Lecture 26: Final Review

CSE 123: Computer Networks
Alex C. Snoeren

Project 2 due TONIGHT

UCSDCSE

Exam Overview

- Focus on topics since the midterm
  - But everything is fair game…
- Roughly the same style and length of the midterm
  - It won't take the whole time
  - We'll spend the first few minutes of the exam period announcing the Espresso Prize winners
- As before, you can bring a crib sheet
  - One double-sided 8.5x11" paper with anything on it

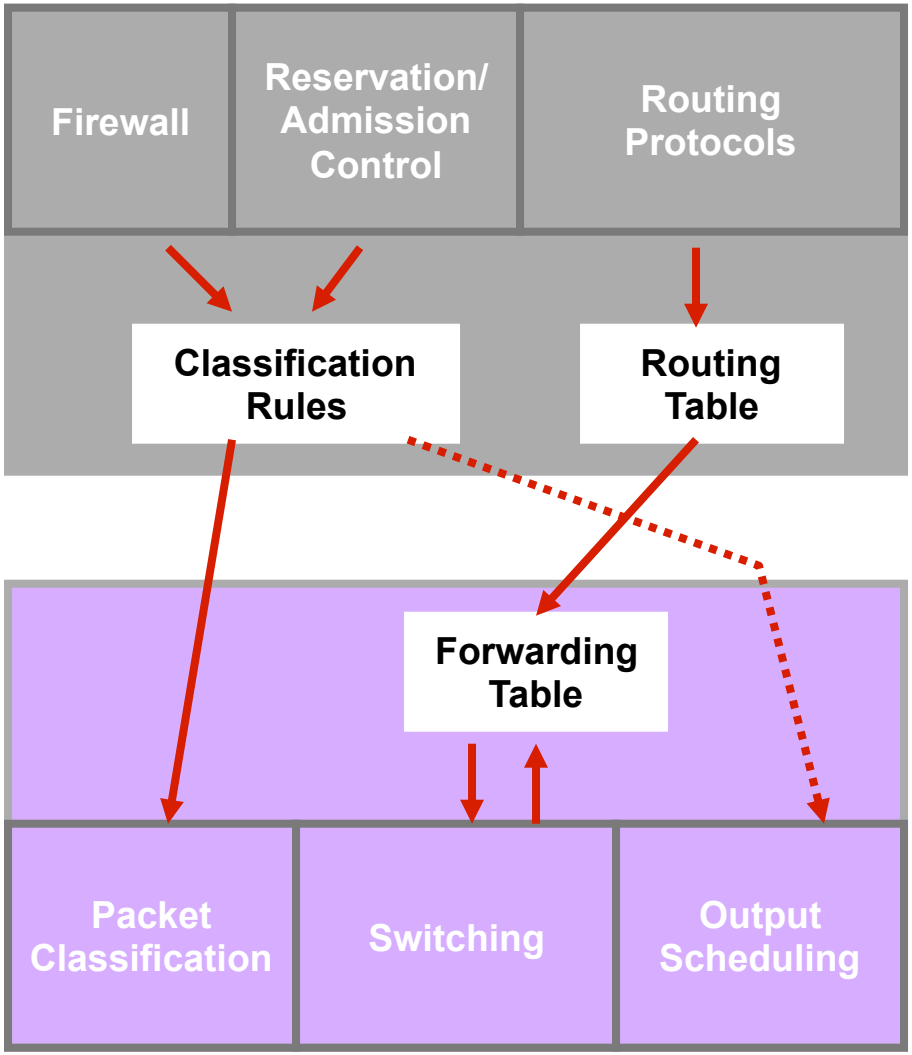
Take Home Portion

- Recall you completed a pre-survey at the beginning
  - Was worth a portion of your HW1 grade
- We are now asking you to complete a post-survey
  - Will be worth a portion of the final exam grade

https://www.surveymonkey.com/s/FRX5LG9

- Please complete anytime before the Final exam
  - And the CAPEs, too!

Router Functional Architecture

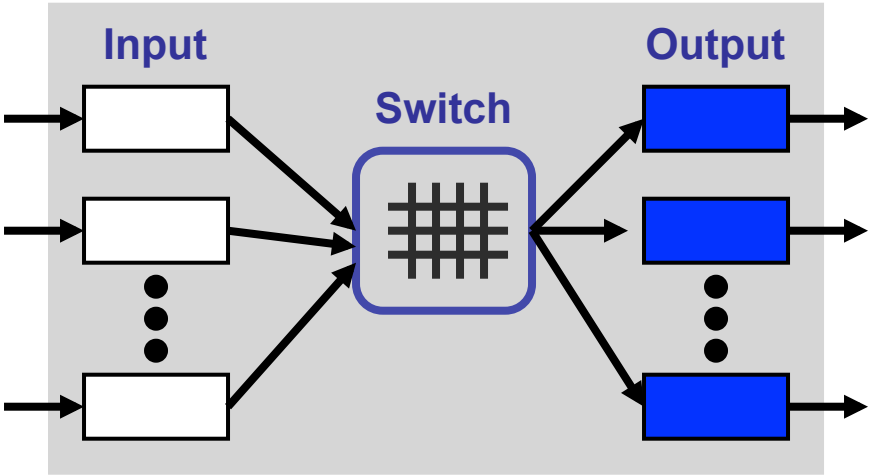


Control Plane
- Complex
- Per-control action
- May be slow

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?
  - Input queuing
  - Output queuing



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
  - Divide task into intra- and inter-AS
- 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

- Routing is a distributed algorithm
  - React to changes in the topology
  - Compute the paths through the network
- Shortest-path link state routing
  - Flood link weights throughout the network
  - Compute shortest paths as a sum of link weights
  - Forward packets on next hop in the shortest path
- Motivation
  - Global information allows optimal route computation
  - Straightforward to implement and verify

Dijkstra's Shortest Path

- Graph algorithm for single-source shortest path tree

```
S ← {}
Q ← <remaining nodes keyed by distance>
While Q != {}
        u ← extract-min(Q)
        S ← S plus {u}
        for each node v adjacent to u
                "relax" the cost of v
```

← u is done

Distance Vector Routing

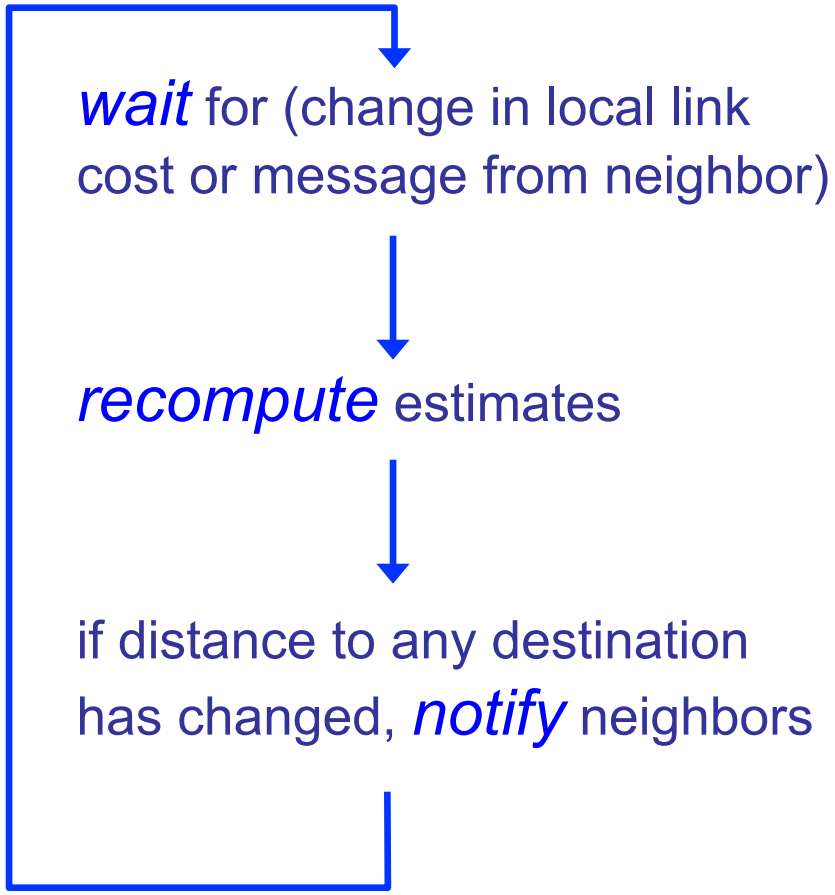
Iterative, asynchronous: each local iteration caused by:

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

Distributed:

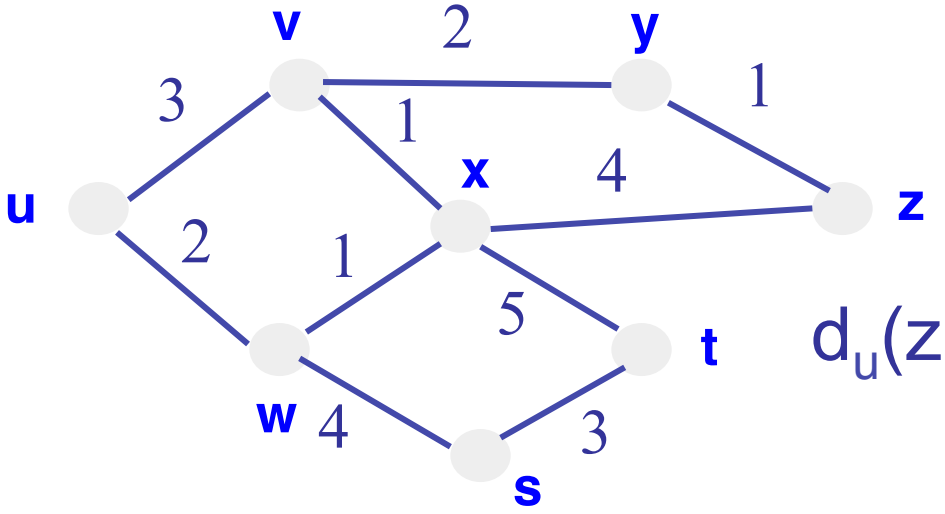
- Each node notifies neighbors *only* when its DV changes
- Neighbors then notify their neighbors if necessary

Each node:



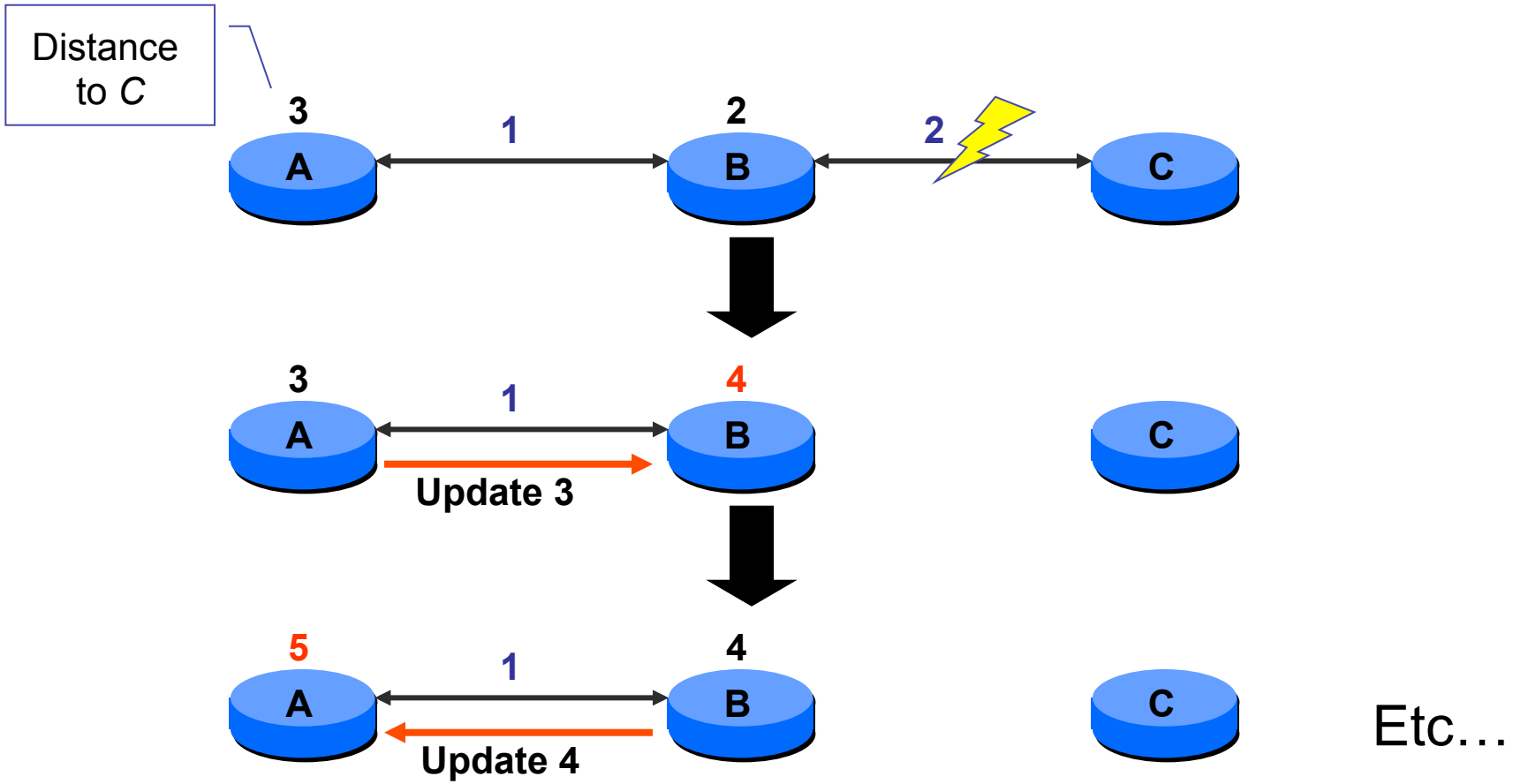
Bellman-Ford Algorithm

- Define distances at each node *X*
  - $d_x(y)$ = 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*



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

Counting to Infinity Problem



Mitigation Strategies

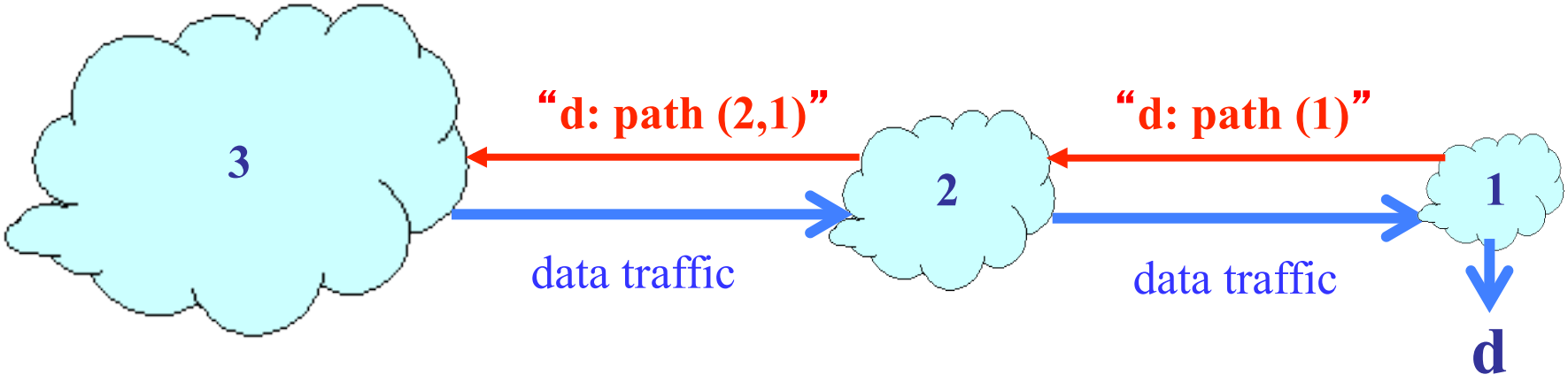
- Hold downs
  - As metric increases, delay propagating information
  - Limitation: Delays convergence
- Loop avoidance
  - Full path information in route advertisement
  - Explicit queries for loops (e.g. DUAL)
- Split horizon
  - Never advertise a destination through its next hop
    - » *A* doesn't 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

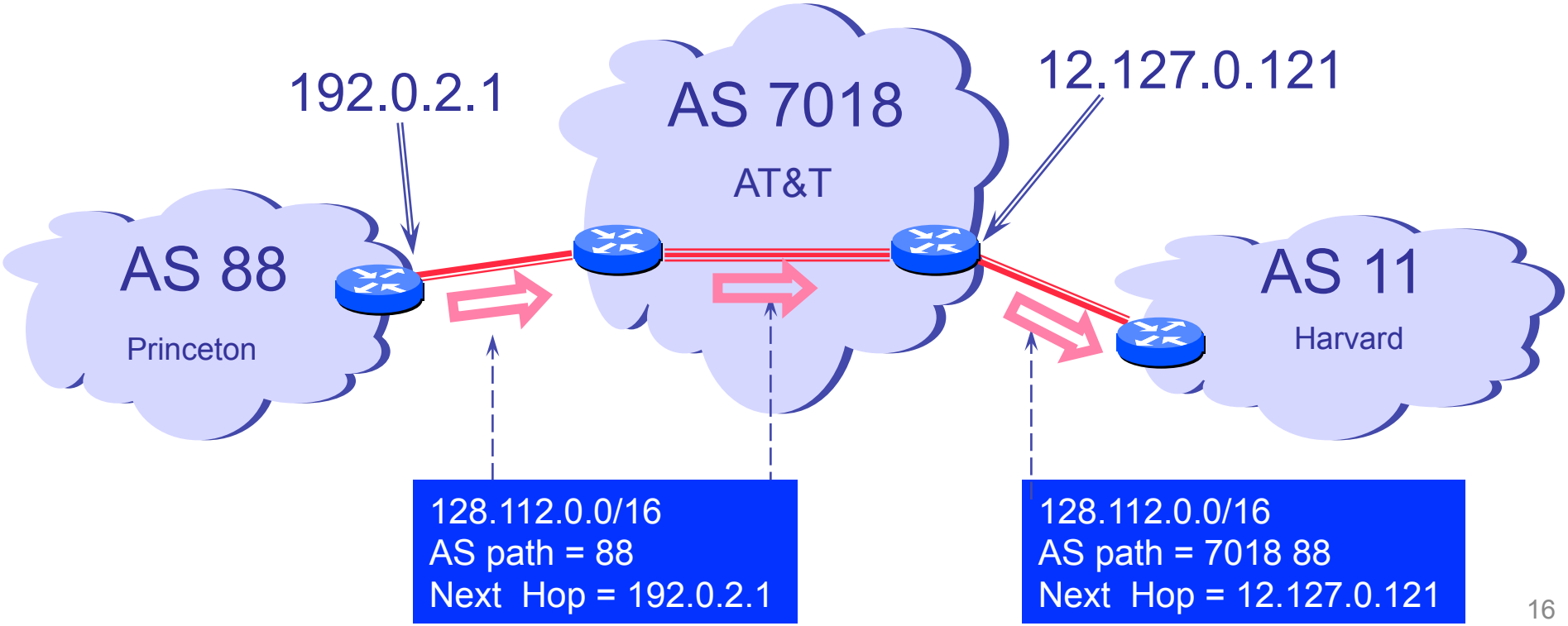
Path-vector Routing

- 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)

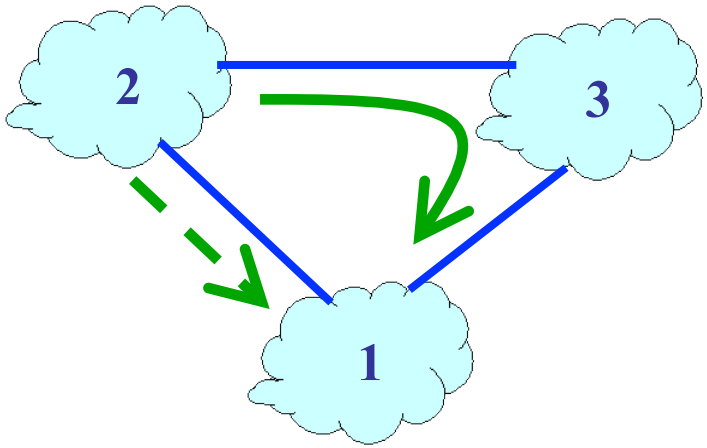


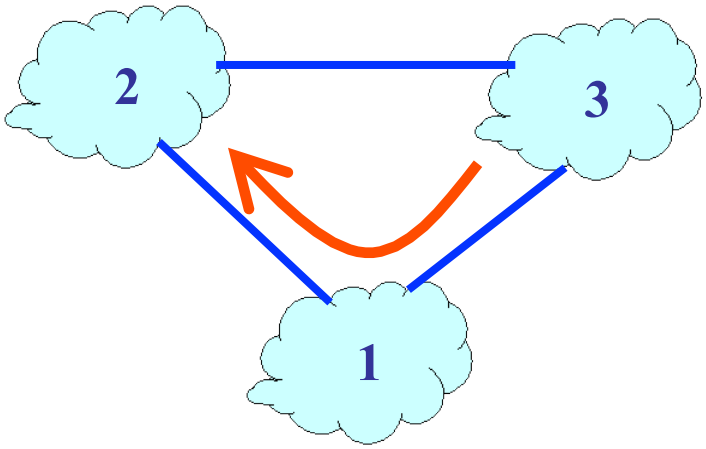
Business Relationships

- 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., Princeton is a customer of USLEC
    - » 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

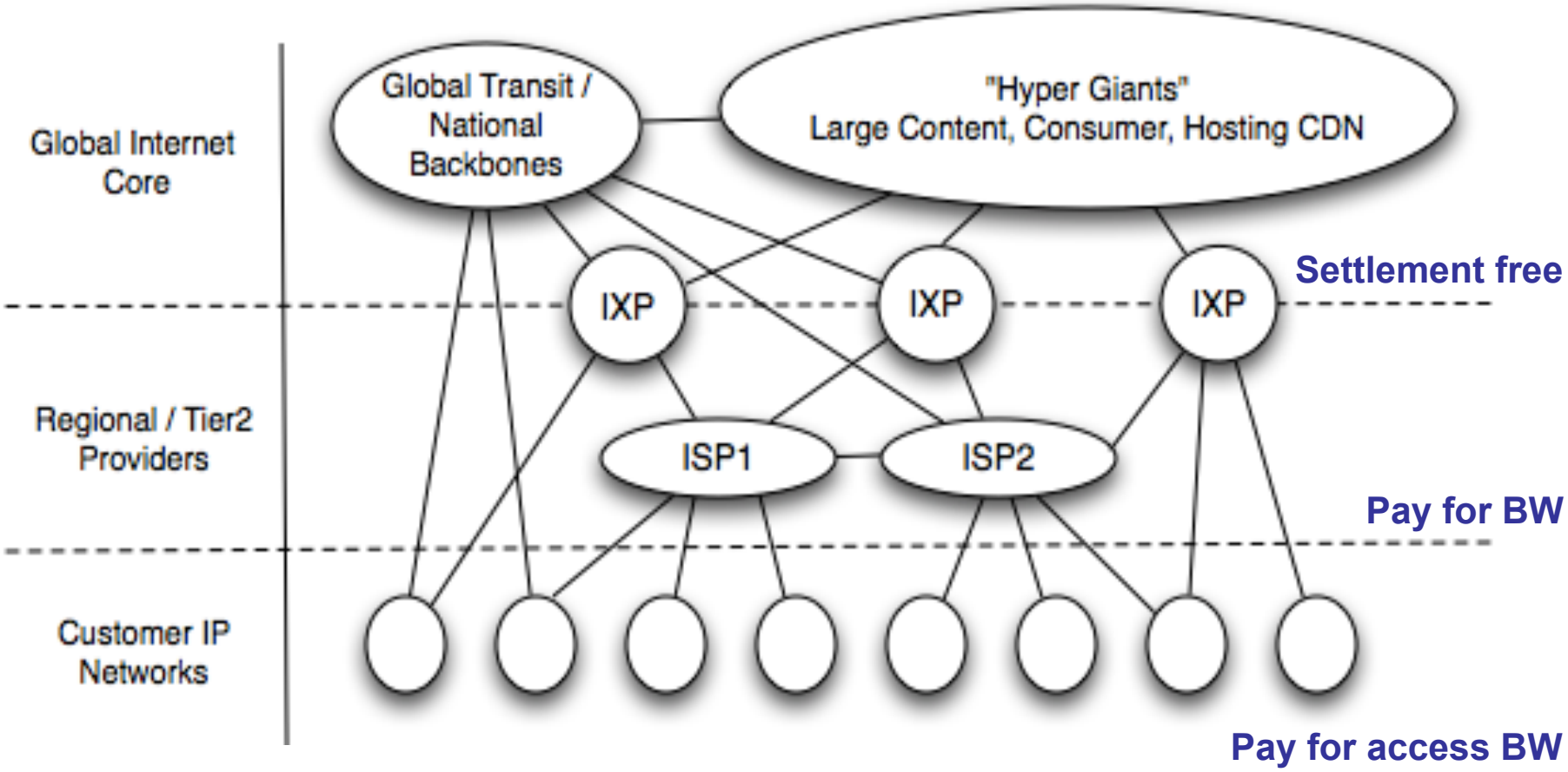
Policy Support

- Each node can apply local policies
  - Path selection: Which path to use?
  - Path export: Which paths to advertise?
- Examples
  - Node 2 may prefer the path “2, 3, 1” over “2, 1”
  - Node 1 may not let node 3 hear the path “1, 2”



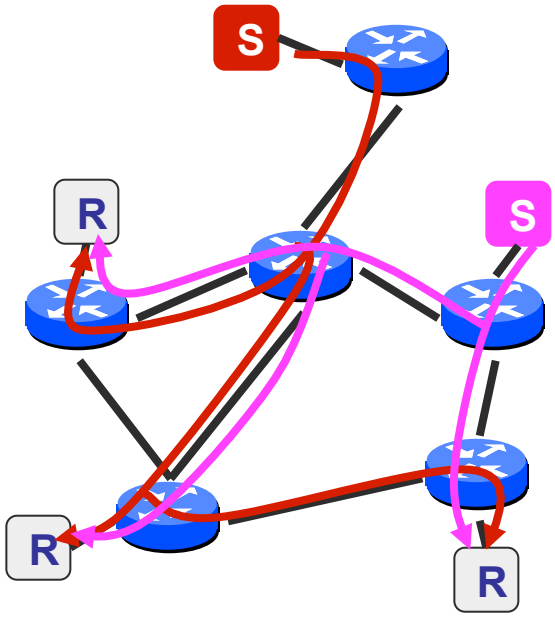


AS Relationships



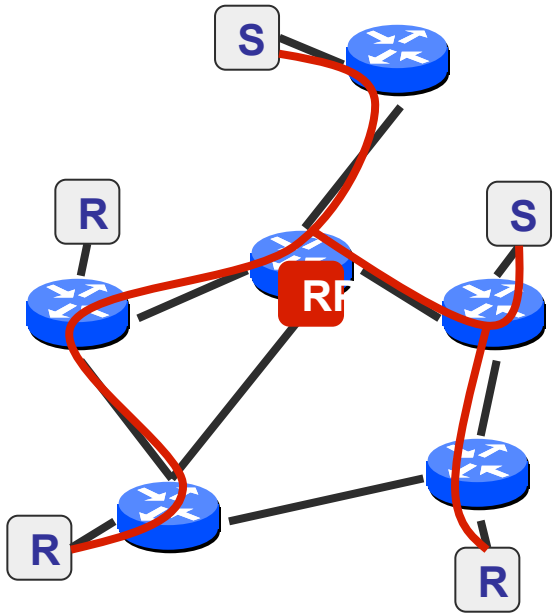
Multicast: Different Options

Source-based tree



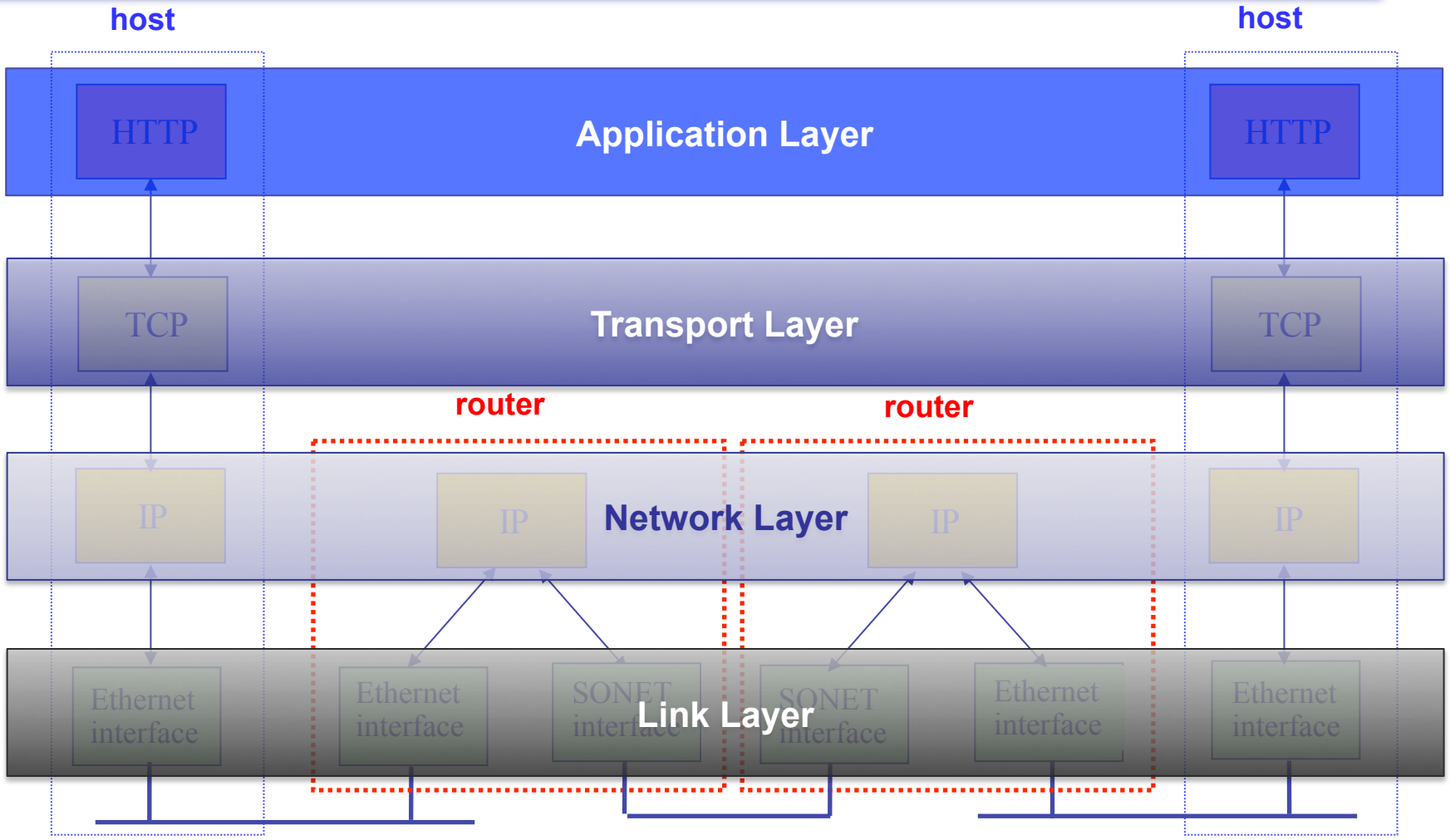
- Efficient trees; low delay, even load
- Per-source state in routers (S,G)

Shared-tree



- Higher delay, skewed load
- Per-group state only (G)

Transport Layer

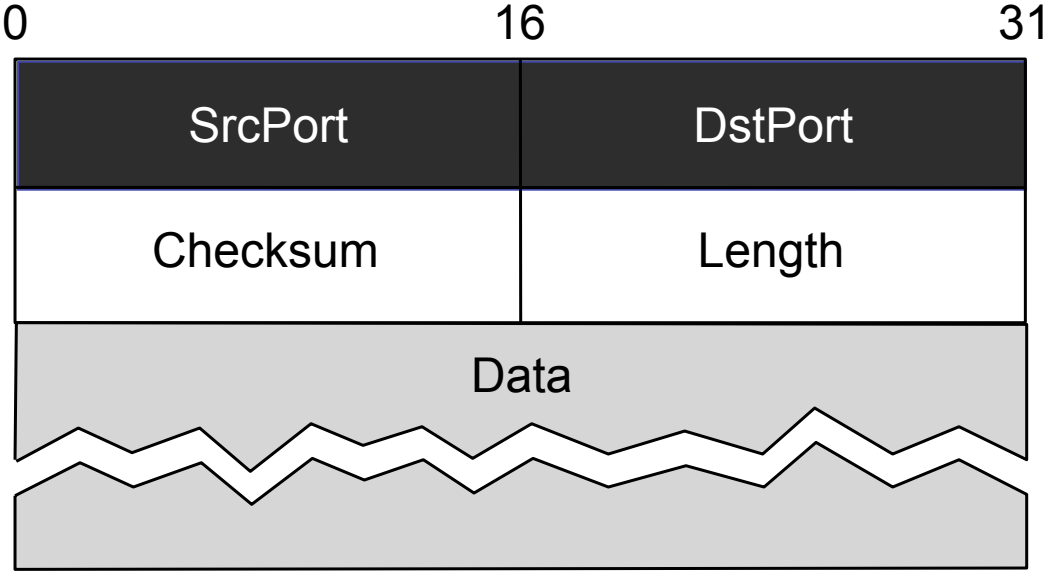


Naming Processes/Services

- Process here is an abstract term for your Web browser (HTTP), Email servers (SMTP), hostname translation (DNS)
- How do we identify for remote communication?
  - Process id or memory address are OS-specific and transient
- So TCP and UDP use Ports
  - 16-bit integers representing mailboxes that processes "rent"
  - Identify process uniquely as (IP address, protocol, port)

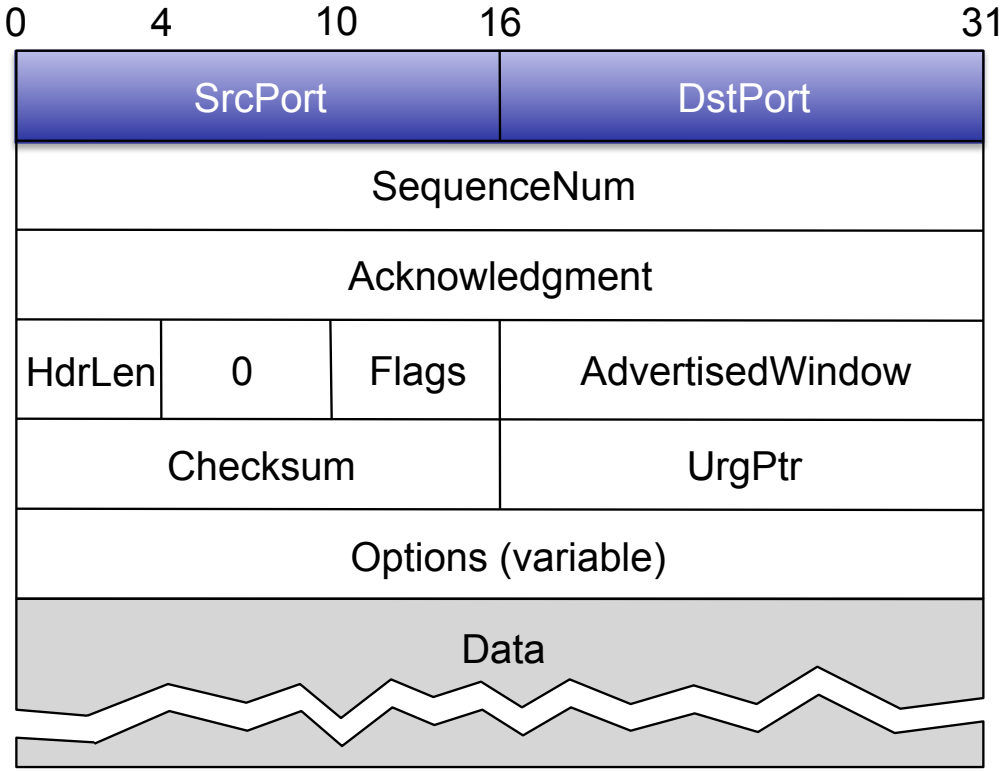
User Datagram Protocol (UDP)

- Provides *unreliable message delivery* between processes
  - Source port filled in by OS as message is sent
  - Destination port identifies UDP delivery queue at endpoint
- Connectionless (no state about who talks to whom)

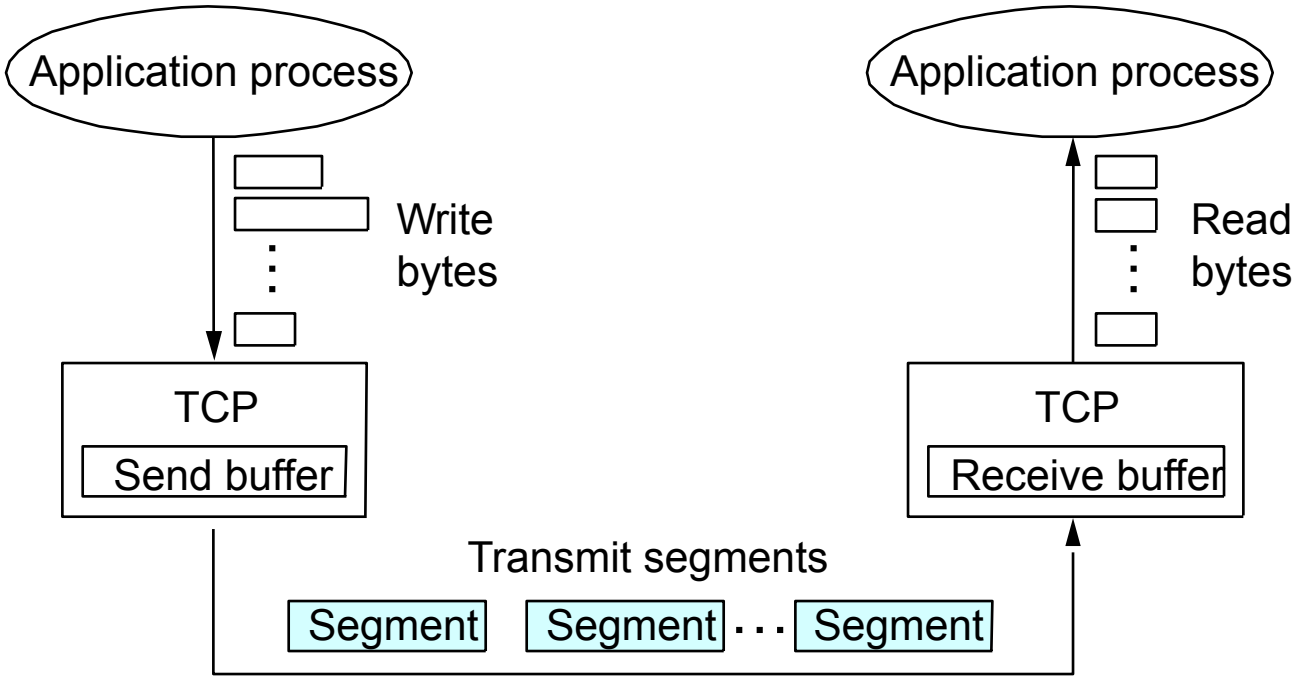


TCP Header Format

- Ports plus IP addresses identify a connection (4-tuple)

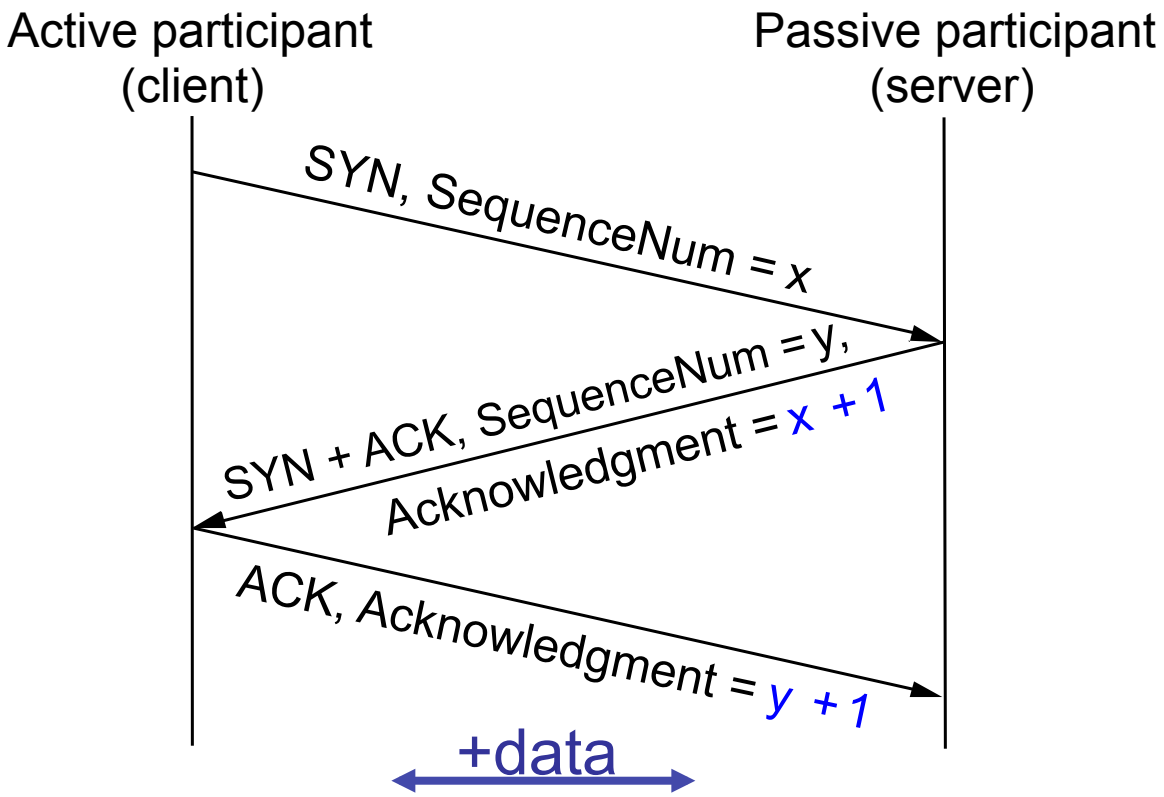


TCP Delivery



Three-Way Handshake

- Opens both directions for transfer



Congestion Control

- How fast should a sending host transmit data?
  - Not to fast, not to slow, just right…
- Should not be faster than the sender's share
  - Bandwidth allocation
- Should not be faster than the network can process
  - Congestion control
- Congestion control & bandwidth allocation are separate ideas, but frequently combined

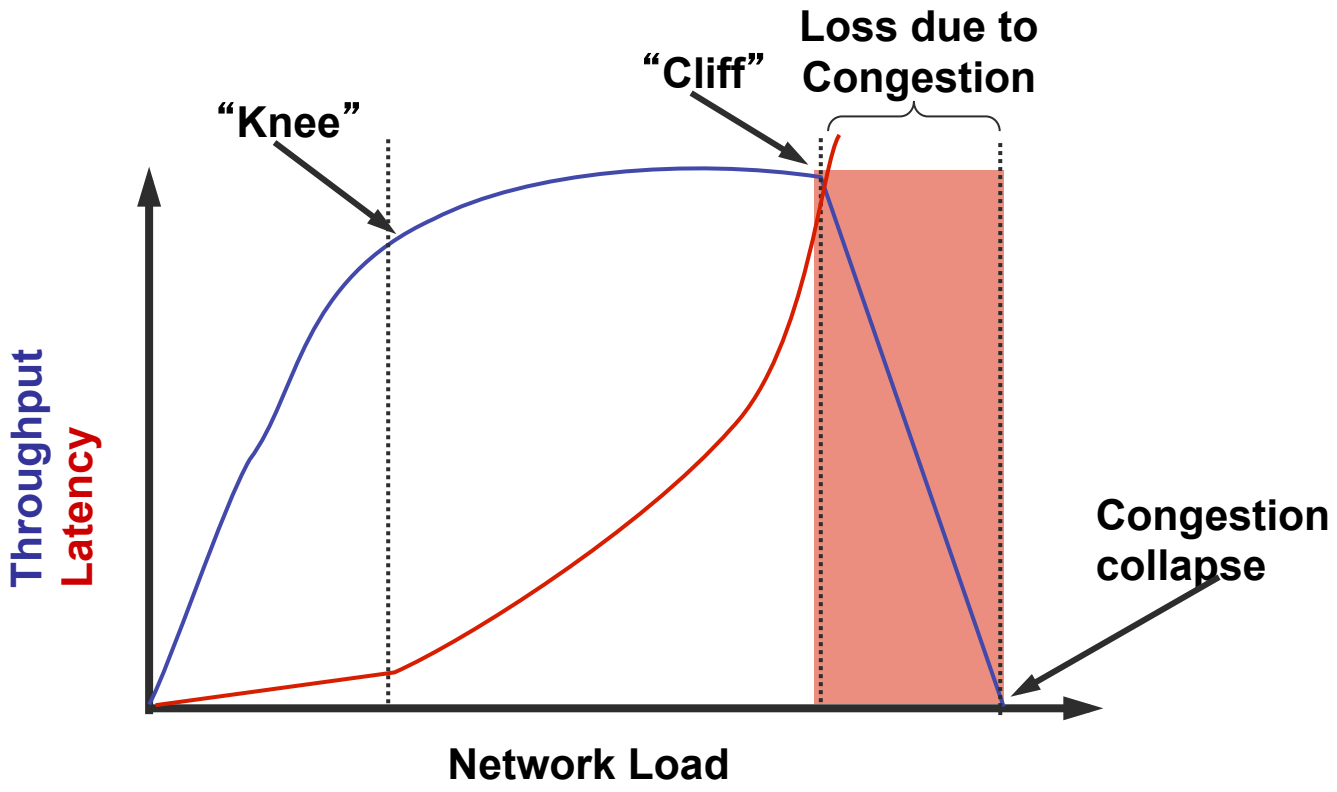
Congestion Collapse

- Rough definition: “When an increase in network load produces a decrease in useful work”

- Why does it happen?
  - Sender sends faster than bottleneck link speed
  - Packets queue until dropped
  - In response to packets being dropped, sender retransmits
  - All hosts repeat in steady state…

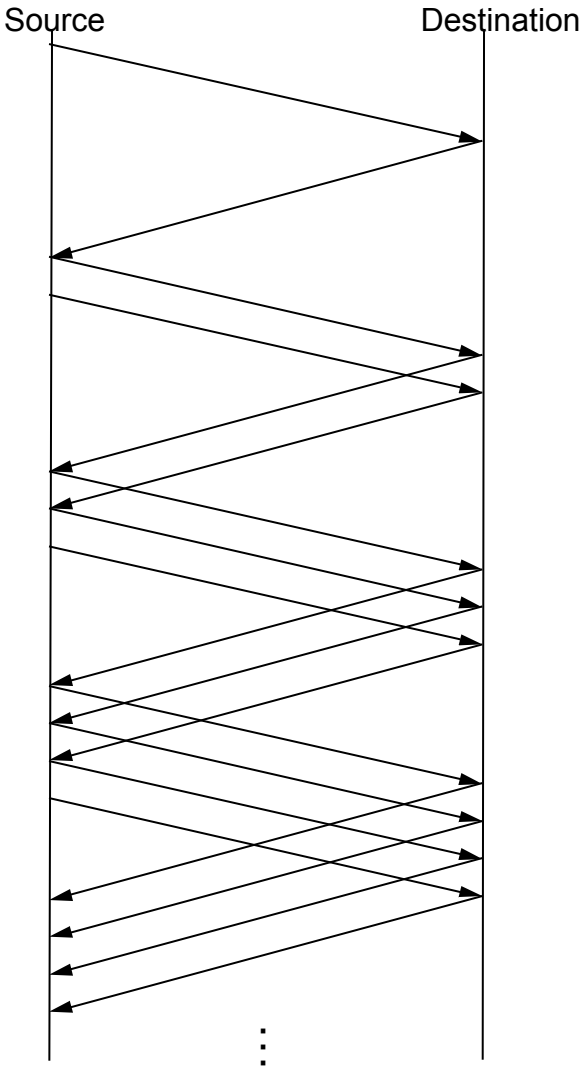
Proactive *vs.* Reactive

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

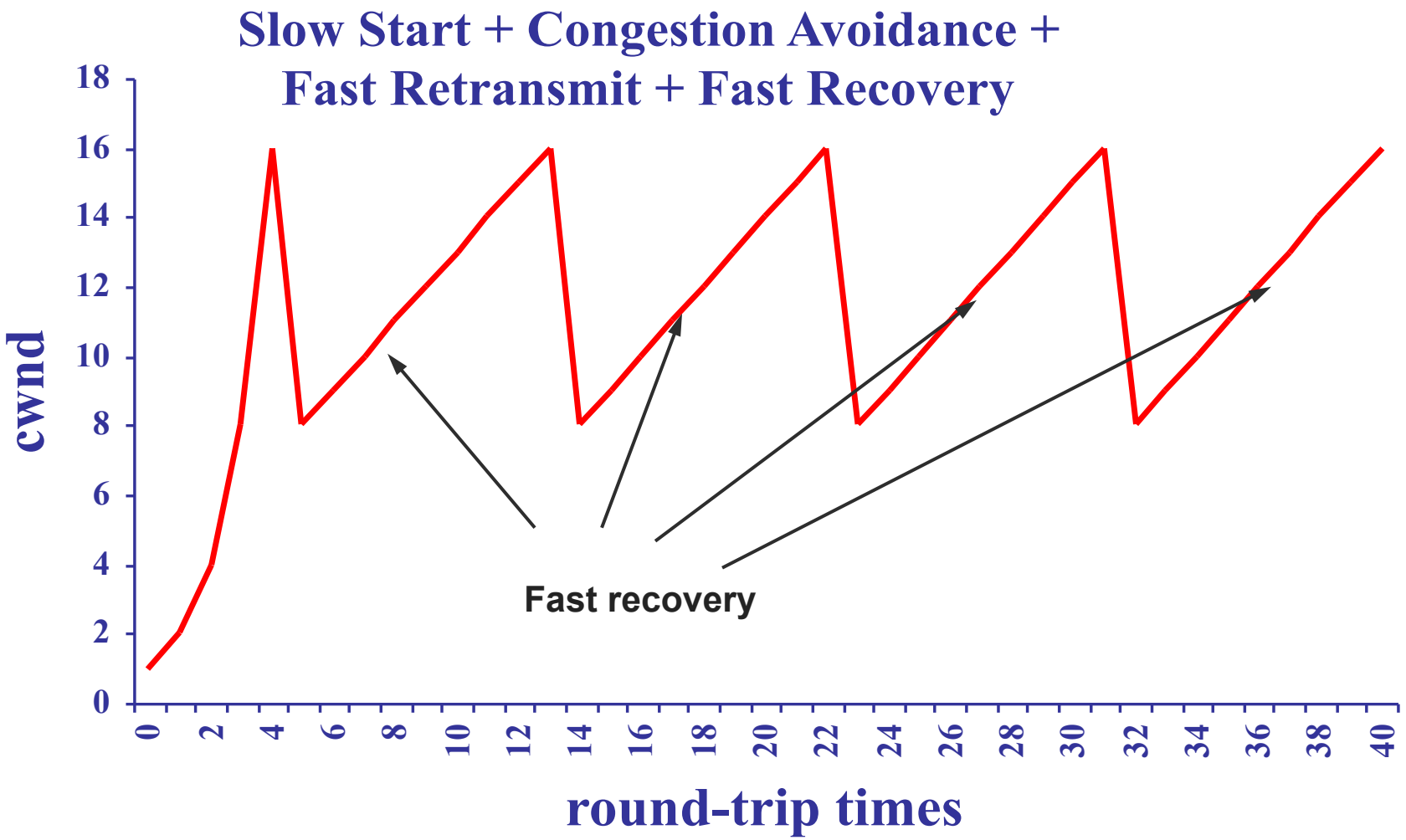


AIMD

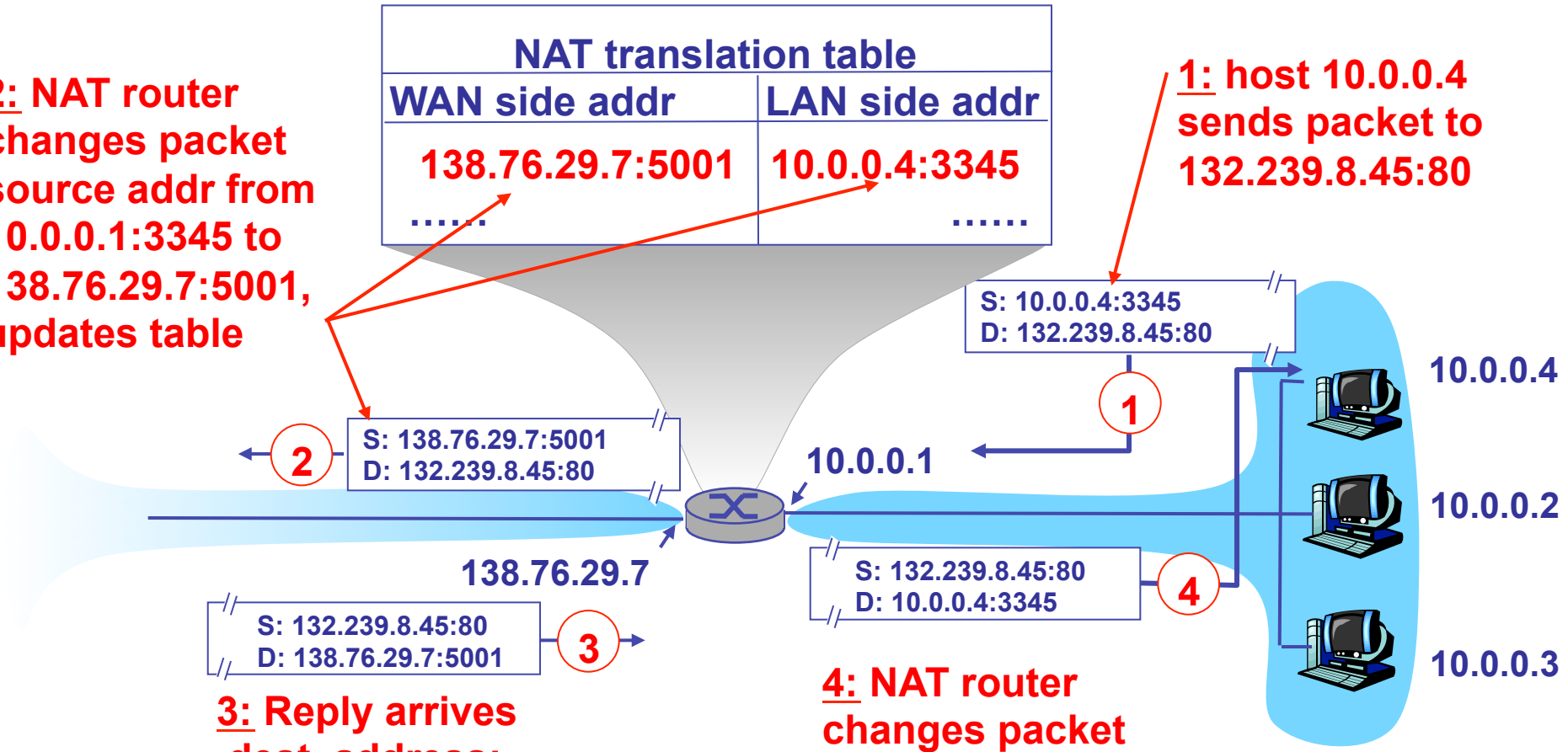
- Increase slowly while we believe there is bandwidth
  - Additive increase per RTT
  - *cwnd* += 1 full packet / RTT
- Decrease quickly when there is loss (went too far!)
  - Multiplicative decrease
  - *cwnd* /= 2



TCP's Various Phases



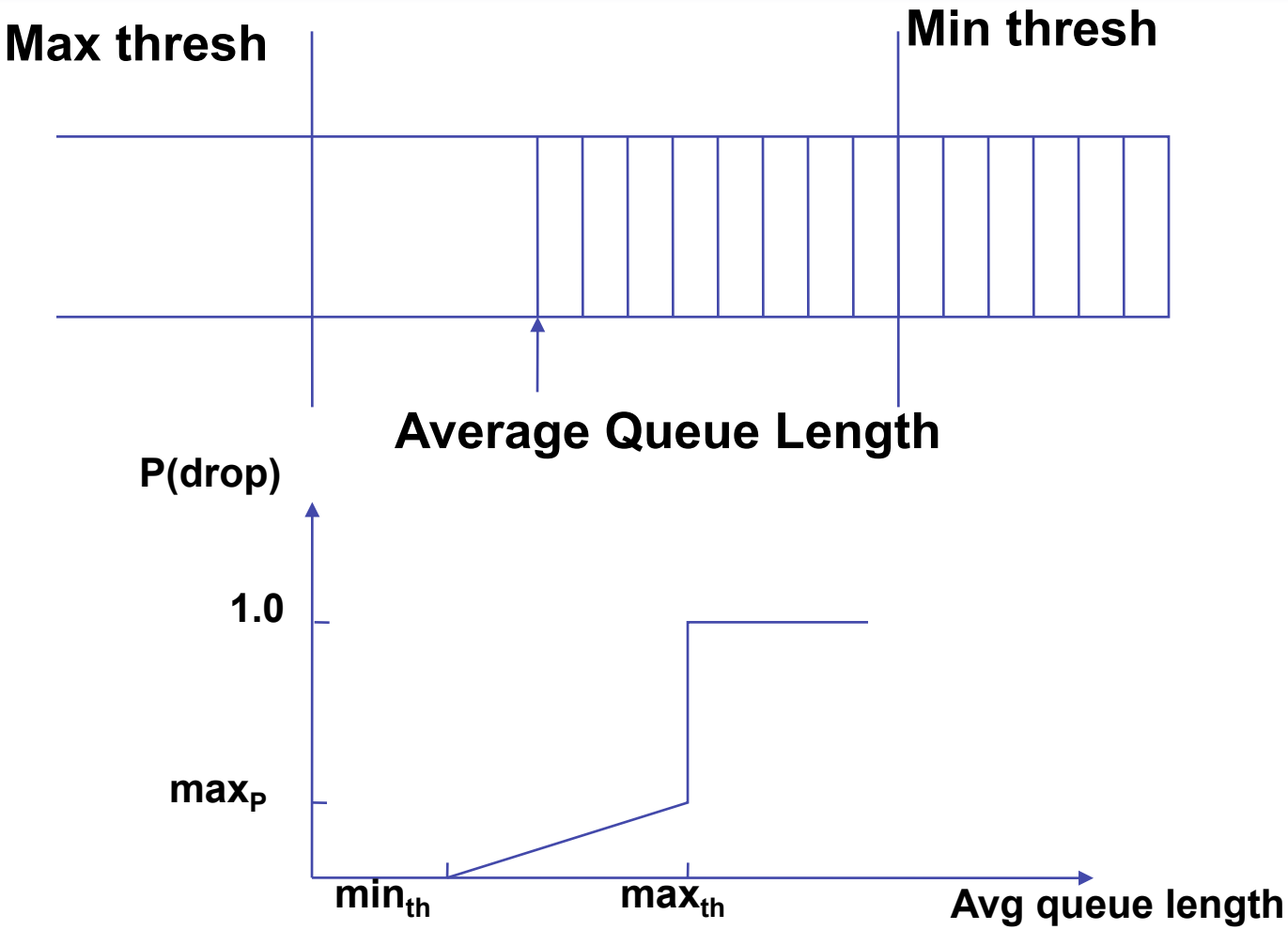
NA(p)T Example



Managing Overload

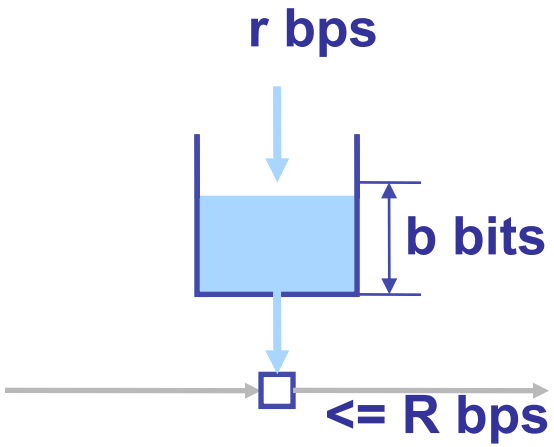
- Buffer Management
  - FIFO
  - RED
- Traffic Policing and Scheduling
  - Token Buckets
  - WFQ

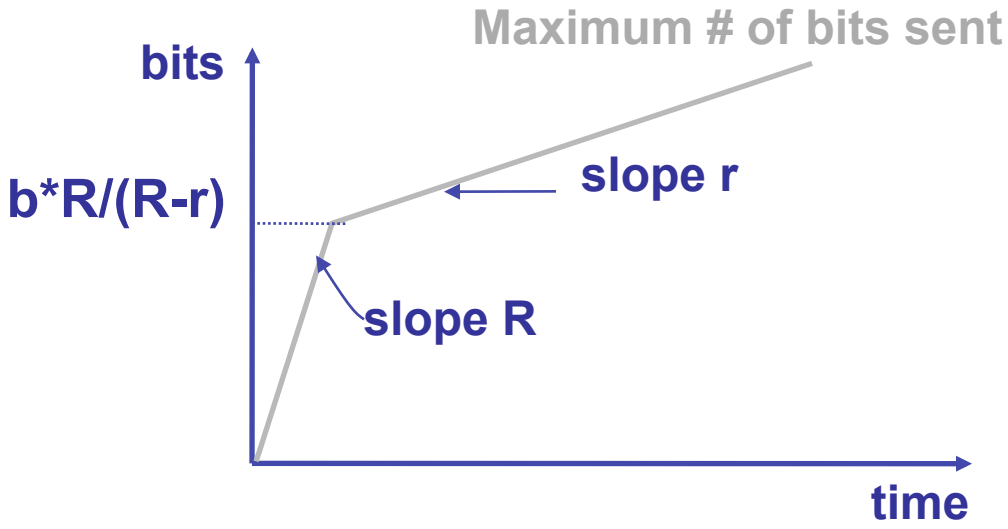
RED Operation



Token Bucket Basics

- Parameters
  - r – average rate, i.e., rate at which tokens fill the bucket
  - b – bucket depth
  - R – maximum link capacity or peak rate (optional parameter)
- A bit is transmitted only when there is an available token

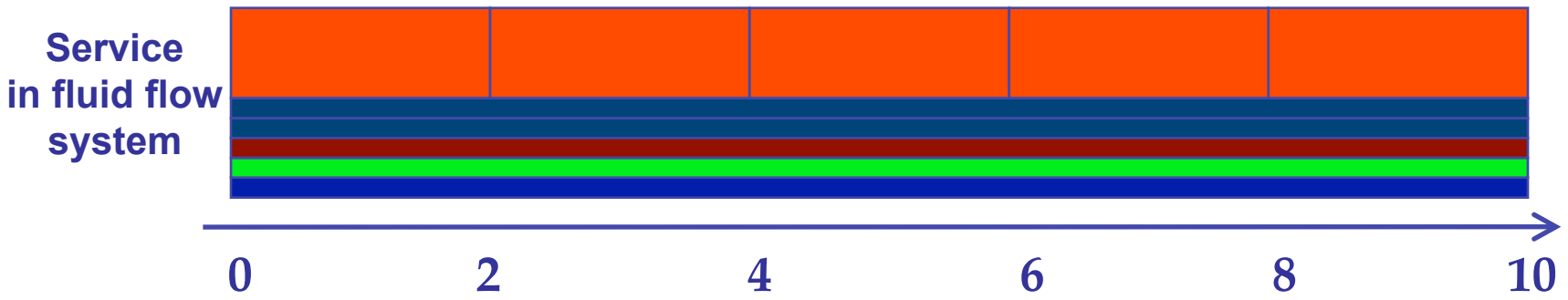




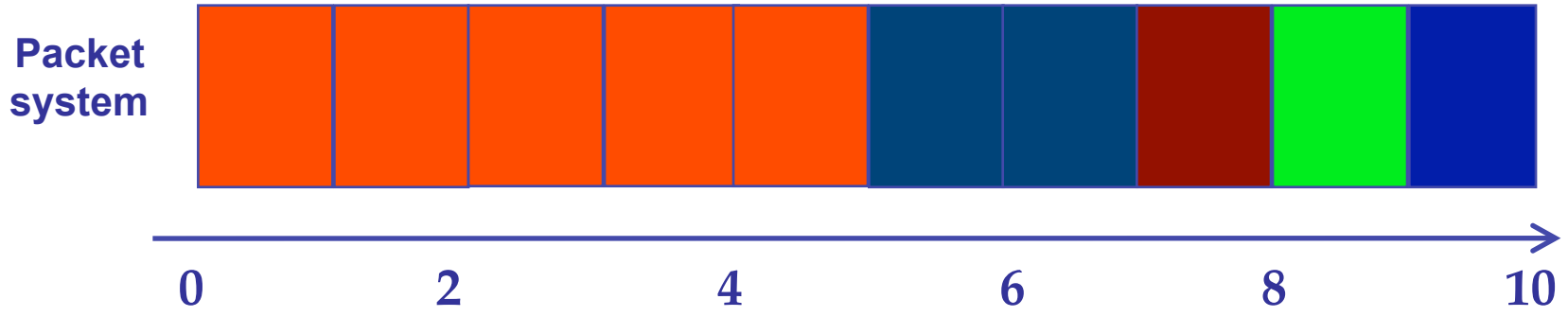
Fair Queuing

- 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

Packet-Based WFQ



- Select the first packet that finishes in the fluid flow system



Network-wide QoS

- Integrated services
  - Motivated by need for end-to-end guarantees
  - On-line negotiation of per-flow requirements
  - End-to-end per-router negotiation of resources
  - Complex
- Differentiated services
  - Motivated by economics (multi-tier pricing)
  - No per-flow state
  - Not end-to-end and not guaranteed services
  - Simple

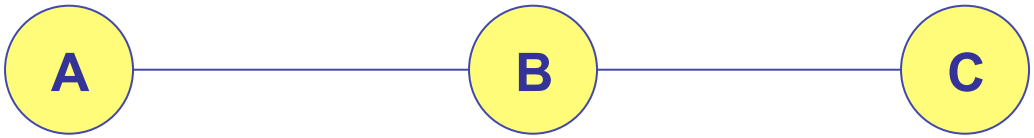
802.11 Summary

- Common technology for local-area wireless
- Uses CSMA/CA
- Needs to handle hidden terminal problem
- Challenges due to asymmetric ranges

CSMA/CA

- Cannot detect collision w/half-duplex radios
- 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.
  - Once channel becomes idle, the node waits for a randomly chosen duration before attempting to transmit.

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…

- Good luck finishing up Project 2
  - We'll announce Espresso prize winners at Final
- Final exam next Friday
  - 8-11am
  - WLH 2005 (right here)
  - You can bring one 2-sided 8.5x11 crib sheet
  - Must turn crib sheet in with your exam
- Have a great break!