Computer Networks

CMP2205

Lecture 6

Error Detection and Correction

- *Add control information to the original data being transmitted.*
- *Error detection: enough info to detect error.*
  - *Need retransmissions.*
- *Error correction: enough info to detect and correct error.*
  - *A.k.a., forward error correction (FEC).*

Why?

- *Error detection versus error correction.*
- *Cost-efficiency?*
  - *Environment.*
  - *Application.*

What's an error?

- *Frame =* **m** *data bits +* **r** *bits for error control.*
  - *$n = m + r$.*
- *Given the original frame* **f** *and the received frame* **f',** *how many corresponding bits differ?*
  - *Hamming distance (Hamming, 1950).*

Hamming Distance: Examples

```
00110
00111
-----
00001      Hamming distance one

00110
10001
1  111                     four
```

•

Hamming Distance

- *If f and f' are Hamming distance of* **d** *apart, there needs to be* **d** *single-bit errors to convert f to f'.*
- *Error detecting/correcting properties of a code depend on the code's Hamming distance.*
  - *To detect* **d** *errors, need code with Hamming distance* **d+1**.
    - *Need d+1 single-bit errors to change a valid f to a valid f'.*
    - *If receiver sees invalid f', it knows an error occurred.*

Parity Bit

- *Simple error detecting code.*
- *Even- or odd parity.*
- *Example:*
  - *Transmit 1011010.*
  - *Add parity bit 1011010 0 (even parity) or 1011010 1 (odd parity).*
- *Code with single parity bit has Hamming distance of 2!*
  - *Any single bit error produces frame with wrong parity.*

Error Correcting Codes

- *To correct d errors, need 2d+1distance code.*
  - *Code words are 2d+1 apart.*
  - *With d changes, original frame is closer than any other valid frame.*

Error Correction: Example

- *Suppose code with 4 valid words: 0000000000, 0000011111, 1111100000, 1111111111.*
  - *Hamming distance is 5.*
  - *Possible to correct double errors.*
  - *Example: If 0000000111 arrives at receiver, receiver assumes original must have been 0000011111.*
  - *But if triple error changes 0000000000 into 0000000111, not able to correct it properly.*

Hamming Code

- *Bits in positions that are power of 2 are check bits. The rest are data bits.*
- *Each check bit used in parity (even or odd) computation of collection of bits.*
  - *Example: check bit in position 11, checks for bits in positions, 11 = 1+2+8. Similarly, bit 11 is checked by bits 1, 2, and 8.*

Hamming Code: Example

Char.	7-bit ASCII	Check bits
H	1001000	00110010000
a	1100001	10111001001
m	1101101	11101010101
m	1101101	11101010101
i	1101001	01101011001
n	1101110	01101010110
g	1100111	01111001111
	0100000	10011000000
c	1100011	11111000011
o	1101111	10101011111
d	1100100	11111001100
e	1100101	00111000101

Order of bit transmission

. Hamming codes can only correct single errors.

.

Error Detecting Codes

- *Typically used in reliable media.*
- *Examples: parity bit, polynomial codes (a.k.a., CRC, or Cyclic redundancy Check).*

Polynomial Codes

- *Treat bit strings as representations of polynomials with coefficients 1's and 0's.*
- *K-bit frame is coefficient list of polynomial with k terms (and degree k-1), from x^{k-1} to x^0.*
  - *Highest-order bit is coefficient of x^{k-1}, etc.*
  - *Example: 110001 represents $x^5 + x^4 + x^0$.*
- *Generator polynomial G(x).*
  - *Agreed upon by sender and receiver.*

CRC

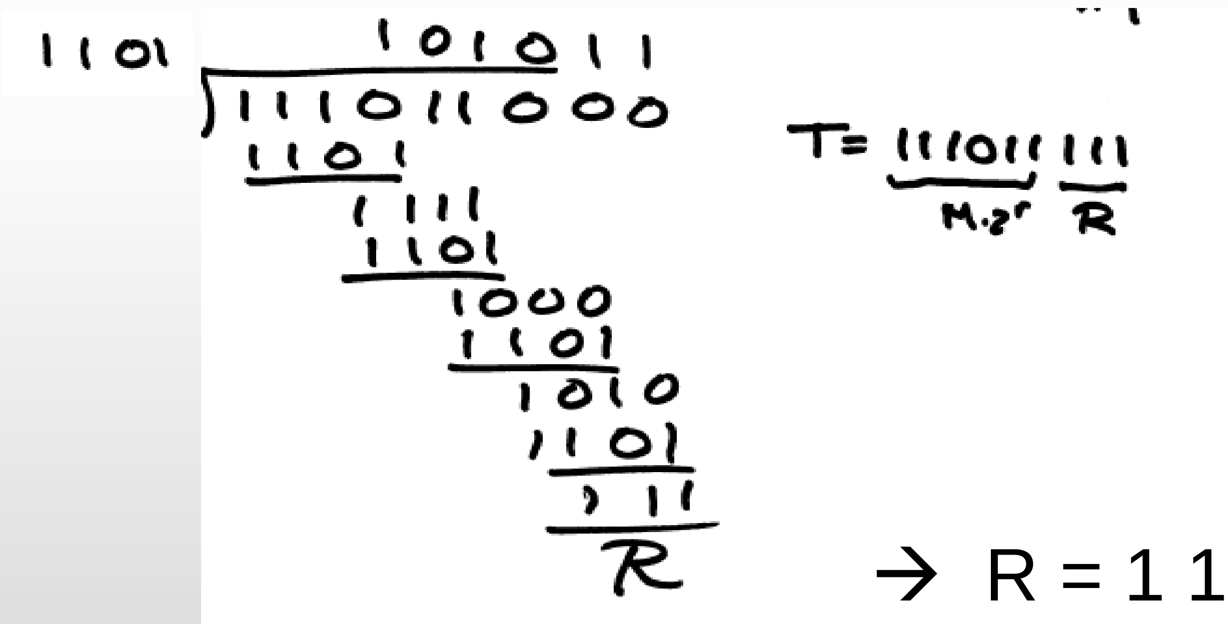
- *Checksum appended to frame being transmitted.*
  - *Resulting polynomial divisible by G(x).*
- *When receiver gets checksummed frame, it divides it by G(x).*
  - *If remainder, then error!*

Cyclic Redundancy Check

At ***Transmitter***, with M = 1 1 1 0 1 1, compute

2^rM= 1 1 1 0 1 1 0 0 0 with G = 1 1 0 1

T = 2^rM + R [note G starts and ends with “1”]

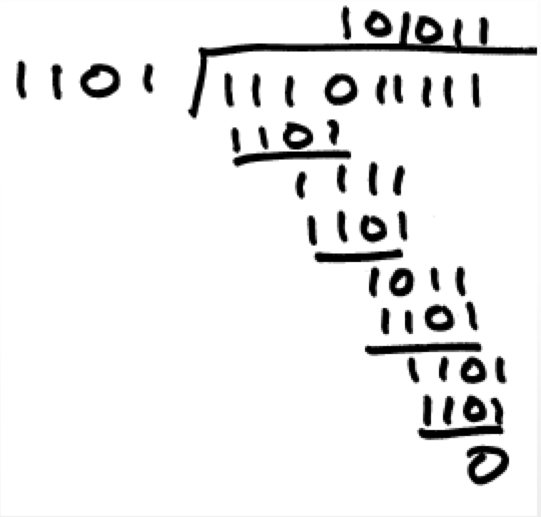


→ R = 1 1 1

Transmit T= 1 1 1 0 1 1 1 1 1

Cyclic Redundancy Check

At the ***Receiver***, compute:

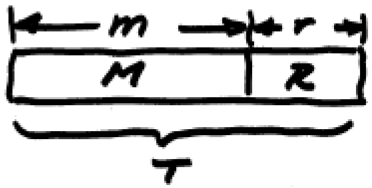


Note remainder = 0 ←→ no errors detected

CRC Performance

- *Errors go through undetected only if divisible by G(x)*
- *With "suitably chosen" G(x) CRC code detects all single-bit errors.*
- *And more…*

More on CRC



M= message (m bits)

T= frame (m+r bits)

R= check sum (sequence
Remainder (r bits)

G= generator polynomial
(r+1 bits)

Want

$$\frac{T}{G} = A + \frac{0}{G} \leftarrow \text{remainder} = 0$$

Recall $\frac{N}{D} = A + \frac{R}{D}$ $\qquad N = AD + R$

$$\frac{N-R}{D} = A = \frac{AD+R-R}{D} = A$$

-

Hamming Code

- *Bits in positions that are power of 2 are check bits. The rest are data bits.*
- *Each check bit used in parity (even or odd) computation of collection of bits.*
  - *Example: check bit in position 11, checks for bits in positions, 11 = 1+2+8. Similarly, bit 11 is checked by bits 1, 2, and 8.*

Hamming Code: Example

Char.	7-bit ASCII	Check bits
H	1001000	00110010000
a	1100001	10111001001
m	1101101	11101010101
m	1101101	11101010101
i	1101001	01101011001
n	1101110	01101010110
g	1100111	01111001111
	0100000	10011000000
c	1100011	11111000011
o	1101111	10101011111
d	1100100	11111001100
e	1100101	00111000101

Order of bit transmission

. Hamming codes can only correct single errors.

. But, to correct bursts of errors, send column by column.

Error Detecting Codes

- *Typically used in reliable media.*
- *Examples: parity bit, polynomial codes (a.k.a., CRC, or Cyclic redundancy Check).*

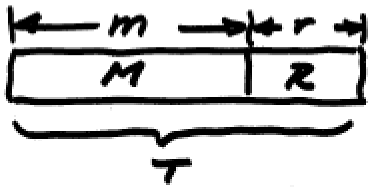
Polynomial Codes

- *Treat bit strings as representations of polynomials with coefficients 1's and 0's.*
- *K-bit frame is coefficient list of polynomial with k terms (and degree k-1), from x^{k-1} to x^0.*
  - *Highest-order bit is coefficient of x^{k-1}, etc.*
  - *Example: 110001 represents $x^5 + x^4 + x^0$.*
- *Generator polynomial G(x).*
  - *Agreed upon by sender and receiver.*

CRC

- *Checksum appended to frame being transmitted.*
  - *Resulting polynomial divisible by G(x).*
- *When receiver gets checksummed frame, it divides it by G(x).*
  - *If remainder, then error!*

More on CRC



M = message (m bits)
T = frame (m+r bits)
R = check sum (sequence) Remainder (r bits)
G = generator polynomial (r+1 bits)

Want

$$\frac{T}{G} = A + \frac{0}{G} \leftarrow \text{remainder} = 0$$

Recall $\frac{N}{D} = A + \frac{R}{D}$ $\quad N = AD + R$

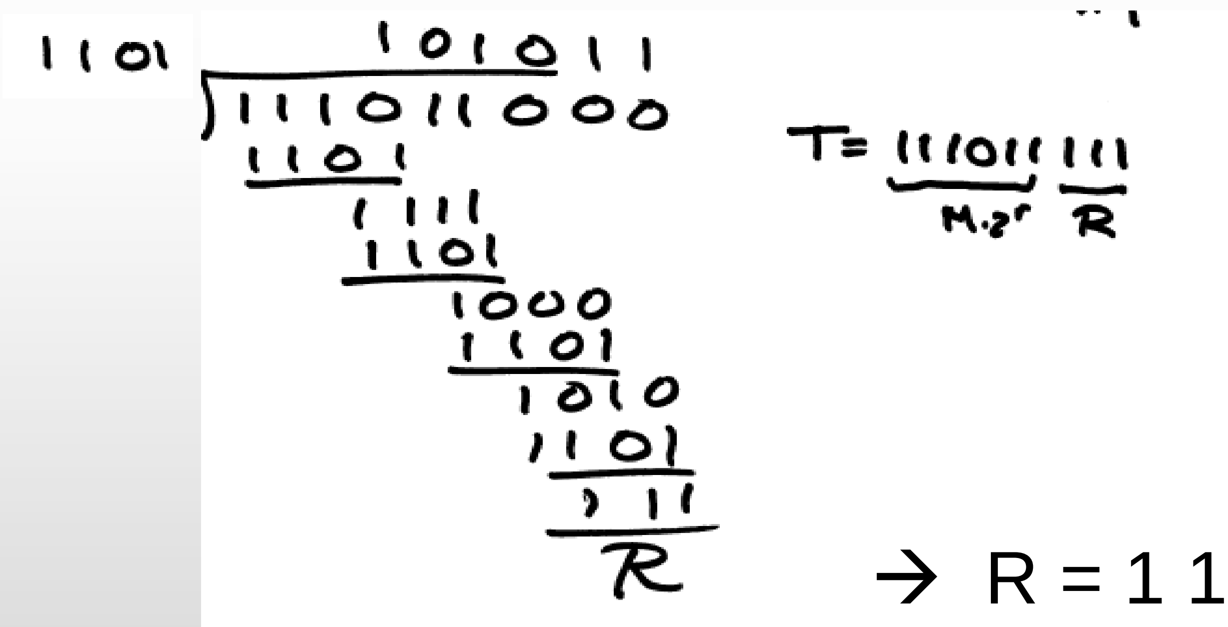
$$\frac{N-R}{D} = A = \frac{AD+R-R}{D} = A$$

•

Cyclic Redundancy Check

At ***Transmitter***, with M = 1 1 1 0 1 1, compute
2^rM= 1 1 1 0 1 1 0 0 0   with G = 1 1 0 1

T = 2^rM + R   [note G starts and ends with "1"]

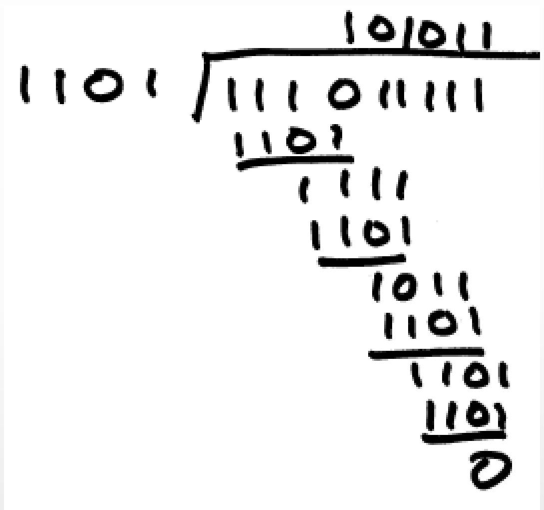


→ R = 1 1 1

Transmit   T= 1 1 1 0 1 1 1 1 1

Cyclic Redundancy Check

At the ***Receiver***, compute:



Note remainder = 0 ←→ no errors detected

CRC Performance

- *Errors go through undetected only if divisible by G(x)*
- *With "suitably chosen" G(x) CRC code detects all single-bit errors.*
- *And more…*

Flow + Error Control

- *How do Layer 2 protocols implement them?*

- *What's a frame?*

F	H	Payload	T	F

. What's F?
. What's in T?
. What's in H?

Header and Trailer

- *Trailer typically has checksum.*
  - *How is it used/processed?*
- *Header has: type, sequence number, and ack.*

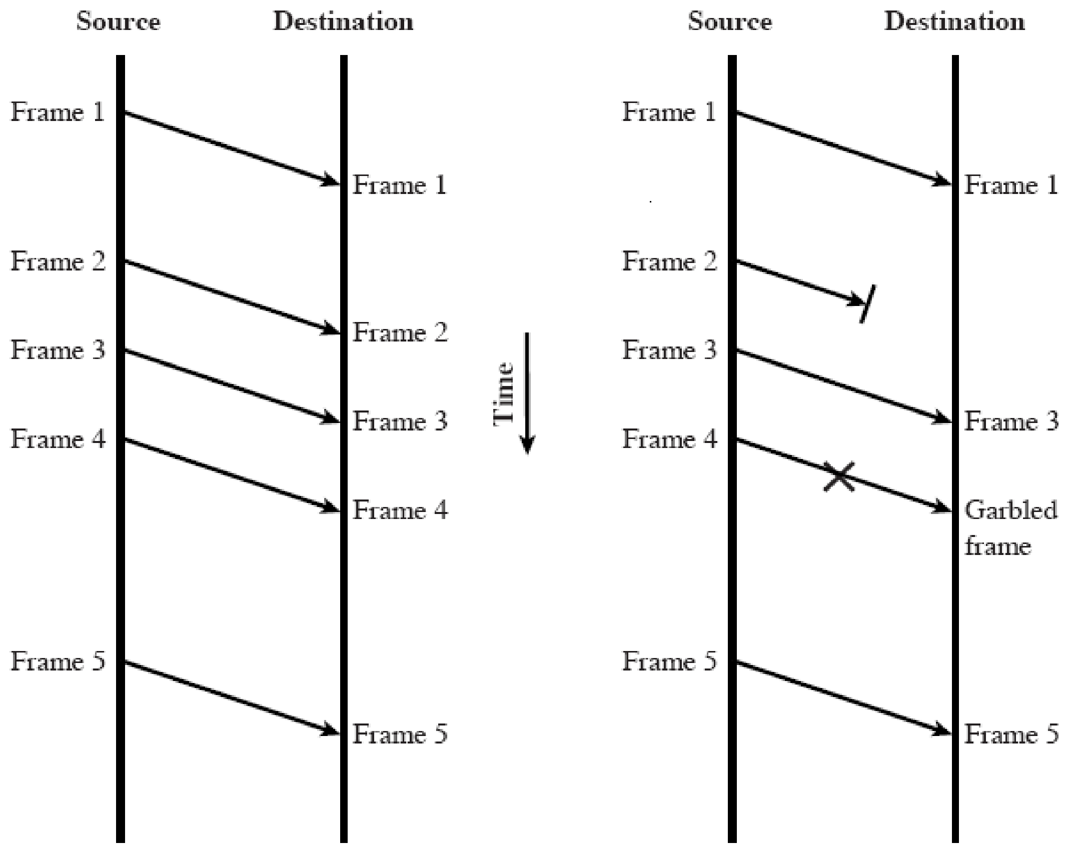
Stop-and-Wait

- *Simplest form of flow control.*
- *How does it work? (assume error-free channel)*
  - *(1) Send 1 frame;*
  - *(2) Wait for ACK.*
  - *(3) Go to 1.*

Stop-and-Wait: Pros and Cons

- *Very simple!*
- *But, poor link utilization.*
  - *High data rates.*
  - *Long propagation delay.*

Noisy Channels



(a) Error-free transmission

(b) Transmission with losses and errors

From Stallings: Data and Computer Communications

Figure 7.1 Model of Frame Transmission

Stop-and-Wait in Noisy Channels

- *Need timers, retransmissions, and duplicate detection.*
- *Use sequence numbers.*
  - *Why?*
  - *Distinguish frames.*
  - *How large (e.g., in number of bits) are sequence numbers?*

ARQ Protocols

- *Automatic Repeat Request.*
  - *Protocols that wait for ACK before sending more data.*
- *ACKs now are used for flow AND error control.*
- *What can happen?*
  - *At receiver: frame arrives correctly, frame arrives damaged, frame does not arrive.*
  - *At sender: ACK arrives correctly, ACK arrives damaged, ACK does not arrive.*

ARQ Protocols

- *Sender:*
  - *Send frame 0.*
  - *Start timer.*
  - *If ACK 0, arrives, send frame 1.*
  - *If timeout, re-send frame 0.*

- *Receiver:*
  - ***Waits for frame.*
  - *If frame arrives, check if correct sequence number.*
  - *Then send ACK for that frame.*
  - *Go to (**)*

Simplex versus Duplex Transmission

- *Simplex:*
  - *Send data in one channel and control in another channel.*
- *Duplex:*
  - *Send data and control on the same chanel.*

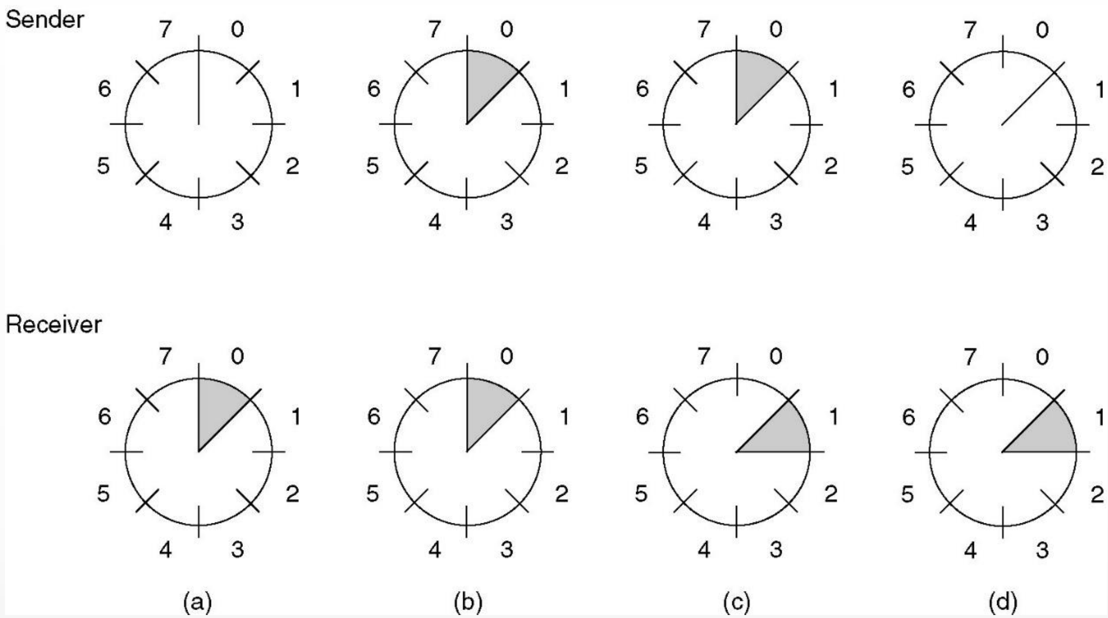
Can we do better?

- *Can we do better?*
  - *Piggybacking.*
  - *Bi-directional transmission.*
  - *Wait for data packet and use that to piggyback the ACK.*
  - *Use ACK field: only a few additional bits in the header.*
- *But, how long should Layer 2 wait to send an ACK?*
  - *ACK timers!*

Sliding Window Protocols

- *Window: number of "outstanding" frames at any given point in time.*
- *Every ACK received, window slides…*

Sliding Window Example



- *A sliding window of size 1, with a 3-bit sequence number.(a) Initially; (b) After the first frame has been sent; (c) After the first frame has been received;(d) After the first acknowledgement has been received.*