# Rappresentazioni di numeri al calcolatore

Stefano Gualandi
Università di Pavia, Dipartimento di Matematica

email: stefano.gualandi@unipv.it
twitter: @famo2spaghi
blog: http://stegua.github.com

# Numeri Arabi

Sin dalle elementari siamo abituati alla rappresentazione simbolica dei numeri con i numeri arabi, che è basato sulle 10 cifre (in inglese *digits*, che viene dal latino *digitus* per significa dito):

$$0, 1, 2, 3, 4, 5, 6, 7, 8, 9$$

Bastano 10 cifre per rappresentare i numeri, in quanto con la notazione **posizionale**, data la posizione della cifra, possiamo capire come interpretare una data sequenza di cifre. Per esempio:

$$\begin{aligned} 563.6 \quad \text{rappresenta} \quad & 5 \times 10^2 + 6 \times 10^1 + 3 \times 10^0 + 6 \times 10^{-1} \\ & = 500 + 60 + 3 + \frac{6}{10} \\ & = 536.6 \end{aligned}$$

Questo viene chiamato sistema **decimale** in quanto la **base** di riferimento è 10. Il "." indica il punto base (decimale).

# Moltiplicare e dividere per la base

Osserviamo che quando moltiplichiamo un numero per la base, in pratica spostiamo il punto base di una posizione verso destra ($563.6 \times 10 = 5636.0$).

Se dividiamo per la base, spostiamo il punto base di una posizione a sinistra ($563.6/10 = 56.36$).

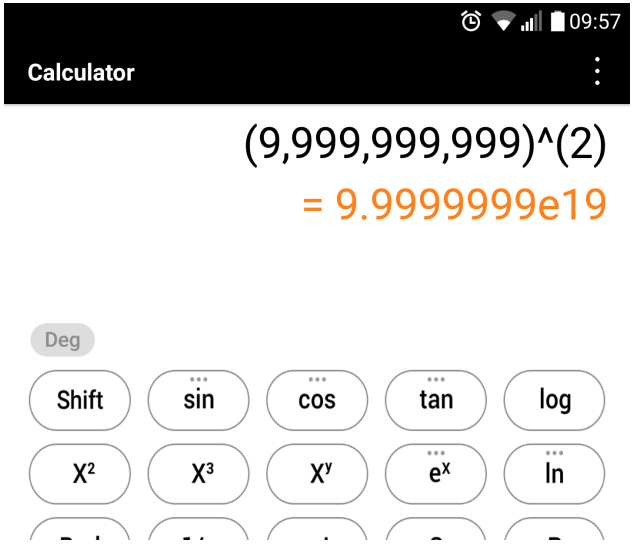
Se $k$ è un numero intero positivo, $10^k$ è la cifra 1 seguita da $k$ zeri ($10^4 = 10000$). Invece, $10^{-k}$ è il punto base seguito prima da $(k-1)$ zeri, e poi da uno ($10^{-4} = .0001$).

# Notazione Scientifica

I numeri nelle calcolatrici e nei computer vengono rappresentati con la notazione in *virgola mobile*.

Se su una calcolatrice proviamo a calcolare il quadrato di 9 999 999 999 otteniamo:

$$9.9999999\,e19 \quad \text{in cui e19 } \textit{significa} \quad \times 10^{19}$$




La parte alla sinistra della “e” viene chiamata la **mantissa**, la parte alla destra è l’**esponente**.

# Errori di Approssimazione

Supponiamo di dover calcolare $\frac{625}{26}$ come imparato alle elementari.

```
   24.038 461 538 461 538 …
 ┌──────────────────────────
26 ) 625.000 000 000 000 000 …
     52
     105
     104
       1 0  .   .   .   .   .   .   R1 =  1
         0
       ---
       1 00 .   .   .   .   .   .   R2 = 10 ←
         78
       ---
         220    .   .   .   .   .   R3 = 22
         208
         ---
          12 0  .   .   .   .   .   R4 = 12
          10 4
          ----
           1 60 .   .   .   .   .   R5 = 16
           1 56
           ----
              40    .   .   .   .   R6 =  4
              26
              --
              14 0  .   .   .   .   R7 = 14
              13 0
              ----
               1 00     .   .   .   R8 = 10 ←
```

# Errori di Approssimazione

Supponiamo di dover calcolare $\frac{625}{26}$ come imparato alle elementari.

```
     24.038 461 538 461 538 …
   ┌──────────────────────────
26 ) 625.000 000 000 000 000 …
     52
     105
     104
       1 0   .   .   .   .   .   .   R1 =  1
         0
       1 00  .   .   .   .   .   .   R2 = 10 ←
         78
         220     .   .   .   .   .   R3 = 22
         208
          12 0   .   .   .   .   .   R4 = 12
          10 4
           1 60  .   .   .   .   .   R5 = 16
           1 56
              40     .   .   .   .   R6 =  4
              26
              14 0   .   .   .   .   R7 = 14
              13 0
               1 00      .   .   .   R8 = 10 ←
```

L'algoritmo della divisione e usiamo la notazione $\frac{625}{26} = 24.0\overline{384615}$

# Errori di Approssimazione: Troncamento

I **numeri razionali** sono i numeri che possono essere rappresentati come rapporti tra numeri interi.

Ogni numero razionale rappresentato con una notazione posizionale deve avere una forma di ripetizione, anche se solitamente non scriviamo la ripetizione di 0 (per esempio, $\frac{1}{625} = 0.00160\overline{0}$),

Nessun oggetto fisico può contenere un numero infinito di elementi. In pratica, per rappresentare un numero razionale quello che viene fatto è di *troncare* la rappresentazione usando solo le cifre più significative: tutte le cifre meno significative vengono semplicemente ignorate.

Per esempio, dato $A = \frac{625}{26} = 24.0\overline{384615}$

$$A_1 = 24.03 \quad \text{troncamento alle 4 cifre più significative}$$
$$A_2 = 24.0384 \quad \text{troncamento a 4 cifre dopo il punto base}$$

# Errori di Approssimazione: Arrotondamento

In alternativa, in modo migliore per rappresentare un numero razionale è di "arrotondare" il numero.

In base 10, la solita regola di arrotondamento è:

> *Se la prossima cifra è maggiore o uguale a 5, allora aggiungi 1 alla cifra meno significativa al numero troncato.*

Nell'esempio precedente, dato $A = \frac{625}{26} = 24.0\overline{384615}$

$$\begin{aligned} A_3 &= 24.04 && \text{la prima cifra dopo il 3 è 8 } (> 5) \\ A_4 &= 24.0385 && \text{la prima cifra dopo il 4 è 6 } (> 5) \end{aligned}$$

# Errore di Approssimazione

Abbiamo due modi di quantificare l'errore tra un numero $A$ e la sua rappresentazione $B$, dove per errore si intende la differenze $e = A - B$:

1. Usare l'**errore assoluto in B** che è il modulo dell'errore:

$$|A - B|$$

# Errore di Approssimazione

Abbiamo due modi di quantificare l'errore tra un numero $A$ e la sua rappresentazione $B$, dove per errore si intende la differenze $e = A - B$:

1. Usare l'**errore assoluto in B** che è il modulo dell'errore:

$$|A - B|$$

2. Usare l'**errore relativo in B** che è l'errore assoluto normalizzato:

$$\frac{|A - B|}{|A|}$$

L'errore relativo viene spesso dato in percentuale.

# Esempi

| | **Error** | **Relative error** |
|---|---|---|
| For $A1 = 24.03$ | $24.03 - \text{A}$ | $\lvert\text{Error}\rvert \div A$ |
| | $= \frac{2403}{100} - \frac{625}{26}$ | $= \frac{22}{2600} \times \frac{26}{625}$ |
| | $= \frac{62478 - 62500}{100 \times 26}$ | $= 0.000352$ |
| | $= \frac{-22}{2600}$ | $= 0.0352\%$ |
| | $= -0.00\overline{846153}$ | |
| | | // similarly, you can show |
| For $A2 = 24.0384$ | $\frac{-16}{260000} = -0.0000\overline{615384}$ | 0.000 256% |
| For $A3 = 24.04$ | $\frac{+4}{2600} = 0.00\overline{153846}$ | 0.006 4% |
| For $A4 = 24.0385$ | $\frac{+10}{260000} = 0.0000\overline{384615}$ | 0.000 16% |

# Sistema binario

I calcolatori sono in grado di gestire due cifre:

$$0, 1$$

Anche con solo due cifre, possiamo usare una notazione posizionale per rappresentare i numeri:

$$\begin{aligned} 101111.011 \quad \text{means} \quad & 1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 \\ & + 0 \times 2^{-1} + 1 \times 2^{-2} + 1 \times 2^{-3} \\ & = 32 + 8 + 4 + 2 + 1 + \frac{1}{4} + \frac{1}{8} \\ & = 47 + 0.25 + 0.125 \\ & = 47.375 \end{aligned}$$

# Somma e prodotto di cifre binarie

| $\times$ | **0** | **1** |
|---|---|---|
| **0** | 0 | 0 |
| **1** | 0 | 1 |

# Somma e prodotto di cifre binarie

| $\times$ | **0** | **1** |
|---|---|---|
| **0** | 0 | 0 |
| **1** | 0 | 1 |

| $+$ | **0** | **1** |
|---|---|---|
| **0** | 0 | 1 |
| **1** | 1 | 10 |

# Somma e prodotto di cifre binarie

| $\times$ | **0** | **1** |
|---|---|---|
| **0** | 0 | 0 |
| **1** | 0 | 1 |

| $+$ | **0** | **1** |
|---|---|---|
| **0** | 0 | 1 |
| **1** | 1 | 10 |

Come per la notazione in base 10, anche in base 2 se moltiplichiamo un numero per la base, spostiamo il punto base di una posizione verso destra, mentre se dividiamo per la base, spostiamo il punto base di una posizione a sinistra:

$$\begin{aligned} 101111.011 \times 2 &= 1011110.11 \\ 101111.011 \,/\, 2 &= 10111.1011 \end{aligned}$$

Se $k$ è un numero intero positivo, $2^k$ è la cifra 1 seguita da $k$ zeri

($2^4 = 10000$), e $2^{-k}$ è il punto base seguito prima da $(k-1)$ zeri, e poi da uno ($2^{-4} = .0001$).

# La moltiplicazione del contadino russo

**Input**: Dati due numei interi M=73 e N=41

| A | B |
|---|---|
| 73 | 41 |
| ~~146~~ | *20* (+1/2) |
| ~~292~~ | *10* |
| 584 | 5 |
| ~~1168~~ | *2* (+1/2) |
| 2336 | 1 |
| ***2993*** | |

**Output**: 2993

# Moltiplicare per due

Nell'algoritmo del contadino russo, ad ogni iterazione moltiplichiamo per due il primo numero:

| $A\{10\}$ | $A\{2\}$ | |
|---|---|---|
| 73 | 100 100 1 | // $73 = 64 + 8 + 1 = 2^6 + 2^3 + 2^0$ |
| 146 | 100 100 10 | |
| 292 | 100 100 100 | |
| 584 | 100 100 100 0 | |
| 1168 | 100 100 100 00 | |
| 2336 | 100 100 100 000 | |

# Dividere per due

Nell'algoritmo del contadino russo, ad ogni iterazione dividiamo per due il secondo numero e tronchiamo il risultato:

| $B\{10\}$ | $B\{2\}$ | |
|---|---|---|
| 41 | 101001 | $// 41 = 32 + 8 + 1 = 2^5 + 2^3 + 2^0$ |
| 20 | 10100 | $// 20 = 16 + 4 = 2^4 + 2^2$ |
| 10 | 1010 | $// 10 = 8 + 2 = 2^3 + 2^1$ |
| 5 | 101 | $// 5 = 4 + 1 = 2^2 + 2^0$ |
| 2 | 10 | |
| 1 | 1 | |

# Moltiplicazione tra numeri binari

Se facciamo attenzione, a questo punto dovremmo riconoscere che l'algoritmo del contadino russo non è altro che la solita moltiplicazione svolta in un sistema binario:

| | | | |
|---|---|---|---|
| 73{10}= | 1001001 | | |
| ×41{10}= | ×101001 | | |
| | 1001001 | 73 | // multiply by the digit 1 |
| | ~~1001001x~~ | ~~146~~ | // shift and multiply by 0 |
| | ~~1001001xx~~ | ~~292~~ | // shift and multiply by 0 |
| | 1001001xxx | 584 | // shift and multiply by 1 |
| | ~~1001001xxxx~~ | ~~1168~~ | // shift and multiply by 0 |
| | 1001001xxxxx | 2336 | // shift and multiply by 1 |
| | 101110110001 | 2993 | |

$$\begin{aligned} \text{// } 101\ 110\ 110\ 001\{2\} &= 2^{11} + 2^9 + 2^8 + 2^7 + 2^5 + 2^4 + 2^0\{10\} \\ \text{//} &= 2048 + 512 + 256 + 128 + 32 + 16 + 1\{10\} \\ \text{//} &= 2993\{10\} \end{aligned}$$

# Notazione Esponenziale (Normalizzata)

La notazione in virgola mobile (*floating points*) si basa sulla **Notazione Esponenziale (Normalizzata)** di un numero $x$ in base $\beta$, che è:

$$\begin{aligned} x &= (-1)^s d_0.d_1 d_2 d_3 \dots d_{t-1} \beta^e \\ &= (-1)^s \left( \sum_{i=0,\dots,t-1} d_i \beta^{-i} \right) \beta^e \\ &= (-1)^s m \beta^e \end{aligned}$$

- $s$ è il segno di $x$
- $\beta$ è un intero $> 1$ e corrisponde alla base
- $e$ è l'esponente
- $m$ è la mantissa $m = d_0.d_1 d_2 d_3 \dots d_{t-1}$, in cui $0 \leq d_i \leq \beta - 1$

**Esempio:** La rappresentazione esponenziale normalizzata in base dieci di 0.00078391 è $7.8391e^{-4}$

# Insieme dei numeri in virgola mobile

Dati due numeri $L$ e $U$ entro cui è compreso l'esponente, possiamo definire il seguente insieme di numeri:

$$F(\beta, t, L, U) := \{0\} \cup \{x \mid (-1)^s \left( \sum_{i=0,\ldots,t-1} d_i \beta^{-i} \right) \beta^e, L \leq e \leq U\}$$

Si osservi che:

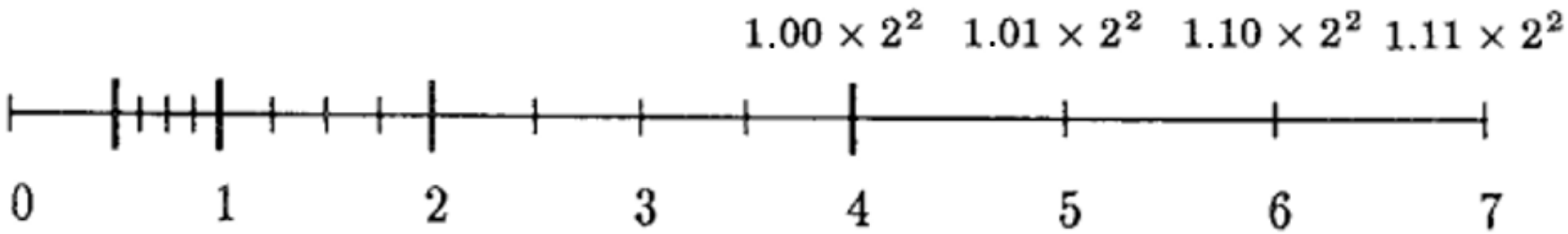
- Si ha che $x \in F$ se e solo se $-x \in F$
- Per ogni $x \in F, x \neq 0$, si ha che:

$$x_{min} = \beta^L \leq |x| \leq \beta^U (2 - \beta^{-t+1}) = x_{max}$$

# Esempio: $F(\beta = 2, t = 3, L = -1, L = 2)$

Rappresentazione grafica dei numeri 16 (più lo 0) normalizzati (ovvero con $d_0 > 0$) contenuti nell'insieme:

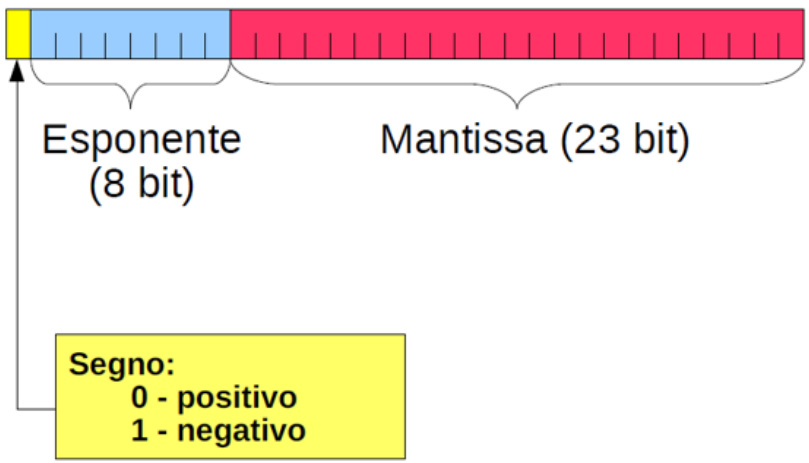
$$F(\beta = 2, t = 3, L = -1, L = 2)$$



# Standard IEEE 754

Sui calcolatori esistono due notazioni standard per i numeri in virgola mobile:

1. Singola precisione, usa $t = 32$ bits: 1 bit di segno, 8 bits per l'esponente, e 23 per la mantissa




2. Doppia precisione, usa $t = 64$ bits: 1 bit di segno, 11 bits per l'esponente, e 52 per la mantissa

# Standard IEEE 754: Esempio


Es. $34.5_{10} = 100010.1_2 = 1.000101_2 \cdot 2^5 = 1.000101_2 \cdot 10_2^{101}$

Segno: 0 (positivo)

Mantissa: 000101 (la parte intera pari a 1 si sottintende)

Esponente: $5+127 = 132 = 10000100_2$
(rappresentazione in ***eccesso 127***)

# Riferimenti


- Tom Jenkyns, Ben Stephenson. Fundamentals of Discrete Math for Computer Science: A Problem-Solving Primer (Capitolo 1). Springer, 2012.

- David Goldberg. *What every computer scientist should know about floating-point arithmetic*. ACM Computing Surveys (CSUR). Vol 23(1), pp. 5-48.