

CMPE 150: Introduction to Computer Networks

Dr. Chane L. Fullmer
chane@cse.ucsc.edu

Mid Term

- Next Tuesday – April 29th
 - **Bring Scantron (large pink one)**
 - 30-50 questions
 - Multiple choice
 - Questions come from both text and lectures
 - Some may be similar to homework
 - ***Closed book, closed notes***

Homework Assignments

Homework assignment #2

Due by May 1.....

CMPE 150: Introduction to Computer Networks

LECTURE 8:

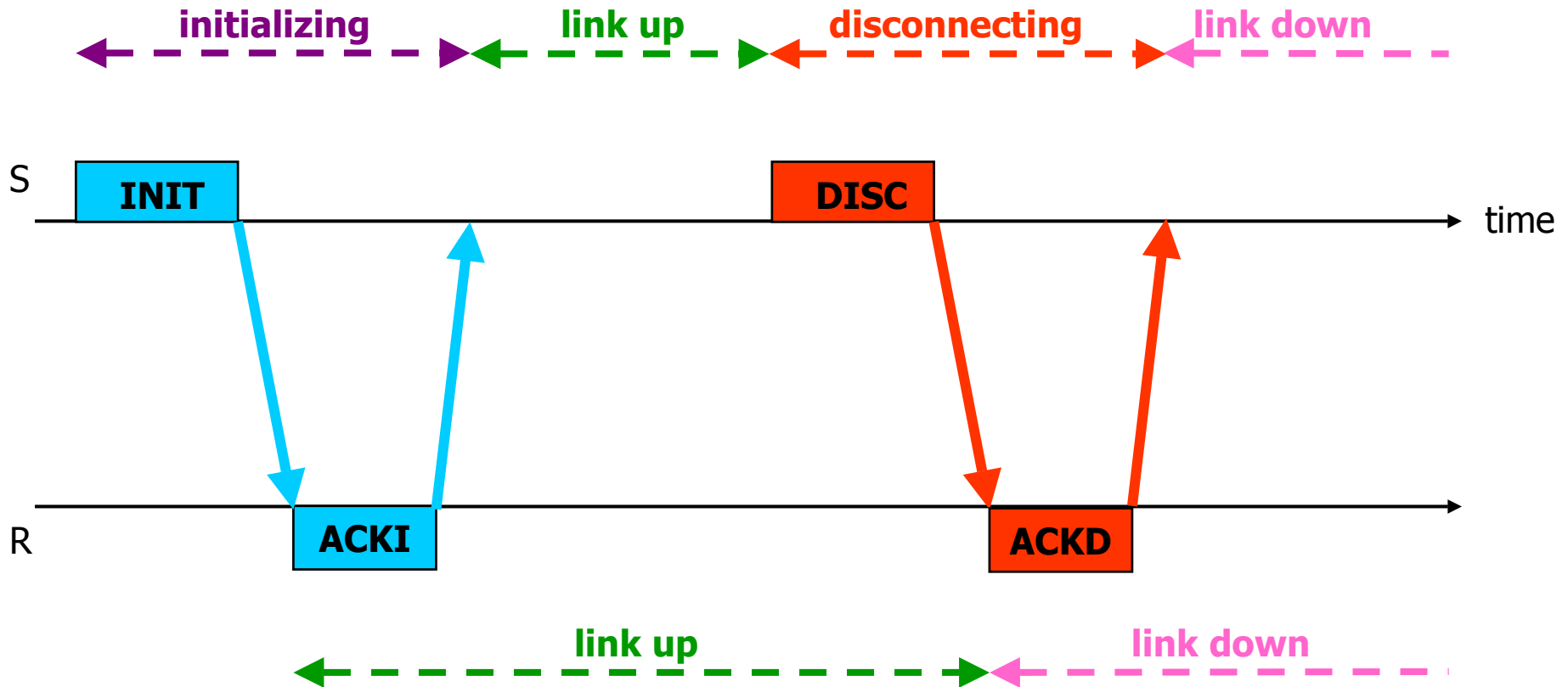
Logical Link Control, Part II

Internetworking with Bridges

Initializing a Link

- Operation of initialization protocol leads to up and down periods of the link
- S and R can have different views of the state of the link
- S and R should not accept packets from prior incarnations of the link.
- We can have a master-slave or peer-to-peer protocol, which consists of two M-S protocols.
- Master sends two types of packets: INIT and DISC, and must stop for ACKs
- **This is the same as the SWP protocol!**

Initializing a Link



Initialization of ARQ

- The basic problem is that an initialization protocol between two parties requires feedback.
- The simplest scheme is a link-up packet and its ACK used to set up a link, followed by a link-down packet and its ACK to tear down the link.
- This, however, is just the SWP, which needs to be initialized!
- ***Ad hoc approach:***
 - Node starts assuming that link is down, and must wait a long time period after coming up before it accepts or sends an INIT packet to the link.
 - In e-t-e protocols (e.g., TCP) the connection is assigned an ID randomly.

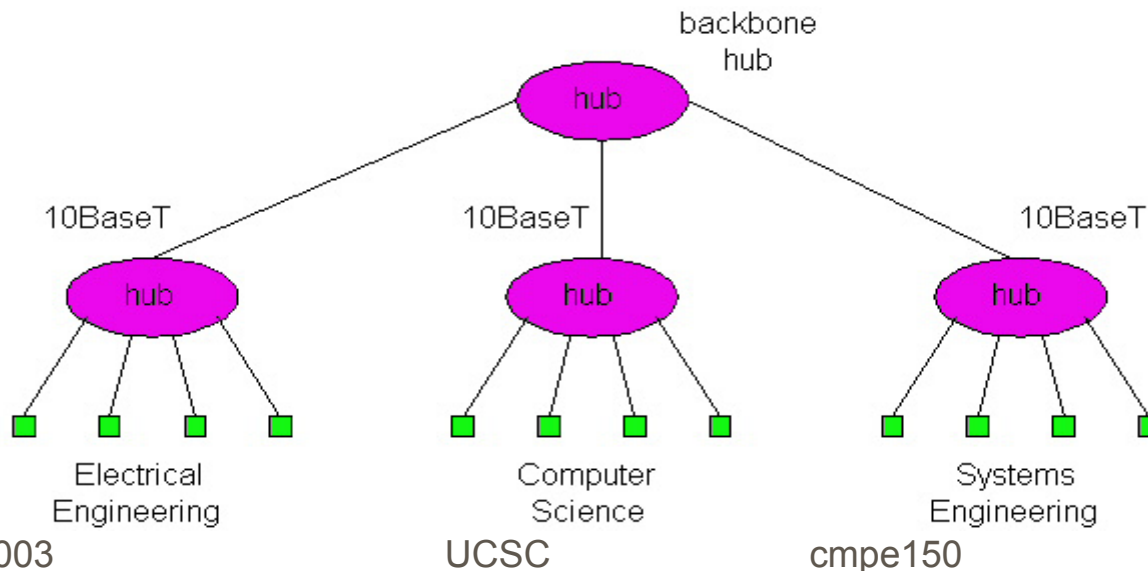
LAN Interconnection below Network Layer

- Hubs
- Bridges
- Switches

Section 5.6 of textbook

Interconnecting with hubs

- Backbone hub interconnects LAN segments
- Extends maximum distance between nodes
- Individual segment collision domains become one large collision domain!
 - ▣ If a node in CS and a node EE transmit at same time: collision
- Cannot interconnect 10BaseT & 100BaseT

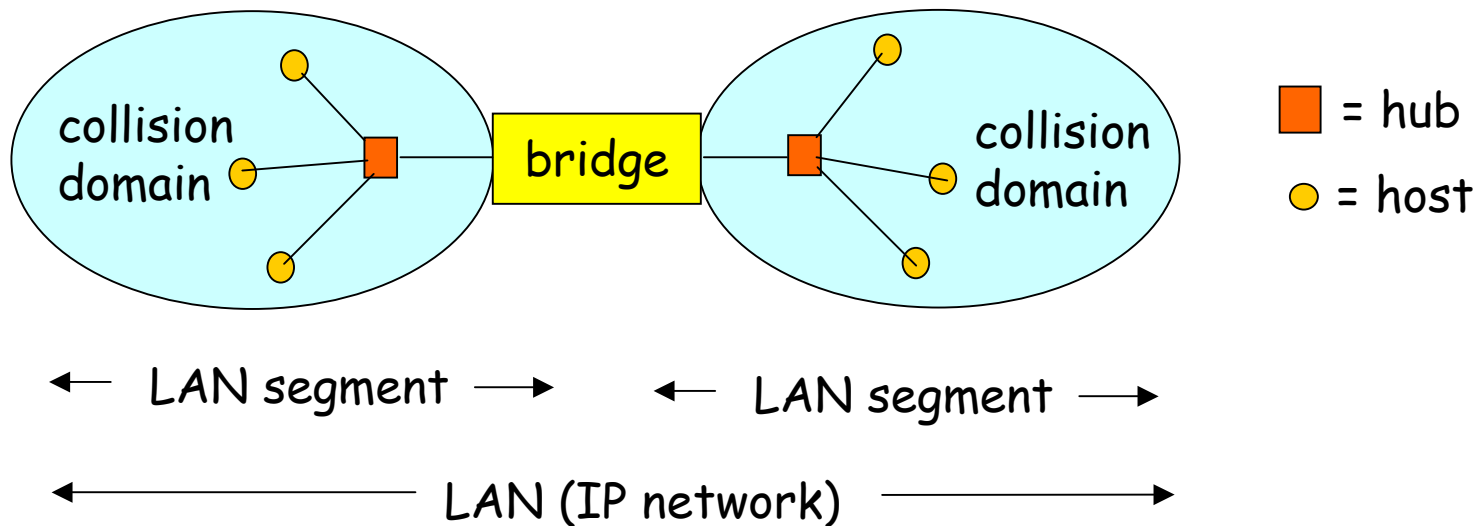


Internetworking with Bridges

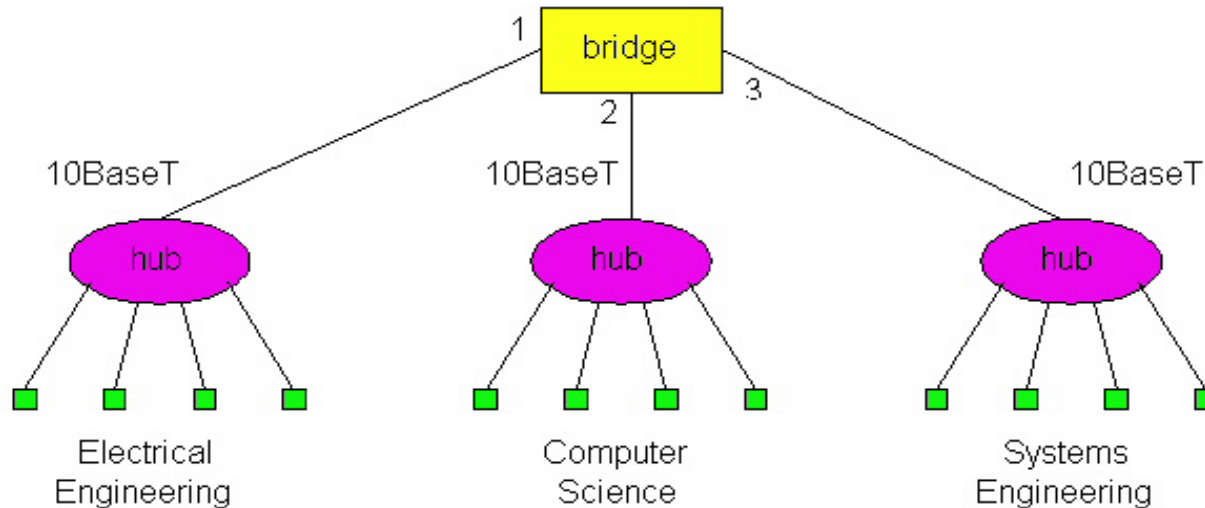
- Bridges are used to interconnect LANs at the **link layer**.
- Frame forwarding from one LAN to another is based on the destination's link-level address (MAC address) **without making any changes to the frame**.
- A MAC address is a name, and for a bridge the address of the destination is the adjacent LAN over which the frames to the destination should be forwarded.
- Plug-and-play, self-learning
 - bridges do not need to be configured.

Traffic Isolation with Bridges

- Bridge installation breaks LAN into LAN segments
- Bridges **filter** packets:
 - Same-LAN-segment frames not usually forwarded onto other LAN segments
 - LAN segments become separate **collision domains**



Internetworking with Bridges



- To which LAN segment should the bridge forward a frame?
- A routing problem!
- There are two types of bridges that have been used:
 - **Transparent**
 - **Source routing**

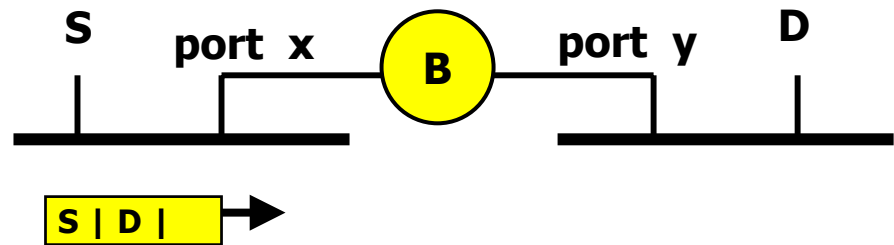
Transparent Bridges: Summary

- The purpose of transparent bridges is to keep the packet forwarding functionality transparent to the hosts.
- Transparent bridges establish and manage a spanning tree of the network to eliminate packet looping.
- The address of a station is always the LAN over which packets from that station came last; this is a dynamic process.
- If no address is known, a bridge broadcasts packets for a station over all its ports (or those in the spanning tree).

Addressing in Transparent Bridges

- Assume for now that the topology of the internet is a tree.
- Bridge listens to every packet it receives over any LAN.
- It builds a **station cache** consisting of the source addresses of packets it hears and the IDs of the ports over which the packets were heard.
- For the bridge, the address of a station is the port over which packets from the station were received.

B assigns port x as the address of station S after hearing the packet from S.



Routing in Transparent Bridges

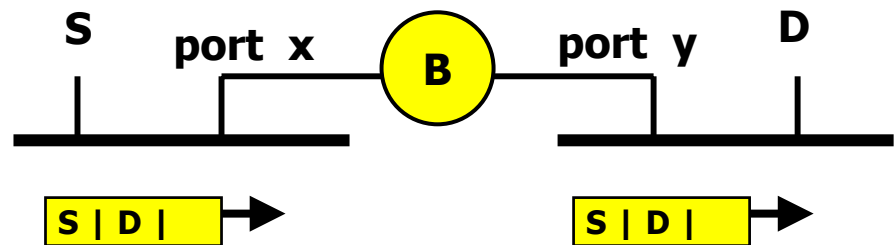
- Consider a tree topology.
- When a bridge receives a packet, it looks up its station cache for the destination MAC address in the packet.
- If match is found then:
 - If port in the cache is the same port over which packet came, the packet is filtered (dropped)
 - Otherwise, the bridge forwards the packet to the port specified in the cache.
- If no matching is found, the bridge forwards the packet over all ports other than the port from which the packet came.

B has station cache entry:

D - port y

or

B does not know about D and forwards to port y (and other ports other than x)

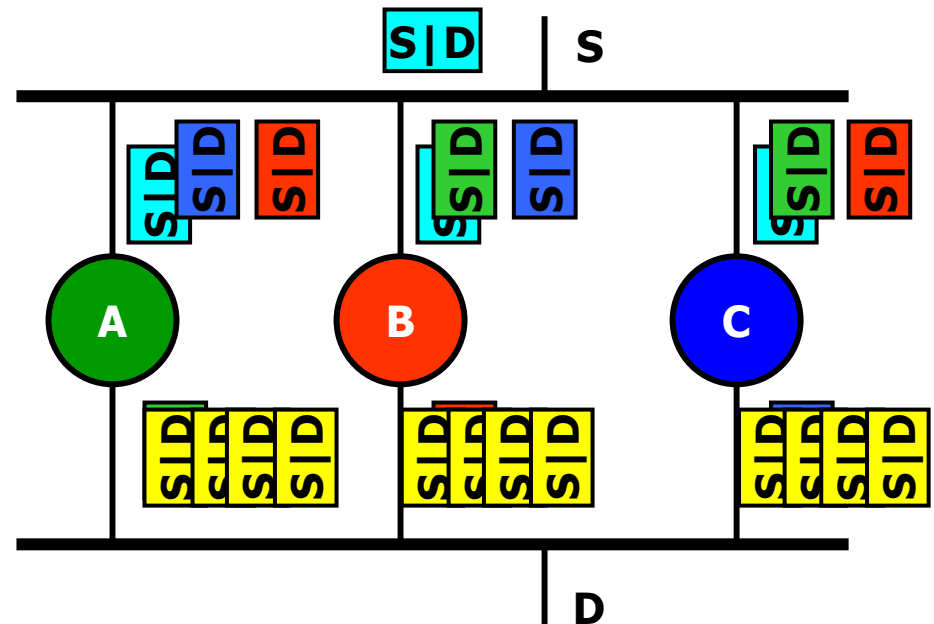


Looping Problems

- The topology of the internet need not be a tree!
- The address learning process is such that packets will traverse loops, and worse, replicas of such packets will be produced and sent over the same loops!
- **S sends a packet to D, D is silent, and bridges do not know about D.**

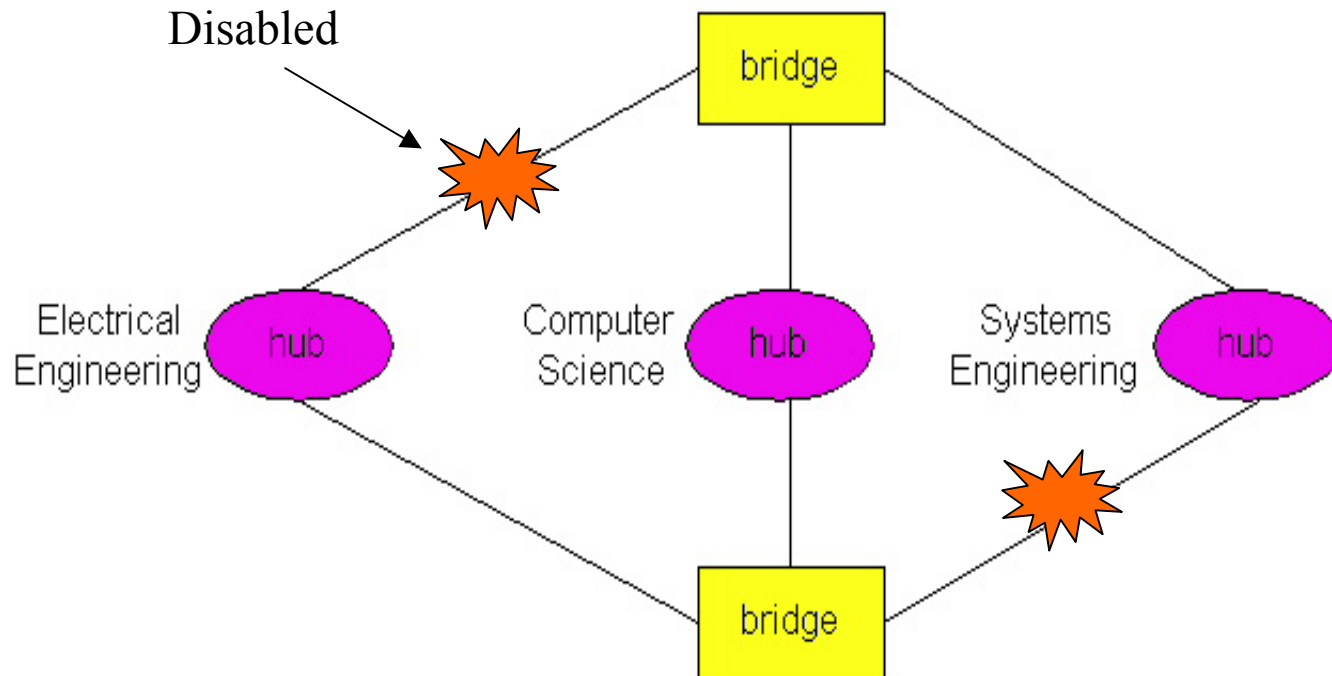
Packet is replicated three times at each LAN!

Looping with bridges is REALLY BAD and must be avoided!



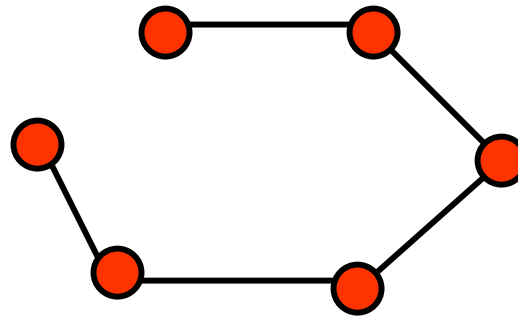
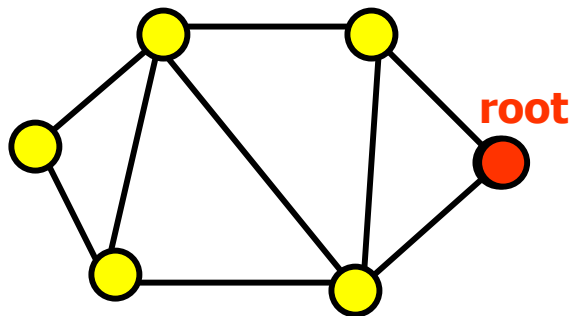
Bridges in Mesh Topologies

- Alternative paths from source to destination LANs are desirable for increased reliability.



Spanning Tree Algorithm (STA)

- The objective is to define a single spanning tree in the internet over which packets flow without looping.
- Basis of operation (Perlman 1992, part of IEEE standard):
 - ▢ Elect distributedly a single bridge as the **root** of the tree
 - ▢ Calculate distance (in hops) on a shortest path to root
 - ▢ Elect a **designated bridge** for each LAN (e.g., closest to the root in the LAN)
 - ▢ Allow only designated bridge to forward packets to and from its LAN



A distributed election process is used to build the spanning tree!

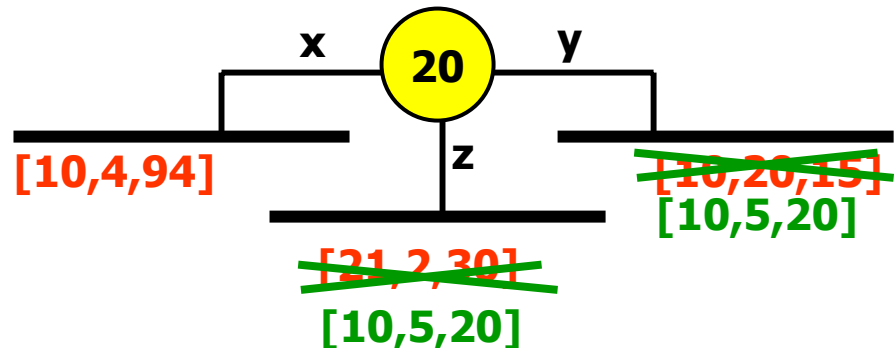
STA Operation

- Each bridge has multiple MAC addresses (one per port)
- A bridge has a bridge-wide ID (one of the MAC addresses)
- **HELLOs**: messages used to build tree, sent to all bridges of a LAN
- **HELLO** specifies:
 - **Root ID**: The MAC address of the bridge assumed to be the root
 - **Transmitting bridge ID**: MAC address of bridge sending HELLO
 - **Cost**: Length (in hops) of path from bridge to root
- A bridge starts by considering itself the proposed root
- Bridge starts election process by sending
 - **HELLO = own ID, 0, own ID**

STA Operation

- Bridges **adopt the smallest HELLO they hear**:
 - Minimum root ID
 - Smallest distance to root
 - Minimum reporting bridge ID
- Bridge compares its own HELLO with its neighbors' HELLOs, and chooses the smallest
- Its **root port** becomes the port to neighbor bridge with smallest HELLO
- Bridge composes a new HELLO, adding 1 to the distance to adopted root

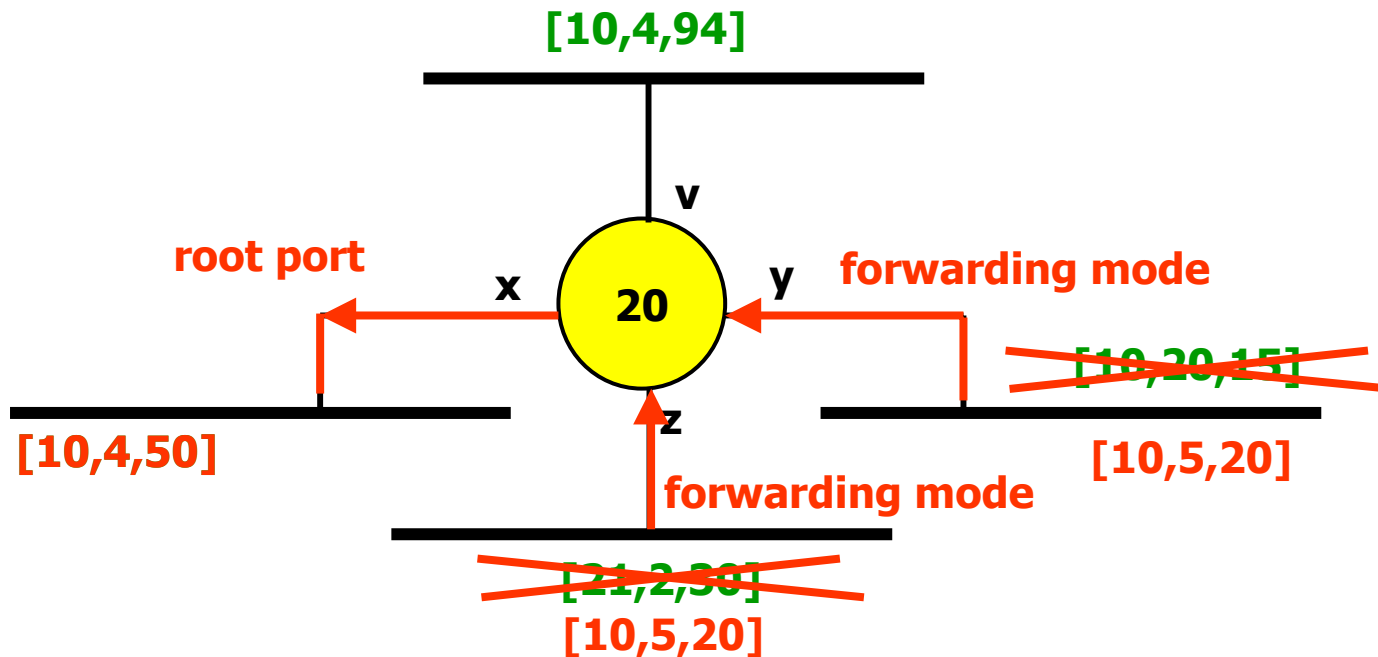
Bridge 20 must adopt HELLO from neighbor 94 over port x: smallest root ID and smallest distance to root!



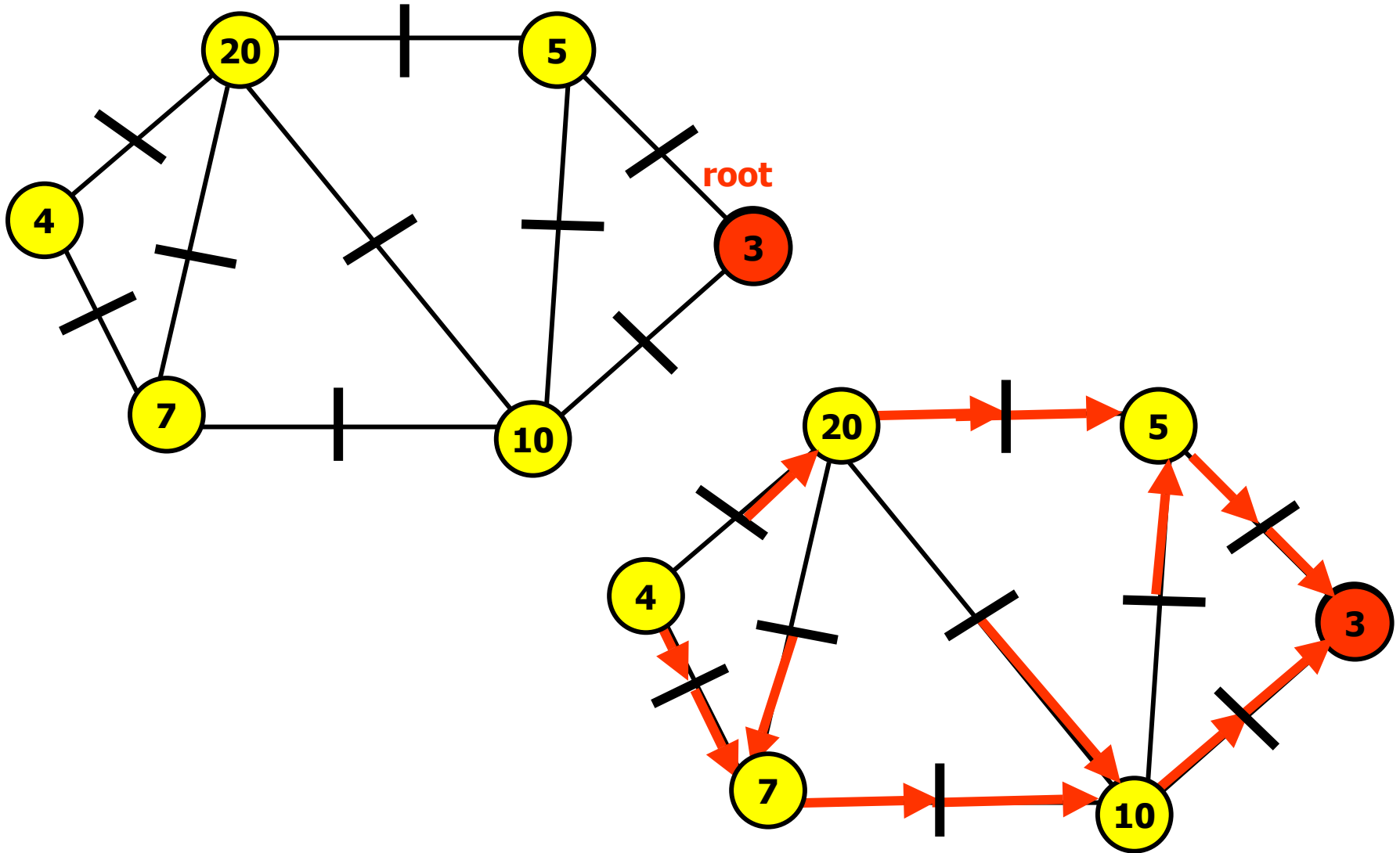
STA Operation

- Bridge sends new HELLO over all ports from which “larger” HELLOs were received.
- Bridge knows if it is the designated bridge for a LAN if it does not hear a “smaller” HELLO than its own.
- Its root port is the port from which the smallest HELLO was received.
- Bridge puts its root port and all ports for which it is the designated bridge in **forwarding state**.
- Bridge puts all other ports in **blocking state**.
- Data packets, control packets, and learning of addresses take place only over ports in forwarding state (over the spanning tree).

Example of STA Operation



Example



Handling Failures

- Problem is that failed bridges stop sending HELLOs.
- Procedure:
 - Each HELLO has an age field
 - Root sends HELLO periodically with 0 age
 - Receiving bridge forwards HELLO (with new distance and its own ID for reporting bridge) for all ports for which it is designated bridge

HELLO from root propagates through spanning tree in the absence of failures!

If root or other bridge fails, bridges down-tree stop receiving HELLOs originated by the root

Handling Failures

- ❑ Age of HELLO is incremented over each hop and each time unit while in storage
 - ❑ Stored HELLOs are discarded when ages reach maximum value
 - ❑ Bridge recomputes best HELLO (with a valid age) for port for which HELLO is deleted
 - ❑ Bridge can decide to become root if it provides the best new HELLO
- Spanning tree calculation occurs when:
 - ❑ HELLO is received from a port
 - ❑ HELLO is discarded

Temporary Loops

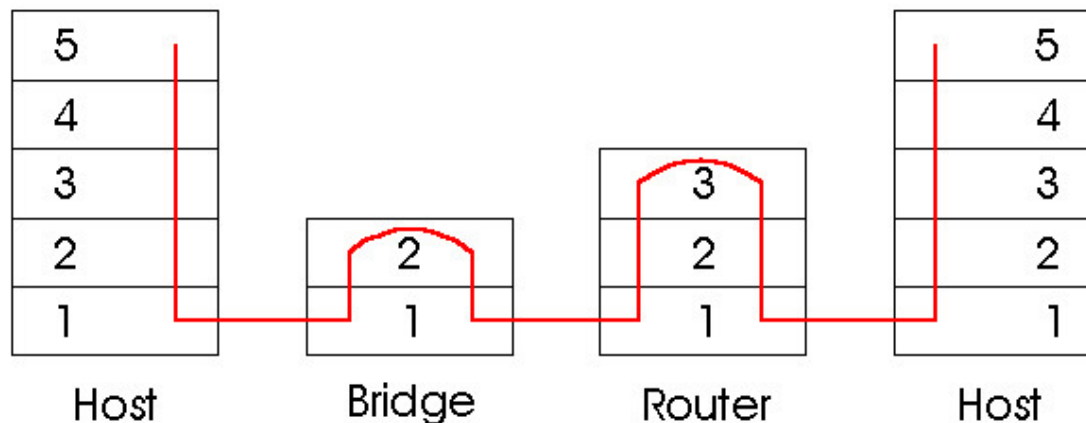
- Because packets have no TTL, they loop indefinitely and multiply, until tree is correct.
- STA is not absolutely free of looping!
- Ad hoc approach: Make a bridge wait a long time before moving a port to forwarding mode.
- Hold-down timer is set to twice the maximum transit time in the internet (say 30 sec).
- This solution is very slow in large internets or does not work at all, because we don't know the maximum transit time after a topology change.

Refreshing Station Cache

- The tree can change due to link and bridge failures or stations physically moving.
- Bridges must distinguish between the two types (stations move very slowly) to decide how often to refresh station cache (so that packets are not forwarded in vain).
- Approach:
 - Bridge that changes the state of a port sends a topology change notification (TCN) to the root (on its root port) persistently, until the "parent" bridge ACKs (with a flag in its HELLO)
 - Bridge receiving a TCN forwards it towards the root
 - If root detects a topology change or receives a TCN, it sets the TCN flag in its own HELLO for a period of time (say 30 sec).
 - The HELLOs with TCN set force bridges to refresh their station caches more often, until the TCN is reset.

Bridges vs. Routers

- Both store-and-forward devices
 - ▣ Routers: network layer devices (examine network layer headers)
 - ▣ Bridges are link layer devices
- Routers maintain routing tables, implement routing algorithms
- Bridges maintain bridge tables, implement filtering, learning and spanning tree algorithms



Routers vs. Bridges

Bridges + and -

- + Bridge operation is simpler requiring less packet processing.
- + Bridge tables are self learning.
- All traffic confined to spanning tree, even when alternative bandwidth is available.

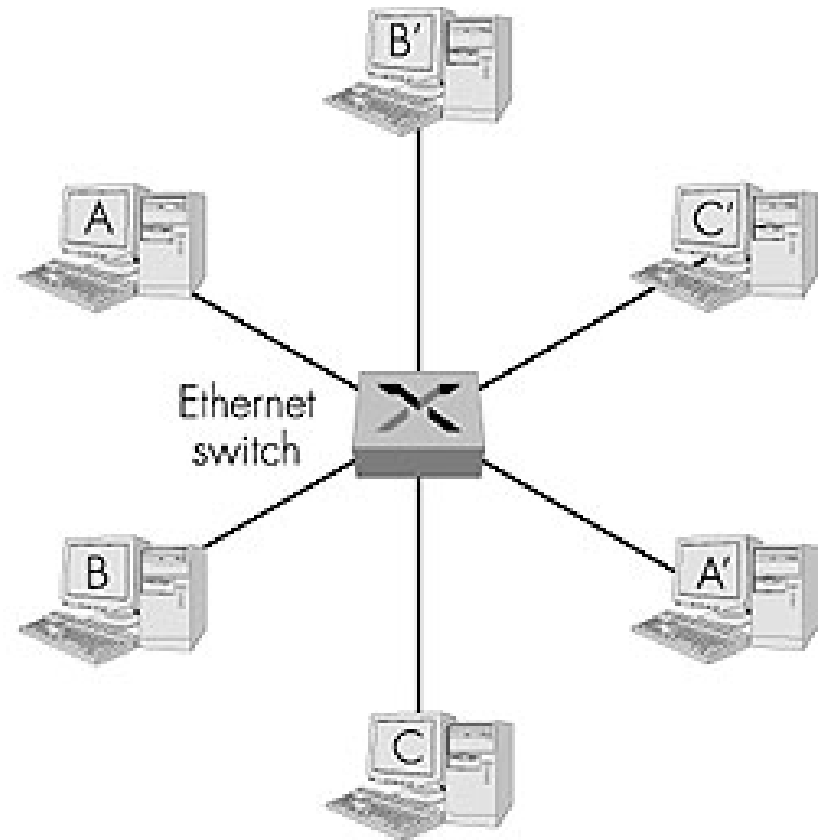
Routers vs. Bridges

Routers + and -

- + arbitrary topologies can be supported, cycling is limited by TTL counters (and good routing protocols)
- + provide protection against broadcast storms
- require IP address configuration (not plug and play)
- require higher packet processing
- bridges do well in small (few hundred hosts) while routers used in large networks (thousands of hosts)

Ethernet Switches

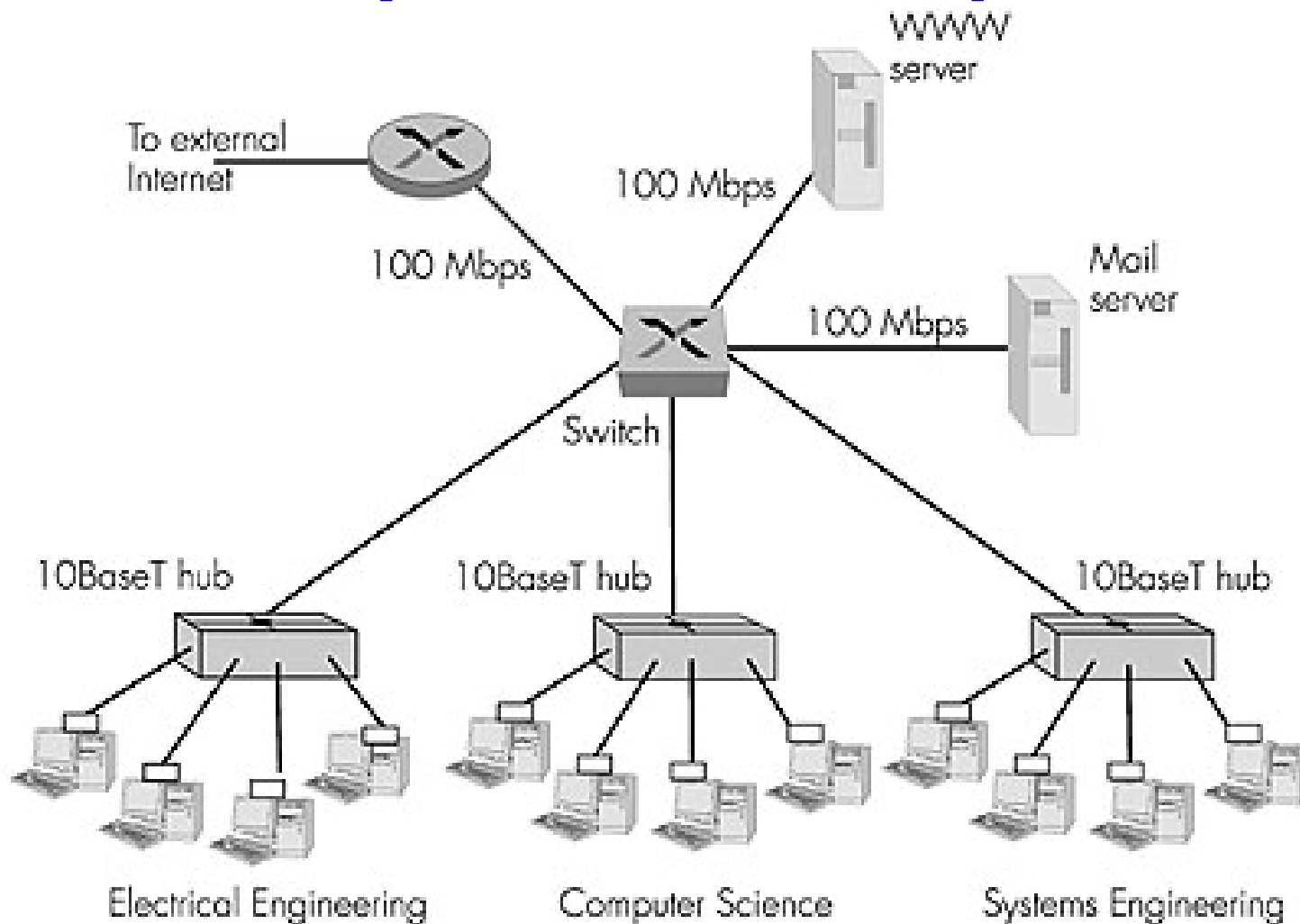
- Essentially a multi-interface bridge
- layer 2 (frame) forwarding, filtering using LAN addresses
- **Switching:** A-to-A' and B-to-B' simultaneously, no collisions
- large number of interfaces
- often: individual hosts, star-connected into switch
 - Ethernet, but no collisions!



Ethernet Switches

- **cut-through switching:** frame forwarded from input to output port without awaiting for assembly of entire frame
 - ▣ slight reduction in latency
- combinations of shared/dedicated, 10/100/1000 Mbps interfaces

Not an Atypical LAN (IP network)



Summary Comparison

	<u>hubs</u>	<u>bridges</u>	<u>routers</u>	<u>switches</u>
traffic isolation	no	yes	yes	yes
plug & play	yes	yes	no	yes
optimal routing	no	no	yes	no
cut through	yes	no	no	yes

Have A Nice

Weekend