Algoritmi Numerici (Parte I)

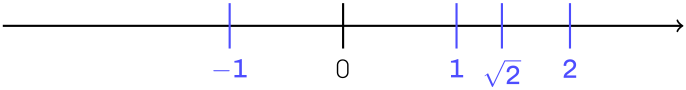
[Lezione 4] Numeri reali e formato float

Alessandro Antonucci
alessandro.antonucci@supsi.ch

https://colab.research.google.com/drive/1uSKMuhZCE8eOLDgGtKhwDqIupy0sh3dw

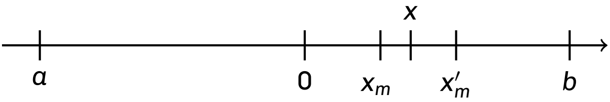
I numeri reali

- Operazioni con risultato irrazionale (es. $\sqrt{2}$)
- Si allarga l'insieme $\mathbb{Q}$ a $\mathbb{R}$ insieme numeri reali
- Diversamente da $\mathbb{Q}$, elementi $\mathbb{R}$ infinità non contabile
- Corrispondenza con i punti di un asse (cartesiano)



Rappresentare i numeri reali

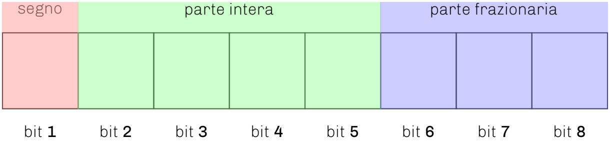
consideriamo range limitato, $[a, b] \subset \mathbb{R}$



- Infiniti elementi, non rappresentabili con n $(< \infty)$ bit
- Rappresentazione $\mathbb{R}$ necessariamente approssimata
- Usare $\mathbb{Q}$? Infiniti razionali dentro all'intervallo
- Esatta solo per (un numero finito di) numeri macchina
- Num non-macchina? Approx al num macchina piu vicino
- Errore di rappresentazione (al massimo $\frac{x'_m - x_m}{2}$)

Scegliere i numeri macchina

- Dati ***n*** bit, rappresentare ($\boldsymbol{2^n}$) numeri macchina
- Un bit al segno, k bit per parte intera, altri a frazionaria
- Sistema molto semplice e facile da leggere,
  ma distanza fra i numeri macchina costante
- Es. m=8 k=4



$0\ 0000\ 000 \rightarrow +\ \mathbf{0000.000}_2 = 0$

$0\ 0000\ 001 \rightarrow +\ \mathbf{0000.001}_2 = 0.125$

$0\ 0000\ 010 \rightarrow +\ \mathbf{0000.010}_2 = 0.25$

$0\ 0000\ 011 \rightarrow +\ \mathbf{0000.011}_2 = 0.375$

$0\ 0000\ 100 \rightarrow +\ \mathbf{0000.100}_2 = 0.5$

$0\ 0000\ 101 \rightarrow +\ \mathbf{0000.101}_2 = 0.625$

$0\ 0000\ 110 \rightarrow +\ \mathbf{0000.110}_2 = 0.75$

$0\ 0000\ 111 \rightarrow +\ \mathbf{0000.111}_2 = 0.875$

$0\ 0001\ 000 \rightarrow +\ \mathbf{0001.000}_2 = 1$

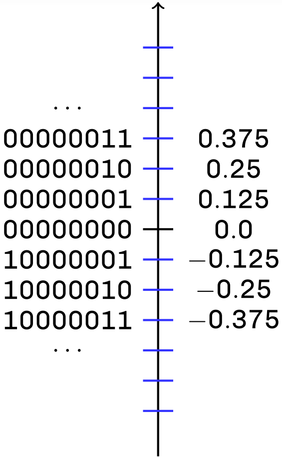
$0\ 0001\ 001 \rightarrow +\ \mathbf{0000.001}_2 = 1.125$

$0\ 0001\ 010 \rightarrow +\ \mathbf{0000.010}_2 = 1.25$

$0\ 0001\ 011 \rightarrow +\ \mathbf{0000.001}_2 = 1.375$

$\cdots$

$1\ 1011\ 101 \rightarrow -\ \mathbf{1011.101}_2 = -11.625$



numeri macchina
tutti equidistanti
(distanza $\mathbf{2}^{-3}$)

Errore assoluto e relativo

- Grandezza (reale) x approssimata da x_m
- Errore assoluto $\epsilon_A := |x - x_m|$
- Errore relativo $\epsilon_R := \frac{\epsilon_A}{|x|} = \left|\frac{x - x_m}{x}\right|$

Es. 7CHF approssimati a 10CHF, $\epsilon_A = 3$CHF, $\epsilon_R \simeq 43\%$

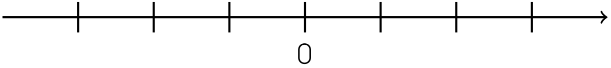
12'007CHF approssimati a 12'010CHF, $\epsilon_A = 3$CHF,

$\epsilon_R \simeq 0.025\%$

per rappresentare i numeri reali

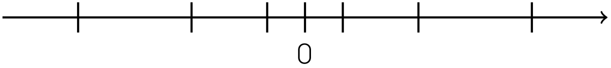
meglio ϵ_R indipendente dal numero che ϵ_A

quindi anziché numeri macchina equidistanti



aumentare la distanza più il numero diventa grande

(in valore assoluto)



Notazione scientifica

- Sistema posizionale scomodo per rappresentare numeri grandi/piccoli
- Ex. gravitazione universale
  $\mathbf{0.000000000066725985} \frac{N \cdot m^2}{kg^2}$,
- Alternativa? Notazione scientifica:

segno per mantissa per base elevata all' esponente "

$$+ \; 6.6725985 \cdot 10^{-11}$$

- Non univoco: $+ \; 66.725985 \cdot 10^{-12}$
  lo diventa se fisso una regola sulla mantissa
- Stessa cosa in base due $+ \; 1011.0011_2 \cdot 2^5$

Il formato float a 32 bit

- 1 bit per il segno
- 8 bit per l'esponente
- 32 - 1 - 8 = 23 bit per la mantissa

Regole formato float (ci saranno eccezioni)

- Segno: $+$ se il primo bit vale 0, $-$ altrimenti
- Esponente: leggi gli 8 bit con Horner e togli 127
- Mantissa: 1.xxxx con xxx sequenza di 23 bit
- Base: 2

Esercizi sul formato float

1. Identificare a quale numero corrisponde la sequenze FA800000 e 42F91FE6 (compattata in esadecimale) sulla base delle regole del formato float
2. Identificare la sequenza di 32 bit (opportunamente compattata in esadecimale) che corrisponde al numero $x = -\mathbf{137.12}$ secondo le regole del formato float
3. Il formato `double` è l'analogo a 64 bit del formato float. Sapendo che il formato utilizza 11 bit per l'esponente ricostruire per analogia le regole del formato

FA800000 $\rightarrow$ $\mathbf{1}|\mathbf{111\ 1010\ 1}|\mathbf{000\ 0000\ 0000\ 0000\ 0000\ 0000}$

$b_1 = 0 \Rightarrow x$ negativo

esponente = horner(**1111 0101**)-127=118

mantissa = 1.**0**

$x = -2^{118} \simeq -3.3231 \cdot 10^{35}$

42F91FE6 $\rightarrow$ $\mathbf{0}|\mathbf{100\ 0010\ 1}|\mathbf{111\ 1001\ 0001\ 1111\ 1110\ 0110}$

$b_1 = 0 \Rightarrow x$ positivo

esponente =horner(**1000 0101**)-127=6

mantissa = **1.11110010001111111110011**

$x = 1.11110010001111111110011 \cdot 2^6 =$
$1111100.10001111111110011 = 124.5623016357421875$

1/2+1 = 1.5
1.5/2+0 = 0.75
0.75/2+0 = 0.375
0.375/2+1 = 1.1875
1.1875/2+1 = 1.59375
1.59375/2+1 = 1.796875
1.796875/2+1 = 1.8984375
1,8984375/2+1 = 1.94921875
1,94921875/2+1 = 1.974609375
1.974609375/2+1 = 1.9873046875
1.9873046875/2+1 = 1.99365234375
1.99365234375/2+0 = 0.996826171875
0.996826171875/2+0 = 0.4984130859375
0.4984130859375/2+0 = 0.24920654296875
0.24920654296875/2+1 = 1.124603271484375
1.124603271484375/2 = 0.5623016357421875

Sequenza 32 bit associata a $-\mathbf{137.12_{10}}$ secondo float

$\mathbf{137_{10}}$ in base 2

$$137 \mod 2 = 1$$
$$68 \mod 2 = 0$$
$$34 \mod 2 = 0$$
$$17 \mod 2 = 1$$
$$8 \mod 2 = 0$$
$$4 \mod 2 = 0$$
$$2 \mod 2 = 0$$
$$1 \mod 2 = 1$$
$$0/$$

Sintesi:

$$137_{10} = 10001001_2$$

Verifica:

$$137 = 2^7 + 2^3 + 1$$

$$0.12_{10} = 0.\overline{00011110101110000100}_2$$

$int(0.12 \cdot 2) = 0$
$int(0.24 \cdot 2) = 0$
$int(0.48 \cdot 2) = 0$
$int(0.96 \cdot 2) = 1$
$int(0.92 \cdot 2) = 1$
$int(0.84 \cdot 2) = 1$
$int(0.68 \cdot 2) = 1$
$int(0.36 \cdot 2) = 0$
$int(0.72 \cdot 2) = 1$
$int(0.44 \cdot 2) = 0$
$int(0.88 \cdot 2) = 1$
$int(0.76 \cdot 2) = 1$
$int(0.52 \cdot 2) = 1$
$int(0.04 \cdot 2) = 0$
$int(0.08 \cdot 2) = 0$
$int(0.16 \cdot 2) = 0$
$int(0.32 \cdot 2) = 0$
$int(0.64 \cdot 2) = 1$
$int(0.28 \cdot 2) = 0$
$int(0.56 \cdot 2) = 0$
$int(0.12 \cdot 2) = 0$
periodico!

$$-137.12 = -10001001.\overline{00011110101110000100}_2$$

Notazione scientifica

$$-10001001.\overline{00011110101110000100}_2 \cdot 2^0$$

Mantissa normalizzata

$$-1.0001001\overline{00011110101110000100}_2 \cdot 2^7$$

Quasi-formato float

$$-1.0001001\overline{00011110101110000100}_2 \cdot 2^{134-127}$$

Rappresento 134 a 8 bit

$$\begin{aligned}
&134 \mod 2 = 0\\
&67 \mod 2 = 1\\
&33 \mod 2 = 1\\
&16 \mod 2 = 0\\
&8 \mod 2 = 0\\
&4 \mod 2 = 0\\
&2 \mod 2 = 0\\
&1 \mod 2 = 1
\end{aligned}$$

$$134_{10} = 10000110_2$$

$-137.12 =$
$-1.0001001\overline{00011110101110000100}_2 \cdot 2^{\text{horner}(10000110)-127}$

23 bit per la mantissa

$-1.\,\colorbox{yellow}{00010010001111010111000}\,0100\ldots \cdot 2^{\text{horner}(10000110)-127}$

segno

1

esponente

1	0	0	0	0	1	1	0

mantissa

0	0	0	1	0	0	1	0	0	0	1	1	1	1	0	1	0	1	1	1	0	0	0

Sintesi **1|10000110|00010010001111010111000**

Zip esadecimale

1100 0011 0000 1001 0001 1110 1011 1000 $\rightarrow$ C 3 0 9 1 E B 8

IEEE 754 Converter (JavaScript), V0.13

Note: This JavaScript-based version is still under development, please report errors here.

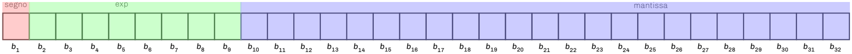
	Sign	Exponent	Mantissa
Value:	-1	2^7	1.0712499618530273
Encoded as:	1	134	597688
Binary:			

Decimal Representation	-137.12
Binary Representation	11000011000010010001111010111000
Hexadecimal Representation	0xc3091eb8
After casting to double precision	-137.1199951171875

From 32 to 64

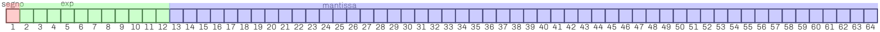
- Float (32 bit)

$$x = \mathrm{sgn}(b_1) \cdot 1.[b_{10}, \ldots, b_{32}] \cdot 2^{\mathrm{HORNER}(b_2,\ldots,b_9)-127}$$



- Double (64 bit)

$$x = \mathrm{sgn}(b_1) \cdot 1.[b_{13}, \ldots, b_{64}] \cdot 2^{\mathrm{HORNER}(b_2,b_{12})-1023}$$



$$\mathrm{sgn}(b) = (-1)^b,\ 127 = 2^{8-1} - 1,\ 1023 = 2^{11-1} - 1$$