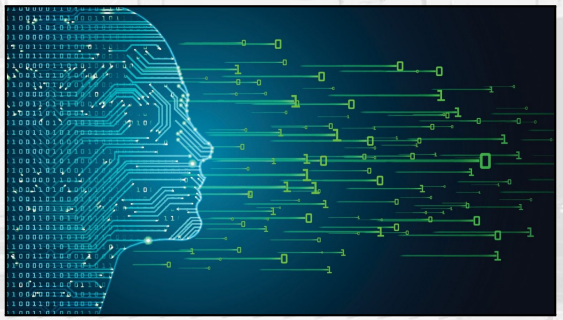
Programmazione Informatica
2. Rappresentazione delle informazioni nel calcolatore

Riccardo Zamolo
rzamolo@units.it

Università degli Studi Trieste
Ingegneria Civile e Ambientale



A.A. 2023-24

- *Bit* (BInary digiT): unità minima di informazione (binaria):
  - due possibili stati logici: 0/1, vero/falso, on/off, +/-, ecc.;
  - rappresentabile fisicamente in un calcolatore tramite una tensione o una corrente elettrica (alta/bassa), una direzione di magnetizzazione o polarizzazione, un'intensità luminosa, ecc.

- Qualsiasi informazione in forma discreta (numeri, testi, immagini, suoni, ecc.) è rappresentabile come sequenza di un certo numero finito di bit utilizzando una determinata *codifica*:

RGB 24 bit

$\longleftrightarrow$

(BMP)

0101101000101011
0110101000111101
1110100101000111
1101000101010101...

- Le operazioni algebriche utilizzate per trattare quantità binarie come i bit sono descritte dall'algebra Booleana.

- Numeri naturali $\mathbb{N}$: rappresentazione posizionale in base 2 dove ogni cifra binaria (0/1) è rappresentata da un bit:

$$(13)_{10} = 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = (1101)_2 \quad \Rightarrow \quad \underbrace{\boxed{1}\boxed{1}\boxed{0}\boxed{1}}_{4 \text{ bit}}$$

- Nei calcolatori si utilizza un numero fisso k di bit per ogni numero, ad esempio 8, 16, 32, 64: sempre potenze di 2. Il range dei numeri naturali rappresentabili va perciò da 0 a $2^k - 1$ e sono quindi 2^k in tutto.
- *Overflow*: si verifica quando il risultato di un'operazione tra due numeri esce dalla capacità di rappresentazione scelta. Ad esempio: somma o prodotto di due numeri, utilizzando 4 bit, il cui risultato eccede $2^4 - 1 = 15$:

1 1 1 0	14	+	
0 0 1 0	2	=	
1 0 0 0 0	16 → 0		
0			

1 0 1 0	10	×
0 0 1 1	3	=
1 0 1 0	10	+
1 0 1 0	20 → 4	=
4		
1 1 1 1 0	30 → 14	
14		

- Numeri relativi $\mathbb{Z}$: è necessario memorizzare anche l'informazione sul segno (+/-). Si potrebbe impiegare il bit più a sinistra per il segno, per esempio 0 (segno +) o 1 (segno −):

$$-6 = \boxed{1}\underbrace{\boxed{1}\boxed{1}\boxed{0}}$$

bit modulo (6)

bit segno (-)

  - questa soluzione è scomoda perchè la somma di due numeri di segno diverso non si può più effettuare sommando le cifre della loro rappresentazione in base 2 (basta sommare due numeri opposti: non si otterrebbe 0);
  - in un calcolatore sarebbe quindi necessario utilizzare circuiti diversi per la somma di numeri di segno diverso.

- Rappresentazione con *complemento a 2*: per mantenere lo stesso procedimento di somma indipendentemente dal segno, è sufficiente sommare una costante B ai numeri negativi, fissato un numero totale di bit da impiegare. Un numero intero negativo $q < 0$ è quindi codificato dal numero intero positivo $B + q$.

- Utilizzando k bit è ragionevole utilizzare metà dei numeri a disposizione per i numeri positivi e l'altra metà per quelli negativi: $\Rightarrow B = 2^k$. I numeri positivi saranno quelli con il bit più significativo (quello più a sinistra) pari a 0, quelli negativi avranno il bit più significativo pari a 1. Esempio con $k = 4$ bit ($B = 2^4 = 16$):

Bit	RP_2	C_2	Bit	RP_2	C_2
0 0 0 0	0	+0	1 0 0 0	$8 = B - 8$	−8
0 0 0 1	1	+1	1 0 0 1	$9 = B - 7$	−7
0 0 1 0	2	+2	1 0 1 0	$10 = B - 6$	−6
0 0 1 1	3	+3	1 0 1 1	$11 = B - 5$	−5
0 1 0 0	4	+4	1 1 0 0	$12 = B - 4$	−4
0 1 0 1	5	+5	1 1 0 1	$13 = B - 3$	−3
0 1 1 0	6	+6	1 1 1 0	$14 = B - 2$	−2
0 1 1 1	7	+7	1 1 1 1	$15 = B - 1$	−1

RP_2 = rappresentazione posizionale in base 2
C_2 = rappresentazione con complemento a 2

- Utilizzando k bit, sono rappresentabili con complemento a 2 gli interi da $\min = -2^{k-1}$ a $\text{MAX} = 2^{k-1} - 1$, sempre 2^k in tutto.
- Si osserva che la rappresentazione con complemento a 2 di un intero negativo $(-n)$ si scrive $2^k - n = [(2^k - 1) - n] + 1$, cioè i bit di $(-n)$ si ottengono negando ciascun bit di n ed aggiungendo 1.

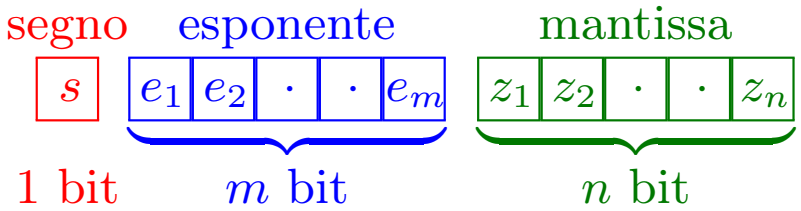
- La somma $s = a + b$ con complemento a 2 è corretta in ogni caso:
  - segni positivi: $a, b \geq 0$. È la "solita" somma, con possibile overflow se il risultato eccede il limite superiore MAX: utilizzando $k = 4$ bit, $6+7 = 13$ eccede il limite $\text{MAX} = 7$ e verrebbe interpretato come -3 ($13 = B - 3$);
  - segni opposti: $a \geq 0$, $b < 0$. Il range di rappresentabilità con complemento a 2 richiede inoltre $a \leq \text{MAX}$, $b \geq \text{min}$, perciò la somma s soddisfa $\text{min} \leq s < \text{MAX}$ ed è quindi sempre rappresentabile con complemento a 2 senza alcun overflow. La somma delle rappresentazioni è quindi:
  $$(a) + (B + b) = B + (a + b) = B + s$$
  la cui interpretazione con complemento a 2 coincide sempre con il risultato voluto s: se $s \geq 0$ si verifica overflow che "cancella" B lasciando s, se $s < 0$ il risultato viene correttamente interpretato con complemento a 2 con segno negativo;
  - segni negativi: $a, b < 0$. Per il range di rappresentabilità con complemento a 2 si ha $a, b \geq \text{min}$, perciò $2\text{min} \leq s < 0$ e vi è quindi la possibilità di overflow. La somma delle rappresentazioni è:
  $$(B + a) + (B + b) = 2B + (a + b) = B + \underbrace{(B + s)}_{\geq 0} \overset{\text{o.f.}}{\equiv} B + s$$
  che viene interpretato correttamente con complemento a 2 con segno negativo se $s \geq \text{min}$, altrimenti si verifica overflow: utilizzando $k = 4$ bit, $(-3) + (-6) = -9$ risulta minore del limite inferiore $\text{min} = -8$ e verrebbe interpretato come $+7 : B - 9 = 7$ dalla formula, oppure dalla somma delle rappresentazioni $(-3)_{\text{C}_2} + (-6)_{\text{C}_2} = (13)_{\text{RP}_2} + (10)_{\text{RP}_2} = (23)_{\text{RP}_2} \overset{\text{o.f.}}{\equiv} 7$.

- Rappresentazione approssimata in virgola mobile in base 2 di un numero $x \in \mathbb{R}$:

$$x \approx (-1)^s \cdot (1\ .\ z_1\ z_2 \ldots z_n)_2 \cdot 2^{(e_1\ e2\ \ldots\ e_m)_2 - cost}$$

  dove s, z_i e e_i sono cifre binarie (0/1) e $cost = 2^{m-1} - 1$ è un intero positivo.

- Rappresentazione standardizzata:



- Dato il numero finito di bit a disposizione, non tutti i numeri reali possono essere rappresentati esattamente: ci si accontenta di loro approssimazioni: es. $\pi, e, 0.2_{10} = 0.0011001100110011\ldots_2$.
- Standard IEEE 754:
  - a 32 bit (*single precision*): $m = 8$ bit per l'esponente e $n = 23$ bit per la mantissa, $cost = 127$ (notazione *a eccesso 127*);
  - a 64 bit (*double precision*): $m = 11$ bit per l'esponente e $n = 52$ bit per la mantissa, $cost = 1023$ (notazione *a eccesso 1023*);

- Range rappresentabile:

IEEE 754-32	da $\pm 2^{-126}$	a $\pm(2-2^{-23})\cdot 2^{127}$
(single precision)	da $\pm 1.18\cdot 10^{-38}$	a $\pm 3.40\cdot 10^{38}$
IEEE 754-64	da $\pm 2^{-1022}$	a $\pm(2-2^{-52})\cdot 2^{1023}$
(double precision)	da $\pm 2.23\cdot 10^{-308}$	a $\pm 1.80\cdot 10^{308}$

- Minima differenza tra $x = \pm(1\,.\,z_1\ldots z_n)_2\cdot 2^{(e_1\ \ldots\ e_m)_2 - cost}$ e $x+\varepsilon$:

$$\varepsilon = \pm(0.\underbrace{00\ldots01}_{n\text{ cifre}})_2\cdot 2^{(e_1\ \ldots\ e_m)_2 - cost}$$

- Minima differenza relativa, detta *epsilon di macchina*:

$$\epsilon = \left|\frac{\varepsilon}{x}\right| = \frac{(0.\overbrace{00\ldots01}^{n\text{ cifre}})_2}{(1\,.\,z_1\ldots z_n)_2} = \frac{2^{-n}}{(1\,.\,z_1\ldots z_n)_2} < 2^{-n}$$

  - IEEE 754-32 (single precision): $\epsilon = 2^{-23} = 1.19\cdot 10^{-7} \Rightarrow$ circa 7 cifre significative (in base 10);
  - IEEE 754-64 (double precision): $\epsilon = 2^{-52} = 2.22\cdot 10^{-16} \Rightarrow$ circa 16 cifre significative (in base 10);

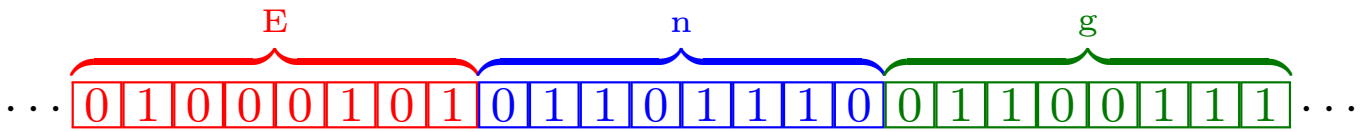
- Particolari combinazioni di mantissa ed esponente sono impiegate per rappresentare $\pm\infty$ e NaN (Not a Number: $\pm 0/\pm 0$, $\pm\infty/\pm\infty$, $\pm\infty\times\pm 0$).

- Il bit è la più piccola quantità d'informazione (0/1), mentre 1 byte = 8 bit è la più piccola quantità di memoria indirizzabile in un calcolatore.
- *File*: sequenza di byte memorizzata su qualsiasi supporto, con un inizio ed una dimensione ben definiti. È un contenitore di informazioni. L'organizzazione delle informazioni all'interno del file dipende dal *formato di file* scelto.
- In ambito informatico sarebbe più corretto utilizzare le potenze di 2 per definire i multipli del byte: con n bit si possono indirizzare 2^n celle di memoria, una per ciascun numero rappresentabile (le possibili combinazioni di n bit).
- Tuttavia, è entrato nell'uso comune l'impiego delle potenze di 10 del Sistema Internazionale (SI) per definire i multipli del byte:

Prefissi SI			Prefissi binari		
kilobyte	kB	10^3	*kibibyte*	KiB	2^{10}
megabyte	MB	10^6	*mebibyte*	MiB	2^{20}
gigabyte	GB	10^9	*gibibyte*	GiB	2^{30}
terabyte	TB	10^{12}	*tebibyte*	TiB	2^{40}
petabyte	PB	10^{15}	*pebibyte*	PiB	2^{50}

- I multipli SI del byte **non coincidono** con quelli binari ma ci si avvicinano, è un'approssimazione basata sul fatto che $2^{10} = 1024 \approx 10^3$.

- *Codifica*: procedimento che permette di stabilire una *corrispondenza biunivoca* tra gli elementi di un insieme I e gli elementi di un sottoinsieme $\mathcal{Q}$ delle parole di un alfabeto $\mathcal{A}$.
- Finora la codifica dei numeri da rappresentare nel calcolatore è stata definita in maniera *implicita*: la codifica era definita dalle rappresentazioni posizionali/in virgola mobile in base 2 con un determinato ordine, $\mathcal{Q}$ era l'insieme delle parole di k caratteri binari (sequenze di k bit) ed I era un certo insieme finito di numeri, determinato come conseguenza delle precedenti scelte.
- Nel caso della rappresentazione del testo, la codifica va definita *esplicitamente*: ad ogni carattere del testo che vogliamo rappresentare va associata una e una sola sequenza di k bit. Questa corrispondenza biunivoca può essere rappresentata da una tabella, per esempio.
- Fissata la codifica, un testo può essere rappresentato in un calcolatore come un file contenente la sequenza delle codifiche (cioè dei bit/byte) dei caratteri del testo da rappresentare. Esempio con 1 byte/carattere:



- A seconda del formato di file impiegato, possono essere incluse altre informazioni relative ad impaginazione, colore e font dei caratteri, ecc.

- ASCII (American Standard Code for Information Interchange):
  - *base*: nato per rappresentare i caratteri di una tastiera composta da $26 \cdot 2$ lettere maiuscole e minuscole, $10 \cdot 2$ numeri e simboli associati, e 39 altri caratteri per un totale di 111 caratteri. Sono suficienti quindi $k = 7$ bit poichè $2^7 = 128$.

	MSB							
LSB	000	001	010	011	100	101	110	111
0000	NUL	DLE	SP	0	@	P	‘	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	’	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	|
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL

LSB: bit meno significativi, MSB: bit più sigificativi.

  - *esteso*: utilizza $k = 8$ bit $= 1$ byte per estendere il numero di caratteri rappresentabili a $2^8 = 256$. Non è una codifica unica, ma esistono diversi standard denominati ISO 8859-n, con $1 < n < 16$.

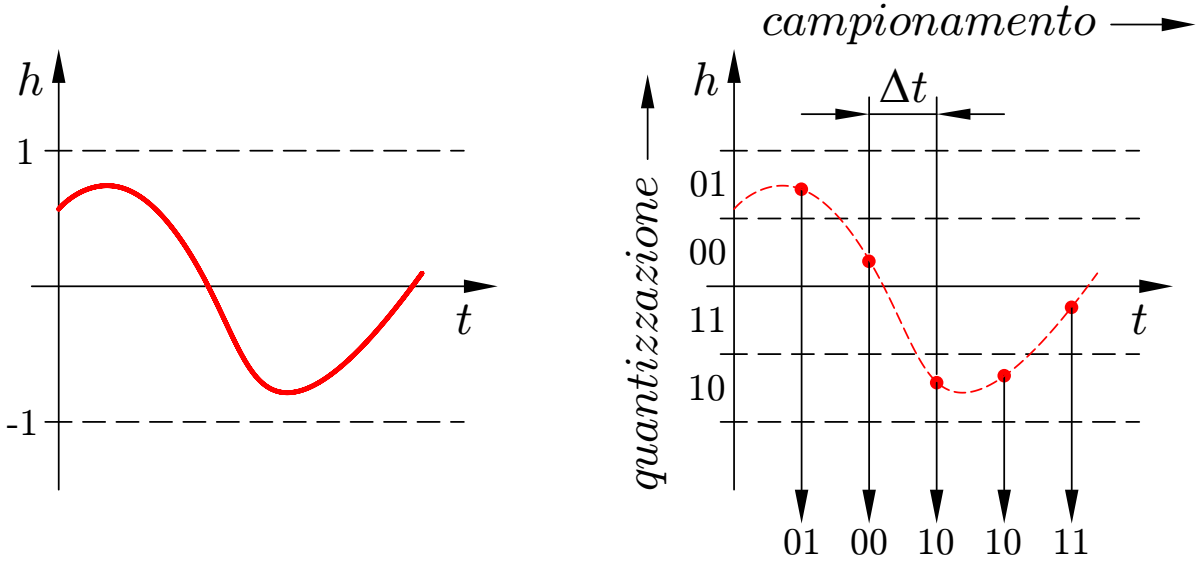
- UTF (Unicode Transformation Format):
  - UTF-32: codifica a lunghezza fissa utilizzando 32 bit (4 byte);
  - UTF-8 e UTF-16: *codifica a lunghezza variabile* utilizzando da 1 a 4 byte. Ciò permette un impiego ottimale dello spazio occupato (la dimensione del file in bytes) avendo comunque a disposizione un set amplissimo di caratteri. Alcuni caratteri in UTF-8:

Unicode code point	Carattere	UTF-8 (esadecimale)	Nome	Unicode code point	Carattere	UTF-8 (esadecimale)	Nome
U+2200	∀	e2 88 80	FOR ALL	U+1F61E	😞	f0 9f 98 9e	DISAPPOINTED FACE
U+2201	∁	e2 88 81	COMPLEMENT	U+1F61F	😟	f0 9f 98 9f	WORRIED FACE
U+2202	∂	e2 88 82	PARTIAL DIFFERENTIAL	U+1F620	😠	f0 9f 98 a0	ANGRY FACE
U+2203	∃	e2 88 83	THERE EXISTS	U+1F621	😡	f0 9f 98 a1	POUTING FACE
U+2204	∄	e2 88 84	THERE DOES NOT EXIST	U+1F622	😢	f0 9f 98 a2	CRYING FACE
U+2205	∅	e2 88 85	EMPTY SET	U+1F623	😣	f0 9f 98 a3	PERSEVERING FACE
U+2206	∆	e2 88 86	INCREMENT	U+1F624	😤	f0 9f 98 a4	FACE WITH LOOK OF TRIUMPH
U+2207	∇	e2 88 87	NABLA	U+1F625	😥	f0 9f 98 a5	DISAPPOINTED BUT RELIEVED
U+2208	∈	e2 88 88	ELEMENT OF	U+1F626	😦	f0 9f 98 a6	FROWNING FACE WITH OPEN
U+2209	∉	e2 88 89	NOT AN ELEMENT OF	U+1F627	😧	f0 9f 98 a7	ANGUISHED FACE
U+220A	∊	e2 88 8a	SMALL ELEMENT OF	U+1F628	😨	f0 9f 98 a8	FEARFUL FACE
U+220B	∋	e2 88 8b	CONTAINS AS MEMBER	U+1F629	😩	f0 9f 98 a9	WEARY FACE
U+220C	∌	e2 88 8c	DOES NOT CONTAIN AS MEMBER	U+1F62A	😪	f0 9f 98 aa	SLEEPY FACE
U+220D	∍	e2 88 8d	SMALL CONTAINS AS MEMBER	U+1F62B	😫	f0 9f 98 ab	TIRED FACE
U+220E	∎	e2 88 8e	END OF PROOF	U+1F62C	😬	f0 9f 98 ac	GRIMACING FACE
U+220F	∏	e2 88 8f	N-ARY PRODUCT	U+1F62D	😭	f0 9f 98 ad	LOUDLY CRYING FACE
U+2210	∐	e2 88 90	N-ARY COPRODUCT	U+1F62E	😮	f0 9f 98 ae	FACE WITH OPEN MOUTH
U+2211	∑	e2 88 91	N-ARY SUMMATION	U+1F62F	😯	f0 9f 98 af	HUSHED FACE
U+2212	−	e2 88 92	MINUS SIGN	U+1F630	😰	f0 9f 98 b0	FACE WITH OPEN MOUTH AND
U+2213	∓	e2 88 93	MINUS-OR-PLUS SIGN	U+1F631	😱	f0 9f 98 b1	FACE SCREAMING IN FEAR
U+2214	∔	e2 88 94	DOT PLUS	U+1F632	😲	f0 9f 98 b2	ASTONISHED FACE

  - sono rappresentabili circa 1 milione di caratteri diversi;
  - i primi 128 caratteri delle codifiche UTF coincidono con quelli della codifica ASCII base.

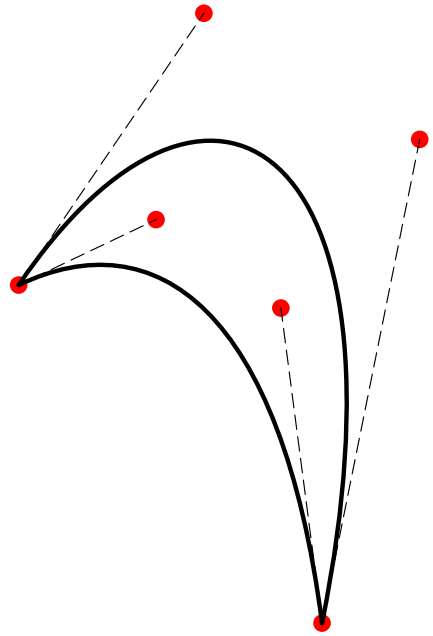
- *Segnale*: grandezza fisica che varia nel tempo, matematicamente espressa da una funzione $h(t) : \mathbb{R} \to \mathbb{R}$. I segnali possono essere di tipo acustico, elettrico, elettromagnetico, meccanico, ecc.
- Un segnale fisico è tipicamente *analogico*: h può assumere qualsiasi valore (reale) ad ogni istante di tempo (anch'esso reale). Il calcolatore è invece digitale, cioè opera su quantità discrete. È quindi necessario eseguire due operazioni per poter rappresentare un segnale analogico mediante informazioni discrete:
  - *campionamento*: il segnale h viene registrato ad intervalli di tempo regolare Δt, ossia $h_i = h(i \cdot \Delta t)$ e partendo da $t = 0$ per semplicità. La frequenza di campionamento è $f_s = 1/\Delta t$. La frequenza di campionamento dev'essere sufficientemente alta ($f_s > 2B$) affinchè non ci sia perdita di informazione tra segnale analogico, limitato in banda (B), e segnale campionato (teorema di Nyquist–Shannon), altrimenti si verifica *aliasing*.
  - *quantizzazione*: il valore scalare di ogni campione h_i viene rappresentato in maniera approssimata utilizzando un numero finito k di bit. Supponendo per semplicità che $-1 < h_i < 1$, tale intervallo $[-1, 1]$ viene suddiviso in $N = 2^k$ livelli (non necessariamente uniformi) e ad ognuno di questi livelli si associa una e una sola combinazione di k bit (codifica). L'informazione digitale rappresentativa del campione h_i è quindi data dalla codifica del livello in cui il campione h_i cade. C'è sempre perdita di informazione e l'errore introdotto diminuisce ovviamente all'aumentare del numero di bit impiegati.

- Esempio di conversione analogico-digitale con quantizzazione a 2 bit (4 livelli):

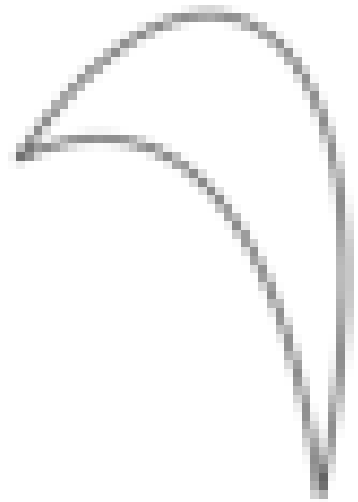


- Formati di file più comuni per i segnali acustici (audio):
  - Compact Disc Digital Audio (CDDA): *senza perdita d'informazione*, due canali, $f_s = 44100$ Hz in quanto la massima frequenza udibile dall'orecchio umano è circa $B = 20000$ Hz, $k = 16$ bit per canale con codifica Linear Pulse-Code Modulation (LPCM) cioè con $2^k = 65536$ livelli uniformi $\Rightarrow$ $2 \cdot f_s \cdot k = 1411200$ bit/s $= 176.4$ kB/s ≈ 10.6 MB/min;
  - MP3 (MPEG Audio Layer III): *compresso con perdita d'informazione* mediante l'utilizzo di modelli psicoacustici per ridurre efficacemente lo spazio impiegato e mantenere contemporaneamente una buona qualità di riproduzione all'orecchio umano. Tipici *bitrate* vanno da 32 a 320 kbit/s (4-40 kB/s) e possono essere costanti (Constant Bit Rate, CBR) o variabili (Variable Bit Rate, VBR).

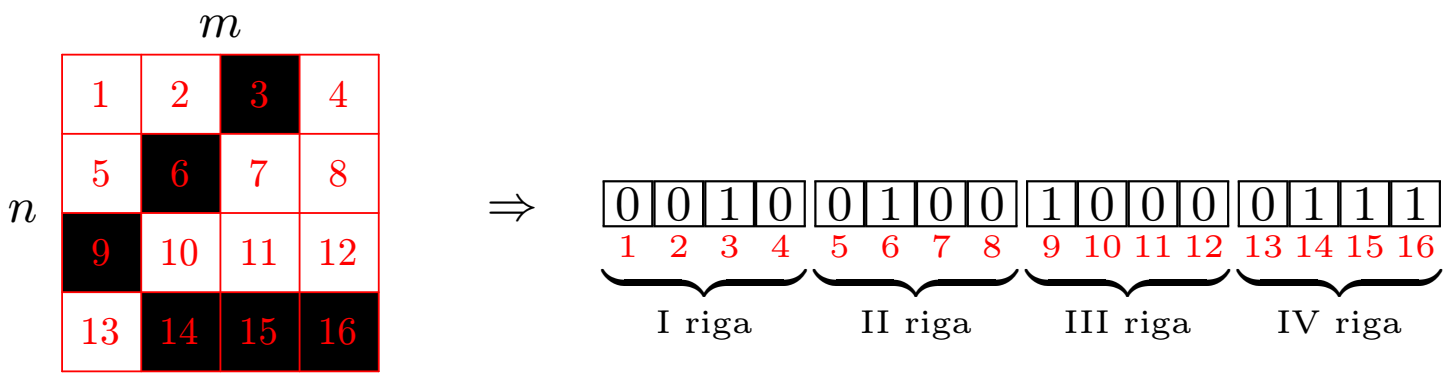
- Immagine *vettoriale*: è definita dalle proprietà di entità geometriche definite a priori (linee, segmenti, poligoni, cerchi, curve, caratteri con determinati font, ecc.). Un'immagine vettoriale può essere perciò ridotta e/o ingrandita a piacere senza alcuna perdita di dettaglio. Viene tipicamente usata da chi si occupa di grafica. Formati di file più utilizzati: Portable Document Format (PDF), Encapsulated PostScript (EPS), Scalable Vector Graphics (SVG) ed i formati di disegno tecnico: AutoCAD Drawing Exchange Format (DXF), ecc.



- Immagine *raster*: è definita da una griglia o *raster* di elementi detti *pixel*, ognuno caratterizzato da un colore definito secondo un modello che è a tutti gli effetti una codifica. Il numero di righe e di colonne della griglia definiscono la *risoluzione* dell'immagine, mentre il numero bit utilizzati per il colore di ogni pixel definisce la *profondità del colore*. Formati di file più utilizzati: Bitmap (BMP), JPEG (JPG), Portable Network Graphics (PNG).



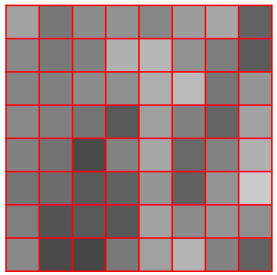
- Nel caso di immagini in bianco e nero (B/N), ogni pixel può assumere solo due valori, perciò basta l'informazione di un bit per identificarlo: 0 (bianco) e 1 (nero), per esempio ⇒ codifica B/N.
- Un'immagine B/N con risoluzione m colonne × n righe può essere quindi rappresentata in un calcolatore come una sequenza di $m \cdot n$ bit in un determinato ordine (per righe/colonne):



- Al di fuori dell'informatica il termine B/N comprende anche le immagini in *scala di grigi*, ad es. le fotografie su pellicola analogica B/N.
- Analogamente ad un generico segnale analogico, la rappresentazione digitale di una fotografia/immagine analogica in scala di grigi passa attraverso due operazioni:
  - *media spaziale*, simile (ma non uguale) al campionamento;
  - quantizzazione.

- Media spaziale: viene eseguita implicitamente (fisicamente), per ogni pixel, dal sensore del dispositivo di acquisizione impiegato (scanner, macchina fotografica). Differisce dal campionamento perchè si ottiene un valore medio e non il valore puntuale. Esempio con risoluzione 8×8:

$\Longrightarrow$
media spaziale



- Quantizzazione: il livello di grigio (analogico) di ogni pixel viene rappresentato in maniera approssimata utilizzando un numero finito k di bit, cioè utilizzando 2^k livelli di grigio, scegliendo il livello di grigio più vicino. Esempio con 4 livelli di grigio ($k = 2$ bit):

256×256

$\Longrightarrow$
quantizzazione
a 2 bit

256×256

- Un'immagine in B/N si ottiene quindi con una quantizzazione a 1 bit.
- Un'immagine $m \times n$ con 2^k livelli di grigio è rappresentata da una sequenza di $m \cdot n \cdot k$ bit.

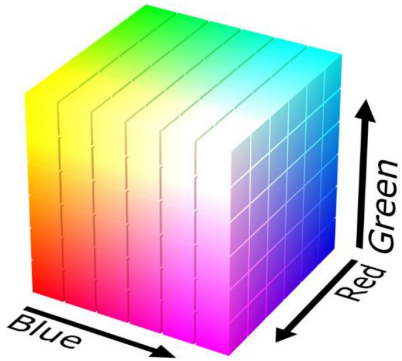
- Se la quantizzazione è fatta "ingenuamente", cioè operando su ciascun pixel indipendentemente dai pixel adiacenti, l'errore di quantizzazione si accumula su vaste zone dell'immagine portando ad una rappresentazione di cattiva qualità. Esempio: vecchie fotocopiatrici analogiche B/N.
- *Dithering*: tecnica che permette di distribuire l'errore di quantizzazione sui pixel adiacenti in maniera da garantire un errore di quantizzazione mediamente nullo. Si applica anche nella quantizzazione dei segnali generici (es. audio). Esempio con quantizzazione a 1 bit (B/N) e risoluzione 256×256:

Scala di grigi

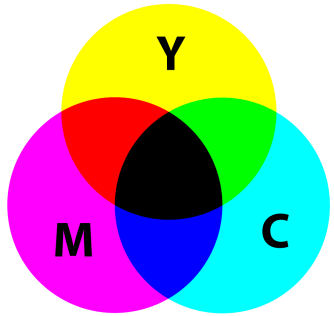
B/N

B/N con dithering

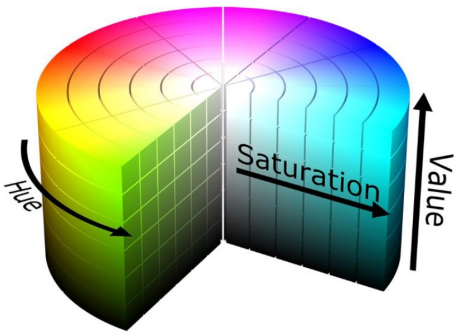
- RGB (Red Green Blue): modello *additivo* dove le emissioni luminose dei tre colori principali rosso, verde e blu vengono sommate. Il bianco è quindi ottenuto dalla somma di tutti e tre i colori R, G e B. Utilizzato nei display di dispositivi elettronici come TV, computer, telefoni cellulari, ecc.



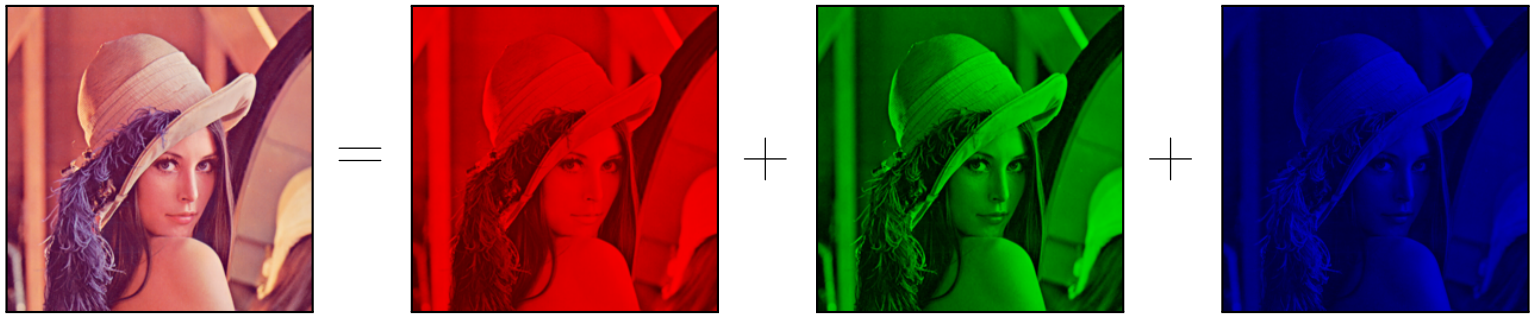
- CMY[K] (Cyan, Magenta, Yellow, [Black]): modello *sottrattivo* dove i tre colori principali ciano, magenta e giallo vengono miscelati (pensando a vernici) o sovrapposti (pensando a lucidi colorati). È sottrattivo perchè ogni colore aggiunto sottrae una parte di spettro alla luce biance incidente: il nero è ottenuto dalla composizione di tutti e tre i colori C, M e Y. L'eventuale quarto parametro K (nero) è utilizzato per motivi di convenienza pratica in fase di stampa.



- HSV (Hue, Saturation, Value) o l'equivalente HSL (Hue, Saturation, Lightness): modello nel quale un colore è definito da tonalità (hue), saturazione (saturation) ed un terzo parametro che può essere brillanza (brightness, value) oppure tono (lightness). Non è nè additivo nè sottrattivo.

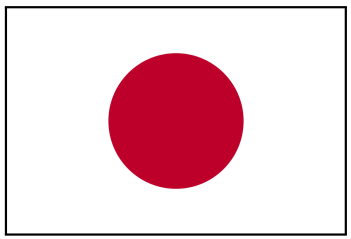


- Il modello RGB è ovviamente quello più utilizzato per la rappresentazione digitale delle immagini a colori.
- Indipendentemente dal modello di colore impiegato, è sempre necessario fornire una terna di numeri per definire un colore: (R, G, B), (C, M, Y) oppure (H, S, V). Esistono opportune formule per passare da un modello di colore all'altro.
- L'informazione di un'immagine a colori può essere quindi rappresentata da tre immagini, o *canali*, in scala di grigi. Esempio con modello RGB:

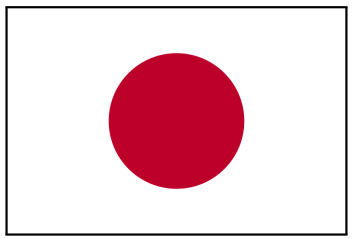


- Ogni canale ha la medesima risoluzione ed il medesimo tipo di quantizzazione, cioè il numero k di bit utilizzati per ogni pixel per ogni canale è lo stesso. Un'immagine $m \times n$ con 2^k livelli per ciascun canale è quindi rappresentata da una sequenza di $3 \cdot m \cdot n \cdot k$ bit.
- La profondità del colore, cioè il numero di bit utilizzati per il colore di ogni pixel, è quindi $3 \cdot k$. La scelta più popolare, nota come profondità *Truecolor*, è quella di usare 8 bit per canale, per un totale di 24 bit (3 byte): si hanno a disposizione $2^{24} = 16,777,216$ colori diversi.

- La memorizzazione diretta di immagini ad alta risoluzione richiede molto spazio: ad es. un'immagine con risoluzione 3840×2160 (4K UHD) con 24 bit di profondità di colore occuperebbe $3840 \cdot 2160 \cdot 24$ bit ≈ 25 MB.
- Esistono varie tecniche di *compressione* per ridurre lo spazio richiesto per la memorizzazione, con e senza perdita d'informazioni.
- Compressione *senza perdita di informazioni* o *lossless*: l'informazione originale può essere completamente ricostruita a partire dai dati compressi. Nelle immagini in formato PNG, per esempio, viene utilizzata una tecnica di compressione lossless che si basa sulla ricerca di sequenze ripetute di dati. Funziona quindi bene su immagini con ampie aree dello stesso colore, meno bene su immagini generiche, ad esempio fotografie:



BMP
2.6 MB



PNG
20 kB
0.77%

BMP
1.4 MB

PNG
1.2 MB
86%

- Compressione *con perdita di informazioni* o *lossy*: parte dell'informazione originale viene persa irreversibilmente nella compressione. Nelle immagini in formato JPG, per esempio, viene utilizzata una tecnica di compressione lossy che si basa su un'ulteriore quantizzazione, oltre a quella del colore di partenza, delle componenti in frequenza di ciascun canale, preventivamente trasformato secondo un particolare modello di colore (luminanza e crominanza). L'accuratezza della quantizzazione, dalla quale dipende l'errore irreversibile di quantizzazione, può essere variata in funzione di un parametro di qualità Q, solitamente compreso tra 0 e 100.

BMP
1.4 MB

JPG, $Q = 50$
76 kB
5.4%

JPG, $Q = 10$
24 kB
1.7%

JPG, $Q = 5$
16 kB
1.1%