"BACK TO CODING"

RE-VISIT : BLOCK CODES

NEW: CONVOLUTIONAL CODES / VITERBI ALGORITHM

$\boxed{1}$ (n,k,d) - CODE : A SET OF 2^k CODEWORDS OF LENGTH n

$$\underbrace{\boxed{\text{MESSAGE}}}_{k} \xrightarrow[\text{CODER}]{} \underbrace{\boxed{\text{CODE WORD}}}_{n \geq k}$$ (+THE CODER FUNCTION)

$d =$ min HD BETWEEN CODEWORDS

* AN (n,k,d)-CODE :
  - CORRECTS $(d-1)\%2$ BIT ERRORS
  - DETECTS UP TO $d-1$ BIT ERRORS

Ex.1 How many errors will A $(15, 11, 3)$ CODE DETECT?

ANSWER : 2

Ex.2 ——— " ——— $(20, 8, 4)$ CODE CORRECT?

ANSWER : 1

$\boxed{2}$ $(?, k, 3)$: HOW MANY EXTRA BITS NEEDED?

TH. ~~m~~ EXTRA BITS ENOUGH ONLY WHEN $k \leq 2^m - m - 1$

! WITH GOOD CODING, THIS IS ACHIEVABLE :

"HAMMING CODE" : $(2^m - 1, 2^m - m - 1, 3)$

(PROOF OF TH.: $(k+m+1)2^k \leq 2^{k+m}$)

Ex.3 FOR A 100-BIT MESSAGE, HOW MANY EXTRA PARITY BITS NEEDED TO GET SINGLE ERROR CORRECTION?

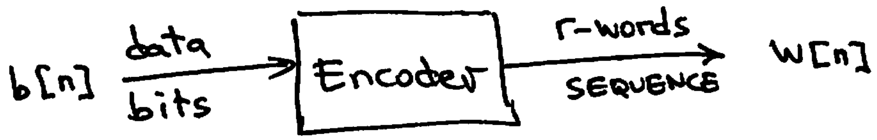
$$2^6 - 6 - 1 = 57 < 100$$
$$2^7 - 7 - 1 = 120 \geq 100 \quad \checkmark \Rightarrow \text{ANSWER} : 7$$

3 LINEAR CODE : WHEN CODEWORD + CODEWORD = ANOTHER CODEWORD

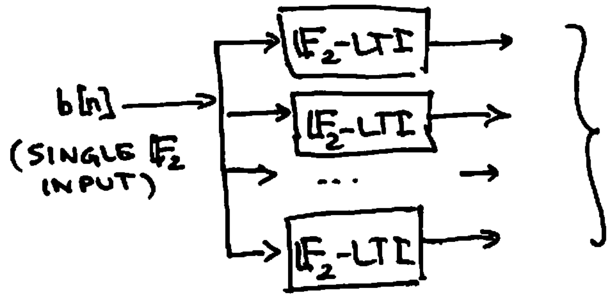
Ex. 4 is
$$\left.\begin{matrix}0011\\0110\\0101\\1000\end{matrix}\right\}\ \text{LINEAR?}\qquad \left.\begin{matrix}000\\010\\001\\100\end{matrix}\right\}\ ?$$

Ex. 5 IS IT POSSIBLE FOR A $\mathbb{F}_2$-LINEAR CODE TO HAVE EXACTLY 17 WORDS ?

CONVOLUTIONAL CODES :
(AS IF) USING ONE BIIIIIG CODEWORD FOR EACH DATA SEQUENCE

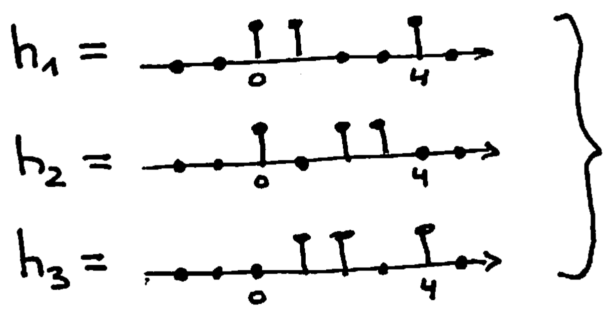


(SPECIFICALLY) CONVOLUTIONAL CODES

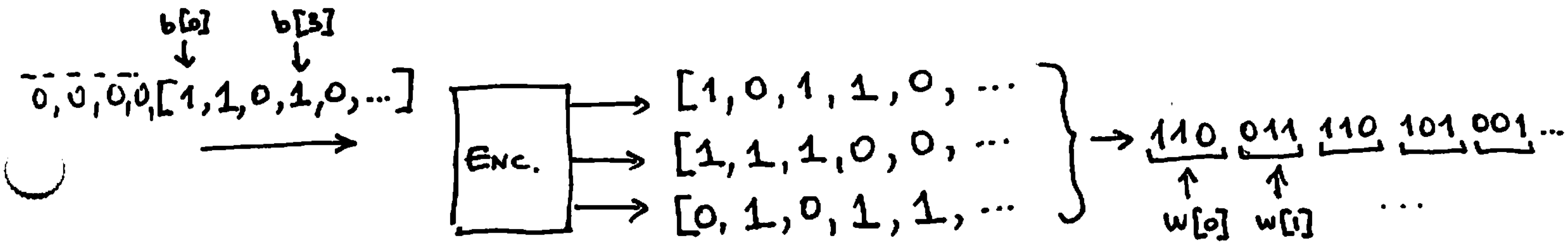


MULTIPLE $\mathbb{F}_2$ OUTPUTS

Ex. 6 $r=3$



$k=4$



TO **DECODE**: NEED "BEST APPROXIMATION" **DECONVOLUTION**:

$$x \longrightarrow \underset{h}{\boxed{\text{LTI}}} \longrightarrow y$$

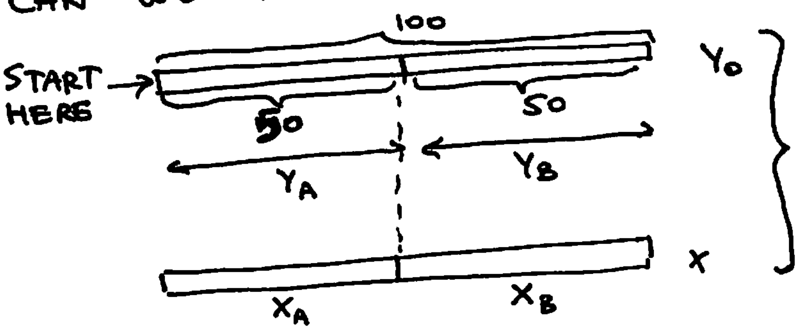
GIVEN h AND Y_0,
FIND X SUCH THAT
$h * X$ IS THE BEST APPROXIMATION
OF Y_0

WE COULD HAVE DONE THIS FOR THE ORIGINAL CONVOLUTION (WITH $\mathbb{R}$ INSTEAD OF $\mathbb{F}_2$), BUT THE QUANTIZED CASE IS EASIER

MAIN PROBLEM: A DIRECT SEARCH THROUGH ALL POSSIBLE SEQUENCES "x" IS TOO LONG

Ex. 7 IF Y IS 100 WORDS LONG, HOW MANY POSSIBLE "x"?
ANSWER: $2^{100} \sim 10^{10}$

CAN WE DIVIDE THE SEARCH INTO SMALLER PIECES?



IF WE KNEW X_A SUCH THAT $h * X_A \big|_{0-49}$ IS THE BEST APPROXIMATION OF Y_A, ~~WOULD~~ IT HELP?

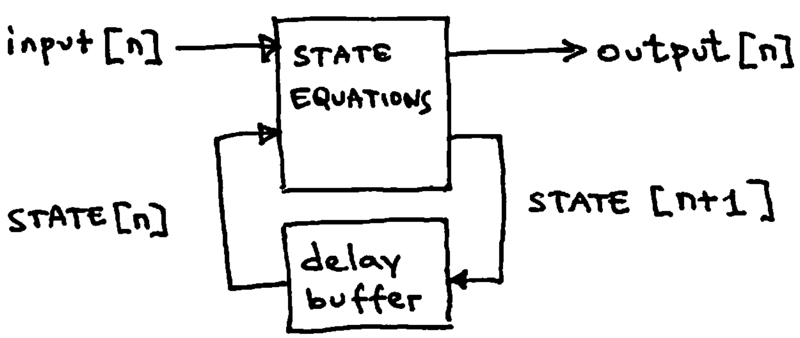
NOT NECESSARILY, BECAUSE THE LTI TRANSFORMATION HAS "MEMORY", HENCE WHAT WAS A GOOD SELECTION IN THE 0-49 RANGE MAY RUIN THE CHANCES OF THE FUTURE

SOLUTION:
FIGURE OUT WHAT IS EXACTLY THE "CURRENT MEMORY", AND OPTIMIZE X_A WHILE **FIXING** THE "MEMORY"

Ex. 8 WHAT IS THE MEMORY IN Ex. 6?

ANSWER: THE 4 PREVIOUS BITS:

$$\text{MEMORY}[n] = \underline{\text{STATE}}[n] = \begin{bmatrix} b[n-1] \\ b[n-2] \\ b[n-3] \\ b[n-4] \end{bmatrix}$$



EVERY CONVOLUTION ENCODER WORKS THIS WAY (AND MANY ~~OTHER~~ OTHER ENCODERS AS WELL

"STEP K" TASK:

FOR EVERY POSSIBLE VALUE X OF STATE[K], FIND THE MINIMAL DISTANCE BETWEEN $Y_o|_{0-K}$ AND $Y|_{0-K}$, WHERE $Y|_{0-K}$ IS THE ENCODER'S RESPONSE TO AN INPUT WHICH BRINGS THE ENCODER'S STATE TO X IN K STEPS.

DYNAMIC PROGRAMMING:

ONCE THE "STEP K" TASK IS SOLVED, A SOLUTION OF THE "STEP K+1" TASK IS EASY

SOLVE "STEP 0" TASK FIRST

THEN: SOLVE "STEP 1" TASK

THEN: SOLVE "STEP 2" TASK

...

ONCE ALL STEPS ARE DONE, RECOVERING THE OPTIMAL INPUT IS EASY

Ex. 9

$x[n] \longrightarrow$ ENC. $\longrightarrow Y_1[n] = x[n]$

$\longrightarrow Y_2[n] = x[n] + x[n-1]$ (in $\mathbb{F}_2$)

$$(r = 2,\ K = 1,\ h_1[n] = \delta[n],\ h_2[n] = \delta[n] + \delta[n-1])$$

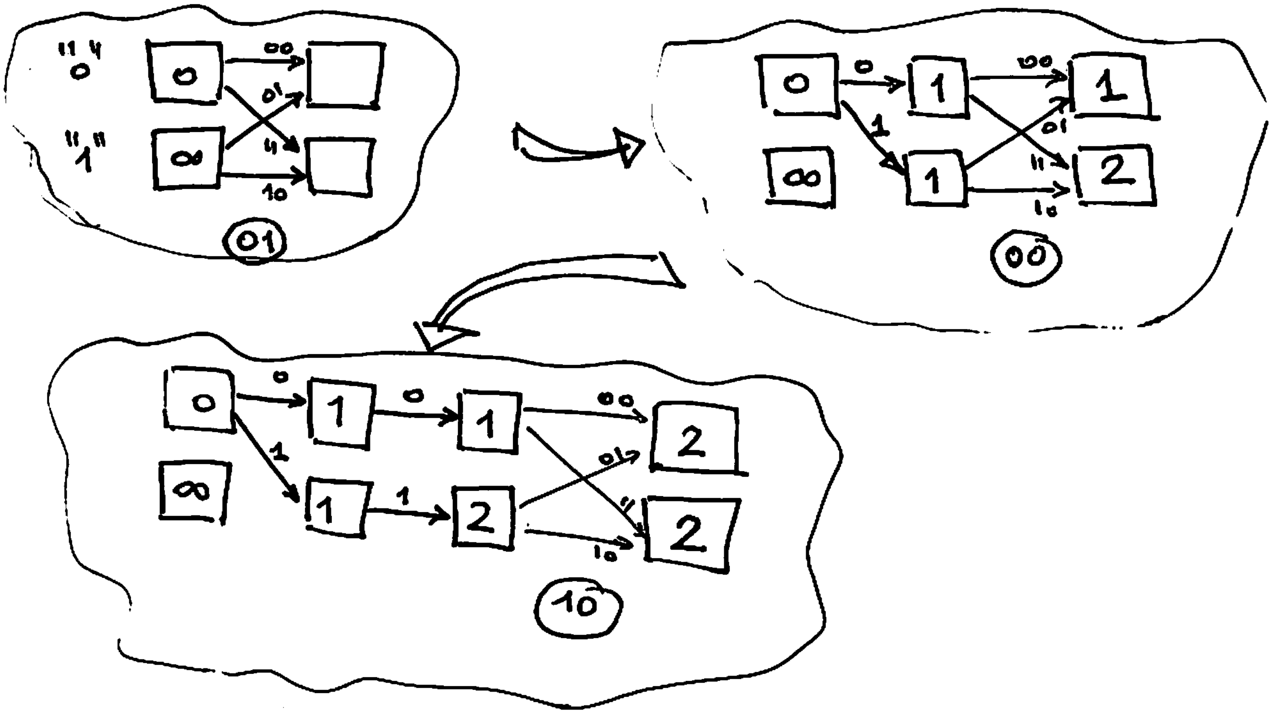
$$\text{"STATE"}[n] = x[n-1]$$

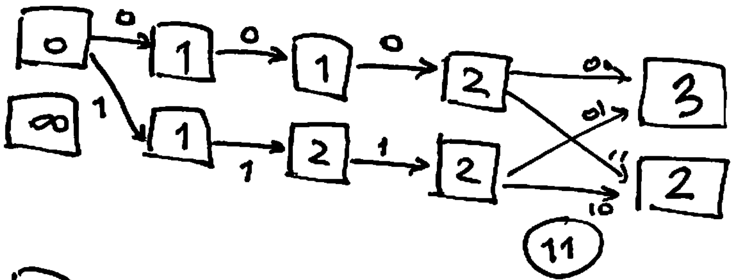
$$x[0], x[1], x[2], \ldots \longrightarrow \underbrace{x[0], x[0]}_{Y[0]},\ \underbrace{x[1], x[0]+x[1]}_{Y[1]},\ \underbrace{x[2], x[1]+x[2]}_{Y[2]}, \ldots$$

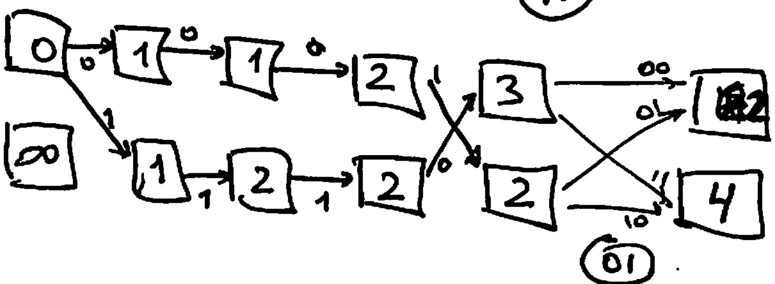
6.02 / 07.10.2010

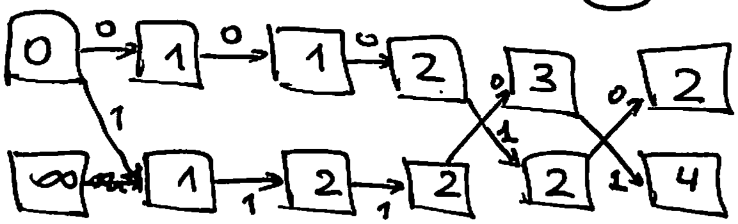
Ex.9 (CONT.)

ASSUME 01 00 10 11 01 : RECEIVED









DECODED SO FAR: 0, 0, 0, 1, 0

→ 00 00 00 11 01