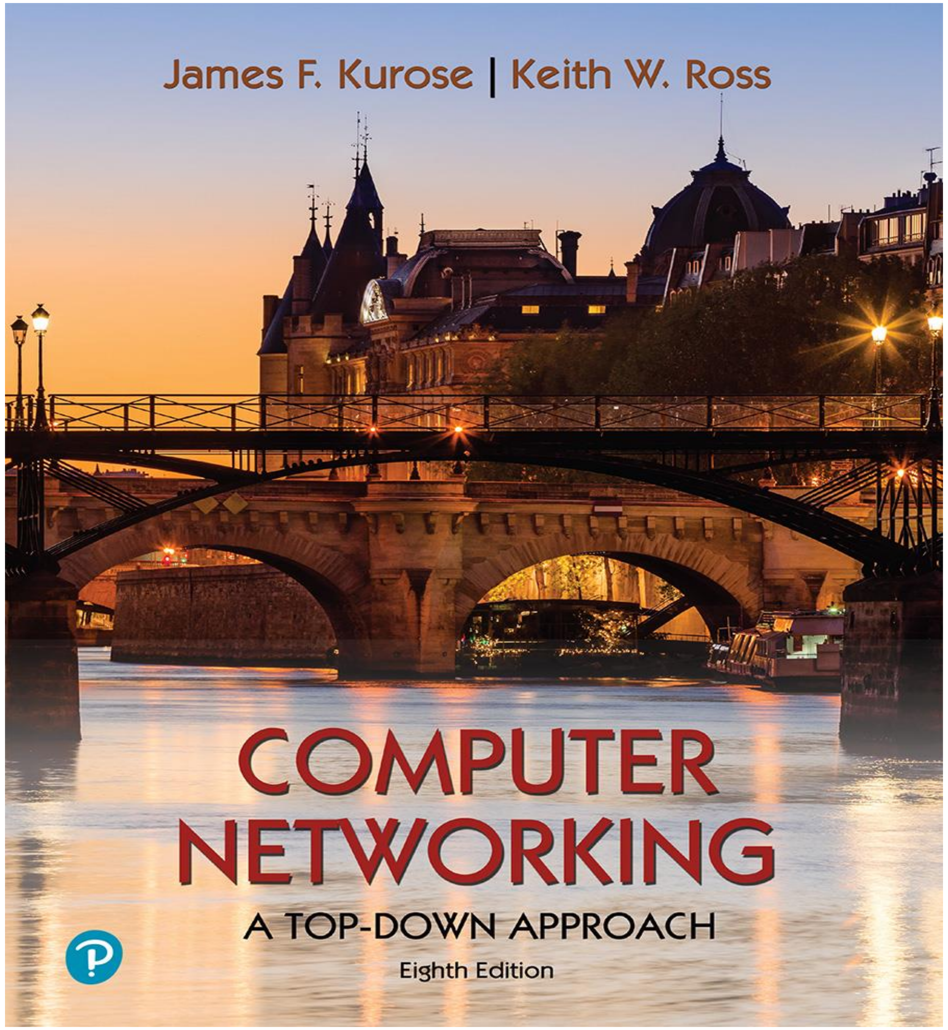
Network Layer – Routing Protocol

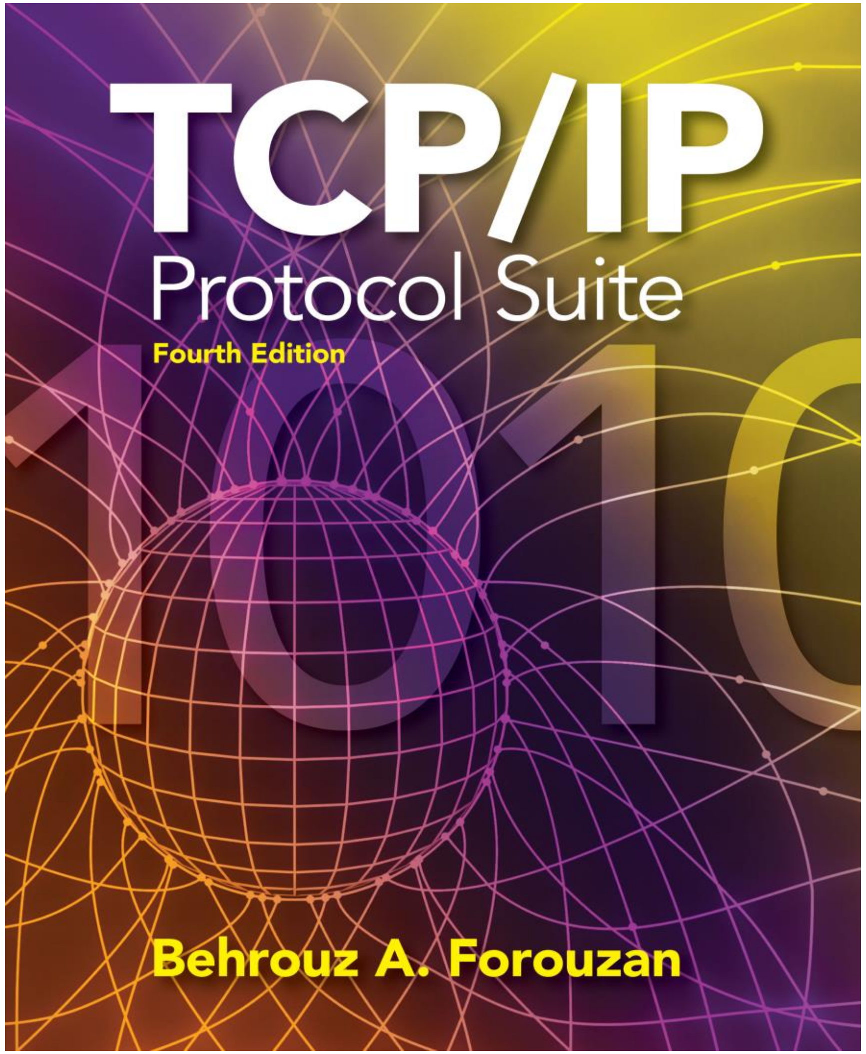
Anand Baswade

anand@iitbhilai.ac.in

Sources



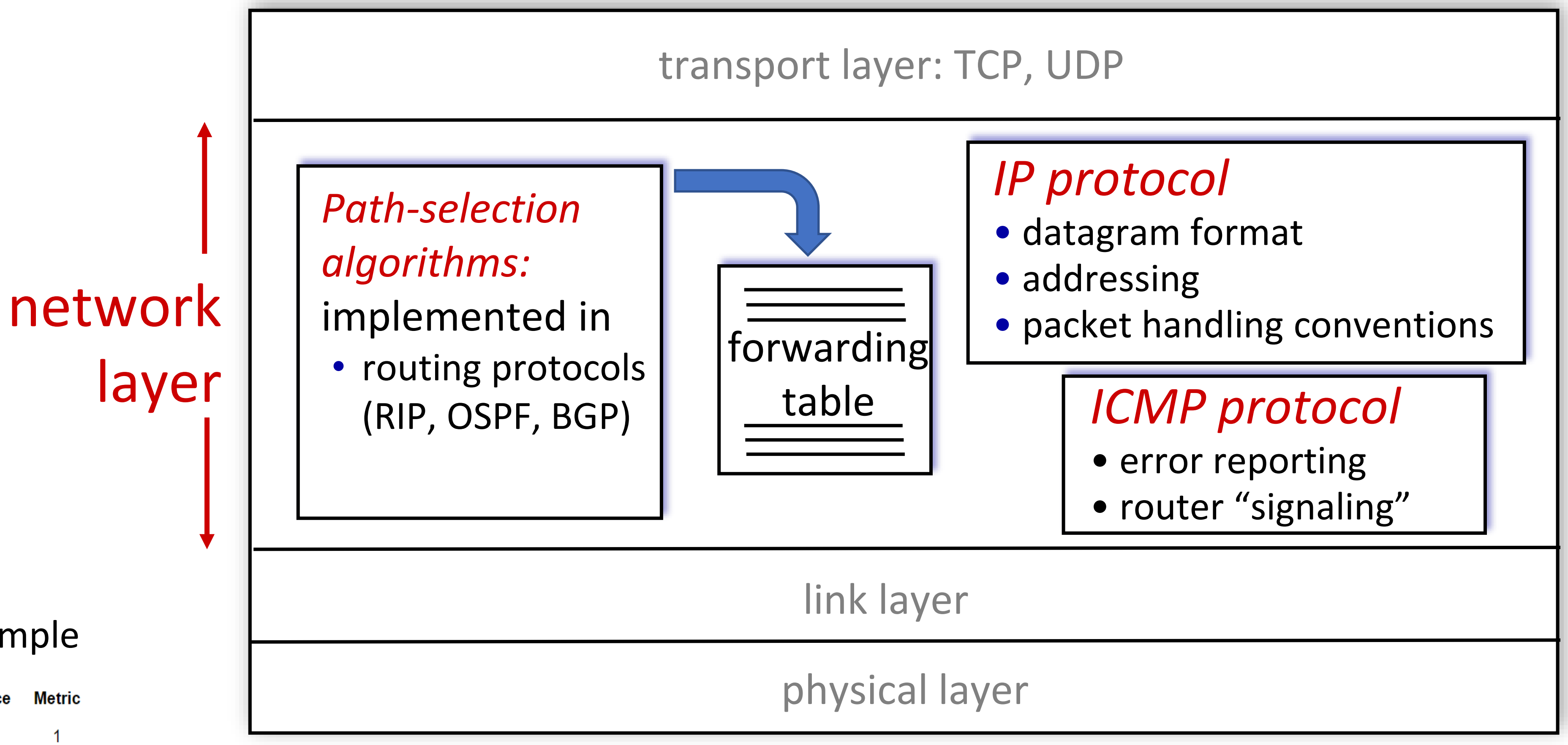
Computer Networking: A Top-Down Approach



TCP/IP Protocol Suite

Network Layer: Internet

host, router network layer functions:



Routing/Forwarding Table Example

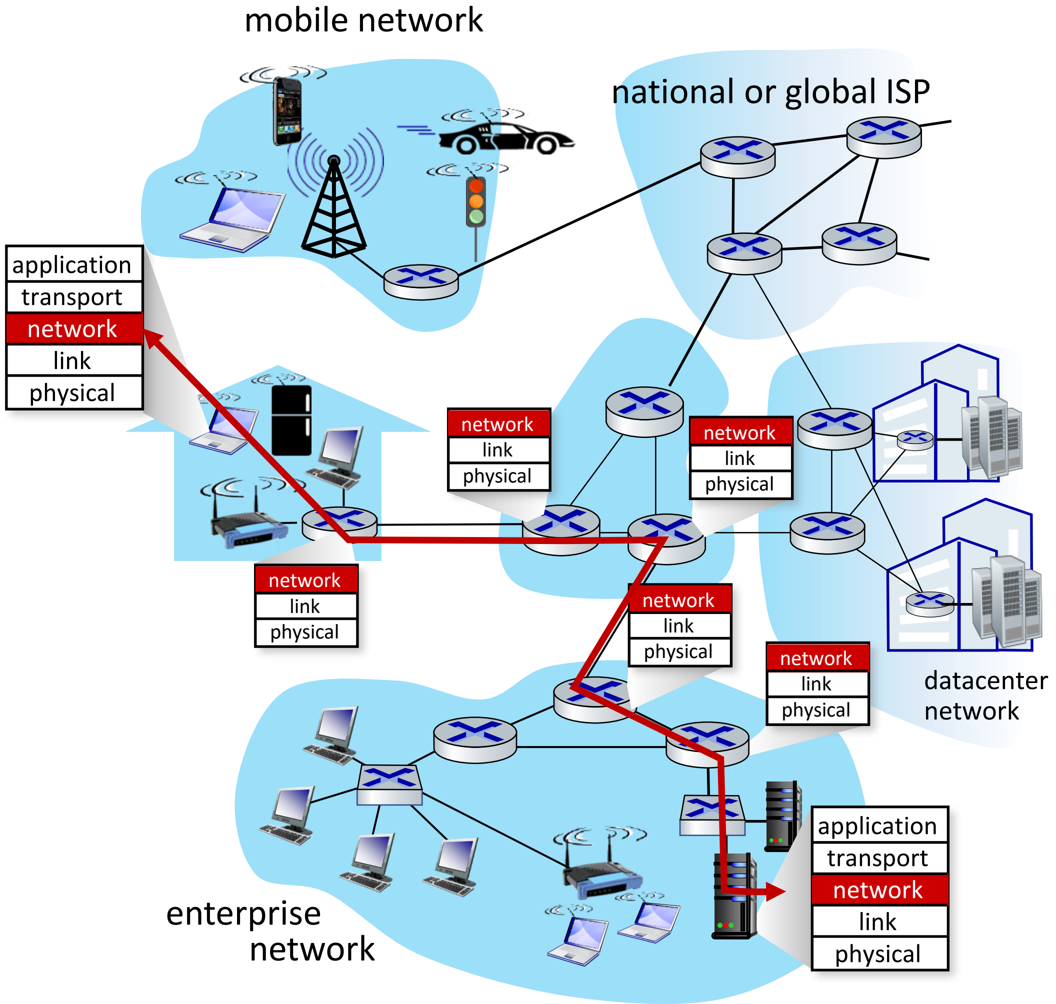
Network Destination	Netmask	Gateway	Interface	Metric
101.25.67.0	255.255.255.0	10.0.0.2	eth3	1
default	0.0.0.0	10.0.0.1	eth0	0
192.25.67.0	255.255.255.0	10.0.0.3	eth5	10

Routing Protocol

Routing protocols

Routing protocol goal: determine "good" paths (equivalently, routes), from sending hosts to receiving host, through network of routers

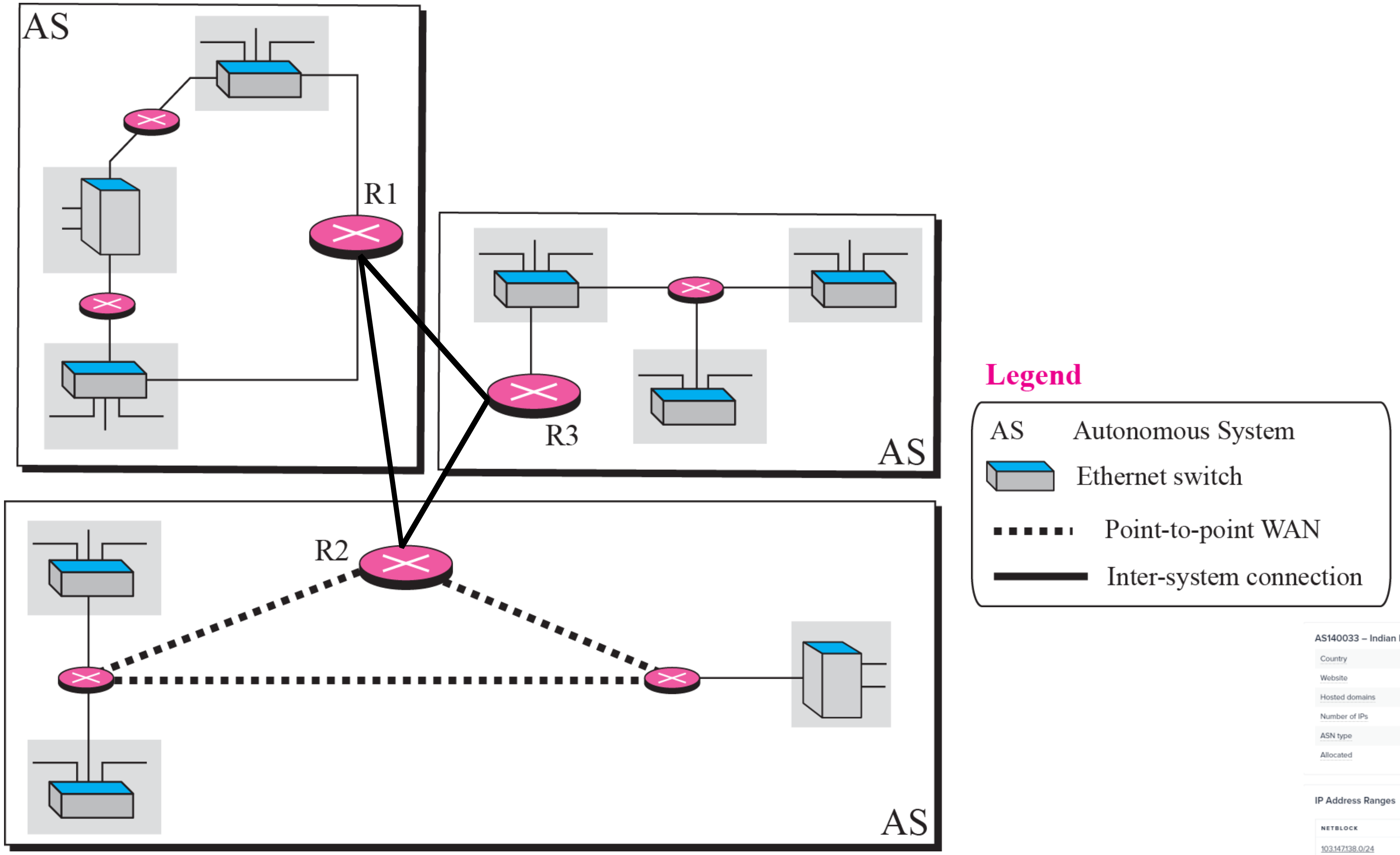
- path: sequence of routers packets traverse from given initial source host to final destination host
- "good": least "cost", "fastest", "least congested"
- routing: a "top-10" networking challenge!

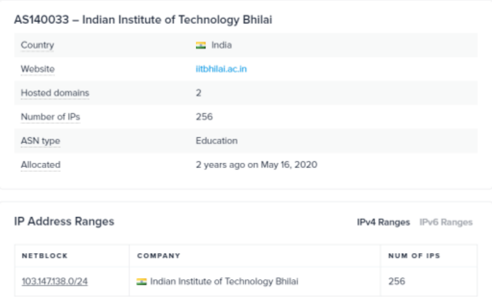


Cont..

- Today, an internet can be so large that one routing protocol cannot handle the task of updating the routing tables of all routers. For this reason, an internet is divided into autonomous systems.
- An autonomous system (AS) is a group of networks and routers under the authority of a single administration. Routing inside an autonomous system is called intra-domain routing. Routing between autonomous systems is called inter-domain routing

Autonomous systems





AS140033 – Indian Institute of Technology Bhilai

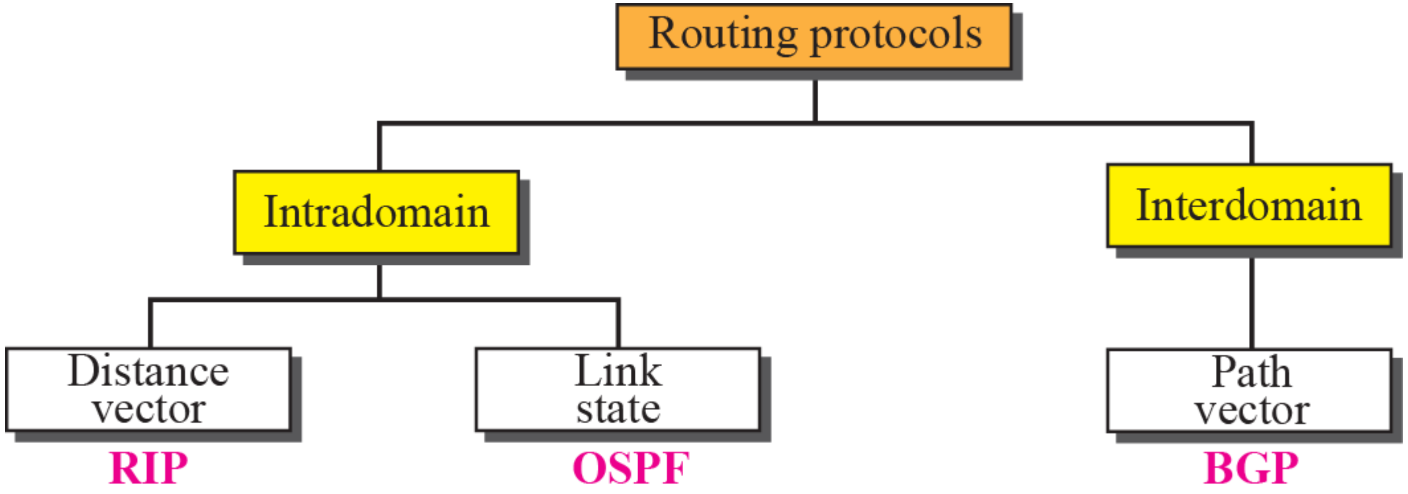
Country	India
Website	iitbhilai.ac.in
Hosted domains	2
Number of IPs	256
ASN type	Education
Allocated	2 years ago on May 16, 2020

IP Address Ranges — IPv4 Ranges IPv6 Ranges

NETBLOCK	COMPANY	NUM OF IPS
103.147.138.0/24	Indian Institute of Technology Bhilai	256

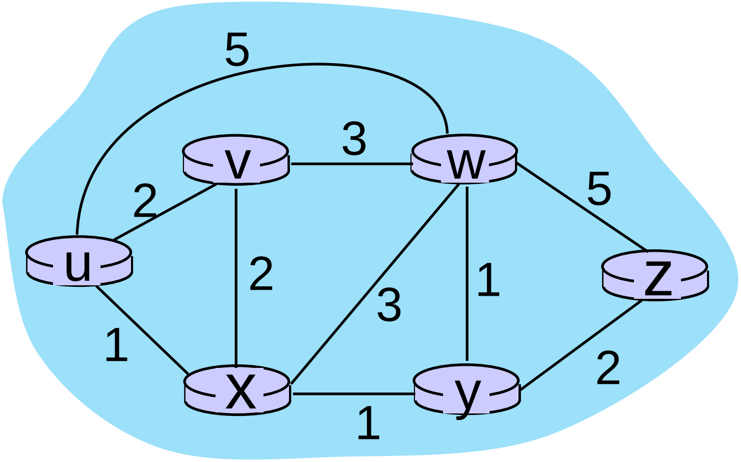
https://ipinfo.io/AS140033

Popular routing protocols



RIP: Routing Information Protocol
OSPF: Open Shortest Path First
BGP: Border Gateway Protocol

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 inversely 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

global: all routers have *complete* topology, link cost info
- "link state" algorithms

How fast do routes change?

static: routes change slowly over time

dynamic: routes change more quickly
- periodic updates or in response to link cost changes

decentralized: iterative process of computation, exchange of info with neighbors
- routers initially only know link costs to attached neighbors
- "distance vector" algorithms

global or decentralized information?

Network layer: "control plane" roadmap

- introduction
- routing protocols
  - distance vector
  - link state
- intra-ISP routing: OSPF
- routing among ISPs: BGP
- SDN control plane
- Internet Control Message Protocol

Distance vector algorithm

- Based on *Bellman-Ford* (BF) equation (dynamic programming): B-F enables us to build a new least cost path from previously established least cost paths.

Bellman-Ford equation

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

Then:

$$\underbrace{\min}_{\text{min taken over all neighbors } v \text{ of } x}\{ \underbrace{c_{x,v}}_{\text{direct cost of link from } x \text{ to } v} + \underbrace{D_v(y)}_{v\text{'s estimated least-cost-path cost to } y} \}$$

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

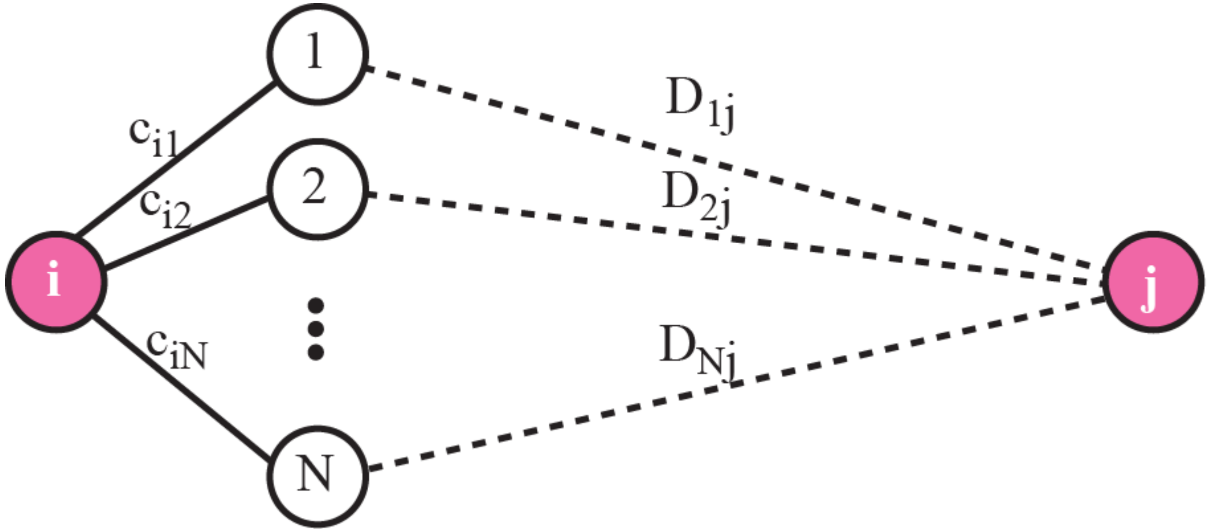
v's estimated least-cost-path cost to *y*

min taken over all neighbors *v* of *x*

direct cost of link from *x* to *v*

The fact behind Bellman-Ford algorithm

$$D_{ij} = \text{minimum } \{(c_{i1} + D_{1j}), (c_{i2} + D_{2j}), \ldots (c_{iN} + D_{Nj})\}$$

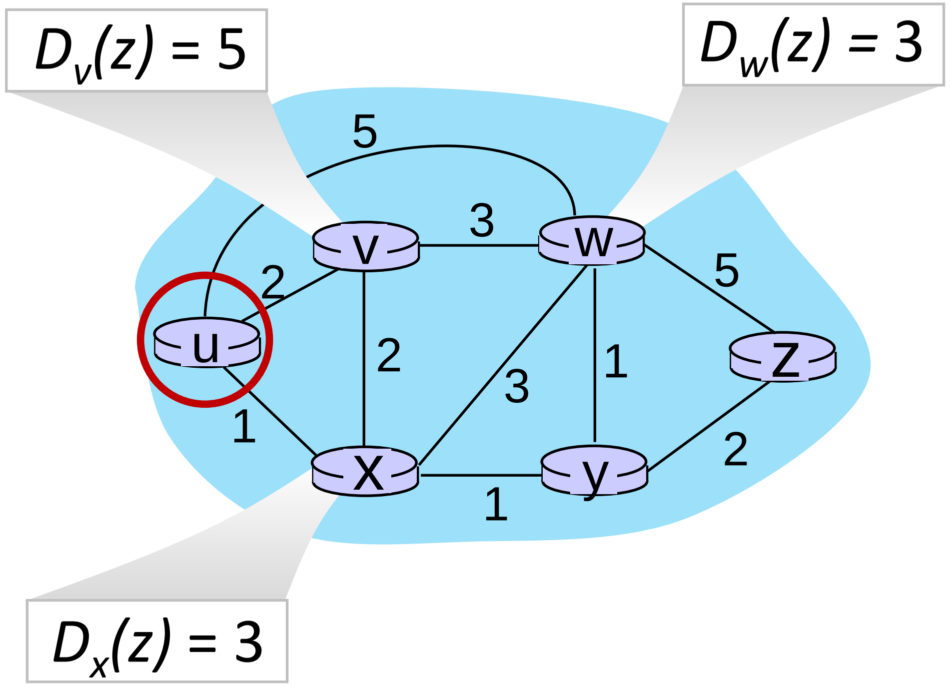


Legend

D_{ij} Shortest distance between i and j
c_{ij} Cost between i and j
N Number of nodes

Bellman-Ford Example

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



Bellman-Ford equation says:

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

node achieving minimum (x) is 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:

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

- the estimate $D_x(y)$ *converge to the actual least cost* $d_x(y)$

Distance vector algorithm:

each node:

wait for (change in local link cost or msg from neighbor)

recompute DV estimates using DV received from neighbor

if DV to any destination has changed, *notify* neighbors

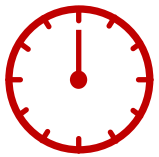
iterative, asynchronous: each local iteration caused by:

- local link cost change
- DV update message from neighbor

distributed, self-stopping: each node notifies neighbors *only* when its DV changes

- neighbors then notify their neighbors – *only if necessary*
- no notification received, no actions taken!

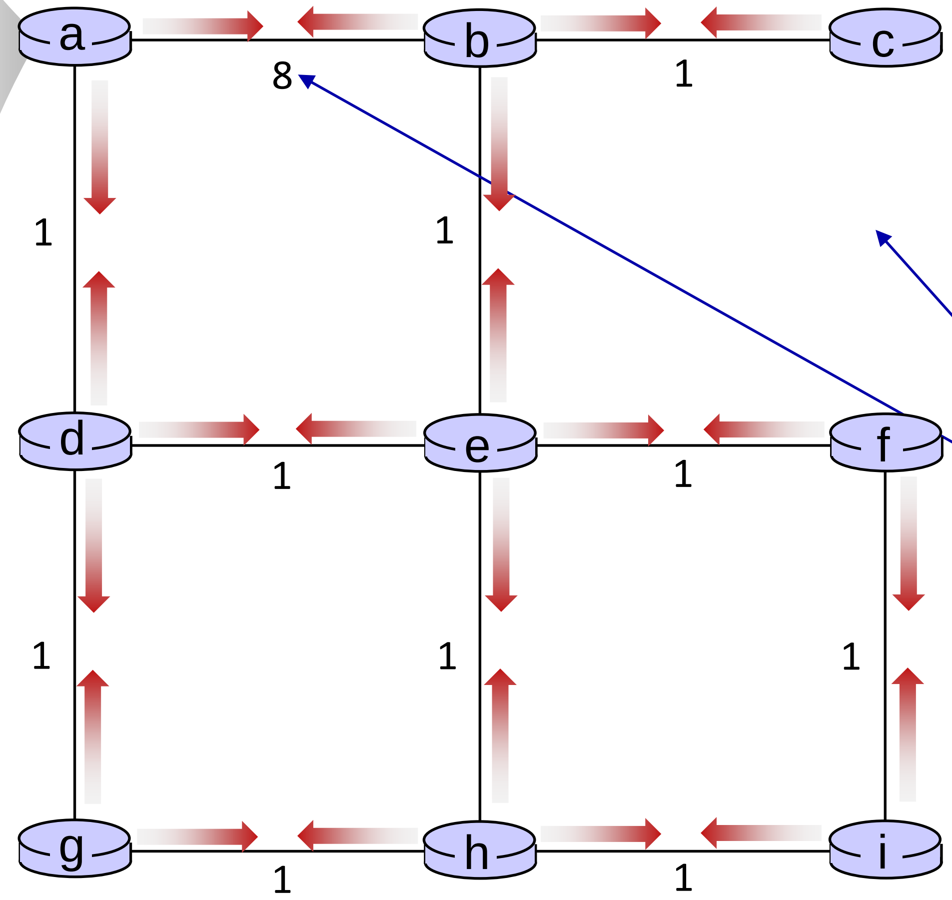
Distance vector: example



t=0

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

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



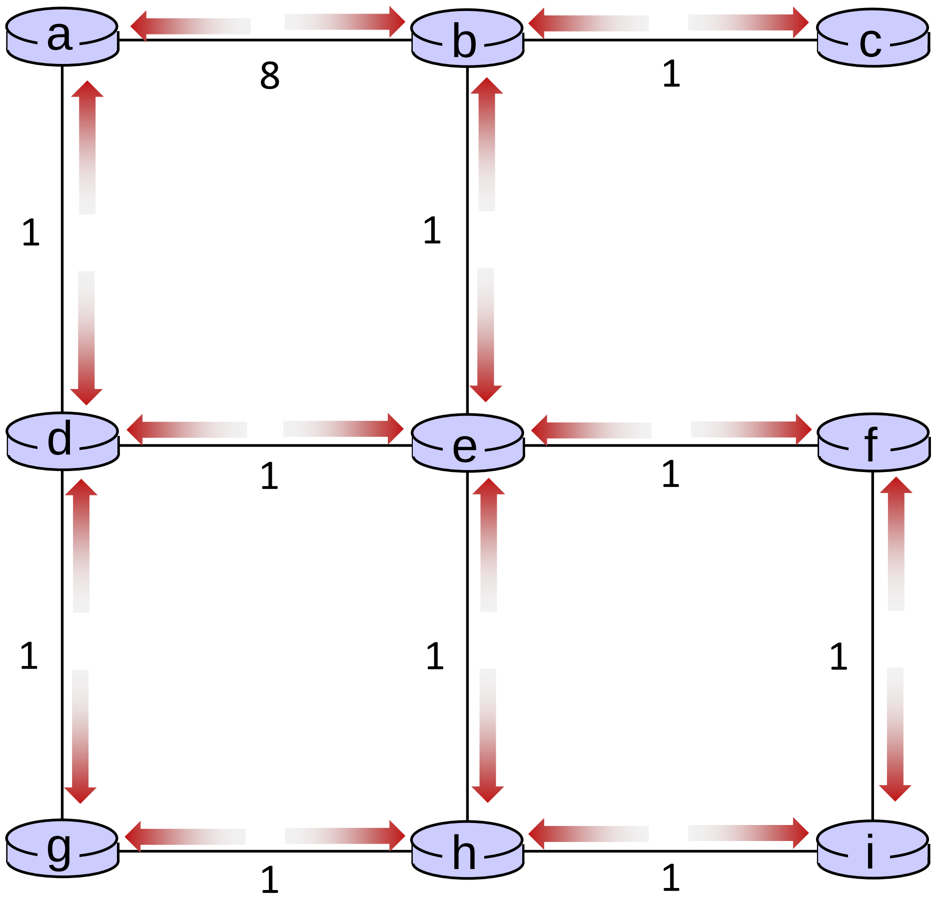
A few asymmetries:

- 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

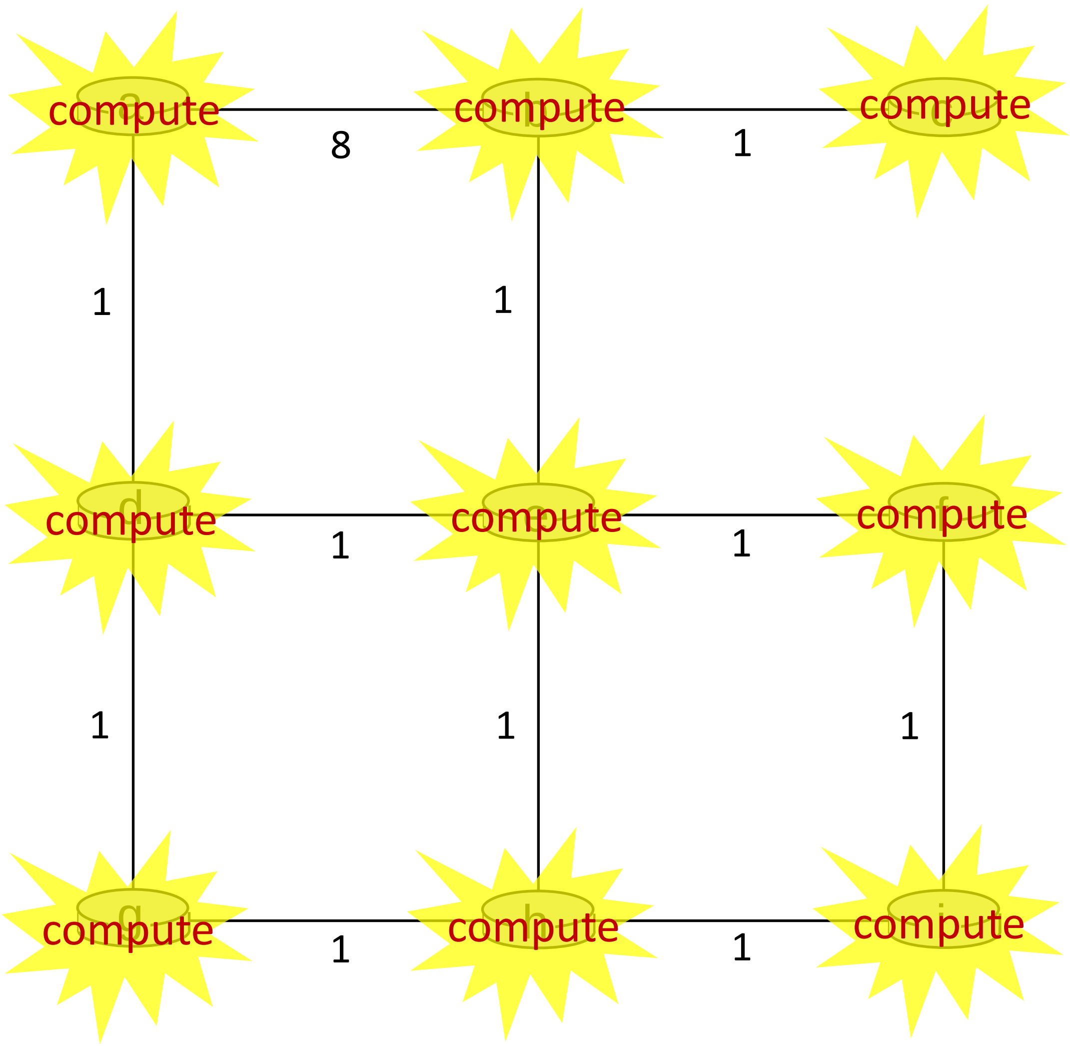


Distance vector example: iteration

t=1

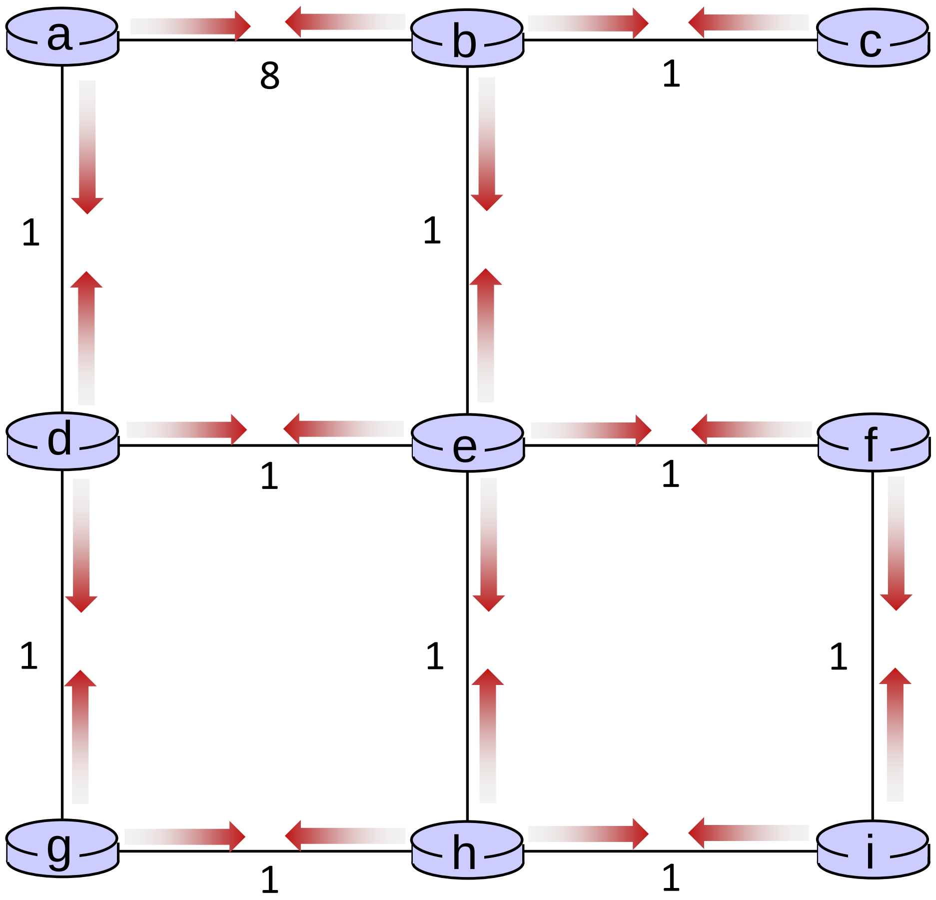
All nodes:

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



Distance vector example: iteration

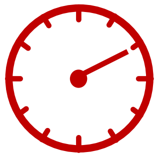
t=1



All nodes:

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

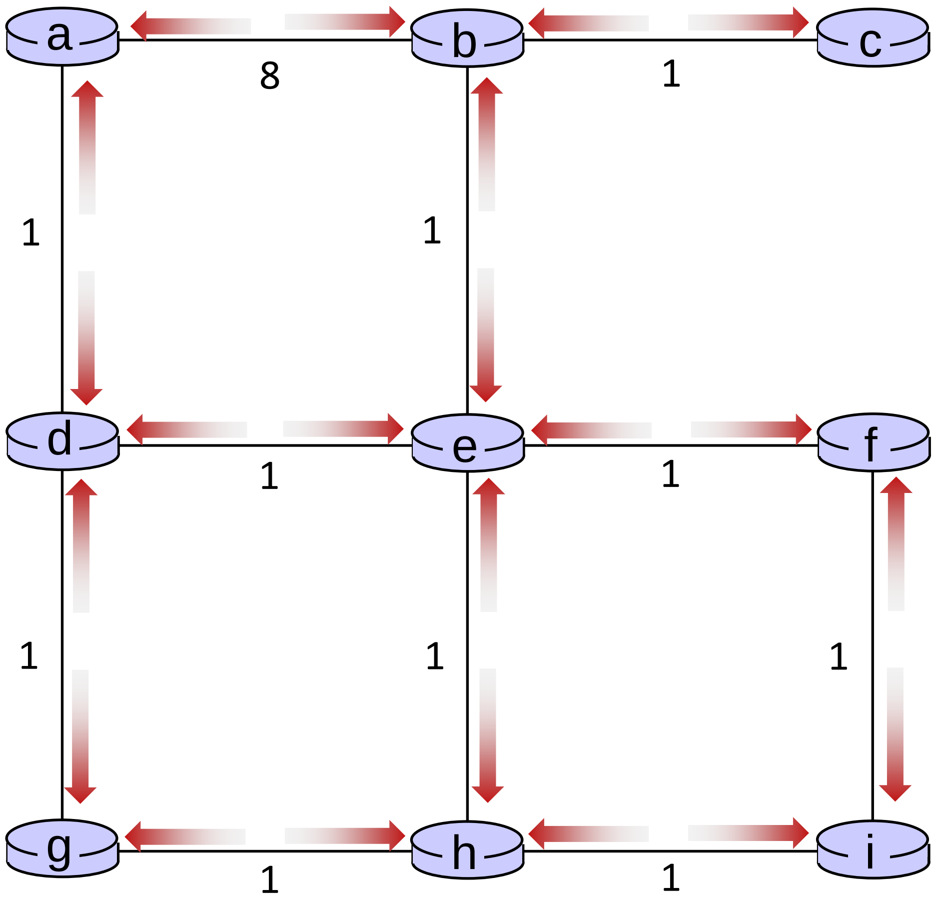
Distance vector example: iteration



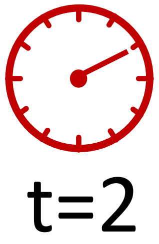
t=2

All nodes:

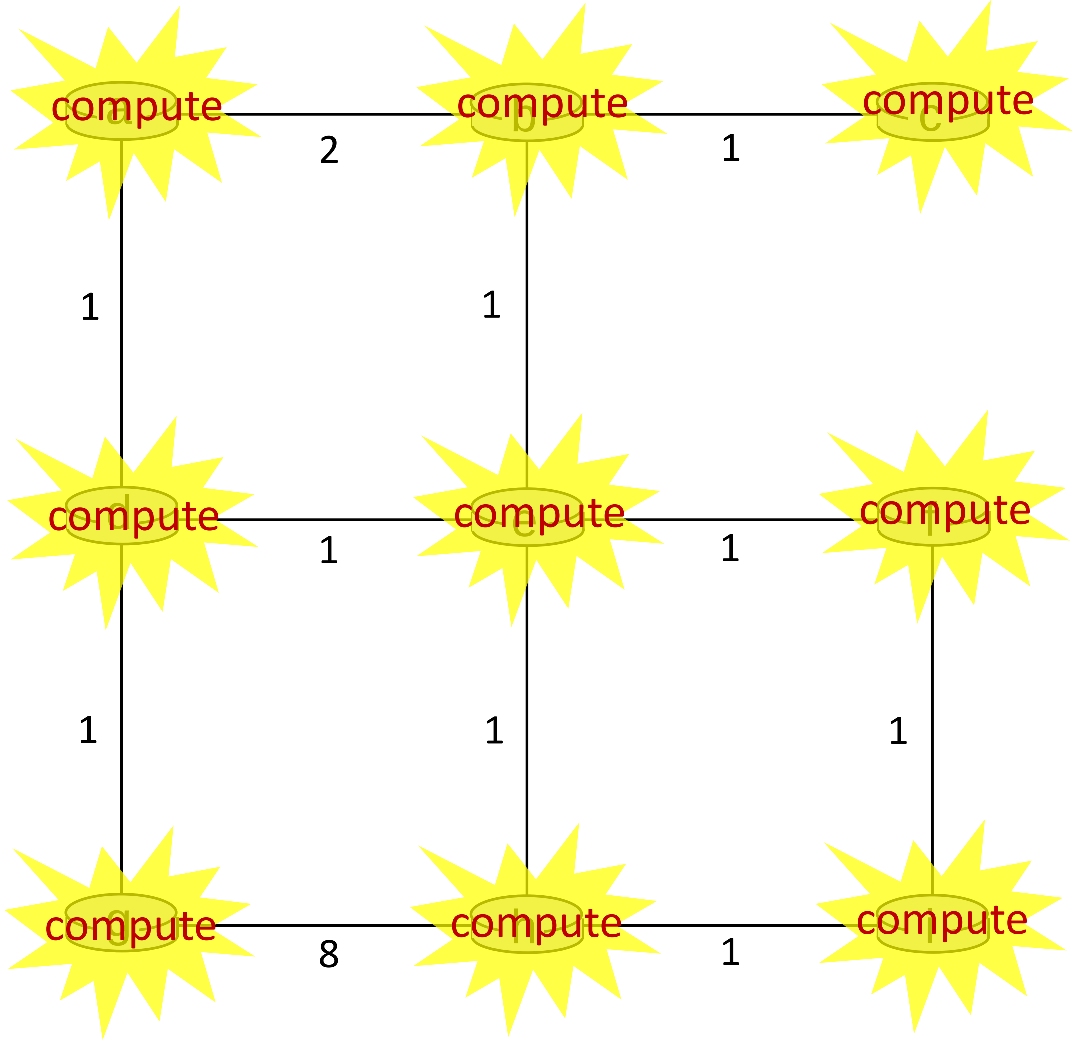
- receive distance vectors from neighbors
- compute their new local distance vector
- send their new local distance vector to neighbors



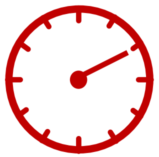
Distance vector example: iteration



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



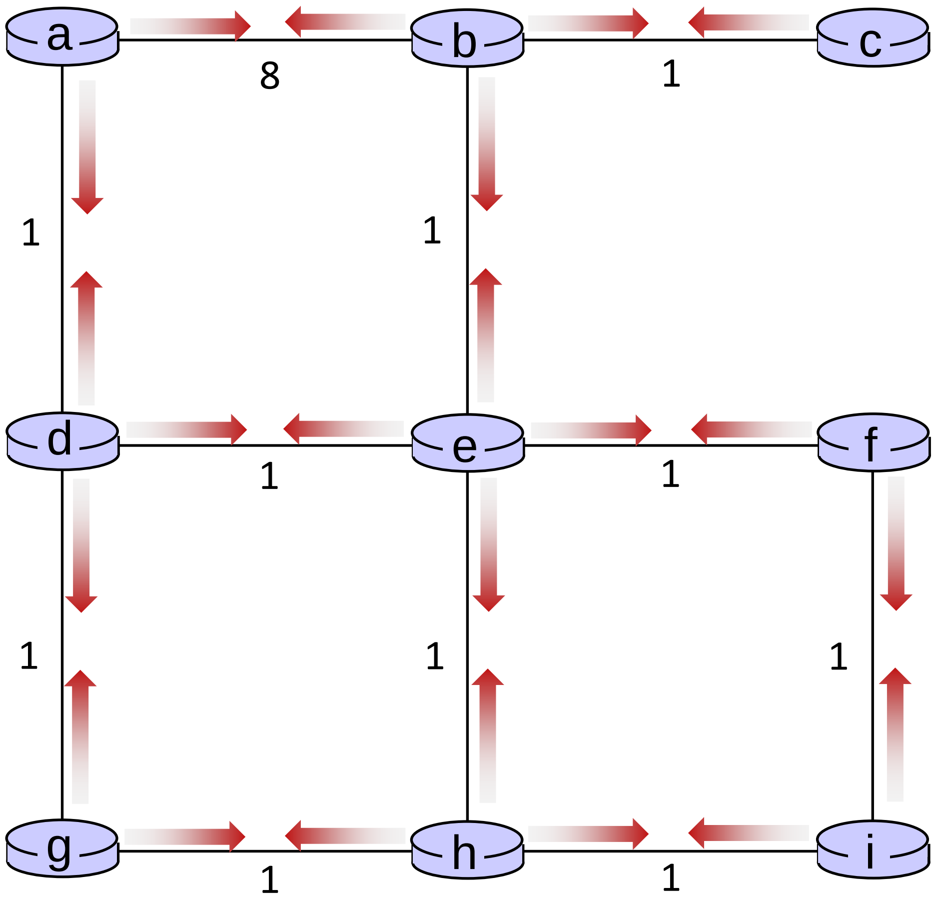
Distance vector example: iteration



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

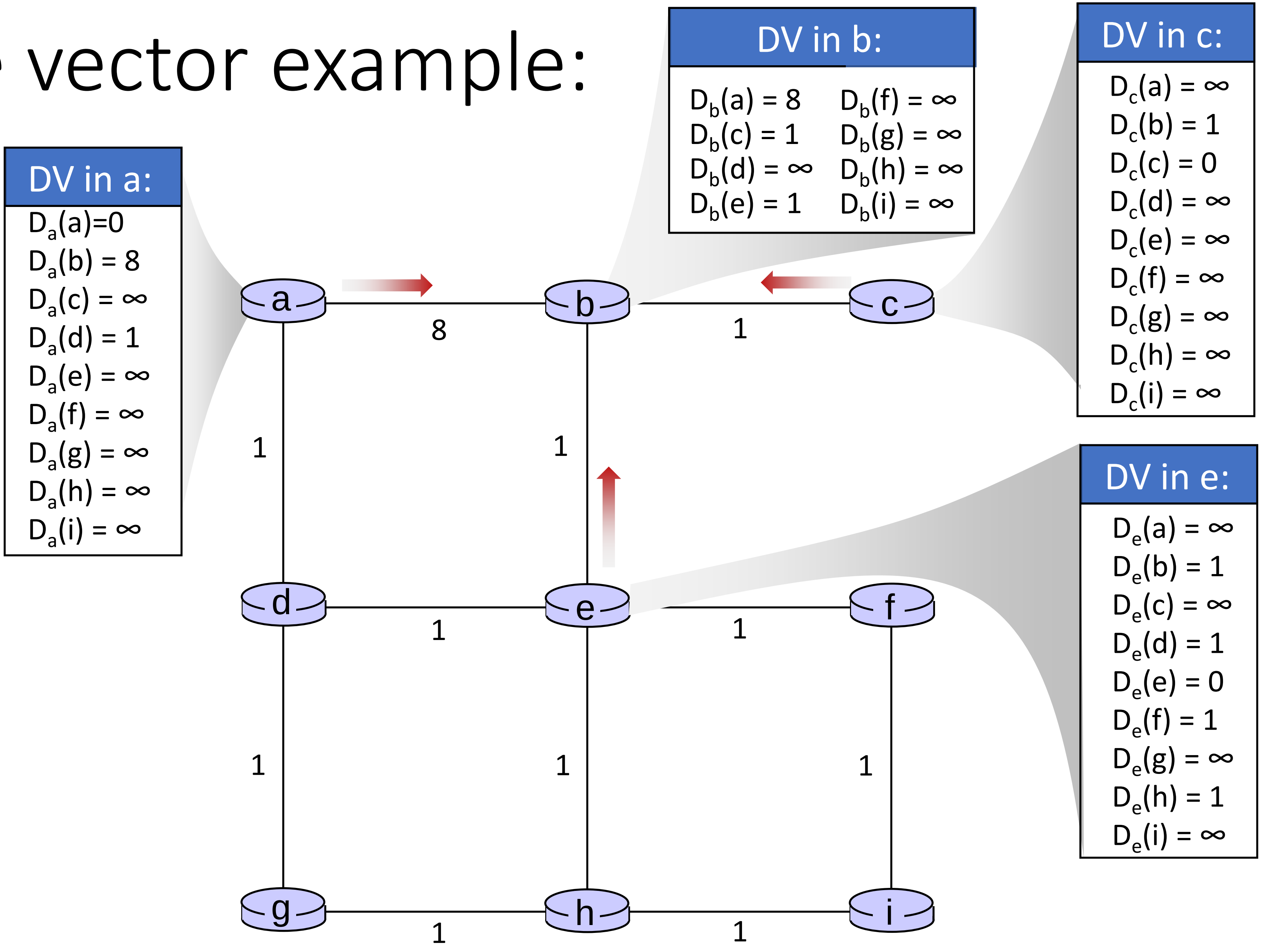
…. and so on

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

Distance vector example:

t=1

- b receives DVs from a, c, e



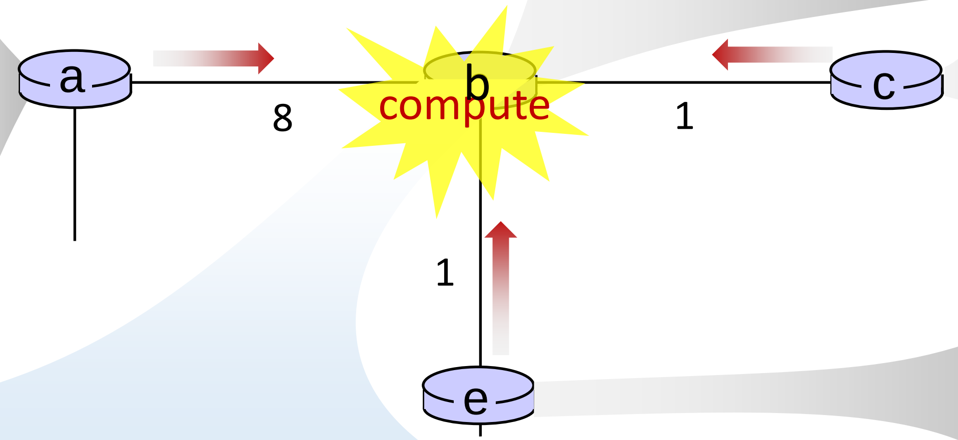
Distance vector example:

t=1

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

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:

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

$D_b(a) = \min\{c_{b,a}+D_a(a),\ c_{b,c}+D_c(a),\ c_{b,e}+D_e(a)\} = \min\{8,\infty,\infty\} = 8$

$D_b(c) = \min\{c_{b,a}+D_a(c),\ c_{b,c}+D_c(c),\ c_{b,e}+D_e(c)\} = \min\{\infty,1,\infty\} = 1$

$D_b(d) = \min\{c_{b,a}+D_a(d),\ c_{b,c}+D_c(d),\ c_{b,e}+D_e(d)\} = \min\{9,2,\infty\} = 2$

$D_b(e) = \min\{c_{b,a}+D_a(e),\ c_{b,c}+D_c(e),\ c_{b,e}+D_e(e)\} = \min\{\infty,\infty,1\} = 1$

$D_b(f) = \min\{c_{b,a}+D_a(f),\ c_{b,c}+D_c(f),\ c_{b,e}+D_e(f)\} = \min\{\infty,\infty,2\} = 2$

$D_b(g) = \min\{c_{b,a}+D_a(g),\ c_{b,c}+D_c(g),\ c_{b,e}+D_e(g)\} = \min\{\infty, \infty, \infty\} = \infty$

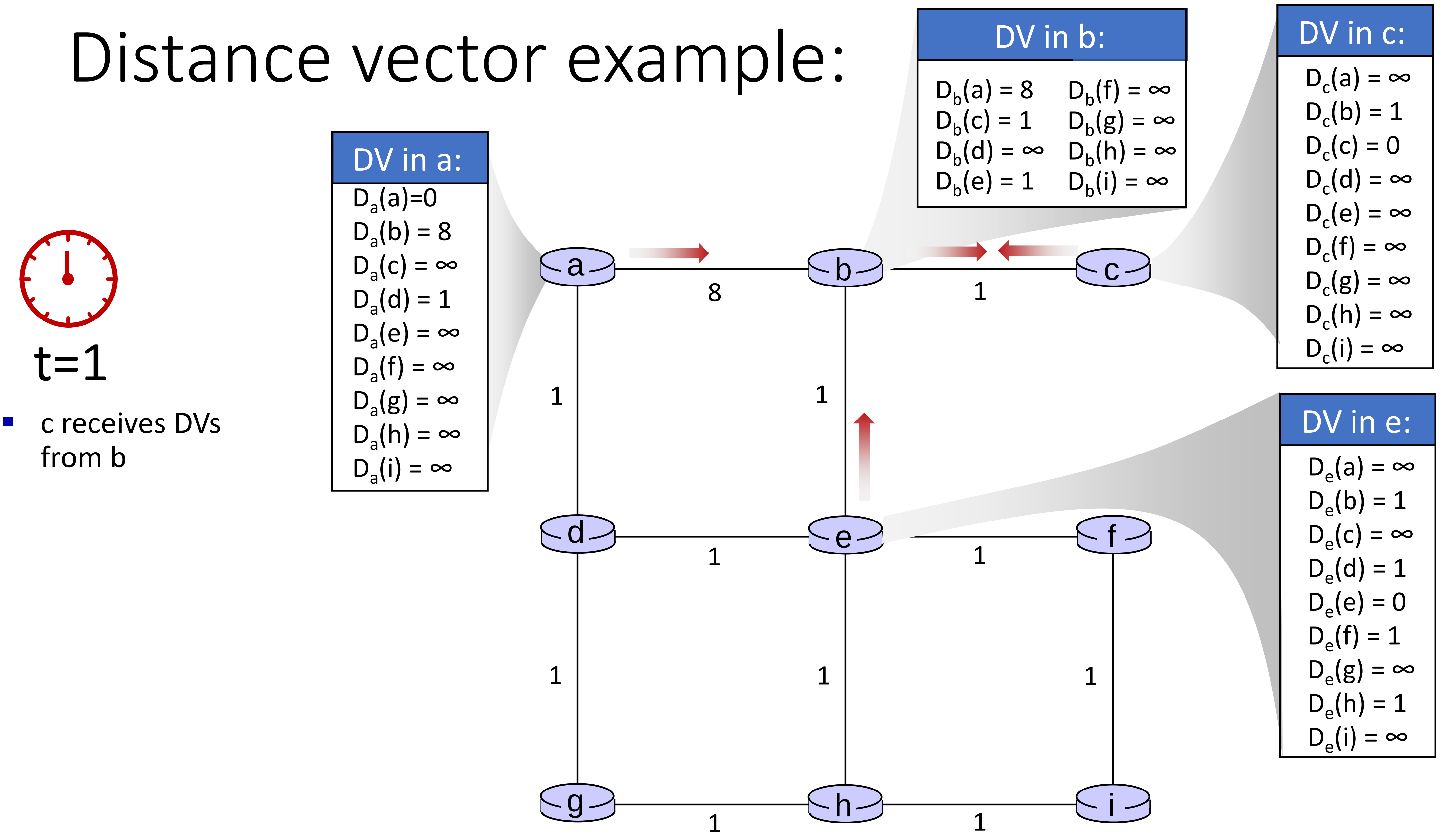
$D_b(h) = \min\{c_{b,a}+D_a(h),\ c_{b,c}+D_c(h),\ c_{b,e}+D_e(h)\} = \min\{\infty, \infty, 2\} = 2$

$D_b(i) = \min\{c_{b,a}+D_a(i),\ c_{b,c}+D_c(i),\ c_{b,e}+D_e(i)\} = \min\{\infty, \infty, \infty\} = \infty$

DV in b:

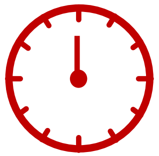
$D_b(a) = 8$	$D_b(f) =2$
$D_b(c) = 1$	$D_b(g) = \infty$
$D_b(d) = 2$	$D_b(h) = 2$
$D_b(e) = 1$	$D_b(i) = \infty$

Distance vector example:



- c receives DVs from b

Distance vector example:

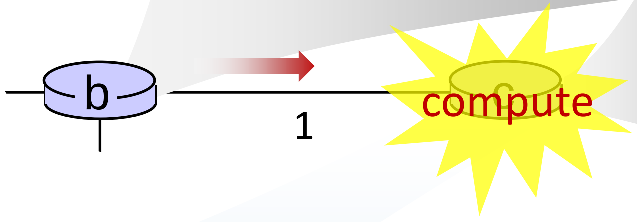


t=1

- c receives DVs from b computes:

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:

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

$D_c(a) = \min\{c_{c,b}+D_b(a)\} = 1 + 8 = 9$

$D_c(b) = \min\{c_{c,b}+D_b(b)\} = 1 + 0 = 1$

$D_c(d) = \min\{c_{c,b}+D_b(d)\} = 1+ \infty = \infty$

$D_c(e) = \min\{c_{c,b}+D_b(e)\} = 1 + 1 = 2$

$D_c(f) = \min\{c_{c,b}+D_b(f)\} = 1+ \infty = \infty$

$D_c(g) = \min\{c_{c,b}+D_b(g)\} = 1+ \infty = \infty$

$D_c(h) = \min\{c_{bc,b}+D_b(h)\} = 1+ \infty = \infty$

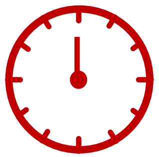
$D_c(i) = \min\{c_{c,b}+D_b(i)\} = 1+ \infty = \infty$

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

- 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 b:

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

DV in e:

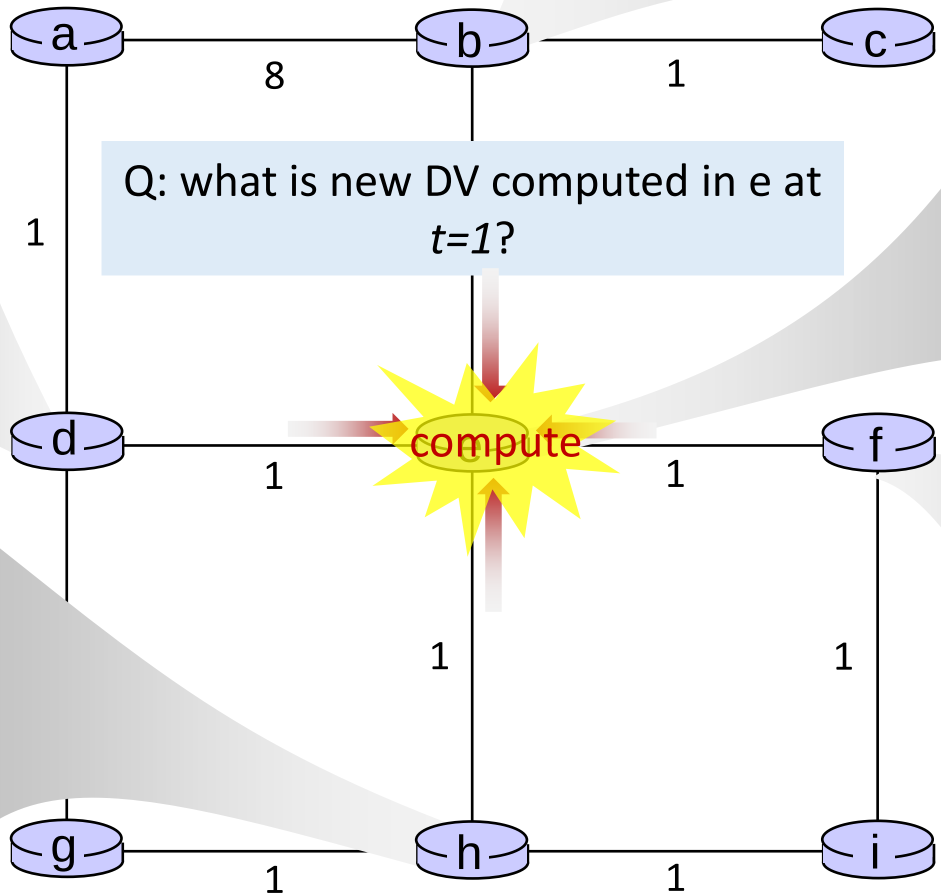
$D_e(a) = \infty$
$D_e(b) = 1$
$D_e(c) = \infty$
$D_e(d) = 1$
$D_e(e) = 0$
$D_e(f) = 1$
$D_e(g) = \infty$
$D_e(h) = 1$
$D_e(i) = \infty$

DV in 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 f:

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



Distance vector: state information diffusion

Iterative communication, computation steps diffuses information through network:

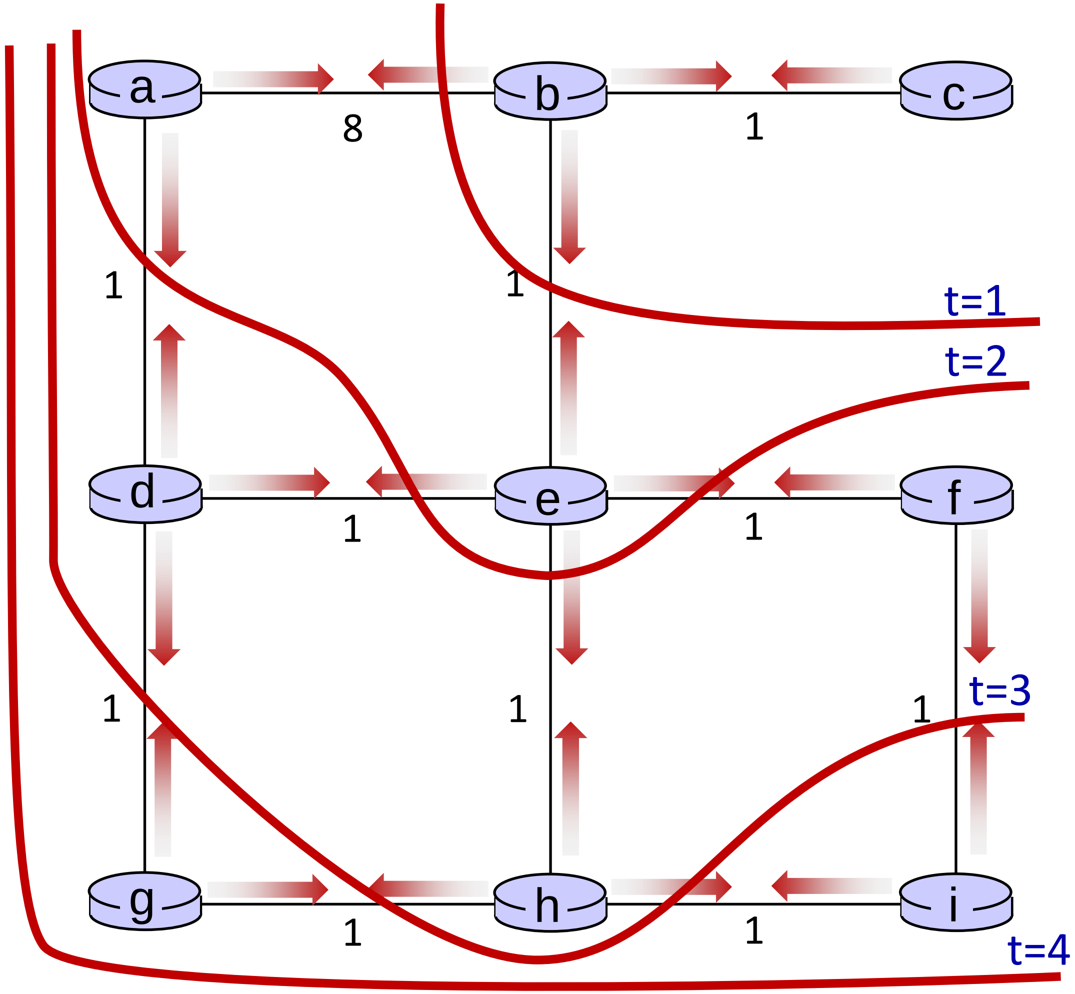
t=0 c's state at t=0 is at c only

t=1 c's state at t=0 has propagated to b, and may influence distance vector computations up to **1** hop away, i.e., at b

t=2 c's state at t=0 may now influence distance vector computations up to **2** hops away, i.e., at b and now at a, e as well

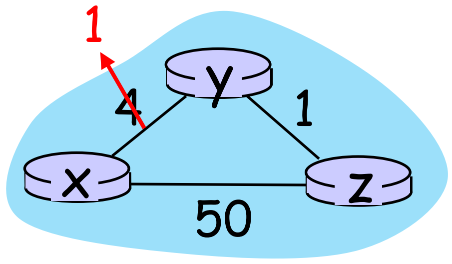
t=3 c's state at t=0 may influence distance vector computations up to **3** hops away, i.e., at b,a,e and now at c,f,h as well

t=4 c's state at t=0 may influence distance vector computations up to **4** hops away, i.e., at b,a,e, c, f, h and now at g,i as well



Distance vector: link cost changes

link cost changes:
- node detects local link cost change
- updates routing info, recalculates local DV
- 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.

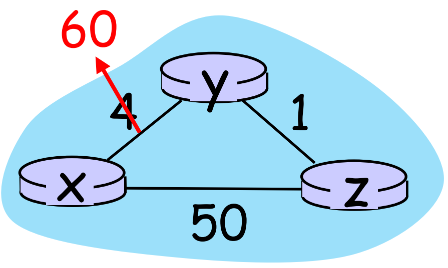
t_2 : *y* receives *z*'s update, updates its distance table. *y*'s least costs do *not* change, so *y* does *not* send a message to *z*.

Distance vector: link cost changes

link cost changes:

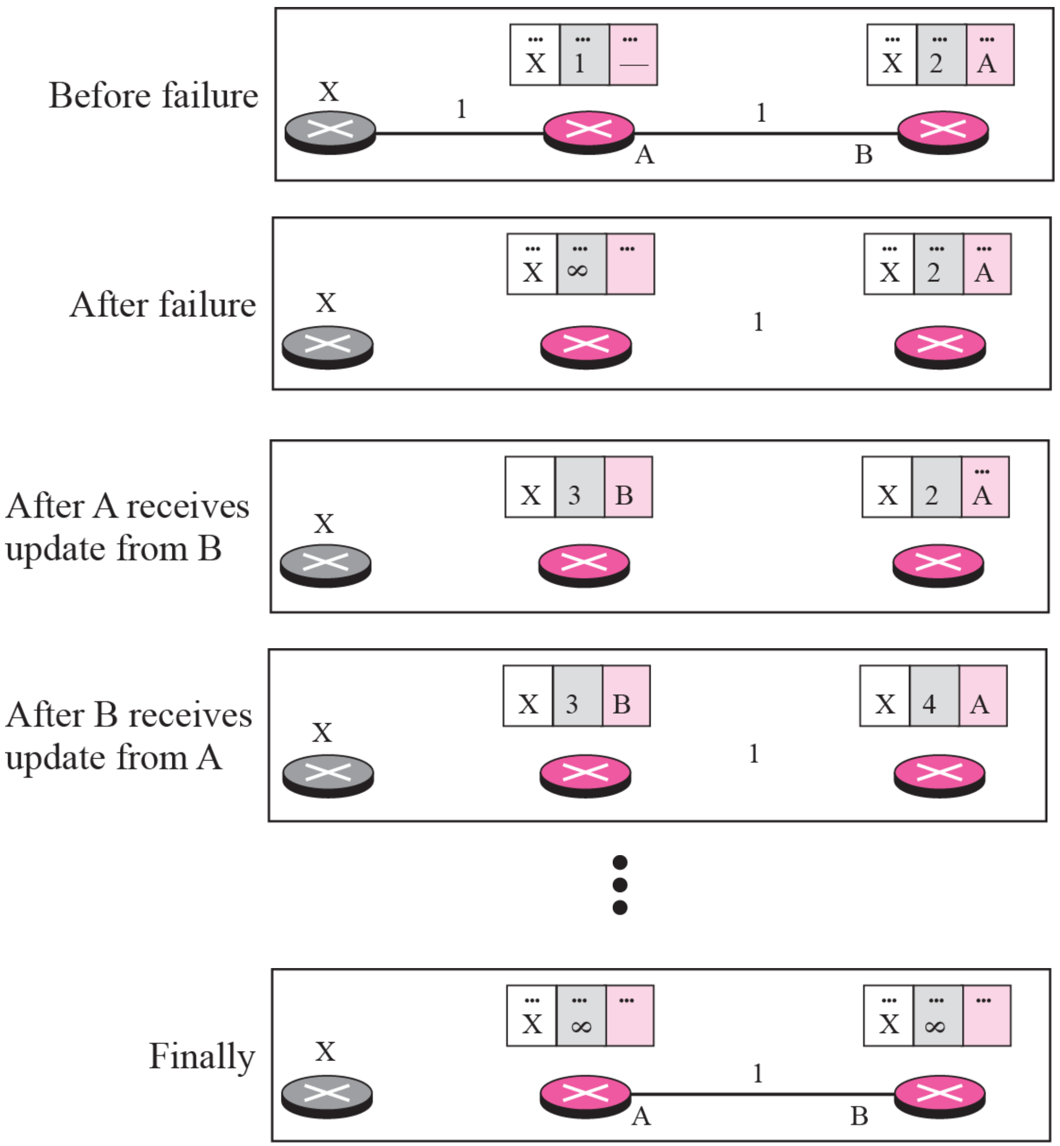
- node detects local link cost change
- "bad news travels slow" – count-to-infinity problem:
  - *y* sees direct link to *x* has new cost 60, but z has said it has a path at cost of 5. So *y* computes "my new cost to x will be 6, via z); notifies *z* of new cost of 6 to *x.*
  - *z* learns that path to *x* via *y* has new cost 6, so *z* computes "my new cost to *x* will be 7 via y), notifies *y* of new cost of 7 to *x.*
  - *y* learns that path to *x* via *z* has new cost 7, so *y* computes "my new cost to *x* will be 8 via y), notifies *z* of new cost of 8 to *x.*
  - *z* learns that path to *x* via *y* has new cost 8, so *z* computes "my new cost to *x* will be 9 via y), notifies *y* of new cost of 9 to *x.*

    ...



- *Distributed algorithms are tricky!*

Two-node instability (Counting to infinity Problem)



Two-Node Instability (1)

- Defining Infinity
  - Most implementations define 16 as infinity
- Split Horizon
  - Instead of flooding the table through each interface, each node sends only part of its table through each interface
  - E.g. node B thinks that the optimum route to reach X is via A, it does not need to advertise this piece of information to A

Two-Node Instability (2)

- Split Horizon and Poison Reverse
  - One drawback of Split Horizon
    - Normally, the DV protocol uses a timer and if there is no news about a route, the node deletes the route from its table
    - In the previous e.g., node A cannot guess that this is due to split horizon or because B has not received any news about X recently
  - Poison Reverse
    - Node B can still advertise the value for X, but is the source of information is A, it can replace the distance with infinity as a warning

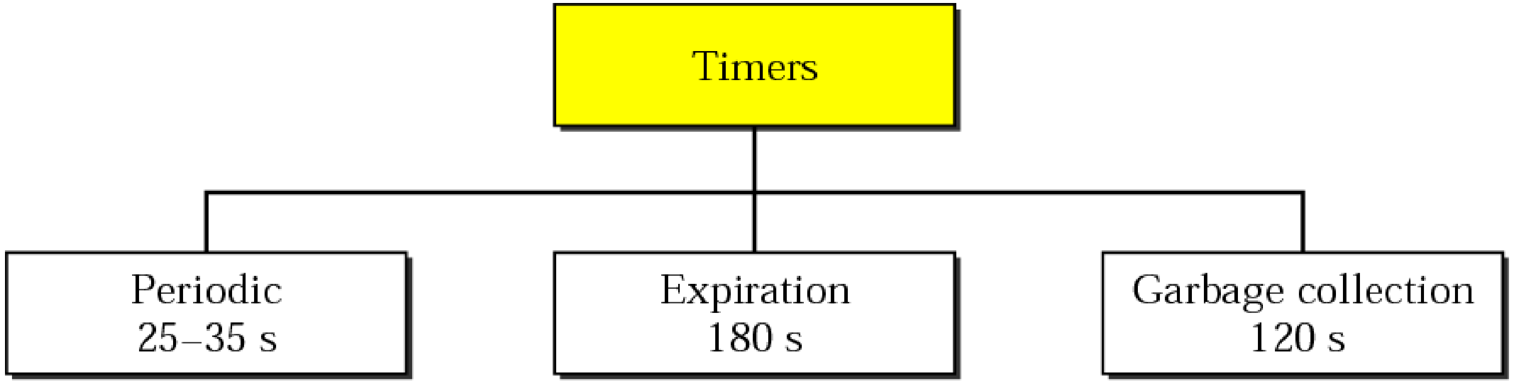
RIP

The Routing Information Protocol (RIP) is an intra-domain (interior) routing protocol used inside an autonomous system. It is a very simple protocol based on distance vector routing. RIP implements distance vector routing directly with some considerations.

RIP messages

- Request
  - A request message is sent by a router that has just come up or by a router that has some time-out entries
  - A request can ask about specific entries or all entries
- Response
  - A response can be either solicited or unsolicited (i.e., periodic) (30s or when there is a change in the routing table)

RIP Timers

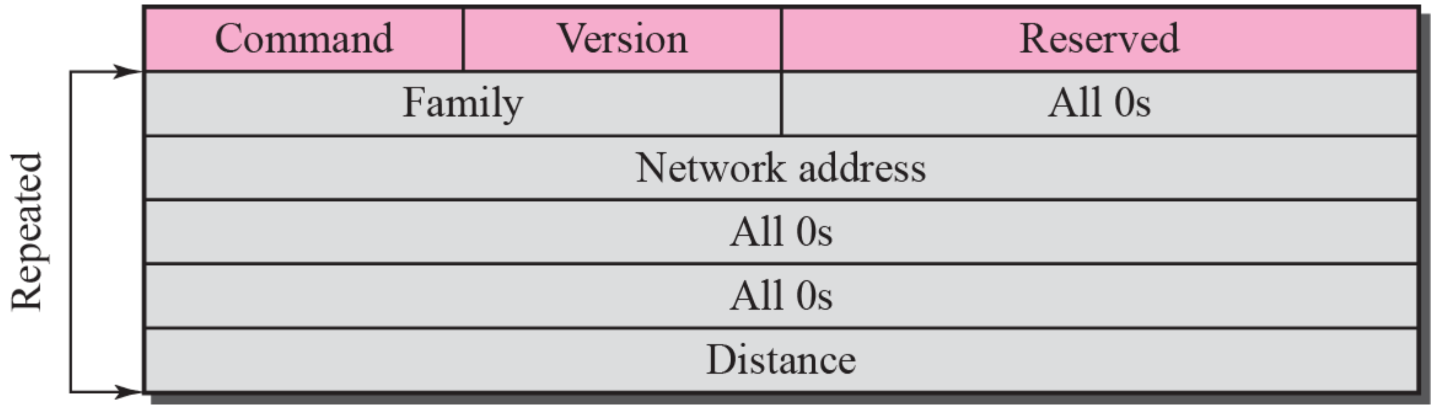


Periodic timer: controls the advertising of regular updates

Expiration timer: governs the validity of a router. When a router receives info, sets timer to 180s. No update within 180s? Route set to 16, which means unreachable.

Garbage collection timer: Set to 120s after route set to 16. When timer expires, then toss route info.

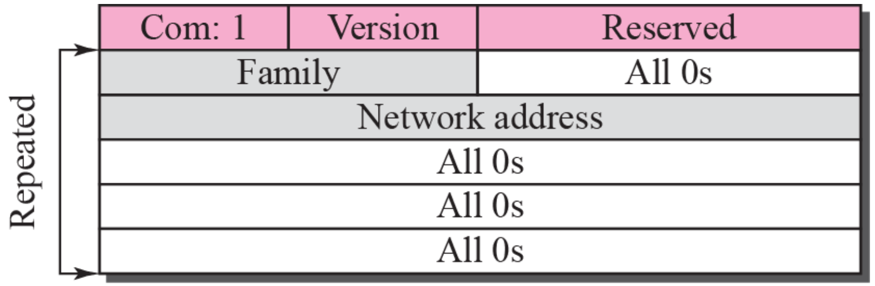
RIPv1 message format



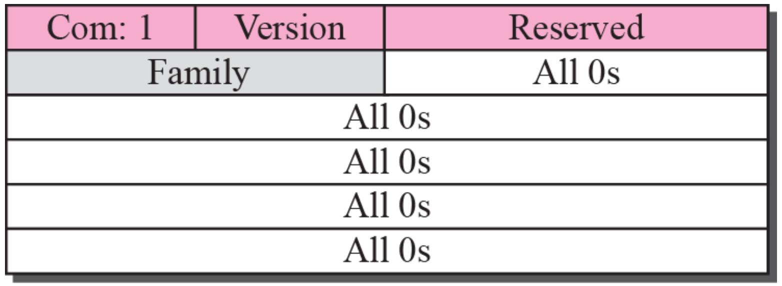
Command: request (1) or response (2)
Version: 1 or 2 (version 2 shown in a couple slides)
Family: TCP/IP has value 2
Network address: address of the destination network
Distance: hop count from the advertising router to the destination network

Request messages

A request message is sent by a router that has just come up or by a router that has some time-out entries.



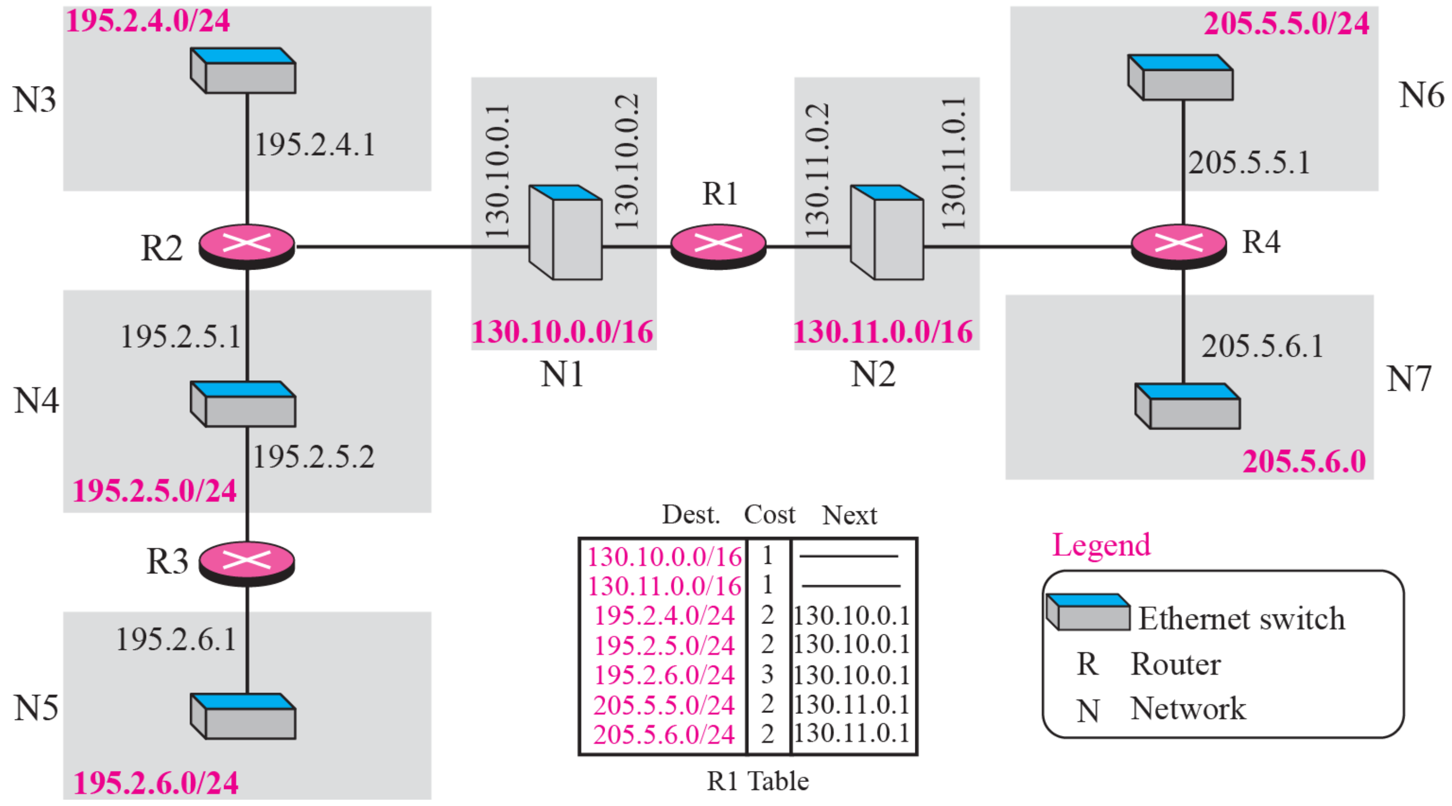
a. Request for some



b. Request for all

A response message is sent in answer to a request (solicited response,
or simply every 30 seconds (unsolicited). Response message format shown in previous slide.

Example of a domain using RIP



Dest.	Cost	Next
130.10.0.0/16	1	——
130.11.0.0/16	1	——
195.2.4.0/24	2	130.10.0.1
195.2.5.0/24	2	130.10.0.1
195.2.6.0/24	3	130.10.0.1
205.5.5.0/24	2	130.11.0.1
205.5.6.0/24	2	130.11.0.1

R1 Table

Dest.	Cost	Next
130.10.0.0/16	1	——
130.11.0.0/16	2	130.10.0.2
195.2.4.0/24	1	——
195.2.5.0/24	1	——
195.2.6.0/24	2	195.2.5.2
205.5.5.0/24	3	130.10.0.2
205.5.6.0/24	3	130.10.0.2

R2 Table

Dest.	Cost	Next
130.10.0.0/16	2	195.2.5.1
130.11.0.0/16	3	195.2.5.1
195.2.4.0/24	2	195.2.5.1
195.2.5.0/24	1	——
195.2.6.0/24	1	——
205.5.5.0/24	4	195.2.5.1
205.5.6.0/24	4	195.2.5.1

R3 Table

Dest.	Cost	Next
130.10.0.0/16	2	130.11.0.2
130.11.0.0/16	1	——
195.2.4.0/24	3	130.11.0.2
195.2.5.0/24	3	130.11.0.2
195.2.6.0/24	4	130.11.0.2
205.5.5.0/24	1	——
205.5.6.0/24	1	——

R4 Table

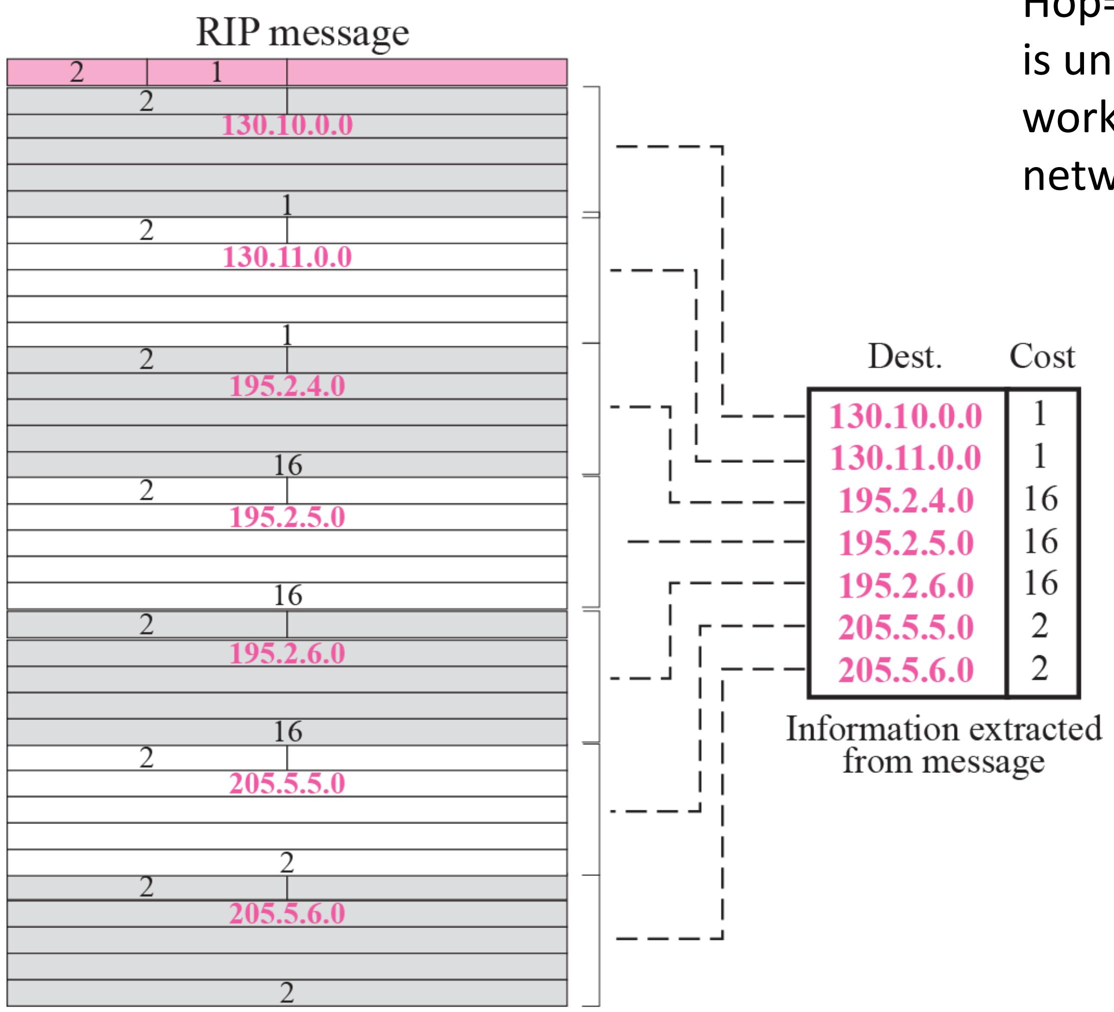
Example

The update message sent from router R1 to router R2 is shown in the next slide for example shown in previous slide. The message is sent out of interface 130.10.0.2.

The message is prepared with the combination of split horizon and poison reverse strategy in mind. Router R1 has obtained information about networks 195.2.4.0, 195.2.5.0, and 195.2.6.0 from router R2. When R1 sends an update message to R2, it replaces the actual value of the hop counts for these three networks with 16 (infinity) to prevent any confusion for R2. The figure also shows the table extracted from the message.

Solution to Example

Hop=16 means dest. is unreachable (only works on smaller networks)



RIPv1

-A classful, Distance Vector (DV) routing protocol

 -Class A : 255.0.0.0 (1.0.0.0-126.255.255.255)

 -Class B : 255.255.0.0 (128.0.0.0-191.255.255.255)

 -Class C : 255.255.255.0 (192.0.0.0-233.255.255.255)

-Does not send subnet masks in routing updates

-Metric = hop count

-Routes with a hop count > 15 are unreachable

-Updates are **broadcast** every 30 seconds

RIPv1 (Encapsulated Message)

Encapsulated RIPv1 Message

Data Link Frame Header	IP Packet Header	UDP Segment Header	RIP Message (504 bytes; Up to 25 routes)

Data Link Frame
MAC Destination Address = Broadcast: FF-FF-FF-FF-FF-FF
MAC Source Address = Address of sending interface

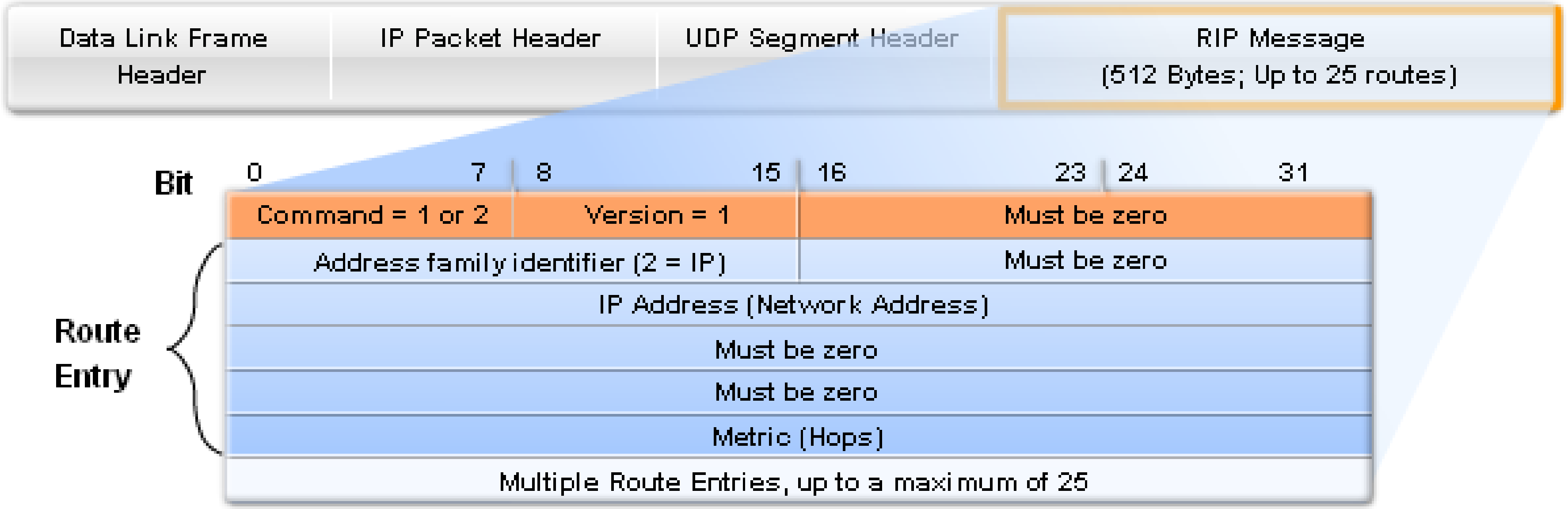
IP Packet
IP Source Address = Address of sending interface
IP Destination Address = Broadcast: 255.255.255.255
Protocol field = 17 for UDP

UDP Segment
Source Port = 520
Destination Port = 520

RIP Message:
Command: Request (1); Response (2)
Version = 1
Address Family ID = 2 for IP
Routes: Network IP Address
Metric: Hop Count

Source: CISCO

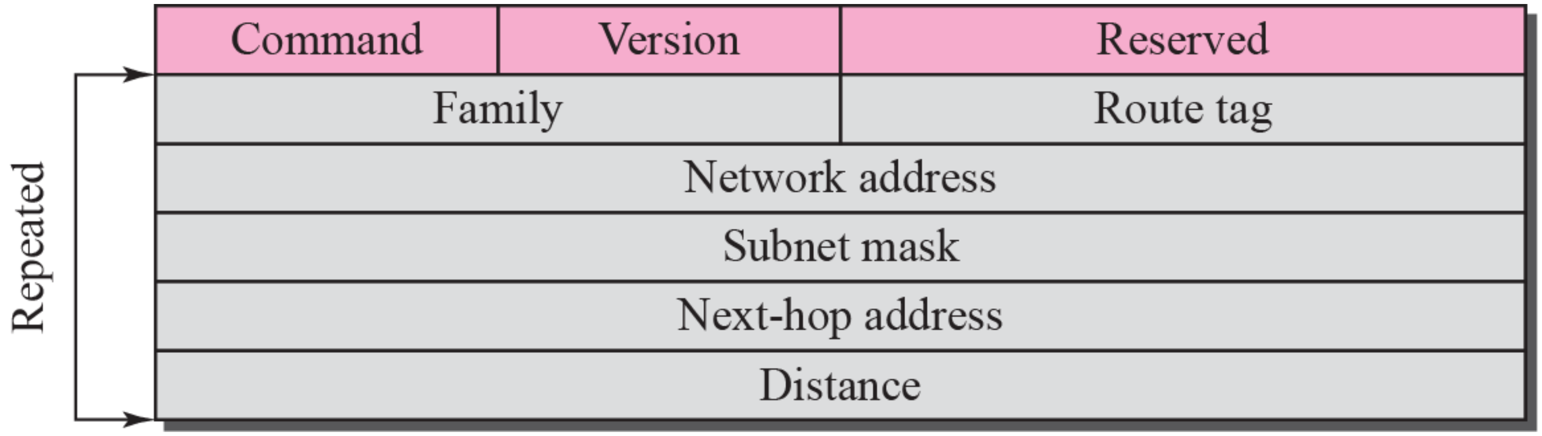
RIPv1 (RIPv1 Message)



Command	1 for a Request or 2 for a Reply.
Version	1 for RIP v 1 or 2 for RIP v 2.
Address Family Identifier	2 for IP unless a Request is for the full routing table in which case, set to 0.
IP Address	The address of the destination route, which may be a network, subnet, or host address.
Metric	Hop count between 1 and 16. Sending router increases the metric before sending out message.

Source: CISCO

RIP version 2 format



Command: request (1) or response (2)
Version: 1 or 2 (version 2 shown in a couple slides)
Family: TCP/IP has value 2
Network address: address of the destination network
Distance: hop count from the advertising router to the destination network
Route tag: Routes imported from EGP or BGP should be able to have their Route Tag either set to an arbitrary value, or at least to the number of the Autonomous System from which the routes were learned.

Ref: https://tools.ietf.org/html/rfc2453

RIP uses the services of UDP on well-known port 520.

Limitations of RIP/DV

- Count to infinity problem. RIP limits the hop count to 15 hops. (Not suitable in large network where number of routers are more than 15)
- The algorithm is slow to converge, needs information from all the nodes.
- Signaling overhead – Periodic broadcasting of DV.