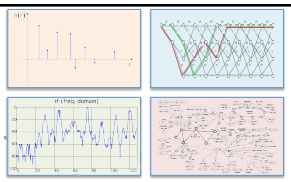


INTRODUCTION TO EECS II

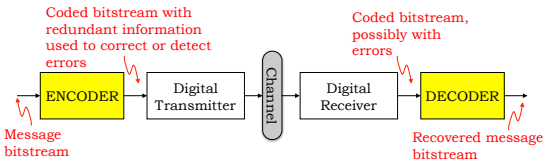
DIGITAL COMMUNICATION SYSTEMS

6.02 Fall 2010
Lecture #6: Coping with Bit Errors

- Channel coding: applying redundancy to detect & correct errors
- Error models: binary symmetric channel & burst error model
- Embeddings and Hamming distance: structural separation
- Parity & linear functions
- Linear (n,k) block codes & Hamming codes

Channel Coding

Our plan to deal with bit errors:



The procedures run at the ENCODER and DECODER constitute **channel coding**.
We will design codes to correct commonly occurring errors, e.g., error bursts of bounded length.
We will also design codes to reduce the effective bit error rate, i.e., the probability of a decoding error

Bit Error Model: Binary Symmetric Channel

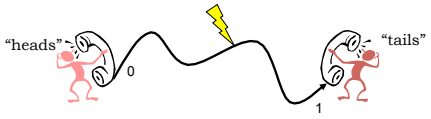
Suppose we wanted to reliably transmit the result of a single coin flip:


Heads: "0"

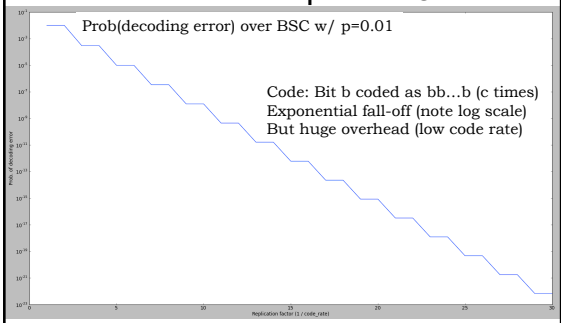
Tails: "1"

This is a prototype of the "bit" coin for the new information economy. Value = 12.5¢

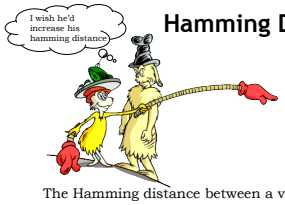
Suppose that during transmission a "0" is turned into a "1" or a "1" is turned into a "0" with probability p.
This is a **binary symmetric channel.**



Performance of Replication Code



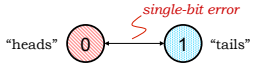
Hamming Distance



The number of bit positions in which the corresponding bits of two encodings of the same length are different

The Hamming distance between a valid binary code word and the same code word with *e* errors is *e*.

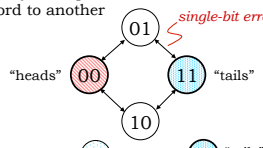
The problem with no coding is that the two valid code words ("0" and "1") also have a Hamming distance of 1. So a single-bit error changes a valid code word into another valid code word…



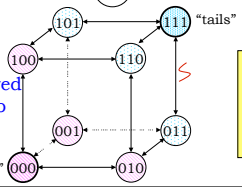
What is the Hamming distance of the replication code?

Embedding for Structural Separation

Encode so that the code words are "far enough" from each other
Likely error patterns shouldn't transform one code word to another



If D is the minimum Hamming distance between code words, we can detect **all** patterns of <= (D-1) bit errors



Decode received code word r to nearest valid code word

If D is the minimum Hamming distance between code words, we can correct **all** patterns of $\left\lfloor \frac{D-1}{2} \right\rfloor$ or fewer bit errors

Linear Block Codes

We can add single-bit error detection to any length code word by adding a *parity bit* chosen to guarantee the Hamming distance between any two valid code words is at least 2.

Parity: addition in GF(2): 0+0=0, 1+0=0+1=1, 1+1=0
multiplication: 0*0=0*1=1*0 =0, 1*1=1

Block code: k message bits encoded to n code bits
Linear block code: sum of any two code words is also a code word
(n,k) code has rate k/n (repetition code is (c,1) code)
Often written as (n,k,d), where d is the Hamming distance of the code

Parity Check

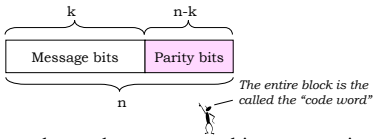
- Add a parity bit to message of length k to make the total number of "1" bits even (aka "even parity").
- If the number of "1"s in the received word and if it's odd, there's been an error. Detects correctly if an odd number of errors have occurred.

0 1 1 0 0 1 0 1 0 0 1 1 → original word with parity
0 1 1 0 0 0 0 1 0 0 1 1 → single-bit error (detected)
0 1 1 0 0 0 1 1 0 0 1 1 → 2-bit error (not detected)

- GF(2) arithmetic: Can count by summing the bits in the word modulo 2 (equivalent to XOR'ing the bits together).

(n,k) Systematic Linear Block Codes

- Split message into *k*-bit blocks
- Add (*n-k*) parity bits to each block, making each block *n* bits long



- Every linear code can be represented in systematic form

If we want to correct single-bit errors, what is the smallest number of parity bits we really need?

How Many Parity Bits?

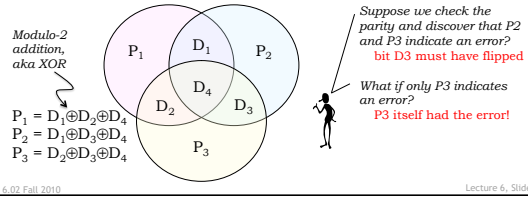
- We have n-k parity bits and so can calculate n-k parity-check bits, which collectively can represent 2^{n-k} possibilities.
- For single-bit error correction, parity bits need to represent
  - Case 1: No error has occurred (1 possibility)
  - Case 2: Exactly one of the code word bits has an error (n possibilities, not k)
- So we need $n+1 \le 2^{n-k}$

$$n \le 2^{n-k} - 1$$

- Hamming codes correct single errors with this minimum number of parity bits (7,4,3), (15,11,3), …

Example: (7,4,3) Hamming Code

- Use multiple parity bits, each covering a subset of the data bits.
- No two message bits belong to exactly the same subsets, so a single-bit error will generate a unique set of parity check errors.



Syndrome Decoding

- After receiving the (possibly corrupted) message, compute a **syndrome** bit (E_i) for each parity bit

$$E_1 = D_1 \oplus D_2 \oplus D_4 \oplus P_1$$
$$E_2 = D_1 \oplus D_3 \oplus D_4 \oplus P_2$$
$$E_3 = D_2 \oplus D_3 \oplus D_4 \oplus P_3$$

- If all the E_i are zero: no errors
- Otherwise the particular combination of the E_i can be used to figure out which bit to correct