Congestion Control for Streaming Media

Sally Floyd
EE290T, UC Berkeley
October 15, 1999

http://www.aciri.org/floyd/talks/streaming_Oct99.ps
http://www.aciri.org/floyd/talks/streaming_Oct99.pdf

Outline of talk:

- Why do we need end-to-end congestion control?
- Characterizing TCP congestion control
- Equation-based congestion control for unicast traffic.
- Equation-based congestion control for multicast traffic.
- Related issues: RED, ECN, FEC, diff-serv, CM (Congestion Manager), and others.

Sub-themes:

- The Internet is a work in progress, with no central control or authority, many players independently making changes, and many forces of change (e.g., new technologies, new applications, new commercial forces, etc.)

- So far, the success of the Internet has rested on the IP architecture's robustness, flexibility, and ability to scale, and not on its efficiency, optimization, or fine-grained control.

- The rather decentralized and fast-changing evolution of the Internet architecture has worked reasonably well to date. There is no guarantee that it will continue to do so.

- The Internet is like the elephant, and each of us is the blind man who knows only the part closest to us.
  - The part of the Internet that I see is end-to-end congestion control.

Why do we need end-to-end congestion control?

- Fairness.

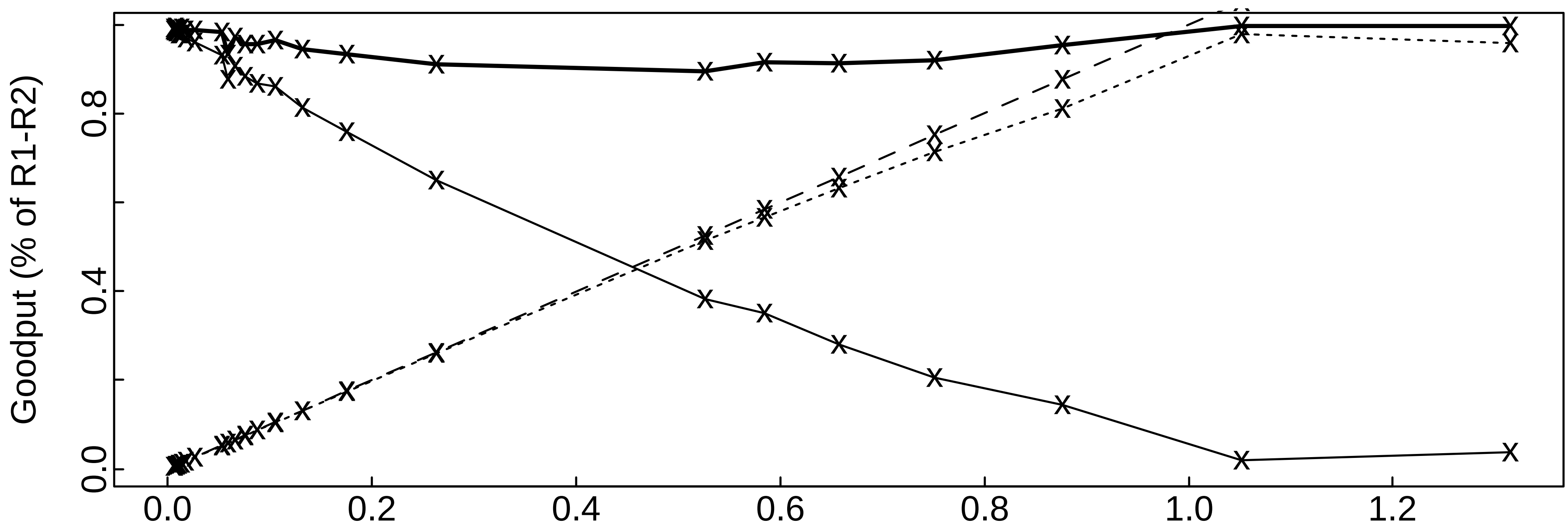
- To avoid congestion collapse.

- As a tool for the application to better achieve its own goals: E.g., minimizing loss and delay, maximizing throughput.

What is the fairness goal? (the pragmatic answer)

- No connection/session/end-node should hog the network resources.
  - TCP is the dominant transport in the Internet (90-95% of the bytes/packets)
  - The current Internet is dominated by best-effort traffic and FIFO scheduling at the routers.
  - New forms of traffic that compete with TCP as best-effort traffic in FIFO queues should not be significantly more aggressive (or significantly less aggressive).

Why is fairness a concern?



X-axis: UDP Arrival Rate (% of R1-R2). Dashed Line: UDP Arrivals; Dotted Line: UDP Goodput; Solid Line: TCP Goodput; Bold line: Aggregate Goodput

Simulations showing three TCP flows and one UDP flow (without end-to-end congestion control), with a congested link using FIFO scheduling.

What is the fairness goal? (other possible answers)

- Fairness goals not dependent upon pricing:

  – Min-max fairness: On each link of the network, each entity has an equal claim to the bandwidth of that link. (e.g., Fair Queueing.)

  – Global fairness: Each entity has an equal claim to the scarce resources (where an entity traversing N congested links is using N times more scarce resources than an entity traversing 1 congested link).

  – Other fairness goals ...

- Fairness goals related to pricing:

  – Pricing: For some services, bandwidth is allocated to those willing to pay for it. (E.g., intserv, diffserv.)

  – Congestion-based pricing: The “cost” of the bandwidth on each link varies as a function of the level of congestion (e.g., the packet drop rate).

Why is congestion collapse a concern?

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].

Fix: Modern TCP retransmit timer and congestion control algorithms [Jacobson 88].

Fragmentation-based congestion collapse:

Problem: Paths clogged with fragments of packets invalidated because another fragment (or cell) has been discarded along the path. [Kent and Mogul, 1987]

Fix: MTU discovery [Kent et al, 1988],
Early Packet Discard in ATM networks [Romanow and Floyd, 1995].

Congestion collapse from undelivered packets:

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

Fix: Either end-to-end congestion control, or a “virtual-circuit” style of guarantee that packets that enter the network will be delivered to the receiver.

TCP

S1 10 Mbps 10 Mbps S3

R1 1.5 Mbps R2

S2 10 Mbps 128 Kbps S4

UDP

How can end-to-end congestion control be useful to an application for its own reasons?

- In an environment of either per-flow scheduling or small-scale statistical multiplexing:
  - The loss and delay experienced by a flow is affected by its own sending rate.
  - The use of end-to-end congestion control can reduce unnecessary loss and delay for that flow.

How can end-to-end congestion control be useful to an application for its own reasons? Part 2:

- In an environment of FIFO scheduling and large-scale statistical multiplexing at all congestion points:
  - The loss rate and delay experienced by a flow is largely independent of its own sending rate (holding the congestion control behavior of all other flows fixed).
  - End-to-end congestion control can be useful to a flow to avoid mechanisms that could be deployed by the network to penalize best-effort traffic that doesn't use end-to-end congestion control in a time of congestion.

- Tragedy of the commons is avoided in part because the "players" are not individual users determining their own end-to-end congestion control strategy and "gaming" against other users.

Characterizing TCP congestion control

- TCP uses Additive Increase Multiplicative Decrease (AIMD).
  - 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, use exponential backoff of the retransmit timer.

- Slow-start: start by doubling the congestion window every roundtrip time.

Why not use TCP for unicast streaming media?

- 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.

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

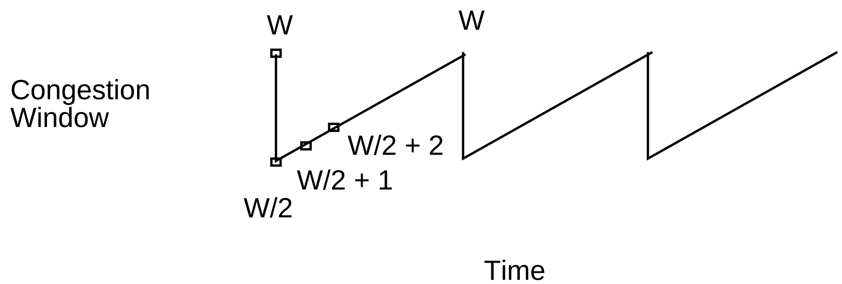
- Use a rate-based version of TCP's congestion control mechanisms, without TCP's ACK-clocking.
  - The Rate Adaption Protocol (RAP) [RH99].

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

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

The "steady-state model" of TCP:

- The model: Fixed packet size B in bytes.
  - Fixed roundtrip time R in seconds, no queue.
  - 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, $\frac{W}{2}$, ...

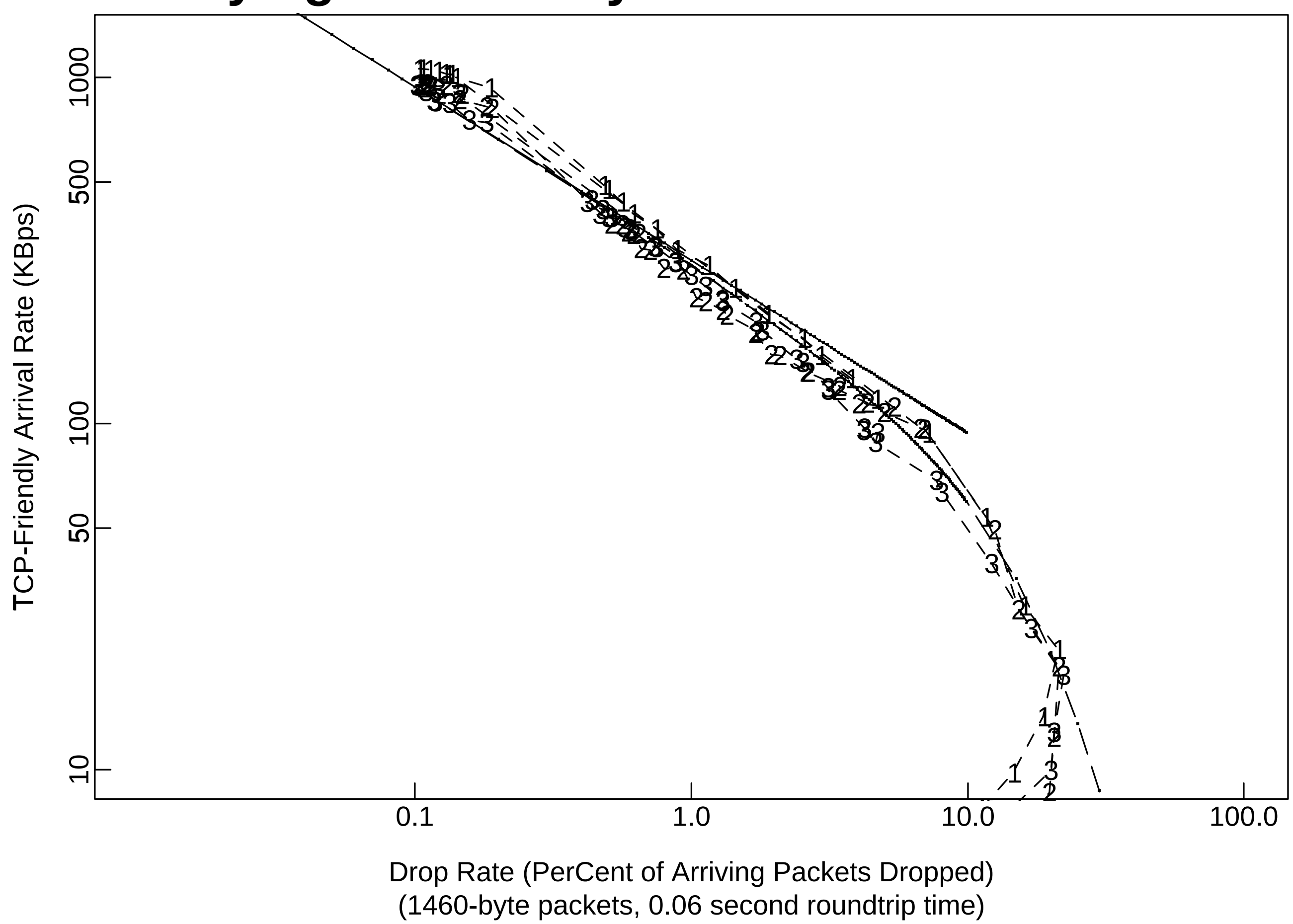


- The maximum sending rate in packets per roundtrip time: W
  - The maximum sending rate in byes per second: WB/R
  - The average sending rate T: $T = (3/4)WB/R$

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

- The result: $T = \frac{\sqrt{6}B}{2R\sqrt{p}} = \frac{\sqrt{3/2}B}{R\sqrt{p}}$

Verifying the “steady-state model” of TCP:



Solid line: the simple equation characterizing TCP

Numbered lines: simulation results

The “steady-state model” of TCP: an improved version.

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

T: sending rate in bytes/sec

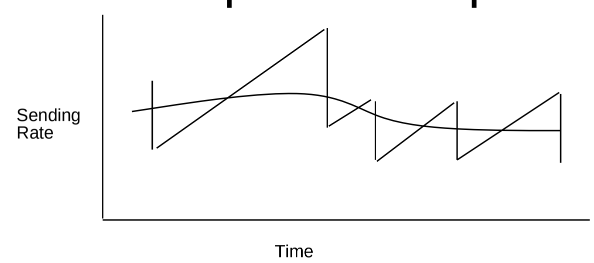
B: packet size in bytes

p: packet drop rate

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

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.
  – Loss event: One or more packet drops/marks in a roundtrip time.

- 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.

Given equation-based congestion control, why use the “TCP-friendly” equation?

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

- Criteria for evaluating an equation for equation-based congestion control:
  - – Stability, potential for oscillations.
  - – Adaptive range: Range in packet drop rate needed for desired range in sending rate.
  - – Sending rate as a function of the roundtrip time?

(How does this generalize to multicast?)

  - – Sending rate as a function of the packet size?

$$T = \frac{\sqrt{3/2}B}{R\sqrt{p}}$$

Further evaluation of equation-based congestion control:

- Stability, oscillations.
- Synchronization among multiple flows.
- Long-term fairness with respect to TCP.
- Transient performance.

Equation-based congestion control in an environment with FIFO scheduling and large-scale statistical multiplexing:

- Packet drop rate is largely independent of individual flow's sending rate.

- The flow monitors the long-term packet drop rate, and the roundtrip time, and adjusts its long-term sending rate accordingly.

- Benefit over TCP: Smoother changes in the sending rate in response to changes in congestion levels.

Equation-based congestion control in an environment with either per-flow scheduling, or small-scale statistical multiplexing:

- Packet drop rate is in part a function of individual flow's sending rate.

- There is an upper bound on the allowed increase in the sending rate.
  – (Increase in sending rate $\rightarrow$ increase in packet drop rate.)

- Concern: The steady-state "model" assumes a fixed roundtrip time. The actual roundtrip time can vary significantly as a function of the sending rate (if queueing delay dominates propagation delay).

Equation-based congestion control: a specific proposal

– Joint work with Mark Handley, Jitendra Padhye, and Joerg Widmer.

- The receiver averages the packet loss rate over the most recent K loss intervals, for K=4.

  – A loss interval is a sending period ending in a loss event (e.g., one or more packet drops in a window of data).

  – The average also takes into account the K+1, K+2, and K+3-rd loss intervals, with reduced weights.

  – The receiver reports the loss average to the sender once per RTT.

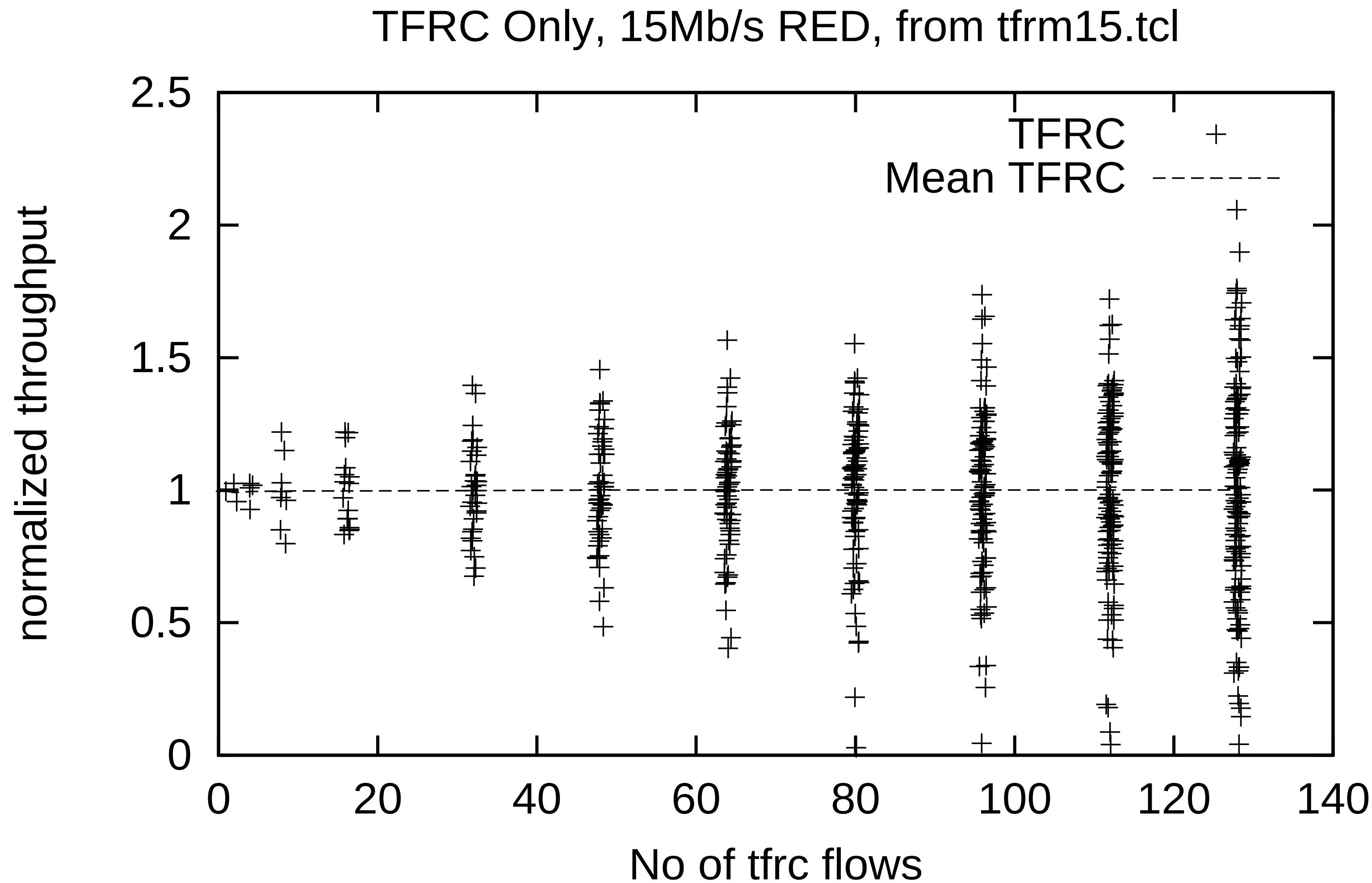
- The sender averages the roundtrip over the most recent several measured roundtrip times, using an exponential weighted moving average.

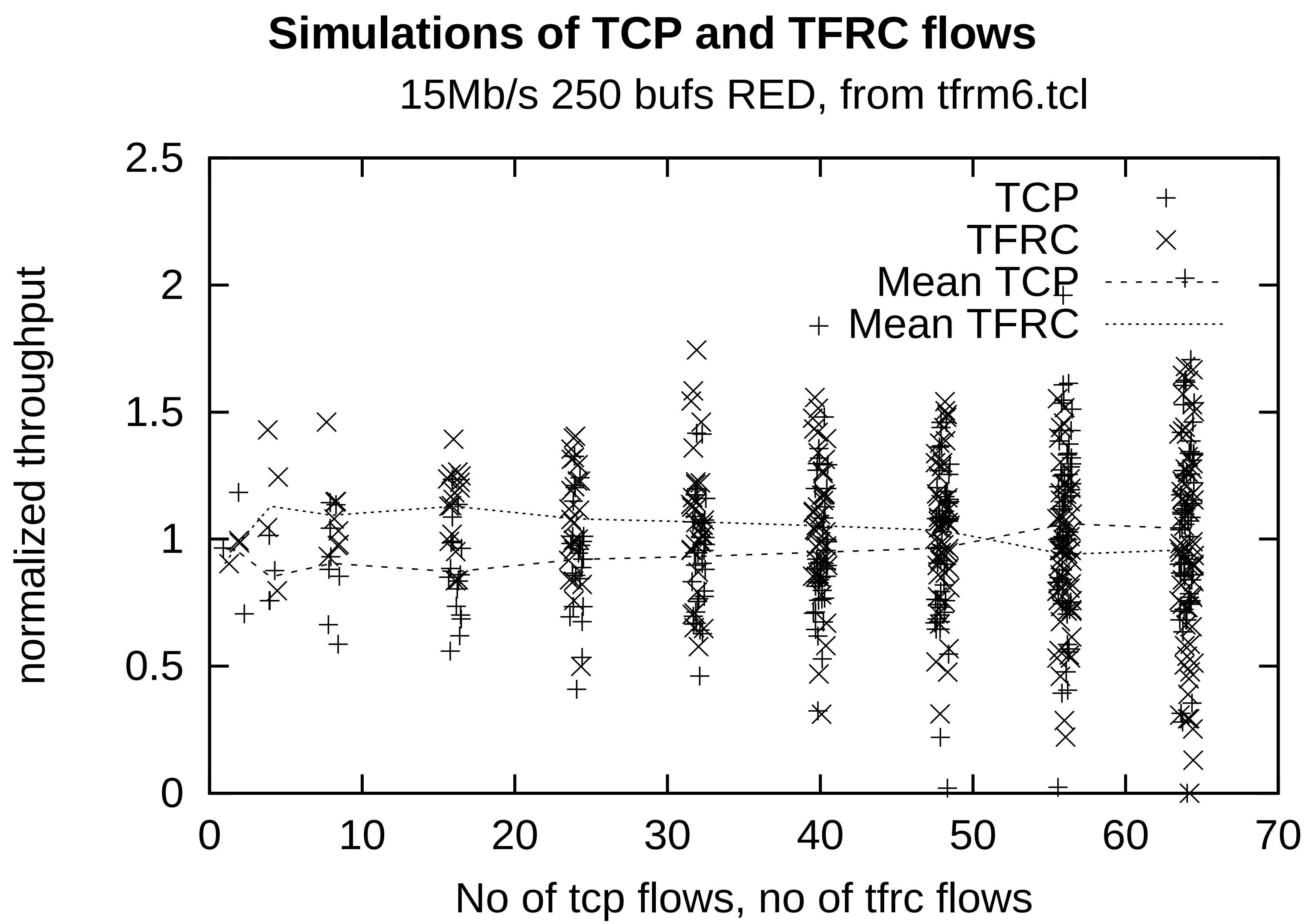
- Using the equation, the sender calculates its allowed sending rate.
  - If allowed sending rate $<$ current sending rate, decrease sending rate down to allowed sending rate.
  - If allowed sending rate $>$ current sending rate, increase sending rate, but by at most one packet/RTT.
  - If the sending rate is less than one packet/RTT, increase the sending rate more slowly.

- Slow-start:
  - Increase the sending rate by a factor ssmult (e.g., 2) each RTT.
  - Rate increases are “smoothed out” over a RTT.
  - Twice the receiver’s reported receive rate is an upper bound on the sending rate.

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

Simulations with TFRC: TCP-friendly Rate Control




Simulations of TCP and TFRC flows
15Mb/s 250 bufs RED, from tfrm6.tcl
normalized throughput
0
0.5
1
1.5
2
2.5
0
10
20
30
40
50
60
70
No of tcp flows, no of tfrc flows
TCP
TFRC
Mean TCP
Mean TFRC

Equation-based congestion control for single-sender multicast traffic:

- Advantages of equation-based congestion control for multicast:
  - The sender does not have to hear about every packet drop from every receiver.
  - The sender responds over slightly-slower time scales than does TCP.

Single-sender multicast: simple congestion control.

- If receivers did not have to measure their RTT to the sender:
  - Each receiver could simply measure its packet drop rate.
  - Some mechanism could be used (probabilistic feedback, tree-structured feedback) for the sender to learn the worst-case packet drop rate.

- Drawback:
  - The sending rate would be limited by the combination of the worst-case RTT and the worst-case packet drop rate, even if these two worst-cases were not experienced by the same receiver [Whetten 98].

Single-sender multicast: simple congestion control, attempt #2:

- Assume that all members of the multicast group have synchronized clocks (e.g., GPS).
  - Each receiver can determine the one-way time from the sender to that receiver.

- The sender reports its current sending rate.

- Receivers know from their combined packet drop rate and RTT whether their feedback would cause the sender to slow down.

- Probabilistic or tree-structured mechanisms are used for feedback to the sender.

Single-sender multicast: more complicated congestion control:

- No assumption of synchronized clocks.

- Receivers with high packet drop rates have to “measure” their RTT to the sender using some mechanism.
  – Receivers know from their combined packet drop rate and RTT whether their feedback would cause the sender to slow down.

Other complications introduced by multicast:

- How aggressively can the sender slow-start?

- Does the sender need positive feedback to keep on sending, or do receivers have the responsibility to unsubscribe if their congestion control feedback is not reaching the sender?

- What are the transient traffic dynamics when round trip times are changing due to increased queueing delay, for example?

Other approaches to congestion control for multicast traffic:

- Intserv (integrated services) and some forms of diffserv (differentiated services) eliminate the need for end-to-end congestion control.

- Layered multicast, with receivers subscribing and unsubscribing from layered multicast groups.

Related issues: Explicit Congestion Notification (ECN)

- Active queue management (e.g., RED) is being incorporated into routers.
  – Routers measure the average queue size, and probabilisticly drop packets before buffer overflow, as an indication of congestion to end nodes.

- Given that routers are not necessarily waiting until buffer overflow to drop a packet, routers can set an ECN bit in the packet header instead of dropping the packet, to inform end-nodes of congestion.

- ECN is an experimental addition to the IP architecture [RFC 2481].
  – ECN-Capable Transport (ECT) indication from sender to router.
  – Congestion Experienced (CE) indication from router to receiver.
  – For TCP, TCP-level feedback from TCP receiver to TCP sender about ECN indications.

Related issues: the Congestion Manager

- The Congestion Manager: a proposal for a congestion control mechanism that would reside below the transport layer (e.g., below UDP and TCP), and provide integrated congestion control for flows that share the same source-destination pair [HRS99].

- The first step: congestion control provided by the sender, for flows that have end-to-end feedback about packet drops/marks.
  - This end-to-end feedback about losses could be at the transport layer (e.g. TCP), or at the application layer (for UDP traffic).

- A longer-term research question: congestion control provided by a collaboration between the sending and receiving node, including detection and feedback about packet drops/marks.

References:

[TCP-Friendly Web Page] The TCP-Friendly Web Page, URL “http://www.psc.edu/n

[Floyd and Fall, 1999] Floyd, S., and Fall, K., Promoting the Use of End-to-End Congestion Control in the Internet, IEEE/ACM Transactions on Networking, August 1999. URL “http://www.aciri.org/floyd/papers.html”.

[HRS99] Hari Balakrishnan, Hariharan S. Rahul, and Srinivasan Seshan, An Integrated Congestion Management Architecture for Internet Hosts, SIGCOMM 99, September 1999. URL “http://inat.lcs.mit.edu/papers/BRS99.html”.

[Jacobson 88] Jacobson, V., Congestion Avoidance and Control. Proceedings of SIGCOMM ’88 (Palo Alto, CA, Aug. 1988) URL “http://www-nrg.ee.lbl.gov/nrg-papers.html”.

[Kent and Mogul, 1987] C. Kent and J. Mogul, ”Fragmentation Considered Harmful,” ACM Computer Communication Review, vol. 17, no. 5, Aug. 1987.

[Kent et al, 1988] J. C. Mogul, C. A. Kent, C. Partridge and K. McCloghrie, ”IP MTU discovery options,” Internet Engineering Task Force, RFC 1063, Jul. 1988.

[Nagle 84] John Nagle, ”Congestion control in IP/TCP internetworks,” ACM Computer Communication Review, vol. 14, no. 4, pp. 11–17, Oct. 1984.

[RFC 2481] Ramakrishnan, K.K., and Floyd, S., A Proposal to add Explicit Congestion Notification (ECN) to IP. RFC 2481, Experimental, January 1999. URL “http://www.aciri.org/floyd/papers.html”.

[RH99] RAP: An End-to-end Rate-based Congestion Control Mechanism for Realtime Streams in the Internet, R. Rejaie, M. Handley, D. Estrin. Proc. Infocom 99. URL “http://www.aciri.org/mjh/papers.html”.

[Romanow and Floyd, 1995] Romanow, A., and Floyd, S., Dynamics of TCP Traffic over ATM Networks. IEEE JSAC, V. 13 N. 4, May 1995, p. 633-641. URL “http://www.aciri.org/floyd/papers.html”.

[Whetten 98] B. Whetten, J. Conlan, A Rate Based Congestion Control Scheme for Reliable Multicast. Technical White Paper, GlobalCast Communications, October 1998.