CS144
An Introduction to Computer Networks

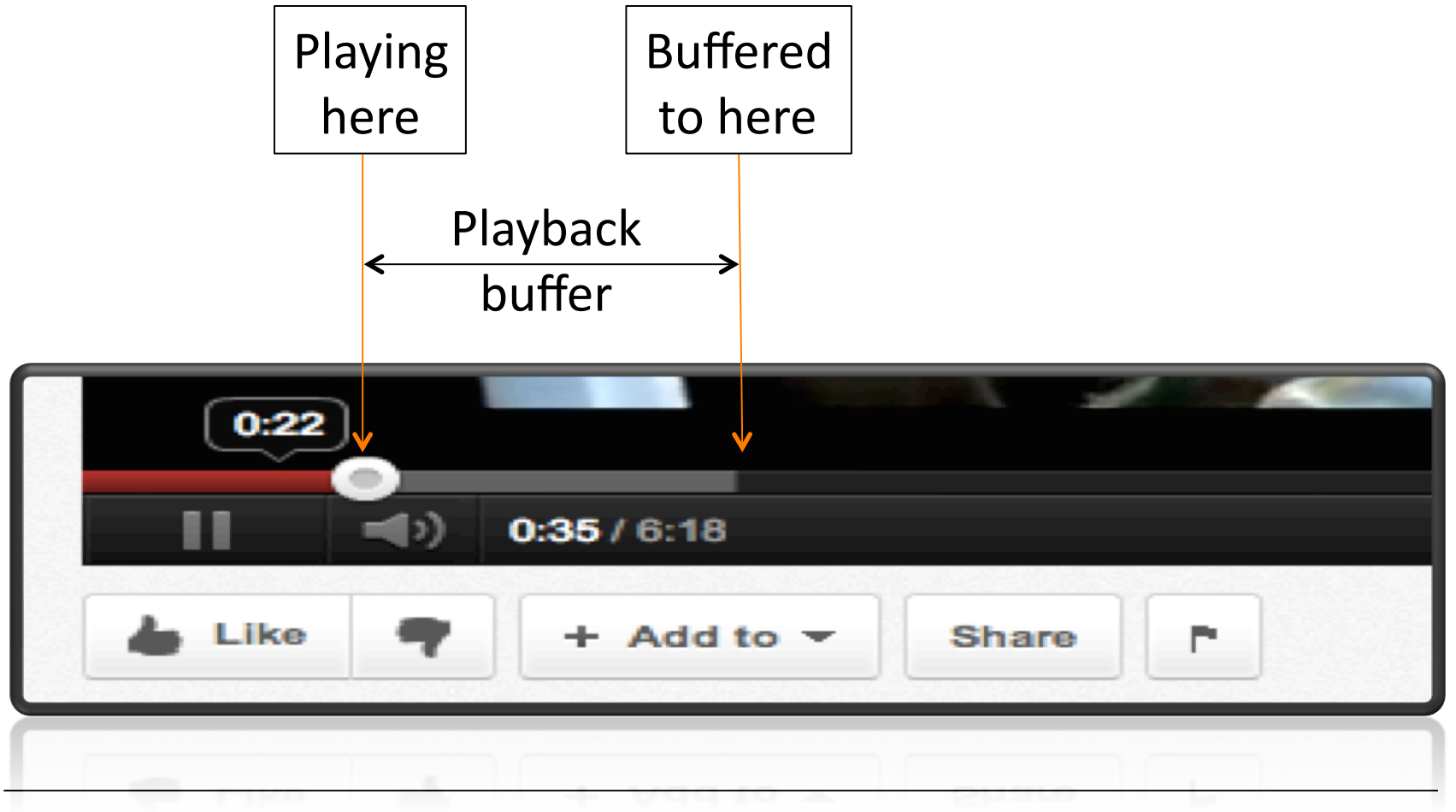
Packet Switching

Playback Buffers

Nick McKeown
Professor of Electrical Engineering
and Computer Science, Stanford University

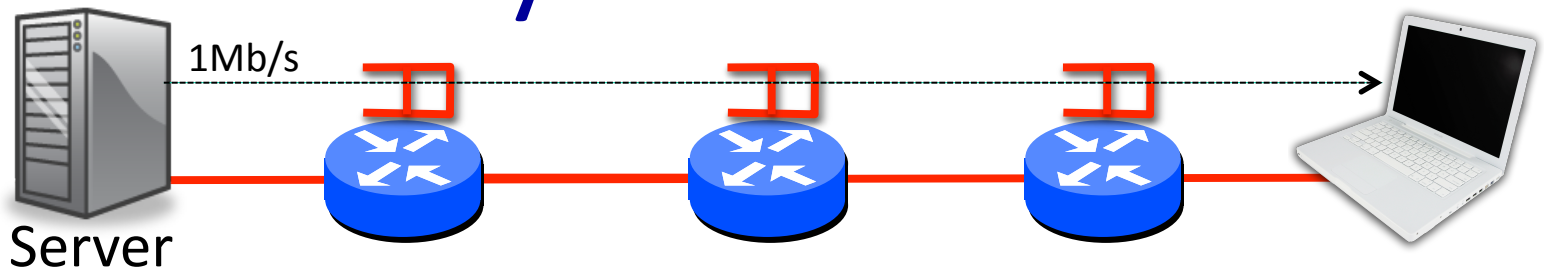
Real-time applications (e.g. YouTube and Skype) have to cope with variable queueing delay

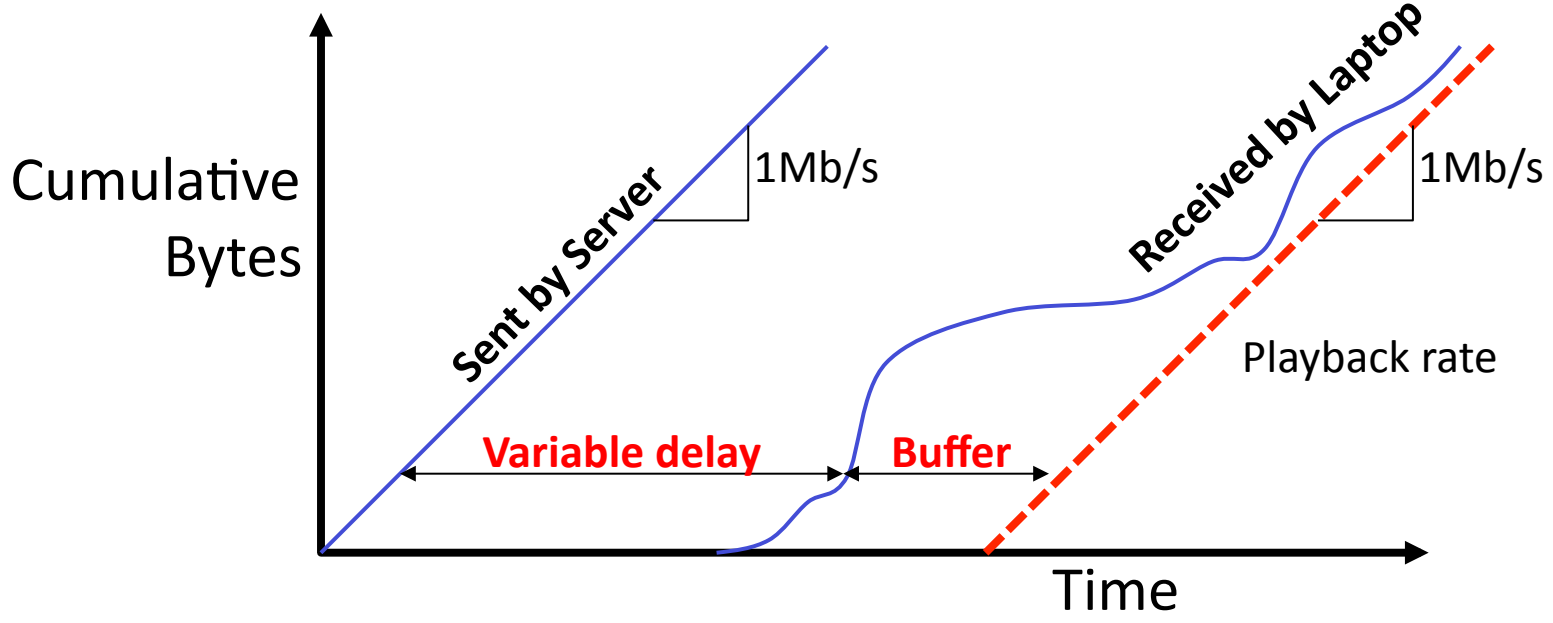
Playback buffers



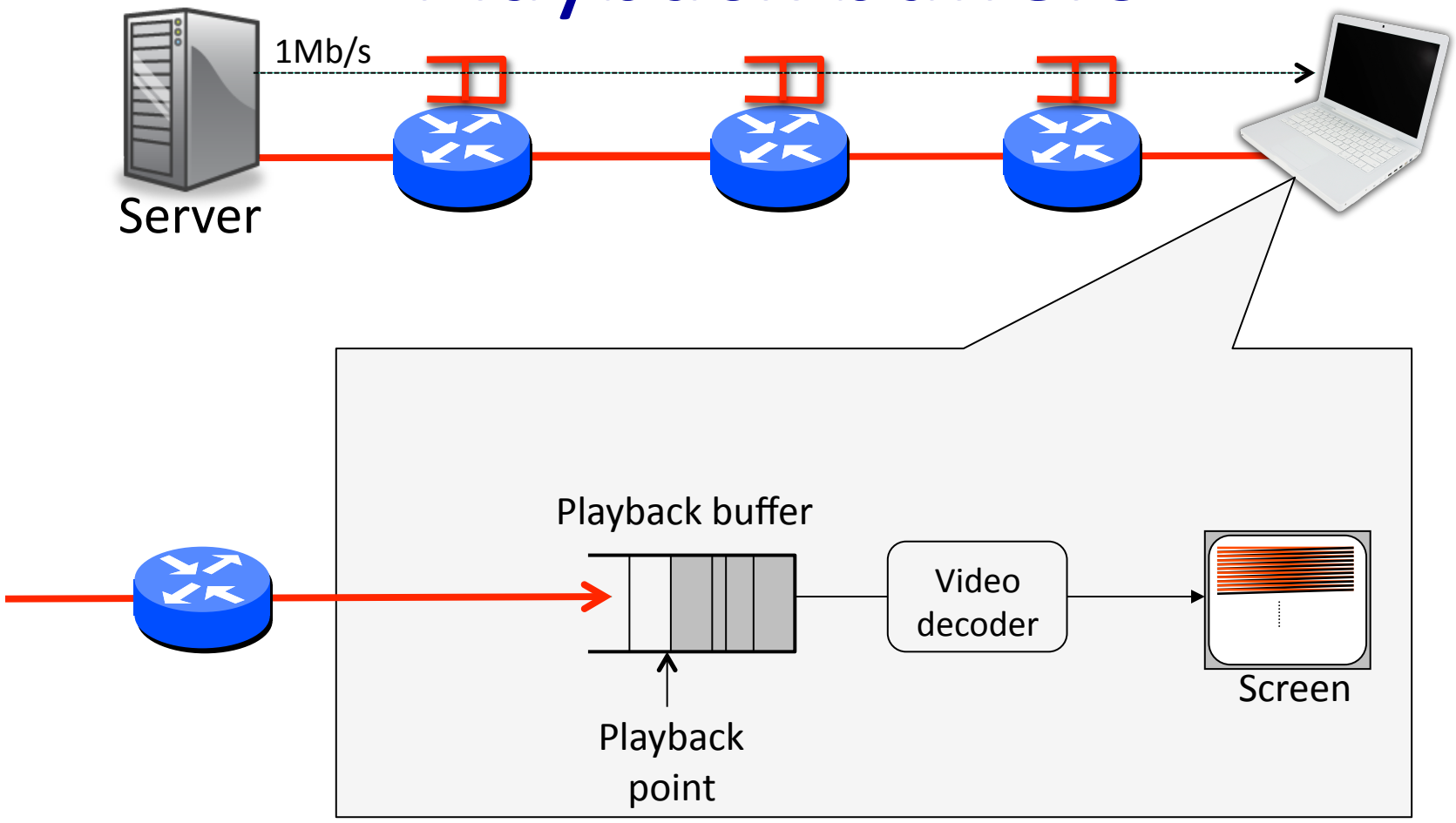
From: youtube.com

Playback buffers

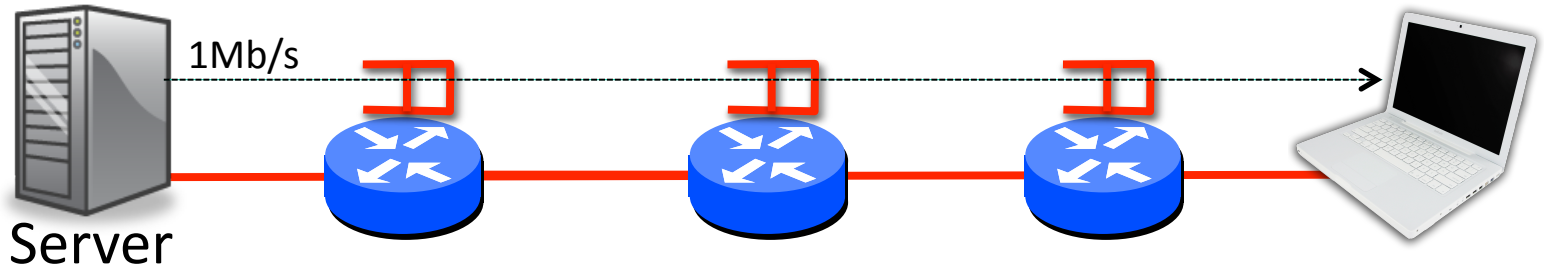


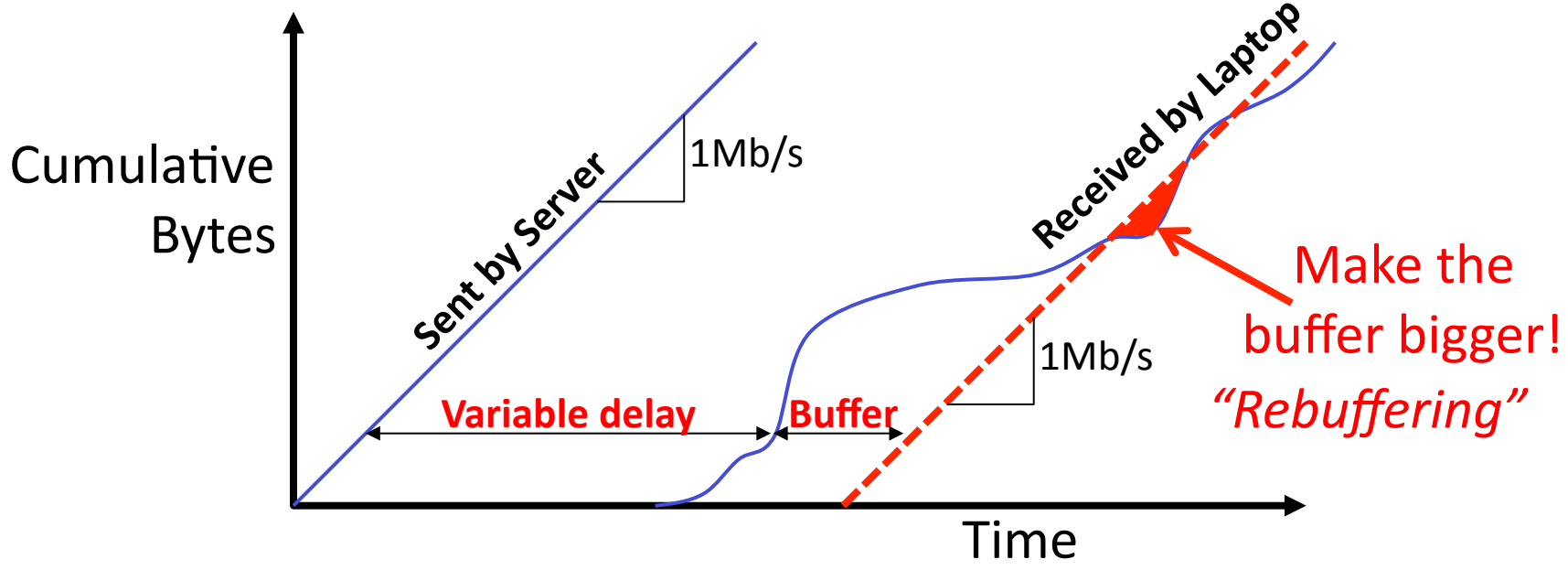


Playback buffers



If the buffer is too small

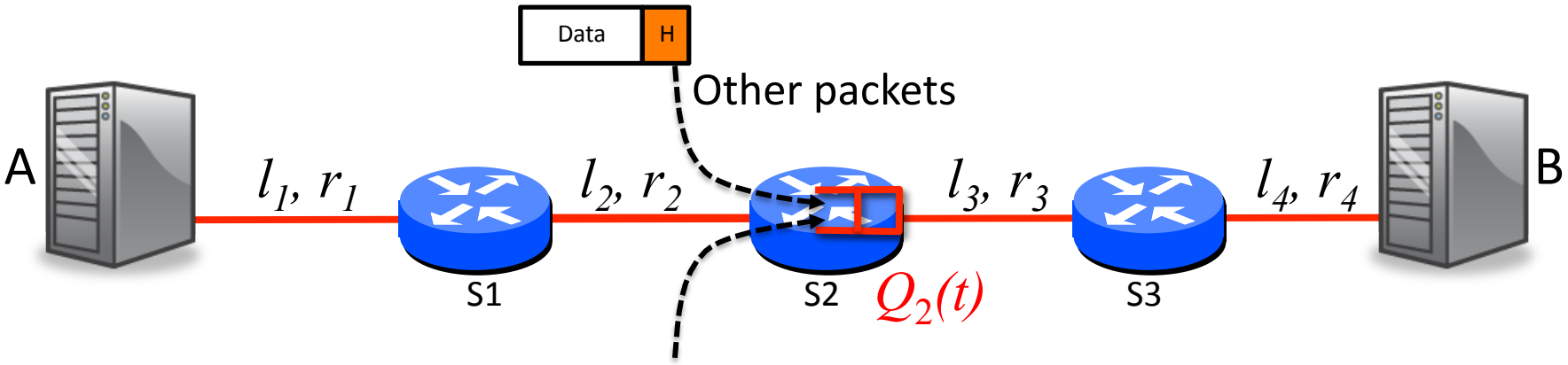


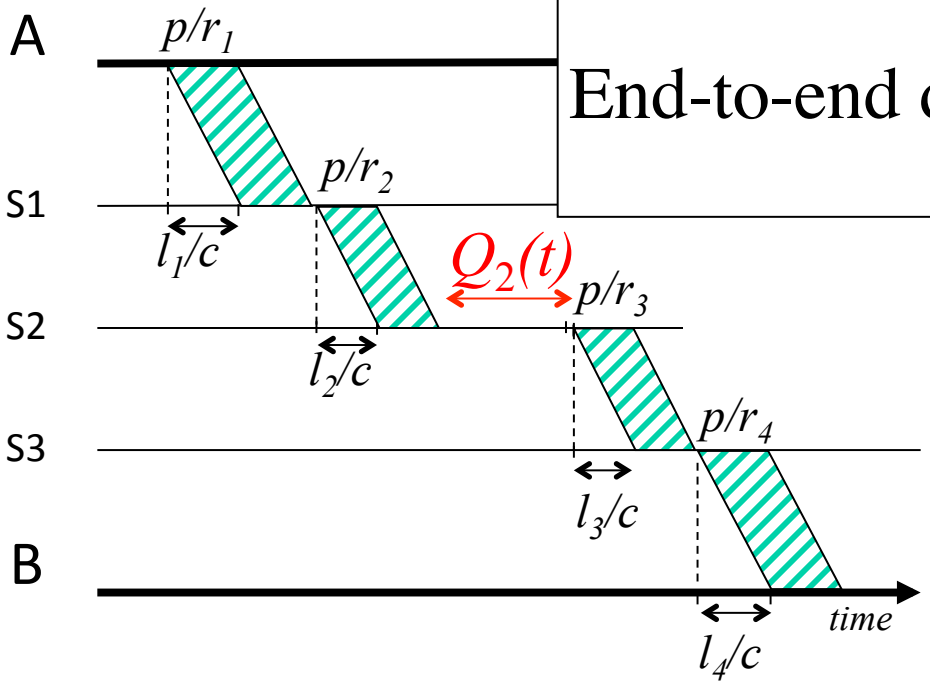


Playback buffer

- With packet switching, end-to-end delay is variable.
- We use a playback buffer to absorb the variation.

- We could just make the playback buffer very big, but then the video would be delayed at the start.
- Therefore, applications estimate the delay, set the playback buffer, and resize the buffer if the delay changes.





$$\text{End-to-end delay, } t = \sum_i \left(\frac{p}{r_i} + \frac{l_i}{c} + Q_i(t) \right)$$

Queueing adds variable and unpredictable delay

Summary

Real-time applications use playback buffers to absorb the variation in queueing delay.

<end>

<Got to here so far….>

Coming up:

1. Simple deterministic model of a queue
2. Some basic results about queues: Little's result, determinism minimizes delay
3. Simple model of queue delay (1/mu-lambda)
4. Leads into next topics:
    1. How to size a playback buffer. (include short demo)
    2. Retransmissions, flow-control, cong control and TCP (3-4 videos)
    3. How to size a router buffer

---

    4. Per-flow queues, fairness, fair queueing, b/w guarantees
    5. Delay guarantess (generalized processor sharing, Parekh/Gallager).

Queueing*

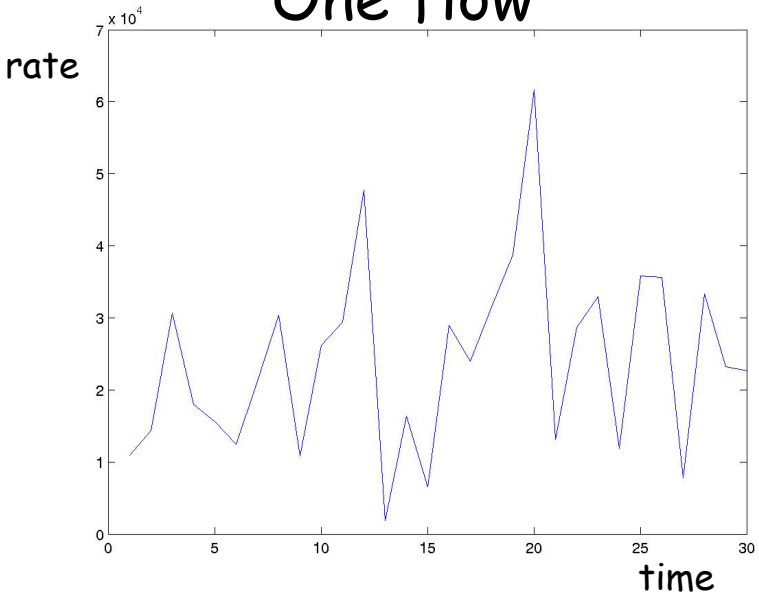
<

*Queueing = UK spelling, adopted by Kleinrock in 1960s.
Queueing and queuing (US spelling) are both widely used.

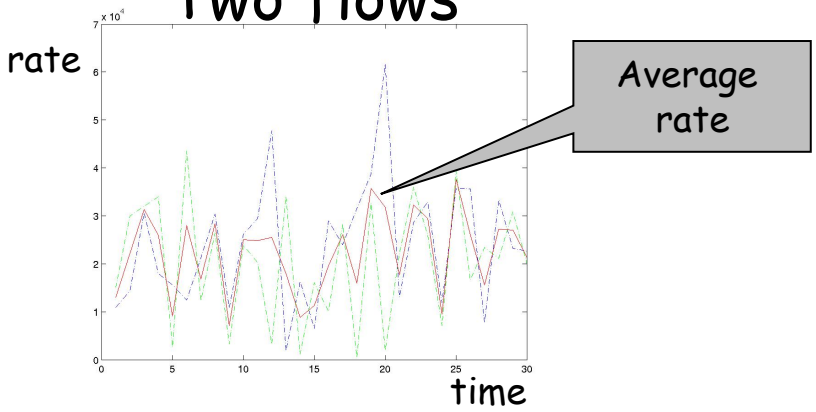
Statistical Multiplexing

Basic idea

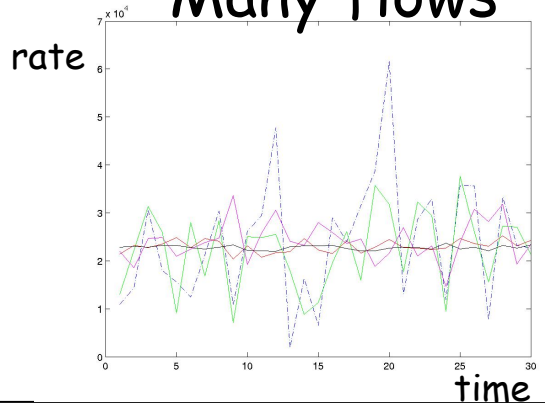
One flow



Two flows



Many flows

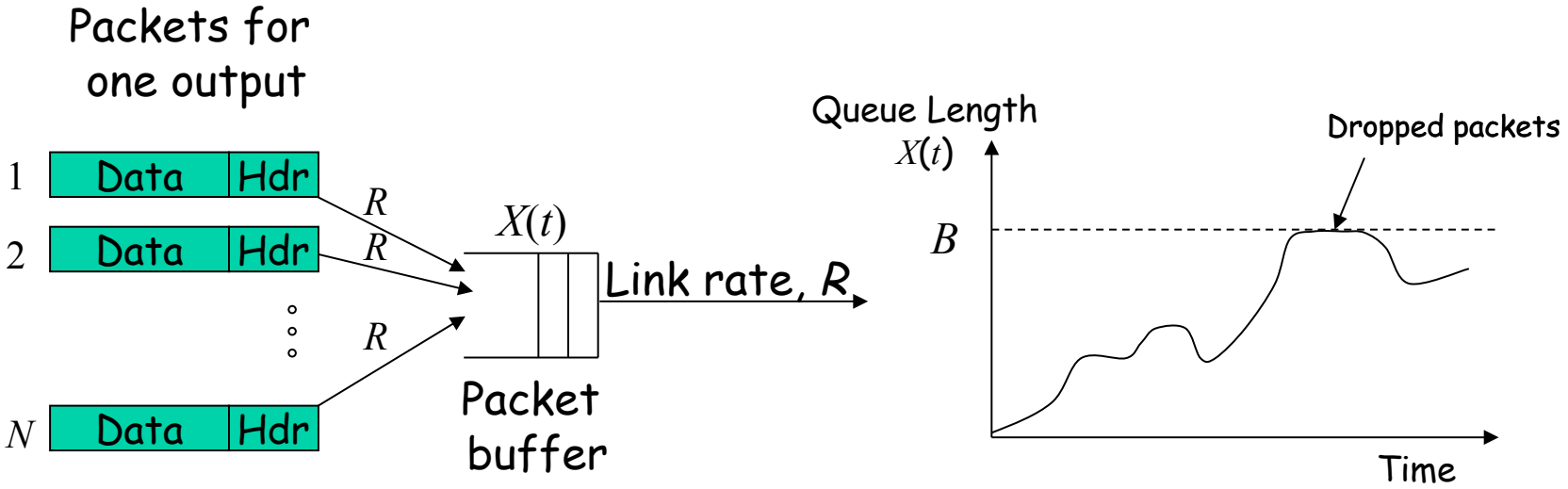


- Network traffic is bursty.
  i.e. the rate changes frequently.
- Peaks from independent flows generally occur at different times.
- Conclusion: The more flows we have, the smoother the traffic.

Average rates of: 1, 2, 10, 100, 1000 flows.

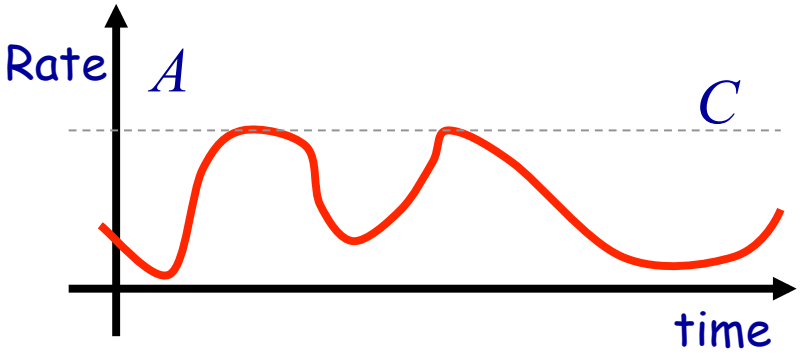
Packet Switching

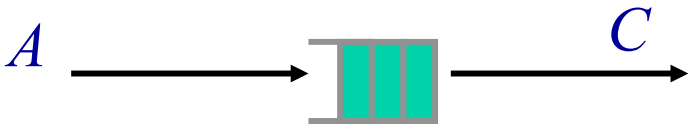
Statistical Multiplexing

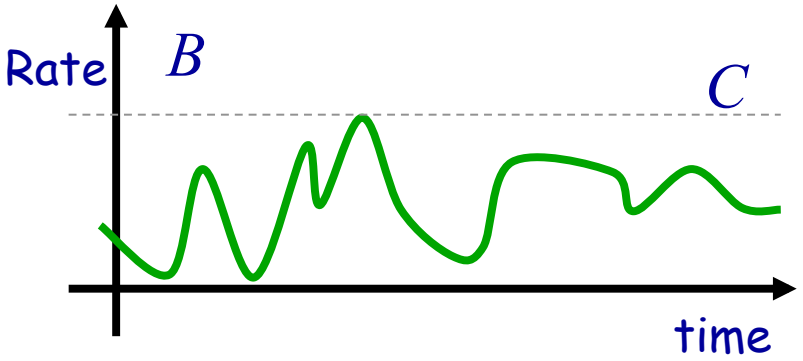


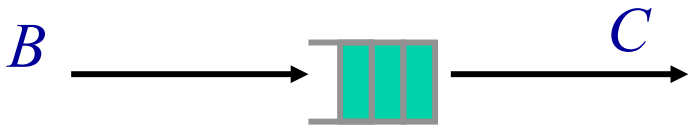
- Because the buffer absorbs temporary bursts, the egress link need not operate at rate $N.R$.
- But the buffer has finite size, B, so losses will occur.

Statistical Multiplexing

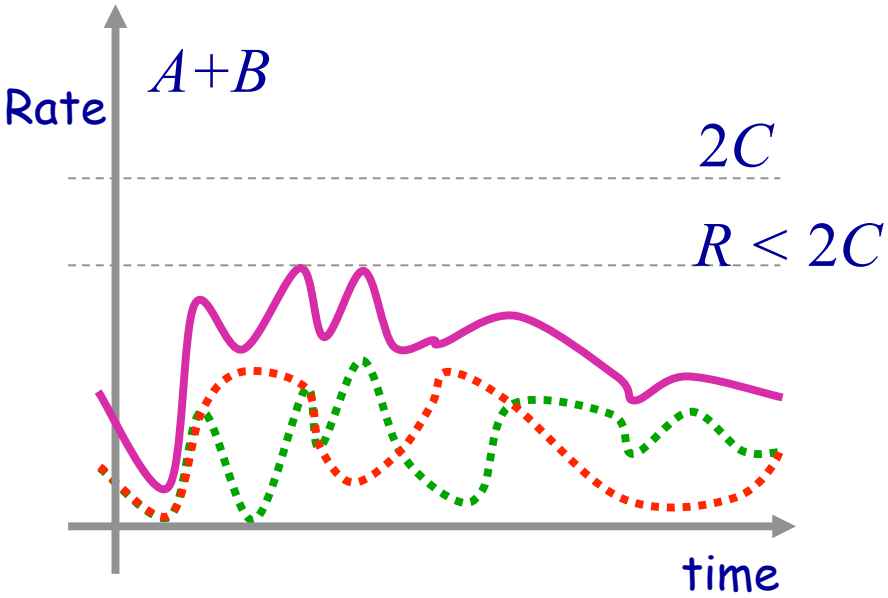


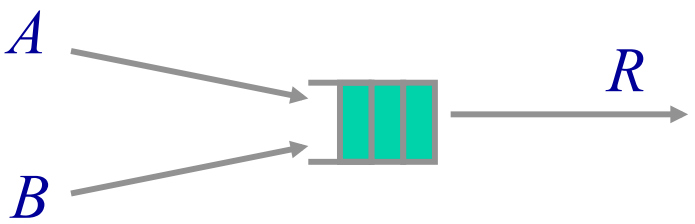






Statistical Multiplexing Gain



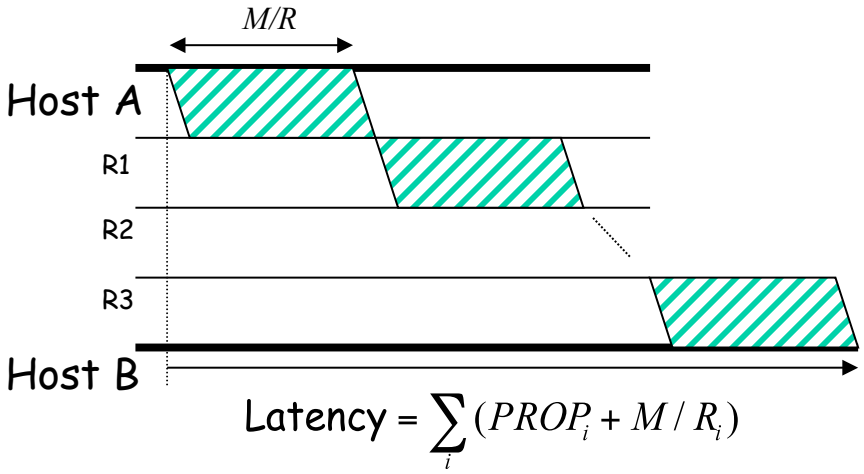


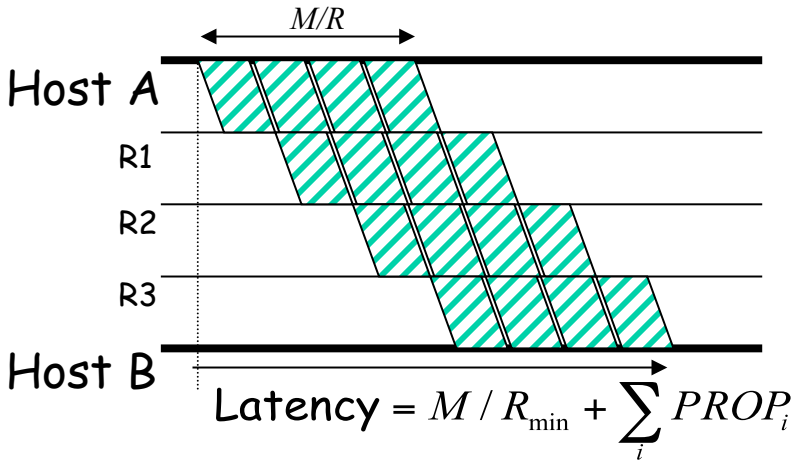
Statistical multiplexing gain = $2C/R$

Other definitions of SMG: The ratio of rates that give rise to a particular queue occupancy, or particular loss probability.

Packet Switching

Why not send the entire message in one packet?



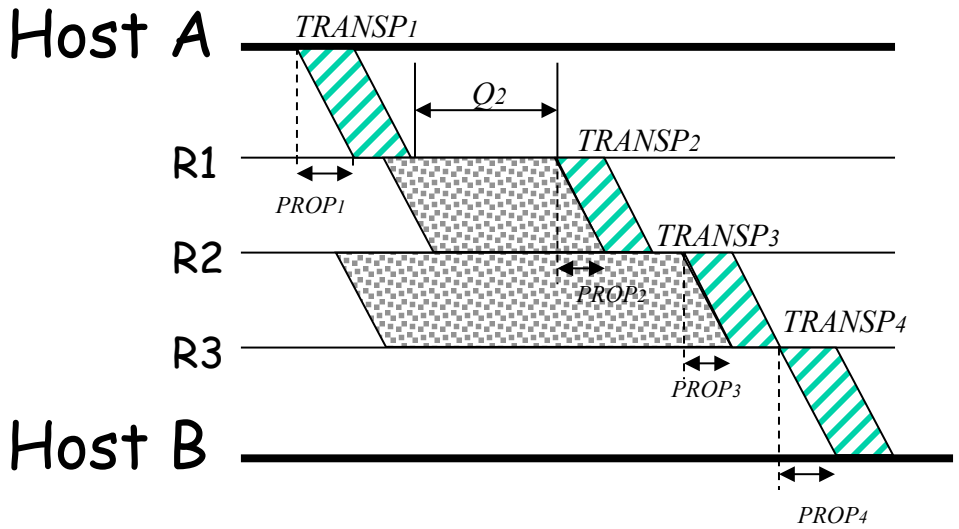


Breaking message into packets allows parallel transmission across all links, reducing end to end latency. It also prevents a link from being "hogged" for a long time by one message.

Packet Switching

Queueing Delay

Because the egress link is not necessarily free when a packet arrives, it may be queued in a buffer. If the network is busy, packets might have to wait a long time.



How can we determine the queueing delay?

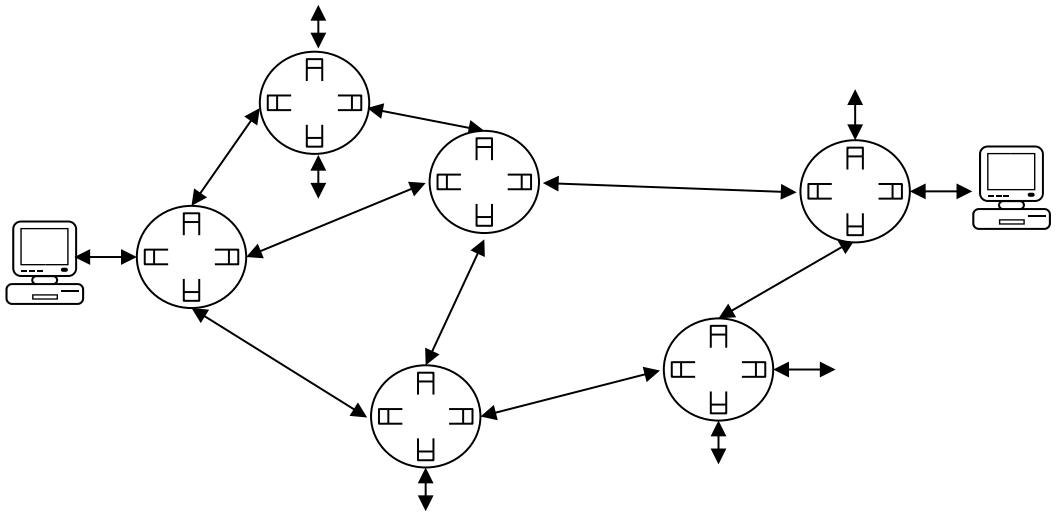
Actual end to end latency = $\sum_i (TRANSP_i + PROP_i + Q_i)$

Queues and Queueing Delay

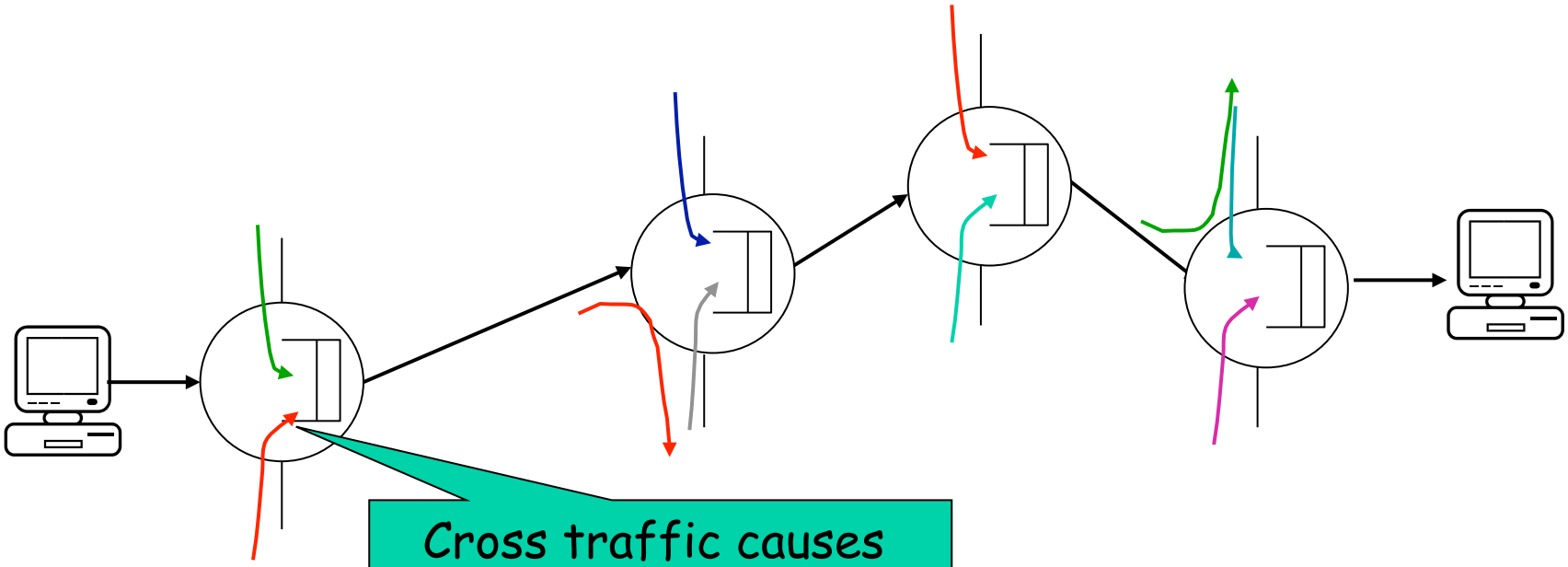
To understand the performance of a packet switched network, we can think of it as a series of queues interconnected by links.

For given link rates and lengths, the only variable is the queueing delay.

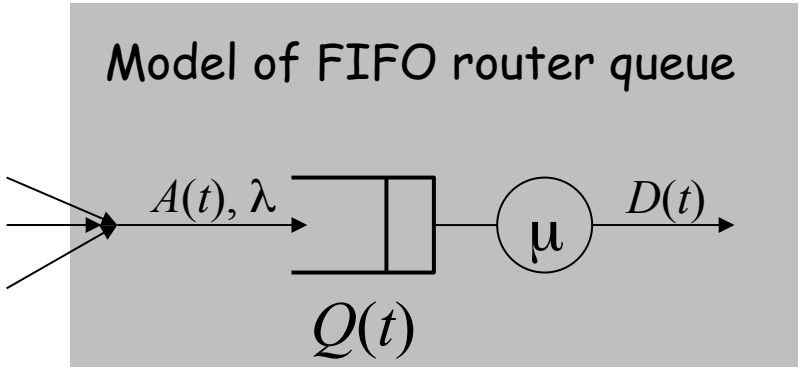
If we can understand the queueing delay, we can understand how the network performs.



Queues and Queueing Delay



A router queue



$A(t)$: The arrival process. The number of arrivals in interval $[0,t]$.

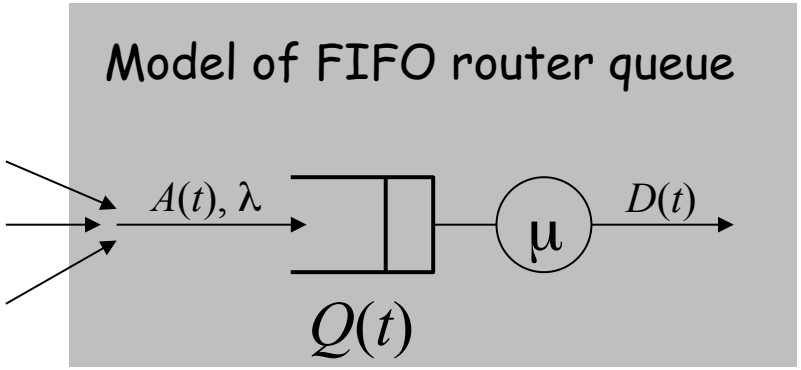
λ: The average rate of new arrivals in packets/second.

$D(t)$: The departure process. The number of departures in interval $[0,t]$.

$1/\mu$: The average time to service each packet.

$Q(t)$: The number of packets in the queue at time t.

A simple deterministic model



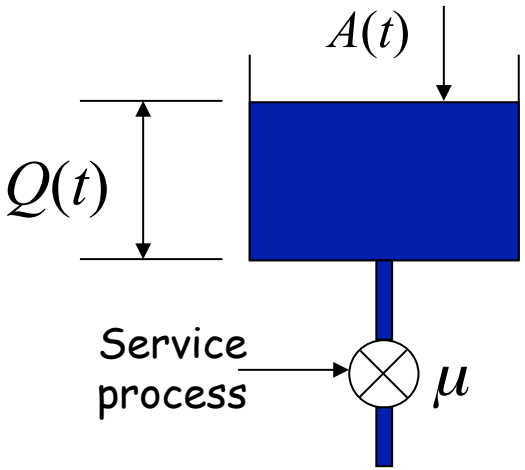
Properties of $A(t)$, $D(t)$:

- $A(t)$, $D(t)$ are non-decreasing
- $A(t) >= D(t)$

A simple deterministic model

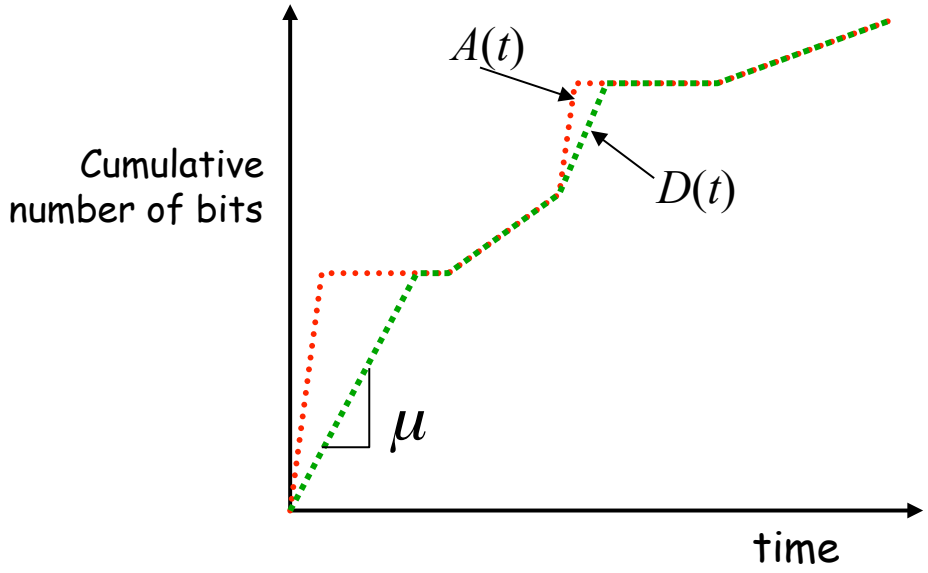
bytes or "fluid"

Cumulative number of bits that arrived up until time t.



$D(t)$

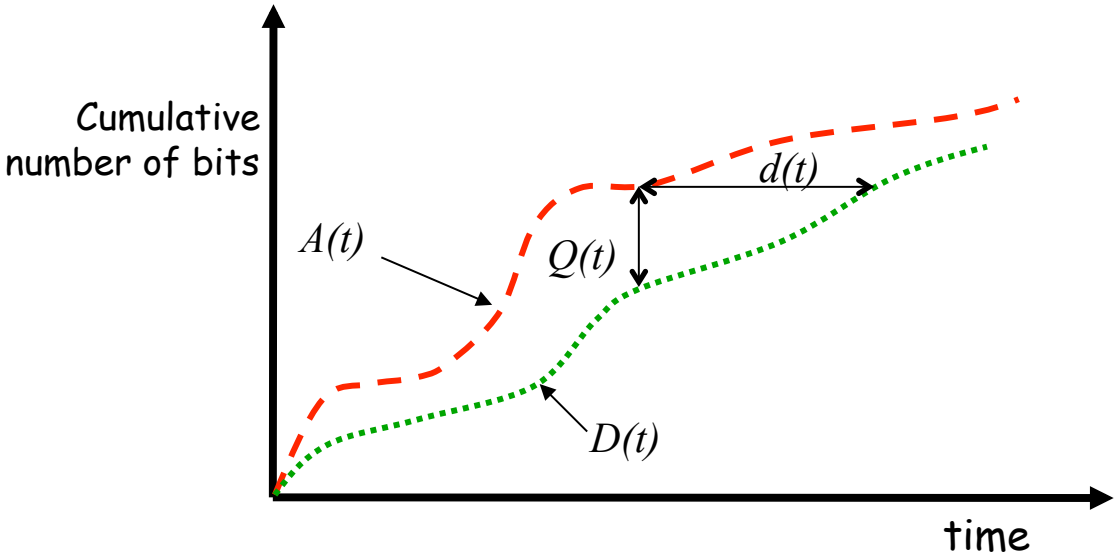
Cumulative number of departed bits up until time t.



Properties of $A(t)$, $D(t)$:

- $A(t)$, $D(t)$ are non-decreasing
- $A(t) >= D(t)$

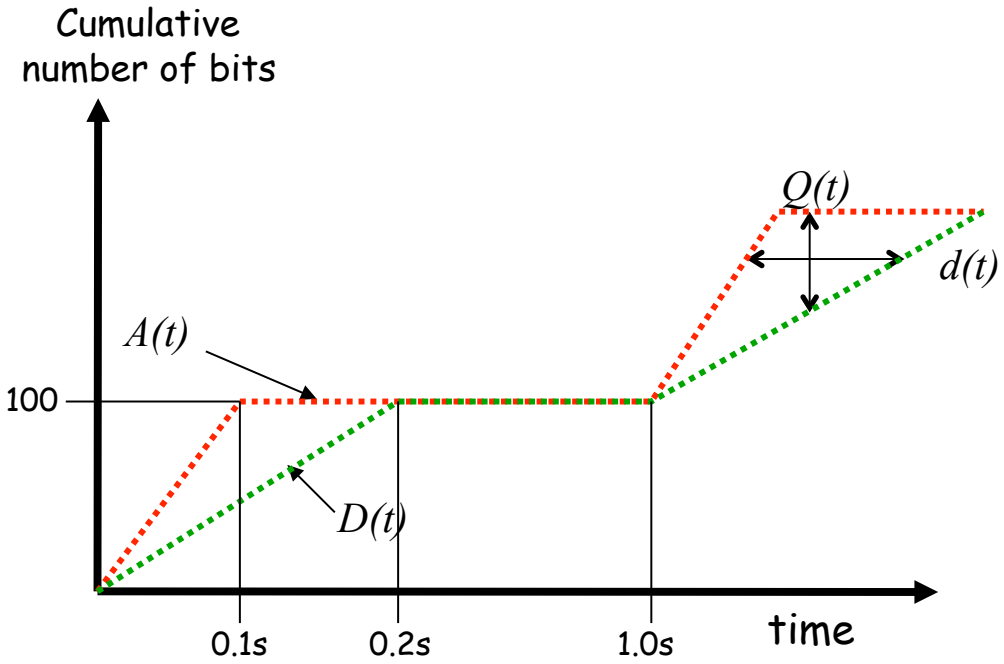
Simple deterministic model



Queue occupancy: $Q(t) = A(t) - D(t)$.

Queueing delay, $d(t)$, is the time spent in the queue by a bit that arrived at time t, and if the queue is served first-come-first-served (FCFS or FIFO)

Example



Example: Every second, a train of 100 bits arrive at rate 1000b/s. The maximum departure rate is 500b/s. What is the average queue occupancy?

Solution: During each cycle, the queue fills at rate 500b/s for 0.1s, then drains at rate 500b/s for 0.1s.The average queue occupancy when the queue is non-empty is therefore: $(\bar{Q}(t)|Q(t) > 0) = 0.5 \times (0.1 \times 500) = 25$ bits.

The queue is empty for 0.8s each cycle, and so: $\bar{Q}(t) = (0.2 \times 25) + (0.8 \times 0) = 5$ bits.

(You'll probably have to think about this for a while...).

Queues with Random Arrival Processes

1. Usually, arrival processes are complicated, so we often model them as random processes.
2. The study of queues with random arrival processes is called Queueing Theory.
3. Queues with random arrival processes have some interesting properties. We'll consider some here.

Properties of queues

Time evolution of queues.

Examples

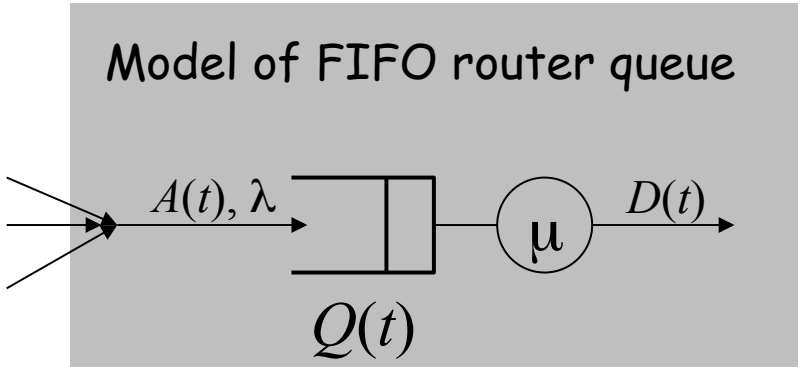
- Burstiness increases delay
- Determinism minimizes delay

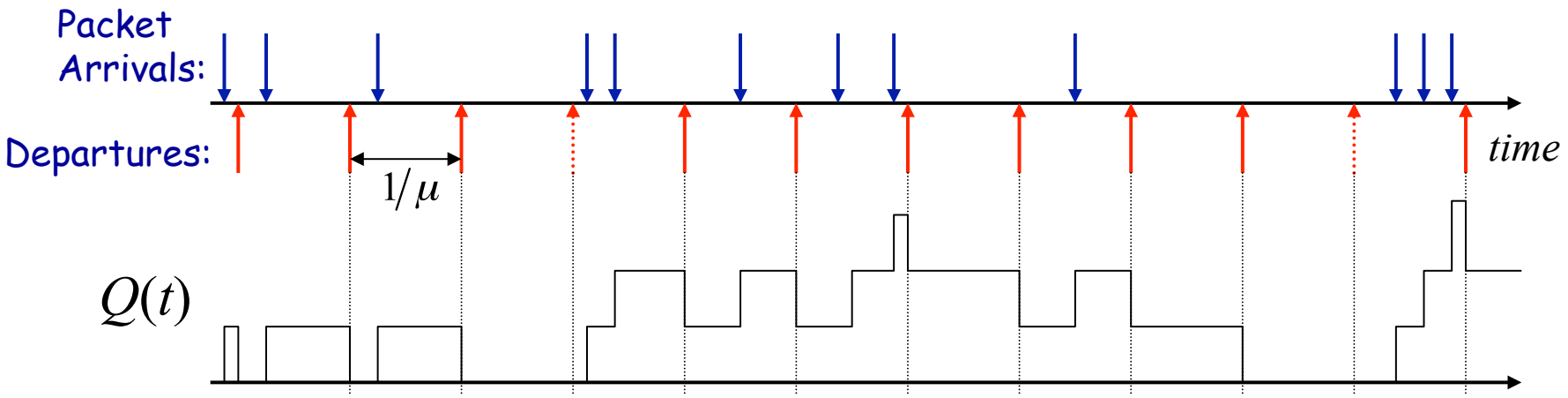
Little's Result.

The M/M/1 queue.

Time evolution of a queue

Packets





Burstiness increases delay

Example 1: Periodic arrivals

- 1 packet arrives every 1 second
- 1 packet can depart every 1 second
- Depending on when we sample the queue, it will contain 0 or 1 packets.

Example 2:

- N packets arrive together every N seconds (same rate)
- 1 packet departs every second
- Queue might contain 0,1, …, N packets.
- Both the average queue occupancy and the variance have increased.

In general, burstiness increases queue occupancy (which increases queueing delay).

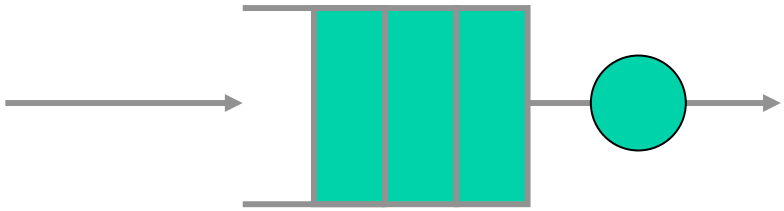
Determinism minimizes delay

Example 3: Random arrivals

- Packets arrive randomly; on average, 1 packet arrives per second.
- Exactly 1 packet can depart every 1 second.
- Depending on when we sample the queue, it will contain 0, 1, 2, … packets depending on the distribution of the arrivals.

In general, determinism minimizes delay.
i.e. random arrival processes lead to larger delay than simple periodic arrival processes.

Little's Result



$$L = \lambda d$$

Where:

L is the average number of customers in the system (the number in the queue + the number in service),

λ is the arrival rate, in customers per second, and

d is the average time that a customer waits in the system (time in queue + time in service).

Result holds so long as no customers are lost/dropped.

The Poisson process

Poisson process is a simple arrival process in which:

1. Probability of k arrivals in an interval of t seconds is:

$$P_k(t) = \frac{(\lambda t)^k}{k!} e^{-\lambda t}$$

2. The expected number of arrivals in interval t is: λt.
3. Successive interarrival times are independent of each other (i.e. arrivals are not bursty).

The Poisson process

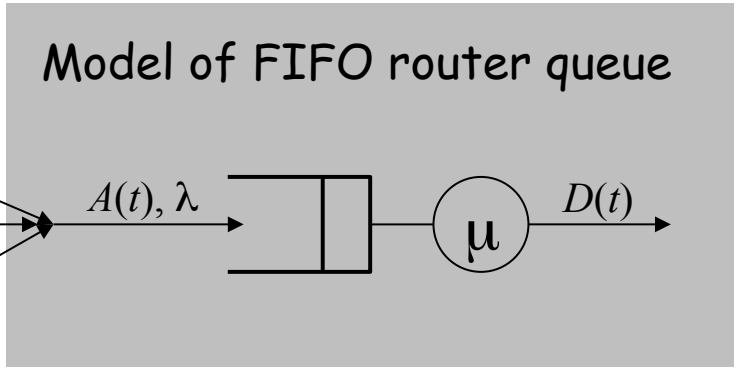
Why use the Poisson process?

- It is the continuous time equivalent of a series of coin tosses.
- The Poisson process is known to model well systems in which a large number of independent events are aggregated together. e.g.
  - Arrival of new phone calls to a telephone switch
  - Decay of nuclear particles
  - "Shot noise" in an electrical circuit
- It makes the math easy.

Be warned

- Network traffic is very bursty!
- Packet arrivals are not Poisson.
- But it models quite well the arrival of new flows.

An M/M/1 queue



If $A(t)$ is a Poisson process with rate λ, and the time to serve each packet is exponentially distributed with rate μ, then:

Average delay, $d = \frac{1}{\mu - \lambda}$; and so from Little's Result: $L = \lambda d = \frac{\lambda}{\mu - \lambda}$