15-441 Computer Networking

Lecture 22 – Queue Management and QoS

Congestion Control Review

- What is congestion control?

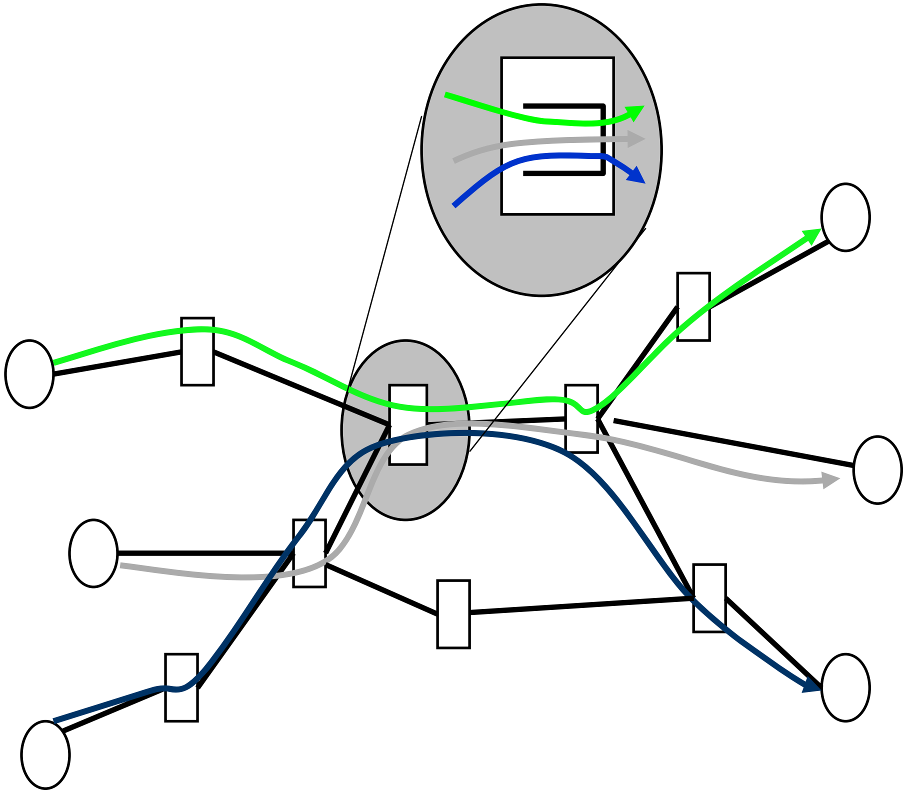
- What is the principle of TCP?

Traffic and Resource Management

- Resources statistically shared

  $$\sum \mathbf{Demand}_i(\mathbf{t}) > \mathbf{Resource}(\mathbf{t})$$

- Overload causes congestion
  - packet delayed or dropped
  - application performance suffer
- Local vs. network wide
- Transient vs. persistent
- Challenge
  - high resource utilization
  - high application performance



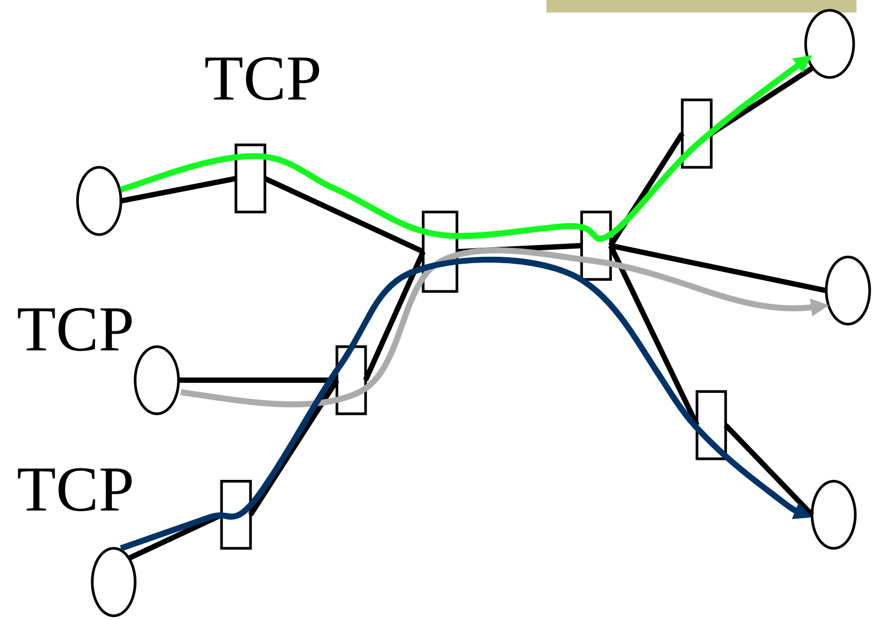
Resource Management Approaches

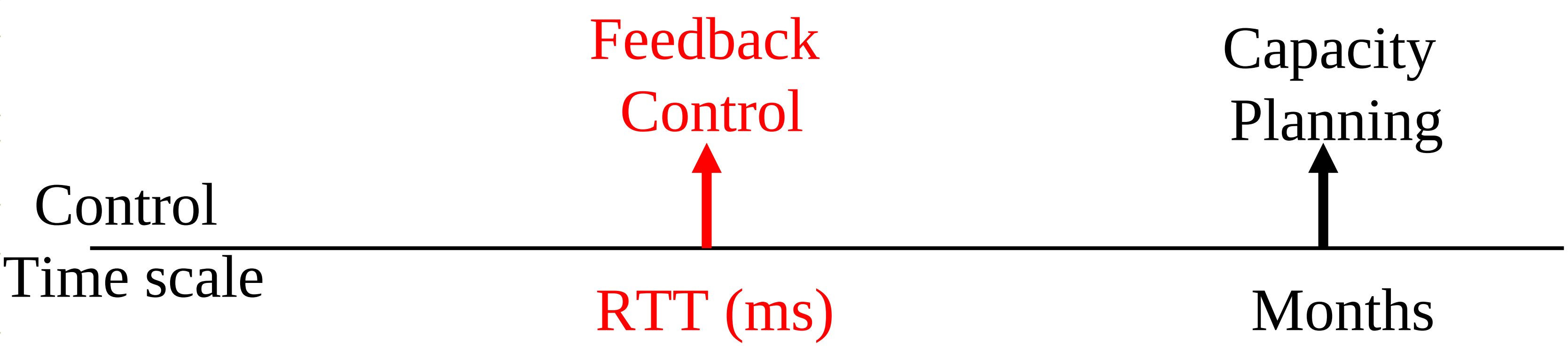
$$\sum \mathbf{Demand}_i(\mathbf{t}) > \mathbf{Resource}(\mathbf{t})$$

- Increase resources
  - install new links, faster routers
  - capacity planning, provisioning, traffic engineering
  - happen at longer timescale
- Reduce or delay demand
  - Reactive approach: encourage everyone to reduce or delay demand
  - Reservation approach: some requests will be rejected by the network

Congestion Control in Today's Internet

- End-system-only solution (TCP)
  - dynamically estimates network state
  - packet loss signals congestion
  - reduces transmission rate in presence of congestion
  - routers play little role



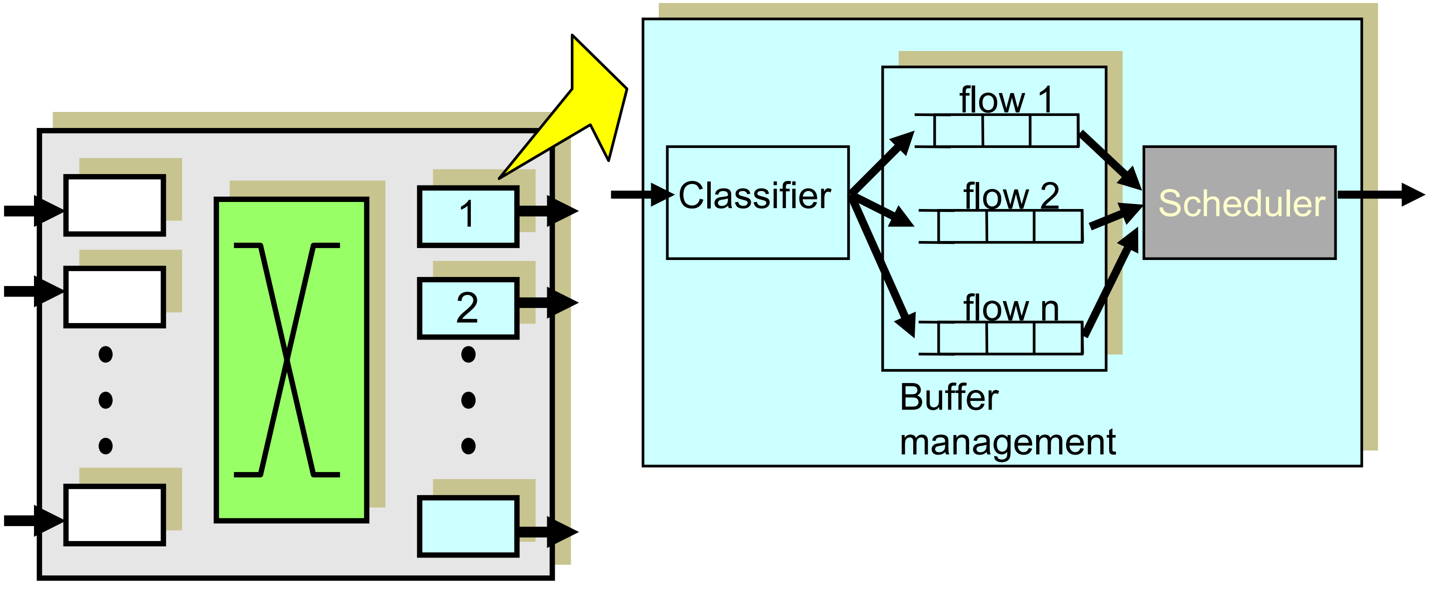


More Ideas on Traffic Management

- Improve TCP
  - Stay with end-point only architecture
- Enhance routers to help TCP
  - Random Early Discard
- Enhance routers to control traffic
  - Rate limiting
  - Fair Queueing
- Provide QoS by limiting congestion

Router Mechanisms

- Buffer management: when and which packet to drop?
- Scheduling: which packet to transmit next?



Typical Internet Queuing

- FIFO + drop-tail
  - Simplest choice
  - Used widely in the Internet
- FIFO (first-in-first-out)
  - Implies single class of traffic
- Drop-tail
  - Arriving packets get dropped when queue is full regardless of flow or importance
- Important distinction:
  - FIFO: scheduling discipline
  - Drop-tail: drop policy

FIFO + Drop-tail Problems

- Leaves responsibility of congestion control completely to the edges (e.g., TCP)
- Does not separate between different flows
- No policing: send more packets → get more service
- Synchronization: end hosts react to same events

FIFO + Drop-tail Problems

- Full queues
  - Routers are forced to have have large queues to maintain high utilizations
  - TCP detects congestion from loss
    - Forces network to have long standing queues in steady-state
- Lock-out problem
  - Drop-tail routers treat bursty traffic poorly
  - Traffic gets synchronized easily → allows a few flows to monopolize the queue space

Active Queue Management

- Design active router queue management to aid congestion control
- Why?
  - Router has unified view of queuing behavior
  - Routers see actual queue occupancy (distinguish queue delay and propagation delay)
  - Routers can decide on transient congestion, based on workload

Design Objectives

- Keep throughput high and delay low
  - High power (throughput/delay)
- Accommodate bursts
- Queue size should reflect ability to accept bursts rather than steady-state queuing
- Improve TCP performance with minimal hardware changes

Lock-out Problem

- Random drop
  - Packet arriving when queue is full causes some random packet to be dropped
- Drop front
  - On full queue, drop packet at head of queue
- Random drop and drop front solve the lock-out problem but not the full-queues problem

Full Queues Problem

- Drop packets before queue becomes full (early drop)
- Intuition: notify senders of incipient congestion
  - Example: early random drop (ERD):
    - If qlen > drop level, drop each new packet with fixed probability *p*
    - Does not control misbehaving users

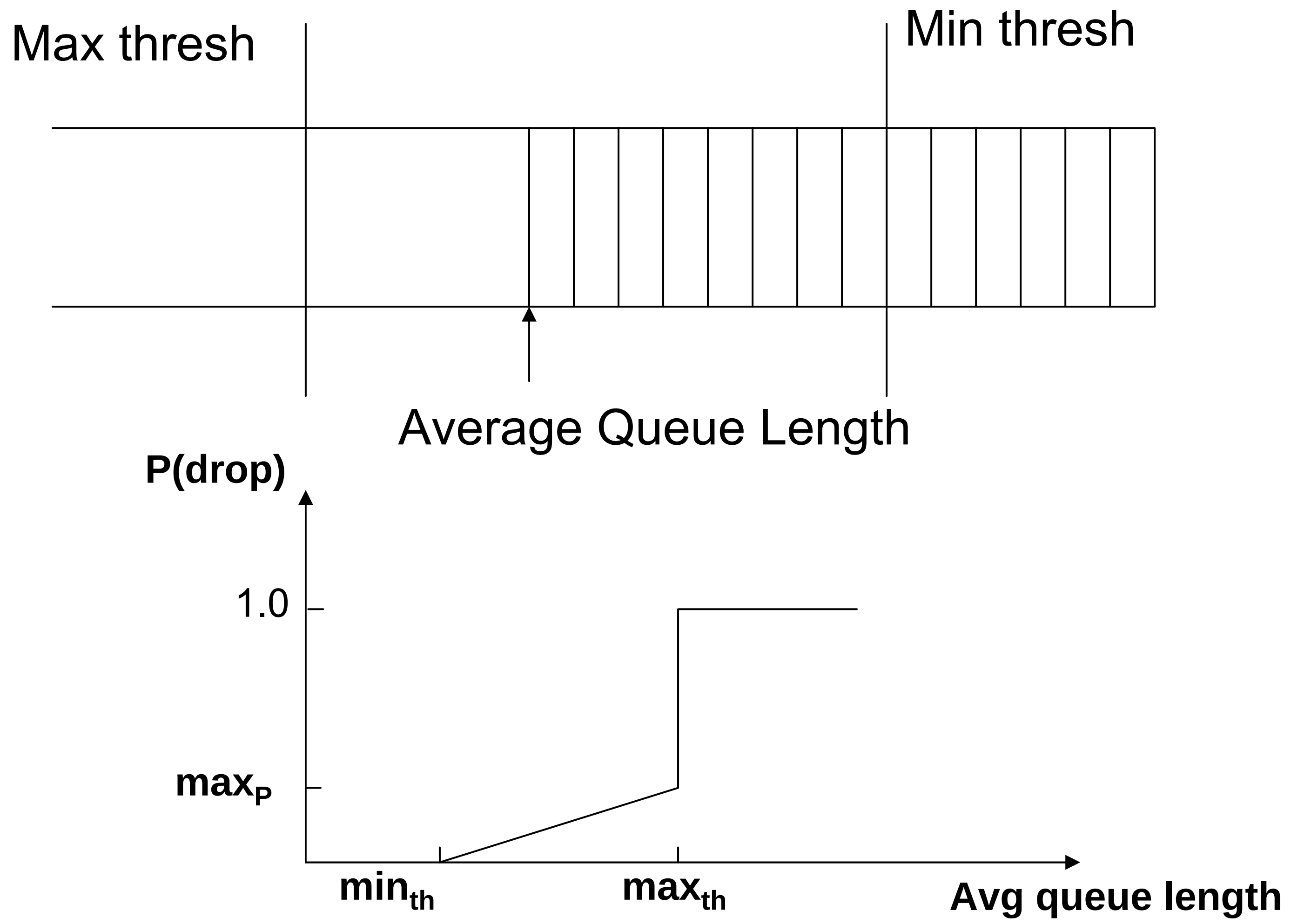
Random Early Detection (RED)

- Detect incipient congestion
- Assume hosts respond to lost packets
- Avoid window synchronization
  - Randomly mark packets
- Avoid bias against bursty traffic

RED Algorithm

- Maintain running average of queue length
- If avg < min_{th} do nothing
  - Low queuing, send packets through
- If avg > max_{th}, drop packet
  - Protection from misbehaving sources
- Else mark packet in a manner proportional to queue length
  - Notify sources of incipient congestion

RED Operation



Explicit Congestion Notification (ECN) [Floyd and Ramakrishnan 98]

- Traditional mechanism
  - packet drop as implicit congestion signal to end systems
  - TCP will slow down
- Works well for bulk data transfer
- Does not work well for delay sensitive applications
  - audio, WEB, telnet
- Explicit Congestion Notification (ECN)
  - borrow ideas from DECBit
  - use two bits in IP header
    - ECN-Capable Transport (ECT) bit set by sender
    - Congestion Experienced (CE) bit set by router

Congestion Control Summary

- Architecture: end system detects congestion and slow down
- Starting point:
  - slow start/congestion avoidance
    - packet drop detected by retransmission timeout RTO as congestion signal
  - fast retransmission/fast recovery
    - packet drop detected by three duplicate acks
- TCP Improvement:
  - NewReno: better handle multiple losses in one round trip
  - SACK: better feedback to source
  - NetReno: reduce RTO in high loss rate, small window scenario
  - FACK, NetReno: better end system control law

Congestion Control Summary (II)

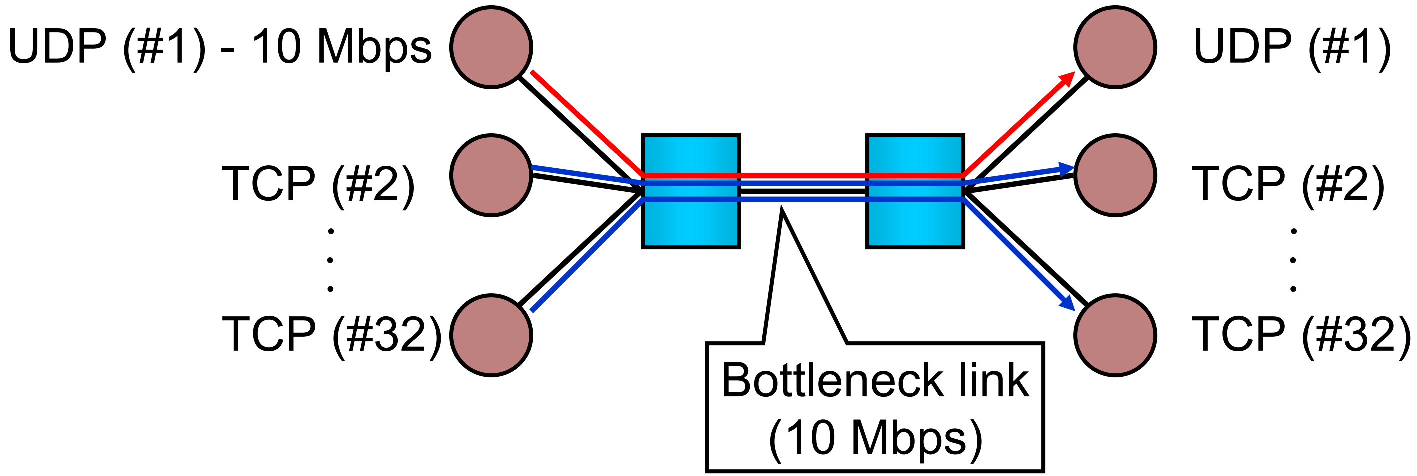
- Router support
  - RED: early signaling
  - ECN: explicit signaling

What are the Problems?

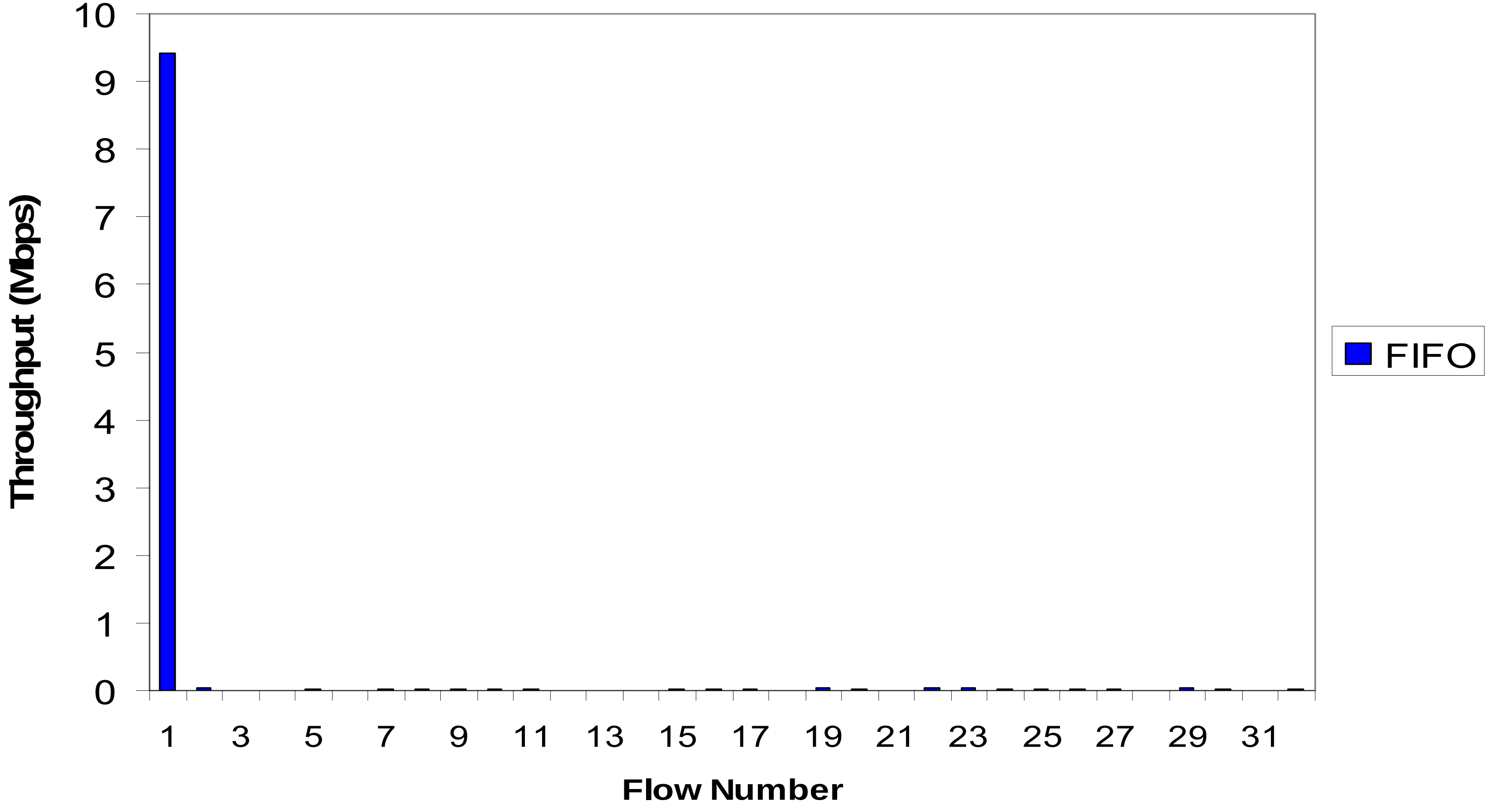
- Works only if most sources implement TCP
  - most sources are cooperative
  - most sources implement homogeneous/compatible control law
    - compatible means less aggressive than TCP
- What if sources do not play by the rule?

An Example

- 1 UDP (10 Mbps) and 31 TCPs sharing a 10 Mbps line



Throughput of UDP and TCP Flows With FIFO



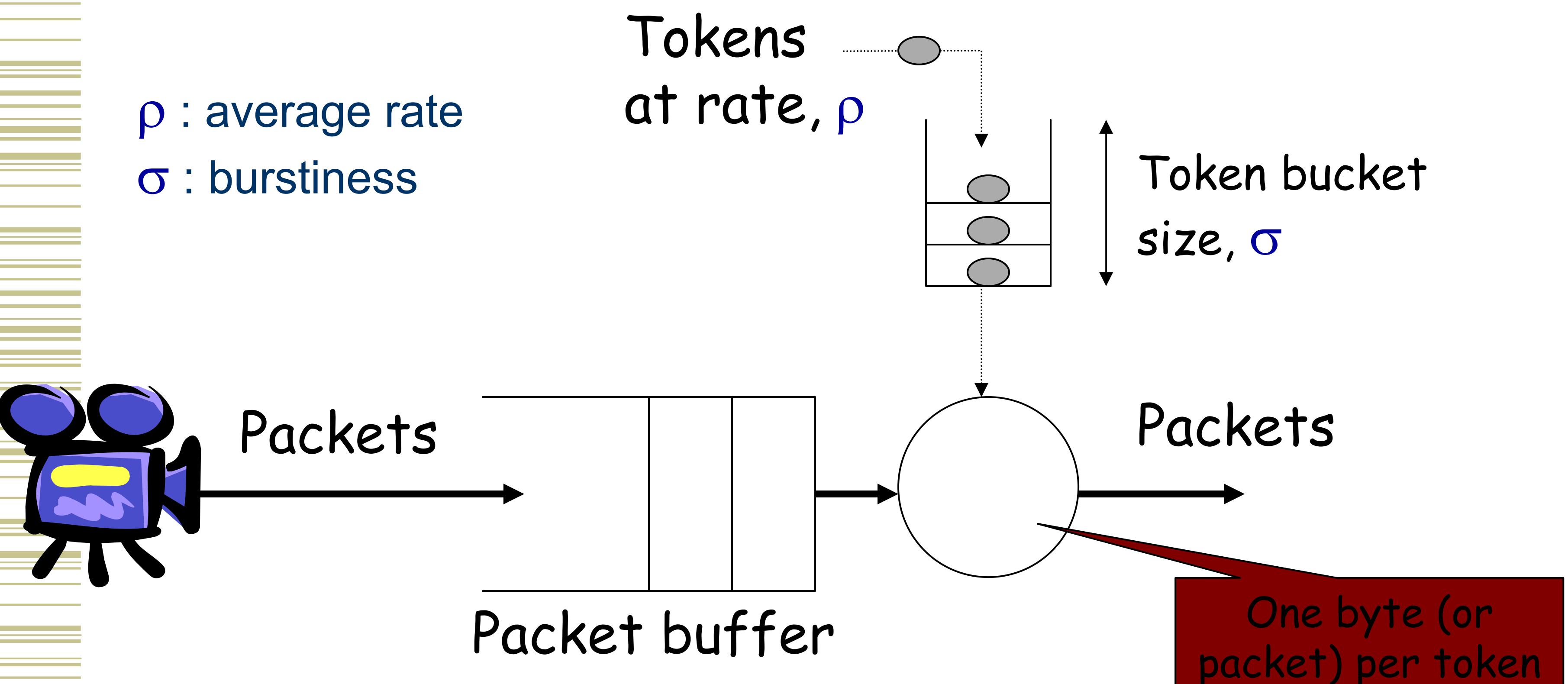
What Is the Solution?

- Enforcement mechanism inside the network
  - Rate limiting, scheduling

The Token Bucket

ρ : average rate

σ : burstiness

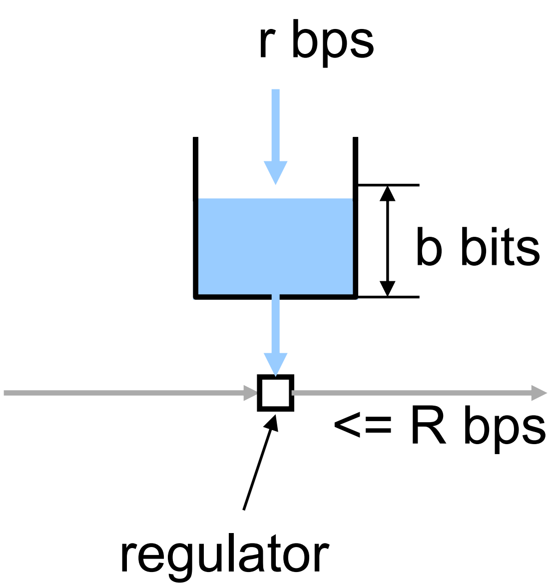


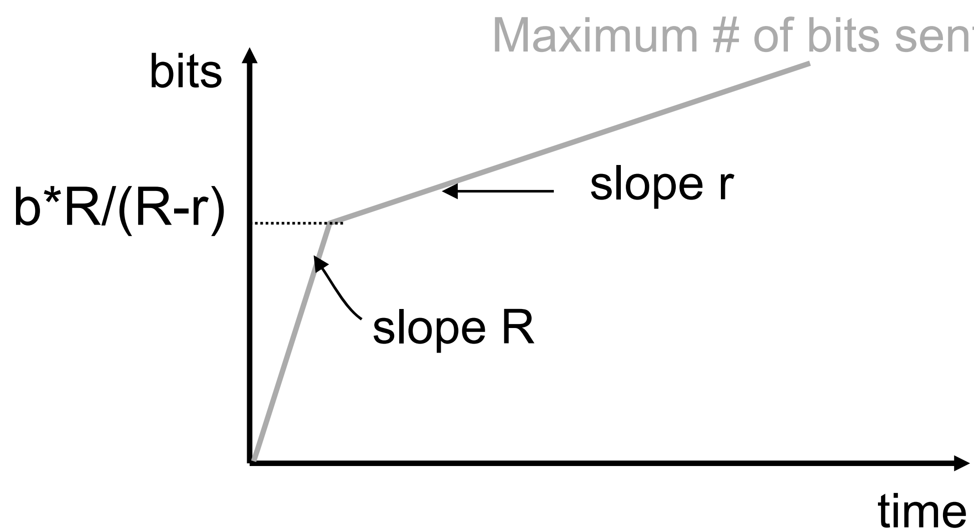
Bits sent between times s and t: $A(s,t) \leq \sigma + \rho(t-s)$

Nick McKeown

Token Bucket

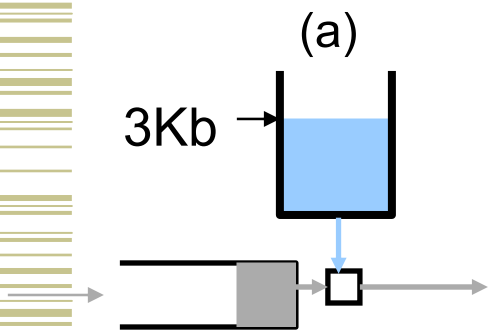
- Parameters
  - r – average rate, i.e., rate at which tokens fill the bucket
  - b – bucket depth
  - R – maximum link capacity or peak rate (optional parameter)
- A bit is transmitted only when there is an available token



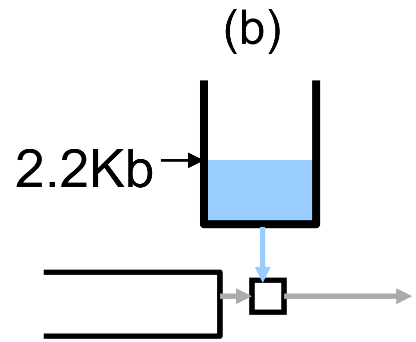


Traffic Enforcement: Example

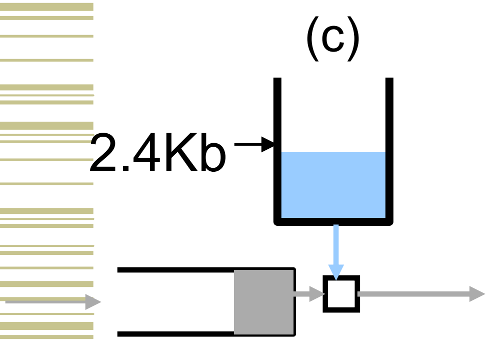
- r = 100 Kbps; b = 3 Kb; R = 500 Kbps



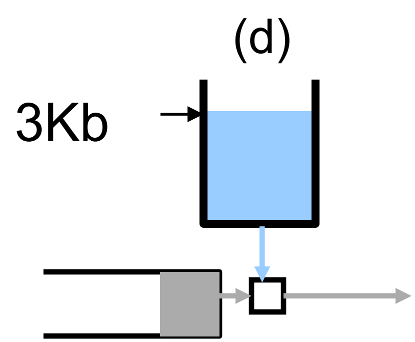
T = 0 : 1Kb packet arrives



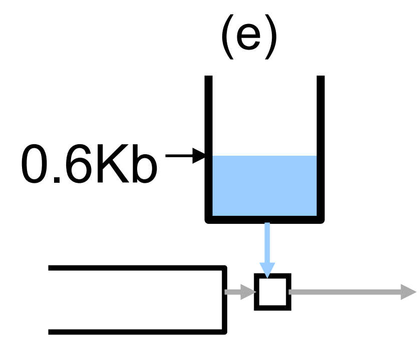
T = 2ms : packet transmitted
b = 3Kb – 1Kb + 2ms*100Kbps = 2.2Kb



T = 4ms : 3Kb packet arrives



T = 10ms :

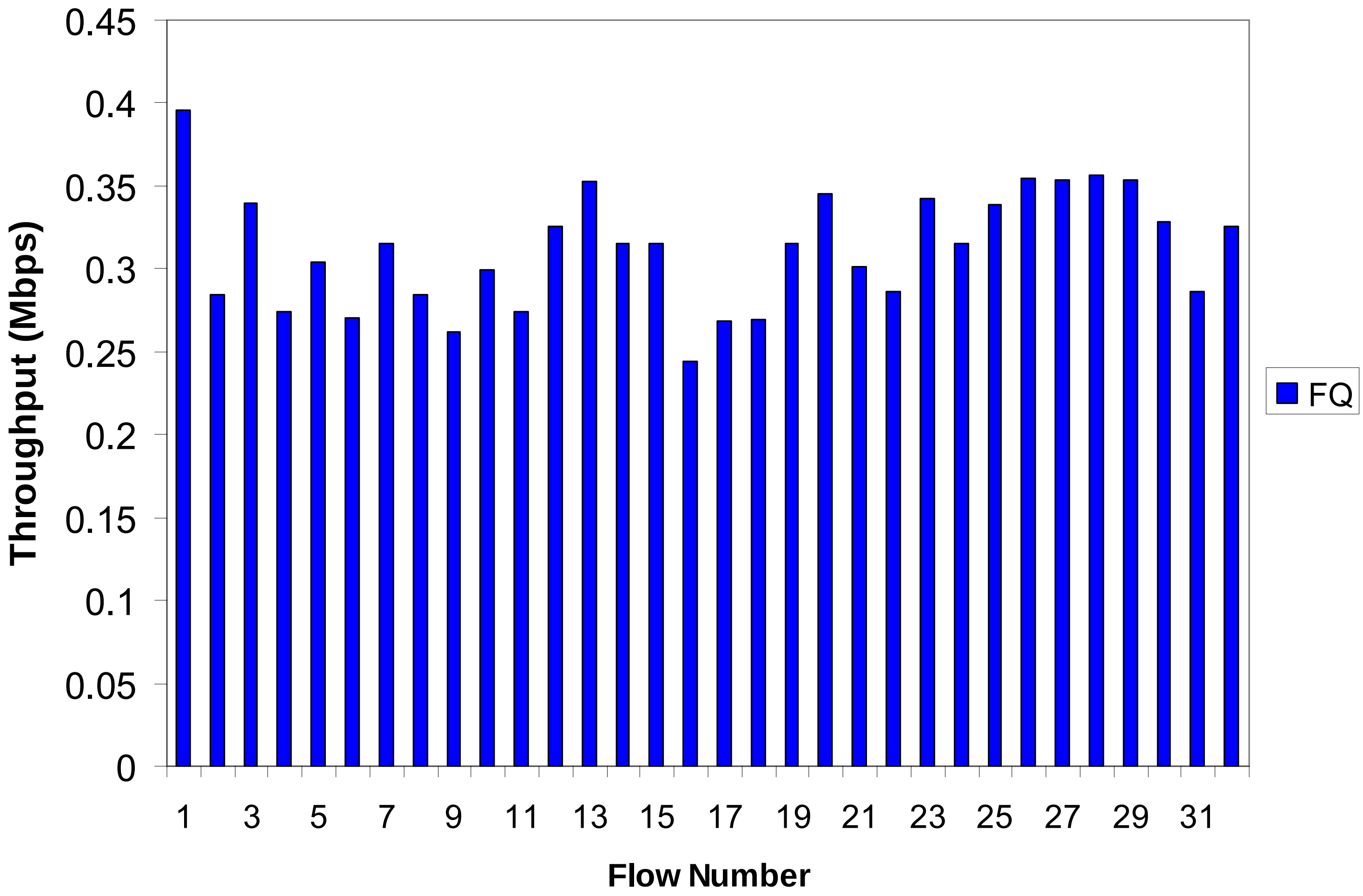


T = 16ms : packet transmitted

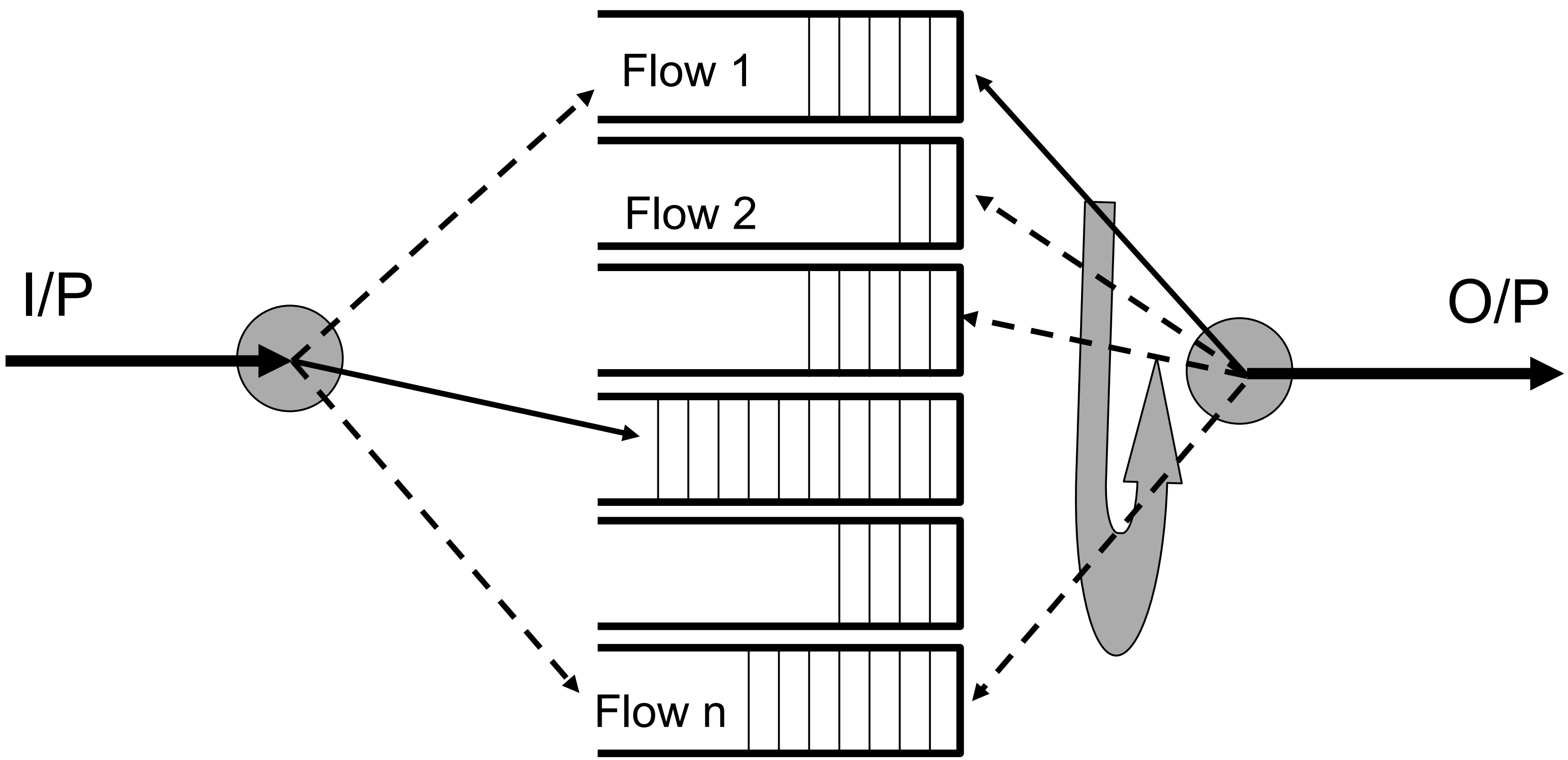
Rate-Limiting and Scheduling

- Rate-limiting: limit the rate of one flow regardless the load in the network
- Scheduling: dynamically allocates resources when multiple flows competing

Example Outcome: Throughput of TCP and UDP Flows With Fair Queueing Router



Fair Queueing



Variation: Weighted Fair Queuing (WFQ)

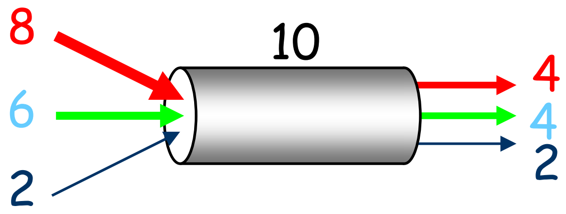
Fair Queueing

- 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 Queueing (WFQ) – associate a weight with each flow

Fair Rate Computation: Example 1

- If link congested, compute f such that

$$\sum_i \min(r_i, f) = C$$

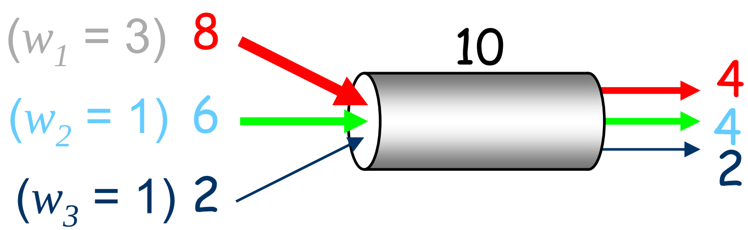


$f = 4$:
min(8, 4) = 4
min(6, 4) = 4
min(2, 4) = 2

Fair Rate Computation: Example 2

- Associate a weight w_i with each flow i
- If link congested, compute f such that

$$\sum_i \min(r_i, f \times w_i) = C$$



f = 2:
min(8, 2*3) = 6
min(6, 2*1) = 2
min(2, 2*1) = 2

Flow i is guaranteed to be allocated a rate >= w_i*C/($\Sigma_k w_k$)

⇓

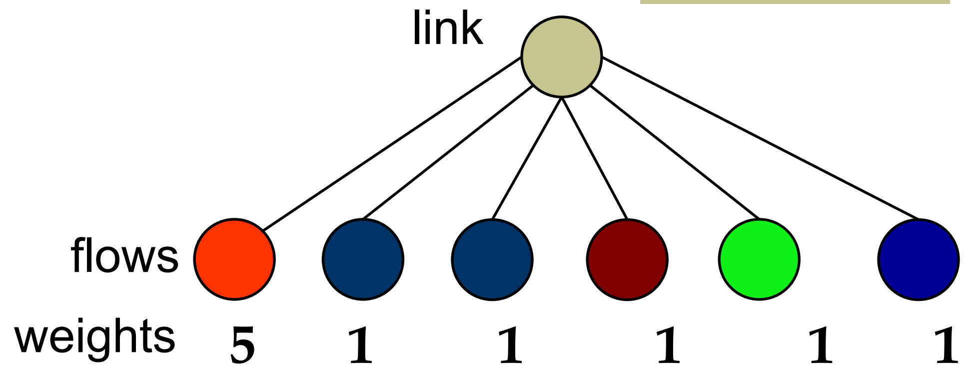
If $\Sigma_k w_k <= C$, flow i is guaranteed to be allocated a rate >= w_i

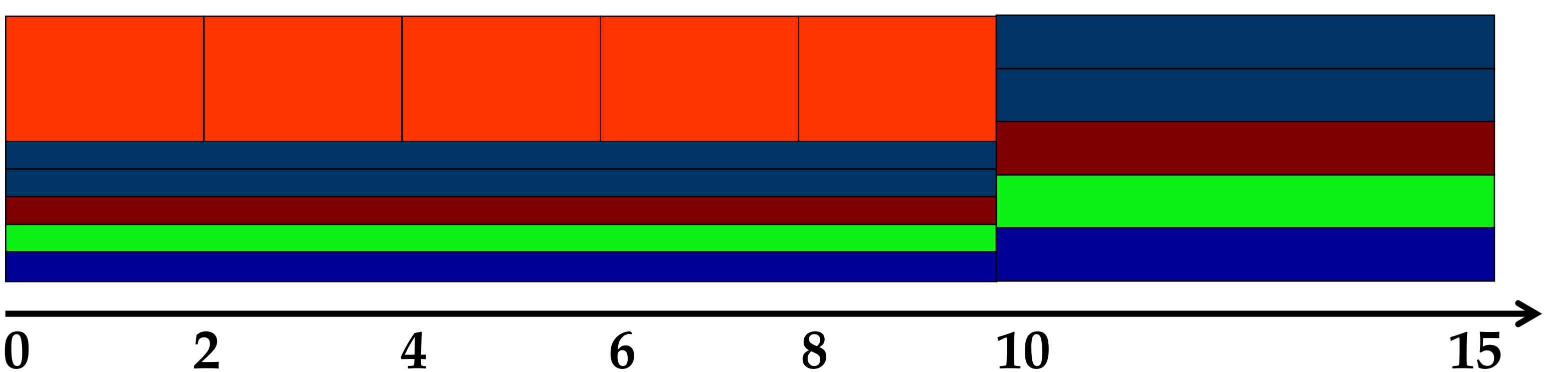
Fluid Flow System

- Flows can be served one bit at a time
- WFQ can be implemented using bit-by-bit weighted round robin
  - During each round from each flow that has data to send, send a number of bits equal to the flow’s weight

Fluid Flow System: Example

- Red flow has packets backlogged between time 0 and 10
  - Backlogged flow → flow's queue not empty
- Other flows have packets continuously backlogged
- All packets have the same size

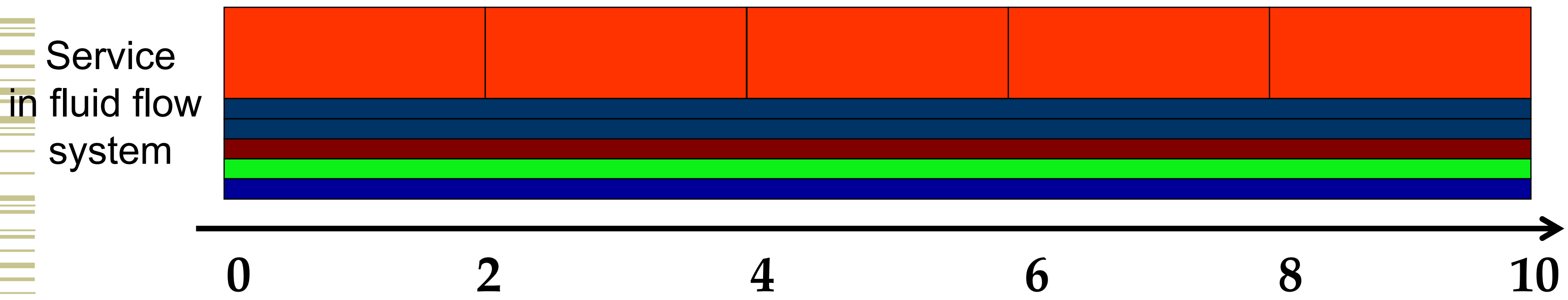




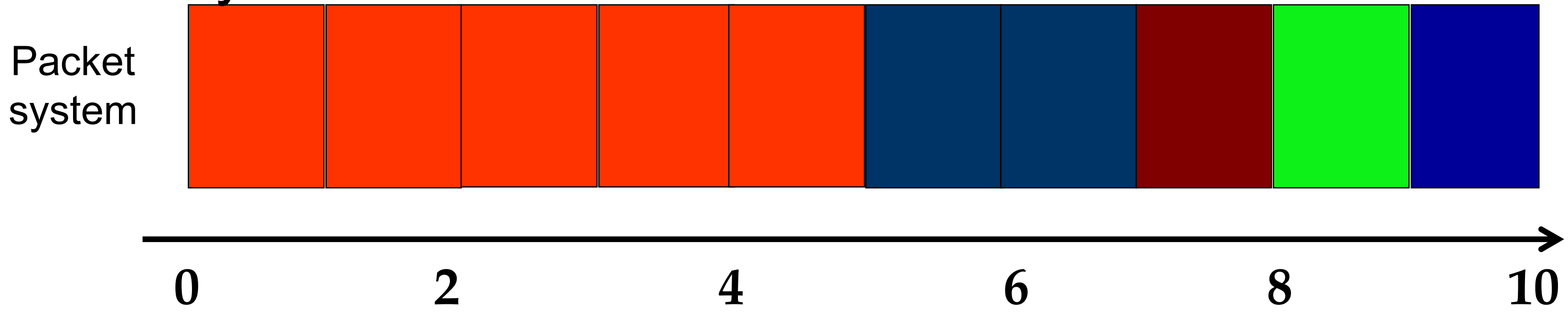
Implementation In Packet System

- Packet (Real) system: packet transmission cannot be preempted. Why?
- Solution: serve packets in the order in which they would have finished being transmitted in the fluid flow system

Packet System: Example



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



Limitations of Resource Management Architecture Today (II)

- IP provides only best effort service
- IP does not participate in resource management, thus cannot provide Quality of Service (QoS)
- Quality of Service
  - flow-based vs. class-based
  - absolute vs. relative (assurance vs. differentiation)
    - absolute: performance assurance regardless of behaviors of other traffic
    - relative: QoS defined with respect to other flows, e.g. priority, weighted fair share

Resource Management Approaches

$$\sum \mathbf{Demand}_i(\mathbf{t}) > \mathbf{Resource}(\mathbf{t})$$

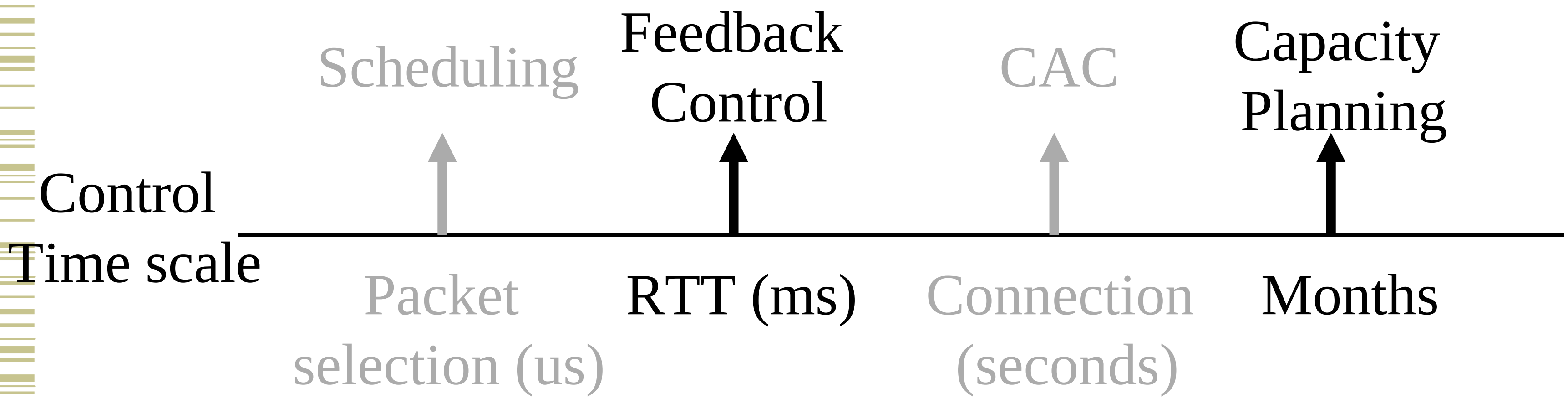
- Increase resources
  - install new links, faster routers
  - capacity planning, provisioning, traffic engineering
  - happen at longer timescale
- Reduce or delay demand
  - Reactive approach: encourage everyone to reduce or delay demand
  - Reservation approach: some requests will be rejected by the network

Components of Integrated Services Network

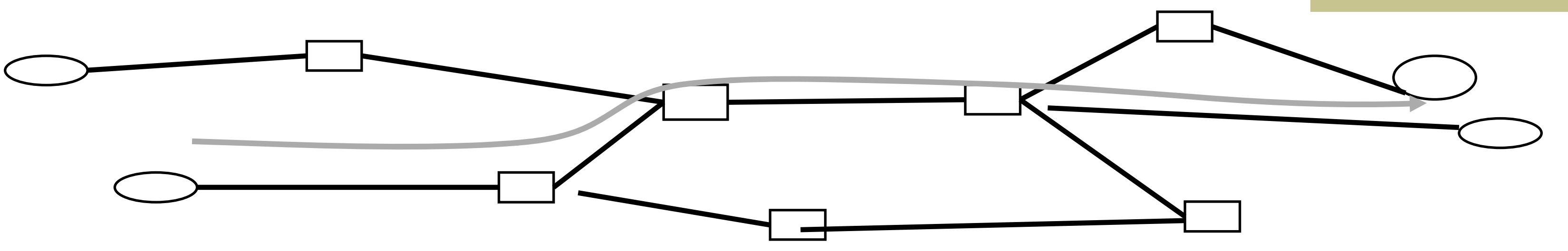
- Service models
  - end-to-end per flow guaranteed, controlled load, best-effort
  - hierarchical link-sharing
- Protocols and mechanisms
  - RSVP: signaling protocol to set-up and tear-down per flow reservation state
  - Admission control
    - determines whether there is enough resource and policy allows
  - Traffic control
    - classify packet to each flow
    - schedule packets transmission according to per flow state

Control Time Scale

- Two levels of control
  - connection admission control (CAC)
  - packet scheduling algorithm



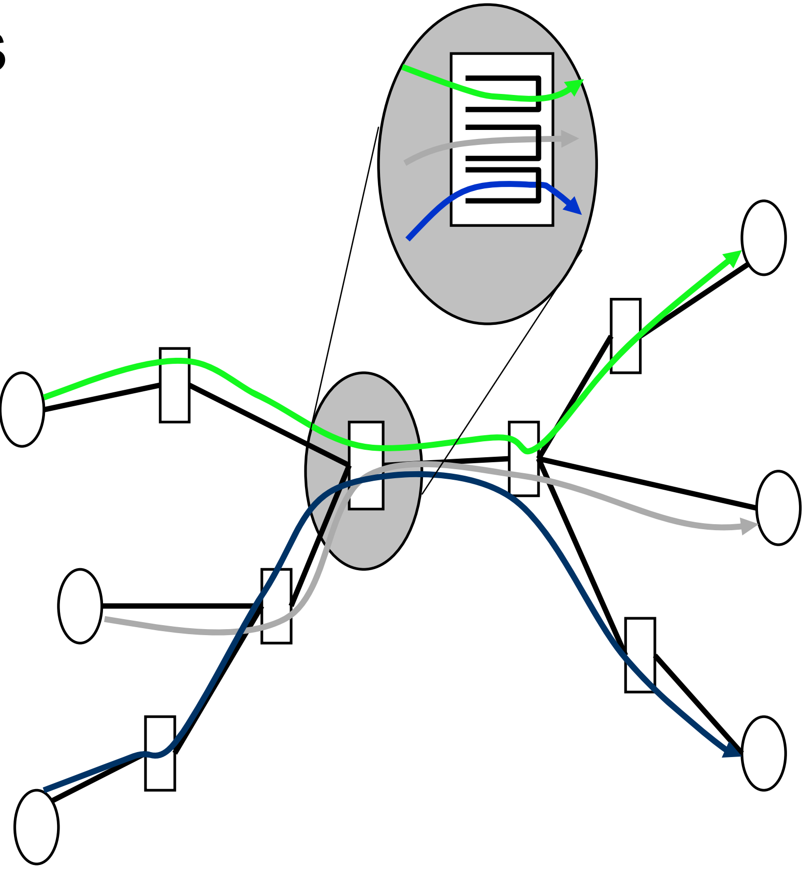
Observations of Reservation Scheme



- Network recognizes a higher abstraction: flow, session, virtual circuit, connection
  - a set of related packets that network treats as a group
  - dynamic created/deleted (switched vs permanent)
  - fixed or stable path for the flow
- Connection-oriented vs. connectionless
  - one of the most bitter, long-lasting religious contention points in computer networks

Integrated Services Network

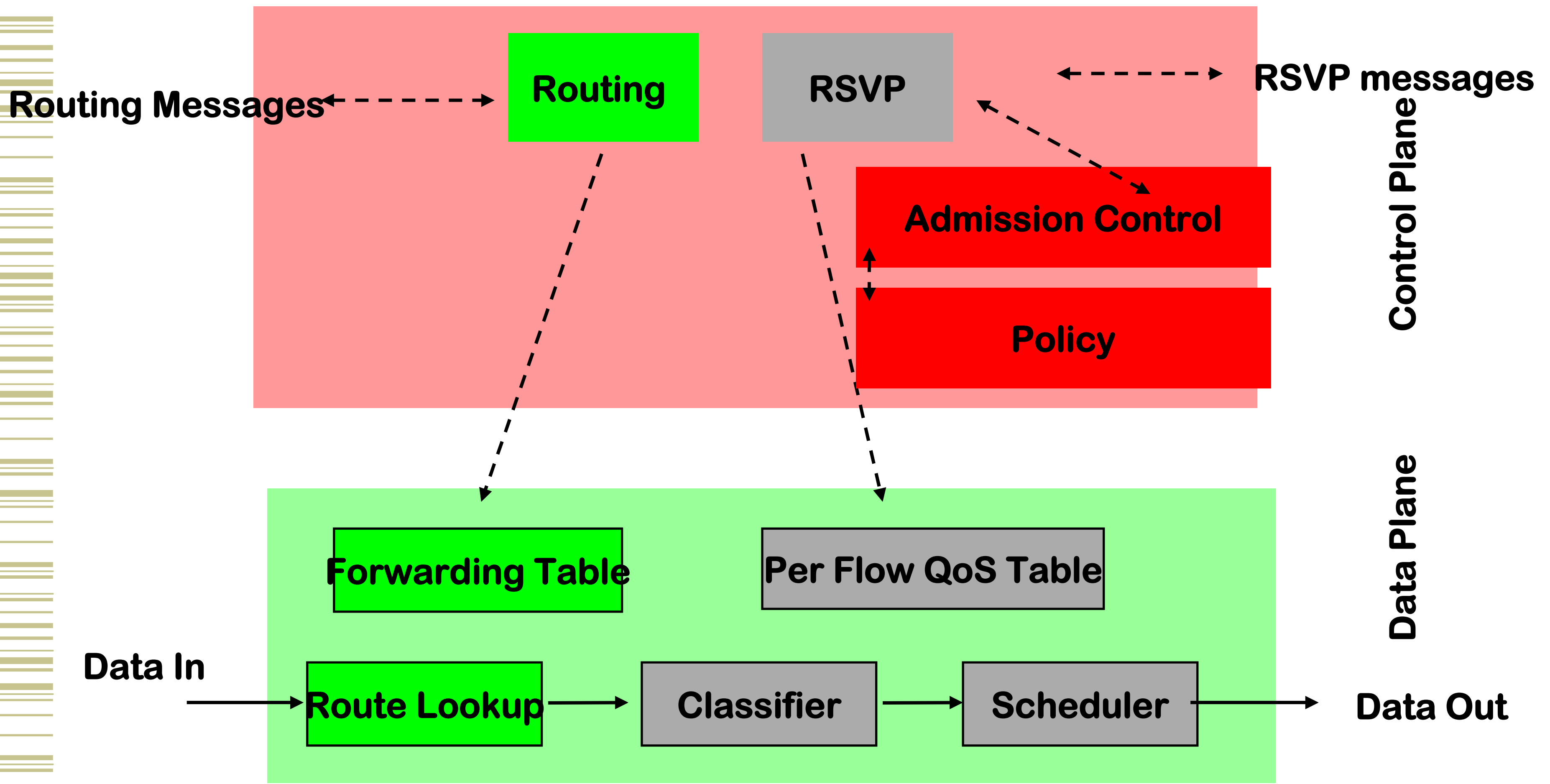
- Flow or session as QoS abstractions
- Each flow has a fixed or stable path
- Routers along the path maintain the state of the flow



Components of Flow QoS Network

- Service models: end-to-end per flow
  - IETF Intserv: guaranteed, controlled load, best-effort
- Protocols and mechanisms
  - Signaling protocol: set-up and tear-down per flow state
    - IETF: RSVP
  - Admission control
    - determines whether there is enough resource inside network
  - Traffic control
    - classify packet to each flow
    - schedule packets transmission according to per flow state

How Things Fit Together



Packet Classification Algorithm

- Map a packet to a flow
- Flow identified by
  - <srcIP, destIP, srcPort, destPort, protocol>
- Sometimes only prefixes of srcIP, destIP are specified
  - e.g <128.2.x.x, 140.247.x.x, x, 80, 6>
    - all web traffic from CMU to Harvard
- Different fields have different matching rules
  - IP addresses: longest prefix match
  - port numbers: exact match or range match
  - protocol: exact match

Resource Management Approaches

$$\sum \mathbf{Demand_i(t) > Resource(t)}$$

- Increase resources
  - install new links, faster routers
  - capacity planning, provisioning, traffic engineering
  - happen at longer timescale
- Reduce or delay demand
  - Reactive approach: encourage everyone to reduce or delay demand
  - Reservation approach: some requests will be rejected by the network