

## **Introduzione**

Il modo migliore di vedere un codice correttore consiste nel considerare un canale binario simmetrico che commette errori con probabilità maggiore di quella accettabile. Se il canale è stazionario e senza memoria, le posizioni degli errori sono casuali e le prestazioni dei codici sono molto più facili da analizzare. Canali più complessi, con memoria o con rapporto segnale rumore variabile nel tempo, tendono a produrre errori a pacchetto (o *burst*). In tal caso sia la scelta del codice sia la valutazione delle prestazioni sono più difficili.

Le strategie nell'uso dei codici da usare nel caso di canali con ingresso ed uscita discreti sono:

- correzione degli errori, quando il loro numero non superi la capacità di correzione del codice; si deve accettare il rischio, di cui si calcolerà la probabilità, di non riconoscere errori commessi dal canale o di correggere in modo errato, introducendo ulteriori errori;
- rivelazione degli errori e richiesta di una nuova trasmissione, purché esista un canale di ritorno ed i ritardi introdotti siano accettabili; anche in questo caso possono sfuggire blocchi errati, usualmente con probabilità molto minore rispetto al caso di correzione (usando gli stessi codici);
- strategie miste di correzione parziale, limitata ad un numero di errori minori di quelli correggibili dal codice, e di rivelazione degli errori nei restanti casi; la probabilità di avere un blocco errato in uscita dal decodificatore è intermedia tra quella dei casi precedenti.

Restando nel campo discreto, si possono avere codici binari e non binari.

I tipi di codici più comuni sono così classificati:

- codici a blocco, nei quali  $K$  cifre d'informazione sono seguite da  $N - K$  cifre di parità, calcolate in modo deterministico dalle  $K$  cifre d'informazione;
- codici convoluzionali, in cui una cifra d'informazione è seguita, ad esempio, da una di parità alternativamente; a sua volta ogni cifra di parità dipende da un numero non grande di cifre d'informazione precedenti;
- codici composti: codici concatenati, codici prodotto, turbo-codici.

Noi considereremo, per semplicità, solo i codici a blocco: dato un blocco di  $K$  cifre d'informazione, le regole del codice determinano la  $N$ -pla di cifre da inviare sul canale ( $N > K$ ). Il codice è

generalmente indicato come codice a blocco  $(N,K)$ . Il rapporto  $\frac{K}{N}$  è il rate del codice.

**Codici lineari**

Per ridurre la complessità della codifica e soprattutto della decodifica, sono utilizzati i codici lineari, le cui cifre trasmesse sono combinazioni lineari delle cifre d'informazione.

Il codice è detto *sistematico* se le prime  $K$  cifre del blocco corrispondono alle cifre d'informazione, mentre le restanti  $N - K$  cifre sono dette cifre di parità. E' ovvio che non ha alcuna importanza se le cifre di parità sono all'inizio del blocco o alla fine. Generalmente, in ogni modo, seguono le cifre d'informazione.

Su un canale di trasmissione binario, la capacità di correzione e di rivelazione degli errori dipendono dalla distanza minima di Hamming tra le parole di codice. Nei codici (naturale, BCD, ASCII, Gray, etc.) in cui la distanza minima è uno, non è possibile rivelare ed eventualmente correggere alcun errore.

Pertanto, vengono aggiunti dei bit di ridondanza, fermo restando il numero di combinazioni e quindi di simboli.

Il caso più semplice è l'aggiunta del bit di parità dispari PO (Parity Odd) o di parità pari PE (Parity even).

Nel caso di PO viene aggiunto un bit fra 0 e 1 in modo da avere un numero dispari di uni nel blocco. Nel caso di PE, al contrario, il numero di uni totali deve essere pari.

In ogni caso la distanza minima di Hamming vale 2 ed è, quindi, possibile rivelare un solo errore ma non correggerlo.

Se, per esempio, si devono trasmettere  $K = 7$  cifre d'informazione (1001011), ad esse viene aggiunto un 1 per la PO (10010111) ed uno 0 per la PE (10010110). In ricezione, sapendo il tipo di parità usato, si controlla il numero degli 1 e si rivela solo la presenza di un errore.

Per riuscire a correggere l'errore è necessario utilizzare il sistema di controllo della parità incrociata. In altre parole, per ogni singola sequenza binaria relativa ad un dato si determina il bit di parità e si aggiungono il bit di start all'inizio e di stop alla fine. Si calcola, inoltre, il controllo per ogni posizione binaria dei dati, facendone la parità. Tale controllo viene trasmesso alla fine dei dati. In ricezione si eliminano i bit di start e di stop e si calcolano la parità dei bit ricevuti ed il nuovo controllo. Si fa il confronto fra le parità ricevute e quelle calcolate. In questo modo, è possibile determinare per incrocio il bit errato, ed essendo binario basta negarlo.

Supponiamo di dover trasmettere due caratteri ASCII a 7 bit con parità pari. Vengono trasmessi due dati più il controllo così organizzati:

**Codici correttori**

Dato 1	Dato 2	Controllo
--------	--------	-----------

I bit ricevuti sono descritti dalla seguente sequenza (10 bit per dato più 10 bit per il controllo):

010011100101110001110001111101

e sono strutturati nel seguente modo:

start – D<sub>0</sub> D<sub>1</sub> D<sub>2</sub> D<sub>3</sub> D<sub>4</sub> D<sub>5</sub> D<sub>6</sub> – E – Stop

Costruiamoci la tabella dei dati ricevuti:

	Start	D <sub>0</sub>	D <sub>1</sub>	D <sub>2</sub>	D <sub>3</sub>	D <sub>4</sub>	D <sub>5</sub>	D <sub>6</sub>	Parità	Stop
Dato 1	0	1	0	0	1	1	1	0	0	1
Dato 2	0	1	1	1	0	0	0	1	1	1
Controllo	0	0	0	1	1	1	1	1	1	1

Eliminando i bit di start e di stop e calcolando la parità dei bit ricevuti ed il nuovo controllo, si ottiene una nuova tabella. Dal confronto tra la parità ricevuta e quella calcolata dal ricevitore si trova che l'errore è nel Dato 2; dal confronto tra il controllo ricevuto ed il controllo calcolato si trova, invece, che l'errore è relativo al bit D<sub>1</sub>. Pertanto, dall'incrocio tra la colonna D<sub>1</sub> e la riga Dato 2 (in rosso) si può individuare e correggere l'errore.

Se gli errori crescono non sono più rivelabili. In questo caso è necessario crescere la distanza minima di Hamming introducendo ulteriori bit di ridondanza.

Se ad esempio la distanza minima di Hamming è  $d = 3$ , il codice può essere usato come correttore di un errore (in posizione qualsiasi della N-pla). Lo stesso codice può essere anche usato per rivelare due errori. Vengono anche rivelate alcune configurazioni di tre o più errori, ma non tutte.

Analogamente, se  $d = 4$ , il codice può essere usato come correttore di un errore (con due errori si può finire a metà strada fra due parole di codice concorrenti), oppure come rivelatore di tre; se  $d = 5$ , il codice può correggere due errori oppure rivelarne quattro. Da ciò è facile costruire la regola generale.

Da questa regola si vede come per i codici di parità pari o dispari, avendo  $d = 2$ , non è possibile correggere alcun errore e se ne rivela solo uno.

E' possibile rinunciare ad una parte della capacità di correzione per ridurre la probabilità di correzione errata. Ad esempio, se  $d = 5$  ci si può limitare a correggere un solo errore, lasciando la capacità di rivelarne altri due.

Un altro importante metodo per il controllo del flusso di dati trasmessi in un sistema di comunicazione seriale è la *checksum*.

**Codici correttori**

La *checksum* può essere realizzata facendo l'operazione di XOR dei byte trasmessi. In alcuni casi viene fatta l'operazione di somma binaria e di somma binaria con il complemento a due. In ogni caso la *checksum* viene posta alla fine del blocco dei dati.

In ricezione viene calcolata la *checksum* e così dal confronto con quella trasmessa si nota la presenza dell'errore e si corregge.

**Codici di Hamming**

Un codice ancora molto semplice è il codice di Hamming (7,4) in cui le tre cifre di parità sono combinazioni *diverse* delle quattro cifre d'informazione. Infatti, fin dai primi tentativi di costruzioni di codici fu evidente che trasmettere più volte una stessa combinazione di cifre d'informazione non fosse una soluzione intelligente ed utile.

Le tre cifre di parità vengono calcolate eseguendo la somma senza riporto di tre delle quattro cifre d'informazione, seguendo il seguente schema:

$$\begin{cases} p_1 = i_2 + i_3 + i_4 \\ p_2 = i_1 + i_3 + i_4, \\ p_3 = i_1 + i_2 + i_4 \end{cases}$$

dove le  $i_k$  per  $k = 1..4$  sono le quattro cifre a partire dalla più significativa; in definitiva, la sequenza di cifre è  $i_1 i_2 i_3 i_4 p_1 p_2 p_3$ .

Si ottengono così le sedici parole di codice seguenti:

0000 → 0000 000

0001 → 0001 111

0010 → 0010 110

0011 → 0011 001

0100 → 0100 101

0101 → 0101 010

0110 → 0110 011

0111 → 0111 100

1000 → 1000 011

1001 → 1001 100

1010 → 1010 101

1011 → 1011 010

**Codici correttori**

1100 → 1100 110

1101 → 1101 001

1110 → 1110 000

1111 → 1111 111.

Come si vede in totale si utilizzano  $2^k = 2^4 = 16$  delle  $2^N = 2^7 = 128$  configurazioni di 7 bit. Si può facilmente verificare che le quindici parole non nulle contengono almeno tre uni, e quindi che le parole di codice differiscono fra loro di almeno tre posizioni. La distanza minima di Hamming è, pertanto,  $d = 3$  ed il codice può rivelare tutti gli errori semplici e doppi ed è in grado di correggere tutti gli errori semplici: per la correzione è sufficiente trovare la parola del codice che differisce in un solo bit da quella ricevuta.

**Matrice generatrice e di parità**

Si può rappresentare il codice di Hamming (7,4) nella forma matriciale:

$$c = i \cdot G,$$

dove  $c$  è il vettore riga di 7 elementi che deve essere trasmesso, cioè la parola di codice,  $i$  è il vettore riga di 4 elementi delle cifre d'informazione,  $G$  è la matrice generatrice di dimensione  $4 \cdot 7$ :

$$G = \begin{bmatrix} 1000 & 011 \\ 0100 & 101 \\ 0010 & 110 \\ 0001 & 111 \end{bmatrix}.$$

I tre bit finali per ogni riga si calcolano come visto in precedenza.

Le parole di codice sono tutte le combinazioni lineari delle righe della matrice generatrice  $G$ .

Le equazioni di parità

$$\begin{cases} c_5 = c_2 + c_3 + c_4 \\ c_6 = c_1 + c_3 + c_4 \\ c_7 = c_1 + c_2 + c_4 \end{cases}$$

possono essere scritte come  $c \cdot H = 0$ , dove  $H$  è la matrice  $7 \cdot 3$  detta matrice di parità:

**Codici correttori**

$$H = \begin{bmatrix} 011 \\ 101 \\ 110 \\ 111 \\ 100 \\ 010 \\ 001 \end{bmatrix} .$$

le righe sono la rappresentazione binaria dei numeri 1, 2, ..., 7. Non è importante l'ordine delle righe, in quanto si ottengono matrici di parità di un diverso codice di Hamming che, in ogni modo, risolve il problema.

**Sindrome**

La sindrome è un vettore di  $N - K$  componenti, che dice se le regole di parità sono o no soddisfatte. Si supponga di ricevere una N-pla (nel nostro caso 7 bit) che può essere scritta sotto forma matriciale nel seguente modo:

$$y = c + e,$$

dove  $e$  è il vettore errore.

La sindrome si calcola moltiplicando tale vettore per la matrice  $H$ , ottenendo:

$$s = y \cdot H = c \cdot H + e \cdot H = e \cdot H.$$

Nel nostro caso, la sindrome è un vettore che ha  $N - K = 3$  componenti. La sindrome può presentarsi in  $2^3 = 8$  modi, mentre le possibili configurazioni dell'errore sono  $2^7 = 128$ . Si verifica

che la stessa sindrome corrisponde a  $\frac{128}{8} = 16$  diverse configurazioni d'errore. In ricezione fra queste si sceglie la più probabile, cioè quella con il minor numero di errori.

Per i codice semplici e spesso di non interesse, la ricerca può essere fatta a priori e memorizzata.

Nel caso in esame si ha:

000 → 0000000;

001 → 0000001;

010 → 0000010;

011 → 1000000;

**Codici correttori**

100 → 0000100;

101 → 0100000;

110 → 0010000;

111 → 0001000.

Risultano correggibili tutti e soli gli errori singoli. Infatti, con un errore in una sola posizione la sindrome coincide con la corrispondente riga della matrice di parità. Poiché le righe di H sono tutte diverse, ed in numero pari a possibili errori singoli (cioè N), tutti gli errori singoli sono correggibili. In generale si hanno infiniti codici di Hamming con parametri N e K legati dalla relazione:

$$N = 2^{N-K} - 1.$$

Alcuni esempi possono essere:

N = 3, K = 1 (codice banale 000,111);

N = 7, K = 4;

N = 15, K = 11;

N = 31, K = 26;

...

N = 1023, K = 1013;

...

Si noti che il rate  $\frac{K}{N}$  tende a 1 per  $N \rightarrow \infty$ , ma si corregge comunque un solo errore. Quindi, N grande va bene solo per canali con probabilità di errore già molto piccola, che viene ulteriormente ridotta dal codice. In tutti gli altri casi occorrono codici correttori di più di un errore.

A parità di ridondanza, codici con blocco lungo e correttori di molti errori sono migliori di codici con blocco corto e correttori di pochi errori, o di un solo errore. Però, la complessità del decodificatore aumenta fortemente all'aumentare del numero di errori correggibili.

In ricezione, in linea di principio basta calcolare la sindrome  $s = y \cdot H$ , e se il codice è usato solo come rivelatore basta verificare se  $s = 0$ . Se, invece, il codice è usato come correttore, dalla sindrome si ricava la N-pla d'errore più probabile, e si provvede a correggere la parola ricevuta.

Se  $N - K$  è elevato, la correzione diventa molto pesante. Le due matrici generatrice e di parità sono semplici ed utili solo nel caso raro di codici di piccole dimensioni. Pertanto, occorre una descrizione del codice più sintetica e con una struttura algebrica più potente.

**Codici correttori****Codici ciclici**

Alla parola  $c_{N-1} c_{N-2} \dots c_0$  di lunghezza  $N$  è possibile associare il polinomio in  $x$ :

$$c_{N-1} x^{N-1} + c_{N-2} x^{N-2} + \dots + c_0.$$

La variabile  $x$  individua solo la posizione della cifra.

I codici polinomiali sono tali per cui  $c(x)$  è una parola di codice se e solo se è divisibile per un polinomio detto generatore del codice  $g(x)$ , di grado  $N - K$ .

I codici ciclici sono codici polinomiali in cui il polinomio generatore del codice  $g(x)$  è un divisore di  $x^{N-1} + 1$ . Ad esempio il codice di Hamming (15,11), in versione ciclica, ha

$$N = 15$$

$$K = 11$$

$$N - K = 4 \text{ (grado del polinomio generatore)}$$

$$g(x) = x^4 + x + 1 \text{ (polinomio generatore)}$$

$$m(x) = (x^{14} + 1) / g(x) \text{ (polinomio del messaggio).}$$

I codici ciclici hanno la proprietà, da cui deriva il nome, che se  $c_{N-1} c_{N-2} \dots c_0$  è parola di un codice lo è anche  $c_{N-2} c_{N-3} \dots c_0 c_{N-1}$ . Codificatori e decodificatori dei codici ciclici sono più semplici di quelli per codici non ciclici.

Facciamo un esempio del codice di Hamming (15,11):

$$m = 11000000001 \text{ (} K = 11 \text{)}$$

$$m(x) = x^{10} + x^9 + 1$$

$$c(x) = g(x) \cdot m(x) = (x^4 + x + 1)(x^{10} + x^9 + 1) = x^{14} + x^{13} + x^{11} + x^9 + x^4 + x + 1$$

(da notare che nel prodotto si hanno due  $x^{10}$  la cui somma senza riporto o EX-OR dà zero)

$$c = 110101000010011 \text{ (codice da trasmettere).}$$

In questa forma però il codice non è sistematico. Infatti, i bit vengono mischiati e si perde la possibilità di trasmettere primi i bit d'informazione e solo dopo quelli parità.

Per ottenere la versione sistematica si utilizza il codice a ridondanza ciclica CRC (Cyclic Redundancy Check).

Il blocco da trasmettere composto da una sequenza di  $K$  bit, è interpretato come l'insieme dei coefficienti di un ipotetico polinomio di grado  $(K - 1)$ ; a questo vengono aggiunti  $N - K$  zeri nelle ultime posizioni e successivamente il polinomio così ottenuto viene diviso, mediante l'aritmetica modulo 2, per un polinomio generatore prestabilito a priori e di grado  $(N - K)$ . I coefficienti del

**Codici correttori**

polinomio resto sostituiscono i coefficienti nulli aggiunti nel polinomio di partenza, ottenendo un polinomio di grado  $N$  che è sicuramente divisibile per quello generatore.

In ricezione, il blocco ricevuto è virtualmente diviso per il polinomio generatore e solo se il resto è nullo comporterà l'assenza di errori. Se invece il resto sarà diverso da zero, avremo rivelato uno o più errori.

E' da notare come il CRC è basato sull'aritmetica modulo 2 e non sulla distanza di Hamming per rivelare, ma non correggere, vari tipi di errori. Infatti, possono essere rivelati:

- errori singoli, se il polinomio generatore  $g(x)$  ha il termine noto (cioè il bit meno significativo diverso da 0);
- errori doppi, senza particolari regole;
- errori dispari, se  $g(x)$  contiene fra i suoi fattori primi il binomio  $(x + 1)$ ;
- errori a burst, se la loro lunghezza è minore del grado del polinomio  $g(x)$ .

La scelta del polinomio generatore, collegata direttamente al tipo di errore che si vuole rivelare, viene effettuata a livello statistico, ed ha portato alla definizione delle seguenti standardizzazioni:

$$\text{CRC\_16} = 11000000000000101 \quad g(x) = (x^{16} + x^{15} + x^2 + 1)$$

$$\text{CRC\_CCITT} = 10001000000100001 \quad g(x) = (x^{16} + x^{12} + x^5 + 1).$$

Vediamo, adesso, un esempio di costruzione del codice CRC per trasmettere il seguente messaggio in binario con codice di Hamming (15,11):  $m = 10110101101$  ( $K = 11$ ).

Il polinomio corrispondente di grado  $K - 1 = 10$  è il seguente  $m(x) = x^{10} + x^8 + x^7 + x^5 + x^3 + x^2 + 1$

Si aggiungono  $N - K = 4$  coefficienti nulli, ottenendo il seguente messaggio 10110101101 0000.

Il polinomio generatore deve essere di grado  $N - K = 4$ , ad esempio  $g(x) = x^3 + x + 1$  (1011).

Si fa la divisione con l'aritmetica modulo 2 fra il messaggio di 15 bit (compresi gli zeri) ed il polinomio generatore, si ottiene un resto a 4 bit che si deve mettere al posto dei quattro zeri, ottenendo il messaggio finale da trasmettere: 10110101101 **0110**.

Se non ci sono errori in trasmissione, in ricezione si fa la divisione con il polinomio generatore ottenendo il resto nullo.

Supponiamo, adesso, che il polinomio generatore venga rappresentato in questo modo del tutto generale:  $g(x) = x^m + g_{m-1} \cdot x^{m-1} + \dots + g_2 \cdot x^2 + g_1 \cdot x + 1$ .

Il circuito atto a generare direttamente il CRC corrispondente al polinomio generatore è realizzato con uno shift-register come mostrato in fig. 1, dove:

**Codici correttori**

- il simbolo  $\oplus$  rappresenta un sommatore modulo 2 (un EX-OR);
- i rettangoli rappresentano dei flip-flop di tipo D;
- il simbolo  $\otimes$  rappresenta un corto circuito o un circuito aperto a seconda del corrispondente valore di  $g_i$  nel polinomio generatore e precisamente un corto circuito se  $g_i = 1$  e un circuito aperto se vale 0.

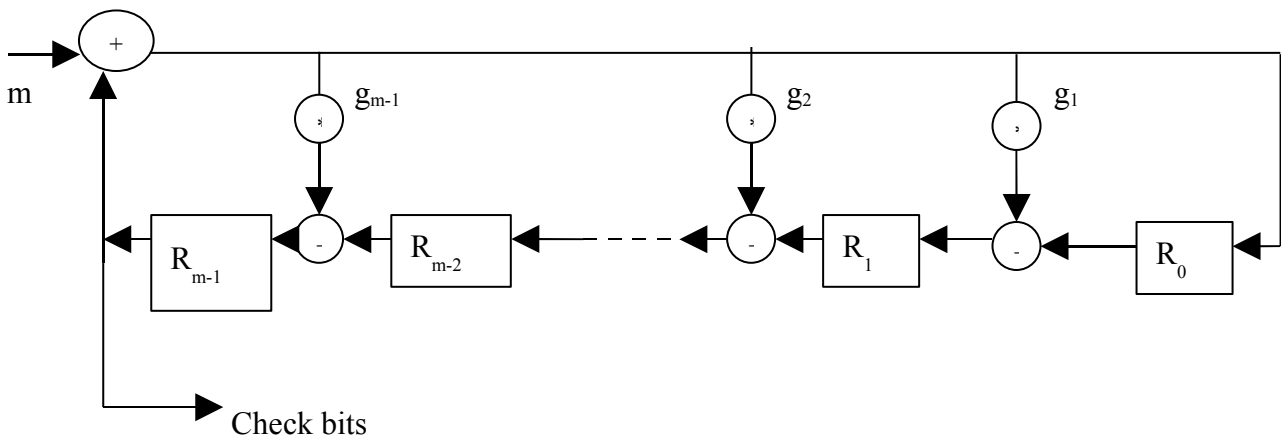


Fig.1 – Shift-register per la creazione del CRC.

La determinazione del valore del CRC è fatta tramite la tabella degli stati. Nel caso di un polinomio generatore di terzo grado si ha  $g(x) = x^3 + g_2 \cdot x^2 + g_1 \cdot x + 1$ .

	Stato attuale			Stato futuro		
M	$R_2$	$R_1$	$R_0$	$R'_2$	$R'_1$	$R'_0$
	0	0	0			

La colonna relativa ad M contiene i bit del messaggio. I valori di  $R_2, R_1, R_0$  e di  $R'_2, R'_1, R'_0$  si ricavano dalle equazioni che regolano il circuito stesso. Nel nostro caso di polinomio generatore di terzo grado si ha un circuito generatore del tipo di fig. 1, le cui equazioni che regolano il circuito sono:

**Codici correttori**

$$\begin{cases} R'_0 = M \oplus R_2 \\ R'_1 = [g_1(M \oplus R_2)] \oplus R_0 \\ R'_2 = [g_2(M \oplus R_2)] \oplus R_1 \end{cases}$$

**Codici BCH**

Sono codici binari correttori di più errori, che includono come caso particolare i codici di Hamming. E' molto complicato spiegare anche in modo elementare la proprietà di distanza dei codici BCH. Infatti è necessaria la conoscenza dell'algebra dei campi finiti.

Tra le possibili modificazioni dei codici, e quindi anche dei codici BCH, le più semplici e comuni sono:

- *estensione*: aggiunta di cifre di parità; l'esempio più comune è l'aggiunta di un bit di parità;
- *accorciamento*: si possono modificare i valori di N e K, ponendo a zero e ovviamente non trasmettendo le prime b cifre d'informazione. In questo modo si ottiene un codice (N-b,K-b) La distanza non diminuisce, anzi in alcuni casi potrebbe aumentare. Nel caso di codici ciclici è sufficiente saltare i primi b passi della codifica. Il codice avvenuto per accorciamento non è ciclico.

**Decodifica hard e soft**

La decodifica algebrica, chiamata hard, dei codici BCH è incomprensibile se non si conoscono i campi di Galois. Tuttavia non è molto complessa per codici correttori di 5 ÷ 20 errori.

In uscita dal canale di trasmissione si prendono decisioni indipendenti bit per bit. Poi, si effettua la ricerca dell'unica parola di codice che non dista dalla ricevuta più del potere correttore t. Se sono avvenuti più di t errori la decodifica può fallire in due modi:

- non si trova una parola che disti meno di t; in tal caso si rinuncia alla correzione, ma l'errore viene rivelato;
- si trova una parola, ma non è quella trasmessa; in tal caso si accetta una parola con errori non rivelati.

**Codici correttori**

Per ridurre la probabilità di errori non rivelati si può limitare la correzione a  $C < t$  errori.

Se all'uscita del canale è disponibile l'informazione soft  $r_n$  sul livello ricevuto, prendendo decisioni hard bit a bit si perde informazione.

In generale si preferisce la decodifica soft, come ad esempio quella a massima verosimiglianza (ML).

L'esperienza mostra che la decodifica soft guadagna circa 2 dB rispetto alla hard. Il problema è che le parole di codice sono  $2^K$  e se  $K = 100$  se hanno circa  $10^{30}$  parole.

Sono stati proposti metodi approssimati (Chase) basati sulla ricerca di un piccolo numero di parole di codice candidate, individuate decodificando in modo algebrico la parola ricevuta ed altre con alcuni bit (i meno affidabili) complementati. Si utilizza la decodifica soft in questo insieme di parole sicuramente più limitato.

Tra le tecniche di decodifica soft vi è anche quella a massima probabilità a posteriori (MAP bit per bit): si calcola, per ciascun bit d'informazione, la probabilità che sia pari a 0, e si decide per lo zero se la probabilità è superiore a 0,5. Esistono due metodi esatti di complessità  $N \cdot 2^{N-K}$ , proibitiva per la maggior parte dei codici.

**Codici Reed-Solomon**

Fra i codici non binari, la classe più importante è certamente quella dei redd-Solomon. L'alfabeto delle cifre d'informazione ha  $q$  elementi, dove  $q$  è un numero primo o una sua potenza. Il caso più comune indica per  $q$  il valore di  $2^m$ . In tal caso una cifra  $q$ -aria è rappresentabile anche con una sequenza di  $m$  bit.

I parametri dei codici Reed-Solomon sono i seguenti:

$$N = q - 1$$

$$K < N$$

$$d = N - K + 1.$$

Un semplice esempio è il seguente:

$$m = 8$$

$$N = 2^8 - 1 = 255$$

$$K = 239$$

$$d = 17.$$

**Codici correttori**

Tale codice riesce a correggere 8 errori, cioè fino a 8 byte errati (non importa quanti bit errati contiene un byte errato).

Se vogliamo considerarlo come un codice binario si hanno:  $239 \cdot 8 = 1912$  bit di informazione e  $255 \cdot 8 = 2040$  bit totali.

Il codice può essere accorciato; per esempio:  $N = 204$ ,  $K = 188$ .

**Codici per errori concentrati a pacchetti**

Esistono molte classi di errori per codici concentrati e di solito la decodifica è agevole.

Il problema principale è avere un buon modello di canale e quindi conoscere come possono concentrarsi gli errori. Un codice non adatto potrebbe addirittura essere peggiore di una trasmissione non codificata.

Una tecnica di codifica per canali con errori a pacchetti (burst) è l'interleaving a blocchi (fig.2).

In trasmissione si scrive in memoria per righe in una matrice  $B \cdot N$  e si rilegge per colonne. In ricezione si effettua l'operazione inversa. In questo modo eventuali errori consecutivi commessi dal canale vengono separati di almeno  $N$  simboli, purché la lunghezza del burst non superi  $B$ .

Usando un codice a blocco di lunghezza  $N$  gli errori cadono in parole diverse ed il codice può essere progettato per errori casuali e non a burst.

Esistono anche interleaver convoluzionali. A parità di prestazioni richiedono circa metà memoria ed introducono circa metà ritardo.

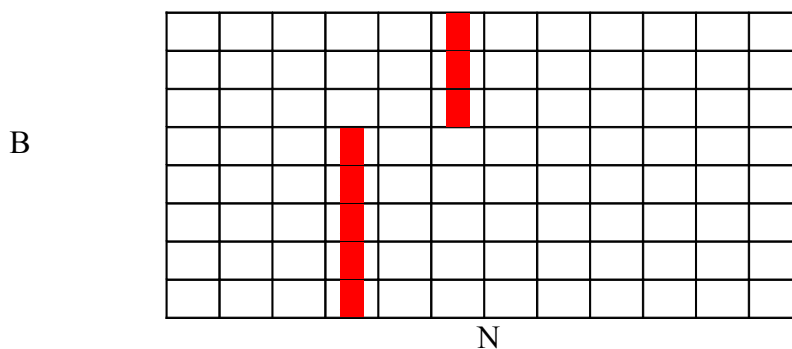


Fig.2 – Interleaving a blocchi. In rosso sono evidenziati gli errori.

**Codici concatenati**

In fig. 3 viene mostrato un esempio di concatenazione serie dei codici, con codice interno Hamming (7,4) correttore di un errore e codice esterno Reed-Solomon (15,11) correttore di due errori, che opera su byte di 4 bit. Un blocco ha 44 bit d'informazione, suddivisi in 11 byte di 4 bit. Il codificatore esterno aggiunge 4 byte di parità (16 bit). Ad ogni byte di 4 bit il codificatore interno aggiunge 3 bit di parità. Il tutto equivale ad un codice  $(15 \cdot 7, 11 \cdot 4) = (105, 44)$ .

Le prestazioni sono inferiori a quelle di un BCH accorciato (106,43), correttore di 10 errori, ma la decodifica è molto più semplice.

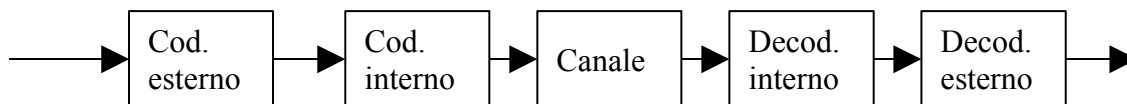


fig.3 – Codifica concatenata

Nei codici tradizionali concatenati il codice esterno è sempre un *Reed-Solomon*. Quello interno è un semplice codice a blocco, con decodifica hard o soft, oppure più spesso un convoluzionale con decodifica di Viterbi (soft). In questo caso occorre introdurre un *deinterleaver* per sparpagliare i blocchi di errori prodotti dal decodificatore interno, ed un corrispondente *interleaver* tra i due codificatori. Altrimenti verrebbe facilmente superato il potere correttore del codice esterno.