1.2 Concetti base dell'Informatica: Informazione

Insegnamento di Informatica

Elisabetta Ronchieri

Corso di Laurea di Economia, Universitá di Ferrara

I semestre, anno 2014-2015

Argomenti

Notazioni

- Rappresentazione numeri reali
- Rappresentazione dei caratteri

Altre informazioni

- Rappresentazione delle immagini
- Rappresentazione dei video
- Rappresentazione dei suoni

Argomenti

Notazioni

- Rappresentazione numeri reali
- Rappresentazione dei caratteri

Altre informazioni

- Rappresentazione delle immagini
- Rappresentazione dei video
- Rappresentazione dei suoni

Rappresentazione numeri reali

- Possono essere rappresentati:
  - virgola fissa;
  - virgola mobile o floating point.

Rappresentazione in virgola fissa

- Per un numero rappresentato da N cifre, usa 1 bit per il segno s, un insieme I di bit per la parte intera e un insieme D di bit per la parte decimale.

+/-	I-1	I-2	...	0	-1	-2	...	-D
Segno	Parte Intera				Parte Frazionaria			

- Per convertire un numero binario rappresentato in virgola fissa con 1 bit di segno, I bit di parte intera e D bit di parte decimale
  $N_2 = sc_{I-1}c_{I-2}\ldots c_0 c_{-1}c_{-2}\ldots c_{-D}$
- usa la formula:
  $N_{10} = (-1)^s \cdot (\sum_{i=0}^{i=I-1} c_i \cdot 2^i + \sum_{d=0}^{d=-D} c_d \cdot 2^d)$

Usa un certo numero di bit per rappresentare la parte intera.

Usa i rimamenti per la parte frazionaria.

Esempio di conversione di base in virgola fissa a base 10

Problema: convertire in decimale il numero binario in virgola fissa $(1010.101)_2$.

- Segno: 1;
- Parte intera: $0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 2$;
- Parte frazionaria:
  $1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3} = 0.5 + 0.125 = 0.625$;
- Risultato: $(1010.101)_2 = -(2 + 0.25) = -2.625$.

Problema: convertire in decimale il numero binario in virgola fissa $(101.01)_2$.

- Segno: 0;
- Parte intera: $1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 4 + 1 = 5$;
- Parte frazionaria: $0 \times 2^{-1} + 1 \times 2^{-2} = 0.25$;
- Risultato: $(101.01)_2 = (4 + 1 + 0.25) = 5.25$.

Conversione da base 10 a base 2 in virgola fissa

- La parte intera si codifica in binario con il segno.
- La parte frazionaria si converte come segue:
  - si moltiplica per 2;
  - si separa la parte intera dal risultato che é sempre pari a 0 o 1;
  - si ripete il procedimento nella parte frazionaria fino a quando il risultato della moltiplicazione é 1.000 . . ., oppure si raggiunge il numero di cifre binarie dedicate alla codifica della parte frazionaria.

Esempio di conversione da base 10 a base 2 in virgola fissa

Problema: convertire il numero -2.625 in virgola fissa.
Nota: si usa 1 bit per il segno, 4 bit per la parte intera, 3 bit per la parte frazionaria.

- Segno: 1;
- Parte intera: $2 = (0010)_2$;
- Parte frazionaria:
  $0.625 \times 2 = 1.250; 0.250 \times 2 = 0.500; 0.500 \times 2 = 1.000;$

Parte frazionaria	Parte frazionaria $\times 2$	Parte intera
0.625	$0.625 \times 2 = 1.250$	1
0.250	$0.250 \times 2 = 0.500$	0
0.500	$0.500 \times 2 = 1.000$	1

- Risultato: si separa 1, $(10010.101)_2 = -2.625$.

Rappresentazione in virgola fissa: svantaggi

- Rigiditá della posizione assegnata alla virgola.
- Sono fissi i bit assegnati per codificare la parte intera e la parte frazionaria.
- Impatta sulla precisione nel codificare i numeri.
- Maggiore é il numero di bit per codificare la parte intera, piú bassa sará la precisione nel codiciare i numeri piccoli.

Rappresentazione in virgola mobile

- Usa un bit per rappresentare il segno s.
- Usa altri bit per rappresentare la mantissa m.
- Usa altri per codificare l'esponente e.

s	e	e	...	e	m	m	...	m
Segno	esponente				mantissa			

- Dal 1985 adota lo standard IEEE[1] 754[2] nella base 2.
- IEEE 754 prevede due forme di codifica:
  1. normalizzata;
  2. denormalizzata.

[1] IEEE sta per Institute of Electrical and Electronics Engineers.

[2] Adottata da tutti i programmi e le componenti di calcolo dell'elaboratore.

Standard IEEE 754: normalizzata

- La codifica da binario a decimale dell'esponente assume un valore intero compreso tra 1 e 254 (estremi inclusi).
- La mantissa prevede una qualunque sequenza di bit.
- Numero corrispondente ad un float (32 bit) risulta:
  $N_{10} = (-1)^s \cdot 1.m \cdot 2^{e-127}$

  m é il numero intero ottenuto dalla codifica da binario a decimale della mantissa;

  e é il numero intero ottenuto dalla codifica da binario a decimale dell'esponente;

  s é il segno;

  la costante 127 é detta di polarizzazione.

Standard IEEE 754: normalizzata

- Il numero reale float piú piccolo in valore assoluto si ha quando la mantissa m é composta da tutti 0 e l'esponente e assume il valore minimo, ossia 1:
  $N_{min} = 1.(0000000000\ldots0)_2 \cdot 2^{1-127} = 1.0 \cdot 2^{-126}$
- Il numero reale float piú grande si ha quando la mantissa m é composta da tutti 1 e l'esponente e assume il valore massimo, ossia 254:
  $N_{max} = 1.(1111111111\ldots1)_2 \cdot 2^{254-127} =$
  $1.(1111111111\ldots1)_2 \cdot 2^{127} \approx 3.4 \times 10^{38}$

Standard IEEE 754: normalizzata

- La codifica da binario a decimale dell'esponente e assume un valore intero compreso tra 1 e 2046 (estremi inclusi).
- Numero corrispondente ad un double (64 bit) risulta: $N_{10} = (-1)^s \cdot 1.m \cdot 2^{e-1023}$

  m é il numero intero ottenuto dalla codifica da binario a decimale della mantissa;

  e é il numero intero ottenuto dalla codifica da binario a decimale dell'esponente;

  s é il segno;

  la costante 1023 é detta di polarizzazione.

Standard IEEE 754: normalizzata

- Il numero reale double piú piccolo in valore assoluto si ha quando la mantissa m é composta da tutti 0 e l'esponente e assume il valore minimo, ossia 1:
  $N_{min} = 1.(0000000000\ldots0)_2 \cdot 2^{1-1023} = 1.0 \cdot 2^{-1022}$
- Il numero reale double piú grande si ha quando la mantissa é composta da tutti 1 e l'esponente assume il valore massimo, ossia 2046:
  $N_{max} = 1.(1111111111\ldots1)_2 \cdot 2^{2046-1023} =$
  $1.(1111111111\ldots1)_2 \cdot 2^{1023} \approx 1.7 \times 10^{308}$

Standard IEEE 754: denormalizzata

- L'esponente é pari a 0.
- La mantissa assume una qualunque sequenza di bit tranne la sequenza di tutti 0, quindi va da $(0000000000\ldots1)_2$ a $(1111111111\ldots1)_2$.
- Permette di codificare numeri reali inferiori, in valore assoluto, al minimo codificabile in forma normalizzata, ossia 2^{-126} per i float, e 2^{-1022} per i double.
- Numero corrispondente ad un float (32 bit) risulta:
  $N_{10} = (-1)^s \cdot 0.m \cdot 2^{-126}$

  m é il numero intero ottenuto dalla codifica da binario a decimale della mantissa.
- Numero corrispondente ad un double (64 bit) risulta:
  $N_{10} = (-1)^s \cdot 1.m \cdot 2^{-1022}$

  m é il numero intero ottenuto dalla codifica da binario a decimale della mantissa.

Standard IEEE 754: denormalizzata

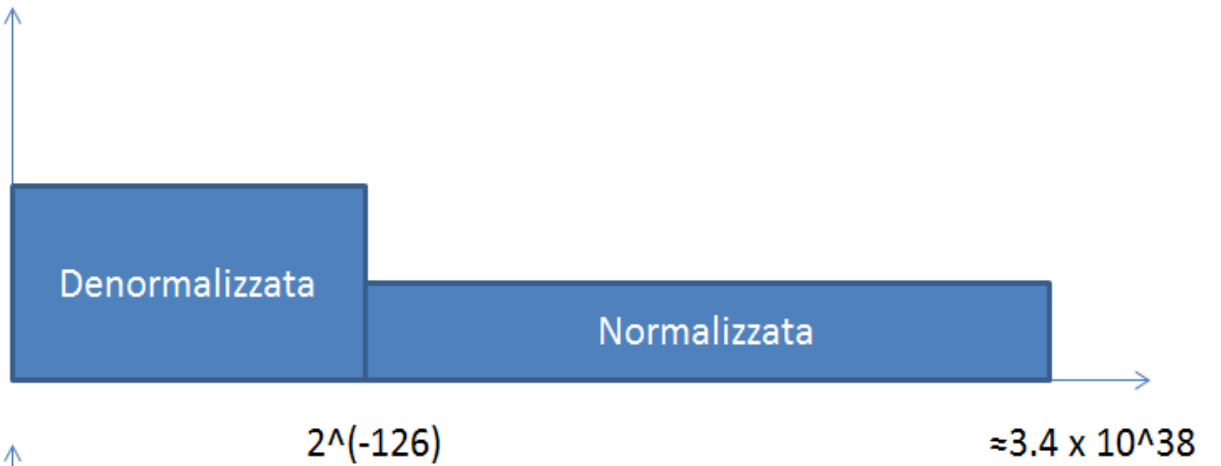
- Il numero reale float piú piccolo in valore assoluto si ha quando la mantissa m é pari a $(0000000000\ldots1)_2$:
  $N_{min} = 1.(0000000000\ldots1)_2 \cdot 2^{-126}$
- Il numero reale float piú grande si ha quando la mantissa m é composta da tutti 1:
  $N_{max} = 1.(1111111111\ldots1)_2 \cdot 2^{-126}$

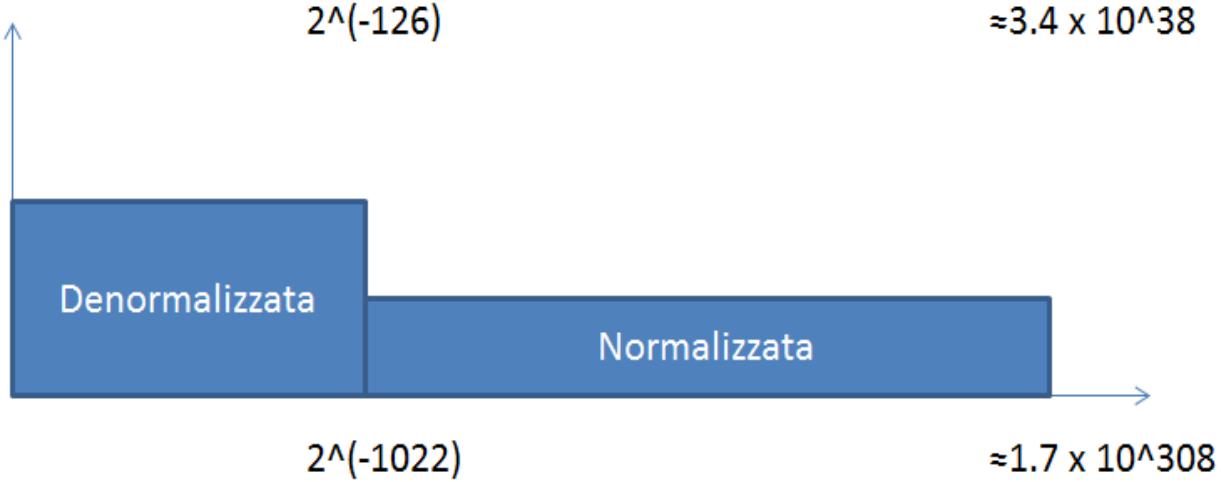
Standard IEEE 754: denormalizzata

- Il numero reale double piú piccolo in valore assoluto si ha quando la mantissa é pari a $(0000000000\ldots1)_2$, leggermente inferiore a 2^{-126}:
  $N_{min} = 1.(0000000000\ldots1)_2 \cdot 2^{-1022}$
- Il numero reale piú grande si ha quando la mantissa é composta da tutti 1, leggermente inferiore a 2^{-1022}:
  $N_{max} = 1.(1111111111\ldots1)_2 \cdot 2^{-1022}$

Standard IEEE 754

Intervalli di codifica del valore assoluto di un numero reale nelle forme normalizzate e denormalizzate.
I due schemi sono, partendo dall'alto, per float e per double rispettivamente.





Standard IEEE 754

- Normalizzato permette:
  - $1 \leq e \leq max$, con $max = 254$ per float e $max = 2046$ per double;
  - m qualunque sequenza di bit;
  - $N_{10} = (-1)^s \cdot 1.m \cdot 2^{e-p}$, con $p = 127$ per float e $p = 1023$ per double.
- Denormalizzato permette:
  - $e = 0$;
  - m qualunque sequenza di bit non 0;
  - $N_{10} = (-1)^s \cdot 1.m \cdot 2^{p}$, con $p = -126$ per float e $p = -1022$ per double.
- Permette di codificare altri valori:
  - ∞, quando $e = (11111111111\ldots1)_2 = 255$ e $m = 0$ ($N_{10} = \infty$);
  - NaN (Not a Number, valore indefinito), quando $e = 255$ e $m <> 0$ ($N_{10} = NaN$);
  - zero (valore particolare della forma denormalizzata), quando $e = 0$ e $m = 0$ ($N_{10} = 0$).

Conversione di un numero reale in base 10 nella base 2 in virgola mobile

Si consideri la forma normalizzata.

- Si effettua la conversione in complemento a 2, considerando la parte intera e la parte decimale.
- Si normalizza il numero binario in virgola mobile float calcolando separatamente s, m e e.
- m nel caso di float é composto da 23 cifre, altrimenti 52 nel caso di double.
- e nel caso di float é composto da 8 cifre, altrimenti 11 nel caso di double.
- Si mettono insieme segno, esponente e mantissa per la rappresentazione in virgola mobile.

Conversione di un numero reale in base 10 nella base 2 in virgola mobile

Esempio: convertire $(7.5)_{10}$ in base 2.

1. si converte la parte intera $(7)_{10} = (111)_2$;
2. si considera la parte frazionaria $(0.5)_{10} = (1.0)_2$;
3. si considera il numero binario ottenuto convertendo parte intera e parte frazionaria
   $(111.1)_2 = 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 + 1 \cdot 2^{-1}$

Conversione di un numero reale in base 10 nella base 2 in virgola mobile

Esempio: convertire $(7.5)_{10}$ in base 2.

4. si normalizza il numero binario ottenuto al passo precedente, spostando la virgola verso sinistra di $p = 2$ posizioni fino ad ottenere il numero binario nella forma normalizzata, il quale viene moltiplicato per la potenza di 2 elevato a 2:
   $(1.111)_2 \times 2^2$, dove
   $(1.111)_2 = 1 \cdot 2^0 + 1 \cdot 2^{-1} + 1 \cdot 2^{-2} + 1 \cdot 2^{-3}$
   $(1.111)_2 \times 2^2 = (111.1)_2$ (rivedere la rappresentazione in virgola fissa).
5. si calcola la mantissa m pari alla sequenza di bit dopo la virgola codificata in 23 bit $m = 11100000000000000000000$.
6. si calcola l'esponente e sommando a 2 il numero 127:
   $e = 2 + 127 = 129 = (10000001)_2$
7. si calcola il segno $s = 0$.

Rappresentazione dei caratteri

- Possiamo associare a ogni carattere (quale lettera minuscola, lettera maiuscola, vocale accentata e segno di interpunzione) un numero.
- I caratteri possono essere rappresentati:
  - in ASCII[3] standard: 1 carattere é rappresentato con 7 bit per un totale di 128 simboli rappresentabili (quali cifre, lettere maiuscole e lettere minuscole);
  - in ASCII estesa: 1 carattere é rappresentato con 8 bit rappresentabili fino a 256 simboli (i caratteri in piú sono usati per esempio per caratteri accentati);
  - in UNICODE: 1 carattere é rappresentato con 16 bit.

[3] ASCII sta per American Standard Code for Information Interchange.

Rappresentazione dei caratteri

bit		000	001	010	011	100	101	110	111
	esad.	0	1	2	3	4	5	6	7
0000	0	NUL	DLE	spz	0	@	P	`	p
0001	1	SOH	DC1	!	1	A	Q	a	q
0010	2	STX	DC2	"	2	B	R	b	r
0011	3	ETX	DC3	#	3	C	S	c	s
0100	4	EOT	DC4	$	4	D	T	d	t
0101	5	ENQ	NAK	%	5	E	U	e	u
0110	6	ACK	SYN	&	6	F	V	f	v
0111	7	BEL	ETB		7	G	W	g	w
1000	8	BS	CAN	(	8	H	X	h	x
1001	9	HT	EM	)	9	I	Y	i	y
1010	A	LF	SS	*	:	J	Z	j	z
1011	B	VT	ESC	+	;	k	[	k	{
1100	C	FF	FS	,	<	L	\	l	|
1101	D	CR	GS	-	=	M	]	m	}
1110	E	SOH	RS	.	>	N	^	n	~
1111	F	SI	US	/	?	O		o	DEL

ρ(A) = 100 0001

➔ 65

65 = 1 x 2^0 + 1 x 2^6

ρ({) = 111 1011

123=1 x 2^0+ 1 x 2^1 + 1 x 2^3 + 1 x 2^4 + 1 x 2^5 + 1 x 2^6 ➔ 123

La conversione da b=2 a b=10 va da dx verso sx.

Rappresentazione dei caratteri

Le parole sono sequenze di caratteri.

01101001 i	01101110 n	01100110 f	01101111 o	01110010 r
01101101 m	01100001 a	01110100 t	01101001 i	01100011 c
01100001 a				

Osservazioni

- ASCII é un codice accettato da tutti i computer.
- Usato dai tempi delle telescriventi durante la prima guerra modiale.
- Tuttavia non considera i caratteri internazionali di numerose lingue straniere.
- Esiste allora un'altra codifica, UNICODE che usa 2 byte per ciascun carattere.
- UNICODE riesce a rappresentare 65536 caratteri, permettendo di rappresentare tutti gli altri alfabeti.

Argomenti

Notazioni
- Rappresentazione numeri reali
- Rappresentazione dei caratteri

Altre informazioni
- Rappresentazione delle immagini
- Rappresentazione dei video
- Rappresentazione dei suoni

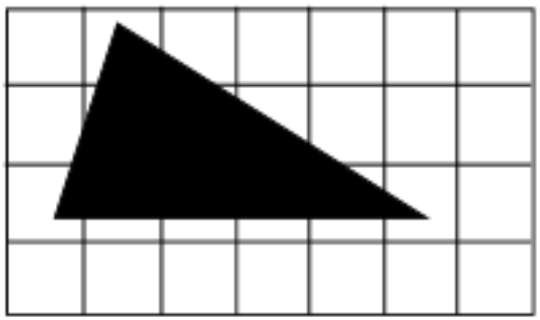
Rappresentazione delle immagini

- L'immagine viene suddivisa mediante una griglia di pixel (risoluzione).
- Ad ogni pixel é associata una rappresentazione binaria.
- Assegnando 1 bit ad ogni pixel si possono rappresentare solo immagini in bianco e nero.
- Per rappresentare immagini a diversi livelli di grigio o a colori, si associa ad ogni pixel una sequenza di bit:
  - con 8 bit per pixel, $2^8 = 256$ livelli di grigio;
  - con 24 bit per pixel, $2^{24} = 16777216$ milioni di colori (codifica RGB[4]).

[4]RGB sta per Red, Green e blue. Sistema di codici di colori internazionale le cui specifiche sono state definite dalla Commission internationale de l'éclairage (CIE).)

Rappresentazione delle immagini

2 colori
bianco: 0, nero: 1



un codice per ogni pixel

0_{22}	1_{23}	0_{24}	0_{25}	0_{26}	0_{27}	0_{28}
0_{15}	1_{16}	1_{17}	0_{18}	0_{19}	0_{20}	0_{21}
0_{8}	1_{9}	1_{10}	1_{11}	1_{12}	0_{13}	0_{14}
0_{1}	0_{2}	0_{3}	0_{4}	0_{5}	0_{6}	0_{7}

codifica digitale dell'immagine = sequenza dei suoi pixel

0000000011110001100000100000

Rappresentazione delle immagini

- Immagini bitmap:
  - rappresentate pixel per pixel;
  - il file ha solitamente estensione .bmp;
  - hanno elevate dimensioni.
- Immagine bitmap compresse:
  - GIF (Graphics Interchange Format) e JPEG (Joint Photographic Experts Group) sono esempi.
- Immagini vettoriali:
  - SVG (Scalable Vector Graphics) é un esempio.
  - sono rappresentate specificando gli elementi geometrici, quali punti, segmenti e poligoni, che le compongono.
  - hanno dimensioni ridotte.

Rappresentazione delle immagini

- Nei monitor é utilizzato lo standard RGB.
- Ogni colore é ottenuto mescolando tre diverse gradazioni dei colori primari, quali rosso, verde e blu.
- Il numero di colori é detto profonditá di colore e si esprime generalmente in bit:
  - 1 bit per immagini bianco/nero (line art)
  - 8 bit per immagini in toni di grigio o 256 colori (grayscale)
  - 16 bit per immagini a 65536 colori (high color)
  - 32 bit per immagini a 32,8 milioni di colori (true color)

Rappresentazione delle immagini

- Il numero di pixel (quadratini) determina la risoluzione e si esprime generalmente in dpi (dot per inch, punti per pollice).
- Maggiore é la risoluzione, migliore é la qualitá dell'immagine e maggiori sono le dimensioni dell'immagine in byte.
- La risoluzione del monitor é data dal numero di pixel presenti sullo schermo:
  - 800×600;
  - 1024×768;
  - 1600×1200.

Esercizi sulla rappresentazione delle immagini

Dire quanti byte occupa un'immagine di 100×100 pixel in bianco e nero.

- Si tenga conto della dimensione complessiva dell'immagine data da 100×100.
- Anche del numero di bit necessari per un pixel di un'immagine in bianco e nero.

Dire quanti byte occupa un'immagine di 100×100 pixel a 256 colori.

- Si tenga conto della dimensione complessiva dell'immagine data da 100×100.
- Anche del numero di bit necessari per un pixel di un'immagine a colori.

Dire da quanti pixel é composta un'immagine di 16777216 milioni di colori che occupa 2400 byte.

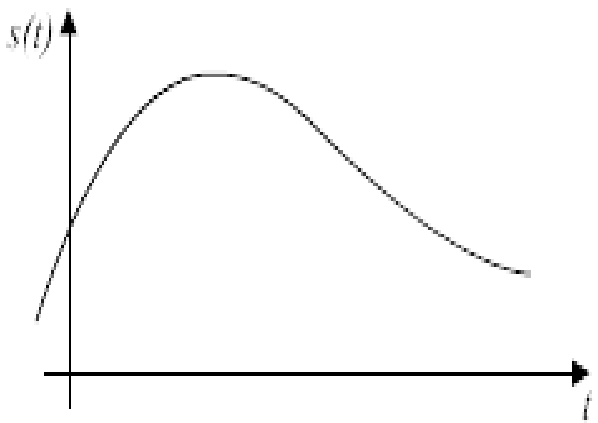
- Si tenga conto del numero di byte occupati da un'immagine di 16777216 milioni di colori.

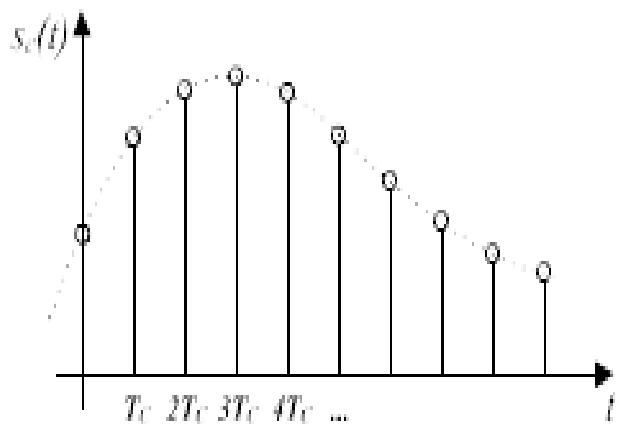
Rappresentazione di video

- Un filmato é una sequenza temporale di immagini ciascuna detta frame.
- Per rappresentare un filmato si digitalizzano i suoi frame.
- La compressione di un video rappresenta solo le differenze tra frame successivi.
- AVI (Audio Video Interleave) e MPEG sono esempi di formati.

Rappresentazione dei suoni

- Fisicamente é rappresentato come un'onda che descrive la variazione della pressione dell'aria nel tempo (onda sonora).
- L'onda sonora é campionata, ossia si misura l'ampiezza ad intervalli di tempo regolari.
- L'ampiezza misurata in un dato istante determina il campione.
- Il numero di misure effettuate in un secondo definisce la frequenza di campionamento (Hz, Herz).





Frequenza di campionamento

- É la misura del numero di volte al secondo in cui un segnale analogico viene misurato e memorizzato in forma digitale.
  - Segnale analogico é la rappresentazione di una grandezza fisica in modo analogico (come segnale acustico con segnale elettrico).
- Traduce un fenomeno naturale in una rappresentazione numerica comprensibile al computer.

Rappresentazione dei suoni

- La precisione della ricostruzione del suono dipende:
  - dalla frequenza di campionamento;
  - dal numero di bit usati per rappresentare il campione;
- Per avere una maggiore accuratezza é necessario occupare piú memoria.

Esercizio sulla rappresentazione dei suoni

Rispondere al quesito: quanto spazio occupa un suono della durata di 10 secondi campionato a 100 Hz, in cui ogni campione occupa 4 byte?

- Si consideri l'informazione proveniente dalla frequenza di campionamento, che dice il numero di campioni in un secondo.
- Si consideri il numero complessivo dei campioni utilizzando l'informazione della durata del suono.
- Tramite il numero complessivo dei campioni e il numero di occupazione in byte di ogni campione si determina quanto spazio occupa il suono.

Esercizio sulla rappresentazione dei suoni

Un secondo di suono campionato a 512 Hz occupa 1KB. Quanti valori distinti si possono avere per campione?

- Ci sono in tutto 512 campioni essendo memorizzati 512 campioni al secondo.
- Il file sonoro occupa 1KB = 1024 byte.
- Ogni singolo campione occuperá $\frac{1024}{512} = 2$ byte, ossia 16 bit.

Risultato: si possono avere $2^{16} = 65536$ diversi valori.