Rechnernetze – Computer Networks

Lecture 3: Direct Link Networks
Forward Error Correction and Error Detection

Prof. Dr.-Ing. Markus Fidler

Institute of Communications Technology
Leibniz Universität Hannover

April 19, 2024

Repetition

Transmission media: bandwidth-limited channel with bandwidth B

- signals, e.g., square pulses, are distorted
- maximal symbol rate that can be reconstructed: $2B$ [Nyquist]
- so far no errors, e.g., due to noise → today

Encoding of bits

- line coding
- modulation

Framing

Outline

Channel Capacity

Forward Error Correction

- Triple Modular Redundancy
- Linear Block Codes

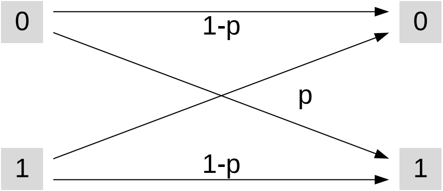
Error Detection

- Internet Checksum
- Cyclic Redundancy Check

Binary symmetric channel

Given information bits are transmitted over an error-prone channel

- bit error probability $p \leq 1$

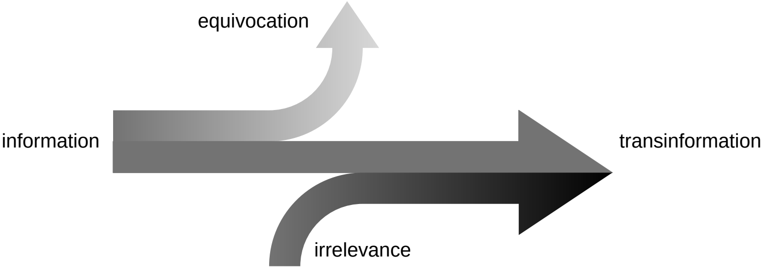


- correct transmission $\mathsf{P}[0|0] = \mathsf{P}[1|1] = 1 - p$
- erroneous transmission $\mathsf{P}[0|1] = \mathsf{P}[1|0] = p$

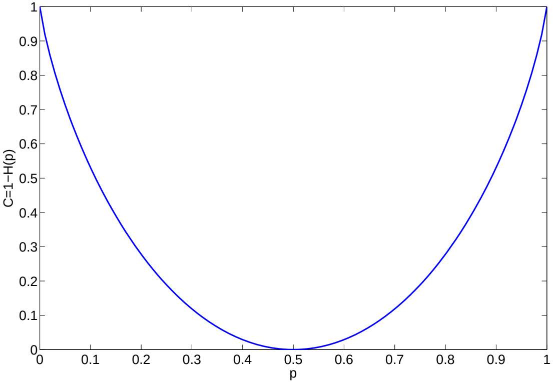
Transinformation

Without formal definition:

- equivocation: amount of information lost
- irrelevance: amount of (useless) information added
- transinformation: amount of the original information received



Binary symmetric channel: channel capacity



The channel capacity C is the amount of transinformation that is feasible at most.

Channel coding theorem

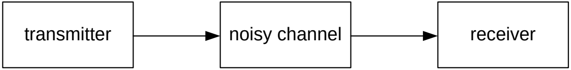
The connection of the channel capacity and the maximum rate at which data can be transmitted without error is [Shannon 1948]:

Theorem (Channel coding)

Consider a discrete, memoryless channel with channel capacity C and a source with bit rate R. If $R < C$ there exists a code such that the data generated by the source can be transmitted with arbitrarily small probability of error.

The proof of the channel coding theorem assumes codes with infinite length. It does not specify how to construct such codes.

AWGN channel



Transmissions over a channel are subject to noise. A useful model is the additive white Gaussian noise (AWGN) channel

- linear addition of noise
- noise is white i.e.
  - it applies to all frequency bands
  - it has a constant spectral density independent of frequency
- noise samples have a Gaussian distribution

Gaussian noise has many natural causes such as thermal noise.

Noise distorts transmitted symbols causing symbol resp. bit errors at the receiver resulting in a loss of information.

Shannon limit

Consider a bandwidth-limited channel that is subject to thermal (white Gaussian) noise. Shannon's limit states that

$$C = B \log_2 \left(1 + \frac{S}{N}\right)$$

where

- $C =$ theoretical limit of the data rate [bit/s]
- $B =$ channel bandwidth [Hz]
- $S =$ signal power [W]
- $N =$ noise power [W]
- $S/N =$ signal-to-noise ratio (in dB $10 \log_{10}(S/N)$)

Example: Shannon limit

Shannon limit

$$C = B \log_2 \left(1 + \frac{S}{N}\right)$$

Example

- analogue telephone line: $B = 3$ kHz
- signal-to-noise ratio: 30 dB, i.e. $S/N = 1000$
- Shannon capacity: 30 kb/s

Shannon limit continued

Given bits of duration $T = 1/C$ the energy per bit is $E_b = S/C$.

The noise can be expressed as $N = N_0 \cdot B$ where N_0 is the spectral noise power density [W/Hz] and B is the bandwidth [Hz].

By insertion

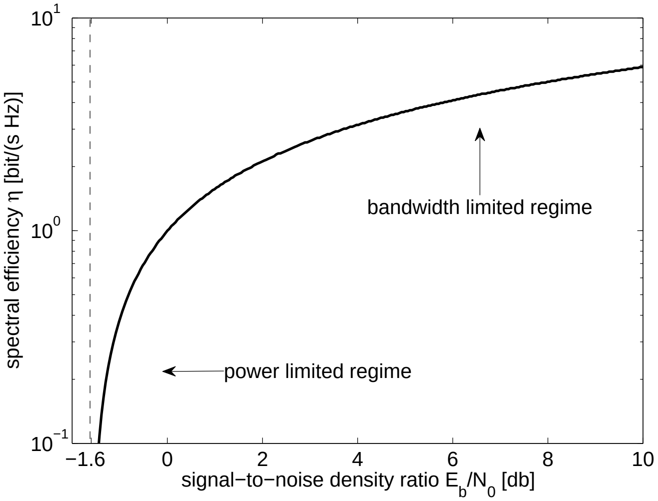
$$\frac{C}{B} = \log_2\left(1 + \frac{E_b}{N_0}\frac{C}{B}\right)$$

where

- $E_b/N_0 =$ signal-to-noise density ratio
- $C/B = \eta$ spectral efficiency [bit/(s·Hz)]

For $\eta \to 0$ it follows that $E_b/N_0 = \ln(2) \approx 0.7 \approx -1.6$ dB, i.e. information transfer is impossible if $E_b/N_0 < 0.7$ even if the bandwidth is arbitrarily large.

Shannon limit continued



Outline

Channel Capacity

Forward Error Correction

- Triple Modular Redundancy
- Linear Block Codes

Error Detection

- Internet Checksum
- Cyclic Redundancy Check

Error correction

Typical bit error rates (BER) for different media are

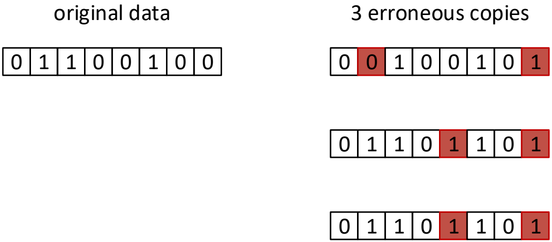
- 10^{-2} for mobile/wireless communications
- 10^{-5} for copper cable
- 10^{-12} for fiber optic cable

Error control coding comprises

- forward error correction (FEC)
  - uses codes that can correct certain transmission errors
- automatic repeat request (ARQ) (next lecture)
  - uses codes that can detect certain transmission errors
  - erroneous data are discarded and retransmitted

Triple modular redundancy

Send three independent copies of the data.



original data

0	1	1	0	0	1	0	0

3 erroneous copies

0	0	1	0	0	1	0	1
0	1	1	0	1	1	0	1
0	1	1	0	1	1	0	1

At any bit position triple modular redundancy

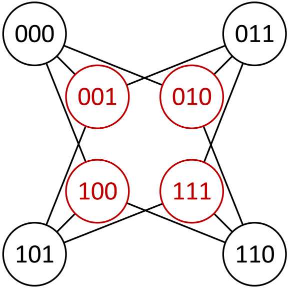
- can correct 1 bit error (majority decision)
- can detect up to 2 bit errors (but not correct)
- cannot detect 3 bit errors

The approach is used, e.g., in Bluetooth, because of its simplicity. It is, however, not efficient.

Parity bits

The parity bit is the modulo-2 sum of each data word.

data word	codeword
00	00 0
01	01 1
10	10 1
11	11 0



Valid codewords differ in (have a distance of) 2 bit positions:

- can detect a single bit error
- cannot correct bit errors (to be able to correct bit errors, a larger distance is required)

Linear block codes

Block coding

- data words of fixed length (e.g. 4 bit) are used
- each data word is mapped to a unique (longer) codeword (e.g. 7 bit) thereby creating redundancy
- the codewords are transmitted over the channel
- the receiver compares the received codeword with the set of possible codewords
  - if the received codeword matches a possible codeword the receiver decodes the codeword
  - otherwise it detects a transmission error and performs error correction if possible

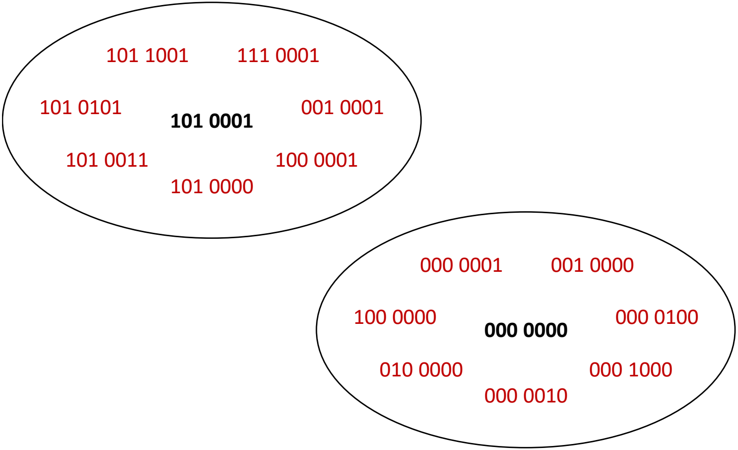
Example: (7,4)-Hamming code

Table: (7,4)-Hamming code

data word	codeword	data word	codeword
0000	000 0000	0001	101 0001
0010	111 0010	0011	010 0011
0100	011 0100	0101	110 0101
0110	100 0110	0111	001 0111
1000	110 1000	1001	011 1001
1010	001 1010	1011	100 1011
1100	101 1100	1101	000 1101
1110	010 1110	1111	111 1111

Note that the codewords differ in at least three bit positions.

Intuition



Each valid codeword can be surrounded by a protecting shell of invalid codewords that each differ in one single bit position.

Hamming distance

The Hamming distance specifies the number of bit positions at which two codewords differ, e.g. the two codewords 0000000 and 1010001 have a Hamming distance of $d = 3$.

The minimal Hamming distance of a code is the minimum of the distance between any two codewords. In case of the (7,4)-Hamming code it is $d_{\min} = 3$.

- the code can detect $d_{\min} - 1 = 2$ bit errors
- the code can correct $\lfloor (d_{\min} - 1)/2 \rfloor = 1$ bit errors

Modulo-2 arithmetic

Coding frequently uses modulo-2 arithmetic (Galois field 2)

- addition $\oplus$ coincides with the XOR operation
- multiplication $\odot$ coincides with the AND operation

$\oplus$	0	1
0	0	1
1	1	0

$\odot$	0	1
0	0	0
1	0	1

The operations $\oplus$ and $\odot$ are (among other properties)

- commutative $\oplus$: $a \oplus b = b \oplus a$
- commutative $\odot$: $a \odot b = b \odot a$
- associative $\oplus$: $(a \oplus b) \oplus c = a \oplus (b \oplus c)$
- associative $\odot$: $(a \odot b) \odot c = a \odot (b \odot c)$
- distributive: $a \odot (b \oplus c) = a \odot b \oplus a \odot c$

Generator matrix

The (7,4)-Hamming code (and other block codes as well) is determined by its generator matrix $\mathbf{G}$.

$$\mathbf{G} = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Codewords can be generated by modulo-2 matrix multiplication

$$\mathbf{v} = \mathbf{u} \odot \mathbf{G}$$

where $\mathbf{u}$ is the data word and $\mathbf{v}$ the corresponding codeword.

Example

(7,4)-Hamming code for the data word (0 1 1 1)

$$\begin{pmatrix}0 & 1 & 1 & 1\end{pmatrix} \odot \begin{pmatrix}1 & 1 & 0 & 1 & 0 & 0 & 0\\ 0 & 1 & 1 & 0 & 1 & 0 & 0\\ 1 & 1 & 1 & 0 & 0 & 1 & 0\\ 1 & 0 & 1 & 0 & 0 & 0 & 1\end{pmatrix} = \begin{pmatrix}0 & 0 & 1 & 0 & 1 & 1 & 1\end{pmatrix}$$

Example

(7,4)-Hamming code for the data word (0 1 1 1)

$$\begin{pmatrix} 0 & 1 & 1 & 1 \end{pmatrix} \odot \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}$$

The data part can generally be read from the last four bits of the codeword, since the last four columns of the generator matrix are the identity matrix. Codes with this property are called systematic.

- the first three bits of the codeword are parity bits
- the last four bits of the codeword are the data bits

Syndrome decoding

Given a systematic code the receiver can itself construct the parity bits for comparison and error detection.

In case of the (7,4)-Hamming code the codeword
$\mathbf{v} = \begin{pmatrix} v_0 & v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \end{pmatrix}$ consists of

- parity bits v_0, v_1, v_2
- data bits v_3, v_4, v_5, v_6

Bit v_0 is computed from the generator matrix as $v_3 \oplus v_5 \oplus v_6$, hence

$$v_0 \oplus v_3 \oplus v_5 \oplus v_6 = s_0$$

equals zero if no transmission error occurred. Similar conditions apply for bits v_1 and v_2.

Syndrome decoding

The parity check can be formulated in matrix notation as

$$\mathbf{s} = \mathbf{v} \odot \mathbf{H}$$

where the parity check matrix is

$$\mathbf{H} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}$$

In case of error free transmission the syndrome is $\mathbf{s} = \begin{pmatrix} 0 & 0 & 0 \end{pmatrix}$.

Example

Continuing the previous example the codeword $\begin{pmatrix}0 & 0 & 1 & 0 & 1 & 1 & 1\end{pmatrix}$ is checked

$$\begin{pmatrix}0 & 0 & 1 & 0 & 1 & 1 & 1\end{pmatrix} \odot \begin{pmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\\ 1 & 1 & 0\\ 0 & 1 & 1\\ 1 & 1 & 1\\ 1 & 0 & 1\end{pmatrix} = \begin{pmatrix}0 & 0 & 0\end{pmatrix}$$

and found to be correct since the syndrome is zero.

Venn diagram, unique syndroms

Codeword

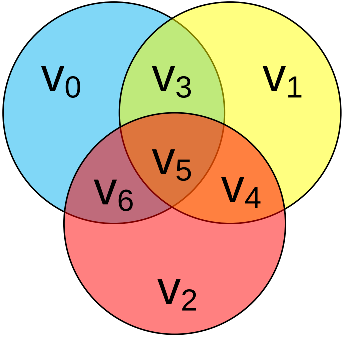
- parity bits v_0, v_1, v_2
- data bits v_3, v_4 ,v_5, v_6

Generation of parity bits

- $v_0 = v_3 \oplus v_5 \oplus v_6$
- $v_1 = v_3 \oplus v_4 \oplus v_5$
- $v_2 = v_4 \oplus v_5 \oplus v_6$

Single bit errors (examples)
Received codeword $(r_0, .., r_6)$

- r_0 erroneous $\Rightarrow$
  $r_0 \neq r_3 \oplus r_5 \oplus r_6$
- r_3 erroneous $\Rightarrow$
  $r_0 \neq r_3 \oplus r_5 \oplus r_6$ and
  $r_1 \neq r_3 \oplus r_4 \oplus r_5$



Syndrome table

The (7,4)-Hamming code has the nice property that any single bit error results in a unique syndrome such that single bit errors can be corrected.

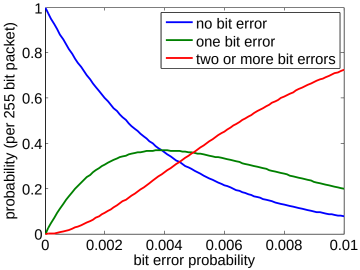
Denote $\mathbf{r}$ the codeword at the receiver. The syndrome table provides the position of single bit errors

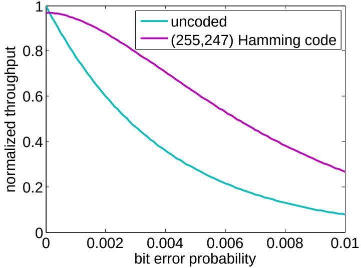
bit error position	r_0	r_1	r_2	r_3	r_4	r_5	r_6
syndrome $\mathbf{s}$	100	010	001	110	011	111	101

The attempt to correct a bit error fails, however, if more than one bit is erroneous. In this case:

- the syndrome may but need not be unequal zero
- if nonzero the syndrome cannot indicate whether there are one or more bit errors

Coding gain





Example: no coding versus $(255, 247)$ Hamming code

- bit errors are statistically independent
- bit errors per packet are binomial distributed
- packet size: 255 bit
- in case of Hamming code: 8 parity bits
- Hamming distance $d = 3$: corrects one bit error per packet

Outline

Channel Capacity

Forward Error Correction
- Triple Modular Redundancy
- Linear Block Codes

Error Detection
- Internet Checksum
- Cyclic Redundancy Check

Frame check sequence

- some bit errors are corrected by FEC, e.g. the Hamming-(7,4) code can correct 1 bit error in any 4 bit data word
- bit errors occur, however, typically in bursts, i.e. several consecutive bits are erroneous
  - errors likely affect many bits of a frame (or none)
  - such burst errors are more difficult to correct
  - instead, erroneous frames can be discarded and resent
  - need strong error detection codes for entire frames
- data frames have a trailer that contains parity bits, aka frame check sequence (FCS), for error detection

Internet checksum

Internet protocols (IP, UDP, TCP) use a simple checksum to detect bit errors:

- at the sender
  - the frame is divided into 16 bit words
  - the 16 bit checksum field is initialized using all 0s
  - the one's complement sum of all 16 bit words of the frame including the checksum field is computed
  - the one's complement of the result is stored as the checksum
- at the receiver
  - the one's complement sum of all 16 bit words of the frame including the checksum field is computed
  - the one's complement of the result is
    - all 0s if no bit error occurred
      (also possible if multiple bit errors occurred)
    - otherwise an error has occurred

One's complement

One's complement

- negation of all bits of a binary number, e.g. 0100 has one's complement 1011
- used for negative numbers in binary algebra, e.g. 0100 is decimal +4 and 1011 is decimal -4
- addition uses carry around, i.e. a carry over at the MSB is added as LSB, e.g.
  - 0100 + 1011 = 1111 is decimal -0
  - 0101 + 1011 = 1 0000 with carry around 0001 is decimal 1
- carry around addition is used for checksum computation
- finally the one's complement of the one's complement sum of all words is stored as checksum

Internet checksum: example 1

Sender		Receiver	
0100	word one	0100	word one
0011	word two	0011	word two
1011	word three	1011	word three
0000	initial checksum	1100	checksum

Sender		Receiver	
0100	word one	0100	word one
0011	word two	0011	word two
0111	sum	0111	sum
1011	word three	1011	word three
1 0010	sum	1 0010	sum
0011	carry around	0011	carry around
0000	initial checksum	1100	checksum
0011	sum	1111	sum
1100	one’s complement	0000	one’s complement

Internet checksum: example 2

Receiver, 1 bit error

0100	word one
0010	word two
1011	word three
1100	checksum

0100	word one
0010	word two
0110	sum
1011	word three
1 0001	sum
0010	carry around
1100	checksum
1110	sum
0001	one's complement

Receiver, 2 bit errors

0100	word one
0010	word two
1011	word three
1101	checksum

0100	word one
0010	word two
0110	sum
1011	word three
1 0001	sum
0010	carry around
1101	checksum
1111	sum
0000	one's complement

Polynomial codes

Polynomial codes are more efficient than the Internet checksum. They also have nice and provable algebraic properties

- bit strings are viewed as polynomials with coefficients 0 and 1
- a k bit string represents the list of coefficients for a polynomial with k terms
- the polynomial is of degree $k-1$ containing terms $x^{k-1} \dots x^0$
- e.g. the string 100101 has polynomial $x^5 + x^2 + x^0$

Arithmetic is done modulo-2

- no carries in case of addition: $1 \oplus 1 = 0$
- no borrows in case of subtraction: $0 \ominus 1 = 1$
- hence both addition and subtraction are identical: XOR

Cyclic redundancy check

Generation of the checksum

- sender and receiver agree on a generator polynomial $G(x)$ of degree r, e.g. the CRC-4 polynomial with $r = 4$ is $x^4 + x^1 + x^0$
- the data frame of m bits is viewed as polynomial $M(x)$
  - the sender appends r zeros at the end of $M(x)$
  - the resulting polynomial $x^r M(x)$ has $m + r$ bits
- the sender divides $x^r M(x)$ modulo-2 by $G(x)$
- the sender subtracts the division remainder from $x^r M(x)$ resulting in the polynomial $T(x)$
- from the construction of $T(x)$ it follows that $T(x)$ is divisible by $G(x)$ (with zero remainder)

Cyclic redundancy check

Error detection from the checksum

- the sender transmits the polynomial $T(x)$ (divisible by $G(x)$)
- during transmission errors occur written as polynomial $E(x)$
  - the error $E(x)$ is added modulo-2 to $T(x)$
  - e.g. an error 00001001 indicates bit errors at position 3 and 0
- the receiver receives polynomial $T(x) \oplus E(x)$
  - e.g. given 10100111 is transmitted 10101110 is received
- the receiver divides $T(x) \oplus E(x)$ modulo-2 by $G(x)$
  - $(T \oplus E)/G = T/G \oplus E/G$ where T/G has zero remainder
  - a non-zero remainder of E/G indicates transmission errors

Cyclic redundancy check example

- CRC-4 generator polynomial $G(x) = x^4 + x^1 + x^0$ that is 10011
- data frame 11010001 with appended zeros 110100010000

$$\begin{array}{cccccccccccc}
1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 & 1 & \downarrow & & & & & & \\
\hline
 & 1 & 0 & 0 & 1 & 0 & & & & & &
\end{array}$$

Cyclic redundancy check example

- CRC-4 generator polynomial $G(x) = x^4 + x^1 + x^0$ that is 10011
- data frame 11010001 with appended zeros 110100010000

$$\begin{array}{cccccccccccc}
1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 & 1 & \downarrow & & & & & & \\
\hline
 & 1 & 0 & 0 & 1 & 0 & & & & & & \\
 & 1 & 0 & 0 & 1 & 1 & \downarrow & \downarrow & \downarrow & \downarrow & & \\
\hline
 & & & & & 1 & 0 & 1 & 0 & 0 & &
\end{array}$$

Cyclic redundancy check example

- CRC-4 generator polynomial $G(x) = x^4 + x^1 + x^0$ that is 10011
- data frame 11010001 with appended zeros 110100010000

$$\begin{array}{cccccccccccc}
1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 & 1 & \downarrow & & & & & & \\
\hline
 & 1 & 0 & 0 & 1 & 0 & & & & & & \\
 & 1 & 0 & 0 & 1 & 1 & \downarrow & \downarrow & \downarrow & \downarrow & & \\
\hline
 & & & & & 1 & 0 & 1 & 0 & 0 & & \\
 & & & & & 1 & 0 & 0 & 1 & 1 & \downarrow & \downarrow \\
\hline
 & & & & & & & 1 & 1 & 1 & 0 & 0
\end{array}$$

Cyclic redundancy check example

- CRC-4 generator polynomial $G(x) = x^4 + x^1 + x^0$ that is 10011
- data frame 11010001 with appended zeros 110100010000

$$
\begin{array}{cccccccccccc}
1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 & 1 & \downarrow & & & & & & \\
\hline
 & 1 & 0 & 0 & 1 & 0 & & & & & & \\
 & 1 & 0 & 0 & 1 & 1 & \downarrow & \downarrow & \downarrow & \downarrow & & \\
\hline
 & & & & & 1 & 0 & 1 & 0 & 0 & & \\
 & & & & & 1 & 0 & 0 & 1 & 1 & \downarrow & \downarrow \\
\hline
 & & & & & & & 1 & 1 & 1 & 0 & 0 \\
 & & & & & & & 1 & 0 & 0 & 1 & 1 \\
\hline
 & & & & & & & & 1 & 1 & 1 & 1
\end{array}
$$

- remainder 1111
- frame to be transmitted 110100011111

Cyclic redundancy check example

- CRC-4 generator polynomial $G(x) = x^4 + x^1 + x^0$ that is 10011
- received frame 110100011111

$$
\begin{array}{cccccccccccc}
1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\
1 & 0 & 0 & 1 & 1 & \downarrow & & & & & & \\
\hline
 & 1 & 0 & 0 & 1 & 0 & & & & & & \\
 & 1 & 0 & 0 & 1 & 1 & \downarrow & \downarrow & \downarrow & \downarrow & & \\
\hline
 & & & & & 1 & 0 & 1 & 1 & 1 & & \\
 & & & & & 1 & 0 & 0 & 1 & 1 & \downarrow & \downarrow \\
\hline
 & & & & & & & 1 & 0 & 0 & 1 & 1 \\
 & & & & & & & 1 & 0 & 0 & 1 & 1 \\
\hline
 & & & & & & & & 0 & 0 & 0 & 0
\end{array}
$$

- remainder is $0000 \Rightarrow$ no error detected
- not a proof that the transmission was error-free

Cyclic redundancy check example

- CRC-4 generator polynomial $G(x) = x^4 + x^1 + x^0$ that is 10011
- transmitted frame $11010\mathbf{0011}111$
- burst error $00000\mathbf{1111}000$ of length 4
- received frame $11010\mathbf{1100}111$

$$
\begin{array}{cccccccccccc}
1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \\
1 & 0 & 0 & 1 & 1 & \downarrow & & & & & & \\
\hline
 & 1 & 0 & 0 & 1 & 1 & & & & & & \\
 & 1 & 0 & 0 & 1 & 1 & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \\
\hline
 & & & & & & 1 & 0 & 0 & 1 & 1 & \\
 & & & & & & 1 & 0 & 0 & 1 & 1 & \downarrow \\
\hline
 & & & & & & & & 0 & 0 & 0 & 1
\end{array}
$$

- remainder is $0001 \Rightarrow$ error detected

Cyclic redundancy check example

- CRC-4 generator polynomial $G(x) = x^4 + x^1 + x^0$ that is 10011
- transmitted frame $11010\mathbf{00111}11$
- burst error $00000\mathbf{10011}00$ of length 5
- received frame $11010\mathbf{10100}11$

$$
\begin{array}{cccccccccccc}
1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \\
1 & 0 & 0 & 1 & 1 & \downarrow & & & & & & \\
\hline
 & 1 & 0 & 0 & 1 & 1 & & & & & & \\
 & 1 & 0 & 0 & 1 & 1 & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\
\hline
 & & & & & & & 1 & 0 & 0 & 1 & 1 \\
 & & & & & & & 1 & 0 & 0 & 1 & 1 \\
\hline
 & & & & & & & & 0 & 0 & 0 & 0
\end{array}
$$

- remainder is $0000 \Rightarrow$ error not detected

Error detection capability

CRC detects any error pattern $E(x)$ that is not divisible by $G(x)$

- if $G(x)$ has degree r and a non-zero x^0 term any burst error up to length $k \leq r$ can be detected
  - a burst error of length k means that two or more errors are spread over at most k successive bits
  - a burst error of length k can be written as $E(x) = x^i(x^{k-1} + \cdots + 1)$
    - x^i is the position of the burst error
    - $x^{k-1} + \cdots + 1$ is the burst error pattern
  - need to prove that $x^i(x^{k-1} + \cdots + 1)$ is not a multiple of $G(x)$
    - since $G(x)$ has a term x^0 the term x^i can never be canceled
    - the degree of $G(x)$ is larger than the degree of $(x^{k-1} + \cdots + 1)$ such that $(x^{k-1} + \cdots + 1)$ cannot be divisible by $G(x)$

Some standard polynomials

Some examples for the use of standard polynomials

- ATM: $x^8 + x^2 + x^1 + x^0$
- USB: $x^{16} + x^{15} + x^2 + x^0$
- HDLC, Bluetooth: $x^{16} + x^{12} + x^5 + x^0$
- IEEE 802.3 Ethernet: $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x^1 + x^0$

Efficient implementations

- in hardware using shift registers with XOR gates
- in software using lookup tables
  - e.g. indexed by byte words
  - contains the remainder of the respective index

Literature

- Martin Werner, Information und Codierung, Vieweg+Teubner, 2. Auflage, 2008.