Equation-based TCP-friendly Congestion Control

Mark Handley and Sally Floyd

Reliable Multicast Research Group

November 1999

Outline of presentation:

- Brief review of ”Strawman Specification for TCP Friendly (Reliable) Multicast Congestion Control (TFMCC)”

- New work on unicast equation-based congestion control.

- Implications for multicast equation-based congestion control.

Brief review of "Strawman Specification":

- Handley, Floyd, and Whetten, "Strawman Specification for TCP Friendly (Reliable) Multicast Congestion Control (TFMCC)", June 1999,
URL "http://www.aciri.org/mjh/rmcc.ps".

- The response function:

$$T = \frac{s}{RTT\sqrt{\frac{2p}{3}} + t_0(3\sqrt{\frac{3p}{8}})p(1+32p^2)} \qquad (1)$$

T: sending rate in bytes/sec
s: packet size in bytes
p: packet drop rate
t_0: retransmission timeout value

– J. Padhye et al., Modeling TCP Throughput: A Simple Model and its Empirical Validation, SIGCOMM 98.

Strawman: Calculating loss fraction at the receiver:

- The receiver calculates the expected packet drop rate p for the current sending rate, and measures the number of loss events over k/p arrivals, for $k = 4$.

- The measurement interval should be extended to include at least four loss events.

Strawman: Calculating the RTT at the receiver:

- Different mechanisms for NACK-based and Hierarchical ACK feedback mechanisms.

Estimating the retransmit timeout value t_0

- Set $t_0 = Max[20ms, 4t_{RTT}]$

Strawman: Increase mechanism:

- Increase at most up to the rate specified by the equation?
  – Increase limited to one packet per RTT, or limited by a fraction of the current rate?

Strawman: Decrease mechanism:

- Decrease down to the rate specified by the equation.

Strawman: Slow-start?

Strawman: Behavior after idle or application-limited periods?

Unicast equation-based congestion control:

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

- Implementation in NS:
  - NS Simulations of TCP-Friendly Congestion Control,
  - URL ”http://www.aciri.org/floyd/friendly.html”.

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.

  – Most of the weight is on the most recent four lost intervals, with slowly decaying weight on older loss intervals.

  – (The average weighs the K+1, K+2, and K+3-rd loss intervals, for $K = 4$, with reduced weights.)

  – A loss interval is a sending period ending in a loss event (e.g., one or more packet drops in a window of data); or the most recent interval without a loss, if longer than the average loss interval.

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

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 equation of the response function is based on the model of a fixed roundtrip time:
  - In environments with high levels of statistical multiplexing, the delay and packet drop rate is largely independent of the flow's sending rate.
  - This is not true with small-scale statistical multiplexing.

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 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 actually 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;

Unicast: slow-start:

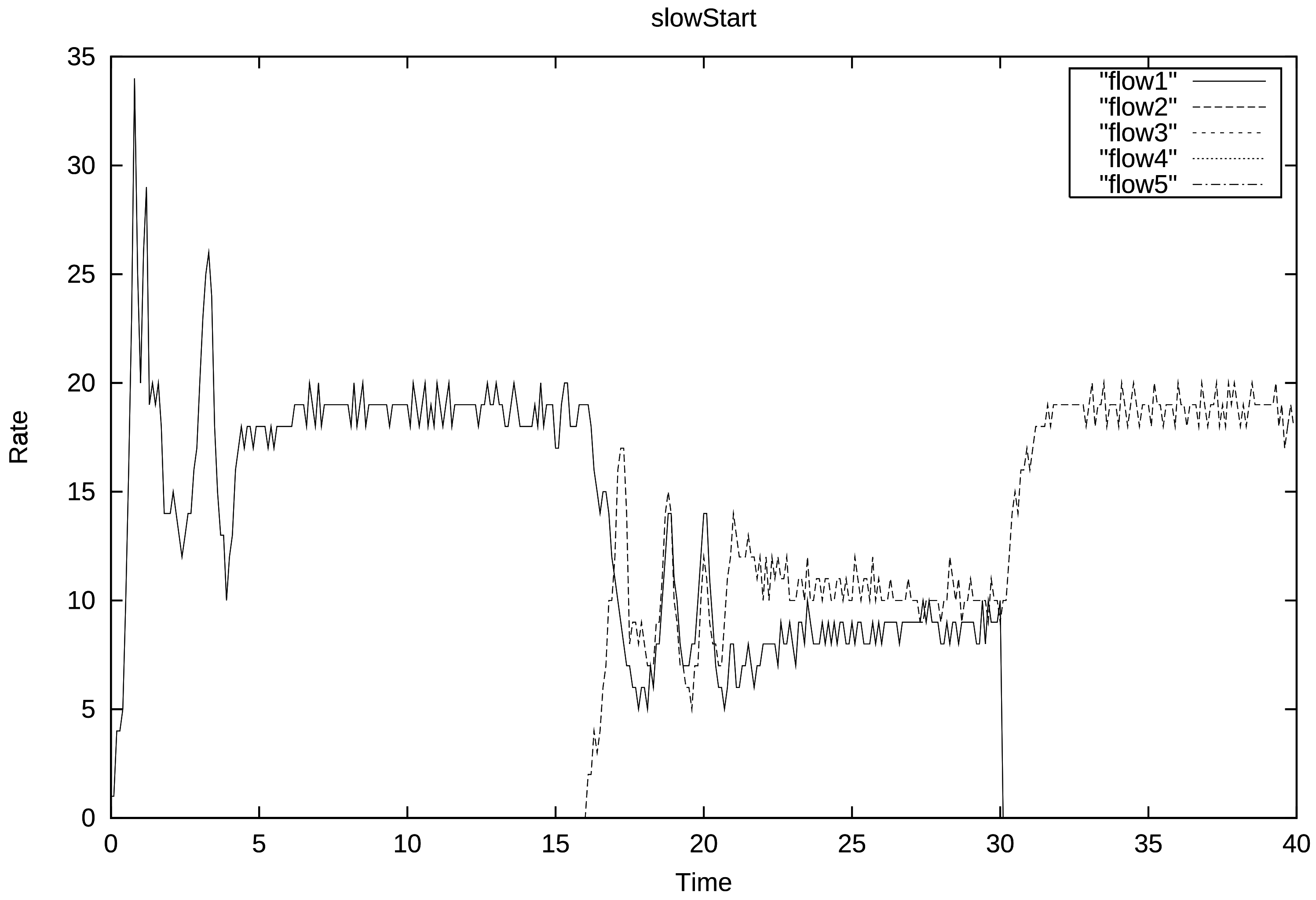
- Increase the sending rate by a factor ssmult (e.g., 2) each RTT.
  - Rate increases are "smoothed out" over a RTT.
  - Upper bound on sending rate:
    Twice the receiver's reported receive rate.

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

Unicast: Dealing with a changing RTT:

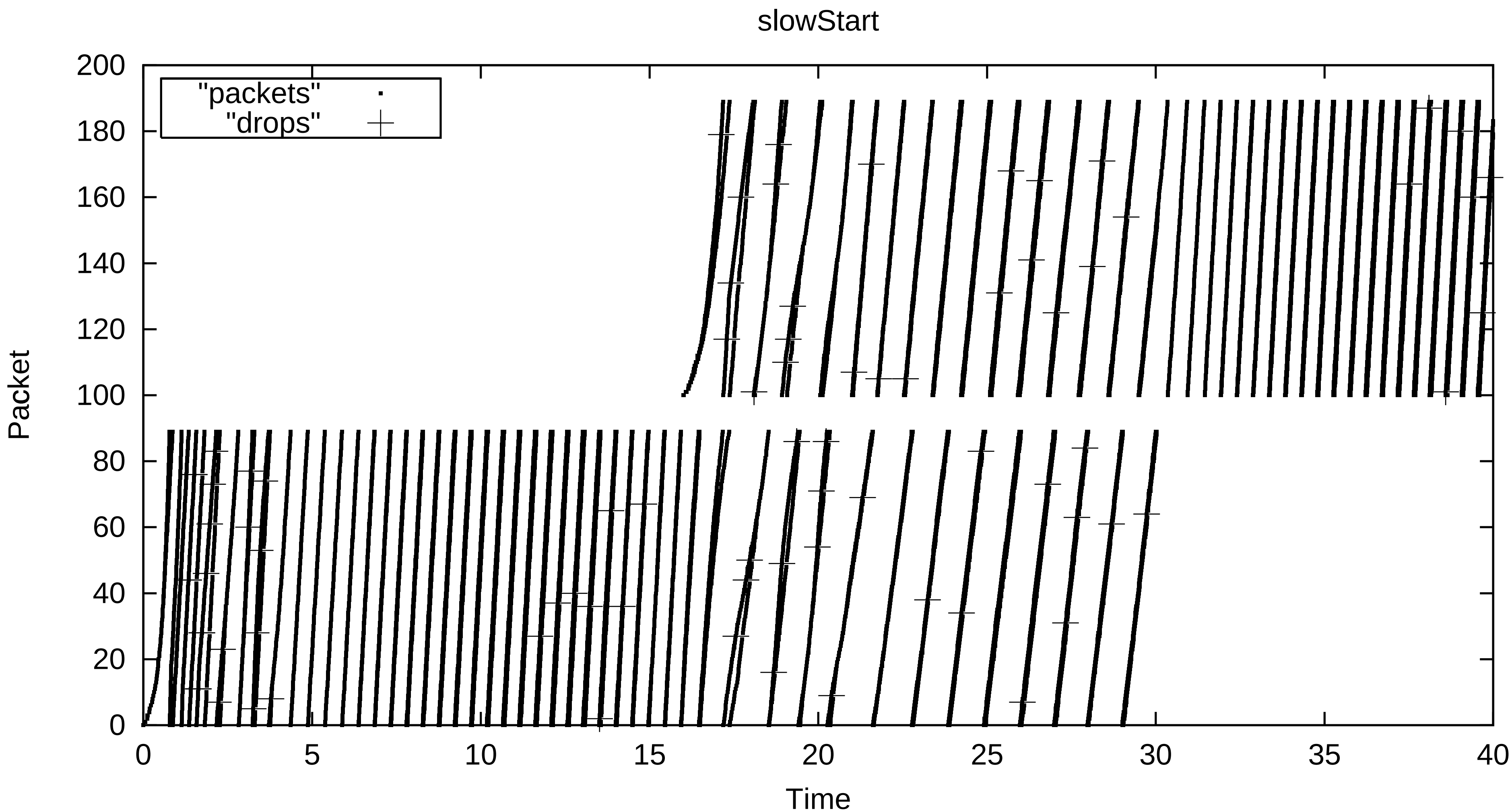
- Proposal: If the RTT in increasing for four RTTs, and the sending rate has also been increasing over those four RTTs, then stop increasing the sending rate.

Unicast: The validation test in NS:



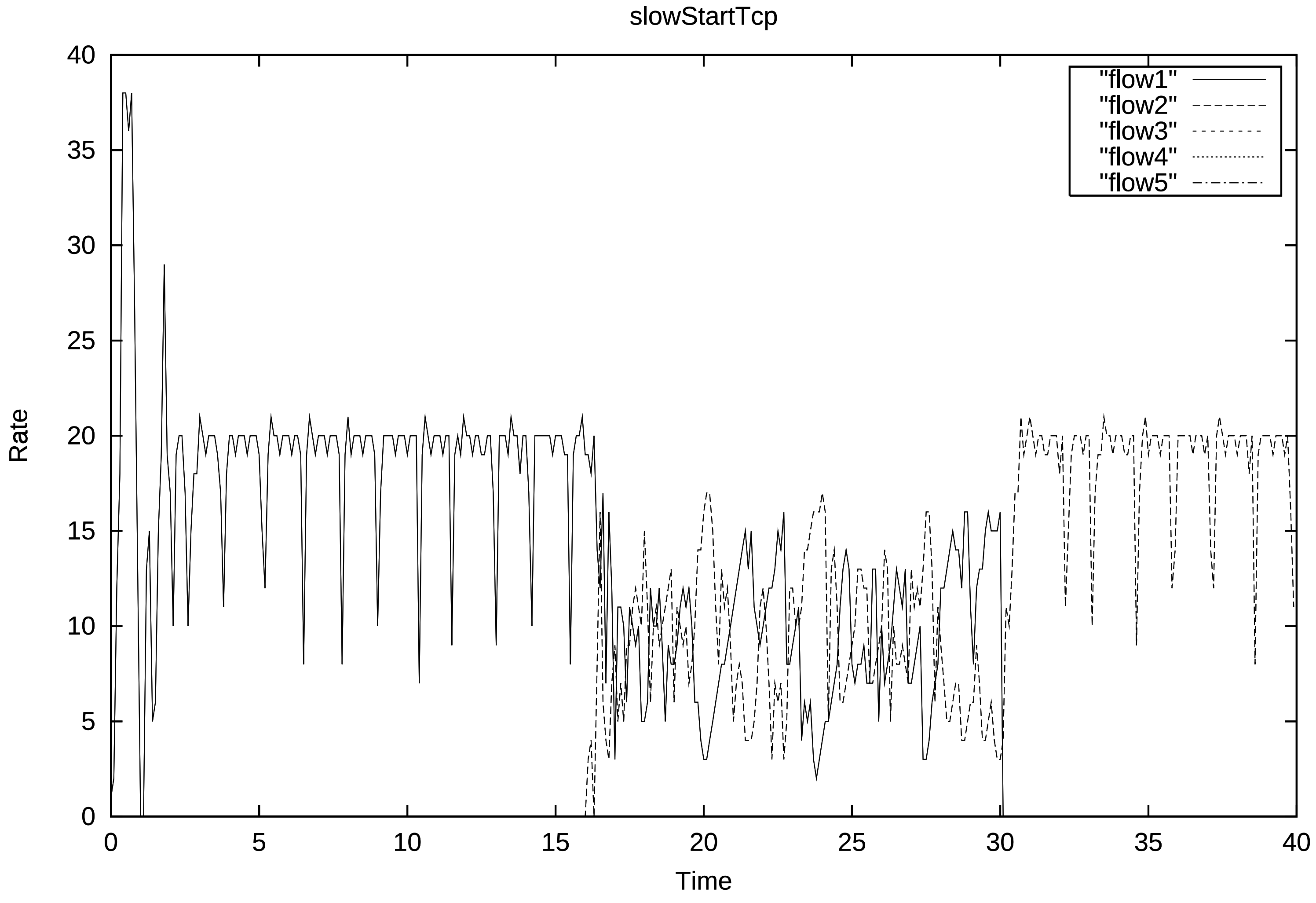
- Two TRFC (TCP-friendly rate control) connections.

Unicast: The validation test in NS, cont.:



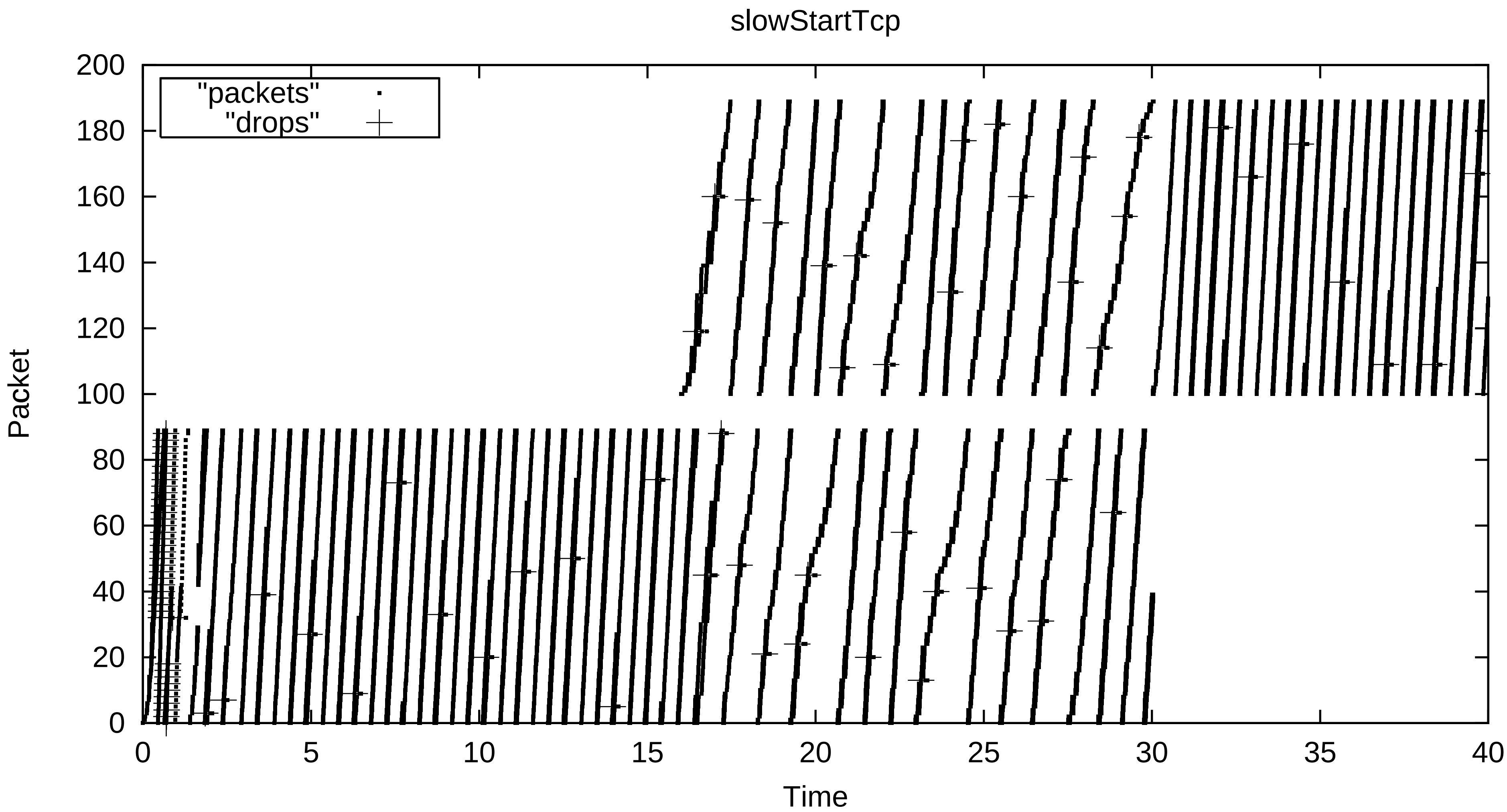
- Two TFRC connections.

Unicast: The validation test in NS:



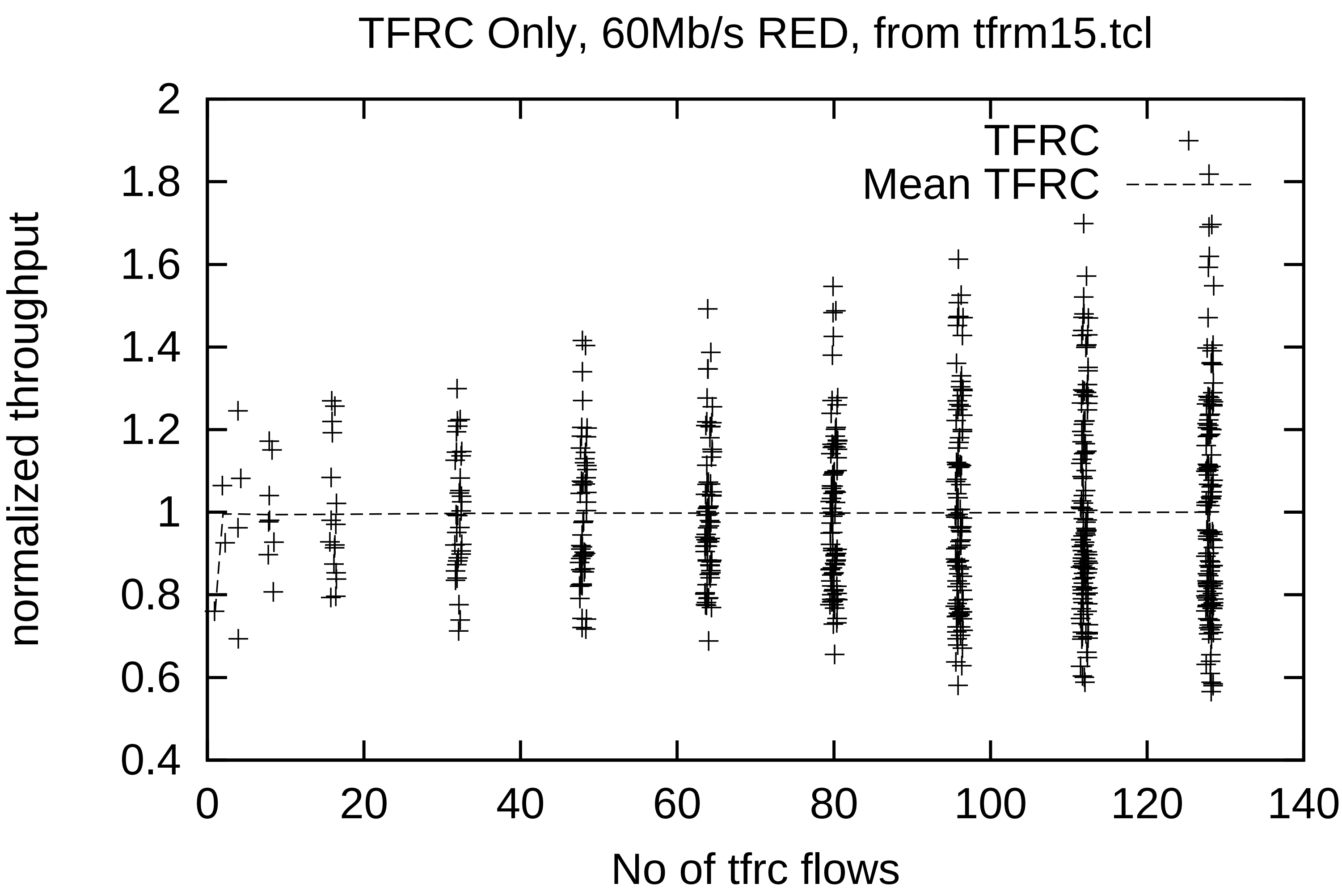
- Two TCP connections.

Unicast: The validation test in NS, cont.:



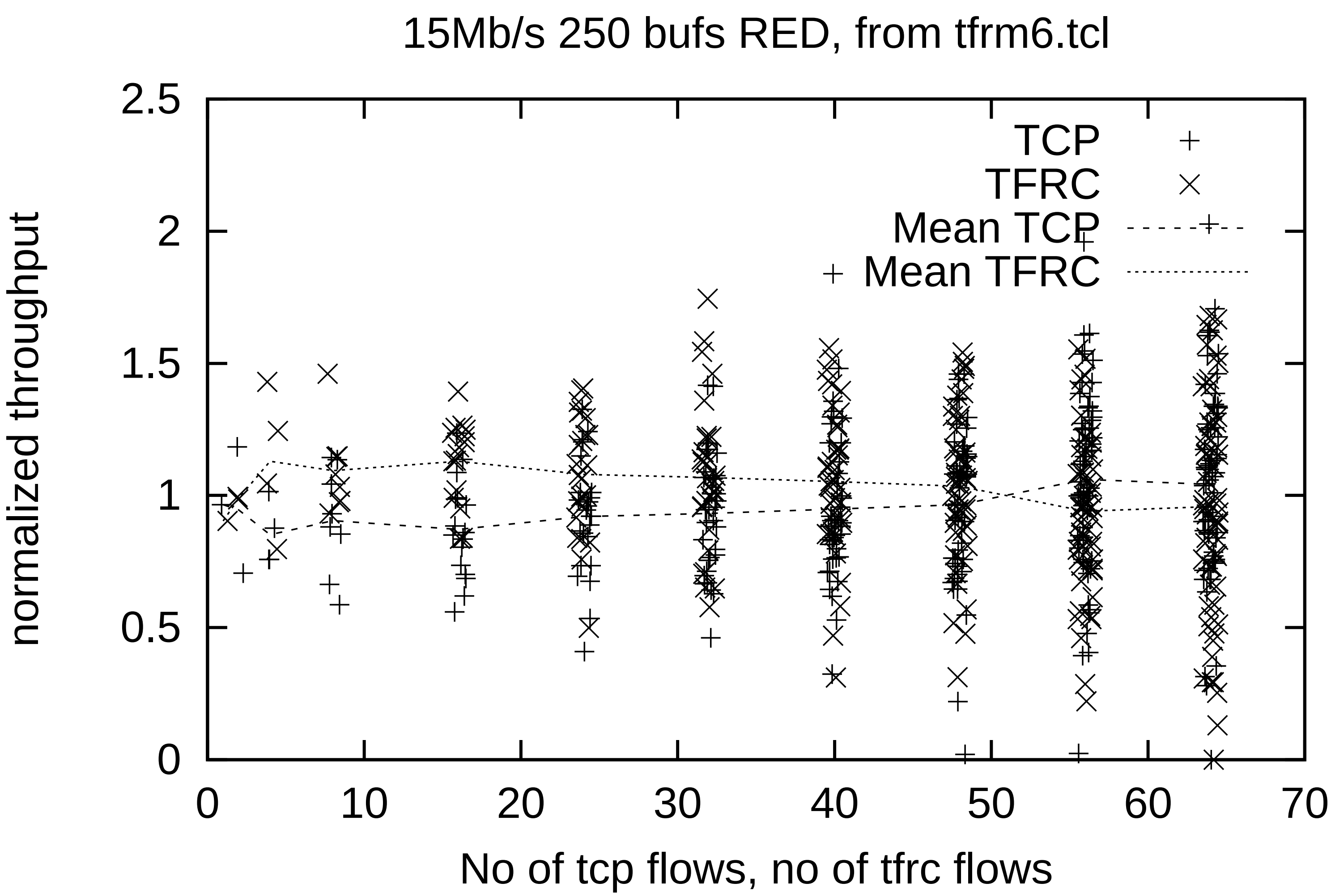
- Two TCP connections.

Unicast: Simulations exploring oscillations:



- More analysis would be useful...

Unicast: Simulations exploring fairness with TCP:

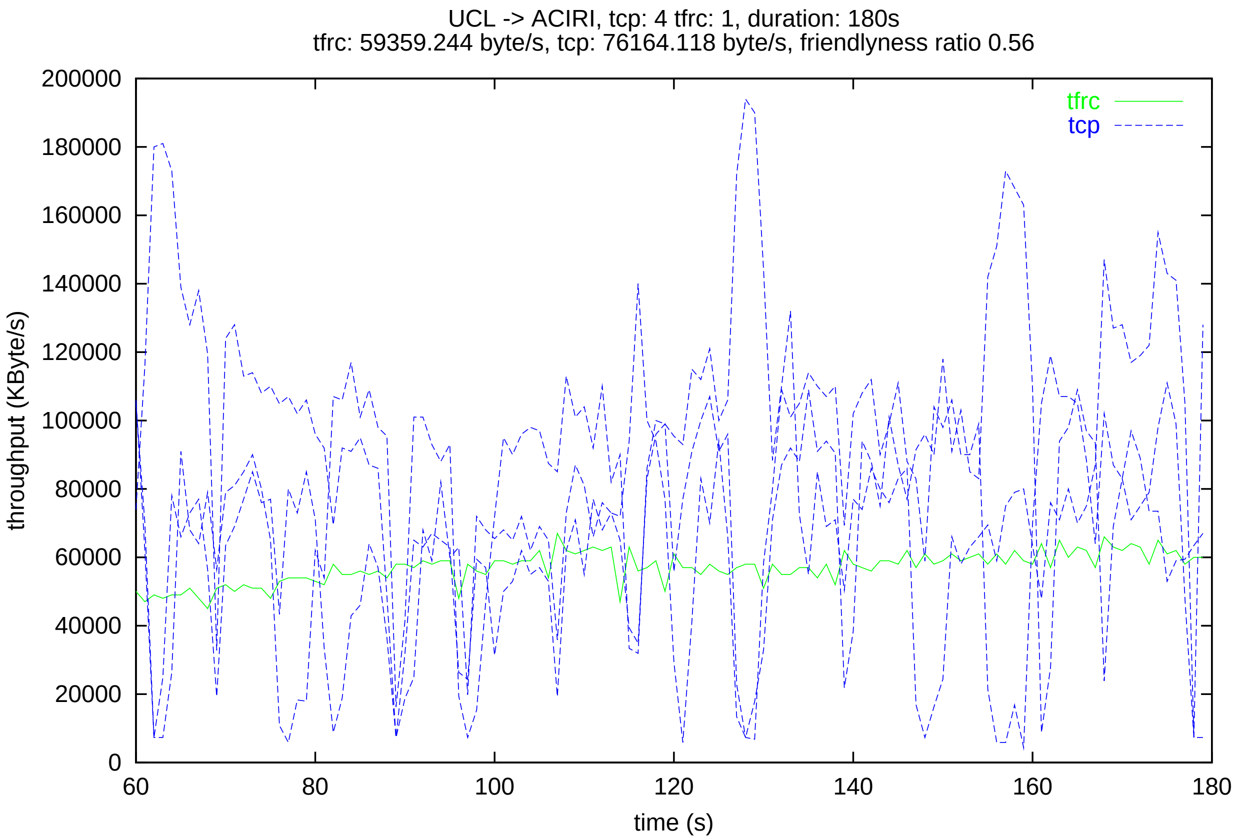


- Simulations with a range of bandwidths, packet sizes, etc..

Unicast: Simulations about delay in making use of available bandwidth:

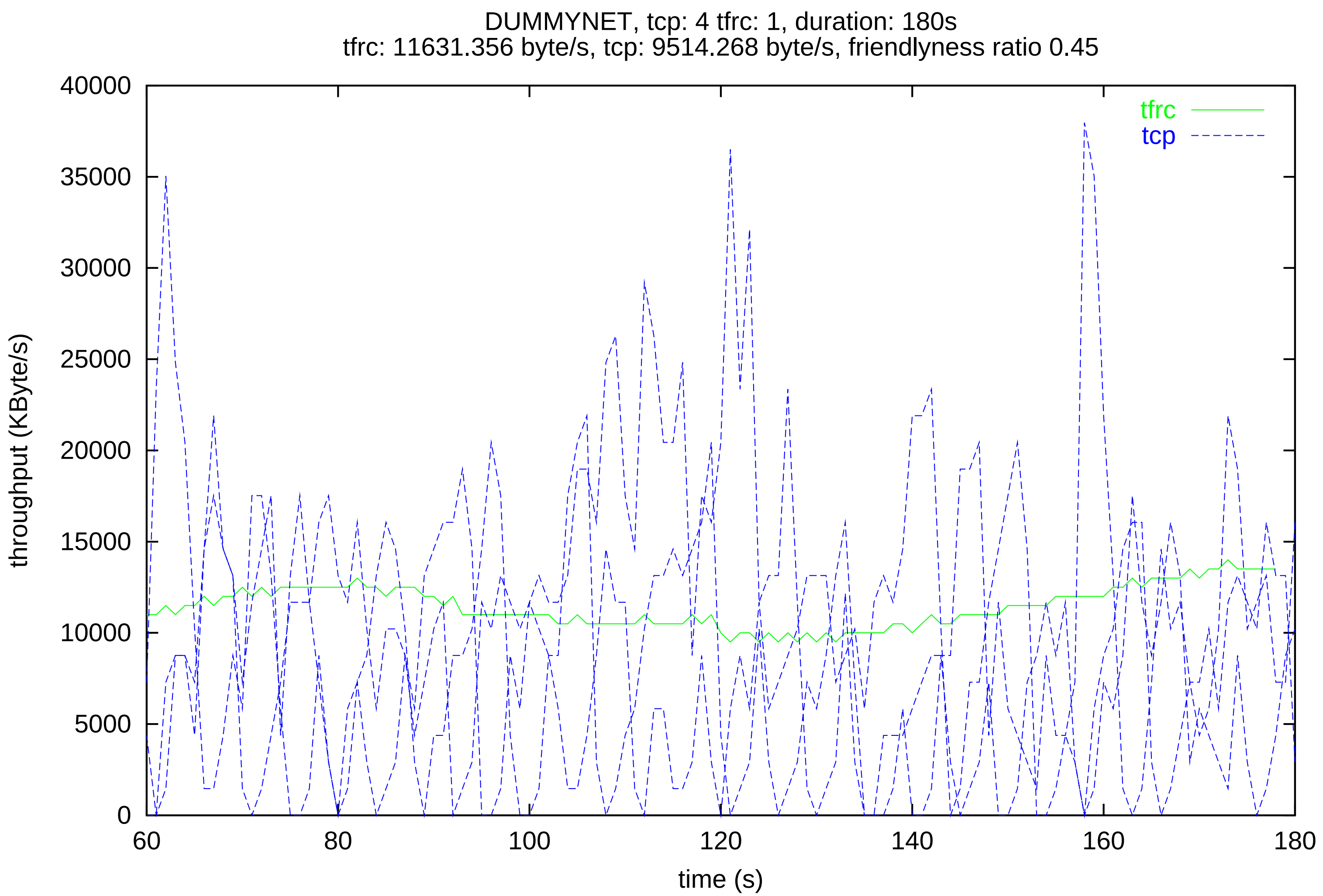
Unicast: Simulations of the autocorrelation function:

Unicast experiments: London to Berkeley

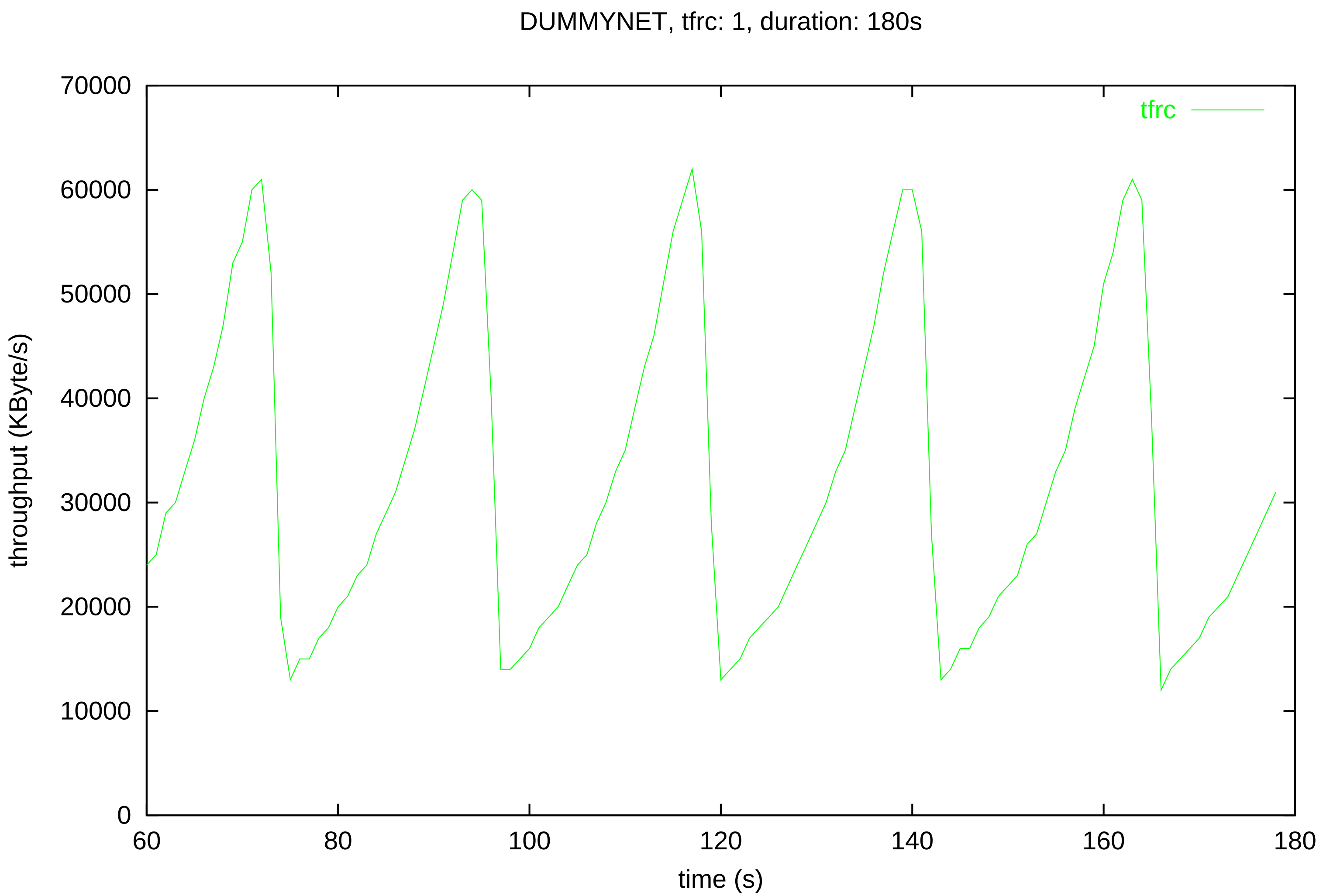


- Experiments by Joerg Widmer. Four TCPs, one TFRC.

Unicast experiments: Dummynet



Unicast experiments: Dummynet



Unicast: Issues that need further work.

- Receiver's algorithm for estimating the packet drop rate when it has been a long time since the most recent packet drop.

- Interactions with changing RTTs.

- Analysis of stability, oscillations.
  - How to avoid overshooting or undershooting on adjustments in the sending rate.

- Interactions in more complex environments.

- Idle and application-limited periods.

Complications introduced by multicast:

- How aggressively can the sender slow-start?

- In unicast, the sender needs positive feedback to keep on sending.

For multicast, receivers can have the responsibility to unsubscribe if their congestion control feedback is not reaching the sender.

- Transient traffic dynamics with changing round trip times?