Distance vector routing

CE 352, Computer Networks

Salem Al-Agtash

Lecture 16

Slides are adapted from Computer Networking: A Top Down Approach, 7th Edition © J.F Kurose and K.W. Ross

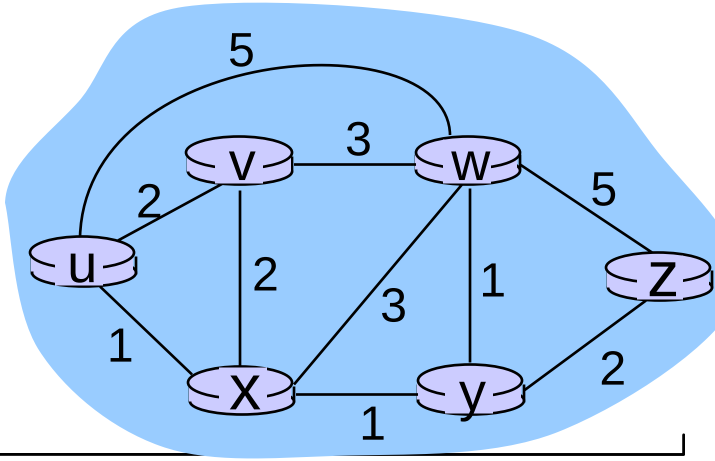
Recap (Routing protocols)

Routing protocols implement the core function of a network

- determine “good” paths (equivalently, routes), from sending hosts to receiving host, through network of routers

Network modeled as a graph

- 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) }
- Edges have cost (distance, loss, capacity, congestion)



Recap(Routing protocols)

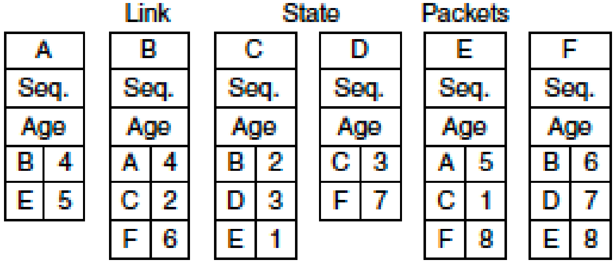
Global:

- "link state"
- Each router floods its LSP, learns (constructs) network topology
- Each router runs Dijkstra to compute the shortest path

Local:

- "distance vector"
- Each router knows its neighbors and costs and constructs DV
- Routers exchange DV with neighbors
- Each router uses Bellman-ford dynamic programming
- No router knows the network topology

Recap(Summary: Link-st



Link State Packets

A	B	C	D	E	F
Seq.	Seq.	Seq.	Seq.	Seq.	Seq.
Age	Age	Age	Age	Age	Age
B 4	A 4	B 2	C 3	A 5	B 6
E 5	C 2	D 3	F 7	C 1	D 7
	F 6	E 1		F 8	E 8

- Each router keeps track of its adjacent links (cost and operational status)

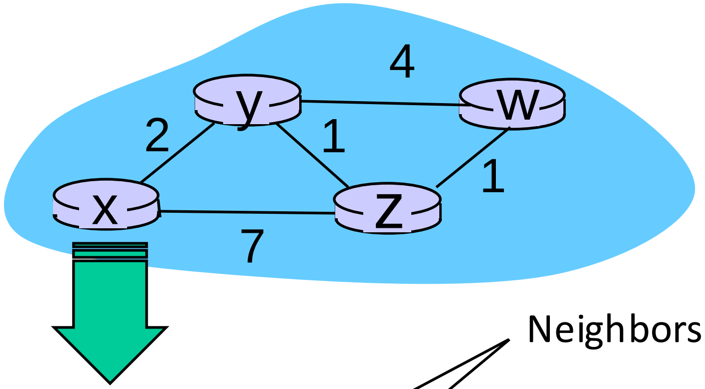
- Each router broadcasts the link state to give every router a complete view of the graph
  - Periodically every 30 minutes (e.g.), or topology change (Link/node failure or recovery)
  - Sequence number (32 bits) is used to label LS packets and keep flooding, when a new one arrives the older is discarded
  - Age (Time-to-Live) is used and when becomes 0, packet is discarded

- Each router runs Dijkstra's algorithm to compute the shortest path and construct the forwarding table

- Widely used protocol: OSPF

Distance vector routing

- Each node knows the links to its neighbors
  - Does not flood this information to the whole network
- Each node has provisional "shortest path" to every other node
  - Vector of distance to every other node
- Nodes exchange this distance vector with neighbors
- Each node iteratively selects the shortest path and updates its distance vector

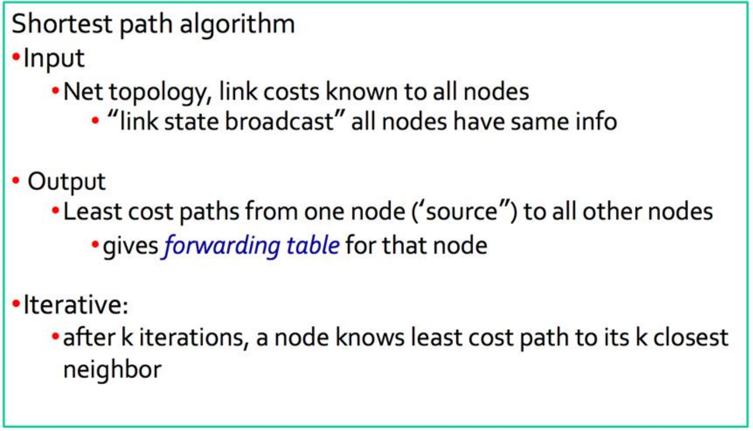


Via / TO	y	z
y	2	8
z	3	7
w	6	8

Neighbors

Best cost from x to w via z

Distance vector



Shortest path algorithm

- Input
  - Net topology, link costs known to all nodes
    - “link state broadcast” all nodes have same info
- Output
  - Least cost paths from one node (‘source”) to all other nodes
    - gives *forwarding table* for that node
- Iterative:
  - after k iterations, a node knows least cost path to its k closest neighbor

Bellman-Ford Dynamic programming

- Input
  - Link costs to neighbors – not the network topology
- Output
  - Next node to each destination and cost (not complete path)
    - Neighbors send vector distance on how far to each destination
- Iterative:
  - Compute cost to neighbors and neighbor’s cost to destination
  - Select minimum path through neighbors and advertise costs to neighbors

Notation

- $c(x,v)$: link cost from node x to its neighbor v (for all neighbors)
- $D_v(y)$: least-cost path from neighbor node v to y
- $D_x(y)$: least-cost path from node x to y, is then which neighbors gives shortest path

$$D_x(y) = \underset{v}{\mathit{min}} \{c(x,v) + D_v(y) \}$$

cost from neighbor v to destination y

cost to neighbor v

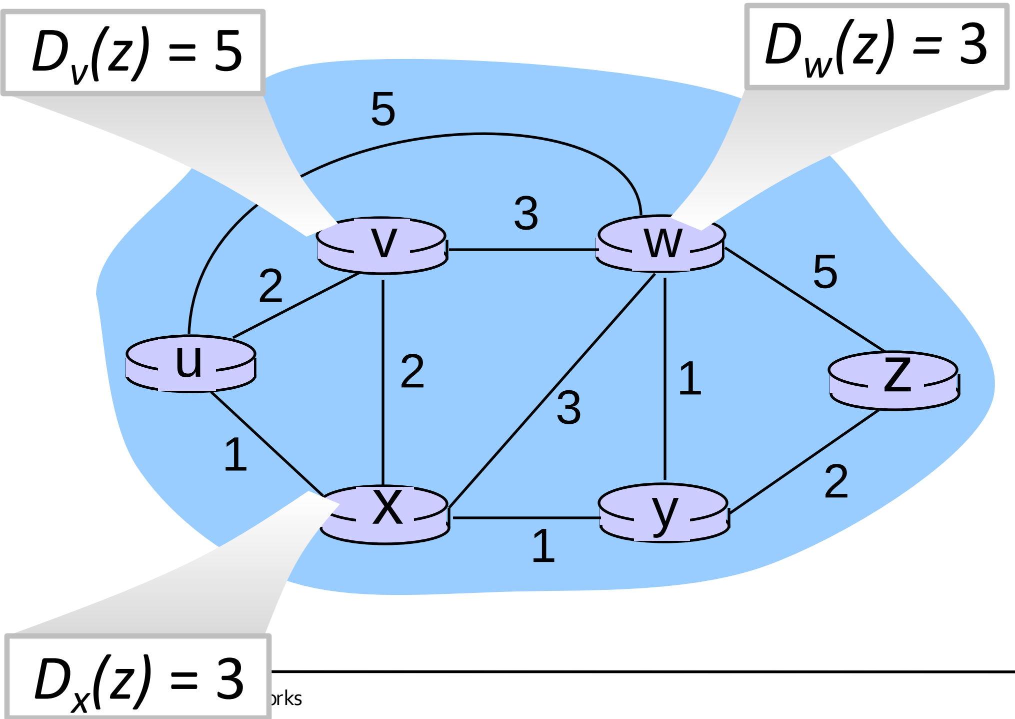
min taken over all neighbors v of x

Bellman-Ford example

Node u has 3 neighbors: V, X, and W

To get the best path to Z, gets costs to Z from neighbors

clearly, $D_v(z) = 5$, $D_x(z) = 3$, $D_w(z) = 3$



B-F equation says:

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

$$= \min \{2 + 5, 1 + 3, 5 + 3\} = \boxed{4}$$

Cost estimate

$D_x(y)$ = calculated least cost from x to y

- x maintains distance vector $\mathbf{D}_x = [D_x(y): y \in N]$

node x:

- knows cost to each neighbor v: $c(x,v)$
- maintains its neighbors' distance vectors. For each neighbor v, x maintains $\mathbf{D}_v = [D_v(y): y \in N]$

Key idea

from time-to-time, each node sends its own distance vector estimate to neighbors

when x receives new DV estimate from neighbor, it updates its own DV using B-F equation:

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

- under minor, natural conditions, the estimate $D_x(y)$ *converges to the actual least cost* D_x(y)

Bellman-Ford

iterative:

each local iteration caused by:

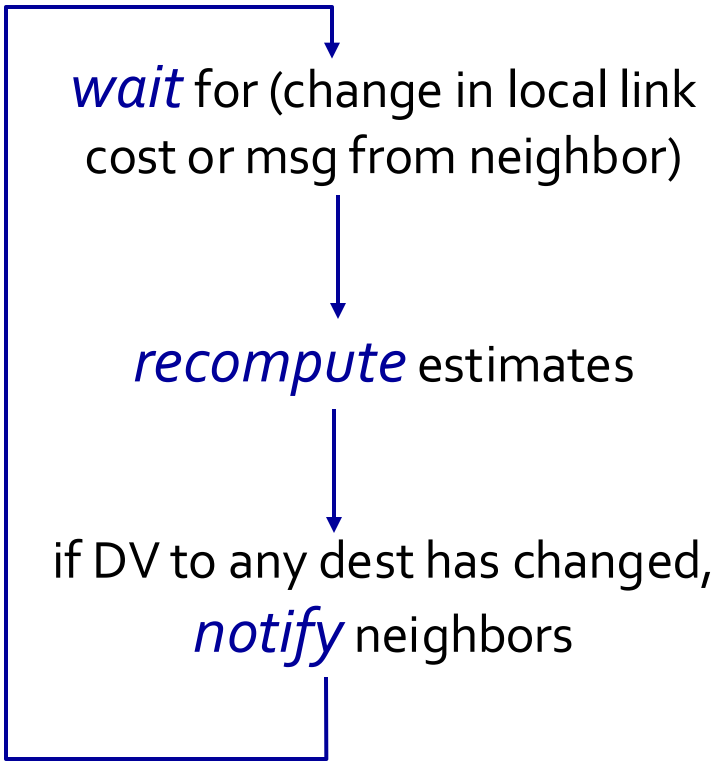
- local link cost change
- DV update message from neighbor

distributed:

each node notifies neighbors *only* when its DV changes

- neighbors then notify their neighbors if necessary

each node:



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

recompute estimates

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

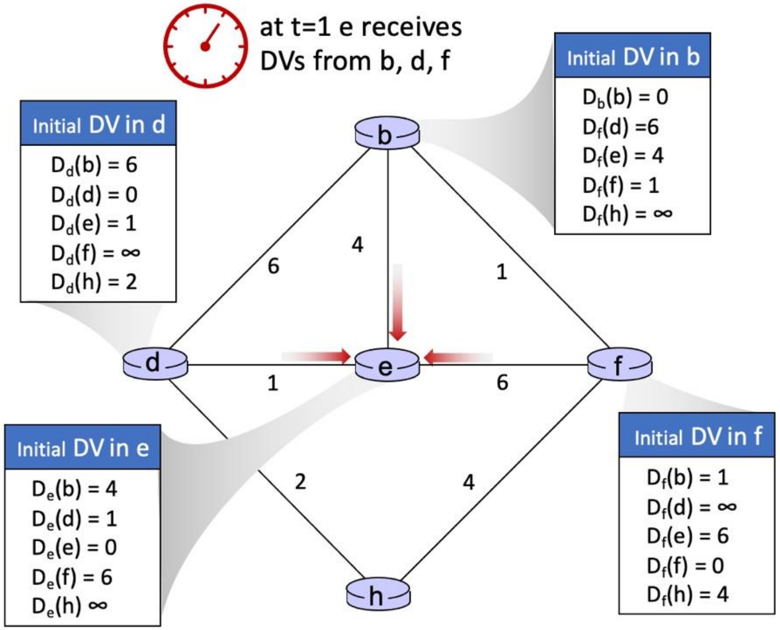
DV Algorithm

```
Initialization:
    for all destinations y in N:
        $D_x(y) = c(x, y)$   /* if y is not a neighbor then $c(x, y) = \infty$ */
    for each neighbor w
        $D_w(y)$ = ? for all destinations y in N
    for each neighbor w
        send distance vector $\mathbf{D}_x = [D_x(y): y \text{ in } N]$ to w

loop
    wait (until I see a link cost change to some neighbor w or
              until I receive a distance vector from some neighbor w)
    for each y in N:
        $D_x(y) = \min_v \{c(x, v) + D_v(y)\}$
    if $D_x(y)$ changed for any destination y
        send distance vector $\mathbf{D}_x = [D_x(y): y \text{ in } N]$ to all neighbors
forever
```

Example

- Consider the scenario shown to the right, where at t=1, node e receives distance vectors (DVs) from neighboring nodes b, d, and f. The (old) DV at e (before receiving the new DVs from its neighbors) is also shown, as well as the DVs being sent from b, d, and f. Compute the new distance vector components at e.
- $D_e(b)$?
- $D_e(d)$?
- $D_e(e)$?
- $D_e(f)$?
- $D_e(h)$?



at t=1 e receives DVs from b, d, f

Initial DV in d

$D_d(b) = 6$
$D_d(d) = 0$
$D_d(e) = 1$
$D_d(f) = \infty$
$D_d(h) = 2$

Initial DV in b

$D_b(b) = 0$
$D_f(d) =6$
$D_f(e) = 4$
$D_f(f) = 1$
$D_f(h) = \infty$

Initial DV in e

$D_e(b) = 4$
$D_e(d) = 1$
$D_e(e) = 0$
$D_e(f) = 6$
$D_e(h) \infty$

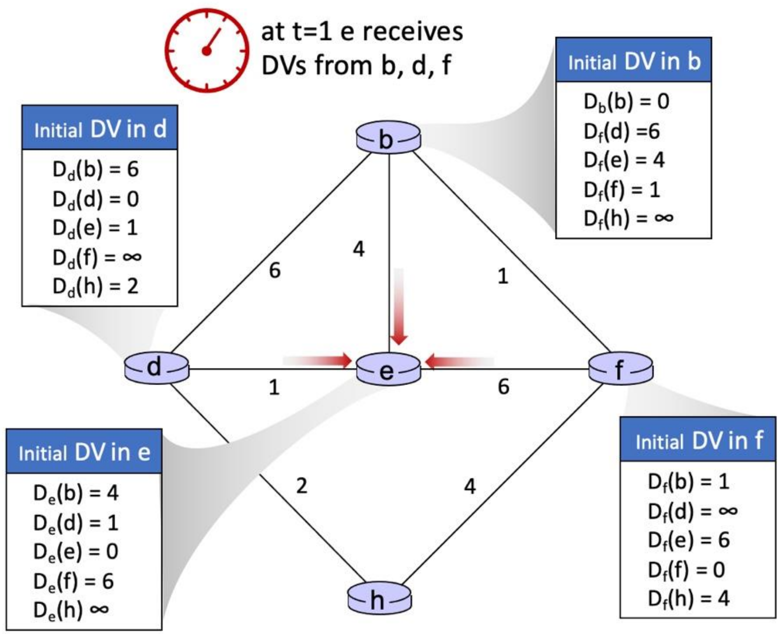
Initial DV in f

$D_f(b) = 1$
$D_f(d) = \infty$
$D_f(e) = 6$
$D_f(f) = 0$
$D_f(h) = 4$

Example

- Consider the scenario shown to the right, where at t=1, node e receives distance vectors (DVs) from neighboring nodes b, d, and f. The (old) DV at e (before receiving the new DVs from its neighbors) is also shown, as well as the DVs being sent from b, d, and f. Compute the new distance vector components at e.

$D_e(b)$	4
$D_e(d)$	1
$D_e(e)$	0
$D_e(f)$	5
$D_e(h)$	3



at t=1 e receives DVs from b, d, f

Initial DV in b

$D_b(b) = 0$
$D_f(d) =6$
$D_f(e) = 4$
$D_f(f) = 1$
$D_f(h) = \infty$

Initial DV in d

$D_d(b) = 6$
$D_d(d) = 0$
$D_d(e) = 1$
$D_d(f) = \infty$
$D_d(h) = 2$

Initial DV in e

$D_e(b) = 4$
$D_e(d) = 1$
$D_e(e) = 0$
$D_e(f) = 6$
$D_e(h)\ \infty$

Initial DV in f

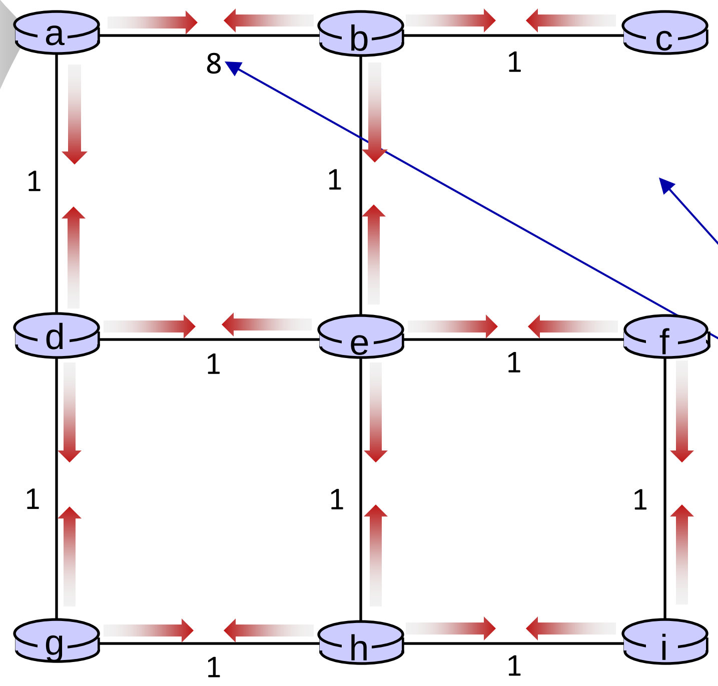
$D_f(b) = 1$
$D_f(d) = \infty$
$D_f(e) = 6$
$D_f(f) = 0$
$D_f(h) = 4$

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:

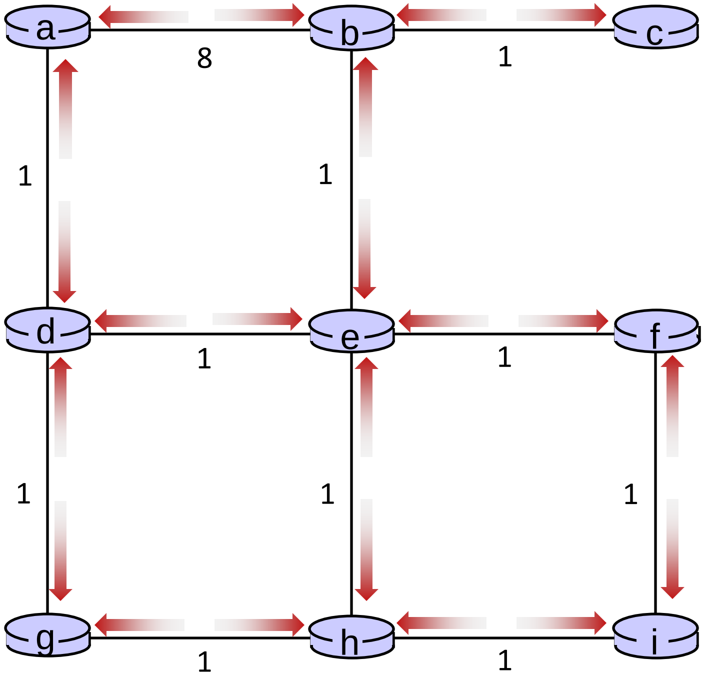
- missing link
- larger cost

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

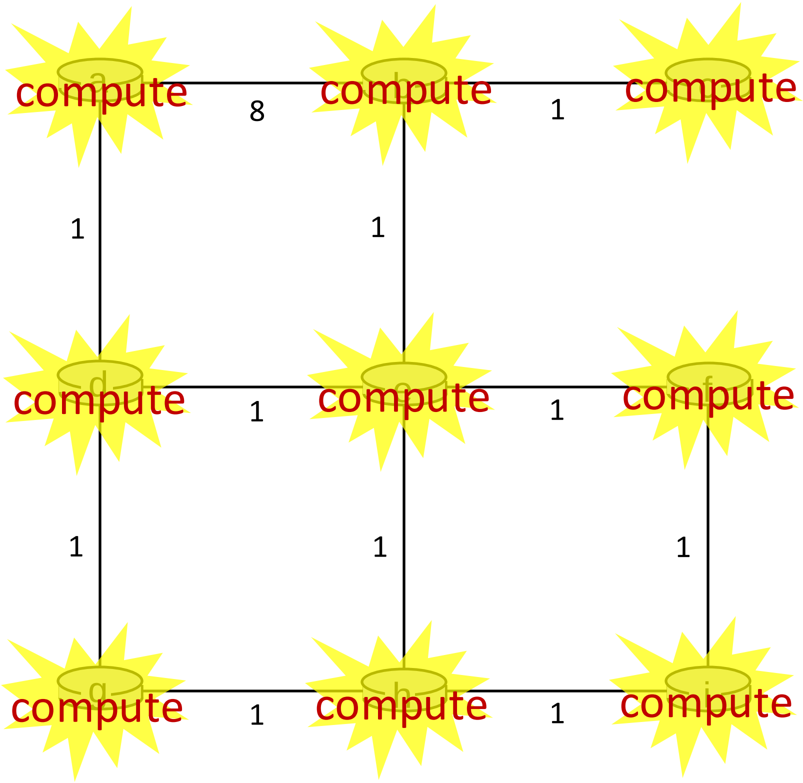


Distance vector example: iteration

t=1

All nodes:

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

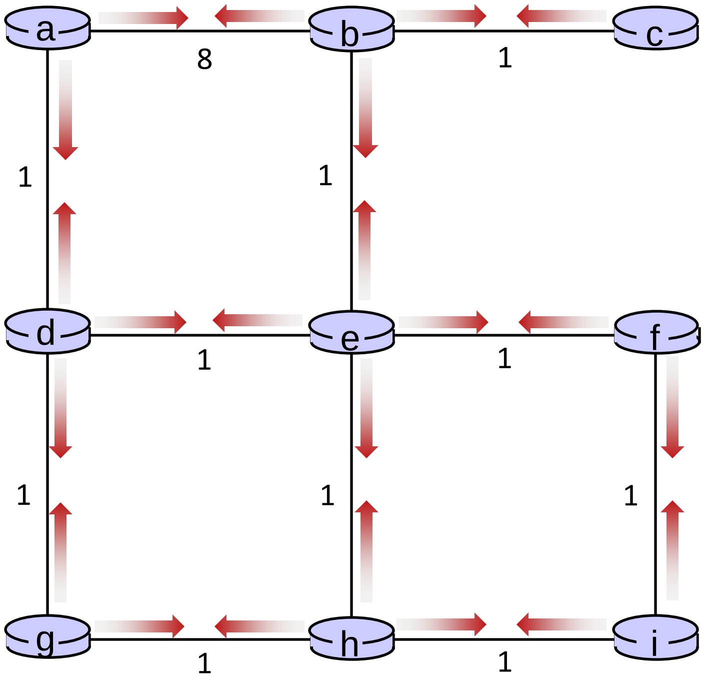


Distance vector example: iteration

t=1

All nodes:

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

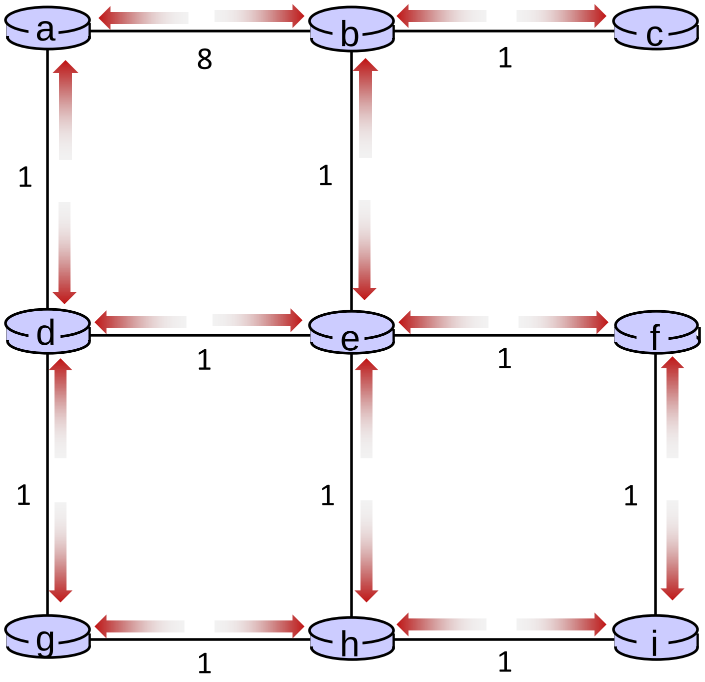


Distance vector example: iteration

t=2

All nodes:

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

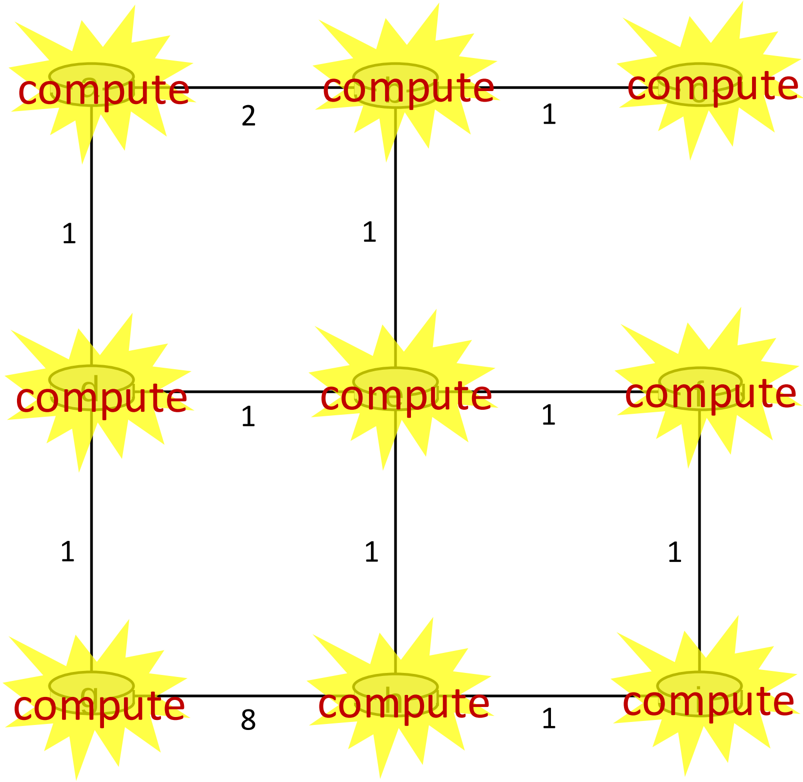


Distance vector example: iteration

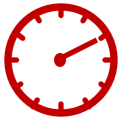
t=2

All nodes:

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



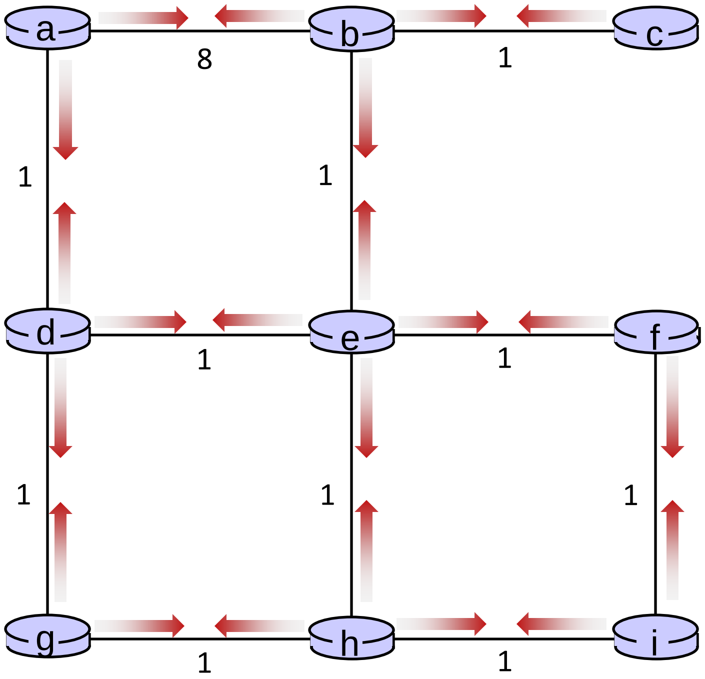
Distance vector example: iteration



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

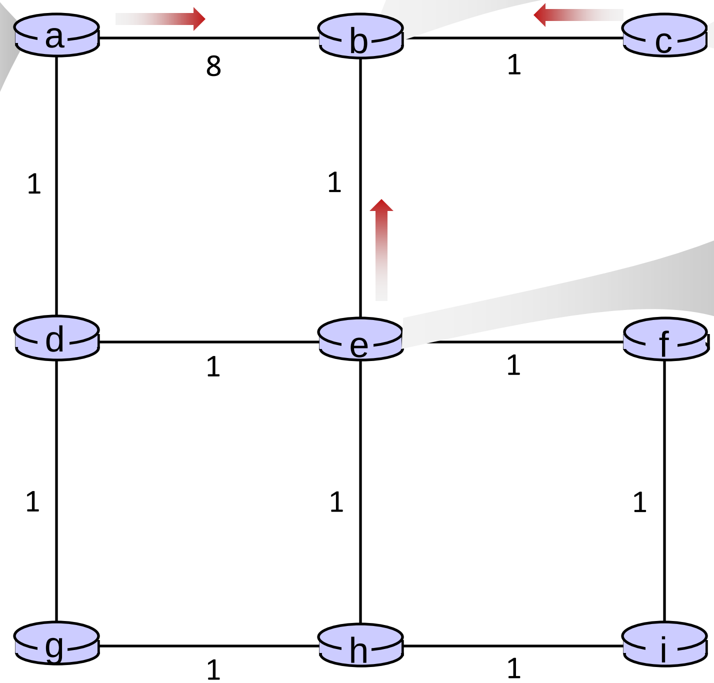
…. and so on

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

Distance vector exam . . ion

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:

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

DV in c:

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

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 exan . . ion

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:

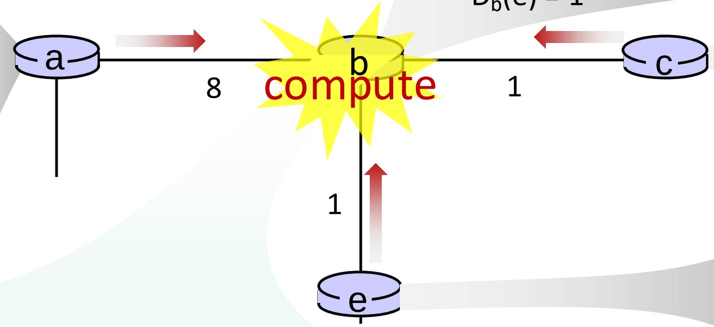
$D_c(a) = \infty$
$D_c(b) = 1$
$D_c(c) = 0$
$D_c(d) = \infty$
$D_c(e) = \infty$
$D_c(f) = \infty$

DV in a:

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

t=1

- b receives DVs from a,



DV in e:

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

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

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

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

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

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

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

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

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

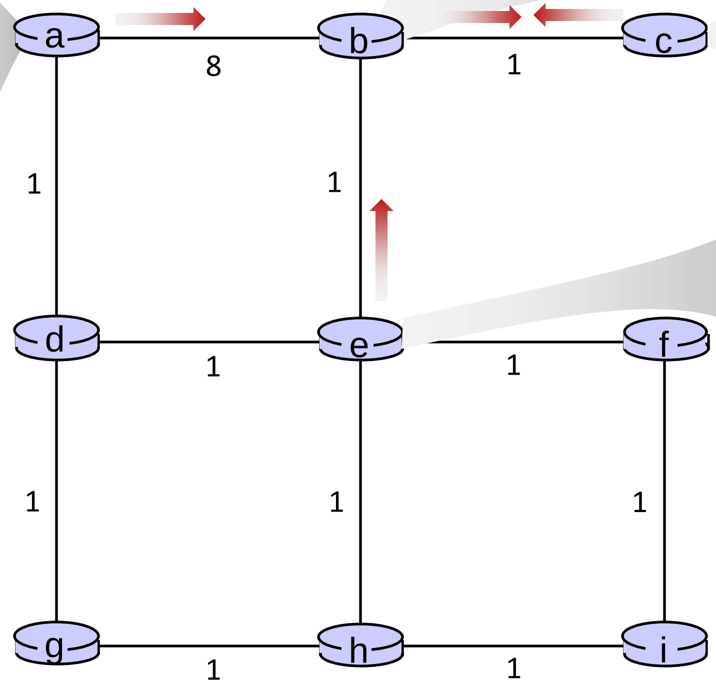
DV in b:

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

Distance vector exan . . ion

t=1

- c receives DVs from b



DV in a:

$D_a(a)$=0
$D_a(b)$ = 8
$D_a(c)$ = ∞
$D_a(d)$ = 1
$D_a(e)$ = ∞
$D_a(f)$ = ∞
$D_a(g)$ = ∞
$D_a(h)$ = ∞
$D_a(i)$ = ∞

DV in b:

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

DV in c:

$D_c(a)$ = ∞
$D_c(b)$ = 1
$D_c(c)$ = 0
$D_c(d)$ = ∞
$D_c(e)$ = ∞
$D_c(f)$ = ∞

DV in e:

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

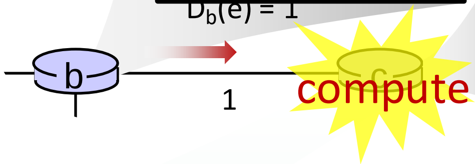
Distance vector exan . . ion

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:

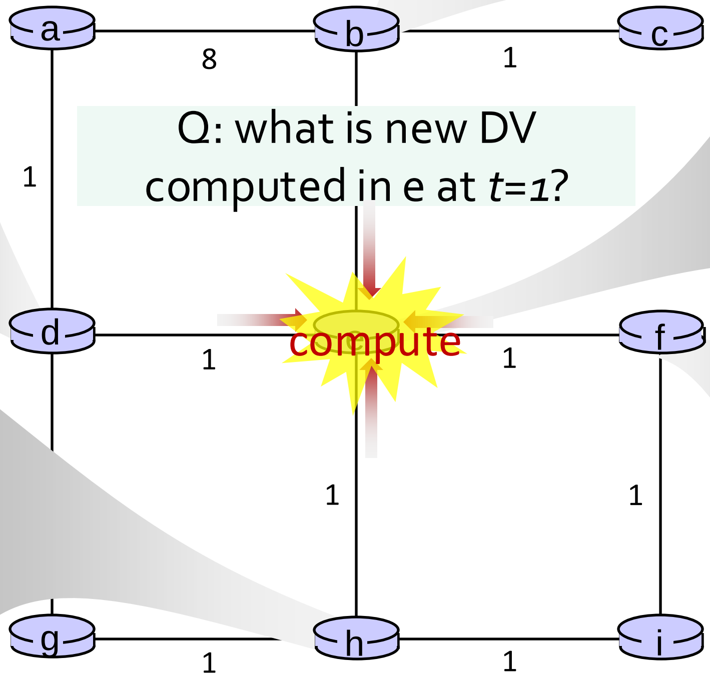
$D_c(a) = 9$
$D_c(b) = 1$
$D_c(c) = 0$
$D_c(d) = 2$
$D_c(e) = \infty$
$D_c(f) = \infty$
$D_c(g) = \infty$
$D_c(h) = \infty$
$D_c(i) = \infty$

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

Distance vector exam . . utation

t=1

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



Q: what is new DV computed in e at *t=1*?

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 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 e:

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

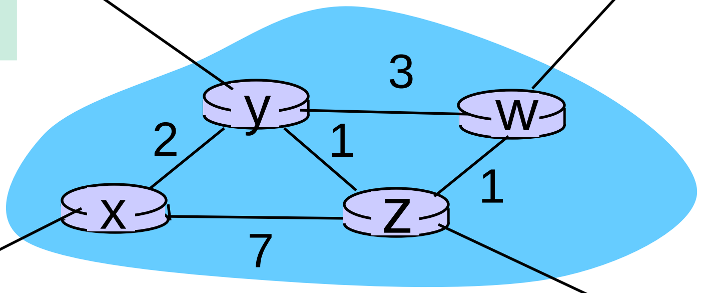
DV in f:

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

Initialization

Via TO	x	z	w
x	2	∞	∞
z	∞	1	∞
w	∞	∞	3

Via TO	y	z
x	∞	∞
y	3	∞
z	∞	1



Via TO	y	z
y	2	∞
z	∞	7
w	∞	∞

Neighbors

cost
from x to z

Via TO	x	y	w
x	7	∞	∞
y	∞	1	∞
w	∞	∞	1

- c(x,y): link cost from node x to y;
- $D_x(y)$: cost of least-cost path from node x to y

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

cost from neighbor v to destination y

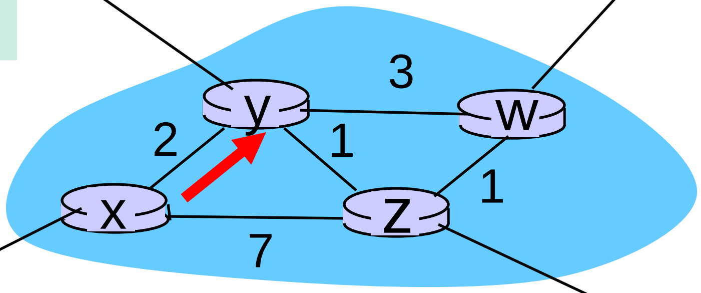
cost to neighbor v

min taken over all neighbors v of x

x sends update to y

Via TO	x	z	w
x	2	∞	∞
z	9	1	∞
w	∞	∞	3

Via TO	y	z
x	∞	∞
y	3	∞
z	∞	1



Via TO	y	z
y	2	∞
z	∞	7
w	∞	∞

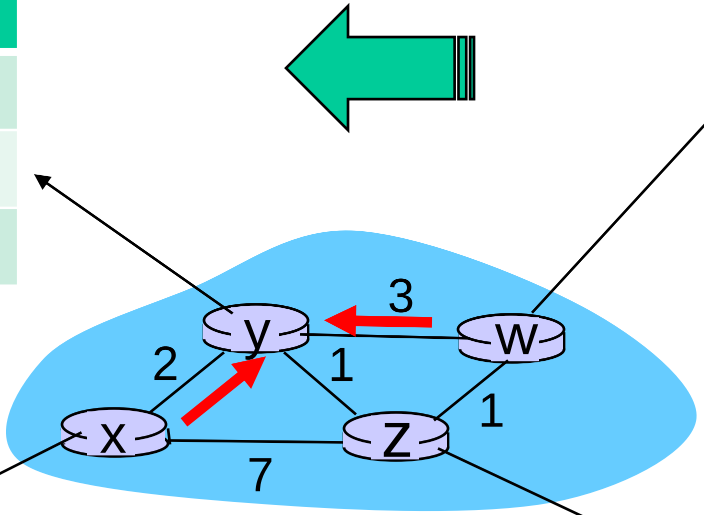
Neighbors

cost
from x to z

Via TO	x	y	w
x	7	∞	∞
y	∞	1	∞
w	∞	∞	1

w sends update to y

Via TO	x	z	w
x	2	∞	∞
z	9	1	4
w	∞	∞	3



Via TO	y	z
x	∞	∞
y	3	∞
z	∞	1

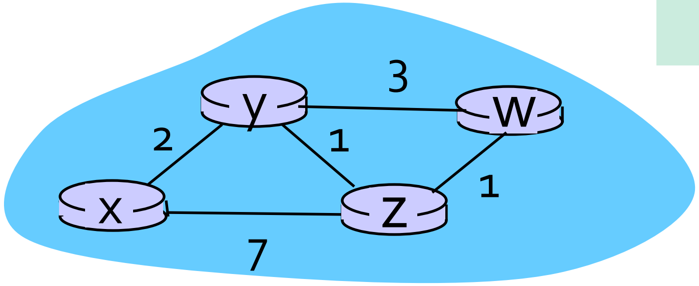
Via TO	y	z
y	2	∞
z	∞	7
w	∞	∞

Neighbors

cost
from x to z

Via TO	x	y	w
x	7	∞	∞
y	∞	1	∞
w	∞	∞	1

First exchanges



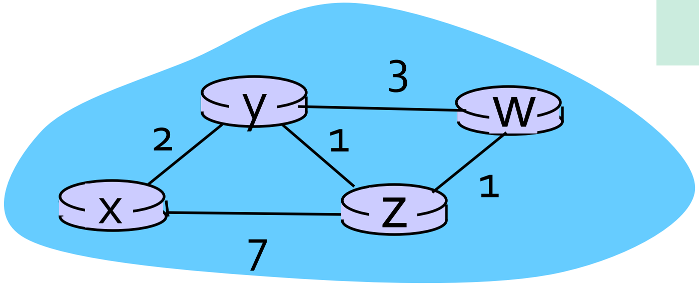
Via TO	x	z	w
x	2	8	∞
z	9	1	4
w	∞	2	3

Via TO	y	z
x	5	8
y	3	2
z	4	1

Via TO	y	z
y	2	8
z	3	7
w	5	8

Via TO	x	y	w
x	7	3	∞
y	9	1	4
w	∞	4	1

Two exchanges



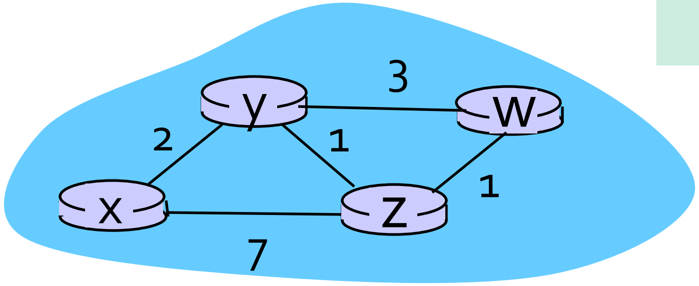
Via / TO	x	z	w
x	2	4	8
z	5	1	4
w	7	2	3

Via / TO	y	z
x	5	4
y	3	2
z	4	1

Via / TO	y	z
y	2	8
z	3	7
w	4	8

Via / TO	x	y	w
x	7	3	6
y	9	1	3
w	12	3	1

Three exchanges



Via TO	x	z	w
x	2	4	7
z	5	1	4
w	6	2	3

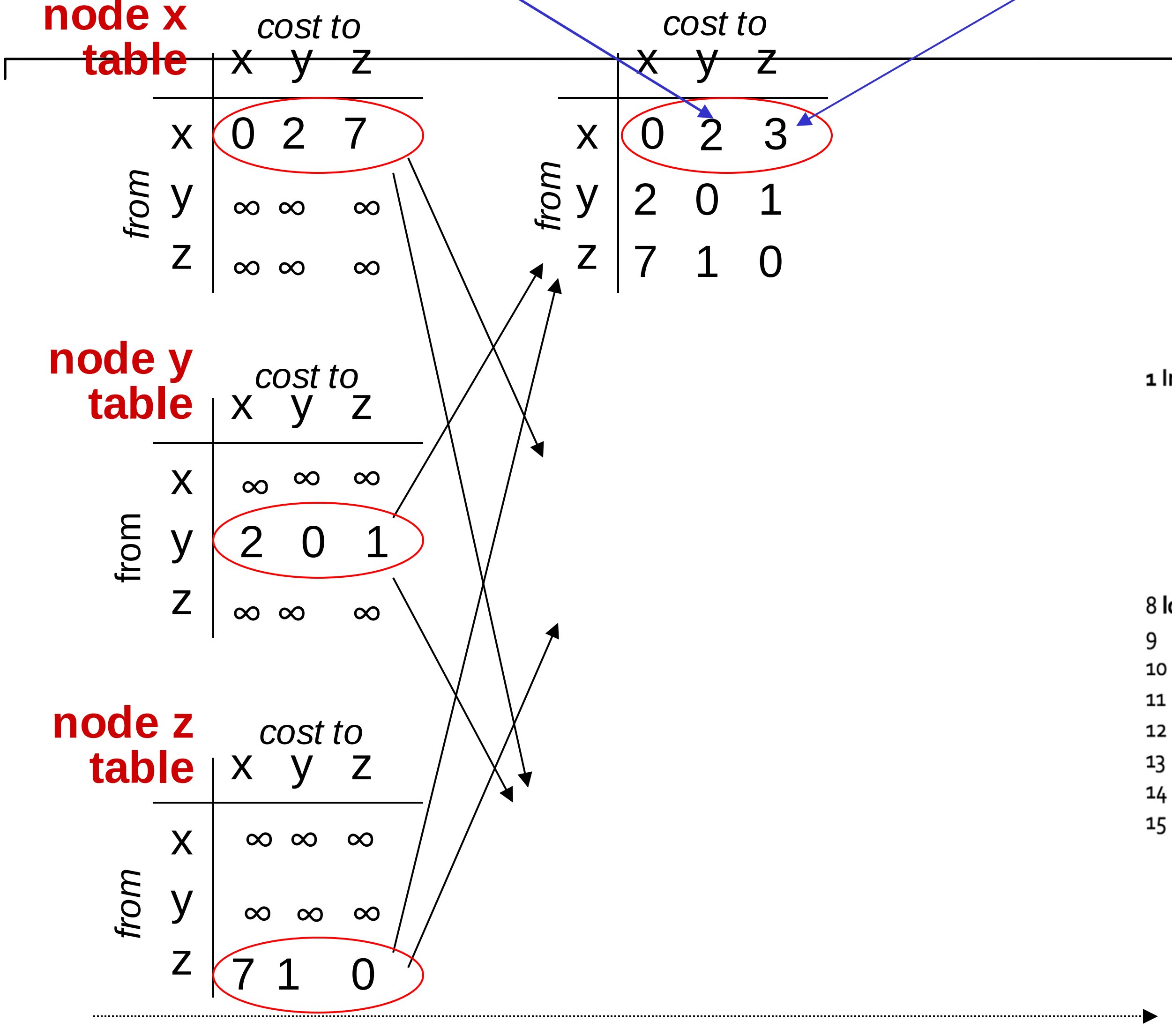
Via TO	y	z
x	5	4
y	3	2
z	4	1

Via TO	y	z
y	2	8
z	3	7
w	4	8

Via TO	x	y	w
x	7	3	5
y	9	1	3
w	11	3	1

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

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



node x table

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

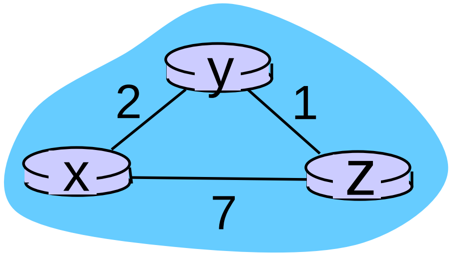
from \ *cost to*	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

node y table

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

node z table

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



```
Initialization:
    for all destinations y in N:
        $D_x(y)= c(x, y)$   /* if y is not a neighbor then $c(x, y)= \infty$ */
    for each neighbor w
        $D_w(y)$ = ? for all destinations y in N
    for each neighbor w
        send distance vector $\mathbf{D}_x = [D_x(y): y \in N]$ to w

loop
    wait (until I see a link cost change to some neighbor w or
            until I receive a distance vector from some neighbor w)
    for each y in N:
        $D_x(y) = \min_v \{c(x, v) + D_v(y)\}$
    if Dx(y) changed for any destination y
        send distance vector $\mathbf{D}_x = [D_x(y): y \in N]$ to all neighbors
forever
```

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

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

node x table

cost to

from	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

cost to

from	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

cost to

from	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

node y table

cost to

from	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

cost to

from	x	y	z
x	0	2	7
y	2	0	1
z	7	1	0

cost to

from	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

node z table

cost to

from	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0

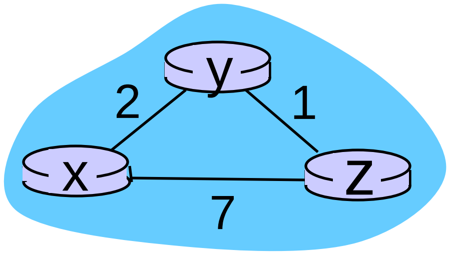
cost to

from	x	y	z
x	0	2	7
y	2	0	1
z	3	1	0

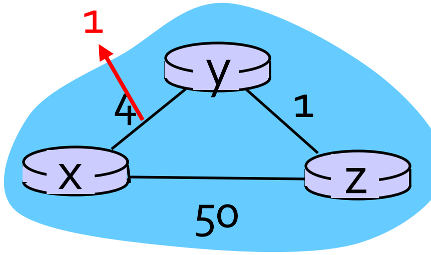
cost to

from	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

time



(x,y) link cost changes to 1



link cost changes:

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

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

cost to

from	x	y	z
x	0	4	5
y	4	0	1
z	5	1	0

cost to

from	x	y	z
x	0	4	5
y	1	0	1
z	5	1	0

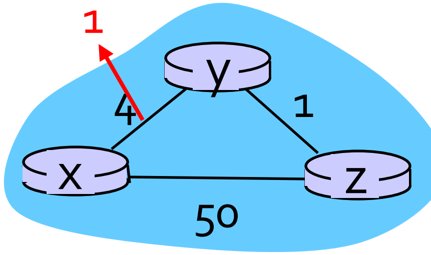
cost to

from	x	y	z
x	0	4	5
y	2	0	1
z	5	1	0

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

$= \min\{1+0 , 1+5\} = 1$

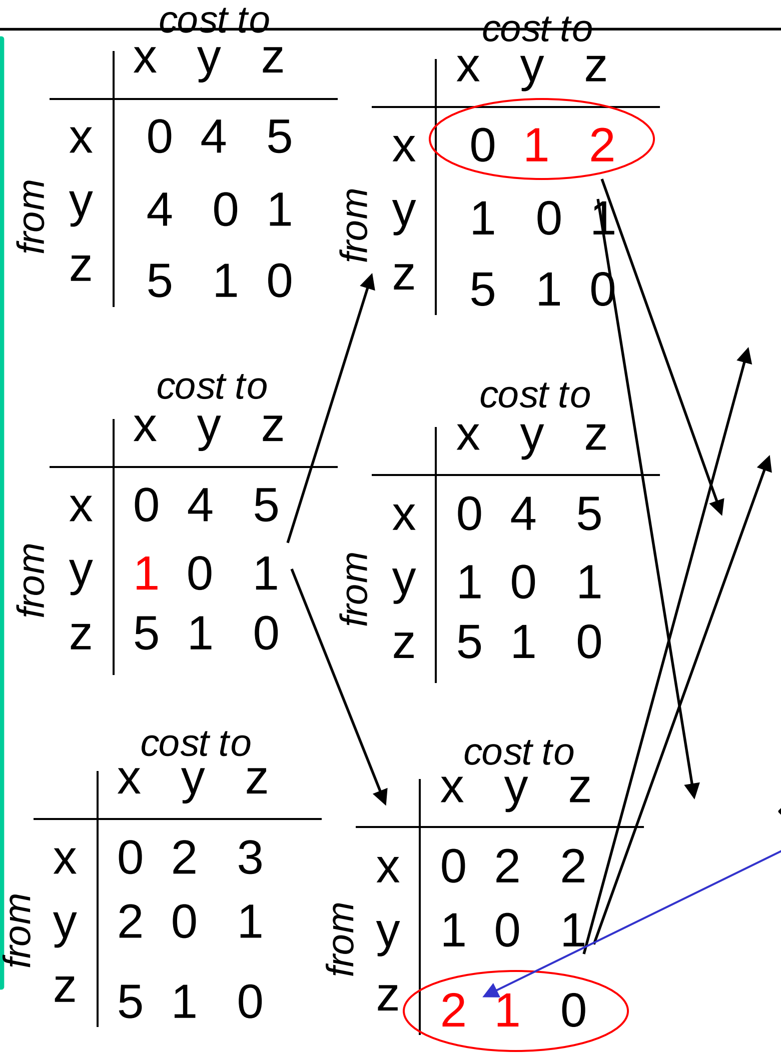
(x,y) link cost changes to 1



link cost changes:

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

***t*$_1$:** *z* receives update from *y*, updates its table, computes new least cost to *x* , sends its neighbors its DV.



from \ cost to	x	y	z
x	0	4	5
y	4	0	1
z	5	1	0

from \ cost to	x	y	z
x	0	4	5
y	1	0	1
z	5	1	0

from \ cost to	x	y	z
x	0	2	3
y	2	0	1
z	5	1	0

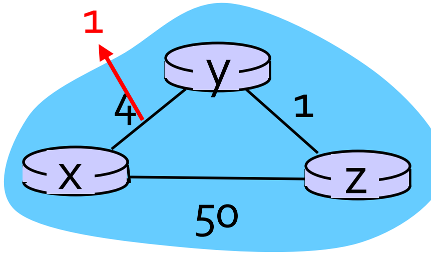
from \ cost to	x	y	z
x	0	1	2
y	1	0	1
z	5	1	0

from \ cost to	x	y	z
x	0	4	5
y	1	0	1
z	5	1	0

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

$D_z(x) = \min\{c(z,x) + D_z(z), c(x,y) + D_y(z)\}$
$= \min\{50+0 , 1+1\} = 2$

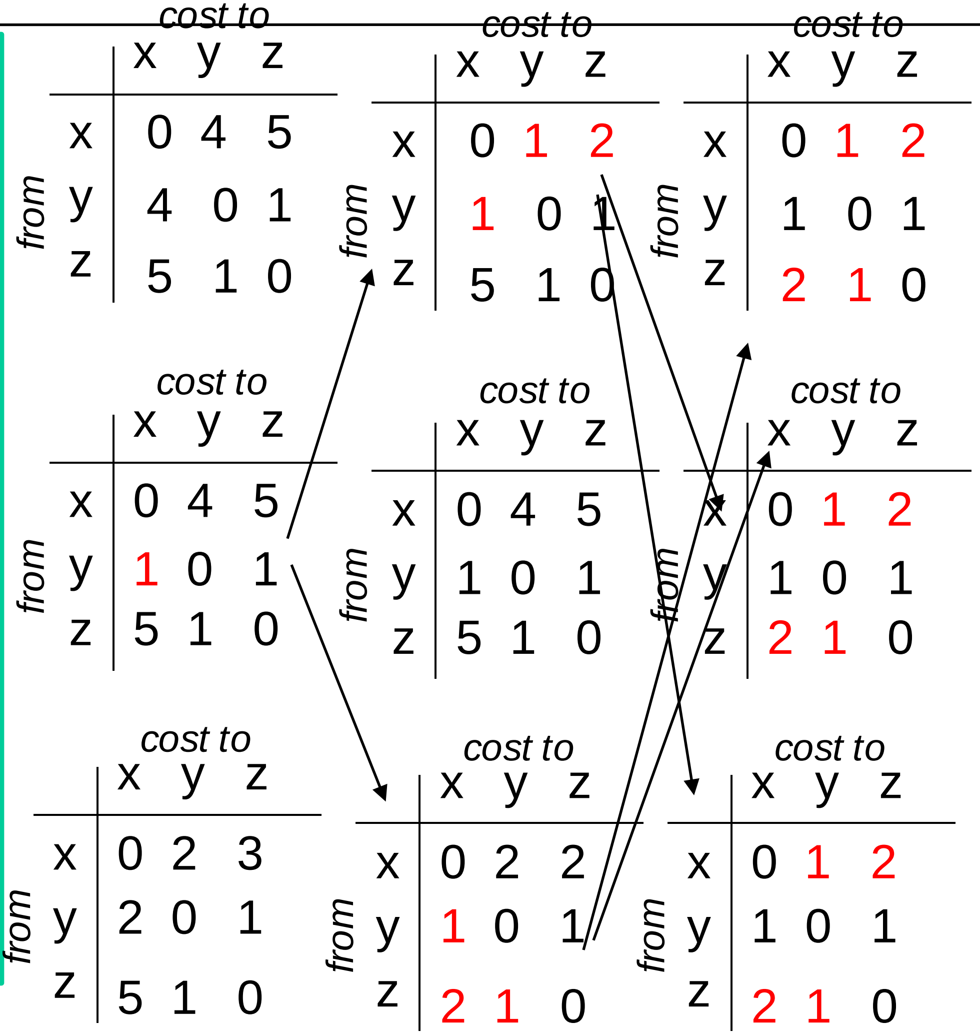
(x,y) link cost changes to 1



link cost changes:

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

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*.



cost to

from	x	y	z
x	0	4	5
y	4	0	1
z	5	1	0

cost to

from	x	y	z
x	0	1	2
y	1	0	1
z	5	1	0

cost to

from	x	y	z
x	0	1	2
y	1	0	1
z	2	1	0

cost to

from	x	y	z
x	0	4	5
y	1	0	1
z	5	1	0

cost to

from	x	y	z
x	0	4	5
y	1	0	1
z	5	1	0

cost to

from	x	y	z
x	0	1	2
y	1	0	1
z	2	1	0

cost to

from	x	y	z
x	0	2	3
y	2	0	1
z	5	1	0

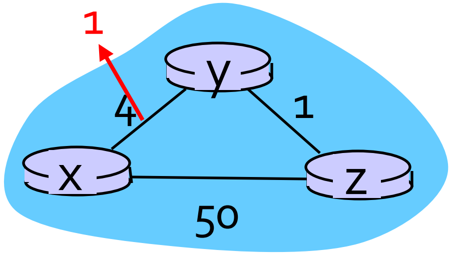
cost to

from	x	y	z
x	0	2	2
y	1	0	1
z	2	1	0

cost to

from	x	y	z
x	0	1	2
y	1	0	1
z	2	1	0

Link cost changes – good news



2 iterations →

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

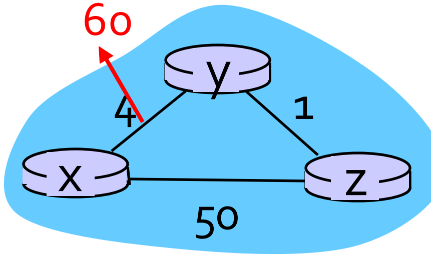
“good news travels fast”

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*.

(x,y) link cost changes to 60



link cost changes:

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

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

cost to

from	x	y	z
x	0	4	5
y	4	0	1
z	5	1	0

$D_y(x) = \min\{c(y,x) + D_x(x), c(y,z) + D_z(x)\}$
$= \min\{60+0 , 1+5\} = 6$

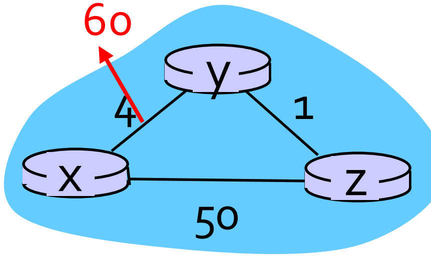
cost to

from	x	y	z
x	0	4	5
y	6	0	1
z	5	1	0

cost to

from	x	y	z
x	0	2	3
y	2	0	1
z	5	1	0

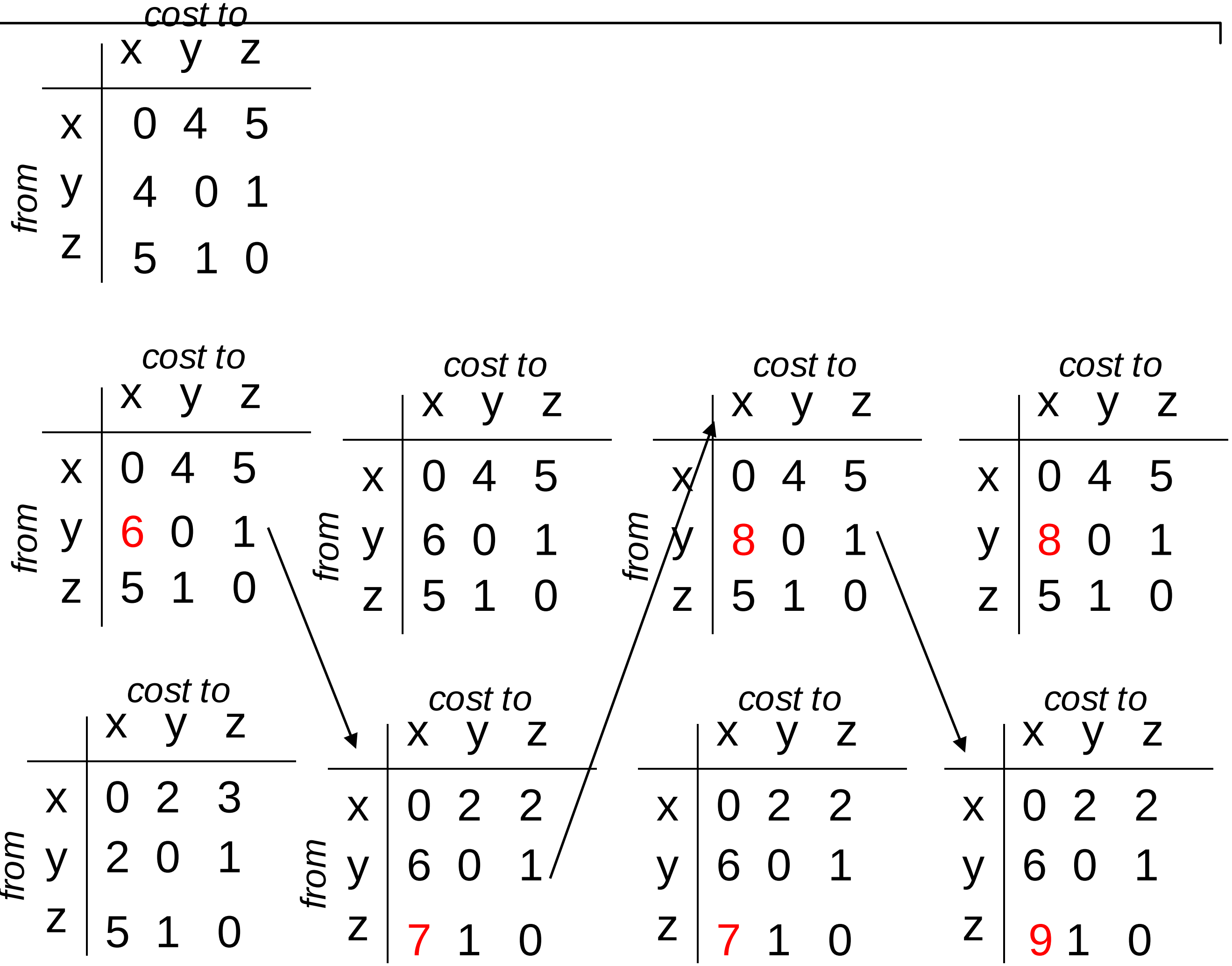
(x,y) link cost changes to 60



link cost changes:

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

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



cost to

from	x	y	z
x	0	4	5
y	4	0	1
z	5	1	0

cost to

from	x	y	z
x	0	4	5
y	6	0	1
z	5	1	0

cost to

from	x	y	z
x	0	4	5
y	6	0	1
z	5	1	0

cost to

from	x	y	z
x	0	4	5
y	8	0	1
z	5	1	0

cost to

from	x	y	z
x	0	4	5
y	8	0	1
z	5	1	0

cost to

from	x	y	z
x	0	2	3
y	2	0	1
z	5	1	0

cost to

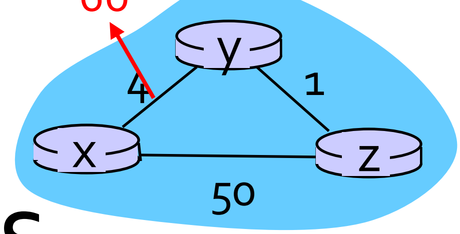
from	x	y	z
x	0	2	2
y	6	0	1
z	7	1	0

cost to

from	x	y	z
x	0	2	2
y	6	0	1
z	7	1	0

cost to

from	x	y	z
x	0	2	2
y	6	0	1
z	9	1	0



Link cost changes – bad news

link cost changes:

- node detects local link cost change
- *bad news travels slow* - "count to infinity" problem!
- 44 iterations before algorithm sees its path via y is greater than 50
- Y will then route to x via z
- What if cost changed from 4 to 10,000, and c(z,x) = 9,999, how many iterations? → count to infinity problem

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)
  - i.e.y poisons the reverse path from z to x by informing z that $D_y(x) = \infty$ (even path is 51)
- will this completely solve count to infinity problem?
- → still does not so solve the count-to-infinity problem if there are more loops

Summary: distance vector routing

- Each router sends to its neighbors least cost distance (distance vector) to all nodes
- Each router maintains a vector of distance to all nodes (forwarding table) and informs its neighbors if a change in cost occurs
- Each router runs Bellman - Ford to find the minimum distance and updates its distance vector
- Widely used protocol: RIP (routing information protocol) and is limited to small networks

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 varies

Speed of convergence

LS: O(n^2) algorithm requires O(ne) msgs

- may have oscillations

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

Desirable routing protocols

- Find least cost paths
- Converge quickly
- Avoid loops
- Are scalable as network size grows
- Acceptable metrics: number of hops, propagation delay, bandwidth, congestion, loss rate, etc.

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 approach to scalable routing

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

intra-AS routing

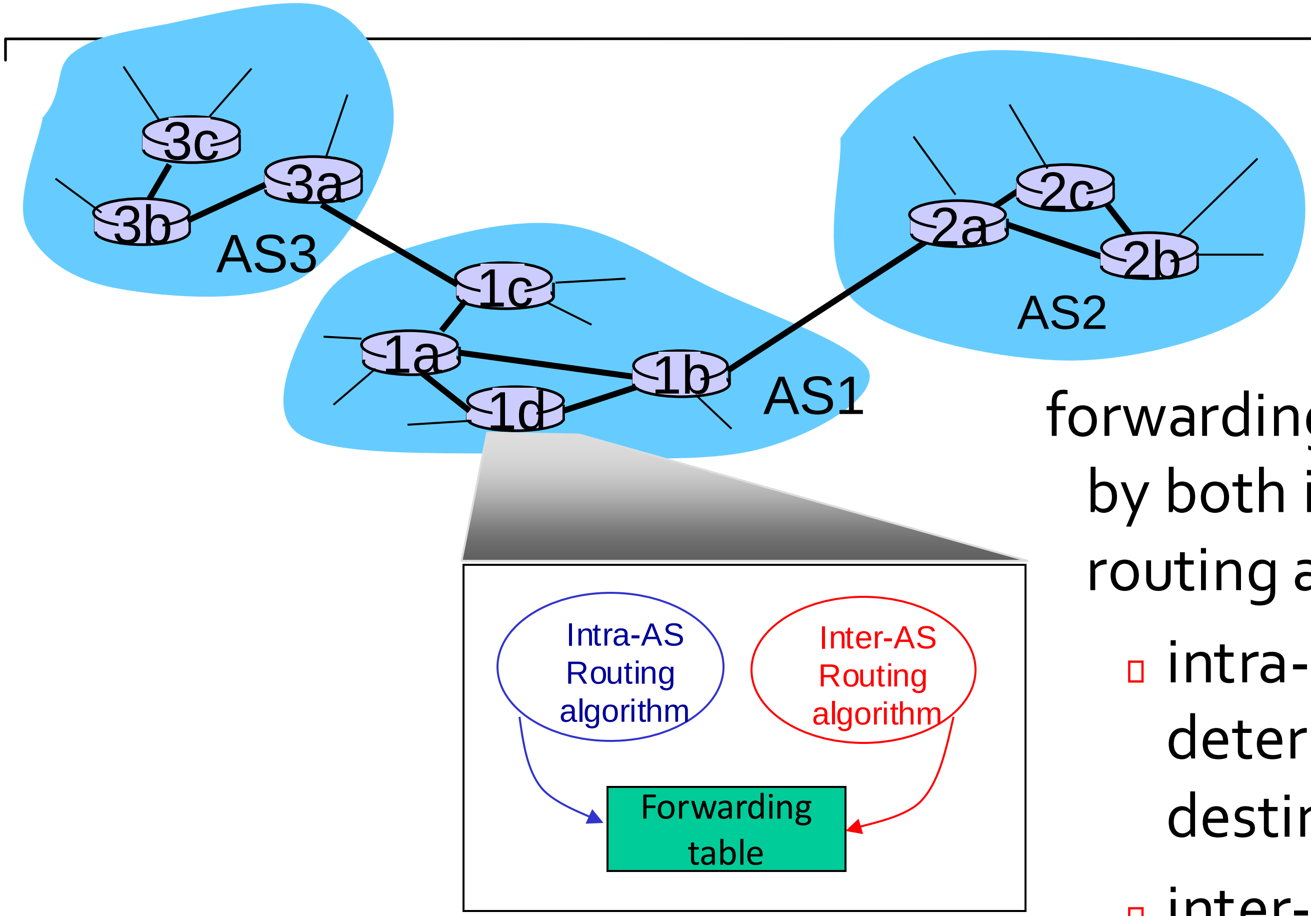
- routing among hosts, routers in same AS ("network")
- all routers in AS must run *same* intra-domain protocol
- routers in *different* AS can run *different* intra-domain routing protocol
- gateway router: at "edge" of its own AS, has link(s) to router(s) in other AS'es

inter-AS routing

routing among AS'es

gateways perform inter-domain routing (as well as intra-domain routing)

Interconnected ASes



forwarding table configured by both intra- and inter-AS routing algorithm

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

Inter-AS tasks

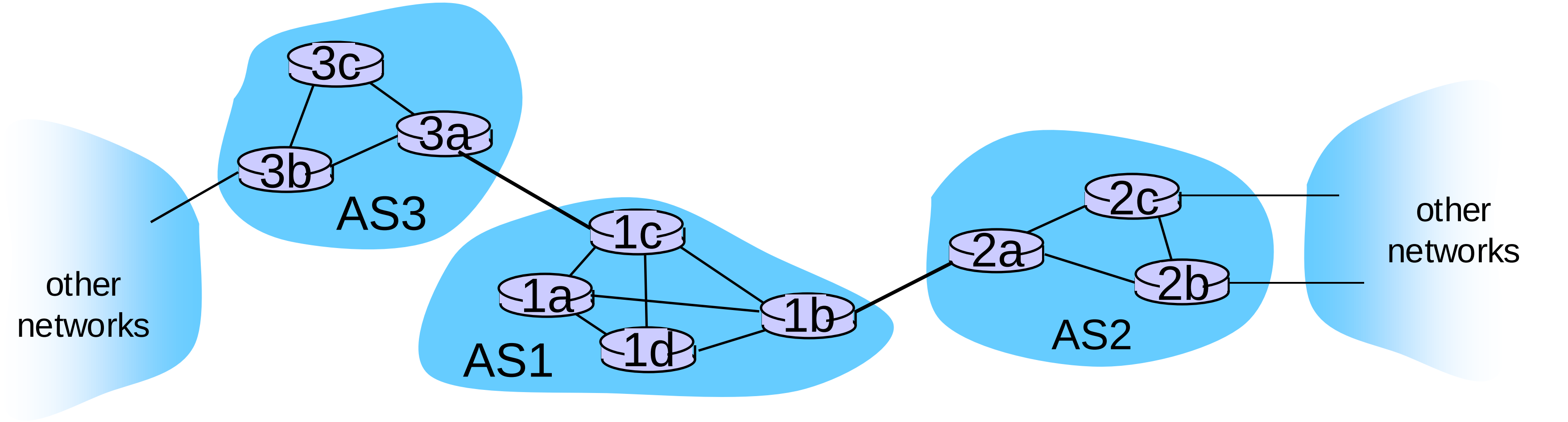
suppose router in AS1 receives datagram destined outside of AS1:

- router should forward packet to gateway router

AS1 must:

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

job of inter-AS routing!



Intra-AS Routing

also known as *interior gateway protocols (IGP)*

most common intra-AS routing protocols:

- RIP: Routing Information Protocol
- OSPF: Open Shortest Path First (IS-IS (Intermediate System to Intermediate System) protocol essentially same as OSPF)
- IGRP: Interior Gateway Routing Protocol (Cisco proprietary for decades, until 2016)

Summary

Today:

- Distance vector routing
- Intra-AS and inter-AS routing

Canvas discussion:

- Reflection
- Exit ticket

Next time:

- read 5.3 (OSPF) and 5.4 (BGP) of K&R
- follow on Canvas! material and announcements

Any questions?