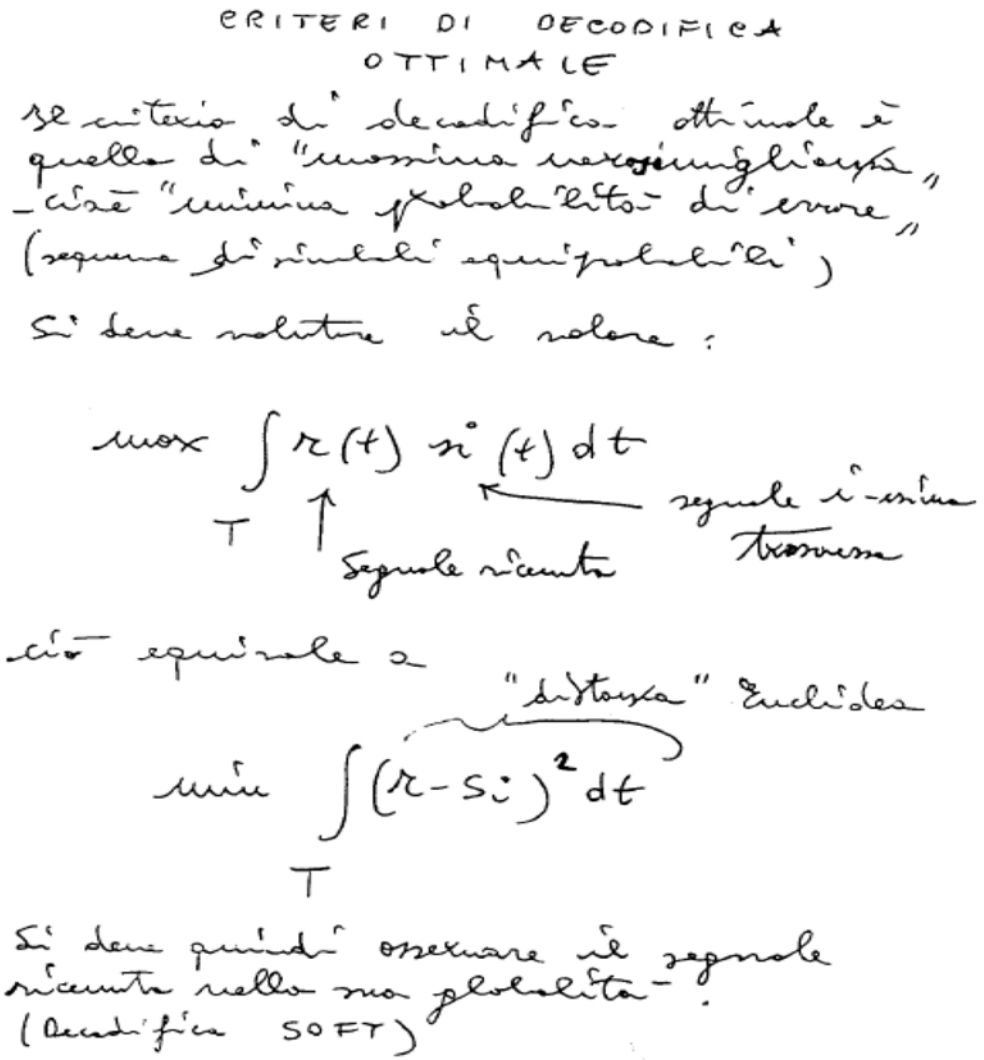
Linear Block Codes (3)
Performance Evaluation

Pierangelo Migliorati

DII - University of Brescia

Optimal Decoding: Soft Detection (or Soft Decision)



The optimal receiver (soft decision)
works in the signal space, using the "Eucledian" distance and estimating the likelihood function
(i.e., minimize the distance or maximize the correlation …)

Hard Detection …

DECODIFICA "HARD"

Per semplificare le operazioni si può operare una decisione preliminare circa il valore del singolo bit che costituisce la parola di codice.

Successivamente si decide sulla parola trasmessa partendo dai singoli bit già decisi.

Si è in tal modo suddiviso l'operazione di decodifica in due distinte operazioni: la DEMODULAZIONE dei singoli bit e la DECODIFICA BINARIA vera e propria

Il sistema composto dal CANALE più il DEMODULATORE DI BIT equivale ad un canale BINARIO.

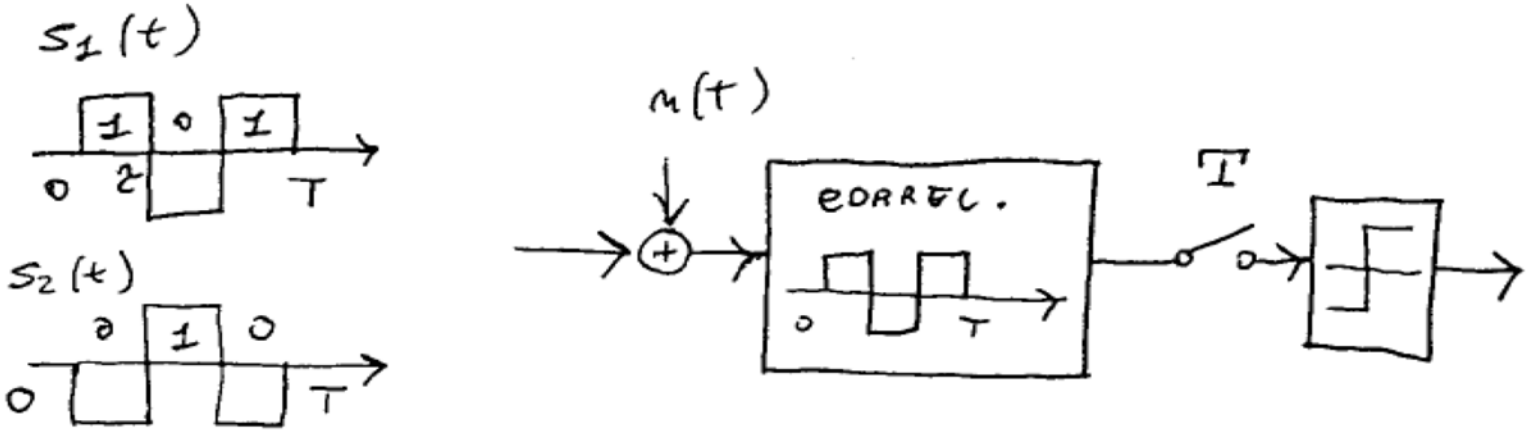
In questo caso si ha un degrado di prestazioni dovuto alla decodifica "hard".

In order to simplify the receiver architecture, it is possible to take a preliminary decision, bit per bit, and then to evaluate the minimum distance (hamming distance in this case) in the vector space of 1s and 0s …

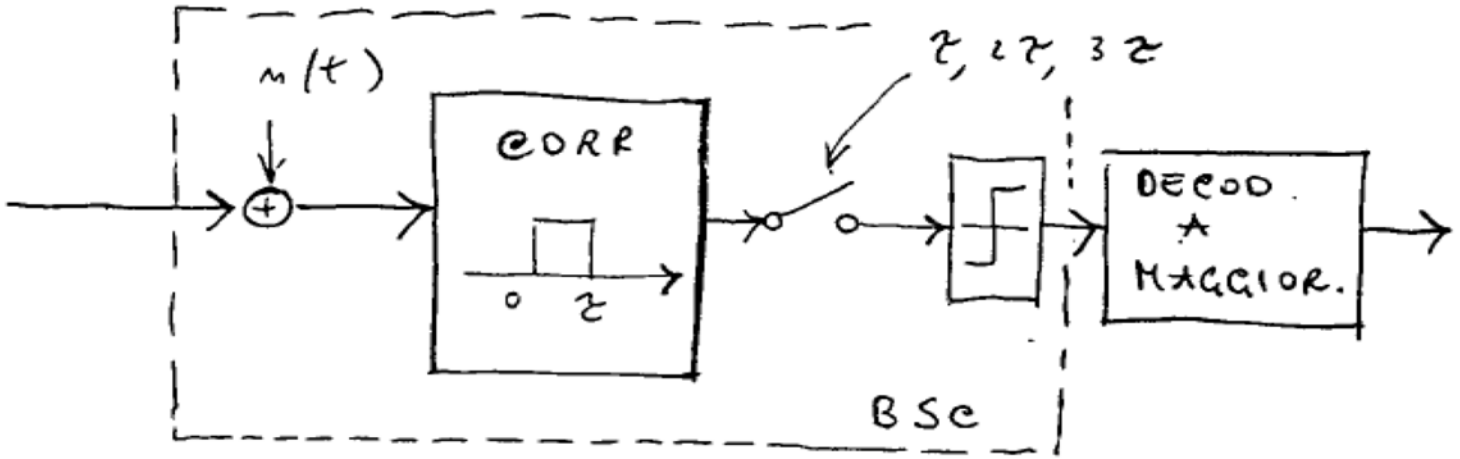
This is indicated as an "hard decision" scheme. It is clearly sub-optimal … but much more simple than the (optimal) soft decision scheme.

Clearly, we will have a performance degradation (usually the loss is around 2 dB, SNR).

"SOFT"


$s_1(t)$
1
0
1
0
T
$s_2(t)$
0
1
0
T
m(t)
CORREL.
T

"HARD"


m(t)
CORR
T, 2T, 3T
DECOD.
*
MAGGIOR.
BSC

Soft / Hard detection

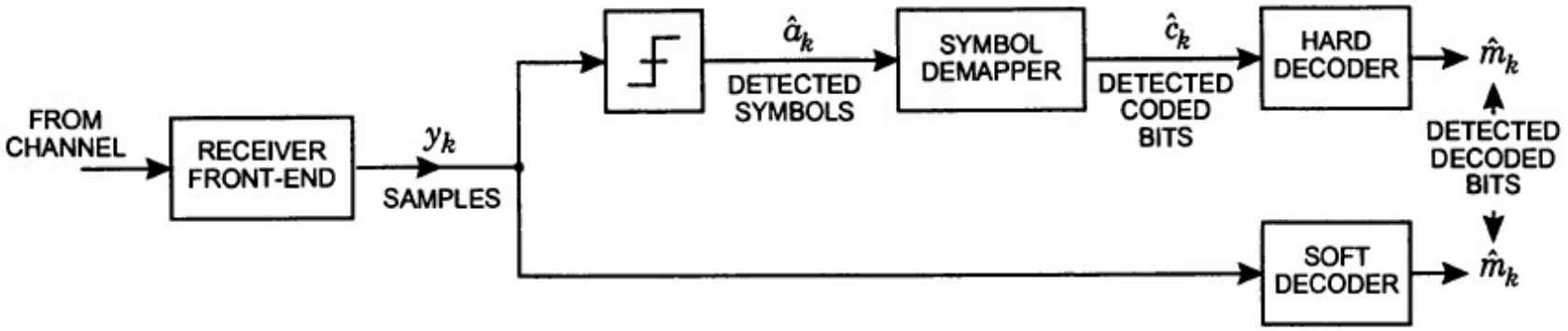


Fig. 12-2. In hard decoding (upper branch), decisions are made about the incoming symbols by a slicer, the symbols are decoded into bits by a demapper, and the channel decoder maps these coded bits into uncoded bits. A soft decoder (lower branch) operates directly on continuous-valued samples of the incoming signal rather than on the detected bits.

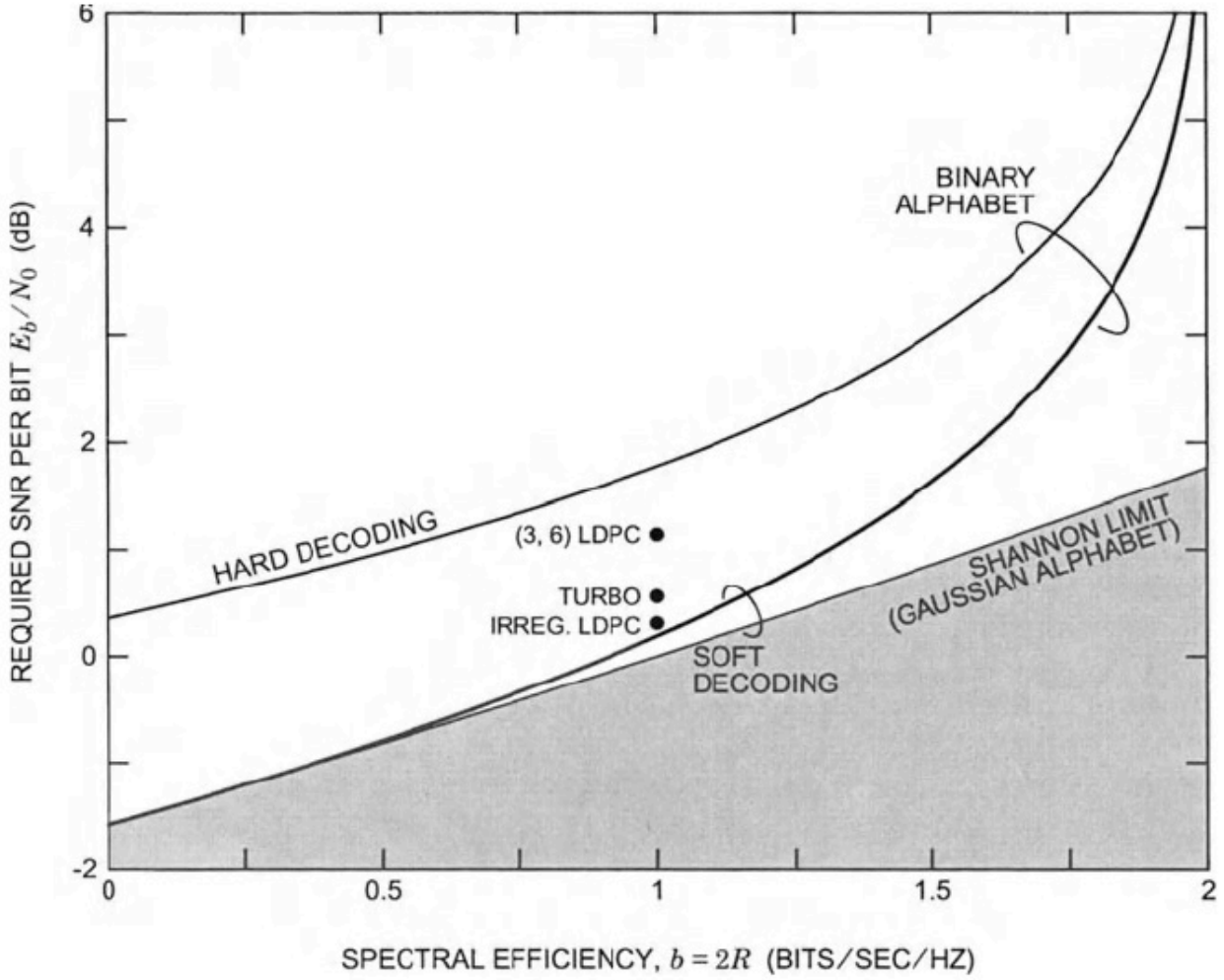


Fig. 12-4. Capacity of the AWGN with various constraints on the input and output. The lower curve is the Shannon limit $(2^b - 1)/b$, which is achieved when the channel inputs have a Gaussian distribution. The next curve above is the Shannon limit when the input alphabet is constrained to be binary. The penalty due to binary inputs disappears at low spectral efficiency. The uppermost curve constrains the input alphabet to be binary and also constrains the receiver to perform hard decoding. The extra penalty due to hard decoding is seen to range from 2 dB down to 1 dB as the binary code rate ranges from 0 to 0.98. The three circles mark the performance of specific codes to be described later in the chapter.

Soft detection

$$\Pr[\text{block error}] \approx KQ\left(\frac{d_{E,\min}}{2\sigma_c}\right)$$

$$= KQ\left(\sqrt{2Rd_{\min}\frac{E_b}{N_0}}\right), \qquad (12.16)$$

where $d_{E,\min} = 2\sqrt{Ed_{\min}} = 2\sqrt{RE_b d_{\min}}$, and where $d_{\min}$ is the minimum *Hamming* distance of the code. The coefficient K is the average number of codewords with Hamming distance $d_{\min}$ from a given codeword.

Soft detection

In contrast to the above expression for a coded system, the bit-error probability for an uncoded binary antipodal system has a simple closed-form solution, namely:

$$\Pr[\text{bit error, uncoded}] = Q\left(\sqrt{\frac{2E_b}{N_0}}\right). \tag{12.17}$$

In fact, the union-bound estimate becomes exact in this case, since (12.16) reduces to (12.17) for the uncoded case where $R = d_{\min} = K = 1$. At high SNR, the probability of error for a block of k uncoded bits is then:

$$\Pr[\text{block error, uncoded}] \approx kQ\left(\sqrt{\frac{2E_b}{N_0}}\right). \tag{12.18}$$

We can now directly compare the block-error performance of the coded and uncoded systems. If we ignore the constant multipliers of $Q(\cdot)$, which is reasonable at high E_b/N_0, we can equate the arguments of the $Q(\cdot)$ in the coded and uncoded cases, yielding:

$$Rd_{\min}(E_b/N_0)^{\text{required}}_{\text{coded}} = (E_b/N_0)^{\text{required}}_{\text{uncoded}}. \tag{12.19}$$

This implies that the *asymptotic coding gain* is simply the product of the code rate and minimum Hamming distance:

$$\text{asymptotic coding gain} = Rd_{\min}. \tag{12.20}$$

This gain is a*symptotic* because it only applies in the limit of large E_b/N_0, or equivalently small probability of error. The true coding gain is usually somewhat less.

Soft detection

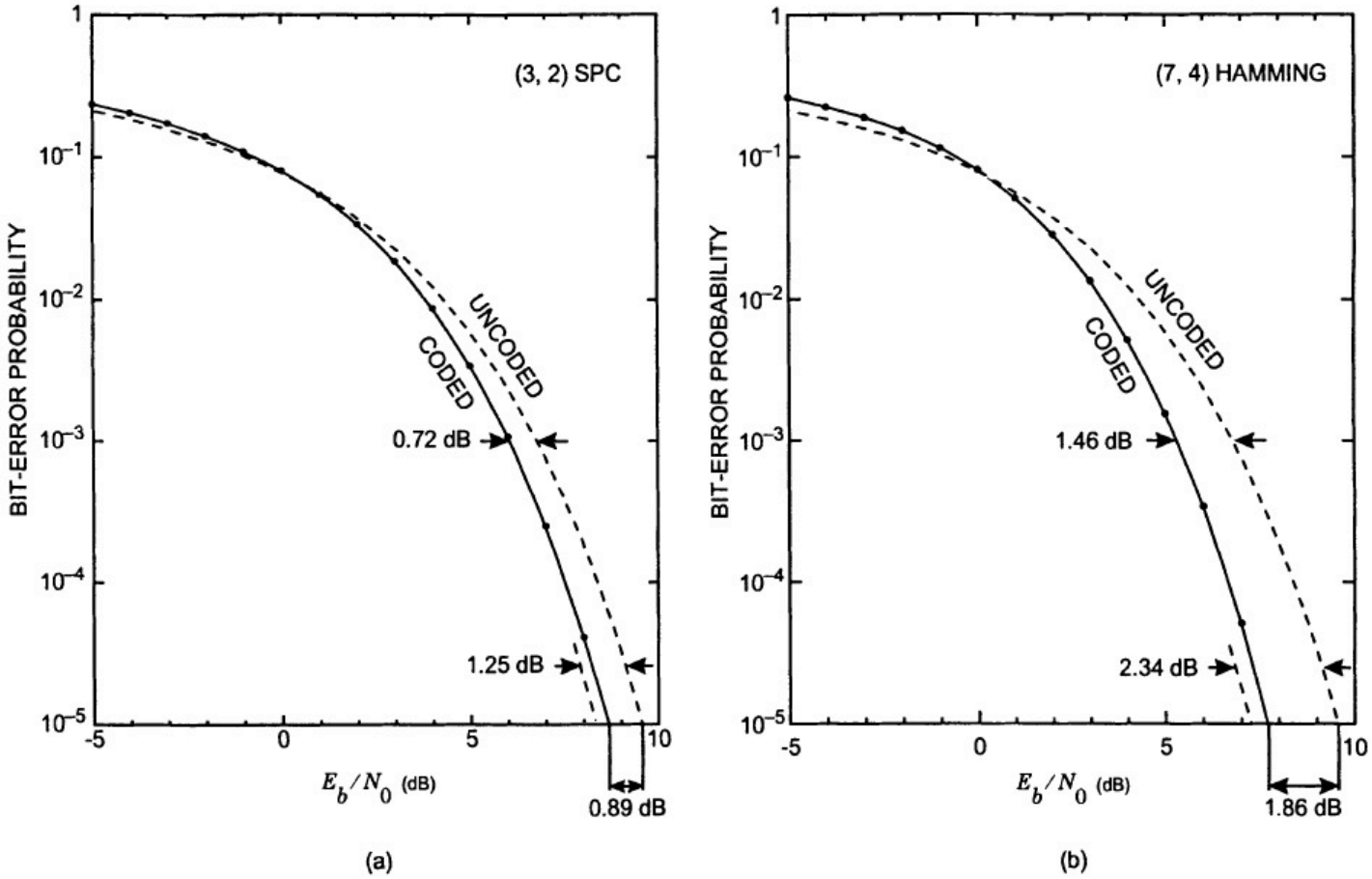


Fig. 12-6. Illustration of coding gain with soft decoding for (a) the (3, 2) single-parity-check code, and (b) the (7, 4) Hamming code, assuming AWGN with binary signaling. Also shown is the bit-error probability for an uncoded system. In (a), we see that the coding gain for the (3, 2) single-parity-check code is 0.89 dB at $P_b = 10^{-5}$, which is close to the 1.25 dB predicted by the asymptotic coding gain. The coding gain for the (7, 4) Hamming code is 1.86 dB at $P_b = 10^{-5}$, about 0.5 dB short of the 2.34 dB asymptotic coding gain.

Hard detection

12.2.2. Performance of Hard Decoders

A hard decoder operates on the output of the slicer. If the discrete-time channel up to the slicer has a noise generation model with independent noise components, then after the slicer the equivalent binary channel will usually be a BSC (Fig. 7-2). With a code of rate R and binary antipodal signaling in AWGN, the crossover probability is:

$$p = Q\left(\sqrt{2R\frac{E_b}{N_0}}\right). \tag{12.23}$$

Clearly, this crossover probability *increases* as the code rate decreases. In other words, the introduction of a code makes the "raw" bit-error rate worse. A useful code will more than compensate for this increase, resulting in an overall bit-error rate after decoding that is smaller.

Hard detection

The question that arises now is how many bit errors can be corrected by an ML detector for a given code. It is clear that if $\boldsymbol{r}$ is closer to $\boldsymbol{c}$ than to any other codeword then any errors in $\boldsymbol{r}$ will be corrected by the ML detector. Certainly if $\boldsymbol{r}$ has fewer than

$$t = \left\lfloor \frac{d_{H,\min} - 1}{2} \right\rfloor \tag{12.24}$$

errors then those errors can be corrected, where $\lfloor \cdot \rfloor$ denotes the "floor" function, or the greatest integer less than or equal to the argument. This value t appeared before (see (7.22)).

Example 12-9.

In the single-parity-check code of Example 12-2 and Example 12-3, each codeword has an even number of ones, so $d_{H,\min} = 2$. The code can correct

$$t = \left\lfloor \frac{d_{H,\min} - 1}{2} \right\rfloor = 0 \tag{12.25}$$

bit errors. Hence this code is not useful at all for hard error correction. It can detect any odd number of bit errors. Note that with soft decoding, this code is useful for reducing the error rate at a given signal level, but not with hard decoding.

$$\Pr[\text{block error}] \approx K \sum_{i=t+1}^{d_{\min}} \binom{d_{\min}}{i} p^i (1-p)^{d_{\min}-i}, \tag{12.26}$$

where $d_{\min}$ is the minimum Hamming distance of the code. For small p, the first term dominates, so that this can be approximated by:

$$\Pr[\text{block error}] \approx K \binom{d}{t+1} p^{t+1}. \tag{12.27}$$

Using $p = Q(\sqrt{2RE_b/N_0})$ and $Q(x) \approx e^{-x^2/2}$ yields:

$$\Pr[\text{block error}] \approx K \binom{d}{t+1} e^{-R(t+1)E_b/N_0}. \tag{12.28}$$

Using the same approximation $Q(x) \approx e^{-x^2/2}$ for the *uncoded* case yields:

$$\Pr[\text{block error, uncoded}] \approx k\,Q(\sqrt{2E_b/N_0}) \approx k e^{-E_b/N_0}. \tag{12.29}$$

Comparing the uncoded and coded cases and ignoring the multiplying coefficients leads to an asymptotic coding gain for hard decoding of:

$$\text{asymptotic coding gain}_{(\text{hard})} = R(t+1). \tag{12.30}$$

This is almost half the asymptotic coding gain of $Rd_{\min}$ that arises from soft decoding. This approximate analysis suggests that hard ML decoding should perform almost 3 dB worse than soft ML decoding. In reality the difference is usually closer to 1.5 dB or 2 dB.

Example 12-11.

Like all Hamming codes, the (15, 11) Hamming code has $d_{\min} = 3$ and $t = 1$, and hence can correct all single bit errors. The asymptotic coding gains for hard and soft decoding are thus:

$$R(t+1) = \frac{22}{15} = 1.66 \text{ dB (hard)}, \qquad Rd_{\min} = \frac{33}{15} = 3.42 \text{ dB (soft)}. \tag{12.31}$$

The anticipated difference is thus 1.76 dB. The bit-error probability for this code with hard decoding over the BSC can be found analytically through a straightforward but tedious counting exercise, yielding:

$$P_{b,\text{hard}} = 21p^2 + 119p^3 + 392p^4 + 1036p^5 + 2093p^6 + 3067p^7 + 3368p^8 + \\ + 2912p^9 + 1967p^{10} + 973p^{11} + 336p^{12} + 84p^{13} + 15p^{14} + p^{15}, \tag{12.32}$$

where $p = Q(\sqrt{2\frac{11}{15}E_b/N_0})$. The bit-error probability with soft decoding can be found via simulations. The results are shown in Fig. 12-7, which reveals that the actual coding gain for hard and decoding at $P_b = 10^{-5}$ are:

$$\text{coding gain} = 1.2 \text{ dB (hard)}, \qquad \text{coding gain} = 2.6 \text{ dB (soft)}. \tag{12.33}$$

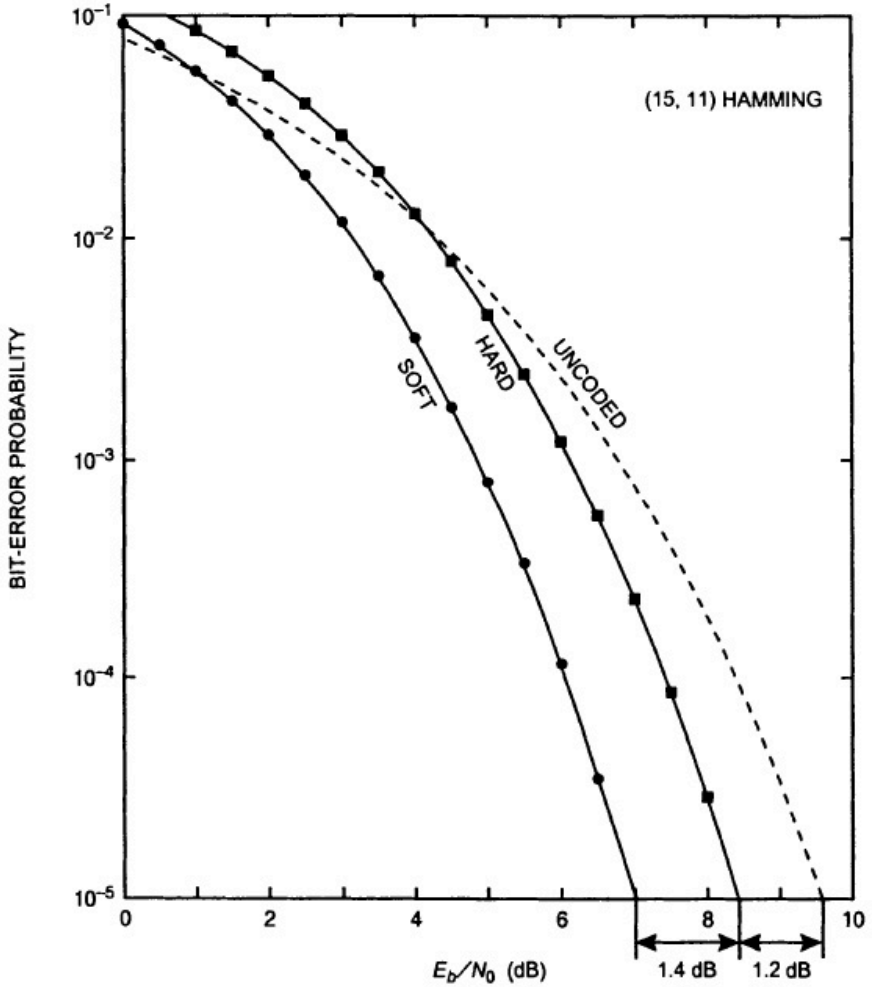


Fig. 12-7. A comparison of bit-error probability for the (15, 11) Hamming code with soft and hard decoding on the AWGN channel with binary signaling. At $P_b = 10^{-5}$, the coding gain is 1.2 dB with hard decoding, and 2.6 dB with soft decoding.

Error Probability

Probability of having more than t errors

$$P(E) = \sum_{h=t+1}^{N} \binom{N}{h} \varepsilon^h (1-\varepsilon)^{N-h}$$

For Gaussian channels with hard receiver

$$\varepsilon = Q\left(\sqrt{\frac{2E_b}{N_0}\frac{K}{N}}\right)$$

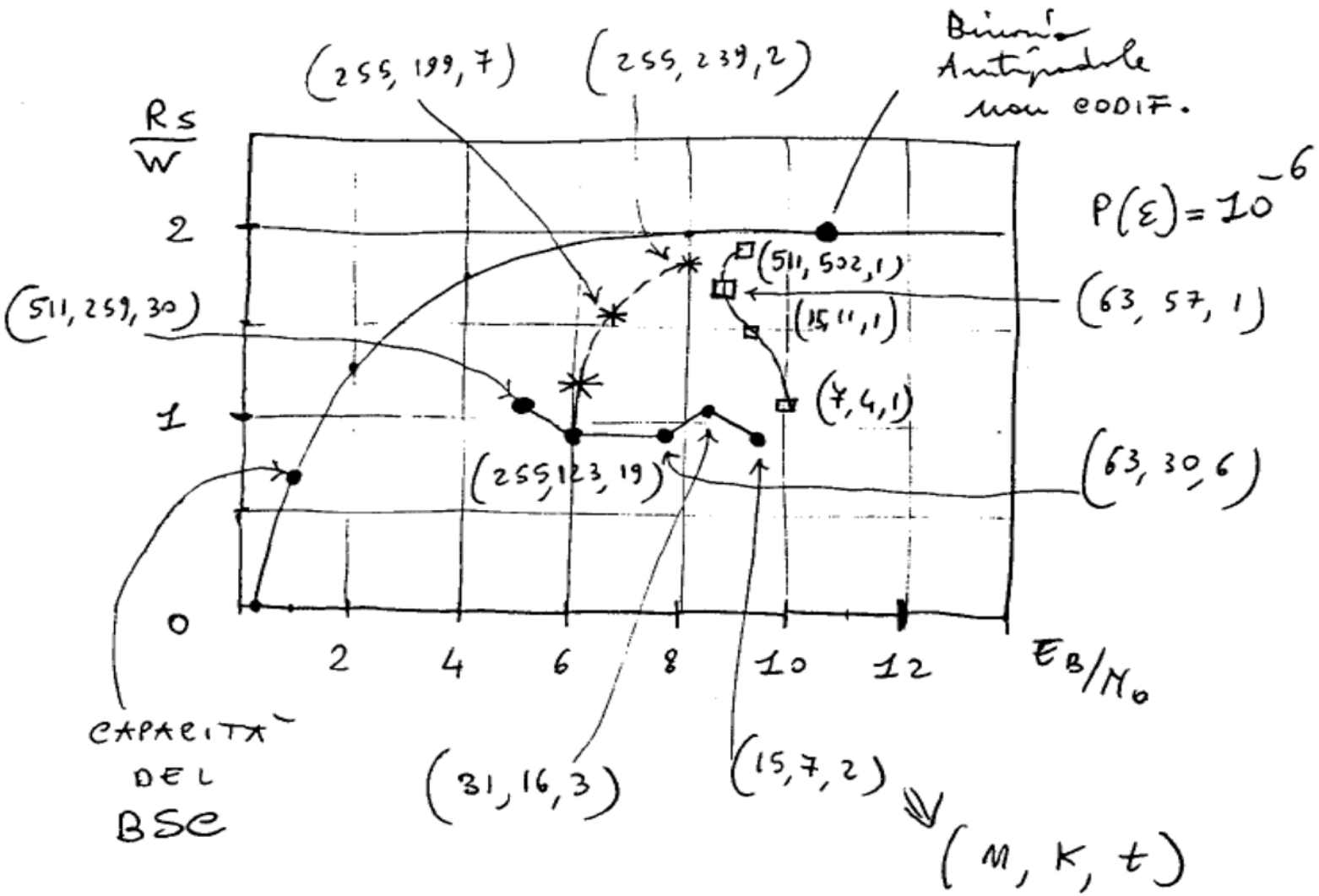
Asymptotic estimate:

$$P(E) \approx Q\left(\sqrt{\frac{2E_b}{N_0}\frac{K}{N}(t+1)}\right)$$

To be compared with the maximum likelihood soft decoding

$$Q\left(\sqrt{\frac{2E_b}{N_0}\frac{K}{N}(d^*)}\right)$$

PRESTAZIONI (Decisioni HARD)



□ CODICI DI HAMMING

✳ BCH con $N = 255$

● CODICI BCH

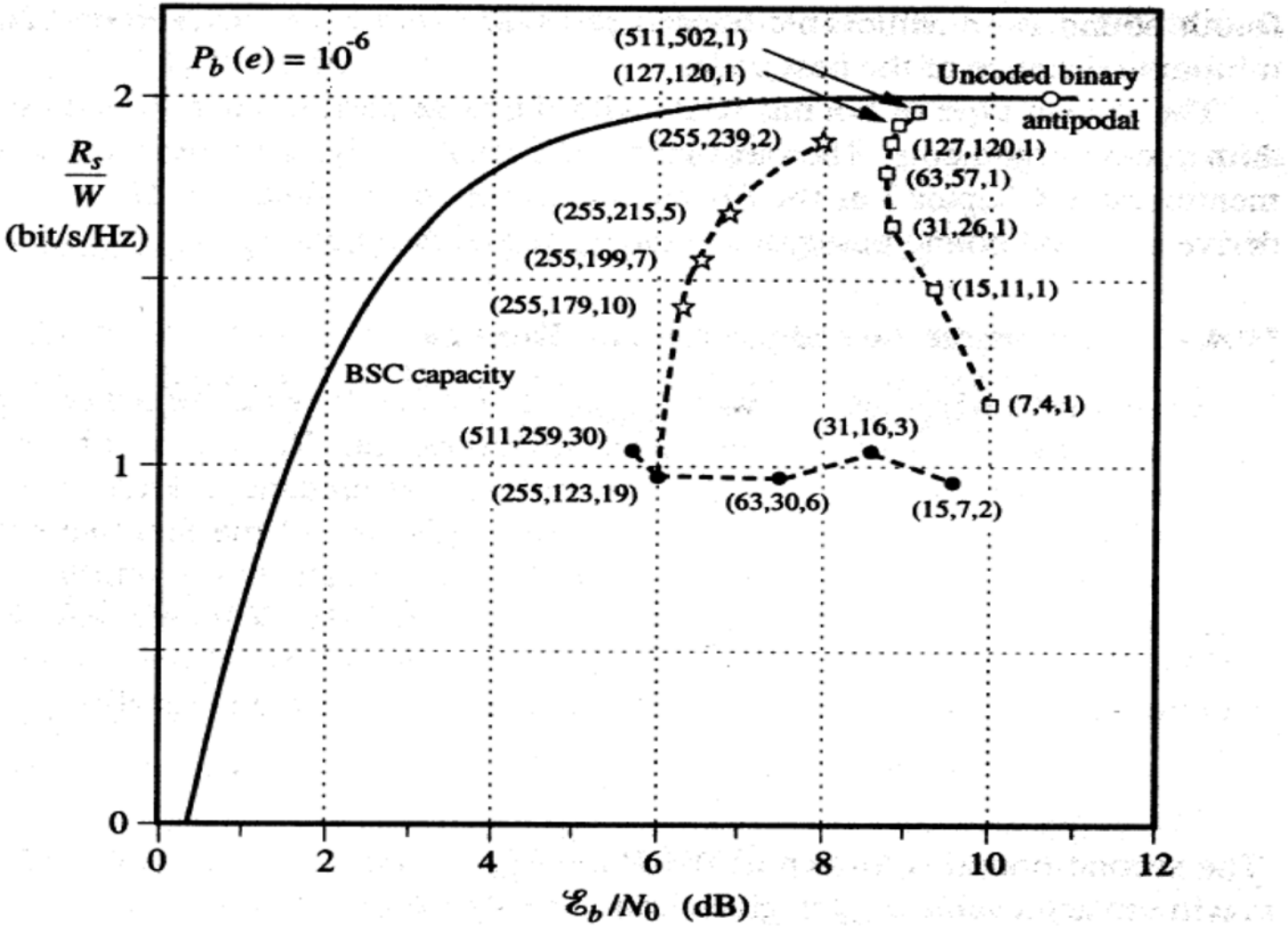


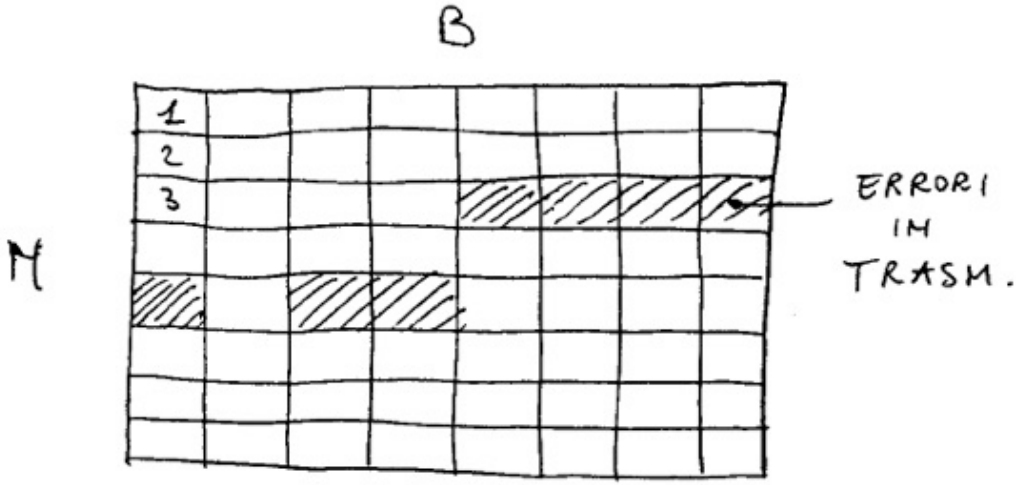
Figure 10.25: *Performance chart of different BCH and Hamming codes. Each code is identified with three numbers: the block length n, the number of information bits k, and the number of corrected errors t.*

Interleaving

Burst Errors

- Burst errors are very common, in particular in wireless environments where a fade will affect a group of bits in transit. The length of the burst is dependent on the duration of the fade.
- One way to counter burst errors, is to break up a transmission into shorter words and create a block (one word per row), then have a parity check per word.
- The words are then sent column by column. When a burst error occurs, it will affect 1 bit in several words as the transmission is read back into the block format and each word is checked individually.

Burst Error Correction



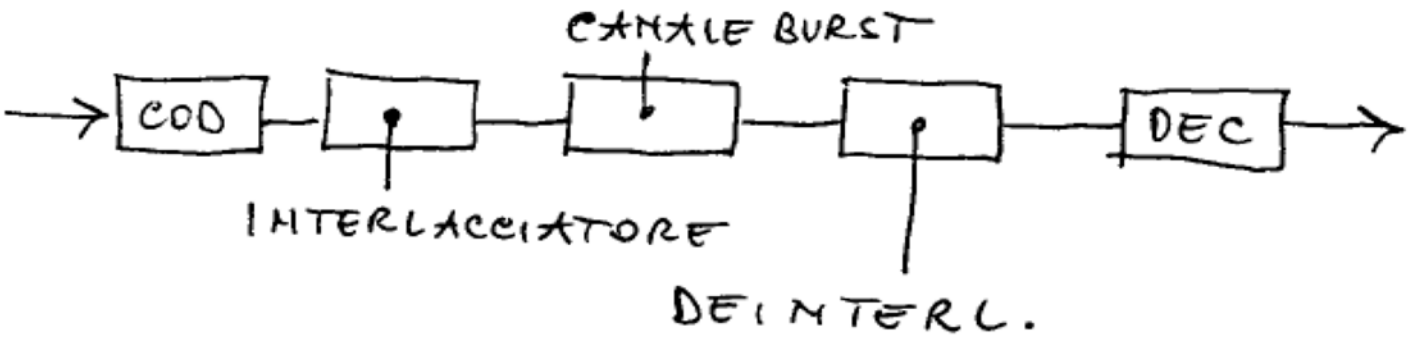
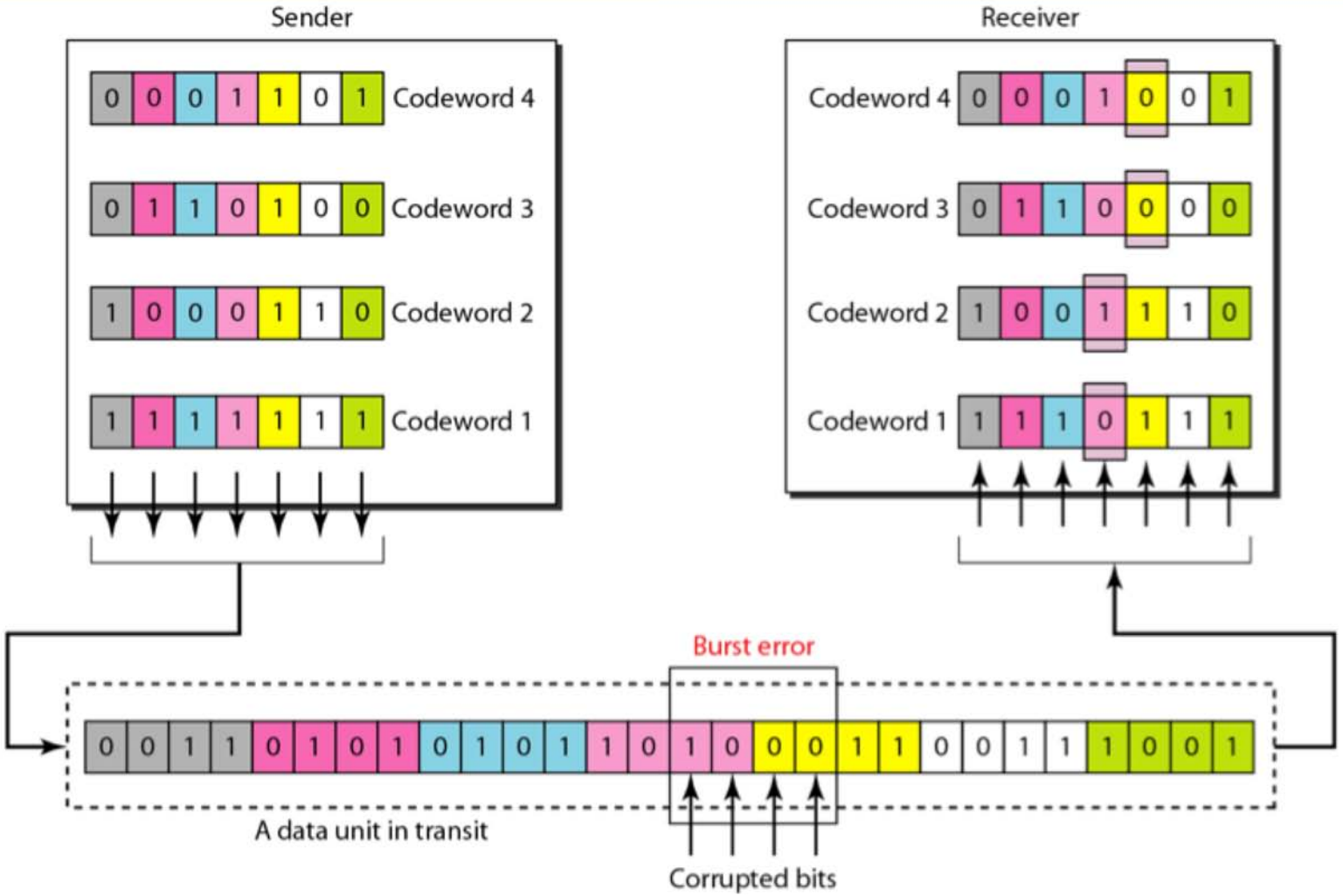


Figure 10.13 *Burst error correction using Hamming code*



Interleaving

- We have *assumed* so far that bit errors are independent from one bit to the next
- In mobile radio, ***fading*** makes ***bursts of error*** likely.
- Interleaving is used to try to make these errors independent again