TCP-Friendly Equation-Based Congestion Control

Jörg Widmer, University of Mannheim

(widmer@informatik.uni-mannheim.de)

09 December 2002

Overview

- Introduction to congestion control
- Equation-based congestion control (TFRC)
- Congestion control for flows with small packets
- Concluding remarks

- (Extending TFRC to multicast)

Why Use Congestion Control?

- Increasing volume of non-TCP traffic
- Multicast transport protocols
- Wireless communication
- High speed Internet connections for end-users

- (Low delay in the network)

Foundations of Equation-Based CC

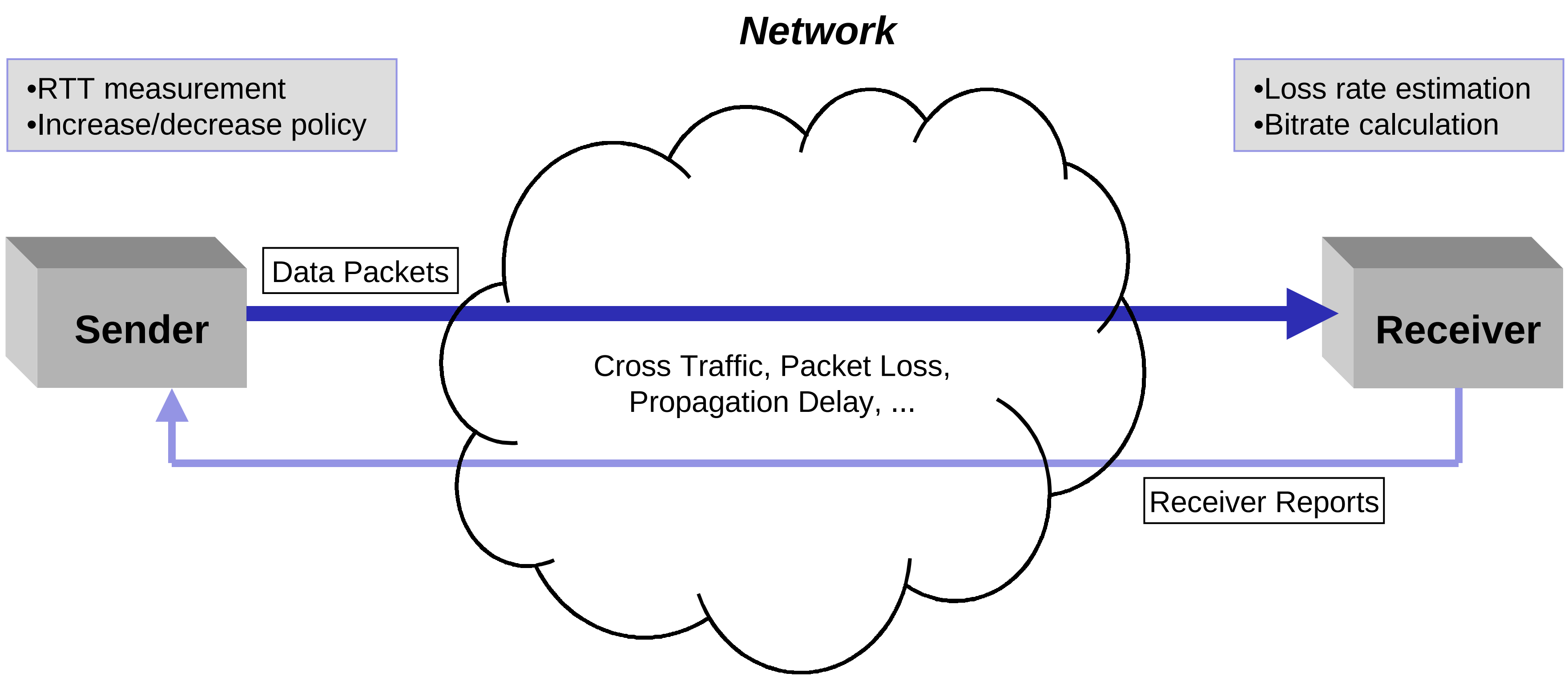
- Adapt rate to long-term steady-state TCP throughput
- Don't reduce rate by half in response to a single congestion indication

Equation for TCP throughput (Padhye, et. al.):

$$T = \frac{s}{t_{RTT}\left(\sqrt{\frac{2p}{3}} + 12\sqrt{\frac{3p}{8}}\,p\,(1 + 32\,p^2)\right)}$$

p = loss rate, s = packet size, t_{RTT} = round-trip time

TCP-Friendly Rate Control (TFRC)



- Sending rate as a function of RTT and loss rate

The measurement of these two parameters is critical.

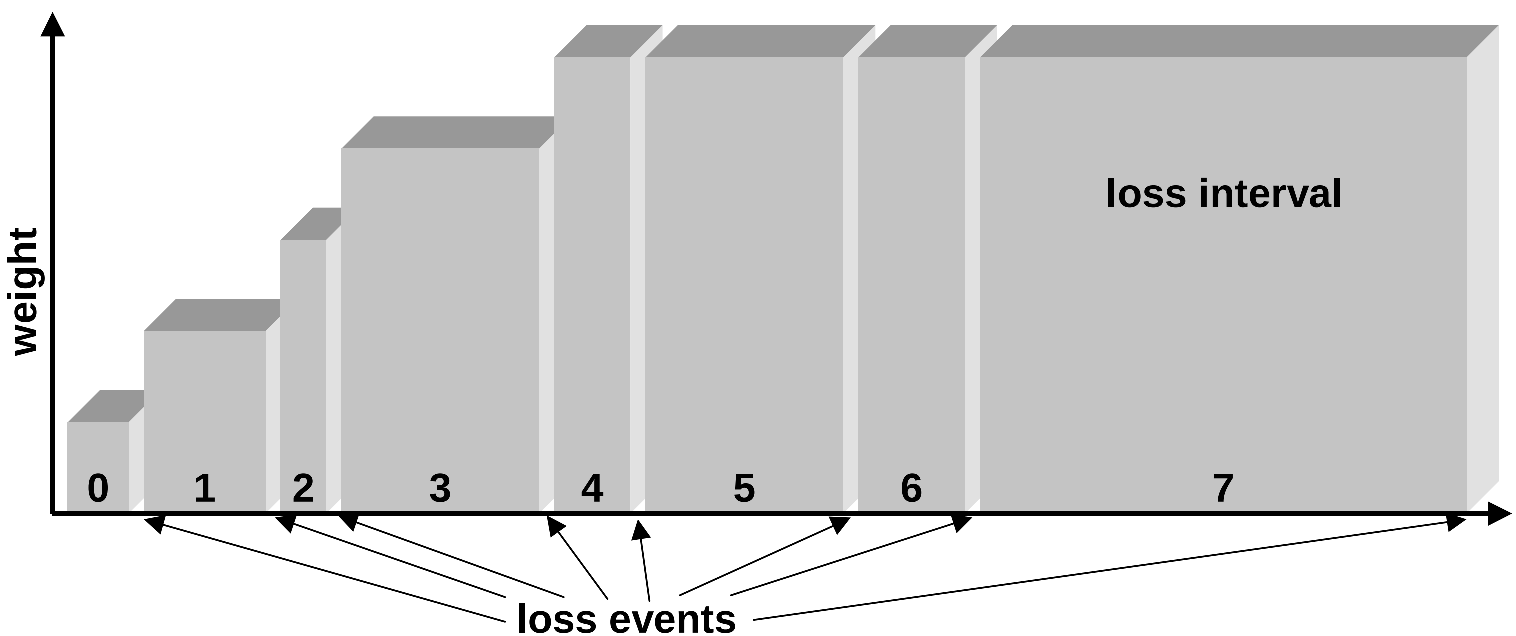
Round-Trip Time Measurements

- Sender timestamps data packets
- Receiver echoes the timestamp in the next report
- Sender calculates instantaneous RTT as the difference of current time and timestamp value

- Smoothe RTT samples using an exponentially weighted moving average

Measuring the Loss Event Rate

- Loss interval: number of packets between loss events (TCP has at most one window reduction per RTT $\longrightarrow$ loss events have to be at least one RTT apart)
- Compute weighted average of *n* loss intervals
- $p = 1/$average loss interval



Slowstart

Roughly similar to TCP slowstart:

- Double sending rate every RTT to quickly reach fair share of bandwidth
- Don’t send faster than twice the receive rate
- Quit slowstart after the first packet loss

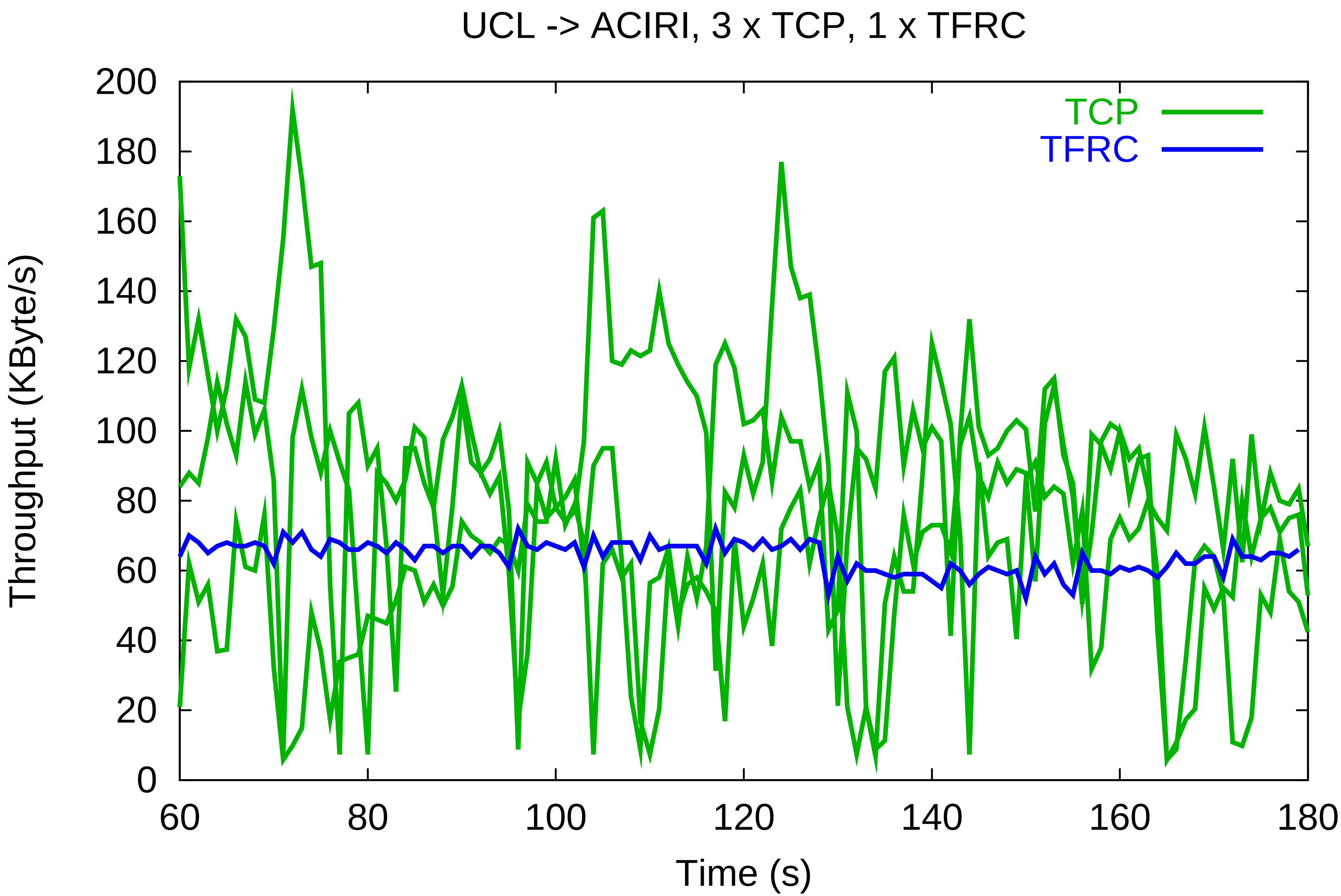
Receiver only has one loss event and cannot compute a loss event rate:

- Initialize loss history so that the current receive rate is achieved

Simulations and Experiments

- Simulations with the *ns*-2 network simulator
- Controlled experiments with Dummynet
- “Real-life” experiments in the Internet

Internet Experiments



So Far So Good ...

- Robust congestion control mechanism that works well for a number of applications

  (e.g. video streaming)

- What if an application needs to modify the packet size instead of the packet rate (particularly in combination with very small packets)?

  (e.g. audio traffic, VoIP)

Flows with Small Packets

TCP throughput scales (roughly) linear with the packet size.
Is it necessary to be that conservative?

Questions:

- What's a fair throughput / packet rate for a flow sending small packets?
- How do we achieve this fair rate?

Fairness

It’s difficult to determine a fair sending rate:

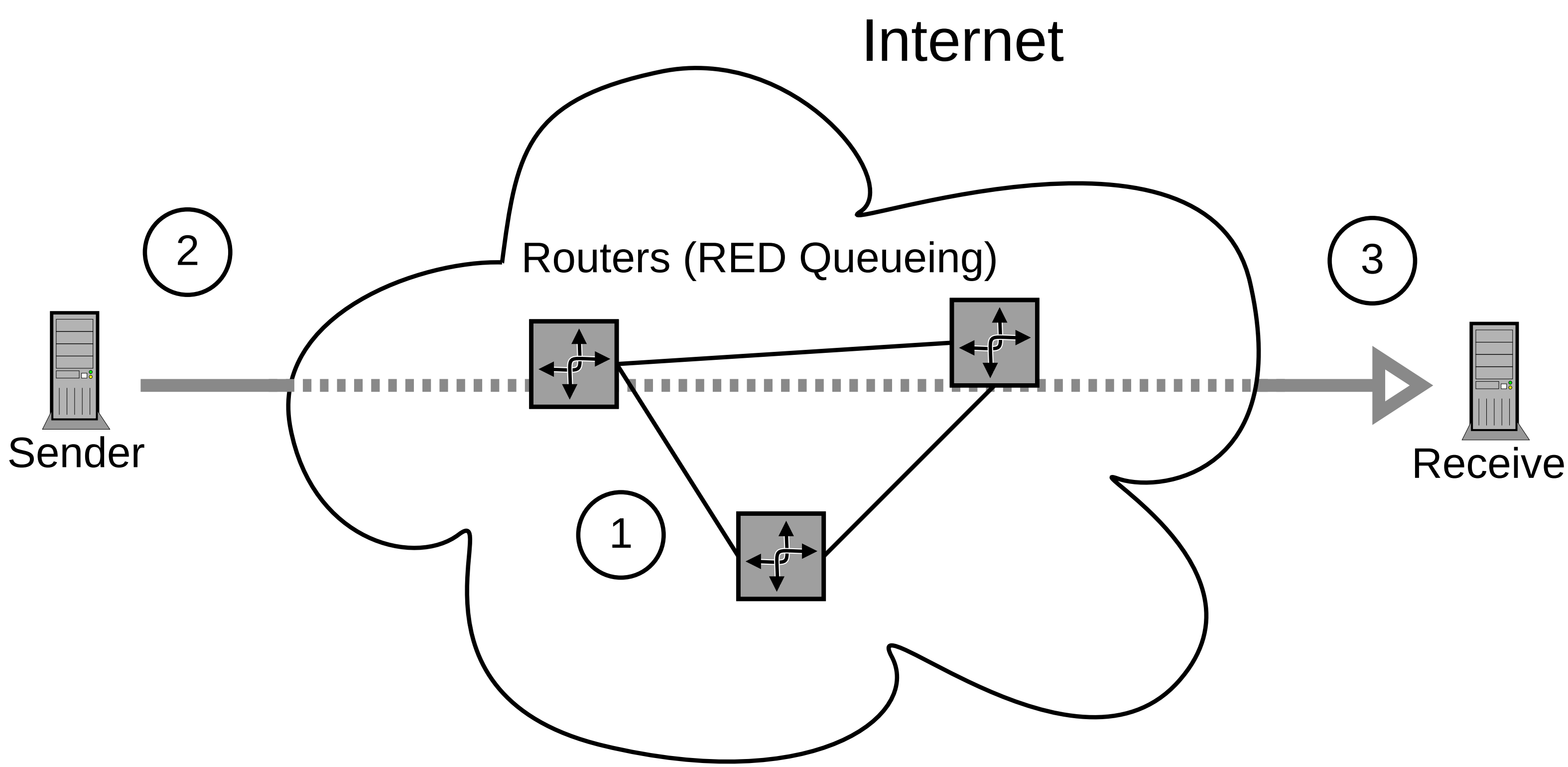
- Do packet drops depend on throughput or only on the number of packets?
- What’s the limited resource at the bottleneck (bandwidth or packet processing overhead)?
- What’s the queuing strategy used (drop-tail, RED, ...)?
- How many flows are competing against each other?

Bottleneck Measurements

- Measurement techniques to estimate bottleneck link bandwidth (e.g., packet pair)
- Reasonable results only for bandwidth limited bottlenecks (mechanism should fail with a packet rate limited bottleneck)
- $\Longrightarrow$ This can possibly be used to distinguish the two types of bottleneck

It seems that most of the bottlenecks in the Internet are bandwidth limited.

Design Space



Modifications to the network (1), the sender (2), or the receivers(3)

Adjusting the Loss Event Rate

If we know what “fair” means, adjusting the congestion control mechanism is doable:

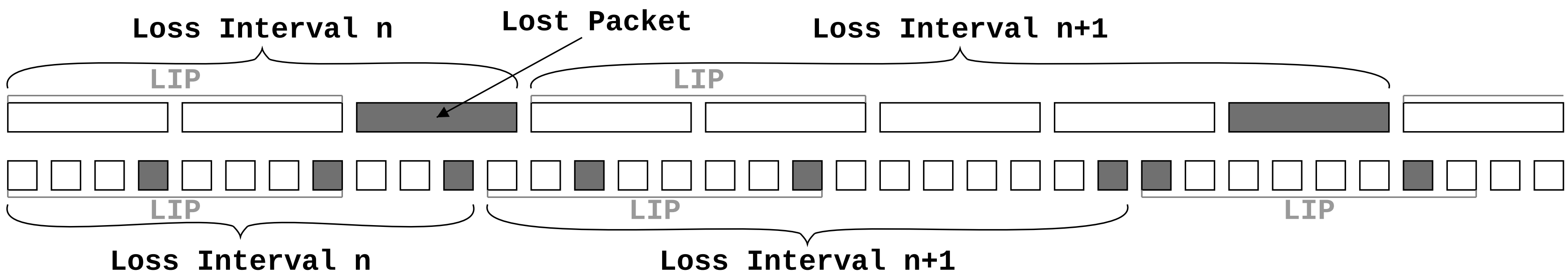
- Too many packets in the denominator of the loss event rate
- Possibly too many loss events in the numerator

$$\Downarrow$$

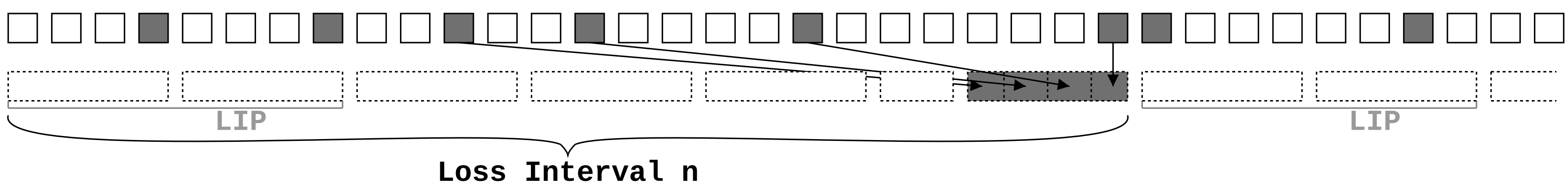
- Sample packets at the receiver at a rate that corresponds to the packet rate of a TCP flow (and ignore all other packets) or
- Aggregate packets (and loss events)

Loss Measurement Mechanisms

Unmodified:

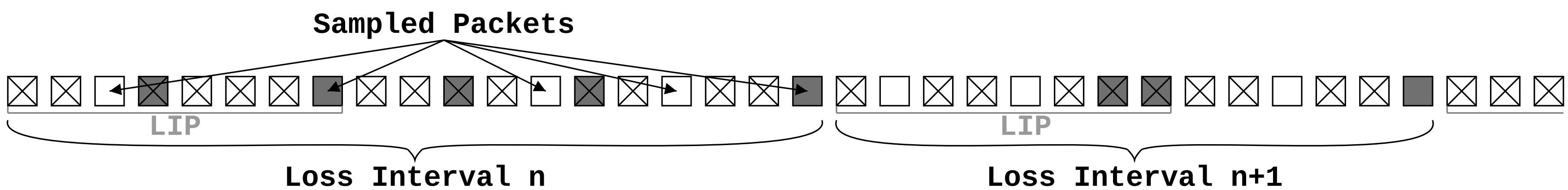


Virtual Packets:

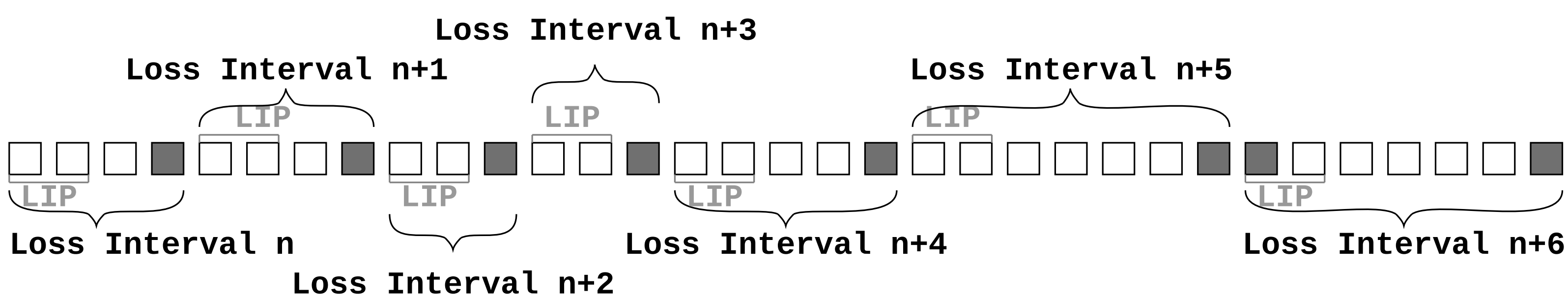


Loss Measurement Mechanisms (2)

Random Sampling:

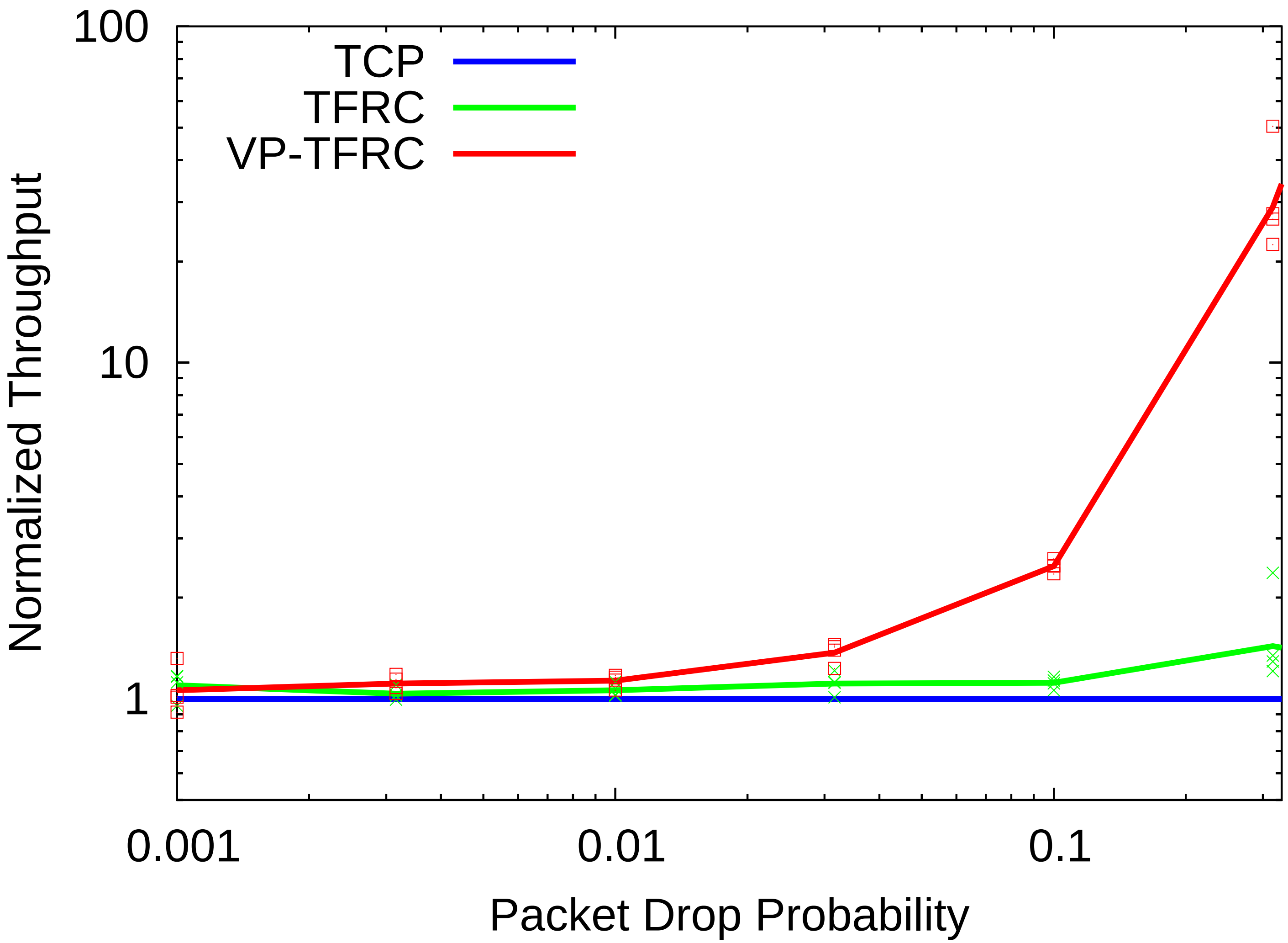


LIP Scaling:



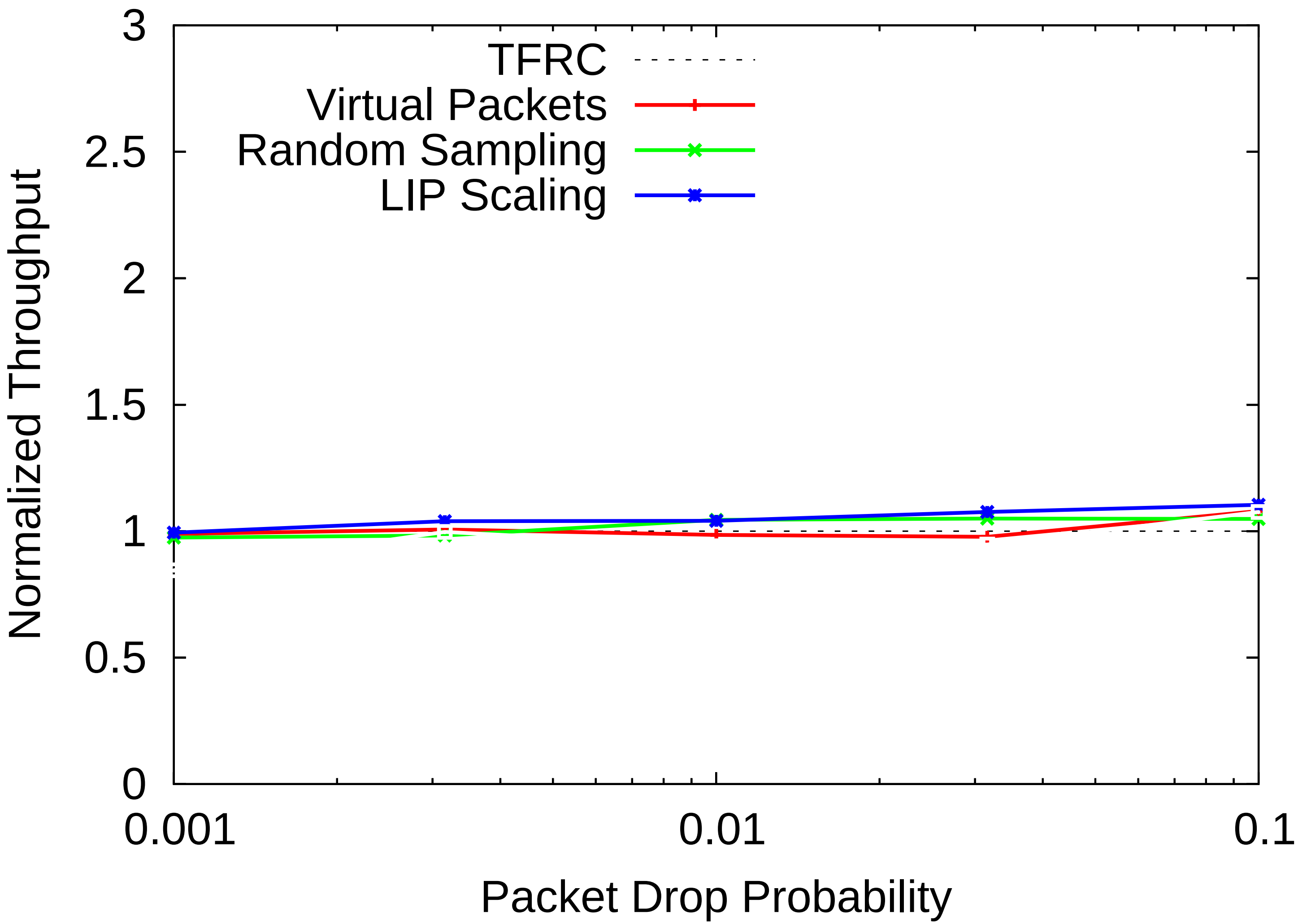
Simulation Results

Unmodified:



Simulation Results

Modified Loss Measurement Mechanisms:



Concluding Remarks

- Characteristics of unicast and multicast congestion control fairly well understood
- Large number of simulations and experiments under various network conditions
- Mechanisms well suited for various applications
- IETF drafts for TFRC as well as TFMCC
- Ongoing work on variable packet size TFRC/TFMCC

References

- Sally Floyd, Mark Handley, Jitendra Padhye, Jörg Widmer. **Equation-Based Congestion Control for Unicast Applications**. Proc. ACM SIGCOMM 2000

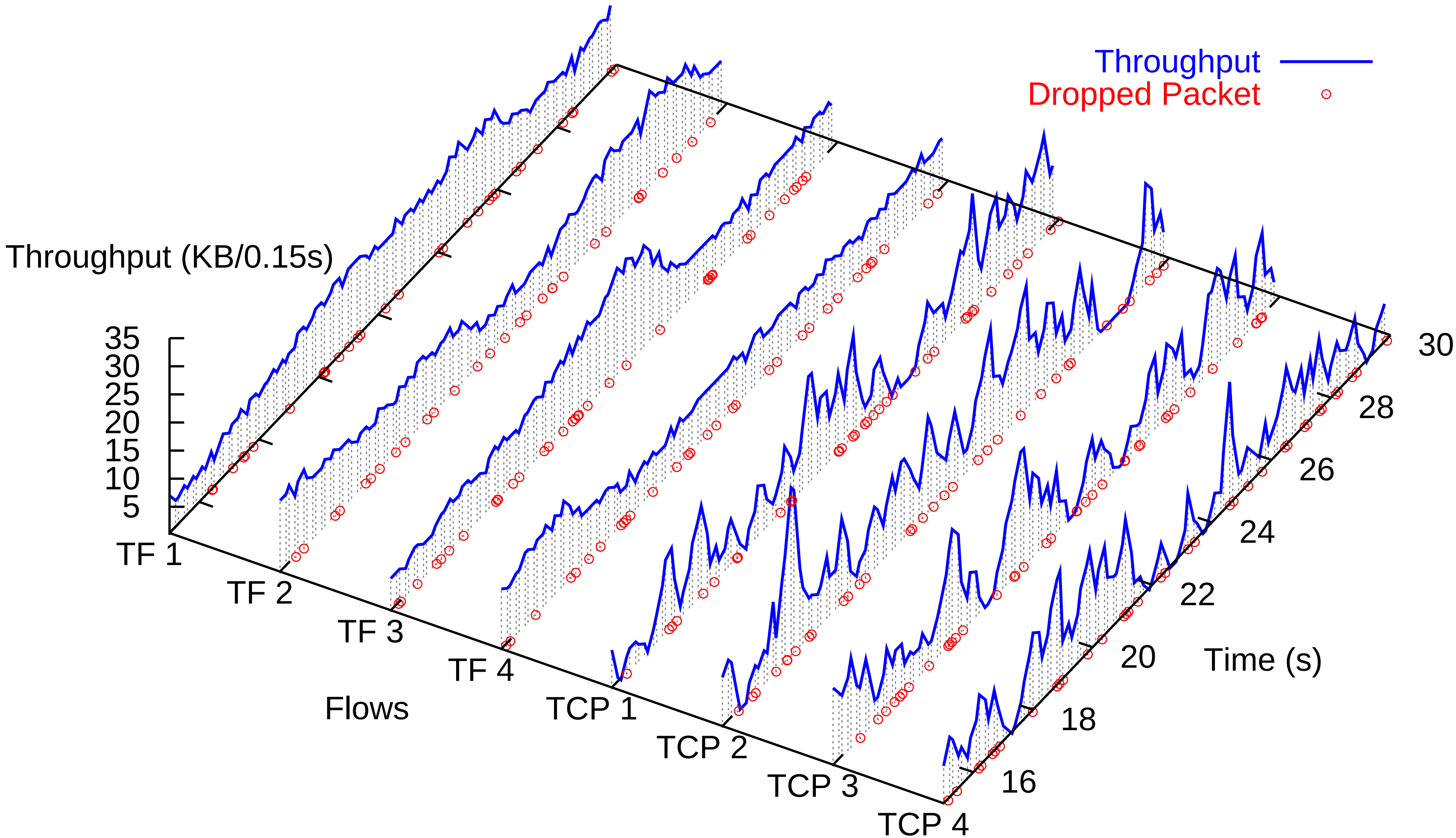
  (draft-ietf-tsvwg-tfrc-03.txt)

- Jörg Widmer, Mark Handley. **Extending Equation-based Congestion Control to Multicast Applications**. Proc. ACM SIGCOMM 2001

  (draft-ietf-rmt-bb-tfmcc-00.txt)

Additional Slides

TFRC Network Simulations



TFRC vs TCP Sack, 32 flows, 15Mb/s link, RED Queue

The Picture So Far

- Working unicast congestion control
- Stable sending rate
- High adaptive range
- Low overhead

How can this scheme be extended to multicast?

Multicast Congestion Control (TFMCC)

TCP-Friendly Rate Control extended to multicast

- Equation-based
  model TCP throughput based on RTT and loss rate
- TCP-friendliness
  no greater medium-term throughput than TCP to *any* of the receivers
- Single-rate congestion control
  adapt the rate of the sender to the slowest receiver
- No network support/overlay network necessary

TFMCC Mechanism

- All receivers measure RTT and loss rate ...
- ... and calculate a TCP-friendly rate
- *Some* receivers report their rate back to the sender who adjusts the sending rate

Challenges:

- Scalable RTT measurements to a large receiver set
  Receivers with a low rate have a higher probability of frequent RTT measurements
- Feedback mechanism preventing a feedback implosion

Adjusting the Sending Rate

Sending rate determined by receiver that is assumed to have the lowest calculated rate (current limiting recv).

Decrease:

- Adjust sending rate whenever lower rate feedback is received (maybe have minimum sending rate)

Increase:

- Only the CLR can cause a rate increase
- Additionally limit rate increase to 1 packet/max. RTT

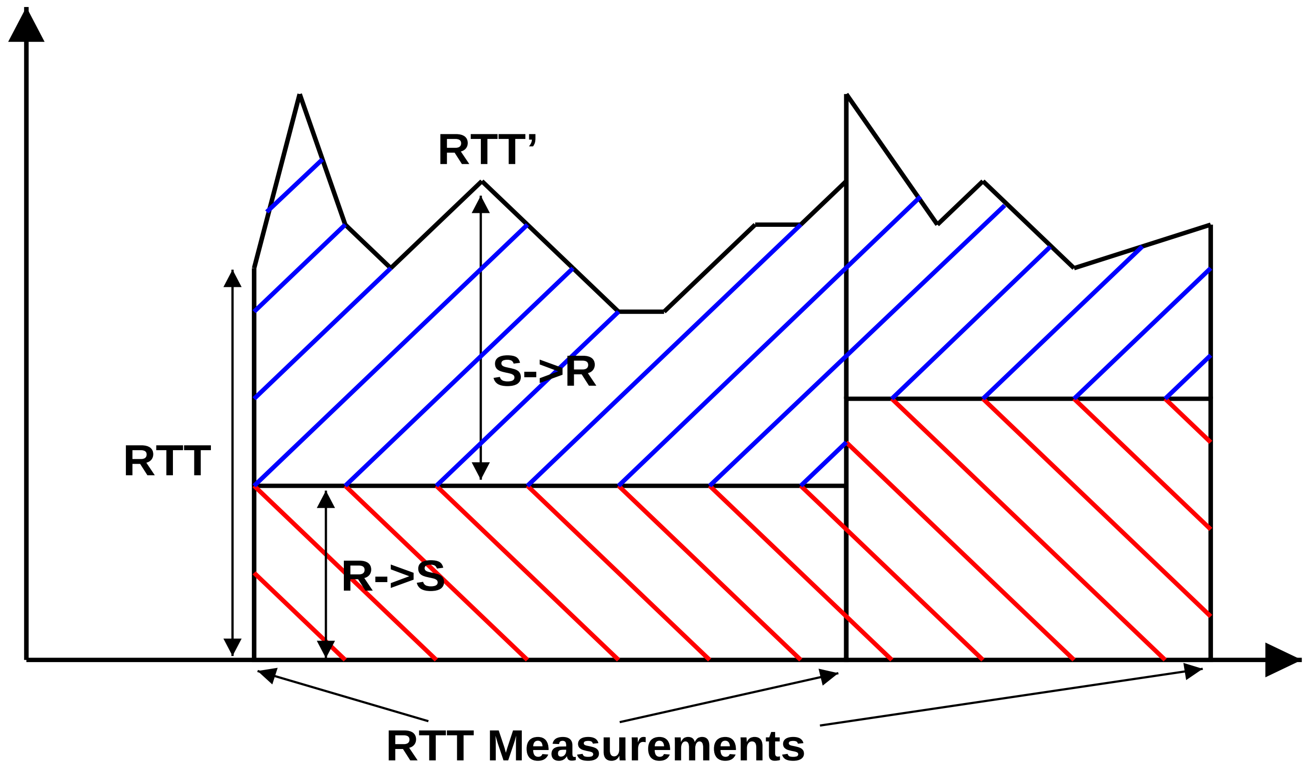
- Time out CLR if no CLR feedback received for 10 RTTs
- Halve rate if no feedback is received at all (for 10 RTTs)

Multicast RTT Measurements

- Well-known RTT measurement mechanism of echoing timestamps
- Priority list of which timestamps to echo in data packets
  - Try to measure RTT to each receiver at least once
  - Receivers with a low calculated rate measure their RTT more frequently
- Continuously update RTT estimate using one-way delay measurements (non-CLR receivers)
- Additional smoothing
- Assume a high initial RTT until the first measurement is made

One-Way RTT Measurements

Infrequent RTT measurements for non-CLR receivers
$\Longrightarrow$ adjust RTT using one-way delay measurements



- RTT' can then be used to detect changes in the RTT

Sender-side RTT Measurements

Timestamps also allow the sender to measure the instantaneous RTT to a receiver.

- Receivers can report a calculated rate without knowing their RTT
- Initial RTT (say 500ms) used instead of real RTT
- Sender adjusts the reported rate to reflect the instantaneous RTT (simple multiplication)

Determining the Max. RTT

Sender-side RTT measurements are also used to determine the maximum RTT. (Receivers don't need to include their RTT in the reports.)

- If instantaneous RTT > max. RTT
  max. RTT = instantaneous RTT
- If no feedback with instantaneous RTT > max. RTT
  max. RTT = max. RTT * 0.95

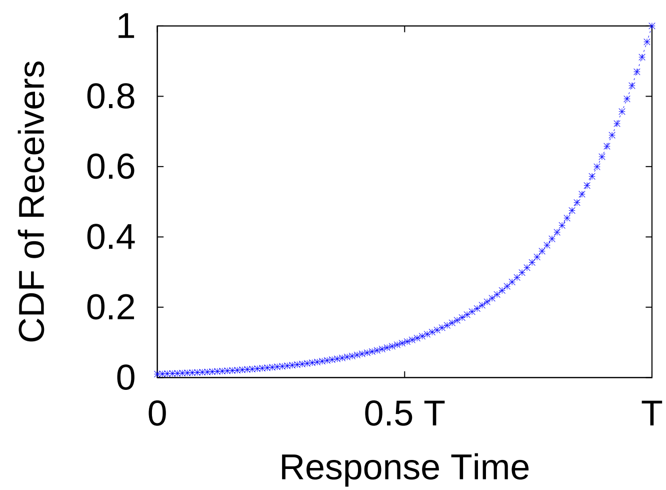
Max. RTT measurements specified differently in the NORM BB, but both mechanisms should work fine for TFMCC.

Feedback Control

Feeedback control using exponentially distributed random timers:

- Receivers set timer at the beginning of a feedback round

$$t = \max\left(T(1 + \log_N x), 0\right)$$



T = feedback delay, N = number of receivers, x = random variable

- Feedback sent after timer expires, if not cancelled
- Successive feedback rounds of duration T

Feedback Suppression

Feedback suppression controlled by *suppression rate* in data packet headers

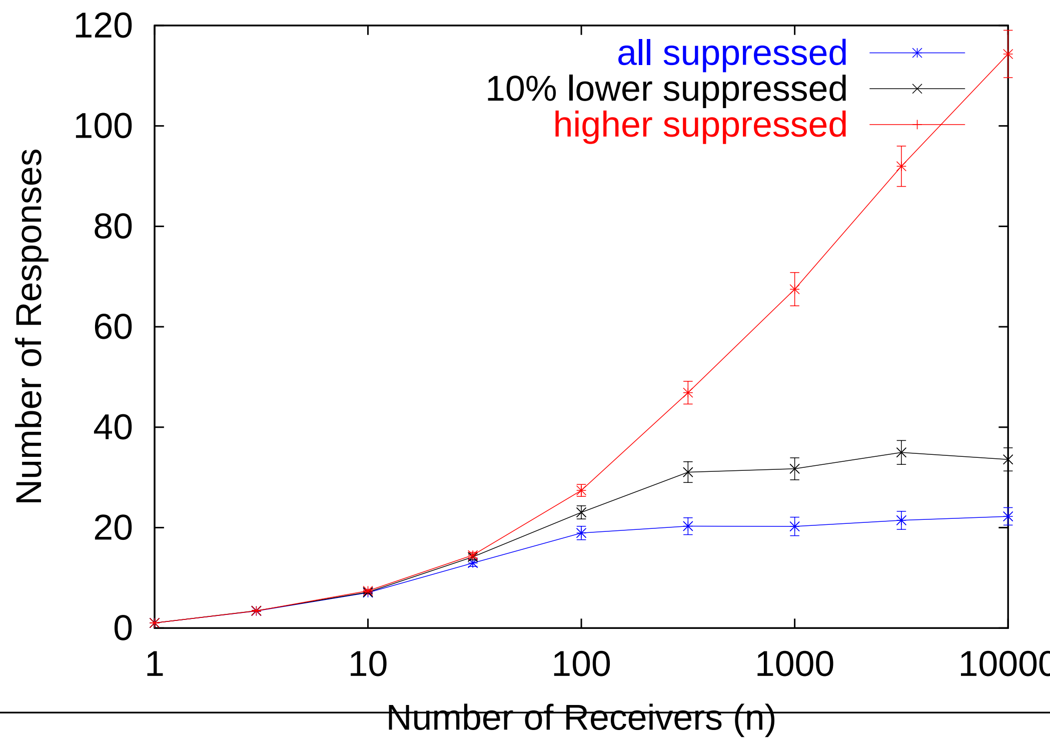
- Suppression rate = ∞ at start of a feedback round
- Supression rate decreased whenever feedback with lower calculated rate arrives at the sender
- Receivers with a calculated rate higher than the suppression rate have to cancel their feedback

Timely sending of data packets critical for feedback suppression

Feedback Suppression

Options what feedback to cancel:

- Cancel timer if any feedback was received
- Cancel timer if “better” feedback was received
- Cancel timer if $T_{fb} - T_{TCP} < \theta\, T_{fb}$



Feedback Bias

Bias feedback timers such that low rate feedback is sent earlier

$$t = \max(T(1 + \log_N x), 0)$$

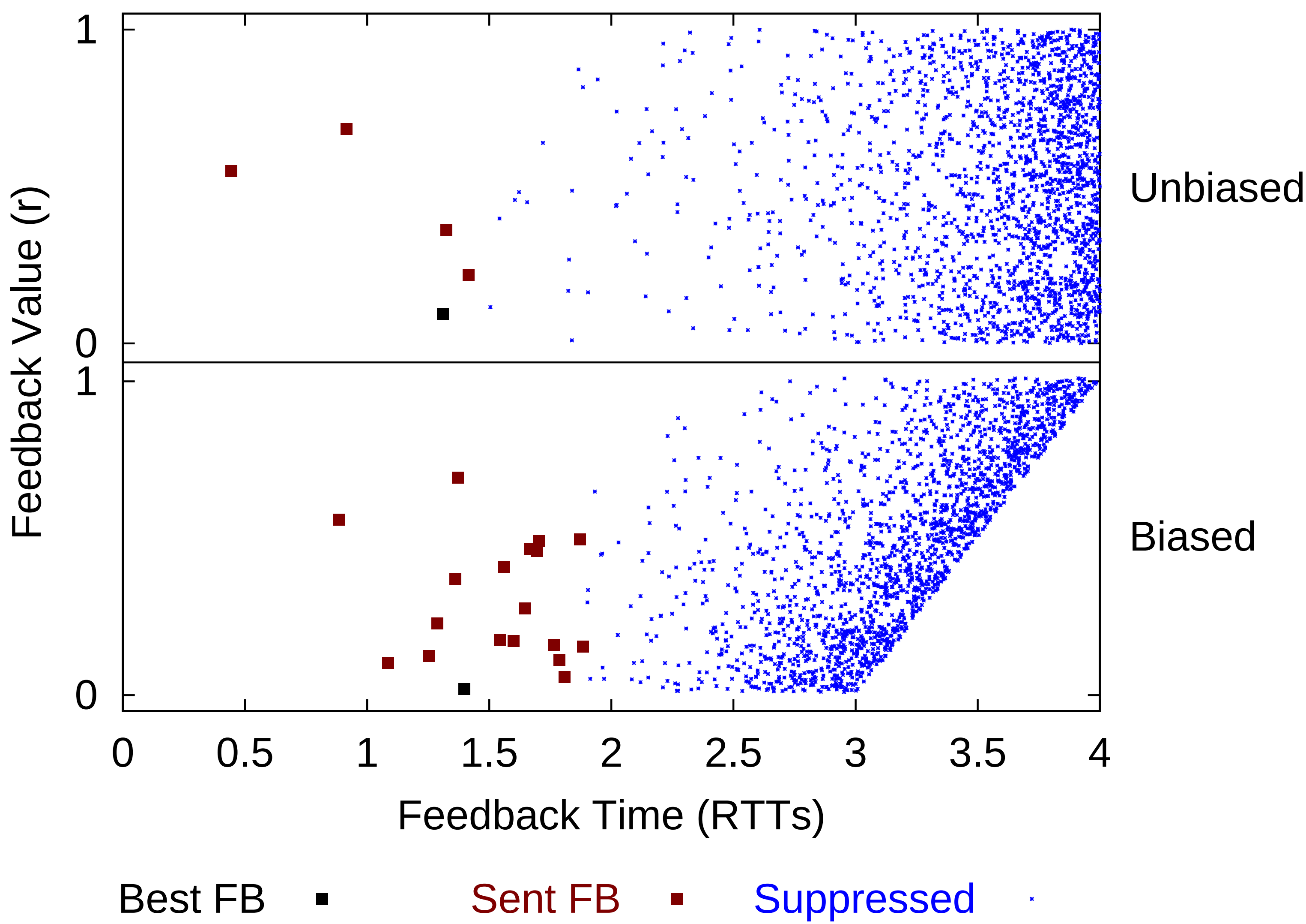
$$\Downarrow$$

$$t = \gamma \max(T(1 + \log_N x), 0) + (1 - \gamma)Tr$$

where

- r is the calculated rate relative to the CLR's rate
- γ is the fraction of T now used for suppression

Feedback Bias



Scalability

- Simulations with up to 1000 receivers
- Feedback mechanisms scales to 1,000,000s
- ... but RTT measurements and receceiver heterogeneity will limit useful scenarios to maybe 10,000

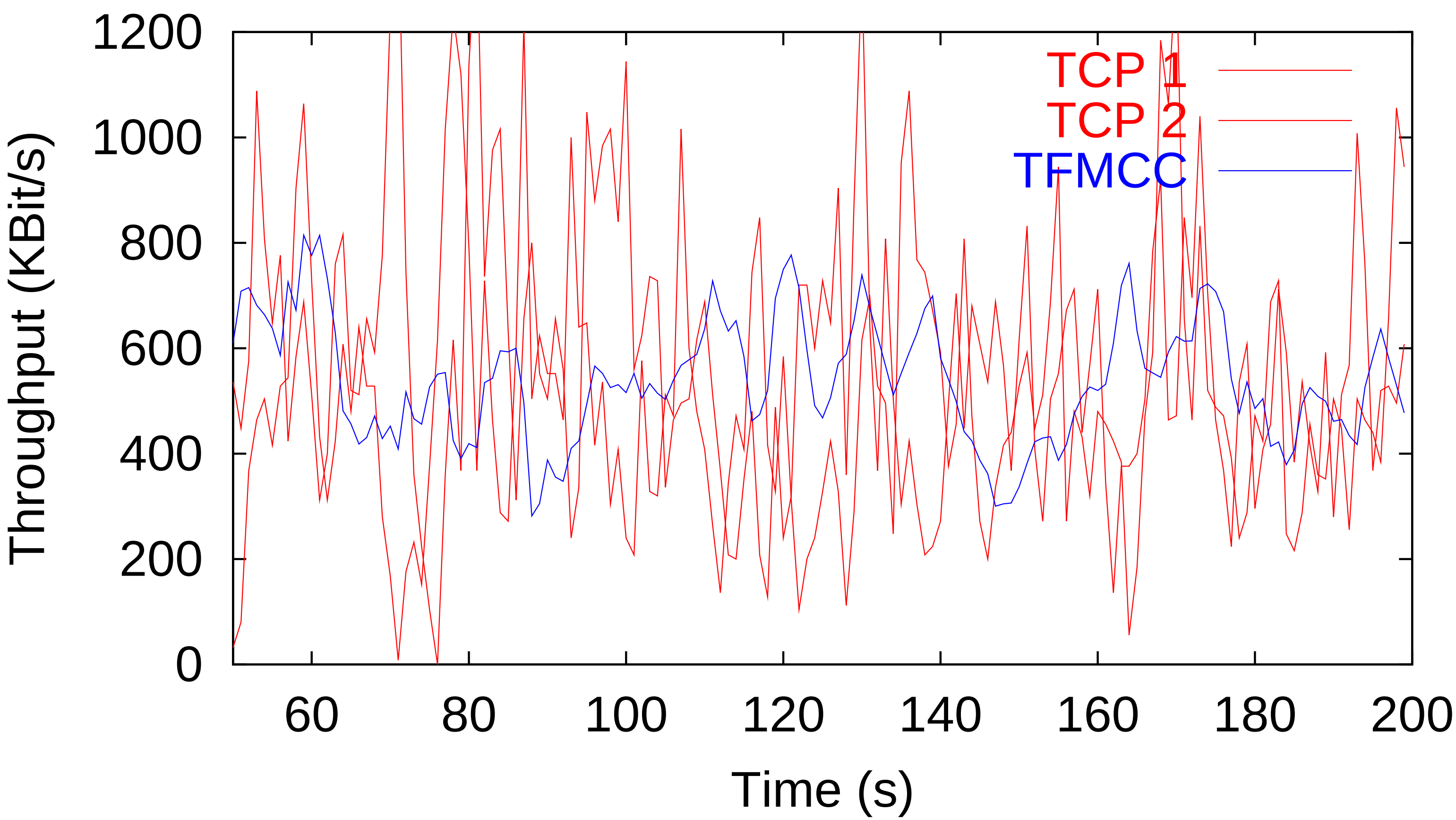
Stochastic variations in the receiver’s loss estimates degrade throughput when the size of the receiver set grows.

Multicast Simulations

Some examples of TFMCC simulations

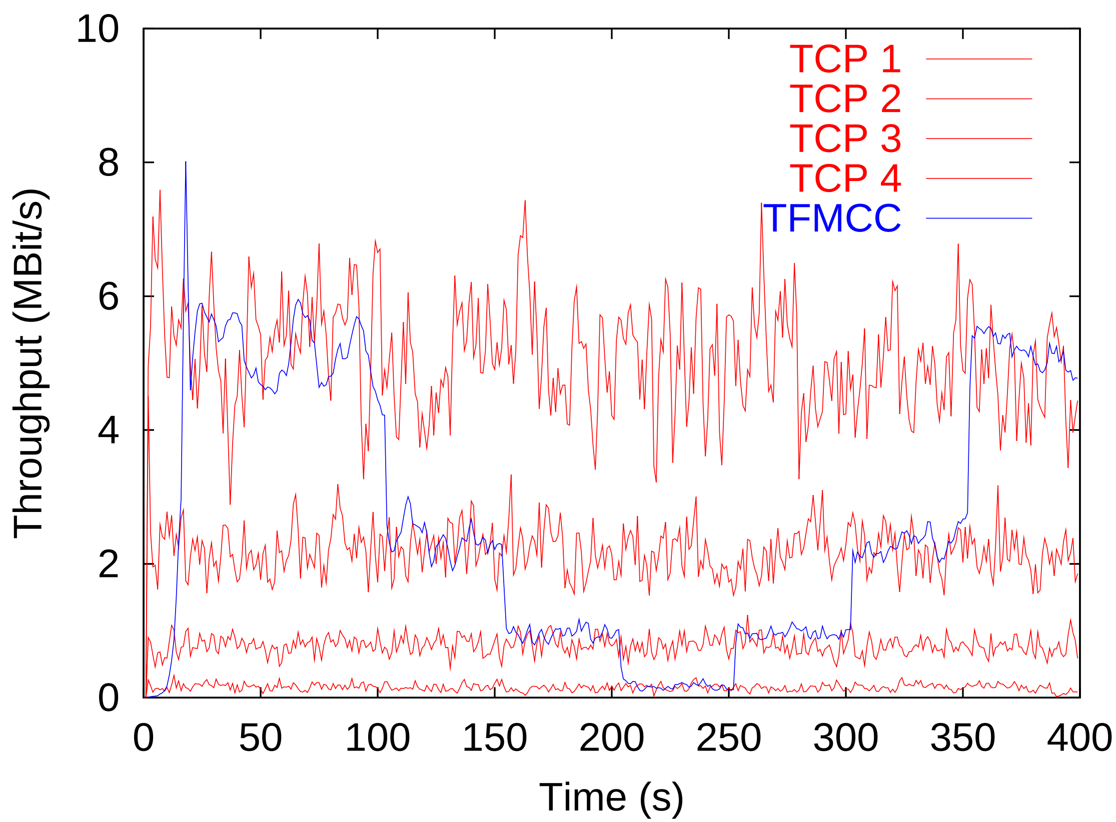
Fairness

One TFMCC flow and 15 TCP flows over a single 8 MBit/s bottleneck with 60ms RTT



Responsiveness

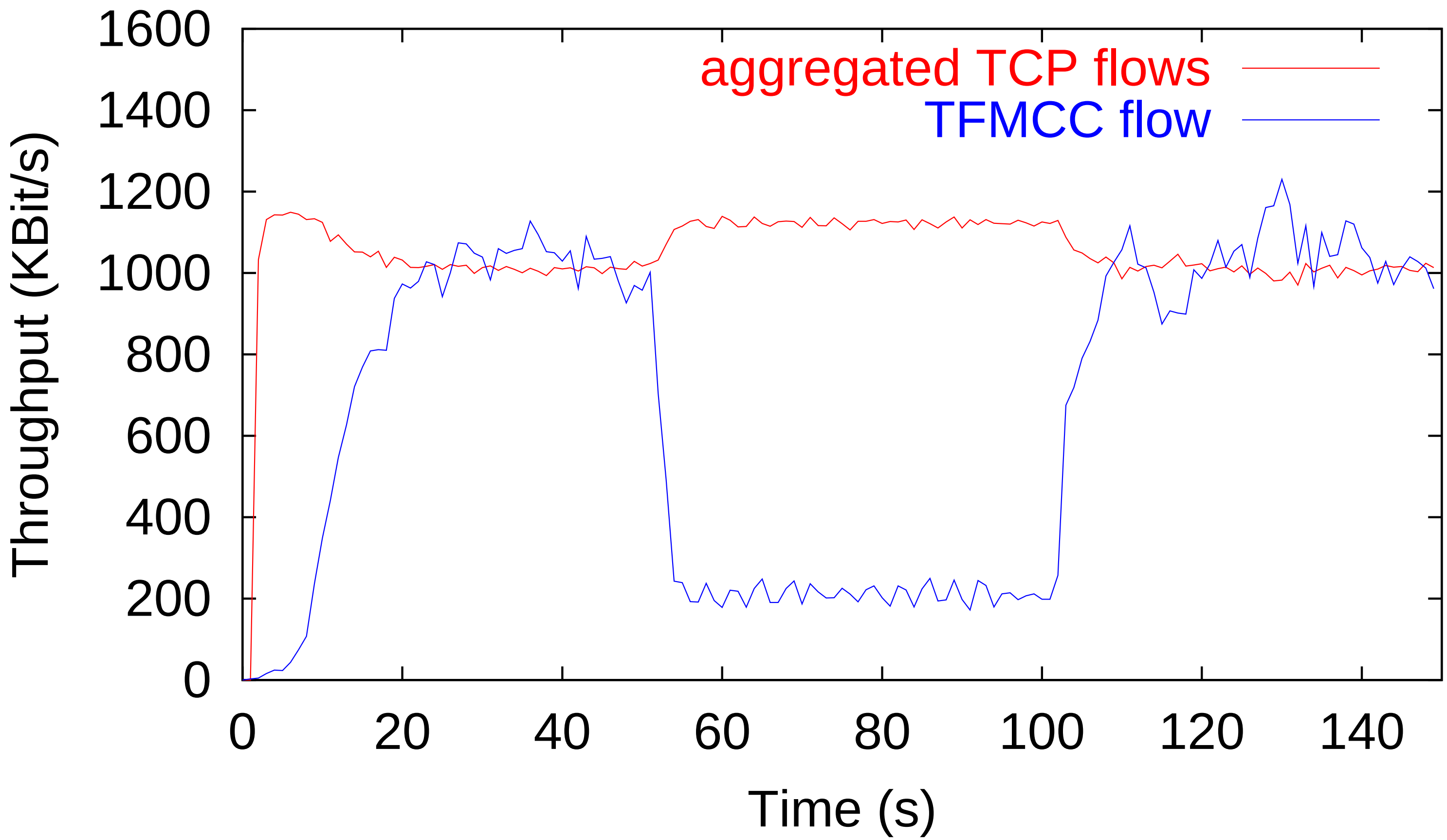
Responsiveness to changes in the loss rate
(60ms RTT and loss rates of 0.1%, 0.5%, 2.5%, and 12.5%)



Correct CLR chosen after ca. 500ms

Late-Join of Low-Rate Receiver

- TFMCC competing with 7 TCPs on 8MBit/s link
- TFMCC receiver 200KBit/s link joins for 50 seconds



RTT Responsiveness

Responsiveness to changes in the RTT (worst case analysis)

- How long does it take to find a single high RTT receiver among a large number of low RTT receivers?