Connecting LANs

Required reading:
Forouzan 17.1 to 17.1
Garcia 6.11 (intro + 6.11.1)

CSE 3213, Fall 20105
Instructor: N. Vlajic

Connecting Devices

Why Connecting Devices in LANs?

(1) **LANs do not normally operate in isolation – they are connected to one another or to the Internet to enable sharing of CPUs, data-bases, programs, etc.**

(2) **as # of devices in a single LAN grows, MAC and error-&-flow control protocols become less effective**

- **way to avoid bottlenecks is to divide LAN into multiple LANs, thus reducing # of devices per LAN**

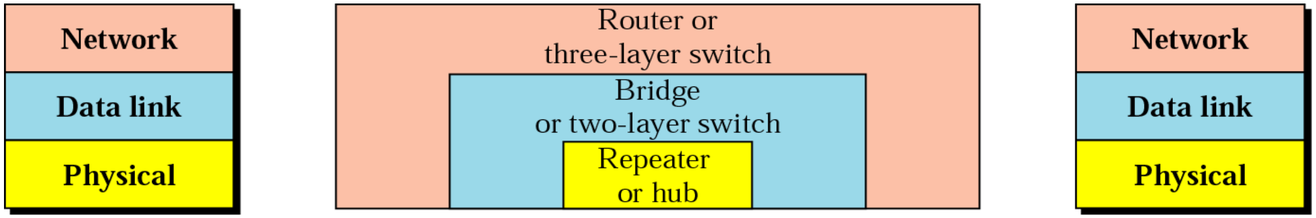
Types of Connecting Devices

- **connecting devices can operate in different layers of the Internet model**

(1) **repeaters and hubs operate in the <u>first layer</u>**

(2) **bridges / link layer switchs operate in the <u>first two layers</u>**

(3) **routers operate in the <u>first three layers</u>**

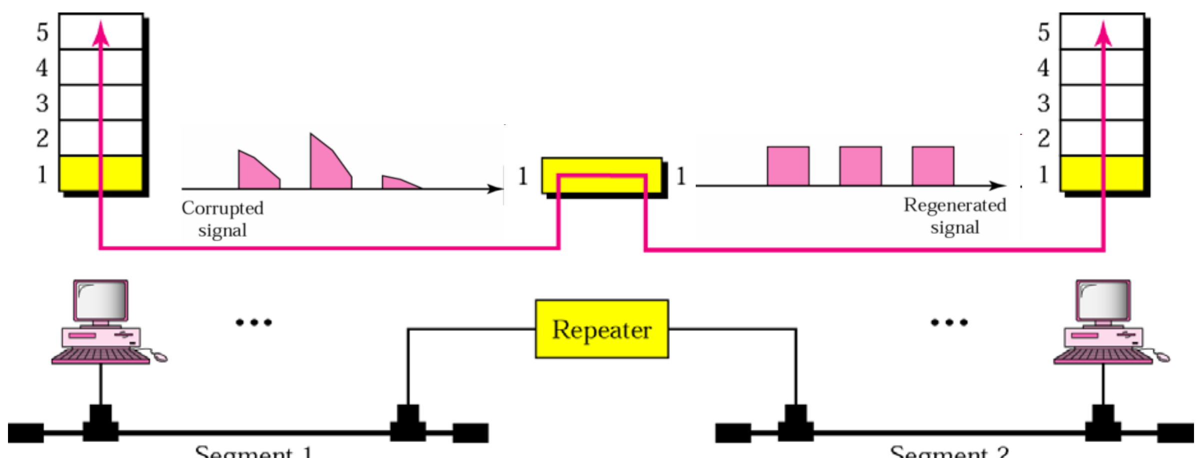


Connecting Devices: Repeaters

Repeater – connecting device that operates only in the physical layer:

(1) receive signal on one end →
(2) regenerate original bit patterns →
(3) send refreshed signal on the other end

- connects only segments of the <u>same</u> LAN
  - segments must run the same protocol
- has no filtering capability
  - every frame received will be <u>regenerated</u> (not amplified) and forwarded
- <u>location of a repeater is crucial</u> – repeater must be placed so that a signal reaches it before noise changes the meaning of any of its bits

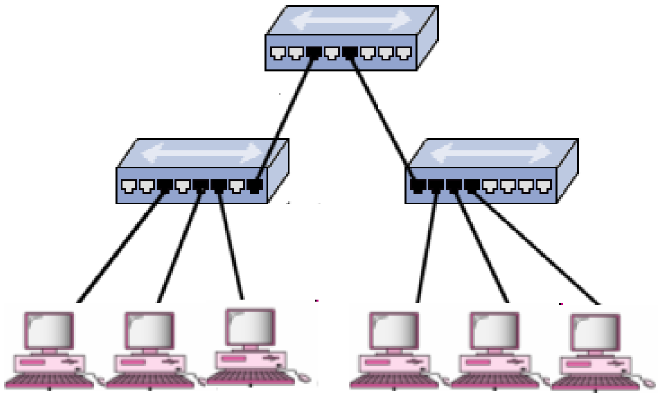


Connecting Devices: Hubs

Hub – multiport repeater !!!

(1) receive signal on one end →
(2) regenerate original bit patterns →
(3) send refreshed signal over all other ports

- **passive hubs**: simply send signal to all connected hosts, without amplifying it
- **active hubs**: are connected to electric power source, and are used to refresh the signal sent to all ports



Repeaters and hubs primarily extend the physical reach of a network, but at the same time they can create problems.

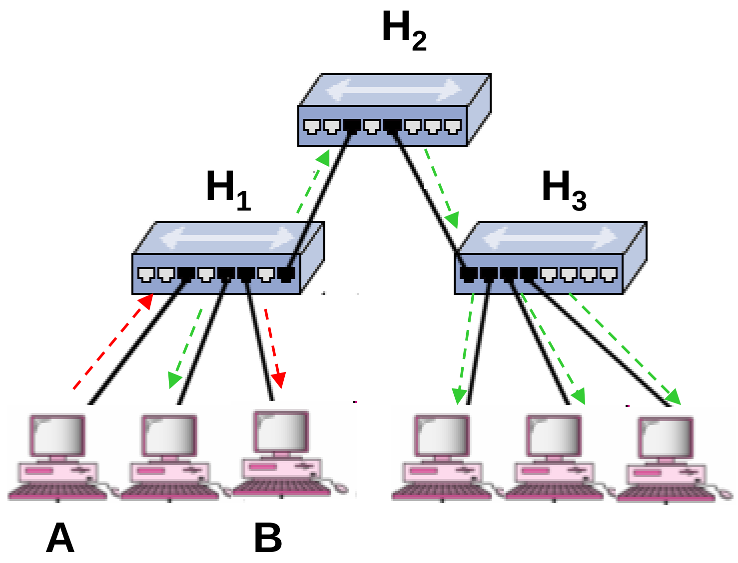
more devices access the medium ⇒ more traffic ⇒ degraded LAN performance

Connecting Devices: Hubs (cont.)

Example [unnecessary frame flooding in LANs with hubs]

If node A sends a frame to node B, hubs H_1, H_2, and H_3 forward the frame to all possible location.

H2, and H3 do not have built-in logic to know that A and B are on the same LAN and connected to the same hub, and that repeating the frame is pointless.

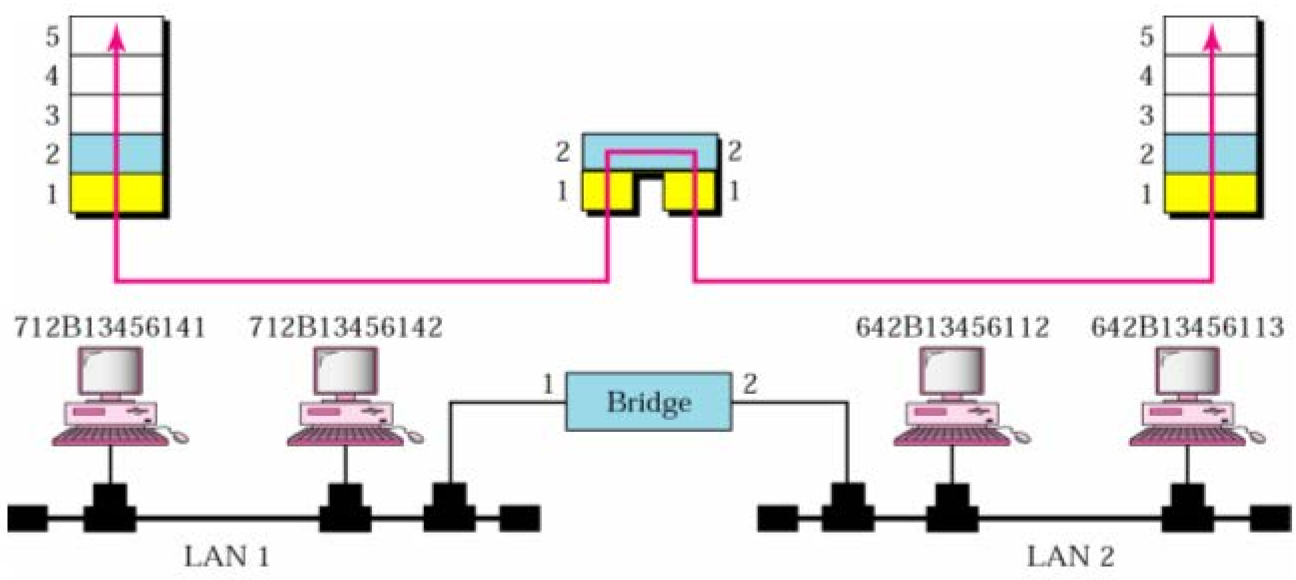


The problem of frame flooding can be resolved by filtering out (not forwarding) frames that have both 'source' and 'destination' address on the same LAN.

Connecting Devices: Bridges

Bridge (Layer 2 Switch) – connecting device that <u>operates in both physical & data link layer</u>

- as a physical-layer device, bridge regenerates the signal it receives
- as a data link layer device, bridge checks physical / MAC addresses (both source and destination) in frames
  - if frame sent in LAN 1 is destined for a device on LAN 2 – receive and forward the frame; otherwise ignore the frame
- to be able to properly forward / filter frames, bridge must build / learn a 'forwarding table', aka 'forwarding database'

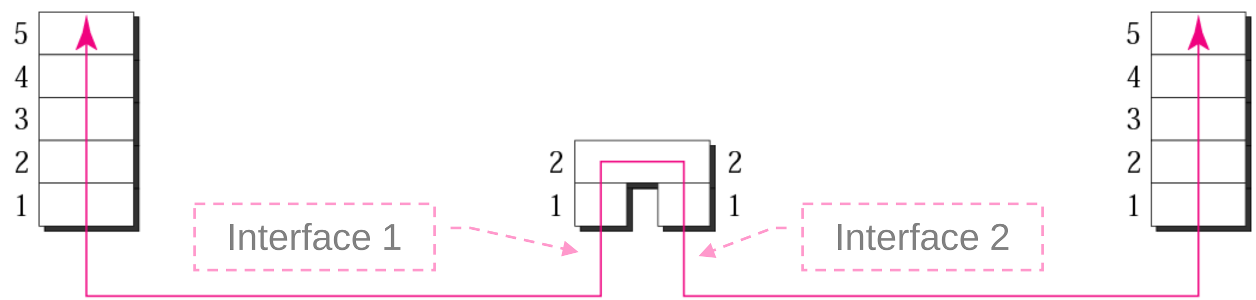


Connecting Devices: Bridges (cont.)

Example [filtering with bridges]

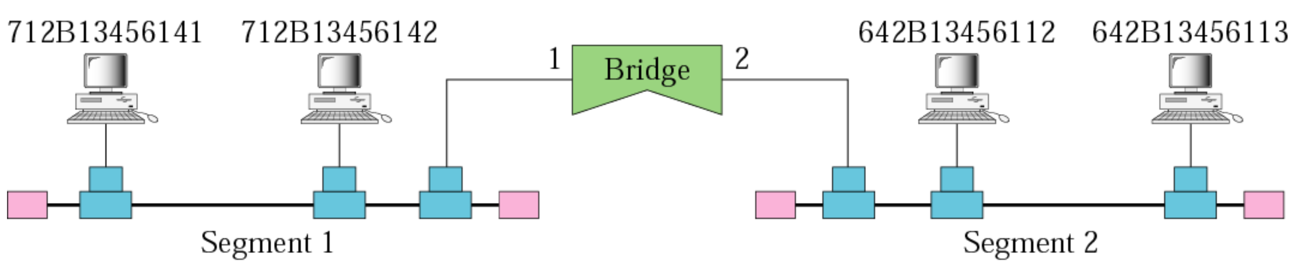
Assume the bridge has a table that maps addresses to ports, i.e. maps the address of each host to the bridge port # through which frames from the given host arrive.

If a frame destined for station 712B1345142 arrives at port 1, the bridge consults its table to find the departing port. As frames for 712B1345142 leave through port 1, there is no need for frame forwarding.



Address	Interface
712B13456141	1
712B13456142	1
642B13456112	2
642B13456113	2

Bridge table



Connecting Devices: Bridges (cont.)

Bridge Learning

- earliest bridges used static forwarding tables
  - system administrators would manually enter each table entry
  - simple but impractical process – whenever a new station was added or removed, the table had to be modified manually
- dynamic / transparent forwarding tables – bridge learns the location of all stations gradually, as it operates, and builds forwarding table automatically
  - bridge inspects both source and destination address of each received frame

learning phase

(a) source address is compared with each entry in table
- if a match is not found, add source address together with port number on which frame was received to table
- if a match is found, do nothing

forwarding phase

(b) destination address is compared with each entry in the table
- if a match is not found, flood frame on all ports except the one on which the frame was received
- if a match is found and port is one on which frame was received, do nothing; otherwise, forward frame to port indicated in table

Connecting Devices: Bridges (cont.)

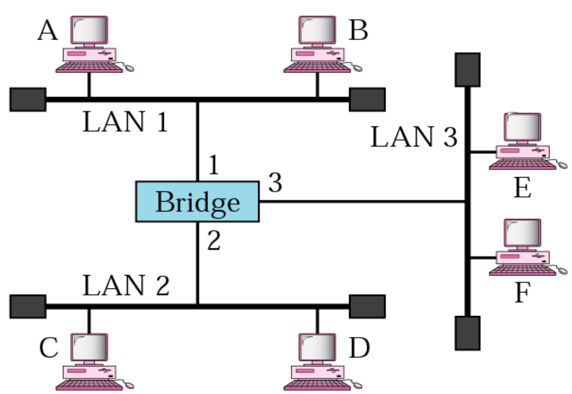
Example [bridge learning]

(a) **When station A sends a frame to station D, the bridge does not have any entry for either A or D. Hence,**

- **frame is flooded on ports 2 and 3**
- **by looking at the source address, the bridge learns that station A must be located on the LAN connected to port 1 (LAN 1) $\Rightarrow$ frames destined for A must be sent out through port 1.**

(b) **When station E sends a frame to station A, the bridge has an entry for A. Hence**

- **the frame is forwarded only to port 1**
- **the source address of the frame is added as a second entry to the table**



Address	Port

a. Original

Address	Port
A	1

b. After A sends a frame to D

Address	Port
A	1
E	3

c. After E sends a frame to A

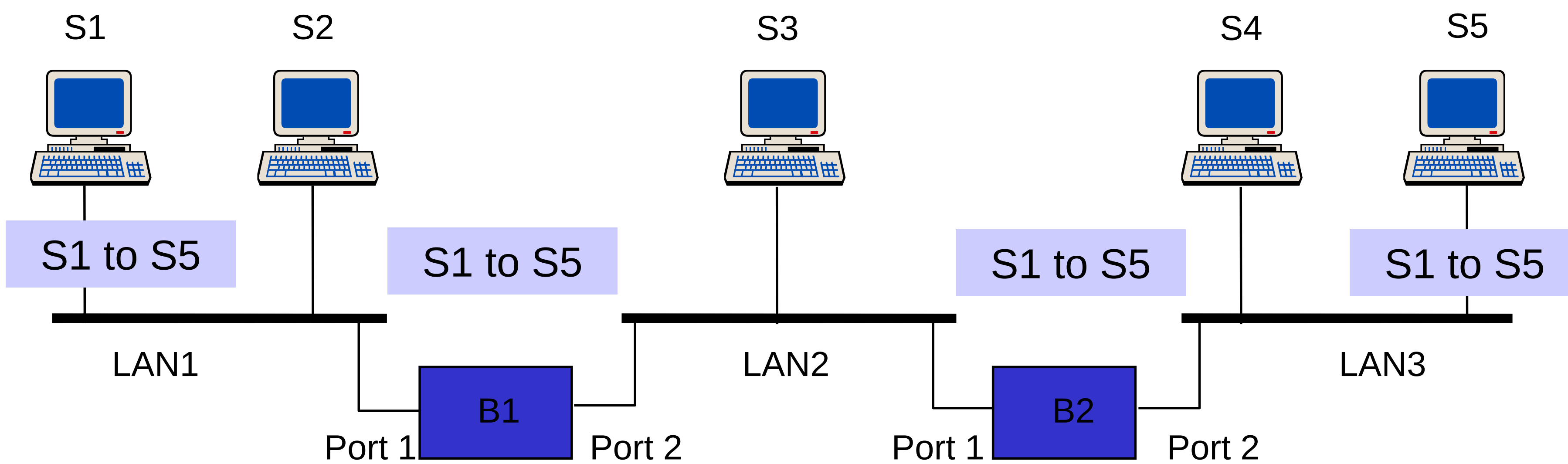
Address	Port
A	1
E	3
B	1

d. After B sends a frame to C

Connecting Devices: Bridges (cont.)

Example **[bridge learning]**

S_1 sends a frame to S_5.

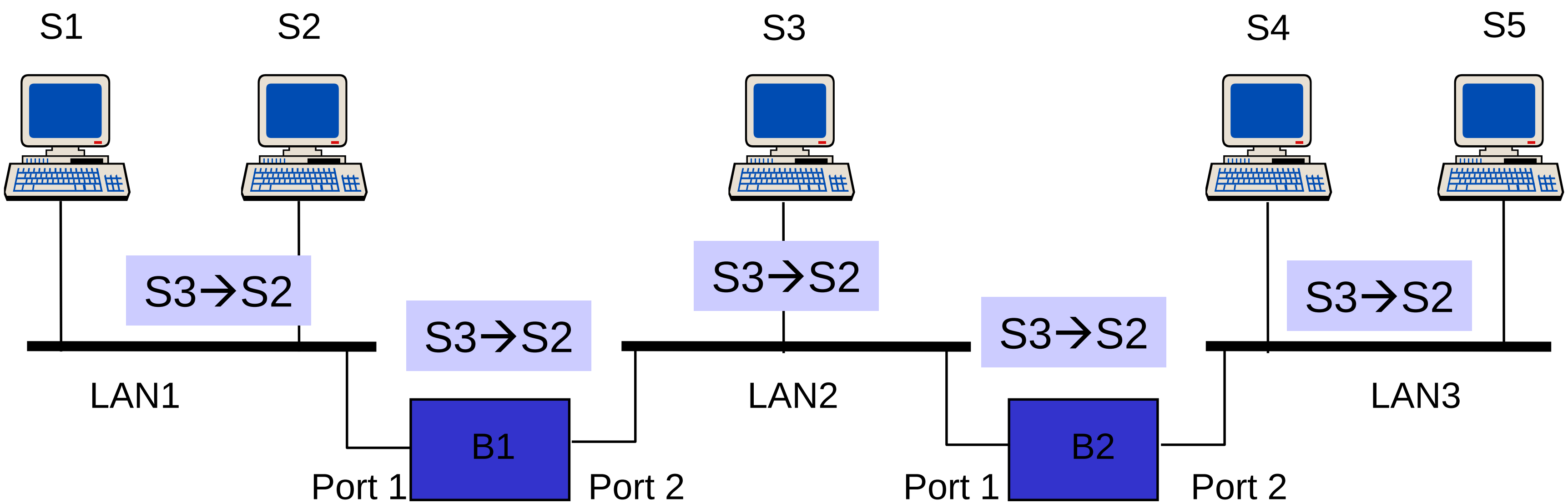


Address	Port
S1	1

Address	Port
S1	1

Connecting Devices: Bridges (cont.)

S_3 sends a frame to S_2.

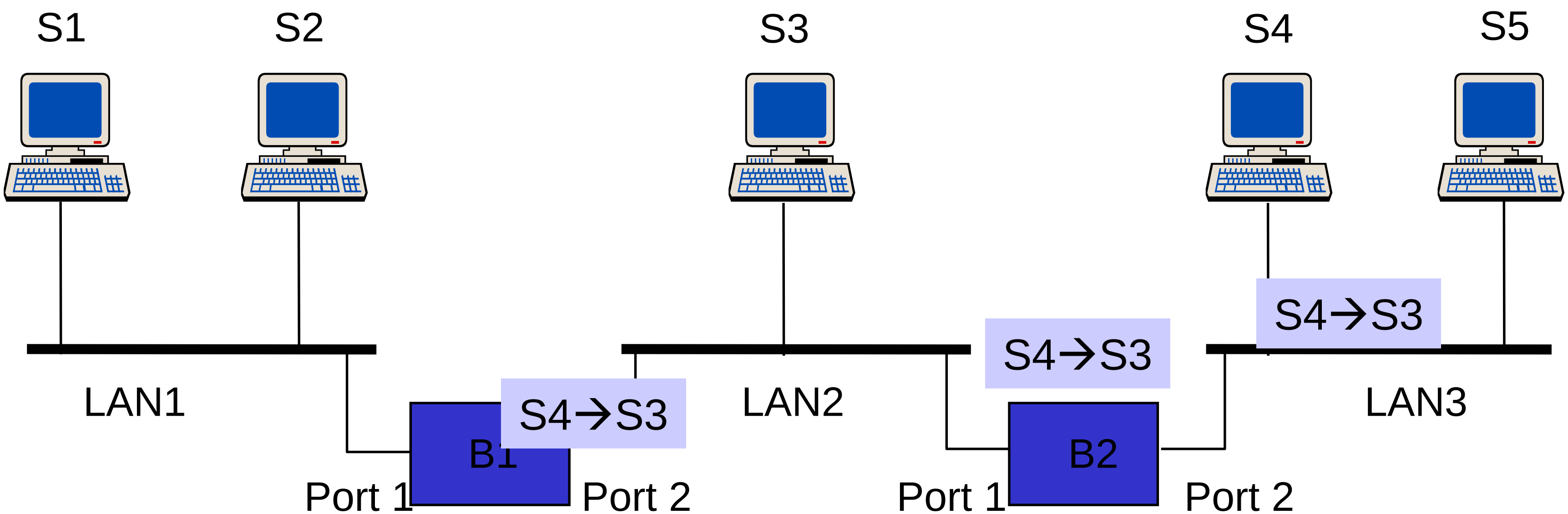


Address	Port
S1	1
S3	2

Address	Port
S1	1
S3	1

Connecting Devices: Bridges (cont.)

S_4 sends a frame to S_3.

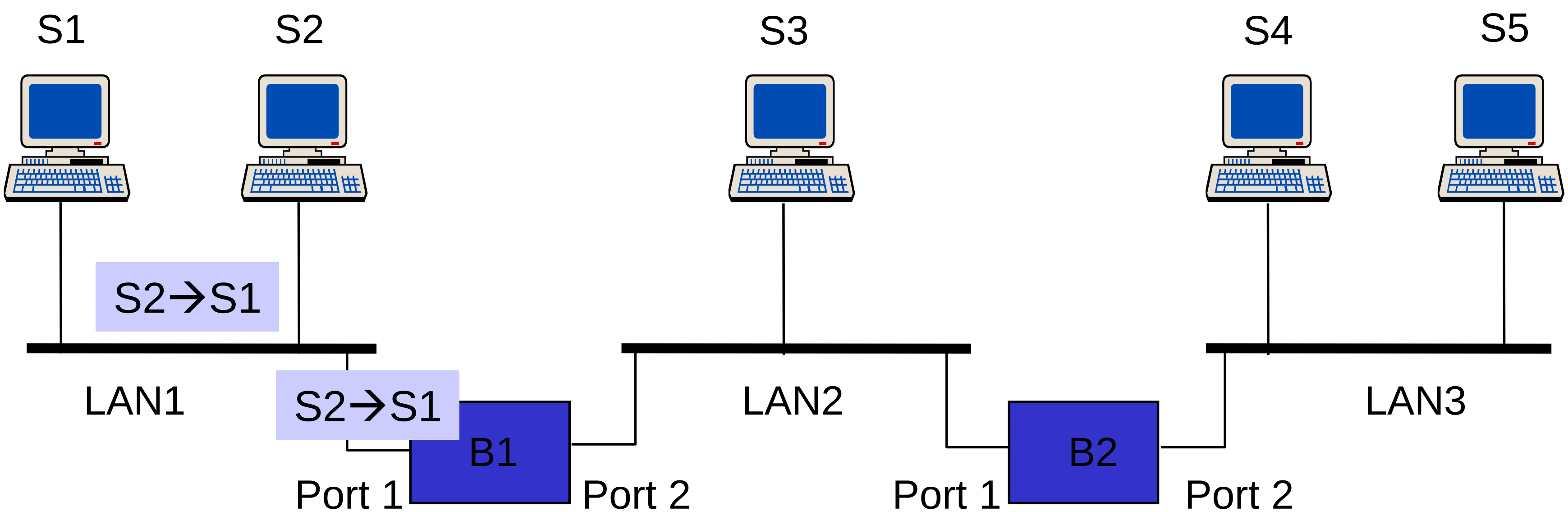


Address	Port
S1	1
S3	2
S4	2

Address	Port
S1	1
S3	1
S4	2

Connecting Devices: Bridges (cont.)

S_2 sends a frame to S_1.



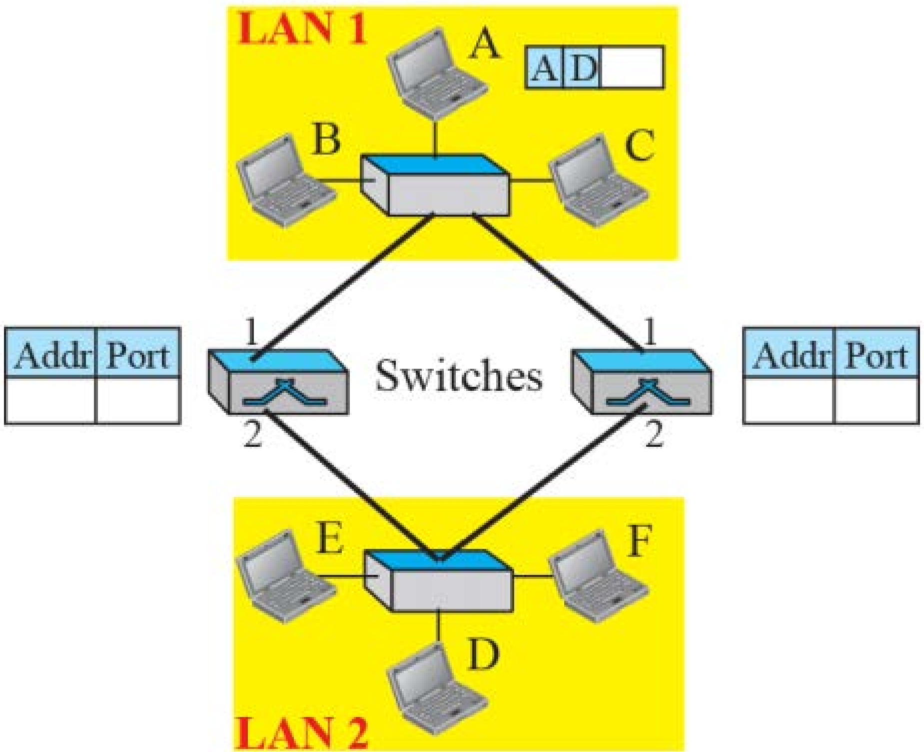
Address	Port
S1	1
S3	2
S4	2
S2	1

Address	Port
S1	1
S3	1
S4	2

Connecting Devices: Bridges (cont.)

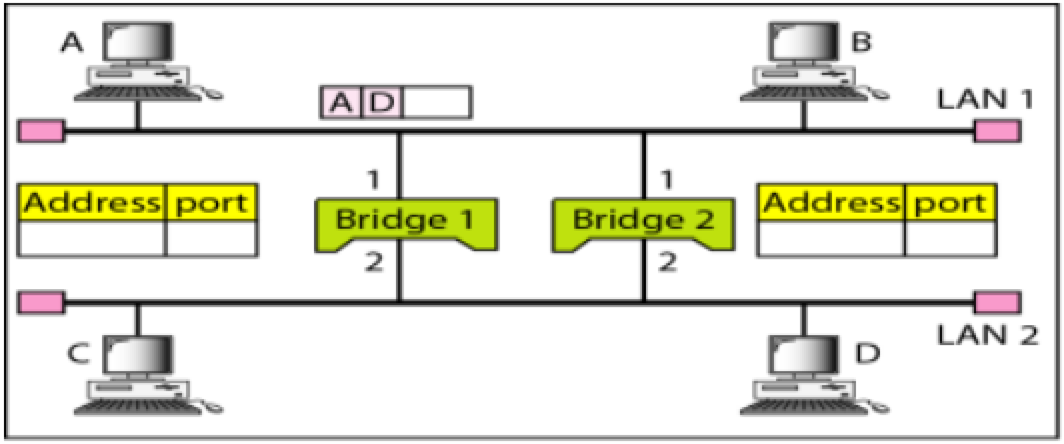
Loop Problem

- transparent switches work fine as long as there are no redundant switches in the system
- system administrators, however, like to have redundant switches to make the system more reliable
  - if one switch fails, another switch takes over
- unfortunately, redundancy can create loops in the system!

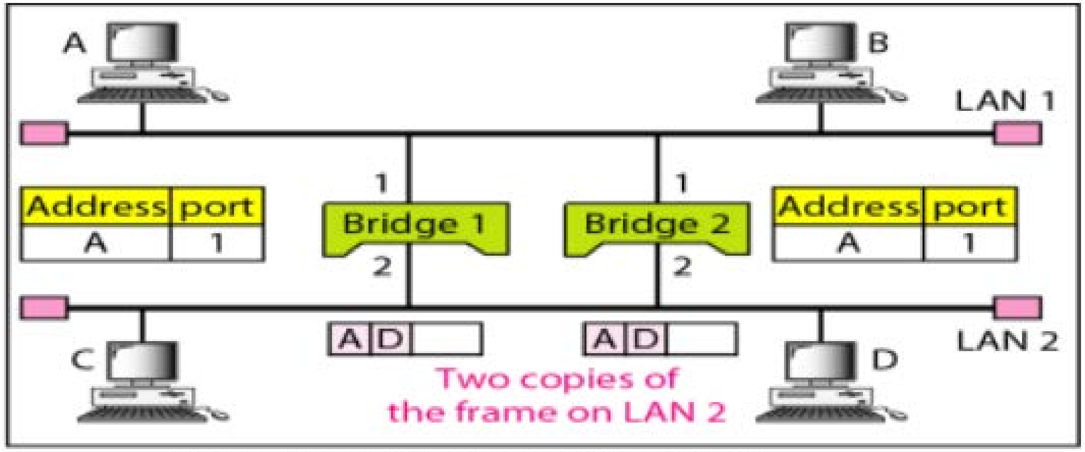


Connecting Devices: Bridges (cont.)

Example [loop problem]



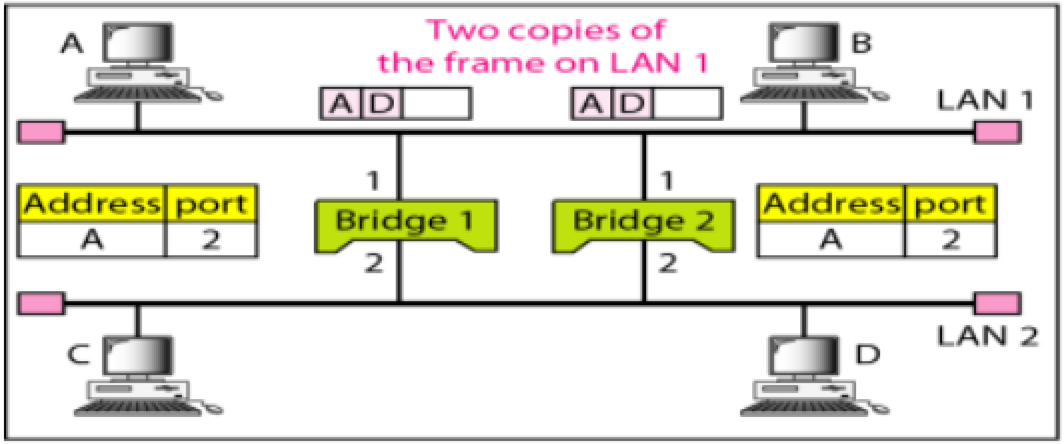
a. Station A sends a frame to station D



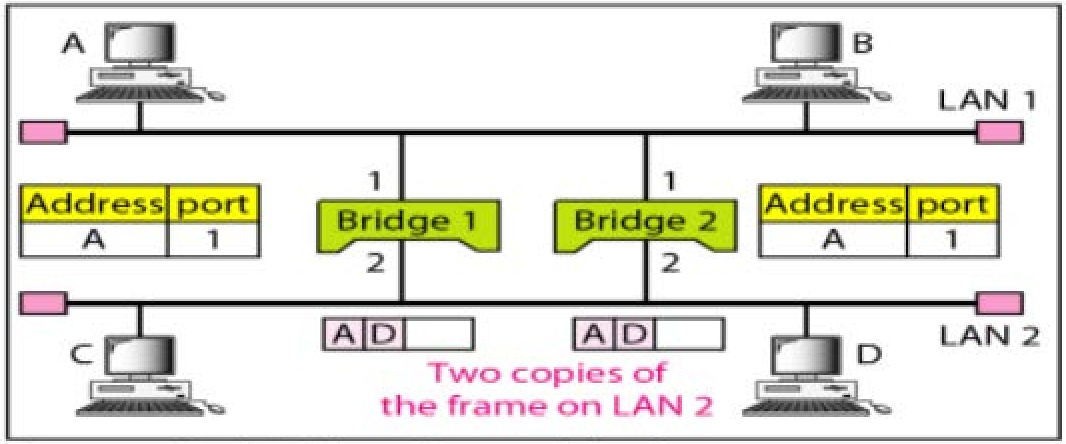
b. Both bridges forward the frame

Connecting Devices: Bridges (cont.)

Example **[loop problem cont.]**



c. Both bridges forward the frame

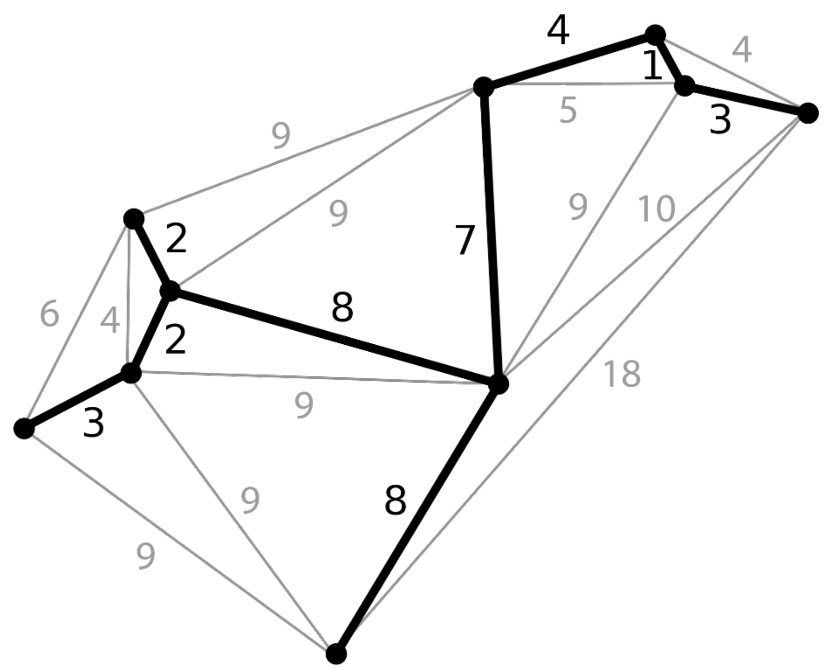


d. Both bridges forward the frame

Connecting Devices: Bridges (cont.)

Solution to Loop Problem

- **spanning tree = graph in which there is no loop**



- **each switch has a built-in ID (serial number), while each link between switches is assigned a cost**
  - **using this information, spanning tree is created**
  - **spanning tree ensures that there is only one path from one LAN to another!**

Connecting Devices: Bridges (cont.)

Spanning Tree Algorithm

1) The switch with the smallest ID is selected as the ROOT SWITCH (root of the tree).

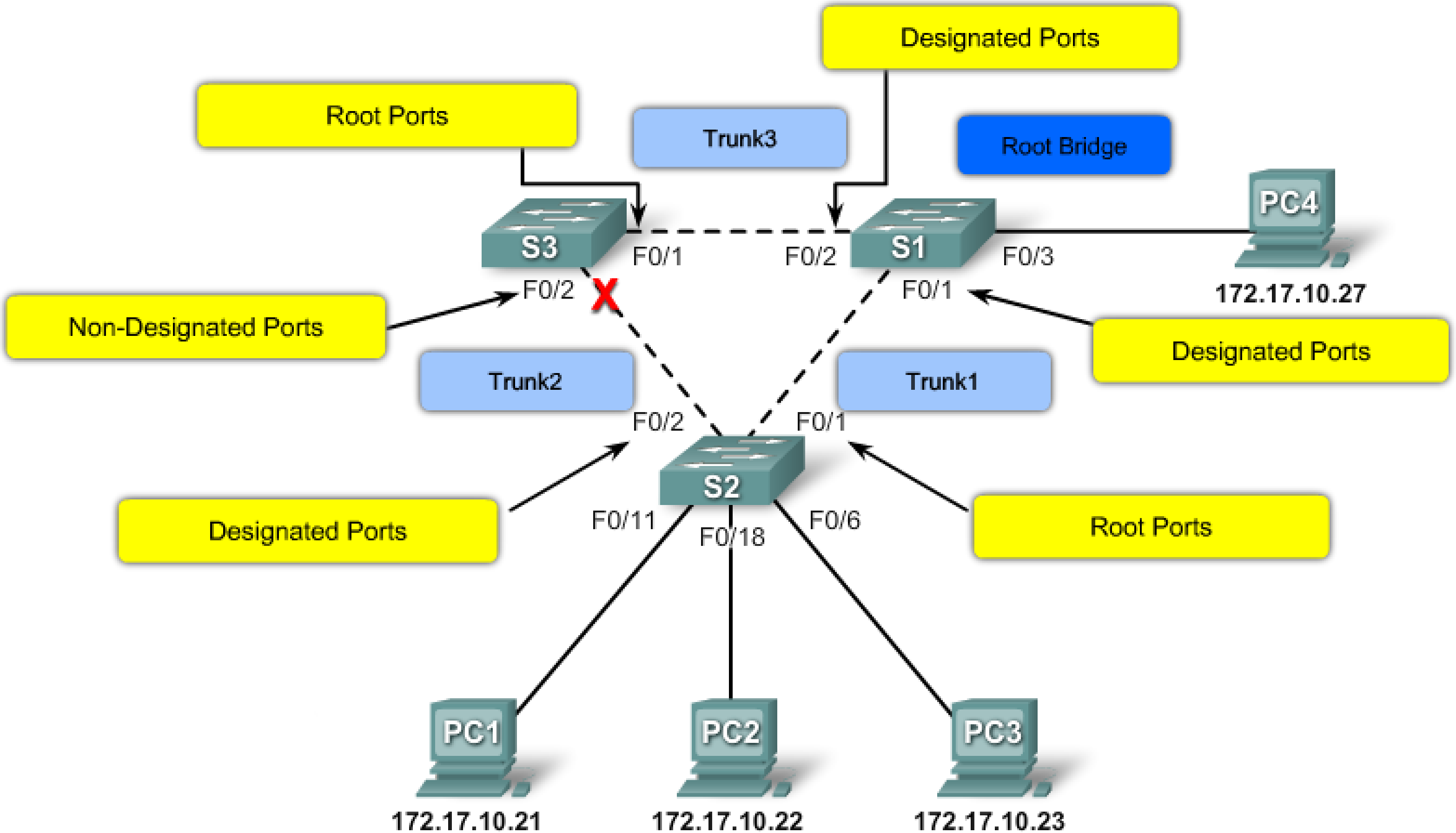
2) For every other switch, mark the port with the minimum hop-distance to the ROOT SWITCH as the ROOT PORT.
   - min-distances are calculated using Dijkstra algorithm
   - if two ports have the same distance, choose one!

3) Choose DESIGNATED SWITCH for each LAN.
   - this would be switch with the min distance to ROOT SWITCH
   - the corresponding port, that is forwarding away from the ROOT SWITCH, is DESIGNATED PORT

4) Mark the ROOT PORT and DESIGNATED PORT as FORWARDING PORT, others as BLOCKING PORT.

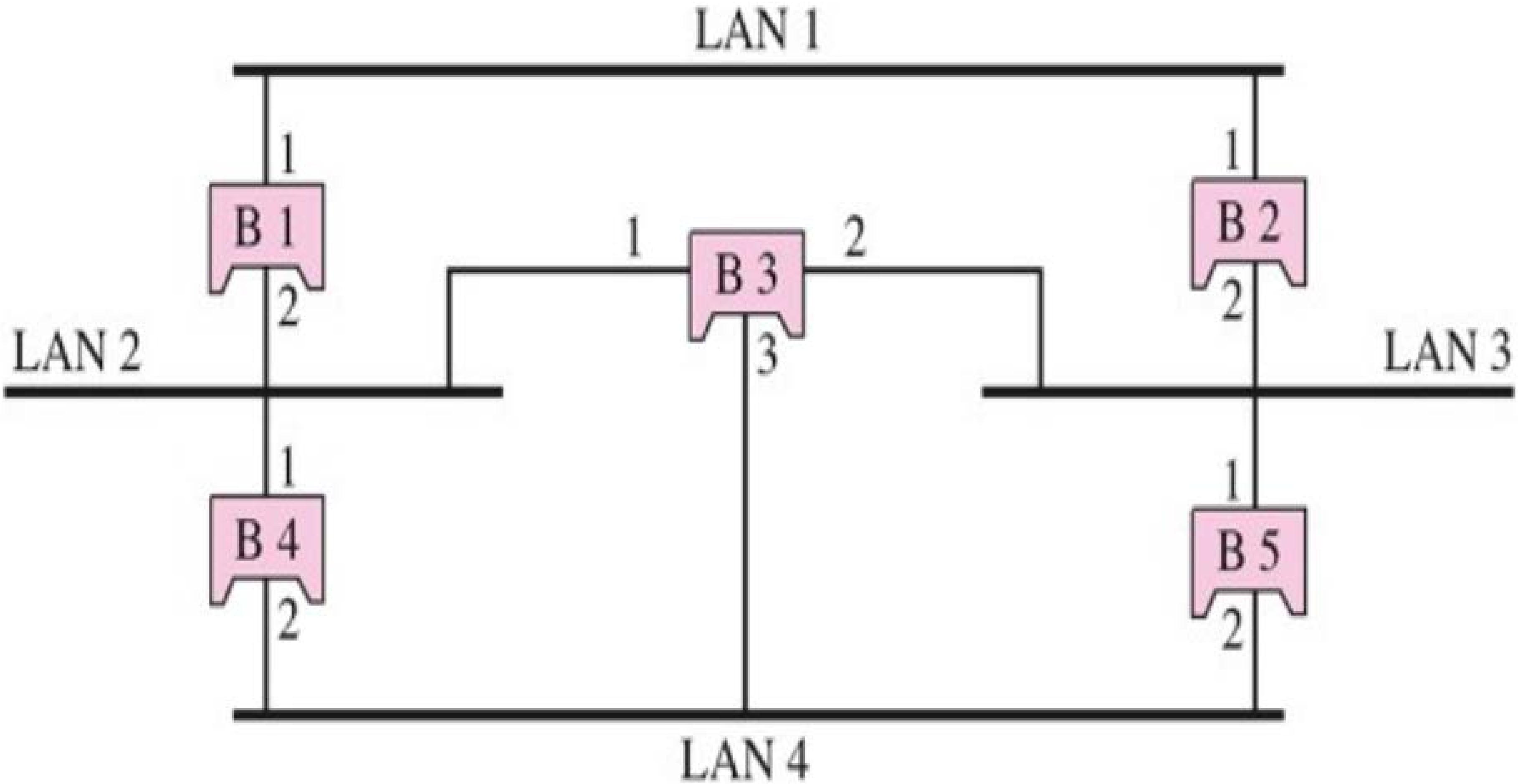
Connecting Devices: Bridges (cont.)

Example [spanning tree algorithm]



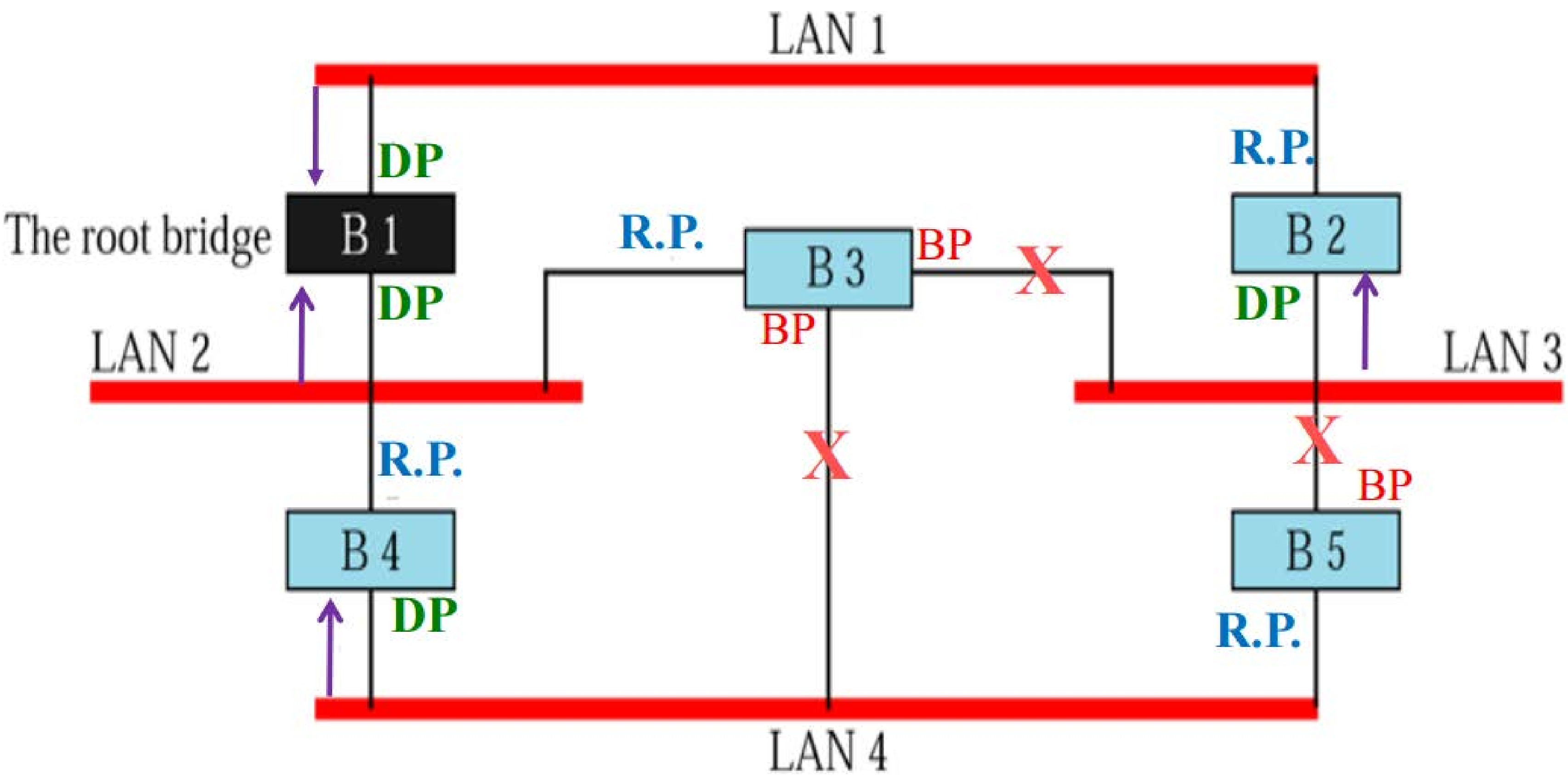
Connecting Devices: Bridges (cont.)

Example **[spanning tree algorithm]**



Connecting Devices: Bridges (cont.)

Example **[spanning tree algorithm]**



Connecting Devices: Bridges (cont.)

Bridges from 802.x to 802.y – theoretically, a bridge should be able to connect LANs using different protocols at the data-link layer; however difficulties encounter by such a bridge include:

- **frame format** – each LAN type has its own frame format – reformatting may be required prior to frame forwarding
- **maximum data size** – if an incoming frame's size is too large for destination LAN, data would have to be fragmented into several frames (this problem is solved at the network layer)
- **data rate** – each LAN has its own data rate – bridge must buffer the frame to compensate for this difference
- **security** – (e.g.) wireless LANs encrypt frames, so bridge need to decrypt frame before forwarding it to a wired LAN