UNIVERSITY OF WATERLOO

CS 456/656
Computer Networks

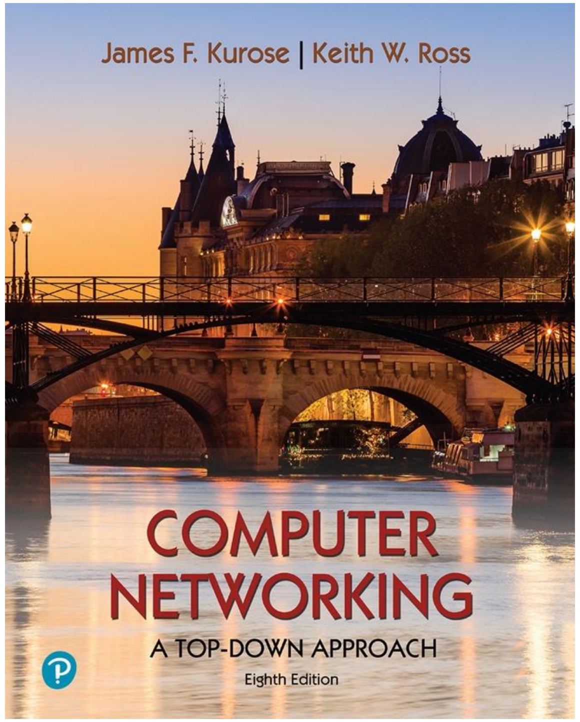
Lecture 10: Network Layer – Part 2

Mina Tahmasbi Arashloo and Uzma Maroof

Fall 2025

A note on the slides

Adapted from the slides that accompany this book.



Computer Networking: A Top-Down Approach
8th edition
Jim Kurose, Keith Ross
Pearson, 2020

Thanks for filling out the survey!

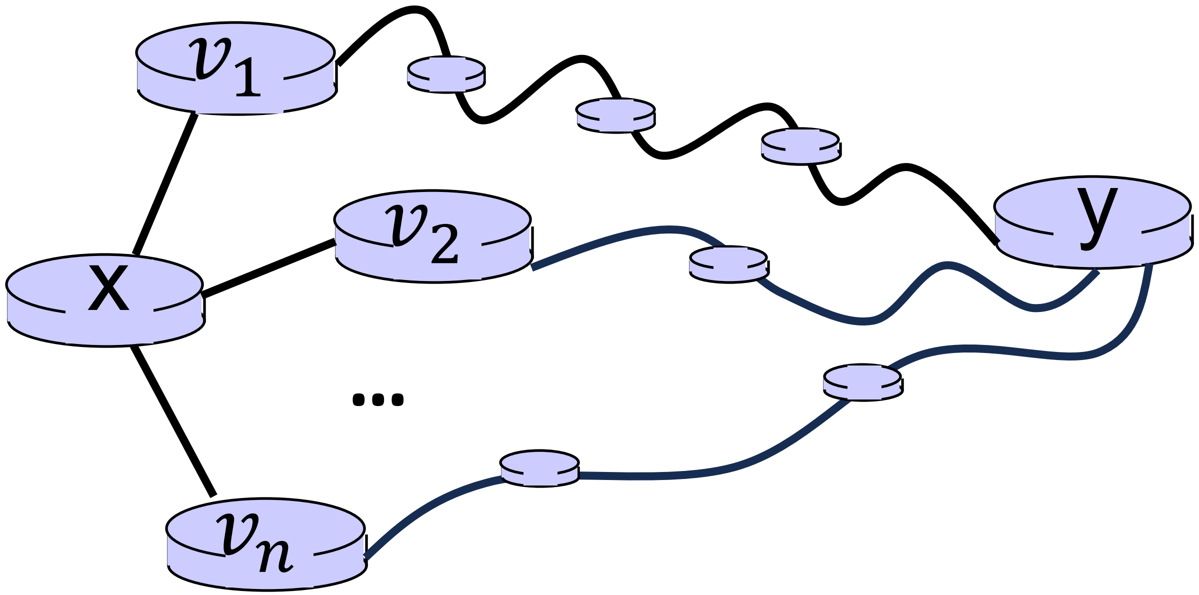
- If you have not, there is still time
- We’ll discuss the results and potential upcoming changes soon

Network layer: roadmap

- Network layer overview
- Routing algorithms
  - Link state
  - <u>Distance vector</u>
- Network layer in the Internet

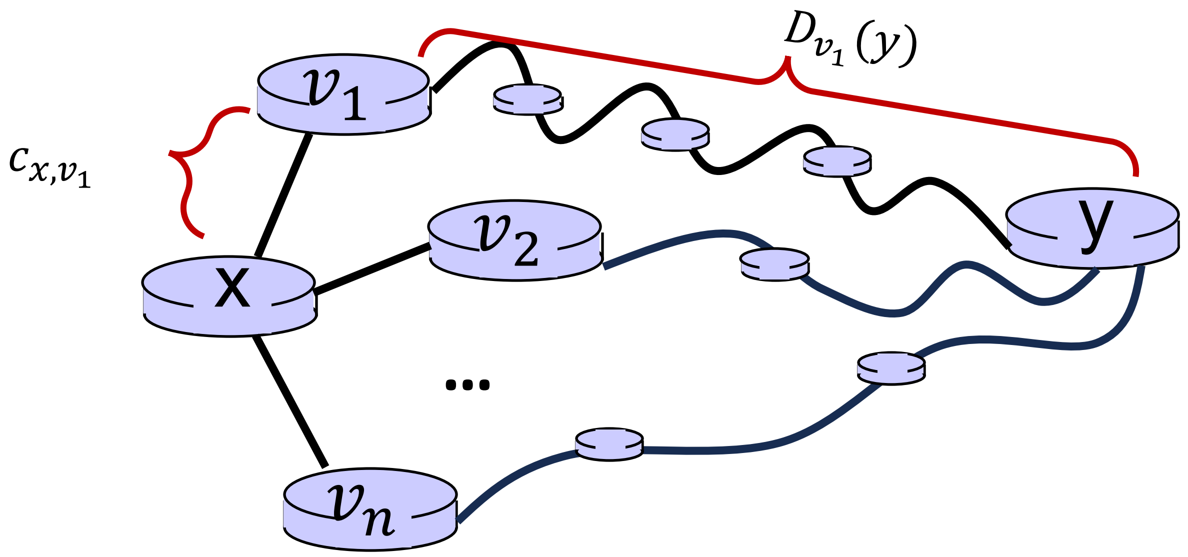
Distance vector routing algorithms

- Suppose node node x has n neighbors, $v_1, v_2, \ldots, v_n$
- The least-cost path from node x to node y will pass one of x's neighbors.



Distance vector routing algorithms

- To find its least-cost path to y, x doesn't necessarily need to build the entire network graph.
- It only need to know
  - $D_{v_i}(y)$: the distance from v_i to y
  - c_{x,v_i} : the cost of the direct link from x to v_i



Distance vector algorithm

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

Bellman-Ford equation

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

Then:

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

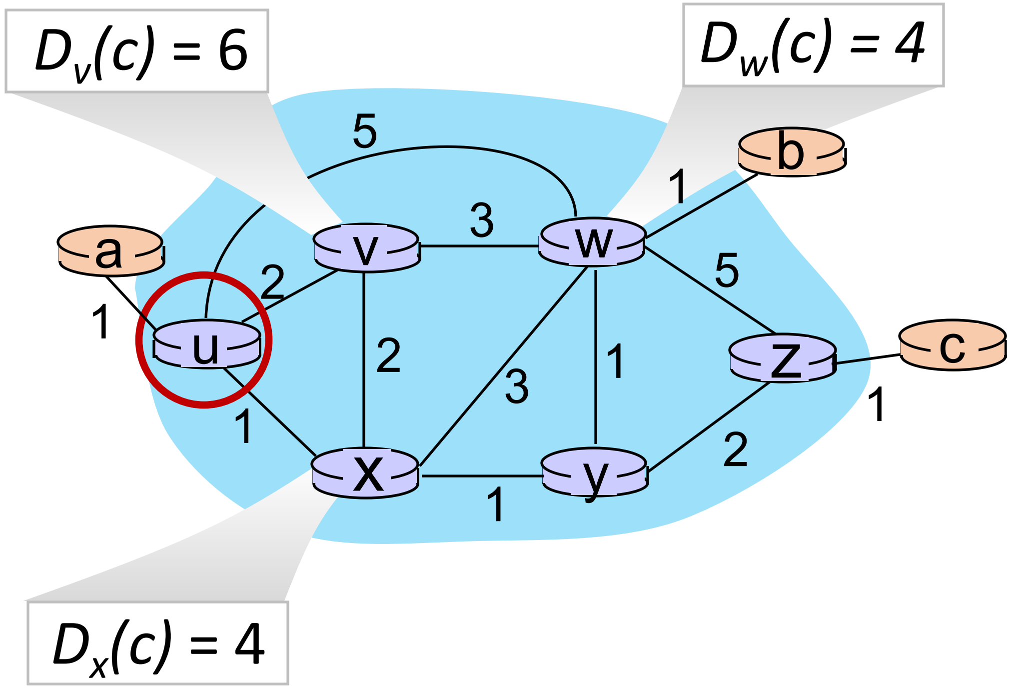
v's estimated least-cost-path cost to *y*

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

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

Bellman-Ford example

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



Bellman-Ford equation says:

$$D_u(c) = \min \{ c_{u,v} + D_v(c), \\ c_{u,x} + D_x(c), \\ c_{u,w} + D_w(c) \}$$

$$= \min \{2 + 6, \\ 1 + 4, \\ 5 + 4\} = 5$$

node achieving minimum (node x) is next hop on estimated least-cost path to destination (node c)

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:

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

- under minor, natural conditions, 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: iteration

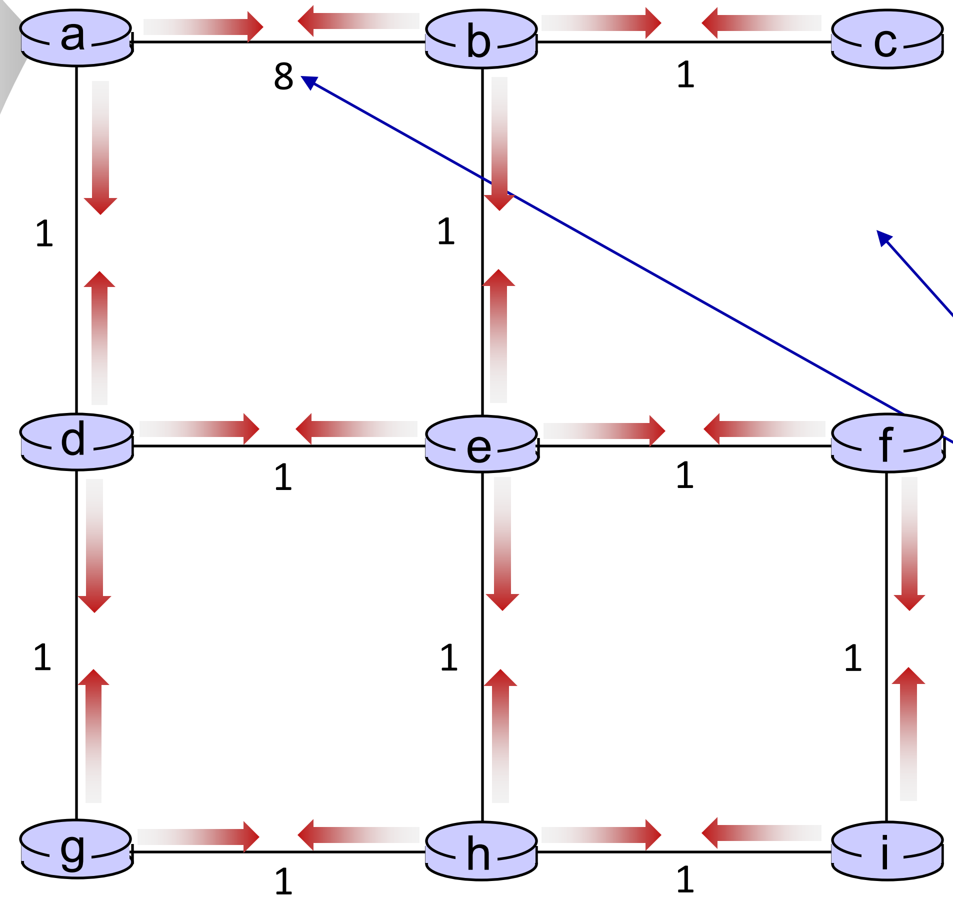
- We will walk through an example of distance vector routing
- For simplicity, we are not adding the end-host (orange) nodes to the example
- They do not participate in routing
- But, the routers will include the distance to them in their advertised distance vectors.

Distance vector example: iteration

t=0

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

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

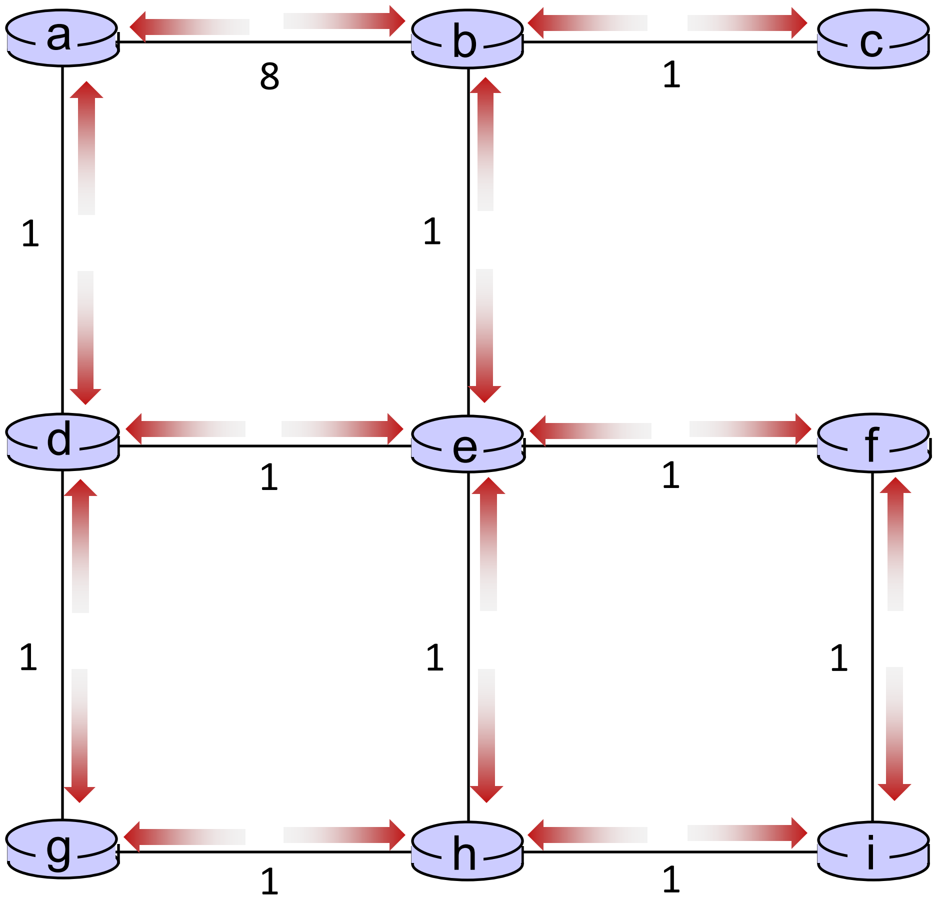


A few asymmetries:
- missing link
- larger cost

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

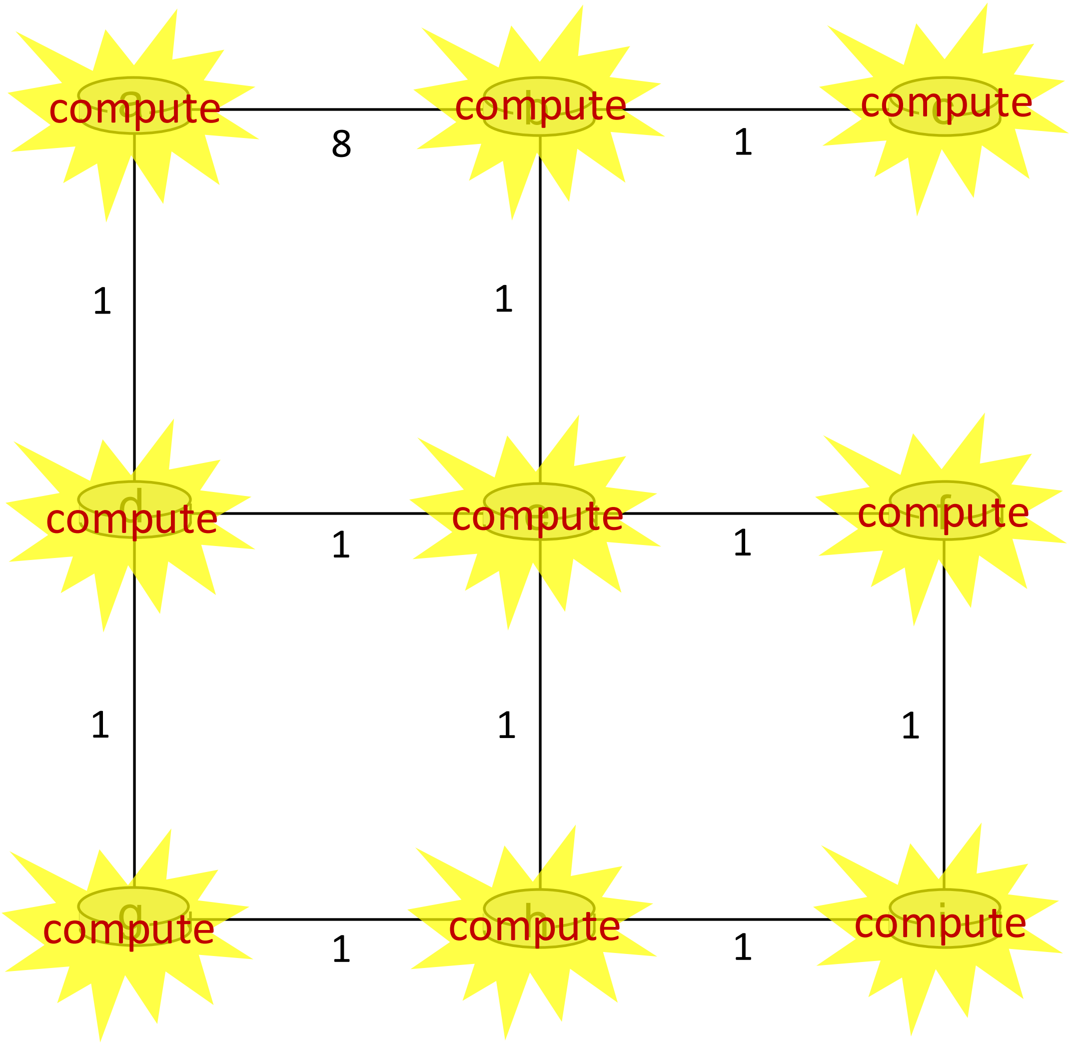


Distance vector example: iteration

t=1

All nodes:

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

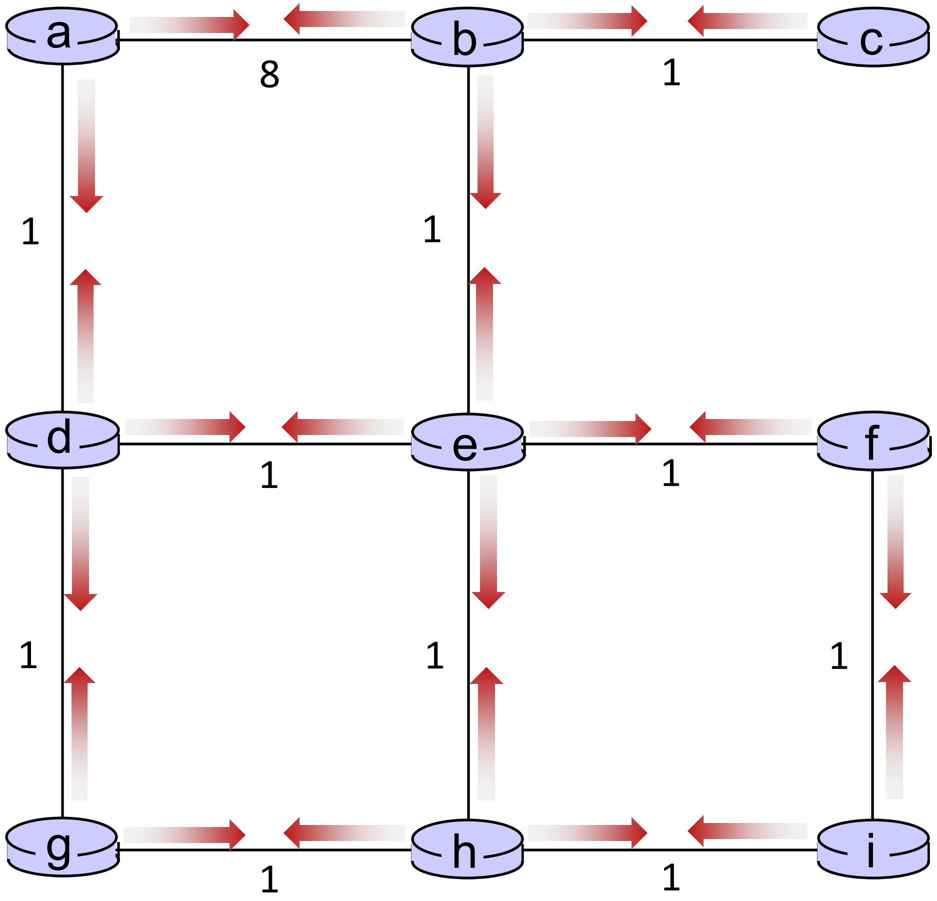


Distance vector example: iteration

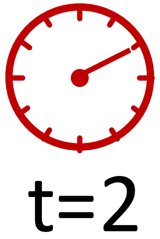
t=1

All nodes:

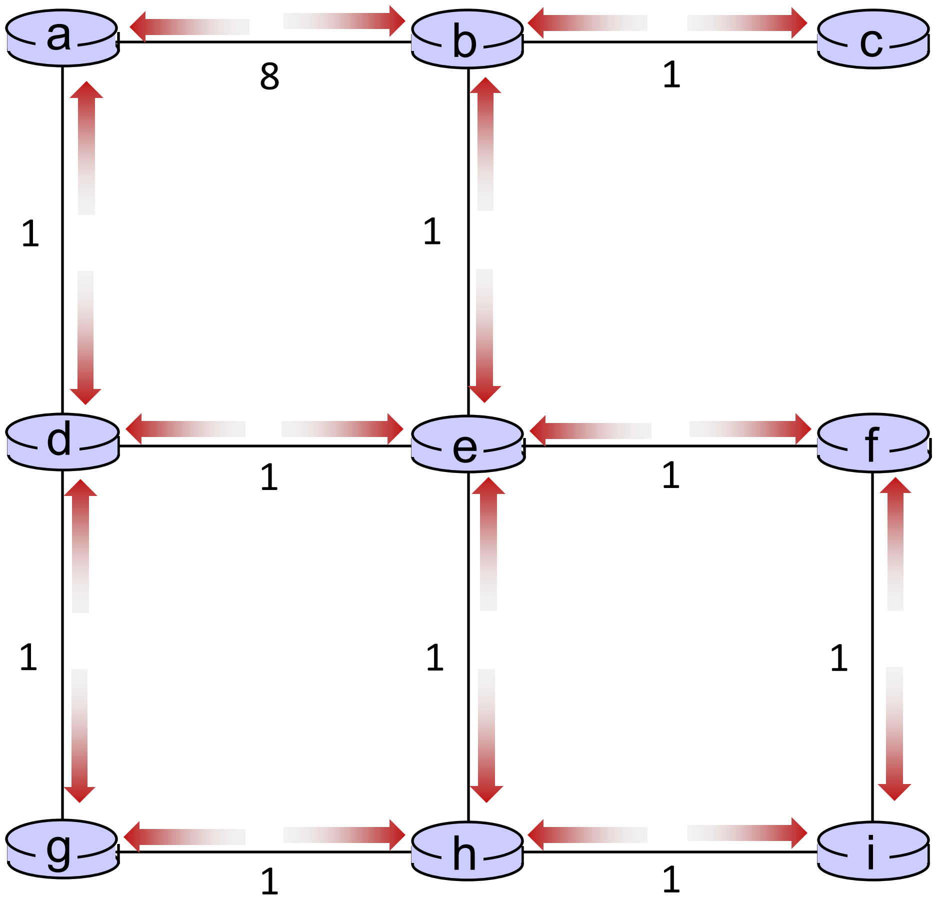
- receive distance vectors from neighbors
- compute their new local distance vector
- send their new local distance vector to neighbors



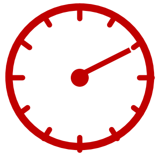
Distance vector example: iteration



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



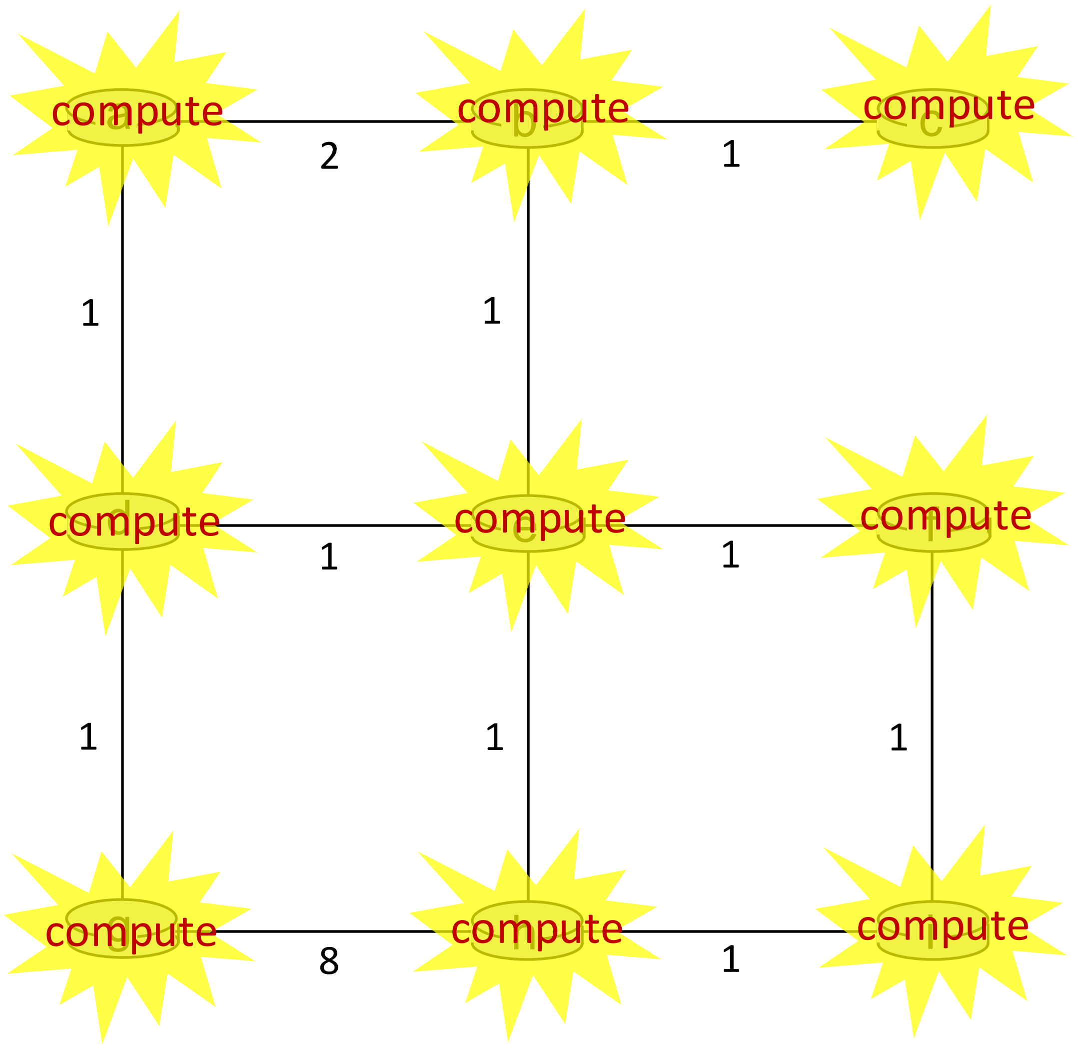
Distance vector example: iteration



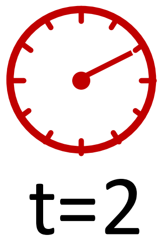
t=2

All nodes:

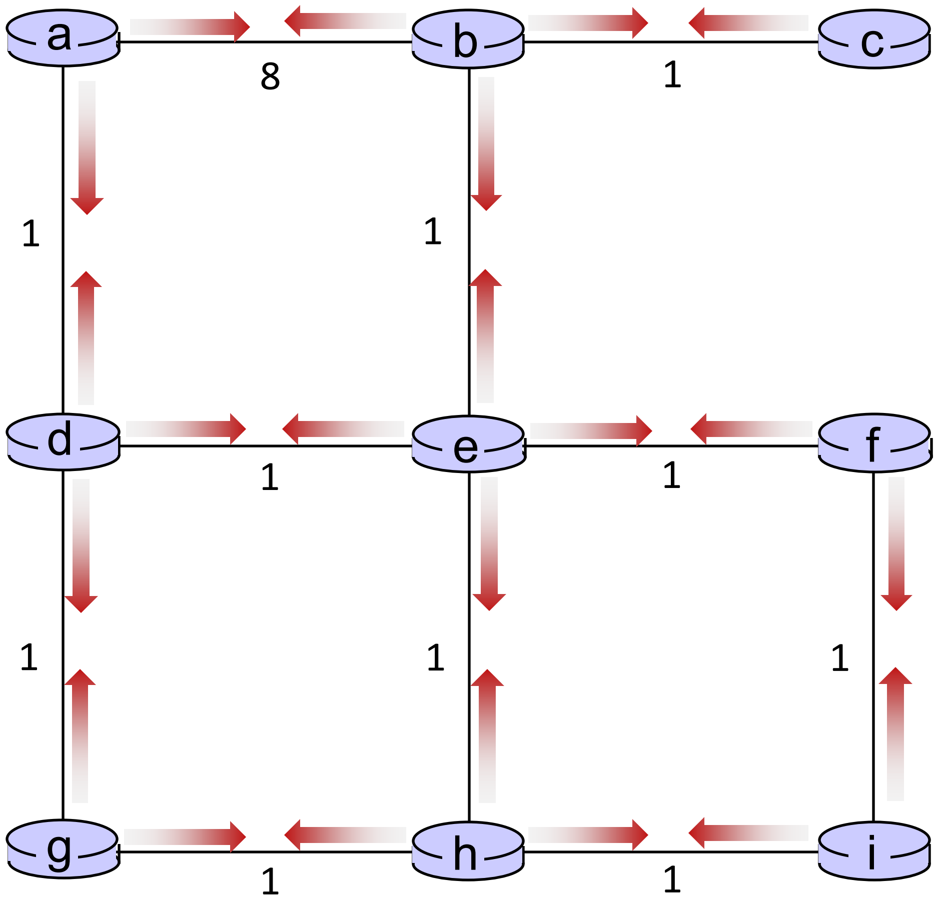
- receive distance vectors from neighbors
- compute their new local distance vector
- send their new local distance vector to neighbors



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

…. and so on

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

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

Distance vector example: iteration

…. and so on

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

Let's look at the computation at node b at t = 1
Remember, b's neighbors have sent b their DV record version at t = 0

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

Distance vector example:

t=1

- b receives DVs from a, c, e

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:

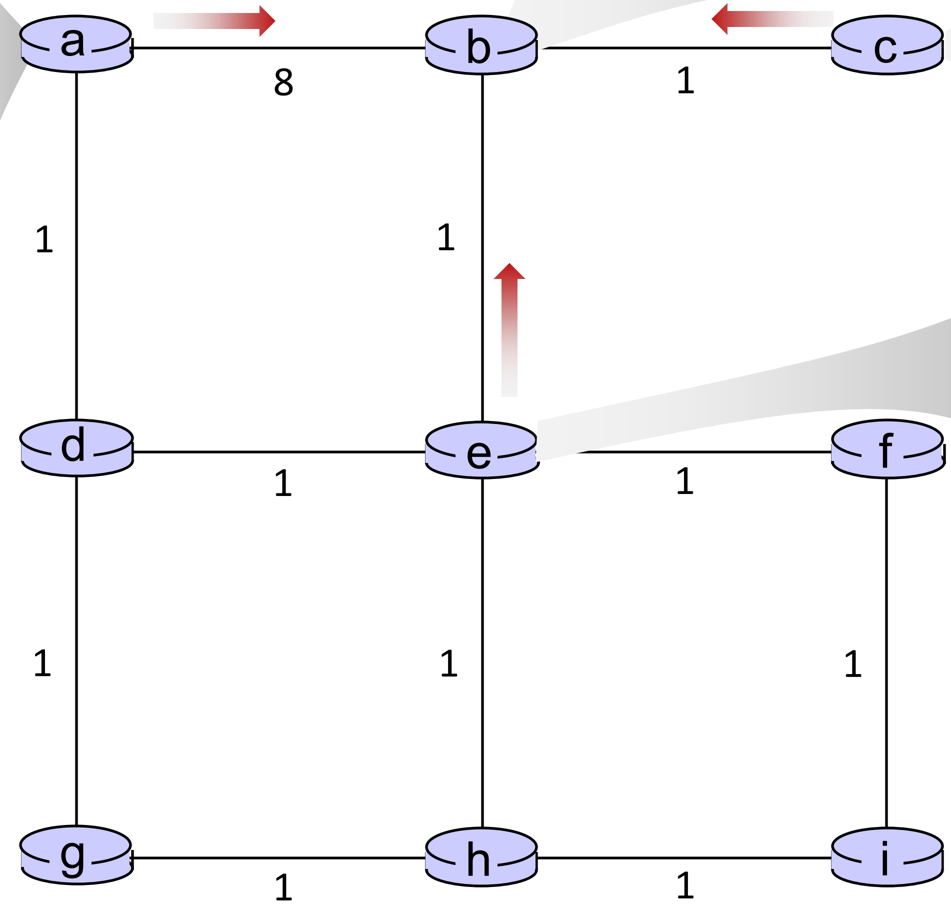
$D_b(a) = 8$	$D_b(f) = \infty$
$D_b(c) = 1$	$D_b(g) = \infty$
$D_b(d) = \infty$	$D_b(h) = \infty$
$D_b(e) = 1$	$D_b(i) = \infty$

DV in c:

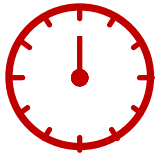
$D_c(a) = \infty$
$D_c(b) = 1$
$D_c(c) = 0$
$D_c(d) = \infty$
$D_c(e) = \infty$
$D_c(f) = \infty$
$D_c(g) = \infty$
$D_c(h) = \infty$
$D_c(i) = \infty$

DV in e:

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



Distance vector example:

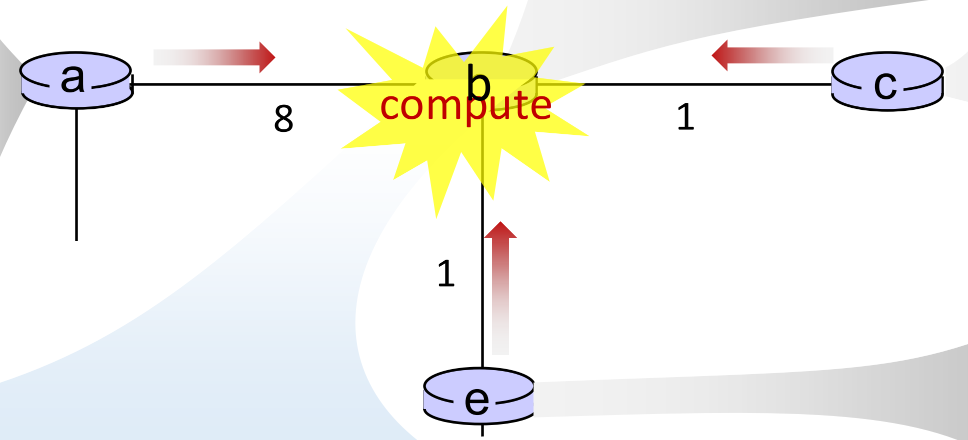


t=1

- 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:

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

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

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

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

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

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

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

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

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

DV in b:

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

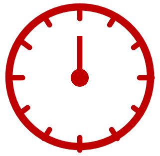
Distance vector example: iteration

Now, let’s look at the computation at node c at t = 1

Remember, c’s neighbors have sent c their DV record version at t = 0

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

Distance vector example: (



t=1

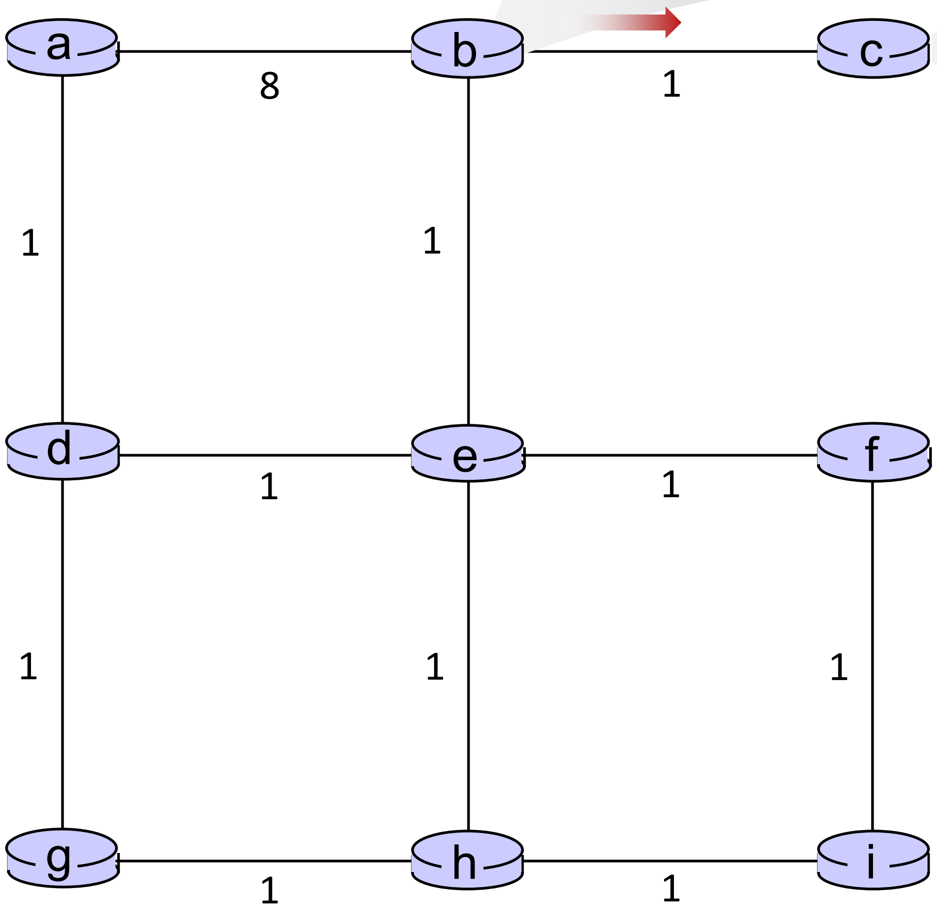
- c receives DVs from b

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:

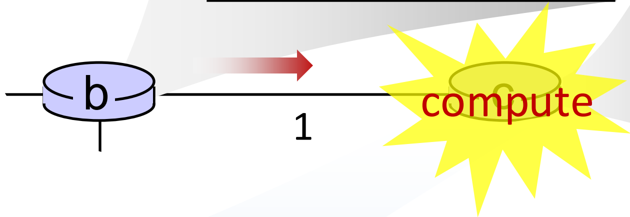
$D_c(a) = \infty$
$D_c(b) = 1$
$D_c(c) = 0$
$D_c(d) = \infty$
$D_c(e) = \infty$
$D_c(f) = \infty$
$D_c(g) = \infty$
$D_c(h) = \infty$
$D_c(i) = \infty$



Distance vector example:

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:

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

t=1

- c receives DVs from b computes:

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

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

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

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

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

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

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

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

DV in c:

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

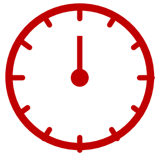
Distance vector example: iteration

Now, let's look at the computation at node e at t = 1

Remember, e's neighbors have sent e their DV record version at t = 0

* 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 b:

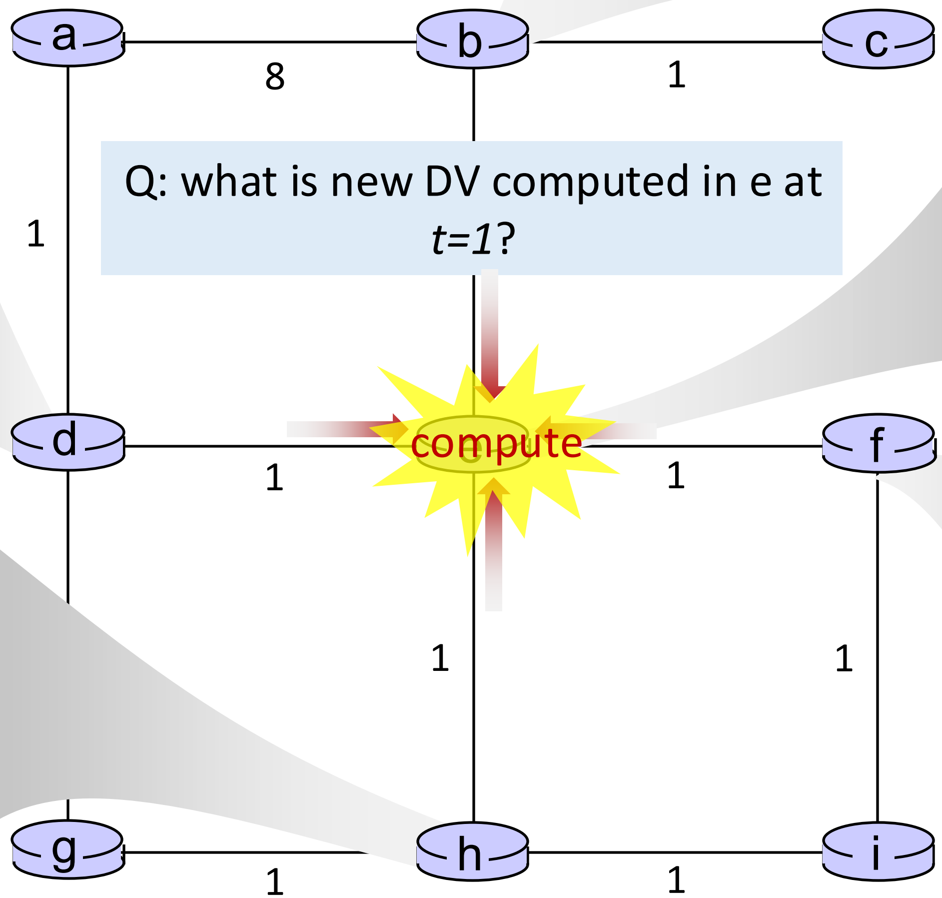
$D_b(a) = 8$ $D_b(f) = \infty$
$D_b(c) = 1$ $D_b(g) = \infty$
$D_b(d) = \infty$ $D_b(h) = \infty$
$D_b(e) = 1$ $D_b(i) = \infty$

DV in 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 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:

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

DV in f:

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

Distance vector: state information diffusion

Iterative communication, computation steps diffuses information through network:

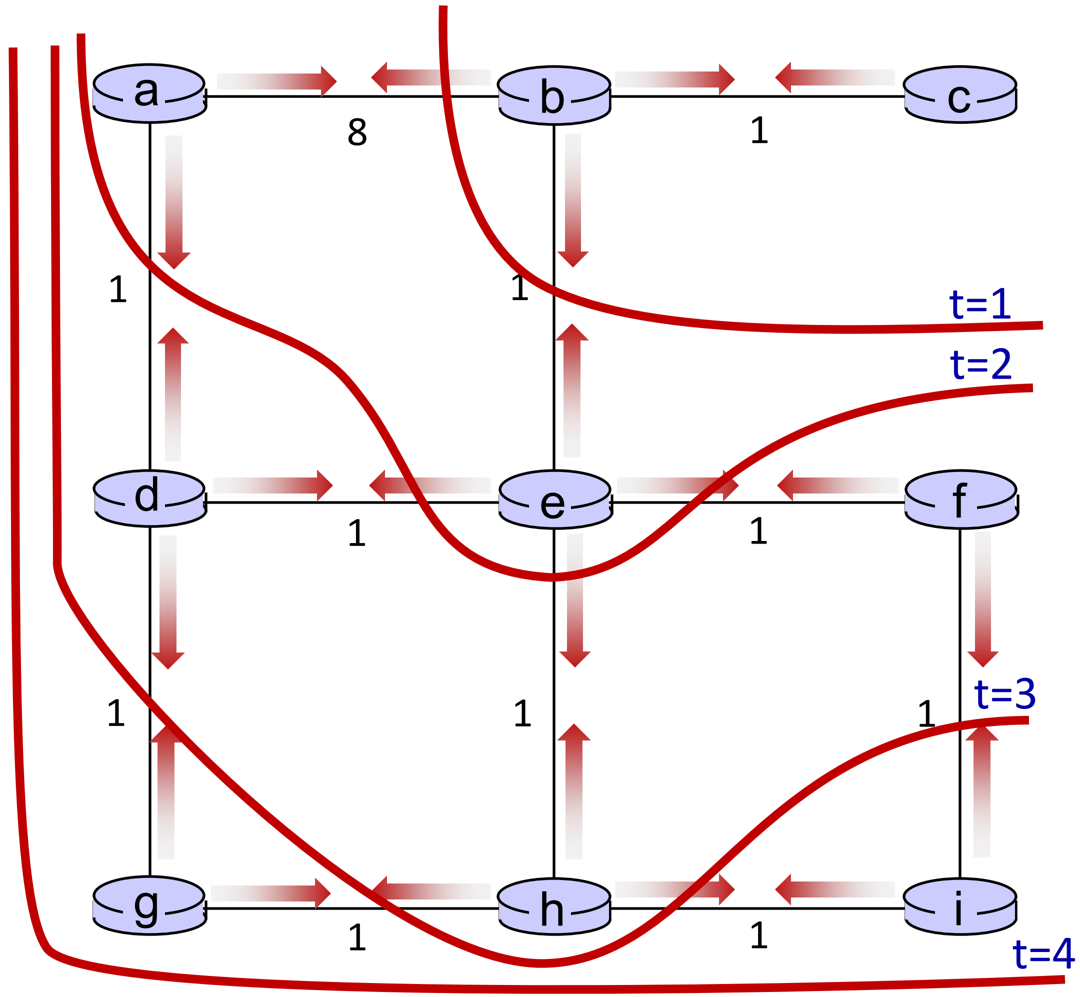
t=0 c's state at t=0 is at c only

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 d, f, h

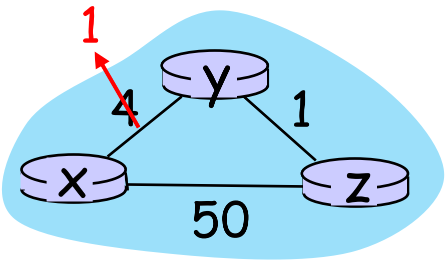
t=4 c's state at t=0 may influence distance vector computations up to **4** hops away, i.e., at g, i



Distance vector is asynchronous

- The example we discussed was simplified…
- We assumed there is a synchronized clock between all routers
  - Syncing the message transfers and computation.
- In reality, the routers are not all synchronized with each other

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

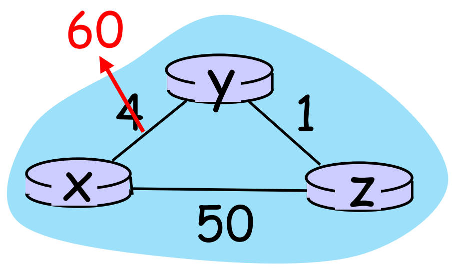
"good news travels fast"

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

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

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

Distance vector: link cost changes

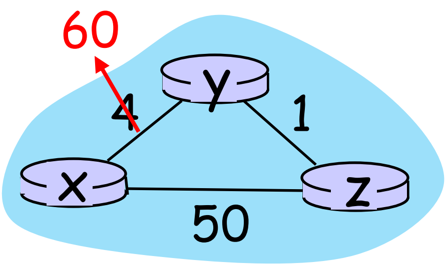


link cost changes:

- 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*.

    …

Distance vector : count-to-infinity problem



link cost changes:

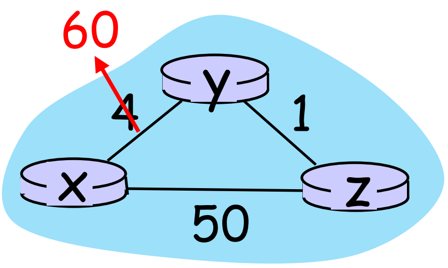
- node detects local link cost change
- "bad news travels slow" – count-to-infinity problem
- In this specific example, the problem happens because:
  - originally z's shortest path to x is through y.
  - But, y doesn't know that! It only knows z has a path of length 5 to x.

Distance vector : count-to-infinity problem

link cost changes:

- node detects local link cost change
- "bad news travels slow" – count-to-infinity problem
- This problem does not only happen between two neighboring nodes
  - See textbook for a solution for the two-node case
- It can happen with loops involving three or more nodes.

- *Distributed algorithms are tricky!*



What you need to know about distance vector routing algorithms

- How they work, i.e.,
  - How routers disseminate information
  - How each router builds its table of distance to different destinations
- E.g., given DV tables and messages from neighboring routers, you should be able to continue executing the algorithm and update DV tables for subsequent timesteps.
- The count-to-infinity problem
  - What it is
  - Why it happens
  - Be able to demonstrate it with an example.

Possible ways to practice more with DV

- Continue the example in the slide for t = 2.
  - Be careful to keep track of which node has received which messages at which time and what is DV looks like.
- Variation: Add an end-host node to the topology and re-do the first two timesteps for a few routers.
- What does the forwarding table look like at each stage?

Comparison of LS and DV algorithms

Messages

LS: Each router's "Advertisement", i.e., link state, will have to be propagated to all the other routers.

DV: Several messages exchanged between neighbors until we converge to the least cost paths; convergence time varies

speed of convergence:

If you change the costs, how long until routes are stable again?

LS :

- Converges when
  - Messages about the change propagate
  - Dijkstra's algorithm for least-cost path computation has to run

DV:

- may have routing loops
- count-to-infinity problem

Comparison of LS and DV algorithms

robustness: what happens if router malfunctions, or is compromised?

LS:

- router can advertise incorrect *link* cost
- each router computes only its *own* table based on the topology

DV:

- DV router can advertise incorrect *path* cost ("I have a *really* low-cost path to everywhere"): *black-holing*
- each router's DV is based on DV of other routers
  - No full picture of the network
  - Harder to detect such problems locally
  - Errors propagate (easier) through the network.

What you need to know about routing algorithms so far

- Link State (LS) algorithms and how they work
- Distance Vector (DV) algorithms and how they work
- How LS and DV are different from each other.