Lezione 4 Codici ridondanti e non

lunedì 2 ottobre 2023 17:37

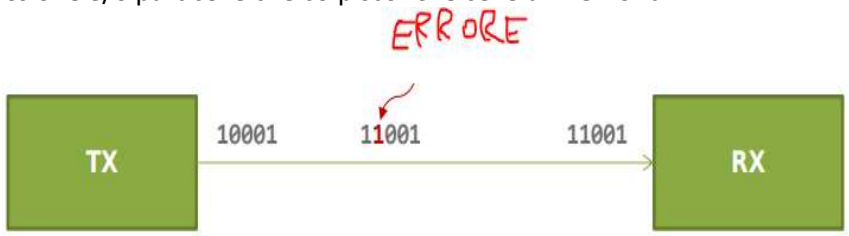
Riepilogando : con n bit possiamo creare una qualsiasi codifica di una qualunque parola, come per esempio con 3 bit possiamo codificare i giorni della settimana, assegnando ad ogni giorno una codifica :

Lunedi	000
Martedi	001
Mercoledi	010
Giovedi	011
Venerdi	100
Sabato	101
Domenica	110

Ma attenzione!!! Finora abbiamo supposto che esista una sola codifica per ciascun elemento che vogliamo rappresentare, ma può capitare che vi sia un codifica particolare : quella ridondante : quella codifica che usa più bit di quelli concessi per la rappresentazione. Vediamo un esempio : dati N elementi da rappresentare e n cifre possibile e m=log2N, si ha che:

1. n=m
   a. Codice irridondante (va bene).
   b. Distanza di humming =1 (va bene)
2. n>m
   a. Codice ridondate
   b. Le k=n-m cifre aggiuntive sono chiamate cifre di controllo.
   c. Distanza di humming <=h-1
   d. In questi codici si riesce a trovare e*o correggere errori grazie alla ridondanza: un esempio può essere l0inversione di bit , la quale può essere dovuta ad errori di trasmissione e/o particelle che colpiscono le celle di memoria:

i.



Vediamo ora un primo modo di vedere se la codifica è "giusta": Usiamo il codice di hamming: numero di bit diversi tra due parole:

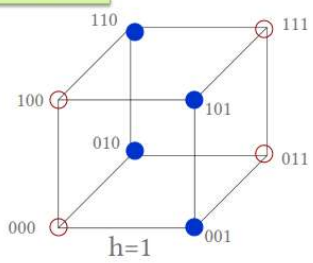
D(10010,01001)=4
D(11010,11001)=2

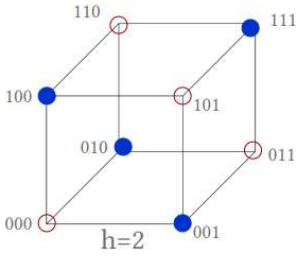
In generale la distanza di hamming è uguale al valore minimo della distanza tra due parole :

H=min(d(x,y)) , per qualunque parola x != parola y

Andiamo ora a vedere i codici rivelatori di errore : in generale questi codici possono rivelare uno o più bit di errore, ed in base al quantitativo di bit si hanno diverse rappresentazioni , ognuna delle quali ha parole ammesse ed altre no :

Elemento	Codice 1	Codice 2
A	000	000
B	100	011
C	011	101
D	111	110





Parole del codice (legali)

Parole non appartenenti al codice

Ricordiamo che i moderni calcolatori interagiscono con i dati attraverso il codice ascii , ovvero una rappresentazione ad 8 bit di ogni singolo carattere.

Addentriamoci ora veramente nei codici rivelatori di errori : BCD (Binary decimal code), Codice Gray , Codice di parità. Il primo è un codice ridondante , che rappresenta le 10 cifre decimali (0-9) usando 4 bit . Ognuna di queste cifre è codificata indipendentemente : di queste 4 cifre 2 sono memorizzate in un singolo byte . Da notare che con questa codifica circa 1/6 dei bit della codifica sono sprecati:

Base 10	BCD	Base 10	BCD
0	0000	5	0101
1	0001	6	0110
2	0010	7	0111
3	0011	8	1000
4	0100	9	1001

Passiamo ora al secondo : questo codice inventato da Grey , ha una lunghezza fissa , ma è caratterizzato da una codifica del bit tale che tra due numeri adiacenti cambi un solo bit :

Base 10	Gray	Base 10	Gray
0	000	4	110
1	001	5	111
2	011	6	101
3	010	7	100

Ma ha una pecca : non di distinguono le configurazioni transitorie da quelle corrette.

Vediamo ora l'ultimo: questo è un semplice codice ridondante con ridondanza h=2, e si ottiene **aggiungendo una cifra di parità al codice irridondante:**

Codice irridondante	Parità	Disparità
000	000 **0**	000 **1**
001	001 **1**	001 **0**
010	010 **1**	010 **0**
011	011 **0**	011 **1**
100	100 **1**	100 **0**
101	101 **0**	101 **1**
110	110 **0**	110 **1**
111	111 **1**	111 **0**

Due tipologie : parità vale 1 se il numero di "1" nella codifica irridondate è dispari e disparità se il numero di "1" nella codifica è pari: vediamo un esempio:

Ricevuto	Parità	Segnale di errore
1010	uguale a zero	**OK**
1110	diversa da 0	**ERRORE**
1111	uguale a zero	**OK**