

Lecture 27: Final Review

CSE 123: Computer Networks
Alex C. Snoeren

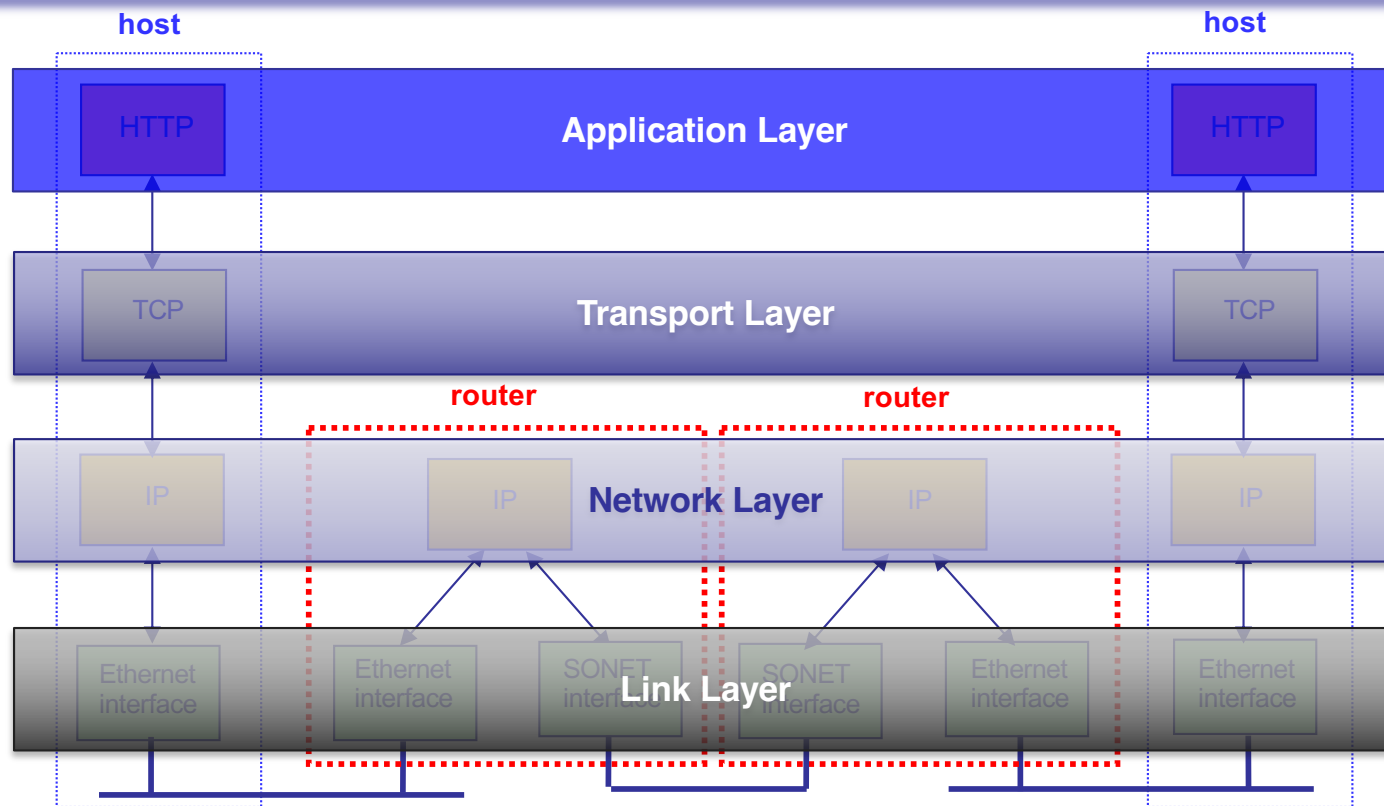


Exam Overview



- Focus on topics since the midterm
 - ◆ But fundamentals remain...
- Roughly the same style & length as midterm
 - ◆ It won't take the whole time
- Exam will be in-person in Center 115
 - ◆ Just like the midterm, you can have a 8.5x11" crib sheet
 - ◆ You will have three hours to complete it once you start
 - ◆ Please bring your ID. NO CALCULATORS

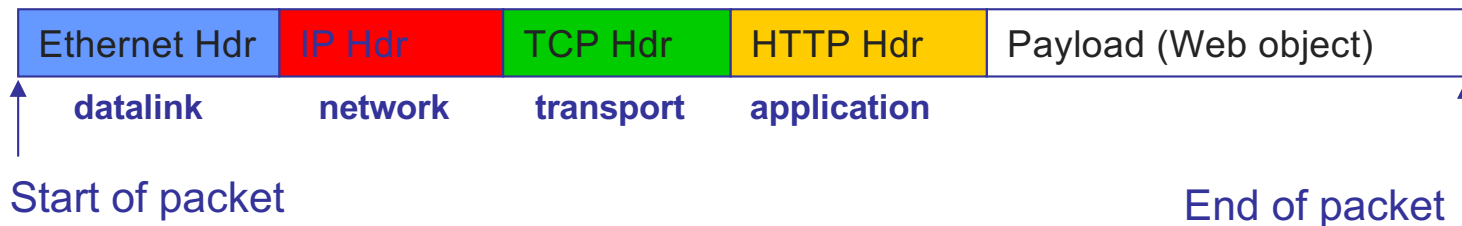
TCP/IP Protocol Stack



Encapsulation via Headers



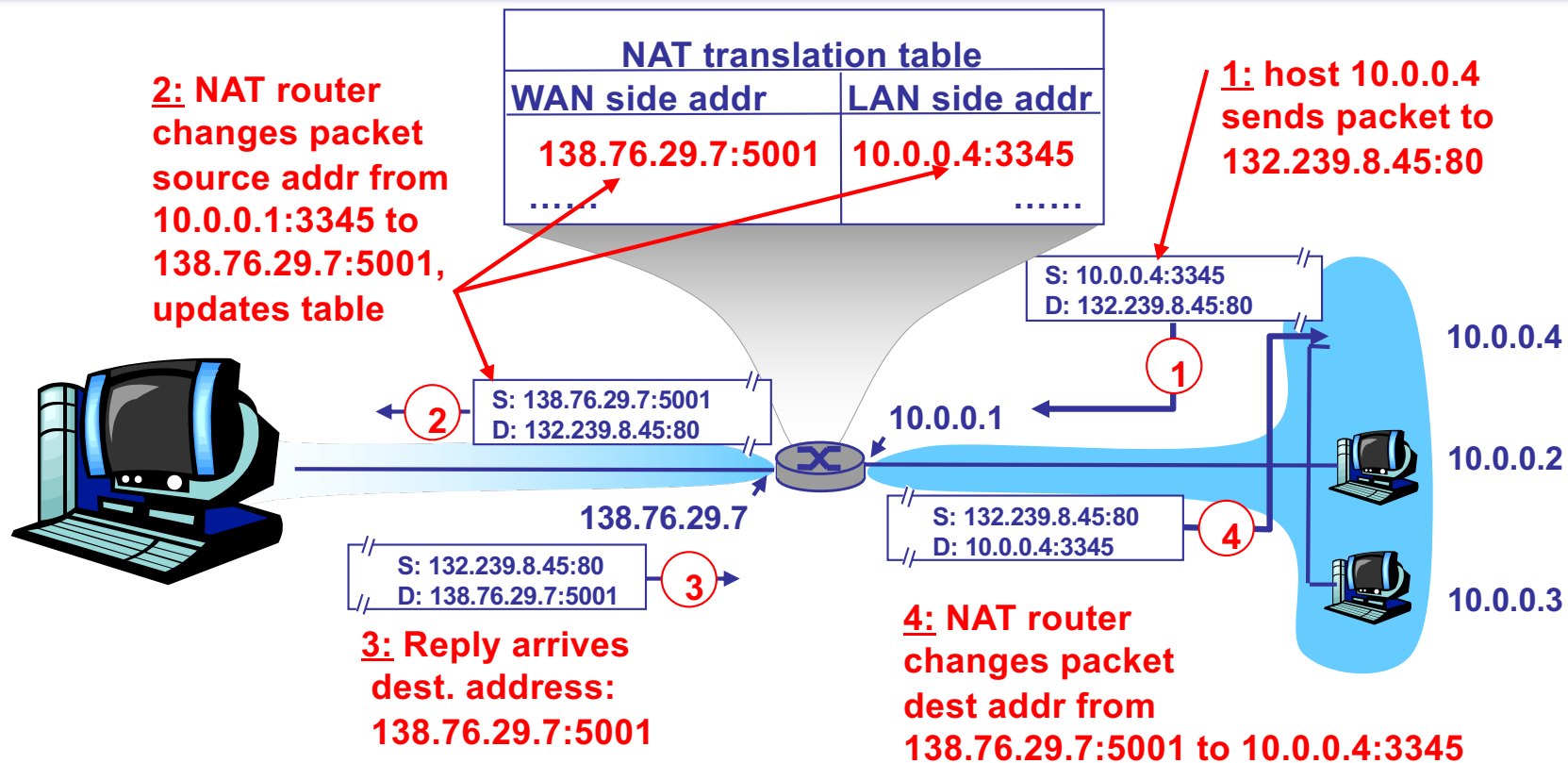
- Typical Web packet



- Notice that layers add overhead
 - ◆ Space (headers), effective bandwidth
 - ◆ Time (processing headers, “peeling the onion”), latency



NA(p)T Example



Routing Challenges



- How to choose best path?
 - ♦ Defining “best” can be slippery
- How to scale to millions of users?
 - ♦ Minimize control messages and routing table size
- How to adapt to failures or changes?
 - ♦ Node and link failures, plus message loss



Forwarding Options

- Source routing
 - ◆ Complete path listed in packet
- Virtual circuits
 - ◆ Set up path out-of-band and store path identifier in routers
 - ◆ Local path identifier in packet
- Destination-based forwarding
 - ◆ Router looks up address in forwarding table
 - ◆ Forwarding table contains (address, next-hop) tuples

Link-state Routing



- Two phases
 - ♦ Reliable flooding
 - » Tell all routers what you know about your local topology
 - ♦ Path calculation (Dijkstra's algorithm)
 - » Each router computes best path over complete network
- Motivation
 - ♦ Global information allows optimal route computation
 - ♦ Straightforward to implement and verify



Dijkstra's Shortest Path Tree

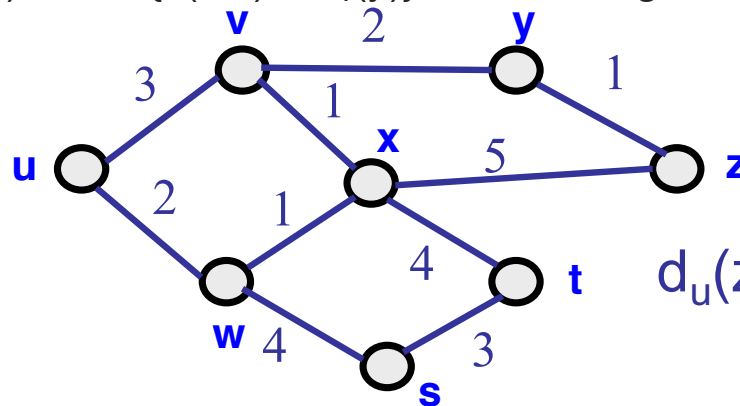
- So you have all of these LSPs. Now what?
- Graph algorithm for single-source shortest path tree (find best route to all nodes)

```
S ← {}  
Q ← <remaining nodes keyed by distance>  
While Q != {}  
    u ← extract-min(Q)  u = node with lowest cost  
    S ← S plus {u}  ← u is done  
    Within Q:  
        for each node v adjacent to u  
            “relax” the cost of v  is it cheaper to go through u?
```



Bellman-Ford Algorithm

- Define distances at each node X
 - ♦ $c(x, v)$ = cost for direct link from X to V
 - ♦ $d_x(y)$ = cost of least-cost *path* from X to Y
- Update distances based on neighbors
 - ♦ $d_x(y) = \min \{c(x, v) + d_v(y)\}$ over all neighbors V



$$d_u(z) = \min\{c(u, v) + d_v(z), c(u, w) + d_w(z)\}$$

Distance Vector Algorithm



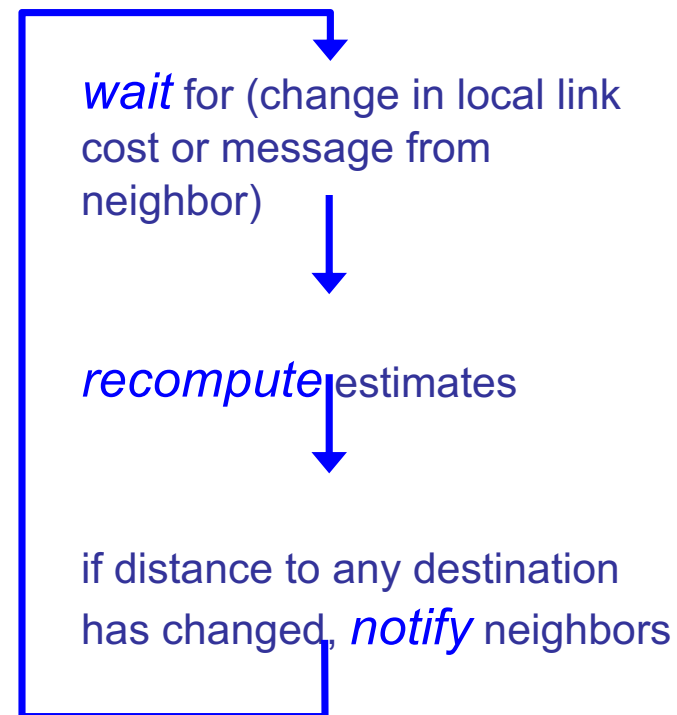
Iterative, asynchronous: each local iteration caused by:

- Local link cost change
- Distance vector update message from neighbor

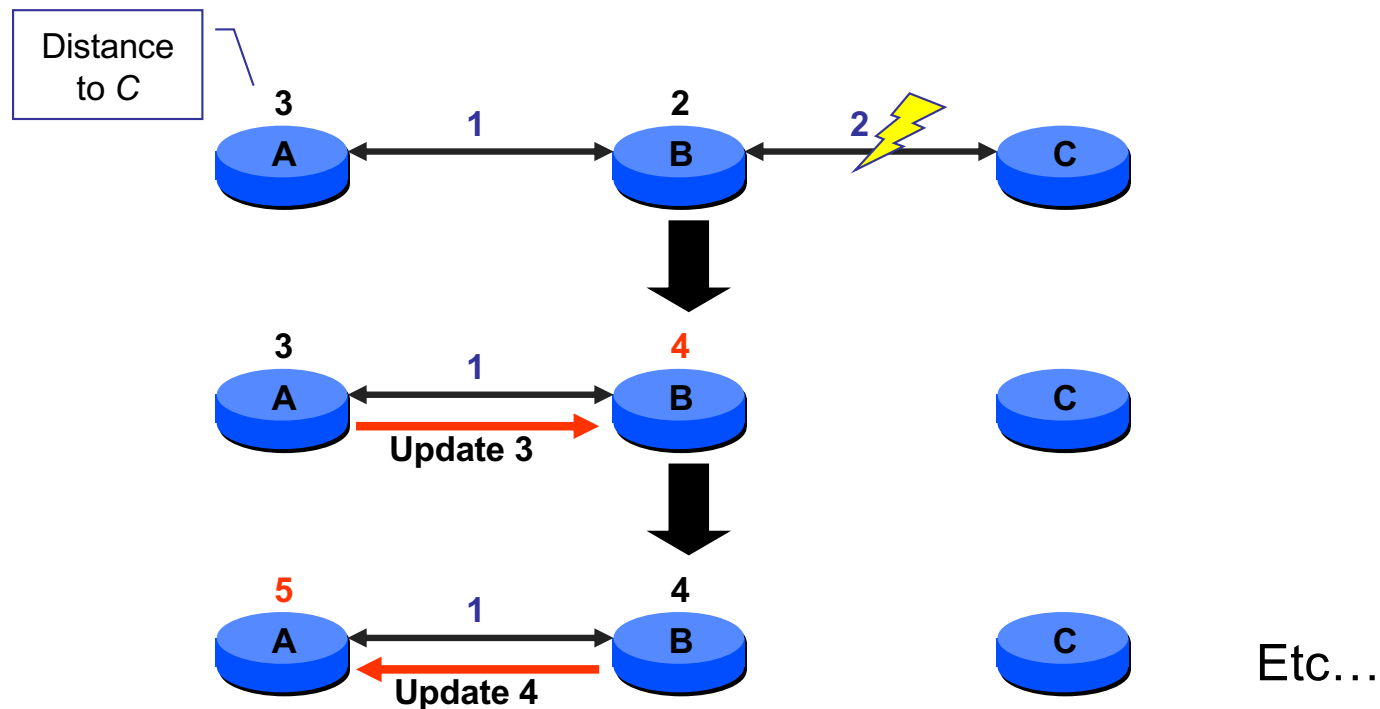
Distributed:

- Each node notifies neighbors when its DV changes
- Neighbors then notify *their* neighbors if necessary

Each node:



Problem: Counting to Infinity





Mitigation Strategies

- Hold downs
 - ♦ As metric increases, delay propagating information
 - ♦ Limitation: Delays convergence
- Loop avoidance
 - ♦ Full path information in route advertisement
 - ♦ Explicit queries for loops
- Split horizon
 - ♦ Never advertise a destination **through its next hop**
 - » *A* doesn't advertise *C* to *B*
 - ♦ **Poison reverse**: Send **negative information** when advertising a destination through its next hop
 - » *A* advertises *C* to *B* with a metric of ∞
 - » Limitation: Only works for “loop”s of size 2

Autonomous Systems

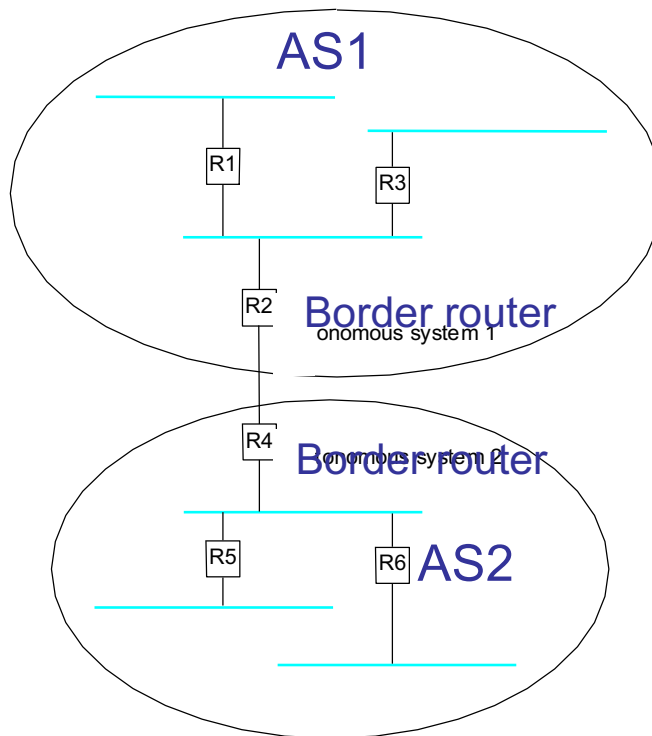


- Internet is divided into **Autonomous Systems**
 - ◆ Distinct regions of administrative control
 - ◆ Routers/links managed by a single “institution”
 - ◆ Service provider, company, university, ...
- Hierarchy of Autonomous Systems
 - ◆ Large, “tier-1” provider with a nationwide backbone
 - ◆ Medium-sized regional provider with smaller backbone
 - ◆ Small network run by a single company or university
- Interaction between Autonomous Systems
 - ◆ Internal topology is not shared between ASes
 - ◆ ... but, neighboring ASes interact to coordinate routing

Inter-domain Routing



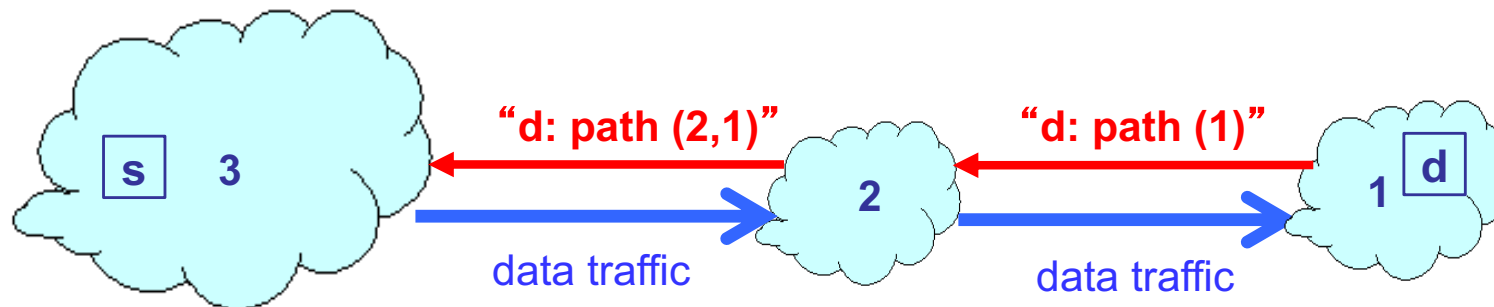
- **Border routers** summarize and advertise their routes to external neighbors and vice-versa
 - ♦ Border routers apply **policy**
- Internal routers can use notion of **default routes**
- Core is **default-free**; routers must have a route to all networks in the world
- But what routing protocol?





Path-vector Routing

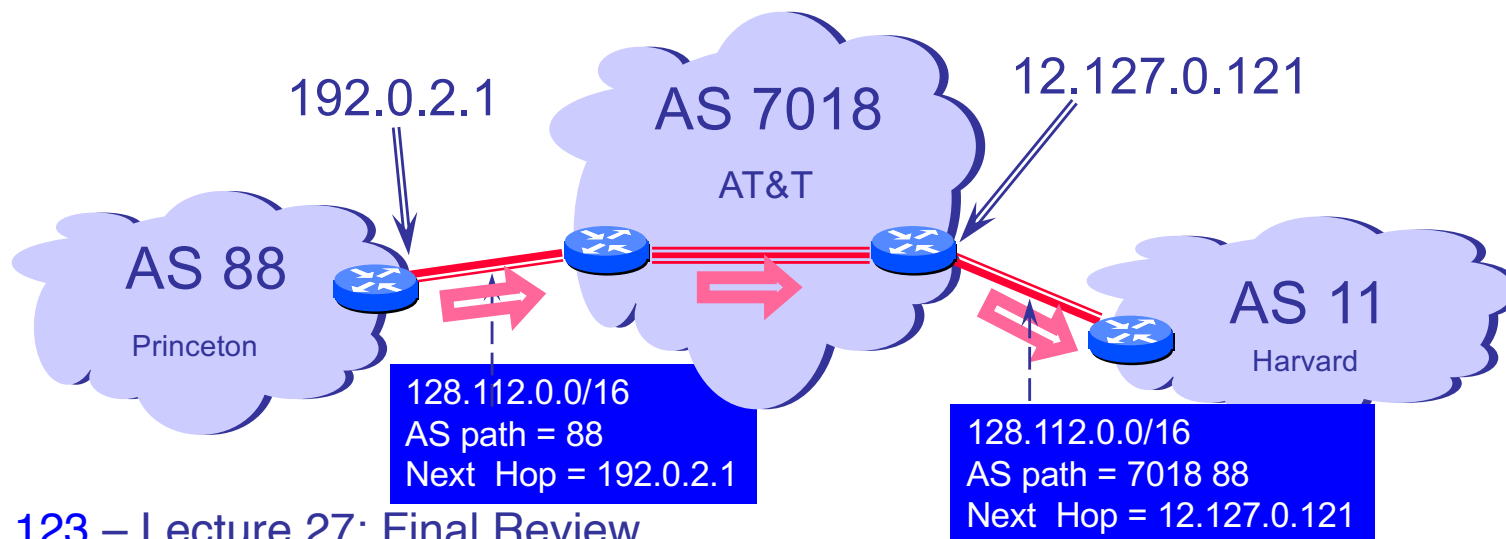
- Extension of distance-vector routing
 - ◆ Support flexible routing policies
 - ◆ Avoid count-to-infinity problem
- Key idea: advertise the entire path
 - ◆ Distance vector: send *distance metric* per destination
 - ◆ Path vector: send the *entire path* for each destination





A Simple BGP Route

- Destination prefix (e.g., 128.112.0.0/16)
- Route attributes, including
 - ♦ AS path (e.g., “7018 88”)
 - ♦ Next-hop IP address (e.g., 12.127.0.121)



Business Relationships



- Neighboring ASes have business contracts
 - ◆ How much traffic to carry
 - ◆ Which destinations to reach
 - ◆ How much money to pay
- Common business relationships
 - ◆ Customer-provider
 - » E.g., UCSD is a customer of Sprint
 - » E.g., MIT is a customer of Level3
 - ◆ Peer-peer
 - » E.g., UUNET is a peer of Sprint
 - » E.g., Harvard is a peer of Harvard Business School

Functional architecture

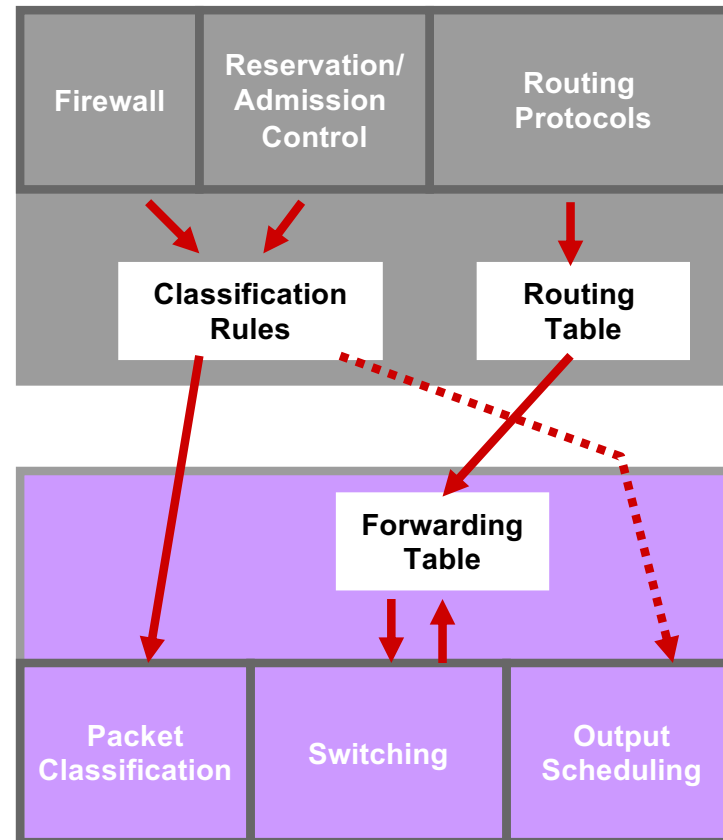


Control Plane

- Complex
- Per-control action
- May be slow

Data plane

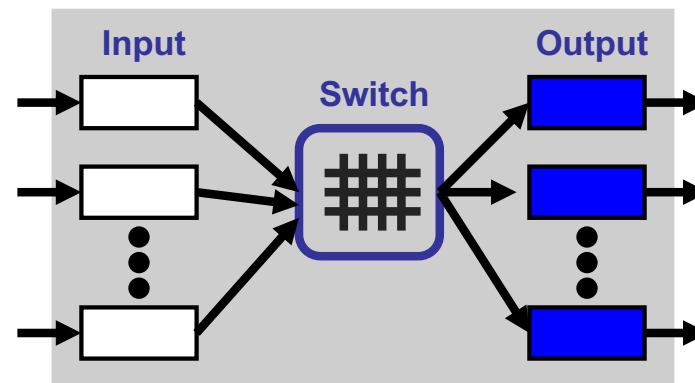
- Simple
- Per-packet
- Must be fast



Interconnect architecture



- Input & output connected via switch fabric
- Kinds of switch fabric
 - ◆ Shared Memory
 - ◆ Bus
 - ◆ Crossbar
- How to deal with transient contention?
 - ◆ Buffering



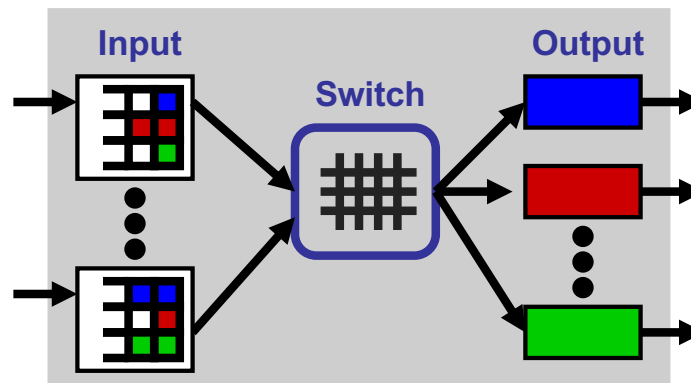


IQ + Virtual Output Queuing

- Input interfaces buffer packets in per-output virtual queues

- Pro
 - ◆ Solves blocking problem

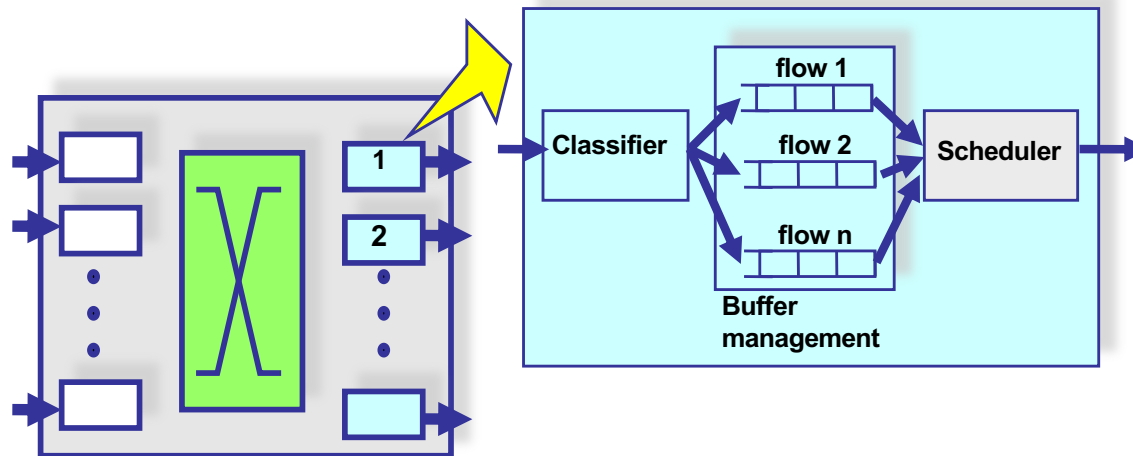
- Con
 - ◆ More resources per port
 - ◆ Complex arbiter at switch
 - ◆ Still limited by input/output contention (scheduler)



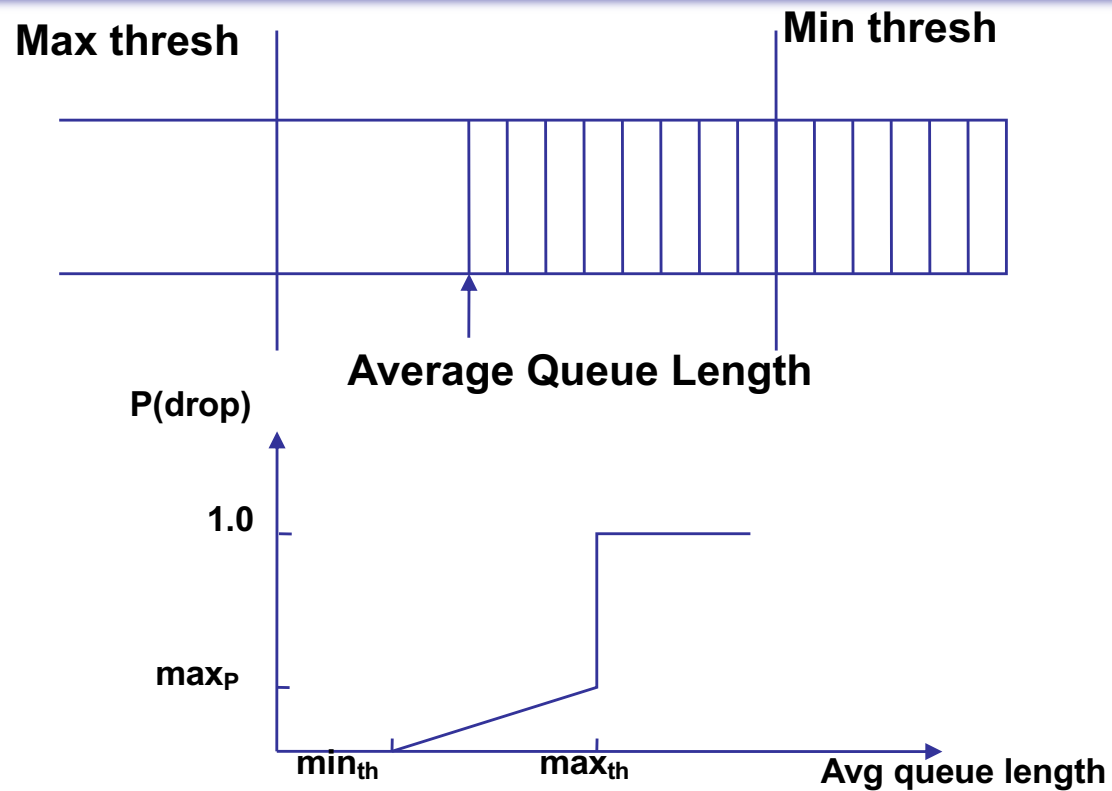


Key Router Challenges

- **Buffer management**: which packet to drop when?
 - ◆ We only have finite-length queues
- **Scheduling**: which packet to transmit next?



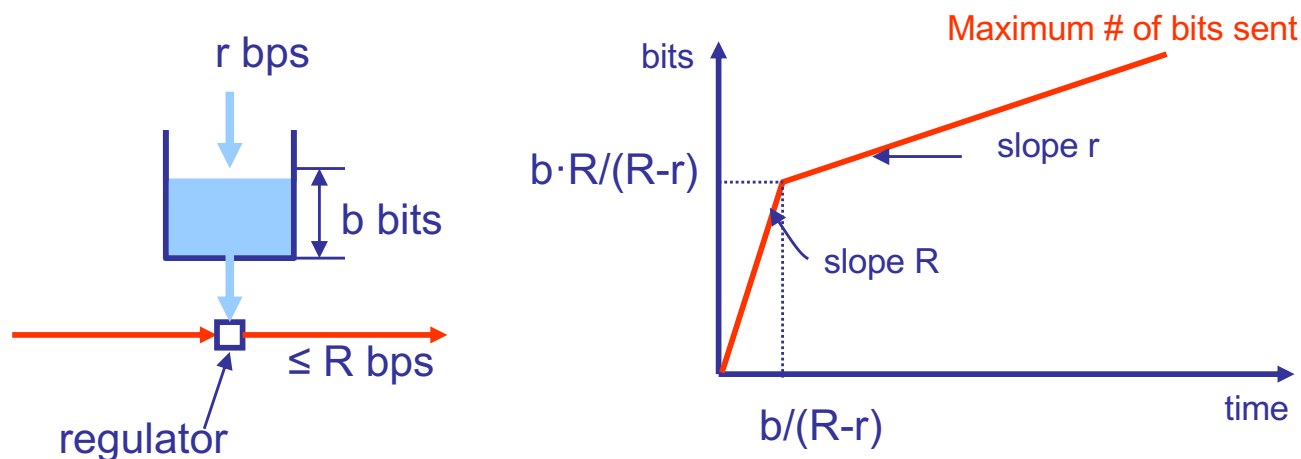
RED Operation





Token Bucket Basics

- Parameters
 - ♦ r – average rate, i.e., rate at which tokens fill the bucket
 - ♦ b – bucket depth (limits size of burst)
 - ♦ R – maximum link capacity or peak rate (optional parameter)
- A bit can be transmitted only when a token is available



Congestion Collapse

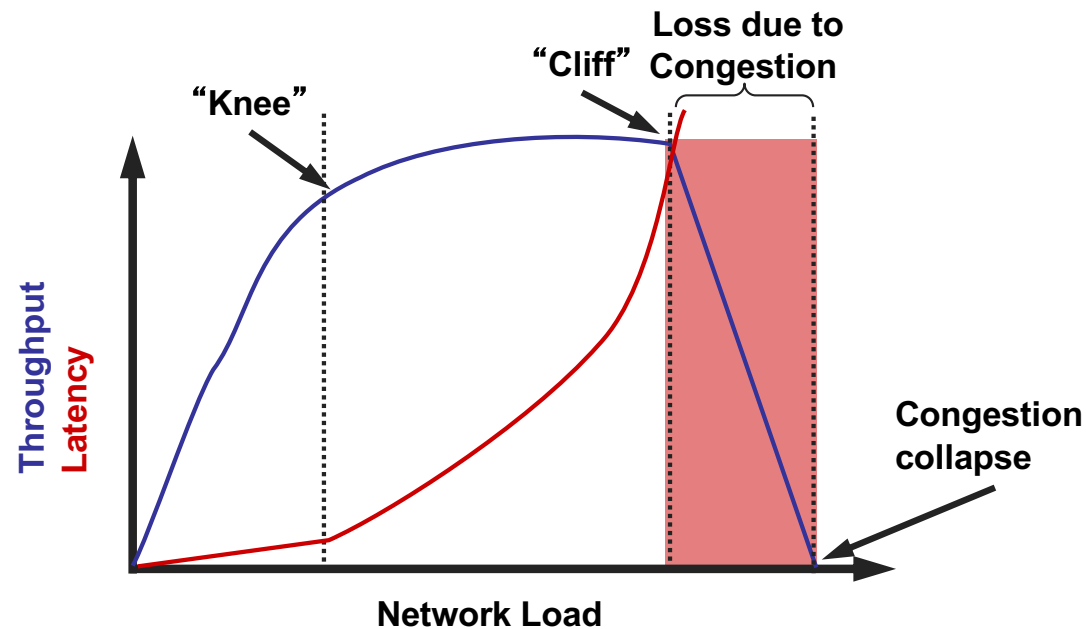


- Rough definition: “When an increase in network load produces a decrease in useful work”
- Why does it happen?
 - ◆ Senders send faster than **bottleneck link** speed
 - ◆ Packets queue until dropped
 - ◆ In response to packets being dropped, senders retransmit
 - ◆ All hosts repeat in steady state...

Proactive vs. Reactive



- Congestion avoidance: try to stay to the left of the knee
- Congestion control: try to stay to the left of the cliff

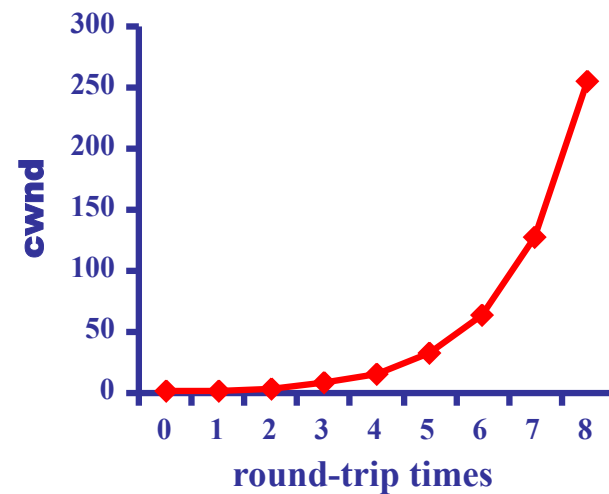
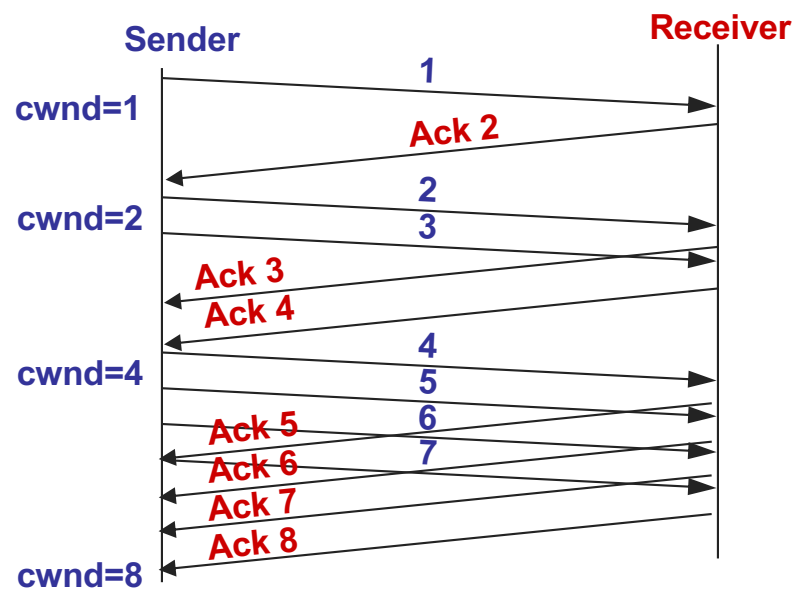


Basic TCP Algorithm

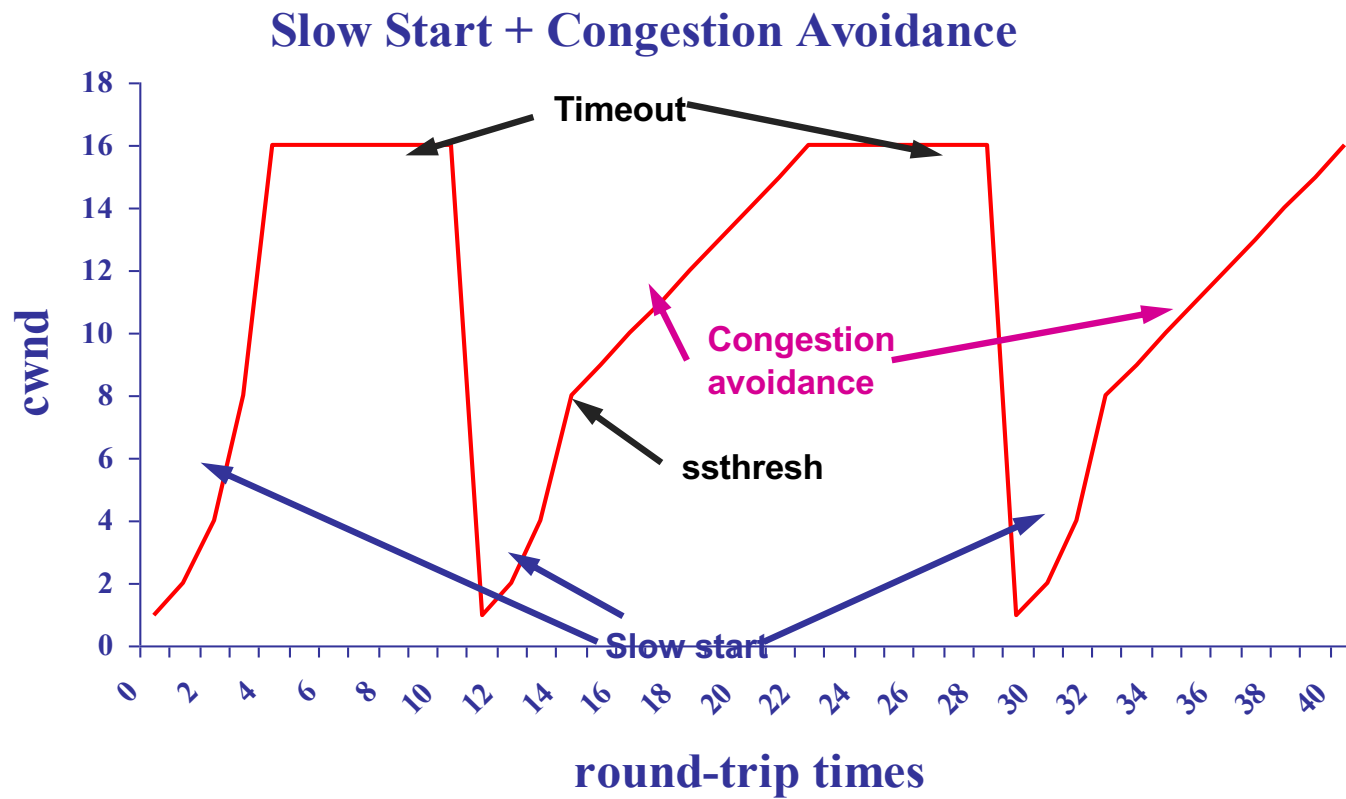


- Window-based congestion control
 - ◆ Unified congestion control and flow control mechanism
 - ◆ *rwin*: advertised flow control window from receiver
 - ◆ *cwnd*: congestion control window
 - » Estimate of how much outstanding data network can deliver in a round-trip time
 - ◆ Sender can only send $\text{MIN}(rwin, cwnd)$ at any time
- Idea: decrease *cwnd* when congestion is encountered; increase *cwnd* otherwise
 - ◆ Question: how much to adjust?

Slow Start Example



Basic Mechanisms



Fair Queuing



- Maintain a queue for each flow
 - ♦ What is a flow?
- Implements **max-min fairness**: each flow receives $\min(r_i, f)$, where
 - ♦ r_i – flow arrival rate
 - ♦ f – link fair rate (see next slide)
- **Weighted Fair Queuing** (WFQ) – associate a weight with each flow to divvy bandwidth up non-equally



Phy/(MAC)Link layer

- Signal encoding
 - ◆ Encode binary data from source node into signals that physical links carry
 - ◆ Signal is decoded back into binary data at receiving node
 - ◆ Work performed by network adapter at sender and receiver
- Media access
 - ◆ Arbitrate which nodes can send frames at any point in time
 - ◆ Not always necessary; e.g. point-to-point duplex links

Signals and Channels

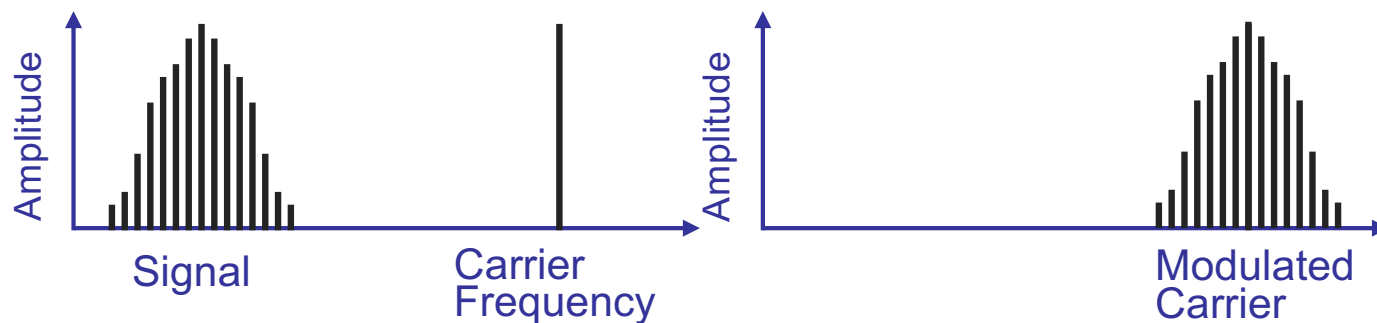


- A **signal** is some form of energy (light, voltage, etc)
 - ◆ Varies with time (on/off, high/low, etc.)
 - ◆ Can be continuous or discrete
- A **channel** is a physical medium that conveys energy
 - ◆ Any real channel will distort the input signal as it does so
 - ◆ How it distorts the signal depends on the signal

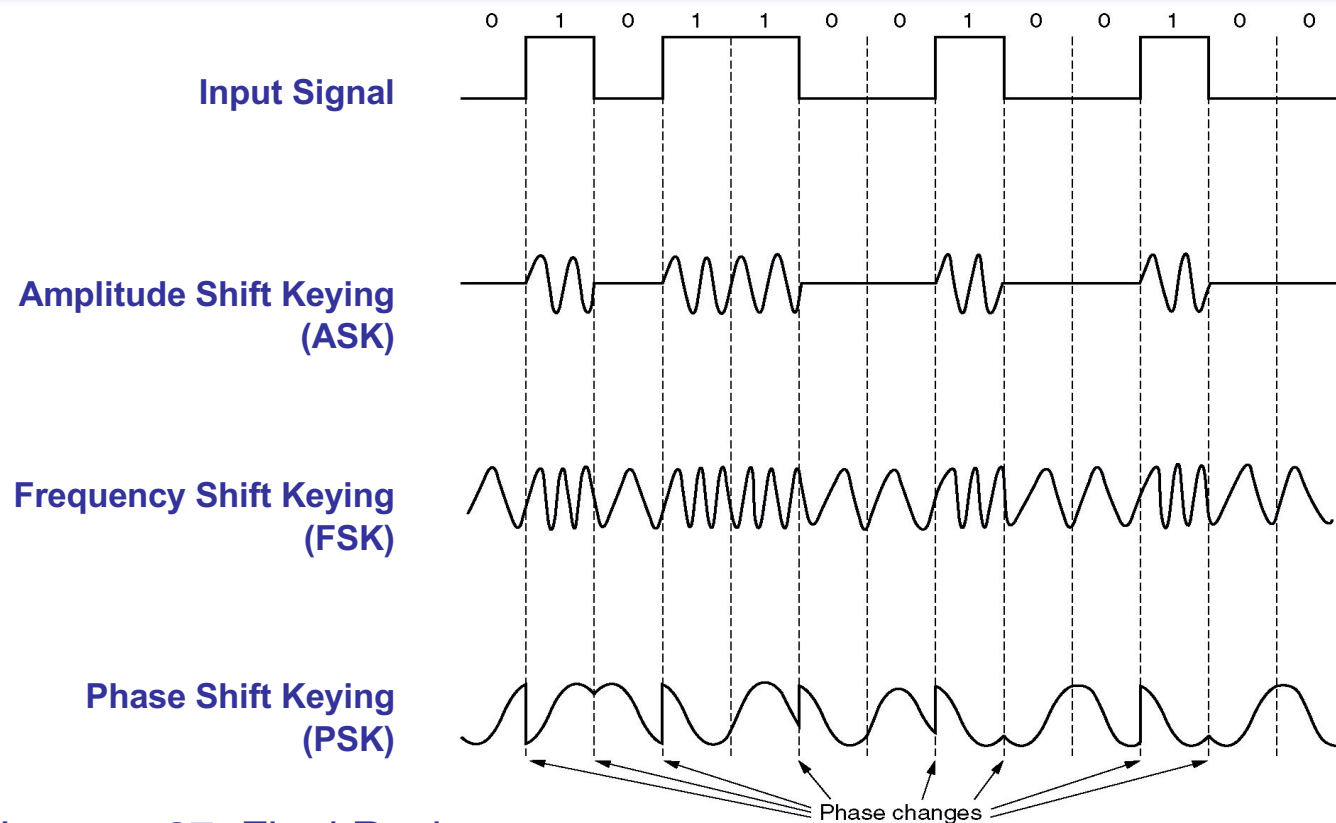
Carrier Signals



- **Baseband** modulation: send the “bare” signal
 - ♦ E.g. +5 Volts for 1, -5 Volts for 0
 - ♦ All signals fall in the same frequency range
- **Broadband** modulation
 - ♦ Use the signal to modulate a high frequency signal (**carrier**).
 - ♦ Can be viewed as the product of the two signals



Forms of Digital Modulation



Shannon's Law



- Shannon considered noisy channels and derived

$$C = B \log_2 (1 + S/N)$$

- Gives us an upper bound on any channel's performance regardless of signaling scheme
- Old school modems approached this limit
 - ◆ $B = 3000\text{Hz}$, $S/N = 30\text{dB} = 1000$
 - ◆ $C = 3000 \times \log(1001) \approx 30\text{kbps}$
 - ◆ 28.8Kbps – anyone remember dialup?

Clock Recovery



- Using a training sequence to get receiver lined up
 - ◆ Send a few, known **initial training bits**
 - ◆ Adds inefficiency: only m data bits out of n transmitted
- Need to combat clock drift as signal proceeds
 - ◆ Use **transitions** to keep clocks synched up
- Question is, how often do we do this?
 - ◆ Quick and dirty every time: **asynchronous** coding
 - ◆ Spend a lot of effort to get it right, but amortize over lots of data: **synchronous** coding

Manchester Encoding (10Mbps Ethernet)

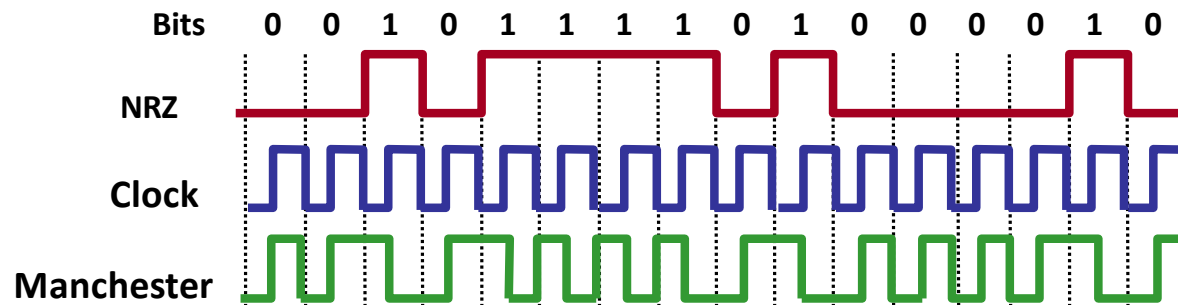


- Signal to Data

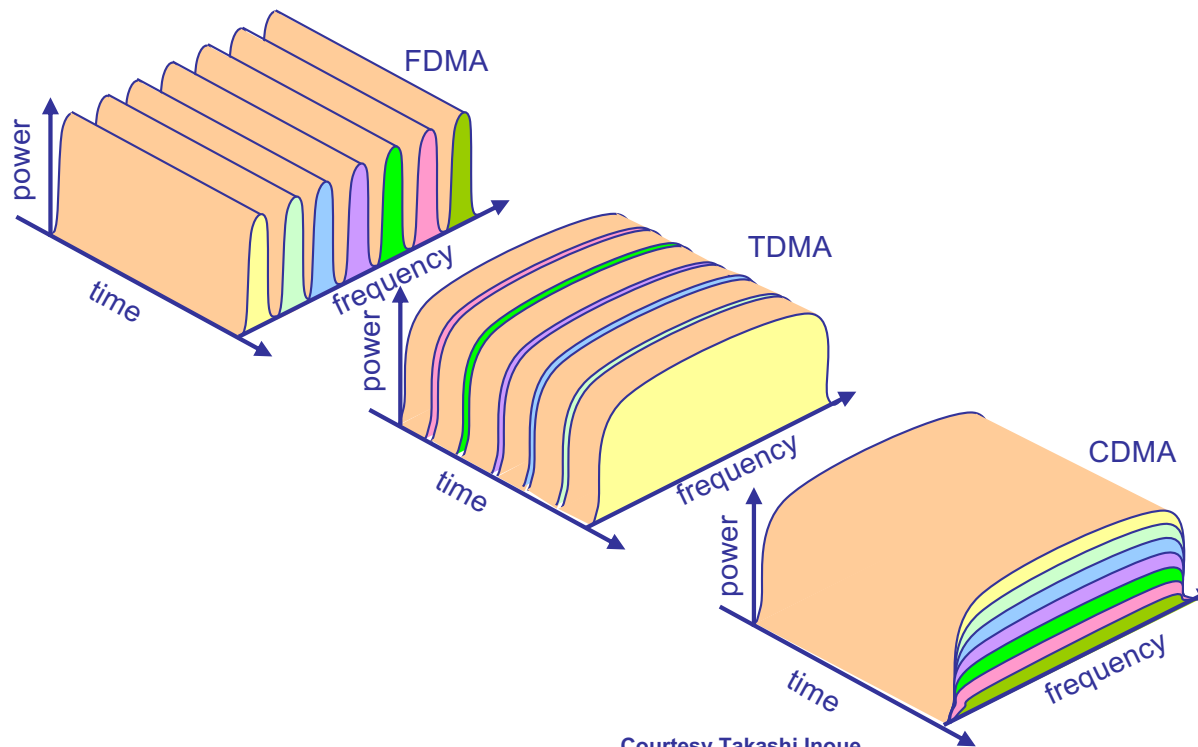
- ♦ XOR NRZ data with senders clock signal
- ♦ High to low transition $\Rightarrow$ 1
- ♦ Low to high transition $\Rightarrow$ 0

- Comments

- ♦ Solves clock recovery problem
- ♦ Only 50% efficient ($\frac{1}{2}$ bit per transition)
- ♦ Still need preamble (typically 0101010101... trailing 11 in Ethernet)



Partitioning Visualization



Courtesy Takashi Inoue

Aloha



- Designed in 1970 to support wireless data connectivity
 - ♦ Between Hawaiian Islands—rough!
- Goal: distributed access control (no central arbitrator)
 - ♦ Over a shared *broadcast* channel
- Aloha protocol in a nutshell:
 - ♦ When you have data **send it**
 - ♦ If data doesn't get through (receiver sends acknowledgement) then **retransmit after a random delay**
 - ♦ Why not a fixed delay?

Channel Efficiency



Q: What is max fraction slots successful?

A: Suppose n stations have packets to send

- ◆ Each transmits in slot with probability p
- ◆ Prob[successful transmission], S , is:

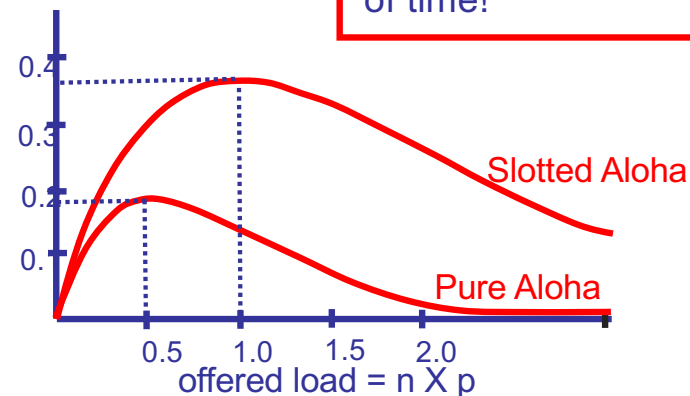
$$S = p (1-p)^{(n-1)}$$

- ◆ any of n nodes:

$$S = \text{Prob}[\text{one transmits}] = np(1-p)^{(n-1)}$$

(optimal p as $n \rightarrow \text{infinity} = 1/n$)

$$= 1/e = .37$$



CSMA/CD



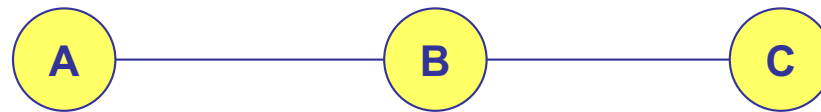
- How can A know that a collision has taken place?
 - ♦ Worst case:
 - » Latency between nodes A & B is d
 - » A sends a message at time t and B sends a message at $t + d - \epsilon$ (just before receiving A's message)
 - ♦ B knows there is a collision, but not A... A must keep transmitting until it can tell if a collision occurred
 - ♦ How long? $2 * d$
- IEEE 802.3 Ethernet specifies max value of $2d$ to be 51.2us
 - ♦ This relates to maximum distance of 2500m between hosts
 - ♦ At 10Mbps it takes 0.1us to transmit one bit so 512 bits take 51.2us to send
 - ♦ So, Ethernet frames must be at least 64B (512 bits) long
 - » Padding is used if data is too small
- Send jamming signal to insure all hosts see collision
 - ♦ 48 bit signal

CSMA/CA



- Cannot detect collision w/half-duplex radios
 - ◆ So collisions are (much) more expensive than in CSMA/CD
- Wireless MAC protocols often use collision avoidance techniques, in conjunction with a (physical or virtual) carrier sense mechanism
- Collision avoidance
 - ◆ Nodes negotiate to reserve the channel before sending data
 - ◆ Still potential for collisions during negotiation, but they are cheaper

RTS/CTS (MACA)



- When A wants to send a packet to B, A first sends a **Request-to-Send (RTS)** to B
- On receiving RTS, B responds by sending **Clear-to-Send (CTS)**, provided that A is able to receive the packet
- When C overhears a CTS, it keeps quiet for the duration of the transfer
 - ♦ Transfer duration is included in both RTS and CTS

Parting thoughts...



- Please complete your CAPE survey online
- Final exam: **MONDAY 8:00-11:00 AM**
- Good luck and have a great (and safe) Spring Break!