TFRC: TCP Friendly Rate Control using TCP Equation Based Congestion Model

ECE 599: Multimedia Networking
Thinh Nguyen
References


Internet Transport Protocols

- **UDP**
  - No congestion control
  - No reliability – no retransmission of lost packets
  - Small throughput and delay fluctuation (jitter)

- **TCP**
  - Congestion control
  - Reliability – automatic retransmission
  - Large throughput and delay fluctuations

- **Multimedia Protocols**
  - Congestion control
  - Small throughput and delay fluctuation
  - Fair with TCP
  - No need for retransmission
TCP Bandwidth Model

Goal: Estimate TCP bandwidth based on the loss rates and round trip times
TCP Behavior

Assume: \( W \) is a Markov renewal random variable

Loss occurs

Window size

Evolution of window size over time

TDP: triple duplicate period
Detail view of TDP model
Estimated Bandwidth

\[ B(p) = \frac{1}{RTT} \sqrt{\frac{3}{2bp}} + o\left(\frac{1}{\sqrt{p}}\right) \]

\( p \) : Loss rate

\( B(p) \) : Bandwidth (throughput)

\( RTT \) : Round trip time
Derivation of \[ B(p) = \frac{1}{RTT} \sqrt{\frac{3}{2bp}} + o\left(\frac{1}{\sqrt{p}}\right) \]

Define

- \( Y_i \): The number of packets sent in period \( A_i \)
- \( \alpha_i \): The first lost packet in period \( A_i \)
- \( X_i \): The round where the loss occurs
Derivation of \[ B(p) = \frac{1}{RTT} \sqrt{\frac{3}{2bp}} + o\left( \frac{1}{\sqrt{p}} \right) \]

\[ B = \frac{E[Y]}{E[A]} \]
(Assume renewal process)

Now, we just need to figure out \( E[Y] \) and \( E[A] \)

\[ Y_i = \alpha_i + W_i - 1 \]
\[ E[Y] = E[\alpha] + E[W] - 1 \]

Now,
\[ P[\alpha = k] = (1 - p)^{k-1} p \]
\[ E[\alpha] = \sum_{k=1}^{\infty} (1 - p)^{k-1} pk = \frac{1}{p} \]
\[ E[Y] = \frac{1 - p}{p} + E[W] \]
Derivation of $B(p) = \frac{1}{RTT}\sqrt{\frac{3}{2bp}} + o\left(\frac{1}{\sqrt{p}}\right)$

$r_{ij}$: Round trip time of the j-th round of TDP$_i$

$A_i = \sum_{j=1}^{X_i+1} r_{ij} 

\rightarrow E[A] = (E[X] + 1)E[r] \quad RTT = E[r]$

Now,

$W_i = \frac{W_{i-1}}{2} + \frac{X_i}{b} \rightarrow E[W] = \frac{2E[X]}{2}$

$Y_i = \sum_{k=0}^{X_i/\beta-1} \left(\frac{W_{i-1}}{2} + k\right)b + \beta_i$

$\rightarrow Y_i = \frac{X_i}{2} \left(\frac{W_{i-1}}{2} + W_i - 1\right) + \beta_i$

$\frac{1-p}{p} + E[W] = \frac{E[X]}{2} \left(\frac{E[W]}{2} + E[W] - 1\right) + E[\beta]$
Derivation of \( B(p) = \frac{1}{RTT} \sqrt{\frac{3}{2bp}} + o(\frac{1}{\sqrt{p}}) \)

\[
E[W] = \frac{2+b}{3b} + \sqrt{\frac{8(1-p)}{3bp}} + \left(\frac{2+b}{3b}\right)^2
\]

\[
\rightarrow E[W] = \sqrt{\frac{8}{3bp}} + o\left(\frac{1}{\sqrt{p}}\right)
\]

\[
E[X] = \frac{2+b}{6} + \sqrt{\frac{2b(1-p)}{3p}} + \left(\frac{2+b}{6}\right)^2
\]

\[
E[A] = RTT\left(\frac{2+b}{6} + \sqrt{\frac{2b(1-p)}{3p}} + \frac{(2+b)^2}{6}\right)
\]

\[
E[A] = \frac{(1-p)/p + E[W]}{E[A]} = \frac{1}{RTT}
\]
Derivation of

\[ B(p) = \frac{1}{RTT} \sqrt{\frac{3}{2bp}} + o\left( \frac{1}{\sqrt{p}} \right) \]

\[ B(p) = \frac{E[Y]}{E[A]} = \frac{(1-p)/p + E[W]}{E[A]} = \frac{1}{RTT} \sqrt{\frac{3}{2bp}} + o\left( \frac{1}{\sqrt{p}} \right) \]
More Accurate Model

\[ T = \frac{s}{R\sqrt{\frac{2p}{3}} + t_{RTO}(3\sqrt{\frac{3p}{8}})p(1 + 32p^2)} \]

• Round trip delay \( R \) (measured at source)
• Packet size \( s \) (measured at source)
• Packet loss (congestion) rate \( p \) (fed back by rcv each RTT)
• Retransmission time out \( t_{RTO} \) (measured at source)
TFRC (TCP-Friendly Rate Control)

- How does TFRC relate to TCP Model?

- Sender receives the feedback on packet loss event rate $p$ from the receiver every RTT.

- Sender calculates new value of allowed sending rate; it increases/decreases current value to match the calculated rate.

- In so doing, TFRC behaves like any other TCP Reno session (same equation); it produced the same external effects.
TFRC Continues

- **Sender**: measures various parameters; calculates the TCP-like rate corresponding to the measured parameters.

- **Receiver**: provides feedback to sender to allow it to calculate RTT; also calculates loss event rate $p$.

- The $p$ rate computation critical for performance of TFRC.

- **Average Loss Interval**: weighted average of loss rate over the last $N$ loss intervals (loss interval = interval of packets between loss episodes)
Calculate loss rates: Sliding window

Lost packet

\[ P = \frac{2}{8} = 0.25 \]

\[ P = \frac{1}{8} \]
Calculate loss rates: Average loss interval
TCP Round Trip Time

\[ \text{EstimatedRTT} = (1 - \alpha) \times \text{EstimatedRTT} + \alpha \times \text{SampleRTT} \]

- Exponential weighted moving average
- Influence of past sample decreases exponentially fast
- Typical value: \( \alpha = 0.125 \)
TCP Round Trip Time

RTT: gaia.cs.umass.edu to fantasia.eurecom.fr

SampleRTT
Estimated RTT
TCP Timeout

- **Mimic TCP timeout computation**
- **EstimatedRTT** plus “safety margin”
  - large variation in **EstimatedRTT** -> larger safety margin
- first estimate of how much SampleRTT deviates from EstimatedRTT:
  \[
  \text{DevRTT} = (1-\beta) \times \text{DevRTT} + \\
  \beta \times |\text{SampleRTT} - \text{EstimatedRTT}|
  \]
  
  (typically, \(\beta = 0.25\))

Then set timeout value in the TFRC equation to

\[
\text{TimeoutInterval} = \text{EstimatedRTT} + 4 \times \text{DevRTT}
\]
NS Simulation results: TCP SACK + TFRC fair sharing
Normalized TCP Thr = 1 means perfect fairness

N TCP flows + N TFRC flows
TFRC less aggressive than TCP
TFRC internally unevenly “fair”
TFRC results in lower throughput jitter than TCP – using RED queuing
TFRC results in lower throughput jitter than TCP – Using DropTail queuing
40 “long lived” flows simulation: the 40 flows start in the first 20 s. We show bottleneck queue dynamics.

Comment: TFRC (bottom) is as stable as TCP (top). TCP drop rate = 4.9%; TFRC drop rate = 3.5%
Internet Measurements: 3 TCP connections – London to Berkeley. Throughput measured over 1 sec intervals.

TFRC much more stable than TCP
Conclusions

- TFRC is better than TCP for low throughput and delay jitter applications.
- TFRC shares bandwidth fairly with TCP.
- Can be used in Multicast?