



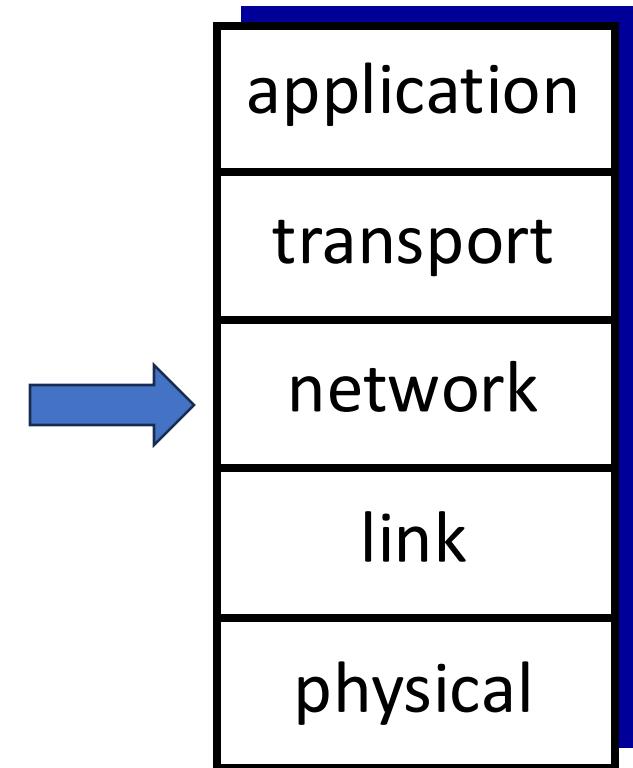
Networks (2IRR20)

Network Layer – Control Plane (05)

Dr. Tanir Ozcelebi

This slide set

- Routing protocols
 - link state
 - distance vector
- Intra-ISP (intra-AS) routing: OSPF
- Inter-ISP (inter-AS) routing: BGP
- Internet Control Message Protocol



Network layer key functions

(recap)

Network-layer functions

- **forwarding:** move packets from router's input to appropriate router output
- **routing:** determine route taken by packets from source to destination

data plane

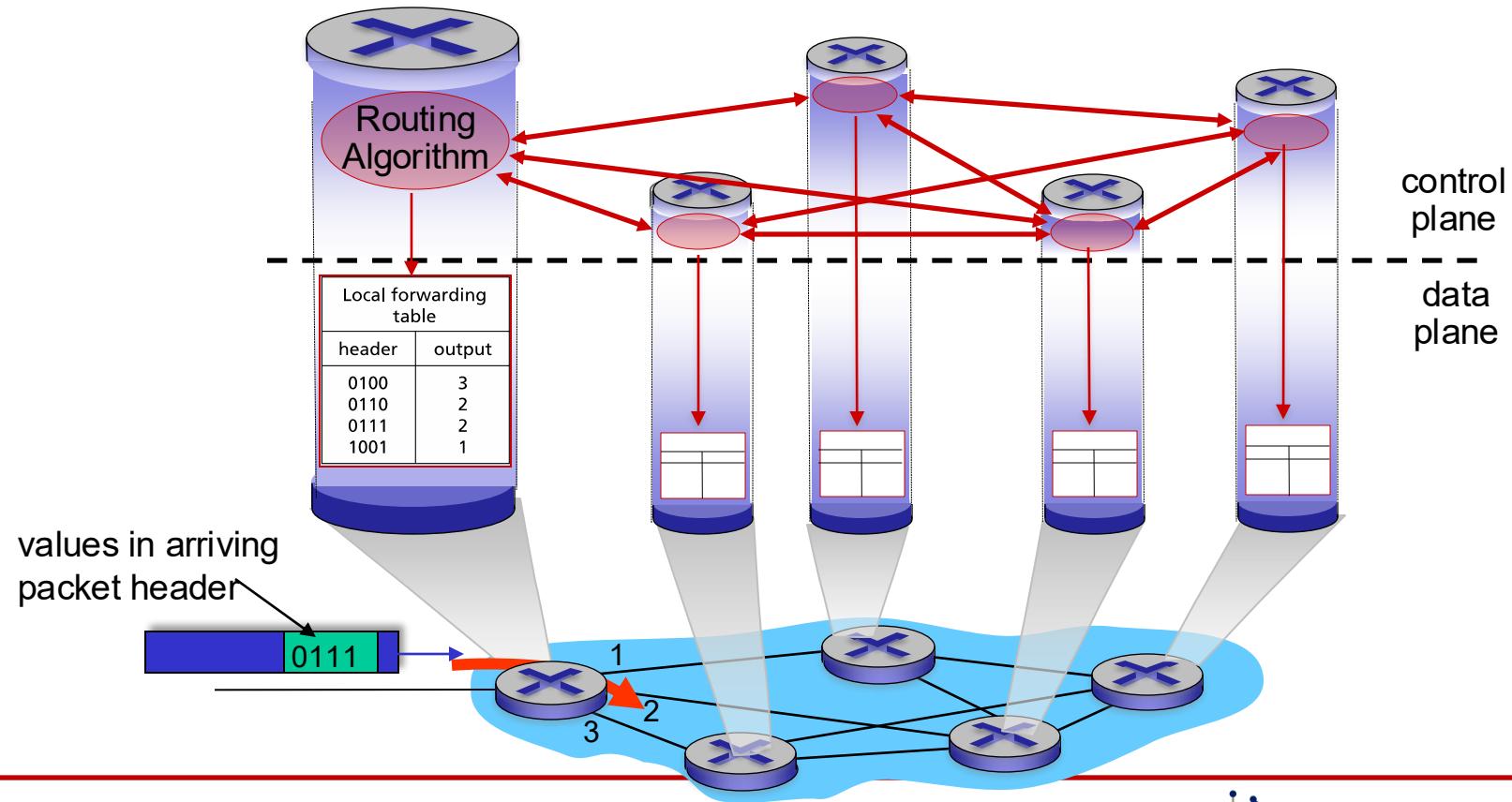
control plane

Two approaches to structuring network control plane:

- per-router control (traditional)
- logically centralized control (software defined networking)

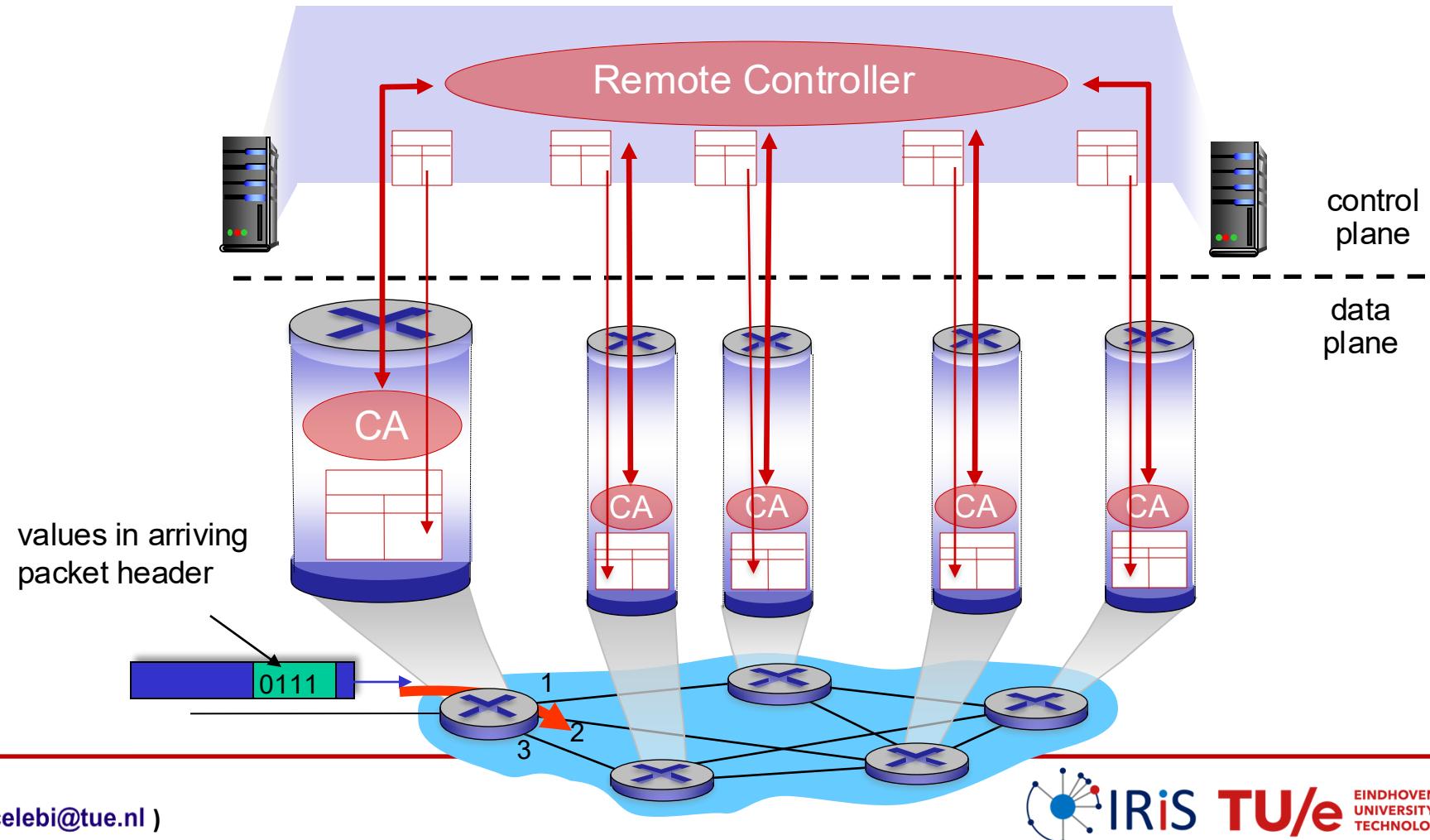
Per-router control plane

Components of a distributed routing algorithm (*in each and every router*) interact/coordinate in the control plane.



Software-Defined Networking (SDN) control plane

Remote controller computes, installs forwarding tables in routers.

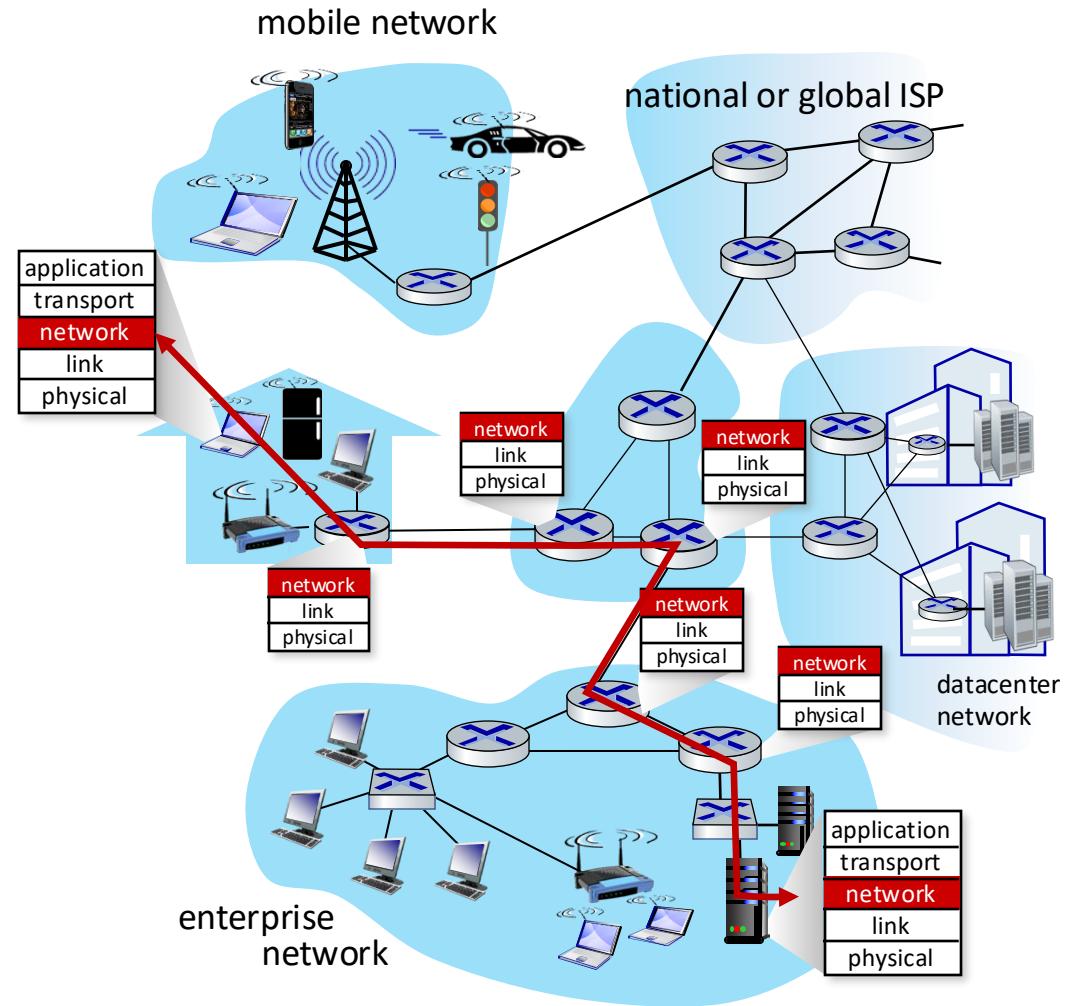


Routing protocols

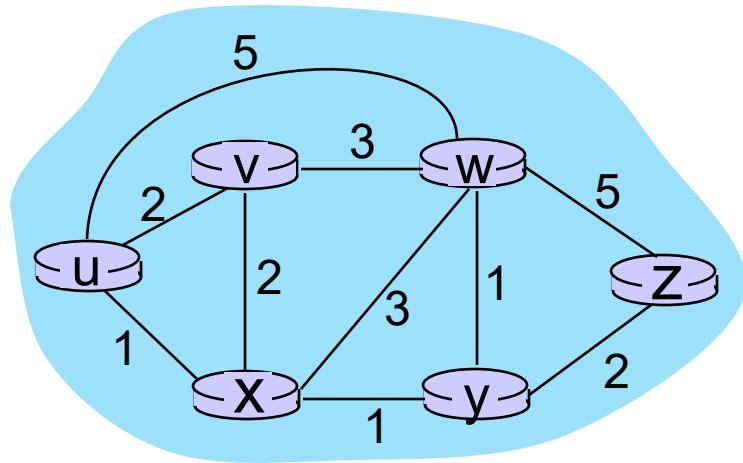
Routing protocols

Routing protocol goal: determine “good” end-to-end routes.

- route/path:
 - sequence of routers packets traverse
- “good”: least “cost” route
 - “fastest”, “least congested”, etc.



Graph abstraction: link costs



$c_{a,b}$: cost of *direct* link connecting a and b

e.g., $c_{w,z} = 5$, $c_{u,z} = \infty$

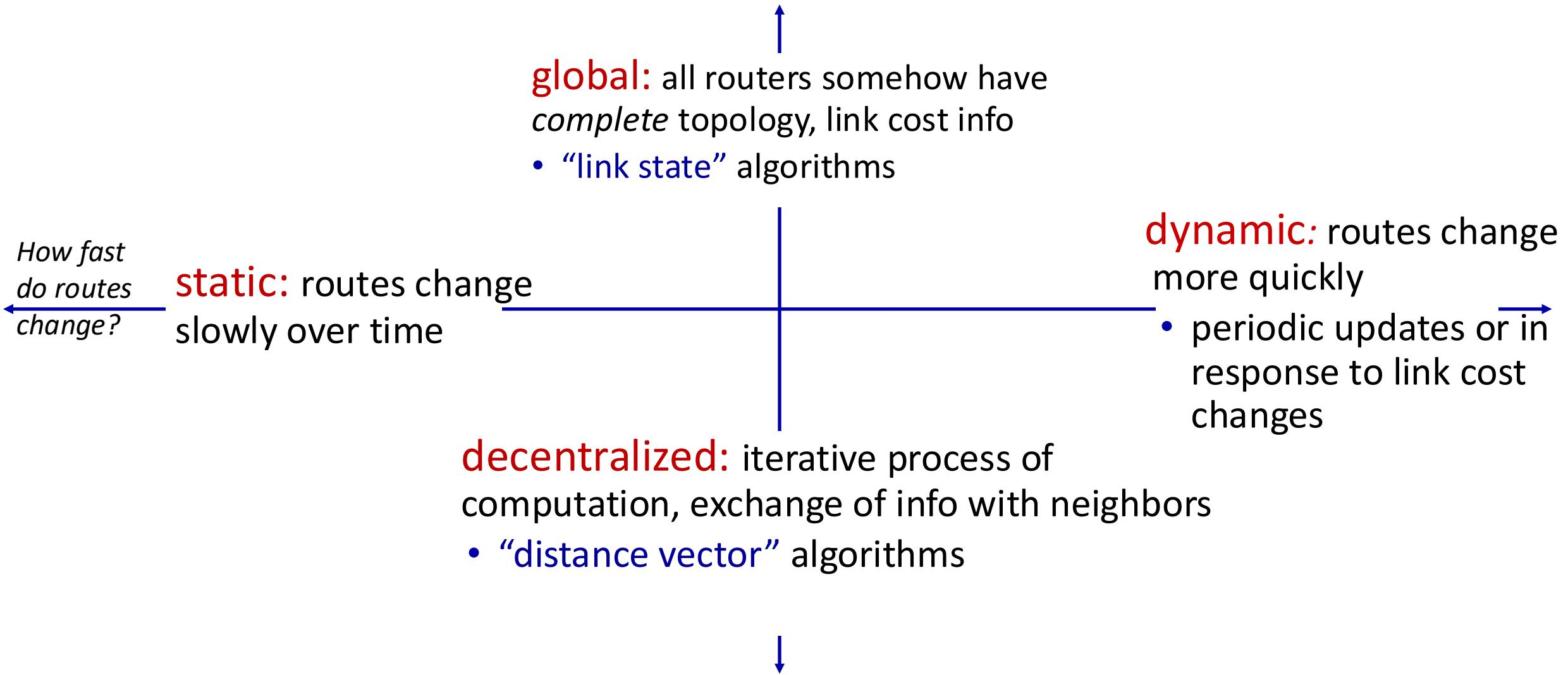
cost defined by network operator:
could always be 1, or inversely related
to bandwidth, or related to congestion

graph: $G = (N, E)$

N : set of routers = { u, v, w, x, y, z }

E : set of links = { $(u,v), (u,x), (v,x), (v,w), (x,w), (x,y), (w,y), (w,z), (y,z)$ }

Routing algorithm classification



Link state routing

Dijkstra's link-state routing algorithm

- **centralized:** network topology, link costs known to *all* nodes
 - via “link state broadcast”
 - all nodes have same info
- computes least cost paths from one node (“source”) to all other nodes
 - gives forwarding table for that node
- **iterative:** after k iterations, know least cost path to k destinations

notation

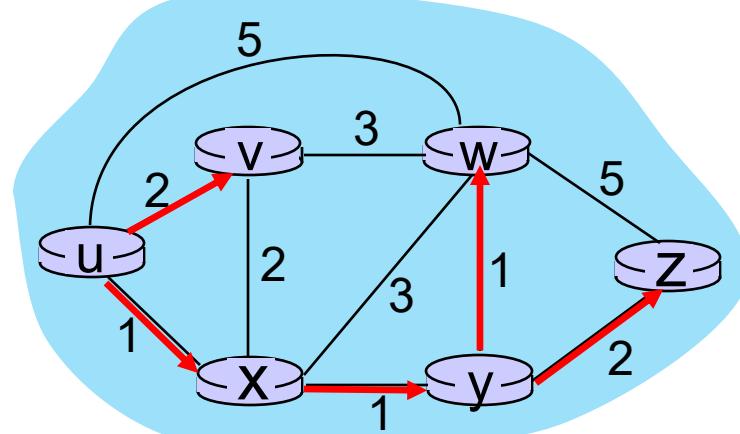
- $c_{x,y}$: direct link cost from node x to y ; $= \infty$ if not direct neighbors
- $D(v)$: *current* estimate of cost of least-cost-path from source to destination v
- $p(v)$: predecessor node along path from source to v
- N' : set of nodes whose least-cost-path *definitively* known

Dijkstra's link-state routing algorithm

```
1 Initialization:
2    $N' = \{u\}$                                 /* compute least cost path from u to all other nodes */
3   for all nodes  $v$ 
4     if  $v$  adjacent to  $u$                       /*  $u$  initially knows direct-path-cost only to direct neighbors */
5       then  $D(v) = c_{u,v}$                       /* but may not be minimum cost!
6     else  $D(v) = \infty$ 
7
8 Loop
9   find  $w$  not in  $N'$  such that  $D(w)$  is a minimum
10  add  $w$  to  $N'$ 
11  update  $D(v)$  for all  $v$  adjacent to  $w$  and not in  $N'$ :
12     $D(v) = \min(D(v), D(w) + c_{w,v})$ 
13  /* new least-path-cost to  $v$  is either old least-cost-path to  $v$  or known
14  least-cost-path to  $w$  plus direct-cost from  $w$  to  $v$  */
15 until all nodes in  $N'$ 
```

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	u, x	2, u	4, x	2, x	∞	∞
2	u, x, y	2, u	3, y	4, y	∞	4, y
3	u, x, y, v	3, y	∞	4, y	∞	4, y
4	u, x, y, v, w					4, y
5	u, x, y, v, w, z					

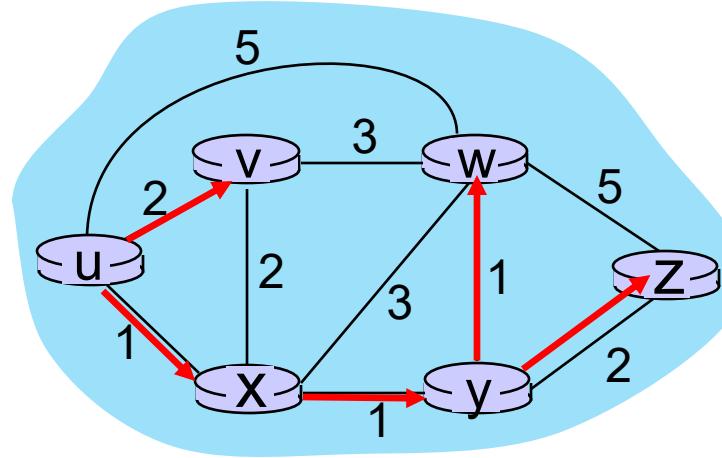


Initialization (step 0): For all a : if a adjacent to then $D(a) = c_{u,a}$

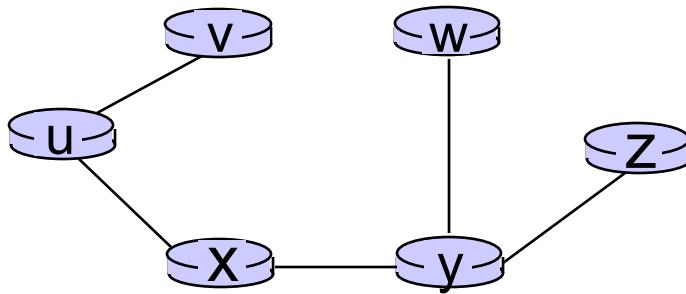
find a not in N' such that $D(a)$ is a minimum
add a to N'
update $D(b)$ for all b adjacent to a and not in N' :

$$D(b) = \min (D(b), D(a) + c_{a,b})$$

Dijkstra's algorithm example



resulting least-cost-path tree from u:



resulting forwarding table in u:

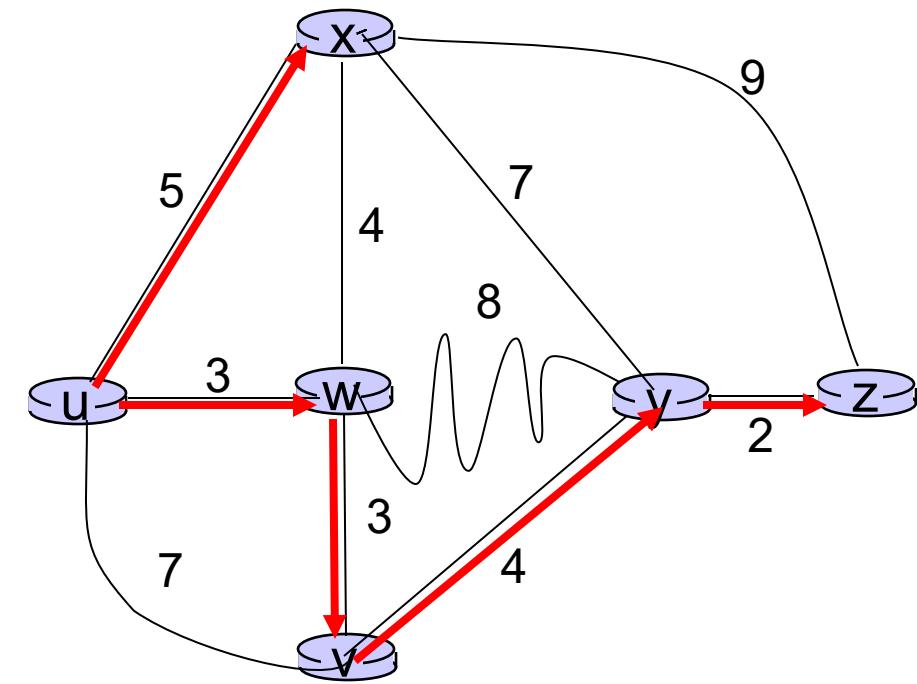
destination	outgoing link
v	(u,v)
x	(u,x)
y	(u,x)
w	(u,x)
z	(u,x)

route from u to v directly

route from u to all other destinations via x

Dijkstra's algorithm: another 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	7, u	3, u	5, u	∞	∞
1	uw	6, w	5, u	11, w	∞	
2	uwx	6, w		11, w	14, x	
3	uwxv		10, v	14, x		
4	uwxvy			12, y		
5	uwxvzy					

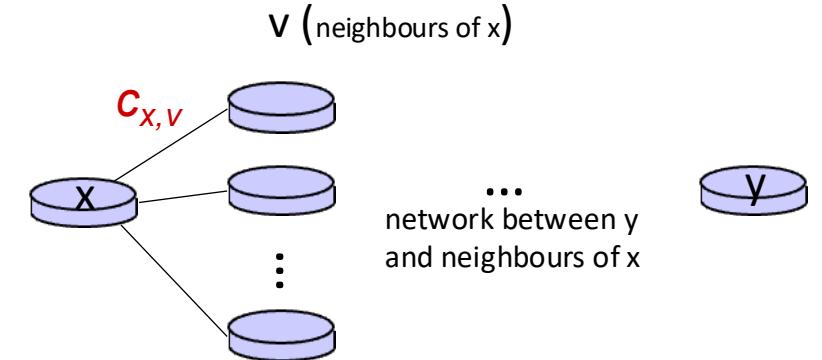


- ties can exist (can be broken arbitrarily)

Distance vector routing

Distance vector algorithm

Based on *Bellman-Ford* (BF) equation:



Bellman-Ford equation

Let $D_x(y)$: cost of least-cost path from x to y .

Then:

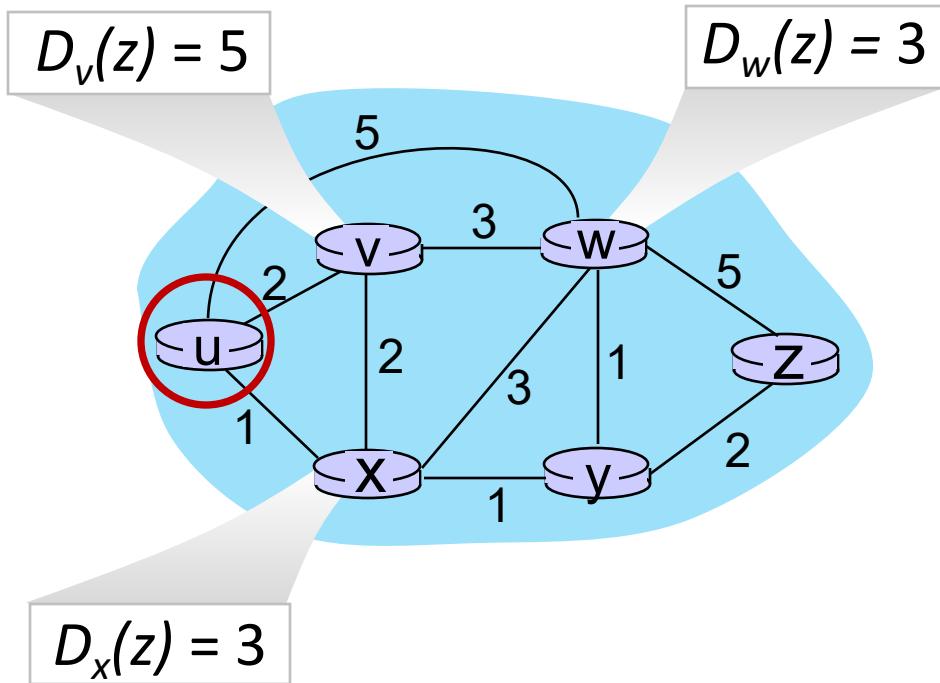
$$D_x(y) = \min_v \{ c_{x,v} + D_v(y) \}$$

\min taken over all neighbors v of x

v's estimated least-cost-path cost to y
direct cost of link from x to v

Bellman-Ford Example

Suppose that u 's neighboring nodes, x, v, w , know that for destination z :



Bellman-Ford equation says:

$$\begin{aligned} D_u(z) &= \min \{ c_{u,v} + D_v(z), \\ &\quad c_{u,x} + D_x(z), \\ &\quad c_{u,w} + D_w(z) \} \\ &= \min \{ 2 + 5, \\ &\quad 1 + 3, \\ &\quad 5 + 3 \} = 4 \end{aligned}$$

Node achieving the minimum (x) is the next hop on estimated least-cost path to destination (z).

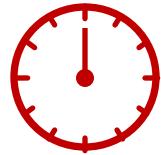
Distance vector algorithm

key idea:

- from time-to-time, each node sends its own distance vector estimate to neighbors
- when x receives new DV estimate from any neighbor, it updates its own DV using B-F equation:

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

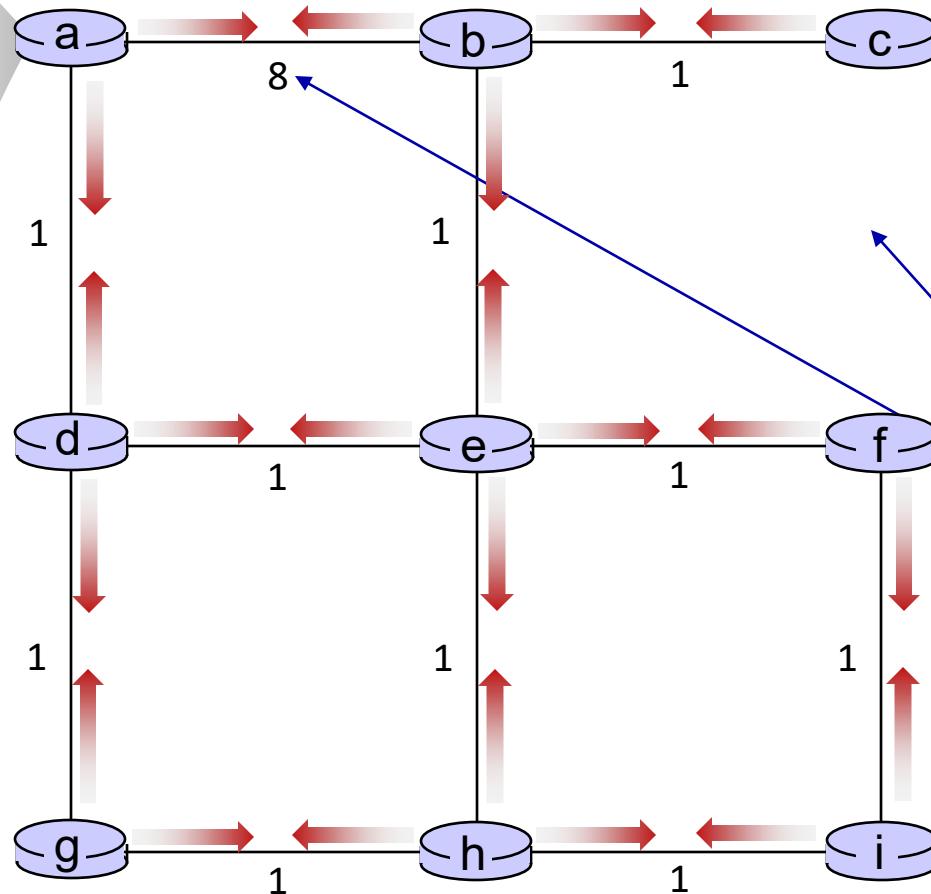
Distance vector: example



$t=0$

- All nodes have distance estimates to nearest neighbors (only)
- All nodes send their local distance vector to their neighbors

DV in a:
$D_a(a) = 0$
$D_a(b) = 8$
$D_a(c) = \infty$
$D_a(d) = 1$
$D_a(e) = \infty$
$D_a(f) = \infty$
$D_a(g) = \infty$
$D_a(h) = \infty$
$D_a(i) = \infty$



- A few asymmetries:
- missing link
 - larger cost

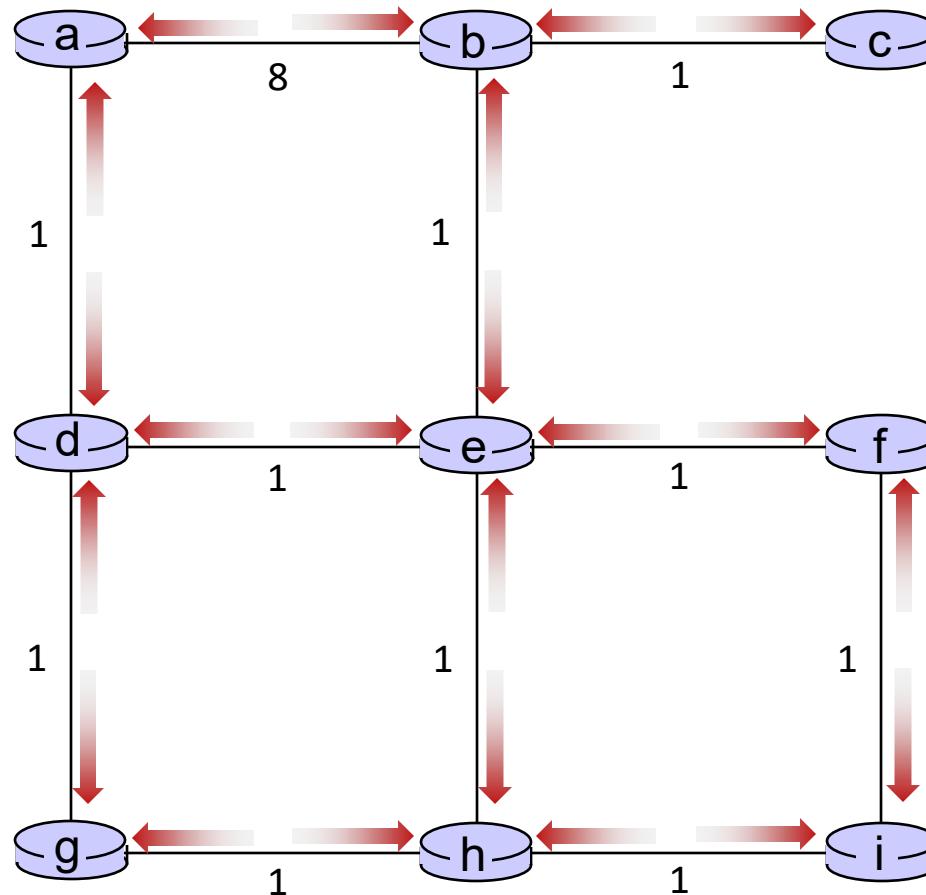
Distance vector example: iteration



$t=1$

All nodes:

- receive distance vectors from neighbors
- compute their new local distance vector
- send their new local distance vector to neighbors



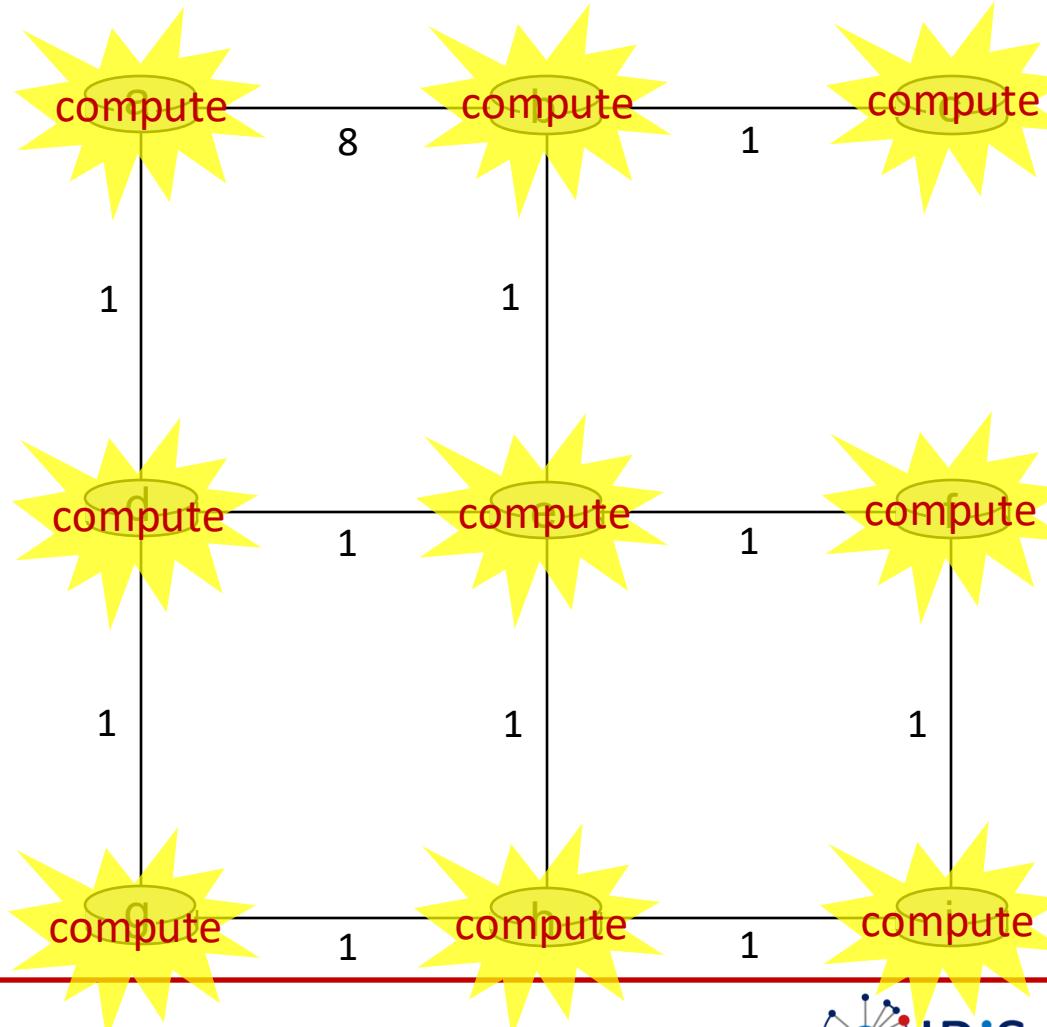
Distance vector example: iteration



$t=1$

All nodes:

- receive distance vectors from neighbors
- compute their new local distance vector
- send their new local distance vector to neighbors



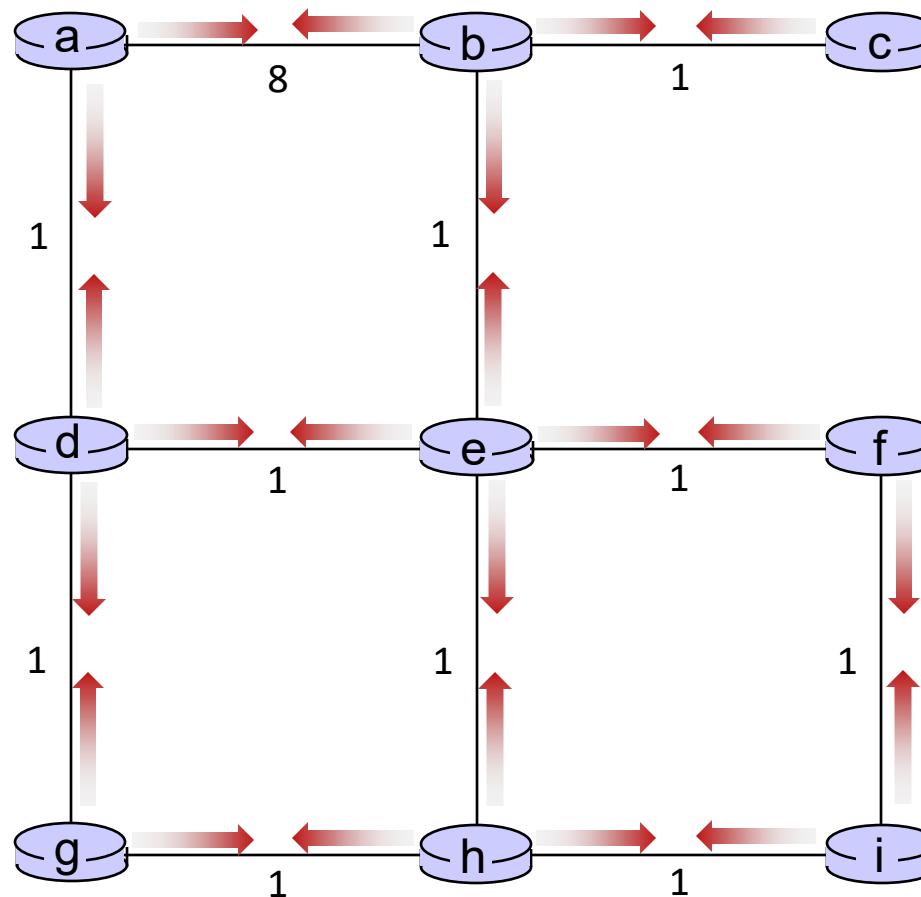
Distance vector example: iteration



$t=1$

All nodes:

- receive distance vectors from neighbors
- compute their new local distance vector
- send their new local distance vector to neighbors



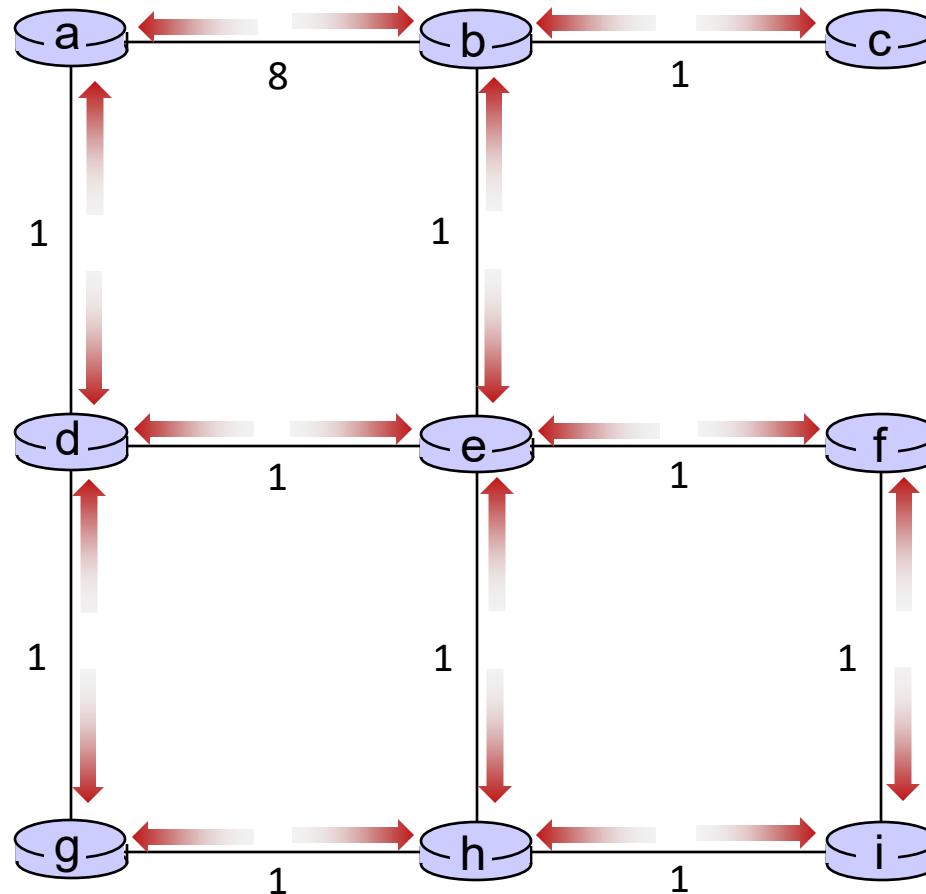
Distance vector example: iteration



$t=2$

All nodes:

- receive distance vectors from neighbors
- compute their new local distance vector
- send their new local distance vector to neighbors



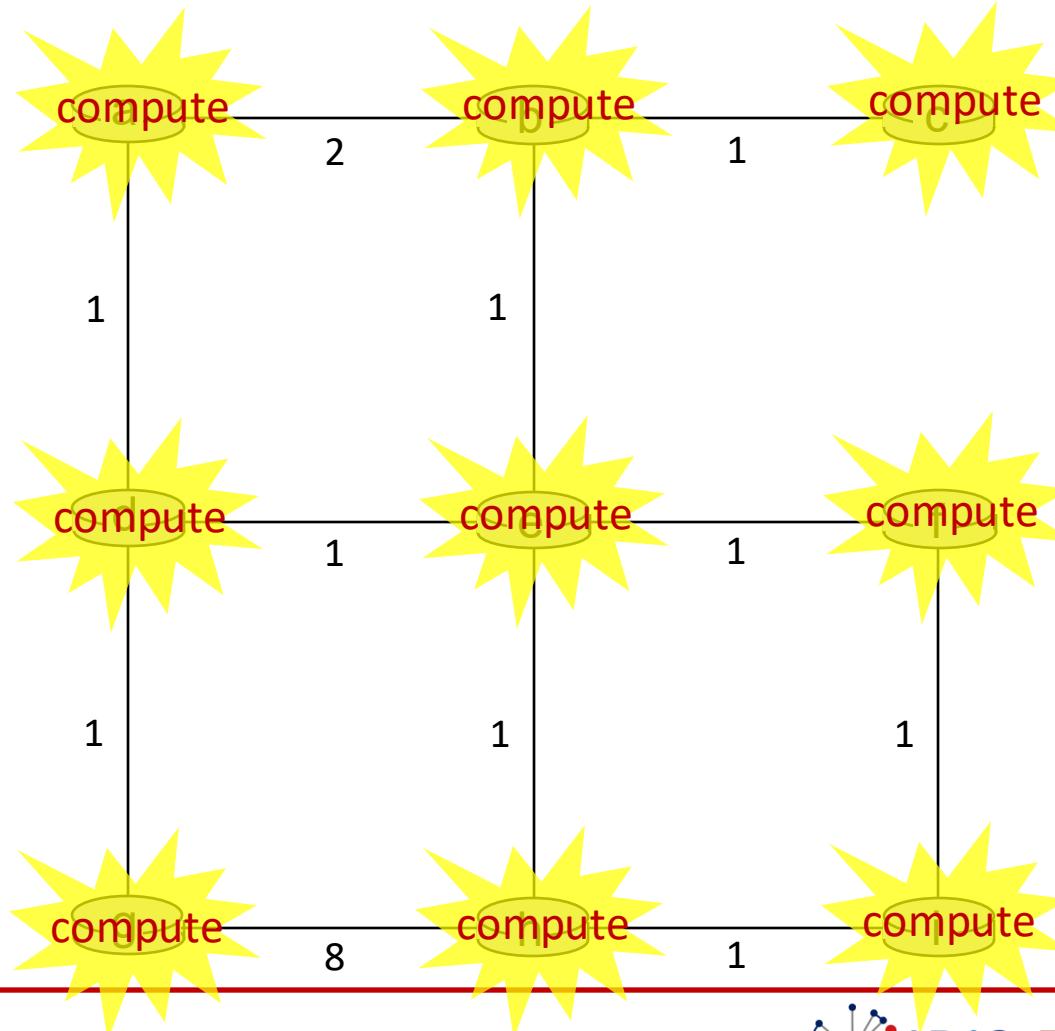
Distance vector example: iteration



$t=2$

All nodes:

- receive distance vectors from neighbors
- compute their new local distance vector
- send their new local distance vector to neighbors



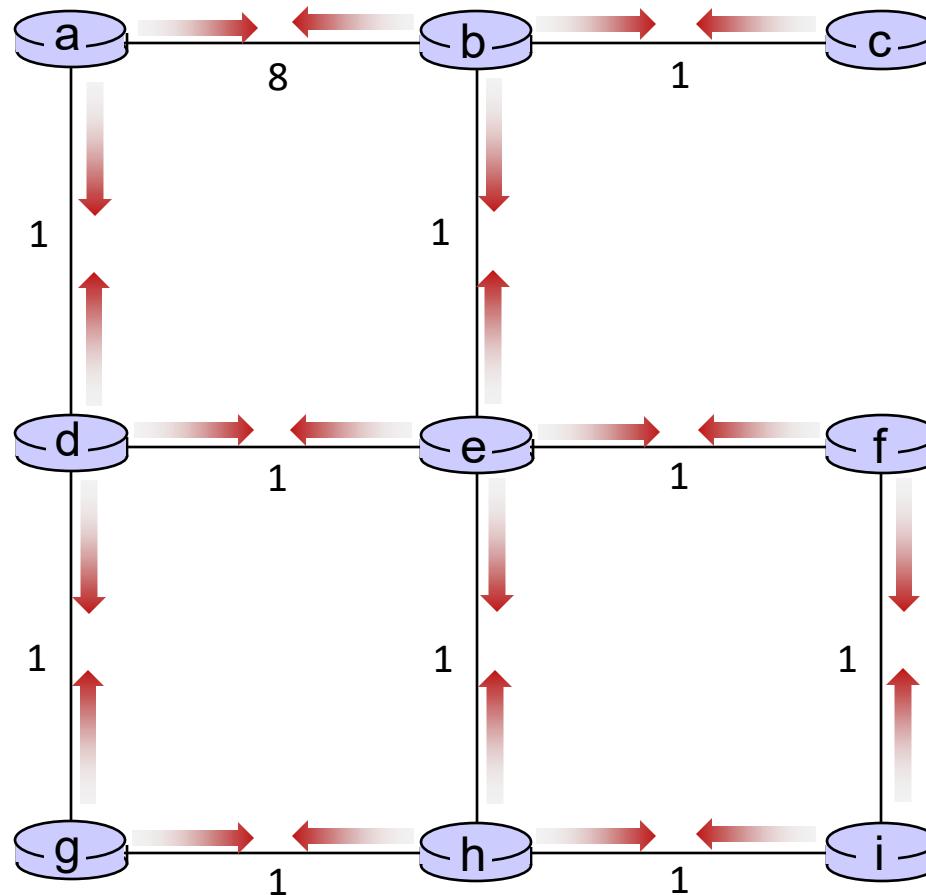
Distance vector example: iteration



$t=2$

All nodes:

- receive distance vectors from neighbors
- compute their new local distance vector
- send their new local distance vector to neighbors



Distance vector example: iteration

.... and so on

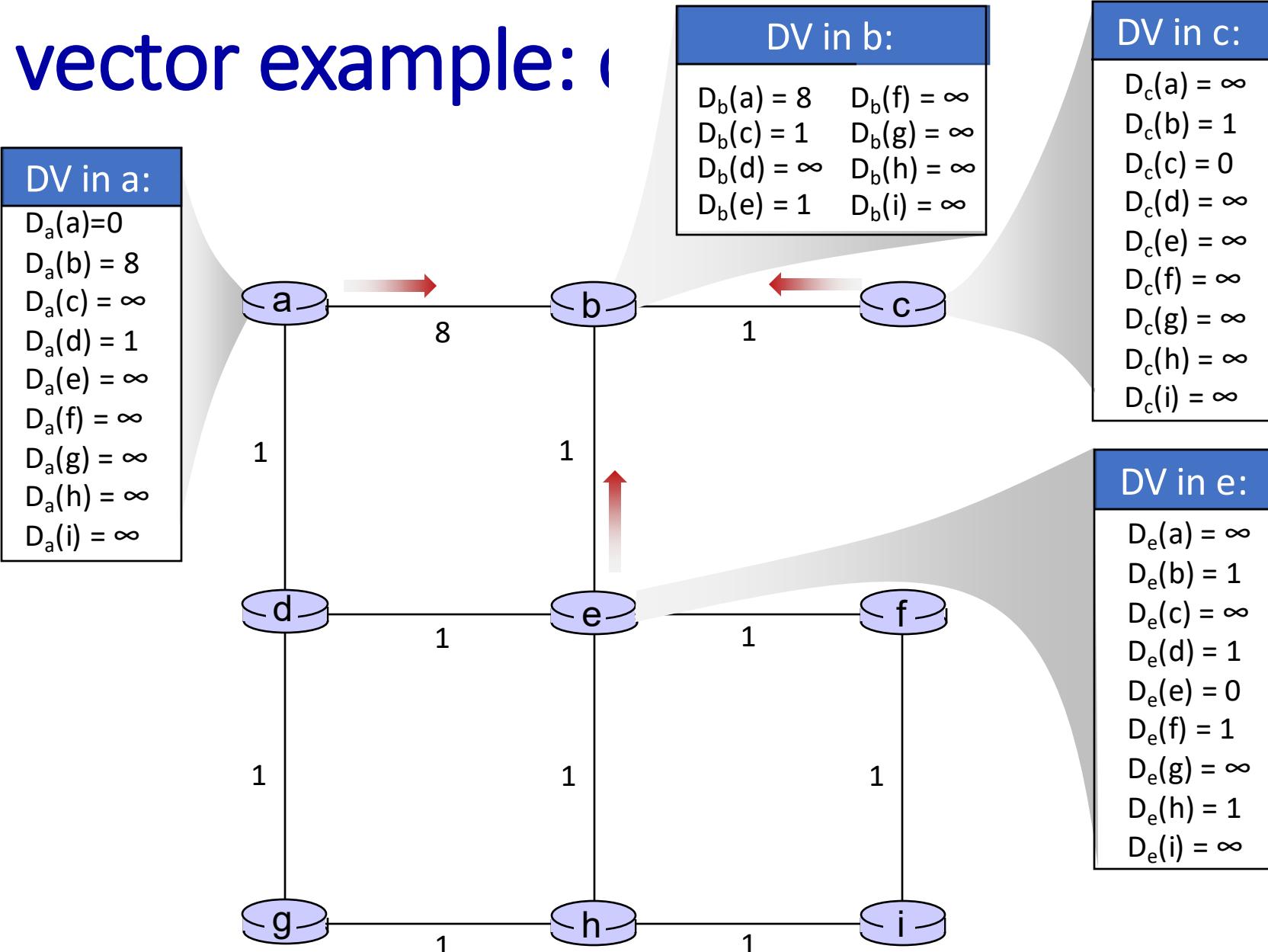
Let's next take a look at the iterative *computations* at nodes

Distance vector example: t=1



$t=1$

- b receives DVs from a, c, e



Distance vector example: t=1

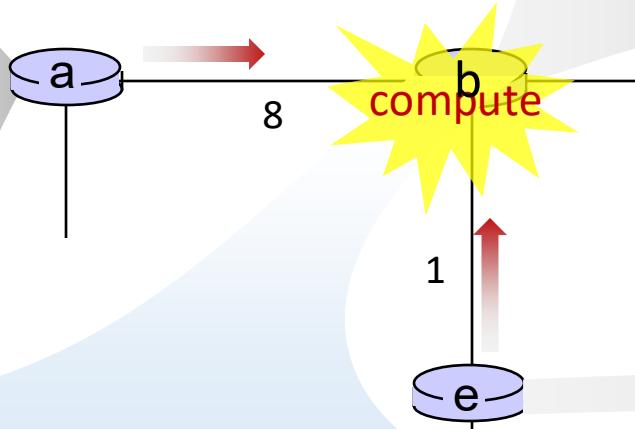


t=1

- b receives DVs from a, c, e, computes:

$$\begin{aligned}
 D_b(a) &= \min\{c_{b,a}+D_a(a), c_{b,c}+D_c(a), c_{b,e}+D_e(a)\} = \min\{8, \infty, \infty\} = 8 \\
 D_b(c) &= \min\{c_{b,a}+D_a(c), c_{b,c}+D_c(c), c_{b,e}+D_e(c)\} = \min\{\infty, 1, \infty\} = 1 \\
 D_b(d) &= \min\{c_{b,a}+D_a(d), c_{b,c}+D_c(d), c_{b,e}+D_e(d)\} = \min\{9, \infty, 2\} = 2 \\
 D_b(e) &= \min\{c_{b,a}+D_a(e), c_{b,c}+D_c(e), c_{b,e}+D_e(e)\} = \min\{\infty, \infty, 1\} = 1 \\
 D_b(f) &= \min\{c_{b,a}+D_a(f), c_{b,c}+D_c(f), c_{b,e}+D_e(f)\} = \min\{\infty, \infty, 2\} = 2 \\
 D_b(g) &= \min\{c_{b,a}+D_a(g), c_{b,c}+D_c(g), c_{b,e}+D_e(g)\} = \min\{\infty, \infty, \infty\} = \infty \\
 D_b(h) &= \min\{c_{b,a}+D_a(h), c_{b,c}+D_c(h), c_{b,e}+D_e(h)\} = \min\{\infty, \infty, 2\} = 2 \\
 D_b(i) &= \min\{c_{b,a}+D_a(i), c_{b,c}+D_c(i), c_{b,e}+D_e(i)\} = \min\{\infty, \infty, \infty\} = \infty
 \end{aligned}$$

DV in a:
$D_a(a)=0$
$D_a(b)=8$
$D_a(c)=\infty$
$D_a(d)=1$
$D_a(e)=\infty$
$D_a(f)=\infty$
$D_a(g)=\infty$
$D_a(h)=\infty$
$D_a(i)=\infty$



DV in b:	
$D_b(a)=8$	$D_b(f)=\infty$
$D_b(c)=1$	$D_b(g)=\infty$
$D_b(d)=\infty$	$D_b(h)=\infty$
$D_b(e)=1$	$D_b(i)=\infty$

DV in c:
$D_c(a)=\infty$
$D_c(b)=1$
$D_c(c)=0$
$D_c(d)=\infty$
$D_c(e)=\infty$
$D_c(f)=\infty$
$D_c(g)=\infty$
$D_c(h)=\infty$
$D_c(i)=\infty$

DV in e:
$D_e(a)=\infty$
$D_e(b)=1$
$D_e(c)=\infty$
$D_e(d)=1$
$D_e(e)=0$
$D_e(f)=1$
$D_e(g)=\infty$
$D_e(h)=1$
$D_e(i)=\infty$

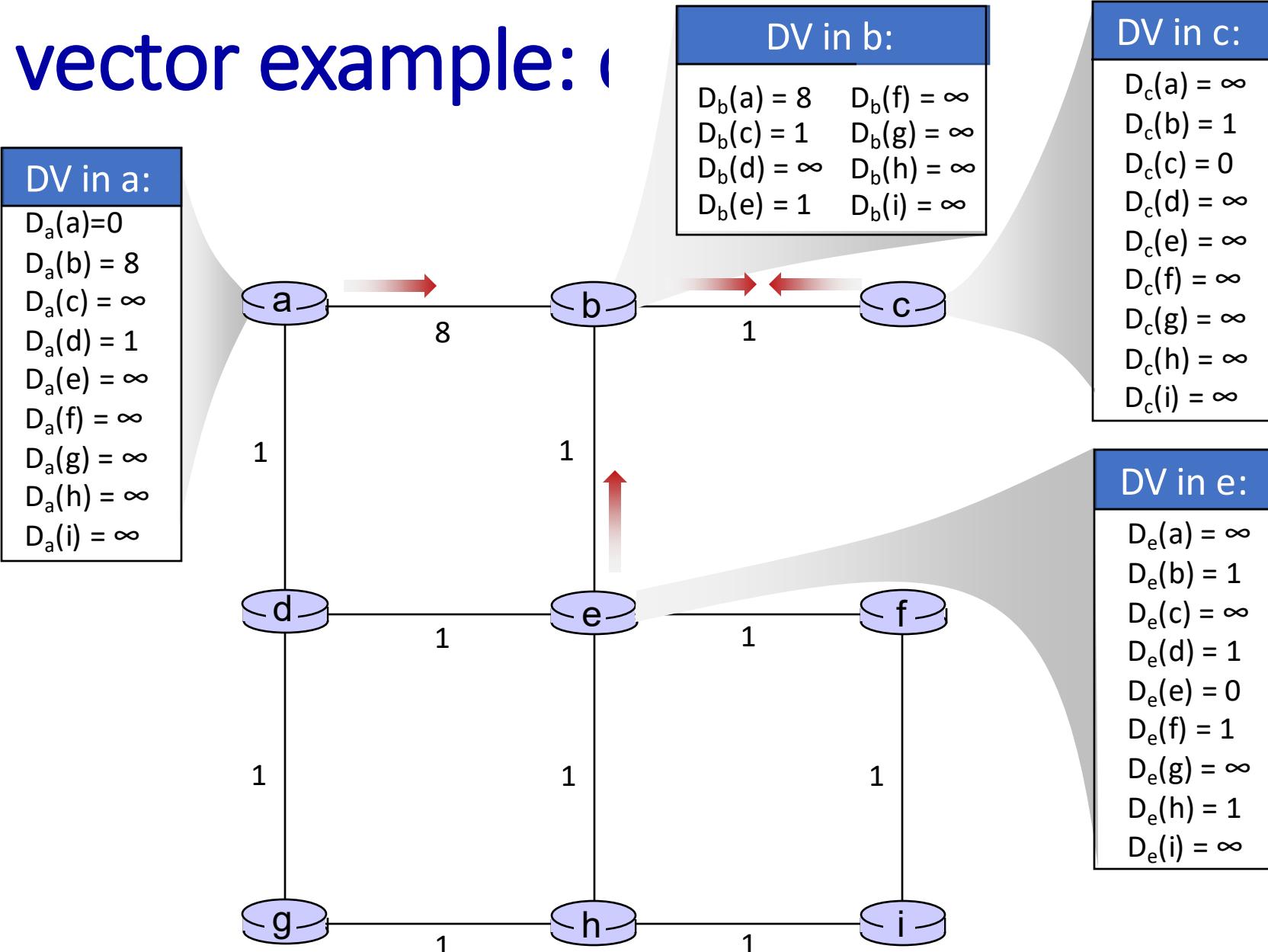
New DV in b:	
$D_b(a)=8$	$D_b(f)=2$
$D_b(c)=1$	$D_b(g)=\infty$
$D_b(d)=2$	$D_b(h)=2$
$D_b(e)=1$	$D_b(i)=\infty$

Distance vector example: t=1



$t=1$

- c receives DVs from b



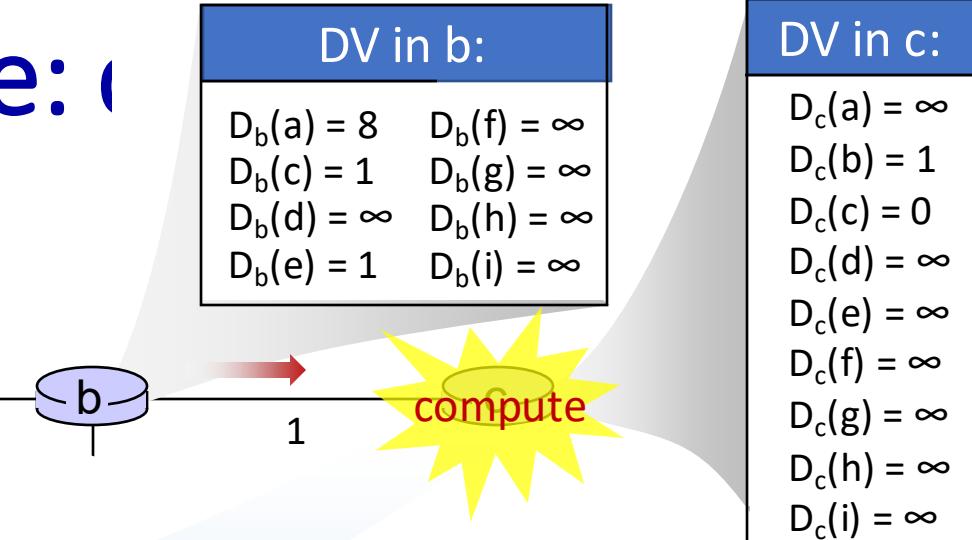
Distance vector example: (t=1)



t=1

- c receives DVs from b computes:

$$\begin{aligned}D_c(a) &= \min\{c_{c,b} + D_b(a)\} = 1 + 8 = 9 \\D_c(b) &= \min\{c_{c,b} + D_b(b)\} = 1 + 0 = 1 \\D_c(d) &= \min\{c_{c,b} + D_b(d)\} = 1 + \infty = \infty \\D_c(e) &= \min\{c_{c,b} + D_b(e)\} = 1 + 1 = 2 \\D_c(f) &= \min\{c_{c,b} + D_b(f)\} = 1 + \infty = \infty \\D_c(g) &= \min\{c_{c,b} + D_b(g)\} = 1 + \infty = \infty \\D_c(h) &= \min\{c_{c,b} + D_b(h)\} = 1 + \infty = \infty \\D_c(i) &= \min\{c_{c,b} + D_b(i)\} = 1 + \infty = \infty\end{aligned}$$



DV in c:

$D_c(a) = 9$
$D_c(b) = 1$
$D_c(c) = 0$
$D_c(d) = 2$
$D_c(e) = \infty$
$D_c(f) = \infty$
$D_c(g) = \infty$
$D_c(h) = \infty$
$D_c(i) = \infty$

* Check out the online interactive exercises for more examples:
http://gaia.cs.umass.edu/kurose_ross/interactive/

Distance vector example: t=1

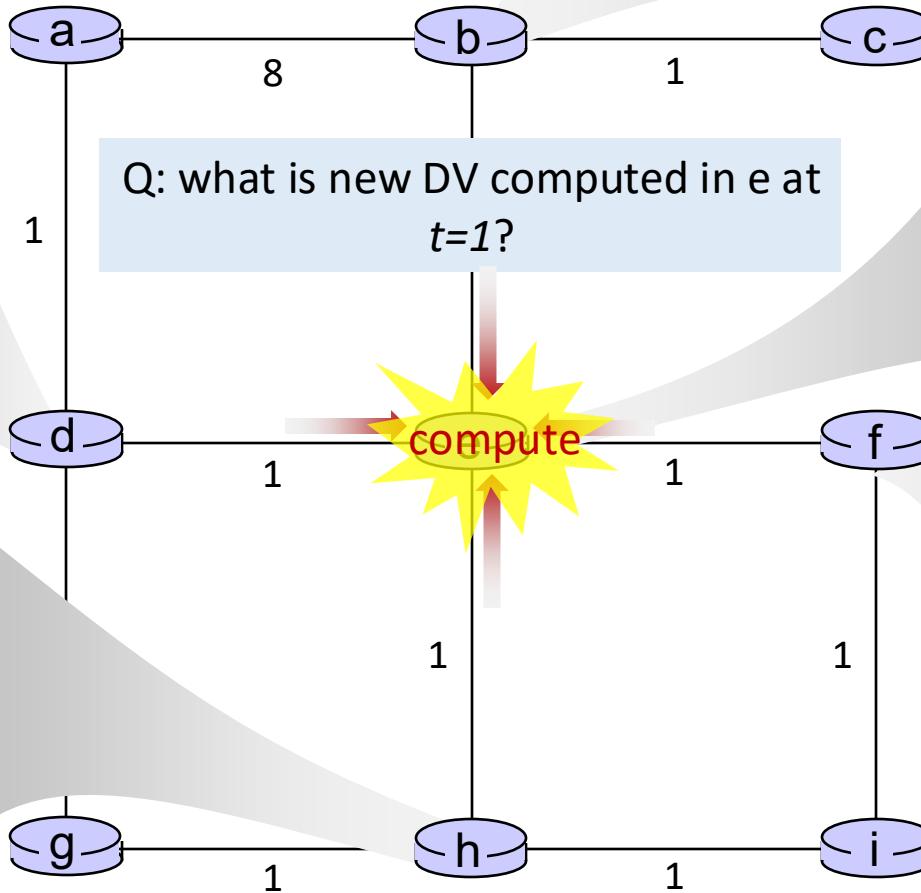


t=1

- e receives DVs from b, d, f, h

DV in d:
$D_c(a) = 1$
$D_c(b) = \infty$
$D_c(c) = \infty$
$D_c(d) = 0$
$D_c(e) = 1$
$D_c(f) = \infty$
$D_c(g) = 1$
$D_c(h) = \infty$
$D_c(i) = \infty$

DV in h:
$D_c(a) = \infty$
$D_c(b) = \infty$
$D_c(c) = \infty$
$D_c(d) = \infty$
$D_c(e) = 1$
$D_c(f) = \infty$
$D_c(g) = 1$
$D_c(h) = 0$
$D_c(i) = 1$



DV in b:
$D_b(a) = 8$
$D_b(f) = \infty$
$D_b(c) = 1$
$D_b(g) = \infty$
$D_b(d) = \infty$
$D_b(h) = \infty$
$D_b(e) = 1$
$D_b(i) = \infty$

DV in e:
$D_e(a) = \infty$
$D_e(b) = 1$
$D_e(c) = \infty$
$D_e(d) = 1$
$D_e(e) = 0$
$D_e(f) = 1$
$D_e(g) = \infty$
$D_e(h) = 1$
$D_e(i) = \infty$

DV in f:
$D_c(a) = \infty$
$D_c(b) = \infty$
$D_c(c) = \infty$
$D_c(d) = \infty$
$D_c(e) = 1$
$D_c(f) = 0$
$D_c(g) = \infty$
$D_c(h) = \infty$
$D_c(i) = 1$

Making routing scalable

our routing study thus far - idealized

- all routers identical
- network “flat”

... not true in practice

scale: billions of destinations:

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

administrative autonomy:

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

Internet approach to scalable routing

aggregate routers into regions known as “autonomous systems” (AS)

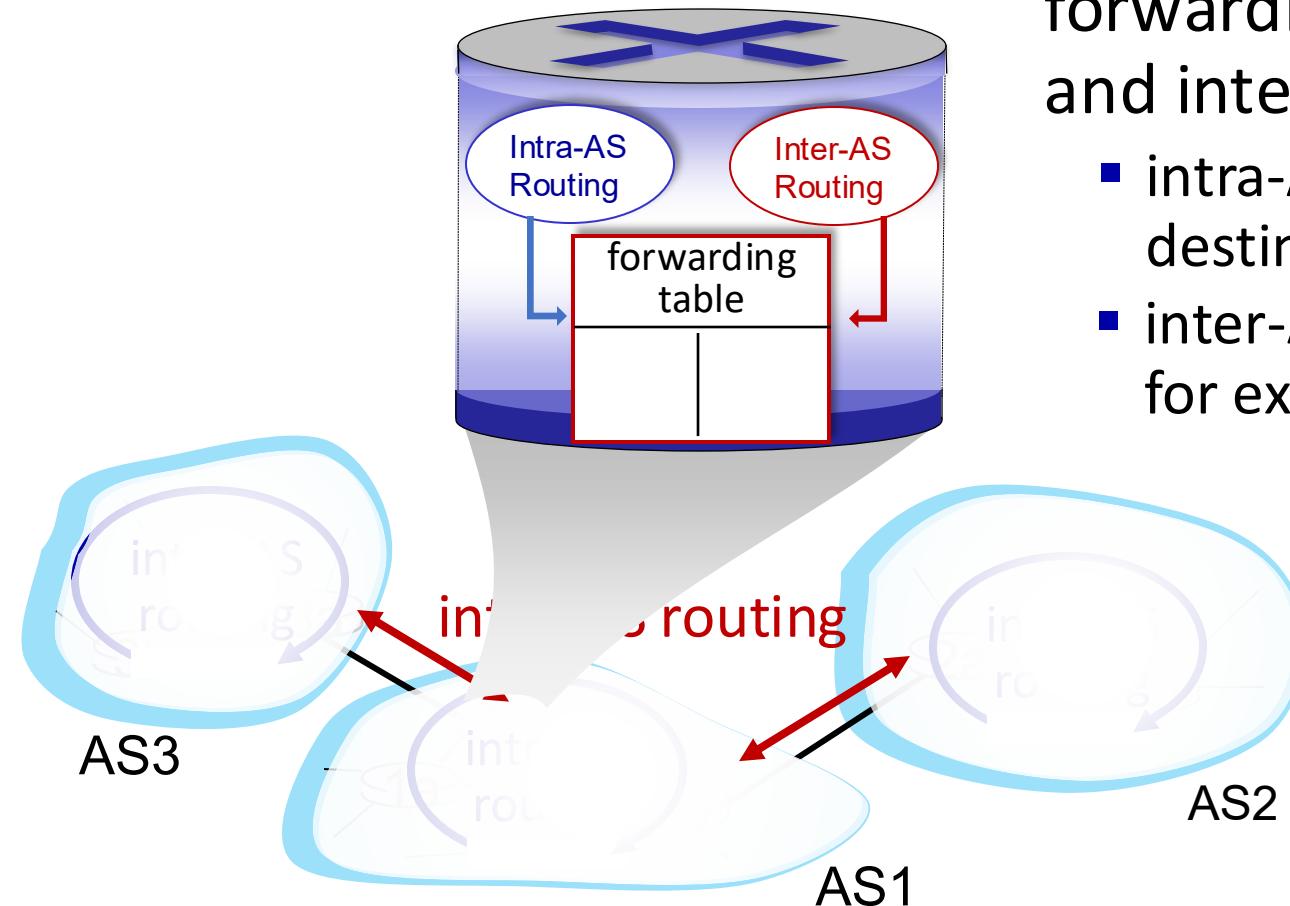
intra-AS: routing *within same AS (“network”)*

- all routers in AS must run same intra-domain protocol
- routers in different AS can run different intra-domain routing protocols

inter-AS: routing to *outside of AS*

- **gateway router:** at “edge” of its own AS, has link(s) to router(s) in other AS
- gateways perform inter-AS routing (as well as intra-AS routing)

Interconnected autonomous systems



forwarding table configured by intra- and inter-AS routing algorithms

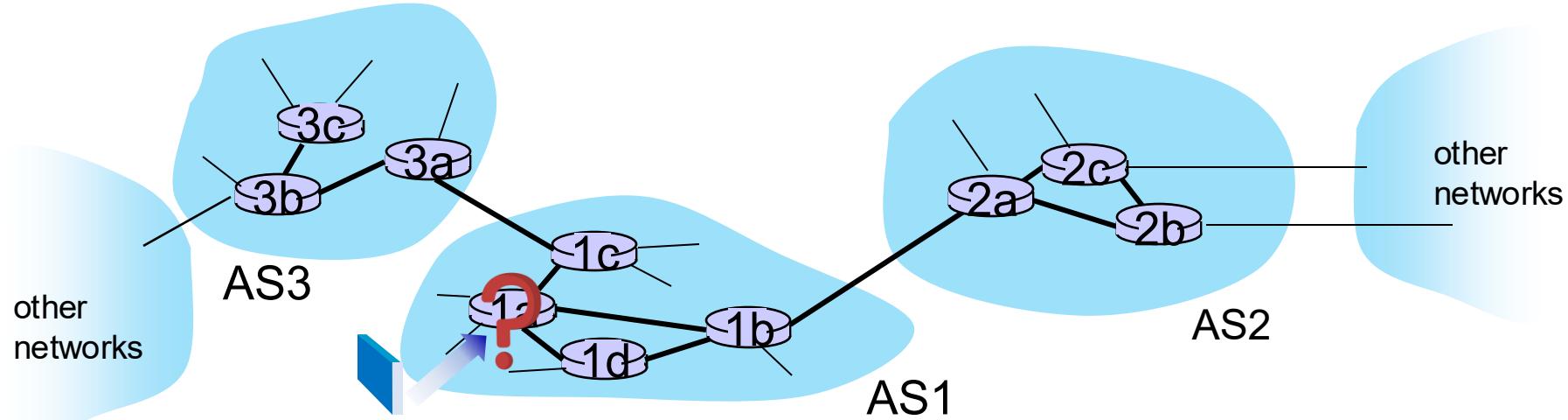
- intra-AS routing determine entries for destinations within AS
- inter-AS & intra-AS determine entries for external destinations

Inter-AS routing

- suppose router in AS1 receives datagram destined outside of AS1:
 - router should forward packet to gateway router in AS1, but which one?

AS1 inter-AS routing must:

1. learn which destinations reachable through AS2, which through AS3
2. propagate this reachability info to all routers in AS1



Intra-AS routing: routing within an AS

most common intra-AS routing protocols:

- RIP: Routing Information Protocol [RFC 1723]
 - classic DV routing: DVs exchanged every 30 secs
 - no longer widely used
- EIGRP: Enhanced Interior Gateway Routing Protocol [RFC 7868]
 - DV based
- OSPF: Open Shortest Path First [RFC 2328]
 - classic link-state routing

OSPF (Open Shortest Path First)

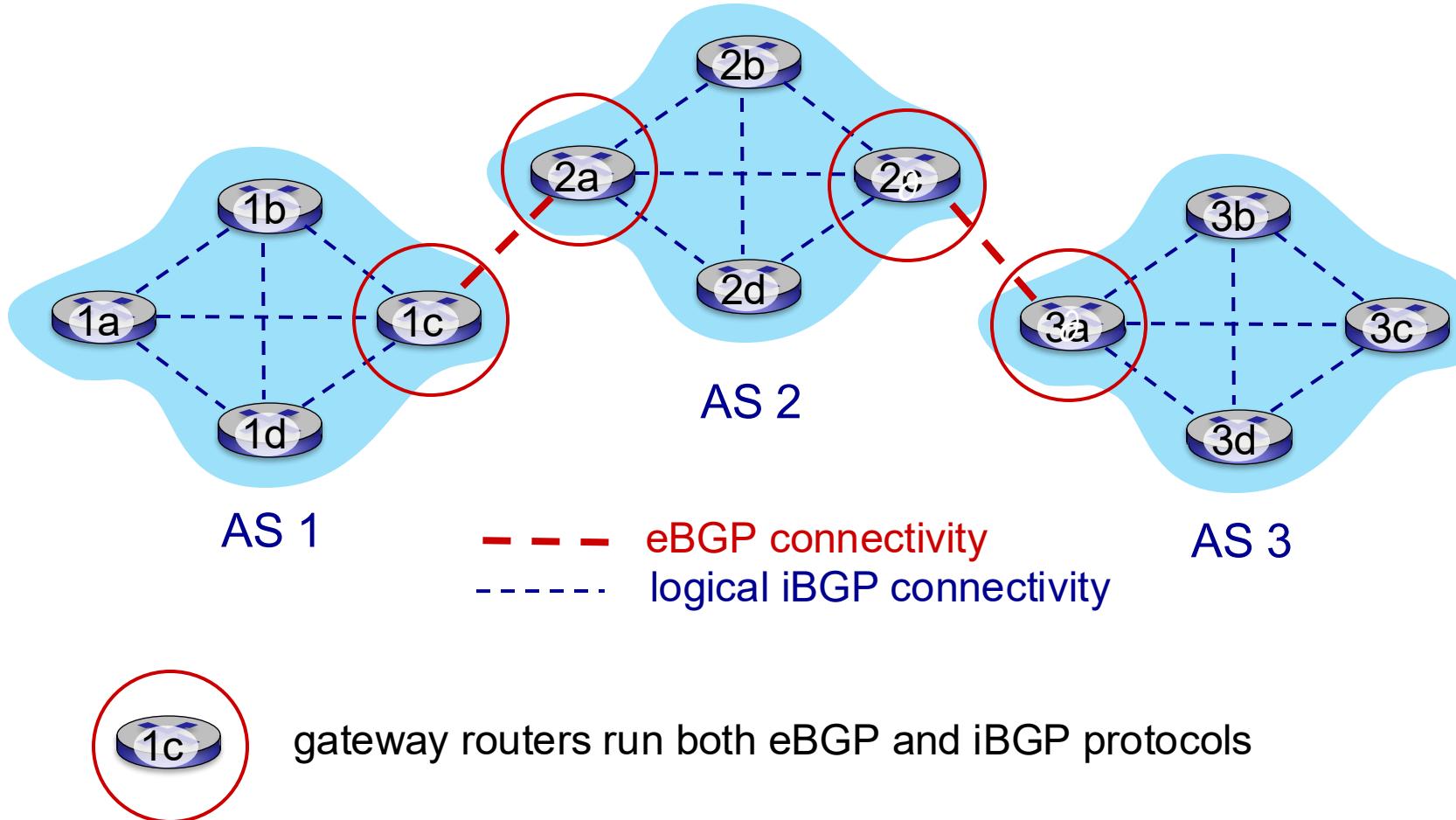
- “open”: publicly available
- classic link-state (Dijkstra’s alg. to compute forwarding table)
 - each router broadcasts OSPF link-state advertisements to entire AS
 - (directly over IP)
 - multiple link costs metrics possible: bandwidth, delay

Internet inter-AS routing: BGP

Internet inter-AS routing: BGP

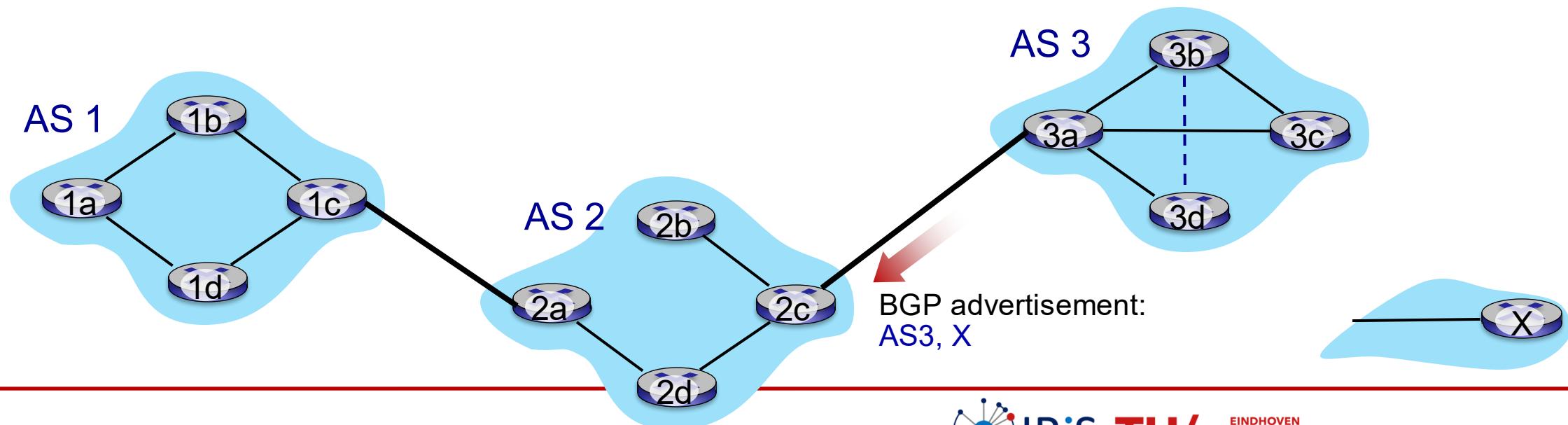
- **BGP (Border Gateway Protocol):** *the de facto inter-domain routing protocol*
 - “glue that holds the Internet together”
- allows subnet to advertise its existence, and the destinations it can reach, to rest of Internet: *“I am here, here is who I can reach, and how”*
- BGP provides each AS a means to:
 - **eBGP:** obtain subnet reachability information from neighboring AS
 - **iBGP:** propagate reachability information to all AS-internal routers.
 - determine “good” routes to other networks based on reachability information and *policy*

eBGP, iBGP connections



BGP basics

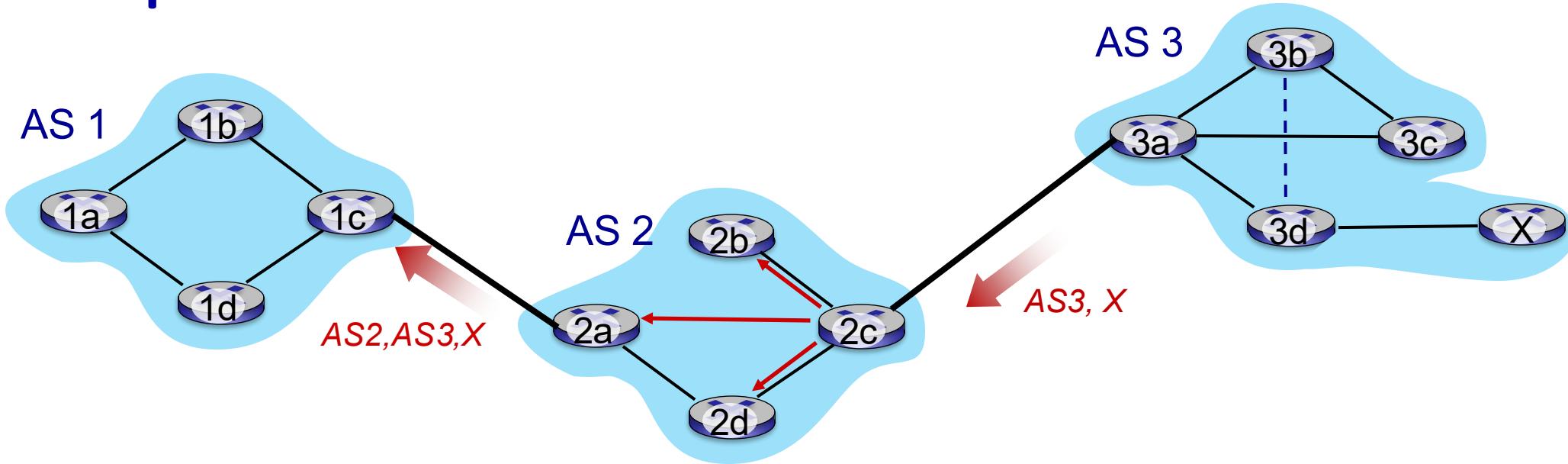
- **BGP session:** two BGP routers (“peers”) exchange BGP messages over a TCP connection:
 - advertising *paths* to different destination network prefixes
- when AS3 gateway 3a advertises **path AS3,X** to AS2 gateway 2c:
 - AS3 *promises* to AS2 it will forward datagrams towards X



Path attributes and BGP routes

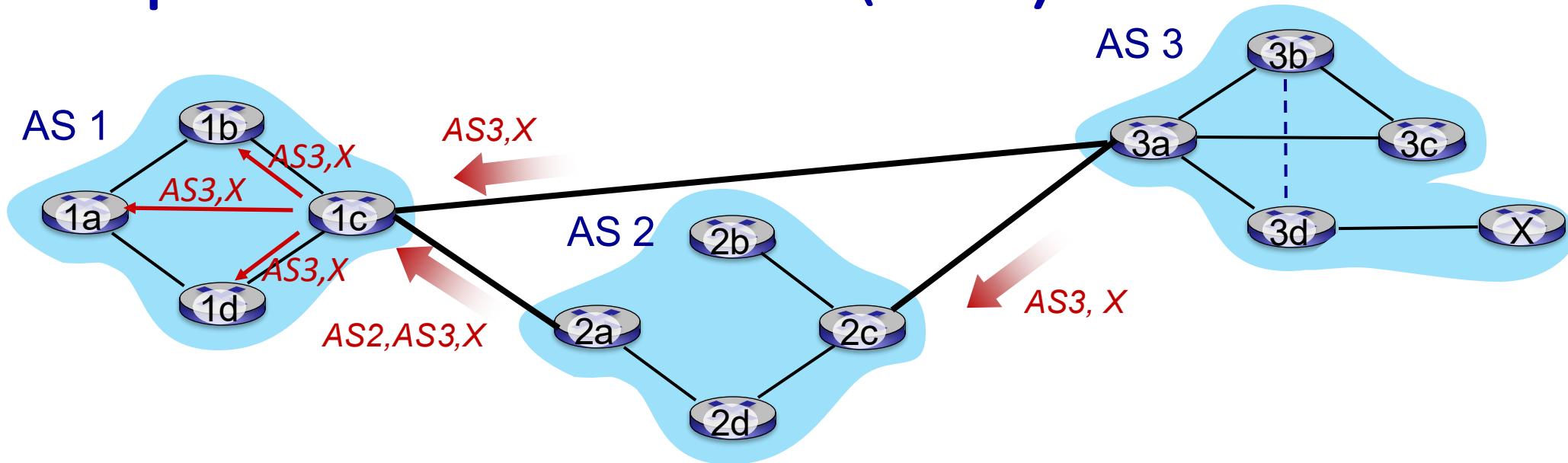
- BGP advertised route: prefix + attributes
 - prefix: destination being advertised
 - two important attributes:
 - AS-PATH: list of AS through which prefix advertisement has passed
 - NEXT-HOP: indicates specific internal-AS router to next-hop AS
- policy-based routing:
 - gateway receiving route advertisement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

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

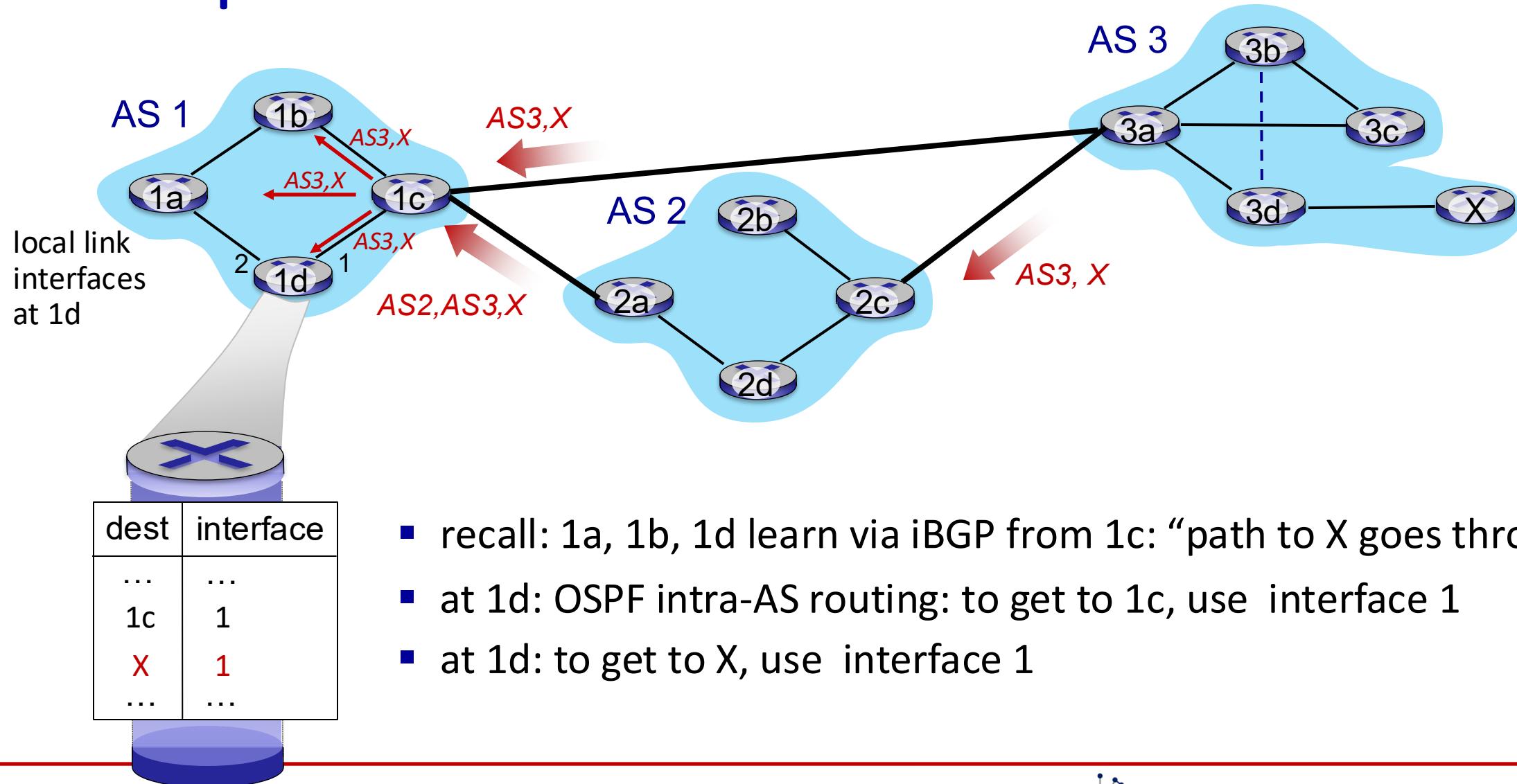
BGP path advertisement (more)



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 path advertisement



Why different Intra-, Inter-AS routing ?

policy:

- inter-AS: admin wants control over how its traffic routed, who routes through its network
- intra-AS: single admin, so policy less of an issue

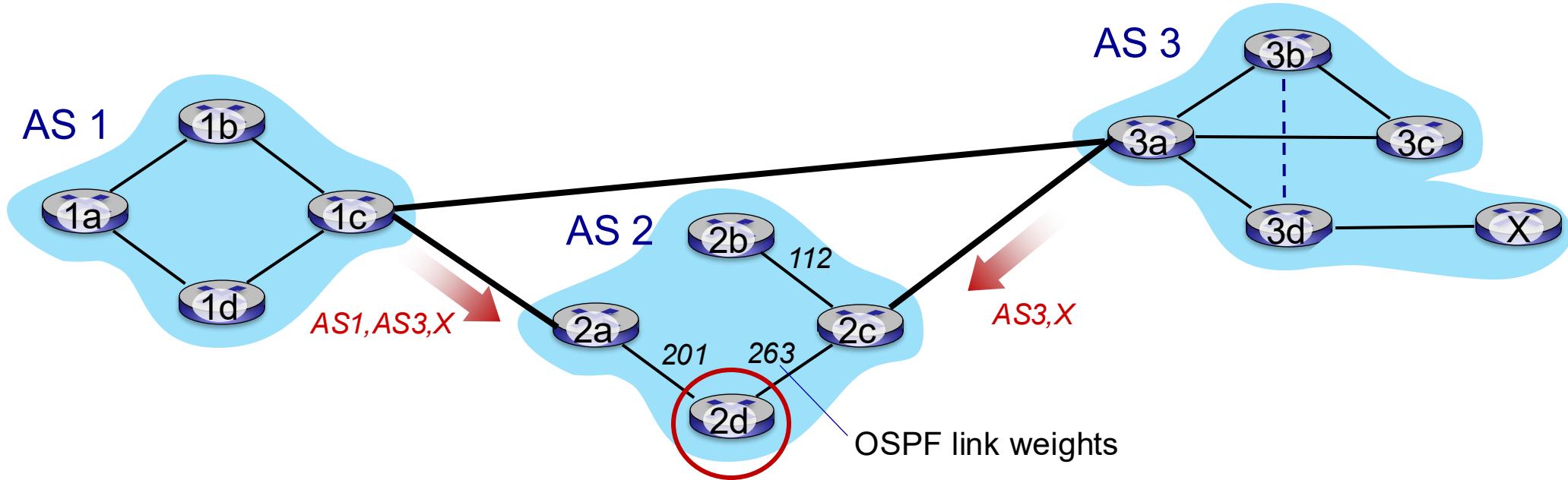
scale:

- hierarchical routing saves table size, reduced update traffic

performance:

- intra-AS: can focus on performance
- inter-AS: policy dominates over performance

Hot potato routing in BGP



- 2d learns (via iBGP) it can route to X via 2a or 2c
- **hot potato routing:** choose local gateway with least *intra-AS* cost (e.g., 2d chooses 2a, even though more AS hops to X): ignoring inter-AS cost!

BGP route selection

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

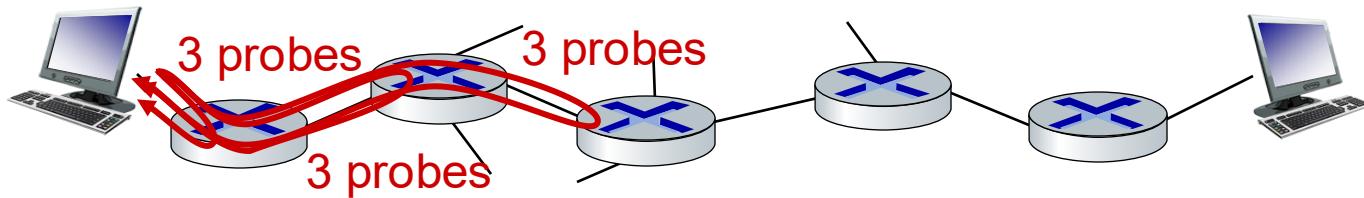
Internet Control Message Protocol

ICMP: internet control message protocol

- used by hosts and routers to communicate network-level information
 - error reporting: unreachable host, network, port, protocol
 - echo request/reply (used by ping)
- network-layer “above” IP:
 - ICMP messages carried in IP datagrams
- *ICMP message*: type, code plus first 8 bytes of IP datagram causing error

Type	Code	description
0	0	echo reply (ping)
3	0	dest. network unreachable
3	1	dest host unreachable
3	2	dest protocol unreachable
3	3	dest port unreachable
3	6	dest network unknown
3	7	dest host unknown
4	0	source quench (congestion control - not used)
8	0	echo request (ping)
9	0	route advertisement
10	0	router discovery
11	0	TTL expired
12	0	bad IP header

Traceroute and ICMP



- source sends sets of UDP segments to destination
 - 1st set has TTL =1, 2nd set has TTL=2, etc.
- datagram in n th set arrives to n th router:
 - router discards datagram and sends source ICMP message (type 11, code 0)
 - ICMP message possibly includes name of router & IP address
- when ICMP message arrives at source: record RTTs

stopping criteria:

- UDP segment eventually arrives at destination host
- destination returns ICMP “port unreachable” message (type 3, code 3)
- source stops