

## Traffic Rate Control for Real-Time Applications

These slides are created by Dr. Yih Huang of George Mason University. Students registered in Dr. Huang's courses at GMU can make a single machine-readable copy and print a single copy of each slide for their own reference, so long as each slide contains the copyright statement, and GMU facilities are not used to produce paper copies. Permission for any other use, either in machine-readable or printed form, must be obtained from the author in writing.

CS 756

1

## Motivations

- ❑ Real-time applications need constant-rate delivery but are not stringent in reliability.
  - Teleconference, multimedia playback
- ❑ TCP Problems
  - Bursty delivery
  - exponential backoff interrupts regular delivery
- ❑ UDP Problem: no congestion control
  - unfair to TCP applications
  - the network may collapse (if no one backs off in congestion, no one's packets get thru)

CS 756

2

## TCP-Friendly Congestion Control

- ❑ In this talk, we discuss congestion control algorithms that are TCP-conformant in terms of bandwidth usages.
  - real-time applications must react to congestion in the same way as TCP so as not to disadvantage existing TCP applications.
- ❑ With such algorithms, one can design realtime transport protocols for use by multimedia applications

CS 756

3

## Realtime Transport Protocols

- ❑ Rate-regulated traffic (no burst traffic)
- ❑ Relaxed reliability model
- ❑ Adjust traffic rate according to network congestion conditions (congestion control)
- ❑ Ideally, a real-time application adjusts its “quality” according to the current transmission rate.
  - Change frame rate/size, color depth, ...
  - Change audio bit rate

CS 756

4

## Performance Criteria for Rate-Control Algorithms

- Smoothness
  - Difficult for multimedia applications to adapt if traffic rate go up and down too much
- Responsiveness
  - How fast do we respond to developing congestion situations ?
  - How fast do we take advantages of newly available resources ?

CS 756

5

## Two Approaches

- Additive increase, multiplicative decrease (**AIMD**)
  - Follow the basic AIMD design of TCP
  - Changes are made to produce smoother changes in traffic rates while maintaining TCP conformance in the long term.
- Equation-base rate control
  - Estimate the packet loss rate  $p$ .
  - Compute TCP sending rate according to  $p$ .

CS 756

6

## AIMD in TCP

- ❑ No packet loss, increase cwnd by 1 (or  $1/b$ ) per RTT.
- ❑ For each loss, decrease cwnd by half.
- ❑ The pure AIMD model is similar to TCP with only triple duplicate events.
- ❑ The pure AIMD model outperforms TCP for the cwnd is never set to 1.

CS 756

7

## Generalized AIMD

- ❑ No packet loss, increase cwnd by  $\alpha$  (or  $\alpha/b$ ) per RTT,  $\alpha > 0$ .
- ❑ For each loss, multiply cwnd by  $\beta$ ,  $0 < \beta < 1$ .
- ❑ In TCP,  $\alpha = 1$  and  $\beta = 0.5$ .
- ❑ **Question:** Are there other  $\alpha$ - $\beta$  combinations that produce the throughput of TCP ?
- ❑ Why ask the question ?
  - A larger-than-0.5  $\beta$  value smooths out the fluctuation of caused by cwnd reduction.

CS 756

8

## Analysis

- TCP performance formula can be generalized to

$$B_{\alpha,\beta}(p, RTT, T_0, b) = \frac{1}{\underbrace{RTT \sqrt{\frac{2b(1-\beta)p}{\alpha(1+\beta)}}}_{TD_{\alpha,\beta}(p, RTT, b)} + T_0 \min\left(1, 3\sqrt{\frac{b(1-\beta^2)p}{2\alpha}}\right) \underbrace{p(1+32p^2)}_{TO_{\alpha,\beta}(p, T_0, b)}}$$

- You can verify that

$$B_{1,0.5}(p, RTT, T_0, b) = \frac{1}{RTT \sqrt{\frac{2bp}{3}} + T_0 \min\left(1, 3\sqrt{\frac{3bp}{8}}\right) p(1+32p^2)}$$

is the performance of TCP with both TD and TO

CS 756

9

## Solving GAIMD Parameters

- We would like to find  $\alpha$  and  $\beta$  values such that, for fixed  $RTT$ ,  $T_0$ , and  $b$ , and for “most”  $p$  values:

$$B_{\alpha,\beta}(p, RTT, T_0, b) \approx B_{1,0.5}(p, RTT, T_0, b)$$

- Focusing on TD:

$$\begin{aligned} TD_{\alpha,\beta}(p, RTT, b) &= TD_{1,0.5}(p, RTT, b) \\ \Rightarrow RTT \sqrt{\frac{2b(1-\beta)p}{\alpha(1+\beta)}} &= RTT \sqrt{\frac{2b(1-0.5)p}{1 \times (1+0.5)}} \\ \Rightarrow \frac{1-\beta}{\alpha(1+\beta)} &= \frac{1-0.5}{\alpha(1+0.5)} \quad \Rightarrow \alpha = \frac{3(1-\beta)}{1+\beta} \end{aligned}$$

CS 756

10

□ Focusing on TO:

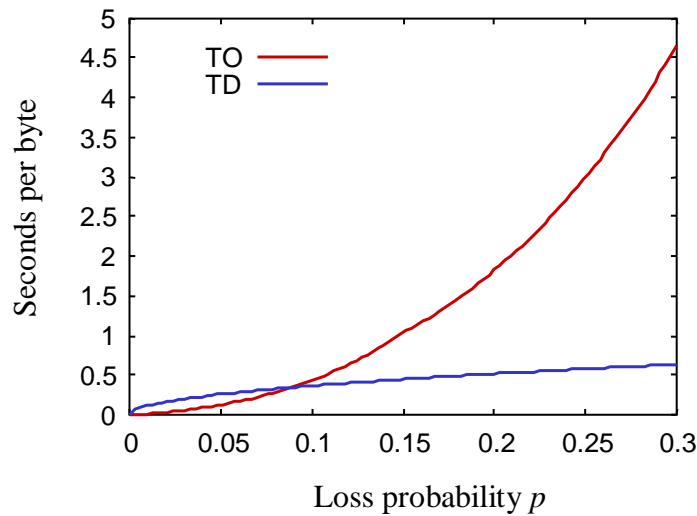
$$TO_{\alpha,\beta}(p, T_0, b) = TO_{1,0.5}(p, T_0, b)$$

$$\Rightarrow T_0 \min\left(1, 3\sqrt{\frac{b(1-\beta^2)p}{2\alpha}}\right) p(1+32p^2) = T_0 \min\left(1, 3\sqrt{\frac{b(1-0.53^2)p}{2}}\right) p(1+32p^2)$$

$$\Rightarrow \sqrt{\frac{1-\beta^2}{\alpha}} = \sqrt{\frac{1-0.5^2}{1}} \Rightarrow \alpha = \frac{4(1-\beta^2)}{3}$$

□ Example: if  $\beta = 0.875$ , then  $\alpha = 0.3125$ .

### TD vs TO ( $RTT=1, T_0=4, b=2$ )



## Discussion

- We use  $\alpha$  and  $\beta$  values such that

$$TO_{\alpha,\beta}(p, T_0, b) = TO_{1,0.5}(p, T_0, b)$$

- These  $\alpha$  and  $\beta$  values will *not* satisfy

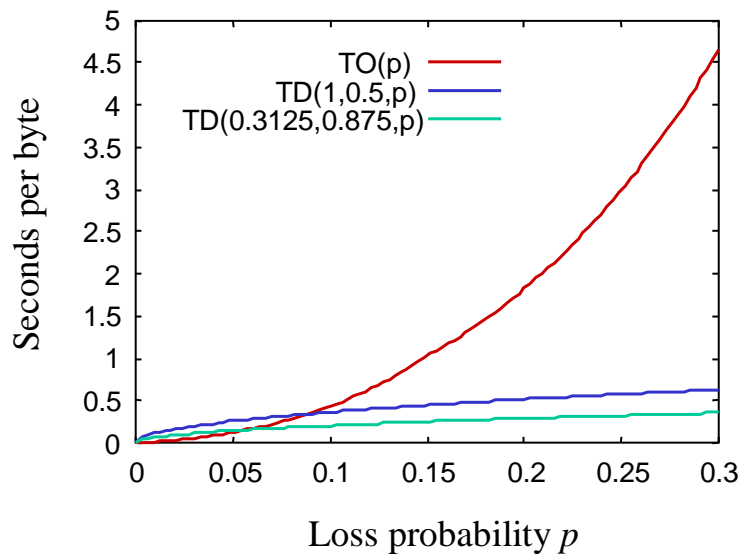
$$TD_{\alpha,\beta}(p, RTT, b) = TD_{1,0.5}(p, RTT, b)$$

- However, since TD are not important for most  $p$  values, the discrepancy is insignificant in the combined result  $B_{\alpha,\beta}(p)$ .
- For very small  $p$ , our choice of  $\alpha$  and  $\beta$  tends to slightly over estimate throughput.

CS 756

13

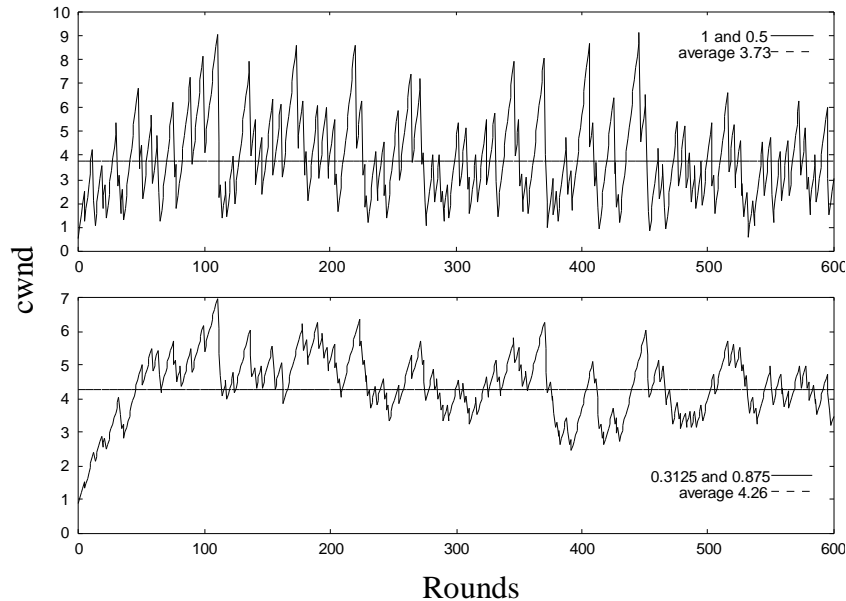
**$\alpha = 0.3125$  and  $\beta = 0.875$**



CS 756

14

## Results ( $p=0.05, b=2$ )



CS 756

15

## Discussion

- ❑ We've discussed a method to produce TCP conformant traffic with smoother reactions to packet losses.
- ❑ This is important to multimedia applications to seamlessly adjust quality.
- ❑ However, as seen in the previous page, GIMD changes cwnd (thus traffic rate and in turn application quality) without changes in packet loss probabilities (that is, network condition is stable).

CS 756

16

## Equation-Based Rate Control

- ❑ Estimate  $p$  and  $RTT$  and computes  $B(p, RTT, T_0, b)$ , where  $T_0$  and  $b$  are implementation parameters.
  - Of course, if  $p$  does not change,  $B$  is stable.
- ❑ Packets are transmitted at a rate determined by  $B(p, RTT, T_0, b)$ .
- ❑ We will discuss an equation-based approach proposed by Sally Floyd, called TCP-Friendly Rate Control (**TFRC**).

CS 756

17

## TFRC Protocol Overview

- ❑ Sender adds sequence numbers to packets.
- ❑ Receiver detects packet losses by gaps in sequence numbers.
- ❑ Receiver computes packet loss probability  $p$ .
- ❑ Receiver piggybacks  $p$  to ACK.
- ❑ Using ACK, sender computes  $RTT$  and  $T_0$  in the same way as TCP.
- ❑ Using the packet loss probability  $p$  from the receiver, sender computes  $B(p)$ , and delivers packets at the rate set by  $B(p)$ .

CS 756

18

- ❑ Packet transmissions are *not* ACK-clocked.
  - You don't have to wait for an ACK to “release” the next packet.
- ❑ Rather packets are transmitted at fixed intervals according to  $B(p)$ .
- ❑ Packet losses are detected through ACK and retransmissions attempted after timeouts.
  - Retransmissions shall not violate  $B(p)$ .
  - A retransmission occupies a “slot” like regular transmissions.

CS 756

19

## Average Loss Rate

- ❑ We measure **loss event rate**, not packet loss rate.
  - What is the difference ?
- ❑ Define a **loss interval** be the number of packets between successive loss events.
- ❑ Computing loss event rate is equivalent to computing the average loss interval.
  - Can you see why ?

CS 756

20

- ❑ Moving exponential average is “considered” too slow to reflect recent conditions.
- ❑ We need a way to give recent rounds high weights while still taking into account the past.

- ❑ Let  $s_i$  be the  $i$ -th sample of loss intervals with  $s_1$  being the most recent sample and  $s_0$  being the present, unfinished sample.
- ❑ We compute the average loss interval from the past  $n$  samples by

$$s(0, n-1) = \frac{\sum_{i=0}^{n-1} w_i s_i}{\sum_{i=0}^{n-1} w_i}, \quad w_i = \begin{cases} 1 & 0 \leq i \leq n/2 - 1 \\ 1 - \frac{i+1-n/2}{n/2+1} & n/2 \leq i < n \end{cases}$$

- ❑ Example:  $n = 8$  gives the weights, from recent to past: 1, 1, 1, 1, 0.8, 0.6, 0.4, 0.2
- ❑ Finally, set loss probability  $p = 1/s(0, n-1)$ .

## Properties of $s(0, n-1)$

- It is obvious that the sum of the first half of the weights is

$$\sum_{i=0}^{n/2-1} w_i = \frac{n}{2}$$

- The sum of the second half is

$$\begin{aligned}\sum_{i=n/2}^{n-1} w_i &= \frac{n}{2} - \sum_{i=n/2}^{n-1} \frac{i+1-n/2}{n/2+1} \\ &= \frac{n}{2} - \frac{1+2+\dots+(n/2)}{n/2+1} \\ &= \frac{n}{2} - \frac{(n/2)(n/2+1)}{2(n/2+1)} = \frac{n}{4}\end{aligned}$$

CS 756

23

- Thus, all the weights add up to  $(3/4) \times n$ .
- The effective weight  $w'$  of the present, unfinished sample  $s_0$  is

$$\frac{1}{\frac{3}{4} \times n}$$

- For  $n = 8$ , we have  $w' = 1/6$ .

CS 756

24

## Rate Increase Per RTT

- For simplicity, we compute traffic rate by the equation below and set  $b$  to 1.

$$B(p) = \frac{1}{RTT} \sqrt{\frac{3}{2bp}}$$

- Let  $A=1/p$  be the average loss interval (in packets).
- $B(p)$  dedicates the transmission rate of

$$\frac{\sqrt{1.5}}{\sqrt{p}} \approx \frac{1.2}{\sqrt{p}} = 1.2\sqrt{A}$$

packets per second

CS 756

25

- After a RTT of no loss, unfinished sample  $s_0$  is increased by  $1.2\sqrt{A}$
- Thus  $A$  is increased to  $A + w'1.2\sqrt{A}$
- In turn, the transmission rate is increased to

$$1.2\sqrt{A + w'1.2\sqrt{A}} = 1.2\sqrt{A} + \Delta_B$$

- The increase in transmission rate per RTT without packet losses is

$$\Delta_B = 1.2\left(\sqrt{A + w'1.2\sqrt{A}} - \sqrt{A}\right)$$

- With  $w = 1/6$ , the increase in transmission rate is approximately 0.12 packets/RTT.

CS 756

26

## **QoS Concerns**

- Do the solutions discussed here address the Quality of Service (QoS) requirements of realtime applications ?