The Network Layer: Routing Algorithms

CS 352, Lecture 16, Spring 2020

http://www.cs.rutgers.edu/~sn624/352

Srinivas Narayana

RUTGERS

UNIVERSITY | NEW BRUNSWICK

Course announcements

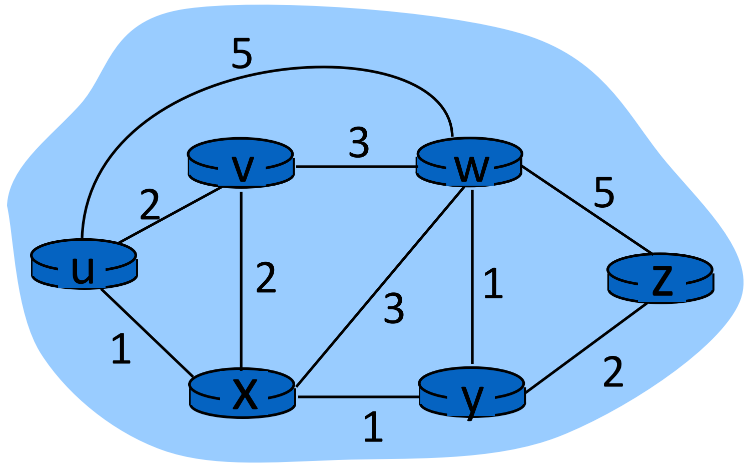
- Mid-term format: out of 45 points
  - 30 points: multiple choice
  - 8 points: short reasoning-style questions
  - 7 points: calculation-style questions

Review of concepts

- Network Address Translation
  - Use one external IP address for many endpoints
  - Translation table maps (WAN IP, WAN port) ←→ (LAN IP, LAN port)
- IP version 6 (IPv6)
  - Large address space (2^128)
  - No fragmentation, no options fields
  - Flow labels
  - Deployment is taking a looong time: architectural “ossification”
- Routing protocols: find a path for packets from src to dst
  - Control plane function
  - Graph abstraction
  - Link-state protocols: flood, then compute shortest paths (e.g., Dijkstra’s algorithm)

Dijkstra’s algorithm: example

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					

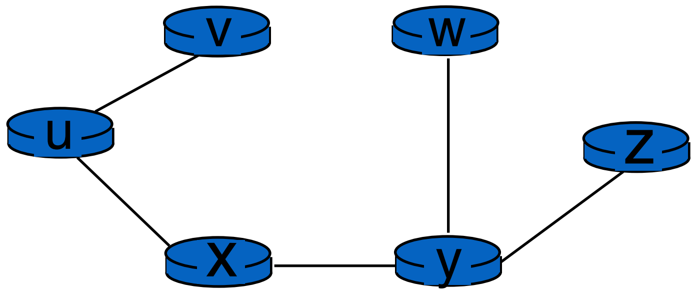


D(n) := current estimate of shortest path length from u to node n.

P(n) := node preceding n in the estimated shortest path from u to n, i.e. the “predecessor” of n.

Constructing forwarding table

resulting shortest-path tree from u:



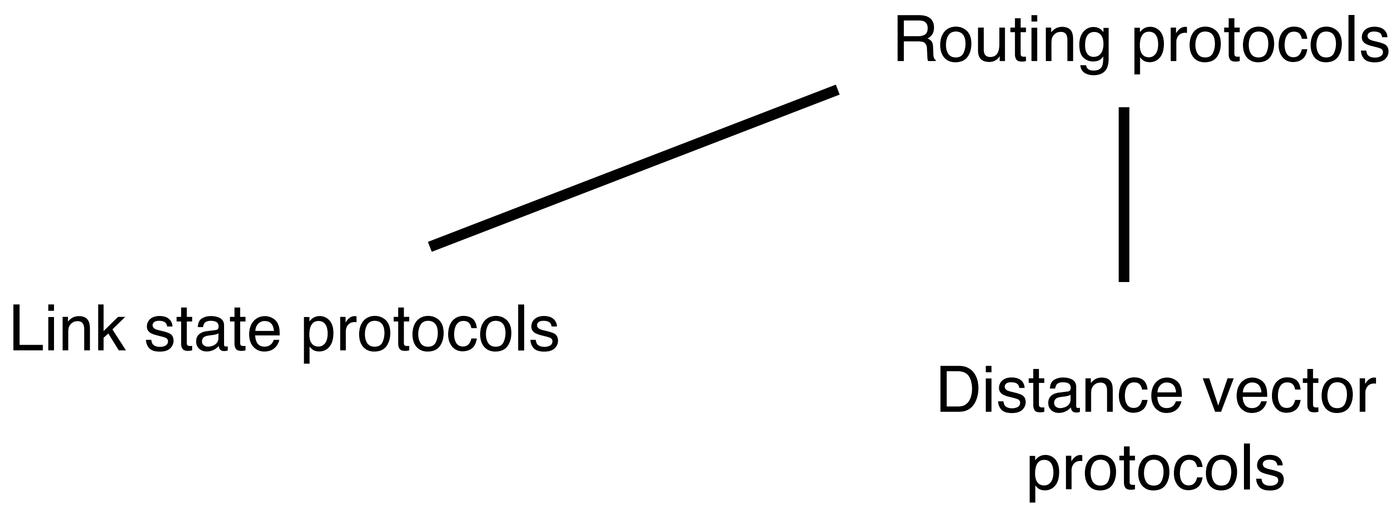
resulting forwarding table in u:

destination	link
v	(u,v)
x	(u,x)
y	(u,x)
w	(u,x)
z	(u,x)

Poll #1

- Link-state information of a router is sent to all routers before computing shortest paths in a link-state protocol.
  - (a) true
  - (b) false

Routing Protocols



Distance Vector Protocols

Distance vector protocol

- $D_x(y)$ = estimate of least cost from x to y
- Distance vector: $\mathbf{D}_x = [D_x(y): y \in N]$
- Node x knows cost of edge to each neighbor v: $c(x,v)$
- Node x maintains $\mathbf{D}_x$
- Node x also maintains its neighbors' distance vectors
  - For each neighbor v, x maintains
    $\mathbf{D}_v = [D_v(y): y \in N]$

Distance vector protocol

Bellman-Ford equation (dynamic programming)

let

$d_x(y)$:= cost of least-cost path from x to y

then

$$d_x(y) = \underbrace{\min_v}_{\text{\textit{min} taken over all neighbors v of x}} \{\underbrace{c(x,v)}_{\text{cost to neighbor v}} + \underbrace{d_v(y)}_{\text{cost from neighbor v to destination y}} \}$$

Distance vector protocol

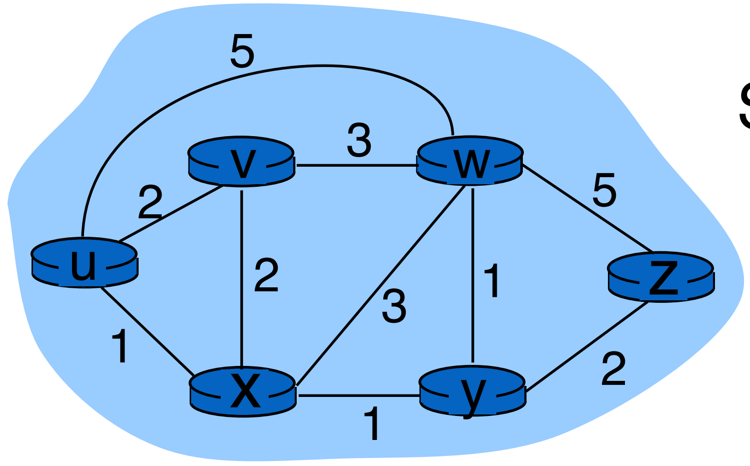
Basic idea:

- Each node periodically sends its own distance vector estimate to neighbors
- When node a node x receives new DV estimate from neighbor, it updates its own DV using Bellman-Ford equation:

$$D_x(y) \leftarrow \min_v\{c(x,v) + D_v(y)\} \quad \textit{for each node } y \in N$$

- Under some conditions, the estimate $D_x(y)$ *converge the actual least cost* $d_x(y)$

Distance vector: example



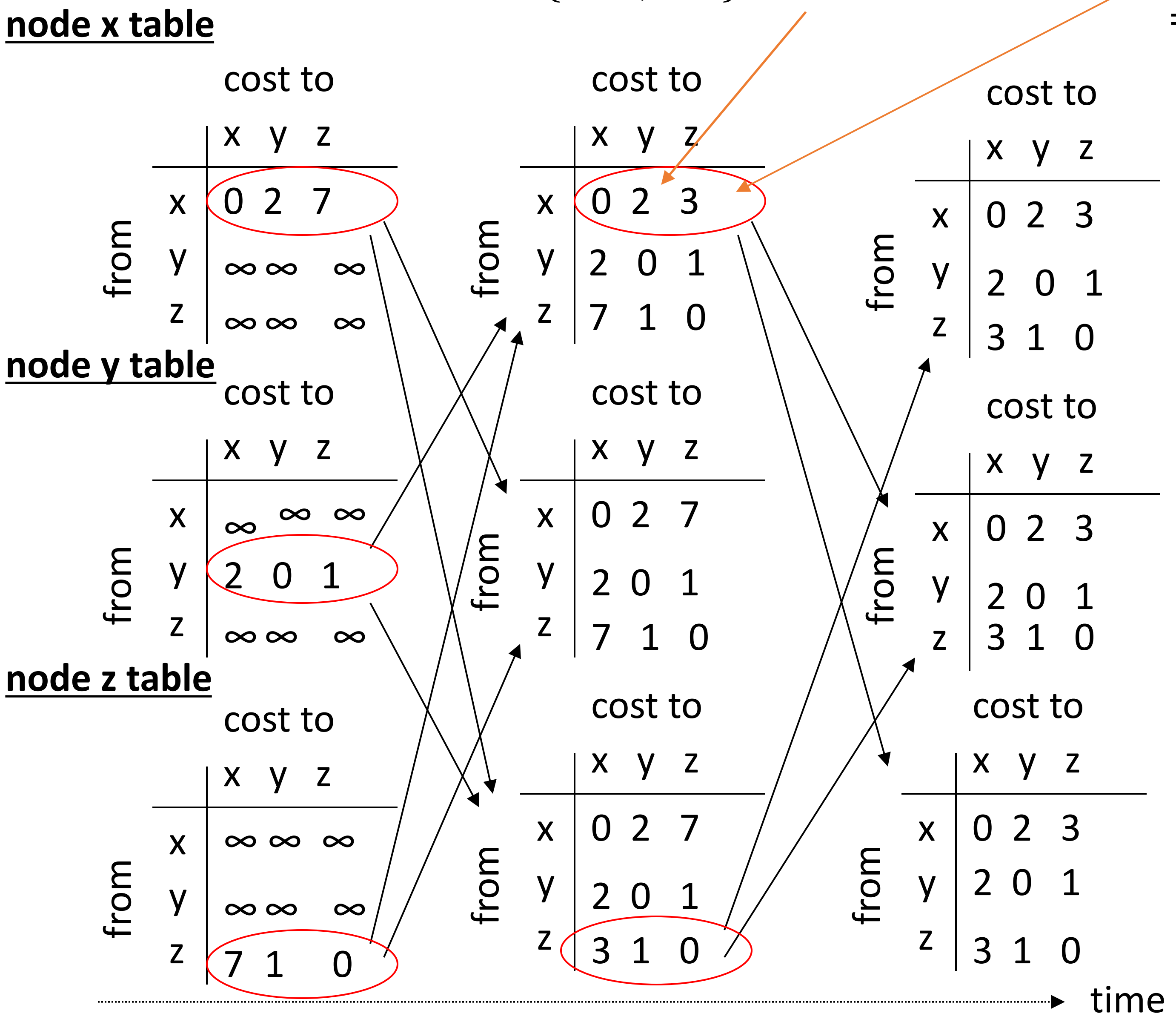
Start with $d_v(z) = 5$, $d_x(z) = 3$, $d_w(z) = 3$

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

Node that achieves minimum is next hop in shortest path ➜ forwarding table

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

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



node x table

from \ cost to	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

from \ cost to	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

from \ cost to	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

node y table

from \ cost to	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

from \ cost to	x	y	z
x	0	2	7
y	2	0	1
z	7	1	0

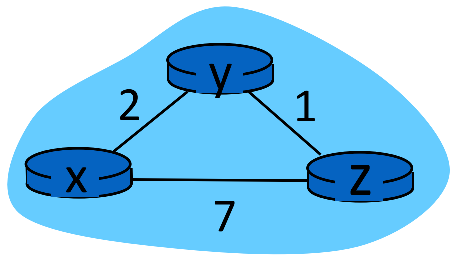
from \ cost to	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

node z table

from \ cost to	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0

from \ cost to	x	y	z
x	0	2	7
y	2	0	1
z	3	1	0

from \ cost to	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0



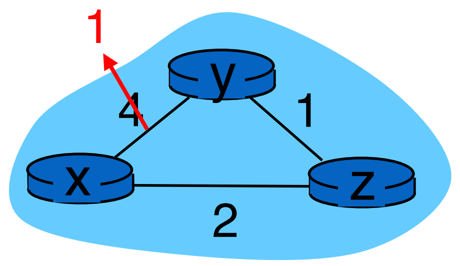
Poll #2

- Suppose v is the node that minimizes $c(x,v) + d_v(y)$ for a fixed destination y.
- The forwarding table entry for y at x chooses the link:
  - (a) (x, y)
  - (b) (x, v)
  - (c) (x, x)
  - (d) (y, v)

Distance vector: link cost changes

link cost changes:

- node detects local link cost change
- updates routing info, recalculates distance vector
- if DV changes, notify neighbors



"good news travels fast"

t_0 : *y* detects link-cost change, updates its DV, informs its neighbors.

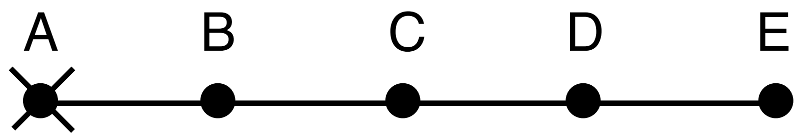
t_1 : *z* receives update from *y*, updates its table, computes new least cost to *x* , sends its neighbors its DV.

t_2 : *y* receives *z*'s update, updates its distance table. *y*'s least costs do *not* change, so *y* does *not* send a message to *z*.

Problem: Count-to-Infinity

- With distance vector routing, good news travels fast, but bad news travels slowly
- When a router goes down, it takes can take a long time before all other routers become aware of it

Count-to-Infinity



B	C	D	E	
1	2	3	4	Initially
3	2	3	4	After 1 exchange
3	4	3	4	After 2 exchanges
5	4	5	4	After 3 exchanges
5	6	5	6	After 4 exchanges
7	6	7	6	After 5 exchanges

etc… to infinity

Count-to-infinity

"Bad news travels slowly"

More precisely, reacting appropriately to bad news requires information that only other routers have.

Poisoned reverse:

- If Z routes through Y to get to X :
  - Z tells Y its (Z's) distance to X is infinite (so Y won't route to X via Z)

- Do you think this would completely solve count to infinity problem?

Comparison of LS and DV algorithms

message complexity

- ***LS:*** with n nodes, E links, O(nE) msgs sent
- ***DV:*** exchange between neighbors only
  - convergence time depends on how many iterations of Bellman-Ford occur, updating the DV

speed of convergence

- ***LS:*** O(n²) algorithm requires O(nE) msgs
- ***DV:*** convergence time varies
  - may be routing loops (count-to-infinity problem)

robustness: what happens if router malfunctions?

LS:

- node can advertise incorrect *link* cost
- each node computes only its *own* table

DV:

- DV node can advertise incorrect *path* cost
- each node's table used by others
  - error propagate thru network

Poll #3

- Which routing protocol(s) ensure that routers have views of the network that are always consistent with each other?
  - (a) Link-state protocols
  - (b) Distance-vector protocols
  - (c) All of the above
  - (d) None of the above

Routing protocols are widely deployed

- Real protocols using a link-state algorithm
  - OSPF “Open Shortest Path First”
  - IS-IS: “Intermediate System to Intermediate System”
- Real protocols using a distance-vector algorithm
  - RIP: “Routing Information Protocol”
    - RFC written by our own Chuck Hedrick from LCSR @ Rutgers CS
  - IGRP: “Interior Gateway Routing Protocol”
- This live lecture is being transmitted over networks deploying one or more of these protocols to compute forwarding tables

Scaling Routing to the Internet

Making routing scalable

our routing study thus far - idealized

- all routers identical
- network "flat"

… *not* true in practice

scale: with billions of destinations:

- can't store all destinations in routing tables!
- routing table exchange would swamp links!

administrative autonomy

- Internet = network of networks
- each network admin may want to control routing in its own network

Internet's approach to scalable routing

aggregate routers into regions known as "autonomous systems" (AS) (a.k.a. "domains")

intra-AS routing

- routing among hosts, routers in same AS ("network")
- all routers in the same AS must run *same* intra-domain protocol
- routers in *different* AS can run *different* intra-domain routing protocols
- gateway router: at "edge" of its own AS, has link(s) to router(s) in other AS'es

inter-AS routing

- routing among AS'es
- gateways perform inter-domain routing as well as intra-domain routing
- All networks run the same inter-domain routing protocol

Making routing scalable

Key principle: Hierarchy
… by separation into ASes
… within ASes, using intra-domain areas.

Hierarchy enables autonomy of separate regions.

Compare: federal -> state -> district -> …

Routing protocols

Link state protocols
e.g., OSPF, IS-IS

Distance vector protocols
e.g., RIP, IGRP

Path vector protocols
e.g., BGP

Intra-AS protocols

- same protocol within an AS
- different algorithms across ASes
- (semi-)global view of the network
- Also called interior gateway protocols (IGP)

Inter-AS protocols

- common across Ases
- each AS knows little about the others

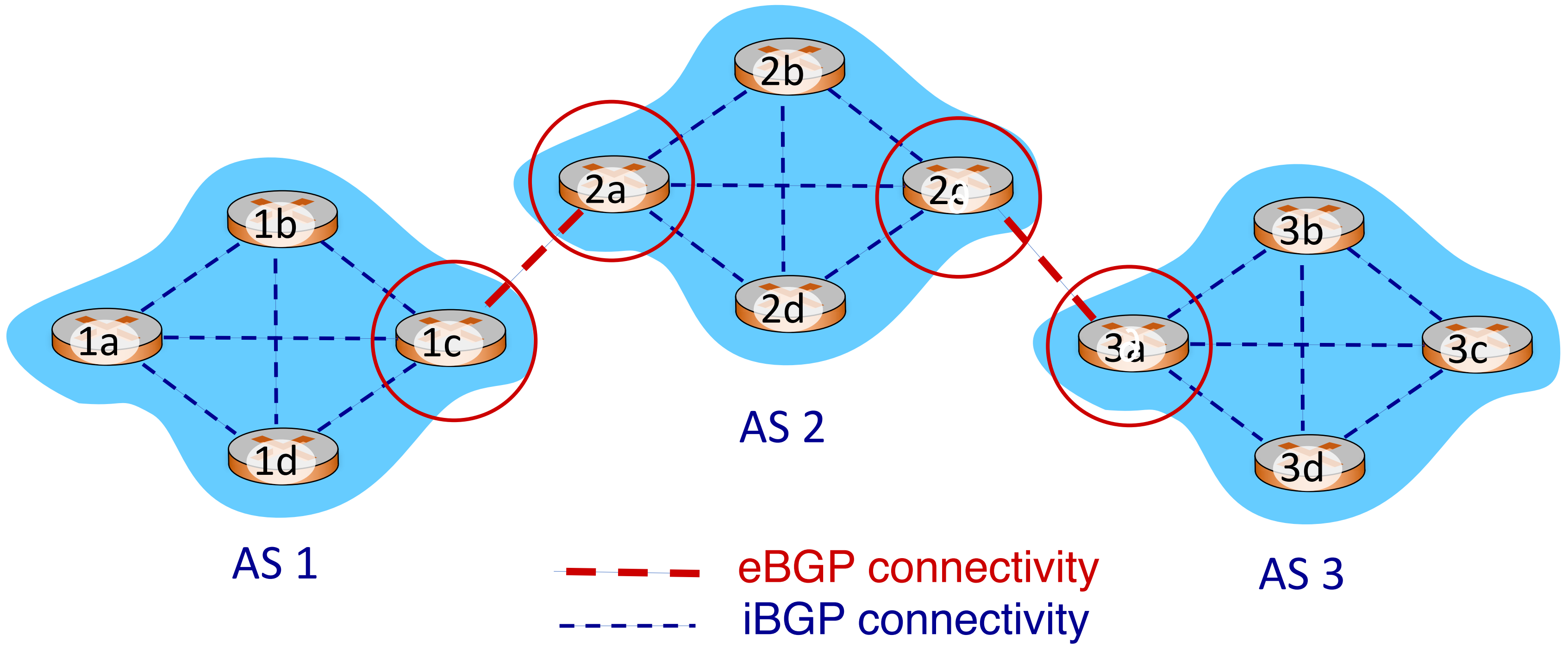
Border Gateway Protocol (BGP)

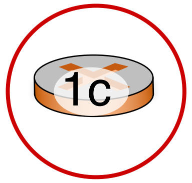
The glue that holds the Internet together

Internet inter-AS routing: BGP

- BGP (Border Gateway Protocol): *the* de facto inter-domain routing protocol
- BGP provides each a way to:
  - eBGP: obtain subnet reachability information from neighboring ASes
  - iBGP: propagate reachability information to all AS-internal routers.
  - determine "good" routes to other networks based on reachability information and *policy*
- eBGP allows a subnetwork to advertise its existence to the rest of Internet:
  - *"I am here"*

eBGP, iBGP connections

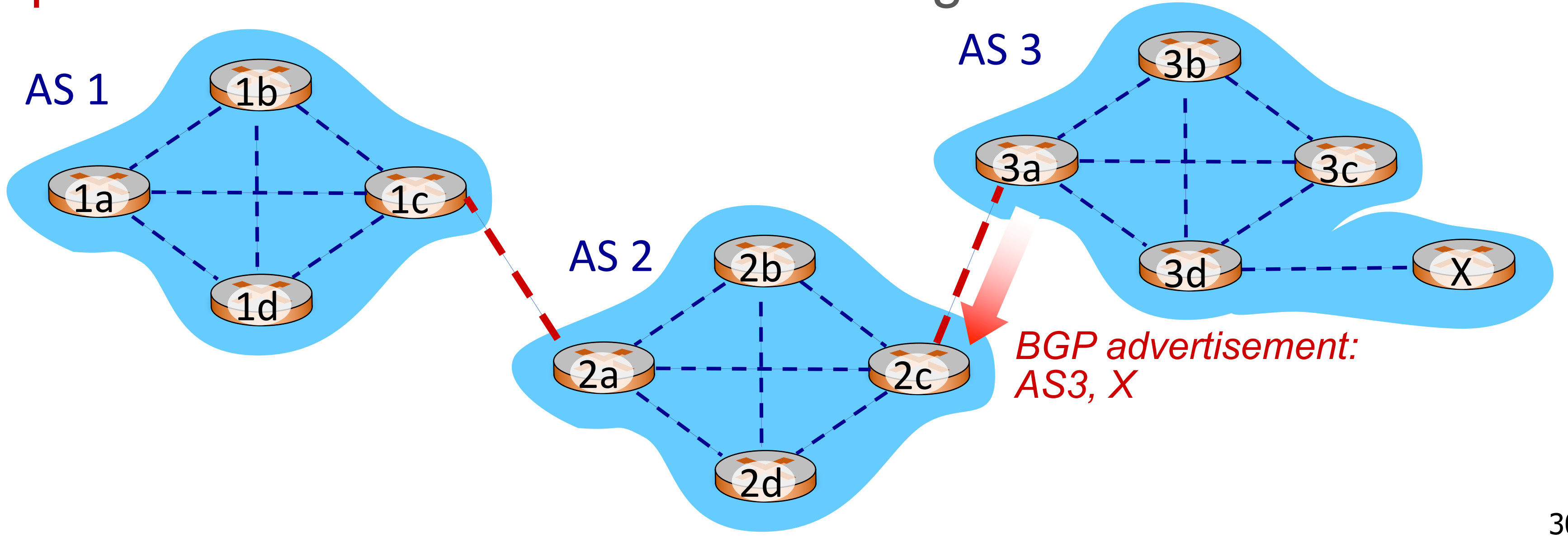




gateway routers run both eBGP and iBGP protools

BGP basics

- **BGP session:** two BGP routers ("peers") exchange BGP messages over semi-permanent TCP connection:
  - advertising paths to different destination network prefixes
  - BGP is a path vector protocol
- When AS3 gateway router 3a advertises path AS3,X to AS2 gateway router 2c,
  - AS3 promises to AS2 it will forward datagrams towards X



Poll #4

- A border gateway router speaks the following protocols.
  - (a) IGP
  - (b) eBGP
  - (c) iBGP
  - (d) All of the above

Poll #5

- Routers inside an AS learn about destinations outside the AS using
  - (a) IGP
  - (b) eBGP
  - (c) iBGP
  - (d) None of the above

Path attributes and BGP routes

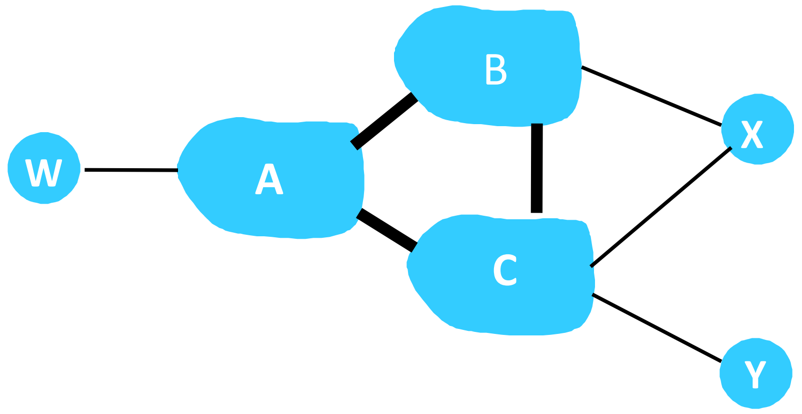
- advertised prefix includes BGP attributes
  - Advertisement of a route = prefix + attributes
- Two important attributes:
  - AS-PATH: list of ASes through which prefix advertisement has passed
  - NEXT-HOP: indicates specific internal-AS router to next-hop AS
- Policy-based routing:
  - gateway receiving route advertisement uses import policy to accept/decline path (e.g., never route through AS Y).
  - AS export policy also determines whether to advertise a path to other other neighboring ASes

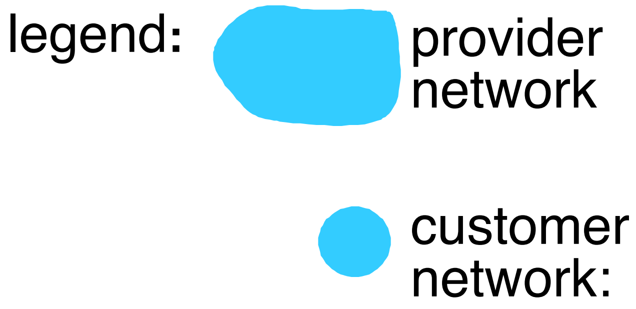
Policies in BGP

Policy comes from business relationships

- Customer-provider relationships:
  - E.g., Rutgers is a customer of AT&T
- Peer-peer relationships:
  - E.g., Verizon is a peer of AT&T
- Business relationships depend on where connectivity occurs
  - “Where”, also called a “point of presence” (PoP)
  - E.g., customers at one PoP but peers at another
- Sometimes, even when there is no direct connectivity
  - E.g., inteliquent (zoom/webex) traffic not to be charged, acc. to the FCC
- Internet-eXchange Points (IXPs) are large PoPs where ISPs come together to connect with each other (often for free)

BGP Export Policy and Advertisements

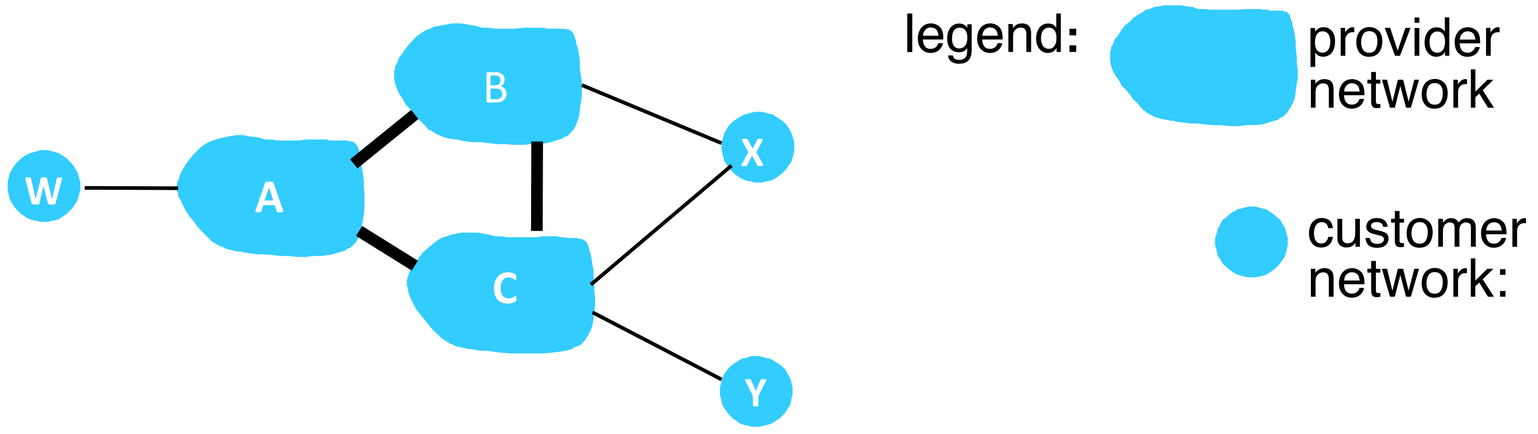




Suppose an ISP only wants to route traffic to/from its customer networks (does not want to carry transit traffic between other ISPs)

- A,B,C are *provider networks*
- X,W,Y are customer (of provider networks)
- X is *dual-homed:* attached to two networks
- *policy to enforce:* X does not want to route from B to C via X
  - .. so X will not advertise to B a route to C

BGP Export Policy and Advertisements



Suppose an ISP only wants to route traffic to/from its customer networks (does not want to carry transit traffic between other ISPs)

- A advertises path Aw to B and to C
- B *chooses not to advertise* BAw to C:
  - B gets no "revenue" for routing CBAw, since none of C, A, w are B's customers
  - C does not learn about CBAw path
- C will route CAw (not using B) to get to w

Policies make BGP a complex protocol.

Advertise entire paths, not just local info (like link state or distance vectors).

Choose to advertise (export) only certain paths.

Choose to accept (import) only certain paths.

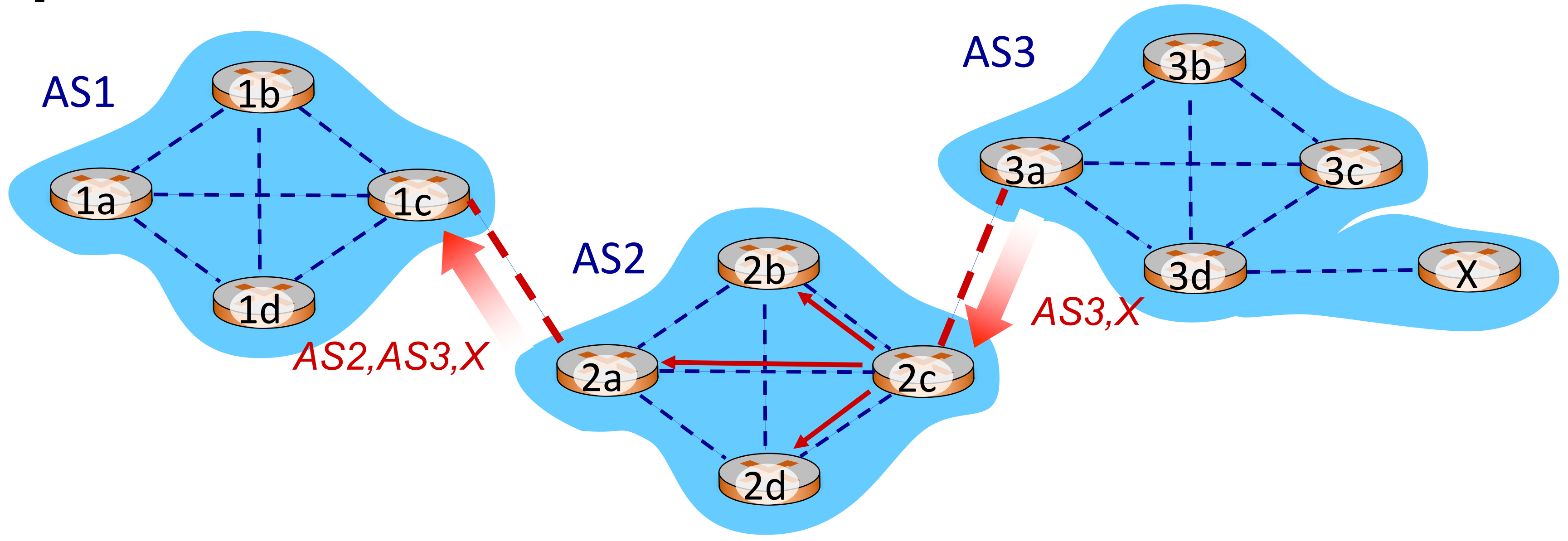
Complex decision process to prefer certain imported paths over others.

Poll #6

- eBGP is a
  - (a) link-state protocol
  - (b) distance-vector protocol
  - (c) path-vector protocol
  - (d) None of the above

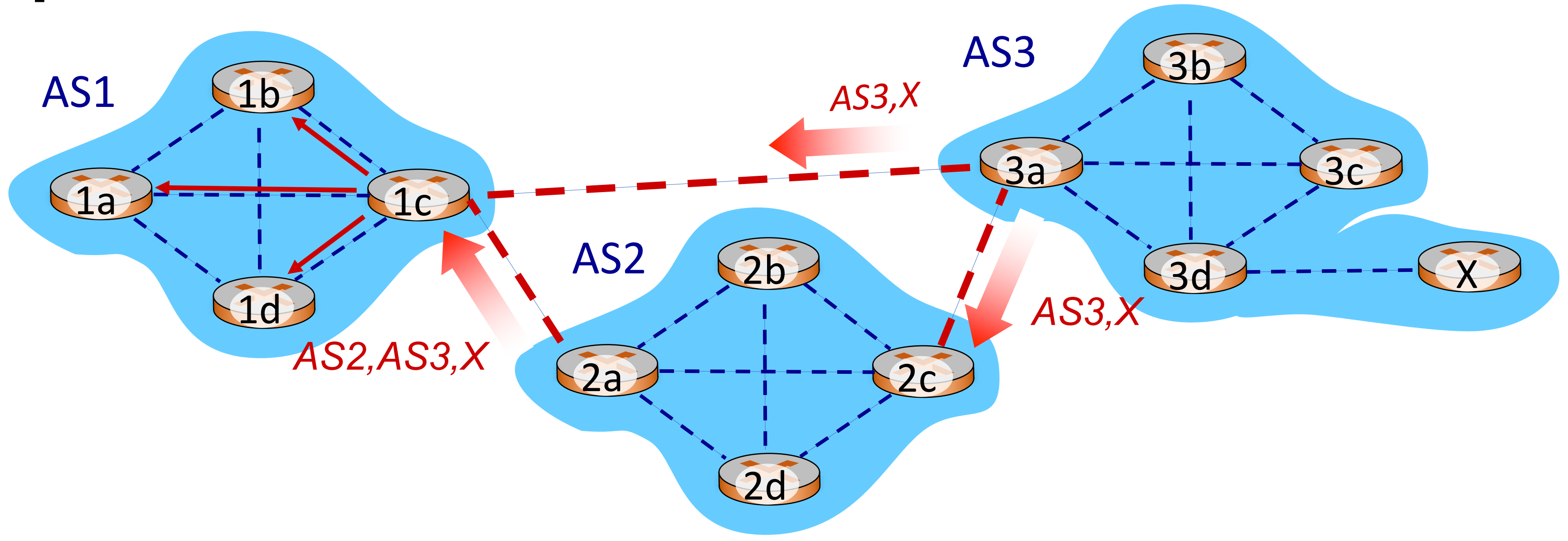
BGP Routing

BGP path advertisement



- AS2 router 2c receives path advertisement AS3,X (via eBGP) from AS3 router 3a
- Based on AS2 policy, AS2 router 2c accepts path AS3,X, propagates (via iBGP) to all AS2 routers
- Based on AS2 policy, AS2 router 2a advertises (via eBGP) path AS2, AS3, X to AS1 router 1c

BGP path advertisement



Gateway router may learn about multiple paths to destination:

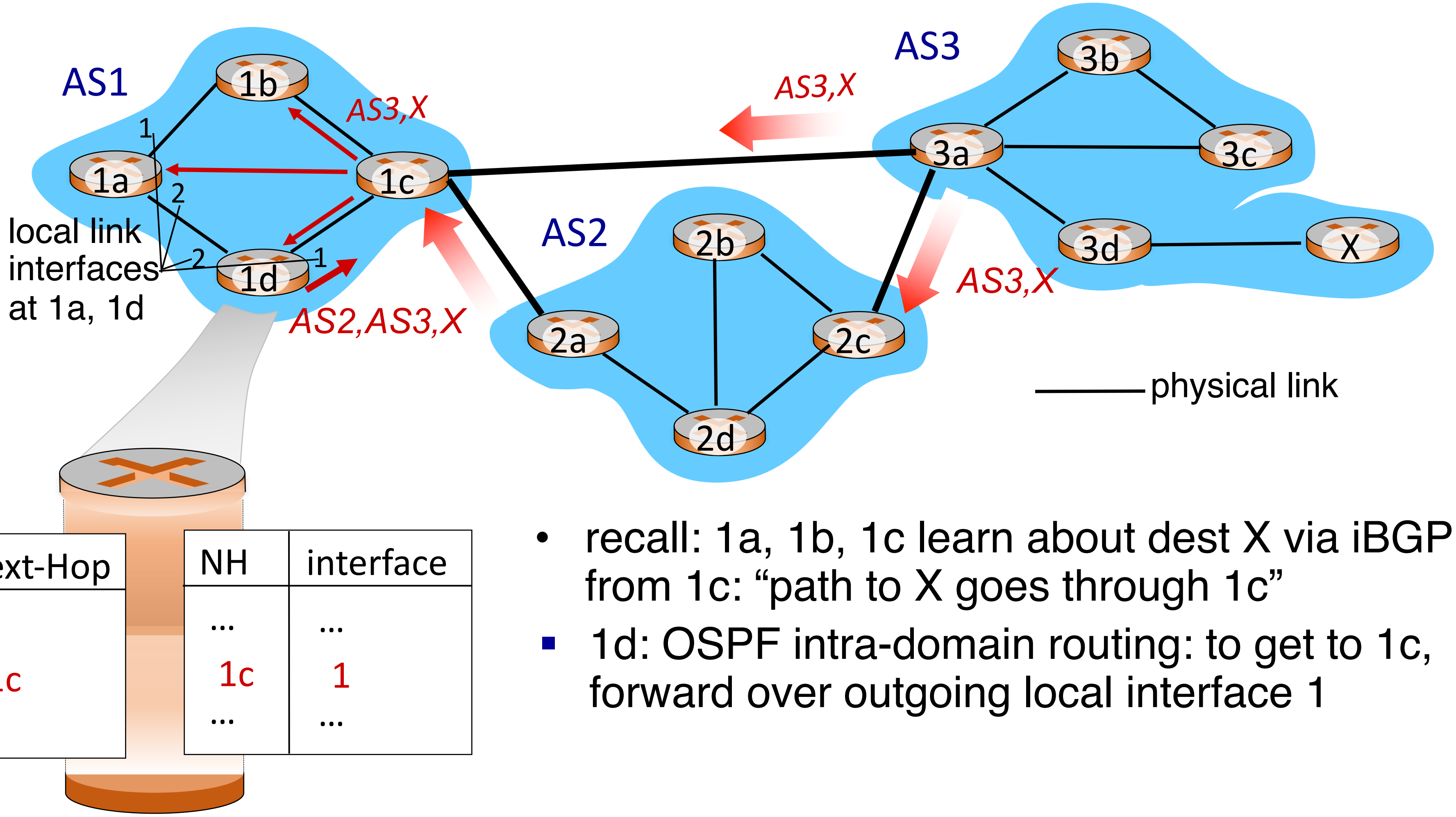
- AS1 gateway router 1c learns path *AS2,AS3,X* from 2a
- AS1 gateway router 1c learns path *AS3,X* from 3a
- Based on policy, AS1 gateway router 1c chooses path *AS3,X, and advertises path within AS1 via iBGP*

BGP messages

- BGP messages exchanged between peers over TCP connection
  - In principle, can establish BGP session with any router
    - Common, but not necessary, that routers are physically adjacent
- BGP messages:
  - OPEN: opens TCP connection to remote BGP peer and authenticates sending BGP peer
  - UPDATE: advertises new path (or withdraws old)
  - KEEPALIVE: keeps connection alive in absence of UPDATES; also ACKs OPEN request
  - NOTIFICATION: reports errors in previous msg; also used to close connection

BGP, OSPF, forwarding table entries

Q: how does router set forwarding table entry to distant prefix?



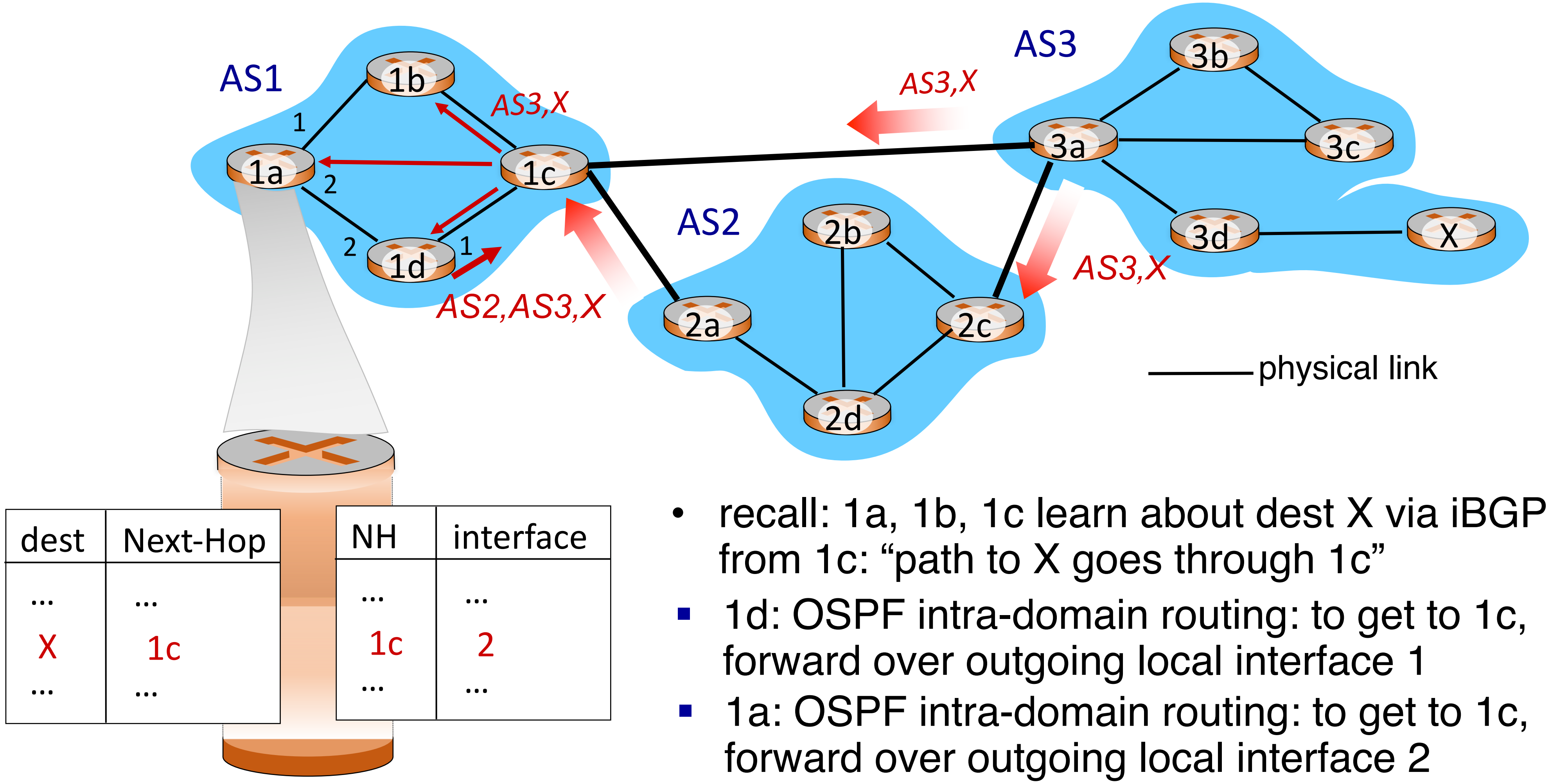
dest	Next-Hop
…	…
X	1c
…	…

NH	interface
…	…
1c	1
…	…

- recall: 1a, 1b, 1c learn about dest X via iBGP from 1c: "path to X goes through 1c"
- 1d: OSPF intra-domain routing: to get to 1c, forward over outgoing local interface 1

BGP, OSPF, forwarding table entries

Q: how does router set forwarding table entry to distant prefix?



dest	Next-Hop
…	…
X	1c
…	…

NH	interface
…	…
1c	2
…	…

- recall: 1a, 1b, 1c learn about dest X via iBGP from 1c: “path to X goes through 1c”
  - 1d: OSPF intra-domain routing: to get to 1c, forward over outgoing local interface 1
  - 1a: OSPF intra-domain routing: to get to 1c, forward over outgoing local interface 2

Poll #7

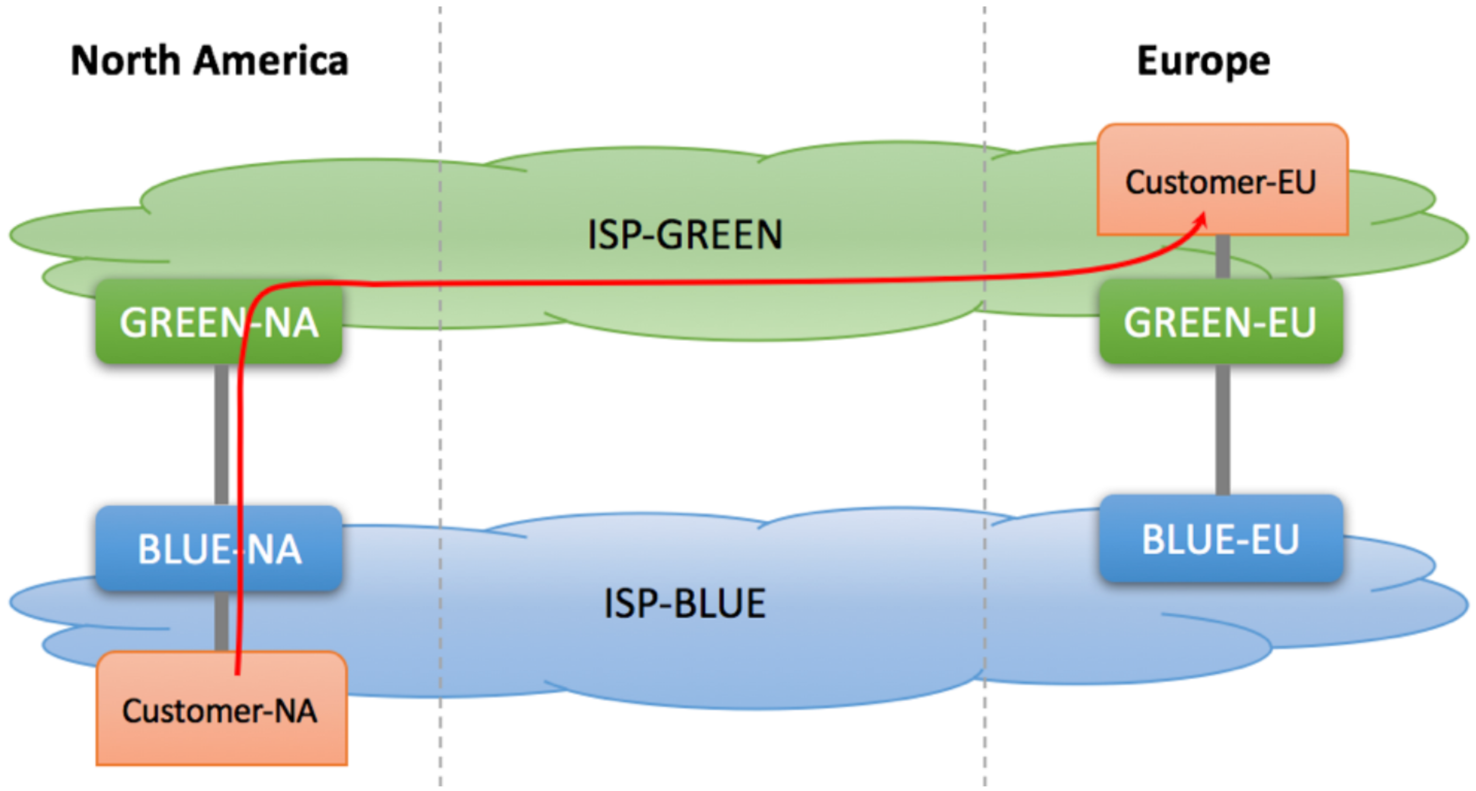
- Suppose an AS uses OSPF as its intra-domain routing protocol.
- Forwarding table entries on AS-internal routers towards destinations outside the AS are computed using information from
  - (a) iBGP
  - (b) OSPF
  - (c) both iBGP and OSPF
  - (d) None of the above

BGP route selection process

- Router may learn about more than one route to destination AS, selects route based on:
  1. local preference value attribute (policy decision)
  2. shortest AS-PATH
  3. closest NEXT-HOP router: “hot potato” routing
  4. additional criteria

You can read up on the full, complex, list of criteria, e.g., at https://www.cisco.com/c/en/us/support/docs/ip/border-gateway-protocol-bgp/13753-25.html

Hot-Potato Routing



BGP Hot Potato Routing

Also called early-exit routing

Choose the “next-hop” router that is closest based on intra-AS routing

Reduces utilization on resources inside the AS

Source: http://bgphelp.com/2017/04/25/hot-potato-vs-cold-potato-routing/

Why different Intra-, Inter-AS routing?

policy:

- inter-AS: admin wants control over how its traffic routed, who routes through its net.
- intra-AS: single admin, so no policy decisions needed

scale:

- hierarchical routing saves table size, reduced update traffic

performance:

- intra-AS: can focus on performance
- inter-AS: policy may dominate over performance

Network layer: the big picture

- The network layer provides connectivity between Internet hosts
  - Split into control plane and data plane
- Data plane: the IP protocol
  - Supported by DHCP, ICMP, NATs
  - Routers implement data plane through ports + fabric + queues
- Control plane: routing protocols
  - Link state: flooding + centralized information + independent computations across routers
  - Distance vector: neighbor exchange + decentralized + dependent computations across routers
  - Path vector: flooding + decentralized + policy-based dependent computations across routers