Data link layer

Jonathan Windle

University of East Anglia

J.Windle@uea.ac.uk

June 6, 2017

Overview I

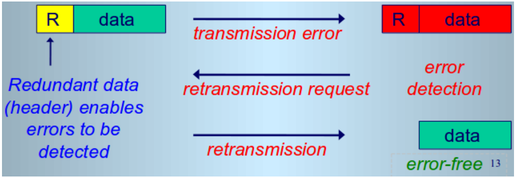
1. Intro
2. Backwards Error Correction
   - Character Parity Bits
   - Longitudinal Parity Bits
   - Cyclic Redundancy Check
3. Forward Error Correction
   - Hamming Codes
     - Simple Hamming Code

Intro

- Main task of data link layer is to take the raw transmission facility, via the physical layer, and transform it into a connection which is free from errors.
- Achieves this by employing error detection and error correction onto the sequence of transmitted data.
- Error detection is used to request re-transmission of corrupt packets.
- Error correction attempts to correct errors without the need for re-transmission.
- Data link layer is also concerned with creating frames from network layer packets.
- Concerned with traffic regulation, to stop fast sender swamping slower receiver.
- Controls data flow on bi-directional channels.

Backwards Error Correction

- Is a two stage process:
  - Error detection
  - Retransmission for correction.
- Common techniques used:
  - Character Parity Bits
  - Longitudinal Parity Bits
  - Cyclic Redundancy Checks

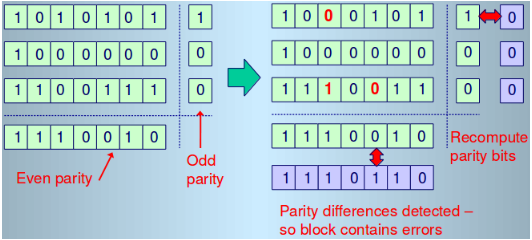


Character Parity Bits

- Utilises an additional bit which is computed and attached to eaach character.
- Either odd parity or even parity.
- Adding parity bit detects errors which affect an odd number of bits. Unfortunately errors typically occur in bursts.

Longitudinal Parity Bits

- Technique also calleda Block Check Sum (BCS).
- A collection of octetes (each using odd/even parity) is treated as a block.
- Bits in position 1 of each octet are checked by a parity bit. Similarly for bits in position 2 checked by parity bit and so on.
- Can detect all 2 and 3 bit errors within the block, but miss some patterns involving 4 bits.



Cyclic Redundancy Check

- The basic principle is that the transmitter treats the message as a binary number and divides the data by a pre-defined constant to give a remainder.
- Remainder is then subtracted from the data and this is then transmitted to the receiver.
- The receiver performs the same division which should result in no remainder.
- If the division does result in a non-zero remainder then the data has been corrupted.

- Polynomial Generators:
  - Bit strings are treated as polynomials with coefficientss of 0 and 1 i.e. $11001 = x^4 + x^3 + x^0$
  - An n bit polynomial generator produces an $n - 1$ bit remainder.

- A 16 bit CRC will identify:
  - All single and double errors
  - All odd numbers of bit corruptions
  - All bursts of errors of length ≤ 16
  - 99.997% of 17 bit burst errors and 99.998% of 18 bit burst errors.

Forward Error Correction

- On unidirectional connections it is not possible to send any feedback from the receiver to the transmitter.
- Makes it impossible to detect errors and ask for retransmission.
- Instead provide sufficient redundant information to enable error correction at receiver.
- Techniques rely on encoding additional redundancy into the data to allow for error correction.
- FEC also important for real-time communication where insufficient time to request retransmission and await new data.
- For any data encoding scheme, the Hamming Distance is the number of bits which differ between valid codes.
- Can use XOR operator to determine the Hamming Distance.
- If two codewords are Hamming Distance d bits apart, then it will require d single bit errors to convert one into another.

Hamming Codes

- Categorised by the number of data bits and parity bits which form the transmitted codeword.
- For example a (7,4) Hamming Code comprises of 7 bits encoding, 4 bits data, hence 3 parity bits.
- More parity bits, gives more chance of detecting bit errors.
- For effective forward error correction a proportion of 50% data 50% parity is necessary.
- This reduces the effective data rate by a factor of 2.
- FEC can be usedon bi-directional channels, butis inefficient for all but the most unreliable channels.

Simple Hamming Code

Creation:

- Bits that are a power of 2 are parity bits. The rest are data bits.
- Determine which data bits are used in each parity bit by using addition.
- For a 7,4 code:
  - P0 = parity(D0,D1,D3)
  - P1 = parity(D0,D2,D3)
  - P2 = parity(D1,D2,D3)

Correction:

- When arrives, the parities are calculated again using the data bits.
- There are then compared to the received parity bits, where K=0:
  - P0 $\neq$ P0' ; error: K=K+1
  - P1 $\neq$ P1' ; error: K=K+2
  - P2 $\neq$ P2' ; error: K=K+4
- If K = 0, codeword is accepted.
- Otherwise K points to the erronous bit in codeword

The End