Network Layer

Routing Algorithm

Distance Vector Routing

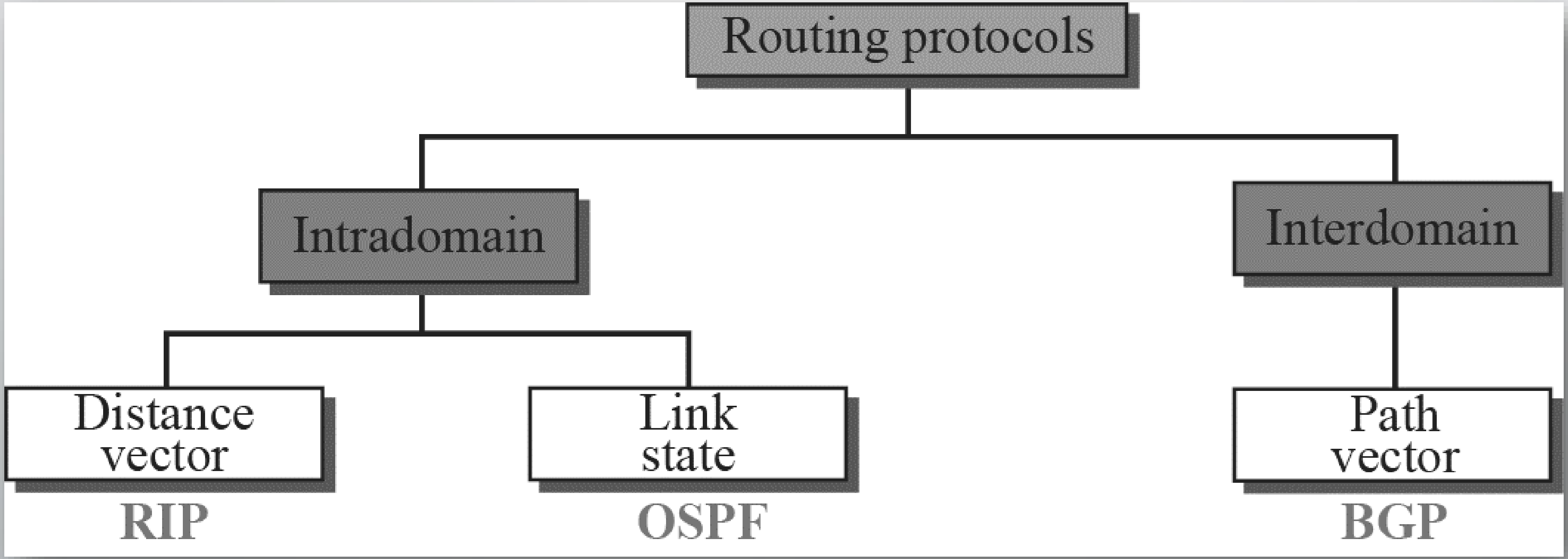
Lecture 13 | Part 1| CSE421 – Computer Networks

Department of Computer Science and Engineering
School of Data & Science

Objectives

- understand principles behind network layer services:
  - network layer service models
  - forwarding versus routing
  - how a router works
  - routing algorithms
    - **distance vector**
    - **link state**
    - hierarchical routing
  - broadcast, multicast

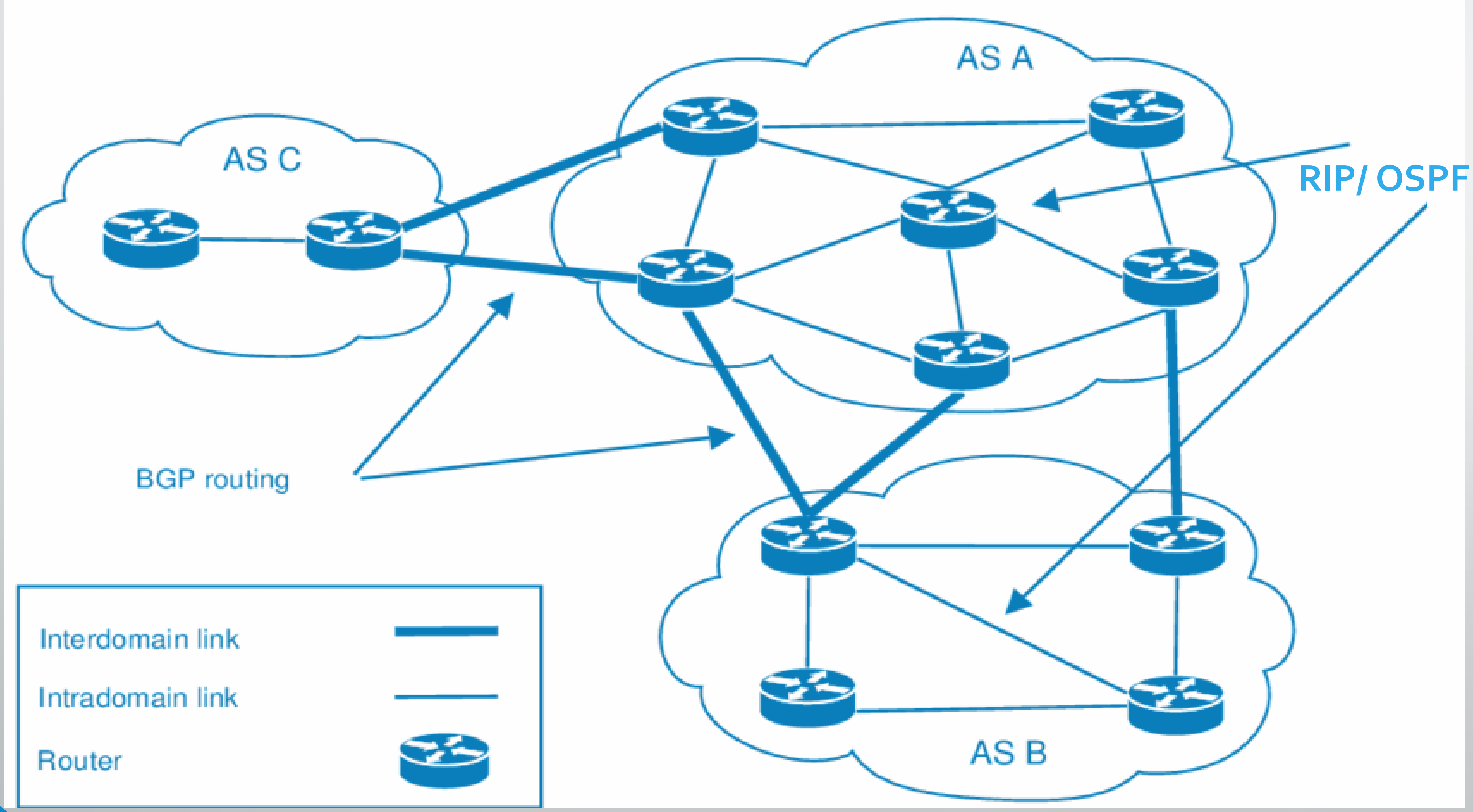
Popular Routing Protocols



Autonomous Systems

- 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



Routing Algorithms

- Given a set of routers and links connecting the routers.
- Routing algorithm finds a "good" path from the source to destination router.
- Good path = Least cost path

Routing Algorithm classification

Static or dynamic?

Static:

- routes change slowly over time
- Manually configured

Dynamic:

- routes change more quickly
  - in response to link cost changes

Routing Algorithm classification

Global or Decentralized

Global:

- all routers have complete topology and link cost info
- "link state" algorithms

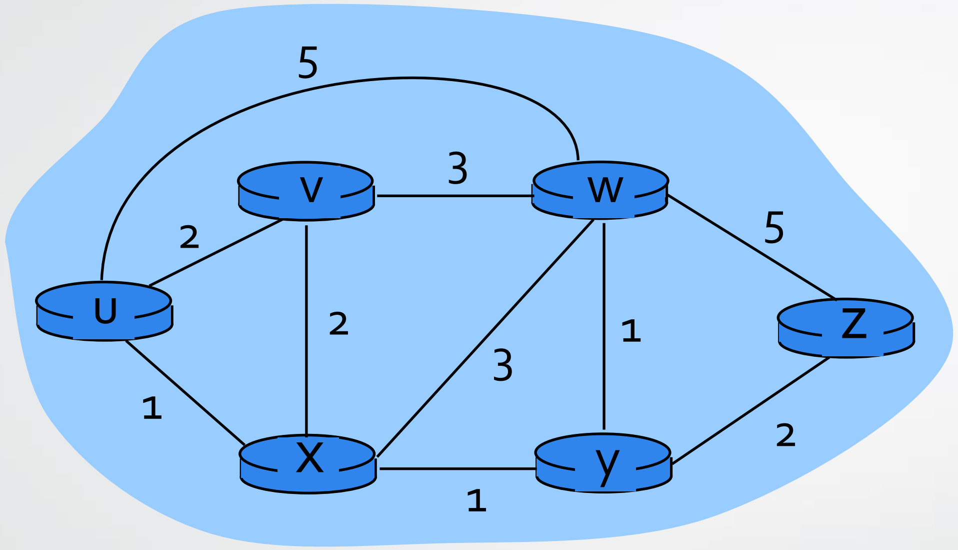
Decentralized:

- router knows physically-connected neighbors, link costs to neighbors
- iterative process of computation, exchange of info with neighbors
- "distance vector" algorithms

Routing on a Graph

- Essentially a graph theory problem
  - Network is a directed graph; routers are vertices
- Find “best” path between every pair of vertices
  - In the simplest case, best path is the shortest path

Graph abstraction

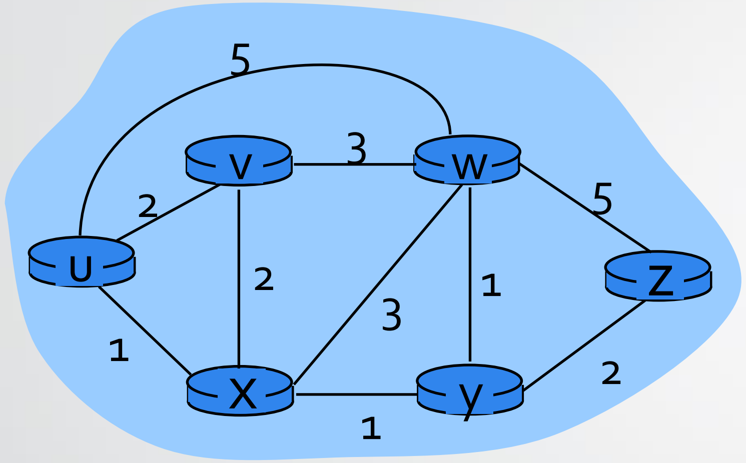


Graph: G = (N,E)

N = set of routers = { u, v, w, x, y, z }

E = set of links ={ (u,v), (u,x), (u,w), (v,x), (v,w), (x,w), (x,y), (w,y), (w,z), (y,z) }

Graph abstraction: costs



c(x,x′) = cost of link (x,x′)

- e.g., c(w,z) = 5

Question: What's the least-cost path between u and z ?

Routing algorithm: algorithm that finds least-cost path

Distance Vector Algorithm

Distance Vector Algorithm

Distributed:

- each node receives info from one or more of its directly connected neighbors
- Performs a calculations
- Distributes the results back to its neighbors

Iterative

- Process continues until no more info to exchange

Asynchronous:

- All nodes operate independently

Distance Vector Algorithm

Iterative, asynchronous: each local iteration caused by:

- Local link cost change
- Distance vector 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 message from neighbor)

↓

recompute estimates

↓

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

Distance Vector Algorithm

Bellman-Ford Equation Algorithm

- computes shortest paths from a single source vertex to all of the other vertices in a weighted digraph.

- Distributed route computation using only neighbor's info

Distance Vector Algorithm

Objective:

$d_x(y)$:= cost of least-cost path from x to y

Then

$$\boxed{d_x(y) = \min_v \{c(x,v) + d_v(y)\}}$$

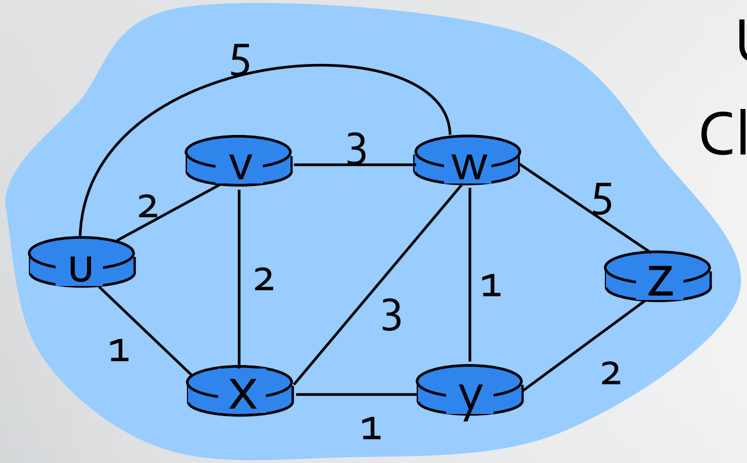
where min is taken over all neighbors v of x

Distance Vector Algorithm

With the Distance Vector Routing algorithm, the node x contains the following routing information:

- For each neighbor v, the cost $c(x,v)$ is the path cost from x to directly attached neighbor, v.
- The distance vector x, i.e., $D_x = [\ D_x(y) : y \text{ in } N\ ]$, containing its cost to all destinations, y, in N.
- The distance vector of each of its neighbors, i.e., $D_v = [\ D_v(y) : y \text{ in } N\ ]$ for each neighbor v of x.

Bellman-Ford example from u to z



U has 3 neighbors v, x and w

Clearly, $d_v(z) = 5$, $d_x(z) = 3$, $d_w(z) = 3$

B-F equation says:

$$
\begin{aligned}
d_u(z) &= \min \{ c(u,v) + d_v(z), \\
&\qquad c(u,x) + d_x(z), \\
&\qquad c(u,w) + d_w(z) \} \\
&= \min \{2 + 5, \\
&\qquad 1 + 3, \\
&\qquad 5 + 3\} = 4
\end{aligned}
$$

- Node that achieves minimum is the next hop in shortest path to a destination,
- To go to z from u, x in the next hop in the forwarding table

Distance vector algorithm

Basic idea:

- Each node periodically sends its own distance vector estimate to neighbors
- When a node x receives new DV estimate from neighbor;
- It updates its own DV using B-F equation:

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

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

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

node y table

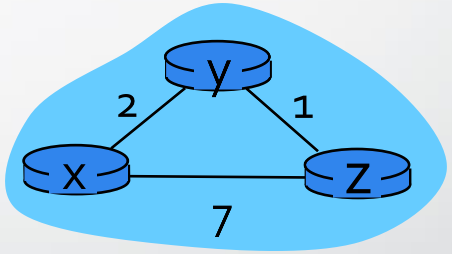
cost to

from	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

node z table

cost to

from	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0



time

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

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

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

node y table

cost to

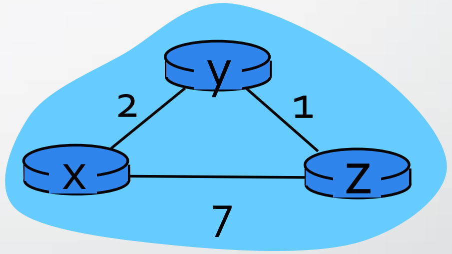
from	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

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

node z table

cost to

from	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0



time

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

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

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

node y table

cost to

from	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

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

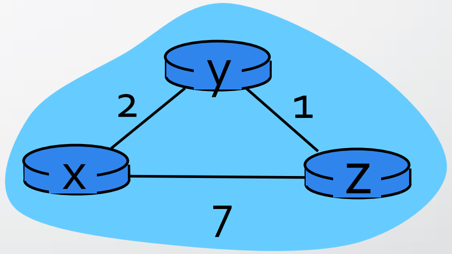
node z table

cost to

from	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0

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

time



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

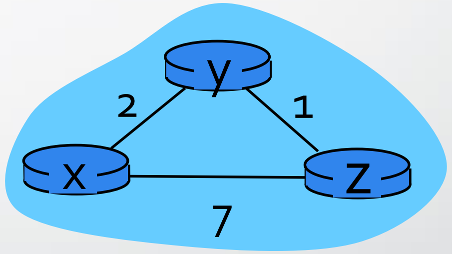
from	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0

cost to

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

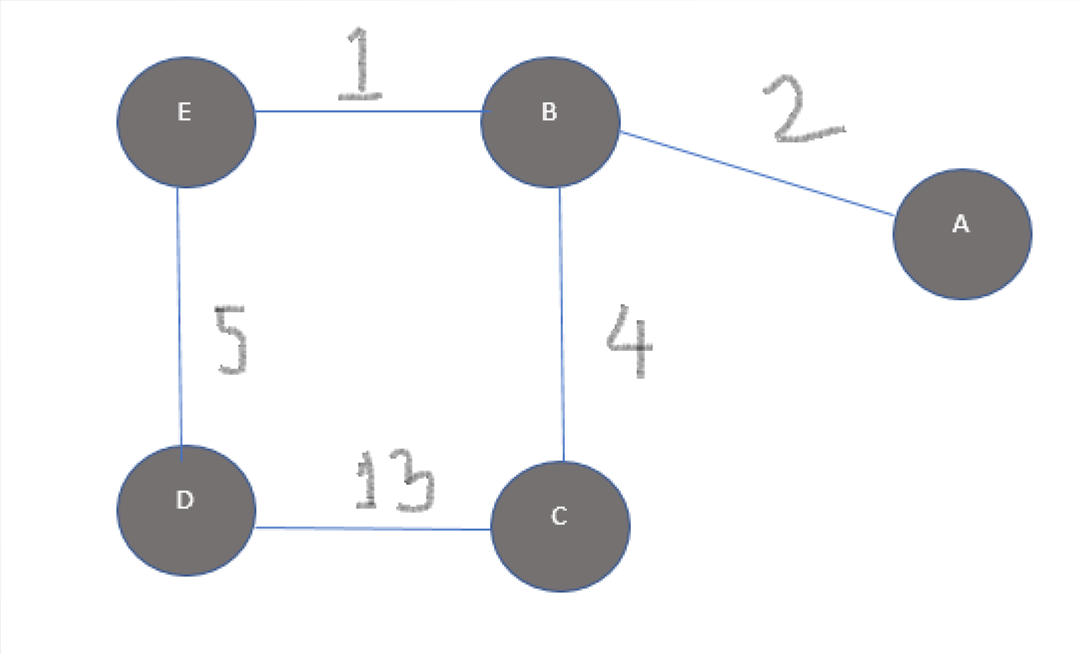
cost to

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



time

Example



- How many iterations are needed to make the final routing tables of each router?
- What will be the routing table of B and D after the 2^{nd} iteration?

BRAC UNIVERSITY
Inspiring Excellence

Network Layer

Routing Algorithm

Distance Vector Routing

Lecture 13 | Part 2| CSE421 – Computer Networks

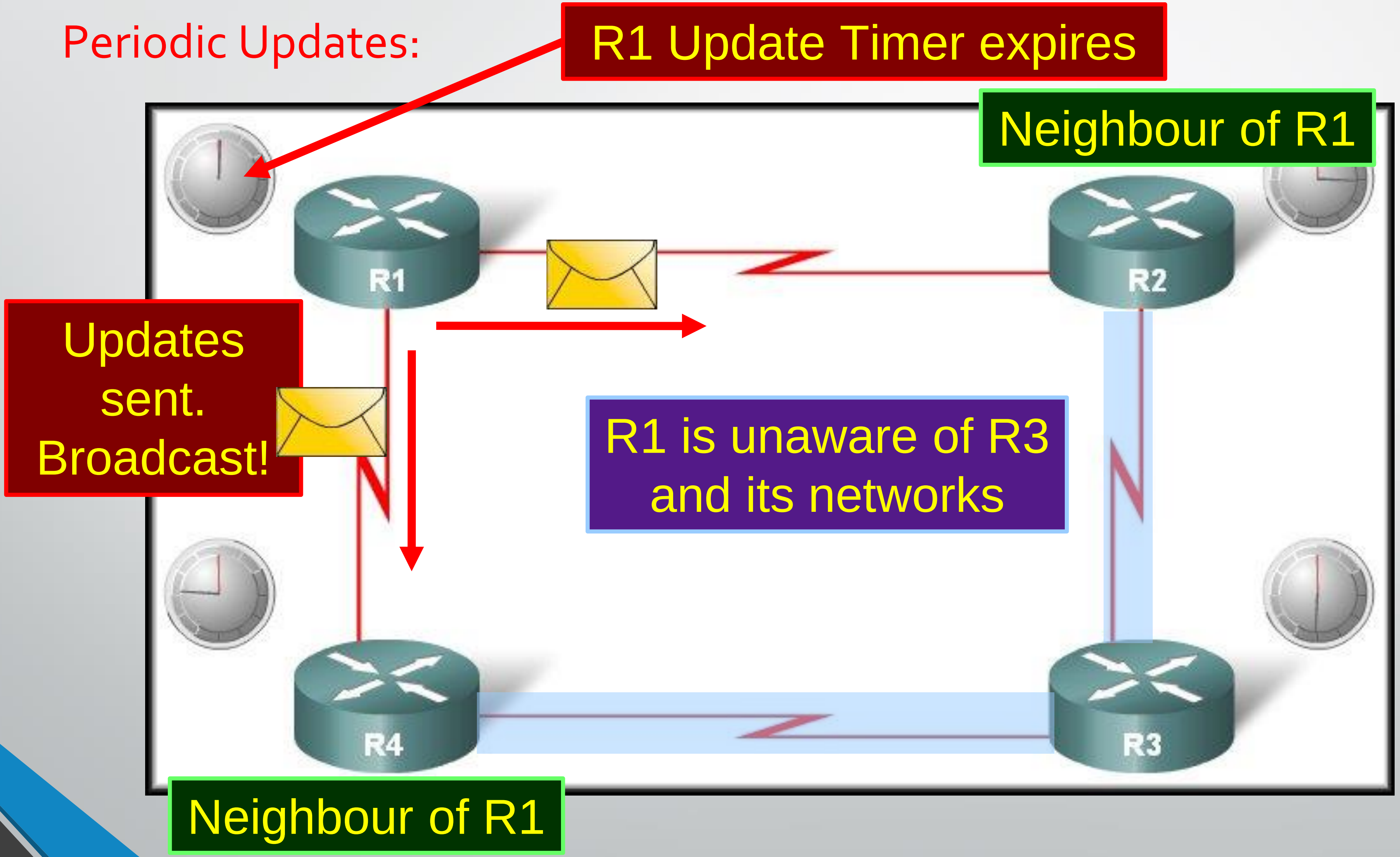
Department of Computer Science and Engineering
School of Data & Science

Operation of Distance Vector

- Periodic Updates:
  - Periodically broadcast the entire routing table to each of its neighbors (RIP – every 30 seconds).
    - Inefficient
  - Router is only aware of the:
    - Network addresses of its own interfaces.
    - Network addresses the neighbors running the same routing protocol.

Operation of Distance Vector

Periodic Updates:



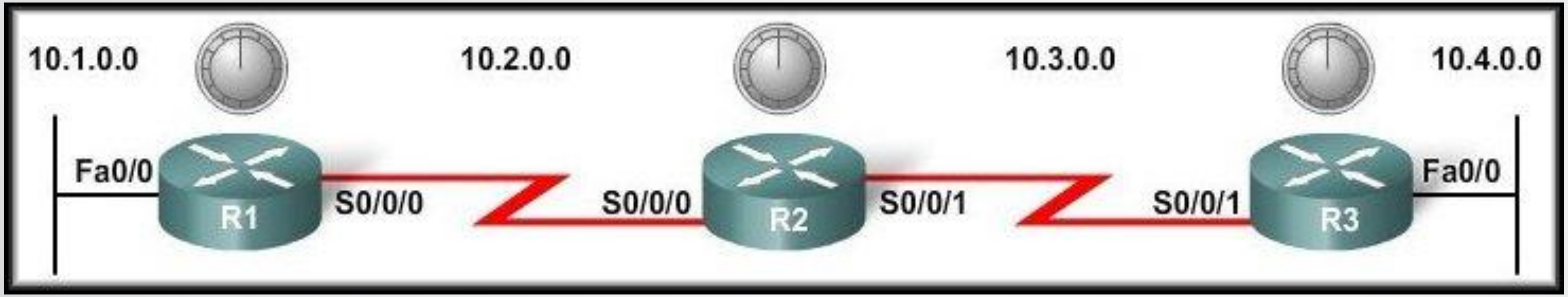
Distance Vector Routing Protocols

Network Discovery

Network	Interface	Hop
10.1.0.0	Fa0/0	0
10.2.0.0	S0/0/0	0
10.3.0.0	S0/0/0	1
10.4.0.0	S0/0/0	2

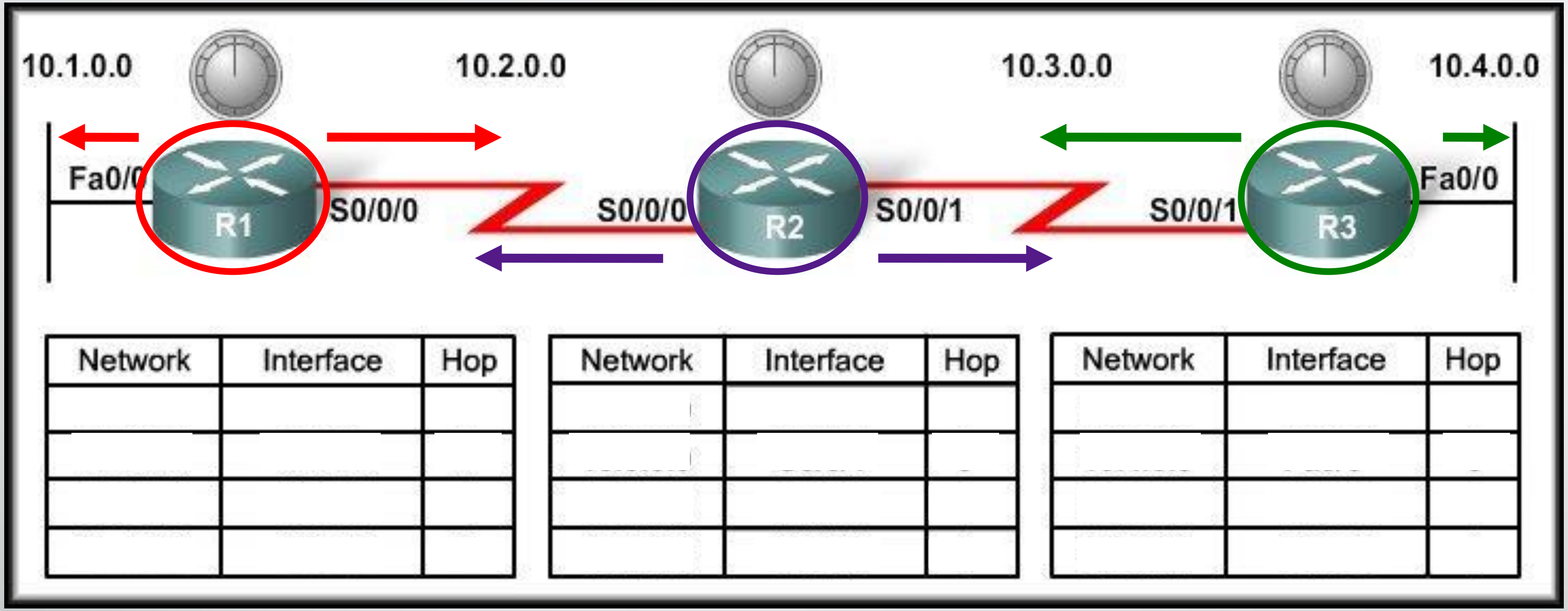
Network	Interface	Hop
10.2.0.0	S0/0/0	0
10.3.0.0	S0/0/1	0
10.1.0.0	S0/0/0	1
10.4.0.0	S0/0/1	1

Network	Interface	Hop
10.3.0.0	S0/0/1	0
10.4.0.0	Fa0/0	0
10.2.0.0	S0/0/1	1
10.1.0.0	S0/0/1	2



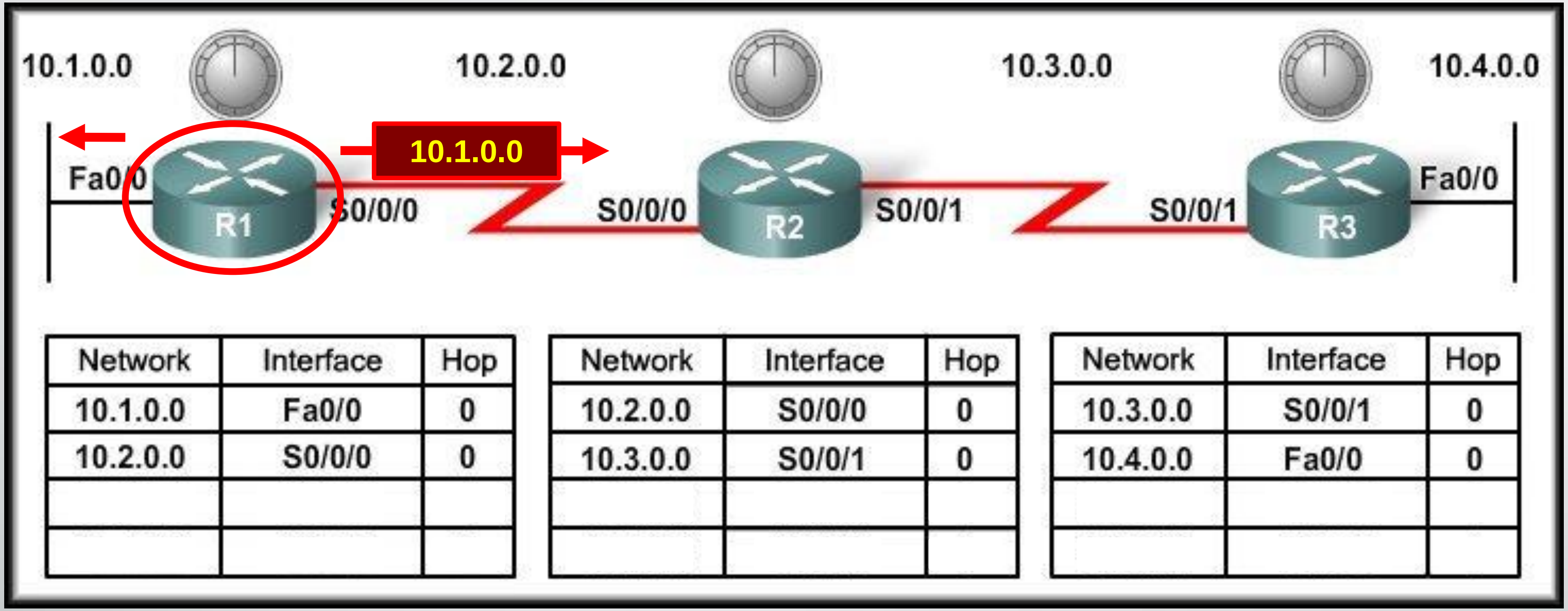
- Network Discovery:
    - Is part of the process of the routing protocol algorithm that enables routers to learn about remote networks for the first time.

Cold Start



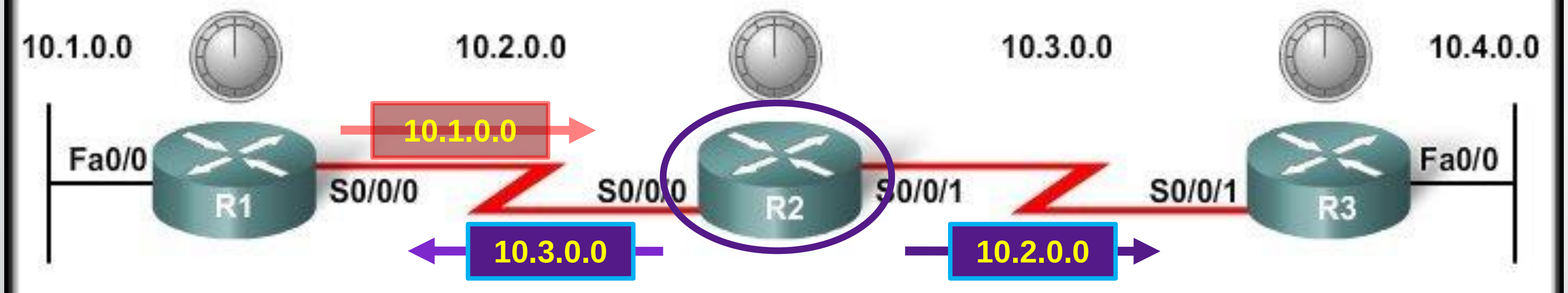
- When a router powers up:
  - Knows nothing about the network topology.
  - Knows only the information saved in NVRAM.
  - Sends updates about its known networks out all ports.

Initial Exchange of Routing Information



- Sends an update about network 10.1.0.0 out the Serial 0/0/0 interface with a metric of 1.
- Sends an update about network 10.2.0.0 out the Fa0/0 interface with a metric of 1.

Initial Exchange of Routing Information

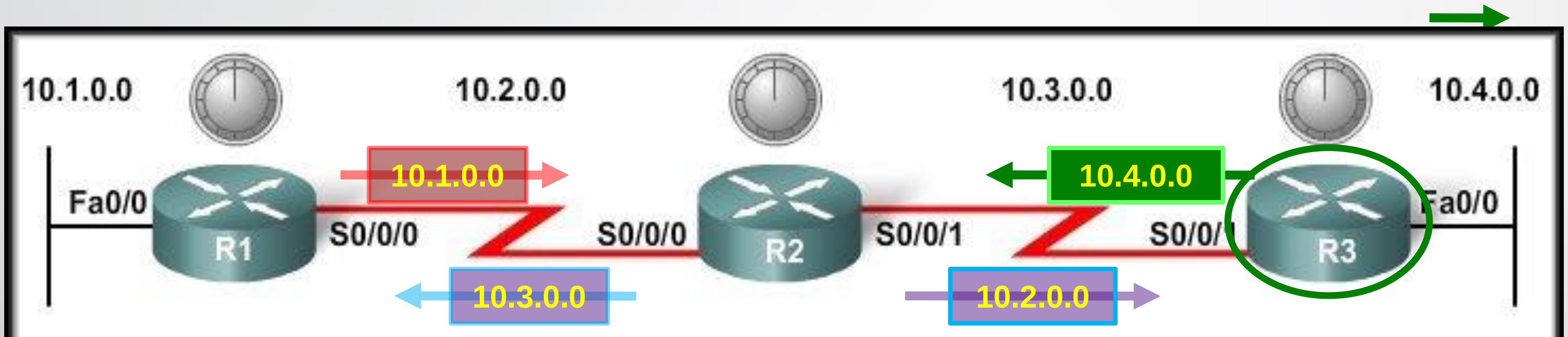


Network	Interface	Hop
10.1.0.0	Fa0/0	0
10.2.0.0	S0/0/0	0

Network	Interface	Hop
10.2.0.0	S0/0/0	0
10.3.0.0	S0/0/1	0

Network	Interface	Hop
10.3.0.0	S0/0/1	0
10.4.0.0	Fa0/0	0

Initial Exchange of Routing Information

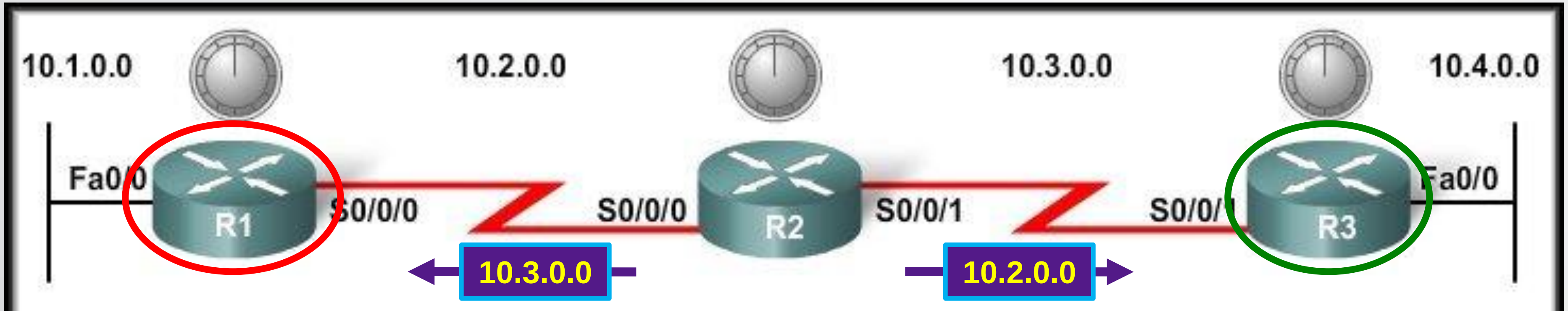


Network	Interface	Hop
10.1.0.0	Fa0/0	0
10.2.0.0	S0/0/0	0

Network	Interface	Hop
10.2.0.0	S0/0/0	0
10.3.0.0	S0/0/1	0

Network	Interface	Hop
10.3.0.0	S0/0/1	0
10.4.0.0	Fa0/0	0

Initial Exchange of Routing Information



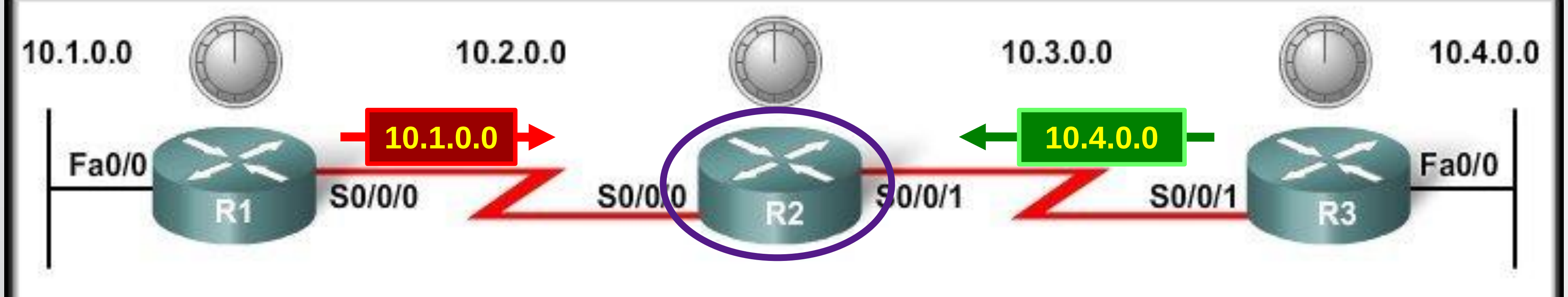
Network	Interface	Hop
10.1.0.0	Fa0/0	0
10.2.0.0	S0/0/0	0

Network	Interface	Hop
10.2.0.0	S0/0/0	0
10.3.0.0	S0/0/1	0

Network	Interface	Hop
10.3.0.0	S0/0/1	0
10.4.0.0	Fa0/0	0

- R1 Receives the update from R2 about network 10.3.0.0 and adds it to its routing table.
- R3 Receives the update from R2 about network 10.2.0.0 and adds it to its routing table.

Initial Exchange of Routing Information



Network	Interface	Hop
10.1.0.0	Fa0/0	0
10.2.0.0	S0/0/0	0
10.3.0.0	S0/0/0	1

Network	Interface	Hop
10.2.0.0	S0/0/0	0
10.3.0.0	S0/0/1	0

Network	Interface	Hop
10.3.0.0	S0/0/1	0
10.4.0.0	Fa0/0	0
10.2.0.0	S0/0/1	1

- R2 Receives the update from R1 about network 10.1.0.0 and adds it to its routing table.
- R2 Receives the update from R3 about network 10.4.0.0 and adds it to its routing table.

Next Exchange of Routing Information

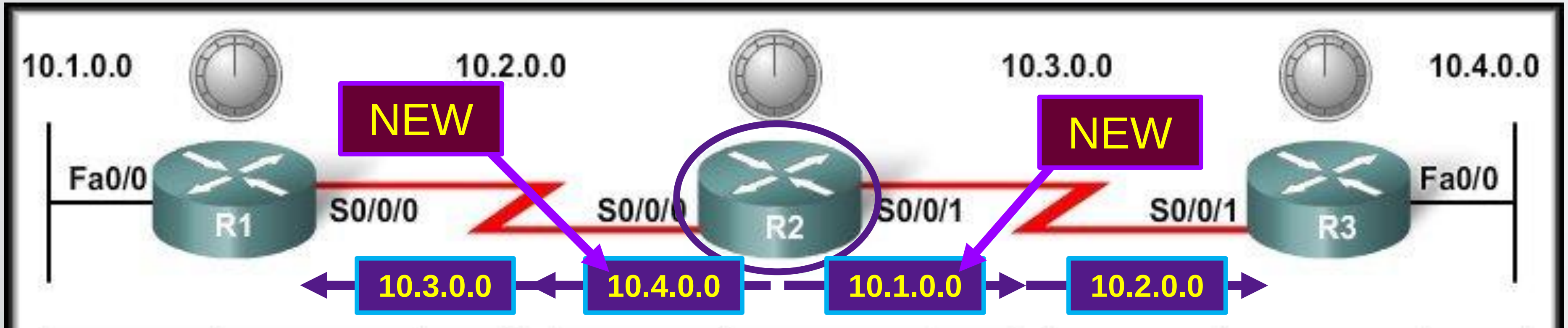
Network	Interface	Hop
10.1.0.0	Fa0/0	0
10.2.0.0	S0/0/0	0
10.3.0.0	S0/0/0	1

Network	Interface	Hop
10.2.0.0	S0/0/0	0
10.3.0.0	S0/0/1	0
10.1.0.0	S0/0/0	1
10.4.0.0	S0/0/1	1

Network	Interface	Hop
10.3.0.0	S0/0/1	0
10.4.0.0	Fa0/0	0
10.2.0.0	S0/0/1	1

- Sends an update about network 10.1.0.0 out the S0/0/0 interface with a metric of 1 - AGAIN!
- When R2 receives the update, there is no change in information so the update is ignored.

Next Exchange of Routing Information



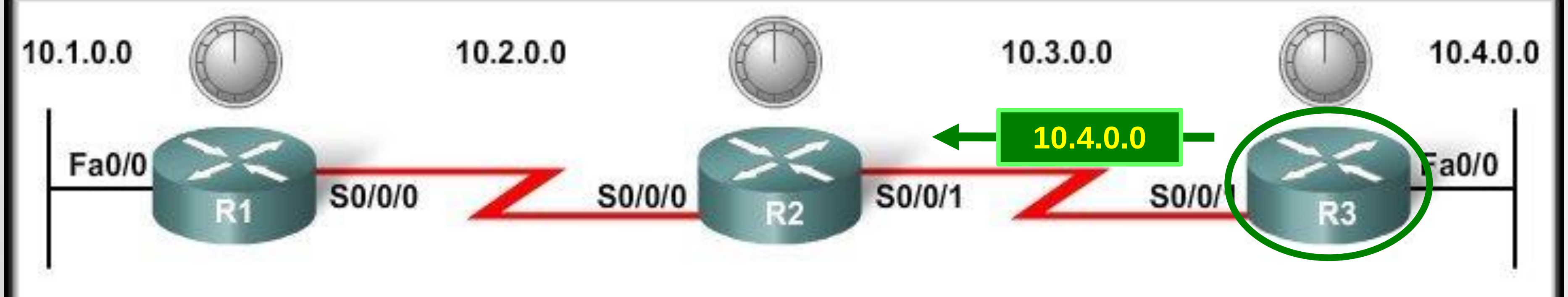
Network	Interface	Hop
10.1.0.0	Fa0/0	0
10.2.0.0	S0/0/0	0
10.3.0.0	S0/0/0	1

Network	Interface	Hop
10.2.0.0	S0/0/0	0
10.3.0.0	S0/0/1	0
10.1.0.0	S0/0/0	1
10.4.0.0	S0/0/1	1

Network	Interface	Hop
10.3.0.0	S0/0/1	0
10.4.0.0	Fa0/0	0
10.2.0.0	S0/0/1	1

- Sends an update about networks 10.3.0.0 with a metric of 1 and 10.4.0.0 with a metric of 2 out the Serial 0/0/0 interface.
- Similarly sends updates about networks 10.1.0.0 with a metric of 2 and 10.2.0.0 with a metric of 1 out the Serial 0/0/1 interface.

Next Exchange of Routing Information



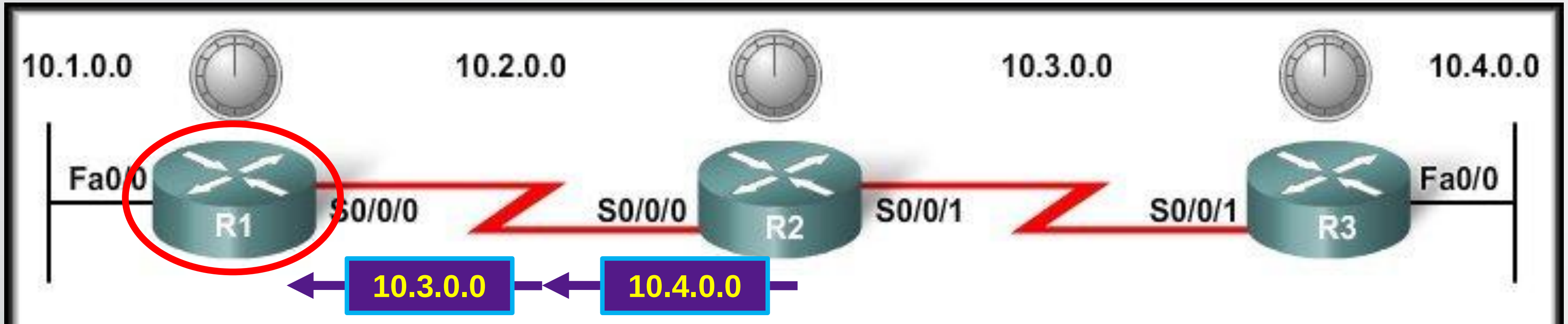
Network	Interface	Hop
10.1.0.0	Fa0/0	0
10.2.0.0	S0/0/0	0
10.3.0.0	S0/0/0	1

Network	Interface	Hop
10.2.0.0	S0/0/0	0
10.3.0.0	S0/0/1	0
10.1.0.0	S0/0/0	1
10.4.0.0	S0/0/1	1

Network	Interface	Hop
10.3.0.0	S0/0/1	0
10.4.0.0	Fa0/0	0
10.2.0.0	S0/0/1	1

- Sends an update about network 10.4.0.0 out the S0/0/0 interface with a metric of 1 - AGAIN!
- When R2 receives the update, there is no change in information so the update is ignored.

Next Exchange of Routing Information



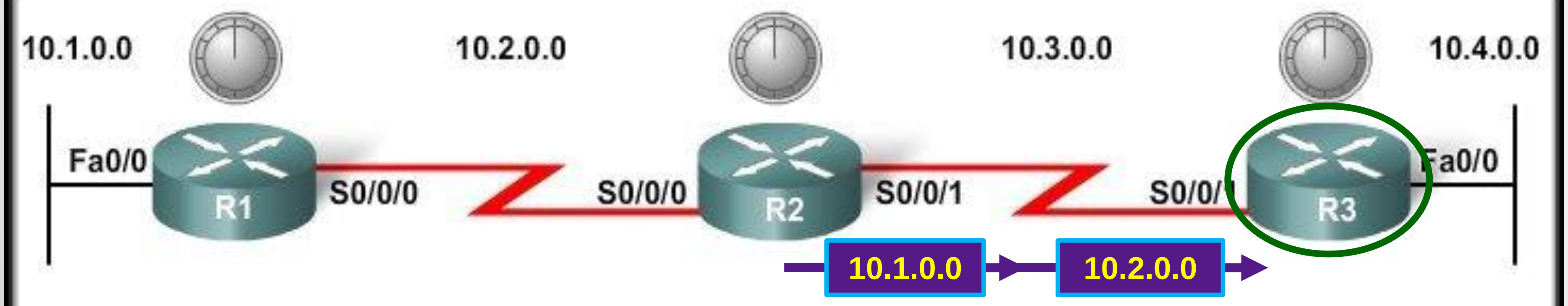
Network	Interface	Hop
10.1.0.0	Fa0/0	0
10.2.0.0	S0/0/0	0
10.3.0.0	S0/0/0	1

Network	Interface	Hop
10.2.0.0	S0/0/0	0
10.3.0.0	S0/0/1	0
10.1.0.0	S0/0/0	1
10.4.0.0	S0/0/1	1

Network	Interface	Hop
10.3.0.0	S0/0/1	0
10.4.0.0	Fa0/0	0
10.2.0.0	S0/0/1	1

- R1 receives an update from R2 about network 10.3.0.0 and there is no change – update ignored.
- R1 receives an update from R2 about network 10.4.0.0 (new) and adds it to its routing table.

Next Exchange of Routing Information



Network	Interface	Hop
10.1.0.0	Fa0/0	0
10.2.0.0	S0/0/0	0
10.3.0.0	S0/0/0	1
10.4.0.0	S0/0/0	2

Network	Interface	Hop
10.2.0.0	S0/0/0	0
10.3.0.0	S0/0/1	0
10.1.0.0	S0/0/0	1
10.4.0.0	S0/0/1	1

Network	Interface	Hop
10.3.0.0	S0/0/1	0
10.4.0.0	Fa0/0	0
10.2.0.0	S0/0/1	1

- R3 receives an update from R2 about network 10.2.0.0 and there is no change – update ignored.
- R3 receives an update from R2 about network 10.1.0.0 (new) and adds it to its routing table.

Next Exchange of Routing Information

Network	Interface	Hop
10.1.0.0	Fa0/0	0
10.2.0.0	S0/0/0	0
10.3.0.0	S0/0/0	1
10.4.0.0	S0/0/0	2

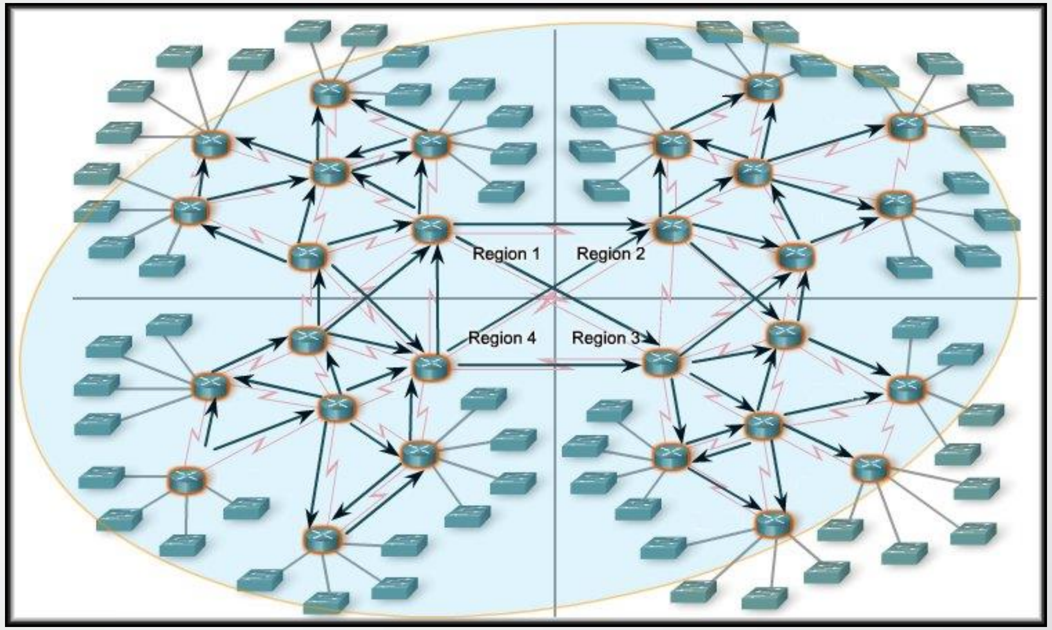
Network	Interface	Hop
10.2.0.0	S0/0/0	0
10.3.0.0	S0/0/1	0
10.1.0.0	S0/0/0	1
10.4.0.0	S0/0/1	1

Network	Interface	Hop
10.3.0.0	S0/0/1	0
10.4.0.0	Fa0/0	0
10.2.0.0	S0/0/1	1
10.1.0.0	S0/0/1	2

- The network has CONVERGED!
  - All routers now know about all of the networks attached to all of their neighbouring routers.

Convergence

- The amount of time it takes for a network to converge is directly proportional to the size of that network.



- Routing protocols are compared based on how fast they can propagate this information - their speed to convergence.

- A network is not completely operable until it has converged.
  - Network administrators prefer routing protocols with shorter convergence times.

THE END