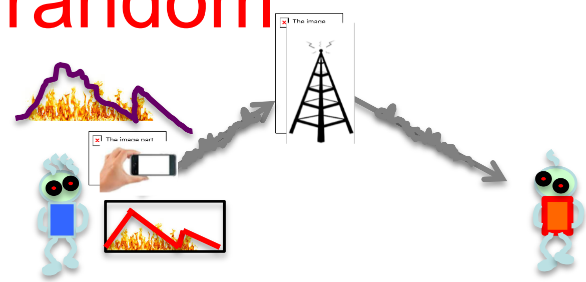
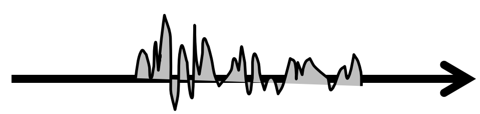
Message Integrity

Message integrity

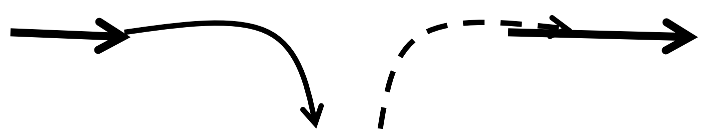
- Sent message is correctly received.

- Messages can be corrupted by random events.



m  m′

- Messages can be corrupted intentionally.

m  m′

Message integrity

Ensuring correct message is received.

1. Probabilistic:
   - Noise, accidents

Reliable communication (error correcting code)

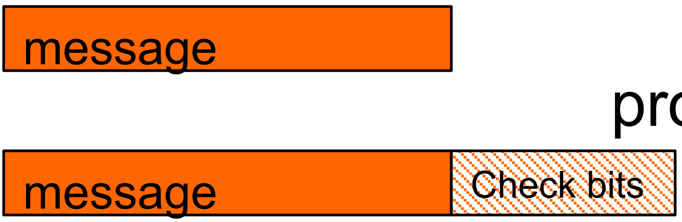
2. Adversarial:
   - Adversary tampering with communication
     - replace messages, inject false messages, block messages...

Authenticated communication (message authentication code)

Message integrity

Changes of a message can be,

probabilistic   intentional



Reliable Comm.
(no secret key)
- Error correction
- Error detection

Secure Comm.
(with secret key)
-Message authentication code

Protection in all cases by adding extra (check) bits to message

Outline

Reliable communication

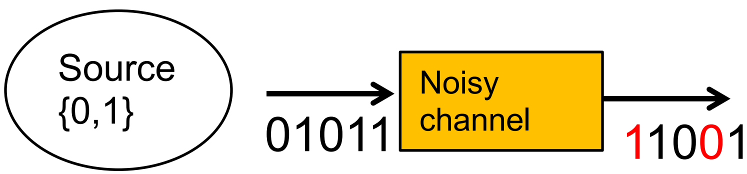
- Error correcting codes
  - Encode/decode
- Linear codes
- Decoding – ML decoding
- Error correcting capability
- Hamming code

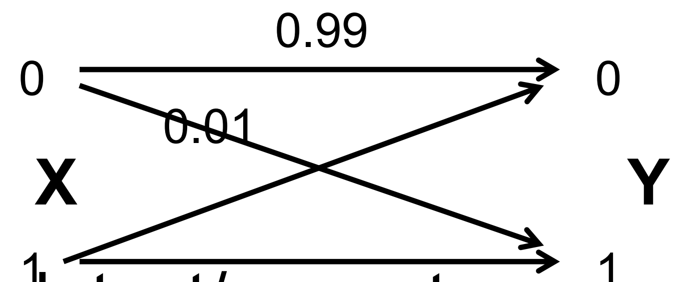
- Efficiency- rate
- Noisy channel theorem

Message authentication

Reliable Communication over Noisy channel

- A probabilistic process corrupts the message:
  - Change is due to probabilistic error.

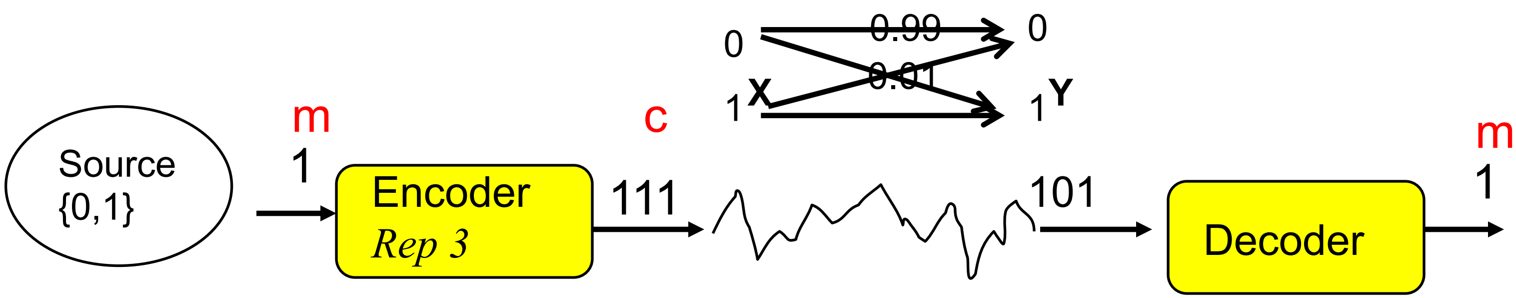


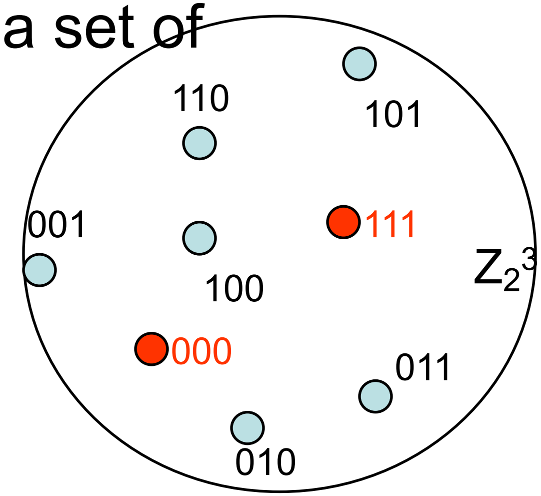


- Error detecting/correcting codes detect/correct errors.

- Example: Repetition code
- *Rep 3* code: repeat each bit three times.

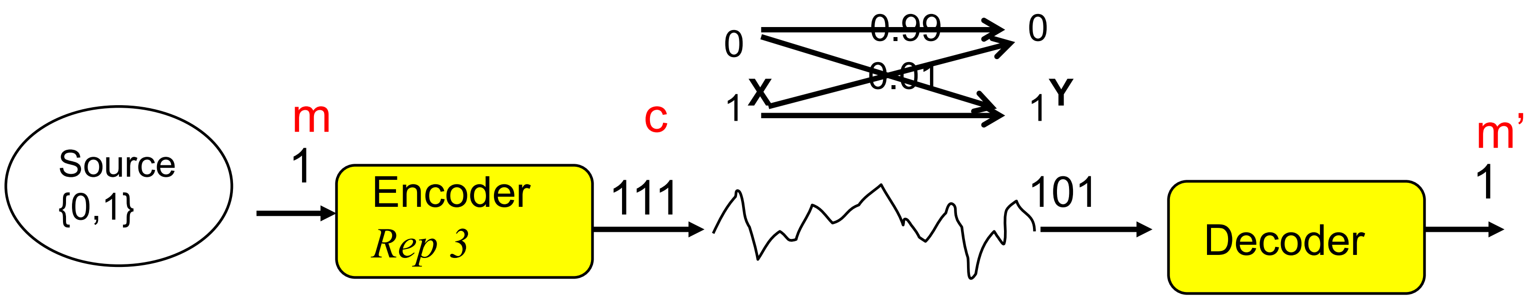
Reliable communication:
Error correction



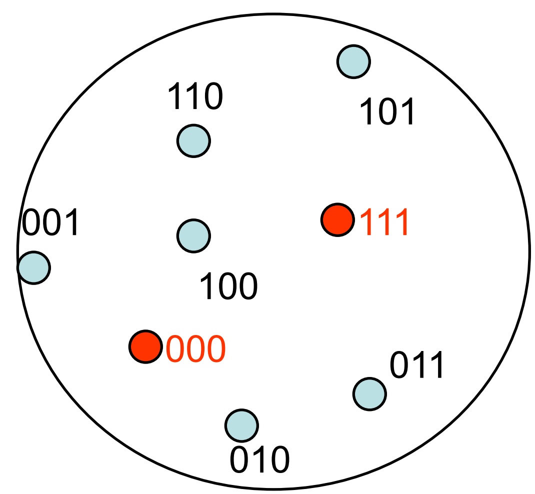


- A binary Error Correcting Code (ECC) C is a set of binary vectors of length n
  - Can be defined over Z_p
- Message space: binary vectors of length k
- An ECC has two algorithms:
- Enc(m)= c is the encoder algorithm:
  - for all m ∈ M maps messages to a codeword in C
- Dec (Z_p^n) = m ∈ {M or ⊥ }

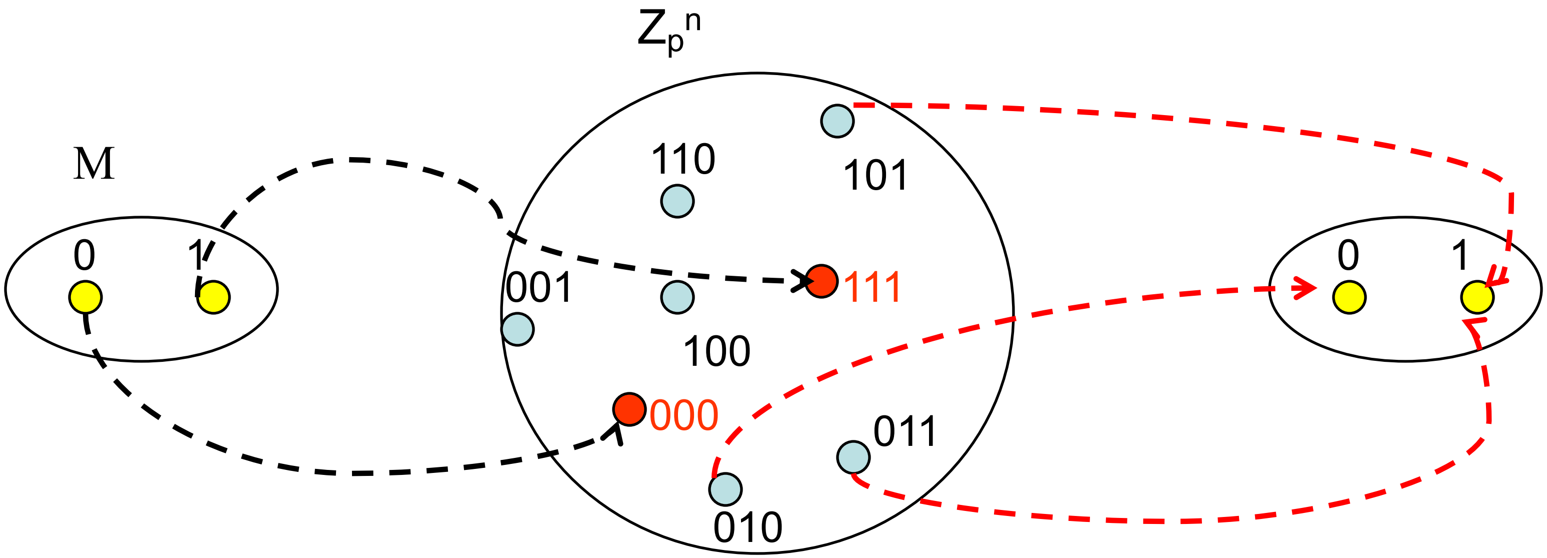
Reliable communication:
Error correction



- For 3-repetition code
- M= {0,1}
- Enc (m) = m m m, $m \in M$
- C= {000, 111}

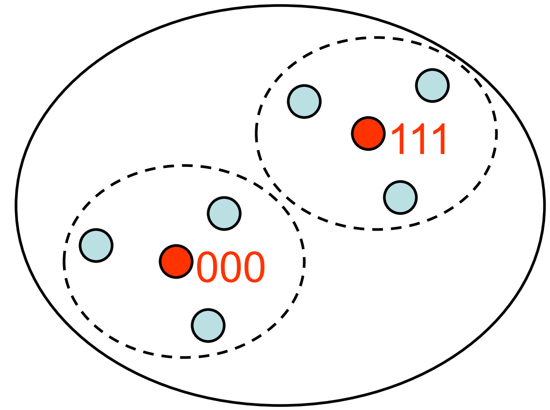


Error correction



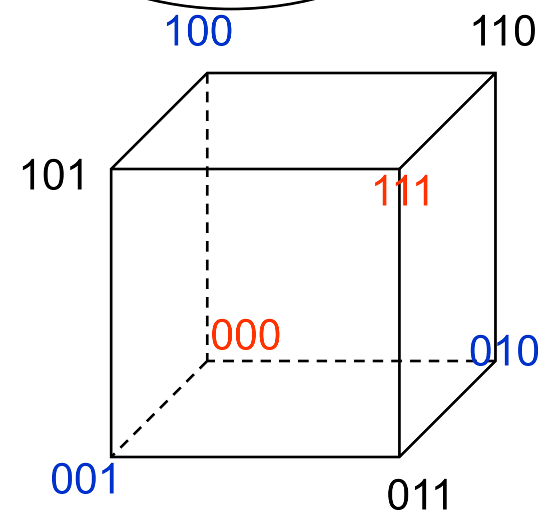
- Decoding is a decision function:
- Given a word in Z_2^3, what message was sent?

Rep3 code



- M= {0,1}
- *Rep 3* code

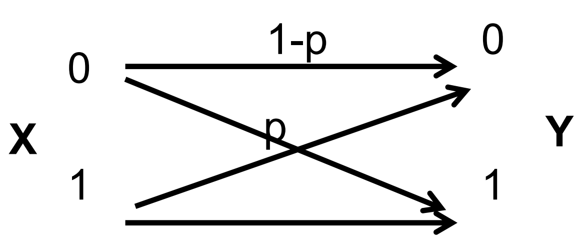
$$C = \left\{ \begin{matrix} 0 & 0 & 0 \\ 1 & 1 & 1 \end{matrix} \right.$$



For BSC_p decoding decision depends on noise level, p.

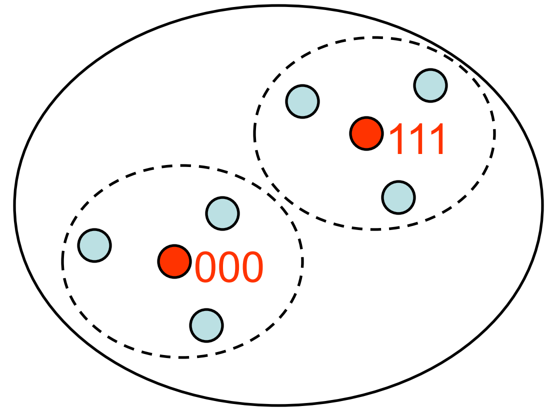
- Is this a "good decision table"?

Decoding algorithm (decision table)



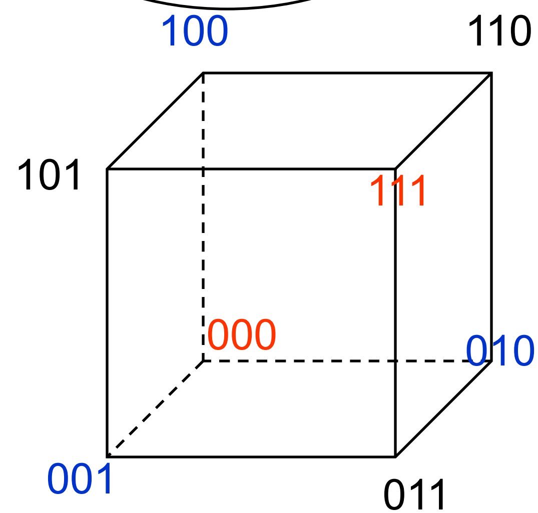
$$\left.\begin{matrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{matrix}\right\} \to 000 \qquad \left.\begin{matrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{matrix}\right\} \to 111$$

Rep3 code



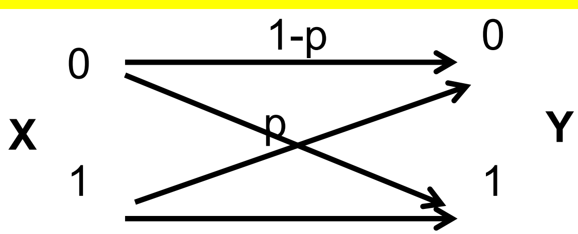
- M= {0,1}
- *Rep 3* code

$$C = \begin{cases} 0 \quad 0 \quad 0 \\ 1 \quad 1 \quad 1 \end{cases}$$



A ”good decision table” results in small number of wrong decisions: decoding error.

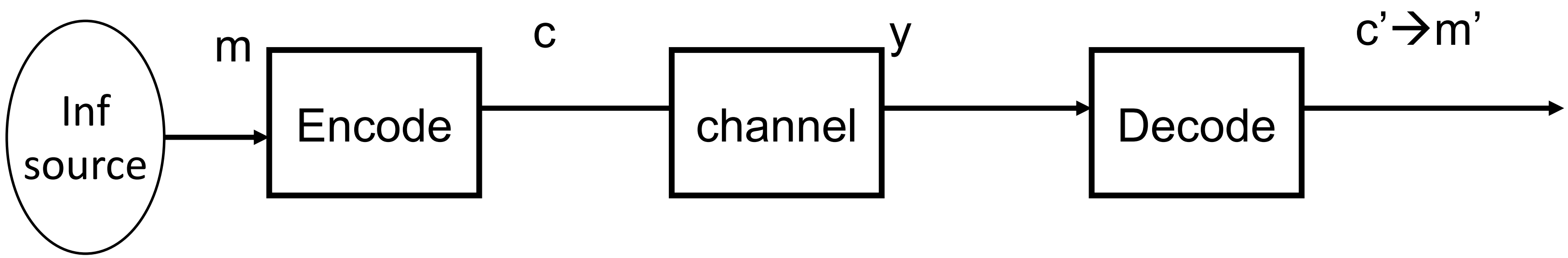
Decoder works correctly if,
- p <1/2
- up to one error occurs.



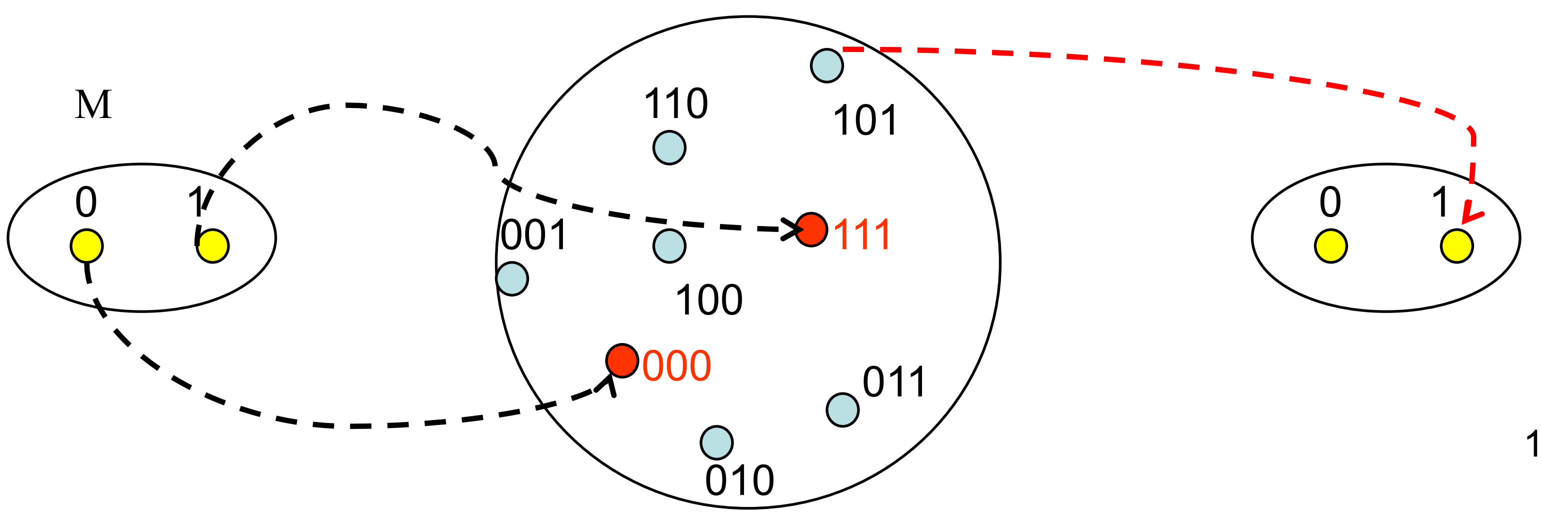
Decoding algorithm (decision table)

$$\left.\begin{matrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{matrix}\right\} \to 000 \qquad \left.\begin{matrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{matrix}\right\} \to 111$$

Decoding



- Decoding error is when decoder outputs a codeword different from the sent one $c \neq c'$



Maximum Likelihood (ML) decoding

- Maximum likelihood decoding minimizes error in decoding.

- Decoding strategy:
- Given y, choose the codeword c with maximum p(c|y)

Find $c \in C$ that Maximizes $p(c \mid y)$

- This depends on the channel probabilities.

Maximizes p(c | y)

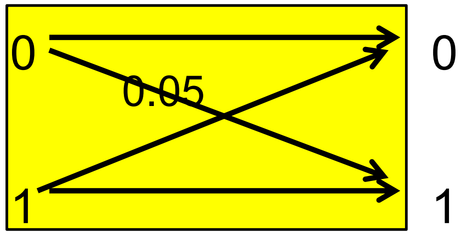
- Note p(c|y)= $\frac{p(y|c)p(c)}{p(y)}$

- y is the received word:
- Finding p(y)= $\Sigma_{\{c \text{ in } C\}}$ p(c) p(y|c)

- Assume M is uniform → p(c) = 1/|M|

Finding p(y)

- p(110 | 000) = p(1|0) p(1|0) p(0|0)



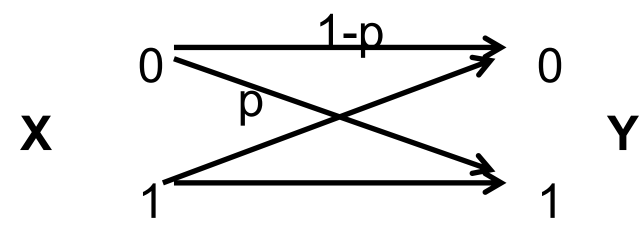
- Each bit flip is independent with probability p
- $p^2 (1-p) \quad = 0.95x\ 0.05x\ 0.05 = 0.0024$

- Similarly, p(110 | 111) = p(1|1) p(1|1) p(0|1)
  = 0.95 x 0.95x 0.05
  =0.045

- p(110)= (1/2) (0.045+ 0.002)=0.0235

Using Hamming distance

- For
- binary codes AND
- BSC channels with p <1/2

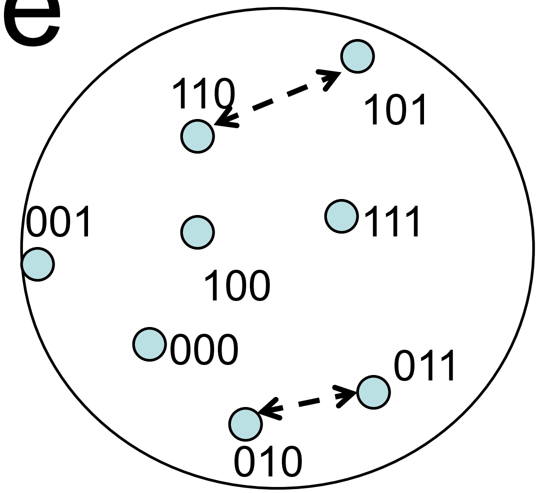


- Maximum likelihood (ML) decoding is equivalent to minimum Hamming distance decoding → find closest code vector

- Example: Rep3

$$\left.\begin{matrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{matrix}\right\} \to 000 \qquad \left.\begin{matrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{matrix}\right\} \to 111$$

Hamming distance



- Hamming distance of two vectors in Z_p^n is the number of places that two vectors are different.

Richard Hamming
1915-1998

Example: Linear codes

- M= {00,01,10,11}
- $G = \begin{bmatrix} 10\ 101 \\ 01\ 110 \end{bmatrix}$

Encoding
each codeword is a linear combination of the rows of a generator matrix.

- $Enc(m_1,m_2) = m_1 . (10101) + m_2 . (01110)$
  - Component-wise multiplication and addition

- c_{00}= Enc(00) = 00 000
- c_{01}= Enc(01)= 01110
- c_{10}=Enc (10)= 10101
- c_{11}= Enc(11)= 11 011

Example: Linear codes

- ML decoding: y= 11111 is received.
- $d_H(11111, 00\ 000) = 5$
- $d_H(11111, 01\ 110) = 3$
- $d_H(11111, 10101) = 2$
- $d_H(11111, 11011) = 1$

- → c = 11011
- → m= 11

Decoding
Find the codeword with minimum Hamming distance

ML decoding

- Encoding

$$C = \left\{ \begin{array}{ccc} 0 & 0 & 0 \\ 1 & 1 & 1 \end{array} \right. \begin{array}{l} \leftarrow 0 \\ \leftarrow 1 \end{array}$$

- Decoding
  - 1 error corrected

$$\left. \begin{array}{ccc} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{array} \right\} \rightarrow 000 \quad \left. \begin{array}{ccc} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{array} \right\} \rightarrow 111$$

- C is a linear code:
- G=[111]
- Enc(0)= 0 x [111]=[000]
- Enc(1)= 1 x [111]=[111]

So far..

- Error correcting codes
- Encoding/decoding
- ML decoding
- Minimum Hamming Distance decoding
- Linear codes

- What is a good ”good code”?

Efficiency

- Rate of binary linear codes = k/n
- k bits of information
- n-bit codeword

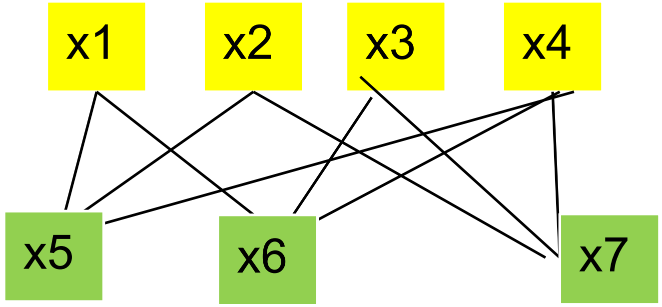
- R= (num info bits)/ (num codeword bits)

- Example: binary repetition code
- R= 1/3

Hamming code

- More efficient codes have higher information rate.
- In Hamming code a block of 4 information bits x1, x2, x3, x4 is "appended" with 7 parity bits

$$\begin{bmatrix} 1000 & 110 \\ 0100 & 101 \\ 0010 & 011 \\ 0001 & 111 \end{bmatrix}$$

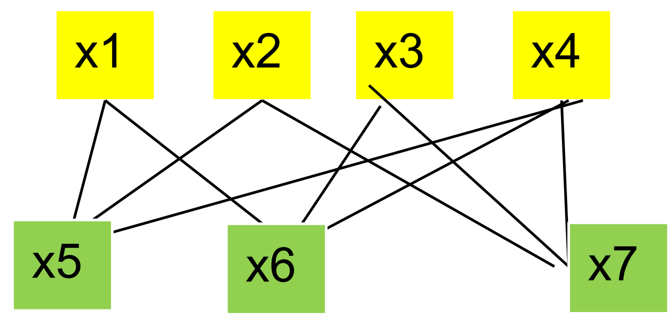


- x1, x2, x3, x4 →
- x1, x2, x3, x4, $\boxed{x1 + x2 + x4}$, $\boxed{x1 + x3 + x4}$, $\boxed{x2 + x3 + x4}$

Hamming code

- The distance between any two codewords is at least 3
- → One error can be corrected

$$\begin{bmatrix} 1000 & 110 \\ 0100 & 101 \\ 0010 & 011 \\ 0001 & 111 \end{bmatrix}$$



4 bit information
7 bit codeword
R= 4/7

- x1, x2, x3, x4 →
- x1, x2, x3, x4, x1 + x2 +x4, x1 +x3 +x4, x2 +x3 +x4

Decoding Hamming code

- y is received: what codeword was sent?

- Find the closest (Hamming distance) code vector
  - Find $d_H(y,c)$ for all c in C
  - Choose c which is closest

- For Hamming codes, there exists an efficient algorithm the finds the location of error.

Comparing with Rep3

- We want to send message $m_1m_2m_3m_4$ over a BSC channel.
- Assume 1 bit error occurs during transmission of the coded 4 message bits

- $m_1m_2m_3m_4 = 1001$

1. Encode each bit separately
   Use Rep3 111 | 000 | 000 | 111
   Rate 1/3

1. Form a block, and use Hamming code:
   1001 100
   Rate: 4/7

→ Block coding provides the same protection with higher efficiency.