Last time: abstractions of the physical layer

* Elasticity buffer: mediate between packets @ bitrate r1 (r\_sender) and packets @ bitrate r2 (r\_receiver)
* Why would r1 be different from r2?
  + R1 may be different from r2 because clocks in the internet is different
  + Say the sender is sending at 10 Mbit/s (with a **clock** of 10 Mhz and 1 bit per cycle)
  + The receiver also has a **clock** of 10 Mhz and reads 1 bit per cycle, this is also 10Mbit/s
  + However, the two **clocks** have a different 10Mhz => r1 is 10 Mbit/s from the sender’s perspective and r2 is Mbit/s from the receiver’s perspective => r1 is not equal to r2.
* Although r1 may be different from r2, they can’t be too different from each other because of they are both a clock of 10Mhz and the there is a limited clock tolerance
  + and
* Since r1 is different from r2, some bad things may happen.
  + Case 1: r\_sender < r\_receiver => buffer underflows (the receiver would try to drain things from the buffer when it’s empty)
  + Case 2: r\_sender > r\_receiver => buffer overflows
* Communications interface parameters
  + MTU = 10 kbit
  + Inter-packet gap must be > (intuitively the receiver should be able to drain enough during this gap)
    - Proposal one:
    - Proposal two: => The receiver can always get back to zero state after the Inter-packet gap
    - The real number: look at the end of this notes
* Case 1: Underflow
  + is 9.99 Mhz and is 10.01 Mhz
  + Every one second, the elasticity buffer is drained by .
  + If the receiver only tells the system to start draining when there is at least 1 Mbytes in the buffer, it takes 400s to drain the buffer.
  + If the sender has a packet that is 4000 exabytes, it takes way more than 400s to drain the buffer, and therefore we need something to limit that — **MTU**
* Case 1: Underflow with MTU and a large buffer
  + Sender has packet that is 10 kbits
  + Receiver is manufactured with buffer that is 10 kbits
  + Receiver strategy:
    - 1) receive entire packet
    - 2) then tell its client to start reading
  + This works, but expensive
* Case 1: Underflow with MTU and a small buffer
  + Sender has packet that is 10 kbits
  + Receiver is manufactured with buffer that is 1 kbits
  + Receive strategy:
    - 1) receive first 1 kbit
    - 2) then tell its client to start reading
  + Starting at t=0, the buffer gets the first 1 kbit after ( 1 kbit / 9.99 Mbit/s is roughly 0.1 milliseconds )
  + Starting at t = 0.1 ms, it takes ( 1 kbit / 20 Kbits/s = 50 milliseconds ) to drain
  + But, at t = 1 ms for the sender to send the whole packet
  + Thus, there is no underflow.
* Given this, what is the smallest elasticity buffer size?
  + Sender has packet that is 10 kbits
  + Receiver is manufactured with buffer that is 20 bits
  + Receive strategy:
    - 1) receive first 20 bits
    - 2) then tell its client to start reading
  + The buffer gets the first 20 bit after roughly 2 microsecond
  + The sender needs another ( 1 ms - 2 microsecond = 998 microsecond ) to send the whole packet
  + So there is ( 20 bit - 998 microsecond \* 20 Kbits /s ~ 0.02 bits)
  + It works!
* So the smallest elasticity buffer X is such that
* Case 2: Overflow
  + Sender has packet that is 10 kbits
  + Receiver is manufactured with buffer that is 10 kbits
  + Strategy:
    - 1) Receiver tells its client to start reading asap
    - 2) Sender needs to wait between packets - **Inter-packet gap**
* Case 2: Overflow with Inter-packet gap
  + Sender has packet that is 10 kbits
  + Receiver is manufactured with buffer that is 10 kbits
  + Strategy:
    - 1) Receiver tells its client to start reading asap
  + When the packet is done, there is ( ~ 1 millisecond \* 20 Kbits/s = 20 bits) in the buffer size.
* So 20 bit buffer works for both cases with a different strategy. So this policy would work for both the cases:
  + Sender has packet that is 10 kbit
  + Receiver with buffer that is 40 bits
  + Receiver Strategy:
    - If there are greater than or equal to 20 bits in the buffer
    - Then tell client to start reading
  + And the buffer size is ( 2 \* the number of smallest buffer size we calculated at the end of Case 1) =
* Last piece: let’s go back to the minimum number of inter packet gap: the inter packet gap needs to be enough for the go back to a buffer size of from either buffer size 0 or buffer size . Therefore, we need at least
* **Summary**: given a clock rate and a clock tolerance , a max transmission unit , we have elasticity buffer size and the proposed strategy works if the inter-packet gap is at least . (We could also motivate these results with a little intuition: the more accurate clocks are, the smaller will be, in other words, we require smaller inter-packet gap and elasticity buffer size for more accurate clocks. If the clock is ideal (no error at all), inter-packet gaps and elasticity buffers are not needed, and the senders and the receivers can just operate at their own clocks. )