CSE/EE 461 – Lecture 4

Error Detection and Correction

David Wetherall
djw@cs.washington.edu

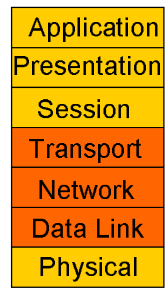
Last Time

- Different media have different properties that affect higher layer protocols
- To send messages we must solve the problems of clock recovery and framing

This Lecture

1. Latency. How long does it take to send messages across a link?

2. Error detection and correction. How do we detect and correct when messages are garbled during transmission?



1. Message Latency

- How long does it take to send a message?

- Two terms:
  - Propagation delay = distance / speed of light in media
  - Transmission delay = message (bits) / rate (bps)
- In effect, slow links stretch bits out in time/space
- Later we will see queuing delay …

One-way Latency Examples

- Either a slow link or long wire makes for large latency
- Dialup with a modem:
  - D = 10ms (say), R = 56Kbps, M = 1000 bytes
  - Latency = 10ms + (1024 x 8)/(56 x 1024) sec = 153ms!
- Cross-country with T3 line:
  - D = 50ms, R = 45Mbps, M = 1000 bytes
  - Latency = 50ms + (1024 x 8) / (45 x 1000000) sec = 50ms!

Terminology

- Latency is typically the one way delay over a link
  - But latency and delay are generic terms
- The round trip time (RTT) is twice the one way delay
  - Measure of how long to signal and get a response
- An important metric is the bandwidth-delay product
  - Measure of how much data can be in-flight at a time

2. Error Detection/Correction

- Noise can flip some of the bits we receive
  - We must be able to detect when this occurs
- Basic approach: add redundant data
  - Error detection codes allow errors to be recognized
  - Error correction codes allow some errors to be repaired too

Motivating Example

- Let's just send two copies. Differences imply errors.
- Question: Can we do any better?
  - With less overhead
  - Catch more kinds of errors
- Answer: Yes – stronger protection with fewer bits
  - But we can't catch all inadvertent errors, nor malicious ones
- We will look at basic block codes
  - K bits in, N bits out is a (N,K) code
  - Simple, memoryless mapping

Detection versus Correction

- Two strategies to correct errors:
  - Error correcting codes and retransmissions (ARQ)
- Question: Which should we choose?
- Answer: Depends on errors and cost of recovery!
- Example: Message with 1000 bits, Prob(bit error) 0.001
  - If random errors, most messages likely to have an error
  - If bursts of 1000 errors typical, only 1 or 2 per 1000 messages
- Satellites, real-time media tend to use error correction
  - Called Forward Error Correction (FEC) in some contexts
- Retransmissions typically at the frame/packet level

The Hamming Distance

- To detect/correct bit errors, errors must not turn one valid codeword into another valid codeword
- Hamming distance is the number of bit differences
  - E.g, code 000 for 0, 111 for 1, Hamming distance is 3
  - This is the number of errors needed to turn one into the other
  - Hamming distance of the entire code is minimum of pairs
- For code with distance d+1:
  - d errors can be detected, e.g, 001, 010, 110, 101, 011
- For code with distance 2d+1:
  - d errors can be corrected, e.g., 001 → 000 iff one error

Parity

- Start with n bits and add another so that the total number of 1s is even (even parity)
  - e.g. 0110010 → 01100101
  - Easy to compute as XOR of all input bits
- Will detect an odd number of bit errors
  - But not an even number
- Does not correct any errors

2D Parity

- Add parity row/column to array of bits
- Detects all 1, 2, 3 bit errors, and many errors with >3 bits.
- Corrects all 1 bit errors

```
          ↓
0101001   1
1101001   0
1011110   1
0001110   1
0110100   1
1011111   0

→ 1111011 0 ←
          ↑
```

Checksums

- Used in Internet protocols (IP, ICMP, TCP, UDP)
- Basic Idea: Add up the data and send it along with sum
- Algorithm:
  - checksum is the 1s complement of the 1s complement sum of the data interpreted 16 bits at a time (for 16-bit TCP/UDP checksum)
- 1s complement: flip all bits to make number negative
  - Consequence: adding requires carryout to be added back

Checksum Example

- Message is e3 4f 23 96 44 27 99 f3
- 2s complement sum is 1e4ff
- So 1s complement sum is e500 (add back carry)
- So checksum is 1aff (flip all bits)
- Advantages: fast to compute; incremental
- Disadvantage: error detection isn't strong

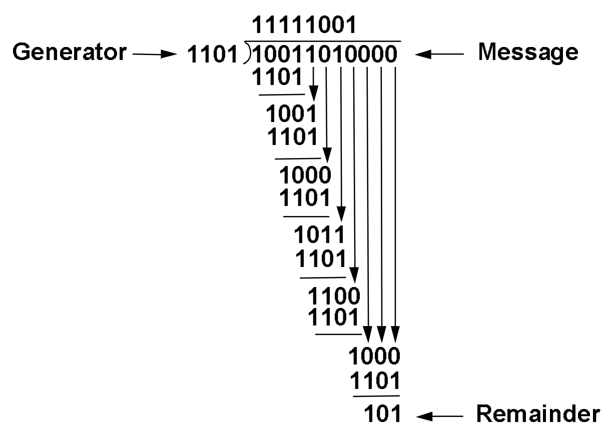
CRCs (Cyclic Redundancy Check)

- Stronger protection than checksums
  - Used widely in practice, e.g., Ethernet CRC-32
  - Easily implemented in hardware (XORs and shifts)
- Algorithm: Given n bits of data, generate a k bit check sequence that gives a combined n + k bits that are divisible by a pre-defined number
- Based on mathematics of finite fields
  - "numbers" correspond to polynomials, use modulo arithmetic
  - e.g, interpret 10011010 as $x^7 + x^4 + x^3 + x^1$

CRC Example

- How do we generate the check sequence?
  - Have our message, e.g., 10011010 (m=8)
  - Have the CRC as a divisor polynomial
    e.g., C(x)=1110 ($x^3 + x^2 + x^1$; k=3)
  - Want to make m + k bits divisible by this divisor …
  - First, add k zeros to end of message
  - Then, divide by C(x) to find the remainder …

Example – Polynomial Division



Example – Remainder to CRC

- So we see the remainder is 101
- Thus the zero extended message – 101 must be evenly divisible by C(x)!
- So perform the subtraction to discover the check bits
  - Subtraction/addition is XOR in modulo 2 arithmetic
  - E.g., we get 10011010000 - 101 = 1011010101
  - The check bits are 101
- Finally, the message we send is 10011010101

How is C(x) Chosen?

- Mathematical properties:
  - All 1-bit errors if non-zero x^k and x^0 terms
  - All 2-bit errors if C(x) has a factor with at least three terms
  - Any odd number of errors if C(x) has (x + 1) as a factor
  - Any burst error < k bits
- There are standardized polynomials of different degree that are known to catch many errors

Standard CRC Polynomials

- CRC-8 100000111
- CRC-10 11000110011
- CRC-12 110000000111
- CRC-16 1000100000100000
- CRC-32 100000100110000010001110110110111

Reed-Solomon / BCH Codes

- Reed-Solomon codes developed to protect data on magnetic disks
- Used for CDs and cable modems too
- Property: 2t redundant bits can correct <= t errors
- Mathematics somewhat more involved …

Key Concepts

- Message latency is the sum of the propagation and transmission delays
- Redundant bits are added to messages to detect, and in some cases correct, transmission errors.