


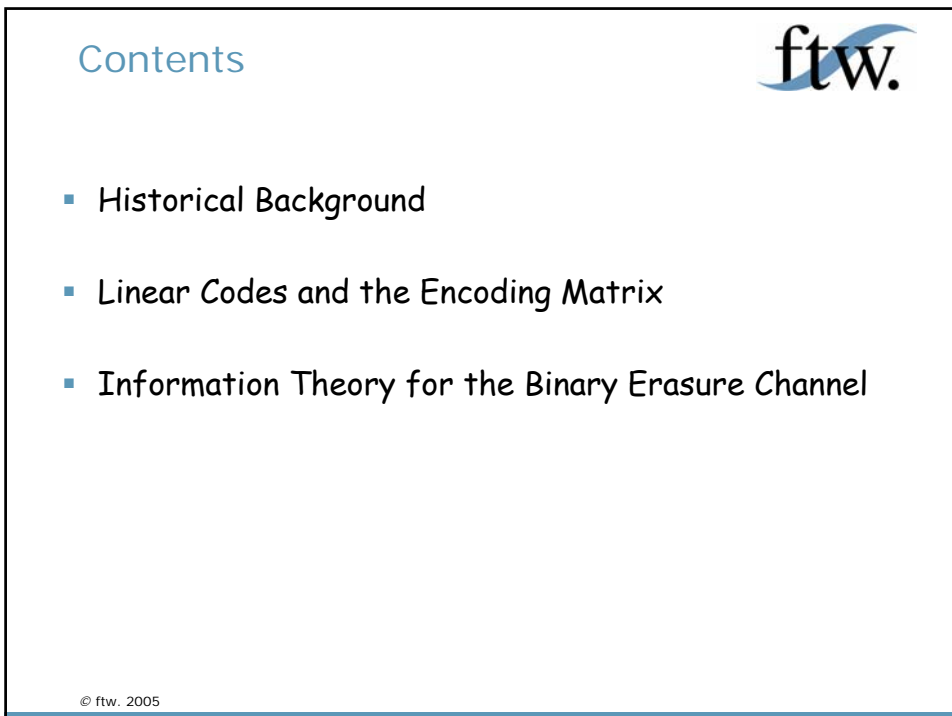
ftw.  
Forschungszentrum Telekommunikation Wien


Theory and Design of  
Turbo and Related Codes  
*Lecture 1*

*Jossy Sayir & Gottfried Lechner*

<http://userver.ftw.at/~jossy/turbo/index.html>

 Kplus  
Kompetenzzentren-Programm



Contents 

- Historical Background
- Linear Codes and the Encoding Matrix
- Information Theory for the Binary Erasure Channel

© ftw. 2005

## Error Correcting Codes



Error Correction is "easy"!

Repeat Codes:

0 → 0000...0

1 → 1111...1

(repeat  $N$  times)

Any desired probability of error can be attained for almost any communication channel!

© ftw. 2005

## Rate vs. Reliability



Is there a Rate / Reliability Tradeoff?

Not for  $\text{Rate} \leq \text{Channel Capacity}$  !!!!  
(Claude E. Shannon, 1948)

© ftw. 2005

# Claude Elwood Shannon 1916 - 2001



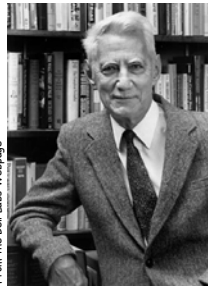
From the homepage of Bill Bertorff

Shannon around 1950



From the Bell Labs Webpage

Shannon playing with  
mechanical mouse



From the Bell Labs Webpage

Shannon in his office  
at Bell Labs

© ftw. 2005

## Figure 1

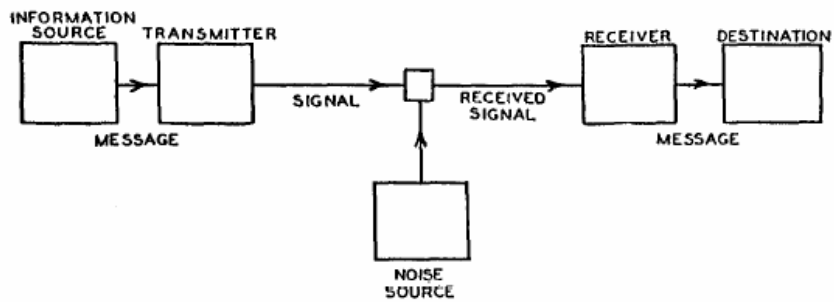


Fig. 1—Schematic diagram of a general communication system.

© ftw. 2005

## The Coding Theorem



It is possible to achieve **arbitrary reliability** (any desired non-zero error probability) while communicating at any data **rate** below **channel capacity**

It is impossible to achieve arbitrary reliability at data rates above channel capacity

But how do we achieve arbitrary reliability? → use coding...

© ftw. 2005



From the website of the U of Michigan

Shannon the prophet



Source unknown

Richard W. Hamming  
1915 - 1998

Worked at Bell Laboratories  
Pioneer of coding research

© ftw. 2005

# ftw. Channel Capacity



Claude Berrou

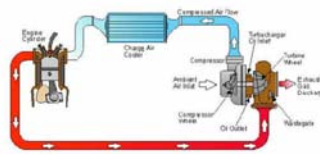


The coding community before 1993... 1993: the invention of Turbo Coding!

© ftw. 2005

## Turbo Coding

NEAR SHANNON LIMIT ERROR-CORRECTING CODING AND BERGER-TURNBOCK'S (1)  
Claude Berrou, Alain Glavieux and Punya Thitimajshima  
Claude Berrou, Integrated Circuits for Telecommunication Laboratory  
Alain Glavieux and Punya Thitimajshima, Digital Communication Laboratory  
Ecole Nationale Supérieure des Télécommunications de Bretagne, France  
(1) France N° 9402279 (France, N° 92060113) (Switzerland, N° 027032483) (USA)



**ABSTRACT**—This paper starts with a new class of convolutional codes called Turbo codes, whose performance in terms of the Error Rate (BER) are close to the SHANNON limit. The Turbo Code consists in using a parallel concatenation of two Recursive Systematic Convolutional codes with an interleaver device, using a feedback decoding rule, is implemented as a repeated identical convolutional decoder.

**1. INTRODUCTION**  
Consider a binary rate  $R=1/2$  convolutional encoder with constraint length  $K$  and memory  $M$  at large SNR. As we know, it is not possible to generate the same way as a classical systematic code with the same memory  $M$  at large SNR. In fact, the only way to generate the same way as a classical systematic code with the same memory  $M$  at large SNR is to use a Turbo code.

A binary rate  $R=1/2$  Turbo code is obtained from a convolutional code by interleaving the two identical rate  $R=1/2$  convolutional codes. For an  $R=1/2$  Turbo code, the input  $x_i$  is equal to the input  $x_{i+1}$  but the output  $y_i$  and  $y_{i+1}$  are different. In fact, the output  $y_i$  (resp.  $y_{i+1}$ ) is equal to the input  $x_i$  (resp.  $x_{i+1}$ ) by substituting  $x_i$  for  $x_{i+1}$  and the variable  $x_i$  is sequentially calculated as:

$$x_i = x_{i+1} \oplus y_{i+1} \quad \text{mod } 2 \quad (1)$$

$$x_{i+1} = x_i \oplus y_i \quad \text{mod } 2 \quad (2)$$

$$x_i = x_{i+1} \oplus y_{i+1} \quad \text{mod } 2 \quad (3)$$

The BER consists with memory that obtained from an NCC decoder defined by generators  $G_1(x), G_2(x)$  is depicted in Fig. 1. Generally, we assume that the input bit  $x_i$  takes values 0 or 1 with the same probability. From equation (1), we can show that variable  $x_i$  exhibits the same statistical property.

$$P_i[x_i = 0] = P_{i+1}[x_{i+1} = 0] = P_i[x_i = 0] = P_{i+1}[x_{i+1} = 0] = 1/2 \quad (4)$$

$$P_i[x_i = 1] = P_{i+1}[x_{i+1} = 1] = P_i[x_i = 1] = P_{i+1}[x_{i+1} = 1] = 1/2 \quad (5)$$

$$P_i[x_i = 0, x_{i+1} = 0] = P_{i+1}[x_{i+1} = 0, x_i = 0] = 1/4 \quad (6)$$

$$P_i[x_i = 0, x_{i+1} = 1] = P_{i+1}[x_{i+1} = 1, x_i = 0] = 1/4 \quad (7)$$

$$P_i[x_i = 1, x_{i+1} = 0] = P_{i+1}[x_{i+1} = 0, x_i = 1] = 1/4 \quad (8)$$

$$P_i[x_i = 1, x_{i+1} = 1] = P_{i+1}[x_{i+1} = 1, x_i = 1] = 1/4 \quad (9)$$

$$P_i[x_i = 0, x_{i+1} = 0, x_{i+2} = 0] = P_{i+2}[x_{i+2} = 0, x_{i+1} = 0, x_i = 0] = 1/8 \quad (10)$$

$$P_i[x_i = 0, x_{i+1} = 0, x_{i+2} = 1] = P_{i+2}[x_{i+2} = 1, x_{i+1} = 0, x_i = 0] = 1/8 \quad (11)$$

$$P_i[x_i = 0, x_{i+1} = 1, x_{i+2} = 0] = P_{i+2}[x_{i+2} = 0, x_{i+1} = 1, x_i = 0] = 1/8 \quad (12)$$

$$P_i[x_i = 0, x_{i+1} = 1, x_{i+2} = 1] = P_{i+2}[x_{i+2} = 1, x_{i+1} = 1, x_i = 0] = 1/8 \quad (13)$$

$$P_i[x_i = 1, x_{i+1} = 0, x_{i+2} = 0] = P_{i+2}[x_{i+2} = 0, x_{i+1} = 0, x_i = 1] = 1/8 \quad (14)$$

$$P_i[x_i = 1, x_{i+1} = 0, x_{i+2} = 1] = P_{i+2}[x_{i+2} = 1, x_{i+1} = 0, x_i = 1] = 1/8 \quad (15)$$

$$P_i[x_i = 1, x_{i+1} = 1, x_{i+2} = 0] = P_{i+2}[x_{i+2} = 0, x_{i+1} = 1, x_i = 1] = 1/8 \quad (16)$$

$$P_i[x_i = 1, x_{i+1} = 1, x_{i+2} = 1] = P_{i+2}[x_{i+2} = 1, x_{i+1} = 1, x_i = 1] = 1/8 \quad (17)$$

$$P_i[x_i = 0, x_{i+1} = 0, x_{i+2} = 0, x_{i+3} = 0] = P_{i+3}[x_{i+3} = 0, x_{i+2} = 0, x_{i+1} = 0, x_i = 0] = 1/16 \quad (18)$$

$$P_i[x_i = 0, x_{i+1} = 0, x_{i+2} = 0, x_{i+3} = 1] = P_{i+3}[x_{i+3} = 1, x_{i+2} = 0, x_{i+1} = 0, x_i = 0] = 1/16 \quad (19)$$

$$P_i[x_i = 0, x_{i+1} = 0, x_{i+2} = 1, x_{i+3} = 0] = P_{i+3}[x_{i+3} = 0, x_{i+2} = 1, x_{i+1} = 0, x_i = 0] = 1/16 \quad (20)$$

$$P_i[x_i = 0, x_{i+1} = 0, x_{i+2} = 1, x_{i+3} = 1] = P_{i+3}[x_{i+3} = 1, x_{i+2} = 1, x_{i+1} = 0, x_i = 0] = 1/16 \quad (21)$$

$$P_i[x_i = 0, x_{i+1} = 1, x_{i+2} = 0, x_{i+3} = 0] = P_{i+3}[x_{i+3} = 0, x_{i+2} = 0, x_{i+1} = 1, x_i = 0] = 1/16 \quad (22)$$

$$P_i[x_i = 0, x_{i+1} = 1, x_{i+2} = 0, x_{i+3} = 1] = P_{i+3}[x_{i+3} = 1, x_{i+2} = 0, x_{i+1} = 1, x_i = 0] = 1/16 \quad (23)$$

$$P_i[x_i = 0, x_{i+1} = 1, x_{i+2} = 1, x_{i+3} = 0] = P_{i+3}[x_{i+3} = 0, x_{i+2} = 1, x_{i+1} = 1, x_i = 0] = 1/16 \quad (24)$$

$$P_i[x_i = 0, x_{i+1} = 1, x_{i+2} = 1, x_{i+3} = 1] = P_{i+3}[x_{i+3} = 1, x_{i+2} = 1, x_{i+1} = 1, x_i = 0] = 1/16 \quad (25)$$

$$P_i[x_i = 1, x_{i+1} = 0, x_{i+2} = 0, x_{i+3} = 0] = P_{i+3}[x_{i+3} = 0, x_{i+2} = 0, x_{i+1} = 0, x_i = 1] = 1/16 \quad (26)$$

$$P_i[x_i = 1, x_{i+1} = 0, x_{i+2} = 0, x_{i+3} = 1] = P_{i+3}[x_{i+3} = 1, x_{i+2} = 0, x_{i+1} = 0, x_i = 1] = 1/16 \quad (27)$$

$$P_i[x_i = 1, x_{i+1} = 0, x_{i+2} = 1, x_{i+3} = 0] = P_{i+3}[x_{i+3} = 0, x_{i+2} = 1, x_{i+1} = 0, x_i = 1] = 1/16 \quad (28)$$

$$P_i[x_i = 1, x_{i+1} = 0, x_{i+2} = 1, x_{i+3} = 1] = P_{i+3}[x_{i+3} = 1, x_{i+2} = 1, x_{i+1} = 0, x_i = 1] = 1/16 \quad (29)$$

$$P_i[x_i = 1, x_{i+1} = 1, x_{i+2} = 0, x_{i+3} = 0] = P_{i+3}[x_{i+3} = 0, x_{i+2} = 0, x_{i+1} = 1, x_i = 1] = 1/16 \quad (30)$$

$$P_i[x_i = 1, x_{i+1} = 1, x_{i+2} = 0, x_{i+3} = 1] = P_{i+3}[x_{i+3} = 1, x_{i+2} = 0, x_{i+1} = 1, x_i = 1] = 1/16 \quad (31)$$

$$P_i[x_i = 1, x_{i+1} = 1, x_{i+2} = 1, x_{i+3} = 0] = P_{i+3}[x_{i+3} = 0, x_{i+2} = 1, x_{i+1} = 1, x_i = 1] = 1/16 \quad (32)$$

$$P_i[x_i = 1, x_{i+1} = 1, x_{i+2} = 1, x_{i+3} = 1] = P_{i+3}[x_{i+3} = 1, x_{i+2} = 1, x_{i+1} = 1, x_i = 1] = 1/16 \quad (33)$$

$$P_i[x_i = 0, x_{i+1} = 0, x_{i+2} = 0, x_{i+3} = 0, x_{i+4} = 0] = P_{i+4}[x_{i+4} = 0, x_{i+3} = 0, x_{i+2} = 0, x_{i+1} = 0, x_i = 0] = 1/32 \quad (34)$$

$$P_i[x_i = 0, x_{i+1} = 0, x_{i+2} = 0, x_{i+3} = 0, x_{i+4} = 1] = P_{i+4}[x_{i+4} = 1, x_{i+3} = 0, x_{i+2} = 0, x_{i+1} = 0, x_i = 0] = 1/32 \quad (35)$$

$$P_i[x_i = 0, x_{i+1} = 0, x_{i+2} = 0, x_{i+3} = 1, x_{i+4} = 0] = P_{i+4}[x_{i+4} = 0, x_{i+3} = 1, x_{i+2} = 0, x_{i+1} = 0, x_i = 0] = 1/32 \quad (36)$$

$$P_i[x_i = 0, x_{i+1} = 0, x_{i+2} = 0, x_{i+3} = 1, x_{i+4} = 1] = P_{i+4}[x_{i+4} = 1, x_{i+3} = 1, x_{i+2} = 0, x_{i+1} = 0, x_i = 0] = 1/32 \quad (37)$$

$$P_i[x_i = 0, x_{i+1} = 0, x_{i+2} = 1, x_{i+3} = 0, x_{i+4} = 0] = P_{i+4}[x_{i+4} = 0, x_{i+3} = 0, x_{i+2} = 1, x_{i+1} = 0, x_i = 0] = 1/32 \quad (38)$$

$$P_i[x_i = 0, x_{i+1} = 0, x_{i+2} = 1, x_{i+3} = 0, x_{i+4} = 1] = P_{i+4}[x_{i+4} = 1, x_{i+3} = 0, x_{i+2} = 1, x_{i+1} = 0, x_i = 0] = 1/32 \quad (39)$$

$$P_i[x_i = 0, x_{i+1} = 0, x_{i+2} = 1, x_{i+3} = 1, x_{i+4} = 0] = P_{i+4}[x_{i+4} = 0, x_{i+3} = 1, x_{i+2} = 1, x_{i+1} = 0, x_i = 0] = 1/32 \quad (40)$$

$$P_i[x_i = 0, x_{i+1} = 0, x_{i+2} = 1, x_{i+3} = 1, x_{i+4} = 1] = P_{i+4}[x_{i+4} = 1, x_{i+3} = 1, x_{i+2} = 1, x_{i+1} = 0, x_i = 0] = 1/32 \quad (41)$$

$$P_i[x_i = 0, x_{i+1} = 1, x_{i+2} = 0, x_{i+3} = 0, x_{i+4} = 0] = P_{i+4}[x_{i+4} = 0, x_{i+3} = 0, x_{i+2} = 0, x_{i+1} = 1, x_i = 0] = 1/32 \quad (42)$$

$$P_i[x_i = 0, x_{i+1} = 1, x_{i+2} = 0, x_{i+3} = 0, x_{i+4} = 1] = P_{i+4}[x_{i+4} = 1, x_{i+3} = 0, x_{i+2} = 0, x_{i+1} = 1, x_i = 0] = 1/32 \quad (43)$$

$$P_i[x_i = 0, x_{i+1} = 1, x_{i+2} = 0, x_{i+3} = 1, x_{i+4} = 0] = P_{i+4}[x_{i+4} = 0, x_{i+3} = 1, x_{i+2} = 0, x_{i+1} = 1, x_i = 0] = 1/32 \quad (44)$$

$$P_i[x_i = 0, x_{i+1} = 1, x_{i+2} = 0, x_{i+3} = 1, x_{i+4} = 1] = P_{i+4}[x_{i+4} = 1, x_{i+3} = 1, x_{i+2} = 0, x_{i+1} = 1, x_i = 0] = 1/32 \quad (45)$$

$$P_i[x_i = 0, x_{i+1} = 1, x_{i+2} = 1, x_{i+3} = 0, x_{i+4} = 0] = P_{i+4}[x_{i+4} = 0, x_{i+3} = 0, x_{i+2} = 1, x_{i+1} = 1, x_i = 0] = 1/32 \quad (46)$$

$$P_i[x_i = 0, x_{i+1} = 1, x_{i+2} = 1, x_{i+3} = 0, x_{i+4} = 1] = P_{i+4}[x_{i+4} = 1, x_{i+3} = 0, x_{i+2} = 1, x_{i+1} = 1, x_i = 0] = 1/32 \quad (47)$$

$$P_i[x_i = 0, x_{i+1} = 1, x_{i+2} = 1, x_{i+3} = 1, x_{i+4} = 0] = P_{i+4}[x_{i+4} = 0, x_{i+3} = 1, x_{i+2} = 1, x_{i+1} = 1, x_i = 0] = 1/32 \quad (48)$$

$$P_i[x_i = 0, x_{i+1} = 1, x_{i+2} = 1, x_{i+3} = 1, x_{i+4} = 1] = P_{i+4}[x_{i+4} = 1, x_{i+3} = 1, x_{i+2} = 1, x_{i+1} = 1, x_i = 0] = 1/32 \quad (49)$$

$$P_i[x_i = 1, x_{i+1} = 0, x_{i+2} = 0, x_{i+3} = 0, x_{i+4} = 0] = P_{i+4}[x_{i+4} = 0, x_{i+3} = 0, x_{i+2} = 0, x_{i+1} = 0, x_i = 1] = 1/32 \quad (50)$$

$$P_i[x_i = 1, x_{i+1} = 0, x_{i+2} = 0, x_{i+3} = 0, x_{i+4} = 1] = P_{i+4}[x_{i+4} = 1, x_{i+3} = 0, x_{i+2} = 0, x_{i+1} = 0, x_i = 1] = 1/32 \quad (51)$$

$$P_i[x_i = 1, x_{i+1} = 0, x_{i+2} = 0, x_{i+3} = 1, x_{i+4} = 0] = P_{i+4}[x_{i+4} = 0, x_{i+3} = 1, x_{i+2} = 0, x_{i+1} = 0, x_i = 1] = 1/32 \quad (52)$$

$$P_i[x_i = 1, x_{i+1} = 0, x_{i+2} = 0, x_{i+3} = 1, x_{i+4} = 1] = P_{i+4}[x_{i+4} = 1, x_{i+3} = 1, x_{i+2} = 0, x_{i+1} = 0, x_i = 1] = 1/32 \quad (53)$$

$$P_i[x_i = 1, x_{i+1} = 0, x_{i+2} = 1, x_{i+3} = 0, x_{i+4} = 0] = P_{i+4}[x_{i+4} = 0, x_{i+3} = 0, x_{i+2} = 1, x_{i+1} = 0, x_i = 1] = 1/32 \quad (54)$$

$$P_i[x_i = 1, x_{i+1} = 0, x_{i+2} = 1, x_{i+3} = 0, x_{i+4} = 1] = P_{i+4}[x_{i+4} = 1, x_{i+3} = 0, x_{i+2} = 1, x_{i+1} = 0, x_i = 1] = 1/32 \quad (55)$$

$$P_i[x_i = 1, x_{i+1} = 0, x_{i+2} = 1, x_{i+3} = 1, x_{i+4} = 0] = P_{i+4}[x_{i+4} = 0, x_{i+3} = 1, x_{i+2} = 1, x_{i+1} = 0, x_i = 1] = 1/32 \quad (56)$$

$$P_i[x_i = 1, x_{i+1} = 0, x_{i+2} = 1, x_{i+3} = 1, x_{i+4} = 1] = P_{i+4}[x_{i+4} = 1, x_{i+3} = 1, x_{i+2} = 1, x_{i+1} = 0, x_i = 1] = 1/32 \quad (57)$$

$$P_i[x_i = 1, x_{i+1} = 1, x_{i+2} = 0, x_{i+3} = 0, x_{i+4} = 0] = P_{i+4}[x_{i+4} = 0, x_{i+3} = 0, x_{i+2} = 0, x_{i+1} = 1, x_i = 1] = 1/32 \quad (58)$$

$$P_i[x_i = 1, x_{i+1} = 1, x_{i+2} = 0, x_{i+3} = 0, x_{i+4} = 1] = P_{i+4}[x_{i+4} = 1, x_{i+3} = 0, x_{i+2} = 0, x_{i+1} = 1, x_i = 1] = 1/32 \quad (59)$$

$$P_i[x_i = 1, x_{i+1} = 1, x_{i+2} = 0, x_{i+3} = 1, x_{i+4} = 0] = P_{i+4}[x_{i+4} = 0, x_{i+3} = 1, x_{i+2} = 0, x_{i+1} = 1, x_i = 1] = 1/32 \quad (60)$$

$$P_i[x_i = 1, x_{i+1} = 1, x_{i+2} = 0, x_{i+3} = 1, x_{i+4} = 1] = P_{i+4}[x_{i+4} = 1, x_{i+3} = 1, x_{i+2} = 0, x_{i+1} = 1, x_i = 1] = 1/32 \quad (61)$$

$$P_i[x_i = 1, x_{i+1} = 1, x_{i+2} = 1, x_{i+3} = 0, x_{i+4} = 0] = P_{i+4}[x_{i+4} = 0, x_{i+3} = 0, x_{i+2} = 1, x_{i+1} = 1, x_i = 1] = 1/32 \quad (62)$$

$$P_i[x_i = 1, x_{i+1} = 1, x_{i+2} = 1, x_{i+3} = 0, x_{i+4} = 1] = P_{i+4}[x_{i+4} = 1, x_{i+3} = 0, x_{i+2} = 1, x_{i+1} = 1, x_i = 1] = 1/32 \quad (63)$$

$$P_i[x_i = 1, x_{i+1} = 1, x_{i+2} = 1, x_{i+3} = 1, x_{i+4} = 0] = P_{i+4}[x_{i+4} = 0, x_{i+3} = 1, x_{i+2} = 1, x_{i+1} = 1, x_i = 1] = 1/32 \quad (64)$$

$$P_i[x_i = 1, x_{i+1} = 1, x_{i+2} = 1, x_{i+3} = 1, x_{i+4} = 1] = P_{i+4}[x_{i+4} = 1, x_{i+3} = 1, x_{i+2} = 1, x_{i+1} = 1, x_i = 1] = 1/32 \quad (65)$$

$$P_i[x_i = 0, x_{i+1} = 0, x_{i+2} = 0, x_{i+3} = 0, x_{i+4} = 0, x_{i+5} = 0] = P_{i+5}[x_{i+5} = 0, x_{i+4} = 0, x_{i+3} = 0, x_{i+2} = 0, x_{i+1} = 0, x_i = 0] = 1/64 \quad (66)$$

$$P_i[x_i = 0, x_{i+1} = 0, x_{i+2} = 0, x_{i+3} = 0, x_{i+4} = 0, x_{i+5} = 1] = P_{i+5}[x_{i+5} = 1, x_{i+4} = 0, x_{i+3} = 0, x_{i+2} = 0, x_{i+1} = 0, x_i = 0] = 1/64 \quad (67)$$

$$P_i[x_i = 0, x_{i+1} = 0, x_{i+2} = 0, x_{i+3} = 0, x_{i+4} = 1, x_{i+5} = 0] = P_{i+5}[x_{i+5} = 0, x_{i+4} = 1, x_{i+3} = 0, x_{i+2} = 0, x_{i+1} = 0, x_i = 0] = 1/64 \quad (68)$$

$$P_i[x_i = 0, x_{i+1} = 0, x_{i+2} = 0, x_{i+3} = 0, x_{i+4} = 1, x_{i+5} = 1] = P_{i+5}[x_{i+5} = 1, x_{i+4} = 1, x_{i+3} = 0, x_{i+2} = 0, x_{i+1} = 0, x_i = 0] = 1/64 \quad (69)$$

$$P_i[x_i = 0, x_{i+1} = 0, x_{i+2} = 0, x_{i+3} = 1, x_{i+4} = 0, x_{i+5} = 0] = P_{i+5}[x_{i+5} = 0, x_{i+4} = 0, x_{i+3} = 1, x_{i+2} = 0, x_{i+1} = 0, x_i = 0] = 1/64 \quad (70)$$

$$P_i[x_i = 0, x_{i+1} = 0, x_{i+2} = 0, x_{i+3} = 1, x_{i+4} = 0, x_{i+5} = 1] = P_{i+5}[x_{i+5} = 1, x_{i+4} = 0, x_{i+3} = 1, x_{i+2} = 0, x_{i+1} = 0, x_i = 0] = 1/64 \quad (71)$$

$$P_i[x_i = 0, x_{i+1} = 0, x_{i+2} = 0, x_{i+3} = 1, x_{i+4} = 1, x_{i+5} = 0] = P_{i+5}[x_{i+5} = 0, x_{i+4} = 1, x_{i+3} = 1, x_{i+2} = 0, x_{i+1} = 0, x_i = 0] = 1/64 \quad (72)$$

$$P_i[x_i = 0, x_{i+1} = 0, x_{i+2} = 0, x_{i+3} = 1, x_{i+4} = 1, x_{i+5} = 1] = P_{i+5}[x_{i+5} = 1, x_{i+4} = 1, x_{i+3} = 1, x_{i+2} = 0, x_{i+1} = 0, x_i = 0] = 1/64 \quad (73)$$

$$P_i[x_i = 0, x_{i+1} = 0, x_{i+2} = 1, x_{i+3} = 0, x_{i+4} = 0, x_{i+5} = 0] = P_{i+5}[x_{i+5} = 0, x_{i+4} = 0, x_{i+3} = 0, x_{i+2} = 1, x_{i+1} = 0, x_i = 0] = 1/64 \quad (74)$$

$$P_i[x_i = 0, x_{i+1} = 0, x_{i+2} = 1, x_{i+3} = 0, x_{i+4} = 0, x_{i+5} = 1] = P_{i+5}[x_{i+5} = 1, x_{i+4} = 0, x_{i+3} = 0, x_{i+2} = 1, x_{i+1} = 0, x_i = 0] = 1/64 \quad (75)$$

$$P_i[x_i = 0, x_{i+1} = 0, x_{i+2} = 1, x_{i+3} = 0, x_{i+4} = 1, x_{i+5} = 0] = P_{i+5}[x_{i+5} = 0, x_{i+4} = 1, x_{i+3} = 0, x_{i+2} = 1, x_{i+1} = 0, x_i = 0] = 1/64 \quad (76)$$

$$P_i[x_i = 0, x_{i+1} = 0, x_{i+2} = 1, x_{i+3} = 0, x_{i+4} = 1, x_{i+5} = 1] = P_{i+5}[x_{i+5} = 1, x_{i+4} = 1, x_{i+3} = 0, x_{i+2} = 1, x_{i+1} = 0, x_i = 0] = 1/64 \quad (77)$$

&lt;

## Low-Density Parity-Check Coding



Invented by Robert G. Gallager  
in his PhD thesis, MIT, 1963



From the Website of the Edward Rhein Stiftung

(re-discovered by MacKay  
from Cambridge, England  
in 1999)

Bob Gallager

© ftw. 2005

## Repeat Accumulate Codes



Invented by Hui Jin and Robert J. McEliece  
from the California Institute of Technology  
(CalTech) in 1998 / 2000

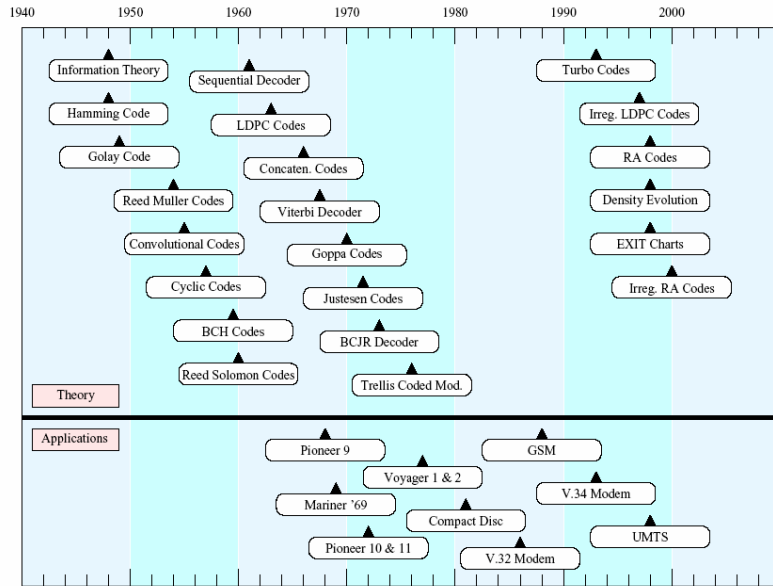


From the Website of CalTech

Bob McEliece

© ftw. 2005

## Coding Timeline



© ftw. 2005

## Course Contents



- History of coding pre- and post-turbo
- Information Theory for the Binary Erasure Channel
- Linear Coding Basics
- Low-Density Parity-Check Codes
- Channels and Decoders
- Factor Graphs and Iterative Decoding
- Irregular Codes, AND-OR Trees and Density Evolution for the BEC
- Decoding on Trees and Trellises
- Turbo Codes
- Encoding and Decoding Complexity
- Repeat-Accumulate Codes
- Fundamentals of Information Theory
- EXIT Charts
- Code Design with EXIT Charts
- Applications, Implementation Issues and Further Extensions

© ftw. 2005

## Something to think about...



$$(x_1 x_2 \dots x_{16}) = (u_1 u_2 \dots u_{10}) \begin{pmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 \end{pmatrix}$$

$$x = (1 \ 1 \ X \ X \ 1 \ 0 \ 0 \ X \ X \ 0 \ 0 \ 0 \ X \ X \ 0 \ 1)$$

- 1.) What is  $(u_1 \ u_2 \dots u_{10})$ ?
- 2.) What happens if there are less erasures?
- 3.) What happens if there are more erasures?

© ftw. 2005

## New Example



$$(x_1 \ x_2 \dots \ x_{16}) = (u_1 \dots \ u_6) \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \end{pmatrix}$$

$$\text{Received: } (1 \ X \ X \ X \ X \ 1 \ 0 \ 0 \ 0 \ 1 \ X \ 1 \ X \ X \ X \ 1)$$

Remove columns corresponding to erasures...

$$(x_1 \ x_6 \ x_7 \ x_8 \ x_9 \ x_{10} \ x_{12} \ x_{16}) = (u_1 \dots \ u_6) \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \end{pmatrix}$$

© ftw. 2005

## New Example (continued)



Drop last 2 columns:  $(x_1 \ x_6 \ x_7 \ x_8 \ x_9 \ x_{10}) = (u_1 \dots u_6)$   
 (system of equations  
 is overdetermined)

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

Triangulize problem:

$$(x_1 \ x_6 \ x_7 \ x_8 \ x_9 \ x_{10}) = (u_1' \dots u_6')$$

$$\begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{matrix} g_3 \\ g_4 \\ g_2 \\ g_6 \\ g_1 \\ g_1+g_4+g_5+g_6 \end{matrix}$$

$$\begin{aligned} \text{Solution: } (x_1 \ x_6 \ x_7 \ x_8 \ x_9 \ x_{10}) &= (1 \ 1 \ 0 \ 0 \ 0 \ 1) \\ &= g_1' + g_2' + g_3' + g_4' + g_5' + g_6' \\ &= g_2 + g_3 + g_5 \\ (u_1 \ u_2 \ u_3 \ u_4 \ u_5 \ u_6) &= (0 \ 1 \ 1 \ 0 \ 1 \ 0) \end{aligned}$$

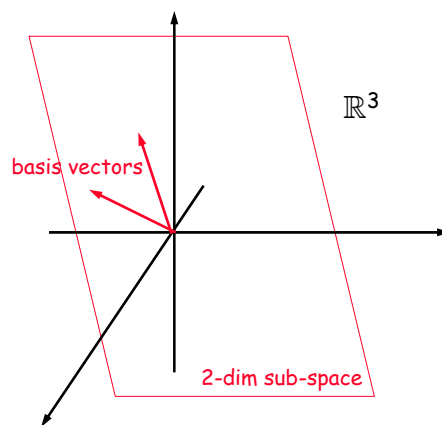
© ftw. 2005

## Binary Linear Codes



A binary linear code  $V$  is  
 a  $K$ -dimensional sub-space  
 of  $GF(2)^N$

$GF(2)$ : Galois Field  $\{0,1\}$   
 with addition and  
 multiplication modulo 2



The sum of any two codewords must give a codeword  
 A linear code contains the all zero codeword

© ftw. 2005

## Simple Codes



- $N = 2, K = 1, \{01, 10\}$
- $N = 2, K = 1, \{00, 11\}$
- $N = 2, K = 1, \{00, 10\}$
- $N = 3, K = 1, \{000, 111\}$
- $N = 3, K = 1, \{000, 011, 110, 111\}$
- $N = 3, K = 2, \{000, 011, 110, 101\}$

Two of the codes in the list are not binary linear codes. Can you tell which?

Elements of  $GF(2)^N$  are alternatively called **words**, **vectors**, binary **N-tuples**, binary **sequences**

An element **v** of a code **V** is usually called codeword

© ftw. 2005

## Encoding Matrix



$$x = uG$$

An encoding matrix specifies the **encoder**, i.e., the assignment of information words to the codewords of the code **V**

- **G** is an  $K \times N$  matrix
- the rows  $g_1, g_2, \dots, g_k$  of **G** are basis vectors of **V**
- the rows of **G** must be linearly independent

The Rate **R** of the encoder/code is  
 $R = K / N$

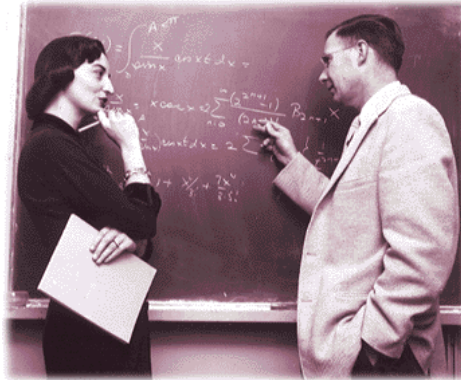
© ftw. 2005

## Distance and Weight



The Hamming Distance between two N-tuples is the number of positions in which they differ

The Hamming Weight of an N-tuple is the number of positions in which it is non-zero



From the homepage <http://www.cs.mcgill.ca/~srosser/hamming.html>

R. Hamming explaining distance? Around 1948...

© ftw. 2005

## Examples



$$d(01001011, 11000011) = 2$$

$$w(00101001) = 3$$

$$d(x,y) = w(x-y) = w(x+y)$$

For a linear code, the number of codewords at Hamming distance  $i$  from any codeword  $v$  is equal to the number  $A_i$  of codewords of Hamming weight  $i$ .

Proof: for every codeword  $v'$  at distance  $i$  from  $v$ ,  $v+v'$  has weight  $i$ . For every codeword  $v''$  of weight  $i$ ,  $d(v,v+v'') = i$ .

© ftw. 2005

## Minimum Distance



The minimum distance  $d_{\min}$  of a code is the smallest distance between any two codewords

For a linear code:

$$\text{minimum distance } d_{\min} = \text{minimum weight } w_{\min}$$

Minimum weight and weight profile are important characteristics to determine the performance of a linear code

© ftw. 2005

## Example



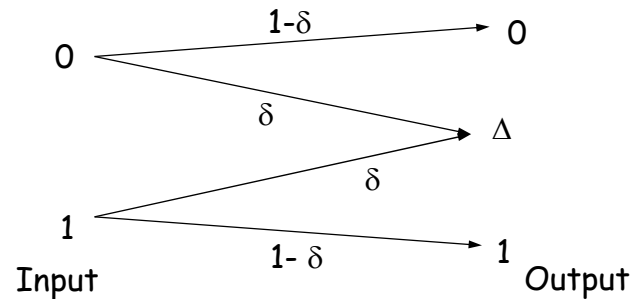
Here is an encoder matrix for the code  $V$

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

Can you think of a trivial upper bound for  $d_{\min}$ ?

© ftw. 2005

## The Binary Erasure Channel



© ftw. 2005

## The Big Question



We have learned how to decode a codeword that has been transmitted over a Binary Erasure Channel (by solving a system of equations, whenever possible)

What is the probability of decoding successfully, in function of the erasure probability  $\delta$ , the code rate  $R$  and the length  $N$ ?

We will suppose that the code  $V$  has been selected at random by choosing an encoding matrix  $G$  at random among all binary  $K \times N$  matrices ( $K = NR$ )

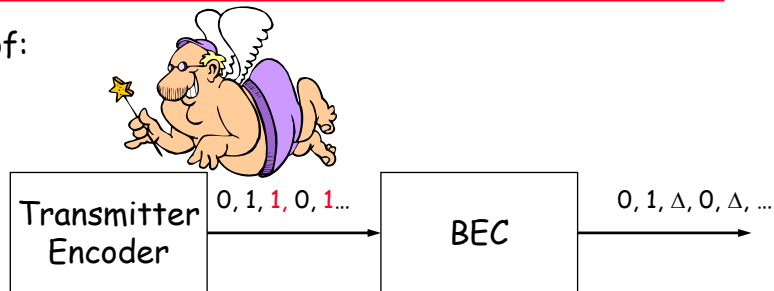
© ftw. 2005

## Converse Coding Theorem



If the information rate  $R$  is higher than  $1-\delta$ , the transmission cannot be arbitrarily reliable

Proof:



Even if a genie tells the transmitter where the erasures will be, the best strategy is to place the data uncoded in all positions that won't be erased. We can only transmit  $1-\delta$  bits per use on average this way.

© ftw. 2005



Let us assume  $R = K/N = 1-\delta-\rho$ . Furthermore, assume the  $K \times N$  encoder matrix has been chosen at random.

On average, there will be  $\delta N = N - K - \rho N$  erasures per block. Our decoding method (from Example 1) will work if the remaining  $K + \rho N$  columns of the encoder matrix are linearly independent, i.e., if the resulting  $K \times (K + \rho N)$  matrix has rank  $K$ .

What is the probability that a  $L \times M$  matrix ( $L \leq M$ ) chosen at random has rank  $L$ ?

© ftw. 2005



If we construct the random matrix row by row, the rule is that the next row is not allowed to be linearly dependent on the previous. Therefore, if we have chosen  $i$  rows so far, we have  $2^M - 2^i$  possible choices for the next row. The number of  $L \times M$  linearly independent matrices is thus:

$$\prod_{i=0}^{L-1} (2^M - 2^i)$$

If our  $L \times M$  matrix is chosen at random among the  $2^{L \times M}$  binary matrices, then the probability that it have rank  $L$  is:

$$P(\text{rank } L) = \frac{\prod_{i=0}^{L-1} (2^M - 2^i)}{2^{L \times M}} = \prod_{i=M-L+1}^M (1 - 2^{-i})$$

© ftw. 2005



There appears to be no closed form expression for this product. If  $L = M$ , we have:

$$P(\text{full rank}) = \frac{1}{2} \frac{3}{4} \frac{7}{8} \frac{15}{16} \frac{31}{32} \frac{63}{64} \dots \frac{2^L - 1}{2^L}$$

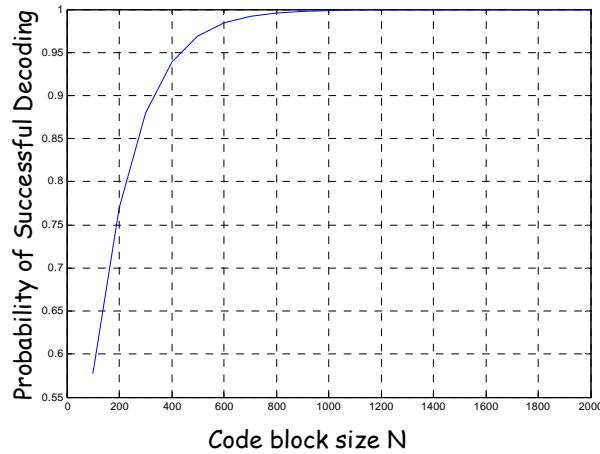
Therefore, if  $K \rightarrow \infty$  and  $R = 1 - \delta$  ( $\rho = 0$ ), then the probability of successful decoding will tend towards 0.2887880950866...

If  $M = L + 1$ , the product starts with  $3/4$ , if  $M = L + 2$ , it starts with  $7/8$ , etc... In our case,  $M = K + \rho N = N(1 - \rho)$  and  $L = (1 - \delta - \rho)N$ .

For a fixed  $\delta$  and  $\rho$  ( $R = 1 - \delta - \rho$ ), we can now plot the probability of successful decoding in function of  $N$ , as the product of  $(1 - 2^{-i})$  for  $i$  from  $\rho N + 1$  to  $\rho N + N$ .

© ftw. 2005

## Prob. of successful decoding



$$\rho = 1 - \delta - R = .01$$

(i.e., R is .01 less than the maximum possible rate)

Solution independent of  $\delta$ !!

Thanks Ralf!

© ftw. 2005

## Information Theory for the Binary Erasure Channel



Peter Elias  
1923 - 2001

Professor at MIT,  
one of the most original  
and prolific researchers  
in information theory,  
inventor of convolutional  
codes

Presented coding  
theorems for linear codes  
for the BEC in London,  
1955 (without proof...)



© ftw. 2005

## What we have learned...



For any information rate  $R < 1 - \delta$ , it is possible to achieve any desired non-zero probability of error provided that we choose  $K$  and  $N$  large enough.

In plain English: (Coding theorem)

Arbitrary reliability is possible up to a certain rate, but we have to work with very large block sizes

And furthermore:

We can expect to do pretty well by simply choosing our codes at random