

CSCI-1680

Building Links and (Local) Networks

Nick DeMarinis

Administrivia

- Snowcast due tomorrow (Wednesday, 9/28) by 11:59pm
- Office hours today
 - Today (Tuesday, 9/27)
 - 1-3pm: Individual/Zoom (Nick)
 - 3-5pm: Collaborative, CIT316 (Nick)
 - 5-7pm: Collaborative, CIT201 (Rhea, Nathan)
 - Tomorrow (Wednesday 9/28)
 - Probably 1-3pm? (Nick)
 - 3-5pm: Individual (Nathan)

Administrivia

- We are here to help!

About ~~autograding~~ auto-testing:

- Not sure why a test is failing? Check Ed.
 - If your test isn't mentioned, please post!
 - There may be known issues
 - Also test with the reference!

Remember: $\text{your grade} := (\# \text{ passing tests}) * C$

Administrivia

- Thursday, 9/29: IP project out
 - You will work in groups of two
 - We will send a form where you can specify your group, or ask to be matched to a group
 - Matching happens based on language, in-person/remote, etc.

Last time

Physical layer

- Different physical media => different requirements and properties
- How to send *bits* over different physical media

Today

How do we build a network with this?

- Link layer challenges: shared media
- Case study: Ethernet (and Wifi)
 - *Network interfaces*: How you interact with the link layer
- How switches work

What does “link layer” mean?

Application

Service: user-facing application.
Application-defined messages

Transport

Service: multiplexing applications
Reliable byte stream to other node (TCP),
Unreliable datagram (UDP)

Network

Service: move packets to any other node in the network
Internet Protocol (IP)

Link

Service: move frames to other node across link.
May add reliability, medium access control

Physical

Service: move bits to other node across link

What does “link layer” mean?

- How to talk to your “neighbors”
- Can use this to build a “small” network
 - Within a building, campus
 - Can build a Local Area Network (LAN)
- Note: hard to talk about without mentioning IP (more on this later)

Link layer challenges

Last time, we talked about a channel...

Link layer challenges

Now, the channel may be shared among multiple hosts!

Medium Access Control (MAC)

Medium Access Control

- Control access to shared physical medium
 - E.g., who can talk when?
 - If everyone talks at once, no one hears anything]
- Two conflicting goals
 - Maximize utilization when one node sending
 - Approach $1/N$ allocation when N nodes sending

High-level: MAC approaches

Partitioned Access: divide the channel into fixed slots

- Time Division Multiple Access (TDMA)
- Frequency Division Multiple Access (FDMA)
- Code Division Multiple Access (CDMA)

High-level: MAC approaches

Random Access: senders “acquire” the channel for a time

- ALOHA/ Slotted ALOHA
- Carrier Sense Multiple Access / Collision Detection (CSMA/CD)
- Carrier Sense Multiple Access / Collision Avoidance (CSMA/CA)
- RTS/CTS (Request to Send/Clear to Send)
- Token-based

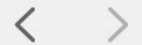
Why does this matter?

- Different types of links solve these problems differently
 - Ethernet (wired) vs. Wifi (wireless)
 - Affects throughput, reliability, etc.
 - Relies on physical layer characteristics (encoding, modulation, etc)
- Useful to understand why modern link layer operates the way it does


Ethernet: how the link layer works

Preamble: Interfaces


















- Every hosts has some number of network *interfaces*
 - Software abstraction of some physical network device
- Common interfaces
 - Loopback: Virtual, only for local host
 - Ethernet, Wifi, etc...



Network

 Search


Location: Automatic 

-  **Wi-Fi**
 Connected
-  Bluetooth PAN
 Not Connected
-  USB 10/1...1000 LAN
 Not Connected
-  Thunder...rnet Slot 2
 Not Connected
-  Thunder...rnet Slot 1
 Not Connected
-  Thunderbolt Bridge
 Not Connected
-  ProtonVPN
 Not Connected
-   

Status: **Connected**

Turn Wi-Fi Off

Wi-Fi is connected to RLAB and has the IP address 138.16.161.155.

Network Name: RLAB 


☒ Automatically join this network

☒ Ask to join Personal Hotspots

☒ Ask to join new networks

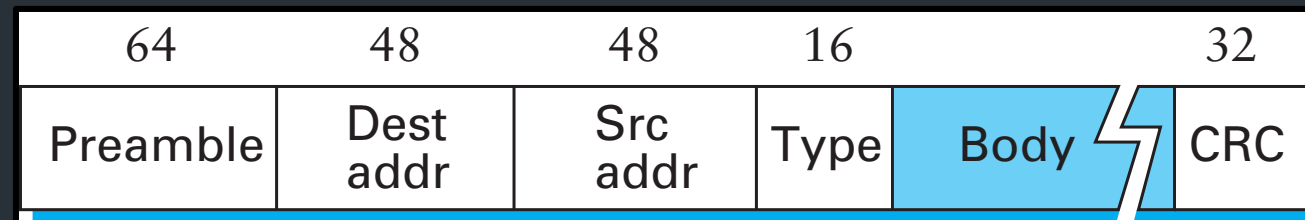
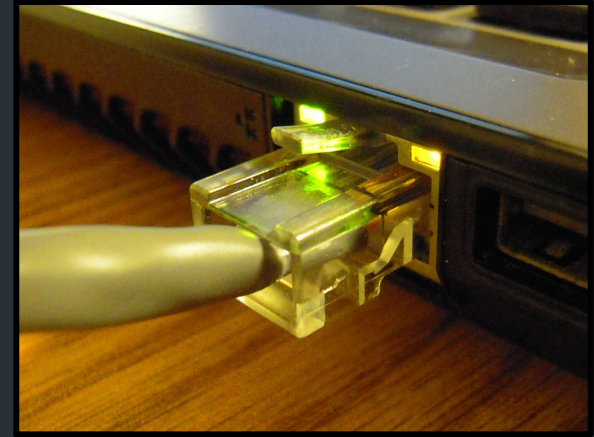
Known networks will be joined automatically. If no known networks are available, you will be asked before joining a new network.

☒ Show Wi-Fi status in menu bar

Advanced... 

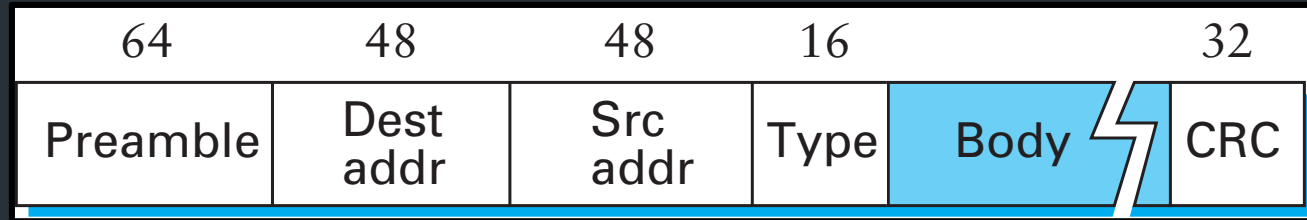
Ethernet (IEEE 802.3)

- Dominant wired LAN technology
- Original version (1983): 10Mbps
- Now: 1Gbps (1000BASE-T), 10Gbps, ...
- CSMA/CD: Carrier Sense / Multiple Access / Collision Detection
- L1: Manchester encoding



The basic idea

Ethernet Addressing



Globally unique, 48-bit unicast address per adapter

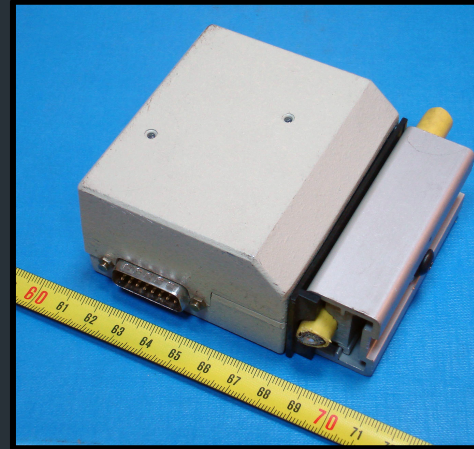
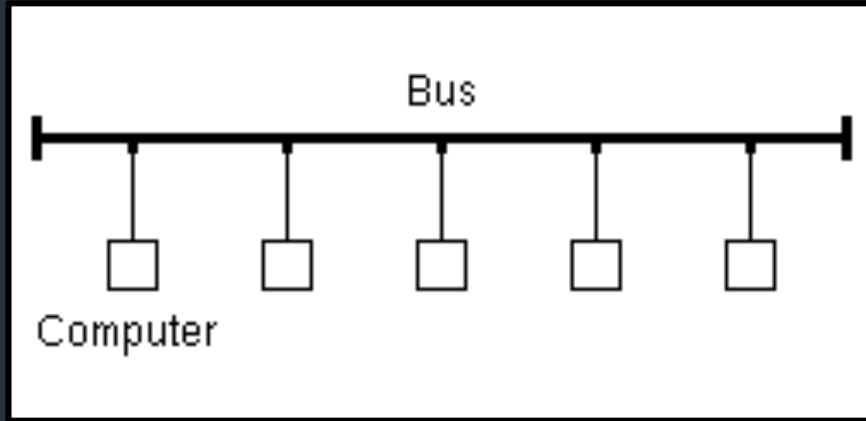
- Example: 00:1c:43:00:3d:09 (Samsung adapter)
- First 24 bits: Registered to manufacturers
- <http://standards.ieee.org/develop/regauth/oui/oui.txt>

Other protocols have adopted this address format (eg. Wifi, Bluetooth, ...)

- Nowadays, we call them “mac addresses” or “hardware addresses”

Ethernet's evolution

Originally, a shared medium with all hosts



- Basic idea: all hosts can see all frames, read a frame if it matches your hardware address
- Implications?

Classical Ethernet: Problems

- Problem: shared medium, all hosts in the same "collision domain"
- Transmit algorithm
 - If line is idle, transmit immediately
 - Upper bound message size of 1500 bytes
 - If line is busy: wait until idle and transmit immediately
- Generally possible to detect collisions

When to transmit again?

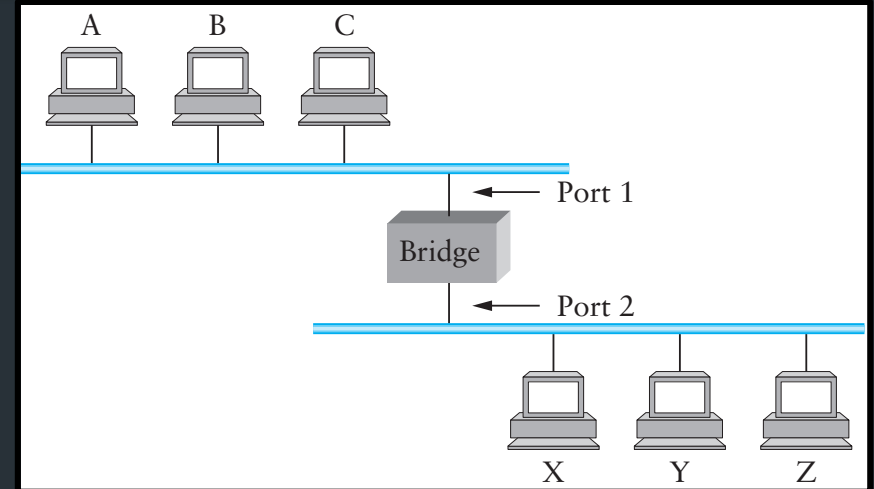
- Delay and try again: exponential backoff
- n th time: $k \times 51.2\mu\text{s}$, for $k = U\{0..(2^{\min(n,10)}-1)\}$
 - 1st time: 0 or $51.2\mu\text{s}$
 - 2nd time: 0, 51.2 , 102.4 , or $153.6\mu\text{s}$
- Give up after several times (usually 16)
- Exponential backoff is a useful, general technique

Ethernet Recap

- Service provided: send frames among stations with specific addresses
- Addresses are just names, no topology information
 - Special broadcast and multicast addresses
- All nodes in the same “broadcast domain”
 - Is this what we want?

Bridges and Extended LANs

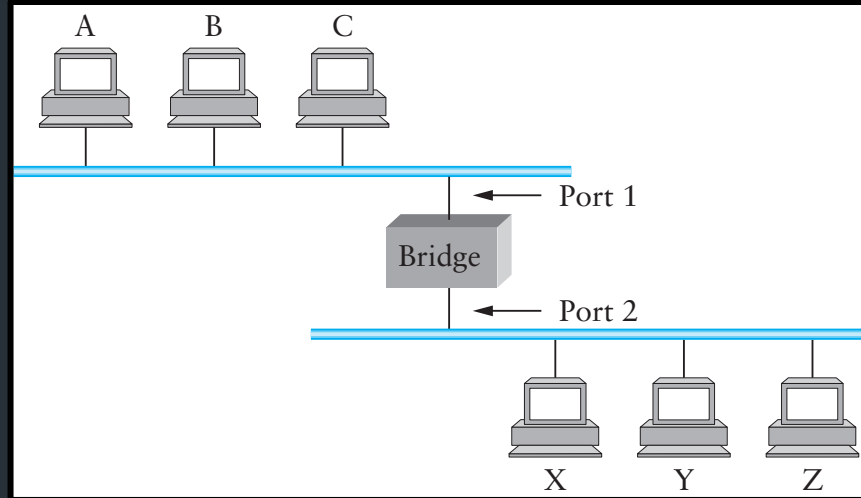
- Single Ethernet collision domain has limitations
 - Limits performance, distance, ...
- Next step: separate collision domains with *bridges*
 - Operates on Ethernet addresses
 - Forwards packets from one collision domain to others
- Modern ethernet uses switches: all hosts directly connected to a bridge



Destinations for packets

- Unicast: forward with filtering
- Broadcast: always forward
- Multicast: always forward or learn groups
- Can try to limit how we direct packets to a destination

Learning Bridges/Switches



- Idea: don't forward a packet where it isn't needed
 - If you know recipient is not on that port
- Learn hosts' locations based on source addresses
 - Build a table as you receive packets
 - Table is a *cache*: if full, evict old entries. Why is this fine?
- Table says when *not* to forward a packet
 - Doesn't need to be complete for *correctness*

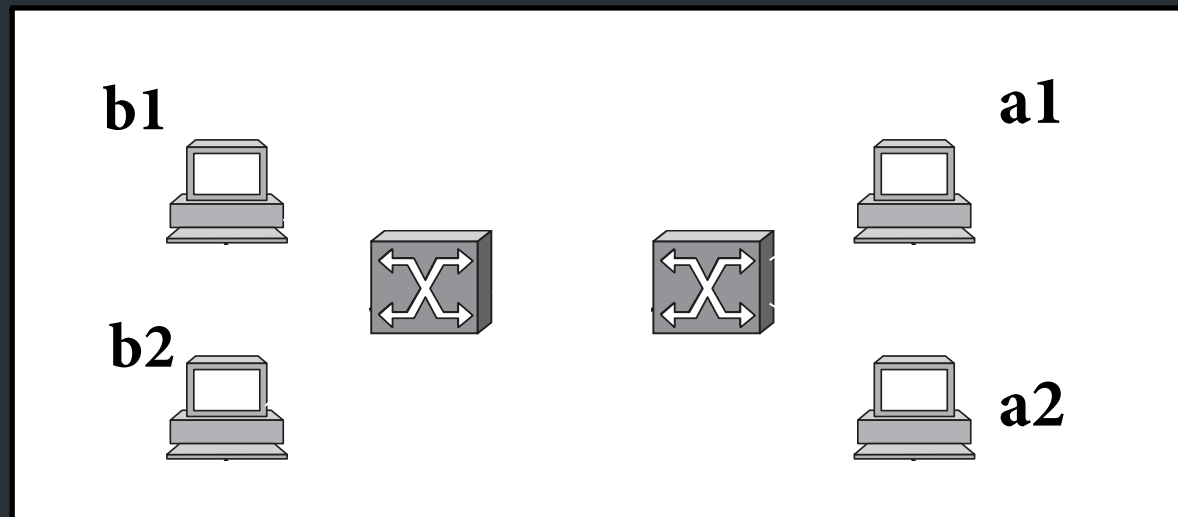
Attack on a Learning Switch

- Eve: wants to sniff all packets sent to Bob
- Same segment: easy (shared medium)
- Different segment on a learning bridge: hard
 - Once bridge learns Bob's port, stop broadcasting
- How can Eve force the bridge to keep broadcasting?
 - Flood the network with frames with spoofed src addr!

VLANs

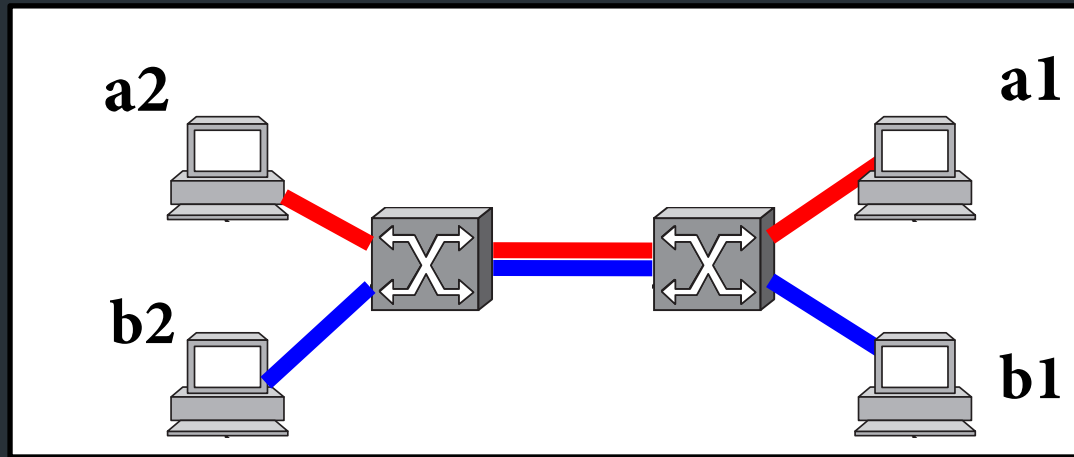
Consider: Company network, A and B departments

- Broadcast traffic does not scale
- May not *want* traffic between the two departments
- Topology has to mirror physical locations
- What if employees move between offices?



VLANs

- Solution: Virtual LANs
 - Assign switch ports to a VLAN ID (color)
 - Isolate traffic: only same color
 - Trunk links may belong to multiple VLANs
 - Encapsulate packets: add 12-bit VLAN ID
- Easy to change, no need to rewire



We did not cover these...

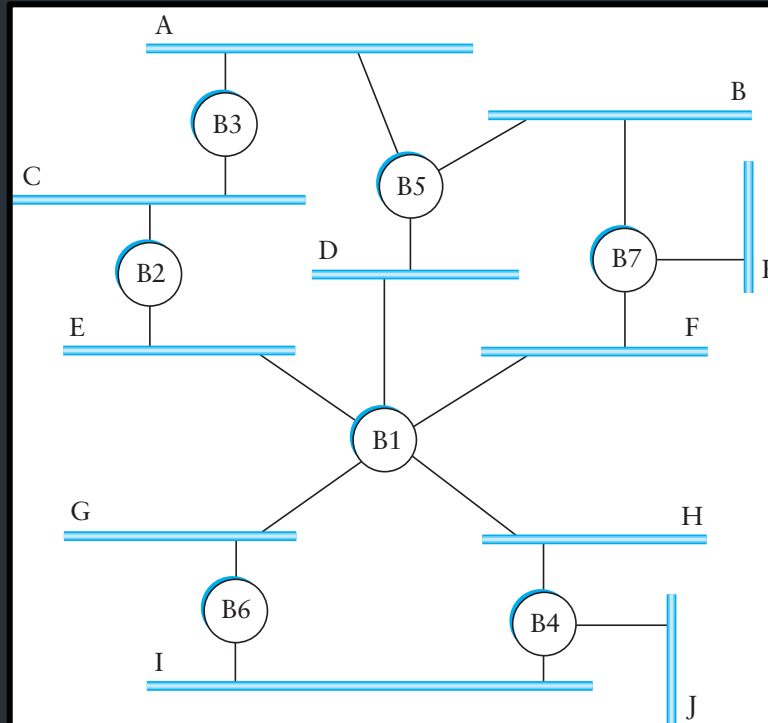
Coming Up

- Connecting multiple networks: IP and the Network Layer

Dealing with Loops

Problem: people may create loops in LAN!

- Accidentally, or to provide redundancy
- Don't want to forward packets indefinitely



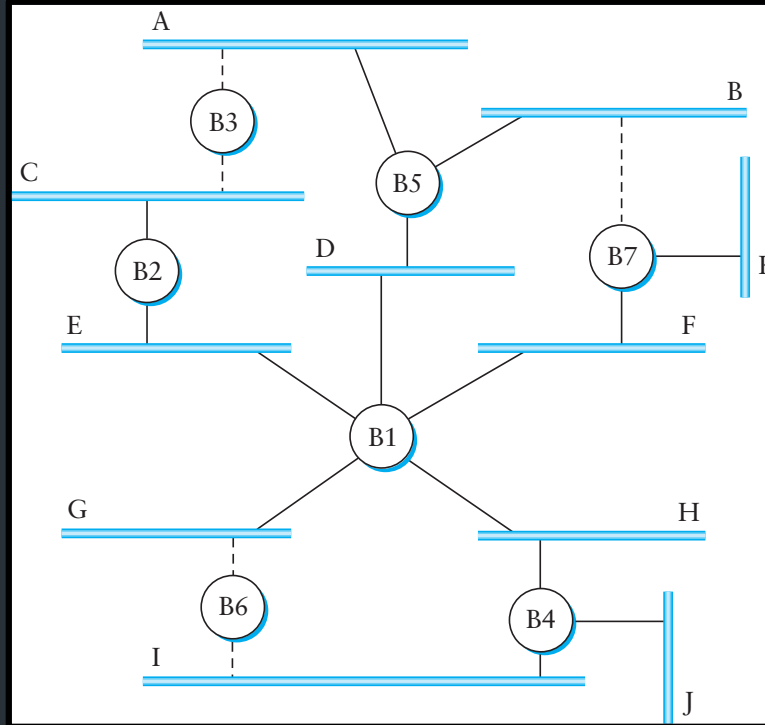
Enter Radia Perlman

"...we have designed an algorithm that allows the extended network to consist of an arbitrary topology. (...) The algorithm (...) computes a subset of the topology that connects all LANs yet is loop-free (a spanning tree)."

Perlman, Radia (1985). "An Algorithm for Distributed Computation of a Spanning Tree in an Extended LAN". *ACM SIGCOMM Computer Communication Review*. **15** (4): 44–53. [doi:10.1145/318951.319004](https://doi.org/10.1145/318951.319004)



Spanning Tree



- Need to disable ports, so that no loops in network
- Like creating a spanning tree in a graph
 - View switches and networks as nodes, ports as edges

Distributed Spanning Tree Algorithm

- Every bridge has a unique ID (Ethernet address)
- Goal:
 - Bridge with the smallest ID is the root
 - Each segment has one designated bridge, responsible for forwarding its packets towards the root
 - Bridge closest to root is designated bridge
 - If there is a tie, bridge with lowest ID wins

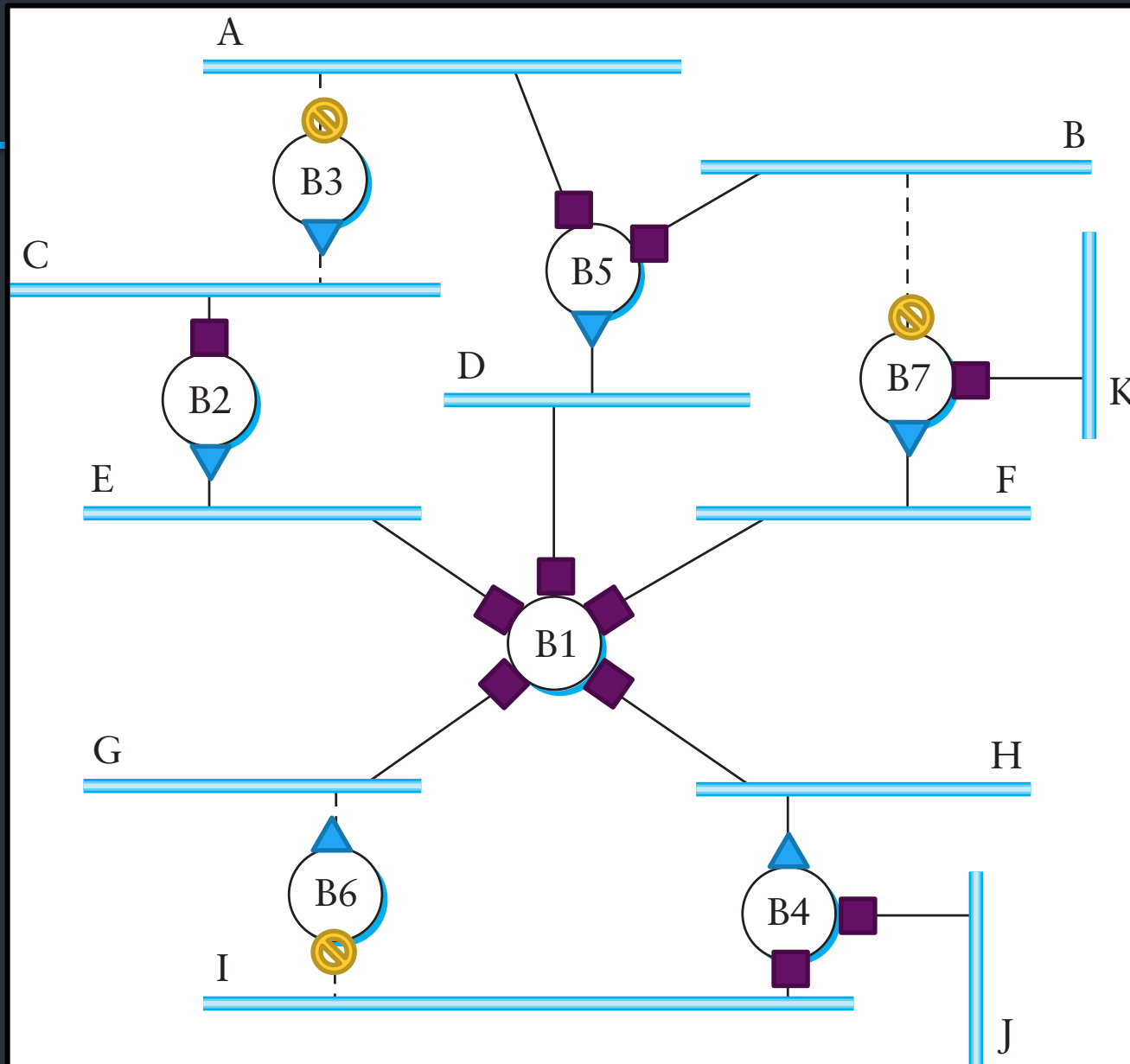
Spanning Tree Protocol

- Send message when you think you are the root
- Otherwise, forward messages from best known root
 - Add one to distance before forwarding
 - Don't forward over *discarding ports* (see next slide)
- Spanning Tree messages contain:
 - ID of bridge sending the message
 - ID sender believes to be the root
 - Distance (in hops) from sender to root
- Bridges remember best config msg on each port
- In the end, only root is generating messages

Spanning Tree Protocol (cont.)

- Forwarding and Broadcasting
- Port states*:
 - **Root port**: a port the bridge uses to reach the root
 - **Designated port**: the lowest-cost port attached to a single segment
 - If a port is not a root port or a designated port, it is a **discarding port**.

* In a later protocol RSTP, there can be ports configured as backups and alternates.



- Root Port
- Designated Port
- Discarding Port

Algorhyme

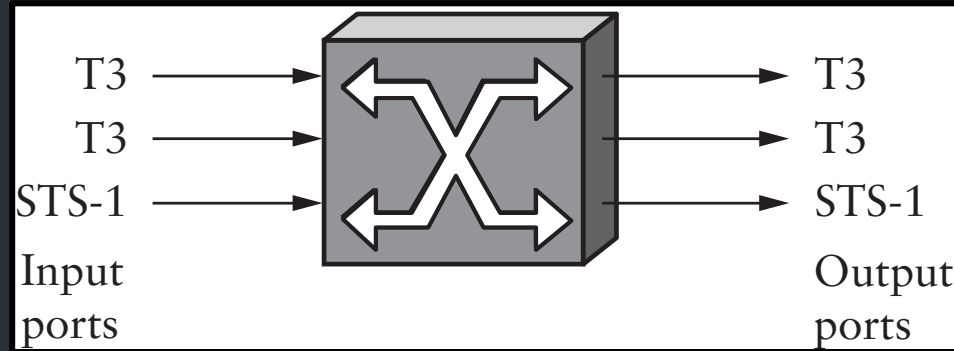
I think that I shall never see
a graph more lovely than a tree.
A tree whose crucial property
is loop-free connectivity.
A tree that must be sure to span
so packet can reach every LAN.
First the root must be selected.
By ID, it is elected.
Least cost paths from root are traced.
In the tree, these paths are placed.
A mesh is made by folks like me,
then bridges find a spanning tree.

Radia Perlman

Limitations of Bridges

- Scaling
 - Spanning tree algorithm doesn't scale
 - Broadcast does not scale
 - No way to route around congested links, *even if path exists*
- May violate assumptions
 - Could confuse some applications that assume single segment
 - Much more likely to drop packets
 - Makes latency between nodes non-uniform
 - Beware of transparency

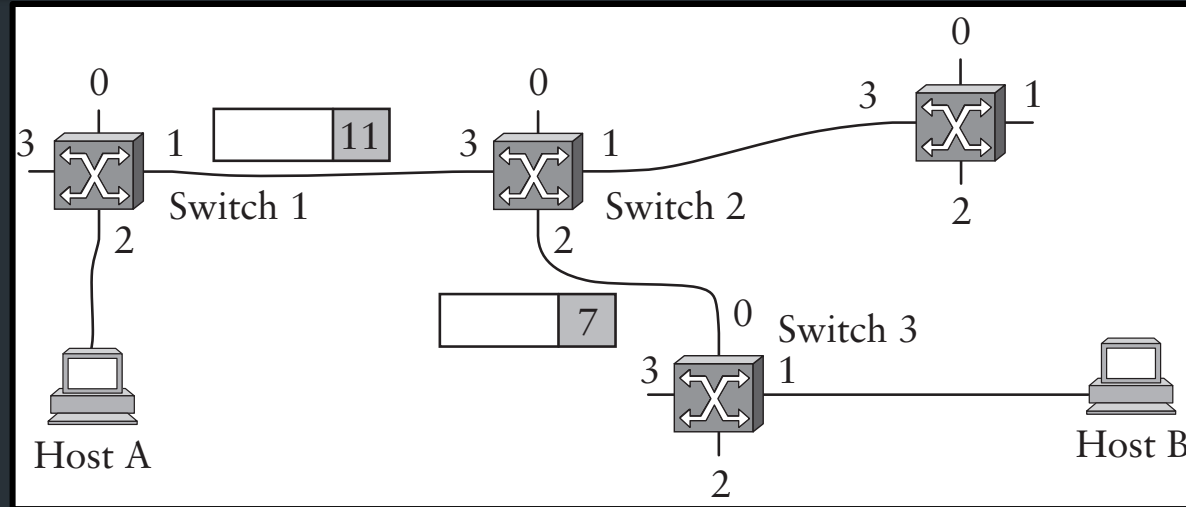
Switching



Switches must be able to, given a packet, determine the outgoing port

- 3 ways to do this:
 - Virtual Circuit Switching
 - Datagram Switching
 - Source Routing

Virtual Circuit Switching



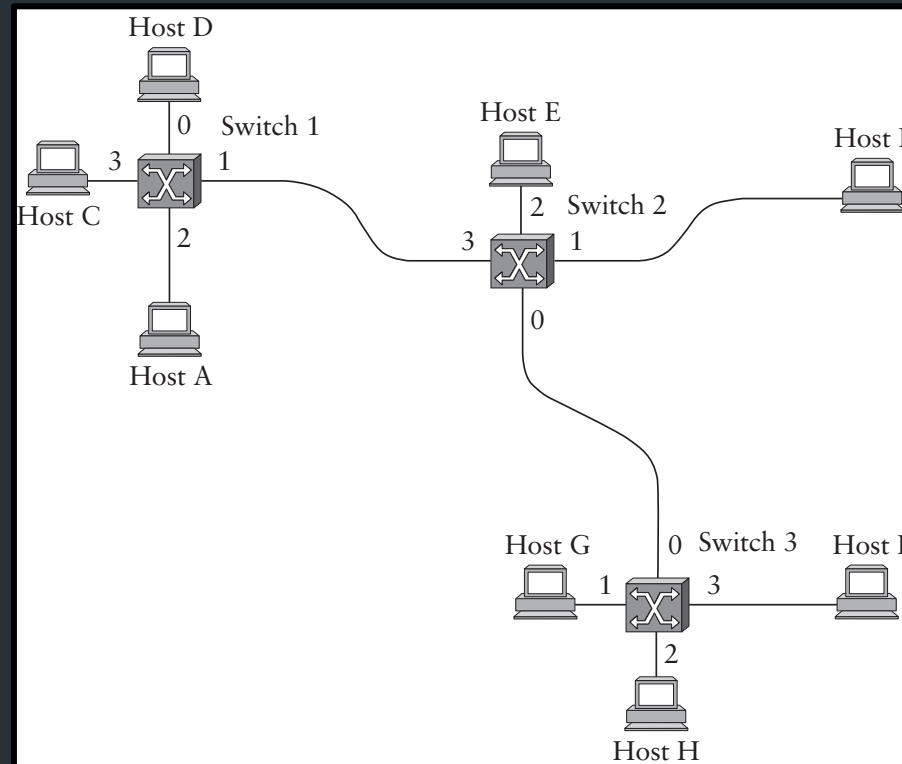
- Explicit set-up and tear down phases
 - Establishes Virtual Circuit Identifier on each link
 - Each switch stores VC table
- Subsequent packets follow same path
 - Switches map [in-port, in-VCID] : [out-port, out-VCID]
- Also called *connection-oriented* model

Virtual Circuit Model

- Requires one RTT before sending first packet
- Connection request contain full destination address, subsequent packets only small VCI
- Setup phase allows reservation of resources, such as bandwidth or buffer-space
 - Any problems here?
- If a link or switch fails, must re-establish whole circuit
- Example: ATM, MPLS

Datagram Switching

- Each packet carries destination address
- Switches maintain address-based tables
 - Maps [destination address]:[out-port]
- Also called *connectionless* model



Switch 2

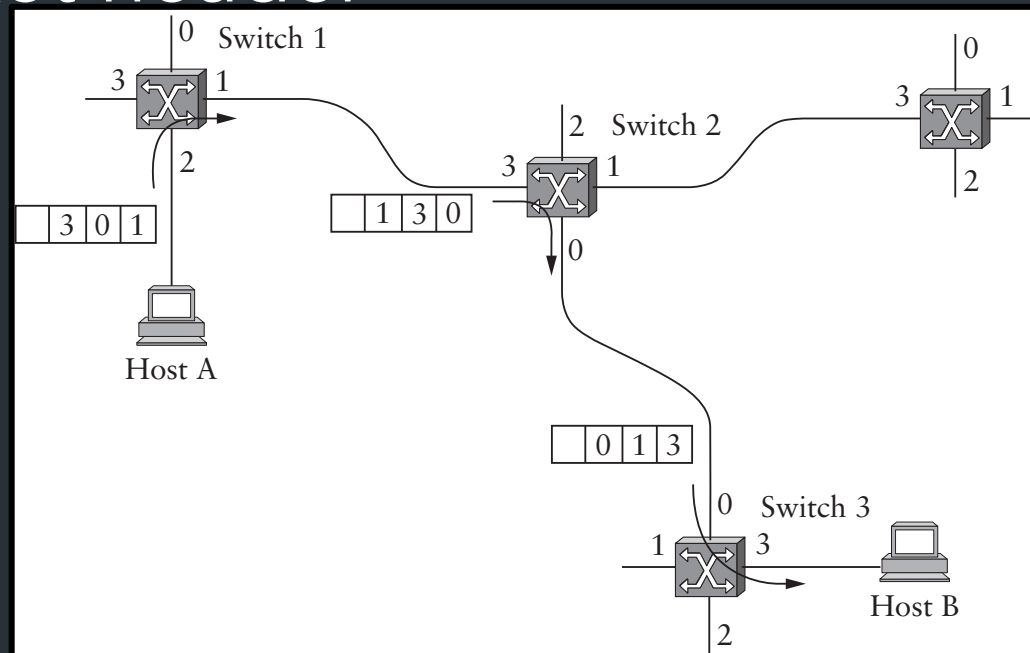
Addr	Port
A	3
B	0
C	3
D	3
E	2
F	1
G	0
H	0

Datagram Switching

- No delay for connection setup
- Source can't know if network can deliver a packet
- Possible to route around failures
- Higher overhead per-packet
- Potentially larger tables at switches

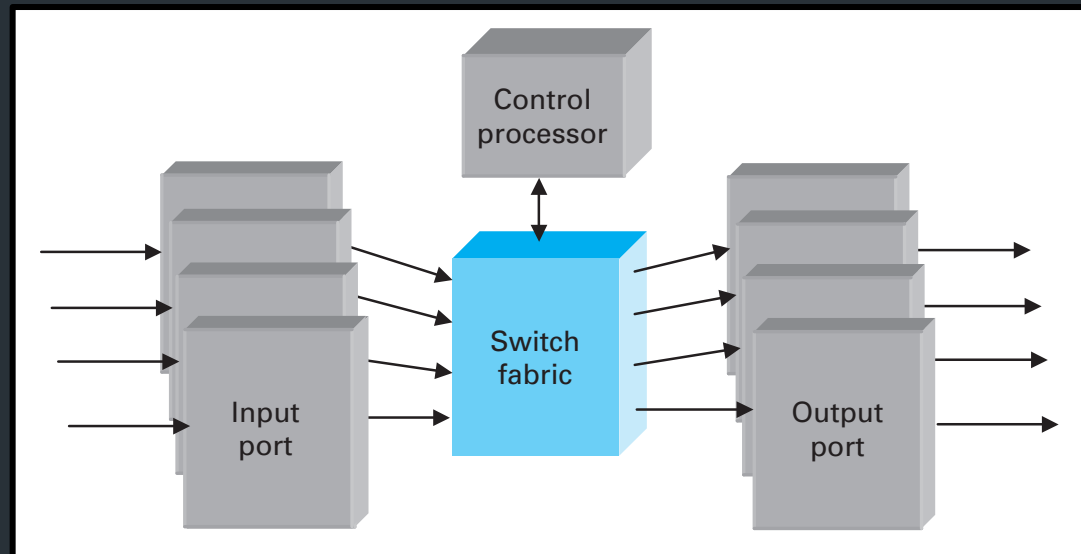
Source Routing

- Packets carry entire route: ports
- Switches need no tables!
 - But end hosts must obtain the path information
- Variable packet header



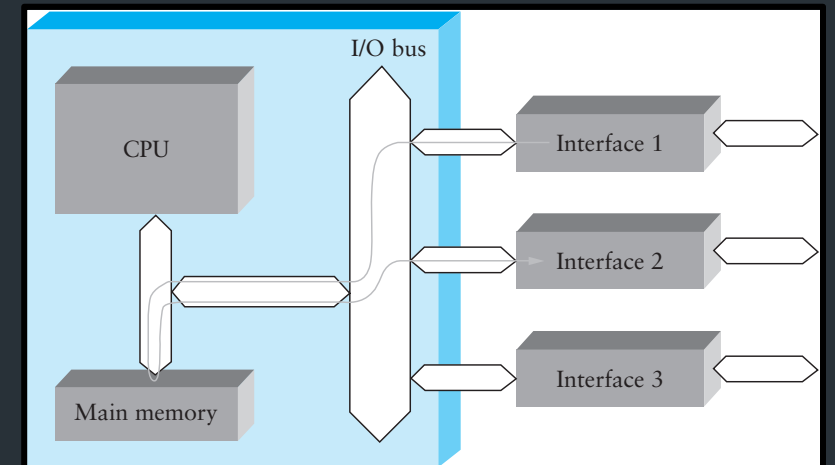
Generic Switch Architecture

- Goal: deliver packets from input to output ports
- Three potential performance concerns:
 - Throughput in bytes/second
 - Throughput in packets/second
 - Latency



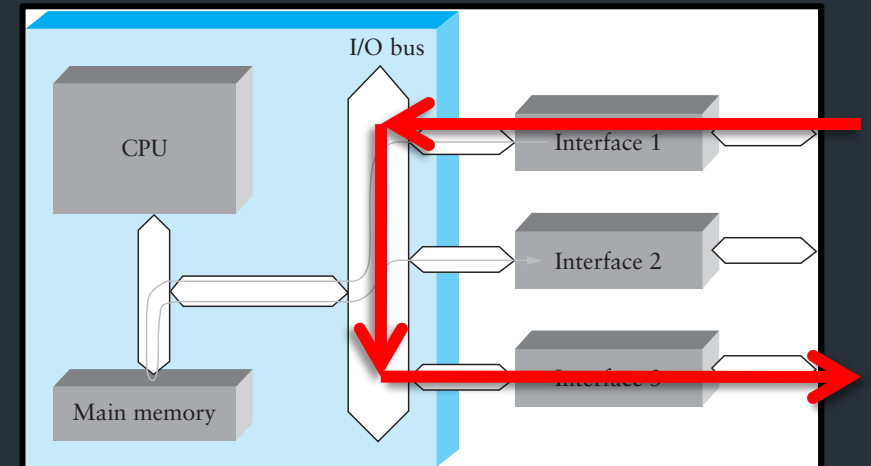
Shared Memory Switch

- 1st Generation – like a regular PC
 - NIC DMA's packet to memory over I/O bus
 - CPU examines header, sends to destination NIC
 - I/O bus is serious bottleneck
 - For small packets, CPU may be limited too
 - Typically < 0.5 Gbps



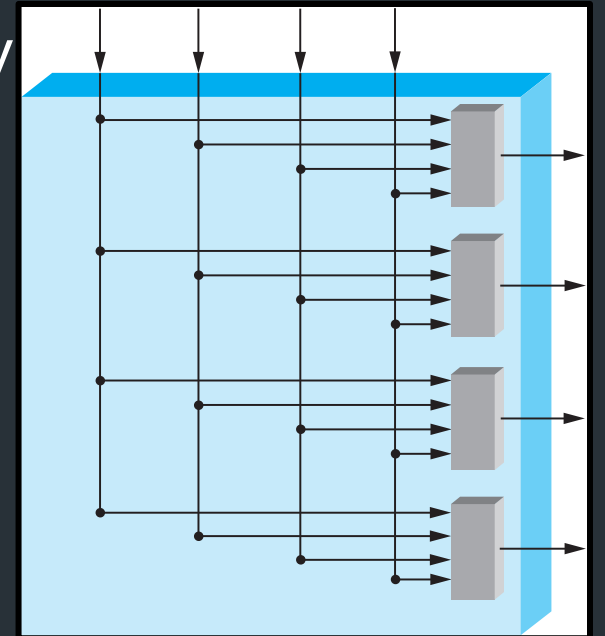
Shared Bus Switch

- 2st Generation
 - NIC has own processor, cache of forwarding table
 - Shared bus, doesn't have to go to main memory
 - Typically limited to bus bandwidth
 - (Cisco 5600 has a 32Gbps bus)



Point to Point Switch

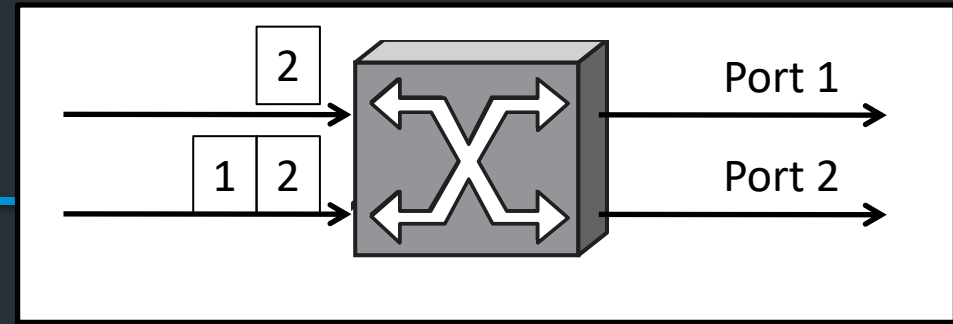
- 3rd Generation: overcomes single-bus bottleneck
- Example: Cross-bar switch
 - Any input-output permutation
 - Multiple inputs to same output requires trickery
 - Cisco 12000 series: 60Gbps



Cut through vs. Store and Forward

- Two approaches to forwarding a packet
 - Receive a full packet, then send to output port
 - Start retransmitting as soon as you know output port, before full packet
- Cut-through routing can greatly decrease latency
- Disadvantage
 - Can waste transmission (classic *optimistic* approach)
 - CRC may be bad
 - If Ethernet collision, may have to send runt packet on output link

Buffering

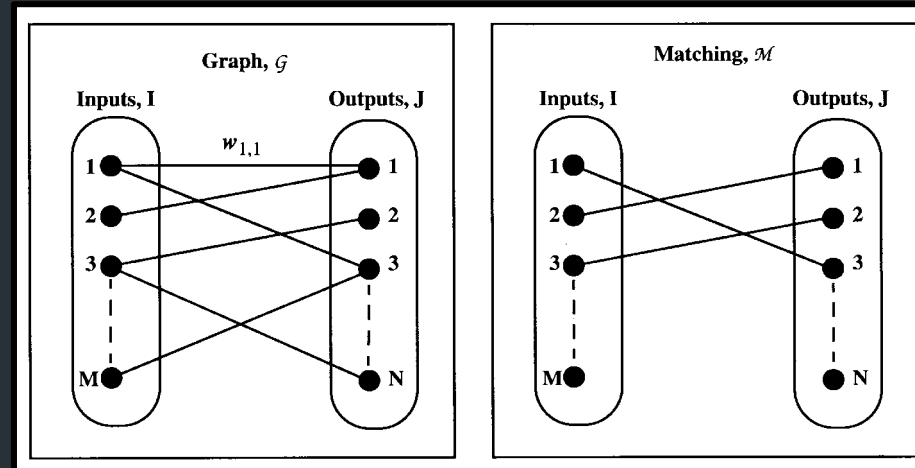
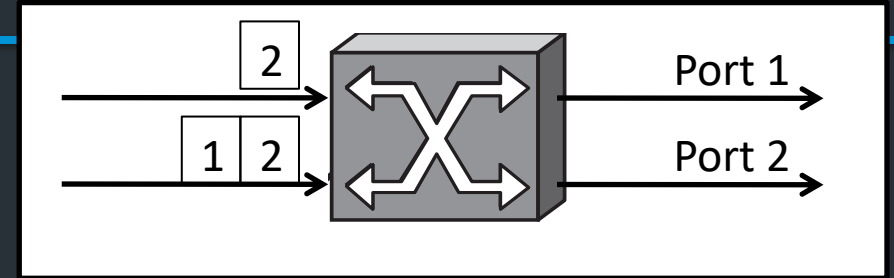


- Buffering of packets can happen at input ports, fabric, and/or output ports
- Queuing discipline is very important
- Consider FIFO + input port buffering
 - Only one packet per output port at any time
 - If multiple packets arrive for port 2, they may block packets to other ports that are free
 - *Head-of-line blocking*: can limit throughput to ~ 58% under some reasonable conditions*

* For independent, uniform traffic, with same-size frames

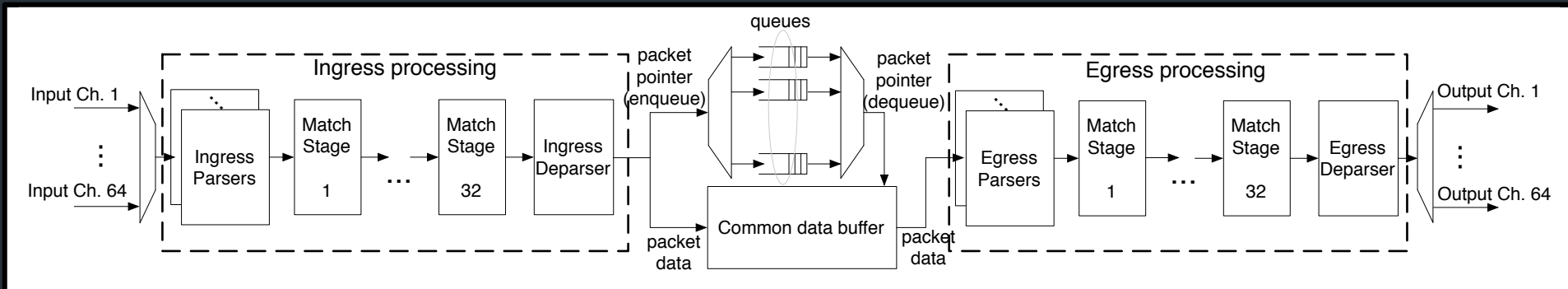
Head-of-Line Blocking

- Solution: Virtual Output Queueing
 - Each input port has n FIFO queues, one for each output
 - Switch using matching in a bipartite graph
 - Shown to achieve 100% throughput*



Current Developments

- Switches are becoming programmable
 - Match-action paradigm
 - Custom protocols, encapsulation, metering, monitoring



- Current speeds reach 12.8Tbps (32x400Gbps or 256x50Gbps) on a single programmable switching chip