E6998-02: Internet Routing

Lecture 2
Bridging

John Ioannidis

AT&T Labs – Research

ji+ir@cs.columbia.edu

Announcements

Class web page: http://www.cs.columbia.edu/~ji/F02/

Lectures 1 and 2 are available.

Homework 1 is available, due 9/12 at 3am.

- Submissions will be electronic, probably over email.
- Only plain-ASCII or PDF files will be accepted.

I’ve added an announcements page.

Class BBoard: still TBA

Office hours: Tuesdays 16:00-17:30 or by appointment.

I know they conflict with W4180!

TA(s): I’m looking for one. Any volunteers?

Mike Schiraldi <mgs21@columbia.edu> is willing to organize a bulk order to Amazon. Talk to him.

Link-layer Addressing

- You should already know about this.
- Ethernet-like LANs, MAC address:
  - 48-bit, unique.
  - Flat namespace as far as addressing is concerned.
  - Appears at the beginning of a frame.
  - It’s a name, really.
  - Unicast/multicast addresses.
- Point-to-point connections:
  - No need for a station address.
  - Still need for service/higher layer protocol identifiers.

Connecting LANs

- Plugging them together usually not an option.
  - Distance limitations.
  - Capacity limitations.
  - Administrative/security limitations.

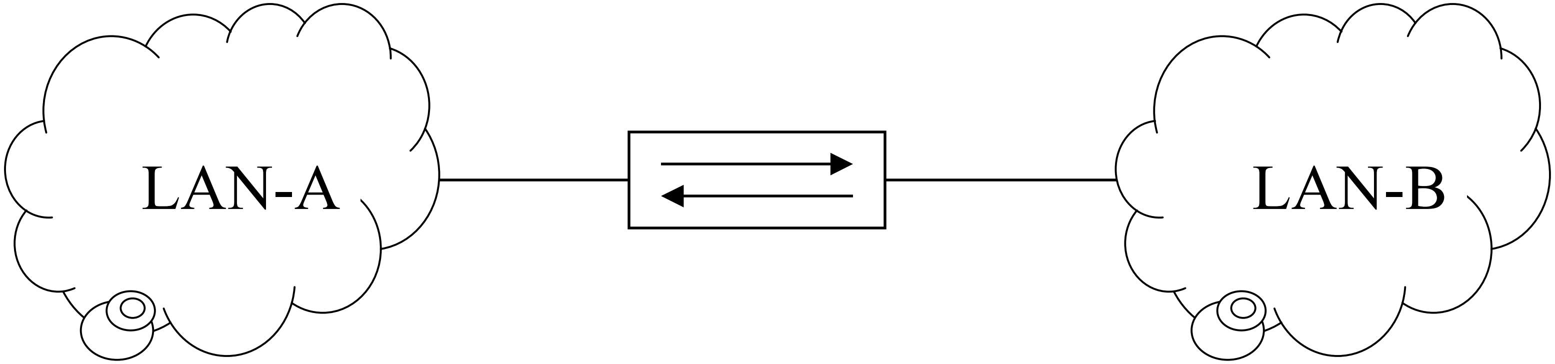
Bridges

- Link two or more broadcast LANs together.
- Work at the link layer.
  - Only look at MAC addresses.
- Why?
  - Capacity.
  - Distance.
  - Some technologies (TR) have problems with # of nodes.
- None of these problems are solved by repeaters (layer 1).

Why Bridges and not Routers?

- Originally, accommodate layer-2 only nodes.
- Then,
  - multiprotocol considerations.
  - Performance (cheaper/faster throughout the 80s).
  - before subnetting/VLSM.
- Still useful to move IP nodes around.
- Modern switches are really bridges.
- Good for linking similar LAN technologies.

Ethernet Bridges



- The box in the middle is a bridge.
- Dumb bridge:
  - – Just copies all traffic between the LANs.
  - – Little more than a repeater (increases distance).
  - – Does not increase total network throughput. But:
    - May have enough buffer capacity.
    - May drop packets that didn’t need forwarding.

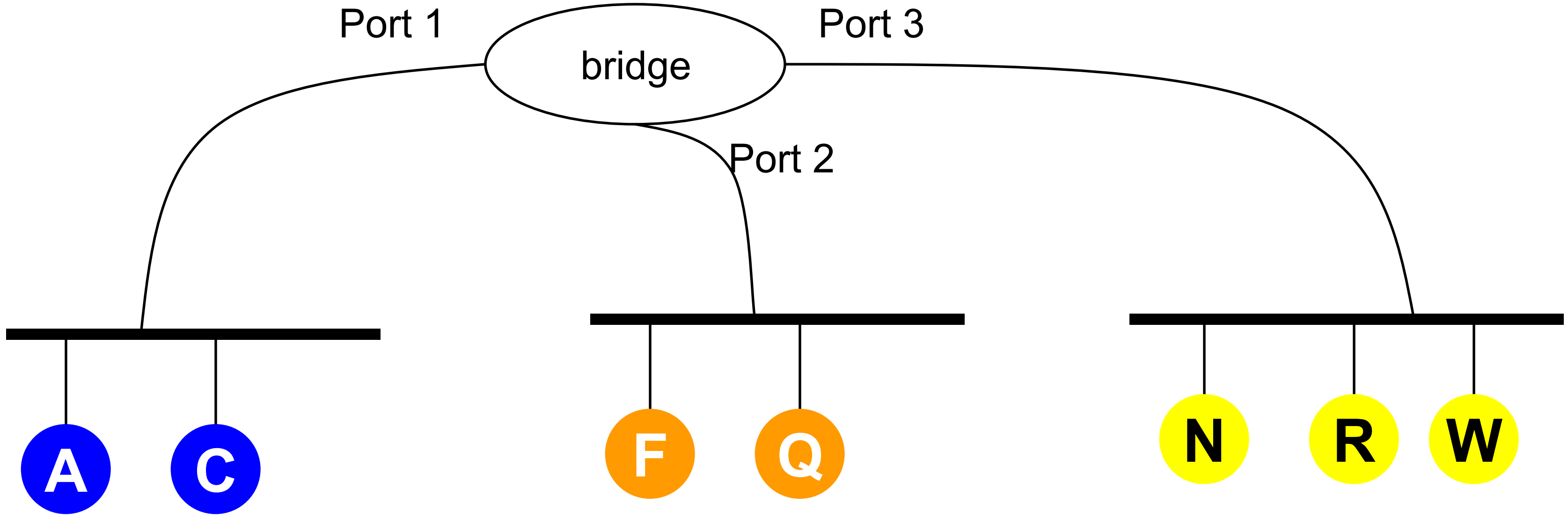
Improving the Dumb Bridge

- Slight improvement:
  - Tell bridge which nodes are on which side, or
  - Tell bridge which MAC addresses to forward.
  - Assign MAC addresses hierarchically.
    - Usually infeasible, MAC address in PROM.
- All of these involve too much configuration.
- They also don’t scale up.

- Things get worse for bridges with multiple interfaces.

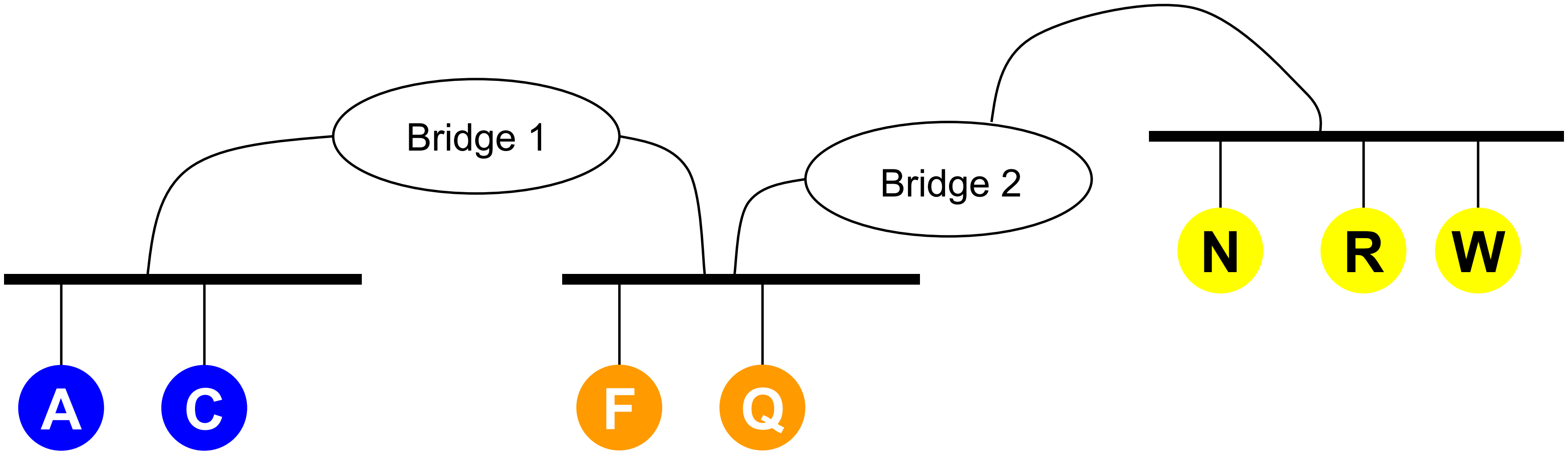
The Learning Bridge

- Bridge listens promiscuously.
- Store source MAC address in cache, indexed by port.
- Look up dest MAC address:
  - If not in cache, forward to all ports.
  - If in cache AND on different segment, forward to port connected to that segment.
- Cache entries are aged, replaced by LRU.



Packet	Appears in	Port 1	Port 2	Port 3
A→C	1,2,3	A		
A →W	1,2,3	A		
F →A	2,1	A	F	F
Q →F	2	A	F,Q	F,Q
R →C	3,1,2	A	F,Q	R
A →C	1	A	F,Q	R
C →A	1,2,3	A,C	F,Q	R
R →C	3,1	A,C	F,Q	R
N →*	3,1,2	A,C	F,Q	R,N
F →N	2,3	A,C	F,Q	R,N

LB works for Loop-Free Topologies



- To Bridge 1, the orange and yellow segments look like one segment.
- To Bridge 2, the blue and orange segments look like one.
- Stations don’t have to worry about it.

- What happens if we have a loop?

Loops and Bridging

- Get rid of bridges, find some other technology.
- Forbid loops.
  - Loops are good for redundancy.
  - Loops may happen accidentally.
- Have bridges complain about loops.
  - Can they detect them?
- Add functionality to handle loops.
  - Could do it manually.
  - More interesting to do it automatically.

Graph theory: what is a spanning tree?

- A subgraph containing all the vertices and has no cycles.
- We can assign *weights* to each edge.
  - Even if it's 1 on all edges.
  - May be interpreted as the cost to traverse the edge.
- Then we can define a Minimum-Weight Spanning Tree.
  - Is it unique?
- Why can't we use what is in the algorithms book?

Thinking about the ST Algorithm

- The bridges don’t have knowledge of the entire network.
- It’s a *distributed computation*!
- Any node in a tree can become the root.
  - “pick it up and shake it!”
- Each node has one parent.
  - The way to the root is through the parent.
- Each non-leaf node has children.
  - For which *it* is the way to the root, and must tell them so.
- Each node selects the best children and the best parent.

- Nodes and LANs can be added and removed; the algorithm must cope with that.

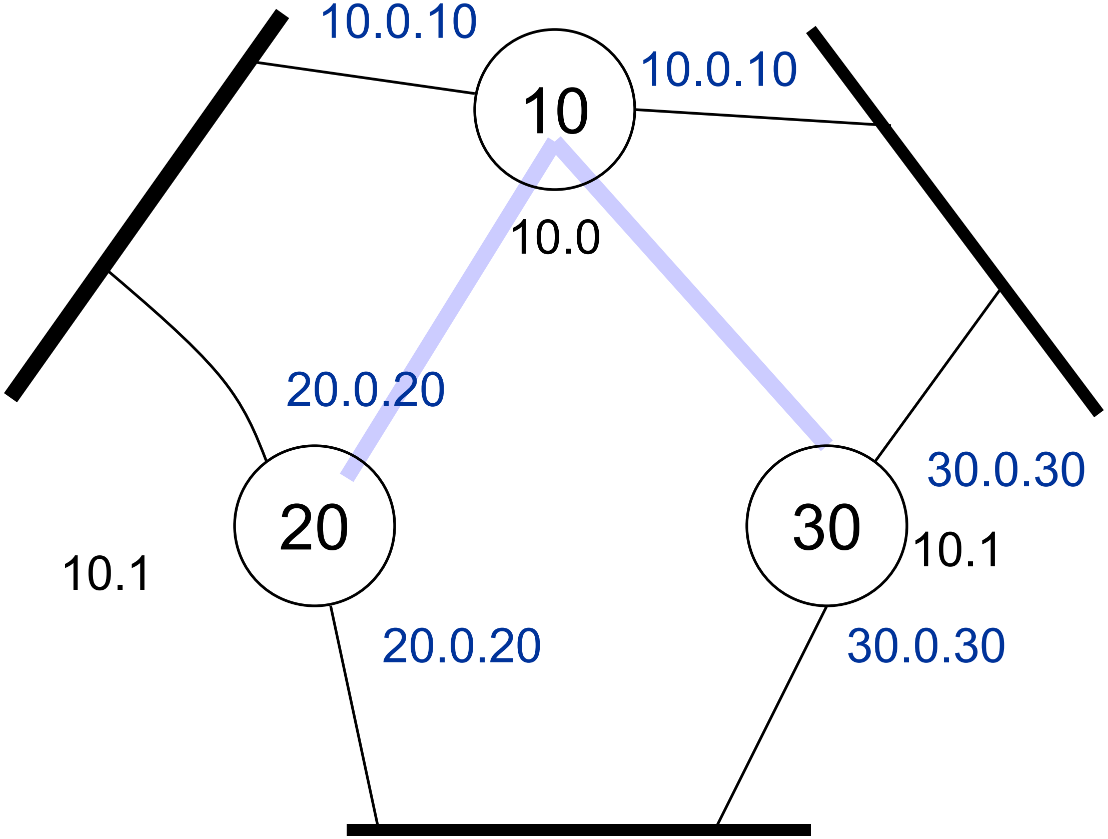
Spanning Tree Algorithm

- Bridges send configuration messages to:
  - Elect a bridge as the *root bridge*.
  - Calculate shortest path to root bridge.
  - For each segment, elect a *designated bridge*: the one closest to the root bridge.
  - Designated bridge forwards packets from that LAN toward the root bridge.
  - Choose a port that gives the best path to the root bridge.
  - Select ports to be included in the spanning tree.
- Once a spanning tree has been formed, bridges act as learning bridges.
- Configuration messages keep being sent to detect topology changes.

Bridge Configuration Messages

- Root ID: ID of bridge assumed to be the root.
- Cost: sum of weights of the least-cost-path to root from transmitting bridge.
    - We are introducing the concept of the *link metric* here.
- Transmitting bridge ID.
- Port ID: transmitting bridge's port id where message was sent out on.

- Ordering of configuration messages:
    - Compare root IDs.
    - If equal, compare costs.
    - If equal, compare transmitting bridge IDs.
    - If equal, compare ports.
        - Only useful when two ports are connected to the same LAN.


10.0.10
10.0.10
10
10.0
20.0.20
30.0.30
20
30
10.1
10.1
20.0.20
30.0.30

When Topology Changes

- Algorithm as described adapts to *additions* of links/nodes.
- To handle failures:
  - Stored configuration message for a port is aged.
    - When it reaches max age it is discarded and ST is recomputed (perhaps causing configuration messages to be resent).
  - Root bridge periodically retransmits *hello* configuration messages, with age field of 0.
  - Downstream bridges do likewise.
- ST recalculation:
  - Receipt of a configuration message.
  - Timing out of a stored configuration message for a port.

Why an Age Field?

- When new bridge comes up it sends configuration message.
- Any bridge hearing that retransmits its (stored) configuration message, but with the current age field.
- Why?

- This way new bridge has a pre-aged configuration message.
- The resulting behavior is the same as it would be if the new bridge had been there since the beginning.

Avoiding Temporary Loops

- Loops are BAD!
  - No TTL for packets at the link layer.
- A loop may form when a bridge turns a port from blocked to forwarding.
- It should wait for some time during which:
  - It propagates hello messages, but
  - It does not propagate data traffic.
- 801.d splits the non-propagating phase in two:
  - Just listen for conf messages.
  - Then listen for data and build the learning cache.

Configurable Parameters

- Max age.
- Hello time.
- Forward delay.
  - Amt of time in the learning/listening states.

- Port ID.
- Bridge priority.
- Port priority.
- Long cache timer.
- Path cost.

Configuration Message Format

- Protocol ID (=0), 16 bits
- Version (=0), 8 bits
- Message type (=0), 8 bits
- Topology Change Ack flag, 1 bit
- Topology Change flag, 1 bit
- Root ID, 64 bits (16 bits priority, 48 bits MAC address)
- Cost of path to root, 32 bits
- Bridge ID, 64 bits
- Port ID, 16 bits (8 bits priority, 8 bits port number)
- Message age, 16 bits, in 1/256$^{\text{ths}}$ of a second.
- Max age, 16 bits
- Hello time, 16 bits
- Forward delay, 16 bits

Topology Change Message

- 0x00000080!

Other Bridge Issues

- Multiply connected stations.
- Filtering.
  - By protocol.
  - By MAC address.
- Multicast.
- Remote bridges/half bridges.

- The entire discussion on bridges applies to switches.

- We may talk about source routing bridges later on; read about it in Perlman.