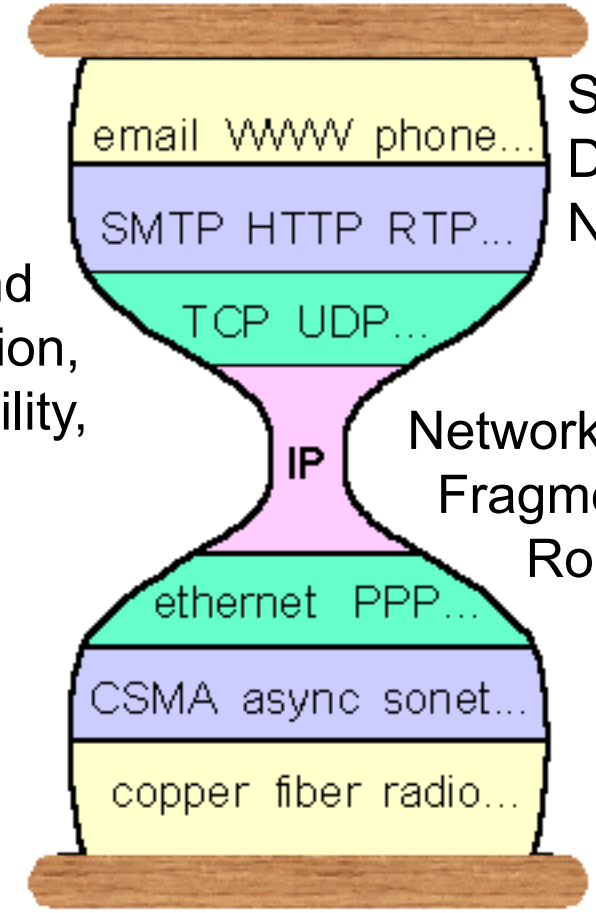
CS 557
Congestion and Complexity

Observations on the Dynamics of a Congestion Control Algorithm: The Effects of Two-Way Traffic
Zhang, Shenker, and Clark, 1991

Spring 2013

The Story So Far….



Some Essential Apps: DNS (naming) and NTP (time).

Transport layer: End to End communication, Multiplexing, Reliability, Congestion control, Flow control,

Network layer: Addressing, Fragmentation, Dynamic Routing, Best Effort Forwarding

Data Layer: richly connected network (many paths) with many types of unreliable links

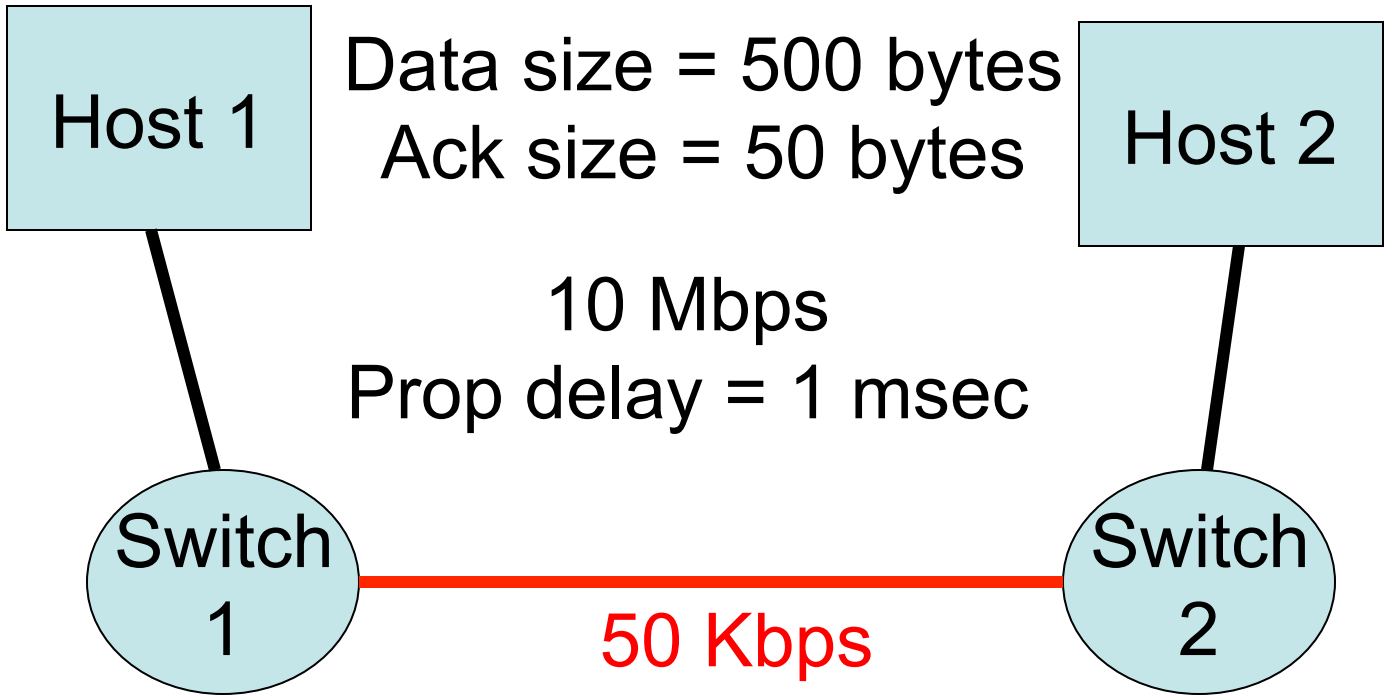
Main Points

- Objective:
  - Analysis of Congestion Control Algorithms
    - TCP Tahoe congestion control in particular
- Approach:
  - Simulations of a simple Network
  - Variations in traffic sources and windows
- Contributions:
  - Clustering, ACK-Compression, Phase Sync.
  - How to properly conduct experiments

Congestion Control Algorithm

- TCP algorithm from last lecture:
- If (cwnd < sstresh)
  cwind += 1;
  else
  cwnd += 1/[cwnd]
- If packet loss (duplicate acks or timeout)
  ssthresh=Max[cwnd/2, 2]
  cwnd =1
- Ignores maxwind (for flow control)
  – Not a factor in this study.

Network Topology



Pipesize P = (Bandwidth*prop delay)/packetsize

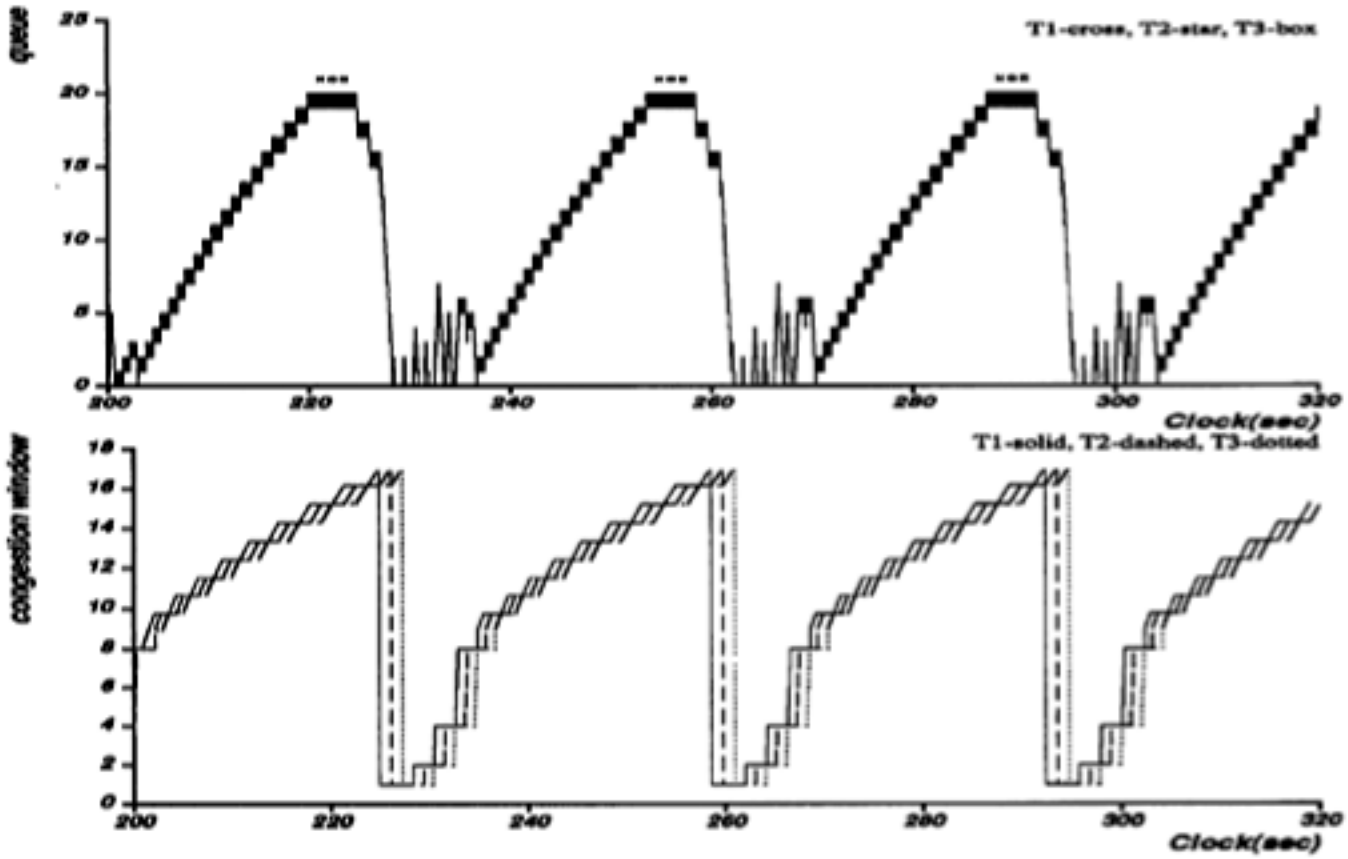
P = .125 packets (.01 sec delay)

P = 12.5 packets (1 sec delay)

Utilization

- Clearly can never get fully utilization on host-switch links.
- Utilization on bottleneck link
  - Want bottleneck link always full
- Pipe size
  - Link can hold 12.5 packets for 1 sec prop delay
  - Link can hold 0.125 packets for .01 sec prop delay
  - Larger pipe size decreases utliization
    - Need to feed more packets to keep the pipe full
- Buffer size
  - Larger buffer size increases utilization
    - Always some packet in the queue ready to send

One-Way Traffic



One-Way Traffic Analysis (1/2)

- Figure follows predictable TCP pattern
  - Window grows exponentially at first
  - Window grows linearly after passing threshold
  - Packet drop cuts the window back to 1
- Buffer Size and Throughput
  - Want full utilization of the bottleneck link
  - Increasing the pipe size decreases utilization
  - Increasing the buffer size increases the utilization
- Primary Goal:
  - Queue size at bottleneck should never drop to 0
  - Tune buffer size based on pipe size
  - “Rule of Thumb” for router buffer sizes
    - See “Sizing Router Buffers”, SIGCOMM 2004

One-Way Traffic Analysis (2/2)

- Packets from a connection are clustered
  - All the packets from one connection linked together
  - Start as a cluster of 1 and always remain clustered
  - Results since each ack triggers data
  - Can be violated if random spacing after ack
    - Here each ack immediately triggers data
  - Can be violated due to retransmit
    - But each connection loses a packet in the congestion epoch.
- Pacing or Non-Pacing
  - Pacing: data or acks not immediately transmitted
  - Non-pacing will exhibit clustering effect.

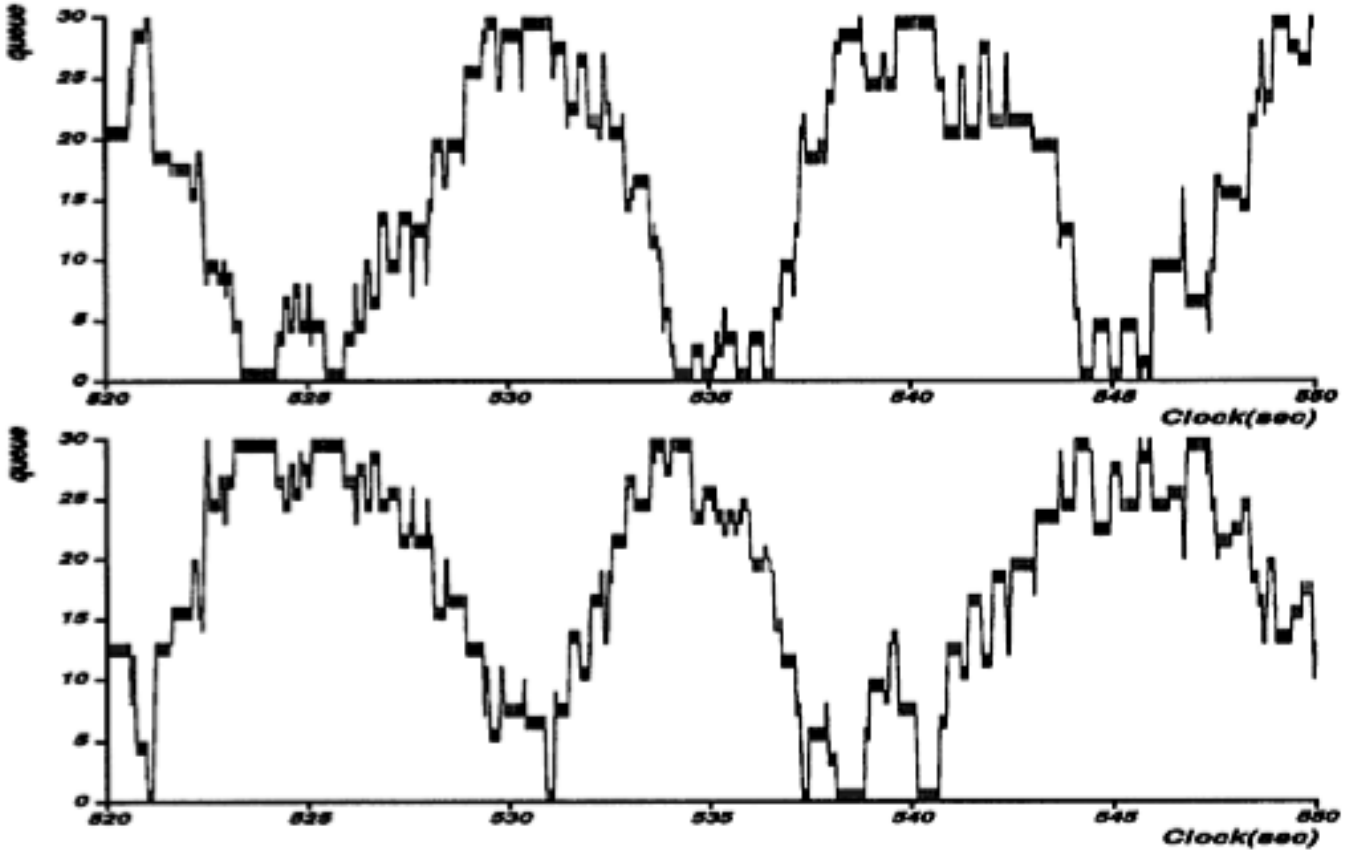
Summary for Part 1

- Observed One-Way Traffic and Found
  - Packet clustering
  - Increasing pipe size reduces utilization
  - Increasing buffer size increases utilization
- Next Observed Two-Way Traffic and Found
  - Packet clustering
  - Increasing buffer size decreased utilization
  - Complex dynamics
- Simplified Further to Fixed Window, Infinite Buffer
  - Fixed window results for stages 1,2,3,4,5
  - Dynamic window results in Figures 4,5, 6, and 7
  - Be prepared to explain these on an exam

Two-Way Traffic

Rapid changes that don't match changes in cwnd??

5 packet change in less than packet trans time



Buffer size = 30, util = 91%

Buffer size = 60, util = 87% (??)

Discussion

- Contradicts conventional wisdom:
  Larger buffer size results in smaller utliization
- Too difficult to understand dynamics
- Solution reduce to less complex model
  - One connection from host1 to host2
  - One connection from host2 to host1

Two-Way Traffic

Prop delay = 0.01 sec

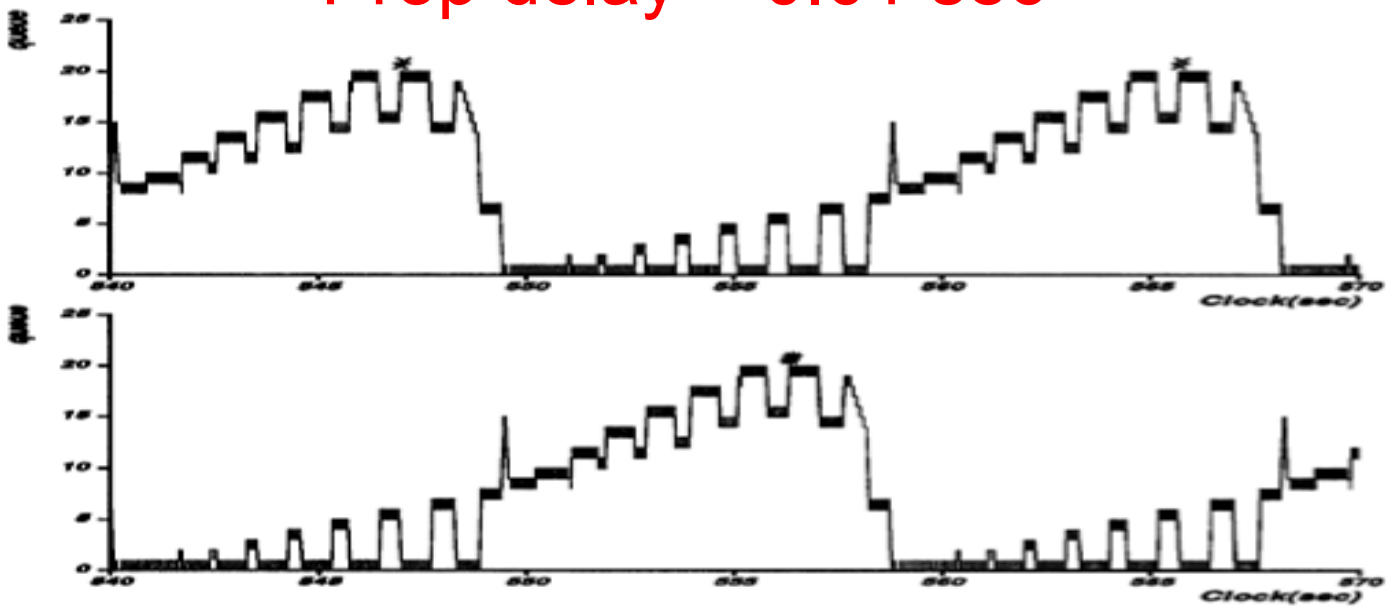


Figure 4: Packet queue at switches 1 and 2 for a configuration with $\tau = 0.01$ sec and with two connections, one having its source on Host-1 and the other having its source on Host-2. The switches have a buffer size of 20 packets. Each mark above the graphs indicates the dropping of a data packet. Note that during a congestion epoch one connection loses two packets while the other has no losses.

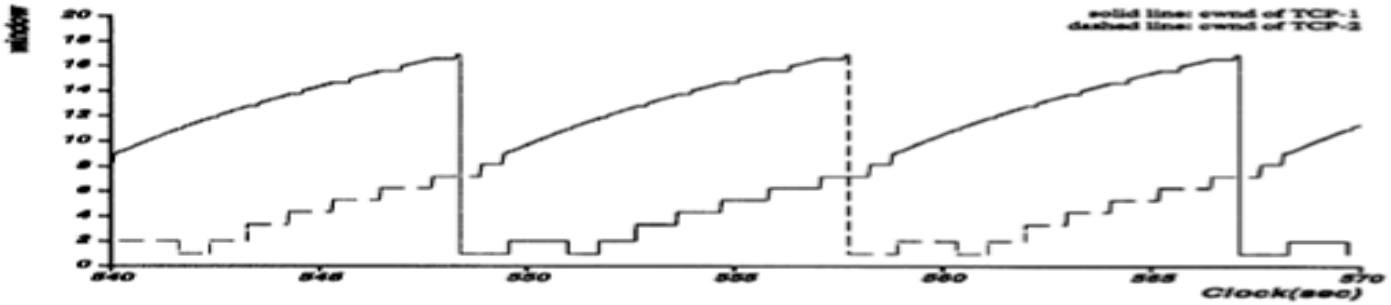


Figure 5: The congestion window sizes for the two connections in the configuration described above. The increase-decrease cycles of the two connections are synchronized out-of-phase.

Two-Way Traffic

Prop delay = 1 sec

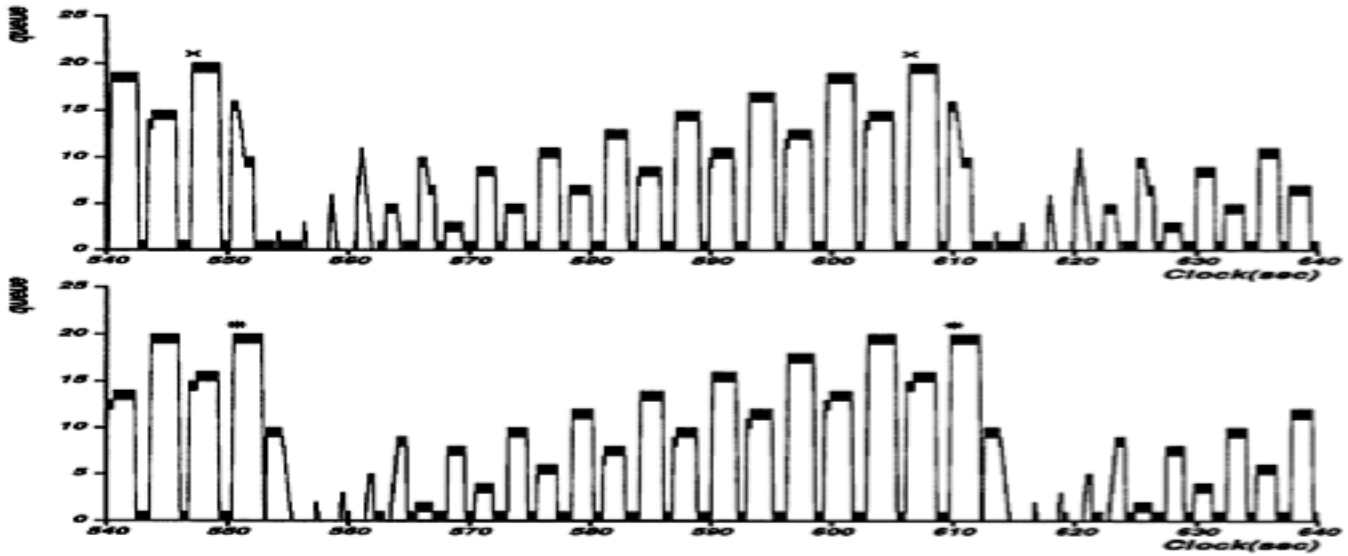


Figure 6: Packet queue at switches 1 and 2 for a configuration with $\tau = 1$ sec and with two connections, one having its source on Host-1 and the other having its source on Host-2. The switches have a buffer size of 20 packets. Each mark above the graphs indicates the dropping of a data packet. Note that during a congestion epoch both connections have a single packet dropped.

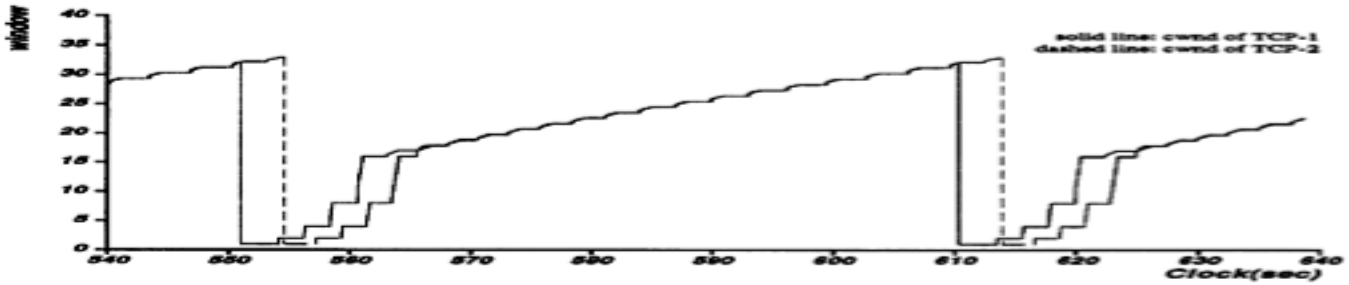


Figure 7: The congestion window sizes for the two connections in the configuration described above. The increase-decrease cycles of the two connections are synchronized in-phase.

Discussion

- Some predictions still hold
  - Packets are clustered
  - Two packets (= acceleration) dropped in congestion epoch
- Some unexpected results
  - High frequency oscillations
  - Out of sync windows when r=0.01
  - Drops are for the same connection when r=0.01
- Still too difficult to understand dynamics
- Solution reduce to less complex model
  - Infinite buffer space at the switches
  - Fixed window size at the hosts

Two-Way Traffic

Prop delay = 0.01 sec

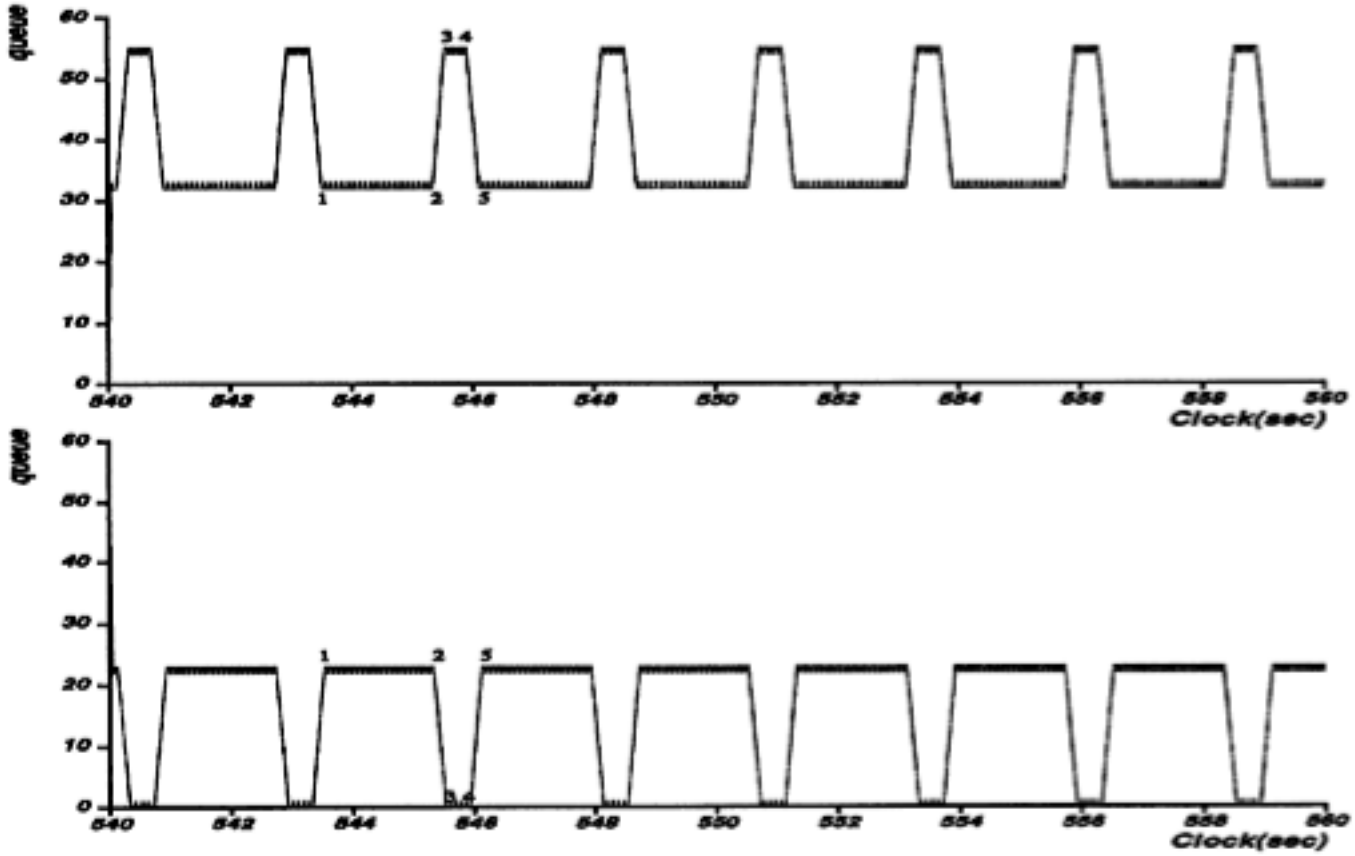


Figure 8: Packet queues at switches 1 and 2 for a configuration with $\tau = 0.01$ sec. There are two connections, one having its source on Host-1 and the other having its source on Host-2, with fixed window sizes of 30 and 25 respectively. The switches have infinite buffers. Note that the two queues have different maximum heights.

Two-Way Traffic

Prop delay = 1 sec

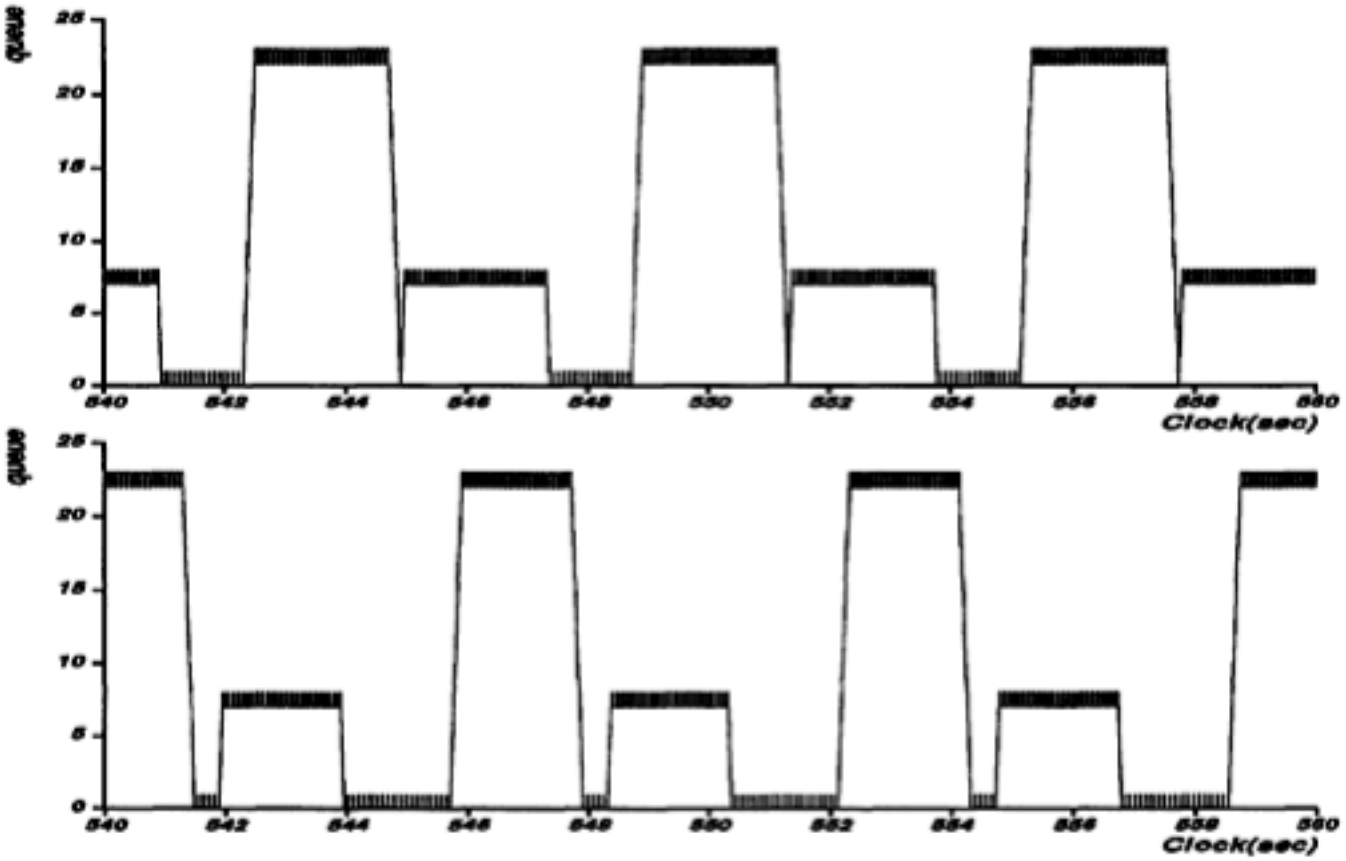


Figure 9: Packet queue at switches 1 and 2 for a configuration with $\tau = 1$ sec and with two connections, which have sources on Host-1 and Host-2 and have fixed window sizes of 30 and 25, respectively. The switches have infinite buffers. Note that the two queues have the same maximum height, and that there is an alternation pattern in the plateau heights.

Ack-Compression

- Ack messages encounter non-empty queues
  - Ack delay no longer simply a function of data packet arrivals
- Acks become compressed together
  - Cluster together in the queue
  - Transmitted out in a cluster
  - Transmit time for acks in much smaller than data packets
- Impact at the host
  - Burst of closely spaced ack packets
  - Send a burst of closely spaced data packets
- Impact at the Queue
  - Burst of data packets arrive at queue and increase queue
  - Cluster of attacks leave queue and decrease queue size
  - Important to note queue is number of packets,
    - not number of bytes

Out of Phase Synchronization

- Ack packets are never dropped
  - First ack never encounters full buffer.
  - Additional acks arrive at rate buffer drains
- One Connection Loses Two Packets
  - First loss: cwind = 1,
    sthresh = cwind/2
  - Second loss: cwind = 1
    sthresh = cwind/2 = 1/2
  - Slow start stops when window size = 2!
    - Low utilization while connection ramps up.

In Phase Synchronization

- Ack packets are never dropped
  - First ack never encounters full buffer.
  - Additional acks arrive at rate buffer drains
- Each Connection Loses One Packets
  - Both cwinds halved by congestion epoch
  - Reduced utilization on both directions.

- In both sync cases, the effective pipesize varies due to the ack delays