Queueing theory for TCP

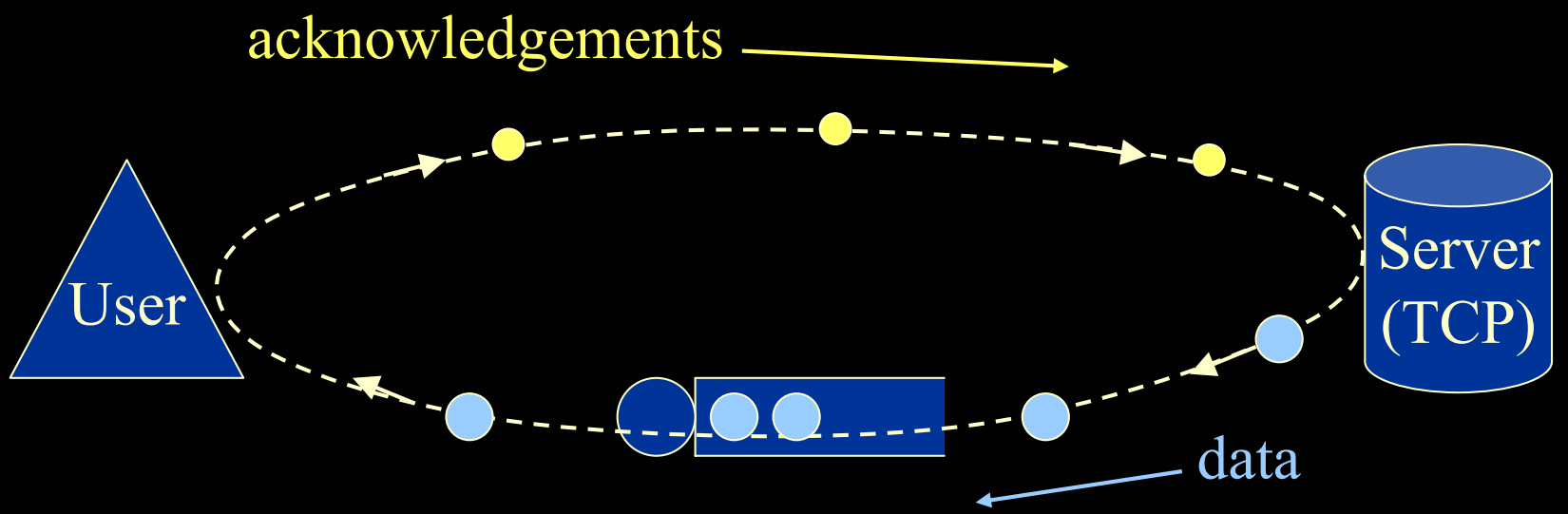
Damon Wischik, UCL
Gaurav Raina, Cambridge

Some Internet history

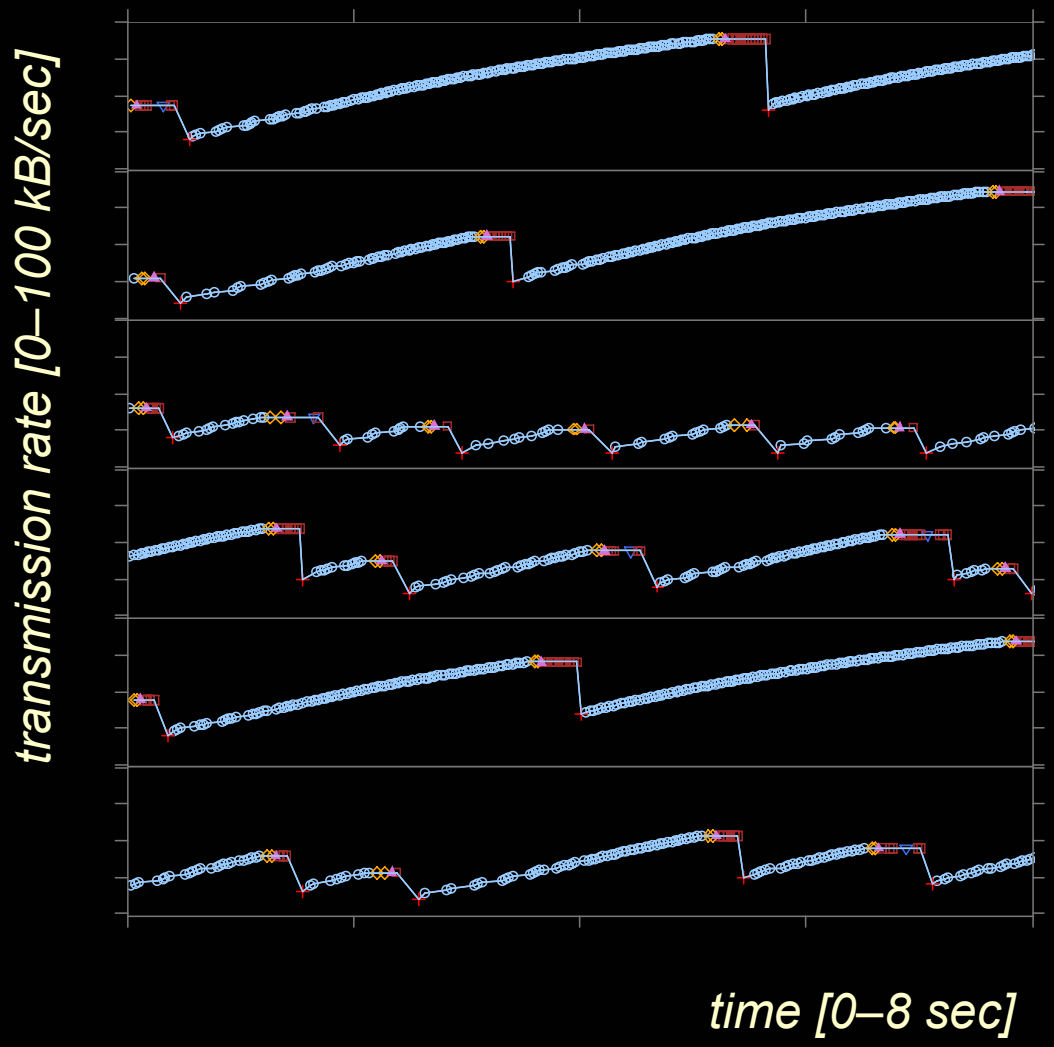
- 1974: First draft of TCP/IP
  [*"A protocol for packet network interconnection"*, *Vint Cerf and Robert Kahn*]
- 1983: ARPANET switches on TCP/IP
- 1986: Congestion collapse
- 1988: Congestion control for TCP
  [*"Congestion avoidance and control"*, *Van Jacobson*]

TCP transmission control protocol

- Internet congestion is controlled by the end-systems
- The TCP algorithm, running on the server, decides how fast to send data
  - It sends data packets, and waits for ACKnowledgement packets
  - It sets a target window size w_t
    the number of packets that have been sent but not yet acknowledged (here 5+3)
  - The average transmission rate is $x_t \approx w_t/RTT$ where RTT is the round trip time
  - It adapts w_t based on the ACKs it receives

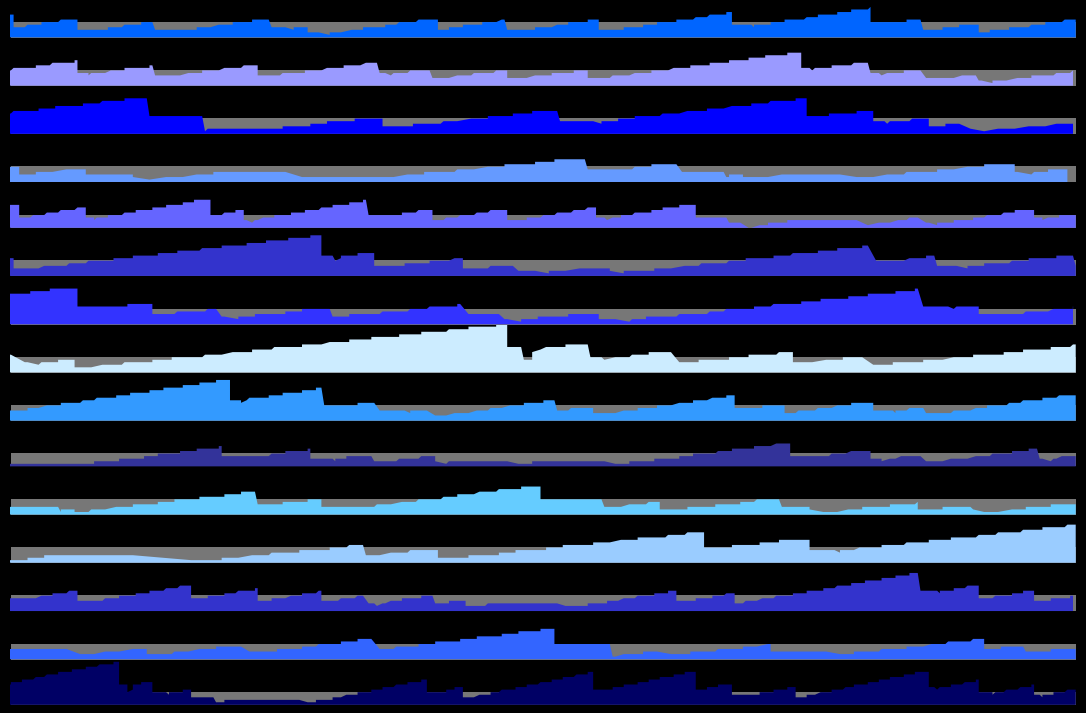


TCP

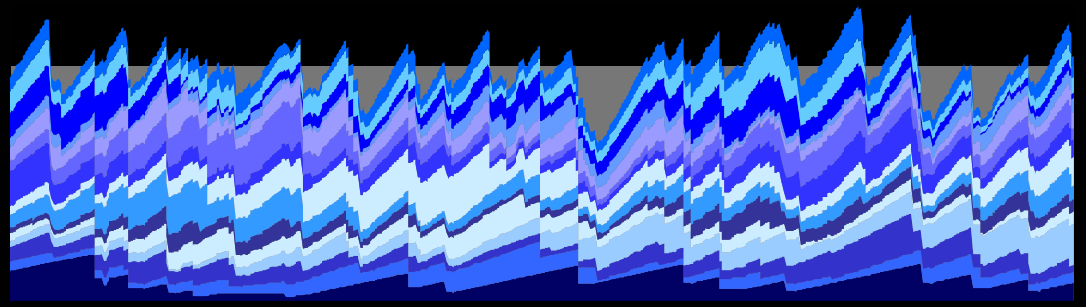



```
if (seqno > _last_acked) {
    if (!_in_fast_recovery) {
        _last_acked = seqno;
        _dupacks = 0;
        inflate_window();
        send_packets(now);
        _last_sent_time = now;
        return;
        }
    if (seqno < _recover) {
        uint32_t new_data = seqno - _last_acked;
        _last_acked = seqno;
        if (new_data < _cwnd) _cwnd -= new_data; else _cwnd=0;
        _cwnd += _mss;
        retransmit_packet(now);
        send_packets(now);
        return;
        }
    uint32_t flightsize = _highest_sent - seqno;
    _cwnd = min(_ssthresh, flightsize + _mss);
    _last_acked = seqno;
    _dupacks = 0;
    _in_fast_recovery = false;
    send_packets(now);
    return;
    }
if (_in_fast_recovery) {
    _cwnd += _mss;
    send_packets(now);
    return;
    }
_dupacks++;
if (_dupacks!=3) {
    send_packets(now);
    return;
    }
_ssthresh = max(_cwnd/2, (uint32_t)(2 * _mss));
retransmit_packet(now);
_cwnd = _ssthresh + 3 * _mss;
_in_fast_recovery = true;
_recover = _highest_sent;
}
```

How TCP shares capacity



individual flow rates



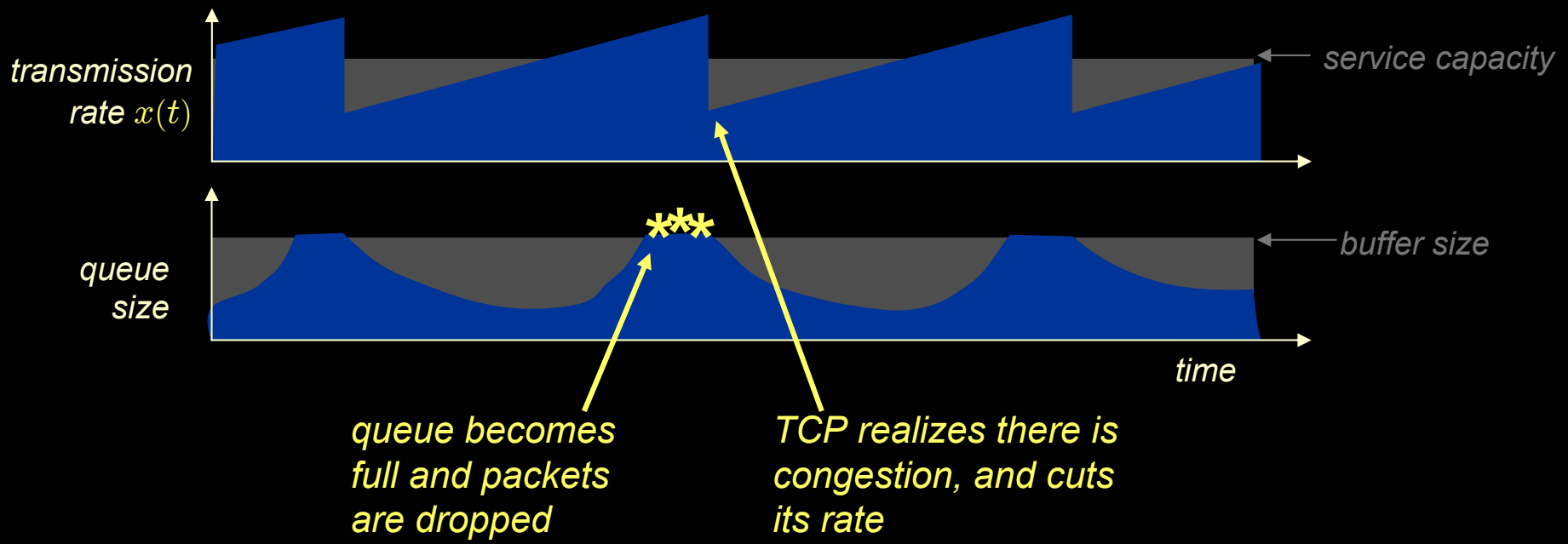
aggregate flow rate

available capacity

time

TCP model I: the sawtooth

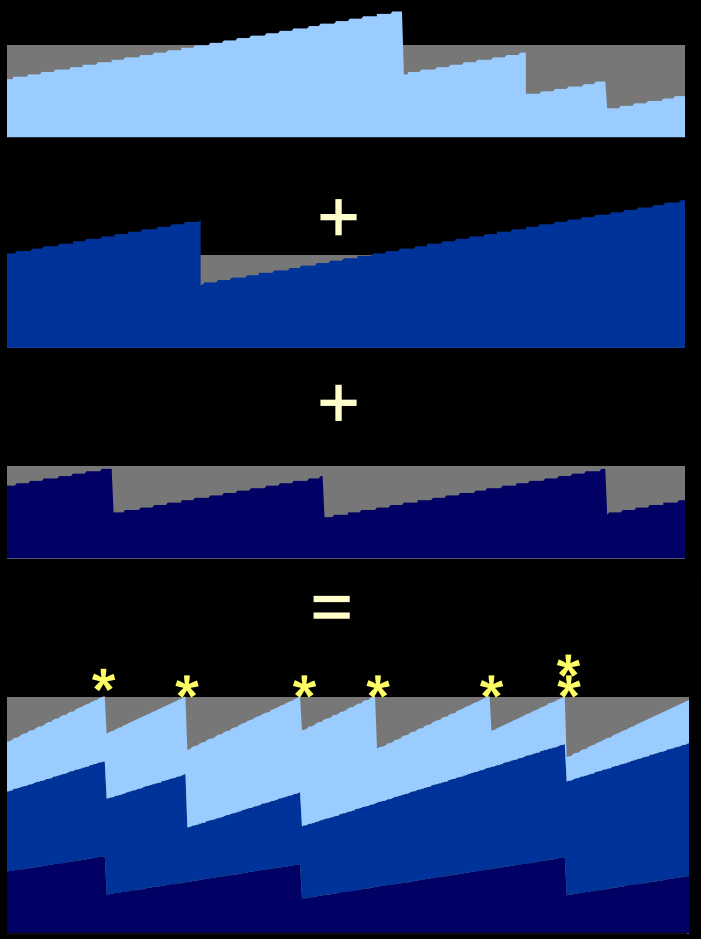
- Consider a single link, and a single flow with round trip time RTT
- When the link is not congested, transmission rate x_t increases by $1/RTT^2$ pkt/sec every sec
- When the link becomes congested, transmission rate is cut by 50%



TCP model II: billiards

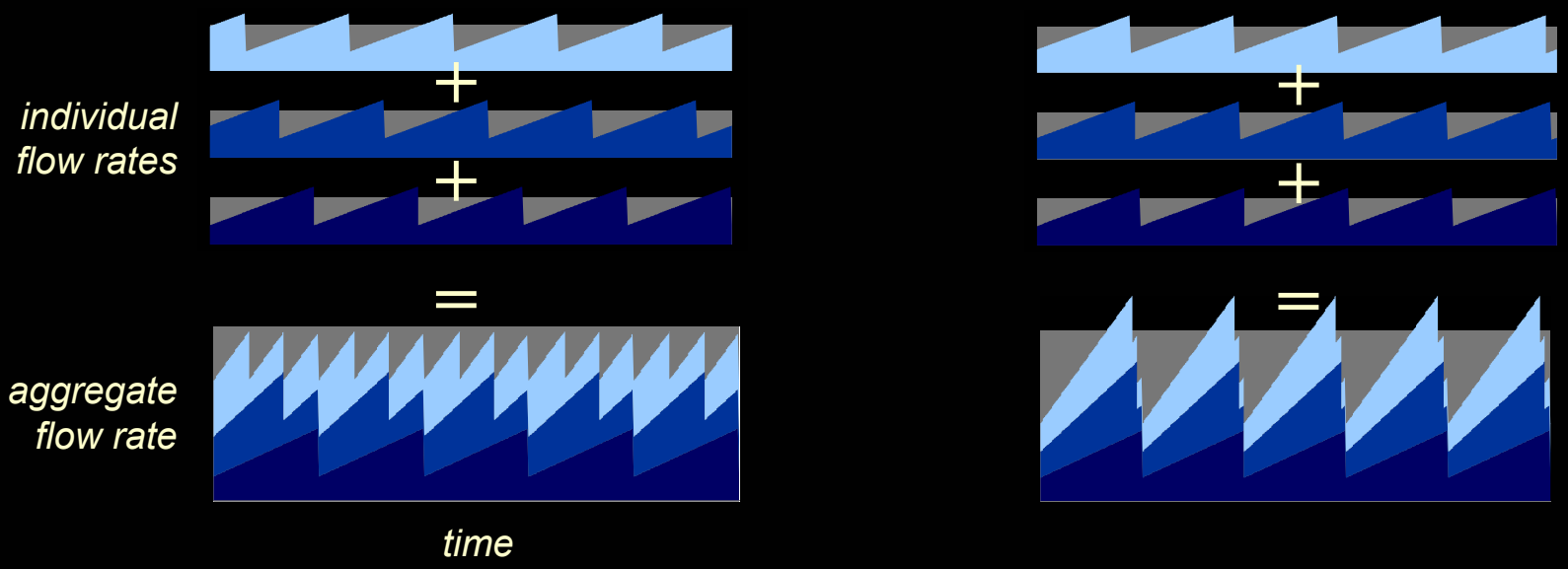
- Consider a single link, and many flows with round trip times $RTT(1)$, $RTT(2)$,...
- When the link is not saturated, flow i increases its transmission rate $x_t(i)$ by $1/RTT(i)$ pkt/sec^2
- When the link becomes saturated, some flows experience packet drops and they cut their rates
- Assume some probabilistic model that says which flows experience drops

[Baccelli+Hong 2002]



TCP model III: fluid

- Individual TCP flows follow a sawtooth

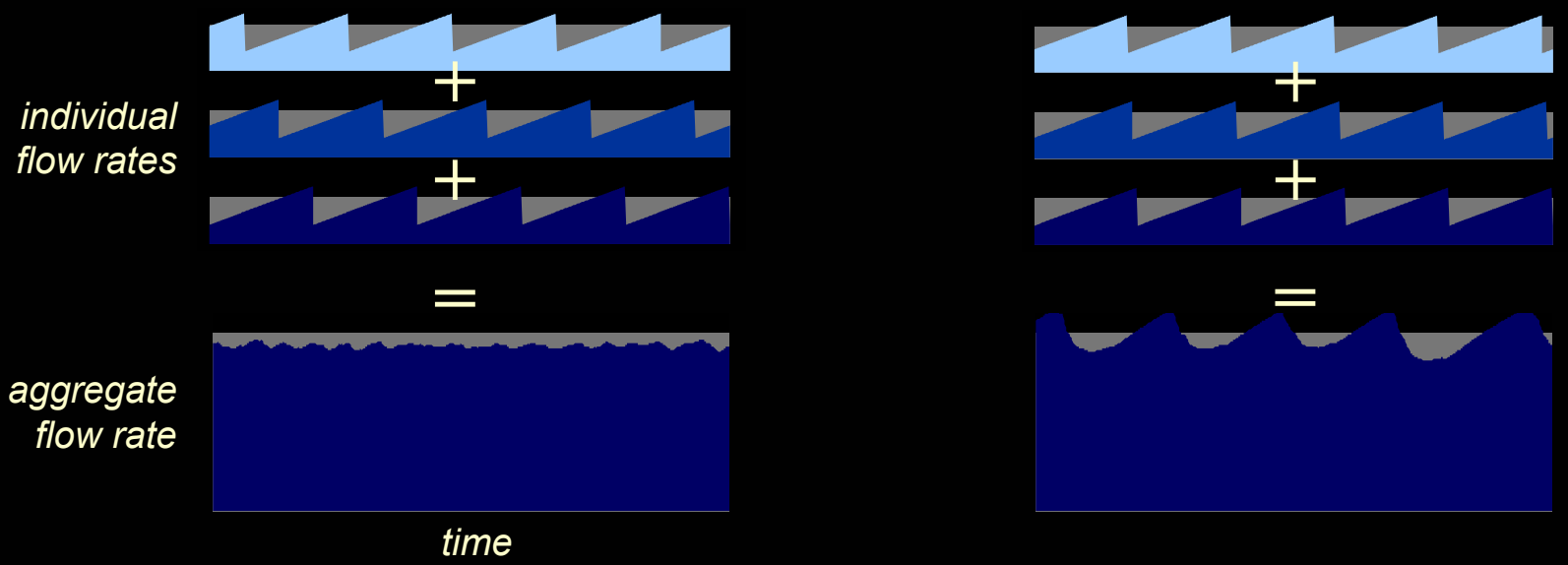


TCP model III: fluid

- Individual TCP flows follow a sawtooth
- ... but many TCP flows added together are smoother
  - the average rate x_t varies according to a time-delayed differential equation
  - eqn involves p_t, the packet drop probability experienced by packets sent at time t

  [Misra+Gong+Towsley 2000, Baccelli+McDonald+Reynier 2002, after Kelly et al. 1998]

$$\frac{dx_t}{dt} = \frac{1}{RTT^2} - p_{t-RTT}\, x_{t-RTT}\, \frac{x_t}{2}$$

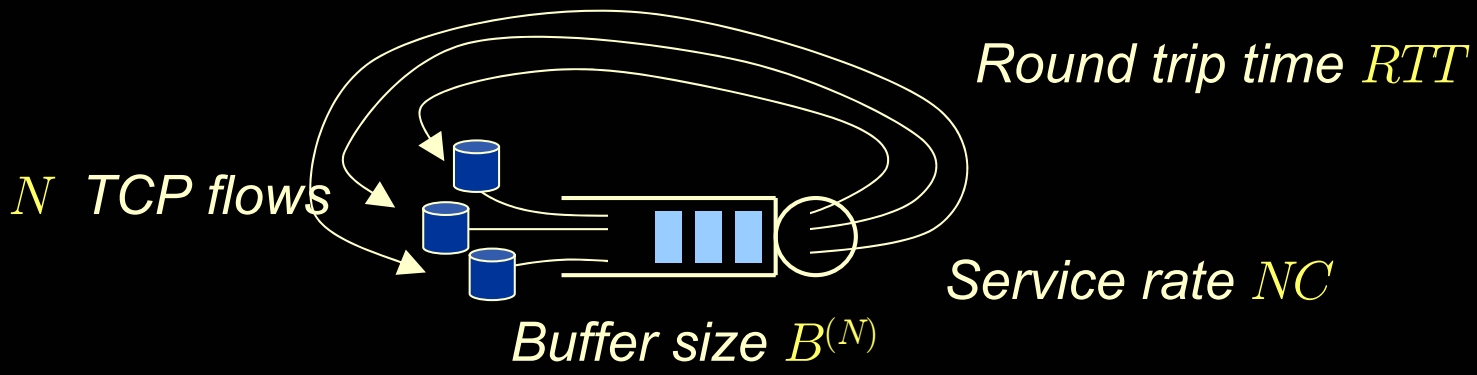


Questions about queues

- What is the packet drop probability p?
  - from a critically loaded or overloaded M/M/1/B model [B+H]
  - from an underloaded M/M/1/B model [Kelly, Key, ...]
  - $p_t = (x_t - C)^+ / x_t$ i.e. the loss rate caused by excess traffic [Srikant]
  - assume that the router sets p explicitly [M+G+T, ...]
- How does it depend on buffer size & traffic statistics?
  - assorted other approximations from queueing theory [Kelly, ...]

- How big should buffers be in Internet routers?
  - 3 GByte? Rule of thumb says buffer = bandwidth×delay
  - 300 MByte? buffer = bandwidth×delay/√#flows [Appenzeller+Keslassy+McKeown, 2004]
  - 30 kByte? constant buffer size, independent of line rate [Kelly, Key, ...]

Formulate a maths question

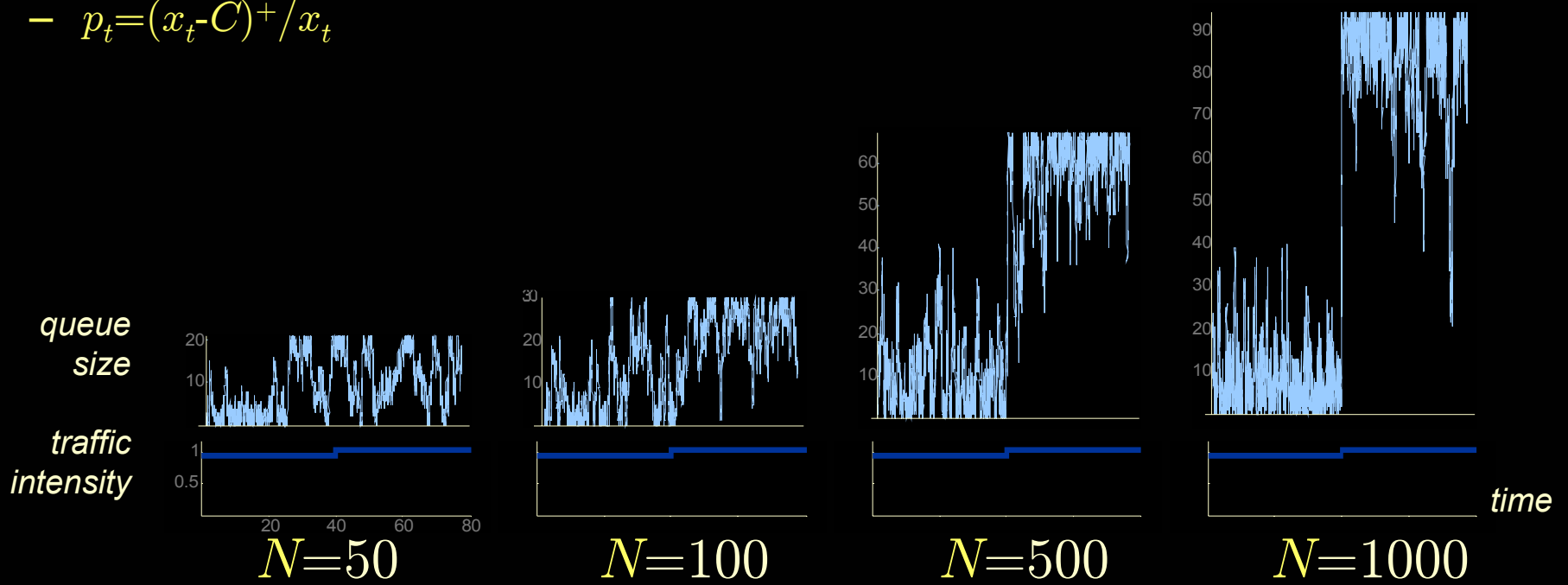


- What is the limiting queue-length process, as $N \to \infty$, in the three regimes
  - *large buffers* $B^{(N)} = BN$
  - *intermediate buffers* $B^{(N)} = B\sqrt{N}$
  - *small buffers* $B^{(N)} = B$

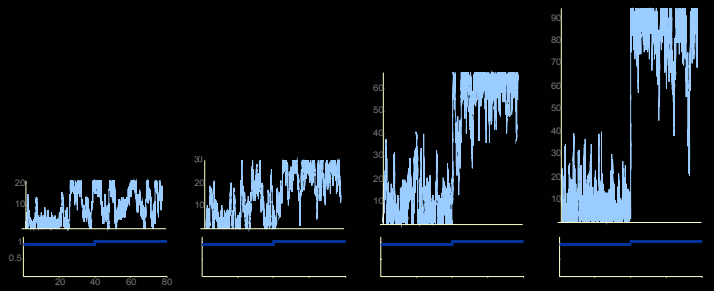
- Why are these the interesting limiting regimes? What are the alternatives? [Bain, 2003]

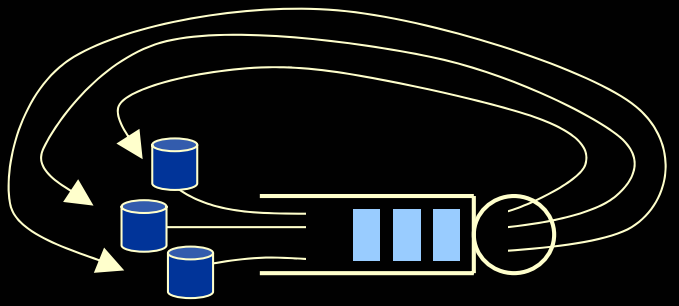
Intermediate buffers $B^{(N)}=B\sqrt{N}$

- Consider a standalone queue
  - fed by N Poisson flows, each of rate x pkts/sec where x varies slowly, served at rate NC, with buffer size $B\sqrt{N}$
  - x_t=0.95 then 1.05 pkts/sec, C=1 pkt/sec, B=3 pkt
- Observe
  - queue size flips quickly from near-empty to near-full
  - queue size fluctuates very rapidly; busy periods have duration $O(1/N)$ so p_t is a function of instantaneous arrival rate x_t
  - $p_t=(x_t\text{-}C)^+/x_t$



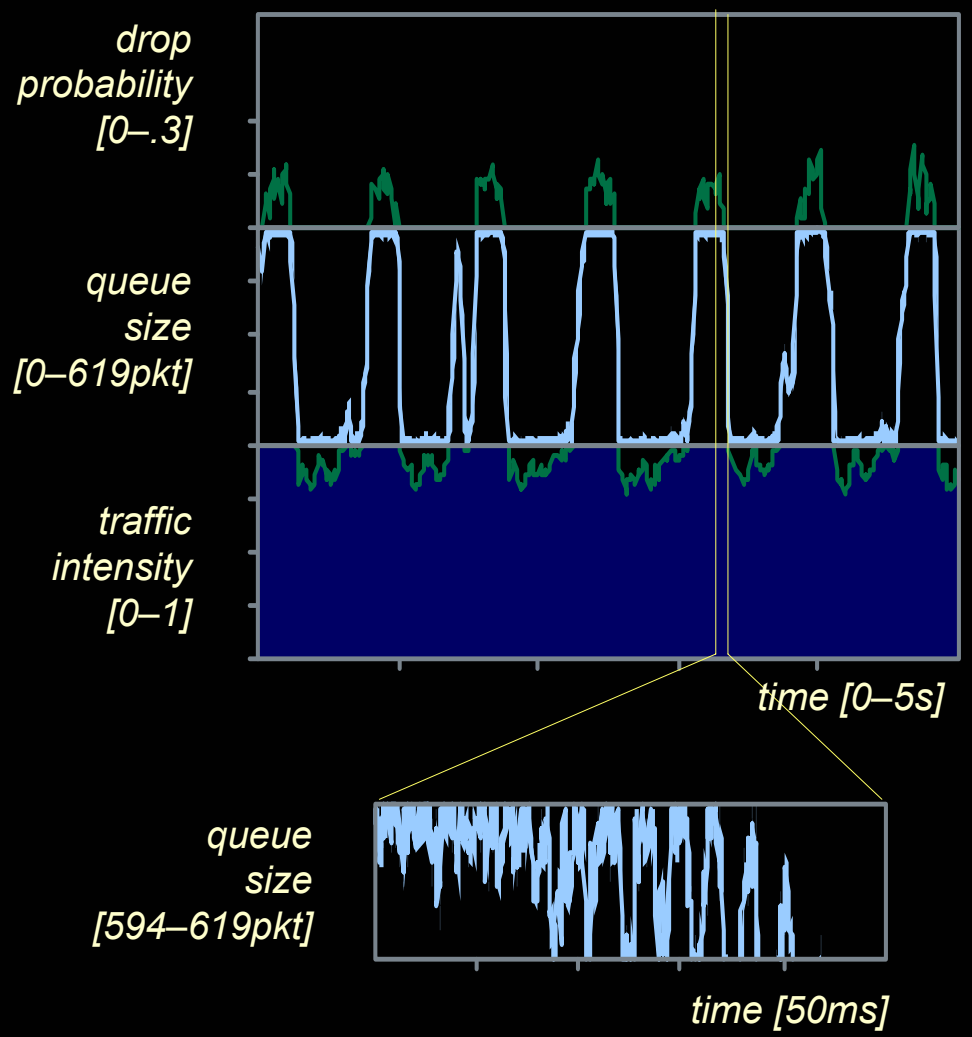
Closing the loop





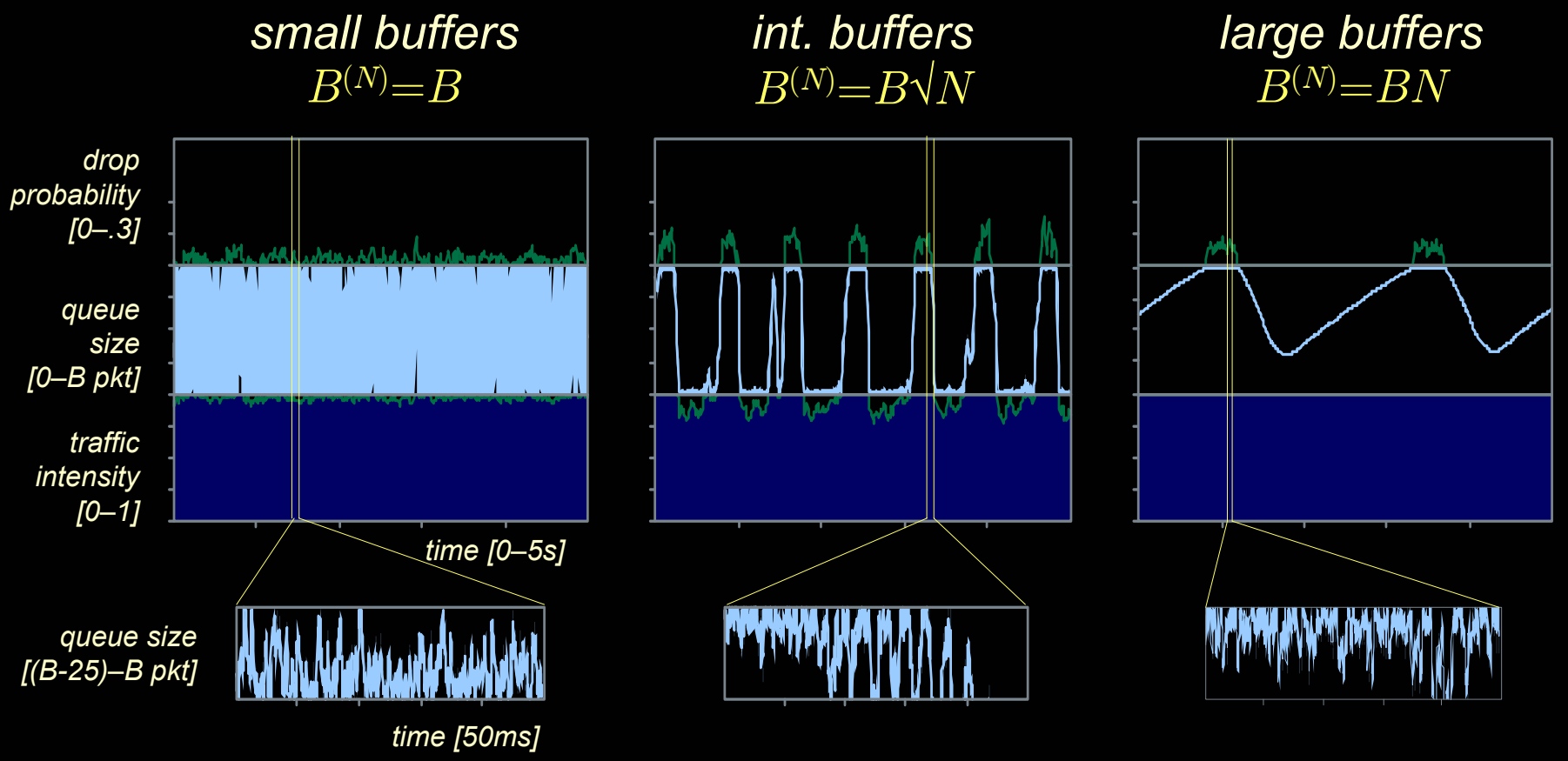
- In the open-loop queue
  - we assumed Poisson inputs with slowly varying arrival rate
  - queue size flips quickly from near-empty to near-full
  - queue size fluctuates very rapidly, with busy periods of duration $O(1/N)$, so drop probability p_t is a function of instantaneous arrival rate x_t
- In the closed-loop TCP system
  - the round trip time means that we can treat inputs over time $[t, t+RTT]$ as an exogenous arrival process to the queue
  - the average arrival rate should vary slowly, according to a differential equation
  - the aggregate of many point processes converges to a Poisson process, and this carries through to queue size [Cao+Ramanan 2002]
- There is thus a separation of timescales
  - queue size fluctuates over short timescales, and we obtain p_t from the open-loop M/D/1/$B^{(N)}$ model
  - TCP adapts its rate over long timescales, according to the fluid model differential equation

Illustration



- 2Gb/s link, C=50 pkt/sec/flow
- 5000 TCP flows
- RTT 150–200ms
- buffer 619pkt
- simulator `htsim` by Mark Handley, UCL

Other buffer-sizing regimes



- The difference in timescales between small and large buffer AQM schemes was first observed by [Deb+Srikant, 2004]

System summary

- x_t = average traffic rate at time t
  p_t = packet loss probability at time t
  C = capacity/flow B = buffer size RTT = round trip time N=# flows

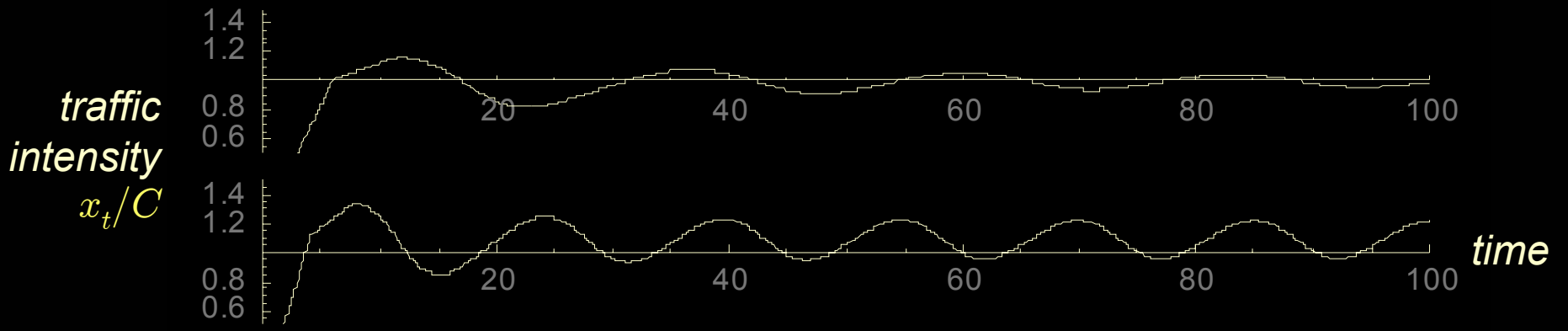
- TCP traffic model
$$\frac{dx_t}{dt} = \frac{1}{RTT^2} - p_{t-RTT}x_{t-RTT}\frac{x_t}{2}$$

- Small buffers
$$p_t \approx (x_t/C)^B$$

- Intermediate buffers
$$p_t = (x_t - C)^+/x_t$$

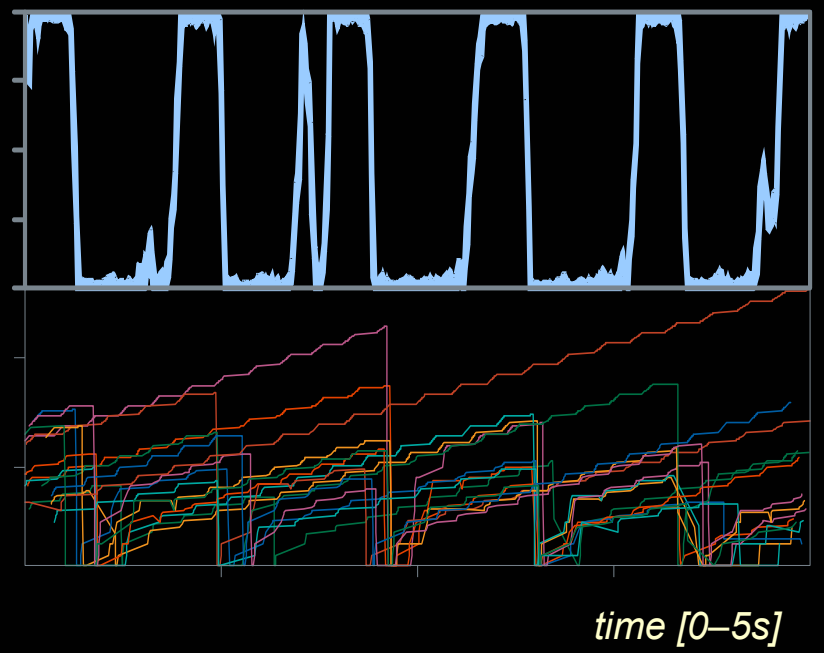
- Large buffers
$$p_t = 1_{\{q_t=B\}}(x_t - C)/x_t$$
$$\frac{dq_t}{dt} = x_t(1 - p_t) - C$$

Stability/instability analysis



- For some parameter values the dynamical system is stable
  - we can calculate the steady-state traffic intensity, loss probability etc.
- For others it is unstable and there are oscillations
  - we can calculate the amplitude of the oscillations, either by numerical integration or by algebraic approximation [Gaurav Raina, PhD thesis, 2005]

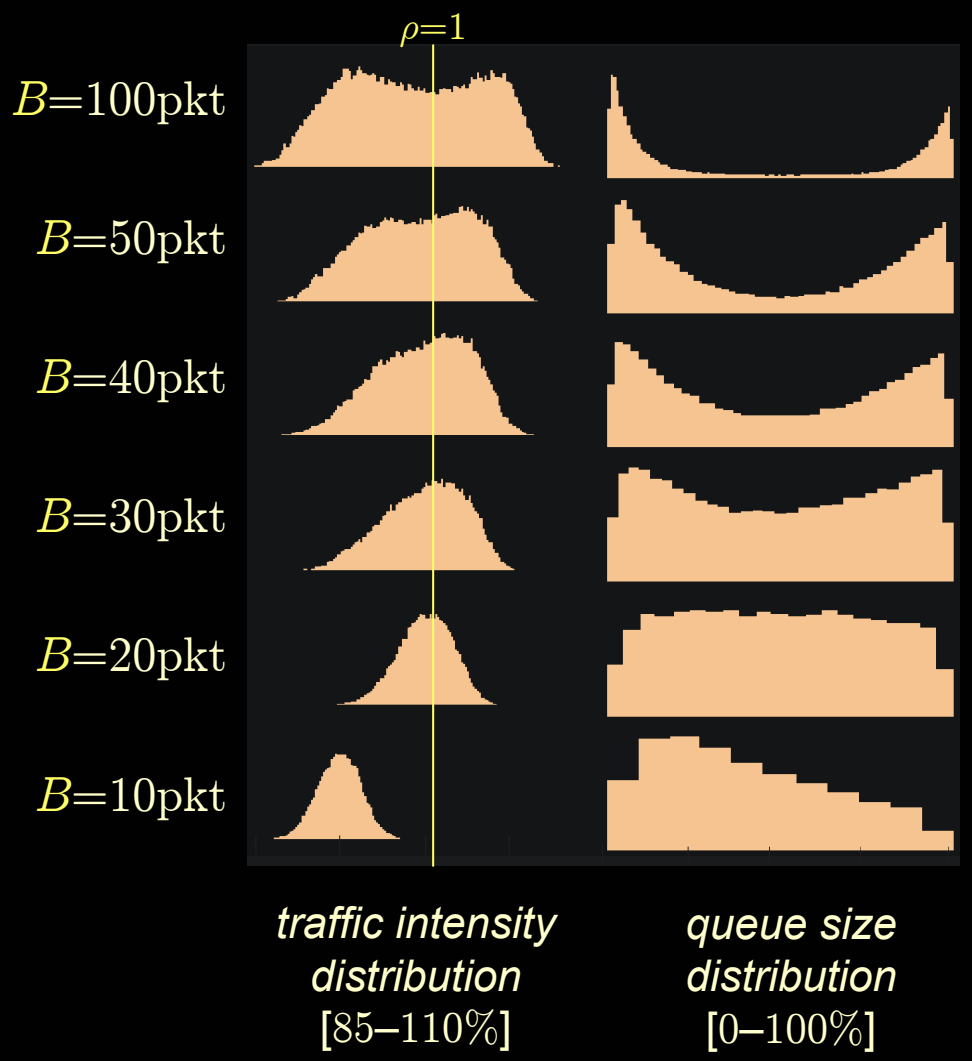
Instability = synchronization



- If the dynamical system is unstable, there will be oscillations in aggregate traffic
- These usually make the queue flip-flop
- Then there are short periods of concentrated packet loss
- This makes the flows become synchronized

- The billiards model for TCP is an extreme case of the fluid model

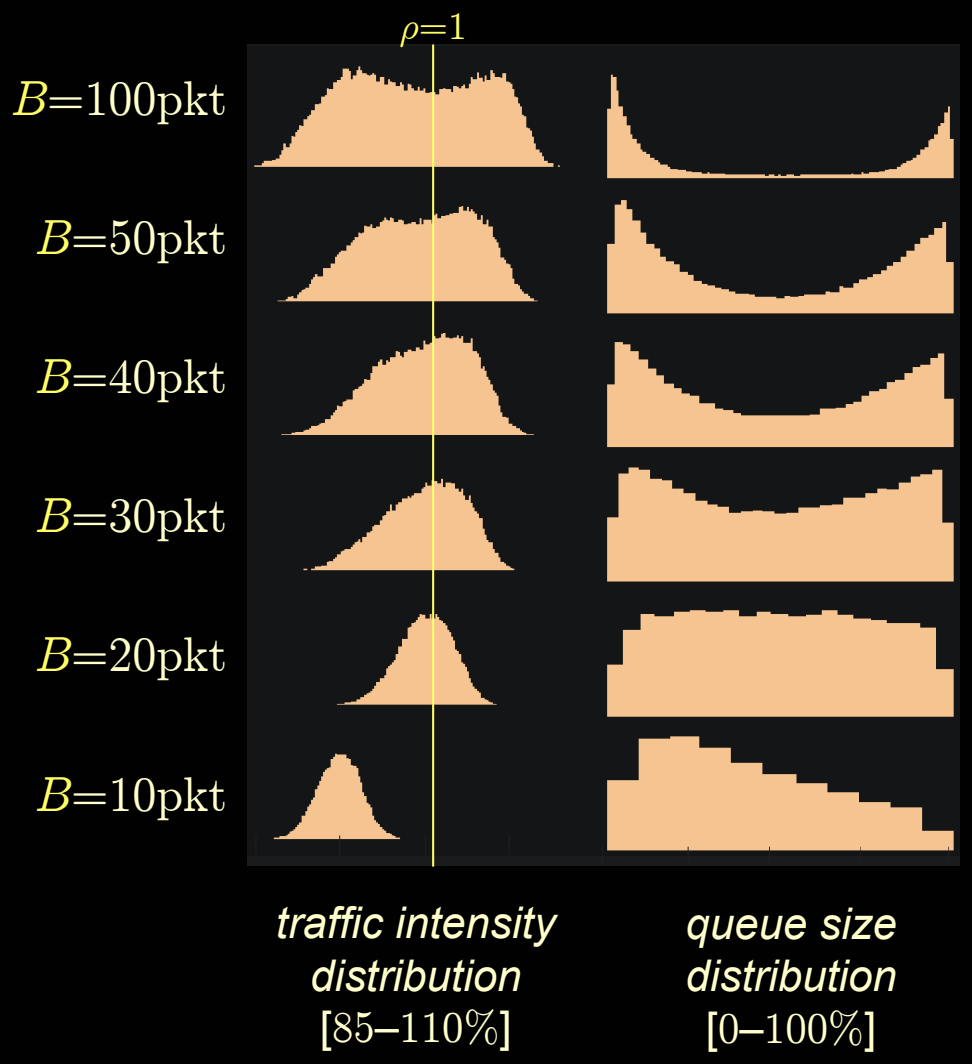
Instability & buffer size



- On the basis of this analysis, we claimed
  - a buffer of 30–50 packets should be sufficient
  - intermediate buffers will make the system become synchronized, and utilization will suffer
  - large buffers get high utilization, but cause synchronization and delay

Plot by Darrell Newman, Cambridge

Instability & buffer size

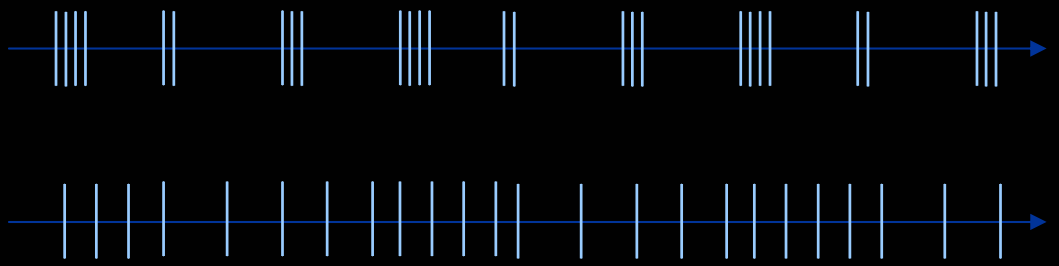


Plot by Darrell Newman, Cambridge

- On the basis of this analysis, we claimed
  - a buffer of 30–50 packets should be sufficient
  - intermediate buffers will make the system become synchronized, and utilization will suffer
  - large buffers get high utilization, but cause synchronization and delay
- Other people ran simulations and found
  - small buffers lead to terribly low utilization
  - intermediate buffers are much better

Burstiness

- The problem is that the Poisson limit broke down
  - a key step in the C+R proof is showing that each flow contributes at most one packet in a busy period
- TCP, on a fast access link, will produce bursty traffic
  - sources can send an entire window-full of packets back-to-back

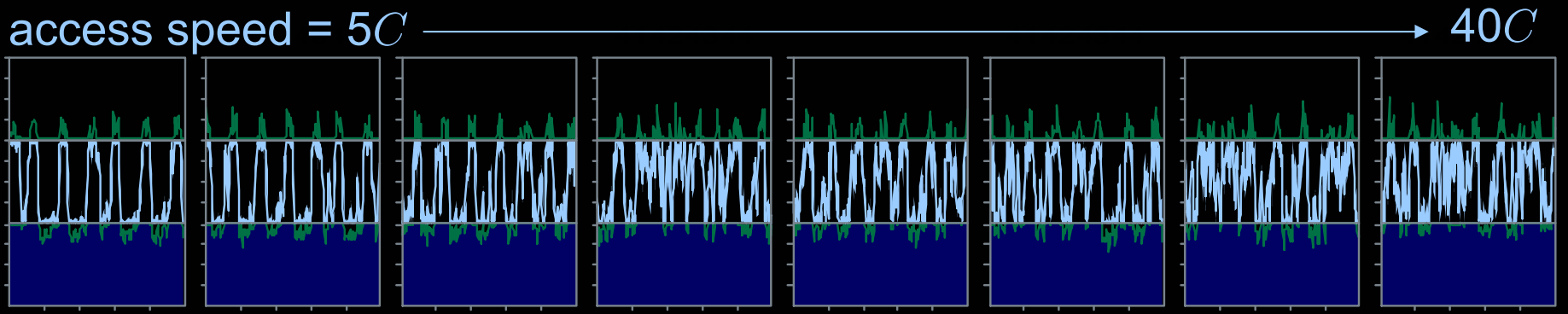


- We can estimate when the M/D/1/B model breaks down
  - calculate the typical duration of a busy period
  - count how many packets could arrive from a single source in this time
  - thus estimate how fast access speeds can be without the model failing
- For very fast access speeds, i.e. very bursty TCP traffic, a batch-M/D/1/B model is appropriate
  - where the batch size reflects the TCP window size

Burstiness & access speeds

- For fast access speeds, the problem of synchronization is reduced
  - the drop probability $p(\rho)$ for a batch-M/D/1/B queue with traffic intensity ρ becomes smoother as batch size increases
  - a smoother $p(\rho)$ function makes the dynamical system more stable

- Simulations of a system with a single bottleneck link with capacity NC, where each flow is throttled by a slow access link

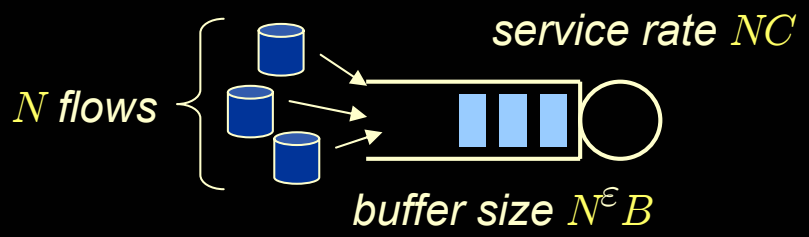


- Smoothing in networks of queues is also likely to have an impact

Queueing theory at short timescales

- Moral: we need better measurements of Internet traffic at short timescales, and we need more tractable techniques for short-timescale queueing theory

- James Cruise (Cambridge) is investigating large deviations for



  - intermediate between the C+R Poisson limit regime and the Weiss many-flows limit regime
  - typical duration of busy period is $O(N^{-(1-\varepsilon)})$
  - loss probability is $\approx \exp(-N^{\varepsilon} I)$
  - parsimonious traffic descriptors:
    it's just like doing large deviations for batch Poisson processes
  - the variational problem is simple to solve

Conclusion

- TCP chooses its own limit regime

- TCP, through its adaptive feedback mechanism, induces a fluid limit over the timescale of a round trip time
- The queue dynamics change fluidly over the timescale of a round trip time, and Poisson/stochastically over much shorter timescales
- The choice of buffer size determines which of three queueing models is relevant