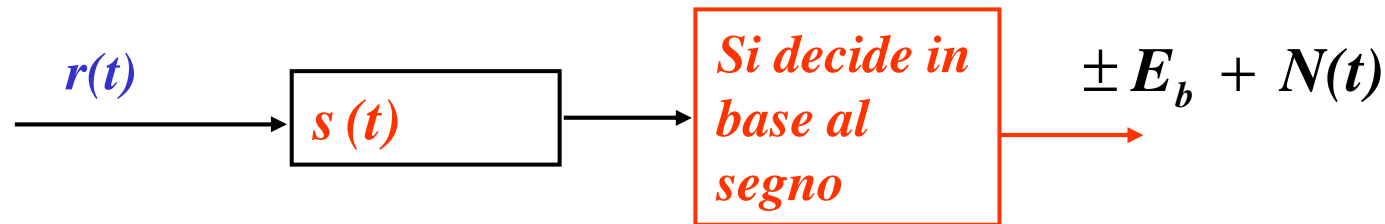


# *Codifica a controllo di errore*

*Mauro Giaconi*

## *Prestazioni dei sistemi di trasmissione binari*

*Con riferimento ai ricevitori ML (Maximum Likelihood), considerando segnali binari reali e antipodali,  $s(t)$  e  $-s(t)$ , la struttura del decisore è la seguente*



*Riguardo ai segnali nell'istante di decisione (massimo della correlazione), si ha*

*ricevuto il segnale  $s(t)$*

$$y = \int_{-\infty}^{\infty} S(f) \cdot S^*(f) df = \int_{-\infty}^{\infty} |S(f)|^2 df = E_b$$

*ricevuto il segnale  $-s(t)$*

$$y = \int_{-\infty}^{\infty} -S(f) \cdot S^*(f) df = - \int_{-\infty}^{\infty} |S(f)|^2 df = -E_b$$

*$E_b$  = energia per bit*

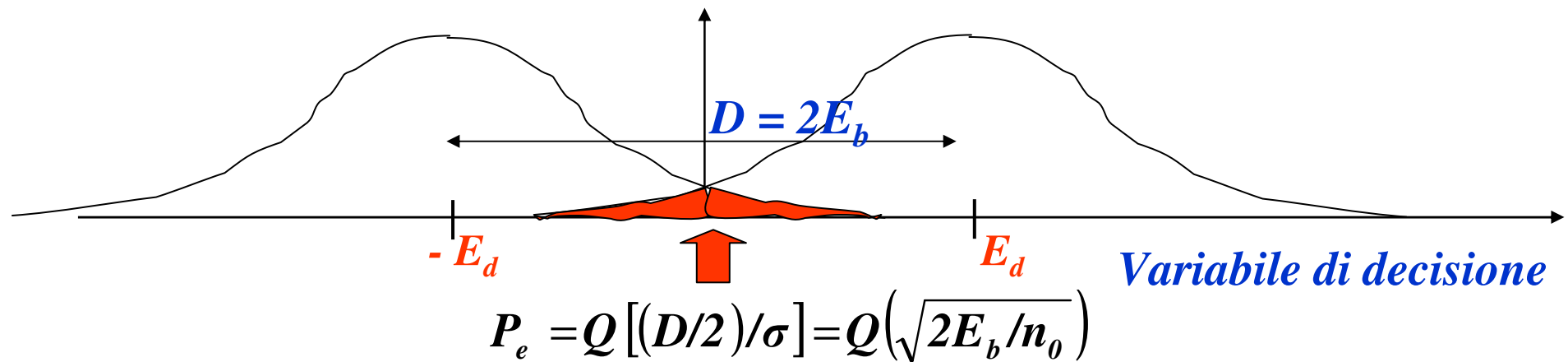
## *Prestazioni dei sistemi di trasmissione binari*

*$2E_b$  distanza tra i segnali; la varianza (valore efficace quadratico), del rumore, è*

$$\sigma^2 = \frac{n_0}{2} \int_{-\infty}^{\infty} |S(f)|^2 df = \frac{n_0 E_b}{2}$$

## Prestazioni dei sistemi di trasmissione binari

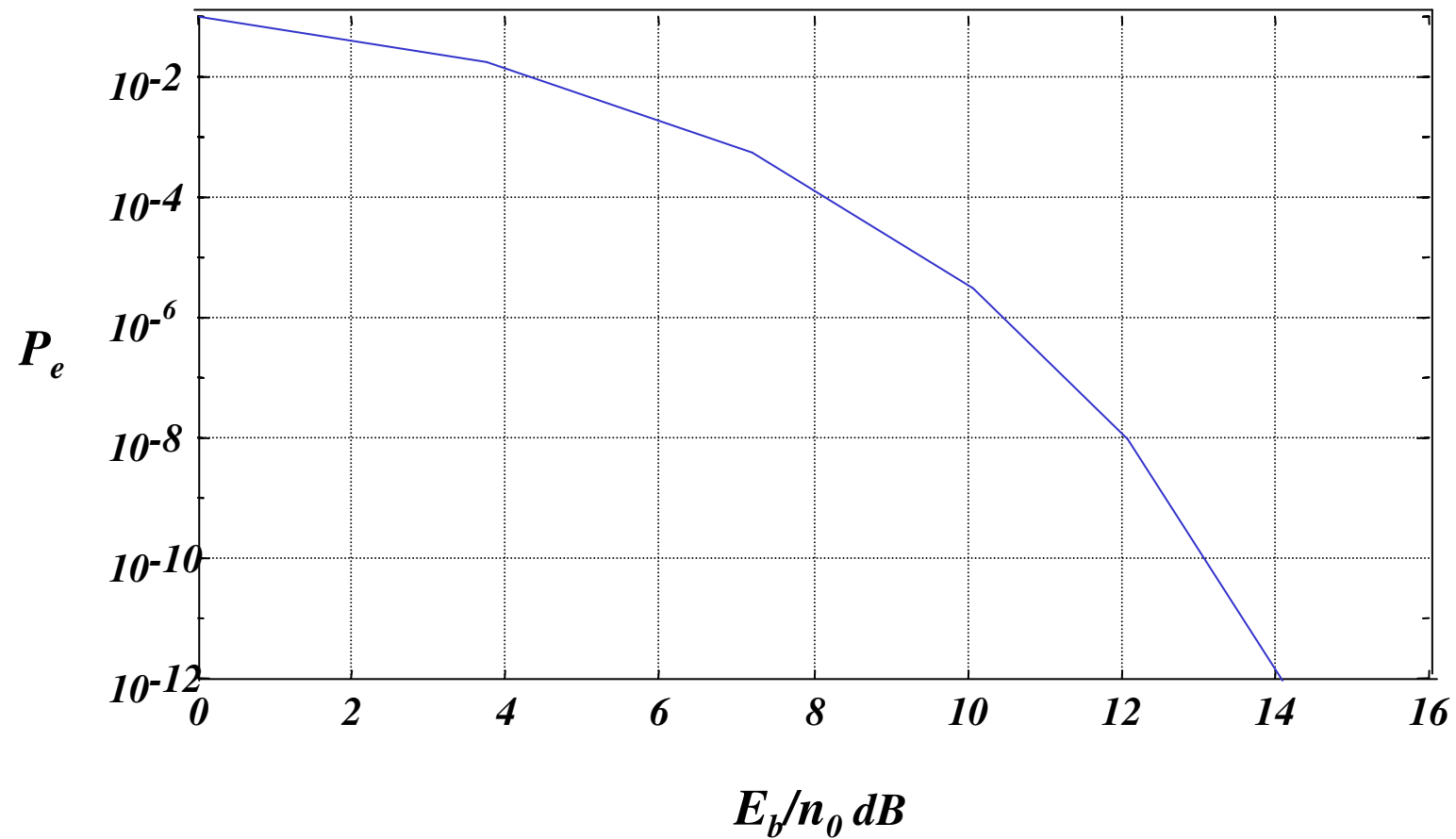
Geometricamente, si ottiene, quindi, per la *decisione simbolo per simbolo*



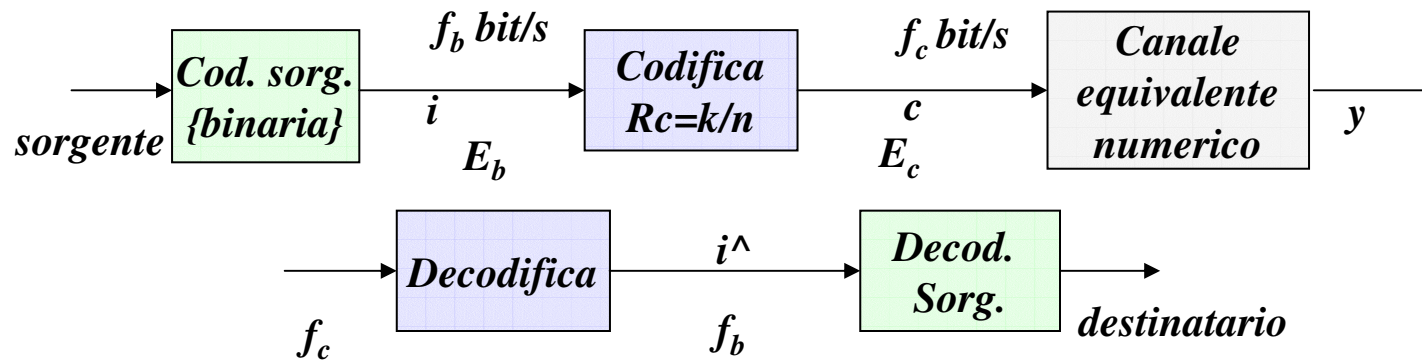
Segnali antipodali (polari BB, NRZ o RZ, 2 PSK),  $P_{bit} = Q\sqrt{2E_b/n_0}$ .

$$Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^{\infty} e^{-\frac{y^2}{2}} dy \approx \frac{e^{-\frac{x^2}{2}}}{2} \Rightarrow P_b \approx \frac{e^{-\frac{E_b}{n_0}}}{2}$$

## *Prestazioni dei sistemi di trasmissione binari*



## Codifica di canale



- La codifica di canale aggiunge **ridondanza e memoria** all'informazione così da rivelare e/o correggere errori. Ad esempio, ogni  $k$  bit informativi si generano blocchi di  $n$  bit codificati

- Il rate o efficienza di codifica è  $R_c = k/n$

- La codifica di canale classica richiede un aumento di banda (questo è evitato con il TCM, Trellis Coded Modulation)

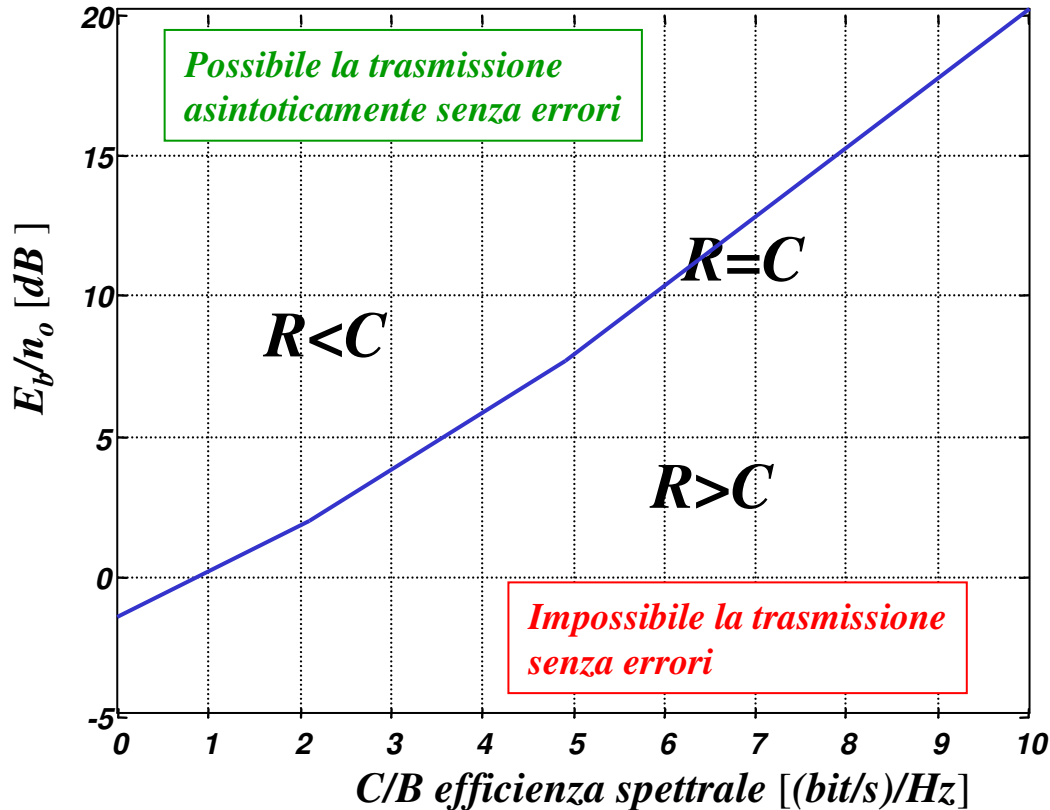
$$f_c = \frac{f_b}{R_c} = f_b \frac{n}{k}$$

- La codifica riduce l'energia per bit

$$E_b^{(c)} = E_b^{(u)} \cdot R_c = E_b^{(u)} \cdot \frac{k}{n}$$

- il canale ha un  $E_b/n_0$  più sfavorevole, la probabilità di errore all'ingresso del decodificatore è maggiore rispetto alla trasmissione non codificata

# Limiti teorici: capacità $C$ del canale ideale gaussiano limitato in banda $B$



$$C = B \log_2 \left( 1 + \frac{S}{N} \right) \text{ [bit/s]}$$

legge di Hartley Shannon

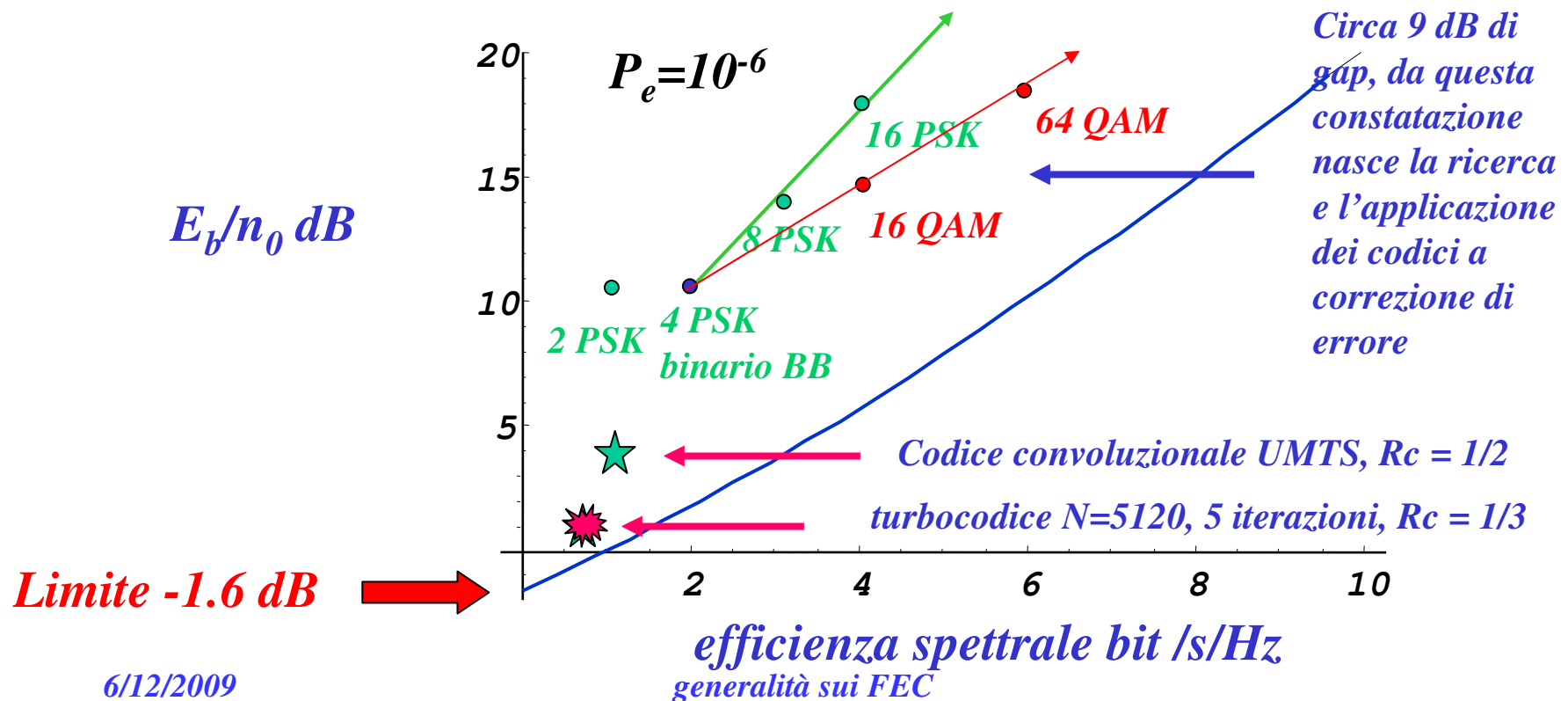
- ❖ Ipotesi: segnale utile a statistica Gaussiana con rumore Gaussiano;
- ❖ limiti sulla potenza del segnale utile,  $S$ , e su quella di rumore,  $N$ .

❖ Il teorema di Shannon stabilisce l'esistenza di una codifica di canale che permette la trasmissione asintoticamente senza errori su un canale di banda  $B$  ad una velocità  $R < C$

❖ ~~é' invece~~ impossibile la trasmissione senza errori per  $R > C$

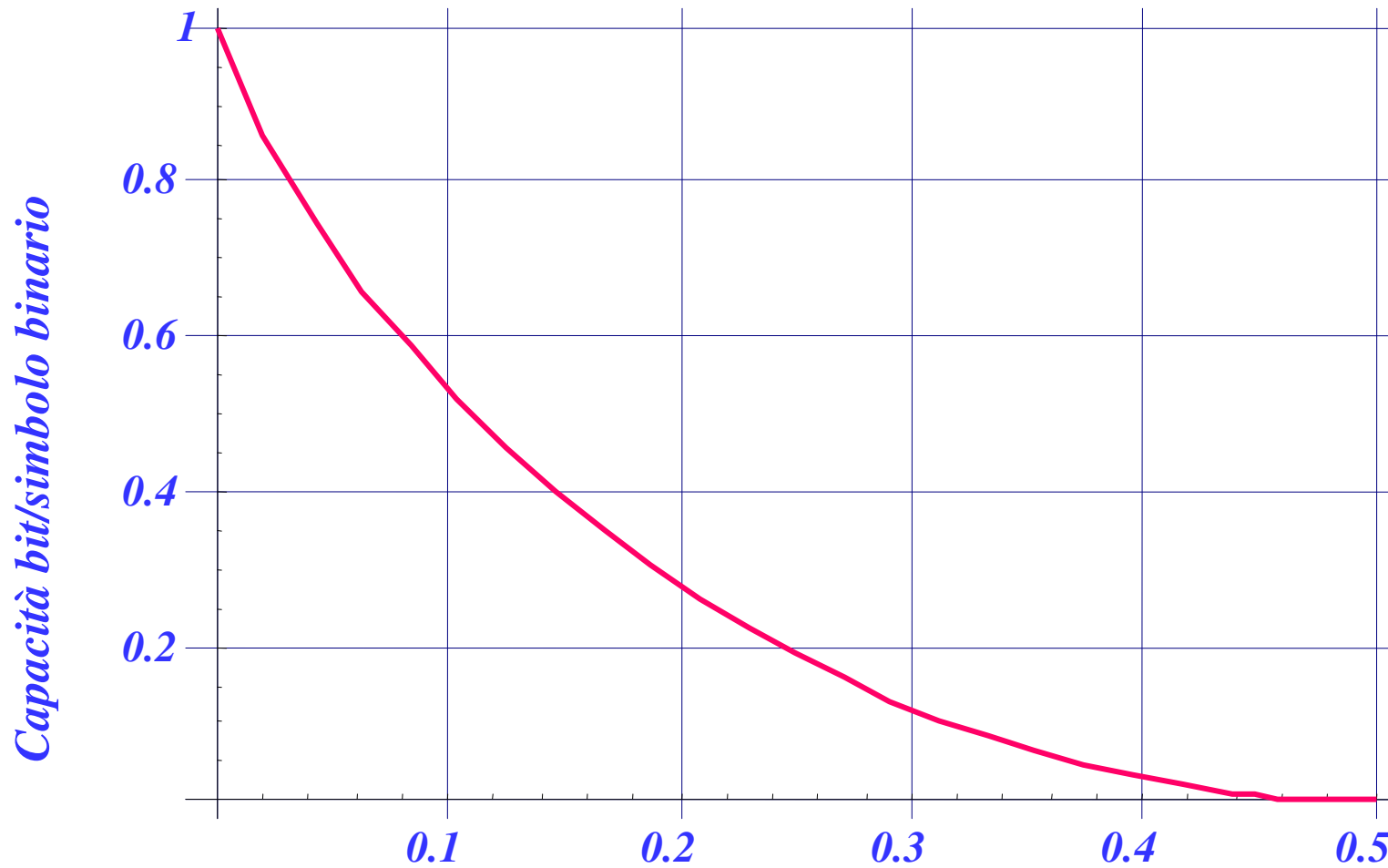
# Capacità del canale gaussiano, sistemi di modulazione e codici

- ❖ Vi è un notevole recupero di efficienza energetica, a parità di banda, confrontando i sistemi tradizionali con i limiti teorici, ciò che ha alimentato, per decenni, lo sviluppo della ricerca sui codici;
- ❖ i codici correttori di errore consentono di avvicinarsi al limite teorico, sebbene a prezzo di notevoli complessità e ritardi di calcolo.





## *Capacità del canale binario simmetrico*



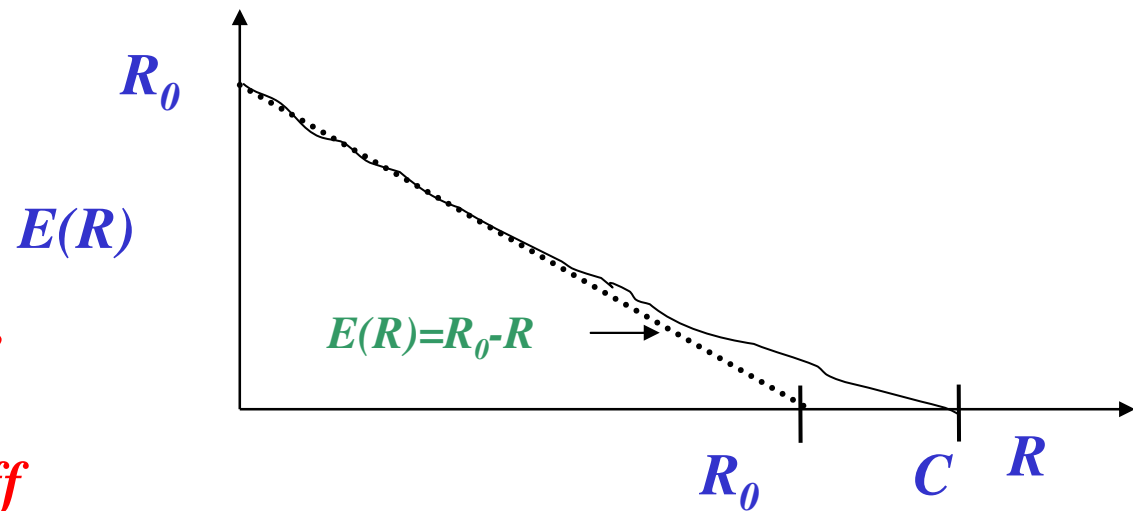
*Probabilità di errore (ad esempio, segnali antipodali,  $p \approx \frac{1}{2} \exp(-E_b/n_0)$ )*

## Limiti teorici e mondo “reale”

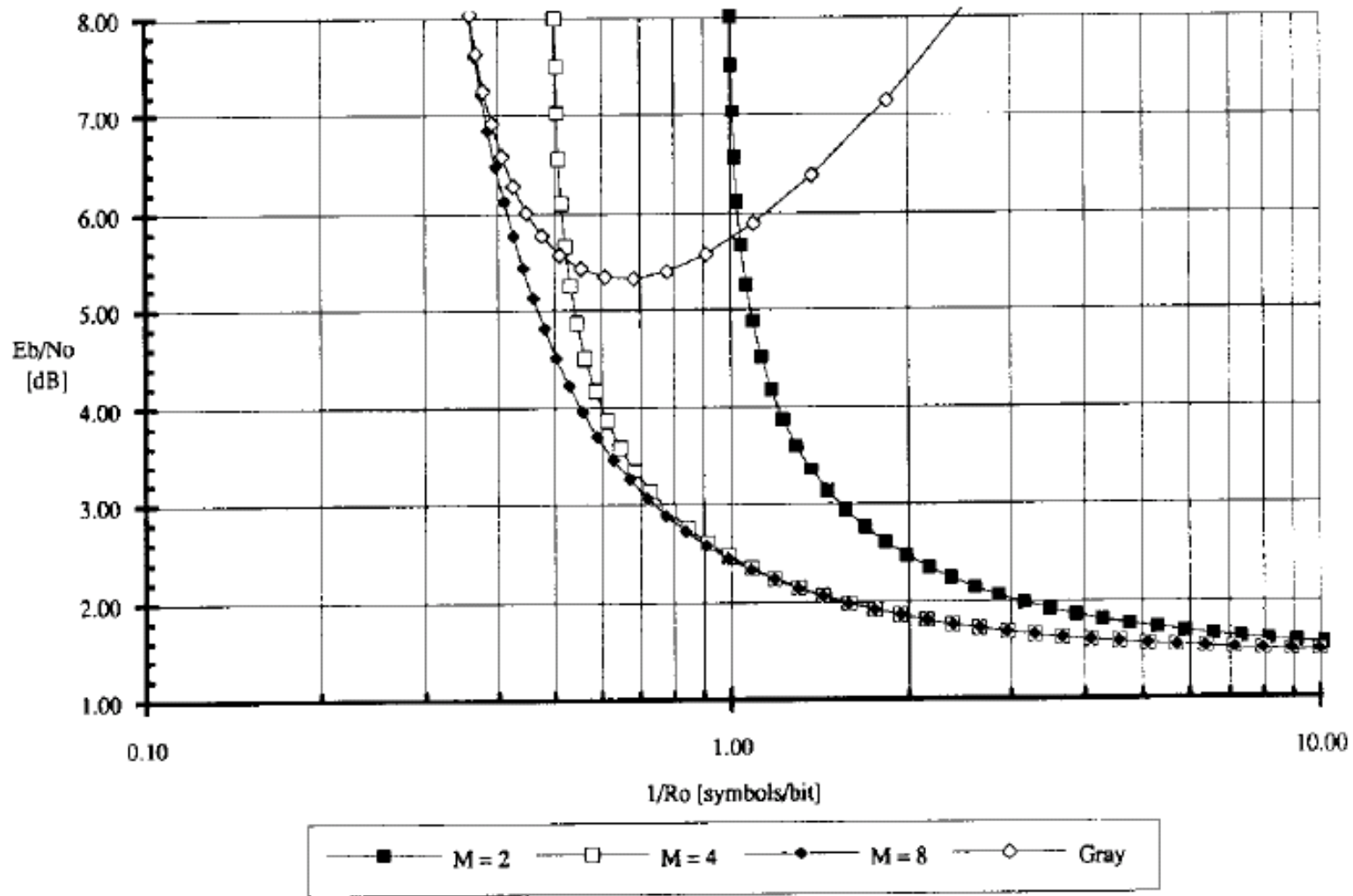
Dato un canale rumoroso di capacità  $C$  (bit/simbolo), ad es. il canale gaussiano prima descritto, si dimostra che esiste una sequenza di codici, di crescente lunghezza di blocco  $n$ , ognuno con un dato rate  $R < C$ , per cui la probabilità di errore di parola  $P(e)$ , con un decodificatore a massima verosimiglianza (sceglie la parola trasmessa massimizzando la probabilità del segnale ricevuto condizionata alla trasmissione di tale parola), va a 0 esponenzialmente con  $n$ . In particolare, si ha

$$P(e) < 2^{-nE(R)}$$

$E(R)$ , esponente di errore, è funzione del rate  $R$ ;  
 $R_0$ , bit/simbolo, detto cut off rate, è legato al canale



# Limiti teorici e mondo "reale"



*Minimo  $E_b/n_0$  vs  $1/R_0$  per modulazioni M-arie su canale gaussiano*

## *Limiti teorici e mondo “reale”*

- ❖  $R_0$ , bit/simbolo, non è solo un parametro di canale;*
- ❖ aumentando  $R$ , rate di trasmissione in bit/simbolo, l'esponente di errore diminuisce linearmente con  $R$ , a partire da  $R_0$ , talché, se tale andamento lineare proseguisse, il massimo rate sarebbe  $R_0$  e non  $C$ ;*
- ❖ in realtà, aumentando  $R$ , l'esponente di errore va a 0 sempre meno velocemente della linearità, e si annulla esattamente a  $C$ , consentendo di raggiungere, almeno teoricamente, la capacità del canale;*
- ❖ tanto minore è l'esponente di errore, tanto maggiore deve essere però  $n$ , per consentire una probabilità di errore molto bassa, e, quindi, tanto maggiore deve essere la corrispondente complessità dei co-decodificatori.*

## *Limiti teorici e mondo “reale”*

❖ *In sostanza, risulta, da un lato la crescente complessità dei codici, legata, alle loro crescenti dimensioni, volendo aumentare  $R$ , dall'altro lato l'accelerato aumento di tale dimensione al di là di  $R_0$ , per il piccolo valore, in tal caso, dell'esponente di errore.*

❖ *inoltre, per i codici convoluzionali, usando per la decodifica lo stack algorithm, il superamento di  $R_0$  pone gravi problemi computazionali;*

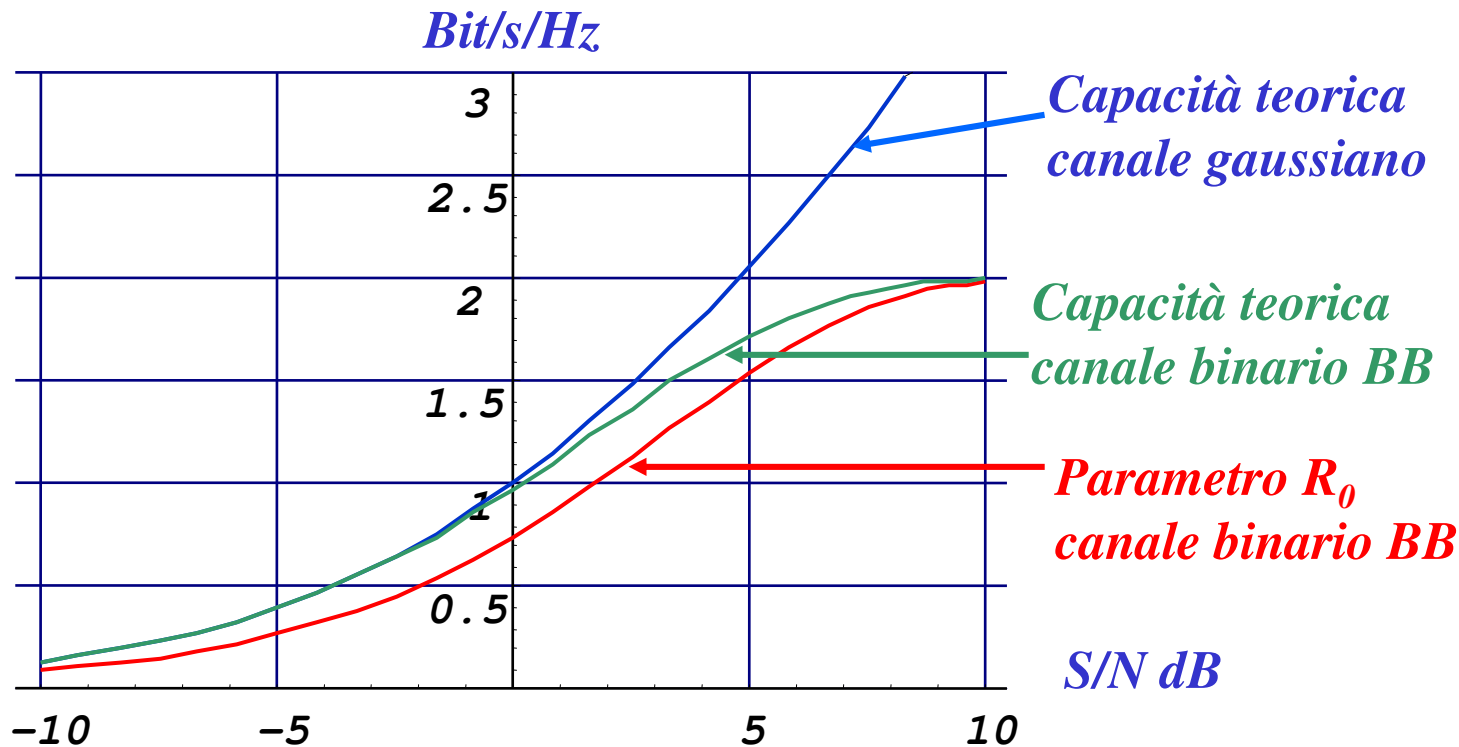
❖ *da queste considerazioni,  $R_0$ , pur non essendo un vero limite per i sistemi realizzabili, dà importanti indicazioni sulla fattibilità tecnologica dei sistemi stessi, in termini di difficoltà computazionali;*

❖ *in altre parole, il cutoff rate  $R_0$  è il parametro più importante di codifica, il più vicino all'effettivo data rate conseguibile con strutture pratiche.*

## Limiti teorici e mondo “reale”

La capacità del canale gaussiano, come già determinata, non è correlata ad una particolare scelta dello schema trasmissivo, ad esempio PSK, QAM, o al numero dei livelli, ma semplicemente pone un vincolo sulla potenza media trasmessa.

Ove si consideri, viceversa, un determinato schema trasmissivo, se ne deve individuare sia la **capacità teorica**, sia il **parametro  $R_0$** . Ad esempio, per il canale binario in banda base si ha

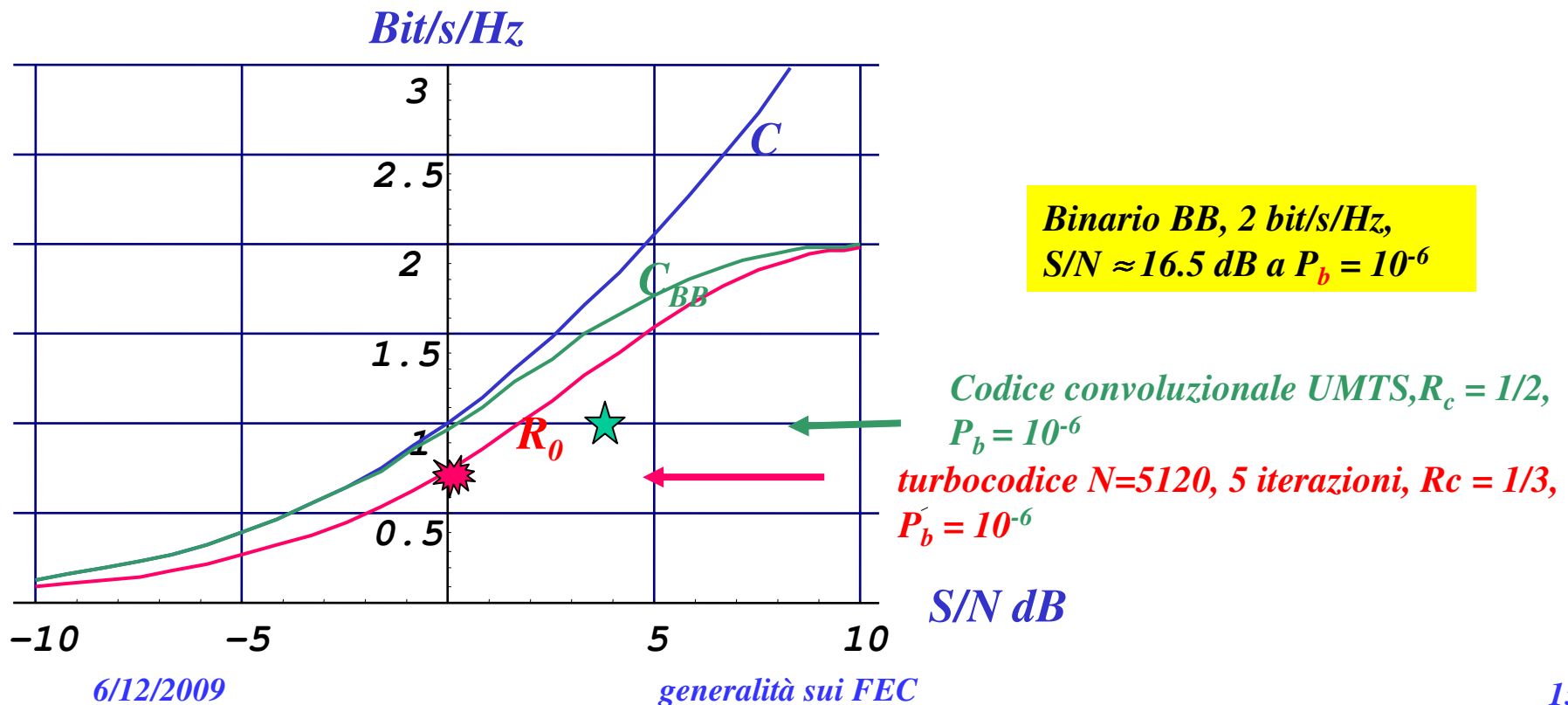


Con questi parametri vanno confrontati i codici reali

## Limiti teorici e mondo “reale”

E' evidente che, con riferimento ad un particolare schema trasmissivo, la capacità teorica è comunque limitata dalla numerosità dei simboli.

Inoltre, il parametro  $R_0$  indica chiaramente un gap tra risultati teorici e fattibilità tecnologiche.



## *La codifica: perché?*

### ➤ *Per **rivelare** gli errori e sorvegliare la qualità*

*La rilevazione degli errori consente di ritrasmettere le informazioni errate qualora si possa tollerare il ritardo (**ARQ**: Automatic Repeat reQuest)*

### ➤ *Per **correggere** gli errori*

*La **ridondanza** e la **memoria** insite nel processo di codifica sono utilizzate per la correzione degli errori se il sistema richiede il tempo reale (**FEC**: Forward Error Correction)*



## *La codifica: un po' di storia*

*I codici a blocco furono i primi sviluppati; essi raccolgono  $k$  bit (simboli) informativi e li mappano in  $n$  bit (simboli) totali, aggiungendo ridondanza. Hamming scoprì i codici omonimi, che correggono un errore, nel 1950.*

*Il passo importante successivo fu la scoperta dei codici RS (Reed Solomon) e BCH (Bose Chaudhuri Hocquenghem) nel 1959 - 1960; i codici binari BCH correggono un numero arbitrario di errori, ma sono asintoticamente deboli; ciò si risolve concatenando codici BCH corti con codici RS, non binari, lunghi (1966). Le prime tecniche di decodifica sono del 1954, ma quelle più importanti, algebriche, sono state sviluppate dal 1960 al 1968.*

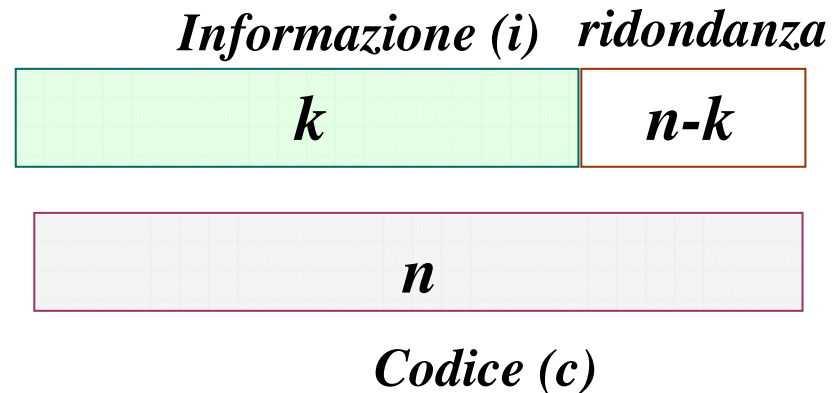
## *La codifica: un po' di storia*

*Completamente diversi dai codici a blocco, i codici convoluzionali (Elias 1955) hanno memoria, così che il mapping dai  $k$  bit agli  $n$  bit tiene conto del passato; le prime tecniche di decodifica sono sequenziali (1961), quelle attuali sono dovute alla eccezionale scoperta dell'algoritmo di Viterbi ('67).*

*Applicando ingegnosamente gli ingredienti esistenti, Berrou e altri (1993) hanno scoperto i turbocodici, che si avvicinano finalmente ai limiti teorici, combinando codificatori tradizionali, interleaver e decodifica iterativa.*

# *Codici a blocco*

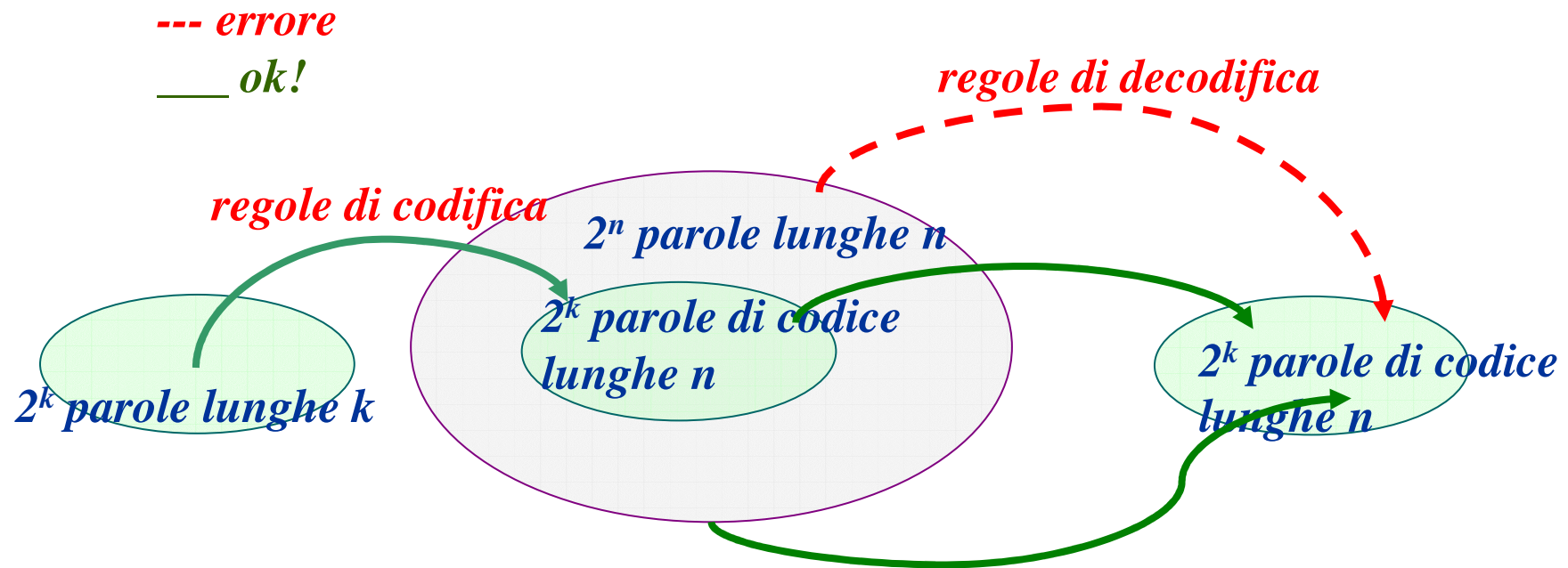
## Codici a blocco



$$R_C = k/n$$

- *L'associazione fra parole di informazione e di codice è biunivoca*
- *gli  $n - k$  bit di ridondanza (e di memoria, essendo relativi a bit informativi) hanno un contenuto informativo associabile, in binario, sino a  $2^{n-k}$  configurazioni di errore.*

# Codici a blocco



- *Selezionano un sottoinsieme di  $2^k$  parole di codice delle  $2^n$  parole totali*
- *Se gli errori trasformano una parola di codice in una parola appartenente al sottoinsieme complementare, l'errore è rivelabile*
- *Capacità di correzione/rivelazione e memoria e ridondanza sono strettamente legate e dipendono dalle regole di codifica*

## *Codificare e decodificare*

*Se codifica e decodifica avvengono senza regole, sono necessarie due tabelle di  $2^k$  elementi, rispettivamente di  $k$  bit (le parole da codificare) e  $n$  bit (le parole codificate) per la codifica, e di due tabelle -una di  $2^n$  elementi di  $n$  bit ed una di  $2^k$  elementi di  $k$  bit - per la decodifica*

*Questo consentirebbe l'implementazione solo di codici piccoli e con scarse capacità correttive*

*E' quindi necessario avere semplici ma potenti regole di codifica e decodifica; in particolare, in codifica, regole agevoli per generare dai  $k$  bit informativi gli  $n-k$  bit di parità*

*Nel seguito si fa riferimento ai codici sistematici, cioè a quei codici tali che i primi  $k$  bit individuano i bit informativi e gli altri  $n-k$  sono i bit di ridondanza*

# *Decodifica*

*Le regole di codifica generano  $n-k$  bit di parità in funzione dei  $k$  bit informativi*

*La decodifica riapplica le regole di codifica ai primi  $k$  bit ricevuti, ricalcolando gli  $n-k$  bit di parità*

*Se gli  $n-k$  bit ricalcolati coincidono con quelli ricevuti:*

- non c'è stato errore*
- l'errore non è rivelabile*

*Se gli  $n-k$  bit ricalcolati non coincidono con quelli ricevuti:*

- la differenza fra i bit di parità ricevuti e ricalcolati è la sindrome degli errori sul blocco, che ne permette l'individuazione (entro i limiti di correzione del codice)*

## *Distanza di Hamming*

- *Le parole dei codici sono associabili a vettori*
- *Il criterio con cui è definita la distanza tra le parole di codice, parametro su cui si basa la correzione, si adatta all'ambiente nel quale il codice opera*
- *In contesti binari è usata la distanza di Hamming, uguale al numero di posizioni in cui i simboli tra due parole di codice sono diversi*
- *La distanza di Hamming è una distanza nel **senso matematico** del termine, poiché soddisfa i requisiti di **nonnegatività**, **simmetria** e **diseguaglianza triangolare***

*a*    **0 0 0 1 0 1 1 1**

*b*    **0 1 0 0 0 1 0 1**

$$d_H(a,b) = 3$$

*generalità sui FEC*



## *Rivelazione e correzione degli errori*

- *La rivelazione si basa sulla constatazione che una parola non appartiene al codice*
- *La decodifica si basa sulla associazione della parola ricevuta alla parola di codice a distanza di Hamming minima*

*a, b: parole di codice*

*trasmissione*                    *a* 00010111;            *b* 01000101  
*ricezione*                        *a* 00010111  
    *y* 00010111

*d(y,a)=0; d(y,b)=3*    *errori rivelati: 0*            *errori corretti: 0*

*decodifica*                    *a* 00010111

## *Rivelazione e correzione degli errori*

*trasmissione*      *a* 0 0 0 1 0 1 1 1      *b* 0 1 0 0 0 1 0 1  
*ricezione*          *a* 0 0 0 1 0 1 1 1  
                         *y* 0 1 0 1 0 1 1 1

$d(z,x)=1; d(z,y)=2$       *errori rivelati: 1*      *errori corretti: 1*

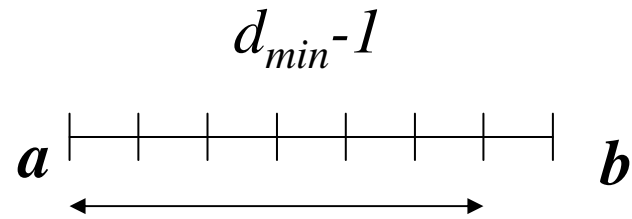
*decodifica*                      *a* 0 0 0 1 0 1 1 1

*trasmissione*      *a* 0 0 0 1 0 1 1 1      *b* 0 1 0 0 0 1 0 1  
*ricezione*          *a* 0 0 0 1 0 1 1 1  
                         *y* 0 1 0 1 0 1 0 1

$d(y,a)=2; d(y,b)=1$       *errori rivelati: 2*      *errori corretti: nessuno*

*decodifica*                      *parola errata*

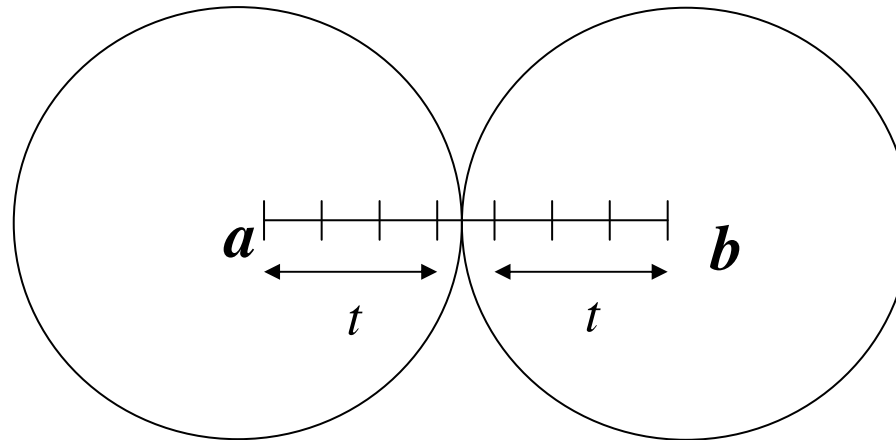
## *Rivelazione degli errori*



## *Rivelazione degli errori*

*Se si intende rivelare gli errori, verificando solamente l'appartenenza della parola al codice, ciò è sicuramente possibile se il numero degli errori è inferiore a  $d_{min} - 1$  perché in questo caso, anche se vi sono errori, non è comunque ricevuta una parola di codice*

## *Correzione degli errori*

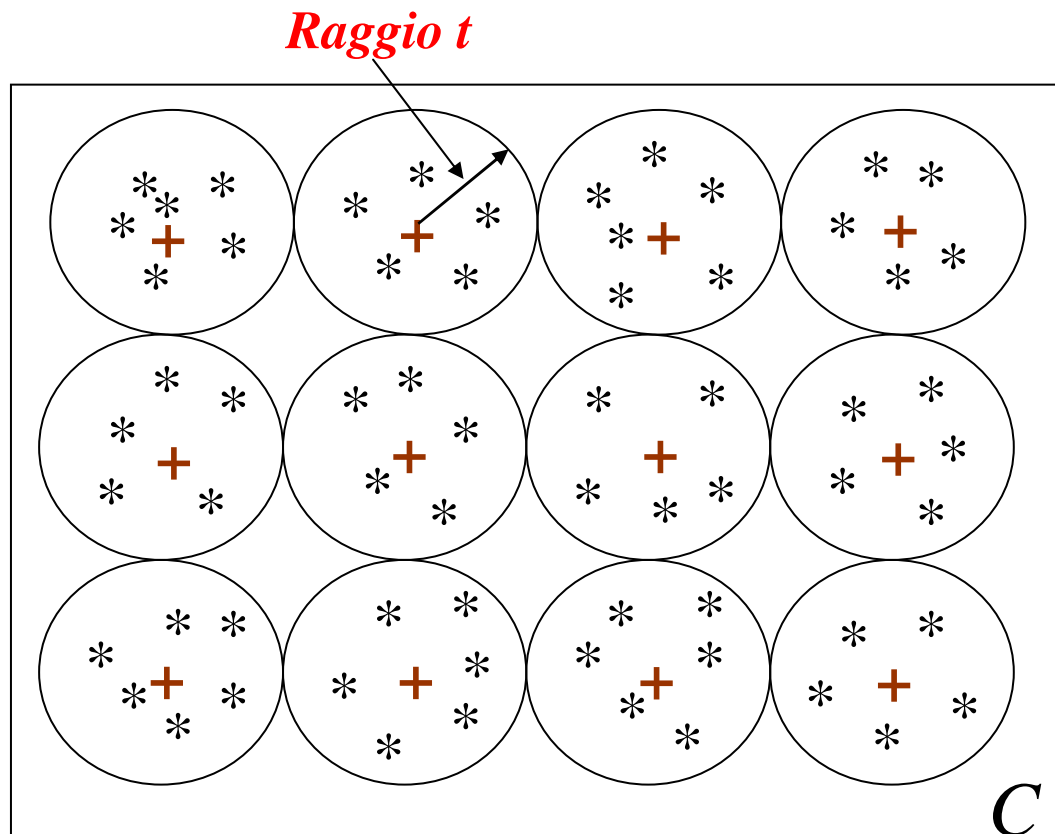


## *Correzione degli errori*

*A ciascuna parola di codice  $c_i$ , detta  $d_{min}$  la minima distanza di Hamming, si può associare una sfera di raggio  $t = (d_{min}-1)/2$ , centrata su  $c_i$ , che raccoglie tutte le possibili  $n$ -ple che distano al massimo  $t$  rispetto alla parola in oggetto.*

*Le sfere sono la regione di decodifica per le parole di codice*

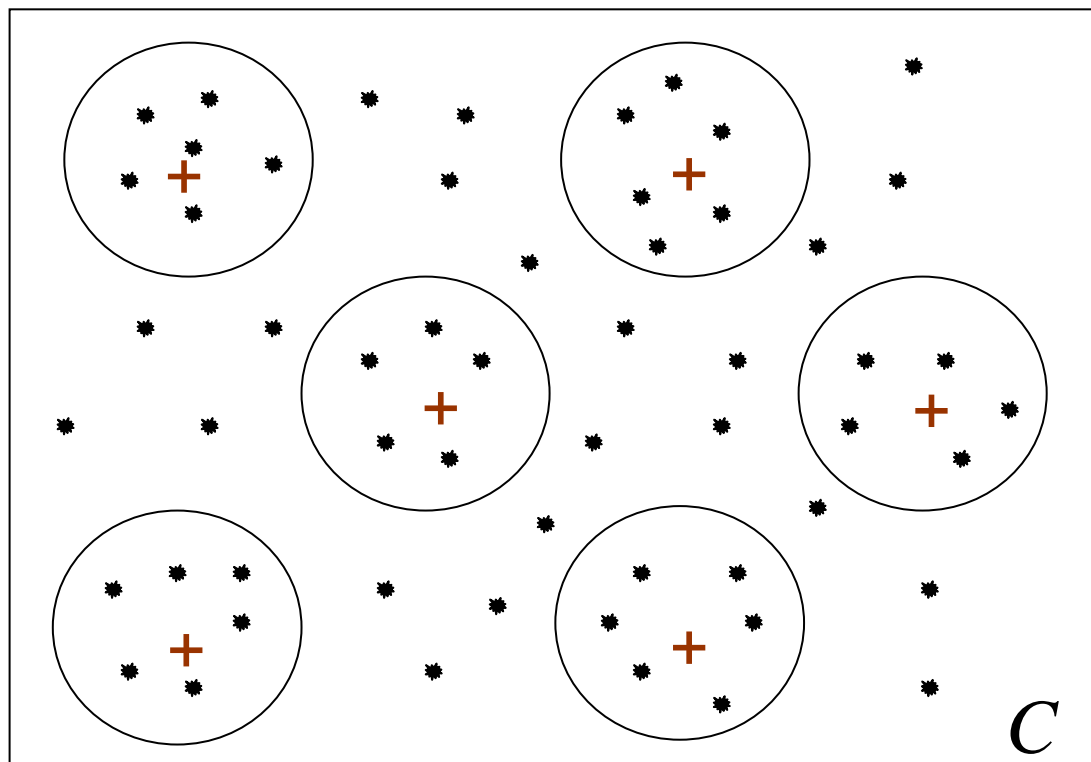
## Codici perfetti



*Nei codici perfetti tutte le parole ricevute sono associabili ad una parola di codice, cioè tutti gli errori rivelabili sono correggibili. Tutte le sfere hanno raggio uguale  $t$ . Non si hanno regioni intra-sfere. Il numero delle possibili parole a distanza  $2t+1$  è massimo, e così  $R_c$ .*

*I codici perfetti sono molto rari, in pratica la famiglia dei codici di Hamming, il codice binario di Golay (23,11),  $t = 3$ , il codice ternario di Golay (11,6),  $t=2$ .*

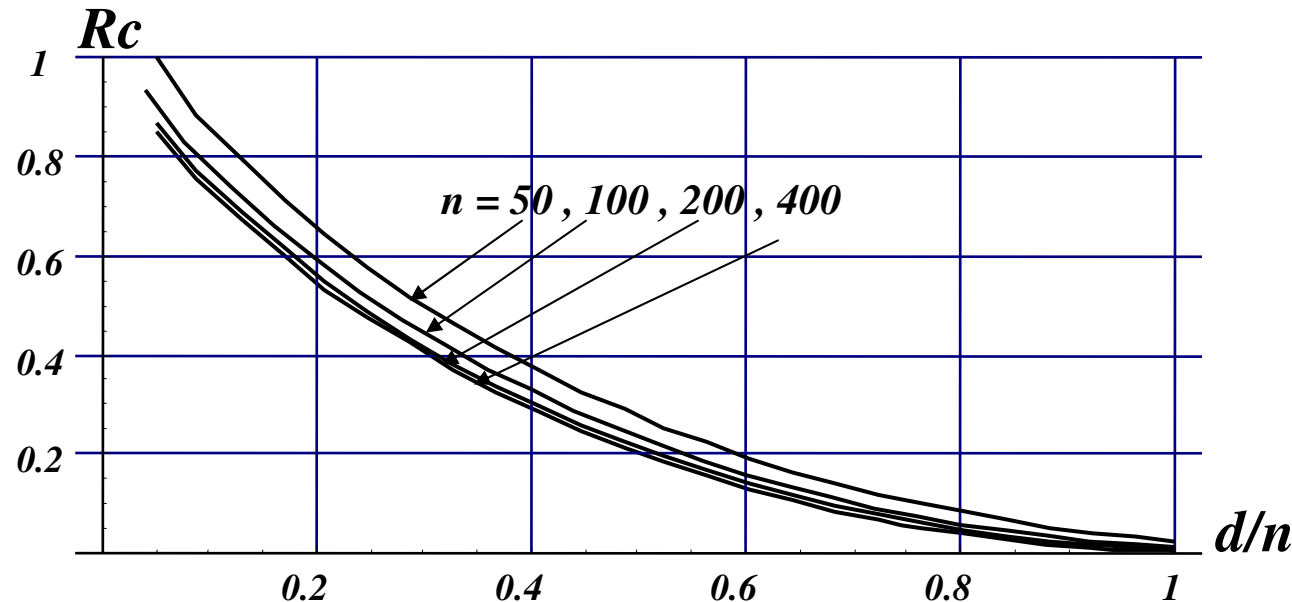
## *Codici non perfetti*



*Nei **codici non perfetti** le parole ricevute non sono tutte associabili ad una parola di codice. Ci sono cioè regioni intra-sfere, con sfere di raggio non necessariamente uguale. A parità di  $n$  e di distanza, si hanno meno parole rispetto al caso ideale,  $k$  è minore, e così anche  $R_c$ .*

*Nei **codici quasi perfetti** tutte le parole ricevute sono o associabili a sfere di uguale raggio  $t$ , o, se fuori da esse, hanno al più distanza  $t+1$  dalle parole del codice. A parità di  $n$  e di distanza, il numero di parole di codice può essere vicino a quello dei codici ideali, con simili valori di  $R_c$ .*

## *Rate di codifica vs. $d/n$ con codici perfetti*



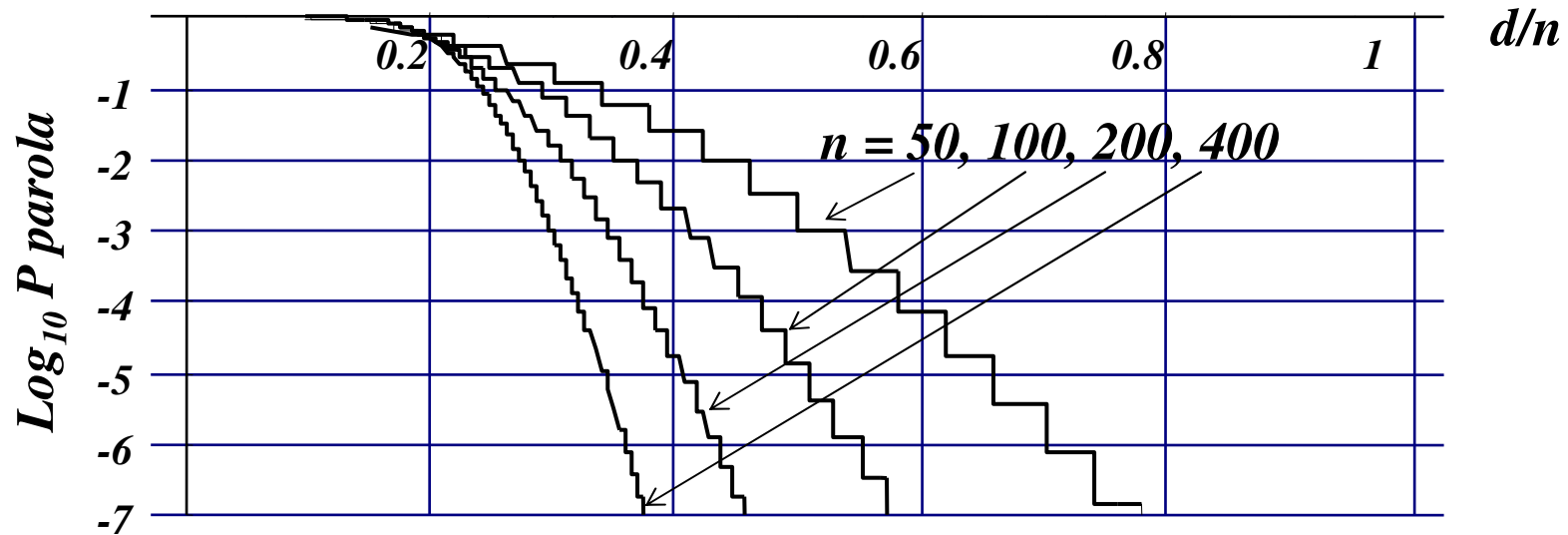
*Le curve convergono assai velocemente con  $n$*

*Dato  $n$ , all'aumentare del rapporto  $d/n \approx 2t/n$ , due volte il rapporto tra numero di errori correggibili e  $n$ , si deve diminuire  $R = k/n$ , aumentando  $n-k$ , con conseguente minor rendimento, e ciò è scarsamente influenzato dal valore di  $n$ , almeno per grande*

*Se si vogliono correggere molti errori si hanno rendimenti limitati*

# *$P_e$ parola vs. $d/n$ con codici perfetti - canale a $P_{bit}$ costante*

*$P_{bit}$  canale = 0.1*

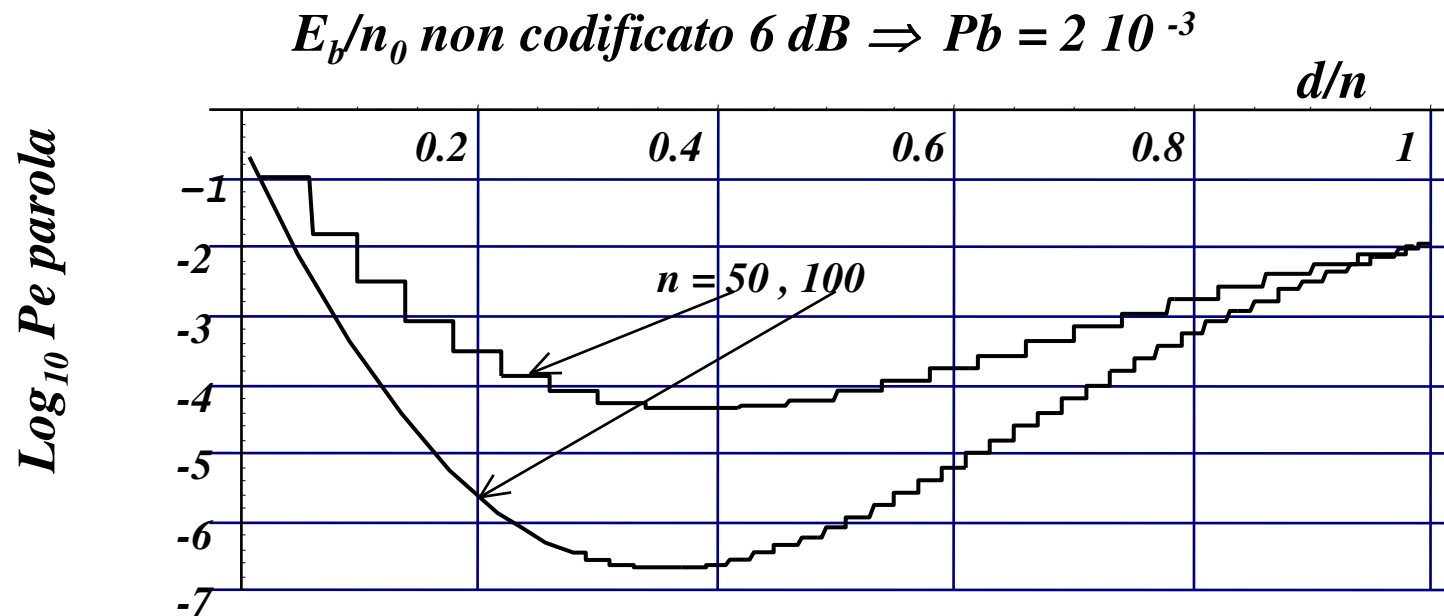


*Aumentando  $n$ , con codici perfetti, a parità di  $P_e$  basta un valore di  $d/n$  minore con vistosi aumenti del rate del codice*

*I codici sono tanto più efficienti quanto più  $n$  è grande, almeno in una visione ideale.*



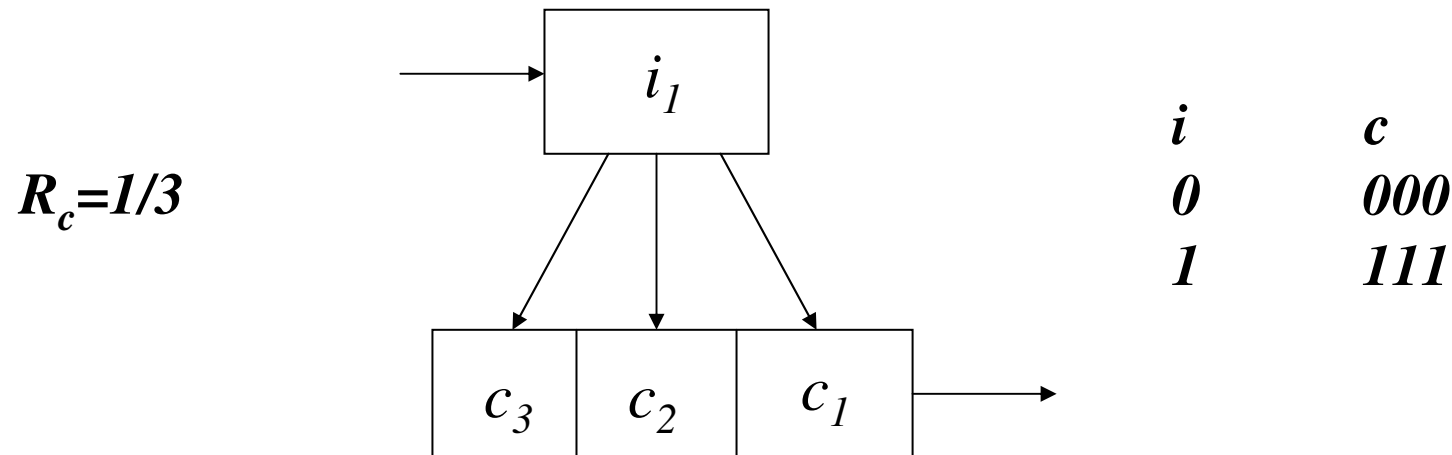
## *$P_e$ parola vs. $d/n$ con codici perfetti - canale gaussiano*



*Nei canali gaussiani la diminuzione del rate del codice comporta la diminuzione di  $E_c/n_0$  (energia per bit codificato/densità spettrale di rumore), quindi, pur valendo le considerazioni generali già fatte, **per ogni valore di  $n$  l'aumento di  $d/n$ , e quindi la diminuzione del rate del codice, all'inizio migliora le prestazioni, poi, dopo un massimo piuttosto piatto, le peggiora.***

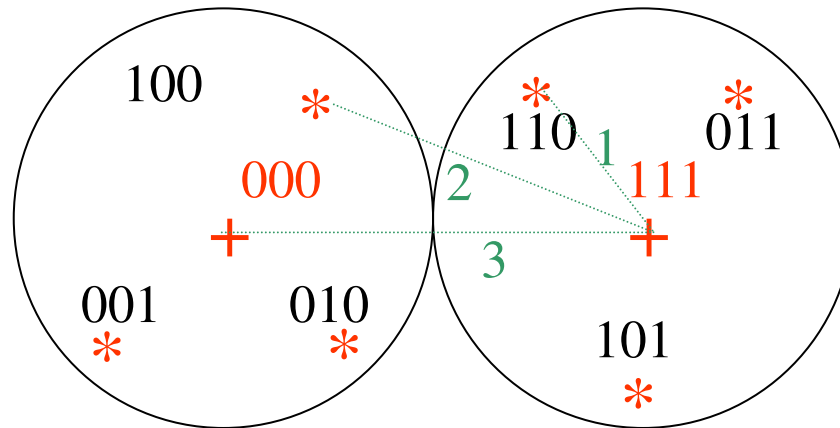
*Si conferma, almeno coi codici ideali, la superiorità dei codici lunghi*

## Esempio, codice a ripetizione (3,1)



- *il codice ripete tre volte i bit informativi*
- *Esempio:*  
*tx: 1 0 0 1 1 0 1*  
*cod: 111 000 000 111 111 000 111*  
*rx: 101 000 011 110 111 100 111*
- *può essere corretto un errore e rivelati due errori*

## Codice a ripetizione (3,1)



*Lo spazio delle parole di codice è dato da 2 parole di 3 bit (000, 111)  
Al ricevitore si possono presentare, a causa di eventuali errori, tutte le  $2^3=8$  possibili combinazioni di 3 bit*

*Il codice può:*

- correggere, associando alla parola ricevuta quella a distanza minima (il centro della sfera di appartenenza)*
- rivelare, limitandosi ad annunciare al trasmettitore che la parola ricevuta non è una parola di codice*

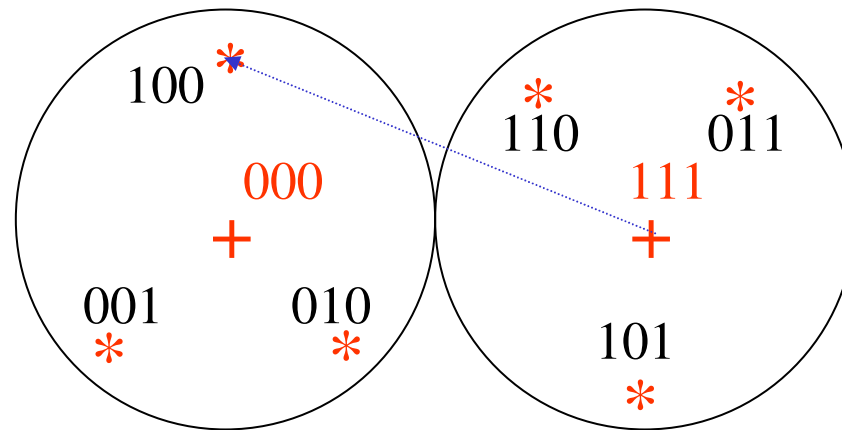
***E' un codice perfetto***

6/12/2009

generalità sui FEC

35

## *Codice a ripetizione (3,1)*



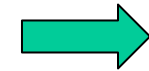
- *Nel caso in cui si hanno due errori (per esempio 111 diventa 001), l'associazione con una parola di codice può determinare un ulteriore aumento degli errori (111 diventa 000)*
- *Per ciò questo codice deve essere usato in canali dove due bit errati in una parola di codice sia un evento molto raro*
- *In generale, stressare il codice oltre le sue capacità correttive produce un danno piuttosto che un beneficio*
- *equivalentemente, l'impiego di qualunque codice deve essere subordinato alla conoscenza precisa delle caratteristiche statistiche del canale, in termini di errori*

## Codici perfetti e mondo “reale”


*Se si devono correggere  $t$  errori, con distanza  $d = 2t+1$ , il minimo numero di bit di ridondanza necessari, ovvero  $n-k$ , è evidentemente dato dal logaritmo del numero delle configurazioni di errori fino a  $t$*

$$n - k = \log_2 \sum_{i=0}^t \binom{n}{i}$$

*Equivalentemente, tutto lo spazio delle  $2^n$  parole deve essere suddiviso esattamente in  $2^k$  sfere ciascuna comprendente tutte le parole a distanza  $t$  dalla parola di codice, quindi con volume*


$$\sum_{i=0}^t \binom{n}{i}$$

*Il numero delle parole di codice (upper bound, essendo chiaramente una situazione ideale) è*


$$2^k \leq \frac{2^n}{\sum_{i=0}^t \binom{n}{i}}$$

## *Codici perfetti e mondo “reale”*

*I codici perfetti costituiscono un upper bound molto ottimista, difficile da conseguire; per questo i codici perfetti sono così rari.*

*Un bound, più rispettoso della realtà, è quello di Gilbert - Varshamov.*

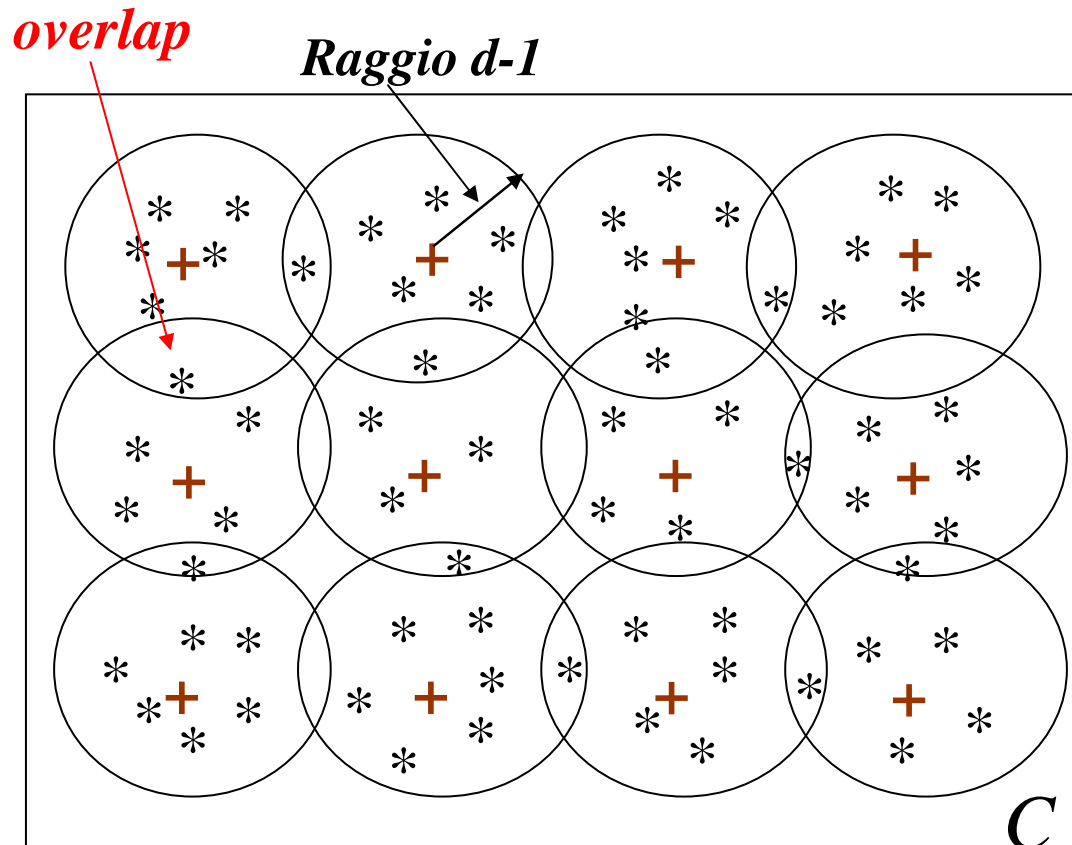
*Si consideri, al riguardo, un codice  $C$ , costruito nello spazio delle  $2^n$   $n$  uple binarie, e si voglia che abbia distanza minima  $d$ .*

*$C$  ha, evidentemente, un massimo numero di parole se nessuna altra parola può essere aggiunta a  $C$  senza ridurre la minima distanza.*

*Se ciò non è possibile, significa che ogni parola, non del codice, è a distanza  $\leq d - 1$  da una parola del codice, cioè ogni parola del codice è il centro di una sfera di raggio  $d-1$ , e tali sfere coprono lo spazio, cioè*

$$2^k \sum_{i=0}^{d-1} \binom{n}{i} \geq 2^n \quad \Rightarrow \quad 2^k \geq \frac{2^n}{\sum_{i=0}^{d-1} \binom{n}{i}}$$

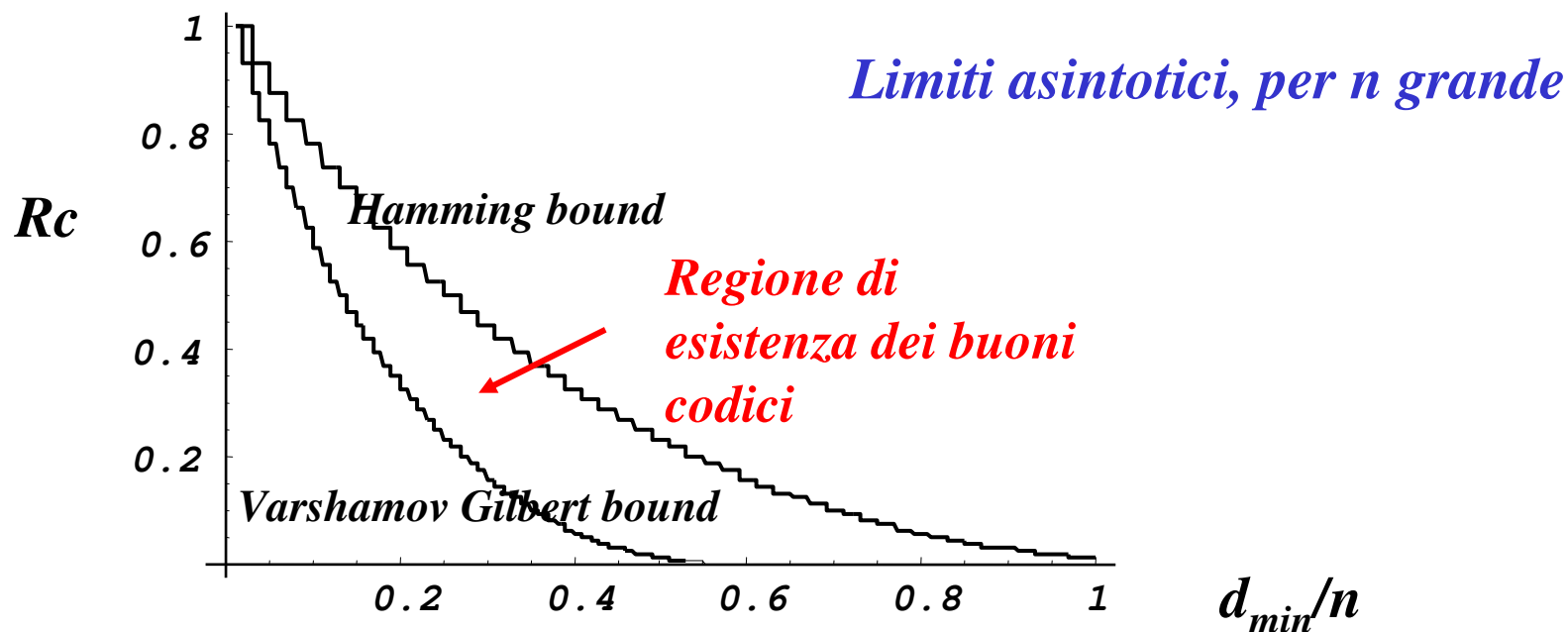
## Limite di Gilbert Varshamov



*Lo spazio delle  $n$  uple si riempie aggiungendo progressivamente parole di codice che sono a distanza  $d$  dalle altre parole di codice. Ci si ferma quando non ci sono più parole che sono a distanza maggiore o uguale a  $d$  rispetto alle altre parole di codice.*

*Il caso peggiore, ovvero il minor numero di parole di codice, e quindi il minimo valore di  $R$  si ha quando tutte le sfere sono disgiunte, in realtà normalmente ci sono sovrapposizioni, con superiori valori di  $R_c$ .*

## *codici perfetti e mondo “reale”*



- *I codici perfetti (Hamming bound) sono molto rari, come già visto*
- *esistono **sicuramente** codici con prestazioni superiori al limite di Varshamov Gilbert*
- *per  $n$  piccolo, molti codici superano il limite di Varshamov Gilbert; quando, viceversa,  $n$  è grande, tale limite è più difficile da conseguire.*
- *Varshamov Gilbert è il limite per confrontare i codici tra loro.*



## *Codici lineari*

*La maggioranza dei buoni codici a blocco noti sono lineari.*

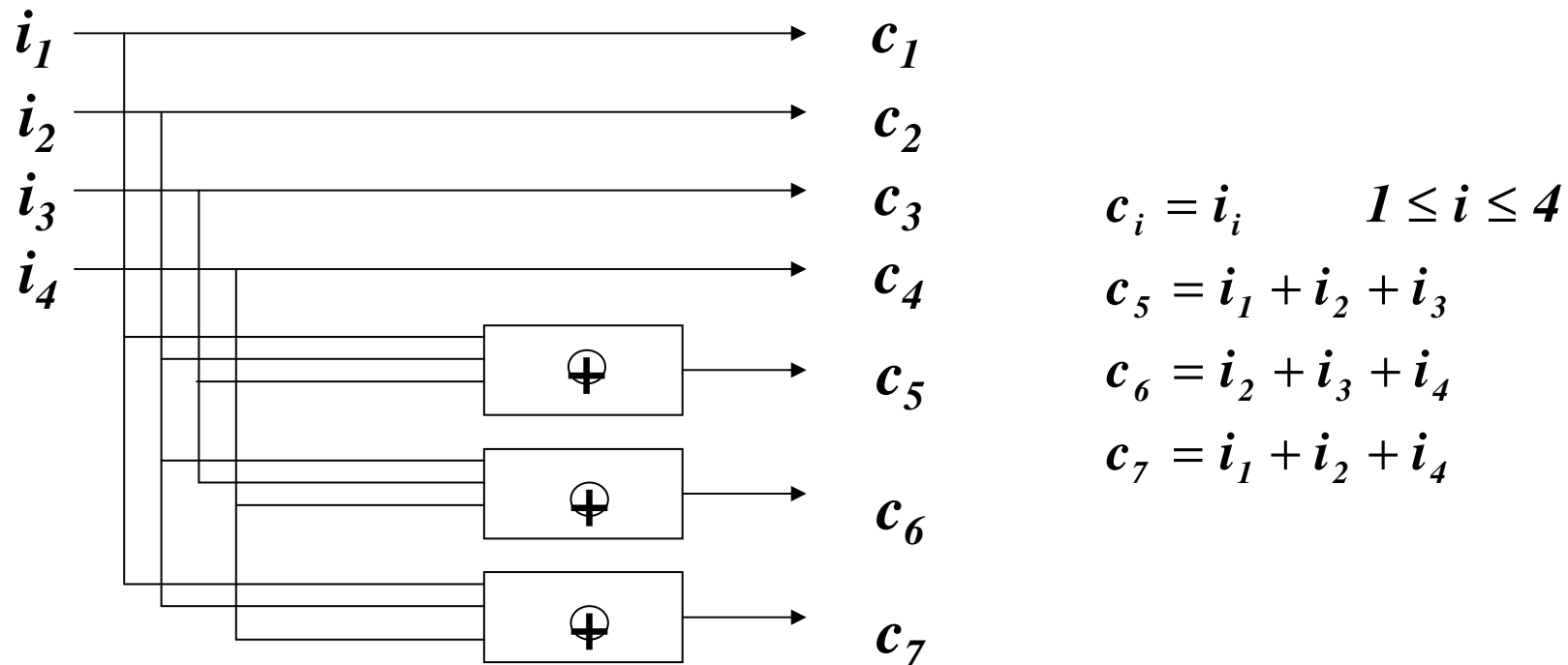
*In pratica sono un **sottospazio vettoriale**, di dimensione **k** (tutte le k uple del codice), immerso in uno **spazio vettoriale** di dimensione **n** (tutte le n uple)*

*Essi hanno forti proprietà strutturali che rendono praticamente fattibili codificatori e decodificatori.*

*In sostanza si possono usare per la codifica e la decodifica semplici operazioni matriciali lineari, ovvero somme (or esclusivo) e prodotti.*

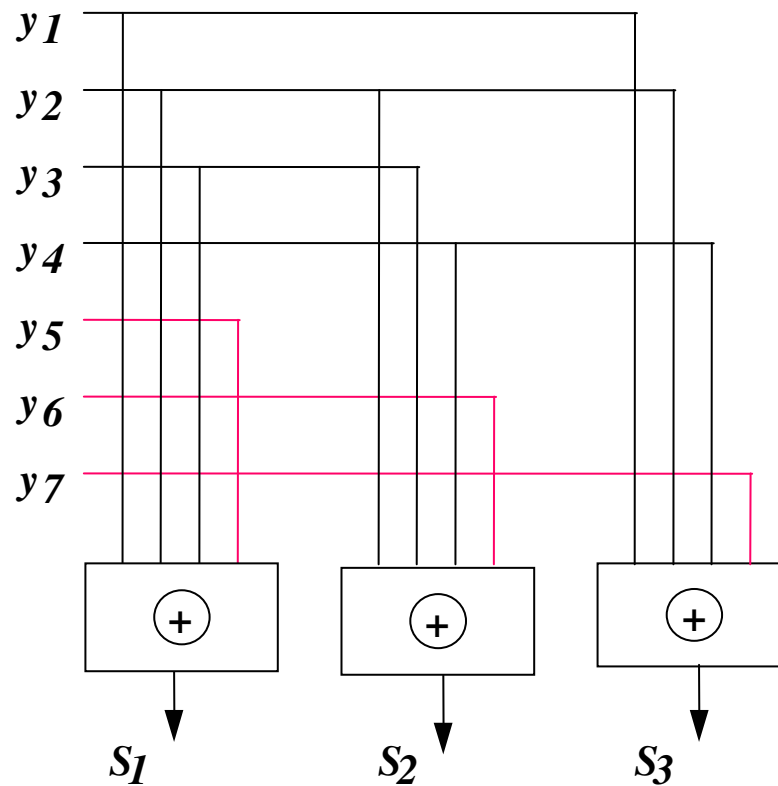
*Tra l'altro, le più potenti tecniche algebriche note sono utili solo per codici di tipo lineare.*

## Codice lineare di Hamming (7,4)



- *La parola codificata è ottenibile con operazioni algebriche (somme e moltiplicazioni) dalla parola informativa*
- *3 bit di ridondanza possono individuare  $2^3 = 8$  configurazioni, cioè nessun errore e un errore in 7 posizioni*

## Decodifica



*In decodifica si ricalcolano i bit di parità e si confrontano con quelli ricevuti; l'eventuale differenza tra tali bit si dice sindrome*

*La sindrome non dipende dalla parola di codice, ma solo dall'errore*

## *Esempio*

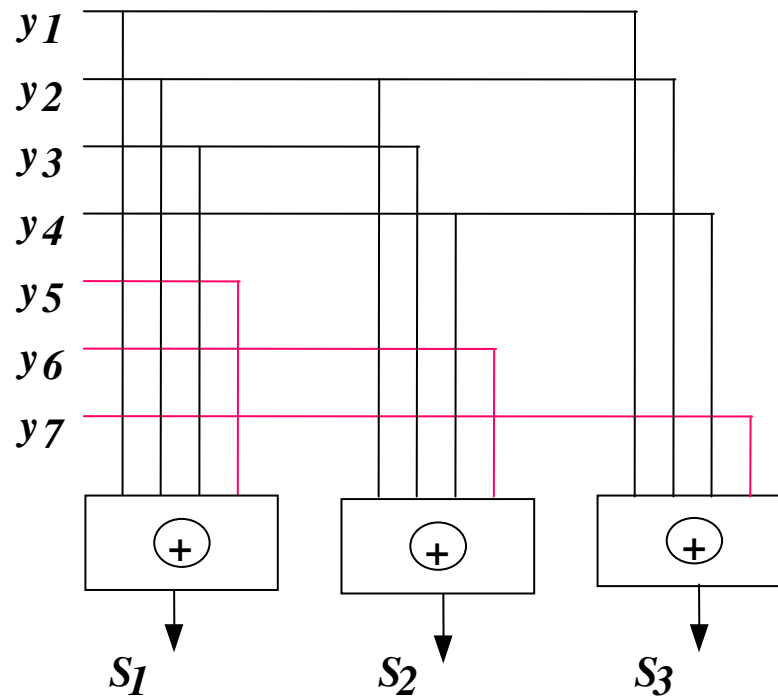
<i>parole di codice</i>	<b>+</b>	<i>errore</i>	<b>=</b>	<i>parola ricevuta</i>	<i>sindrome calcolata</i>	
<i>0000 000</i>		<i>0000001</i>		<i>0000001</i>	<i>001</i>	
<i>0001 011</i>		<i>0000001</i>		<i>0001010</i>	<i>001</i>	
<i>0010 110</i>		<i>0000001</i>		<i>0010111</i>	<i>001</i>	
<i>0011 101</i>		<i>0000001</i>		<i>0011100</i>	<i>001</i>	<i>La</i>
<i>0100 111</i>		<i>0000001</i>		<i>0100110</i>	<i>001</i>	<i>sindrome</i>
<i>0101 100</i>		<i>0000001</i>		<i>0101101</i>	<i>001</i>	<i>identifica</i>
<i>0110 001</i>		<i>0000001</i>		<i>0110000</i>	<i>001</i>	<i>in modo</i>
<i>0111 010</i>		<i>0000001</i>		<i>0111011</i>	<i>001</i>	<i>univoco</i>
<i>1000 101</i>		<i>0000001</i>		<i>1000100</i>	<i>001</i>	<i>l'errore</i>
<i>1001 110</i>		<i>0000001</i>		<i>1001111</i>	<i>001</i>	
<i>1010 011</i>		<i>0000001</i>		<i>1010010</i>	<i>001</i>	
<i>1011 000</i>		<i>0000001</i>		<i>1011001</i>	<i>001</i>	
<i>1100 010</i>		<i>0000001</i>		<i>1100011</i>	<i>001</i>	
<i>1101 001</i>		<i>0000001</i>		<i>1101000</i>	<i>001</i>	
<i>1110 100</i>		<i>0000001</i>		<i>1110101</i>	<i>001</i>	
<i>1111 111</i>		<i>0000001</i>		<i>1111110</i>	<i>001</i>	

## *Esempio*

<i>parola di codice</i>	<b>+</b>	<i>errore</i>	<b>=</b>	<i>parola ricevuta</i>	<i>sindromi calcolate</i>
<i>0000 000</i>		<i>0000000</i>		<i>0000000</i>	<i>000</i>
<i>0000 000</i>		<i>0000001</i>		<i>0000001</i>	<i>001</i>
<i>0000 000</i>		<i>0000010</i>		<i>0000010</i>	<i>010</i>
<i>0000 000</i>		<i>0000100</i>		<i>0000100</i>	<i>100</i>
<i>0000 000</i>		<i>0001000</i>		<i>0001000</i>	<i>011</i>
<i>0000 000</i>		<i>0010000</i>		<i>0010000</i>	<i>110</i>
<i>0000 000</i>		<i>0100000</i>		<i>0100000</i>	<i>111</i>
<i>0000 000</i>		<i>1000000</i>		<i>1000000</i>	<i>101</i>

*Come già osservato, tre bit di parità identificano in modo univoco  
 $2^3=8$  configurazioni di errore (si corregge un errore)*

## Esempio: 2 errori



$$y = c + e_1 + e_2 = 0001011 + 0100000 + 0000100 = 0101111$$

$$s_y = 011$$

$$e = 0001000$$

$$y = y - e = 0101111 + 0001000 = 0100111$$

*Il decoder non ricostruisce la parola trasmessa, ma, erroneamente, considera la parola di ugual sindrome, ma relativa ad un solo errore*

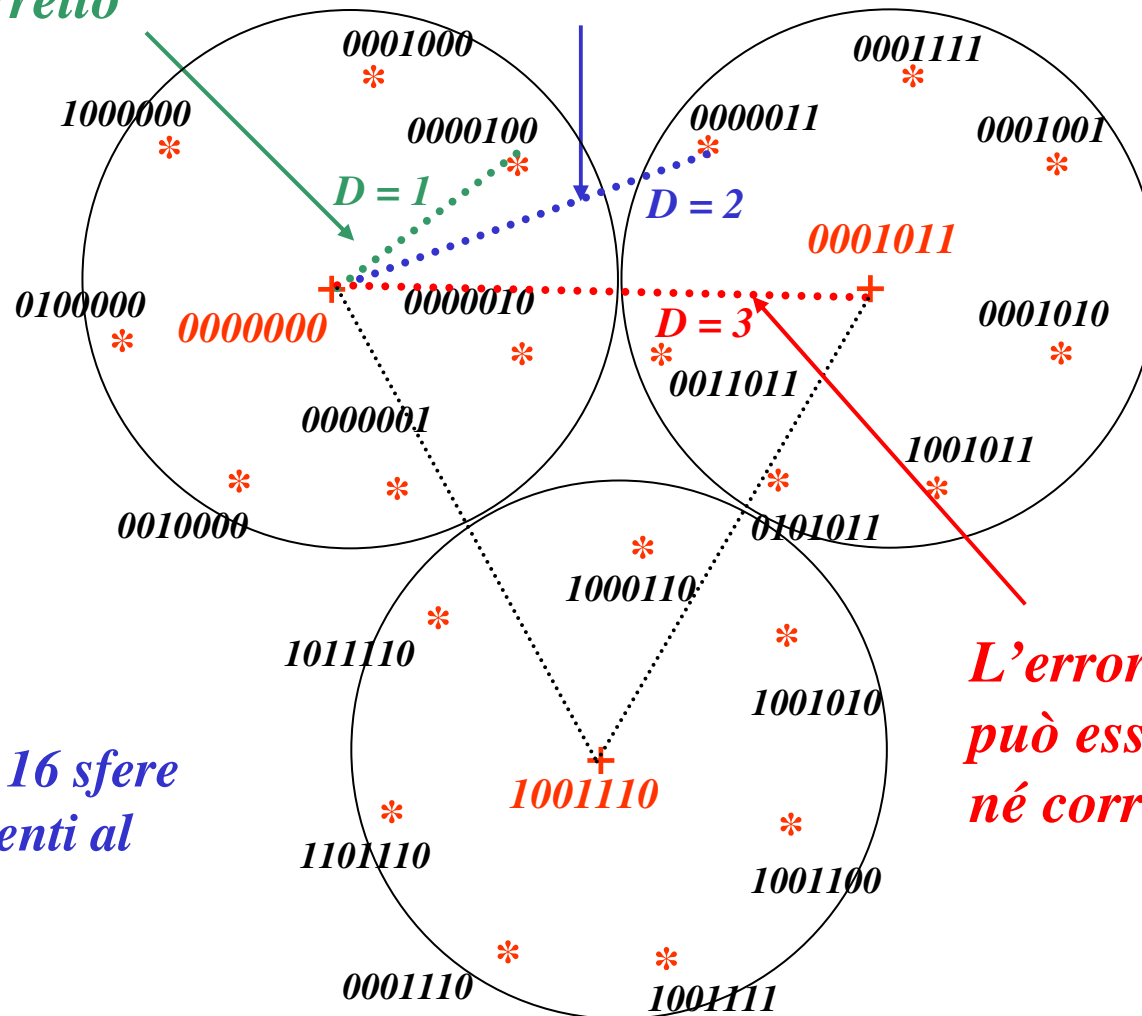
## *Codice di Hamming (7,4)*

- *Il codice di Hamming è perfetto, cioè non ci sono regioni intra-sfere*
- *Le parole informative sono costituite da 4 bit, vi sono  $2^4=16$  parole*
- *Ogni sfera è costituita da 8 parole, cioè dalla parola stessa più tutte le parole che distano 1 da essa*
- *Il numero totale di parole è  $8 \cdot 2^4=2^7$  parole, ovvero tutte le parole di 7 bit*
- *La distanza minima è tre*
- *Tutte le parole che non sono del codice sono a distanza uno da una parola di codice*
- *Il codice è in grado di correggere gli errori singoli*
- *Il codice è in grado di rivelare gli errori doppi*

# Codice di Hamming (7,4)

*L'errore singolo viene corretto*

*L'errore doppio può essere rivelato, ma non corretto*



*Tre delle 16 sfere appartenenti al codice*

*L'errore triplo non può essere né rivelato né corretto*



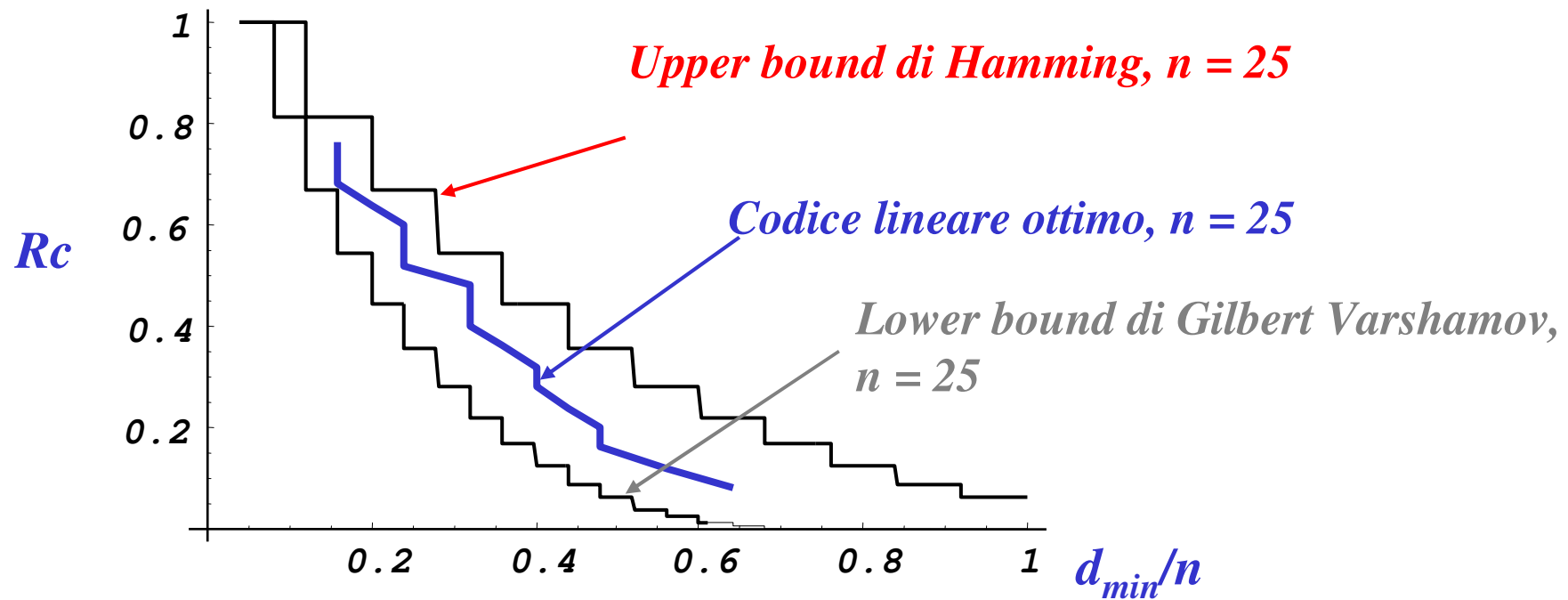
## *Codice di Hamming (7,4)*

$$\begin{array}{cccc} c_1 & c_2 & c_3 & c_4 \end{array} \times \begin{array}{cccc|cccc} 1 & 0 & 0 & 0 & 1 & 1 & 1 & \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & \end{array} = y_1 \ y_2 \ y_3 \ y_4 \ y_5 \ y_6 \ y_7$$

## *Prestazioni dei codici lineari*

*I codici lineari sono da confrontare con i bound generali già mostrati.*

*In generale, i codici corti hanno migliori probabilità di sorpassare i limiti di Gilbert Varshamov; si mostra il caso di  $n = 25$  e si notino le buone prestazioni.*



## *Probabilità di errore*

*La probabilità di errore in un codice a blocco di lunghezza  $n$  e capace di correggere  $t$  errori è data dalla seguente relazione:*

$$P_e = \sum_{i=t+1}^n \binom{n}{i} p^i (1-p)^{n-i}$$

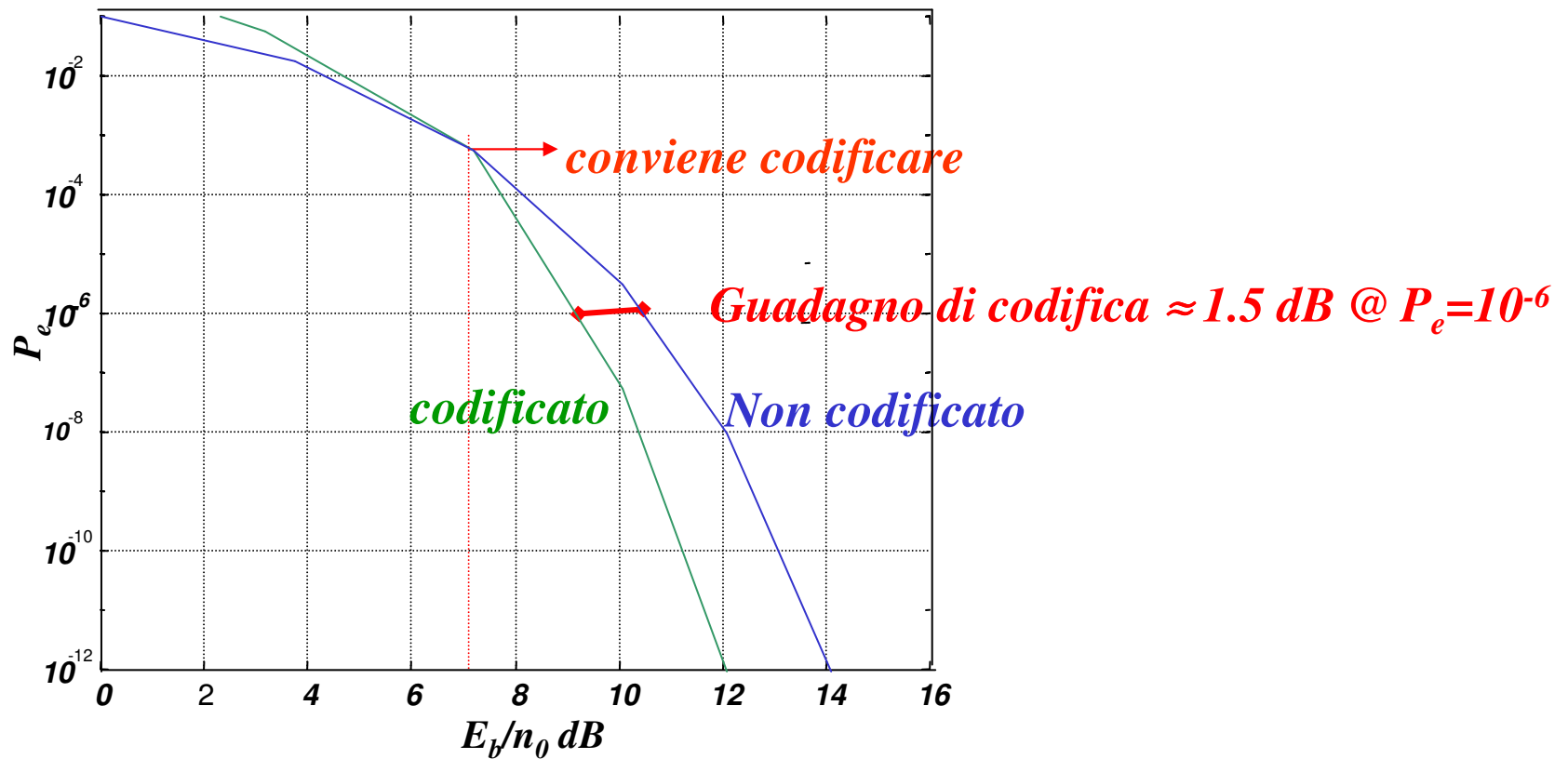
*che esprime la probabilità di commettere, nel blocco, un numero di errori maggiore di  $t$  e quindi non correggibili*

*$p$  è la probabilità di errore di bit nel canale prima della decodifica tale probabilità  $p$  è maggiore rispetto al caso non codificato, perché:*

$$\frac{E_b}{n_0} [\text{coded}] = \frac{E_b}{n_0} [\text{uncoded}] \cdot \frac{k}{n}$$

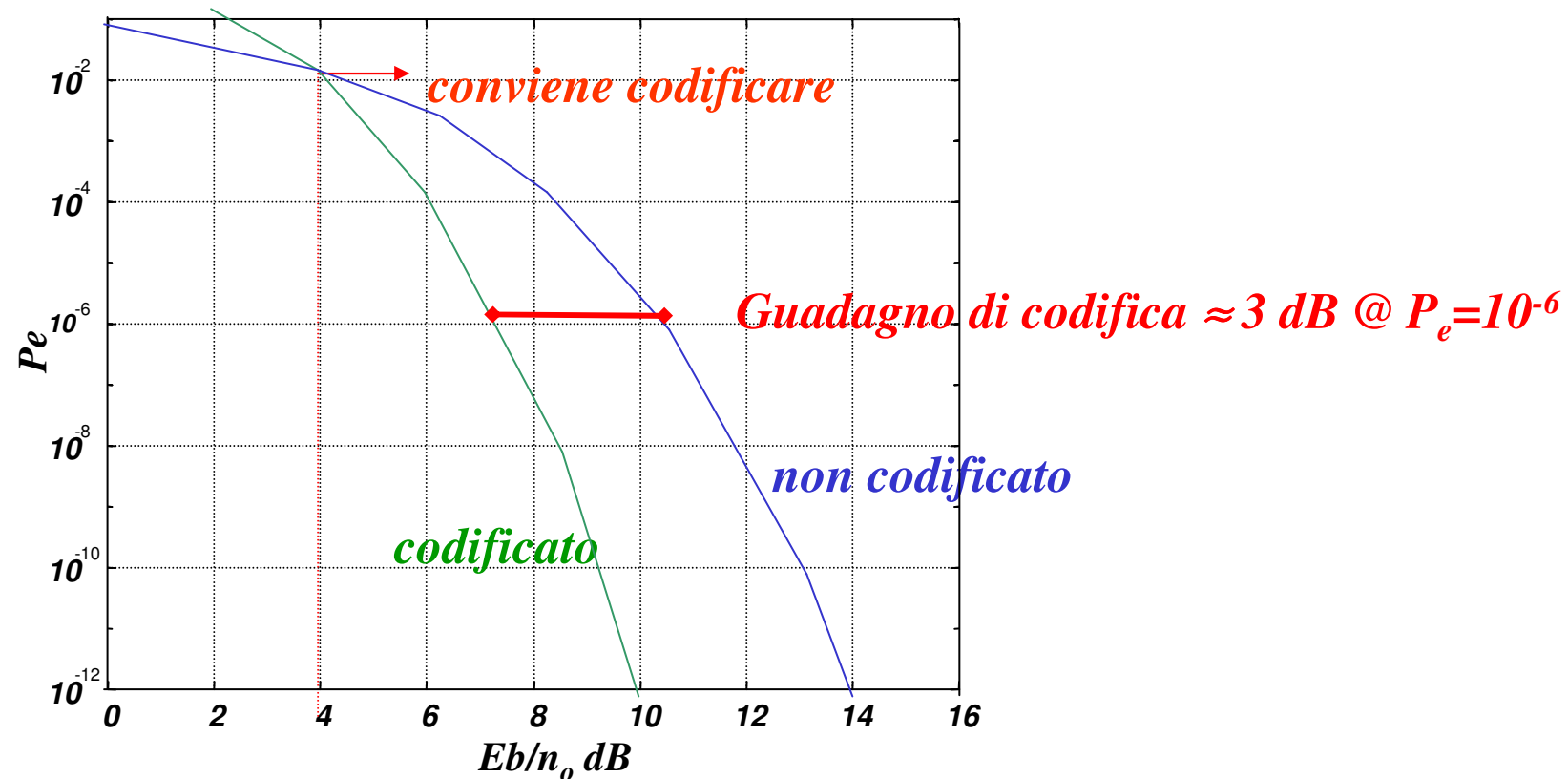
*La capacità di correzione deve superare tale, intrinseca, criticità*

## Probabilità di errore e guadagno di codifica: Hamming(7, 4, t = 1)



Si dice **guadagno di codifica** la diminuzione di  $E_b/n_0$  che il sistema codificato consente rispetto al sistema non codificato, scelta una certa  $P_e$

## Probabilità di errore e guadagno di codifica : BCH(31,16, t = 3)



Poiché il codice può correggere 3 errori, il punto di intercetta è più alto e il guadagno di codifica maggiore del precedente

## ***Codici lineari BCH Bose-Chaudhuri-Hocquenghem***

*Sono molto utilizzati, anche perché sono disponibili metodologie di codifica e decodifica algebrica potenti e relativamente semplici*

*Per ogni coppia di interi positivi  $m$  e  $t$  (numero di errori correggibili) esiste un codice BCH binario con parametri:*

$$n = 2^m - 1 \quad n - k \leq mt \quad d_{\min} \geq 2t + 1$$

*Sono eccellenti nel correggere errori distribuiti casualmente nel blocco, se  $n$  non è eccessivo, anzi sono essenzialmente i più potenti codici lineari, sino a valori moderati di  $n$*

*Codificatori e decodificatori sono strutturalmente analoghi a quello del codice di Hamming, salvo il maggior numero di sindromi; una per ogni errore correggibile*

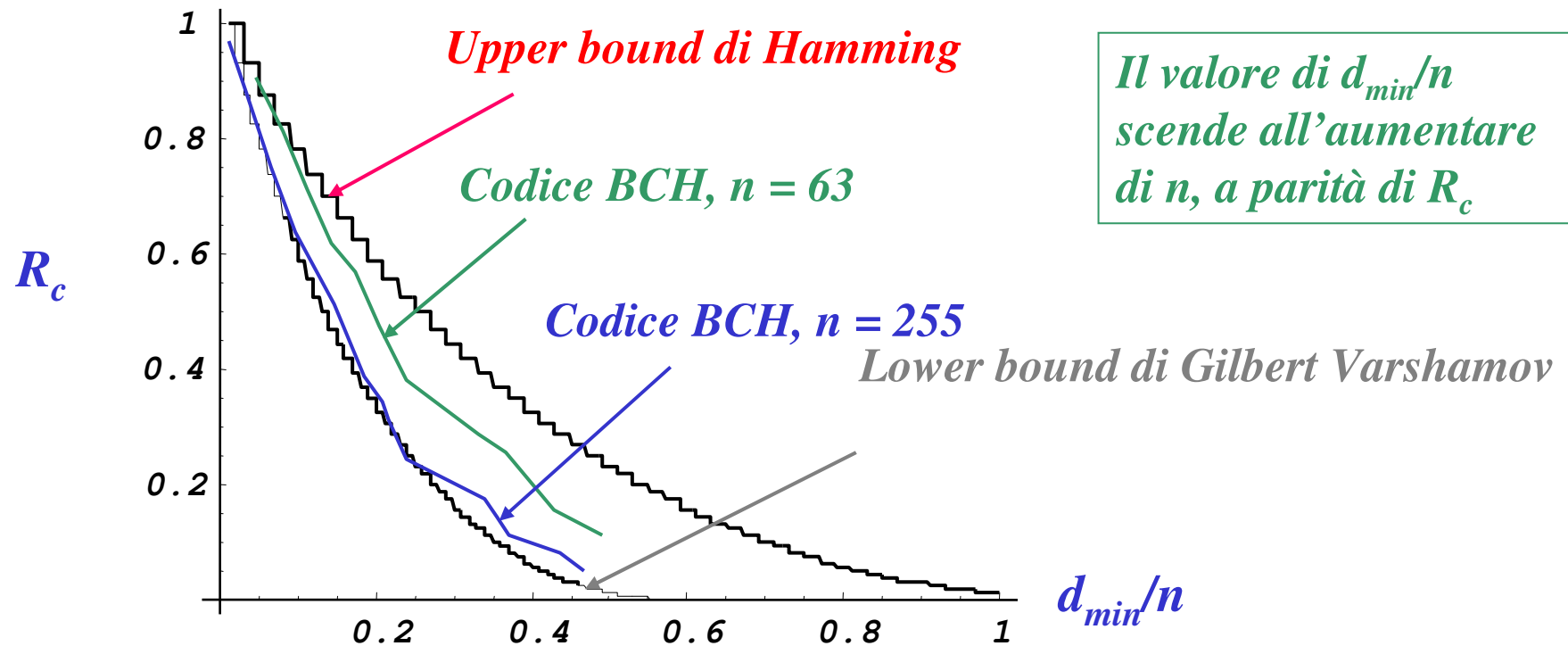
*La classe non binaria di tali codici, i codici di Reed Solomon, hanno rimarchevoli caratteristiche di ottimalità, ovvero  $d_{\min} = n - k + 1$ .*

## *Codici lineari BCH (15,7,5)*

$c_1$	$c_2$	$c_3$	$c_4$	$c_5$	$c_6$	$c_7$	×	$1\ 0\ 0\ 0\ 0\ 0\ 0\   1\ 0\ 0\ 0\ 1\ 0\ 1\ 1$	
								$0\ 1\ 0\ 0\ 0\ 0\ 0\   1\ 1\ 0\ 0\ 1\ 1\ 1\ 0$	
								$0\ 0\ 1\ 0\ 0\ 0\ 0\   0\ 1\ 1\ 0\ 0\ 1\ 1\ 1$	
								$0\ 0\ 0\ 1\ 0\ 0\ 0\   1\ 0\ 1\ 1\ 1\ 0\ 0\ 0$	=
								$0\ 0\ 0\ 0\ 1\ 0\ 0\   0\ 1\ 0\ 1\ 1\ 1\ 0\ 0$	
								$0\ 0\ 0\ 0\ 0\ 1\ 0\   0\ 0\ 1\ 0\ 1\ 1\ 1\ 0$	
								$0\ 0\ 0\ 0\ 0\ 0\ 1\   0\ 0\ 0\ 1\ 0\ 1\ 1\ 1$	
$y_1\ y_2\ y_3\ y_4\ y_5\ y_6\ y_7\ y_8\ y_9\ y_{10}\ y_{11}\ y_{12}\ y_{13}\ y_{14}\ y_{15}$									

## Prestazioni dei codici BCH

I codici BCH sono da confrontare con i bound generali già mostrati. Al solito, i codici corti hanno migliori probabilità di sorpassare i limiti di Gilbert Varshamov; per  $n > 1023$  i codici BCH scendono sotto tale limite.

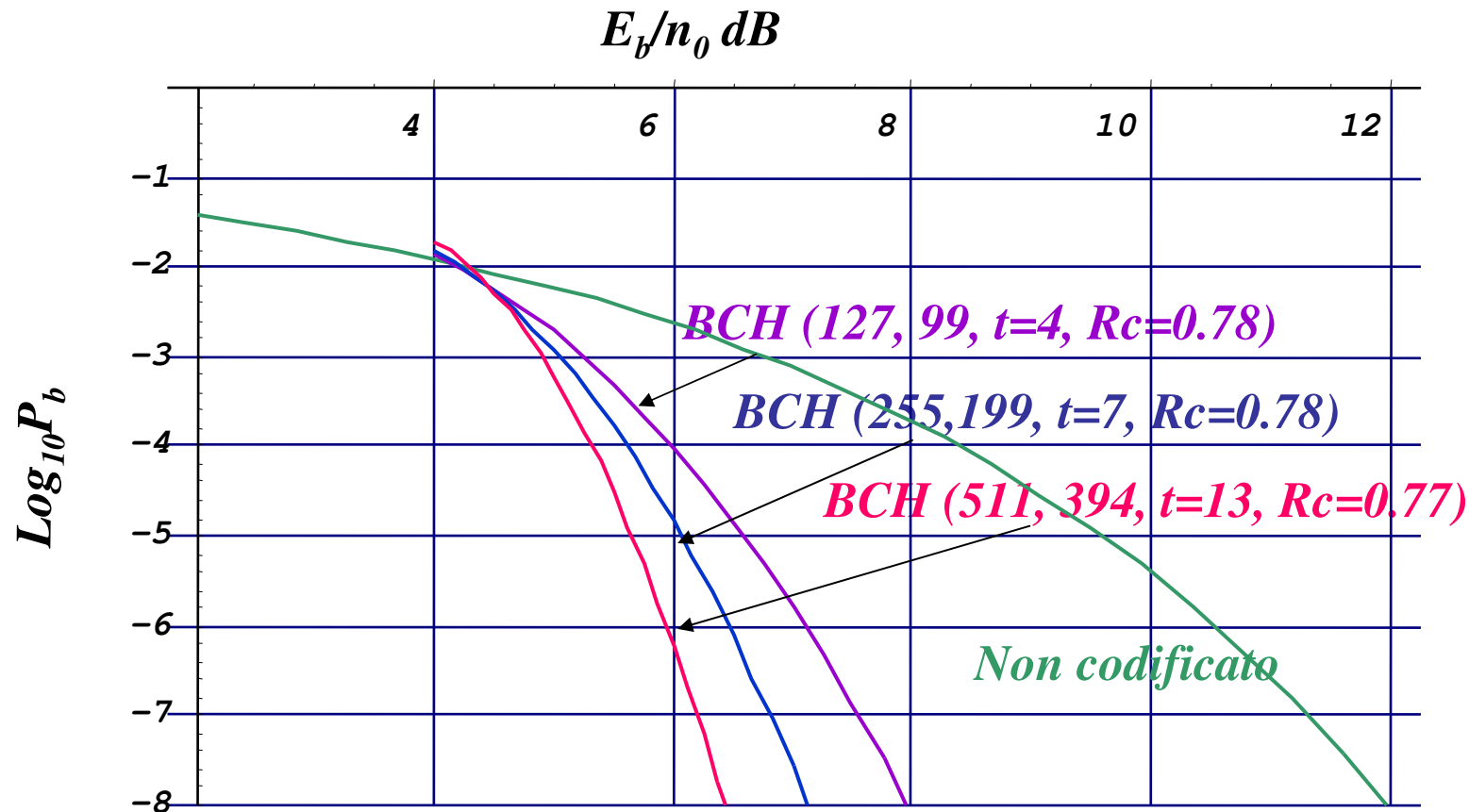




## *Prestazioni dei codici BCH*

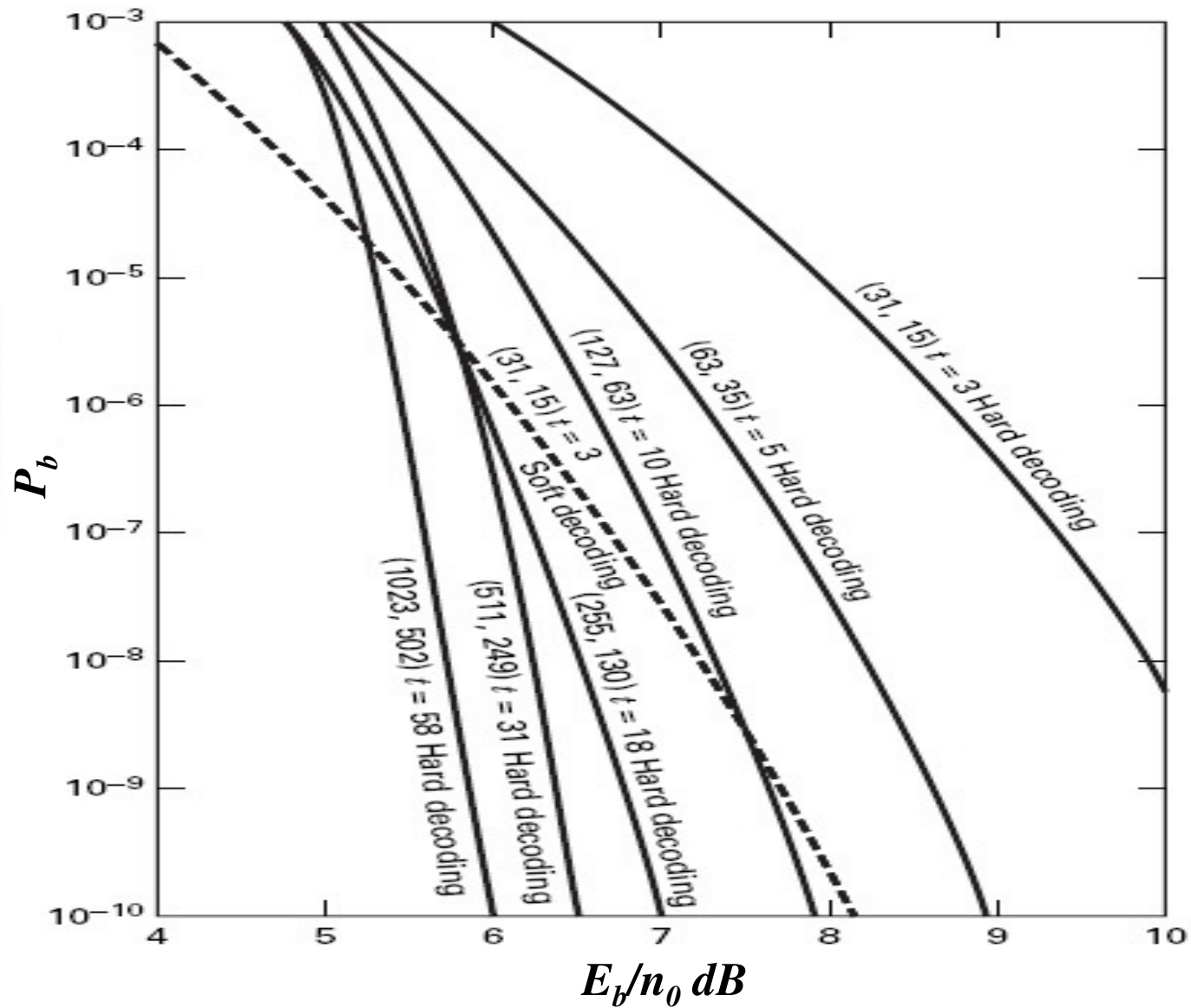
- *I codici BCH, quantunque eccellenti, non sono ottimi, soprattutto per valori molto grandi di  $n$ ;*
- *comunque, il rapporto  $d/n$  supera i limiti di Gilbert Varshamov per le lunghezze di pratico interesse;*
- *inoltre essi hanno algoritmi di decodifica ragionevolmente semplici e si costruiscono con valori generali di  $n$  e  $k$ .*

## Prestazioni dei codici BCH



*Il GSM impiega codici BCH per la rivelazione di errore per vari canali di segnalazione, nel canale di sincronizzazione e per l'accesso iniziale.*

## Prestazioni dei codici BCH



## *Prestazioni dei codici BCH*

- *Prestazioni quasi ottime con il vincolo del rate più grande possibile;*
- *il rate ottimo è quasi uguale per tutti i codici, indipendente da  $n$ ;*
- *il guadagno di codifica @  $BER = 10^{-6}$  è 3.5 - 4.5 dB, crescente con  $n$ ;*
- *diminuendo ancora il rate le prestazioni di fatto non aumentano, anzi, oltre certe diminuzioni, peggiorano, cosa già notata in generale;*
- *più il codice è lungo, più la curva di BER è ripida: ciò per la legge dei grandi numeri, per cui, per  $n$  grandi, se  $P_c$  è la probabilità di errore del bit codificato, il numero di errori è quasi esattamente  $n P_c$ , la soglia è allora  $t \geq n P_c$ .*

## *Prestazioni dei codici BCH*

*Codici BCH generati da elementi primitivi di ordine minore di  $2^{10}$*

<i>m</i>	<i>n</i>	<i>k</i>	<i>t</i>	<i>m</i>	<i>n</i>	<i>k</i>	<i>t</i>	<i>m</i>	<i>n</i>	<i>k</i>	<i>t</i>	<i>n</i>	<i>k</i>	<i>t</i>	<i>n</i>	<i>k</i>	<i>t</i>
3	7	4	1		63	24	7		127	50	13	255	187	9	255	71	29
4	15	11	1			18	10			43	14		179	10		63	30
		7	2			16	11			36	15		171	11		55	31
		5	3			10	13			29	21		163	12		47	42
5	31	26	1			7	15			22	23		155	13		45	43
		21	2	7	127	120	1			15	27		147	14		37	45
		16	3			113	2			8	31		139	15		29	47
		11	5			106	3	8	255	247	1		131	18		21	55
		6	7			99	4			239	2		123	19		13	59
6	63	57	1			92	5			231	3		115	21		9	63
		51	2			85	6			223	4		107	22	511	502	1
		45	3			78	7			215	5		99	23		493	2
		39	4			71	9			207	6		91	25		484	3
		36	5			64	10			199	7		87	26		475	4
		30	6			57	11			191	8		79	27		466	5

$n = 2^m - 1$       per  $t$  piccoli,  $n - k = mt$

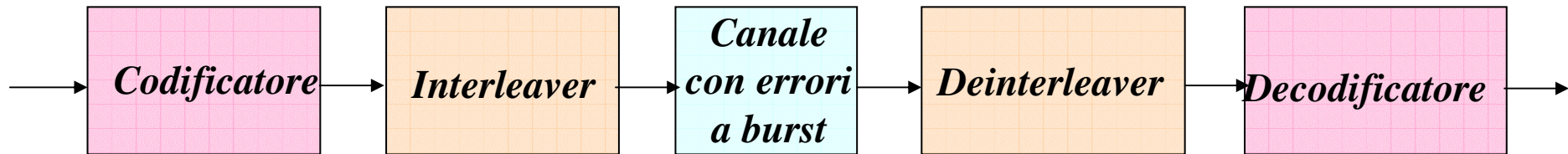
*generalità sui FEC*

## *Prestazioni dei codici BCH*

*Codici BCH generati da elementi primitivi di ordine minore di  $2^{10}$*

<i>n</i>	<i>k</i>	<i>t</i>	<i>n</i>	<i>k</i>	<i>t</i>	<i>n</i>	<i>k</i>	<i>t</i>	<i>n</i>	<i>k</i>	<i>t</i>	<i>n</i>	<i>k</i>	<i>t</i>
<i>511</i>	<i>457</i>	<i>6</i>	<i>511</i>	<i>322</i>	<i>22</i>	<i>511</i>	<i>193</i>	<i>43</i>	<i>511</i>	<i>58</i>	<i>91</i>	<i>1023</i>	<i>933</i>	<i>9</i>
	<i>448</i>	<i>7</i>		<i>313</i>	<i>23</i>		<i>184</i>	<i>45</i>		<i>49</i>	<i>93</i>		<i>923</i>	<i>10</i>
	<i>439</i>	<i>8</i>		<i>304</i>	<i>25</i>		<i>175</i>	<i>46</i>		<i>40</i>	<i>95</i>		<i>913</i>	<i>11</i>
	<i>430</i>	<i>9</i>		<i>295</i>	<i>26</i>		<i>166</i>	<i>47</i>		<i>31</i>	<i>109</i>		<i>903</i>	<i>12</i>
	<i>421</i>	<i>10</i>		<i>286</i>	<i>27</i>		<i>157</i>	<i>51</i>		<i>28</i>	<i>111</i>		<i>893</i>	<i>13</i>
	<i>412</i>	<i>11</i>		<i>277</i>	<i>28</i>		<i>148</i>	<i>53</i>		<i>19</i>	<i>119</i>		<i>883</i>	<i>14</i>
	<i>403</i>	<i>12</i>		<i>268</i>	<i>29</i>		<i>139</i>	<i>54</i>		<i>10</i>	<i>121</i>		<i>873</i>	<i>15</i>
	<i>394</i>	<i>13</i>		<i>259</i>	<i>30</i>		<i>130</i>	<i>55</i>		<i>1013</i>	<i>1</i>		<i>863</i>	<i>16</i>
	<i>385</i>	<i>14</i>		<i>250</i>	<i>31</i>		<i>121</i>	<i>58</i>	<i>1023</i>	<i>1003</i>	<i>2</i>		<i>858</i>	<i>17</i>
	<i>376</i>	<i>15</i>		<i>241</i>	<i>36</i>		<i>112</i>	<i>59</i>		<i>993</i>	<i>3</i>			
	<i>367</i>	<i>16</i>		<i>238</i>	<i>37</i>		<i>103</i>	<i>61</i>		<i>983</i>	<i>4</i>			
	<i>358</i>	<i>18</i>		<i>229</i>	<i>38</i>		<i>94</i>	<i>62</i>		<i>973</i>	<i>5</i>			
	<i>349</i>	<i>19</i>		<i>220</i>	<i>39</i>		<i>85</i>	<i>63</i>		<i>963</i>	<i>6</i>			
	<i>340</i>	<i>20</i>		<i>211</i>	<i>41</i>		<i>76</i>	<i>85</i>		<i>953</i>	<i>7</i>			
	<i>331</i>	<i>21</i>		<i>202</i>	<i>42</i>		<i>67</i>	<i>87</i>		<i>943</i>	<i>8</i>			

## *Interleaving*



- *E' una tecnica utilizzata con i canali con errori a burst;*
- *I codici a blocco permettono di correggere un dato numero di errori per blocco, e sono normalmente progettati per errori uniformemente distribuiti;*
- *con errori a burst, si supera la capacità correttiva del codice in certi blocchi e le prestazioni peggiorano;*
- *d'altronde i codici per errori a burst sono poco usati: il canale non è sufficientemente noto ed è difficile la soft decision.*

## *Interleaving*



➤ *L' interleaver riarrangia e rimescola l'ordine di una sequenza di simboli in modo deterministico;*

➤ *il deinterleaver applica l'operazione inversa per recuperare l'ordine originale della sequenza; il burst di errore può, allora, essere disperso nella sequenza e non più concentrato su pochi blocchi;*

➤ *paradossalmente, l'interleaver abbassa la capacità del canale, poiché a parità di numero di errori la situazione peggiore è loro distribuzione senza memoria; d'altronde sfruttare tale correlazione è praticamente inattuabile, dovendo conoscere con esattezza lo stato del canale dinamicamente.*



# *Codici convoluzionali*

## *Codici convoluzionali*

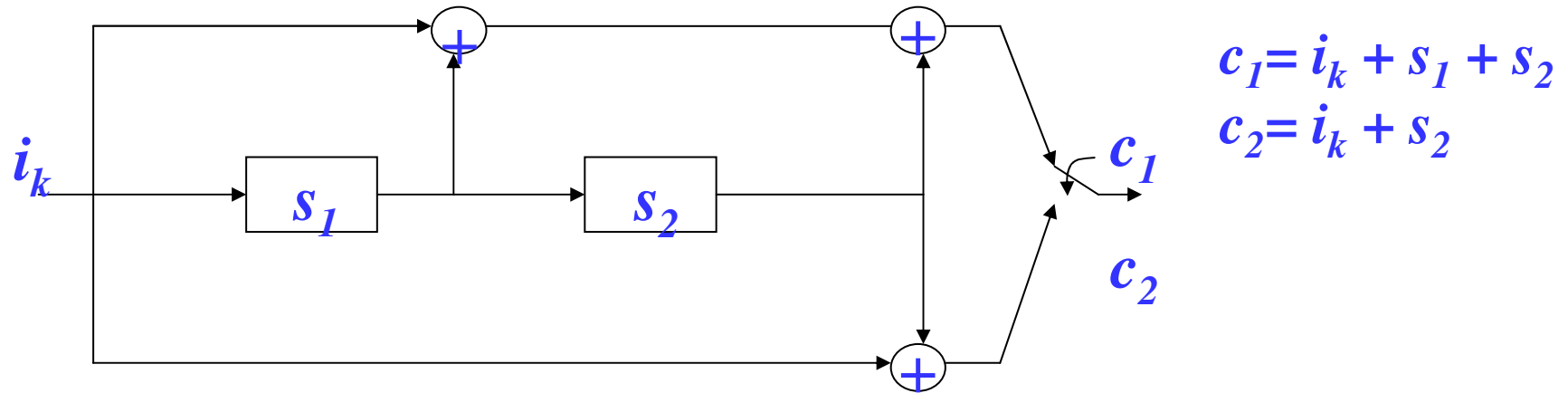
*I codici convoluzionali  $(n,k)$  sono molto impiegati nei sistemi di comunicazione: varie tecniche di decodifica consentono di adattare complessità e prestazioni alle reali necessità; si possono decodificare soft a massima verosimiglianza, con complessità computazionali ragionevoli (**guadagnando dB praticamente gratis**), ciò che non si può fare i codici a blocco; è semplice sincronizzazione i decodificatori.*

*Tali codici non richiedono la suddivisione della sequenza dell'informazione in blocchi*

*Sono generati facendo passare la sequenza dei bit di informazione da trasmettere attraverso un registro a scorrimento; Le uscite dei registri sono sommate opportunamente generando i bit di uscita*

*Il contenuto dello shift register costituisce la memoria del codificatore e ne definisce lo stato; la **constrain length L** è, normalmente, il numero di stadi di tale shift-register, ciascuno di  $k$  bit, più 1.*

## Codice convoluzionale $R_c = 1/2, L = 3$



*Sequenza da codificare: 1 1 0 0*

*Stato iniziale: 0 0*

*Sequenza codificata: 11010111*

*Lo stato è la memoria del sistema  $s_1 s_2$*

$i_k$	$s_1 s_2$	$c_1 c_2$
1	10	11
1	11	01
0	01	01
0	00	11

## *La distanza di Hamming*

- *Le prestazioni del codice dipendono dalla distanza di Hamming tra le parole di codice;*
- *per la linearità del codice, confrontare la distanza fra le parole equivale a confrontare la distanza di tutte le parole con la parola nulla;*
- *il confronto va fatto fra tutti i cammini ed il cammino 0000... ;*
- *la distanza minima è la minima distanza tra le parole di codice.*

## *La distanza nei codici convoluzionali*

*Si confronta la trasmissione della sequenza 0 0 0 ... che viene ancora codificata come 00 00 00 ... con la sequenza 1 0 0 ...*

*Sequenza da codificare: 1 0 0 0*

	<i>Bit in ingresso</i>	<i>bit in memoria</i>	<i>uscite</i>	
	$i_k$	$s_1 s_2$	$u_1 = i + s_1 + s_2$	$u_2 = i + s_2$
<i>La memoria ha</i>	<i>1</i>	<i>1 0</i>	<i>1</i>	<i>1</i>
<i>stato iniziale nullo</i>	<i>0</i>	<i>0 1</i>	<i>1</i>	<i>0</i>
	<i>0</i>	<i>0 0</i>	<i>1</i>	<i>1</i>

*Sequenza in uscita: 11 10 11 00*

*La distanza fra le due sequenze trasmesse è 5, una volta che convergono nello stesso stato 0 0. Siccome l'energia per bit rispetto al non codificato è 1/2, l'incremento reale dato dal codice è di circa 4 dB = 10 log<sub>10</sub> (5/2)*

## Guadagno di codifica

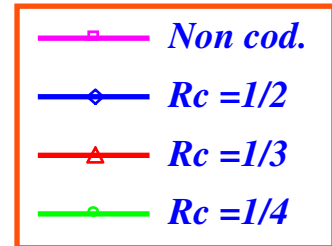
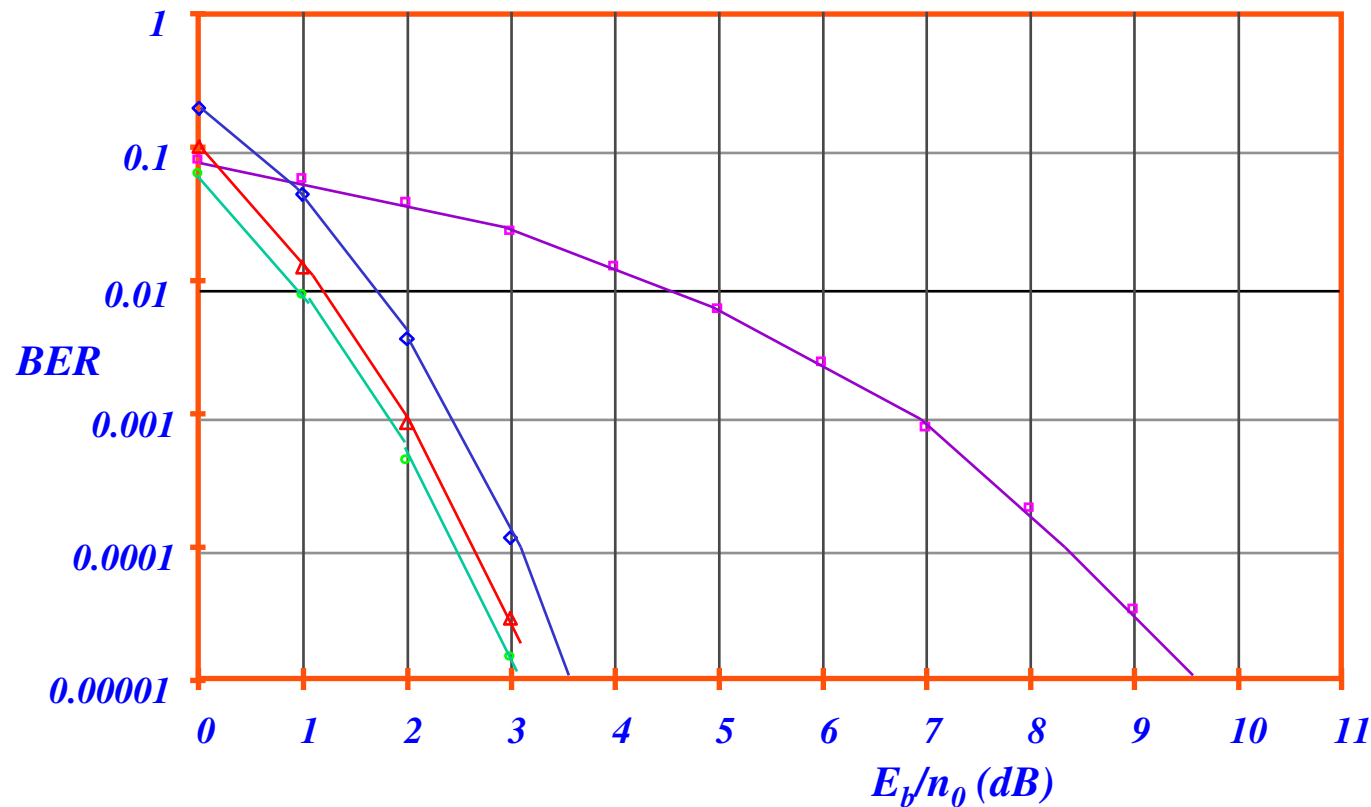
*Guadagni di codifica asintotici G per alcuni codici convoluzionali.*

Rate 1/2			Rate 1/3			<i>Notare l'importanza decisiva della constrain length L; a parità di rate di codifica, il suo aumento determina incrementi di prestazioni. Il prezzo è un notevole incremento (esponenziale con L) della complessità computazionale</i>
<i>L</i>	<i>d<sub>min</sub></i>	<i>G(dB)</i>	<i>L</i>	<i>d<sub>min</sub></i>	<i>G(dB)</i>	
3	5	3.97	3	8	4.26	
4	6	4.76	4	10	5.23	
5	7	5.43	5	12	6.02	
6	8	6.00	6	13	6.37	
7	10	6.99	7	15	6.99	
8	10	6.99	8	16	7.27	
9	12	7.78	9	18	7.78	

$$G(dB) = 10 \log_{10} (d_{min} \cdot R_c)$$

## *I codici convoluzionali nell'UMTS*

- *In UMTS i codici convoluzionali sono usati per traffico real time*
- *I codificatori hanno rate 1/2 e 1/3 e constraint length uguale a 9*



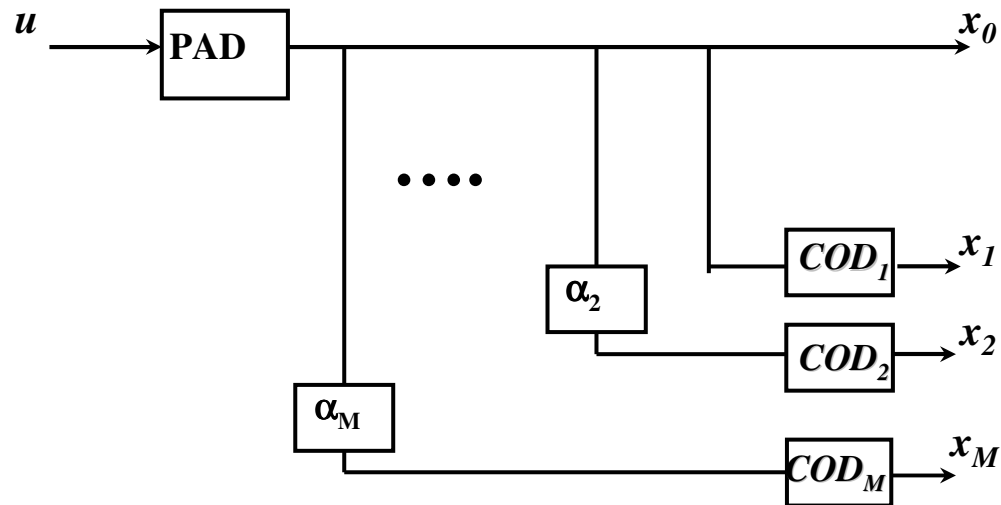
*Da notare il (non forte) incremento delle prestazioni al diminuire di  $R_c$ , in accordo con l'andamento del cutoff rate.*

***constraint length = 9***

# *I turbo codici*



## Turbo codici: schema generale del codificatore



- I turbo codici sono codici concatenati, di tipo seriale o parallelo, con prestazioni eccellenti, ma a prezzo di notevoli complessità computazionali;
- lo schema mostra il codificatore di un turbo codice costituito dalla concatenazione di due o più codici ricorsivi in parallelo;
- il blocco PAD pone in coda alla parola da codificare alcuni bit di “coda” per riportare il codificatore nello stato iniziale a fine codifica;
- gli  $\alpha_i$  sono interleaver, che mescolano i bit in modo pseudocasuale.

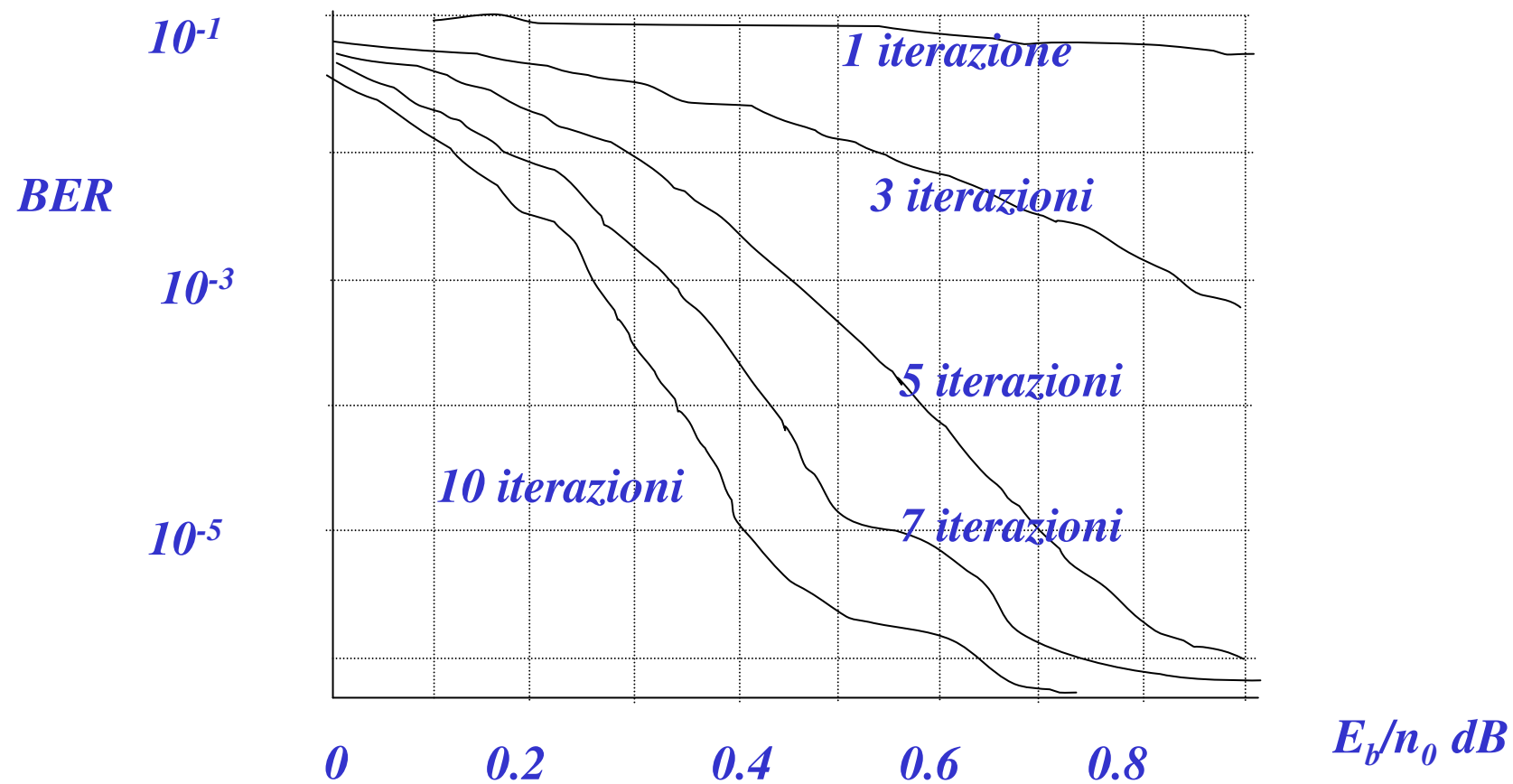
## *I turbo codici*

- *I turbo codici si avvicinano ai limiti di Shannon perché c'è l'interleaver;*
- *un codice sbaglia perché la distanza tra parole codificate è limitata;*
- *i codici costituenti del turbo codice, considerati separatamente, non hanno distanze elevate;*
- *l'interleaver, a N posizioni, rende improbabile che le uscite dei codificatori abbiano tutte bassa distanza rispetto alla parola nulla;*
- *esempio: si supponga che 10100... sia associata a una parola codificata a distanza piccola rispetto alla parola nulla, e che questa sia, quindi, una parola critica per il codice in esame. Supponiamo, inoltre, senza perdita di generalità, che i due codici costituenti siano uguali. La probabilità che questa parola o sue versioni shiftate capitino anche sul secondo codice è  $1/N$ . Globalmente, la  $P_e$  dovuta a quella parola si riduce allora di  $\approx 1/N$ .*

## *Turbo codici: prestazioni*

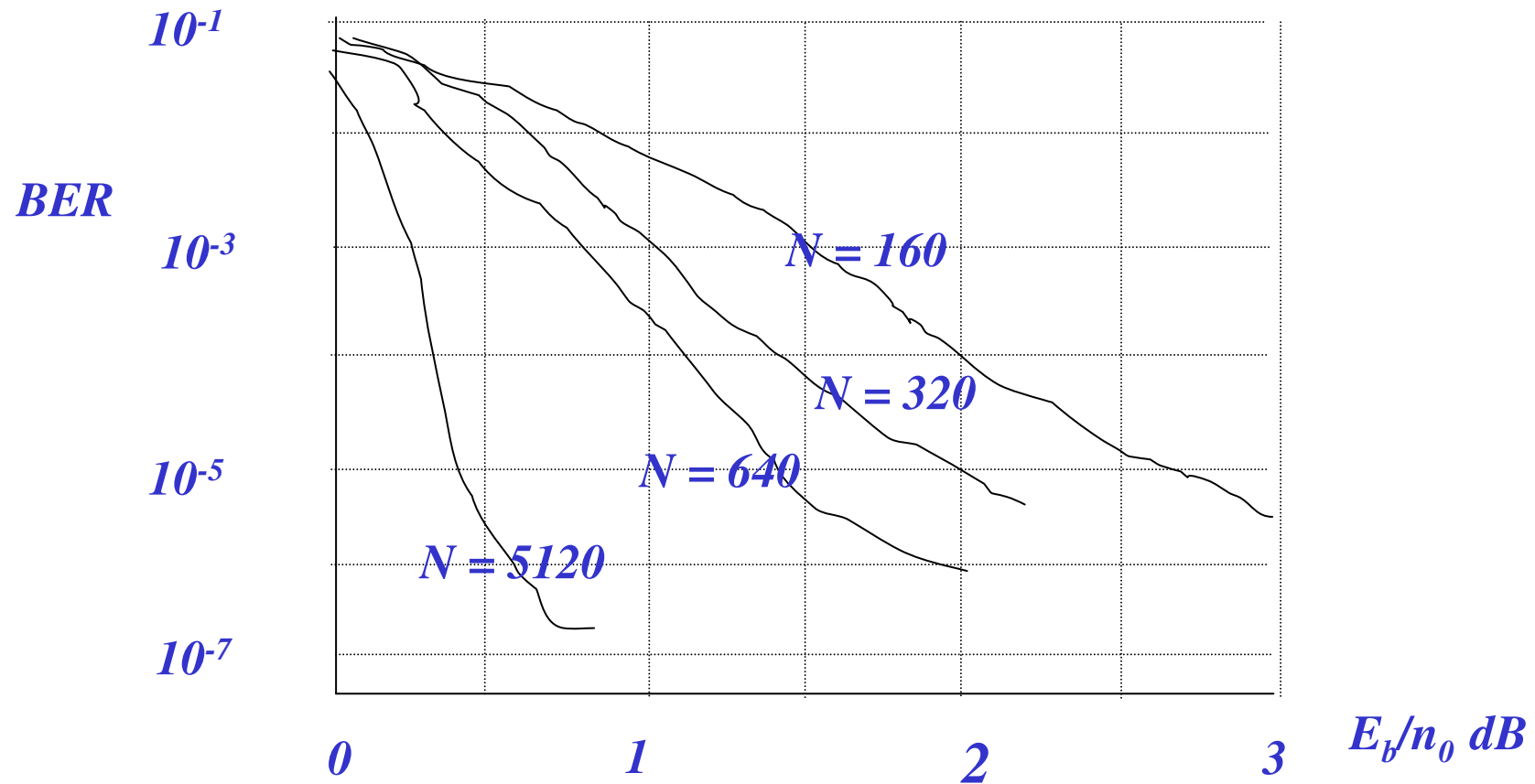
- *I turbo codici vanno molto bene quando  $E_b/n_0$  è molto basso;*
- *a causa della ridotta complessità computazionale dei codici costituenti la distanza tra le parole di codice non è, singolarmente, elevata;*
- *quando  $E_b/n_0$  è elevato i turbo codici presentano un floor nelle prestazioni*

## *BER vs. Eb/No al variare del numero di iterazioni*



$G=(15,17)_{octab}$  Rate=1/3, N=5120

## *BER vs. $E_b/N_0$ al variare della lunghezza di trama $N$*



$G=(15,17)_{octal}$  Rate=1/3, 10 iterazioni

## *Il Turbo Decoder*

- *Con riferimento a due codici costituenti, in decodifica vi sono due decodificatori concatenati serialmente, ognuno dei quali è associato ad uno dei codici costituenti il codificatore*
- *Per ogni bit decodificato il primo decodificatore fornisce al secondo anche l'affidabilità (stima della probabilità a posteriori) della decodifica; questo parametro è utilizzato dal secondo codificatore per effettuare la propria stima e ricalcolare la corrispondente affidabilità su tale bit; questa procedura si ripete successivamente tra i due decodificatori.*
- *La retroazione presente tra il primo ed il secondo decodificatore dà il nome a questo tipo di codici. Infatti tale retroazione ricorda molto ciò che avviene nel motore di una automobile.*
- *L'affidabilità della decodifica è una funzione crescente con il numero di iterazioni: però, anche il ritardo è crescente con il numero di iterazioni.*

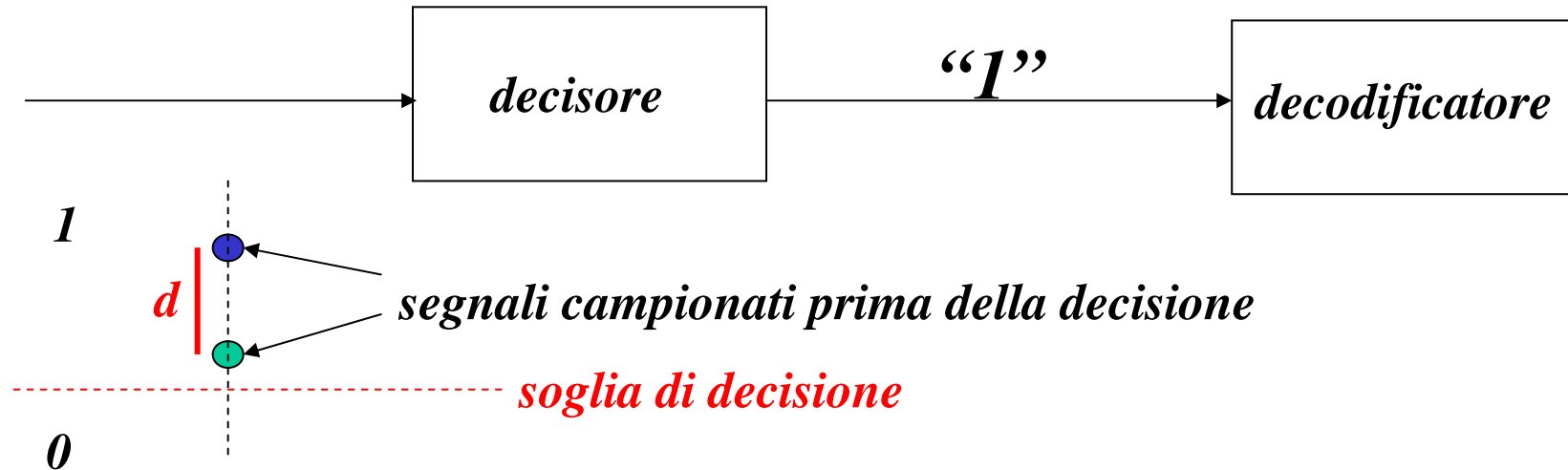
# *Decodifica: hard e soft*

## *Decodifica hard*

- *La decisione avviene bit a bit (simbolo a simbolo)*
- *Il decisore confronta l'energia del segnale ricevuto e il valore del rumore associato, filtrato dal filtro di ricezione, nel tempo di bit (simbolo), con una soglia e prende una decisione*



## Decodifica hard



- *La decisione dopo il demodulatore si basa sul confronto con una soglia e all'ingresso del decodificatore si inviano i simboli decisi;*
- *nell'esempio, la decisione è la stessa per entrambi i segnali;*
- *nell'effettuare tale decisione si è persa un'informazione importante, cioè la distanza (metrica) rispetto alla soglia;*
- *tale distanza è legata all'energia effettiva dei segnali.*

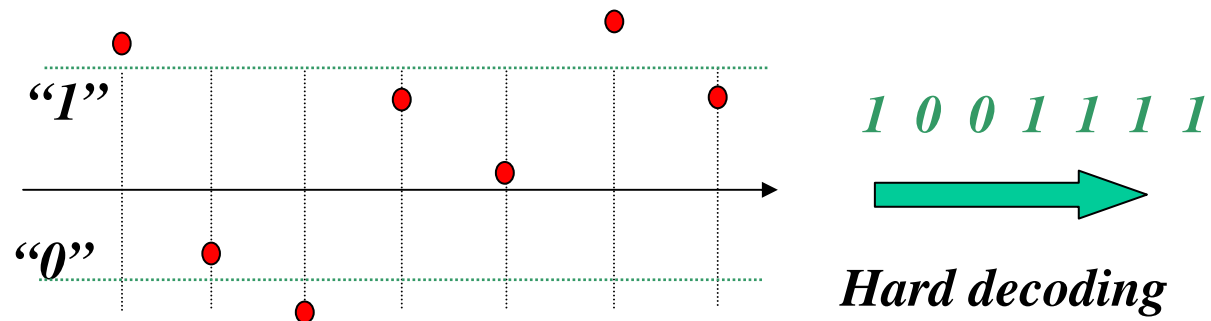
## *Decodifica soft*

- *La decisione non avviene bit a bit;*
- *dopo  $n$  bit si ha una metrica globale (tipicamente associata alla sequenza di uscite dal filtro di ricezione - filtro adattato), che si confronta con la metrica ideale (corrispondente alla struttura energetica dei segnale trasmissibili) associata alle parole di codice, in assenza di rumore;*
- *in altri termini, con la decodifica soft si decide dopo il decodificatore e si tiene conto della differenza energetica globale dei possibili segnali ricevuti;*
- *ciò dà un guadagno di codifica, rispetto alla decodifica hard, di circa 2 dB, a parità di altre condizioni.*

## *Decodifica hard e soft: un esempio (Wagner decoding)*

- *Si fa riferimento ad un codice di parità(7,6), che ad ogni blocco di 6 bit aggiunge un bit in modo che il numero dei bit 1 sia pari;*
- *tale codice sia impiegato ove un errore su 7 bit è in effetti abbastanza raro, altrimenti si dovrebbe procedere troppo spesso alla ritrasmissione, e, quindi, un doppio errore su 7 bit è veramente ancora più raro;*
- *si supponga allora di trasmettere la parola 1 0 0 1 0 1 1;*
- *supponga, altresì, che i segnali in decisione, dopo il filtro di ricezione (filtro adattato) siano i seguenti*

## *Decodifica hard e soft: un esempio (Wagner decoding)*

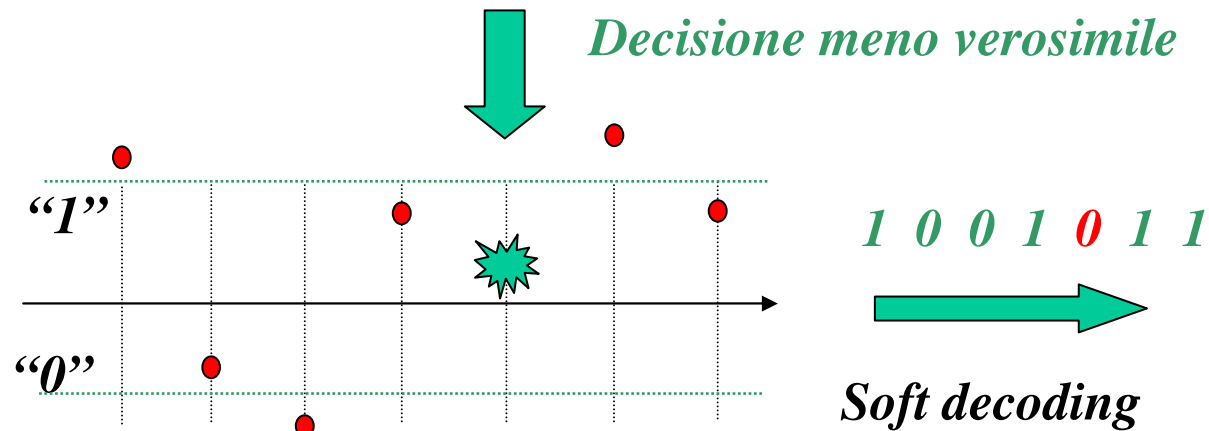


*Con la decodifica hard, la parola in uscita dal decisore a soglia contiene un errore e quindi tale errore è rilevabile, ma non è correggibile*

*Segnali negli istanti di campionamento ai capi del filtro di ricezione (filtro adattato), che presenta in uscita l'energia dei segnali utili, più il rumore filtrato.*

## *Decodifica hard e soft: un esempio (Wagner decoding)*

- *Poiché la parità non è rispettata, il decodificatore, che sa che un doppio errore è molto improbabile, decide di considerare errato il bit cui corrisponde la peggiore situazione in decisione, cioè che ha la più grande distanza energetica rispetto ai livelli in assenza di rumore*
- *Con ciò si rimuove facilmente l'errore*



*Segnali negli istanti di campionamento ai capi del filtro di ricezione (filtro adattato), che presenta in uscita l'energia dei segnali utili, più il rumore filtrato.*

*Con la decodifica soft, cambiando la decisione per il bit meno verosimile, il corrispondente errore è rilevabile ed è **correggibile**, almeno sinché sono veramente rari gli errori doppi*

## *Decodifica hard e soft: confronto di prestazioni*

- Il confronto di prestazioni tra decodifica hard e soft deve esprimersi in termini di capacità e di cutoff rate  $R_0$ .*
- Si mostra, di seguito, il confronto tra canale gaussiano ideale e il canale binario banda base (BB), come capacità teorica e  $R_0$ . Il divario di  $\approx 2$  dB tra decodifica soft e hard è evidente, ed è assolutamente generalizzabile, in teoria e in pratica.*

## Decodifica hard e soft: confronto di prestazioni

