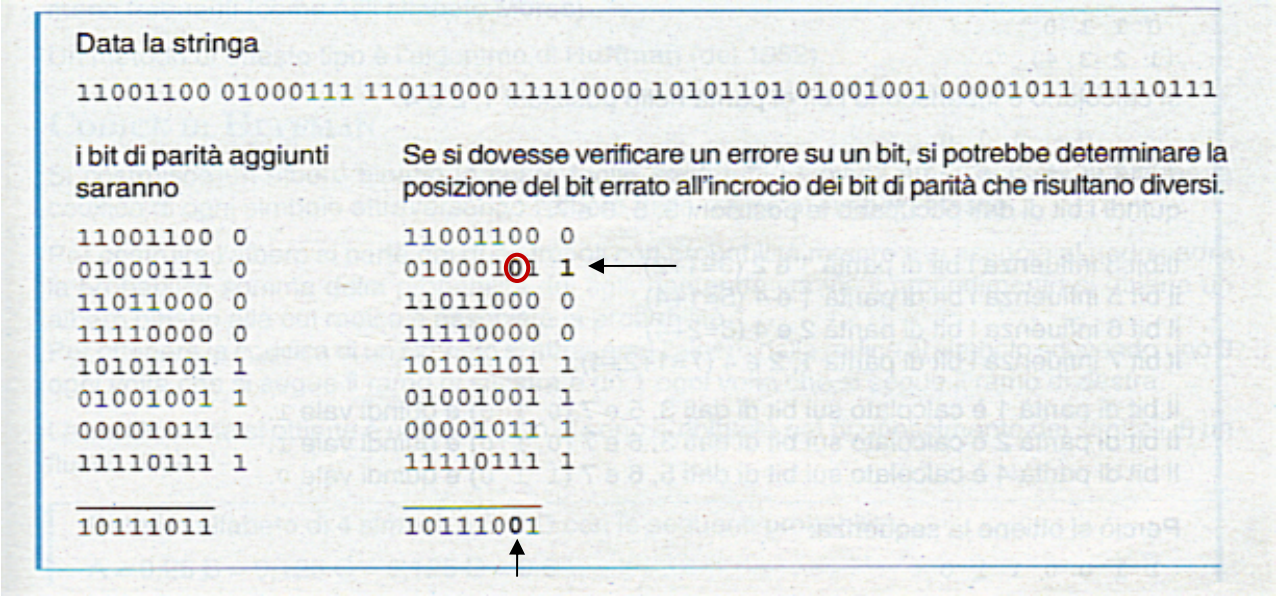
CODICI CORRETTORI

Parità incrociata

La tecnica più elementare di codice correttore è quella a **parità incrociata**. Questa tecnica consiste nel calcolare oltre al bit di parità per ogni parola (parità trasversale) anche un bit di parità per i bit che occupano la stessa posizione in ogni parola di un messaggio (parità longitudinale) e quindi richiede di aggiungere al messaggio un'ulteriore parola. L'aumento della ridondanza porta ad una maggiore efficacia nella gestione degli errori. La parità incrociata permette di individuare la posizione di un errore su un singolo bit (evidenziato sia dal bit di parità trasversale che dal bit di parità longitudinale) e quindi di correggerlo. Se gli errori sono più di uno, possono solo essere rilevati ma non corretti e in certi casi alcuni errori possono non venire individuati.



Codice di Hamming

DISTANZA DI HAMMING

Una **parola di codice** (codeword) su *n* bit consiste di *m* bit di dati e di *r* bit di controllo con *n=m+r.*

Il numero di bit diversi tra due parole di codice viene detto **distanza di Hamming;** si può calcolare il numero di bit diversi facendo l'or *esclusivo* delle due stringhe e contando il numero di bit 1 del risultato; se due parole hanno distanza ***d*** significa che servono *d* errori per trasformare una nell'altra.

es.: le stringhe: 1011011 1010010 hanno distanza 2 perché hanno solo 2 bit diversi

In genere non tutte le possibili stringhe di *n* bit (2^n) sono legali (anche se quelle con *m* bit di dati lo sono); la distanza minima tra le parole legali del codice è la **distanza di Hamming del codice.**

es.: se le parole di un codice sono: 1011011 1010010 1101001 0100110
la distanza minima si determina calcolando la distanza tra tutte le coppie e poi prendendo la minima tra le distanze calcolate.

La distanza di Hamming indica quanti errori si possono rilevare e quanti se ne possono correggere. Per **rilevare *d* errori** serve una distanza di ***d+1;*** per **correggere *d* errori** serve una distanza di ***2d+1*** (la parola originale è la più vicina valida).

CODICE DI HAMMING

Il codice di Hamming è un codice che permette di aggiungere un certo numero di bit ai bit di dati in modo da comporre parole con **distanza 3** in grado di rilevare e correggere errori su un singolo bit. Il numero di bit da aggiungere aumenta all'aumentare del numero dei bit di dati.

I bit aggiunti sono **bit di parità** calcolati su sottoinsiemi di bit della parola di codice; numerando a partire da 1 a sinistra i bit che compongono la parola di codice, i bit di parità vengono inseriti nelle posizioni che sono potenze di 2 (1, 2, 4, 8,16 ...); gli altri bit sono i bit di dati. Ogni bit di parità viene calcolato su un sottoinsieme di bit; ogni bit di dati può essere incluso in diversi sottoinsiemi e influire su diversi bit di parità. Per sapere su quali bit di parità influisce il bit di dati *k* basta riscrivere *k* come somma di potenze di 2 (per esempio 11 =1 +2 + 8); ogni bit di dati è controllato da tutti e soli i bit di parità che appartengono alla sua espansione (il bit 11 è controllato dai bit 1, 2 e 8).

In ricezione, per ogni parola di codice vengono ricalcolati i bit di parità per ogni posizione *k* (con k= 1, 2, 4, 8,16...). Se la parità non è corretta viene aggiunto *k a* un contatore inizializzato a 0; al termine il valore del contatore indica la posizione del bit errato (se non sono corretti i bit di parità 1, 2 e 8 il bit errato è quello in posizione 11).

Esempio:

Partendo dalla sequenza: 0 1 1 0

(1 2 3 4) posizioni

si calcolano e inseriscono i *bit di parità* nelle posizioni 1, 2 e 4:

```
 _ _ 0 _ 1 1 0
(1 2 3 4 5 6 7) posizioni
```

quindi *i bit di dati* occupano le posizioni 3, 5, 6 e 7.

```
(1 2 3 4 5 6 7) posizioni
 _ _ 0 _ 1 1 0
    1   1   1
    2     2 2
        4 4 4
```

Il bit 3 influenza i bit di parità 1 e 2 (3=1+2). il bit 5 influenza i bit di parità 1 e 4 (5=1+4). il bit 6 influenza i bit di parità 2 e 4 (6=2+4). il bit 7 influenza i bit di parità 1, 2 e 4 (7=1+2+4).

Il bit di parità 1 è calcolato sui bit di dati 3, 5 e 7 (0 1 0) e quindi vale 1. Il bit di parità 2 è calcolato sui bit di dati 3, 6 e 7 (0 1 0) e quindi vale 1. Il bit di parità 4 è calcolato sui bit di dati 5, 6 e 7 (1 1 0) e quindi vale 0.

Perciò si ottiene la sequenza: 1 1 0 0 1 1 0

Se un errore modifica la sequenza in: 1 1 0 0 1 **0** 0

(1 2 3 4 5 6 7) posizioni

ricalcolando i bit di parità si ottiene:
bit di parità 1 calcolato sui bit di dati 3, 5 e 7 (0 1 0): 1
bit di parità 2 calcolato sui bit di dati 3, 6 e 7 (0 0 0): 0
bit di parità 4 calcolato sui bit di dati 5, 6 e 7 (1 0 0): 1

Confrontando i bit della sequenza con quelli calcolati viene incrementato il contatore k:
il bit di parità 1 è uguale: k=0;
il bit di parità 2 è diverso: k=2 (+2);
il bit di parità 4 è diverso: k=6 (+4).

Pertanto si può stabilire che c'è un bit errato nella posizione 6 e lo si può correggere riottenendo la sequenza: 1 1 0 0 1 1 0.

Codici ridondanti vengono usati per esempio nei sistemi di comunicazione o nelle operazioni di memorizzazione dei dati. Nei sistemi di comunicazione ciò permette di verificare se le informazioni sono state trasmesse correttamente; se viene usato un codice rilevatore, in caso di errore bisogna chiedere al mittente di ritrasmettere le informazioni.
In una operazione di memorizzazione l'errore si può verificare al momento della scrittura o della successiva lettura dei dati; se viene rilevato un errore si può ritentare la lettura, ma se l'errore si è verificato in scrittura il problema si può risolvere solo se è stato usato un codice correttore.