BRIAN BARRETT SECURITY 10.04.2021 05:03 PM [wired October 4, 2021]

Why Facebook, Instagram, and WhatsApp All Went Down Today

The problem relates to something called BGP routing, and it could take a while longer to resolve.

Class Meeting: Lectures 9 & 10: Routing and Convergence

Kyle Jamieson

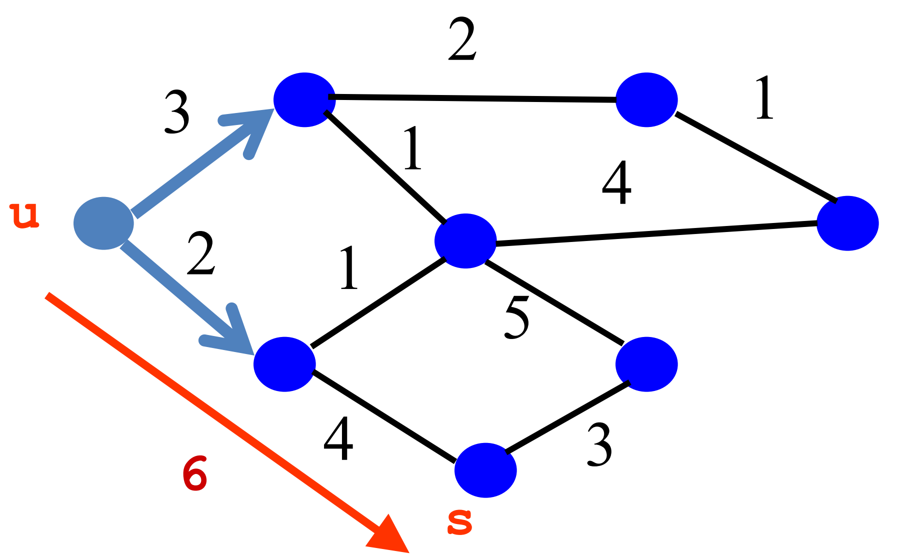
COS 461: Computer Networks

Routing vs. Forwarding

- **Routing:** control plane
  - Computing paths the packets will follow
  - Routers talk among themselves
    - → **Create the forwarding tables**
- **Forwarding:** data plane
  - Directing a data packet to an outgoing link
  - Using the forwarding tables

Shortest-Path Problem

- Compute: *path costs* to all nodes
  - **From** a given source ***u,*** **to** all other nodes
  - Edges: ***Cost*** of the path through each outgoing link
  - Next hop along the least-cost path to s



Link State Routing

Link State: Dijkstra's Algorithm

- Flood the topology information to all nodes
- Each node computes shortest paths to other nodes

Initialization

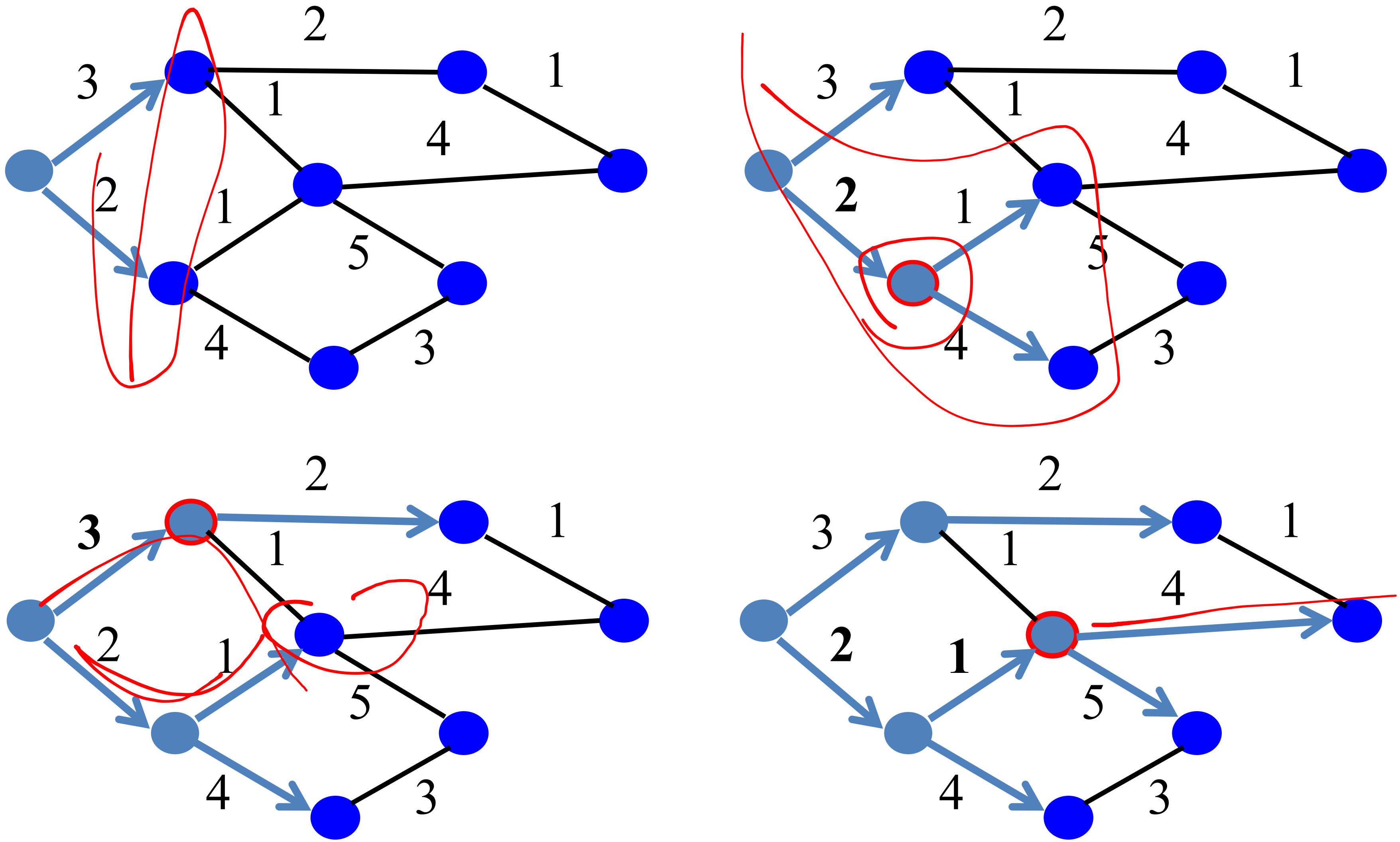
S = {u}
for all nodes v
  if (v is adjacent to u)
    D(v) = c(u,v)
  else D(v) = ∞

Loop

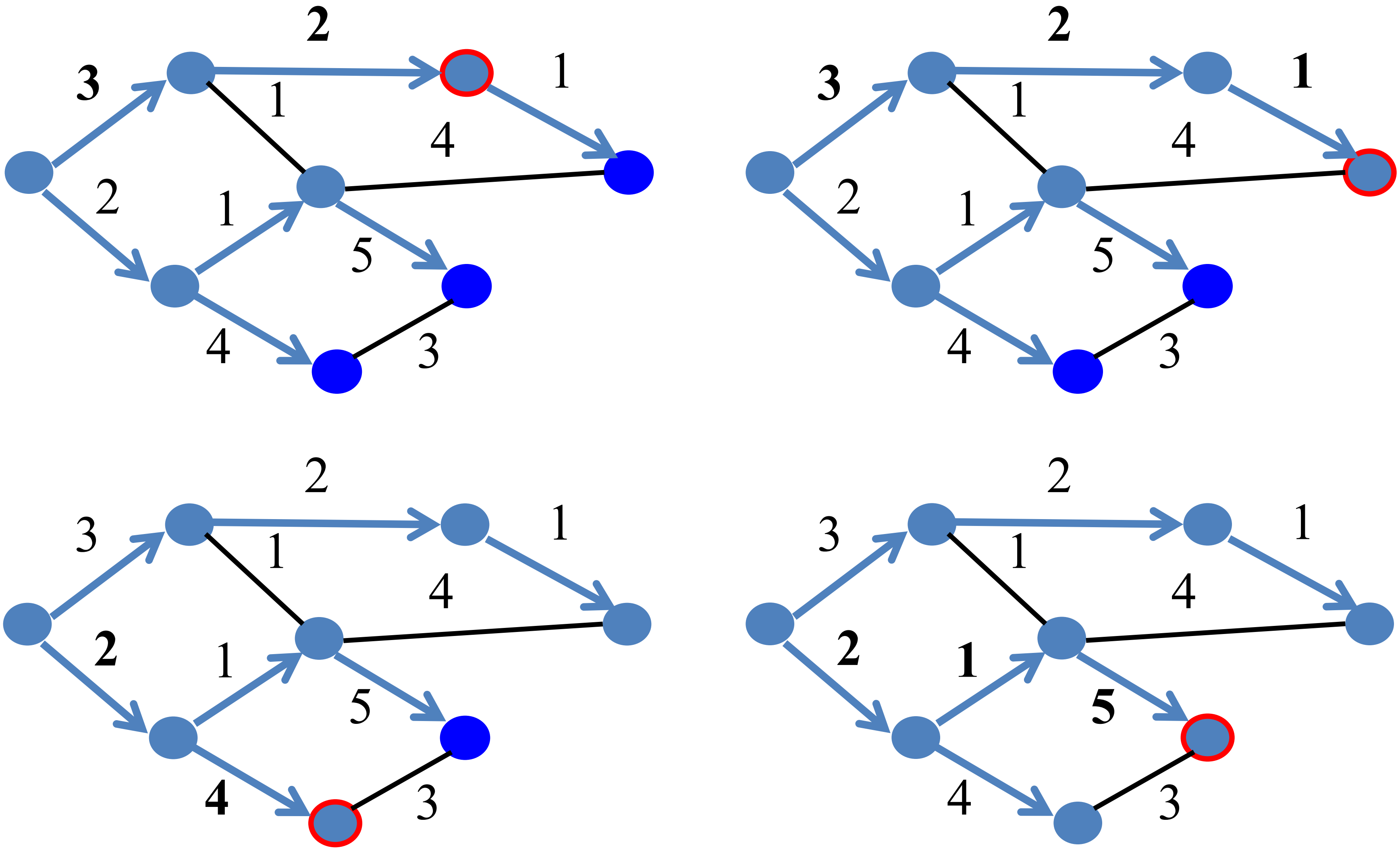
add w with smallest D(w) to S
update D(v) for all adjacent (to w) v:
  D(v) = min{D(v), D(w) + c(w,v)}
until all nodes are in S

Used in OSPF and IS-IS

Link-State Routing Example

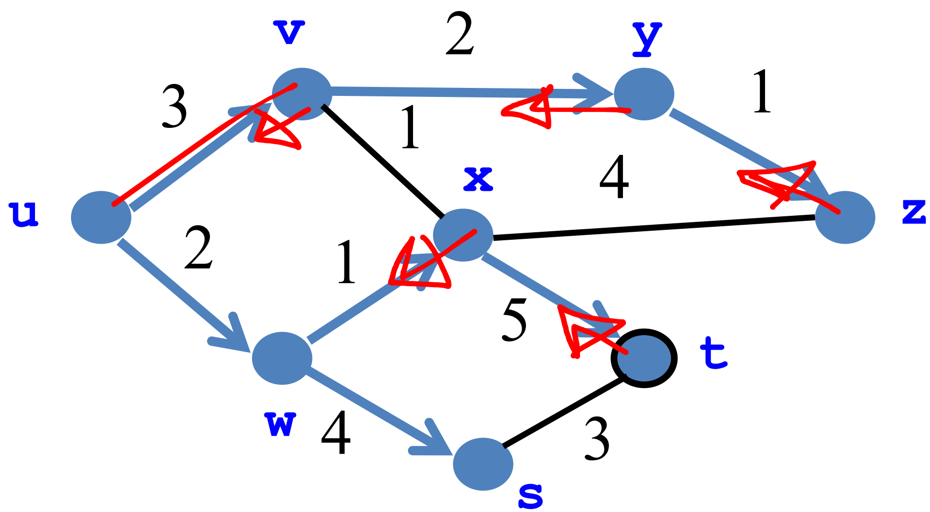


Link-State Routing Example (cont.)



Link State: Shortest-Path Tree

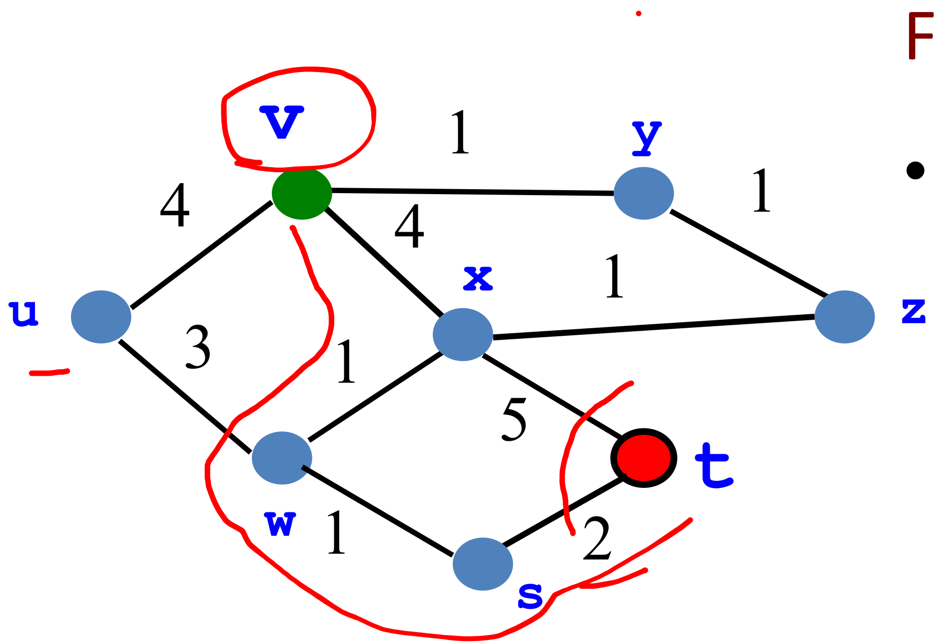
- Shortest-path tree from u



- Forwarding table at u

dest	link
v	(u,v)
w	(u,w)
x	(u,w)
y	(u,v)
z	(u,v)
s	(u,w)
t	(u,w)

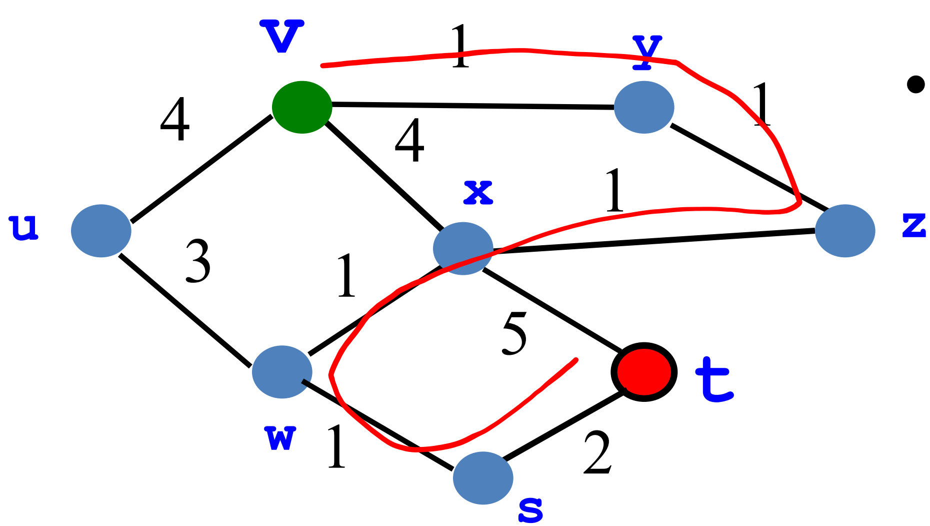
Link State: Shortest-Path Tree



Find shortest path **from** *t* **to** *v*

- Forwarding table entry at *t*?
  A. (t,x)
  B. (t, s)
  C. (t, v)

Link State: Shortest-Path Tree



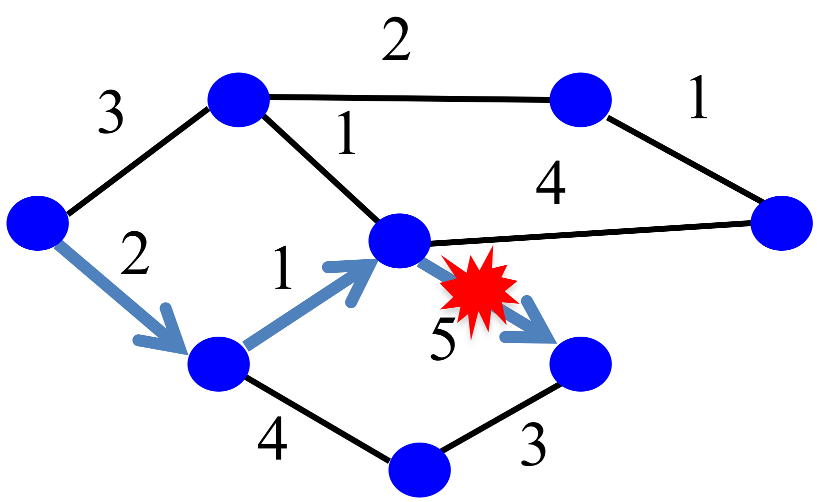
Find shortest path **from *t* to *v***

- Distance from *t* to *v*?
  - A.) 5
  - B.) 6
  - C.) 7
  - D.) 8
  - E.) 9

Routing Convergence: Link-State Routing

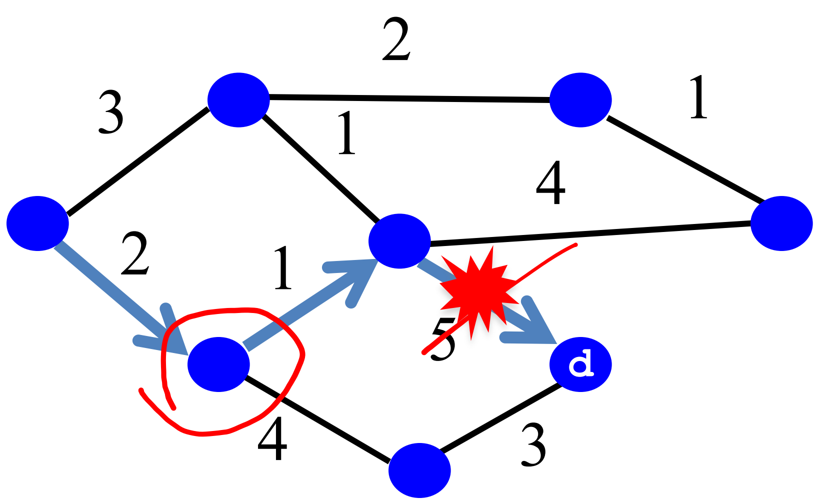
Transient Disruptions

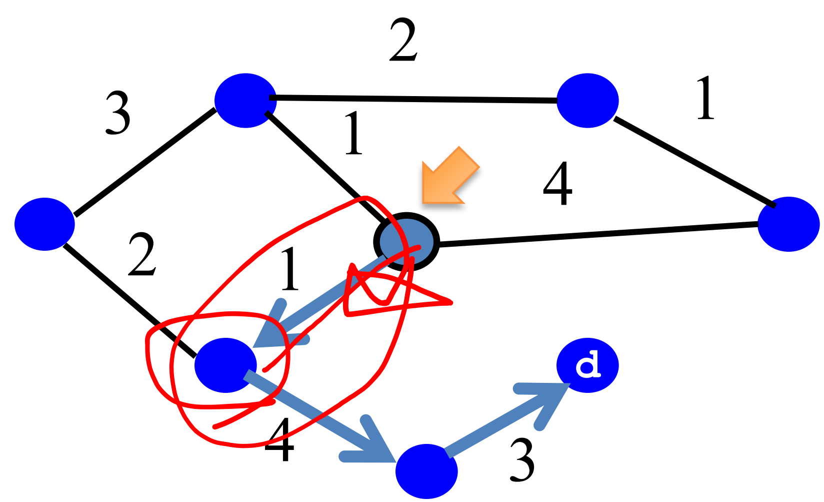
- Detection delay
  - A node does not detect a failed link immediately
  - … and forwards data packets into a “blackhole”
  - Depends on timeout for detecting lost hellos



Transient Disruptions

- Inconsistent link-state database
  - Some routers know about failure before others
  - Inconsistent paths cause transient forwarding loops

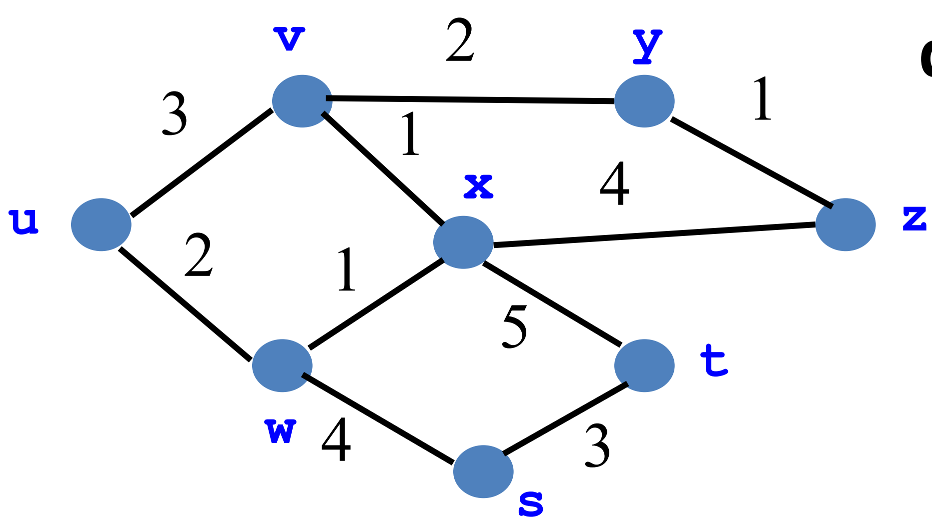




Distance Vector Routing

Distance Vector: Bellman-Ford Algo

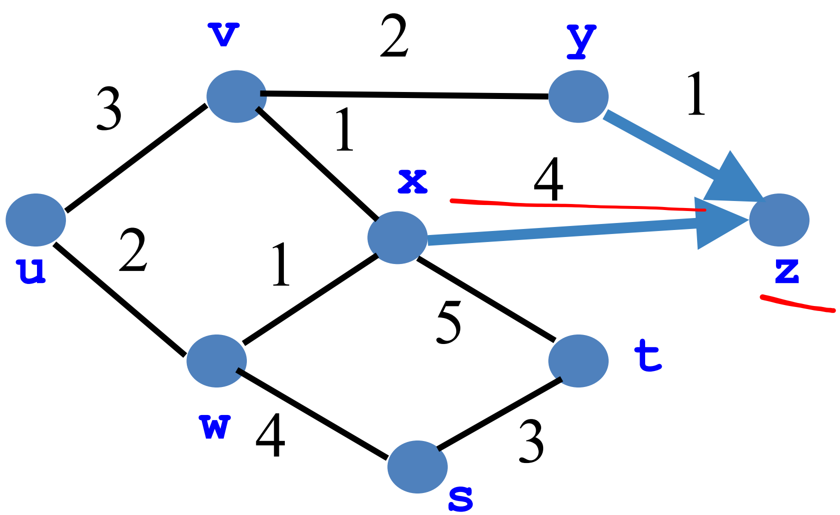
- Define distances at each node x
  - $d_x(y)$ = cost of least-cost path from x to y
- Update distances based on neighbors
  - $d_x(y) = \min \{c(x,v) + d_v(y)\}$ over all neighbors v

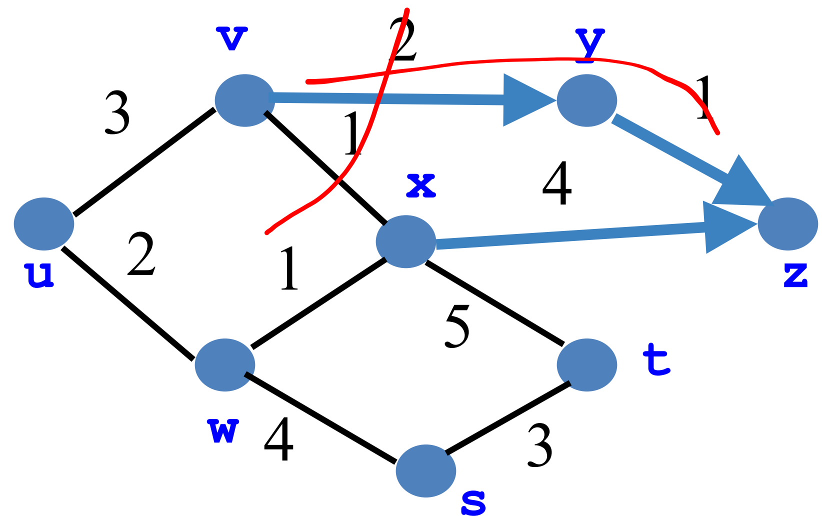


$$d_u(z) = \min\{\, c(u,v) + d_v(z),\; c(u,w) + d_w(z)\}$$

Used in RIP and EIGRP

Distance Vector Example



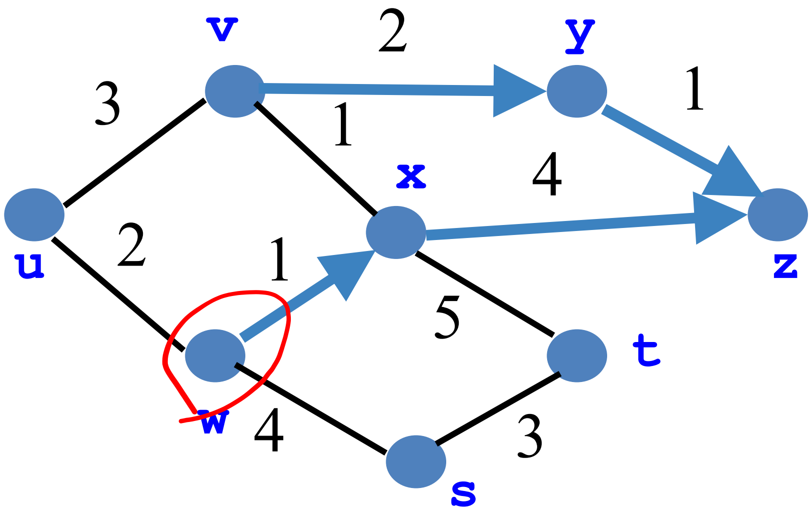


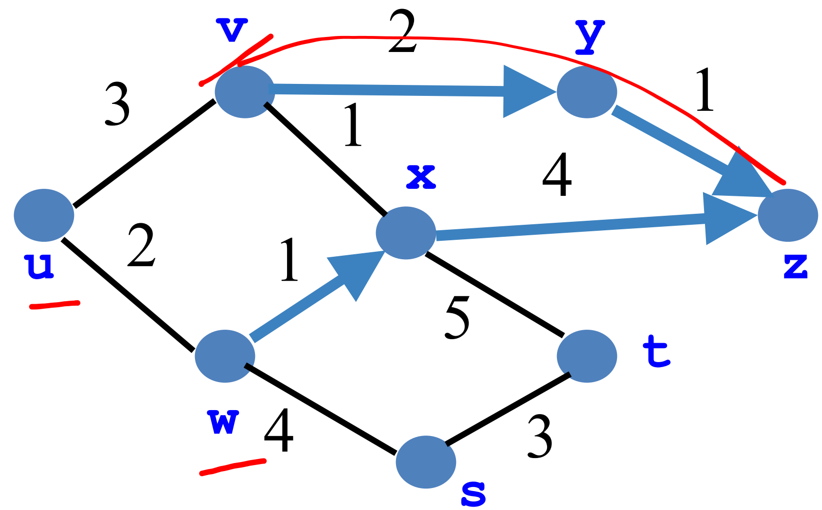
$d_y(z) = 1$

$d_x(z) = 4$

$d_v(z) = \min\{\ 2 + d_y(z),$
$\quad 1 + d_x(z)\ \}$
$= 3$

Distance Vector Example (Cont.)





$d_w(z) = \min\{ 1 + d_x(z),$
$4 + d_s(z),$
$2 + d_u(z) \}$
$= 5$

$d_u(z)$ = ???

A.) 5 (via v)

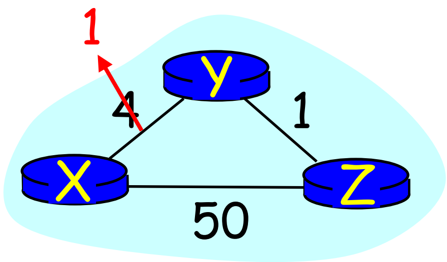
B.) 6 (via v)

C.) 6 (via w)

D.) 7 (via w)

Slow Convergence in Distance-Vector Routing

Distance Vector: Link Cost Changes



- Link cost decreases and recovery
  - Node updates the distance table
  - **Rule:** Least-cost path's cost changed? notify neighbors

D^Y = Distances known to Y

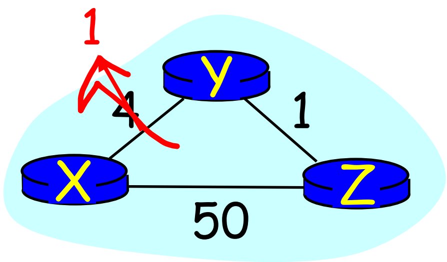
D^Y	via X	via Z
to: X	4	6

D^Z	via X	via Y
to: X	50	5

time → t_0 t_1 t_2

Distance Vector: Link Cost Changes

- Link cost decreases and recovery
  - Node updates the distance table
  - **Rule:** Least-cost path's cost changed? notify neighbors



D^Y = Distances known to Y

D^Y (via)	X	Z
to: X	4	6

D^Y	X	Z
X	1	6

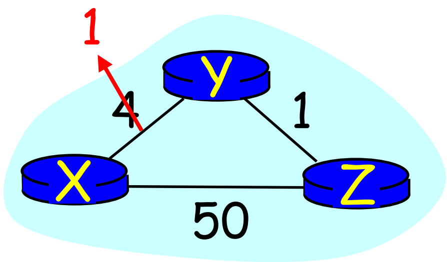
D^Z (via)	X	Y
to: X	50	5

D^Z	X	Y
X	50	5

c(X,Y) change

time t_0 t_1 t_2

Distance Vector: Link Cost Changes



- Link cost decreases and recovery
  - Node updates the distance table
  - **Rule:** Least-cost path's cost changed? notify neighbors

"good news travels fast"

D^Y = Distances known to Y

D^Y (via)	X	Z
to: X	(4)	6

D^Y (via)	X	Z
X	(1)	6

D^Y	X	Z
X	(1)	6

D^Y	X	Z
X	(1)	3

D^Z (via)	X	Y
to: X	50	(5)

D^Z	X	Y
X	50	(5)

D^Z (via)	X	Y
X	50	(2)

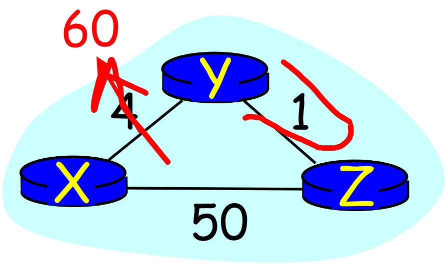
D^Z	X	Y
X	50	(2)

c(X,Y) change

time → t_0 t_1 t_2

Distance Vector: Link Cost Changes

- Link cost increases and failures
  - **"Count to infinity"** problem!



D^Y (via)	X	Z
to: X	4	6

D^Y	X	Z
X	60	6

D^Z (via)	X	Y
to: X	50	5

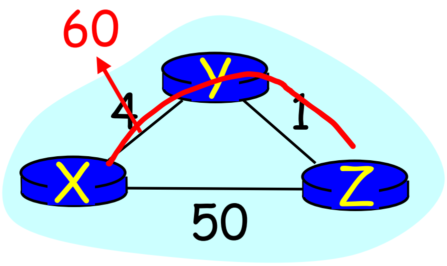
D^Z	X	Y
X	50	5

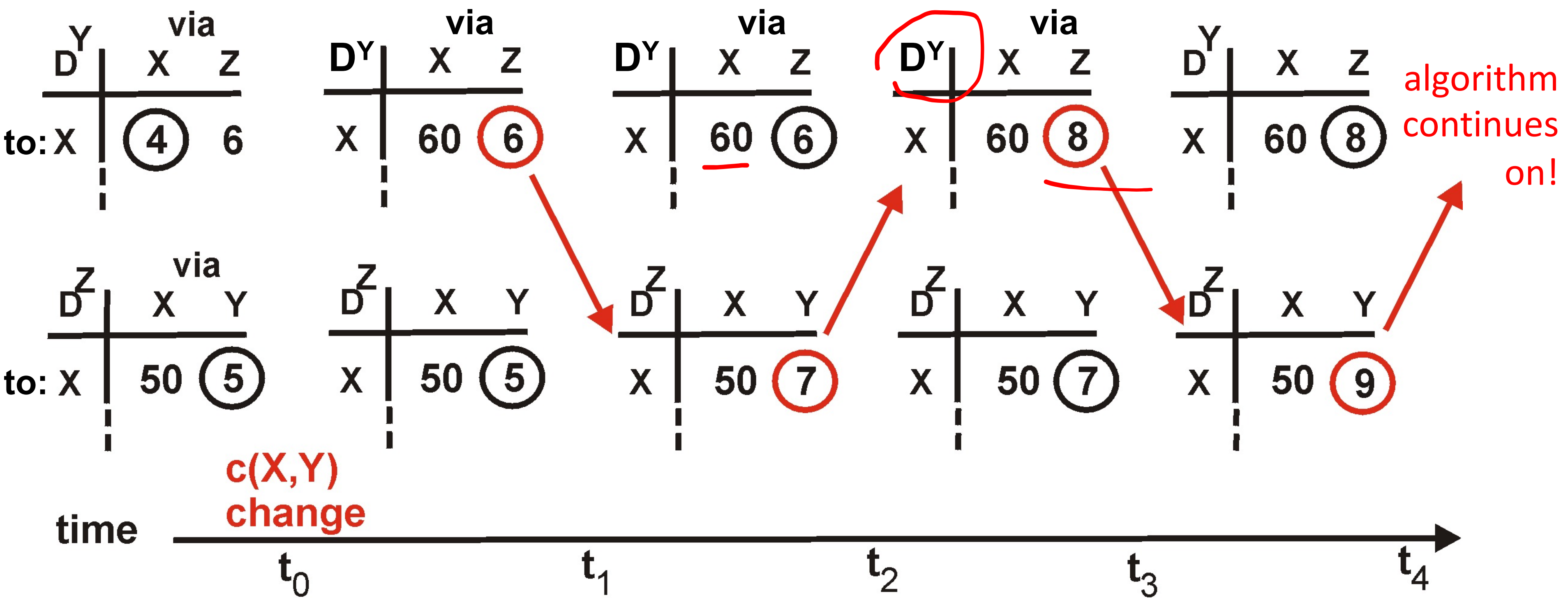
c(X,Y) change

time: t_0 → t_1 → t_2 → t_3 → t_4

Distance Vector: Link Cost Increase

- Link cost **increases and failures**
  - **"Count to infinity"** problem!



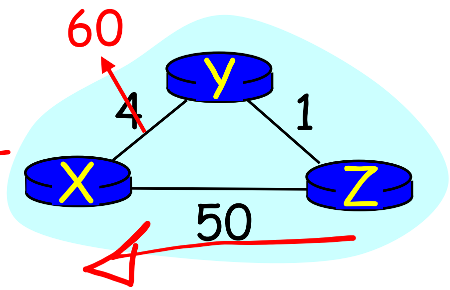


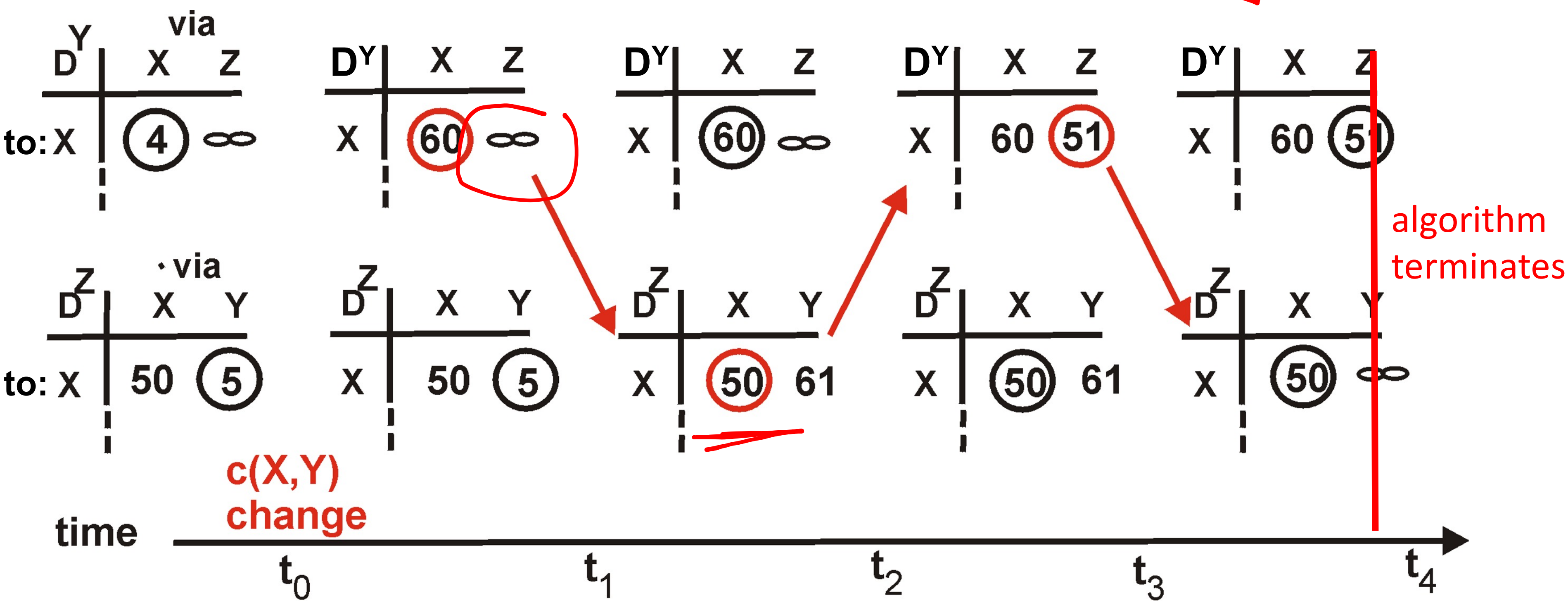
Distance Vector: Poison Reverse

- If Z routes through Y to X,

  then Z tells Y its (Z's) distance to X is ∞

  (so Y won't route to X via Z)

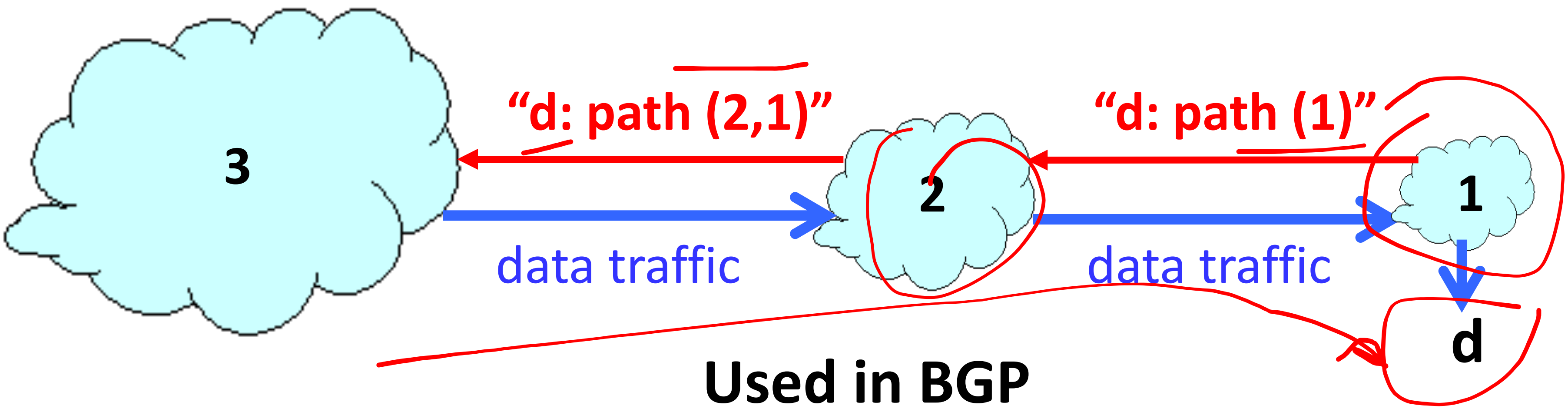




Path Vector Routing

Path-Vector Routing

- Extension of distance-vector routing
  - Support flexible routing policies
- Key idea: advertise the entire path
  - Distance vector: send *distance metric* per dest d
  - Path vector: send the *entire path* for each dest d

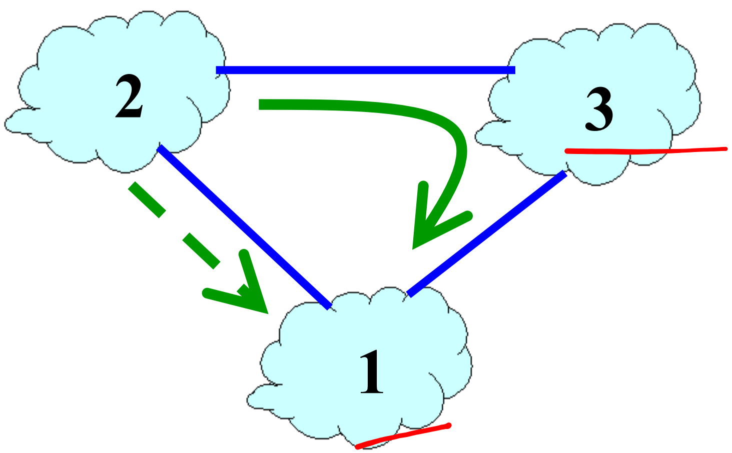


Used in BGP

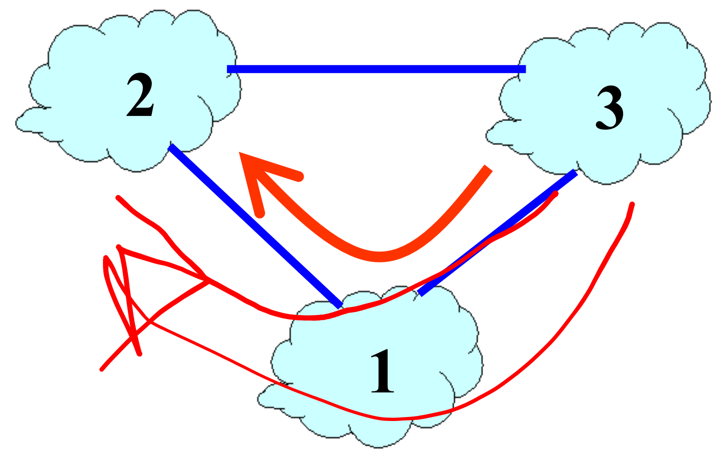
Path-Vector: Flexible Policies

- Each node can apply local policies
  - Path selection: Which path to use?
  - Path export: Which paths to advertise?

Node 2 prefers “2, 3, 1” over “2, 1”



Node 1 doesn’t let 3 hear the path “1, 2”

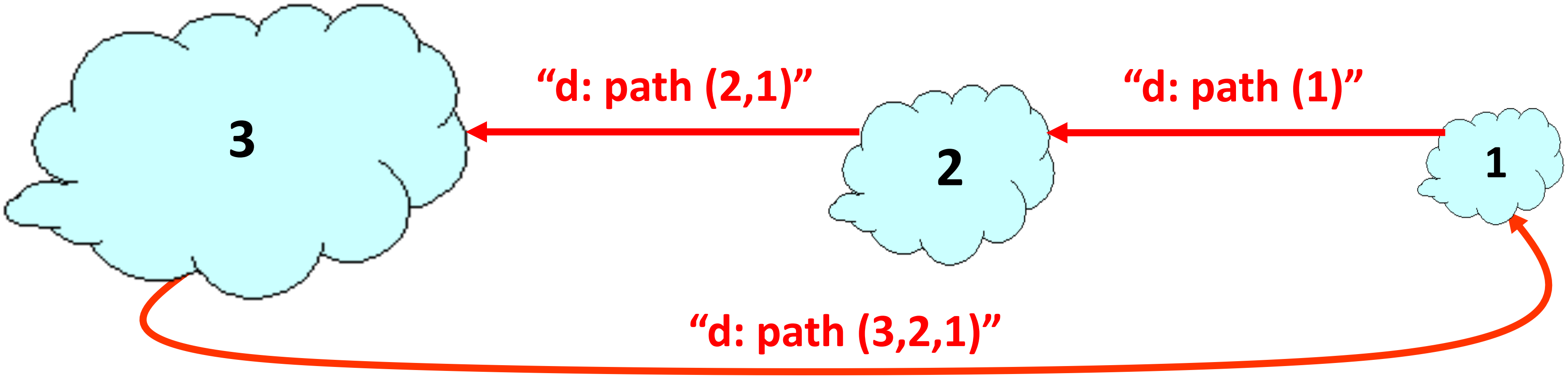


Reducing Convergence Time With Path-Vector Routing

(*e.g.:* Border Gateway Protocol)

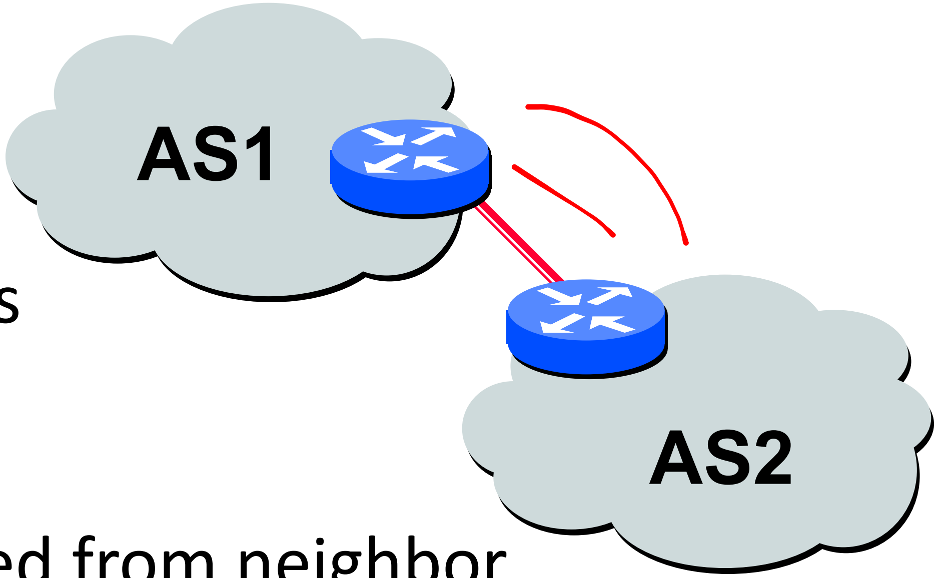
Faster Loop Detection

- Node can easily detect a loop
  - Look for its own node identifier in the path
  - E.g., node 1 sees itself in the path "3, 2, 1"
- Node can simply discard paths with loops
  - E.g., node 1 simply discards the advertisement

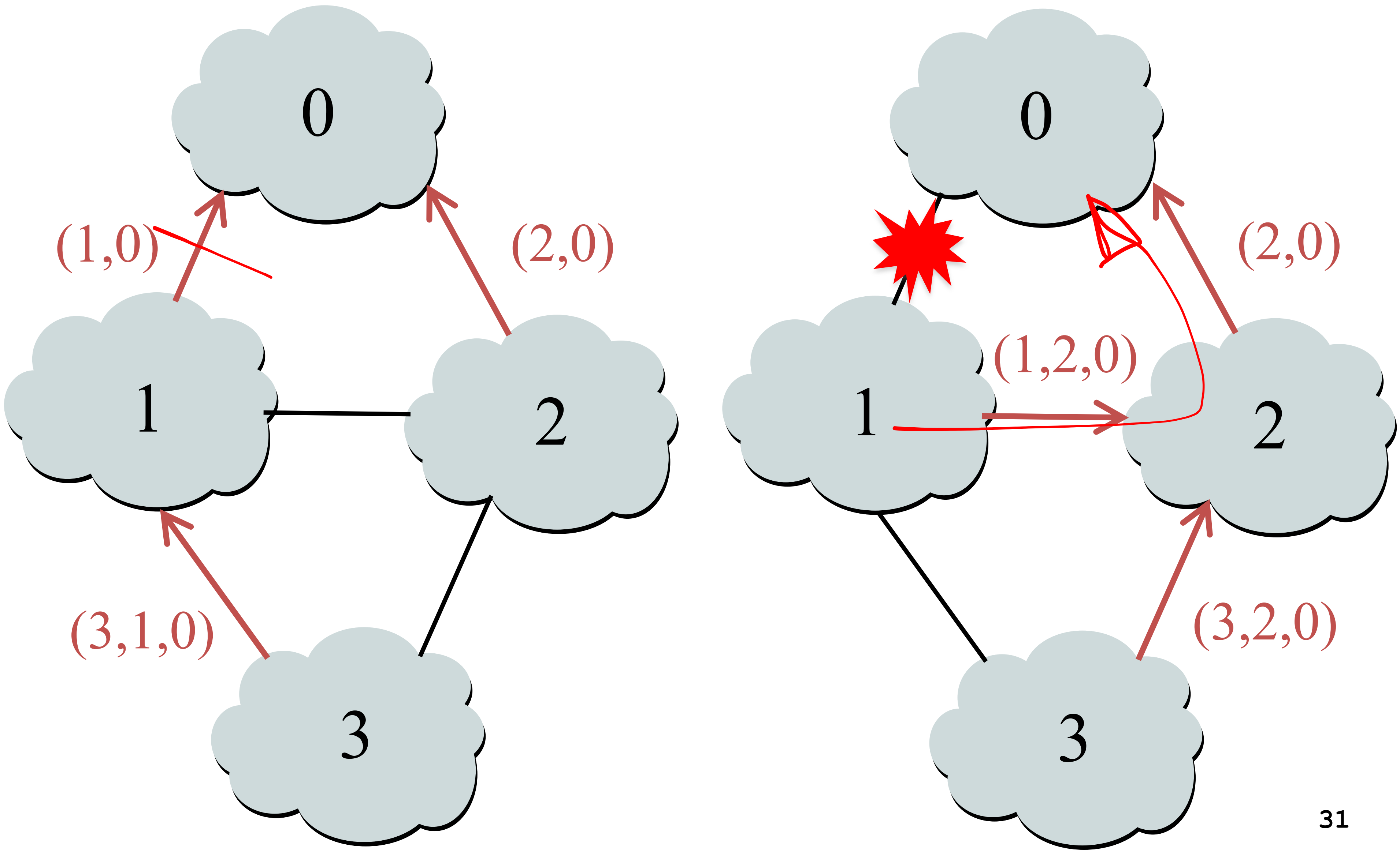


BGP Session Failure

- BGP runs over TCP
  - BGP only sends updates when changes occur
  - TCP doesn't detect lost connectivity on its own
- Detecting a failure
  - Keep-alive: 60 seconds
  - Hold timer: 180 seconds
- Reacting to a failure
  - Discard all routes learned from neighbor
  - Send new updates for any routes that change

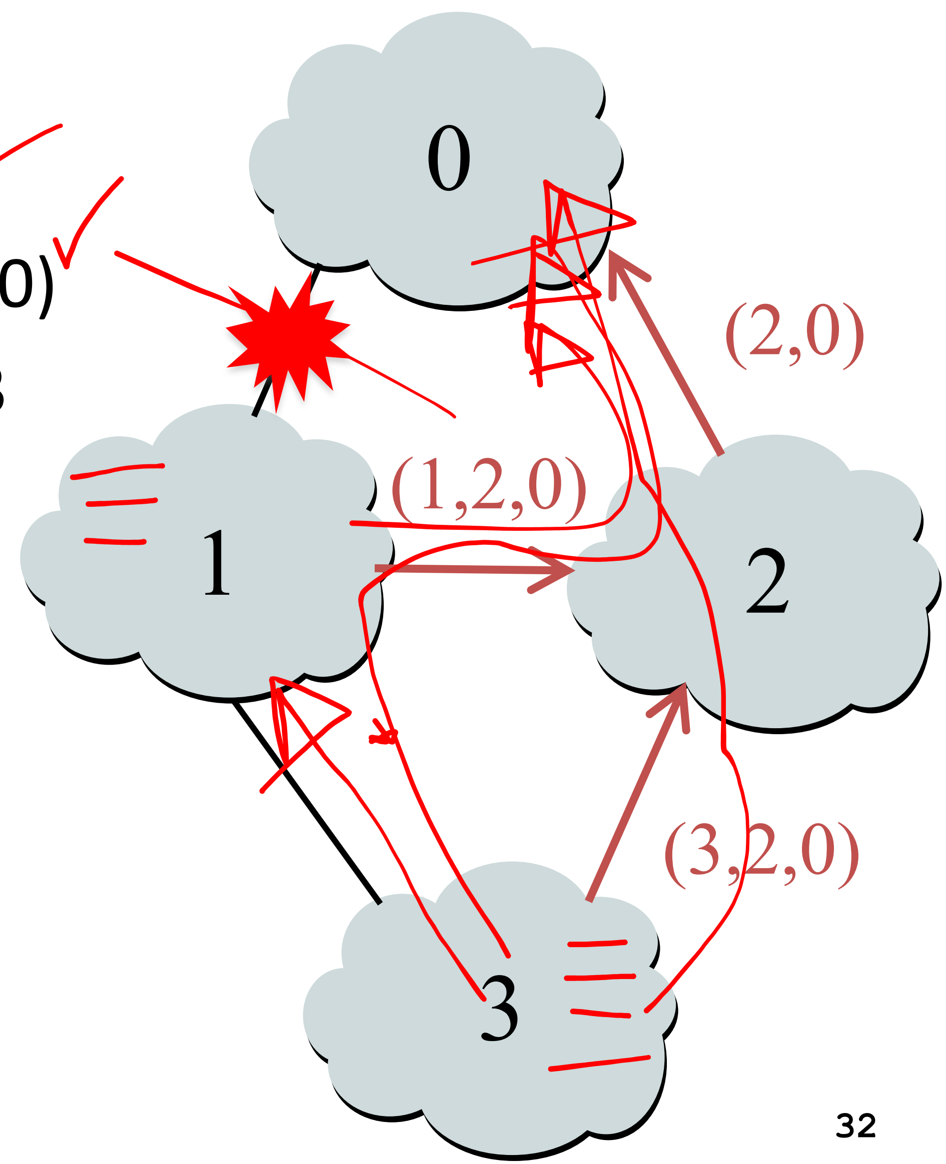


Routing Change: Before and After



Routing Change: Path Exploration

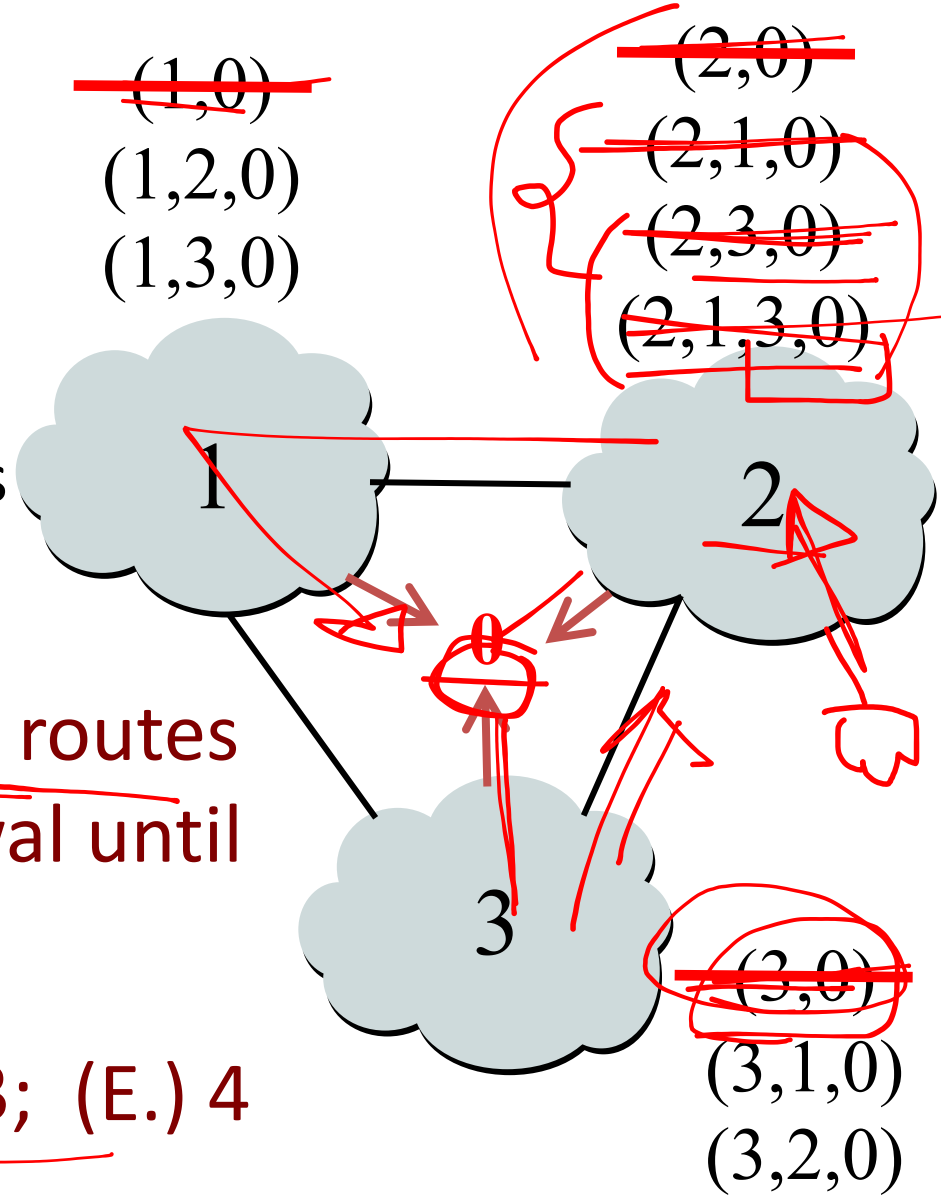
- AS 1
  - Delete the route (1,0)
  - Switch to next route (1,2,0)
  - Send route (1,2,0) to AS 3
- AS 3
  - Sees (1,2,0) replace (1,0)
  - Compares to route (2,0)
  - Switches to using AS 2



Routing Change: Path Exploration

- Initial: All AS use direct
- Then destination 0 dies
  - All ASes lose direct path
  - All switch to longer paths
  - Eventually withdrawn
- How many intermediate routes following (2,0) withdrawal until no route known to 2?

(A.) 0; (B.) 1; (C.) 2; (D.) 3; (E.) 4



BGP Converges Slowly

- Path vector avoids count-to-infinity
  - But, ASes still must explore many alternate paths to find highest-ranked available path
- Fortunately, in practice
  - Most popular destinations have stable BGP routes
  - Most instability lies in a few unpopular destinations
- Still, lower BGP convergence delay is a goal
  - Can be tens of seconds to tens of minutes

Conclusion

- Routing protocols cope with change
  - Planned topology and configuration changes
  - Unplanned failure and recovery
- Routing-protocol convergence
  - Transient period of disagreement
  - Blackholes, loops, and out-of-order packets