Corso di Calcolatori Elettronici I

Rappresentazione dei numeri reali in un calcolatore

Prof. Roberto Canonico

Università degli Studi di Napoli Federico II
Facoltà di Ingegneria
Corso di Laurea in Ingegneria Informatica
Corso di Laurea in Ingegneria dell'Automazione

Rappresentazione di numeri reali

- Con un numero finito di cifre è solo possibile rappresentare un numero razionale *che approssima con un certo errore* il numero reale dato
- Vengono usate due notazioni:

A) Notazione in virgola fissa

Dedica parte delle cifre alla parte intera e le altre alla parte frazionaria

± XXX .YY

B) Notazione in virgola mobile

Dedica alcune cifre a rappresentare un esponente della base che indica l'ordine di grandezza del numero rappresentato

Numeri reali: rappresentazione in virgola fissa

- Quando di un numero frazionario si rappresentano separatamente la parte intera e la parte frazionaria si parla di rappresentazione in *virgola fissa*
- La rappresentazione dei due contributi può essere realizzata secondo una delle tecniche viste in precedenza
- La parte frazionaria è rappresentata con un numero finito m di cifre binarie, scalata di un fattore 2^m che la rende intera
- La posizione della virgola è fissa e resta sottintesa

Numeri reali in virgola fissa

- La stringa

  $,b_{-1}b_{-2}..b_{-m}$

  si interpreta come

  $b_{-1}2^{-1}+b_{-2}2^{-2}+ \ldots +b_{-m}2^{-m}$
- Esempio: .1011

  si interpreta come

  $2^{-1}+2^{-3}+2^{-4}= 1/2 + 1/8 + 1/16 = 0,5 + 0,125 + 0,0625 = 0,6875$

  ovvero come

  $11 / 16 = 0,6875$
- La stringa 1011 è rappresentativa dell'intero $(11)_{10}$ che va scalato del fattore 2^{-4}

Numeri reali: rappresentazione in virgola mobile

- Un numero reale x può essere rappresentato dalla tripla

  (s,m,e)

  tale che:

  $$x = (-1)^s \cdot m \cdot b^e$$

  - *s è il segno (s=0 positivo, s=1 negativo)*
  - *m* è detta *mantissa*
  - *e* è detto *esponente*
  - *b* è la base di numerazione adottata
- In macchina sia m che e hanno un numero prefissato di cifre
  - intervalli limitati ed errori di arrotondamento

Normalizzazione

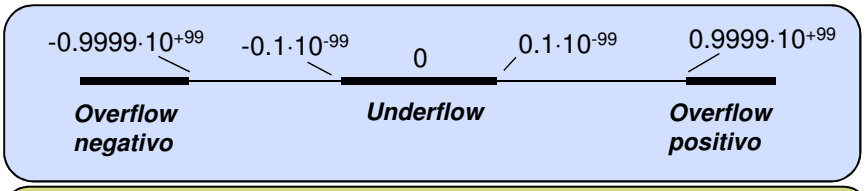
- Per ciascun numero esistono infinite coppie mantissa-esponente che lo rappresentano
- Esempio (b=10):
  - 346.09801 è rappresentato da
    - » m= 346.09801, e=0 oppure
    - » m= 346098.01, e= - 3 oppure
    - » m= 0.034609801, e= 4 ecc…
- Per **rappresentazione normalizzata** del numero si intende convenzionalmente quella in cui la mantissa ha la prima cifra a destra della virgola diversa da zero
- ovvero: $1/b \leq m < 1$
- Esempio:

  (0.34609801, 3)

Esempio: intervallo di rappresentazione

- Con b=10, usando 4 cifre per *m* e 2 per *e* (più due bit per i relativi segni), l'insieme rappresentabile (utilizzando solo rappresentazioni normalizzate) è:

  $[-0.9999\times10^{99},-0.1000\times10^{-99}]\cup\{0\}\cup[+0.1000\times10^{-99},+0.9999\times10^{99}]$



Con le stesse **6=4+2** cifre in virgola fissa ± **XXXX .YY** :
- L'intervallo scende **[-9999.99,+9999.99]**
- Ma si hanno **6** cifre significative invece di **4**

Approssimazione

- Come è facile verificare, in questo tipo di rappresentazione l'approssimazione non è costante
- In particolare la precisione assoluta è molto spinta in prossimità dello zero e va diminuendo progressivamente a mano a mano che il numero aumenta (in valore assoluto)
- Ad esempio:
  - in prossimità dello zero l'errore massimo che può essere commesso è $0.1001*10^{-99} - 0.1000*10^{-99} = \mathbf{0.0001*10^{-99}}$
  - in prossimità dell'estremo superiore dell'intervallo di rappresentazione, invece, l'errore massimo che si può commettere è $0.9999*10^{99} - 0.9998*10^{99} = \mathbf{0.0001*10^{99}}$
- Si commettono quindi "errori piccoli" su "numeri piccoli" ed "errori grandi" su "numeri grandi"
- Quello che resta inalterato è invece l'errore relativo, costante su tutto l'asse di rappresentabilità

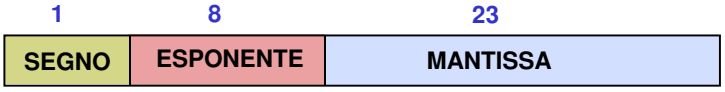
Overflow e Underflow

- L'errore relativo dipende dal numero di cifre della mantissa
- Gli estremi dell'intervallo di rappresentazione dipendono dal numero di cifre dell'esponente

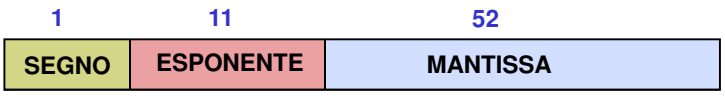
- Nel caso precedente di 2 cifre per l'esponente, si ha overflow per numeri maggiori (in modulo) di 10^{99} e si ha underflow per numeri minori (in modulo) di 10^{-99}

Standard IEEE 754 (1985)

- Formato standard indipendente dall'architettura
- Precisione semplice a 32 bit:

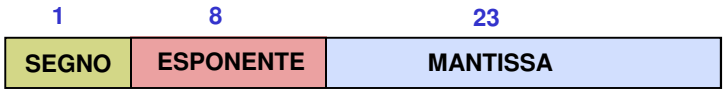


- Precisione doppia a 64 bit



- Notazioni in modulo e segno
- Alcune configurazioni dell'esponente sono riservate

IEEE 754 a 32 bit



- **$x = (-1)^S \times 1.F \times 2^{Exp-bias}$**

- **ESPONENTE**
  - Rappresentato in *eccesso 127*
  - L'intervallo è **[-126, +127]**
  - **Il valore -127 è riservato per rappresentazioni speciali**
- **MANTISSA**
  - Se ne rappresenta *solo la parte frazionaria*

$$\begin{cases} N = (-1)^S \times 1.fraction \times 2^{exponent-127}, & 1 \le exponent \le 254 \\ N = (-1)^S \times 0.fraction \times 2^{exponent-126}, & exponent = 0 \end{cases}$$

IEEE 754: forma normalizzata

- La mantissa binaria normalizzata deve presentare un 1 a sinistra della virgola binaria. L'esponente deve essere aggiustato di conseguenza
- Essendo sempre presente tale cifra non è informativa così come la virgola binaria; esse vengono considerate implicitamente presenti e non vengono memorizzate
- Per evitare confusione con una frazione tradizionale la combinazione dell'1 implicito della virgola binaria e delle 23/52 cifre significative viene chiamata ***significando*** (invece che frazione o mantissa)
  - Tutti i numeri normalizzati hanno un esponente e>0
  - Tutti i numeri normalizzati hanno un significando s tra 1≤s<2
  - I numeri normalizzati non possono avere un esponente composto da soli 1. Tale configurazione serve per modellare il valore infinito (∞)

IEEE 754: forma denormalizzata

- Questa rappresentazione viene utilizzata per rappresentare valori molto piccoli in valore assoluto, che altrimenti produrrebbero underflow con la rappresentazione normalizzata
  - Tutti i bit dell'esponente sono posti a 0 (questa configurazione indica l'utilizzo della forma denormalizzata)
  - Il bit della mantissa a sinistra della virgola binaria è posto implicitamente a 0
  - Il numero più piccolo rappresentabile in questa configurazione è composto da una mantissa con tutti 0 a eccezione del bit più a destra
  - La rappresentazione denormalizzata comporta una progressiva perdita di cifre significative

Esempio

$$-\left(6+\frac{5}{8}\right)=-\left(4+2+\frac{4}{8}+\frac{1}{8}\right)=-\left(4+2+\frac{1}{2}+\frac{1}{8}\right)$$
$$=-\left(1\times 2^2+1\times 2^1+0\times 2^0+1\times 2^{-1}+0\times 2^{-2}+1\times 2^{-3}\right)$$
$$=-\left(110.101_2\right)=-\left(1.10101_2\times 2^2\right)$$

Esponente:

$$exponent-127=2\Rightarrow exponent=129$$

Rappresentazioni speciali

In precisione semplice:

Esponente 255 = 11111111_2 indica un valore speciale

In particolare:

Esponente 255 = 11111111_2 ed f = 0,
Il valore rappresentato è ± infinity

Esponente 255 = 11111111_2 ed f diverso da zero
Il valore rappresentato è Not a Number (NaN)

Convenzioni analoghe sono fatte per I numeri in precisione multipla (su 64 bit, con 11 bit di esponente)