Ch6: Channel Coding

more difficult than source coding

IPO: 更後

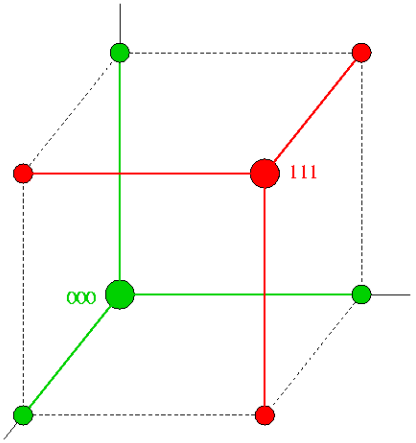
↳ distortion / noise!

Information source and input transducer → Source Coding → Channel Coding → Modulator → Channel → Demodulator (Matched Filter) → Channel Decoding → Source Decoding → Information sink and output transducer

know upper bound what is maximum input rate

insert redundant information at tx data!

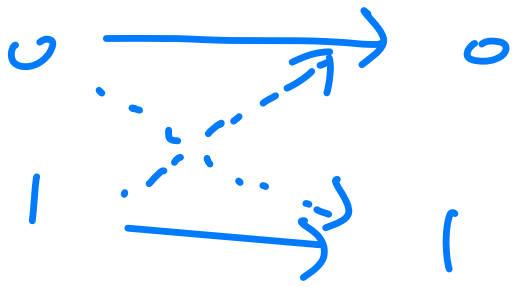
Ch6: Channel Coding



- Questions to be answered:
  - Motivation: Why do we need channel coding?
  - Repetition Code: The simplest channel codes
  - Hamming Codes: Parity check
  - Reed-Solomon Codes

a simple model

$x \in \{0, 1\}$ $\quad\quad$ $\hat{x} \in \{0, 1\}$



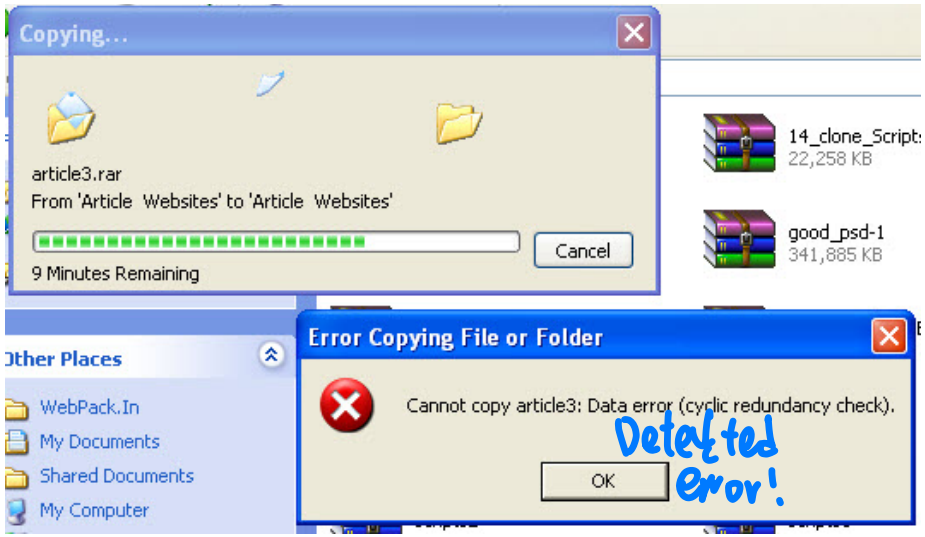
Ch6: Channel Coding

- **Motivation**
- Repetition Codes
- Hamming Codes
- Reed-Solomon Codes
- Channel Coding in GSM Systems
- QR Codes

Channel Coding/ECC

- Provide protection against transmission errors by inserting redundant data
- The channel encoder *inserts redundant information* in a very selective manner. ECC
- Error control coding plays an extremely important role in communication system design
- Do you know any ECC techniques?

error correction! detection



Errors Come from?

- Channel - could be a telephone wire, free space and often presents distorted signal to demodulator
- Channel Effects include
  - Attenuation → serve in wireless communication!
  - Noise (e.g., thermal noise, additive white Gaussian noise or AWGN.)
  - Filtering
    - Channel can have a bandwidth that is small compared to the signal bandwidth (e.g. in a telephone channel).
  - Fading
    - Signal amplitude can change in a random fashion
  - Time Variation
    - Channels change with time. (e.g. call on a bus.)

SNR↑ → error rate↓

Reliability vs. Signal Power

- One way to get a relatively error-free or reliable signal would be to use huge amounts of power to blast over the noise.

SNR↑↑ ↳但耗电快

- How to achieve a higher ***Energy Efficiency?***

- Some coding schemes can achieve reliable transmission at lower powers than others

We want!

Error rate

Error probability

Bit error rate (BER)

Information Theory and Coding

Pe: error prob (can make this small!)

Information theory states that Channel coding theorem:

- In any communications system, there is a fundamental limit, called **Channel Capacity C** [bits/sec], on the maximum possible rate of reliable data transmission.

  不可 rate > C !!!

- In theory, one can transmit over a channel with Bandwidth *B* Hertz at a rate $R \leq C$ with almost no errors.

- However, if we try to transmit at a rate $R > C$ bits/sec, then reliable transmission is impossible.

Source coding

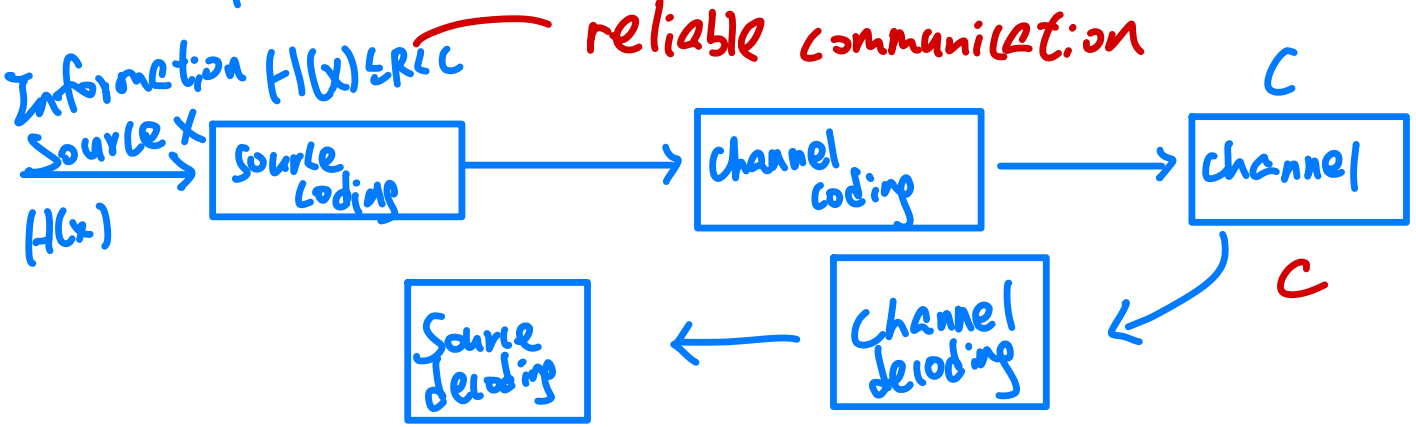
$R \geq H(x) \Rightarrow$ lossless comp.

$R < H(x) \Rightarrow \ldots$ impossible

Channel coding

$R \leq C \Rightarrow$ reliable commu.

$R > C \Rightarrow \ldots$ impossible!



AWGN

More details later.

Additive White Gaussian Noise Channel

noise is Gaussian!

The capacity of AWGN channel is given by

in Ch 7

$$C = B\log_2\left(1+\frac{S}{N}\right)$$

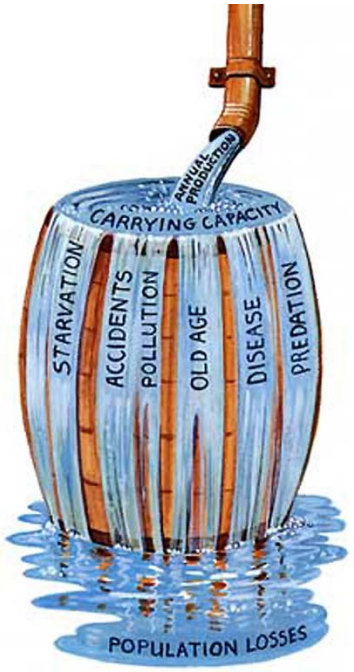
SNR

bits/sec

where

B = **Channel Bandwidth** (Hz)

$\frac{S}{N}$ = **Signal-to-Noise Ratio**

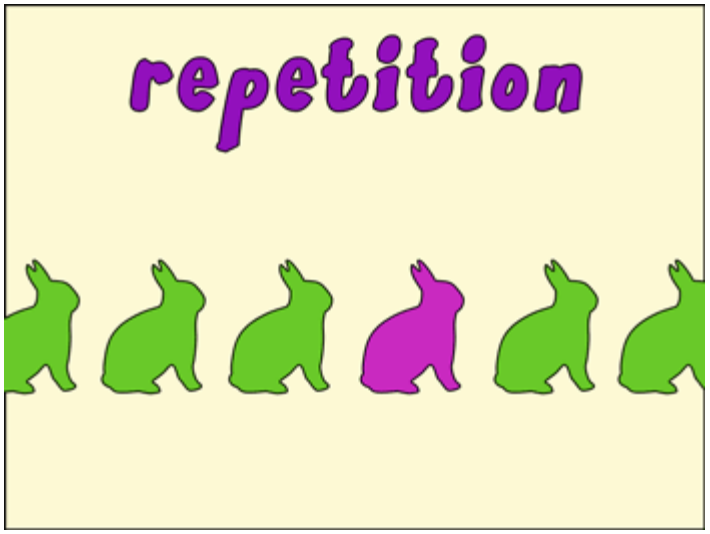


Channel Coding

- **Error control coding** encodes digital data so as to ***introduce dependency among a large number of symbols***. A decoder is then able to more accurately detect the transmitted data.
- There are two types of codes:
  - **Error correcting codes,** e.g. repetition codes
  - **Error detecting codes,** e.g. even parity check codes
- The type of coding should be selected on the basis of
  - Type of **noise** in the communication channel
  - Available **power** budget
  - Available **bandwidth**
  - Other practical consideration just as the amount of **processing power** available.

Ch6: Channel Coding

- Motivation
- **Repetition Codes**
- Hamming Codes
- Reed-Solomon Codes
- Channel Coding in GSM Systems
- QR Codes



Repetition Code by Example

- We know that a telephone link using a modem works with the following conditions:
  - Link bandwidth B = 3kHz
  - The modem can operate up to the speed of 3600 bits/sec at an error probability of $q_c = 8 \times 10^{-4}$
- **Question 1:** What is the maximum transmission rate with transmit SNR = 13dB.
- **Question 2:** Can we transmit data at a rate of 1200 bits/sec **with a probability of error 10^{-4}**

Repetition Code by Example

- For the above communication link, the channel capacity is

$$C = B\log_2\left(1+\frac{S}{N}\right) = 13{,}000 \text{ bits/sec}$$

since $B = 3000$ and $S/N = 20$ (13dB = 10log20).

- Thus, by Shannon's theorem, we CAN transmit the data with **an arbitrarily small error probability**.

- Note that **without coding $P_e = 8 \times 10^{-4}$**. For the given modem, criterion $P_e = 10^{-4}$ is **not met.**

A Simple Code Design

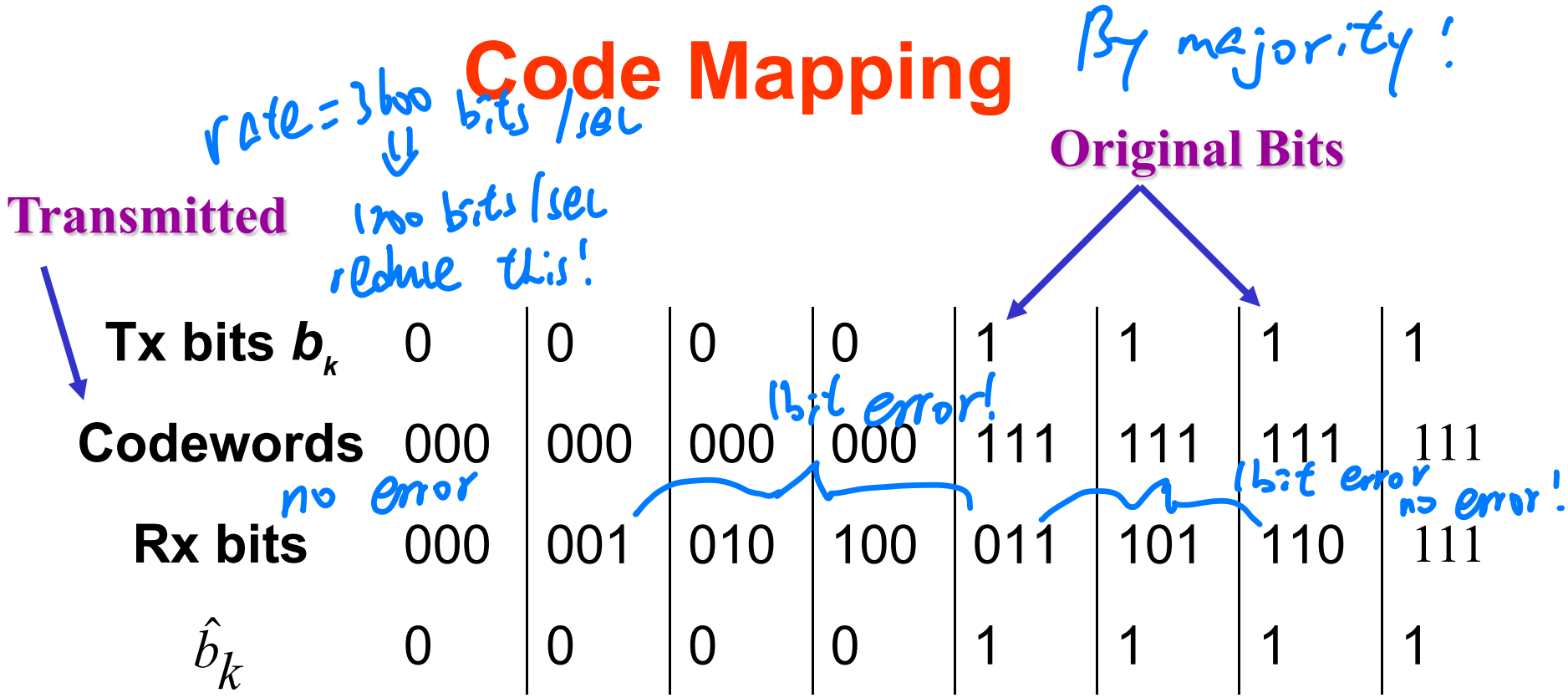
- Design a simple code which yields an overall probability of error of 10^{-4}

- ***Possible Solution:*** **Repetition code**

  When b_k = "0" or "1", the codewords "**000" and "111" are transmitted**.

- The decoder looks at the received codewords and attempts to extract the transmitted bits using a ***Majority-logic decoding scheme***

Code Mapping



- Clearly, the **transmitted bits will be recovered correctly as long as no more than one of the bits in the codeword is wrong**.

Detection Error

- Recall that **without channel coding** the **probability of error** for the given modem is 8×10^{-4}. But **with error control coding**,

transmitted bit　decoded bit　q_c: bit error rate

$$P_e = P(b_k \neq \hat{b}_k)$$ ← Error event!

这俩就有 error!

$$= P(2 \text{ or } 3 \text{ bits in codeword are in error})$$

$$= \binom{3}{2} q_c^2 (1 - q_c) + \binom{3}{3} q_c^3$$ — 3 bits error

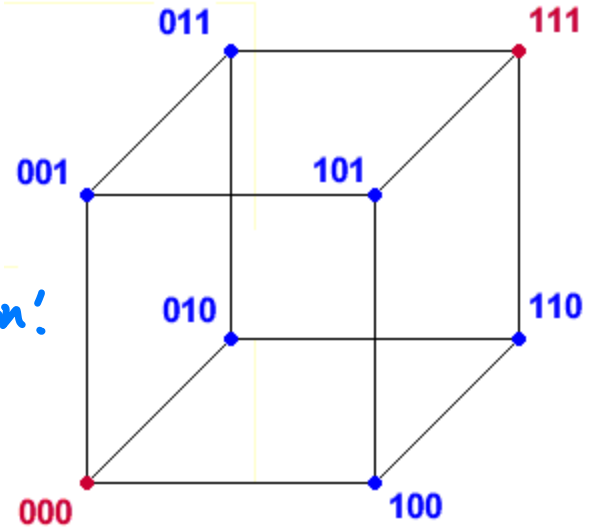
$$= 3q_c^2 - 2q_c^3$$

2 bits error!

$$= 0.0192 \times 10^{-4} \leq \text{Required } P_e \text{ of } 10^{-4}$$

Ch6: Channel Coding

- Motivation
- Repetition Codes
- **Hamming Codes** stronger! Simpler and common!
- Reed-Solomon Codes
- Channel Coding in GSM Systems
- QR Codes



(*n*,*k*) t-Error-Correcting Code

- A (*n*,*k*) ***t*-error-correcting code** encodes ***k*** input bits into an ***n***-bit codeword by adding ***n-k* parity check** bits so that it can correct all **up to** ***t***-bit errors.
- ***k/n*** is called the **code rate**, which measures the amount of coding redundancy.
- The code in previous example is called the **(3,1)** single-error correcting **repetition code.**
- In general, the definition of most useful codes are based on parity check conditions.
- e.g. the well-known single-error correcting code is the **(7,4) Hamming code**

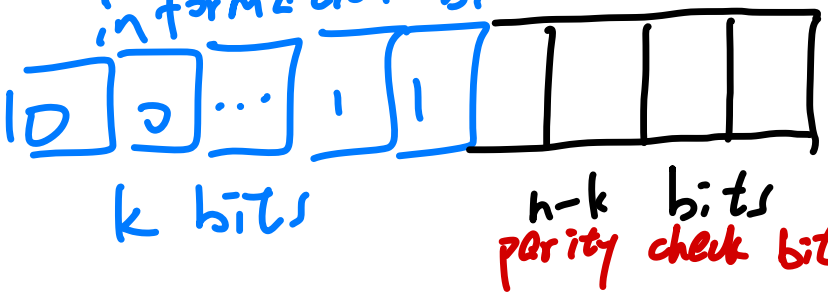
$\frac{4}{7} > \frac{1}{3}$

Hamming Repetition

(n, k) - t code

n: total number of bits!

k: information bits



k bits

$n-k$ bits

parity check bits!

it can correct up to t errors

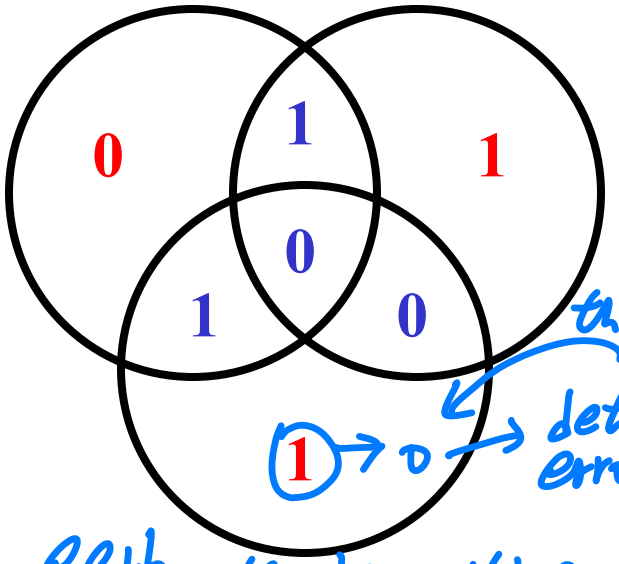
e.g. repetition code $(3,1)$ - 1

$\frac{k}{n}$: code rate

(7,4) Hamming Code

- The parity check bits are generated such that all bits inside the same circle must have even parity, i.e. even number of 1's.

- **Can you find out a way to decode so as to correct all single-bit errors?**
- **Hint**: locate the error by parity checking.



Notation:
- 4 input bits are in blue
- 3 parity checks are in red

Use of Hamming Codes

- Hamming Codes are still widely used in computing, telecommunication, and other applications.

cloud storage!

- Hamming Codes also applied in
  - Data compression
  - Some solutions to the popular puzzle The Hat Game
  - Block Turbo Codes

A [7,4] binary Hamming Code

- Let our codeword be $(x_1\ x_2\ \ldots\ x_7)$
- x_1, x_2, x_3, x_4 are chosen according to the message

- $x_5 := (x_1 + x_2 + x_3) \pmod 2$
- $x_6 := (x_2 + x_3 + x_4) \pmod 2$
- $x_7 := (x_1 + x_2 + x_4) \pmod 2$

Generating Matrix

- Given a matrix

$$G = \left(\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{array}\right)$$

- If $\mathbf{u}=(u_1,u_2,u_3,u_4)$, then we encode it as $\mathbf{c} = \mathbf{uG}$

$$\mathbf{c} = \mathbf{uG} = (u_1, u_2, u_3, u_4)\left(\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{array}\right)$$

$$= (u_1, u_2, u_3, u_4, u_1+u_2+u_3, u_2+u_3+u_4, u_1+u_2+u_4)$$

- For example, if u=(1,0,0,1), then c=(1,0,0,1,1,1,0).

Parity check bits

Parity Check Bits

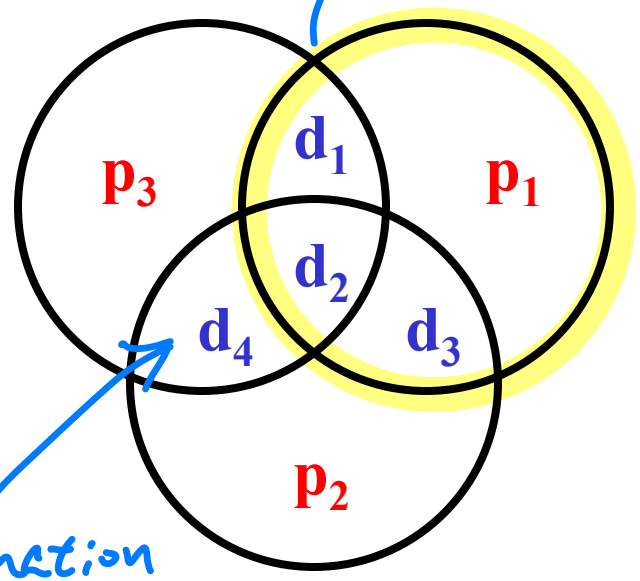
- Graphical depiction of the four data bits and three parity bits and which parity bits apply to which data bits:

$$\mathbf{G} = \left(\mathbf{I}_k \mid \mathbf{P}\right)$$

related to p_1 p_2 p_3 !

$$G = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}$$

$(d_1 + d_2 + d_3 + p_1) \% 2 = 0$



information bits !

Parity Check Matrix

- For a given code generating matrix, we can determine its corresponding parity check matrix.
- Specifically, if $\mathbf{G} = (\mathbf{I}_k \,|\, \mathbf{P})$ then the parity check matrix is

$$\mathbf{H} = (\mathbf{P}^T \,|\, \mathbf{I}_{n-k})$$

H, G one to one mapping!

- So

$$\mathbf{H} = \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}$$

P^T

diag. matrix!

Error Check

- Without errors we have

$$\mathbf{E} = \mathbf{H}\mathbf{c}^T = \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_1+u_2+u_3 \\ u_2+u_3+u_4 \\ u_1+u_2+u_4 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

- If there is an error in the first location c_1, then

$$\mathbf{E} = \mathbf{H}\mathbf{c}^T = \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix} \mathbf{c}^T = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$$

Syndrome decoding

find error → correct that!

- Consider a vector of received word **r**=**c**+**e**, where **c** is a valid codeword transmitted and **e** is the error introduced by the channel.
- The matrix product $\mathbf{rH}^T$ is defined as the syndrome for the received vector **r**

$$\begin{aligned}\mathbf{s} &= \mathbf{rH}^T \\ &= (\mathbf{c}+\mathbf{e})\,\mathbf{H}^T \\ &= \mathbf{cH}^T + \mathbf{eH}^T \\ &= \mathbf{0} + \mathbf{eH}^T \\ &= \mathbf{eH}^T \rightarrow\end{aligned}$$

received word!!! codeword error

codeword

$c = e \cdot H^T$

as $c \cdot H^T = \vec{0}$

Syndrome decoding

- Thus, the syndrome is independent of the transmitted codeword **c** and is solely a function of the error pattern **e**.
- It can be determined that if two error vectors **e** and **e'** have the same syndrome, then the error vectors must differ by a nonzero codeword.

$$\mathbf{s} = \mathbf{e}\mathbf{H}^T = \mathbf{e}'\mathbf{H}^T$$
$$\Rightarrow (\mathbf{e} - \mathbf{e}')\,\mathbf{H}^T = 0$$
$$\Rightarrow (\mathbf{e} - \mathbf{e}') = \mathbf{c} \in \mathbf{C}$$

recover e, then recover c.

codeword!

- It follows from the equation above, that decoding can be performed by computing the syndrome of the received word, finding the corresponding error pattern and subtracting the error pattern from the received word.

Syndrome decoding

不知原来codeword！

$$codeword = [1 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0 \quad 1]$$

$$received\ codeword = [1 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0 \quad \mathbf{0}]$$

error!

$$syndrome = [1 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0 \quad \mathbf{0}] \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = [\mathbf{0} \quad \mathbf{0} \quad \mathbf{1}]$$

last bit is error

error

$$received\ codeword = [1 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0 \quad \mathbf{1}]$$

one bit error!

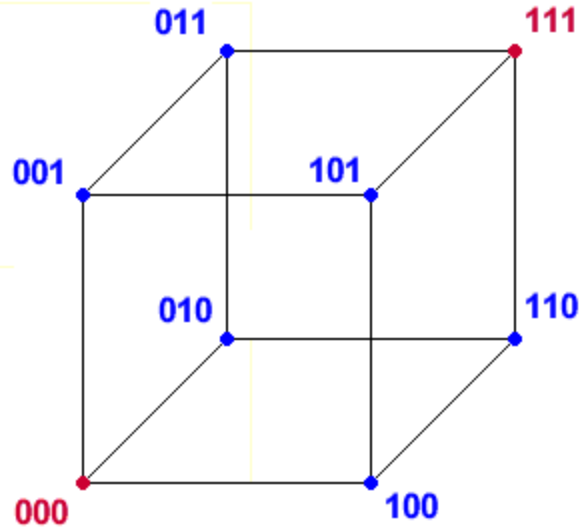
Syndrome $\mathbf{s} = \mathbf{r}\mathbf{H}^T$	Error pattern $\mathbf{e}'$
000	0000000
001	0000001
010	0000010
011	0001000
100	0000100
101	1000000
110	0100000
111	0010000

可以 detect 2 error!

Assume P(≥1 error is small!)

Ch6: Channel Coding

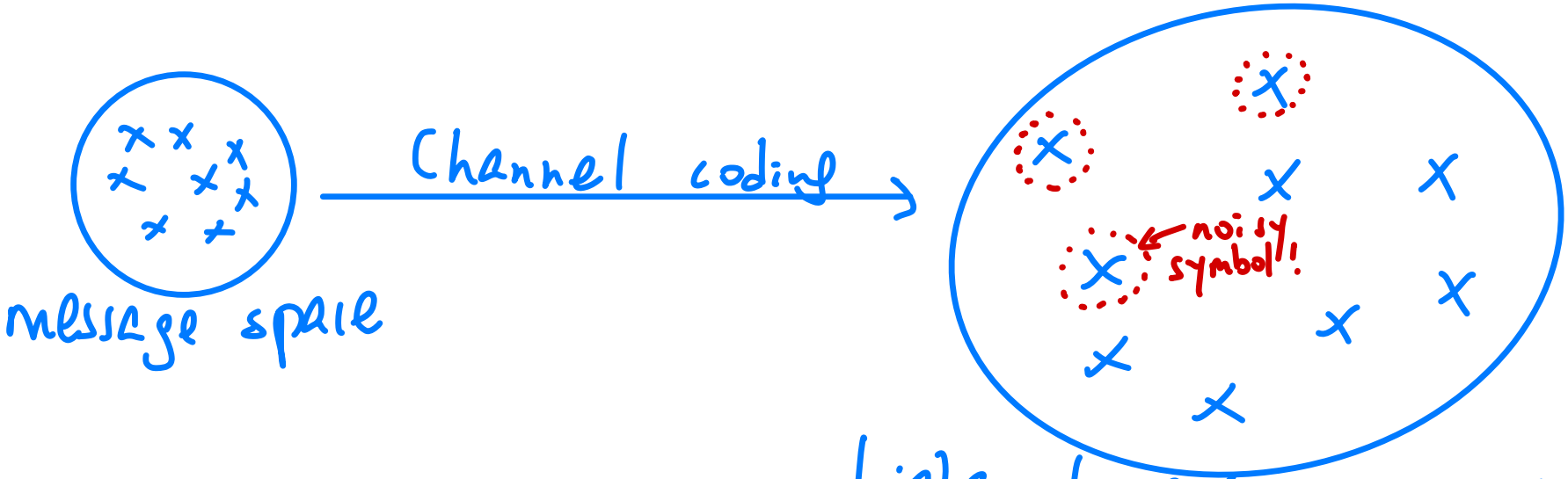
- Motivation
- Repetition Codes
- Hamming Codes
  ↓ extension
- **Reed-Solomon Codes**
- Channel Coding in GSM Systems
- QR Codes



Channel coding

increase reliability

- Enlarge distance!



message space

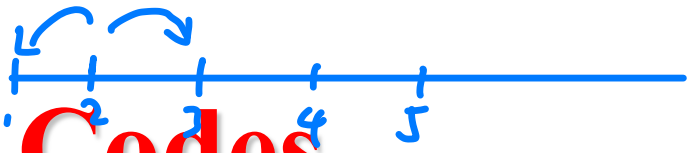
higher dimension space:

- distance much larger
  - Difficult that noise will confuse you!

Reed-Solomon Codes

- (n,k) codes invented in 1960 by Reed and Solomon.
- Original idea: interpret the k source symbols as coefficients of a polynomial and then oversample the polynomial at n points which are then transmitted. The receiver will then use interpolation techniques to recover the original message.
- Today RS codes are implemented in a different way.
- First commercial application in mass-produced consumer products appeared in 1982 with CDs.
- Today, RS codes are used virtually everywhere from digital storage devices to digital comm. standards.
- Recently, they are being slowly replaced by more moderm LDPC codes or turbo codes.

Reed-Solomon Codes



- Imagine we transmit a number among 1,2,3,4,5.
- At the receiver, due to noise, a “2” could be detected as a “1” or a “3” by accident.

- Now suppose we enlarge the alphabet from 1 to 50 and when we want to transmit a number *x* we first multiply it by 10 and then transmit 10*x*.
- If there is an error of one integer, a “20” could become a “19” or “21” but it is still clear that corresponds to a “20”. Only if the error is more than 5 we can go from “20” to detecting a “10” or a “30”.
- This is the idea of RS codes but using polynomials instead.

Reed-Solomon Codes

- Suppose we map a sequence of *k* numbers (the message) into a polynomial:

$$u(x) = u_0 + u_1 x + u_2 x^2 + \ldots + u_{k-1} x^{k-1}$$

- If we transmit directly the *k* coefficients, we are inevitably subject to errors.
- But now suppose we first multiply that polynomial (message) by a generator polynomial of degree *n-k* with roots $\alpha, \alpha^1, \ldots, \alpha^{n-k}$ given by

$$g(x) = (x-\alpha)(x-\alpha^2)\ldots(x-\alpha^{n-k})$$

- Now, the transmitter can send the n coefficients of the product polynomial: $s(x) = u(x)g(x)$

Reed-Solomon Codes

- In the absence of errors, the receiver can recover the message from the received polynomial $r(x)=u(x)g(x)$ simply by dividing it by the generator polynomial.

- In the presence of errors, the received polynomial is given by $r(x)=u(x)g(x)+e(x)$.
- How can we now recover the message *u(x)*?
- There is a simple way: evaluate the received polynomial in the roots of the generator polynomial $\alpha, \alpha^1, ..., \alpha^{n-k}$:

$$r(\alpha^i)=u(\alpha^i)g(\alpha^i)+e(\alpha^i)=e(\alpha^i)$$

or by long-division.

evaluate e(x)

- This way we can estimate the error (assuming not too many errors). Can correct up to *(n-k)/2* errors.

Remark – Decoding of RS Code

The transmitted polynomial is corrupted in transit by an error polynomial $e(x)$ to produce the received polynomial $r(x)$.

$$r(x) = s(x) + e(x)$$
$$e(x) = \sum_{i=0}^{n-1} e_i x^i$$

where e_i is the coefficient for the i-th power of x. Coefficient e_i will be zero if there is no error at that power of x and nonzero if there is an error. If there are ν errors at distinct powers i_k of x, then

$$e(x) = \sum_{k=1}^{\nu} e_{i_k} x^{i_k}$$

The goal of the decoder is to find the number of errors (ν), the positions of the errors (i_k), and the error values at those positions (e_{i_k}). From those, $e(x)$ can be calculated and subtracted from $r(x)$ to get the originally sent message $s(x)$.

Syndrome decoding [edit]

The decoder starts by evaluating the polynomial as received at certain points. We call the results of that evaluation the "syndromes", S_j. They are defined as:

$$S_j = r(\alpha^j) = s(\alpha^j) + e(\alpha^j) = 0 + e(\alpha^j) = e(\alpha^j),\ j = 1, 2, \ldots, n-k$$
$$= \sum_{k=1}^{\nu} e_{i_k} \left(\alpha^j\right)^{i_k}$$

The advantage of looking at the syndromes is that the message polynomial drops out. In other words, the syndromes only relate to the error, and are unaffected by the actual contents of the message being transmitted. If the syndromes are all zero, the algorithm stops here and reports that the message was not corrupted in transit.

Random Errors – Unknown Error Locations – Only Correct up to (n-k)/2 errors

Error locators and error values [edit]

For convenience, define the **error locators** X_k and **error values** Y_k as:

$$X_k = \alpha^{i_k},\ Y_k = e_{i_k}$$

Then the syndromes can be written in terms of the error locators and error values as

$$S_j = \sum_{k=1}^{\nu} Y_k X_k^j$$

This definition of the syndrome values is equivalent to the previous since $(\alpha^j)^{i_k} = \alpha^{j*i_k} = (\alpha^{i_k})^j = X_k^j$.

The syndromes give a system of $n - k \geq 2v$ equations in $2v$ unknowns, but that system of equations is nonlinear in the X_k and does not have an obvious solution. However, if the X_k were known (see below), then the syndrome equations provide a linear system of equations that can easily be solved for the Y_k error values.

$$\begin{bmatrix} X_1^1 & X_2^1 & \cdots & X_\nu^1 \\ X_1^2 & X_2^2 & \cdots & X_\nu^2 \\ \vdots & \vdots & & \vdots \\ X_1^{n-k} & X_2^{n-k} & \cdots & X_\nu^{n-k} \end{bmatrix} \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_\nu \end{bmatrix} = \begin{bmatrix} S_1 \\ S_2 \\ \vdots \\ S_{n-k} \end{bmatrix}$$

Erasure – Known Error Locations – Correct up to (n-k) errors

In the variant of this algorithm where the locations of the errors are already known (when it is being used as an erasure code), this is the end. The error locations (X_k) are already known by some other method (for example, in a FM transmission, the sections where the bitstream was unclear or overcome with interference are probabilistically determinable from frequency analysis). In this scenario, up to $n - k$ errors can be corrected.

The rest of the algorithm serves to locate the errors, and will require syndrome values up to $2v$, instead of just the v used thus far. This is why 2x as many error correcting symbols need to be added as can be corrected without knowing their locations.

Ch6: Channel Coding

- Motivation
- Repetition Codes
- Hamming Codes
- Reed-Solomon Codes
- **Channel Coding in GSM and 3G Systems**
- QR Codes

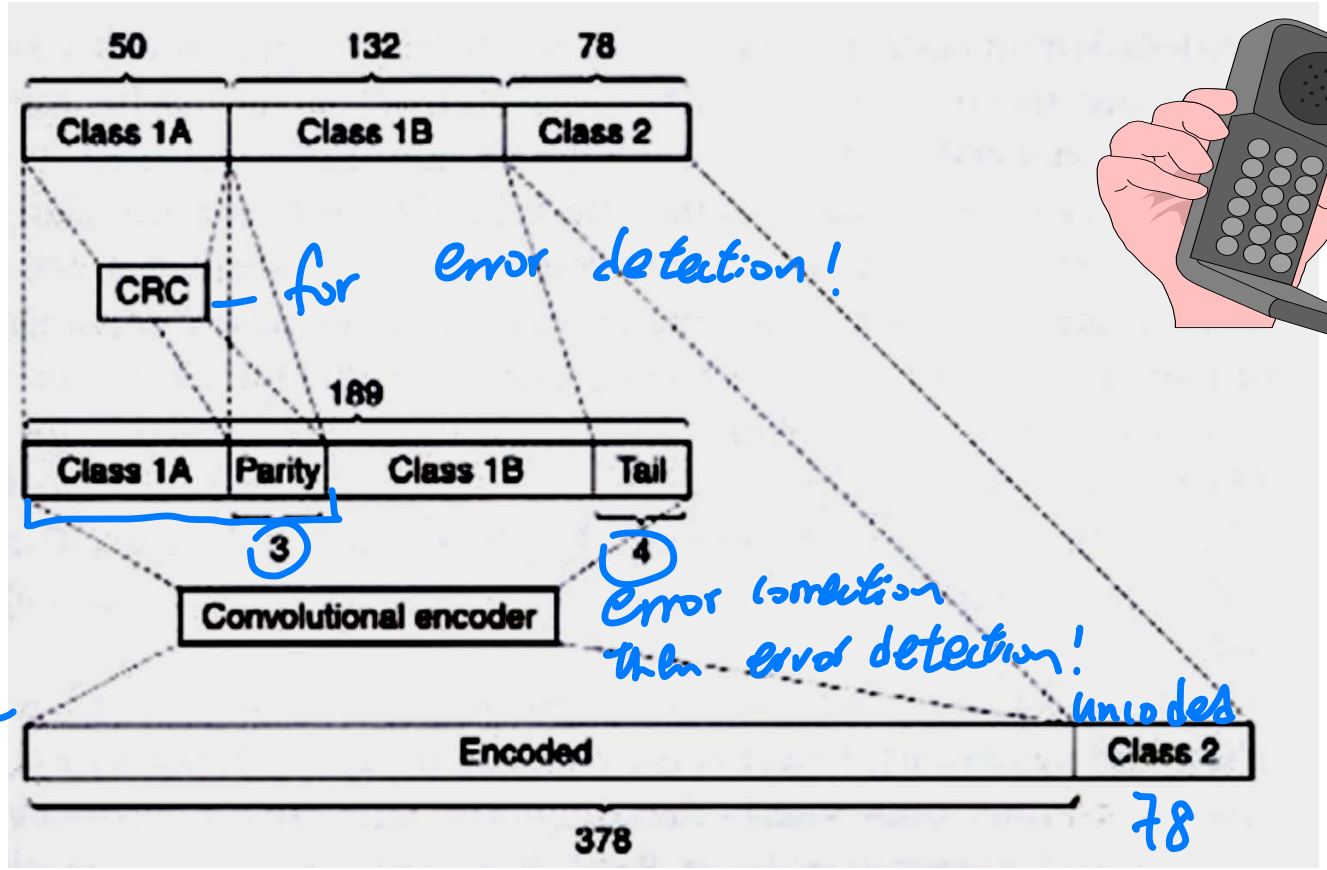
ECC in GSM Phones

- To improve the reliability of mobile communication, ECC is used in **Gobal System for Mobile Communication** (**GSM**).
- Speech coding algorithm in GSM gives a net data rate of 13kbit/s. Speech data are divided into 20ms blocks, resulting in a **frame size of 260 bits**.
- The first **182 bits** are in **Class 1**, and the remaining **78** in **Class 2**.
- Class 1 bits are more important since errors in these bits cause a greater degradation in perceived speech quality.
- Within Class 1, the first **50 bits** labeled **Class 1A** are still more important than the remaining **132 bits** labeled **Class 1B**. If errors occur in Class 1A bits, it is better to discard the speech data frame.

ECC in GSM Phones

- Different levels of protection are provided by combined use of error-correcting and error-detecting codes.
- **50 Class 1A bits** are protected by the **(53,50) CRC error detecting code**, which appends 3 parity check bits.
- **4 tail bits** are added to the Class 1 resulting in **189** (=50+3+132+4) bits.
- A **rate-1/2 (convolutional) error-correcting code** then encodes **all of these Class 1 bits** into a **378-bit codeword** by appending 189 parity check bits. Overall speaking, it results in a **(378,185) error-correcting code**. Note that the tail bits were added to simplify the decoding of the code.
- **78 Class 2 bits** are **uncoded** and will be transmitted as are.

ECC in GSM Phones



make this part no error.

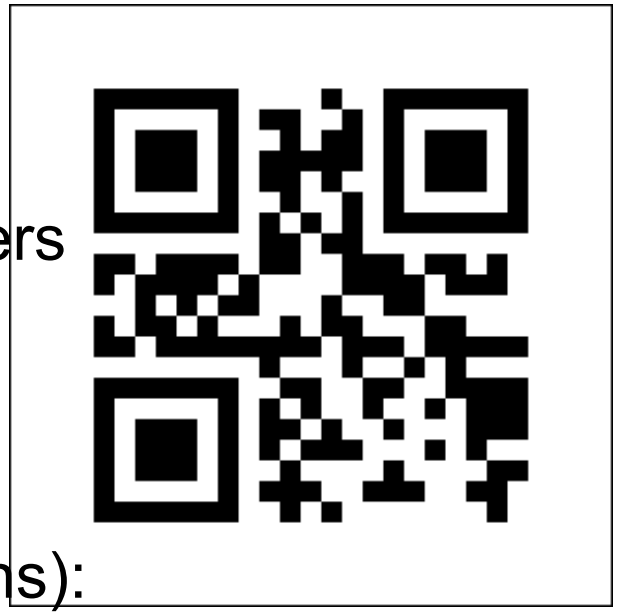
WCDMA uses Convolutional codes and Turbo codes.

Ch6: Channel Coding

- Motivation
- Repetition Codes
- Hamming Codes
- Reed-Solomon Codes
- Channel Coding in GSM and 3G Systems
- **QR Codes** Error control

QR Codes

robotis!



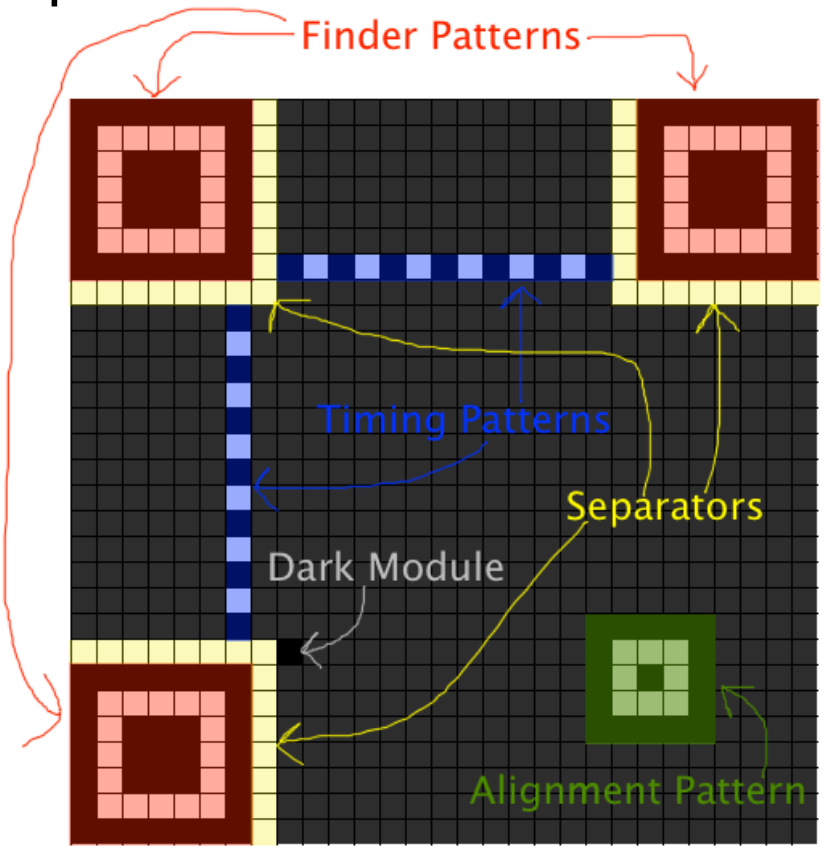
- A QR code is a special type of barcode that can encode information like numbers and letters.
- Was created in 1994 by Japanese company (subsidiary of Toyota).
- There are different sizes (called versions): 21x21 pixel size is version 1.
- QR codes include error correction (channel coding) based on RS codes. There are different levels of protection: 7%, 15%, 25%, and 30% of errors.
- For a tutorial, see http://www.thonky.com/qr-code-tutorial/

QR Codes

- Data Analysis and Encoding: this basically transforms the original information (numeric, alphanumeric, byte, and Kanji) into a string of bits.
- Error Correction Coding: this performs the channel encoding per se.
  - Depending on the QR version different partitions of the bit string are used.
  - Also, different codes (generator polynomials) can be used with . $\alpha = 2$
  - For example, the generator polynomial for correction of 1 error is $g(x) = x^2 + 3x + 2$.
  - Sophisticated tricks are used so that multiplication of numbers in Galois field can be implemented by adding the corresponding exponents (log-tables)

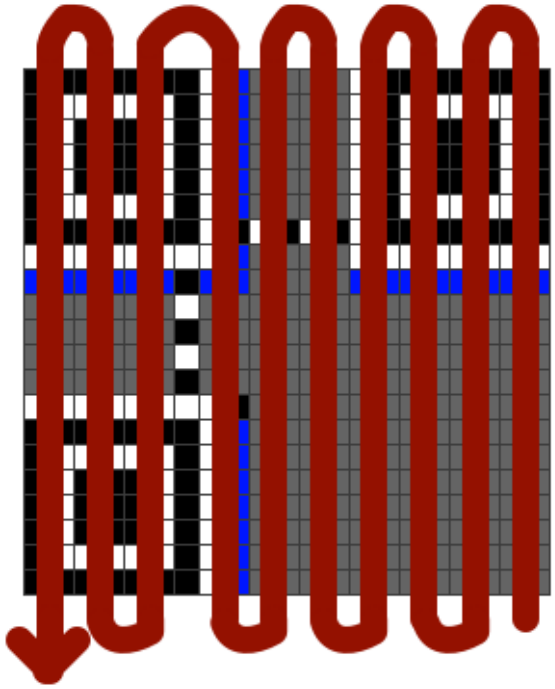
QR Codes

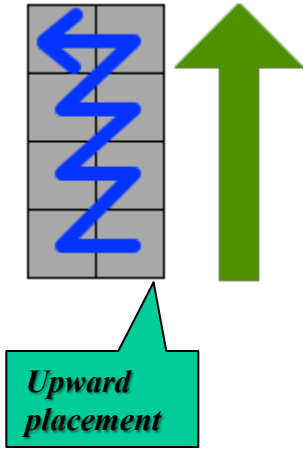
- Interleaving: after encoding different blocks with different codes, in order to spread the errors (avoid bursts), interleaving is used.
- Then, the coded bits are placed in the matrix in a very specific way:

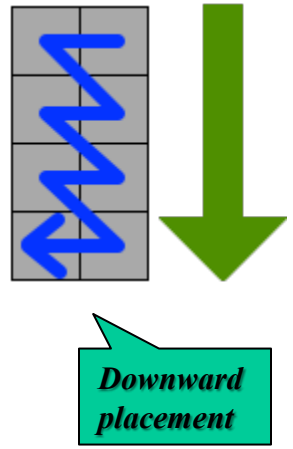


QR Codes

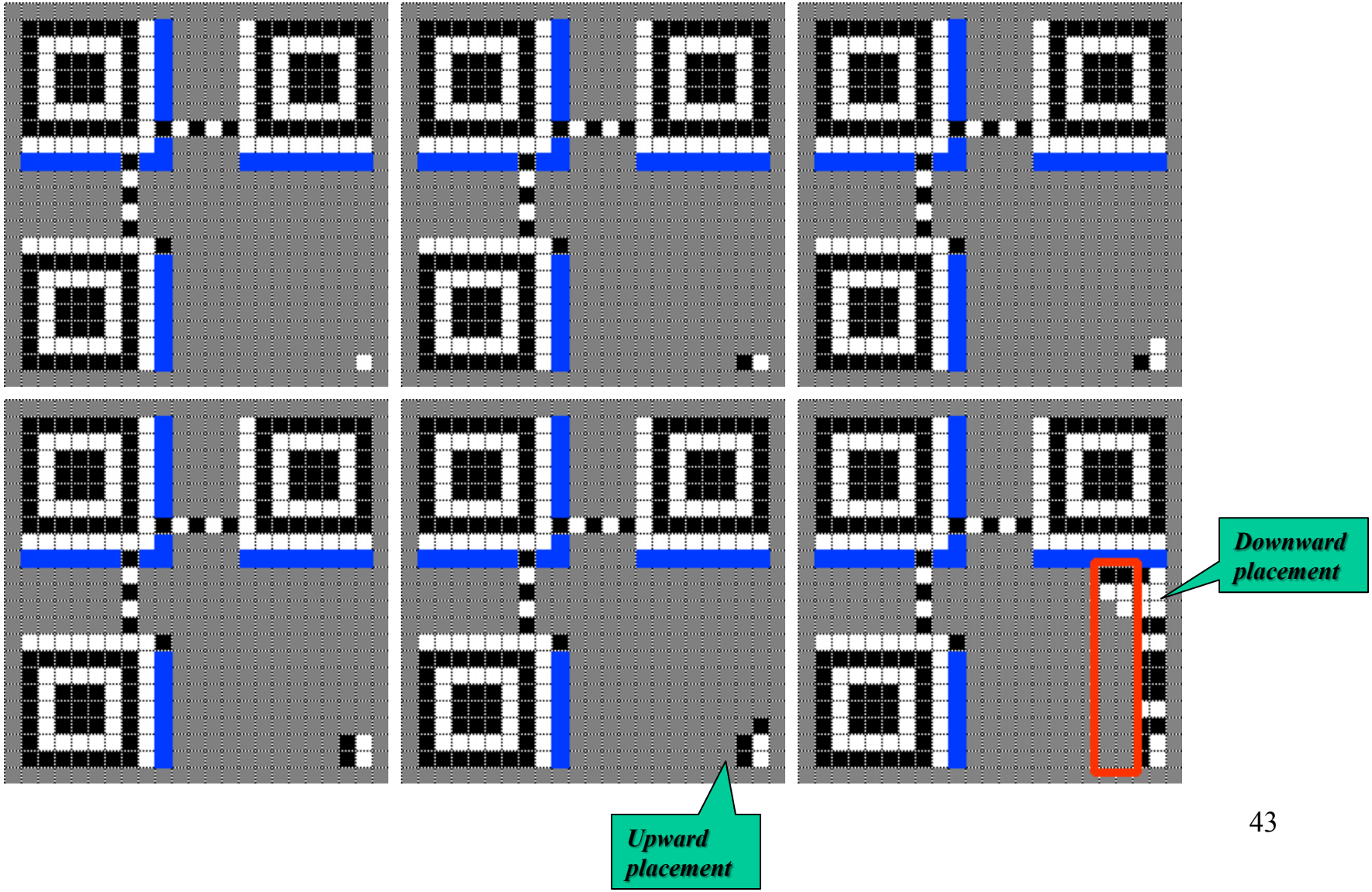
- Bits are written vertically (avoiding restricted areas):





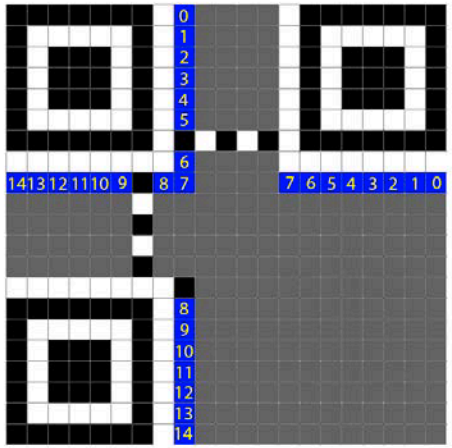


QR Codes



QR Codes

- Format and version information also has to be embedded:



- And the final result is obtained (1-Q code of “HELLO WORLD” encoded in alphanumeric mode):