Data Communications

Error Detection and Correction

Error

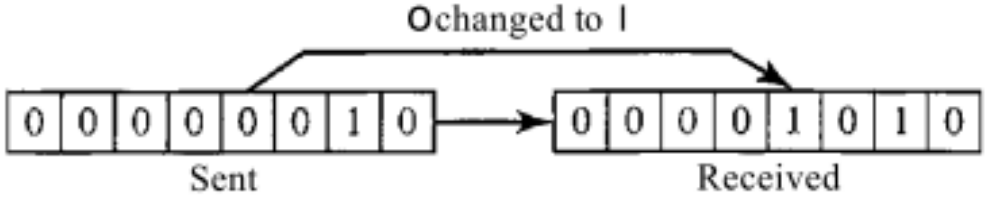
- Data can be corrupted during transmission.
- These errors should be detected and corrected.
- Error is the conversion of a 0 bit to 1 bit (or 1 bit to 0 bit) because of change of the shape of the signal

Types of Errors

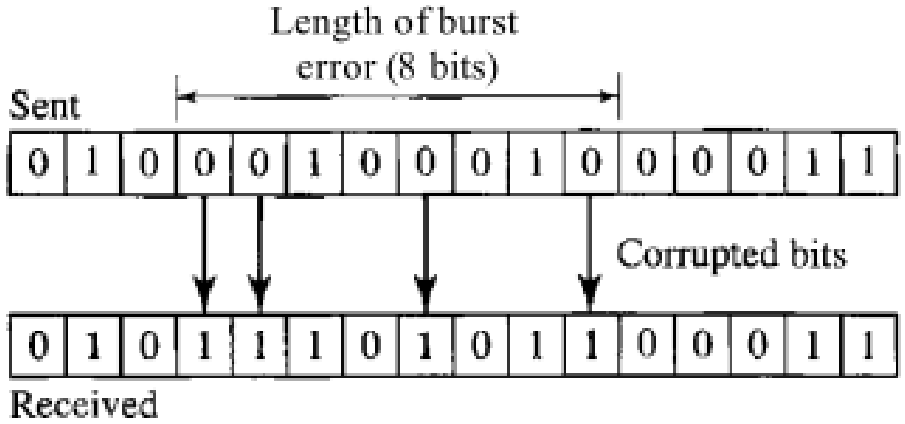
- Single-Bit Error
  - The term single-bit error means that only 1 bit of a given data unit (such as a byte, character, or packet) is changed from 1 to 0 or from 0 to 1.
- Burst Error
  - The term burst error means that 2 or more bits in the data unit have changed from 1 to 0 or from 0 to 1.
  - The length of the burst is measured from the first corrupted bit to the last corrupted bit. Some bits in between may not have been corrupted.

Example

Single-bit error



Burst error of length 8



Detection Versus Correction

- In error detection, we are looking only to see if any error has occurred.
- In error correction, we need to know the exact number of bits that are corrupted and more importantly, their location in the message.
- To detect or correct errors, we need to send extra (redundant) bits with data.

Forward Error Correction Versus Retransmission

- Forward error correction is the process in which the receiver tries to guess the message by using redundant bits.
- Correction by retransmission is a technique in which the receiver detects the occurrence of an error and asks the sender to resend the message.

Error Detection

- If the following two conditions are met, the receiver can detect a change in the original codeword.
  - The receiver has (or can find) a list of valid codewords.
  - The original codeword has changed to an invalid one.

Error Detection Using a Single Parity Bit

- The encoder uses a generator that takes a copy of a dataword and generates a parity bit
- The dataword bits and the parity bit create the codeword.
- The parity bit that is added makes the number of 1s in the codeword even {even parity} or odd {odd parity}.

Example

Datawords	*Codewords*	*Datawords*	*Codewords*
0000	00000	1000	10001
0001	00011	1001	10010
0010	00101	1010	10100
0011	00110	1011	10111
0100	01001	1100	11000
0101	01010	1101	11011
0110	01100	1110	11101
0111	01111	1111	11110

Detecting Burst Error

- A single parity can only detect a single error .
- To detect burst error, the burst error is split between several codewords, one error for each codeword.
- In data communications, we normally send a packet or a frame of data.
- To detect a burst error of size N, we need to make N codewords out of our frame.
- Then, instead of sending one codeword at a time, we arrange the codewords in a table and send the bits in the table a column at a time.

Cyclic Redundancy Check (CRC)

- A generating polynomial is defined for codeword generation
- As many zeros as the order of the generating polynomial are concatenated to the end of the data
- Data and the concatenated zeros are divided by the generating polynomial
- Remainder is sent as the CRC code

CRC Decoder

- CRC code is concatenated to the data
- Data and concatenated CRC code are divided by the generating polynomial
- If the remainder is zero then no error
- Otherwise error(s) have happened

Internet Checksum

- Traditionally, the Internet has been using a 16-bit checksum. The sender calculates the checksum by following these steps.
- Sender site:
  - The message is divided into 16-bit words.
  - The value of the checksum word is set to O.
  - All words including the checksum are added using one's complement addition.
  - The sum is complemented and becomes the checksum.
  - The checksum is sent with the data.

Internet Checksum

- Receiver site:
  - The message (including checksum) is divided into 16-bit words.
  - All words are added using one's complement addition.
  - The sum is complemented and becomes the new checksum.
  - If the value of checksum is 0, the message is accepted; otherwise, it is rejected.

Example

1	0	1	3	Carries
4	6	6	F	
7	2	6	F	
7	5	7	A	
6	1	6	E	
0	0	0	0	Checksum (initial)
8	F	C	6	Sum (partial)
			1	
8	F	C	7	Sum
7	0	3	8	Checksum (to send)

1	0	1	3	Carries
4	6	6	F	
7	2	6	F	
7	5	7	A	
6	1	6	E	
7	0	3	8	Checksum (received)
F	F	F	E	Sum (partial)
			1	
F	F	F	F	Sum
0	0	0	0	Checksum (new}

Error Correction using Multiple Parity Bits

- Algorithm (Encoder)
  1. Create a codeword by putting data bits at positions which are not a power of 2.
  2. The positions that are a power of 2 are used for parity bits.
  3. Write each data bit position as a sum of powers of 2.
  4. To compute each parity bit, use the data bits that have the given parity bit position in their sum (found at step 3)

Error Correction using Multiple Parity Bits

- Decoder:
  - Compute the parity bits using the same algorithm as the encoder
  - Compare the computed parity bits with the delivered parity bits
  - If parity bits are different, consider all possible cases of 1 bit error.
  - Correct the corrupted bit by toggling its value.