Complemento a 2

- Unica rappresentazione dello 0!
- Data una successione di n bit, per codificare un numero intero X, si utilizza il valore binario corrispondente a (2^n+X).
- Es: codifica su 4 bit => $2^4=16$

+6 = 16+6 = +22 => 10110 

0	1	1	0

-6 = 16-6 = +10 => 1010 

1	0	1	0

Somme e sottrazioni in base 2

+	**0**	**1**
0	0	1
1	1	0 (riporto 1)

-	**0**	**1**
0	0	1 (prestito 1)
1	1	0

Esempi

```
 10111011 +
 11010011
----------
110001110
```

```
 10111011 +
        1
----------
 10111100
```

```
 10111011 -
 10000101
----------
 00110110
```

```
 10000000 -
        1
----------
 01111111
```

Complemento a 1

Dato un numero binario $(X)_2$, il

complemento a 1 di $(X)_2$ è un numero binario $(Y)_2$ che si ottiene invertendo i bit di $(X)_2$.

Esempio:

il complemento a 1 di $(00111011)_2$ è $(11000100)_2$

Complemento a 2

- Fatti:
  - il bit più significativo di un numero positivo in complemento a due è sempre 0
  - il bit più significativo di un numero negativo in complemento a due è sempre 1
  - Considerando n bit, questa codifica permette di rappresentare i numeri interi da -2^{n-1} a $(2^{n-1}-1)$
- Problema: dato un numero intero codificato in complemento a due, determinare il corrispondente valore in base 10

Complemento a 2

Problema: Convertire un numero codificato in complemento a 2 $(X)_2$ nel corrispondente valore decimale.

Soluzione:

Se $(X)_2 >= 0$
    stesso procedimento della codifica modulo e segno
Se $(X)_2 < 0$

1. Calcolare il complemento a 1 di $(X)_2$ e chiamarlo $(Y)_2$
2. Sommare 1 a $(Y)_2$, convertire in decimale il risultato, e chiamarlo $(Z)_{10}$
3. Il valore corrispondente è $-(Z)_{10}$

Esempi

Caso numero negativo

Il valore decimale di $(1011)_2$ è $(-5)_{10}$ infatti...

complemento a 1 $0100 +$

$$\begin{array}{r} 1 \\ \hline 0101 \end{array}$$

e convertendo $(0101)_2$ si ottiene 5.

Caso numero positivo

Il valore decimale di $(0011)_2$ è $(+3)_{10}$ infatti...

convertendo $(011)_2$ si ottiene 3.

There are only 10 types
of people in the world:

those who understand binary
and those who don't.

Compressione dei dati

- Finora non abbiamo valutato il costo e l'efficienza delle codifiche scelte.
- La compressione dei dati permette di ridurre il numero di bit necessari alla codifica di un insieme di elementi di informazione.
  - costo di memorizzazione minore
  - trasmissione dati più veloce

Compressione dei dati

- due tipi di algoritmi di compressione
  - lossless (senza perdita di informazione)
    - tutta l'informazione è importante e non si può perdere!
    - Es: archivi compressi zip e rar, immagini tiff
  - lossy (con perdita di informazione)
    - si è disposti a perdere informazione pur di ottenere un maggior tasso di compressione
    - Es.: formati per immagini (JPEG, GIF,...), formati audio (MP3), formati video (famiglia MPEG)

Esempio (lossless)

- supponiamo di avere l'alfabeto A={A,C,G,T} e una successione S di 1.000.000 di caratteri scelti in A.
- Possibile codifica binaria dell'alfabeto: A=00, C=01, G=10, T=11 => la codifica di S prevede l'uso di 2.000.000 di bit.
- Altra possibilità: assegnare codici binari (non ambigui!!) di lunghezza variabile agli elementi dell'alfabeto, in base alla loro frequenza relativa rispetto a S (più la frequenza è alta, più il codice sarà corto). Se A si presenta in S il 50% dei casi, C il 25%, G e T il 12.5%, allora, per esempio,

A=0, C=10, G=110, T=111

in questo caso, la codifica di S risulta più corta, infatti

(1*0.5+2*0.25+3*0.125+3*0.125) bit/carattere * 1.000.000 caratteri = 1.750.000 bit