Class Meeting, Lectures 9 & 10: Routing in Multi-hop Networks

Kyle Jamieson

COS 461: Computer Networks

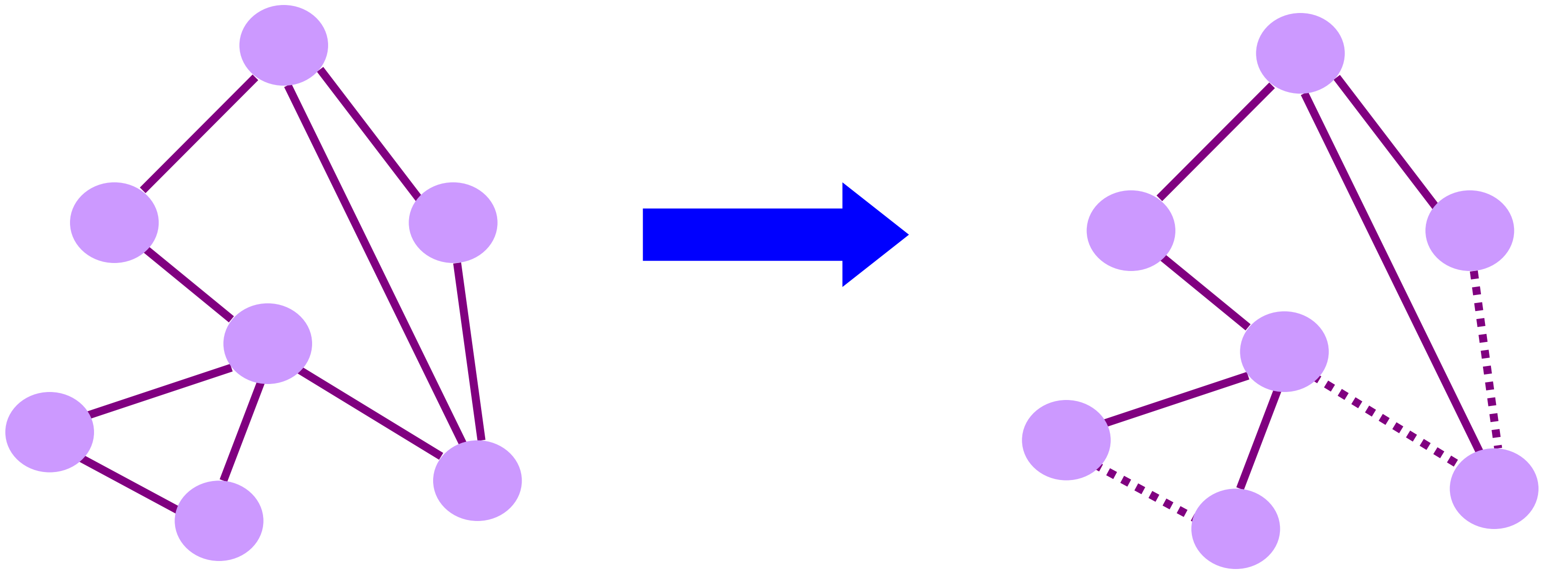
Today

1. Ethernet Spanning Tree Protocol
   - Spanning tree

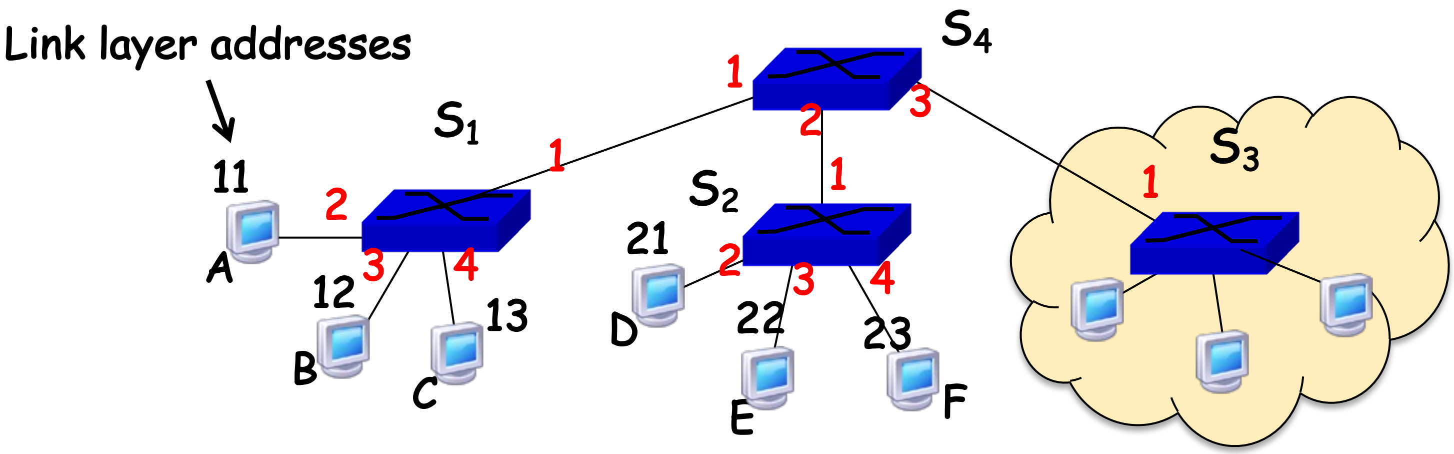
2. IP Interior Gateway Protocol
   - Shortest paths tree

Spanning Tree

- One tree that reaches every node
  - Single path between each pair of nodes
  - No loops, so can support broadcast easily
  - But, paths are long, and some links not used



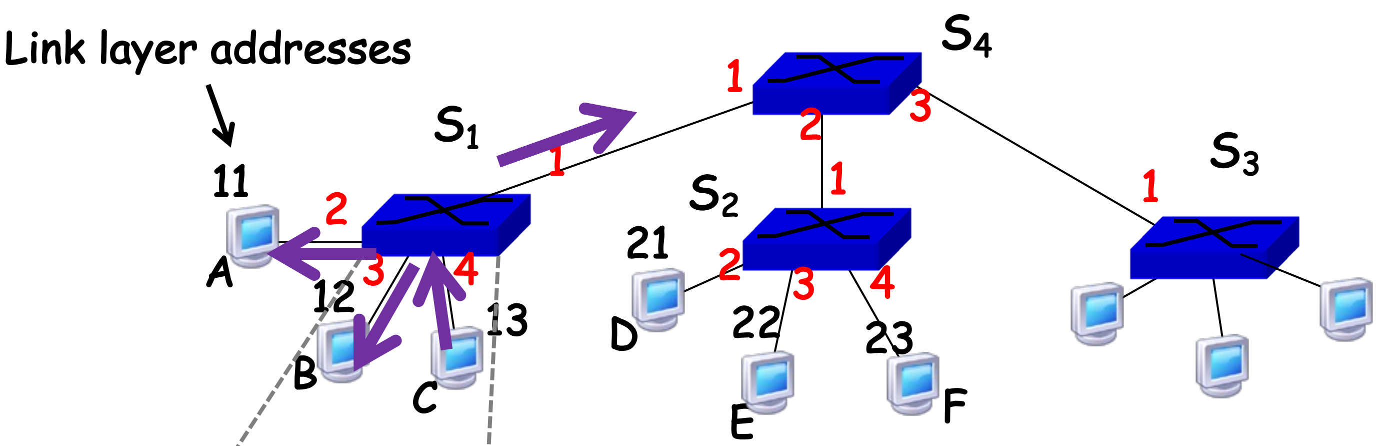
Motivation for Spanning Tree: Extended LANs



- Switches can connect LANs as well as hosts
- Sometimes called bridges in this context
- The entirety is called an extended LAN

Spanning Tree: Motivating Example

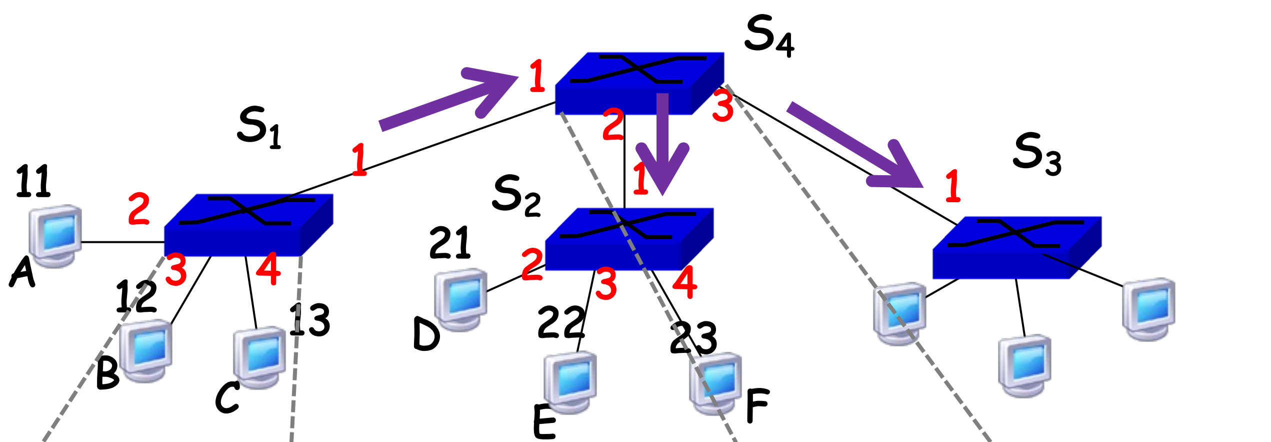
Suppose C sends frame to F, F responds to C



Address	Port
13	4

Spanning Tree: Motivating Example

Suppose C sends frame to F, F responds to C

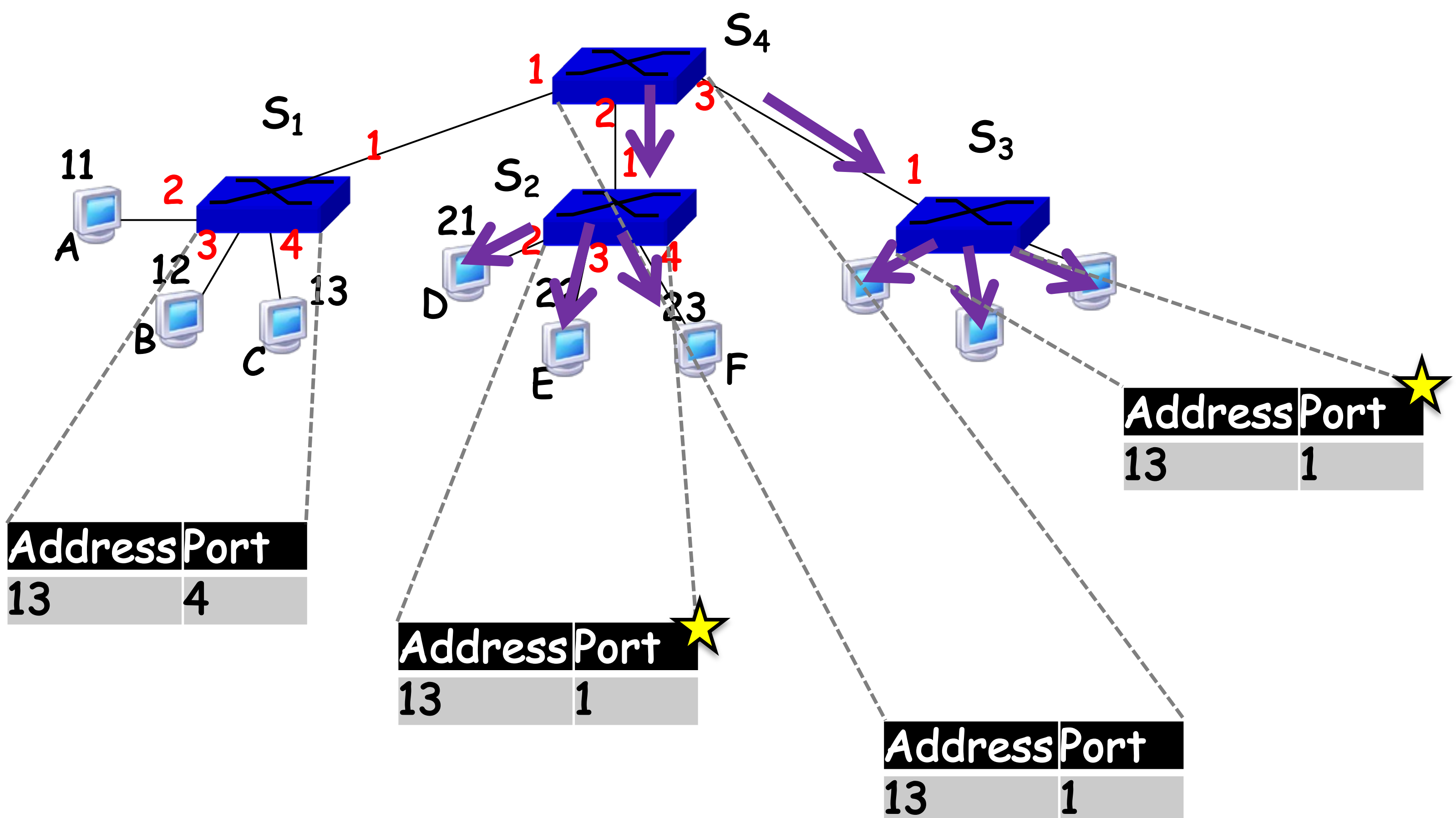


Address	Port
13	4

Address	Port
13	1

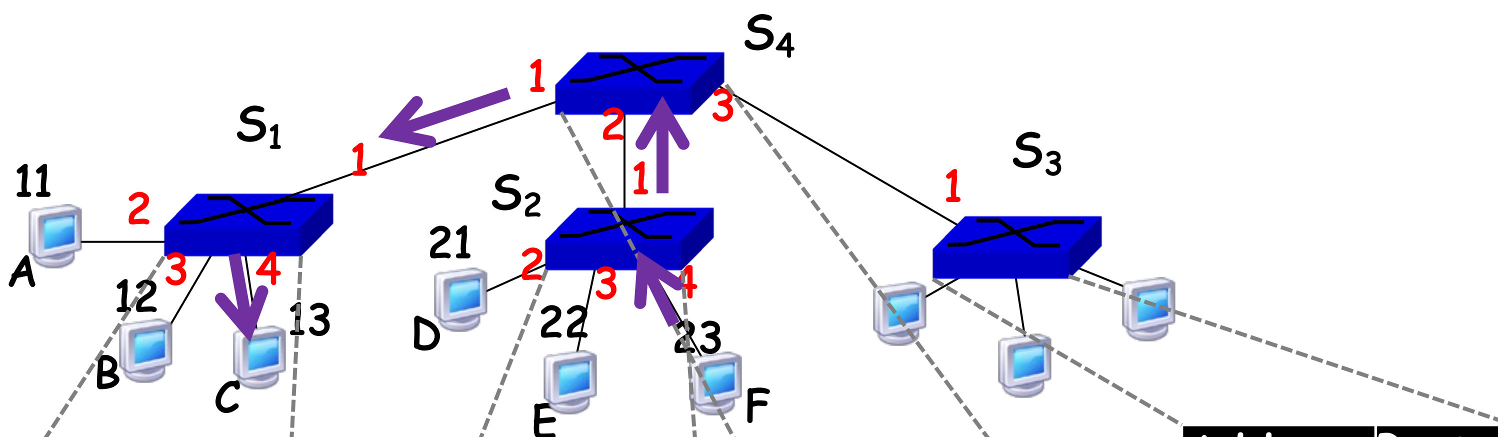
Spanning Tree: Motivating Example

Suppose C sends frame to F, F responds to C



Spanning Tree: Motivating Example

F responds to C



Address	Port
13	4
23	1 ★

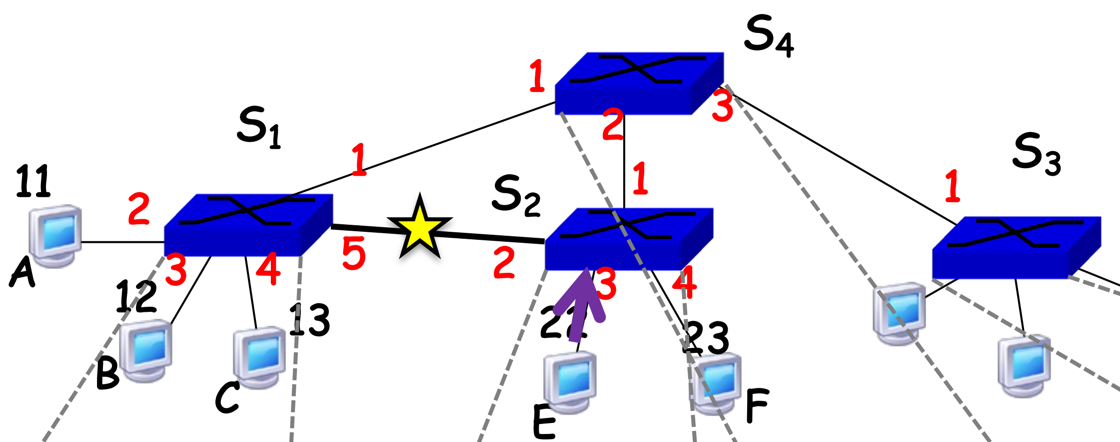
Address	Port
13	1
23	4 ★

Address	Port
13	1
23	2 ★

Address	Port
13	1

Problem: Forwarding loops

E sends a frame to A (MAC address 11)



Address	Port
13	4
23	1

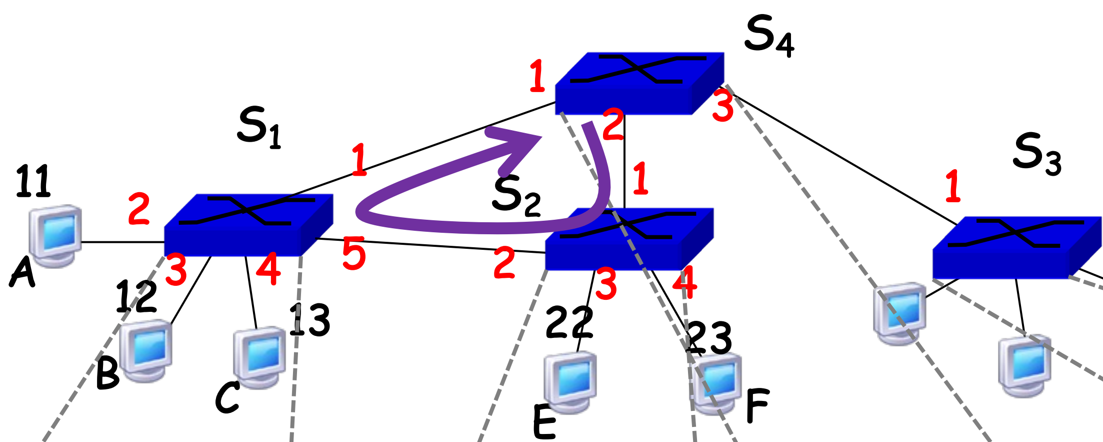
Address	Port
13	1
23	4
22	3

Address	Port
13	1
23	2
22	2

Address	Port
13	1

Problem: Forwarding loops

Incoming frame to E (MAC address 22)



Address	Port
13	4
23	1
22	1

Address	Port
13	1
23	4
~~22~~	~~3~~
22	2

Address	Port
13	1
23	2
22	2

Address	Port
13	1

Problem: Forwarding loops

- Can't learn the direction of a source if it's in more than one direction, so bridge learning algorithm breaks

- Why might loops form?
  - Inadvertently: many people responsible for network, one person adds a bridge

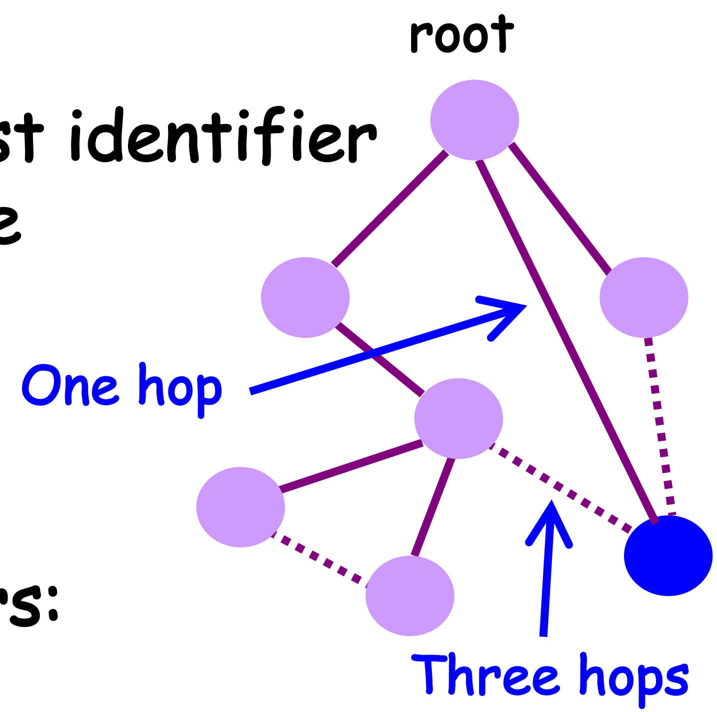
- Intentionally: more connections between bridges increases redundancy, helping to cope with failure
  - So we need to revise the bridge learning algorithm

The spanning tree protocol (STP)

- Manager at DEC asked Radia Perlman to build a switch (bridge) to connect two Ethernets
- Perlman's idea: Switches agree on a loop-free and connected spanning tree
- Implementers at DEC resisted (wanted simplest possible design), first customer site connected bridge to one Ethernet twice, generating a "broadcast storm"
- Once the spanning tree is formed:
  - Switches block some ports from sending or receiving data
  - Switches continue using the learning switch algorithm to forward over the spanning tree

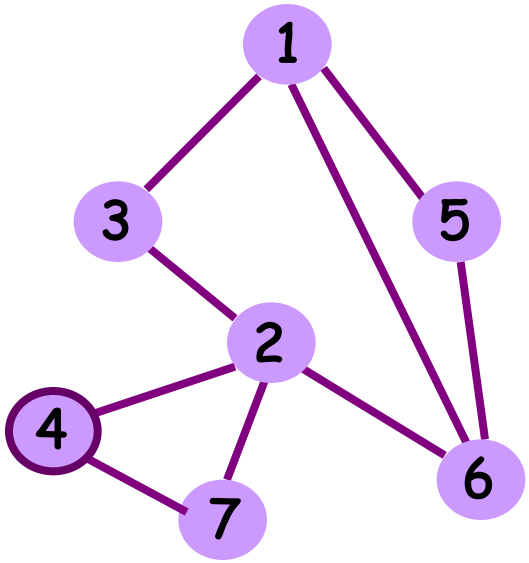
Spanning Tree Algorithm

- Elect a root
  - The switch with the smallest identifier
  - And form a tree from there
- Algorithm
  - Initialize:
    - "I am the root."
  - Repeatedly talk to neighbors:
    - "I think node Y is the root"
    - "My distance from Y is d"
  - Update based on neighbors
    - First priority: Prefer smaller id as the root
    - Second priority: Prefer smaller distance to root d+1



Used in Ethernet LANs

Spanning Tree Example: Switch #4



- **Switch #4 thinks it is the root**
  - Sends (4, 0, 4) message to 2 and 7
  - Notation: (my root, my distance, my ID)
- **Switch #4 hears from #2**
  - Receives (2, 0, 2) message from 2
  - Thinks #2 is root and it's one hop away
- **Switch #4 hears from #7**
  - Receives (2, 1, 7) from 7
  - But, this is a longer path, so 4 prefers 4-2 over 4-7-2
    - And removes 4-7 link from the tree

Today

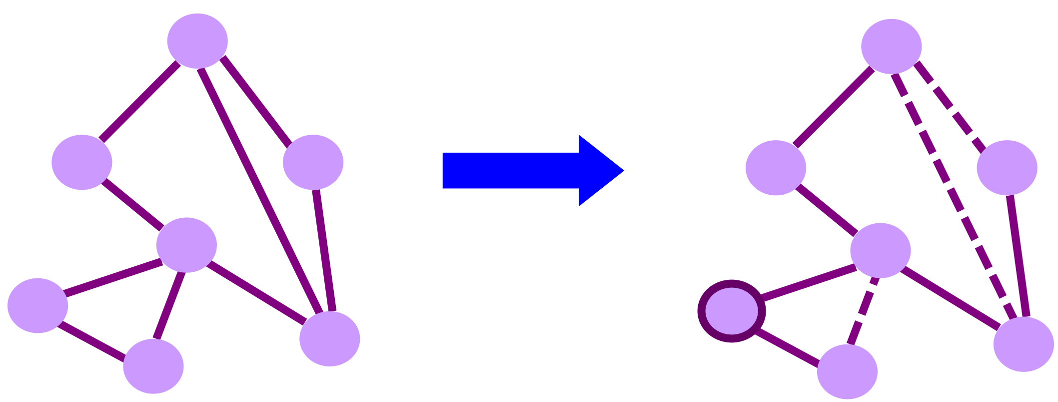
1. Ethernet Spanning Tree Protocol
   – Spanning tree

2. IP Interior Gateway Protocol
   – Shortest paths tree

Shortest Paths Routing: Context

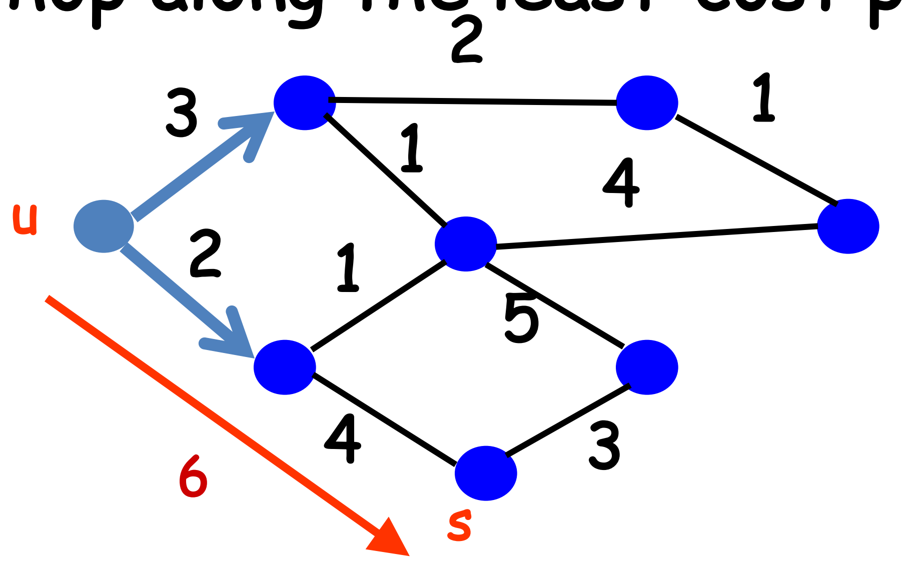
- Intra-domain routing
  - Domain: group of routers owned by a single entity, typically numbering at most 100s
  - Distance Vector, Link State protocols: types of Interior Gateway Protocol (IGP)

- Shortest path(s) between pairs of nodes
  - A shortest-path tree rooted at each node
  - Min hop count or min sum of edge weights



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: Dijkstra's Algorithm

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

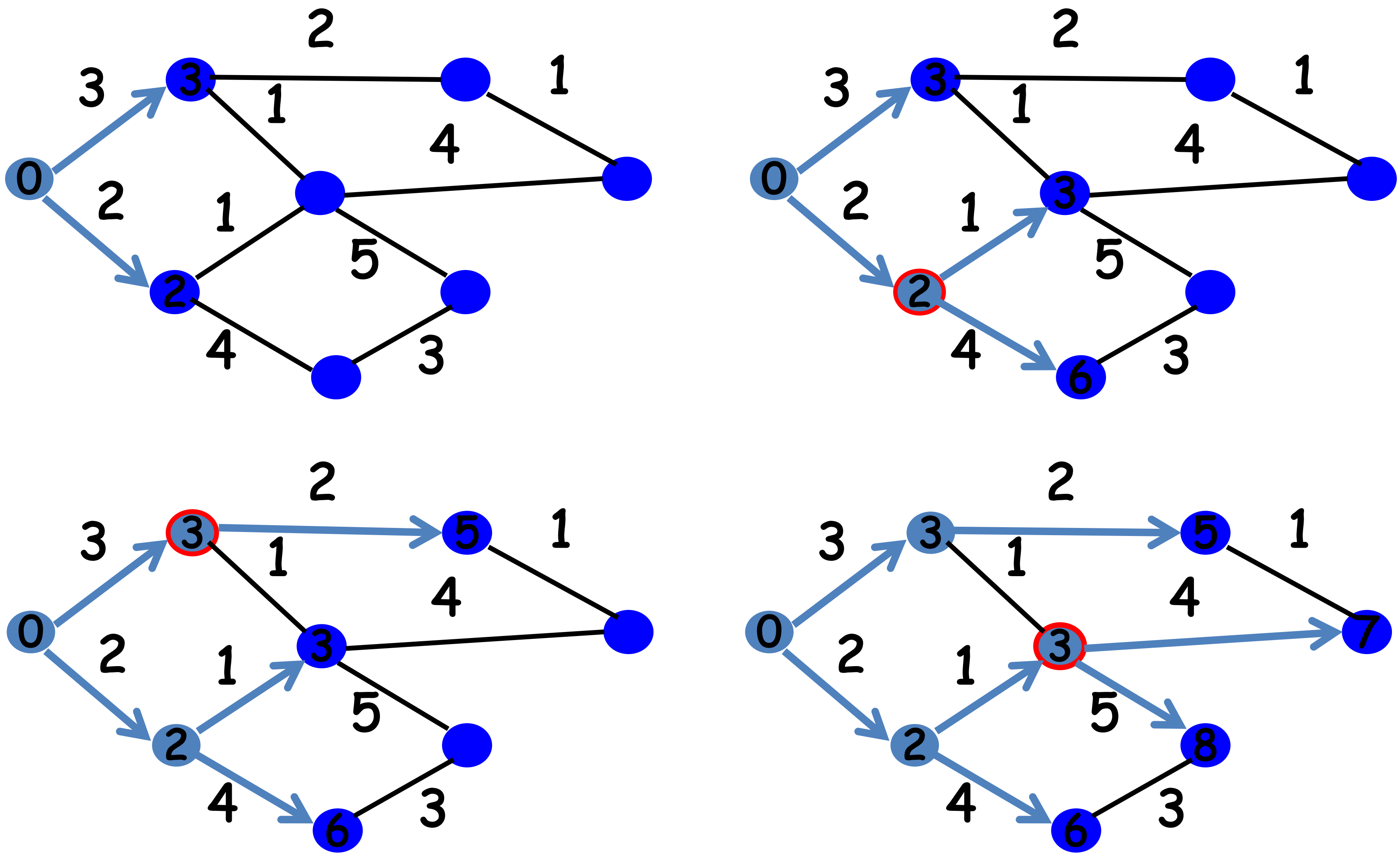
Initialization

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

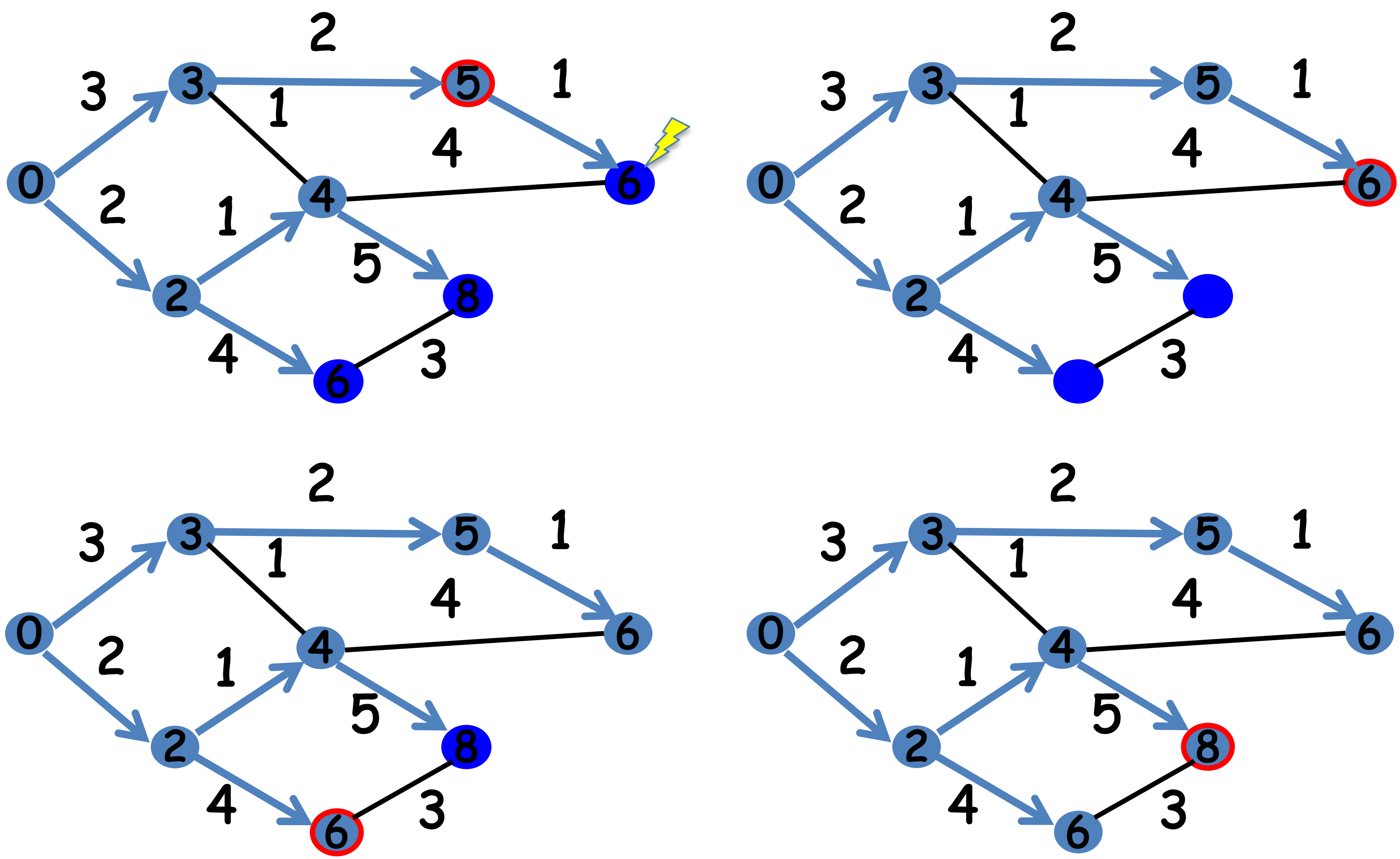
Loop

```
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
```

Link-State Routing Example

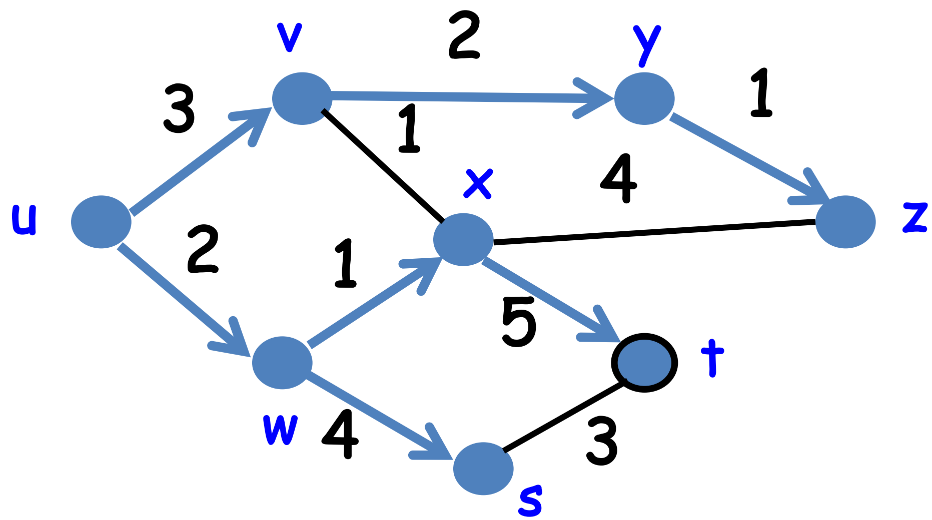


Link-State Routing Example (cont.)



Link State: Shortest-Path Tree

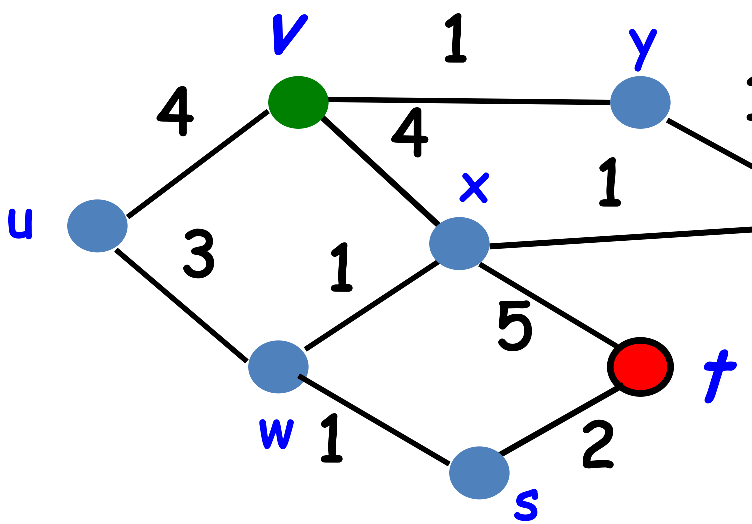
- Shortest-path tree from u



- 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

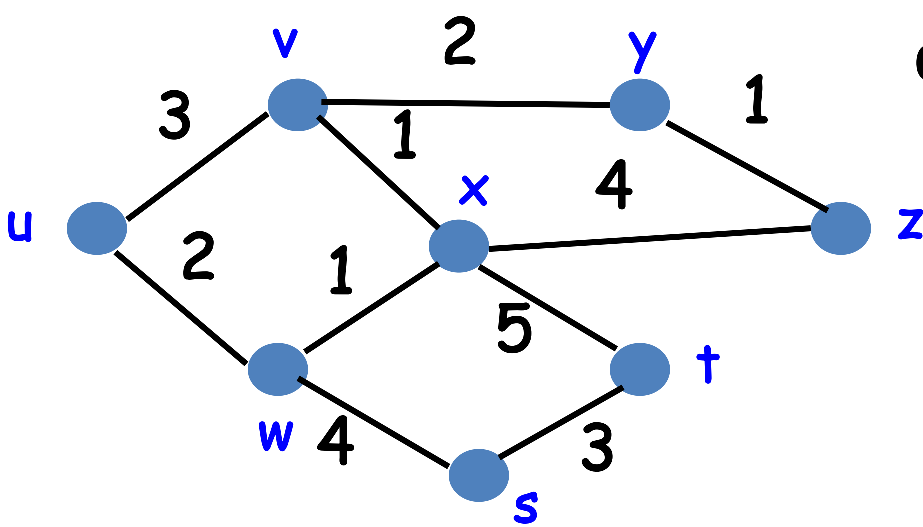


Find shortest path from *t* to *v*

- Forwarding table entry at *t* ?
  (Y) (t, x) (M) (t, s)
- Distance from *t* to *v*?
  (Y) 6 (M) 7 (C) 8 (A) 9

Distance Vector: Bellman-Ford Algorithm

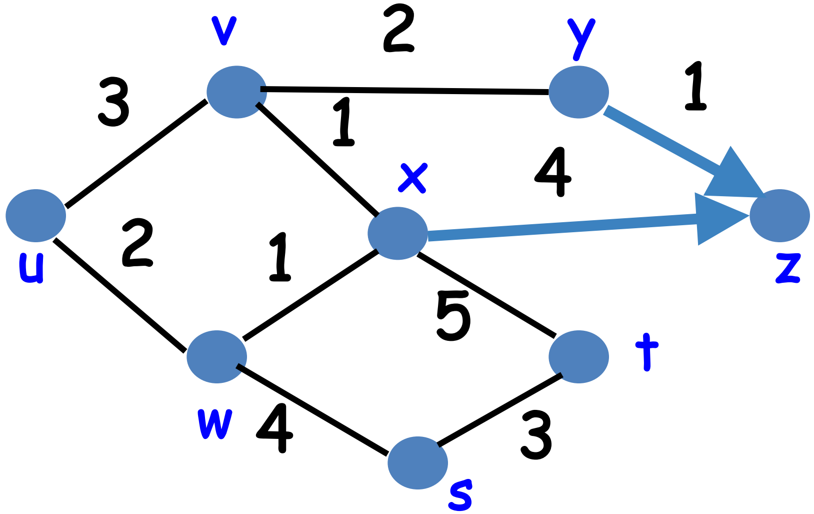
- Define distances at each node x
  - $d_x(y)$ = cost of least-cost path from x to y
- Update distances based on neighbors
  - $d_x(y) = \min \{c(x,v) + d_v(y)\}$ over all neighbors v

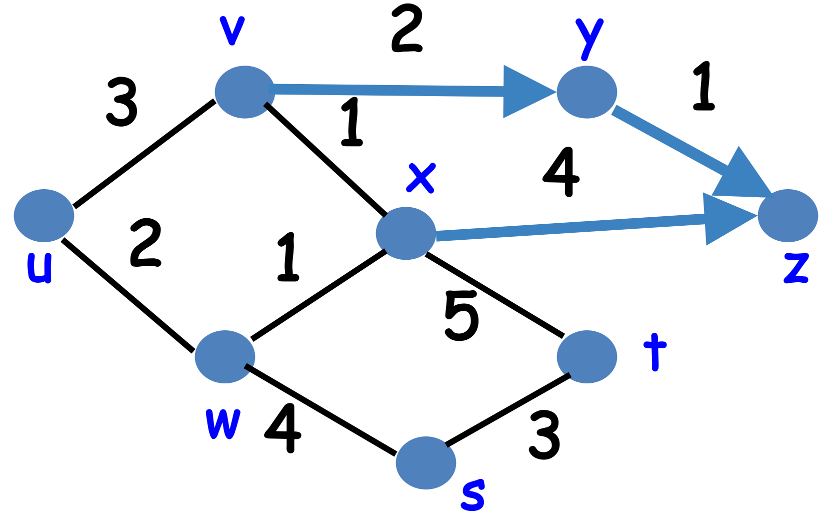


$$d_u(z) = \min\{\, c(u,v) + d_v(z),\; c(u,w) + d_w(z)\}$$

Used in RIP and EIGRP

Distance Vector Example



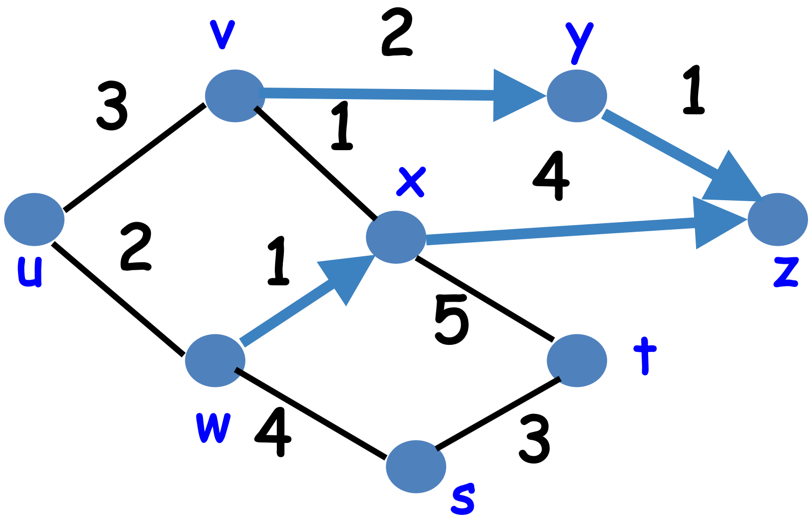


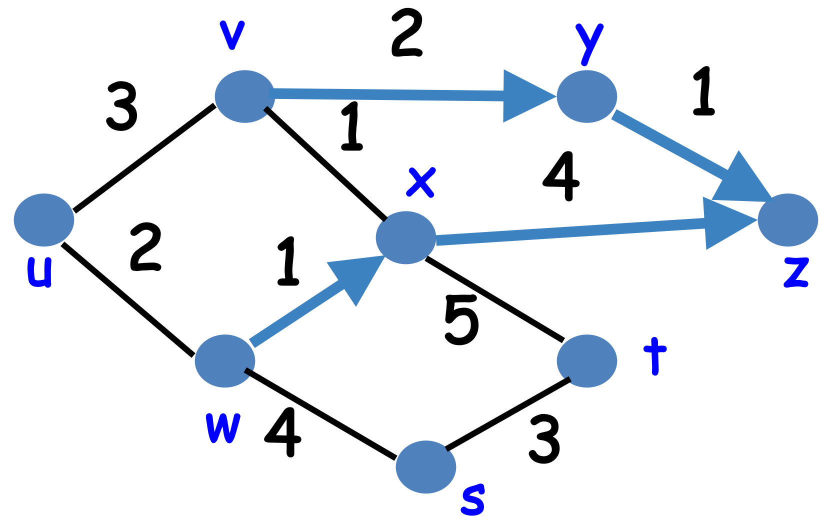
$d_y(z) = 1$

$d_x(z) = 4$

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

Distance Vector Example (Cont.)





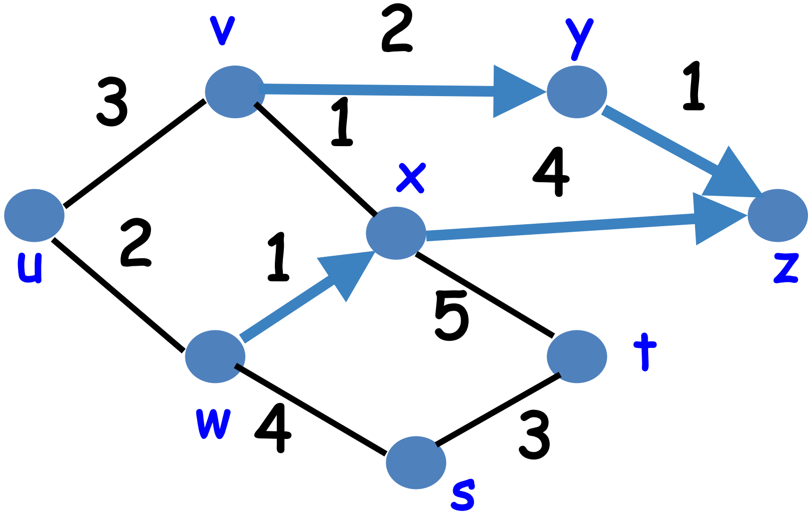
$$d_w(z) = \min\{\ 1+d_x(z),\ 4+d_s(z),\ 2+d_u(z)\ \}$$

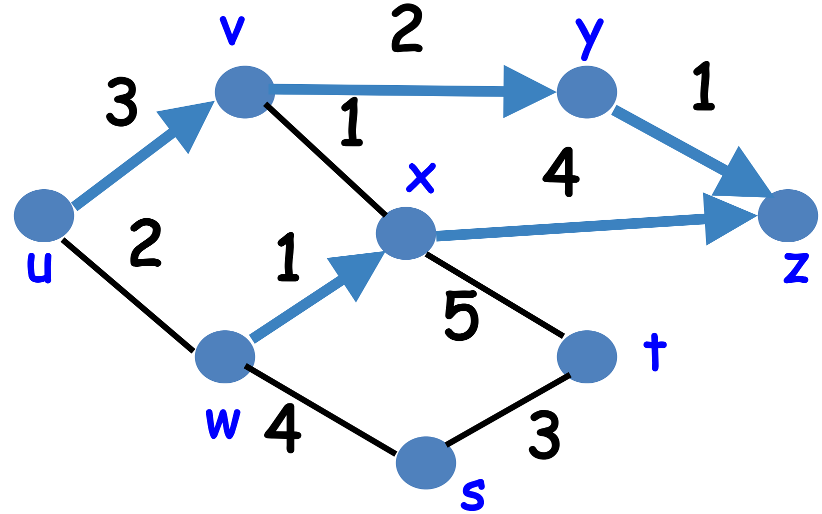
$= 5$

$d_u(z) \quad =$

(Y) 5 (M) 6 (C) 7

Distance Vector Example (Cont.)





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

$d_u(z) = \min\{\ 3+d_v(z),$
$\qquad 2+d_w(z)\ \}$
$= 6$

Next Up in 461

Midterm Exam

Released online March 7, 9:00 AM

Three hour time limit

Precepts this Thursday and Friday:

Midterm review