **Roteamento:**
  - Encontrar caminhos fim-a-fim.
  - **Construir tabela de roteamento**.
    - Consultada no encaminhamento.
- **Grafos**: usados como abstração para representar a rede.
  - Roteadores são nós.
  - Enlaces são arestas.
  - Podem ter pesos: **medida de qualidade do enlace**.
    - Relacionado a banda, atraso, congestionamento, ...
- **Classificações:**
  - Estado de Enlace *vs.* Vetor de Distâncias.
  - Dinâmico *vs.* Estático.
- **Roteamento baseado em Estado de Enlaces:**
  - Algoritmo de Dijkstra.

Algoritmos Baseados em Vetor de Distâncias

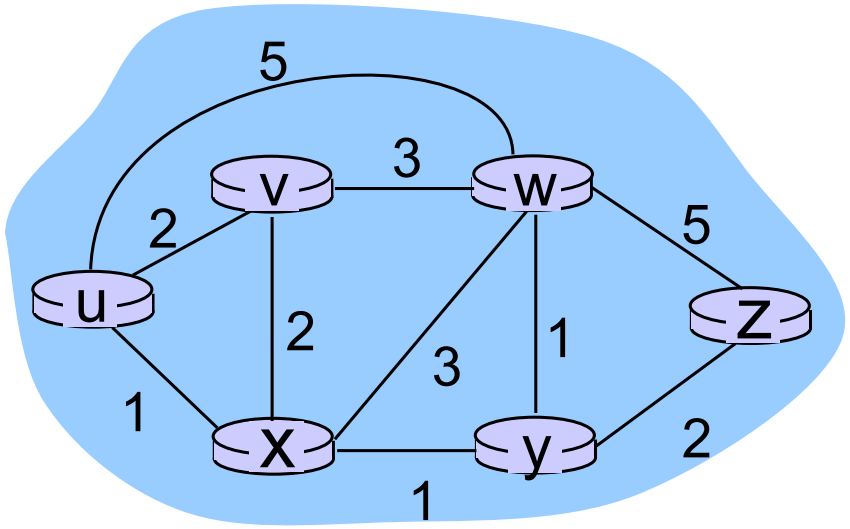
Algoritmo de Vetor de Distâncias

- **Equação de Bellman-Ford**.
  - Programação dinâmica.
- Seja $d_a(b)$ o custo do caminho de menor custo de a para b.
- Digamos que queremos calcular o custo do melhor caminho entre x e y.
- Suponha que, **de alguma forma**, conhecemos o custo dos melhores caminhos de todos os vizinhos v de x até y.
- Então:

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

- Em outras palavras, o melhor caminho de x para y **necessariamente**:
  - Tem como próximo salto um vizinho de x.
  - Utiliza o melhor caminho deste vizinho até y.

Equação de Bellman-Ford: Exemplo



- Claramente:
$$d_v(z) = 5, d_x(z) = 3, d_w(z) = 3$$
- Equação de Bellman-Ford diz que:

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

- Vizinho que resulta no custo mínimo é o próximo salto do caminho mais curto.
- Informação armazenada na tabela de roteamento.

Algoritmo de Vetor de Distâncias (I)

- $D_x(y)$: estimativa do custo mínimo de x para y.
  - Cada nó x mantém **vetor de distâncias** $D_x = [D_x(y), \forall y \in N]$.
- Nó x:
  - Conhece custo para cada vizinho v: $c(x, v)$.
  - Recebe os vetores de distância de seus vizinhos: $D_x = [D_x(y), \forall y \in N]$

Algoritmo de Vetor de Distâncias (II)

- **Ideia chave:**
  - De tempos em tempos, cada nó envia seu próprio vetor de distância com suas estimativas para cada vizinho.
  - Quando x recebe novo vetor de distância de um vizinho, ele atualiza seu próprio vetor, aplicando a equação de Bellman-Ford:

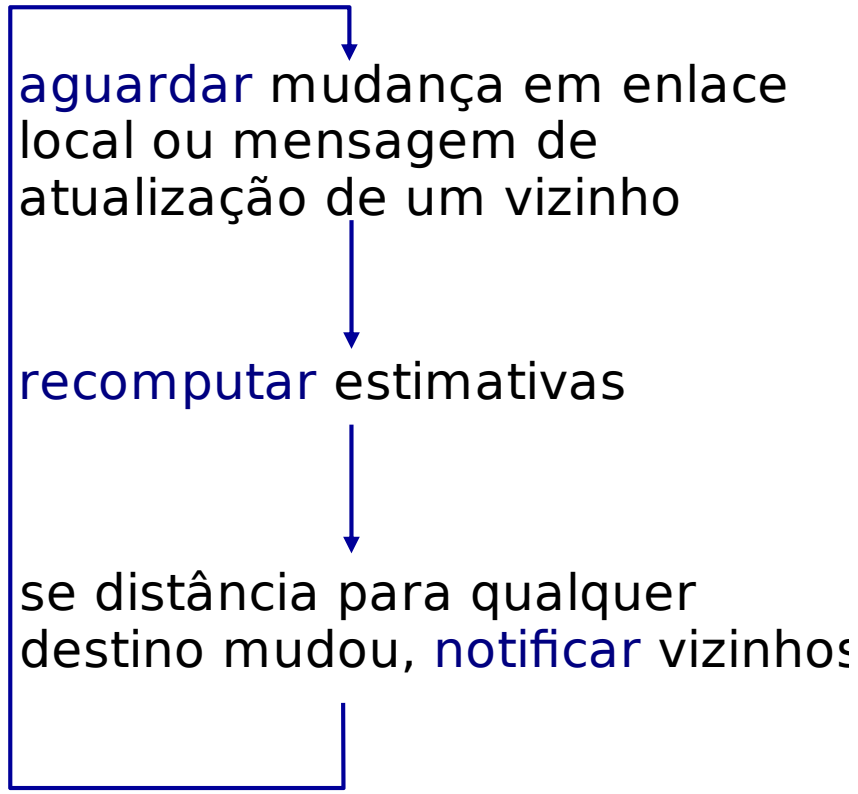
$$D_x(y) = min_v \{c(x, y) + d_v(y)\}$$

  - Sob hipóteses razoáveis na prática, as estimativas $D_x(y)$ convergem para os menores custos reais $d_x(y)$.

Algoritmo de Vetor de Distâncias (III)

- **Iterativo, assíncrono.**
  - Cada iteração local causada por:
    - Alteração no custo de um enlace local.
    - Ou pelo recebimento de um vetor de distâncias atualizado.
- **Distribuído:**
  - Cada nó notifica vizinhos apenas quando seu vetor de distâncias muda.
  - Vizinhos repassam informação da mudança para seus vizinhos, se necessário.

- **Operação em cada nó:**



Vetor de Distâncias: Exemplo (I)

$D_x(y) = \min\{c(x,y) + D_y(y),\ c(x,z) + D_z(y)\}$
$= \min\{2+0\ ,\ 7+1\} = 2$

$D_x(z) = \min\{c(x,y) +$
$D_y(z),\ c(x,z) + D_z(z)\}$
$= \min\{2+1\ ,\ 7+0\} = 3$

Tabela nó x

origem \ destino	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

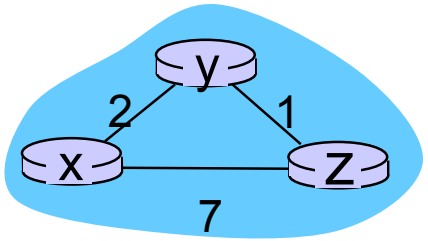
origem \ destino	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

Tabela nó y

origem \ destino	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

Tabela nó z

origem \ destino	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0



tempo

Vetor de Distâncias: Exemplo (II)

$D_x(y) = \min\{c(x,y) + D_y(y),\ c(x,z) + D_z(y)\}$
$= \min\{2+0\ ,\ 7+1\} = 2$

$D_x(z) = \min\{c(x,y) + D_y(z),\ c(x,z) + D_z(z)\}$
$= \min\{2+1\ ,\ 7+0\} = 3$

Tabela nó x

origem \ destino	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

origem \ destino	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

origem \ destino	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

Tabela nó y

origem \ destino	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

origem \ destino	x	y	z
x	0	2	7
y	2	0	1
z	7	1	0

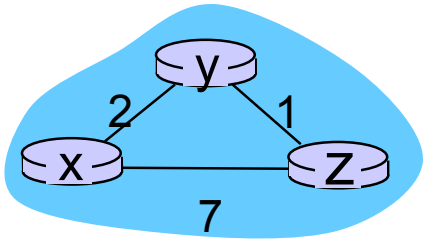
origem \ destino	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

Tabela nó z

origem \ destino	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0

origem \ destino	x	y	z
x	0	2	7
y	2	0	1
z	3	1	0

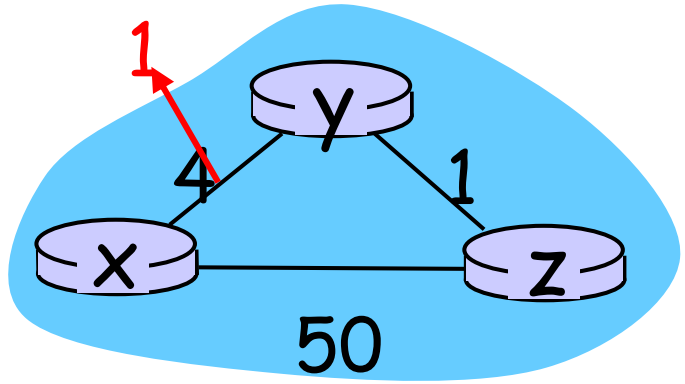
origem \ destino	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0



tempo

Vetor de Distâncias: Mudanças nos Custos dos Enlaces (I)

- **Mudanças nos custos dos enlaces:**
  - Nó detecta alteração em custo de enlace local.
  - Atualiza informação de roteamento, recalcula vetor de distâncias.
  - Se vetor muda, notifica vizinhos.

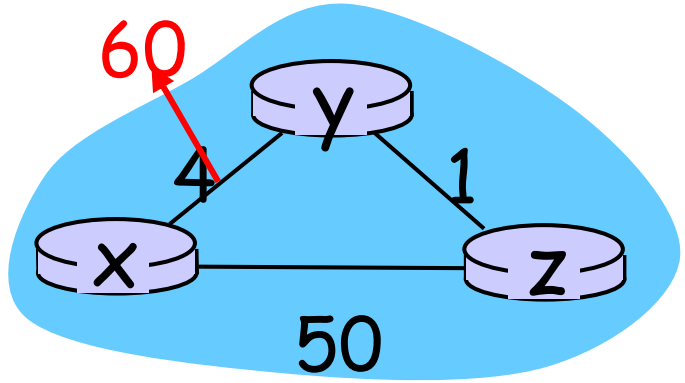


"Noticias boas viajam rápido"

- t_0: y detecta mudança no custo do enlace, atualiza vetor, informa seus vizinhos.
- t_1: z recebe atualização de y, atualiza sua tabela, computa novo custo mínimo para x, envia seu vetor para seus vizinhos.
- t_2: y recebe atualização de z, atualiza sua tabela. Menor custo para x não muda, logo y não envia nova mensagem para z.

Vetor de Distâncias: Mudanças nos Custos dos Enlaces (II)

- **Mudanças nos custos dos enlaces:**
  - Nó detecta alteração em custo de enlace local.
  - **"Notícias ruins demoram"**: problema de **contagem ao infinito**!
  - 44 iterações até que algoritmo se estabilize.



Envenenamento Reverso

- Se z usa y como próximo salto para x:
  - z anuncia para y que sua distância para x é infinita.
  - Assim y não escolherá z como próximo salto para x.
- **Resolve completamente o problema?**

Estado de Enlace *vs.* Vetor de Distância

- **Complexidade de mensagens:**
  - **LS:** com n nós, E enlaces, $O(nE)$ mensagens enviadas.
  - **DV:** mensagens trocadas apenas com vizinhos.
    - O **tempo de convergência** varia.
- **Velocidade de convergência:**
  - **LS:** complexidade de processamento de $O(n^2)$, mais $O(nE)$ mensagens trocadas.
    - Pode apresentar oscilações.
    - Pode haver *loops* no roteamento.
  - **DV:** tempo de convergência depende.
    - Pode haver *loops* nas rotas.
    - Pode haver contagem ao infinito.
- **Robustez:** o que acontece se o roteador funciona incorretamente?
  - **LS:**
    - Roteador defeituoso pode anunciar **custos de enlaces** errados.
    - Cada nó computa apenas a sua tabela.
  - **DV:**
    - Roteador pode anunciar **custo de um caminho** errado.
    - A tabela de roteamento de um nó é usada pelos demais.
      - Erro se propaga pela rede.

Roteamento Hierárquico

Roteamento Hierárquico (I)

- Nosso estudo sobre roteamento tem sido idealizado até aqui.
  - Roteadores são idênticos.
  - Rede é “plana”.
  - ... nada disso é verdade na prática na Internet.

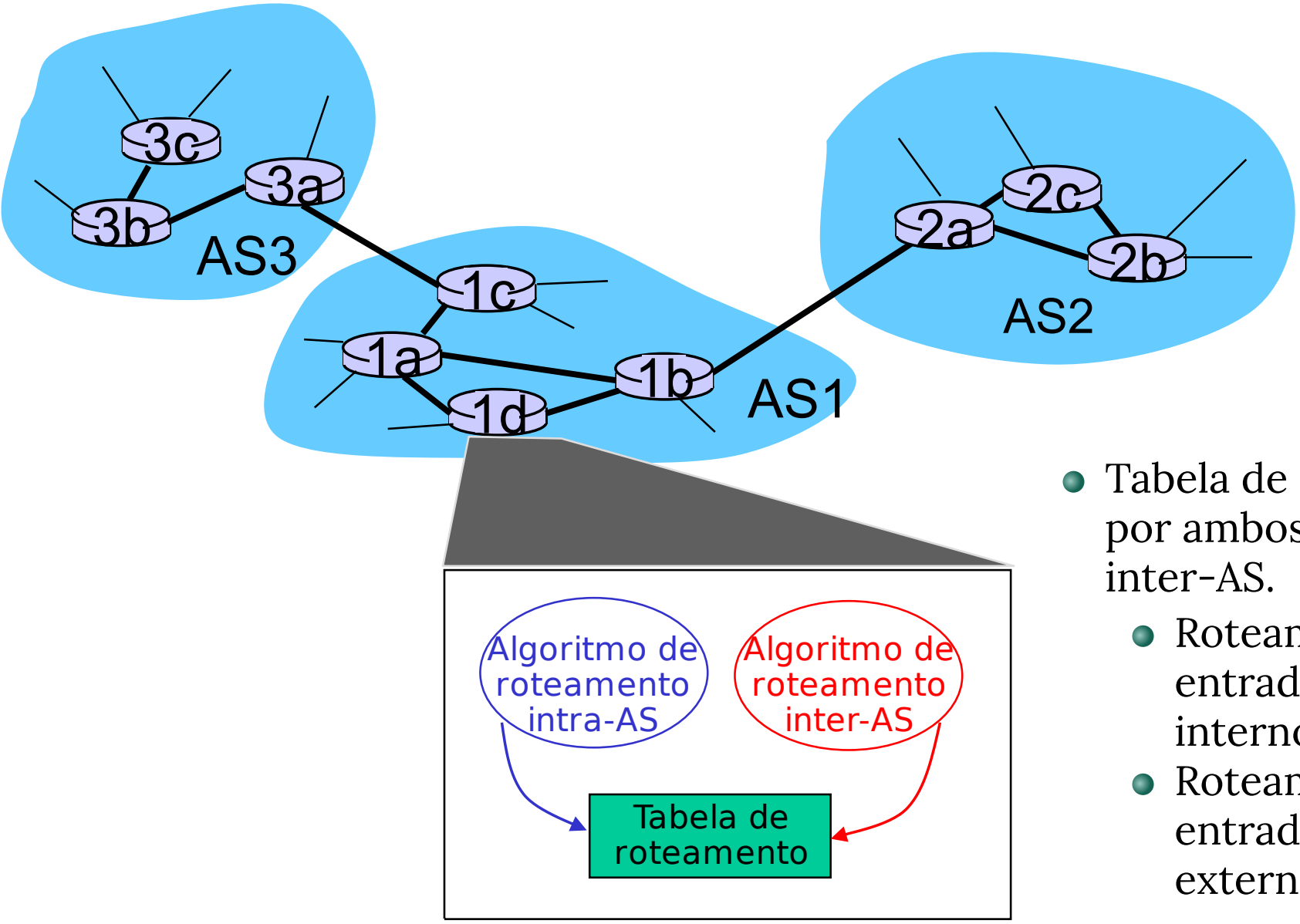
- **Escala:** com 600 milhões de destinos:
  - Não é possível armazenar todos os destinatários em tabelas de roteamento.
  - Trocas de tabelas de roteamento iria afogar os enlaces!

- **Autonomia administrativa:**
  - Internet = Rede de redes.
  - Cada administrador de rede pode querer controlar o roteamento na sua própria rede.

Roteamento Hierárquico (II)

- Agregar roteadores em regiões, **"sistemas autônomos"**.
  - Ou **AS**, da sigla em inglês.
- Roteadores no mesmo AS rodam o mesmo protocolo de roteamento.
  - Protocolo de roteamento **intra-AS**.
  - Roteadores em ASs diferentes podem rodar diferentes protocolos intra-AS.
- **Roteador *gateway*:**
  - Nas "bordas" do seu AS.
  - Possui enlace para roteador(es) de outros ASs.

ASs Interconectados

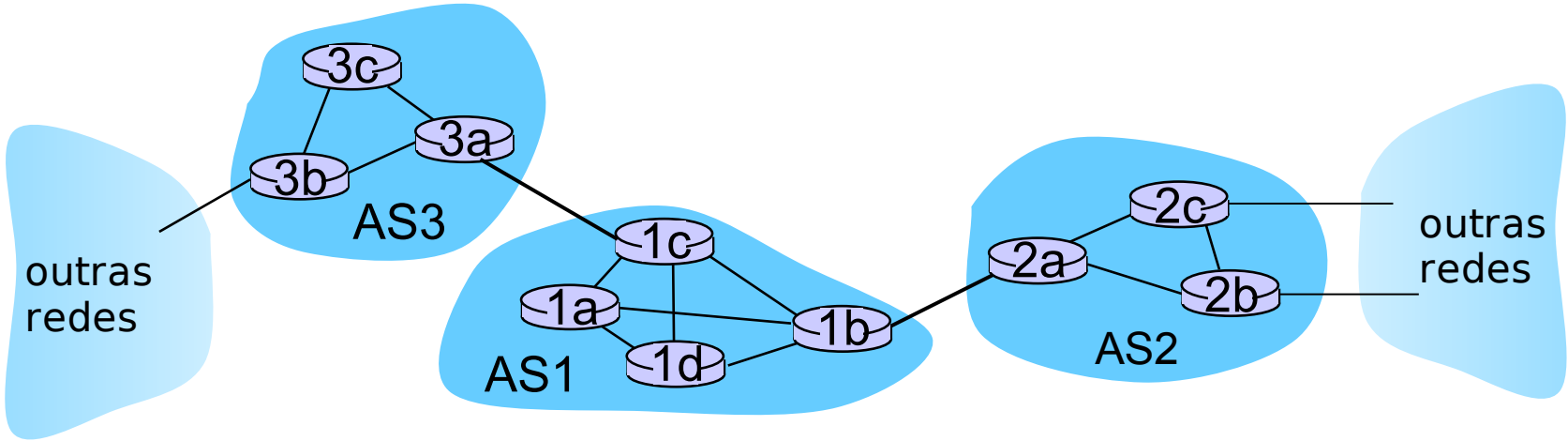


- Tabela de roteamento configurada por ambos os roteamentos intra- e inter-AS.
  - Roteamento intra-AS configura entradas para destinatários internos.
  - Roteamento inter-AS configura entradas para destinatários externos.

Tarefas do Roteamento Inter-AS

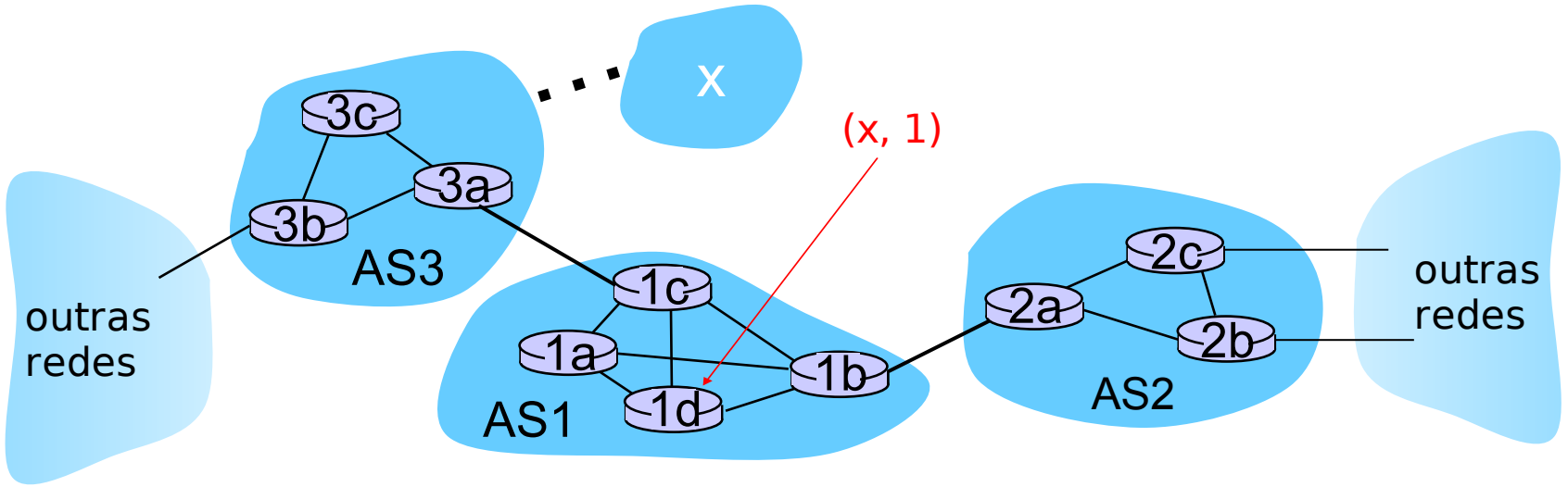
- Suponha que um roteador no AS1 recebe datagrama destinado para fora do AS1:
  - Roteador deve encaminhar pacote para um roteador *gateway*, mas qual?

- **AS1 deve:**
  - Aprender quais destinatártios são alcançáveis através do AS2 e quais através do AS3.
  - Propagar esta informação para todos os roteadores no AS1.
- **Trabalho do roteamento inter-AS!**



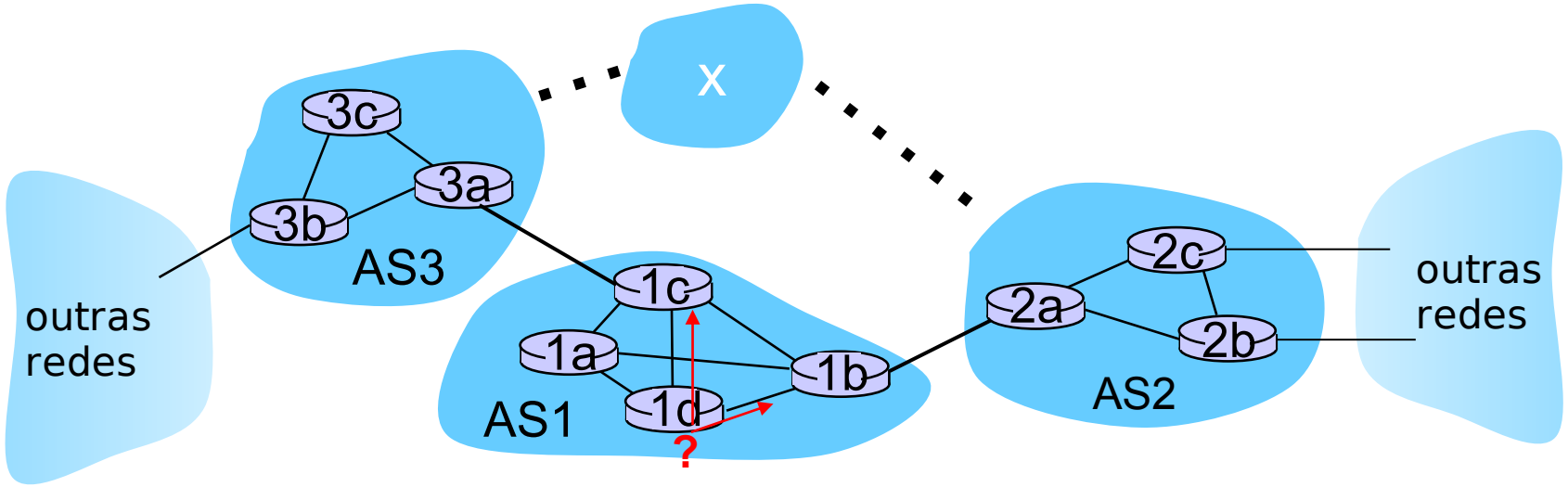
Exemplo: Configurando a Tabela de Roteamento do Roteador 1d

- Suponha que o AS1 aprenda (através do roteamento inter-AS) que a sub-rede **x** é alcançável pelo AS3 (*gateway* 1c), mas não via AS2.
  - Protocolo de roteamento inter-AS propaga esta informação para todos os roteadores internos.
- Roteador 1d determina, usando o roteamento intra-AS, que sua interface 1 está no caminho de menor custo para 1c.
  - Instala entrada **`(x, 1)`** na tabela de roteamento.



Exemplo: Escolhendo entre Múltiplos ASs (I)

- Agora suponha que o AS1 aprenda a partir do protocolo inter-AS que a sub-rede **x** é alcançável por ambos os ASs 3 e 2.
- Para reconfigurar a tabela de roteamento, o roteador 1d precisa determinar para qual *gateway* deve encaminhar os pacotes destinados a **x**.
  - Isto também é uma tarefa do protocolo de roteamento inter-AS!



Exemplo: Escolhendo entre Múltiplos ASs (II)

- Agora suponha que o AS1 aprenda a partir do protocolo inter-AS que a sub-rede **x** é alcançável por ambos os ASs 3 e 2.
- Para reconfigurar a tabela de roteamento, o roteador 1d precisa determinar para qual *gateway* deve encaminhar os pacotes destinados a **x**.
  - Isto também é uma tarefa do protocolo de roteamento intra-AS!
- **Roteamento batata-quente: envie** pacote em direção ao *gateway* mais próximo.

Aprenda pelo protocolo inter-AS que a sub-rede **x** é alcançável por múltiplos *gateways*. → Use informação de roteamento do protocolo intra-AS para determinar custo dos caminhos de menor custo para cada um dos *gateways*. → Roteamento batata-quente: escolha o *gateway* que tem o menor custo. → determine pela tabela de roteamento a interface **I** de próximo salto até o *gateway* de menor custo. Adicione uma entrada `(x, I)` à tabela.

Resumo da Aula...

- **Roteamento baseado em Vetor de Distâncias**:
  - Ideia: melhor caminho até destino composto por enlace até vizinho e melhor caminho do vizinho até destino.
  - Nós anunciam suas **estimativas** de custo até cada destino.
  - Ao **receber** novas estimativas, nó **atualiza** suas próprias.
  - Processo iterativo, **converge para melhores rotas**.
  - Algoritmo **distribuído**: nós precisam conhecer apenas vizinhança.
- **Contagem ao infinito:**
  - Potencial problema, ocorre em caso de grandes pioras nos custos dos enlaces.
  - Solução (parcial): **envenenamento reverso**.
- **Roteamento Hierárquico:**
  - Dois níveis: dentro e fora de **Sistemas Autônomos**.
    - Intra-AS e Inter-AS.
  - Tabela de roteamento construída por **colaboração dos dois processos**.
  - Reduz escopo, complexidade do roteamento.
  - **Nem sempre é globalmente ótimo!**
- **Roteamento batata-quente:**
  - Tirar datagrama do AS o mais rápido possível.

Próxima Aula...

- Dividiremos o problema de roteamento na Internet em duas partes:
  - Roteamento Intra-AS.
  - Roteamento Inter-AS.
- Falaremos sobre alguns protocolos de roteamento usados na Internet:
  - RIP, OSPF (Roteamento Intra-AS).
  - BGP (Roteamento Inter-AS).