Fundamentals of Computer Networks
ECE 478/578

Lecture #3: Encoding and Framing
Instructor: Loukas Lazos
Dept of Electrical and Computer Engineering
University of Arizona

Direct Link (Point-to-Point) Networks
Simplest network possible, consisting of two hosts and one link
Look into different functions at the link level
Encoding  
Framing  
Error detection/correction  
Reliable delivery  
Media access control

Typical Communication System
Compresses a stream of bits  
Adds redundancy to detect/correct errors at the receiver (bits are mapped to codewords)  
Scrambles the bits of consecutive codewords to spread burst errors  
Converts a stream of bits to analog signals modulated for transmission over a medium
Physical Medium

Signals travel through the medium and represent bits

*Nyquist Bandwidth*: Given an error-free medium of bandwidth $B$, the highest signal (symbol) rate (bauds) that can be carried is $2B$

Bauds to bits: $R = 2B \times \log_2 M$ bits, $M$: signal levels

Shannon’s Capacity Theorem

$C = B \log_2 (1+S/N)$

Defines the upper bound on the link capacity $C$ in Hz

Can be used to evaluate the “error-free” bandwidth of a line

---

Shannon’s Theorem: Example

Voice-grade phone line: $B=3,300$–$300$ Hz = 3 KHz

Typical SNR = 30 dB, where

$\text{dB} = 10 \times \log_{10}(S/N)$

For 30 dB $\Rightarrow S/N = 1,000$

$C = 3,000 \times \log_2(1+1,000) = 30$ Kbps

*Higher bandwidth* $B$ (in Hz), *higher capacity*

*Higher S/N*, higher capacity

---

Shannon’s Theorem: Example 2

Can the signal be weaker than noise?

Assume capacity of 50 Kbps over 1MHz bandwidth

The required SNR would then be

$C = B \log_2(1+S/N)$

$S/N = 2^{C/B} - 1 = 0.035$ or -14.5 dB

Think of spread spectrum communications

Transmit a weak signal over a large bandwidth
Relating Nyquist and Shannon

Assume $B = 1 \text{ MHz}$ and $\text{SNR} = 24 \text{ dB}$. How many signal levels are required to achieve the max rate?

$24 \text{ dB} = 10 \log_{10} (S/N) \Rightarrow S/N = 10^{2.4} = 251$

Using Shannon’s formula

$C = 10^6 \times \log_2(1+251) = 10^6 \times 8 = 8 \text{ Mbps} \text{ (Theoretical limit)}$

Using Nyquist’s theorem

$C = 2B \log_2M \Rightarrow 8 \times 10^6 = 2 \times 10^6 \times \log_2M \Rightarrow M = 16$

Encoding

Map binary bits into signals

Example: Low signal represents a 0, high signal represents a 1

NonReturn-to-Zero (NRZ)

Problem: Long periods of silence (zero) or high signals are possible

Baseline wander (receiver loses track of reference sig)
Clock recovery (receiver loses clock synchronization)

Encoding: More Schemes

NRZ Inverted (NRZI): Switch from current state to represent a 1
Manchester: XOR the bit stream with the clock
4B/5B Encoding Scheme

Encode 4-bit symbols into 5-bit codes

$2^4$ symbols must be mapped to $2^4$ codewords out of the possible $2^5$

Each codeword has no more than one starting zero, and no more than two trailing zeros

No more than 3-consecutive zeros

Then use NRZI to solve the consecutive 1s problem

80% efficiency (1 bit is overhead)

---

Example of 4B/5B Encoding

<table>
<thead>
<tr>
<th>4-bit data symbol</th>
<th>5-bit code</th>
<th>4-bit data symbol</th>
<th>5-bit code</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>11110</td>
<td>1000</td>
<td>10010</td>
</tr>
<tr>
<td>0001</td>
<td>01001</td>
<td>1001</td>
<td>10011</td>
</tr>
<tr>
<td>0010</td>
<td>10100</td>
<td>1010</td>
<td>10110</td>
</tr>
<tr>
<td>0011</td>
<td>10101</td>
<td>1011</td>
<td>10111</td>
</tr>
<tr>
<td>0100</td>
<td>01010</td>
<td>1100</td>
<td>11010</td>
</tr>
<tr>
<td>0101</td>
<td>01011</td>
<td>1101</td>
<td>11011</td>
</tr>
<tr>
<td>0110</td>
<td>01110</td>
<td>1110</td>
<td>11100</td>
</tr>
<tr>
<td>0111</td>
<td>01111</td>
<td>1111</td>
<td>11101</td>
</tr>
</tbody>
</table>

---

Framing

The process of grouping bits into frames (messages or packets)

Typically implemented by the network adaptor

Why frames?
Byte-Oriented Framing

BISYNC: Binary synchronous communication
Frame is a collection of bytes
Need to indicate the beginning and end of a frame
Sentinel characters are used

<table>
<thead>
<tr>
<th>8</th>
<th>8</th>
<th>8</th>
<th>Header</th>
<th>STX</th>
<th>ETX</th>
<th>8</th>
<th>CRC</th>
</tr>
</thead>
</table>

SYN: Synchronization character
SOH: Start of header
STX, ETX: Start of text, End of text
CRC: Cyclic redundancy check

Problem with Byte-oriented Framing

ETX may occur in the payload
Precede it with a DLE (data-link-escape) character
Problem propagates, precede DLE with another DLE (extra overhead)
Point-to-Point (PPP) protocol used by IP

<table>
<thead>
<tr>
<th>8</th>
<th>8</th>
<th>8</th>
<th>Protocol</th>
<th>Payload</th>
<th>Checksum</th>
<th>16</th>
<th>8</th>
</tr>
</thead>
</table>

STX: 0111110
Payload: 1,500 bytes
Checksum: 2 or 4 bytes
Overhead: 8/1508 = 0.5%

Byte-counting Framing

Include the # of bytes in the frame as a field in the header

Digital Data Communications Protocol (DDCMP)

<table>
<thead>
<tr>
<th>8</th>
<th>8</th>
<th>8</th>
<th>Count</th>
<th>Header</th>
<th>Body</th>
<th>16</th>
<th>CRC</th>
</tr>
</thead>
</table>

Count: Specifies # of bytes in the body
CRC ensures that count field is not corrupted

---
Bit-oriented Framing

High-Level Data Link Control (HDLC)

- Beginning/end of frame, flag: 01111110
- Instead of inserting bytes do bit stuffing
- Sender adds a 0 after five consecutive 1s
- Receiver removes zero after five 1s

Example of Bit-stuffing

**Sender**

11111101111111111110111110

0 0 0 0 0

**Receiver**

111111X1011111X11111X1011111X0

Length of frame

- Variable, depends on the data
- We can calculate and optimize the overhead of bit stuffing

Maximum Frame Size

Let each frame contain $V$ overhead bits

Let a message of $M$ bits be broken into frames of size $K_{max}$

- # of packets: $\left\lceil \frac{M}{K_{max}} \right\rceil$

The total # of bits for all frames: $M + \left\lceil \frac{M}{K_{max}} \right\rceil V$

If $K_{max}$, $V$, # of frames ↑, and overhead also ↑

If # of frames ↑, then each frame must be processed then more processing delay at each host

**Increase frame length as much as possible**
Maximum Frame Size - Pipelining

Why not make $K_{\text{max}} = M$?

Pipelining delay: frame must be received before forwarding

![Diagram of source and destination with total delay and pipelining](image)

Tradeoff between $K_{\text{max}}$ and $V$

Let message be of length $M$, frame size $K_{\text{max}}$ and overhead $V$, over a path of $j$ equal capacity links.

First packet hop over $(j-1)$ switches + entire message transmission.

Total number of bit transmission times

$$TC = (K_{\text{max}} + V)(j-1) + M + \left\lceil \frac{M}{K_{\text{max}}} \right\rceil V$$

Expected value over message lengths

$$E[TC] = (K_{\text{max}} + V)(j-1) + E[M] + \left( E[M] / K_{\text{max}} + 1/2 \right) V$$

Minimize $E[TC]$ at

$$K_{\text{max}} = \sqrt{E[M]V / (j-1)}$$