CompSci 356: Computer Network Architectures

Lecture 6: Link layer: Error Detection and Reliable transmission
Ref. Chap 2.4, 2.5

Xiaowei Yang
xwy@cs.duke.edu
Overview

• Link layer functions
  – Encoding
    • NRZ, NRZI, Manchester, 4B/5B
  – Framing
    • Byte-oriented, bit-oriented, time-based
    • Bit stuffing
  – Error detection
    • Parity, checksum, CRC
  – Reliability
    • FEC, sliding window
Link-layer functions

- Most functions are completed by adapters
  - Encoding
  - Framing
  - Error detection
  - Reliable transmission
Error detection

• Error detection code adds redundancy
  – Analogy: sending two copies
  – Parity
  – Checksum
  – CRC

• Error correcting code
Cyclic Redundancy Check

• Cyclic error-correcting codes
• High-level idea:
  – Represent an n+1-bit message with an n degree polynomial $M(x)$
  – Divide the polynomial by a degree-k divisor polynomial $C(x)$
  – k-bit CRC: remainder
  – Send Message + CRC that is dividable by $C(x)$
Polynomial arithmetic modulo 2

- B(x) can be divided by C(x) if B(x) has higher degree
- B(x) can be divided once by C(x) if of same degree
  - x^3 + 1 can be divided by x^3 + x^2 + 1
  - The remainder would be 0*x^3 + 1*x^2 + 0*x^1 + 0*x^0 (obtained by XORing the coefficients of each term)
- Remainder of B(x)/C(x) = B(x) – C(x)
- Subtraction is done by XOR each pair of matching coefficients
CRC algorithm

1. Multiply M(x) by x^k. Add k zeros to Message. Call it T(x)

2. Divide T(x) by C(x) and find the remainder

3. Send P(x) = T(x) – remainder
   - Append remainder to T(x)
   - P(x) dividable by C(x)
An example

8-bit msg
- 10011010

Divisor (3bit CRC)
- 1101

Msg sent: 10011010101
How to choose a divisor

- Arithmetic of a finite field
- Intuition: unlikely to be divided evenly by an error
- Corrupted msg is $P(x) + E(x)$
- If $E(x)$ is single bit, then $E(x) = x^i$
- If $C(x)$ has the first and last term nonzero, then detects all single bit errors
- Find $C(x)$ by looking it up in a book
Overview

• Link layer functions
  – Encoding
    • NRZ, NRZI, Manchester, 4B/5B
  – Framing
    • Byte-oriented, bit-oriented, time-based
    • Bit stuffing
  – Error detection
    • Parity, checksum, CRC
  – Reliability
    • FEC, sliding window
Reliable transmission

• What to do if a receiver detects bit errors?
• Two high-level approaches
  – Forward error correction (FEC)
  – Retransmission
    • Acknowledgements
      – Can be “piggybacked” on data packets
    • Timeouts
    • Also called Automatic repeat request (ARQ)
Stop-and-wait

- Send one frame, wait for an ack, and send the next

- Retransmit if times out

- Note in the last figure (d), there might be confusion: a new frame, or a duplicate?
Sequence number

- Add a sequence number to each frame to avoid the ambiguity
Stop-and-wait drawback

• Revisiting bandwidth-delay product
  – Total delay/latency = transmission delay + propagation delay + queuing
    • Queuing is the time packet sent waiting at a router’s buffer
    • Will revisit later (no sweat if you don’t get it now)
Delay * bandwidth product

For a 1Mbps pipe, it takes 8 seconds to transmit 1MB. If the link latency is less than 8 seconds, the pipe is full before all data are pumped into the pipe.

For a 1Gbps pipe, it takes 8 ms to transmit 1MB.
Stop-and-wait drawback

- A 1Mbps link with a 100ms two-way delay (round trip time, RTT)
- 1KB frame size
- Throughput = \( \frac{1\text{KB}}{1\text{KB}/1\text{Mbps} + 100\text{ms}} = 74\text{Kbps} \ll 1\text{Mbps} \)
- Delay \( \times \) bandwidth = 100Kb
- So we could send \( \sim 12 \) frames before the pipe is full!
- Throughput = \( \frac{100\text{Kb}}{1\text{KB}/1\text{Mbps} + 100\text{ms}} = 926\text{Kbps} \)
Sliding window

- Key idea: allowing multiple outstanding (unacked) frames to keep the pipe full
Sliding window on sender

• Assign a sequence number (SeqNum) to each frame

•Maintains three variables
  – Send Window Size (SWS)
  – Last Ack Received (LAR)
  – Last Frame Sent (LFS)

• Invariant: LFS – LAR ≤ SWS
• **Sender actions**
  – When an ACK arrives, moves LAR to the right, opening the window to allow the sender to send more frames
  – If a frame times out before an ACK arrives, retransmit
Sliding window on receiver

• Maintains three window variables
  – Receive Window Size (RWS)
  – Largest Acceptable Frame (LAF)
  – Last frame received (LFR)

• Invariant
  – $\text{LAF} - \text{LFR} \leq \text{RWS}$
When a frame with SeqNum arrives
- Discards it if out of window
  - Seq ≤ LFR or Seq > LAF
- If in window, decides what to ACK
  - Cumulative ack
  - Acks SeqNumToAck even if higher-numbered packets have been received
  - Sets LFR = SeqNumToAck - 1, LAF = LFR + RWS
  - Updates SeqNumToAck

Ex: LFR = 5; RWS = 4, frames 7, 8, 6 arrives
Finite sequence numbers

• Things may go wrong when SWS=RWS, SWS too large
• Example
  – 3-bit sequence number, SWS=RWS=7
  – Sender sends 0, …, 6; receiver acks, expects (7,0, …, 5), but all acks lost
  – Sender retransmits 0,…,6; receiver thinks they are new
• SWS < (MaxSeqNum+1)/2
  – Alternates between first half and second half of sequence number space as stop-and-wait alternates between 0 and 1
Multiple functions of the sliding window algorithm

• Remark: perhaps one of the best-known algorithms in computer networking

• Multiple functions
  – Reliable deliver frames over a link
  – In-order delivery to upper layer protocol
  – Flow control
    • Not to over un a slow slower
  – Congestion control (later)
    • Not to congest the network
Other ACK mechanisms

• NACK: negative acks for packets not received
  – unnecessary, as sender timeouts would catch this information

• SACK: selective ACK the received frames
  – + No need to send duplicate packets
  – - more complicated to implement
  – Newer version of TCP has SACK
Concurrent logical channels

• A link has multiple logical channels
• Each channel runs an independent stop-and-wait protocol
• + keeps the pipe full
• - no relationship among the frames sent in different channels: out-of-order
Exercise

- Delay: 100ms; Bandwidth: 1Mbps; Packet Size: 1000 Bytes; Ack: 40 Bytes
- Q: the smallest window size to keep the pipe full?
• Window size = largest amount of unacked data

• How long does it take to ack a packet?
  – $\text{RTT} = 100 \text{ ms} \times 2 + \text{transmission delay of a packet (1000B)} + \text{transmission delay of an ack (40B)}$
  – $\approx 208 \text{ms}$

• How many packets can the sender send in an RTT?
  – $1\text{Mbps} \times \frac{208\text{ms}}{8000 \text{ bits}} = 26$

• Roughly 13 packets in the pipe from sender to receiver, and 13 acks from receiver to sender
Summary

• CRC

• Reliability
  – FEC, sliding window

• Next
  – Multi-access link