ERRORI
rilevazione & correzione

CODICI & …

- CODICE: una convenzione per rappresentare qualcosa
- Es.: codice binario, rappresenta numeri
- Es.: codice ASCII, rappresenta caratteri
- Es.: codice Morse, rappresenta caratteri
- Esiste pertanto un qualcosa di “naturale” che viene codificato
- Perché viene codificato? Per comodità; è più semplice elaborarlo, trasmetterlo, …
- In informatica e telecomunicazioni (ICT) i codici utilizzano sempre più i bit (mondo digitale)
- Il processo fondamentale è:

CODICI & …

- Esiste evidentemente il processo inverso, detto: decodifica

- fondamentale per recuperare l'informazione
- La codifica è realizzata con HW (circuiti) o SW (programmi)
- Termini comuni sono "encoder" e "decoder"
- Es.: il segnale video viene codificato da un encoder, trasmesso e ricevuto con antenne e alla fine decodificato da opportuni decoder

ERRORI in memoria o in rx/tx

- Occasionalmente i circuiti di memoria o i circuiti di rice/trasmissione possono modificare uno o più bit alterando l'informazione
- Come faccio a rilevare (detect) l'errore?
- Come faccio a correggere (correct) l'errore?

scenario

- Rice/trasmissione tra due circuiti digitali: A emette N bit, B riceve
- Come può B riconoscere un eventuale errore?
- Proposte, idee …
- Esiste un modo infallibile (sempre infallibile, quindi “certo”) per rilevare un errore?

ERRORI in memoria o rx/tx (continua)

- Occasionalmente i circuiti di memoria o i circuiti di rice/trasmissione possono modificare uno o più bit alterando l'informazione
- Come faccio a rilevare (detect) l'errore?
- Come faccio a correggere (correct) l'errore?
- Risposta: con opportuni codici di rilevazione e di correzione dell'errore
- In pratica? Vengono aggiunti bit extra a ogni parola (byte/word/dword ecc.)

Parola di codice (Codeword) aumenta a n bit

Bit ridondanti (bit di controllo)

$$n = m + r$$

Bit dei dati

Distanza di Hamming / 1

- *Distanza di Hamming* fra due parole = numero di bit diversi
- Es.: 10001001 e 10110001 hanno distanza di Hamming = 3
- Distanza di Hamming di un codice: minima distanza di Hamming fra le parole di uno stesso codice
- Distanza di Hamming del codice binario? Del codice ASCII?

Distanza di Hamming / 2

- Se $n = m + r$ solo 2^m delle 2^n parole sono valide
- Se in lettura si ha una parola non valida ➔ è avvenuto un errore
- Esempi…
- La possibilità di rilevazione o di correzione degli errori di un codice dipende dalla distanza di Hamming

Le 2 leggi di rilevazione e correzione dell'errore

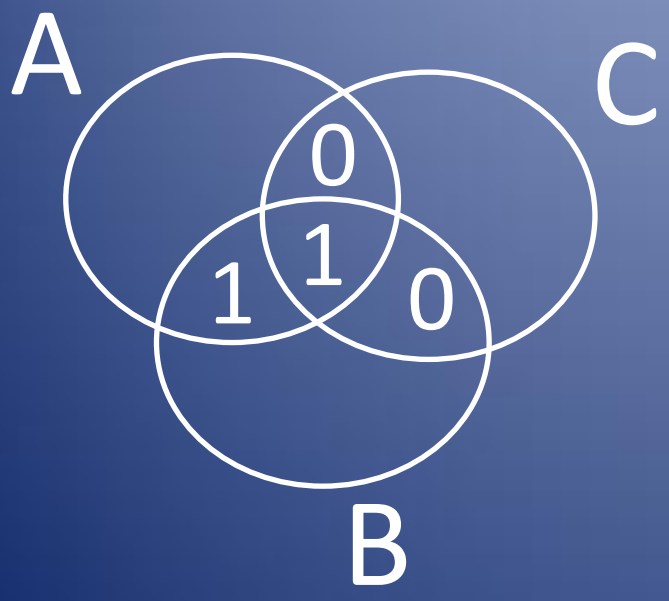
- Dato un certo codice… (con n=m+r)
- per rilevare *E* errori esso deve avere distanza di Hamming = *E + 1*
- per correggere *E* errori deve avere una distanza di Hamming = *2E+1*
- Chi decide E? ehhhhhhhh l'ambiente in cui il codice si applica; nelle RAM delle motherboard un errore è mooooolto raro (si parla di anni), E=1 è già una buona scelta
- Nelle telecomunicazioni si hanno molte sorgenti di rumore e molti disturbi; E=1 non basta

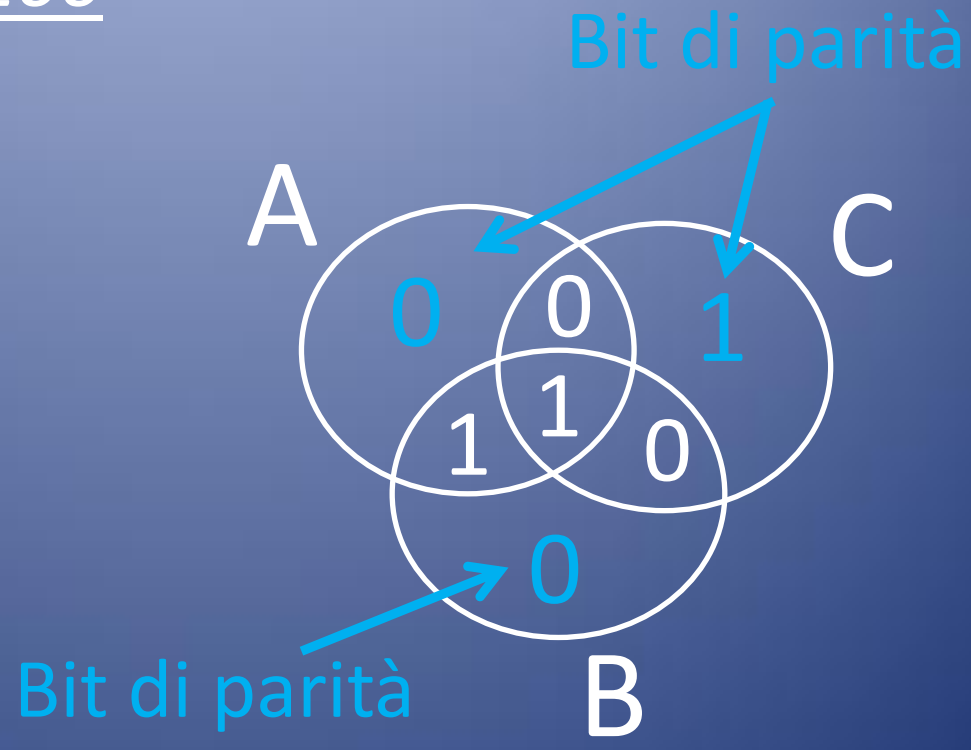
Esempio error detect: bit di parità

- Permette di rilevare (non correggere!) un solo errore
- Si aggiunge ai bit del dato un singolo bit, ridondante, detto **bit di parità**
- scelto in modo che il numero di 1 nella parola di codice sia pari (schema di “parità pari”) o dispari (schema di “parità dispari”).
- Il codice risultante ha distanza di Hamming uguale a ….? Esibire esempi di due codeword valide e calcolare la distanza di Hamming; è la distanza minima? Fare un esempio di codeword non valide

Esempio error correct con m=4, r=3, E=1

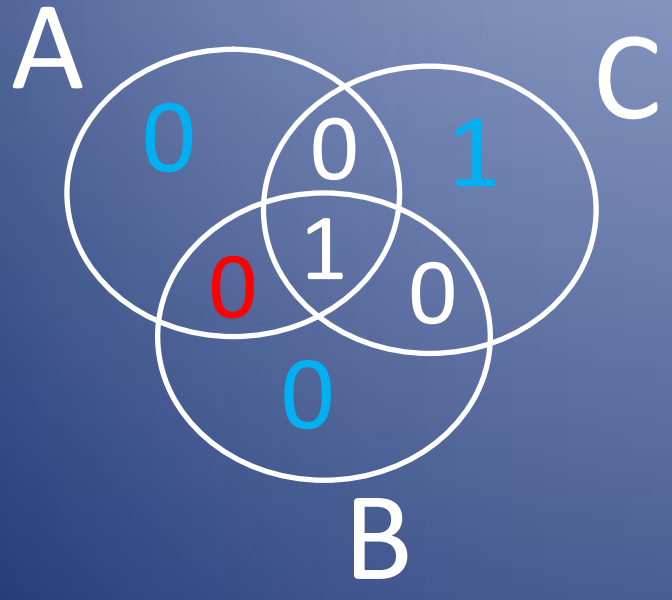
Codeword nativa: 1100

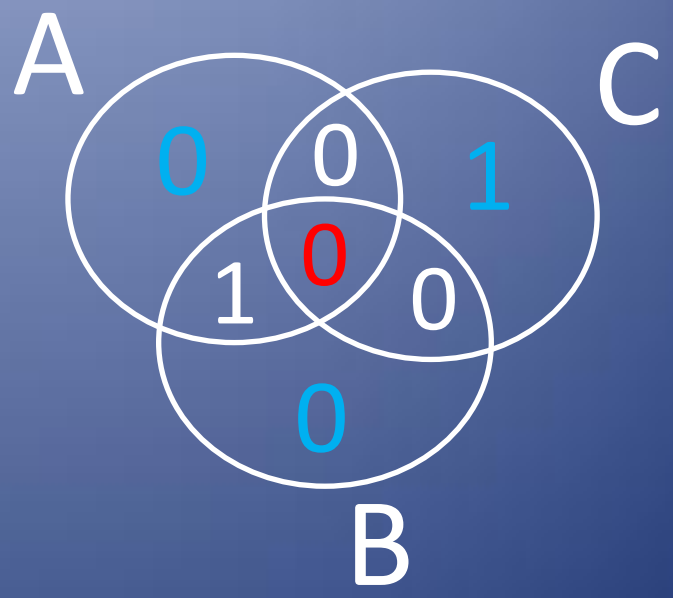




Esempio error correct con m=4, r=3, E=1

Codeword nativa: 1100 -- in rosso l'errore





Che succede se l'errore affligge un bit di parità?

Algoritmo di Hamming / 1

- Codeword originaria a m bit
- Si aggiungono r bit di parità
- r si ricava con la formula: $m < 2^r - r - 1$
- Codeword risultante a $n = m + r$ bit

Algoritmo di Hamming / 2

1. Numerazione bit a partire da 1 (non 0)
2. Il bit 1 è il bit più a sinistra
3. I bit la cui posizione è una potenza di 2 sono bit di parità
4. Gli altri sono bit di dati

Algoritmo di Hamming / 3

Ciascun bit di parità controlla specifiche posizioni:

- Il bit 1 controlla i bit 1,3,5,7,9,11,13,15,17,19,21
- Il bit 2 controlla i bit 2,3,6,7,10,11,14,15,18,19
- Il bit 4 controlla i bit 4,5,6,7,12,13,14,15,20,21
- Il bit 8 controlla i bit 8,9,10,11,12,13,14,15
- Il bit 16 controlla i bit 16,17,18,19,20,21

Algoritmo di Hamming / 4

In generale:

Il bit di posto *p* e' controllato dai bit di posto

$$p1, p2, p3, \ldots, pj$$

tali che

$$p1 + p2 + \ldots + pj = p$$

Esempi:

Il bit di posto 5 e' controllato dai bit di posto 1 e 4.

Il bit di posto 6 e' controllato dai bit di posto 2 e 4.

Esempio con 4 bit di dati / 1

Dati: 0011 (m=4)

d1	d2	d3	d4
0	0	1	1

Aggiungo 3 posizioni per i bit di parità:

1	2	3	4	5	6	7
p1	p2	d1	p3	d2	d3	d4
		0		0	1	1

Aggiungo i bit di parità:

1	2	3	4	5	6	7
p1	p2	d1	p3	d2	d3	d4
1	0	0	0	0	1	1

Esempio con 4 bit di dati / 2

1	2	3	4	5	6	7
p1	p2	d1	p3	d2	d3	d4
1	0	0	0	0	1	1

n = m + r

m = 4 r = 3

Codeword corretta: 1000011

Dati: 0011

In ricezione arriva: 1010011 → errore

Dati arrivati: 1011

Esempio con 4 bit di dati / 3

Calcolo bit di parità per i bit arrivati:

1	2	3	4	5	6	7
p1	p2	d1	p3	d2	d3	d4
		1		0	1	1

1	2	3	4	5	6	7
p1	p2	d1	p3	d2	d3	d4
0	1	1	0	0	1	1

Esempio con 4 bit di dati / 4

Posizione	Arrivati	Calcolati	XOR
p1	1	0	1
p2	0	1	1
p3	0	0	0

Il risultato dello XOR letto dal basso verso l'alto indica la posizione del bit errato: 011

Il bit errato e' in posizione 3