

Lecture 19 Congestion Control

Peter Steenkiste
Departments of Computer Science and
Electrical and Computer Engineering
Carnegie Mellon University

15-441 Networking, Spring 2006
<http://www.cs.cmu.edu/~prs/15-441>

Peter A. Steenkiste, SCS, CMU

1

Outline of the "Transport Lectures"

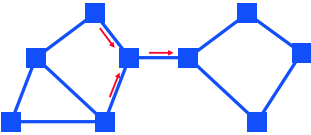
- **Transport protocols introduction.**
 - » Functions, UDP
- **Flow and error control.**
 - » Stop and go, sliding window, ...
- **TCP.**
 - » Connections, flow control, error handling, extensions, ...
- **Congestion control.**
 - » Congestion definition, congestion control strategies
- **Congestion control in TCP.**
 - » More TCP, applied congestion control
- **Other transports.**
 - » TCP conformance

Peter A. Steenkiste, SCS, CMU

2

What is Congestion?

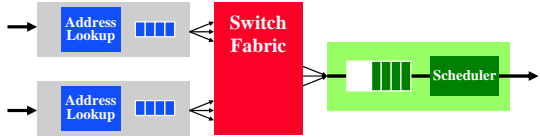
- **The load placed on the network is higher than the capacity of the network.**
 - » Not surprising: independent senders place load on network
- **Results in packet loss: routers have no choice.**
 - » Can only buffer finite amount of data
 - » End-to-end protocol will typically recover, e.g. TCP



Peter A. Steenkiste, SCS, CMU

3

Switch Behavior under Congestion



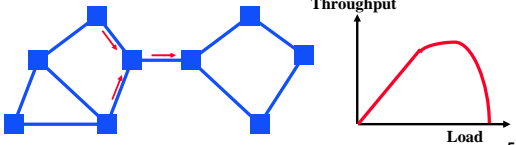
- **Buffering protects flows from short-term congestion.**
 - » Buffer fills up when fill rate exceeds drain rate
 - » Buffer drains when fill rate drops below drain rate
- **Longer term overload will result in overflow of the buffer on the congested bottleneck link.**
 - » Packets are entering the buffer faster than they leave the buffer for extended period of time

Peter A. Steenkiste, SCS, CMU

4

Why is Congestion Bad?

- **Wasted bandwidth: retransmission of dropped packets.**
- **Poor user service : unpredictable delay, reduced throughput.**
- **Increased load can even result in lower network throughput.**
 - » Switched nets: heavy traffic -> long queues -> lost packets -> retransmits
 - » Ethernet: high demand -> many collisions
 - » compare with highways: too much traffic slows down throughput



Peter A. Steenkiste, SCS, CMU

5

Possible Solutions

- **Redesign the network.**
 - » Add capacity to congested links
 - » Very slow solution: takes days to months!
- **Reroute traffic.**
 - » Alternate paths are not always available
 - » Also too slow: takes 10s of seconds
 - » In practice, most routing algorithms are traffic independent
 - Why?
- **Adjust the load in the network.**
 - » What are the options?

Peter A. Steenkiste, SCS, CMU

6

Congestion: Challenge

- **Two views of the congestion problem**
 - › Network has certain amount of bandwidth
 - Sum of the transmission rates of all applications must be less than the aggregate network bandwidth
 - › Network is a giant memory: router buffers + links
 - Number of packets each application has in the network must be smaller than the network memory size
- **Challenge:**
 - › Keep network at a good operating point
 - › “Fair” distribution of bandwidth across users
- **Two approaches to dealing with congestion.**
 - › Open loop
 - › Closed loop

Peter A. Steenkiste, SCS, CMU

7

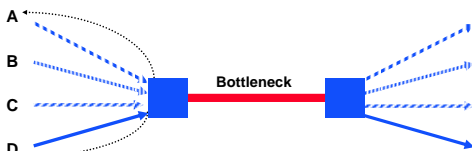
Open Loop Solutions

- **Limit traffic.**
 - › Packet coalescing, ...
- **Limit burstiness of the traffic.**
 - › Make traffic easier to handle by network
 - › Traffic shaping
- **Require reservations: prevent congestion.**
 - › All connections have their own set of resources
 - › No (or limited sharing) of resources so no contention
 - › Does not work well for bursty traffic
 - Performance issues?
 - Efficiency issues?
- **More details in QoS section.**

Peter A. Steenkiste, SCS, CMU

8

The Big Picture: Closed Loop Flow Control



- **How does the network apply back pressure?**
 - › Challenge: delay in feedback loop and vagueness of feedback
- **How do the sources adapt?**
 - › Challenge: diverse sources and malicious users
- **How does the switch distribute link bandwidth?**
 - › Challenge: treat sources fairly in a scalable way

Peter A. Steenkiste, SCS, CMU

9

Feedback Mechanisms: Explicit Feedback

- **The network provides the sender with explicit information on network conditions.**
 - › Explicit information reduces the chance of error
 - › Can be sent as separate packet or can be piggybacked on data packets
- **Always requires support on switch.**
 - › Calculate the feedback and generate the signal
 - › Switch overhead and bandwidth use
- **Many flavors of explicit feedback exist.**
 - › forward versus backward congestion indication
 - › binary versus multivalued
- **Requires some degree of homogeneity.**
 - › Switches have to generate consistent feedback
 - › End-points have to interpret feedback consistently

Peter A. Steenkiste, SCS, CMU

10

Feedback Mechanisms: Implicit Feedback

- **Sender observes quality of connection to estimate congestion status.**
 - › Example: dropped packets indicate congestion
 - usually true
 - › More sophisticated solutions, e.g. packet pair
- **Does not require explicit network support**
 - › Router has no choice - what else will it do?
 - › No additional bandwidth or additional router complexity
 - › But: interpretation often requires some knowledge of the internals of the network
- **Interpreting the information may be difficult for the sender.**
 - › Why was the packet dropped?

Peter A. Steenkiste, SCS, CMU

11

End-Point Behavior

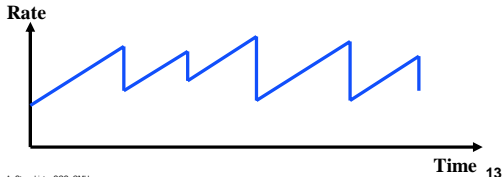
- **Hosts should reduce transmission rate when they receive a congestion indication.**
 - › That much is obvious: important for the stability of the network
- **Hosts are allowed to increase rate if there is no congestion (= use excess capacity).**
 - › Also obvious at least in the case of explicit, multi-valued feedback
- **But feedback typically only gives information about the presence of congestion**
 - › Hard to distinguish between an ideal rate and a low rate
 - › Solution is probing: periodically increase rate to see whether more bandwidth is available

Peter A. Steenkiste, SCS, CMU

12

Adaptation with Binary Feedback

- Network stability requires multiplicative decreases and additive increases in the transmission rate (AIMD).
 - › Instance of simple linear control $W_{i+1} = a + b \times W_i$
- Congestion is a dangerous condition
 - > back off quickly
- Unused bandwidth is undesirable but not dangerous
 - > probe carefully

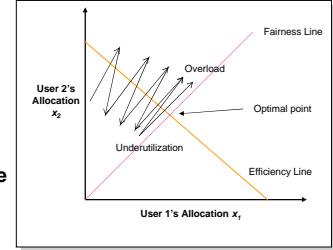


Peter A. Steenkiste, SCS, CMU

13

Phase Plots

- Simple way to visualize behavior of competing connections over time
- Additive: move along line parallel with fairness line.
- Multiplicative: move along line through origine.
- AIMD converges to fair equilibrium.

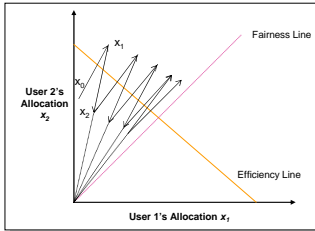


Peter A. Steenkiste, SCS, CMU

14

What is the Right Choice?

- For decrease: $a=0$ & $b < 1$
- For increase: $a > 0$ & $b \geq 0$
- AIMD converges to fair equilibrium.
 - › Can have multiplicative term in increase (MAIMD)
 - › AIMD moves towards optimal point



Peter A. Steenkiste, SCS, CMU

15

End-Point Behavior: Discussion

- Response has to balance congestion and performance considerations.
 - › Large backoff can reduce performance unnecessarily
 - › Slow backoff may not eliminate congestion
- End-point response determines stability of the network.
 - › Congestion collapse: throughput drops to zero when hosts do not back off
 - › Rule: multiplicative slowdown and incremental speedup
- Congestion adaptation must be robust in a heterogeneous environment.
 - › Switches and hosts will all have slightly different behavior
 - › Some applications may want to adapt to congestion in specialized ways

Peter A. Steenkiste, SCS, CMU

16

Fair Bandwidth Distribution

- Distribution of bandwidth of users across users is controlled by the order in which packets are forwarded.
 - › I.e. by the packet scheduler.
- First-In-First-Out: treat all packet the same.
 - › Simple, very widely used
- Is this fair?
- No: greedy users get more bandwidth.

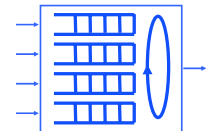


Peter A. Steenkiste, SCS, CMU

17

Packet Scheduling: (Weighted) Fair Queuing

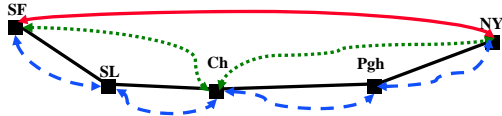
- Separate queues for different flows.
 - › Flow can be application, end-point, organization, ..
- Scheduler distributes bandwidth equally across the flows.
 - › Or according to weights
- Buffering combined with scheduling isolates flows.
 - › Greedy users will see their queue overflow
 - › Other users are protected
- Is this fair?
- Probably more fair than FIFO
 - › But maybe not ...



Peter A. Steenkiste, SCS, CMU

18

Fairness - Part 3



- What is fair?
- All connections get the same bandwidth?
- Red solid connections gets certain bandwidth.
- Blue dashed connections get more bandwidth?
- What about the green dotted connections?

Peter A. Steenkiste, SCS, CMU

19

Discussion: Fairness

- “Fairness” is a very subjective concept.
- Depends on the metric that is used:
 - › Are all packets equal?
 - › Are all users equal?
 - › Does fairness depends on number of hops, distance, ..?
 - › How much are people paying – does it matter?
- Example: max-min fairness.
 - › Model is that all users are equal
 - › Used, e.g., for some of the ATM traffic classes
 - › There is “weighted” version as well

Peter A. Steenkiste, SCS, CMU

20

Max-Min Fair Sharing

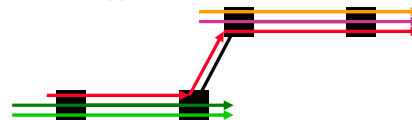
- On each link, flows are divided in two groups.
 - › Flows that are bottlenecked elsewhere
 - › Flows that are bottlenecked here
- The max-min fair share rate R_{fair} of a network link is defined such that
 - › Flows bottlenecked at this link all have rate $r = R_{\text{fair}}$
 - › Flows bottlenecked elsewhere have rate r , where
 - $r < R_{\text{fair}}$
 - r is the max-min fair share rate of their bottleneck link
- Two implementations:
 - › Multi-valued feedback: switch returns rate
 - › Single bit feedback: use congestion bit in the header
 - › Distributed algorithms that converge to max-min fairness

Peter A. Steenkiste, SCS, CMU

21

Max-Min Fair Sharing Example

$$r_{\text{fair}} = \frac{C - \sum_{\text{else}} r_i}{n_{\text{here}}}$$



Assume 10 Mbs links

Peter A. Steenkiste, SCS, CMU

22

Examples of Congestion Control

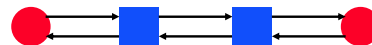
- Explicit feedback.
 - › binary feedback.
 - › multi-valued feedback.
- Implicit feedback.
 - › packet dropping
 - › packet pair

Peter A. Steenkiste, SCS, CMU

23

Example: DEC bit

- Switches set an explicit congestion bit in the packet header if the queue size is larger than one.
 - › Receiver collects the information and forwards it to the sender
- Senders slow down if the bit is set in more than 50% of the packets in a window.
 - › multiplicative slow down
 - › stepwise increase if bit is not set for certain period of time
- Behavior is very similar to TCP, except that it has explicit feedback.

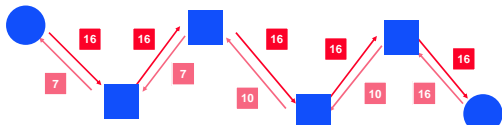


Peter A. Steenkiste, SCS, CMU

24

Example: ATM Rate-based Flow Control

- Provide explicit “per flow” feedback.
 - › Host tells the network how fast it is sending
 - › Switches calculate how fast the host should be sending and include this information in explicit flow control packets that are sent periodically
- Feedback can be binary or explicit.
 - › binary: source slows down or speeds up
 - › explicit: switch specifies the rate



Peter A. Steenkiste, SCS, CMU

25

Example: TCP

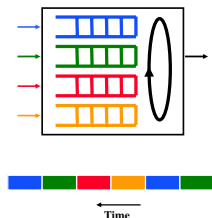
- Very simple mechanisms in network.
 - › FIFO scheduling with shared buffer pool
 - › Feedback through packet drops: the obvious thing to do
- TCP interprets packet drops as signs of congestion.
 - › Multiplicative back off when there are packet drops
 - › This is an assumption: packet drops are not a sign of congestion in all networks
 - e.g. wireless networks
- Periodically probes the network to check whether more bandwidth has become available.
 - › Results in linear speed up

Peter A. Steenkiste, SCS, CMU

26

Example: Packet Pair

- Sender sends two packets in a row and receiver measures the delay between the two received packets.
 - › assume a delay D and packet size P
- The receiver can now estimate the available bandwidth as P/D .
 - › average bandwidth given to the flow assuming a perfect fluid model
 - › other switches do not affect the delay because they are not the bottleneck (no queuing delay)



Peter A. Steenkiste, SCS, CMU

27

Tradeoffs in Congestion Control

- Explicit schemes that isolate traffic flows seem preferable, but require more support inside the networks.
 - › per-flow buffering, weighted fair queuing, policing
 - › scalability???
 - › determining nature of the explicit feedback in heterogeneous environment
- Diversity in networks makes TCP approach a good solution.
 - › Dropping packets is universally a natural response to congestion
 - › But many open issues: how to isolate poorly behaved sources, diversity in TCP implementations, ...

Peter A. Steenkiste, SCS, CMU

28

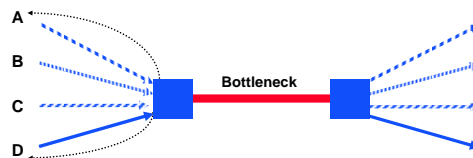
Outline of the “Transport Lectures”

- Transport protocols introduction.
 - › Functions, UDP
- Flow and error control.
 - › Stop and go, go back N, sliding window, ...
- TCP.
 - › Connections, flow control, error handling, extensions, ...
- Congestion control.
 - › Congestion definition, congestion control strategies
- Congestion control in TCP.
 - › More TCP, applied congestion control
- Other transports.
 - › TCP conformance, ..

Peter A. Steenkiste, SCS, CMU

29

Summary: Closed Loop Congestion Control



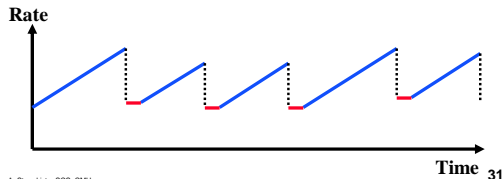
- How does the network apply back pressure?
 - › Dropped packets signal congestion
- How do the sources adapt?
 - › Additive Increase Multiplicative Decrease (AIMD)
- How does the switch distribute link bandwidth?
 - › FIFO queueing

Peter A. Steenkiste, SCS, CMU

30

TCP Congestion Control

- Packet loss is seen as sign of congestion and results in a multiplicative rate decrease
 - › Factor of 2
- TCP periodically probes for available bandwidth by increasing its rate.



Peter A. Steenkiste, SCS, CMU

31

TCP Congestion Control Open Questions

- How can this be implemented?
 - › Operating system timers are very coarse – how do you accurately calculate the transmission rate?
- How does TCP know what is a good initial rate to start with?
 - › Should work both for a CDPD (10s of Kbs or less) and for supercomputer links (10 Gbs and growing)
- Can we avoid the high overheads associated with the timeouts that detect packet loss?
 - › Packet loss will be periodic
 - › Timeouts are expensive!

Peter A. Steenkiste, SCS, CMU

32

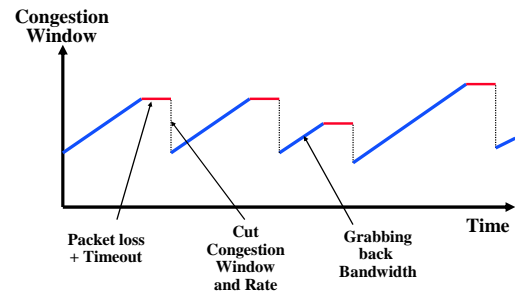
TCP Congestion Control Implementation

- Implemented using a congestion window that limits how much data can be in the network.
 - › Rate = data in transit / roundtrip time
 - Data can only be sent when the amount of outstanding data is less than the congestion window.
 - › The amount of outstanding data is increased on a “send” and decreased on “ack” – packet conservation
 - › (last sent – last acked) < congestion window
- $$\text{Throughput} < \frac{\text{Cong Window Size}}{\text{Roundtrip Time}}$$
- Congestion window is similar to the end-end flow control window.
 - › (last sent – last window update) < receiver window

Peter A. Steenkiste, SCS, CMU

33

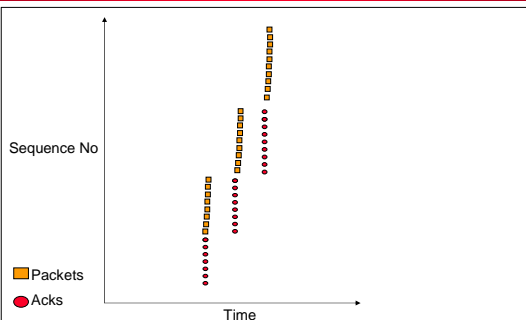
TCP Saw Tooth Behavior



Peter A. Steenkiste, SCS, CMU

34

Congestion Avoidance Sequence Plot



Peter A. Steenkiste, SCS, CMU

35

TCP Packet Pacing

- Congestion window helps to “pace” the transmission of data packets.
- In steady state, a packet is sent when an ack is received.
 - › Data transmission remains smooth, once it is smooth
 - › Self-clocking behavior



Peter A. Steenkiste, SCS, CMU

36

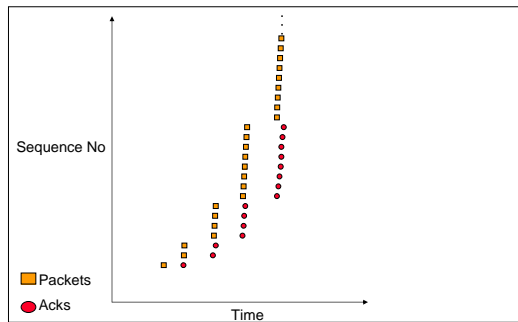
Slow start

- **At startup, no information is available about the congestion state of the network.**
 - › Sending the full flow control window would flood the switches and result in severe packet loss
- **Slow start quickly probes for a good window.**
 - › Additional counter that limits number of packets in network
 - › Initialized to 1
 - › Incremented on each ack: results in exponential increase in traffic - not so slow at all!
- **After packet losses and a timeout, TCP drops into (oscillating) steady state.**
 - › During slow start, TCP remembers the congestion window size before losses occurred and uses that as an initial congestion window

Peter A. Steenkiste, SCS, CMU

37

Slow Start Sequence Plot

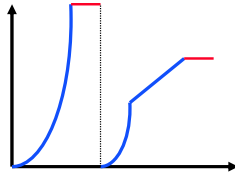


Peter A. Steenkiste, SCS, CMU

38

Slow Start also Starts Packet Pacing

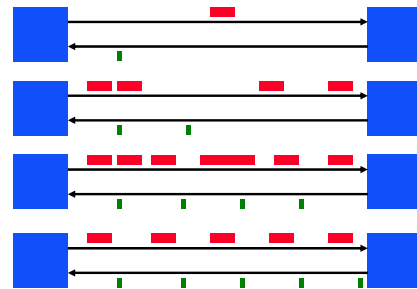
- **Slow start initiates the self-clocking behavior of TCP.**
 - › Sender sends one packet
 - › Receives an ack and sends two packets
 - › Packets will be separated on the bottleneck link, resulting in separate acks
 - › Sender sends two packets in response to each ack, ...
- **Slow start also used after timeout when pipeline has drained.**
 - › but only up to old congestion window



Peter A. Steenkiste, SCS, CMU

39

Starting of Packet Pacing



Peter A. Steenkiste, SCS, CMU

40

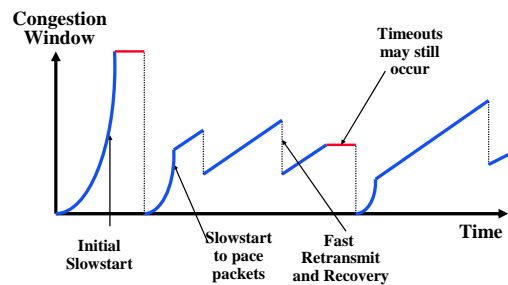
Fast Retransmit and Fast Recovery

- **Timeouts are very expensive.**
 - › Timeout itself is long
 - › Pipe drains - have to go through slow start again
 - › Congestion window is cut in half
- **Lost packets can sometimes be detected without timeout by checking for duplicate acks.**
 - › Receiver sends ack for every packet it receives
 - › A packet loss results in the same packet being acked again
- **After 3 duplicate acks sender retransmits packet without waiting for a timeout.**
 - › Fast retransmit: retransmit the missing packet immediately
 - › Fast recovery: skip slow start
 - Transmission pipeline has not drained so there is no need to start up the self-clocking process
 - But congestion window is still cut in half - why?

Peter A. Steenkiste, SCS, CMU

41

TCP Saw Tooth Behavior



Peter A. Steenkiste, SCS, CMU

42