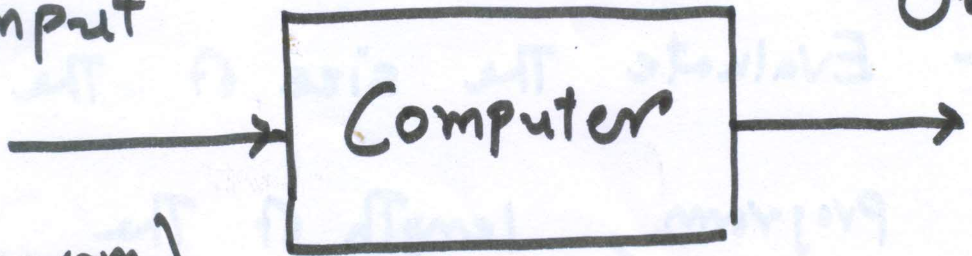
Input 

Output

(Program)

(cause or the intent)

A data pattern
(observation)

Two ways of characterizing the "goodness" or "effectiveness" of the program: 1

i) Computational requirement:

- Minimum possible CPU load in terms of basic computations, e.g. +, ×
- Addressed in a great detail (Big-oh $O$ notation) in an introductory course on Analysis of Algorithms

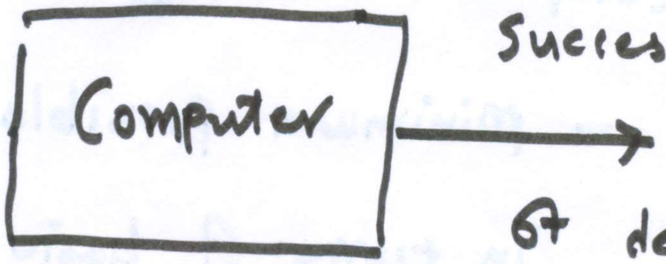
ii) Memory Requirement:

- Evaluate the size of the program, length of the (compiled) program, e.g. in the number of lines of code, in the number of bits of the compiled code. 2

Several Examples in context of (ii)

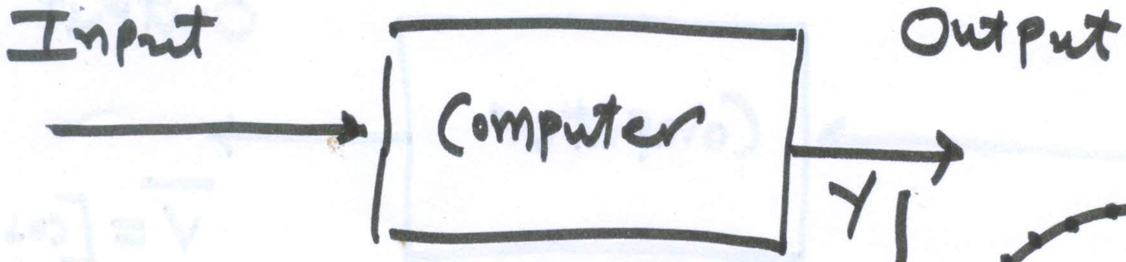
Program 1

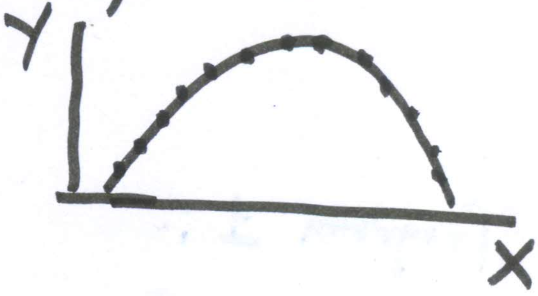
Store these digits in a file and program the computer to read the file and produce the output




Successive digits of decimal expansion of $\pi$

Program 2: Store 22 & 7 and program the computer to do 22/7



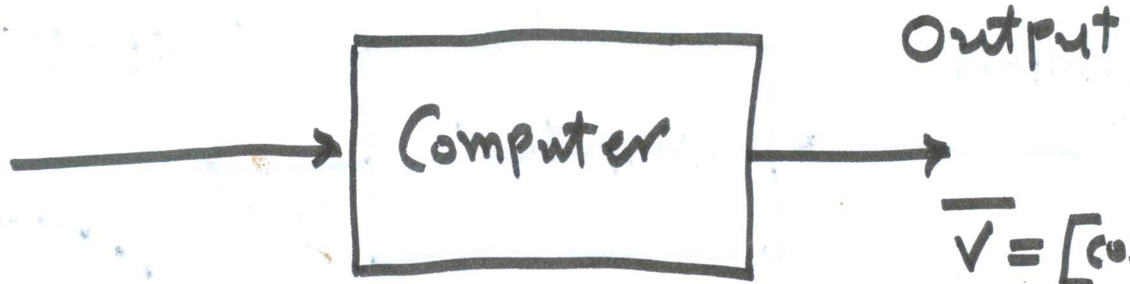




<u>Program 1</u> :

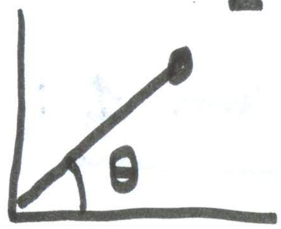
Store each (X, Y) co-ordinate in a file and ask The computer to print each co-ordinate successively.

<u>Program 2</u> (formulated by Sir Isaac Newton):

Program The computer with The parabolic equation, or generally with Newton's Laws of Motion.




$\bar{V} = [\cos\theta, \sin\theta]$




Program 1:

Store in a file

$[\cos\theta, \sin\theta]$

Program 2:

Store only The value of $\theta$

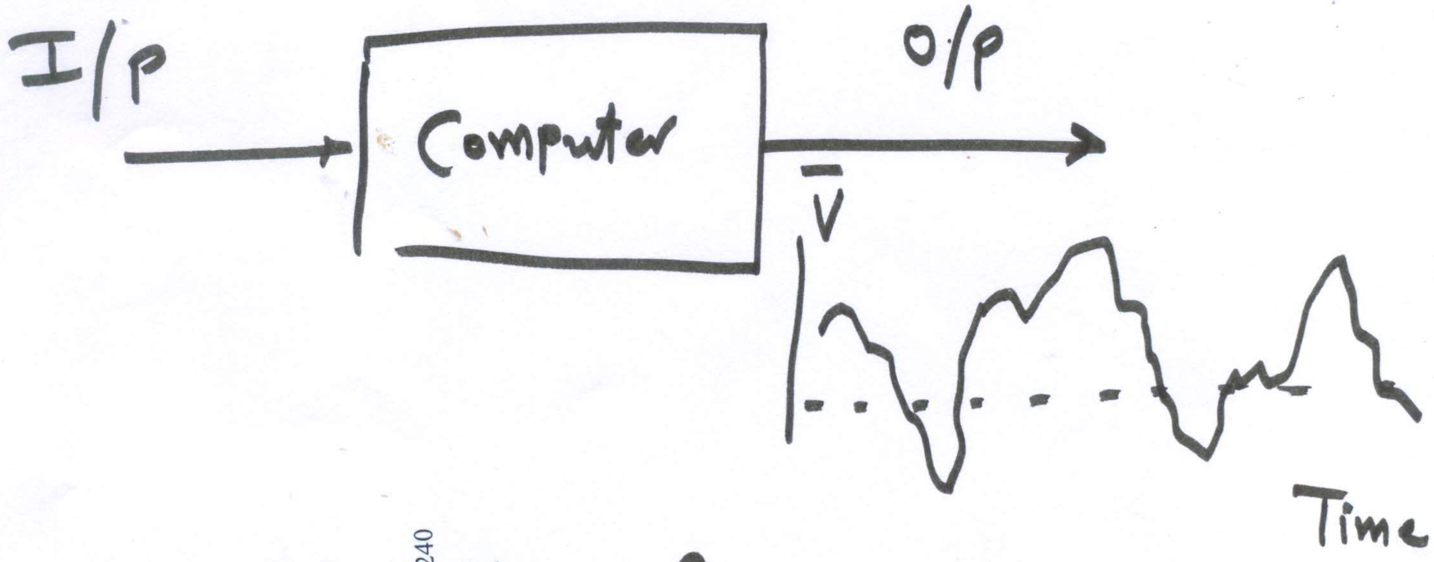
Program 3:

: Store only $\bar{c} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$

: Ask The Computer to do a linear transformation by a matrix $A = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$

$$\bar{V} = A\,\bar{c}$$

4






A certain class of matrices A, (i) Discrete Fourier Transform (DFT); (ii) Discrete Cosine Transfrm (DCT); (iii) Discrete Wavelet Transfrm (DWT)

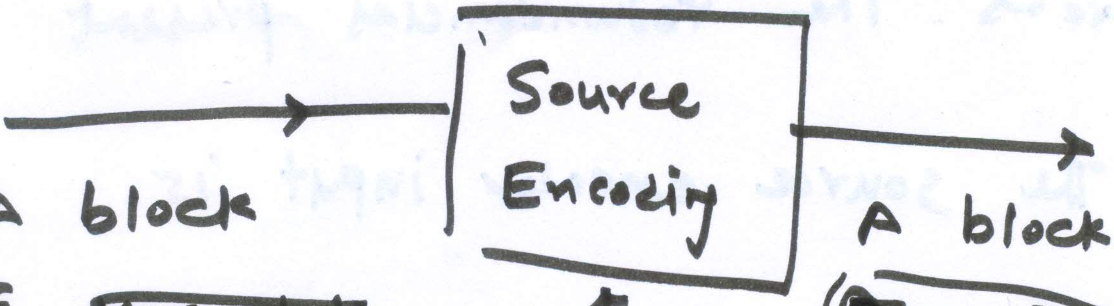
5

$$\bar{V} = A \cdot \bar{C}$$

A complexated ~~set~~ Vector of numbers.

is very simple, mostly made up of few 1's and rest 0's.

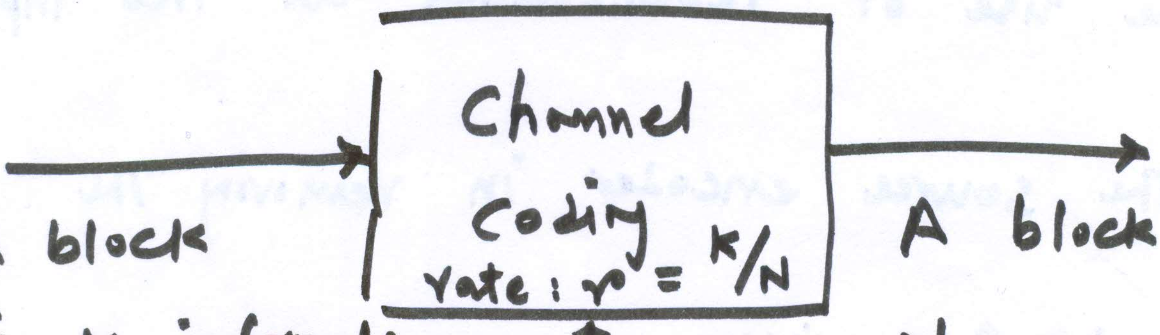
In, nutshell, The source encoder can be represented as follow;




A block of ~~K symbols~~ J information bits; The redundancies may be present

Squeezes out The redundancies

A block of ~~K symbols where~~ K bits where K < J



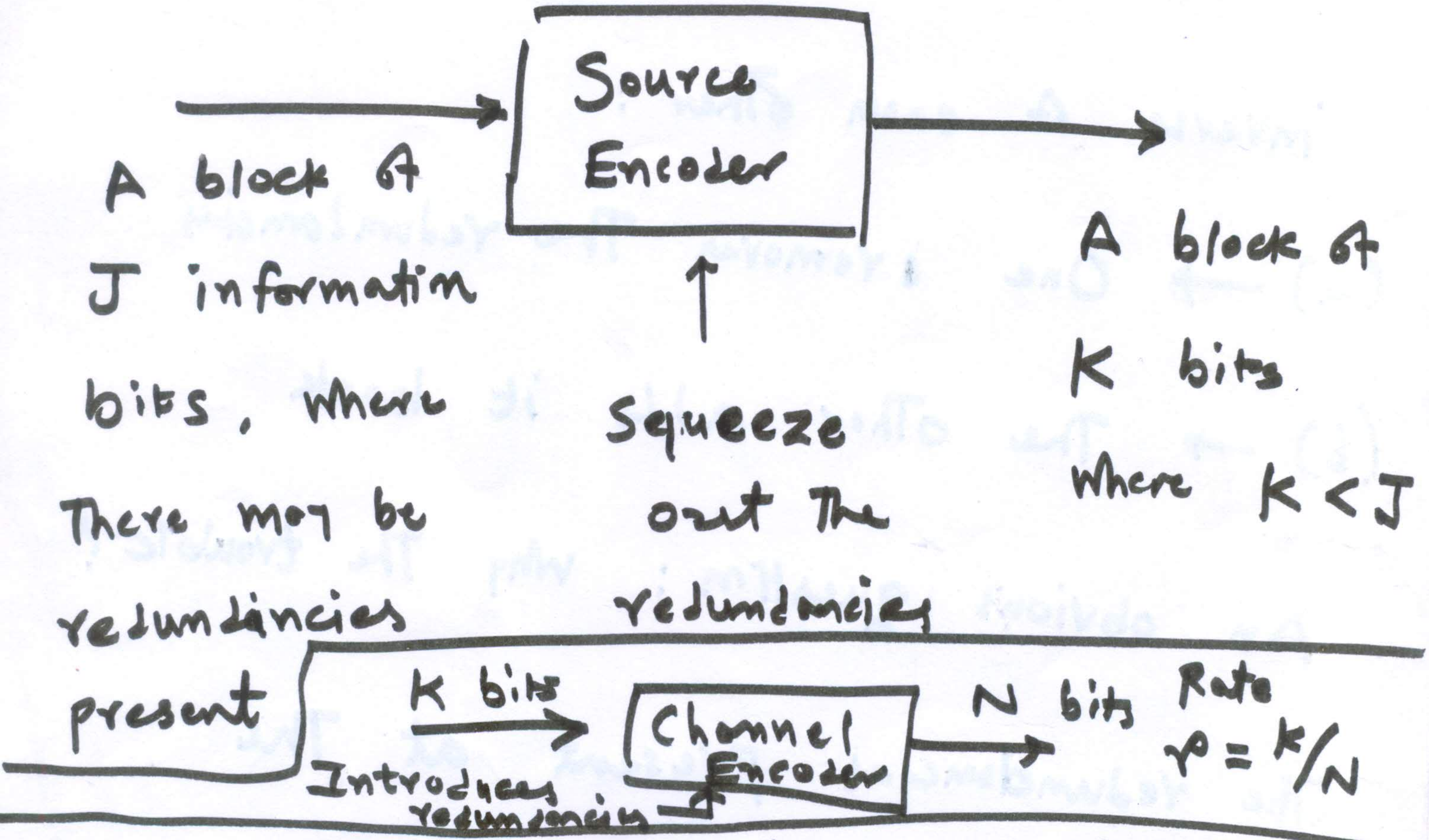

A block of K information bits.

Introduces redundancies

A block of N encoded bits, where N > K

$0 \leqslant r \leqslant 1$

# A summary of Source Encoding:

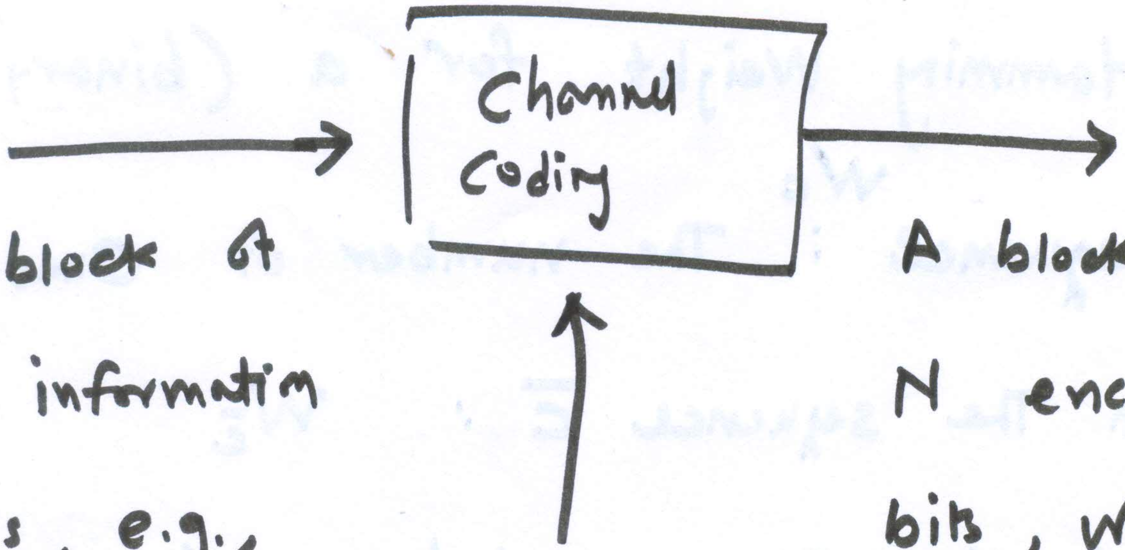



Redundancies in the transmitted / stored data can be viewed in two ways:

+ They provide error detection/correction

− They actually do not convey any information

A summary of channel coding:




A block of K information bits, e.g., at the output of source encoder

Artificially introduces redundancies

A block of N encoded bits, where $N > K$

A metric that defines the amount of redundancy introduced by the channel coder as $r = \frac{K}{N} \in [0, 1]$

Source and channel encoding are dual functions, i.e., They are almost inverse of each other:

(a) → One removes The redundancies

(b) → The other adds it back

An obvious question: Why The trouble?

The redundancies present at The output of The source (in The block of J bits) are not in our control as an engineer / designer.

— ~~[illegible]~~ Maybe The communication channel (e.g., BSC, BEC, Gaussian noise, etc.)

is very weak (i.e., introduces only a small noise),

very strong a very high noise

whereas the redundancies present at the source encoder input is fixed at some level.

— Furthermore, it is very difficult to make use of redundancies at the input to the source encoder in removing the effect of noise.

Define several new terms :

i) Hamming Weight for a (binary) sequence $W_{\bar{c}}$ : The number of Ones in the sequence $\bar{c}$ : $W_{\bar{c}}$

ii) Hamming Distance between two binary sequences $d_H(\bar{c}_1, \bar{c}_2)$ : The number of bits which are different among the two sequences. $\bar{c}_1$ & $\bar{c}_2$

6

$$d_H(\bar{c}_1, \bar{c}_2) = W_{\bar{c}_1 \oplus \bar{c}_2}$$

iii) Minimum Hamming Distance $d_H^{min}$ : minimum of all Hamming Distances between all pairs of a set of binary sequences.

Example 1 : $K = 2$, $N = 2$

Rate $r = \frac{K}{N} = 1$ (uncoded system)

| $d_H$ | | | $W_c$ | $c_1$ | $c_2$ | $c_3$ | $c_4$ |
|---|---|---|---|---|---|---|---|
| $c_1$ | 0 | 0 | 0 | 0 | 1 | 1 | 2 |
| $c_2$ | 0 | 1 | 1 | 1 | 0 | 2 | 1 |
| $c_3$ | 1 | 0 | 1 | 1 | | 0 | 1 |
| $c_4$ | 1 | 1 | 2 | 2 | | | 0 |

$d_H^{min} = 1$ bit

Example 2 : $K = 2$, $N = 3$ SPC code with rate $r = 2/3$

| | | | | $W_c$ | $c_1$ | $c_2$ | $c_3$ | $c_4$ |
|---|---|---|---|---|---|---|---|---|
| $c_1$ | 0 | 0 | 0 | 0 | | 2 | 2 | 2 |
| $c_2$ | 0 | 1 | 1 | 2 | | | 2 | 2 |
| $c_3$ | 1 | 0 | 1 | 2 | | | | 2 |
| $c_4$ | 1 | 1 | 0 | 2 | | | | |

$d_H^{min} = 2$ bits

$t_d$ : maximum # of bits in error (due to the BSC) that the receiver can detect

(8)

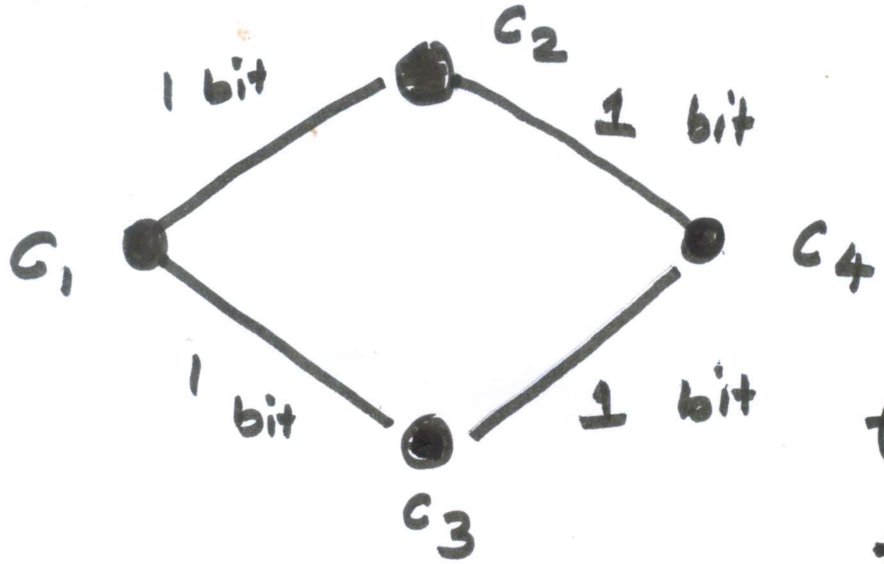
$t_c$ : Maximum # of bits in error that can be corrected at the receiver

---

Rate 1/3 repetition code

| | | $W_c$ | $c_1$ | $c_2$ |
|---|---|---|---|---|
| $c_1$ | 0 0 0 | 0 | 0 | 3 |
| $c_2$ | 1 1 1 | 3 | 3 | 0 |

A map for Example 1

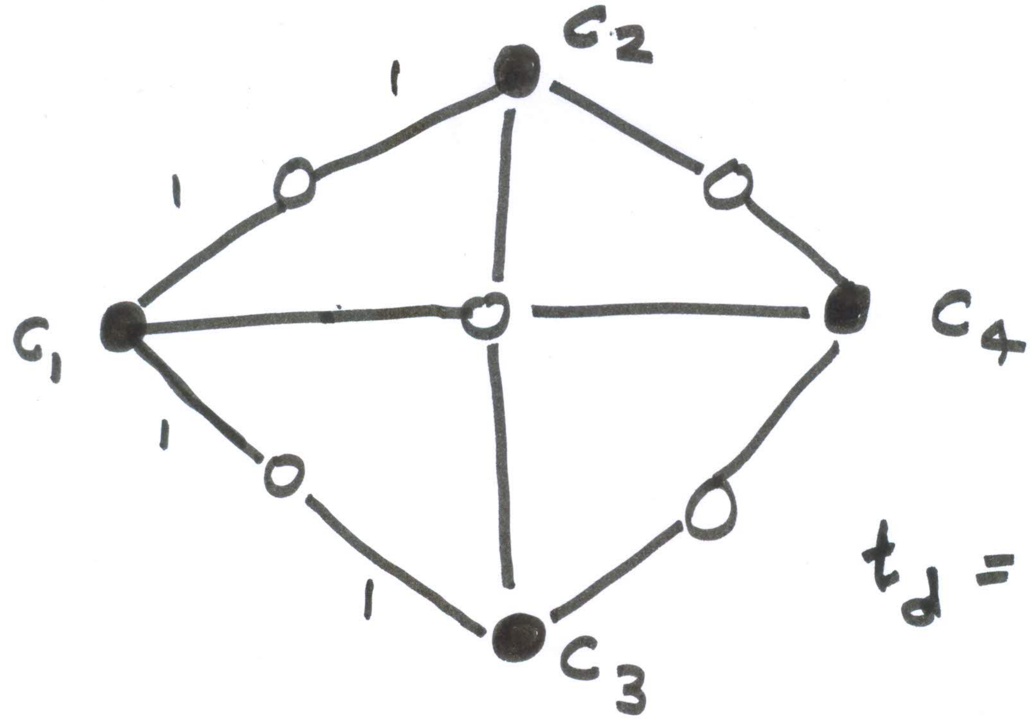



(9)

$t_d = 0$

$t_c = 0$

A map for Example 2 ( $r = \frac{2}{3}$ SPC code)

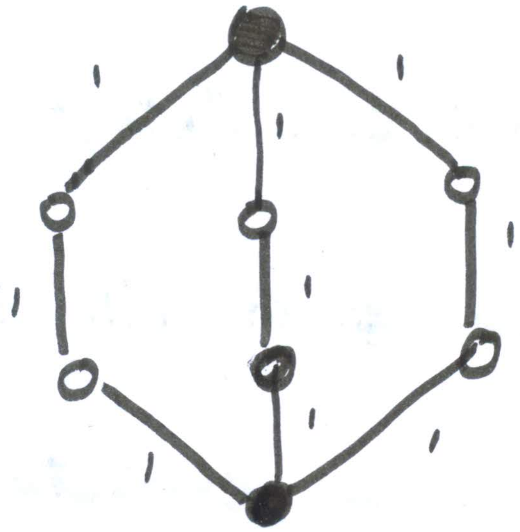



$t_d = 1$ bit

$t_c = 0$ bit

Diagram for the $r = 1/3$ repetition code:

$C_1 = 000$



$C_2 = 111$

$t_d = 2$ bits

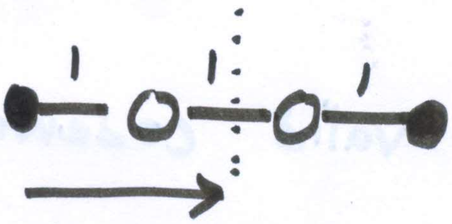
$t_c = 1$ bit.

10

---

In general, $t_d$ & $t_c$ depend on $d_H^{min}$ for the set of codewords.

A simplified diagram:

| $d_H^{min}$ | Example | Simplified Diagram | $t_d$ | $t_c$ |
|---|---|---|---|---|
| 1 bit | No Coding |  | 0 | 0 |
| 2 bits | SPC code |  | 1 | 0 |
| 3 bits | $r\ 1/3$ rep. code. |  $t_c$ bits | 2 | 1 |
| 4 | | | | |
| 5 | | | | |
| 6 | | | | |
| 7 | | | | |

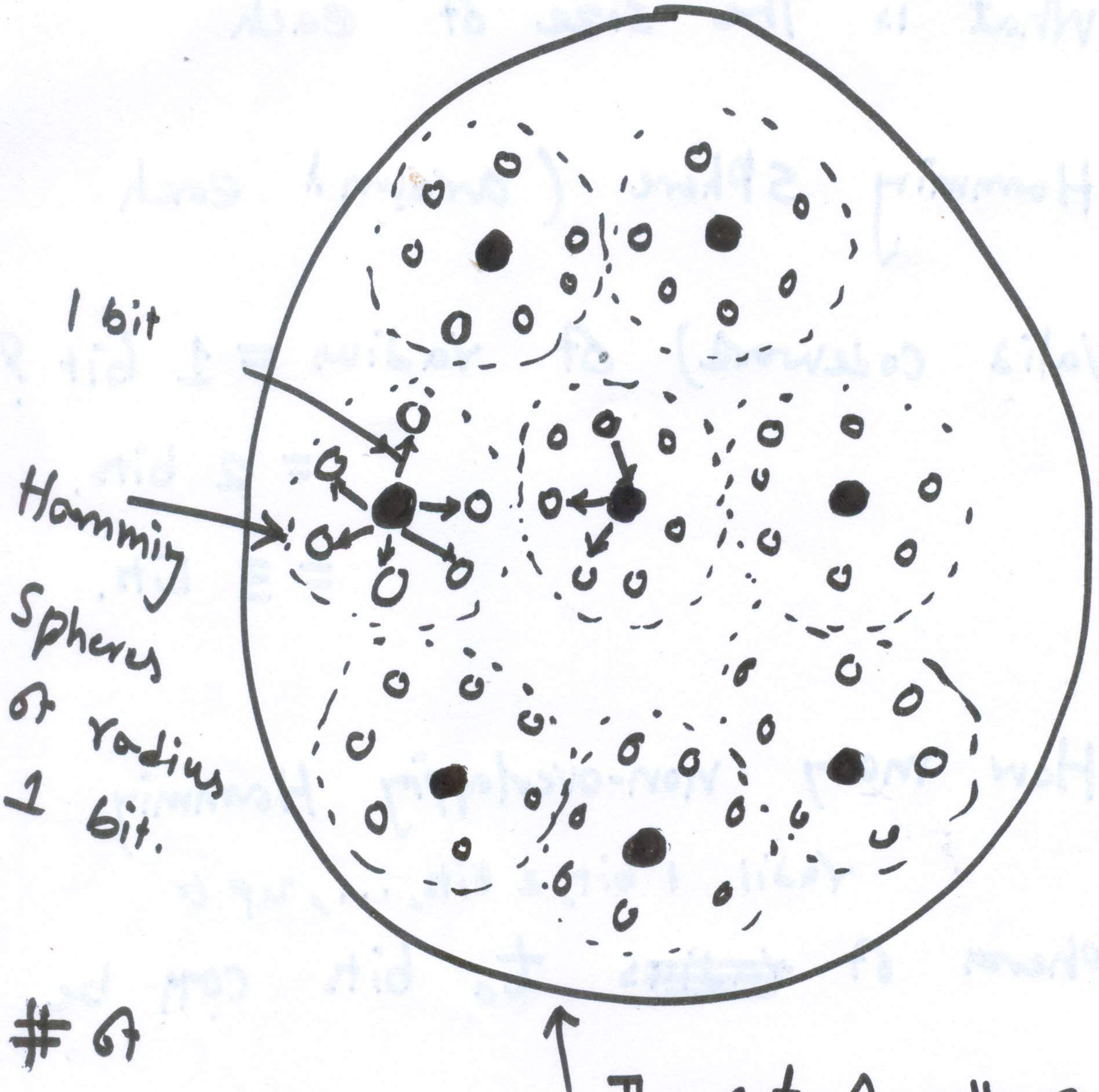
11

Derive expressions for $t_d$ & $t_c$ in terms of $d_H^{min}$

The codewords of rate $r = \frac{K}{N}$ channel code are of length N bits.

$\Rightarrow$ There is a total of $2^N$ possible binary sequences

$\Rightarrow$ However, only $2^K$ of these sequences are valid codewords (i.e., black circles); remaining are never transmitted by the channel encoder (these are the white circles)




13

# 67

Black circles denote the binary sequences (i.e., codewords) sent by the channel encoder. They are also N bits in length.

1.) What is The size of each Hamming sphere (around each valid codeword) of radius = 1 bit?

= 2 bits.

= 3 bits.

2) How many non-overlapping Hamming spheres of ~~radius~~ radii 1 bit, 2 bits, ..., up to $t_c$ bits can be packed inside The large oval of size $2^N$?

An elaboration of Item 1 above:

→ A channel coding

(14)

scheme can correct up to $t_c$ bits of error <u>only if</u> each of The valid codewords is surrounded by $t_c + 1$ concentric Hamming spheres :

→ First sphere has a radius of 0 bits,

i.e., it ~~is~~ contains the codeword itself,

→ The second sphere has a radius of 1 bit,

i.e., it contains all the possible

N bit long sequences that differ from this

codeword by 1 bit

⇒? What is the size of this

Hamming sphere ?

→ The third sphere has a radius of 2 bits,

i.e., it contains all the posible N bit

long sequences that differ from this sequence

by 2 bits

⇒ What is the size of this Hamming

sphere ?

Elaboration of Item (2) on Page 14:

These $t_c + 1$ concentric Hamming spheres whose center is a transmitted codeword (That we have shown in black dot) form The <u>decision region</u> for This codeword.

⇒ The minimum (Hamming) distance decoder will select This codeword as The one likely to have been transmitted if The received sequence ~~is~~ falls in any of These $t_c + 1$ Hamming spheres.

16

→ When The channel coding scheme is $t_c$-bit error correcting scheme, These decision regions for each of The $2^K$ transmitted codewords $\boxed{\text{have to be non-overlapping}}$

The question we want to answer:

What happens to the code-rate $r = \frac{K}{N}$ as the error correction capability $t_c$ of the channel coding scheme is kept on increasing?

→ Intuitive (non-mathematical) answer: $r$ has to reduce (i.e., the redundancies have to increase)

17

→ mathematical answer can be obtained by answering Items 1 and 2 on page 14:

⇒ What is this mathematical answer?