Error Control
(An Overview)

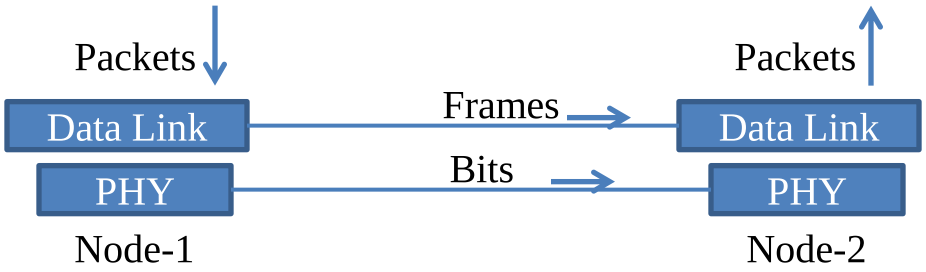
Kameswari Chebrolu

All the figures used as part of the slides are either self created or from the public domain with either 'creative commons' or 'public domain dedication' licensing. The public sites from which some of the figures have been picked include: http://commons.wikimedia.org (Wikipedia, Wikimedia and workbooks); http://www.sxc.hu and http://www.pixabay.com

Recap

errors happen due to noise.
sender sends onething and other end recieves other thing.

- Frame-by-Frame next-hop delivery
- Focus on Error Control
  - Error Detection and recovery



Error Detection

- What cause errors?
  - Distortion of signals due to frequency dependant attenuation, noise (PHY layer)
  - Random single-bit vs Bursty errors

single bit sdiff

a bunch of bits are erros

- Why detect errors?

correctness

if error is at 1st hop itself.... then unnecessary to travel many hops

  - Data fidelity, prevent wastage of resources

noise-->1us
data rate : 1Mbps ----> 1 error   single-bit error
100Mbps=----> 100error --> burst errror of 100 bits

What next?

- After Detection:
  - Drop Frame
    - Higher layers (e.g TCP) will recover or few losses dont hurt applications (e.g. audio)
  - Recover Frame
    - Error Correction: Frame carries enough information to correct errors
    - Retransmission: Receiver signals sender on error, sender retransmits the frame

Error correction vs Retransmission

- Error correction requires more redundant bits per frame than error detection
  - Redundancy bits are sent all the time (every frame)
- Retransmission requires another copy to be transmitted
  - Copy sent only on error

Usage

- Error correction useful when
  - Error rate if high (e.g. wireless)
  - Cost (e.g. latency) of retransmission is too high (e.g. satellite link)

Framework

- Add redundant information to a frame to detect or correct errors
- At Sender: Add **k** bits of redundant data to a **m** bit message
  - k derived from original message through some algorithm
- At Receiver: Reapply same algorithm as sender to detect errors; take corrective action if necessary
- Examples:
  - Detection: $k \ll m$; $k = 32$; $m \sim 12{,}000$ for Ethernet
  - Correction: Code Rate: $m/(m+k)$
    - WiFi code rates range from 1/2 to 5/6

Hamming Distance

m message + k redundant = code word

- Code word: n=m+k bits
- Hamming distance between two codewords: Number of bits they differ in
  - XOR the two codewords
- Example:
  - Codewords: 01110110, 00011101
  - Distance is 5

Hamming Distance of a Code

Legal code words are called 'CODE'

- Number of possible code words is 2^n
- Legal code words = 2^m (determined by the algorithm)

  there are m message bits and 2^m possible messages ..... so and each hs a code word which the algorithm determines.. so total 2^m code words

- Among the list of legal code words, find the smallest hamming distance between two code words
  - This is the hamming distance of a code (=d)

Rules

- The error detection/correction capabilities are a function of the code's hamming distance

errror codes are those present in the remaining 2^n - 2^m words

- Error Detection: Can detect up to d-1 errors
  - To change one codeword to another require atleast d bit changes
- Error Correction: Can correct up to (d-1)/2 errors
  - The received codeword (in error) is still closer to the original codeword than any other codeword

Example

- Repetition code
  - 0 → 000; 1 → 111
  - m=1, k=2; n=3
  - Hamming distance is 3, code rate is 1/3
- Can detect up to 2 errors and correct up to 1 error

n = 3; so 8 possible words..
only 2 legal

Design Considerations of a Code

- Reduce k to achieve high code rate
- For given values of n and k, maximize d
- Easy encoding and decoding
  - Minimal memory and processing time

Focus

- Error Detection
- Reliable Transfer (retransmissions)
- ~~Error Correction~~
  - ~~E.g. Reed-Solomon codes, Convolution codes, Turbo codes~~

Summary

- Important to detect errors in frames
- Error Recovery: FEC or Retransmission
  - Inherent tradeoffs
- Framework (Overview)
  - Hamming distance and error detection/correction capabilities
  - Design considerations
- Going forward: Error detection (in detail)