University of Glasgow

Bridging

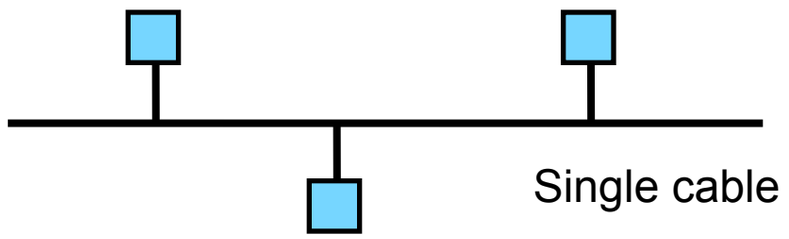
Networked Systems (H)
Lecture 6

Lecture Outline

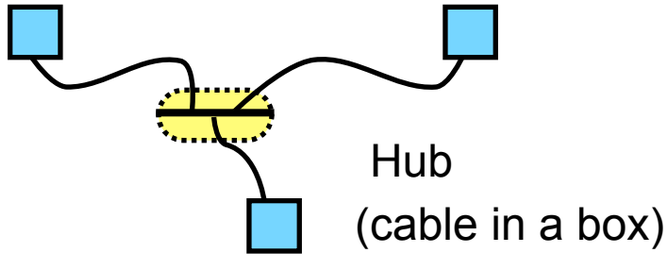
- Link-layer topology evolution
  - Hubs
  - Bridges
- Basic bridge operation
- Loops in bridged networks
- Spanning tree protocol

Bridging

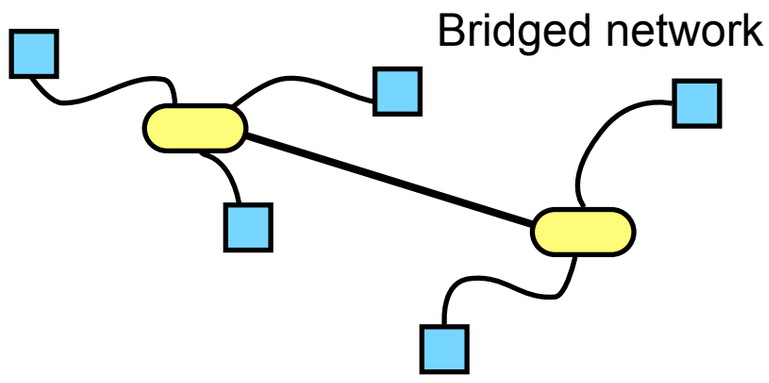
- Link-layer topology evolution:
  - Media access control assumes a single link – on wired networks, a single cable
  - Vulnerable to cable damage



  - A *hub* is a cable in a box – no intelligence
  - Damage to vulnerable cables disconnects only a single host, rather than partitioning the network

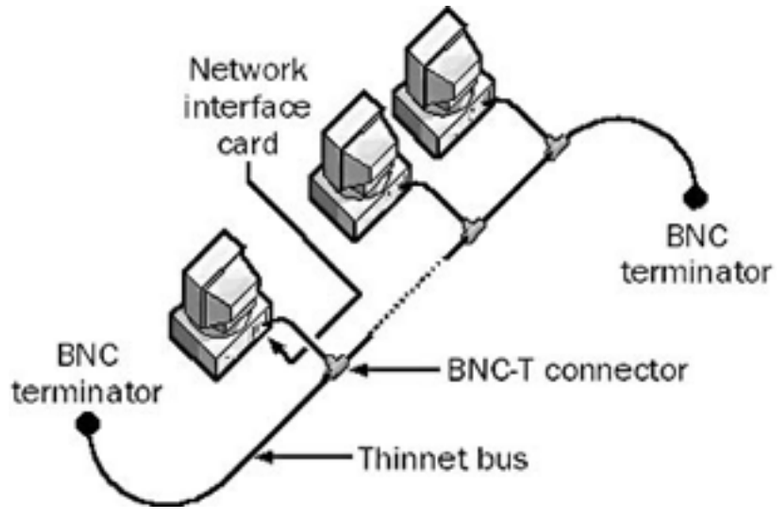


  - A *bridge* is an intelligent device
  - Understands the media access control protocol – joins multiple links together



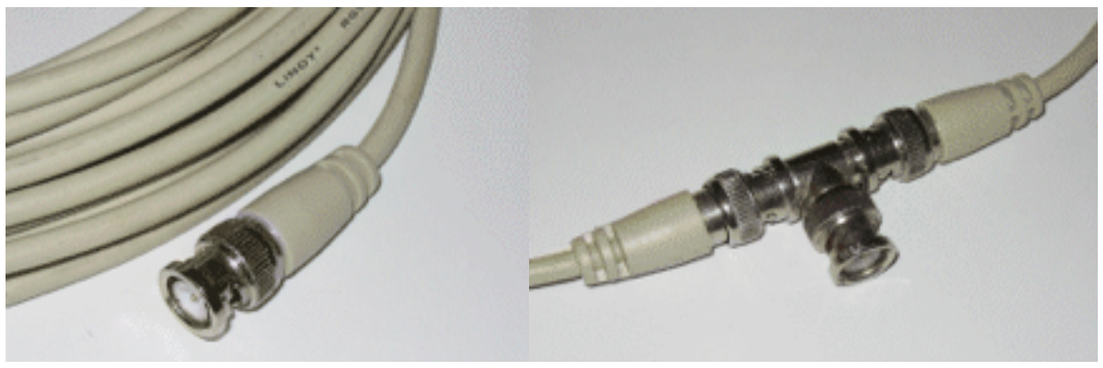
Link Layer Topology Evolution

- Media access control assumes a single link
  - Limits scalability of a system
  - Vulnerable to cable damage
  - Vulnerable to disconnection



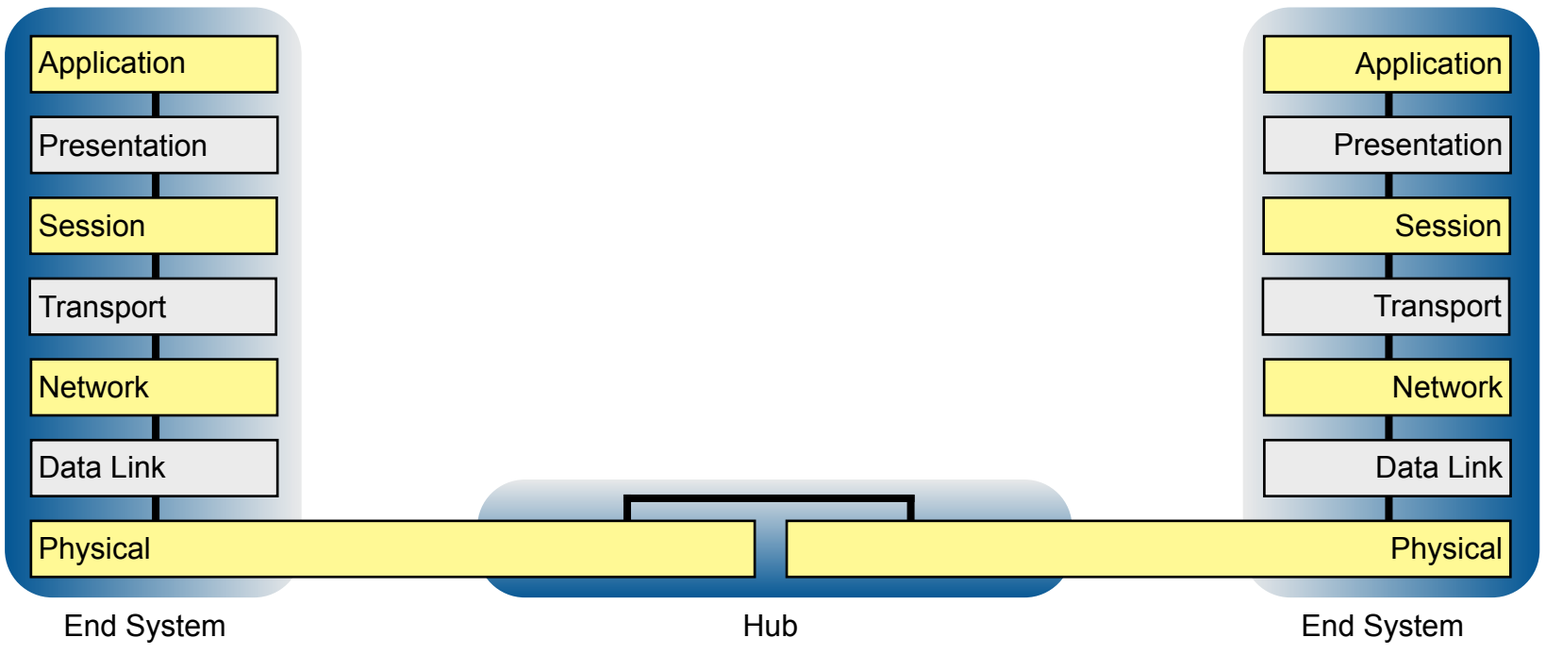
Source: http://www.thenetworkencyclopedia.com/entry/10base2/

Example: 10base2 Ethernet



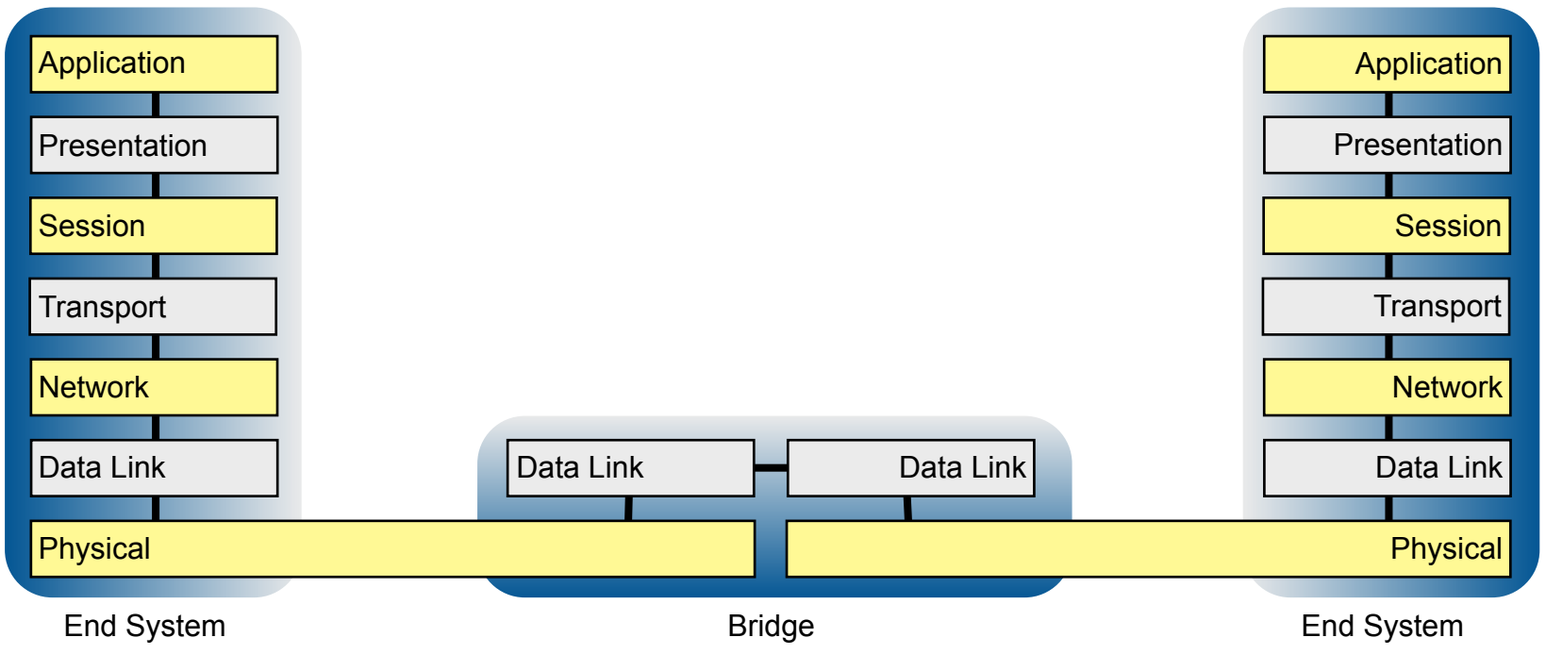
Source: Wikipedia

Extending Link-layer Networks: Hubs



- A *hub* is a physical layer interconnection of links
  - Equivalent to running a longer cable
  - Doesn't improve scalability of the network – but can make physical interconnection of cables/devices easier
  - Largely obsolete – replaced by data link layer bridges

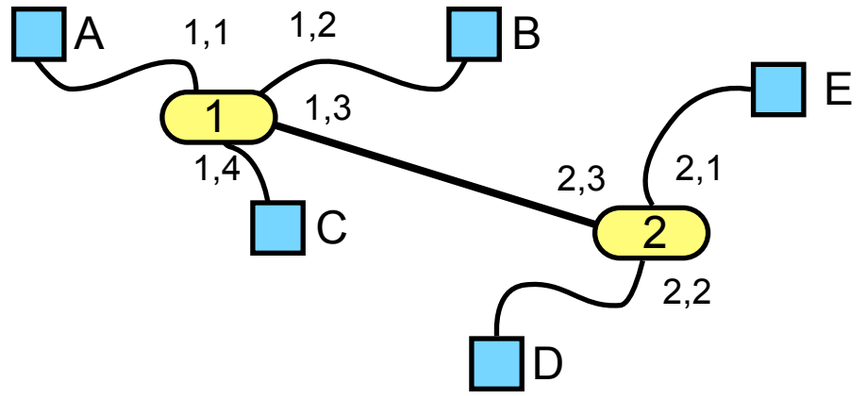
Extending Link-layer Networks: Bridging



- A *bridge* is a data link layer device to interconnect physical networks
  - An intelligent device: understands and processes data link layer frames, identifies location of hosts, forwards only those frames of interest
  - Automatic – needs zero configuration
  - Example: “Ethernet switch”

Basic Bridge Operation

- Learn addresses on each link
  - Observe source addresses of packets
  - *Soft state* time-out allows for graceful response to failure and node mobility

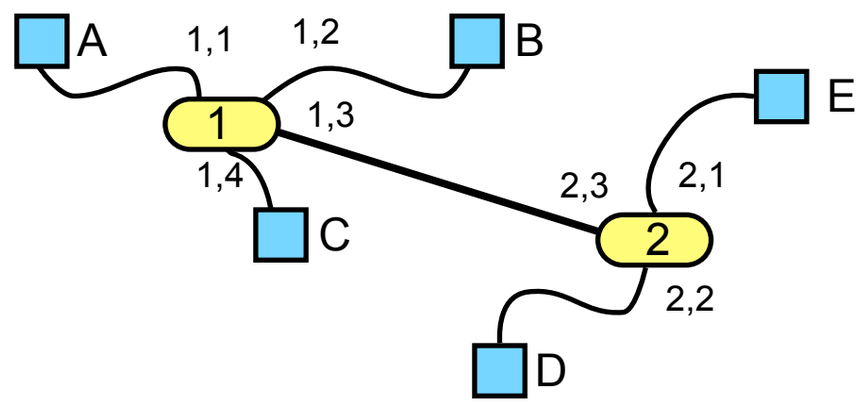


- Forward traffic as appropriate
  - Unicast traffic based on host locations (hash from address to destination link, flooding packets to unknown hosts)
  - Multicast based on group membership
  - Broadcast traffic

Basic Bridge Operation

Bridge 1 state

Link	Host
1,1	
1,2	
1,3	
1,4	



Bridge 2 state

Link	Host
2,1	
2,2	
2,3	

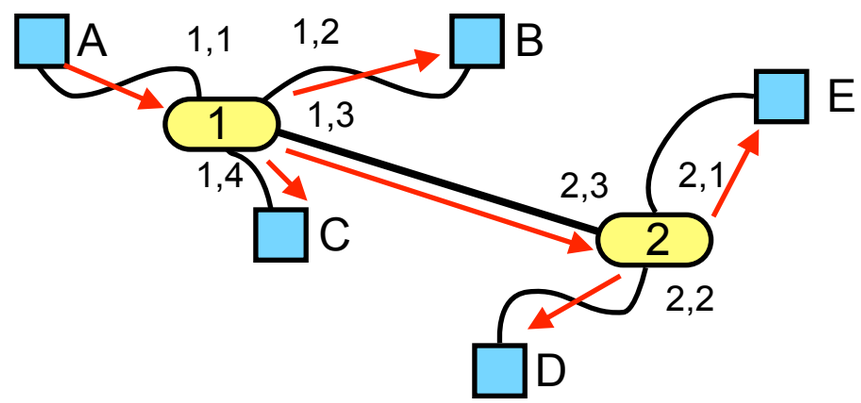
State of network on initialisation:

- Neither bridge knows location of any hosts

Basic Bridge Operation

Bridge 1 state

Link	Host
1,1	A
1,2	
1,3	
1,4	



Bridge 2 state

Link	Host
2,1	
2,2	
2,3	A

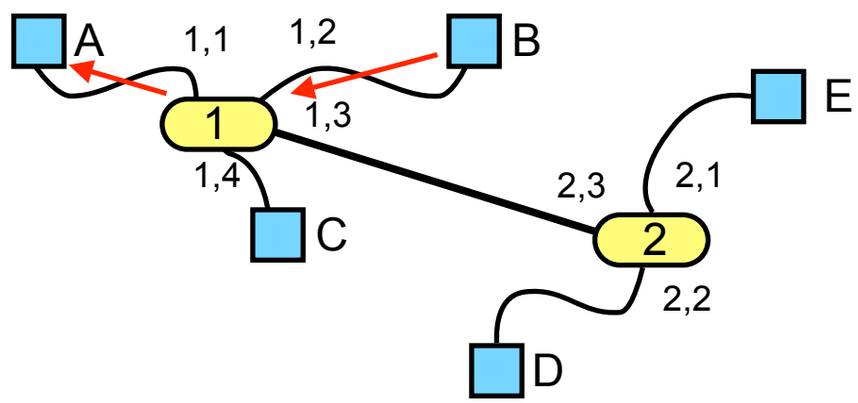
Host A sends packet destined for host B

- Received at bridge 1, which records location of host A
- Location of host B unknown, so bridge 1 floods packet to all outgoing links
- Also received at bridge 2, which doesn't know location of host B, so floods the packet to all outgoing links; records location of host A

Basic Bridge Operation

Bridge 1 state

Link	Host
1,1	A
1,2	B
1,3	
1,4	



Bridge 2 state

Link	Host
2,1	
2,2	
2,3	A

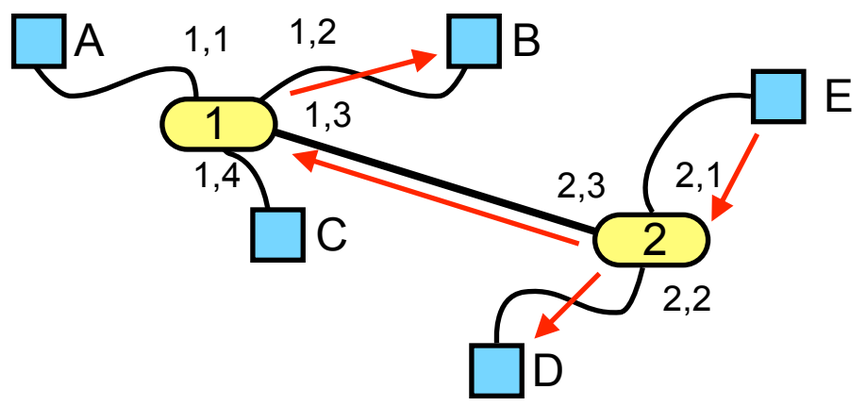
Host B responds with packet destined for host A

- Received at bridge 1, which knows location of host A, and can directly forward the packet without flooding
- Bridge 1 records location of host B

Basic Bridge Operation

Bridge 1 state

Link	Host
1,1	A
1,2	B
1,3	E
1,4	



Bridge 2 state

Link	Host
2,1	E
2,2	
2,3	A

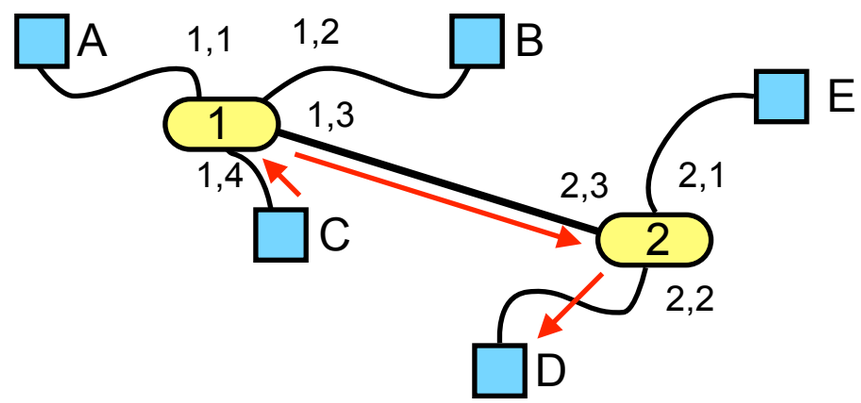
Some time later, host E sends packet destined for host B

- Received at bridge 2, which doesn't know location of B and so floods packet to all outgoing links; records the location of host E
- Received at bridge 1, which *does* know how to reach host B, and directly forwards the packet; records how to reach host E

Basic Bridge Operation

Bridge 1 state

Link	Host
1,1	A
1,2	B
1,3	D,E
1,4	C



Bridge 2 state

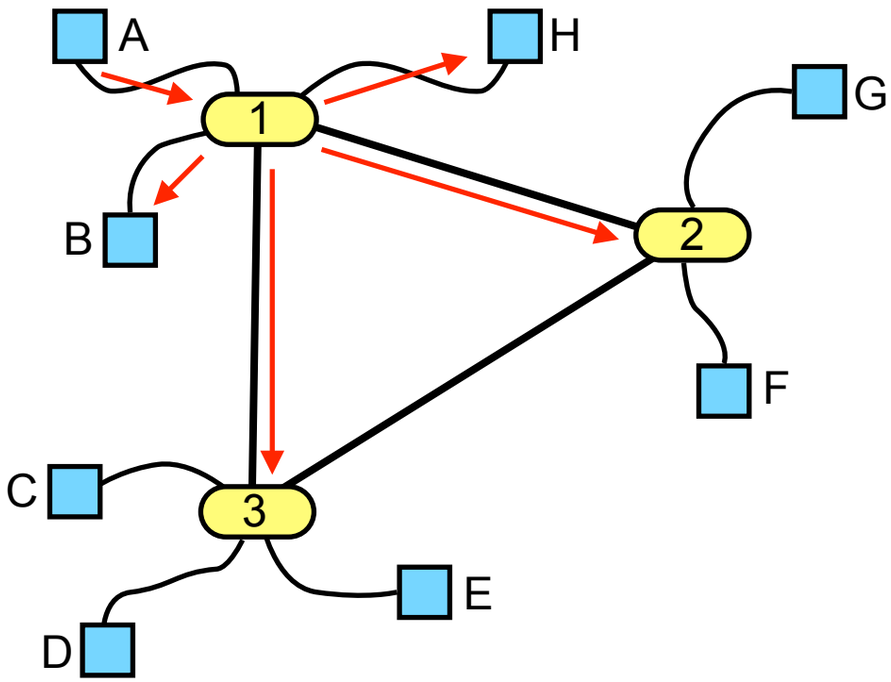
Link	Host
2,1	E
2,2	D
2,3	A,B,C

Over time, bridges learn location of every host, and can forward all packets without flooding

Basic Bridge Operation

- Learning protocol – finds all hosts without needing any configuration
- Flooding ensures connectivity is maintained, even when protocol has no knowledge – performance is never worse than a hub, even when flooding
- Use of *soft state* and timeouts ensures knowledge of failed or disconnected devices disappears
- Poor scalability – every bridge knows about every host

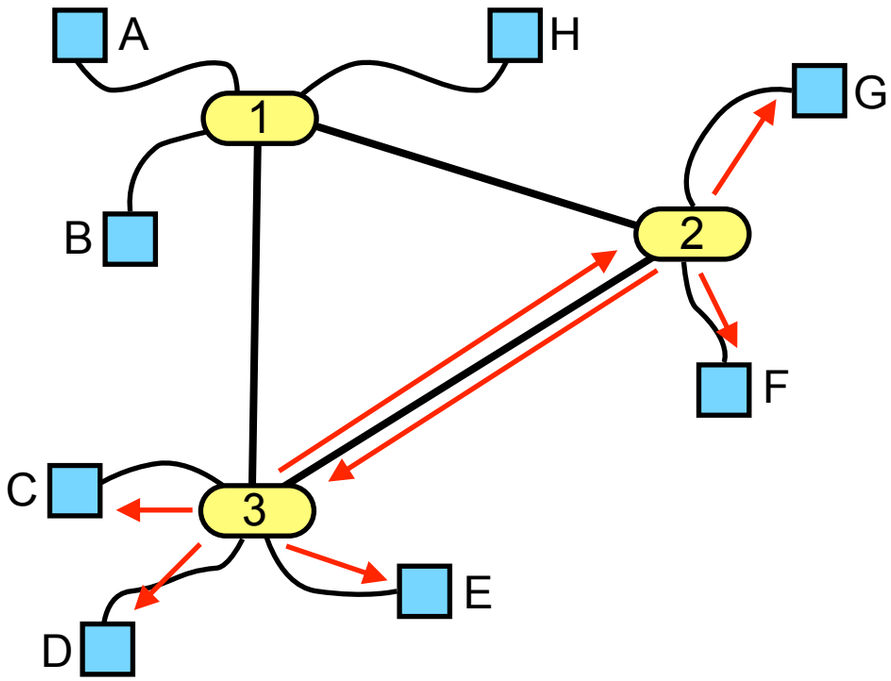
Loops in Bridged Networks



Host A sends packet to host X, that does not exist

- Received at bridge 1, which doesn't know location of X, so floods packet to all outgoing links

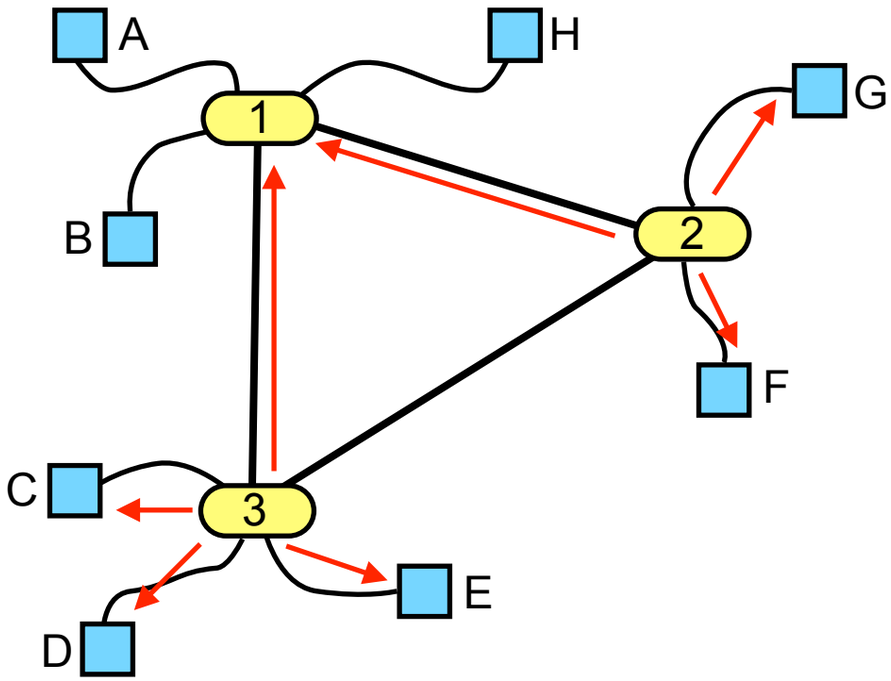
Loops in Bridged Networks



Host A sends packet to host X, that does not exist

- Received at bridge 1, which doesn't know location of X, so floods packet to all outgoing links
- Received at bridges 2 and 3, which also don't know location of X, and so flood packet to all outgoing links

Loops in Bridged Networks

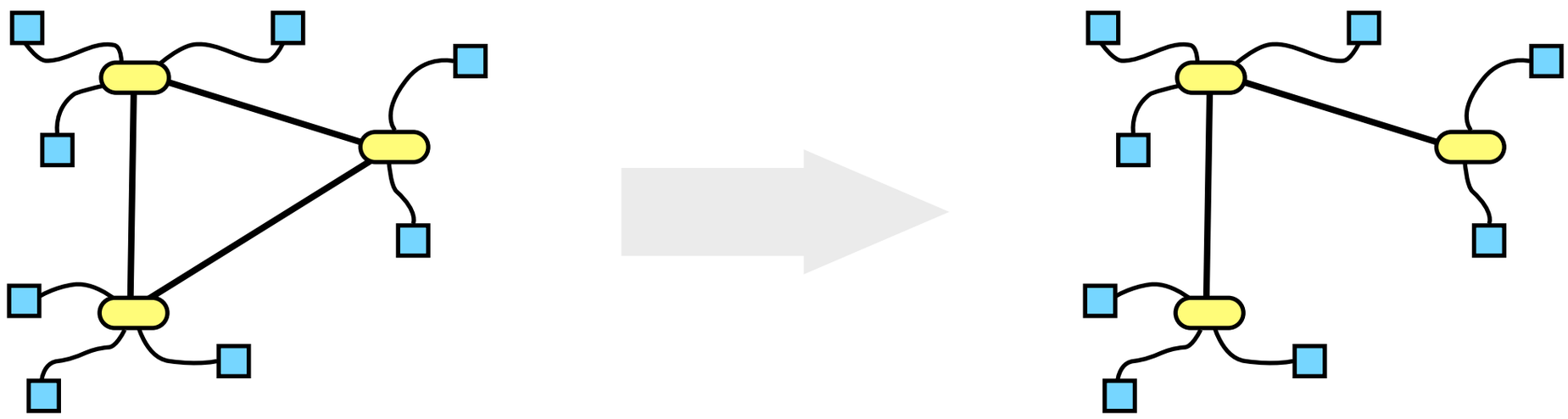


Host A sends packet to host X, that does not exist

- Received at bridge 1, which doesn't know location of X, so floods packet to all outgoing links
- Received at bridges 2 and 3, which also don't know location of X, and so flood packet to all outgoing links
- Packets cross in transit between bridges 2 and 3 – causing a loop unless countermeasures are taken

Loops in Bridged Networks

- Solution: build a *spanning tree* over the network, forward packets along this tree
  - Model network as an undirected graph, *G*
  - A *spanning tree* over that graph is a *tree* comprising all the vertices and some of the edges of *G*
  - Edges are removed to eliminate loops, leaving minimal set of edges that still connect all vertices



Spanning Tree Algorithm

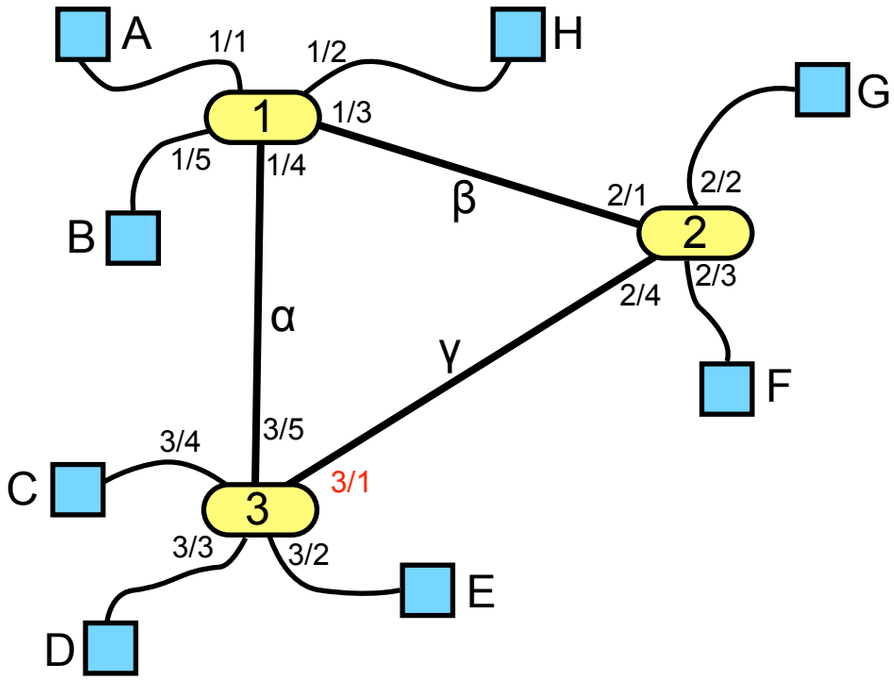
- Distributed algorithm to build spanning tree developed by Radia Perlman



Source: Sun Microsystems

  - Each bridge has a globally unique address
    - Bridge with numerically lowest address is the *root bridge*
    - Each bridge periodically informs it's neighbours what it thinks is address of the root bridge – potentially making them update what they think it the root address
  - Determine *root port* (port with shortest path to root bridge) of each bridge, except the root bridge
  - For each LAN, select *designated bridge* for the LAN (this is the bridge with the shortest path to the root bridge; tie-break based on address)
    - The port connecting the designated bridge to the LAN is a *designated port*
  - Enable all root ports and all designated ports, and disable all other ports – forward traffic using only the enabled ports

Spanning Tree Algorithm



Bridge 1 is the *root bridge* (lowest address)

The *root ports* are 2/1 and 3/5

The *designated bridges* are:

- Bridge 1 for hosts A, B, and H and links α and β
- Bridge 2 for hosts F and G and link γ
- Bridge 3 for hosts C, D, and E

The designated ports are:

- Bridge 1: 1/1, 1/2, 1/3, 1/4, and 1/5
- Bridge 2: 2/2, 2/3, and 2/4
- Bridge 3: 3/2, 3/3, and 3/4

Port 3/1 is neither a root or designated port and is disabled, all others are enabled

Algorhyme

“I think that I shall never see
A graph more lovely than a tree.

A tree whose crucial property
Is loop-free connectivity.

First the Root must be selected.
By ID it is elected.

Least cost paths from Root are traced.
In the tree these paths are placed.

A mesh is made by folks like me.
Then bridges find a spanning tree.”

R. Perlman, “An algorithm for distributed computation of a spanning tree in an extended LAN”, Proc. ACM SIGCOMM ’85, Vancouver, BC, Canada, September 1985, DOI: 10.1145/319056.319004. http://dl.acm.org/citation.cfm?id=319004