Equation-based Congestion Control for Unicast Traffic

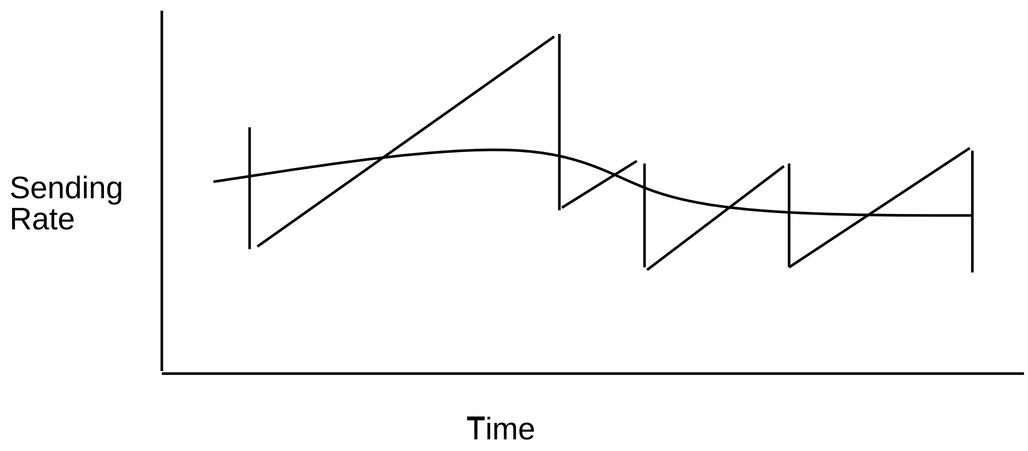
Sally Floyd and Mark Handley

AT&T Center for Internet Research at ICSI

December 1999

Outline of presentation:

- Why work on non-TCP forms of end-to-end congestiol control?
- Characterizing TCP:
- Alternate forms of Additive-Increase Multiplicative-Decrease congestion control (AIMD):
- Developing unicast equation-based congestion control:



Why work on non-TCP forms of end-to-end congestiol control?

- Traffic without end-to-end bandwidth guarantees (e.g., best-effort traffic, better-than-best-effort forms of diff-serv) requires end-to-end congestion control to avoid congestion collapse.

- TCP-based congestion control is not suitable for some unicast applications (e.g., streaming multimedia).

- Understanding equation-based congestion control for unicast is a first step towards designing viable congestion control for multicast applications.

Classical congestion collapse:

Congestion collapse occurs when the network is increasingly busy, but little useful work is getting done.

Problem: Classical congestion collapse:

– Paths clogged with unnecessarily-retransmitted packets [Nagle 84].

Status: A series of congestion collapses beginning in 1986.

Fix: Modern TCP retransmit timer and congestion control algorithms.

– [Jacobson 88].

TCP congestion control:

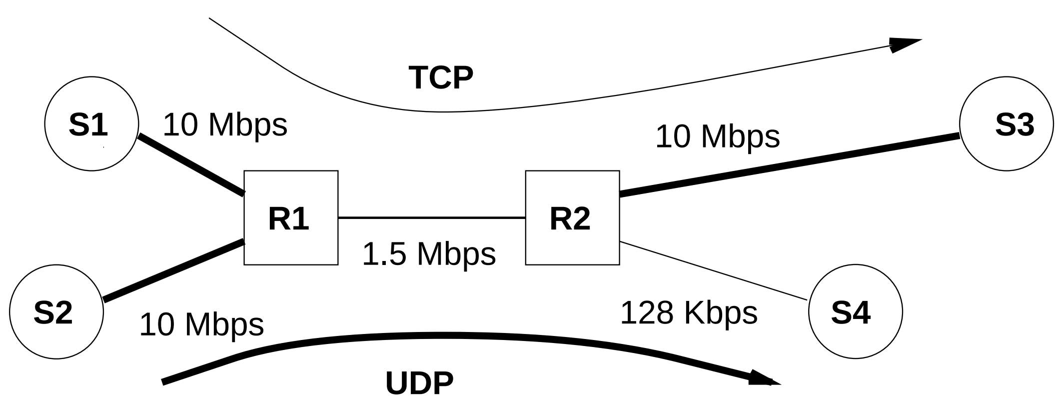
- Packet drops as the indications of congestion.
- TCP uses Additive Increase Multiplicative Decrease (AIMD)
  - from [Jacobson 1988].
  - Decrease congestion window by 1/2 after loss event.
  - Increase congestion window by one packet per RTT.
- In heavy congestion, when a retransmitted packet is itself dropped:
  - exponential backoff of the retransmit timer.
- Slow-start:
  - start by doubling the congestion window every roundtrip time.

Congestion collapse from undelivered packets:

Problem: Paths clogged with packets that are discarded before they reach the receiver [Floyd and Fall, 1999].

Status: There have been no reports of congestion collapse from undelivered packets. (Most traffic in the Internet uses TCP.)

Prevention: For each flow, either end-to-end congestion control, or a guarantee that packets entering the network will be delivered to the receiver.



Why do some unicast applications not use TCP?

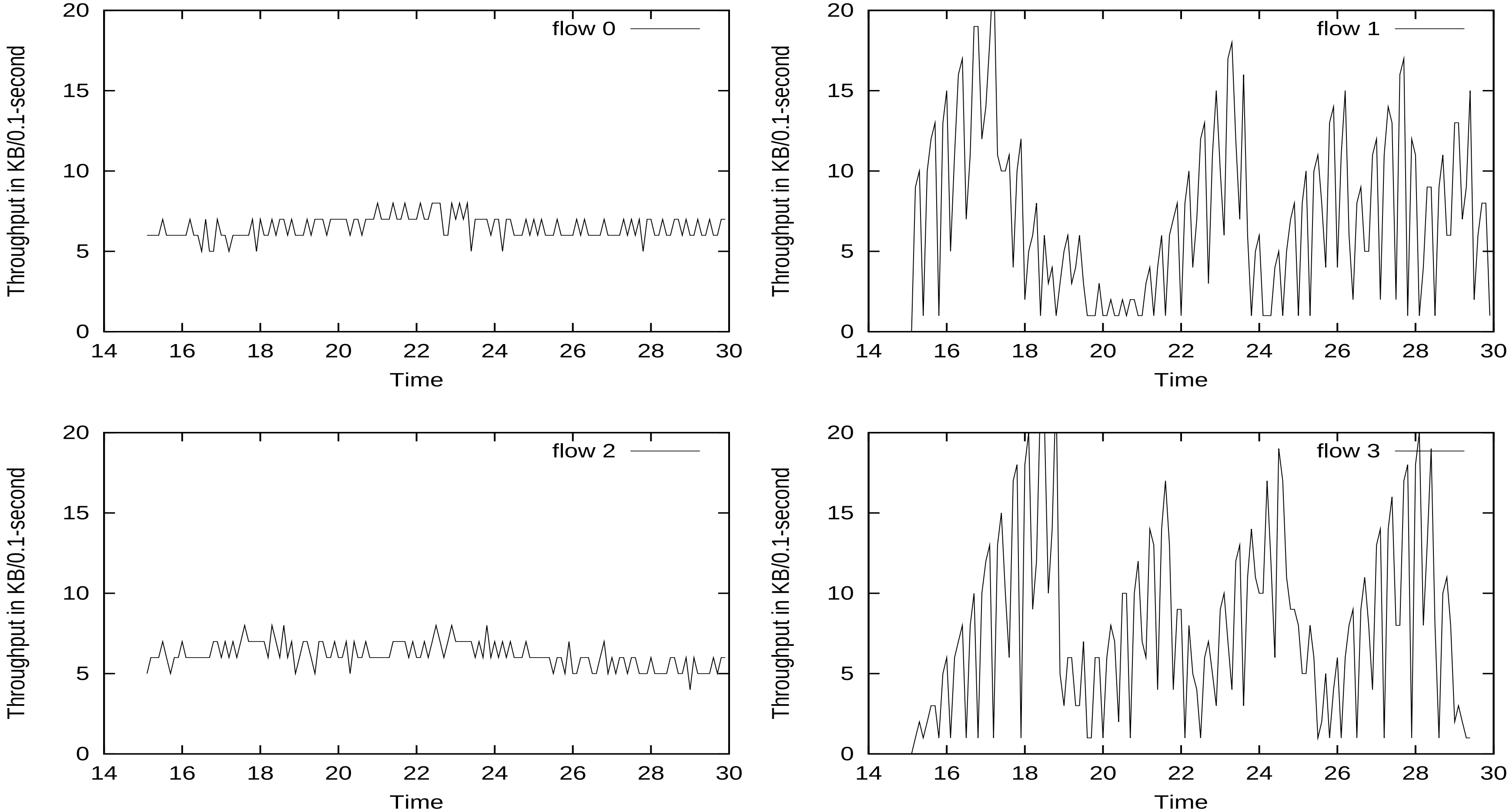
- Reliable delivery is not needed.

- Acknowledgements are not returned for every packet, and the application would prefer a rate-based to a window-based approach anyway.

- Cutting the sending rate in half in response to a single packet drop is undesirable.

- The Internet infrastructure does not yet provide either differentiated services, or standardized protocols with other forms of congestion control, as viable alternatives to TCP or non-congestion-controlled UDP.

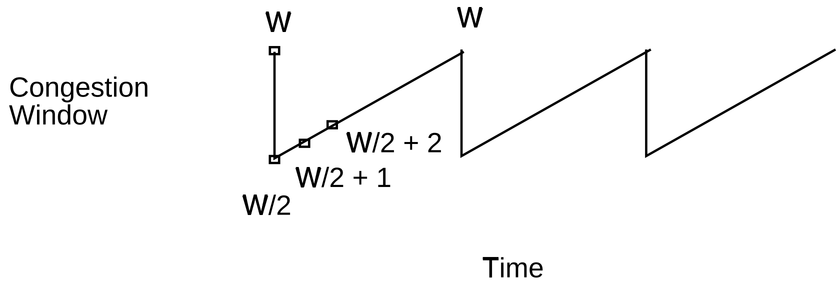
Why do some unicast applications not use TCP?



Equation-based congestion control (left column) and TCP (right column).

The simple “steady-state model” of TCP:

- The model:
  - Fixed roundtrip time R in seconds.
  - A packet is dropped each time the window reaches W packets.
  - TCP’s congestion window: W, $\frac{W}{2}$, $\frac{W}{2}$ + 1, ..., $W - 1$, W, $\frac{W}{2}$, ...



- The average sending rate T in pkts per sec: $T = \frac{3}{4}\frac{W}{R}$

- The packet drop rate p: $p = \frac{1}{(3/8)W^2}$

- T in pkts per sec: $T = \frac{\sqrt{3/2}}{R\sqrt{p}}$
  - or in bytes per sec, given B bytes per pkt: $T = \frac{\sqrt{3/2}B}{R\sqrt{p}}$

The improved “steady-state model” of TCP:

An improved steady-state model of TCP includes a fixed packet drop rate, retranmit timeouts, and the exponential backoff of the retransmit timer.

- The TCP response function:

$$T = \frac{B}{R\sqrt{\frac{2p}{3}} + 2R(3\sqrt{\frac{3p}{8}})p(1+32p^2)} \qquad (1)$$

T: sending rate in bytes/sec
B: packet size in bytes
R: roundtrip time
p: packet drop rate

– J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, Modeling TCP Throughput: A Simple Model and its Empirical Validation, SIGCOMM 98.

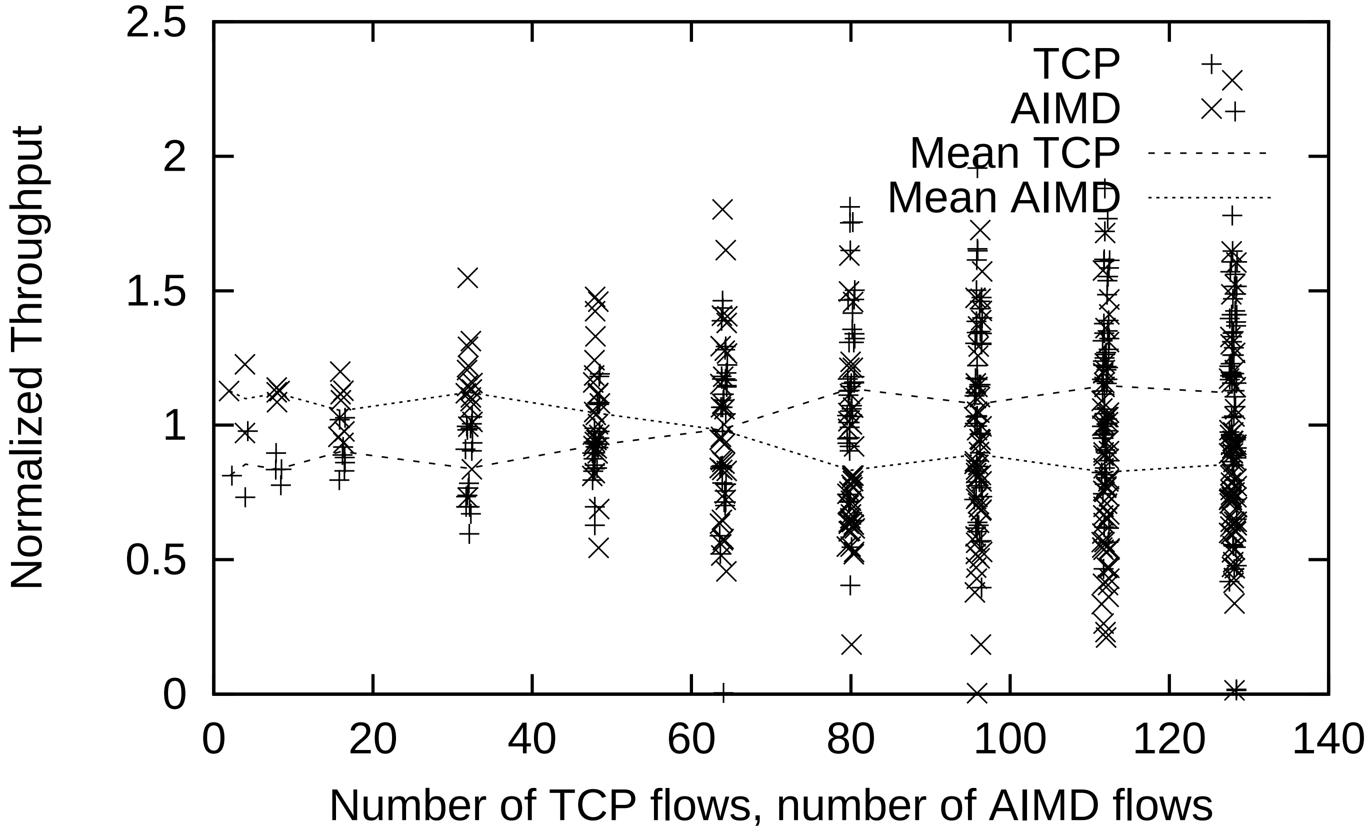
Other possibilities for end-to-end congestion control for unicast streaming media?

- Use a rate-based version of TCP's congestion control mechanisms, without TCP's ACK-clocking.
  - The Rate Adaption Protocol (RAP) [RH99].

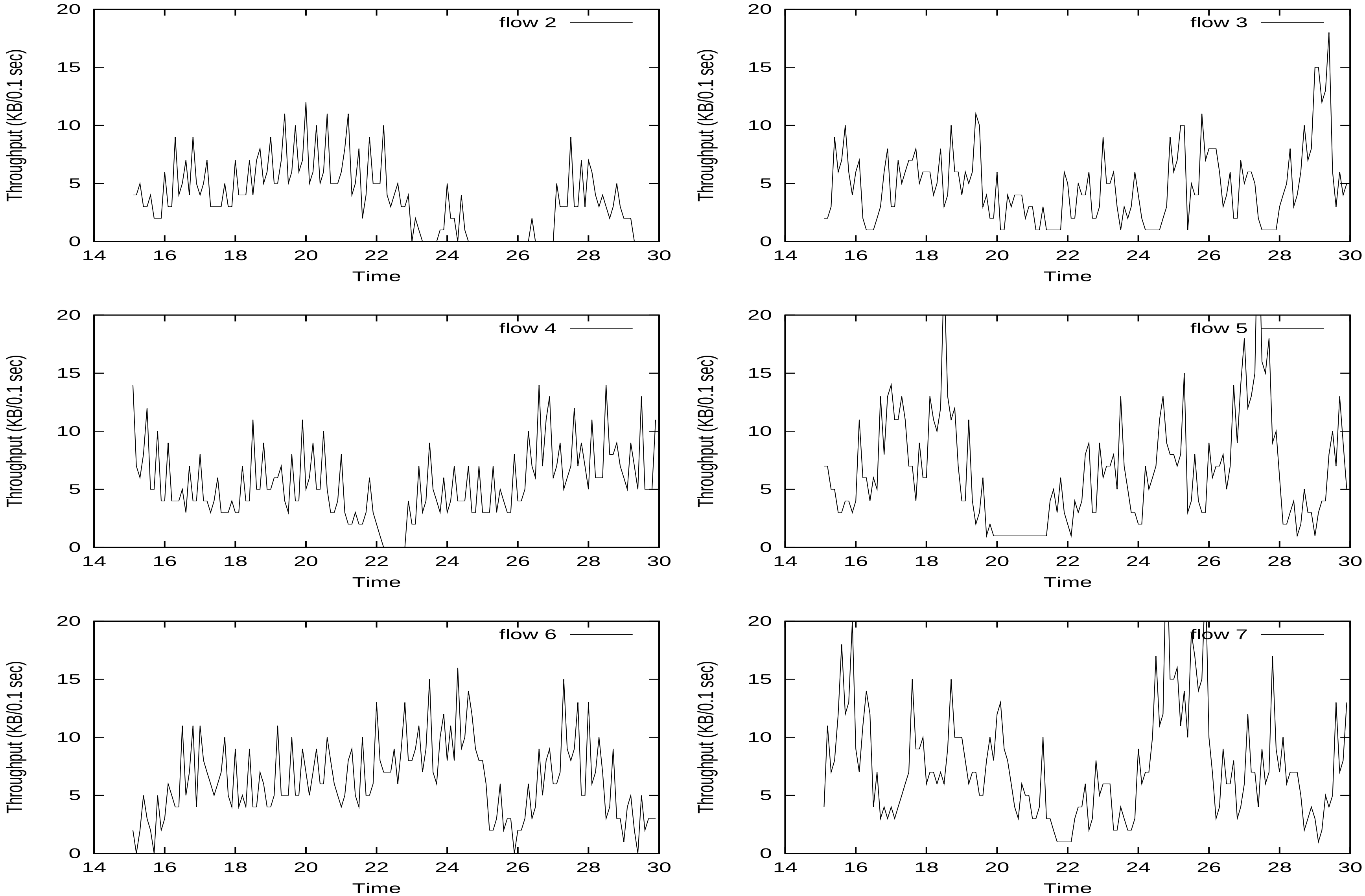
- AIMD with different increase/decrease constants.
  - E.g., decrease multiplicatively by 3/4, increase additively by 3/7 packets per RTT.

- Equation-based congestion control:
  - adjust the sending rate as a function of the longer-term packet drop rate.

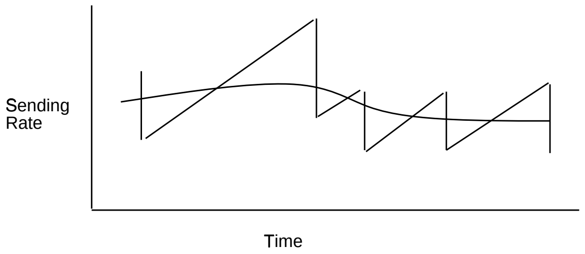
AIMD with different increase/decrease constants:



AIMD: decrease multiplicatively by 7/8, increase additively by 2/5 packets per RTT.



AIMD[2/5, 7/8] (left column) and TCP (right column) flows.



Equation-based congestion control:

• Use the TCP equation characterizing TCP's steady-state sending rate as a function of the RTT and the packet drop rate.

• Over longer time periods, maintain a sending rate that is a function of the measured roundtrip time and packet loss rate.

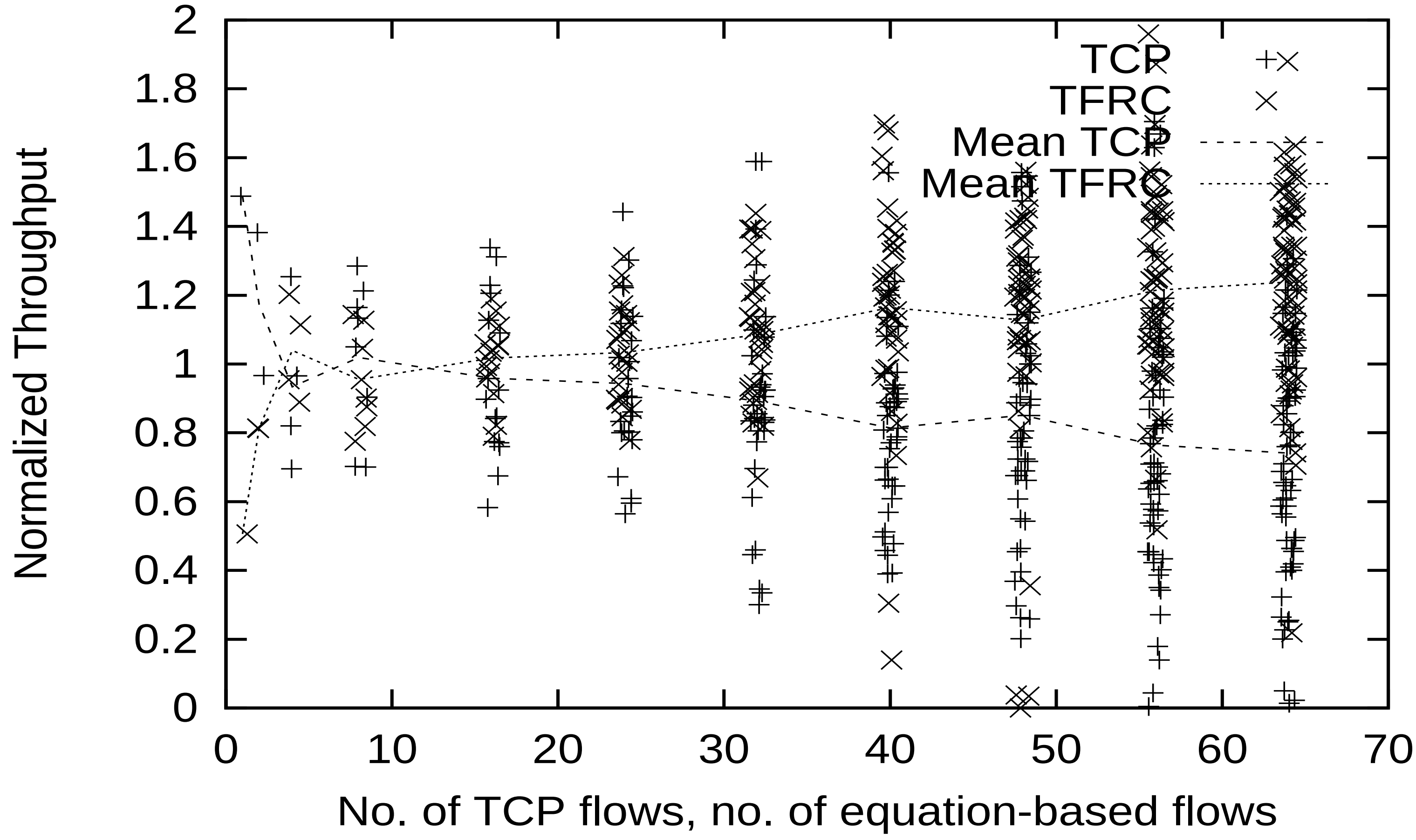
• The benefit: Smoother changes in the sending rate in response to changes in congestion levels.

• The justification: It is acceptable not to reduce the sending rate in half in response to a single packet drop.

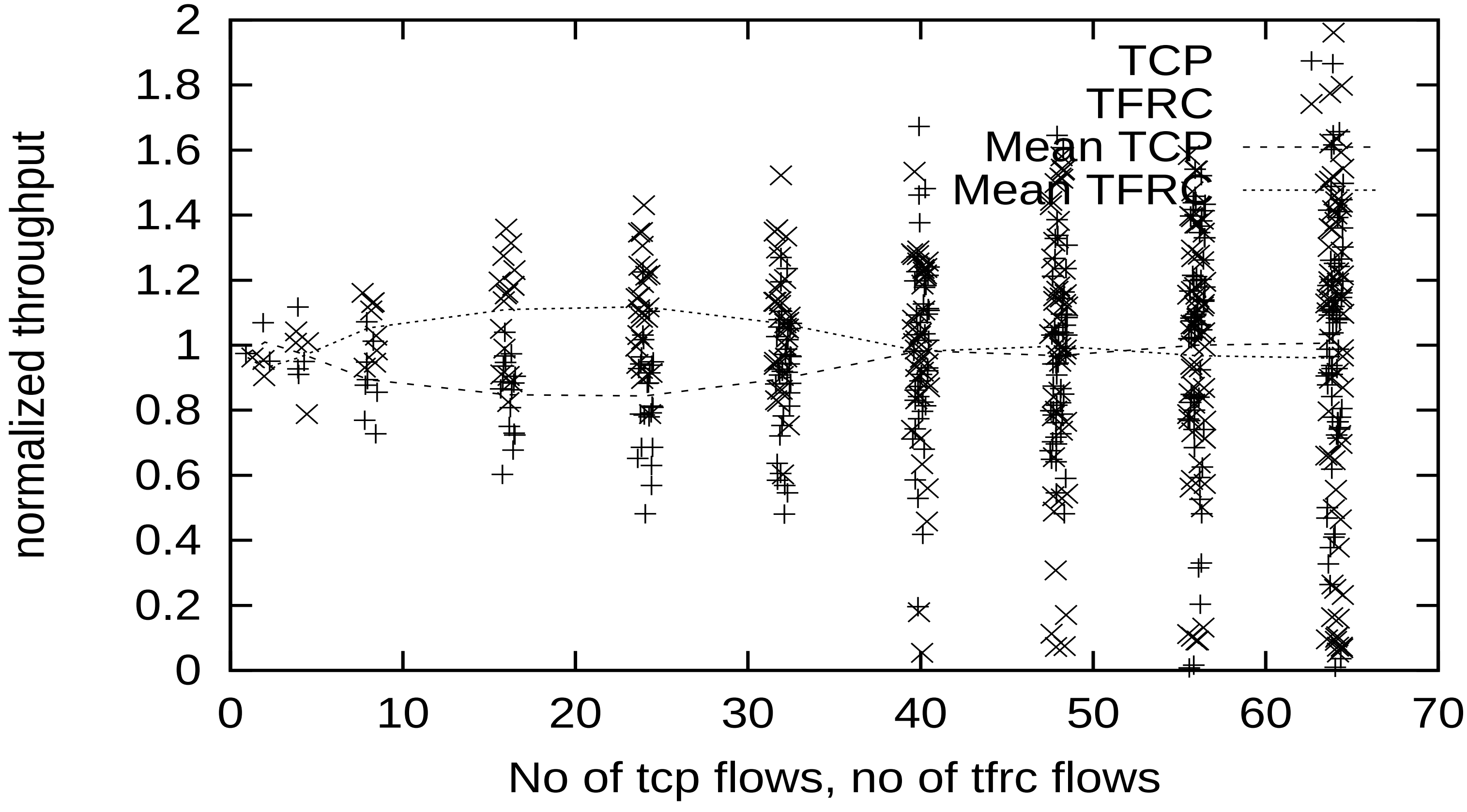
• The cost: Limited ability to make use of a sudden increase in the available bandwidth.

Why use the TCP equation in equation-based congestion control?

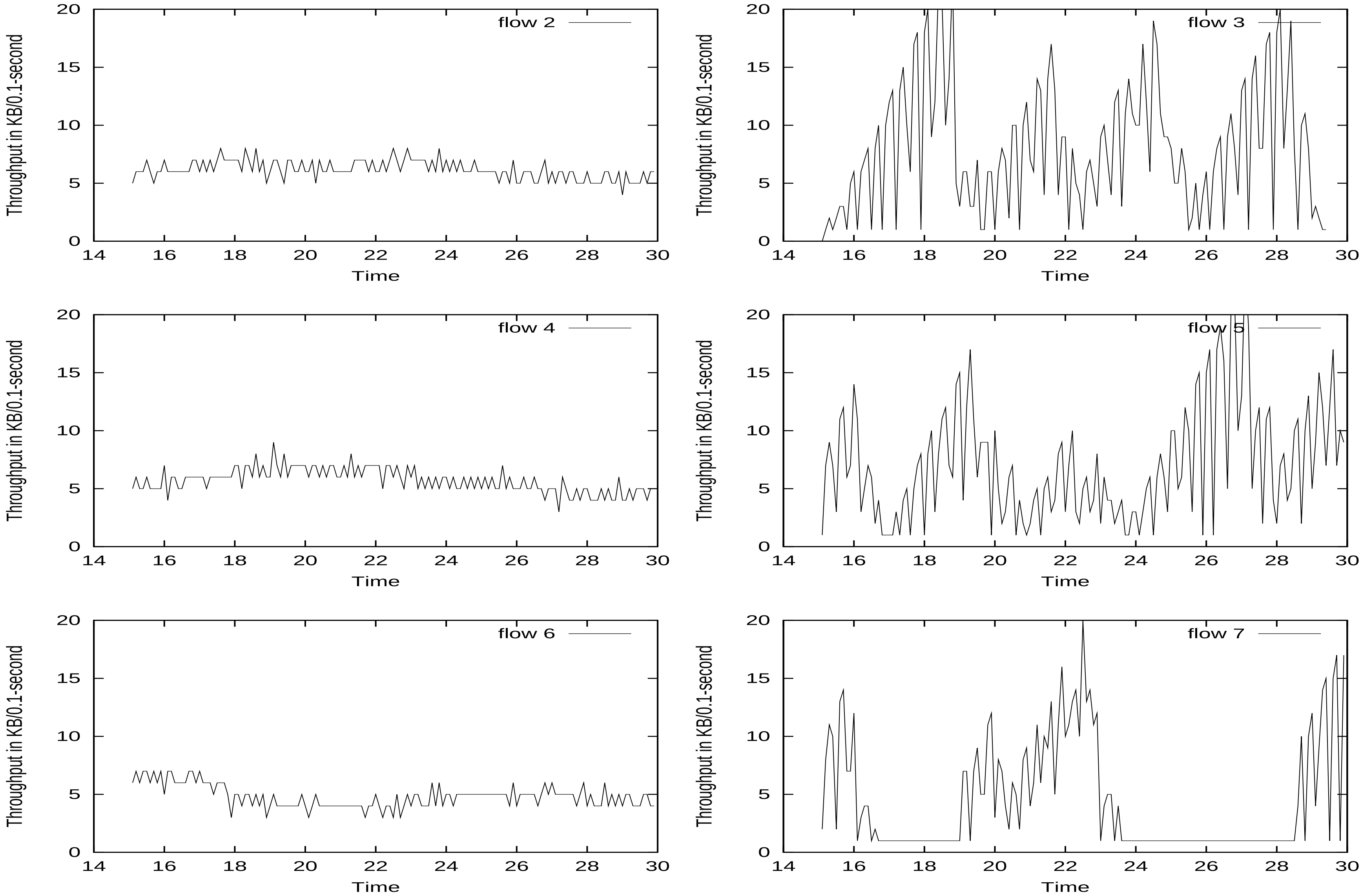
- Because best effort traffic in the current Internet is likely to compete in FIFO queues with TCP traffic.



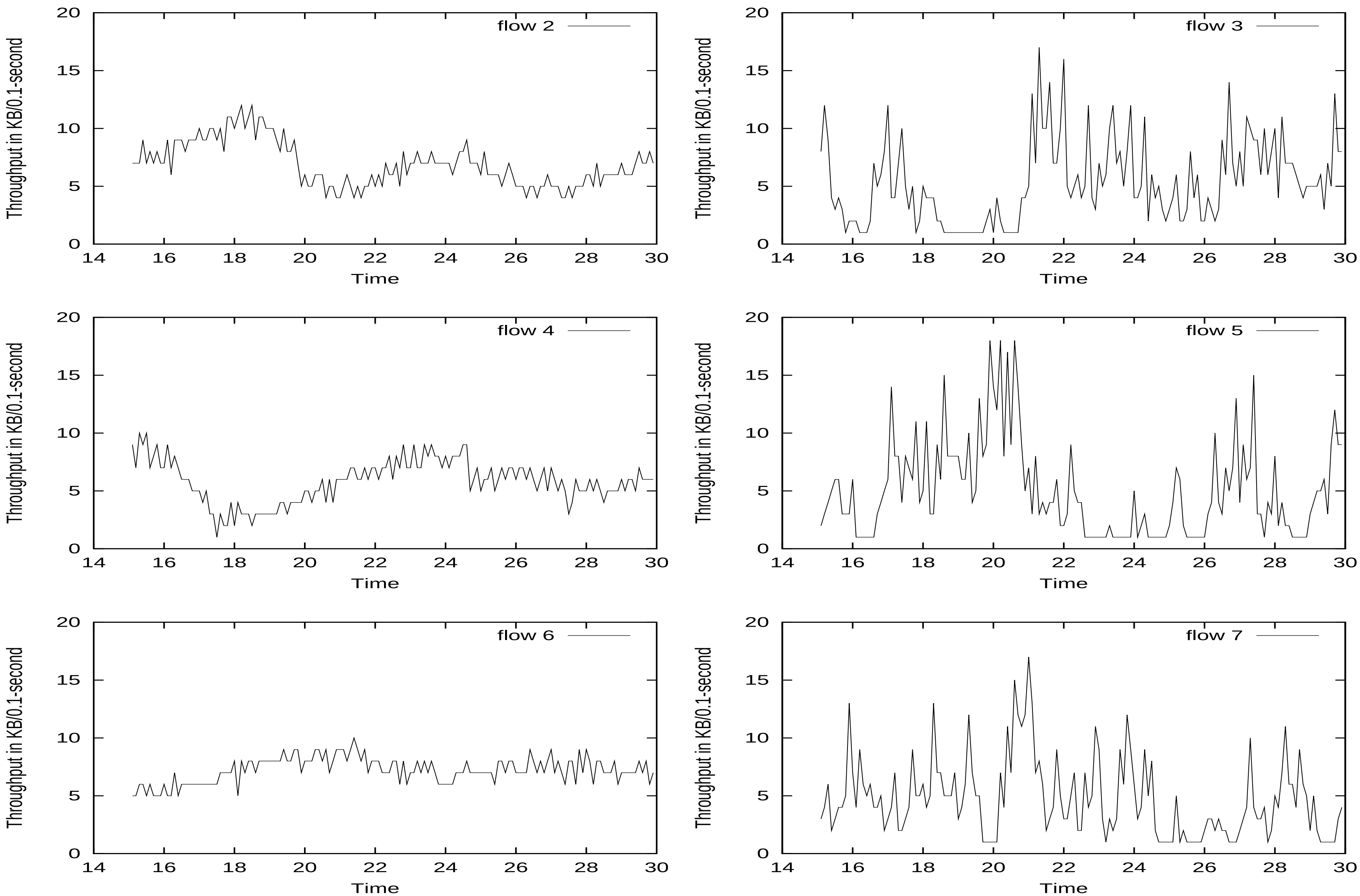
Why use the TCP equation in equation-based congestion control?



These simulation use RED instead of Drop-Tail queue management.



Equation-based congestion control and TCP (with Drop-Tail).



Equation-based congestion control and TCP (with RED).

Unicast: Estimating the packet drop rate:

- Goals for the receiver's estimated packet loss rate:
  - Maintains history of most recent loss events;
  - Estimates loss rate smoothly;
  - Responds promptly to successive loss events;
  - Estimated loss rate increases only in response to a new loss event;
  - Estimated loss rate decreases only in response to a new loss event, or to a longer-than-average interval since the last loss.

Unicast: Estimating the packet drop rate, cont.:

- The receiver estimates the average loss interval (e.g., the number of packet arrivals between successive loss events), and inverts to get the packet loss rate.
  - In estimating the average loss interval, the first four lost invervals are weighed equally.
  - The 5th-8th loss intervals are averaged using reduced weights.
  - The receiver reports the loss average to the sender once per RTT.

- The interval since the most recent packet drop counts as a loss interval, if it is longer than the average loss interval.

Unicast: The sender estimating the roundtrip time:

- The sender averages the roundtrip over the most recent several measured roundtrip times, using an exponential weighted moving average.

- The sender uses the average roundtrip time and packet drop rate in the "response function" to determine the allowed sending rate.

- If two report intervals pass without receiving the expected report from the receiver, cut the sending rate in half.

Unicast: The sender's increase/decrease algorithms:

- If allowed sending rate $<$ current sending rate, decrease sending rate:
  – down to allowed sending rate.

- If allowed sending rate $>$ current sending rate, increase sending rate:
  – by at most one packet/RTT;

If the current sending rate is less than one packet/RTT,

– increase the sending rate more slowly;

– increase half way up to the sending rate indicated by the equation.

Unicast: Goals for slow-start:

- Perform roughly as aggressively as TCP.
- Exit slow-start if regular feedback is not received from the receiver.
- Never send more than twice as fast as the receiver is receiving.
- On exiting slow-start, smoothly transition to equation-based congestion control:
  - Don't use the experienced packet drop rate directly;
  - Receiver estimates the available bandwidth;
  - Receiver computes the packet drop rate that corresponds to that bandwidth;

The future of congestion control in the Internet: several possible views:

- View #1: No congestion, infinite bandwidth, no problems.
- View #2: The "co-operative", end-to-end congestion control view.
- View #3: The game theory view.
- View #4: The congestion-based pricing view.
- View #5: The virtual circuit view.
- The darker views: Congestion collapse and beyond.