Rappresentazione dei numeri reali $\mathbb{R}$ #6.1

Limiti delle rappresentazioni in virgola fissa

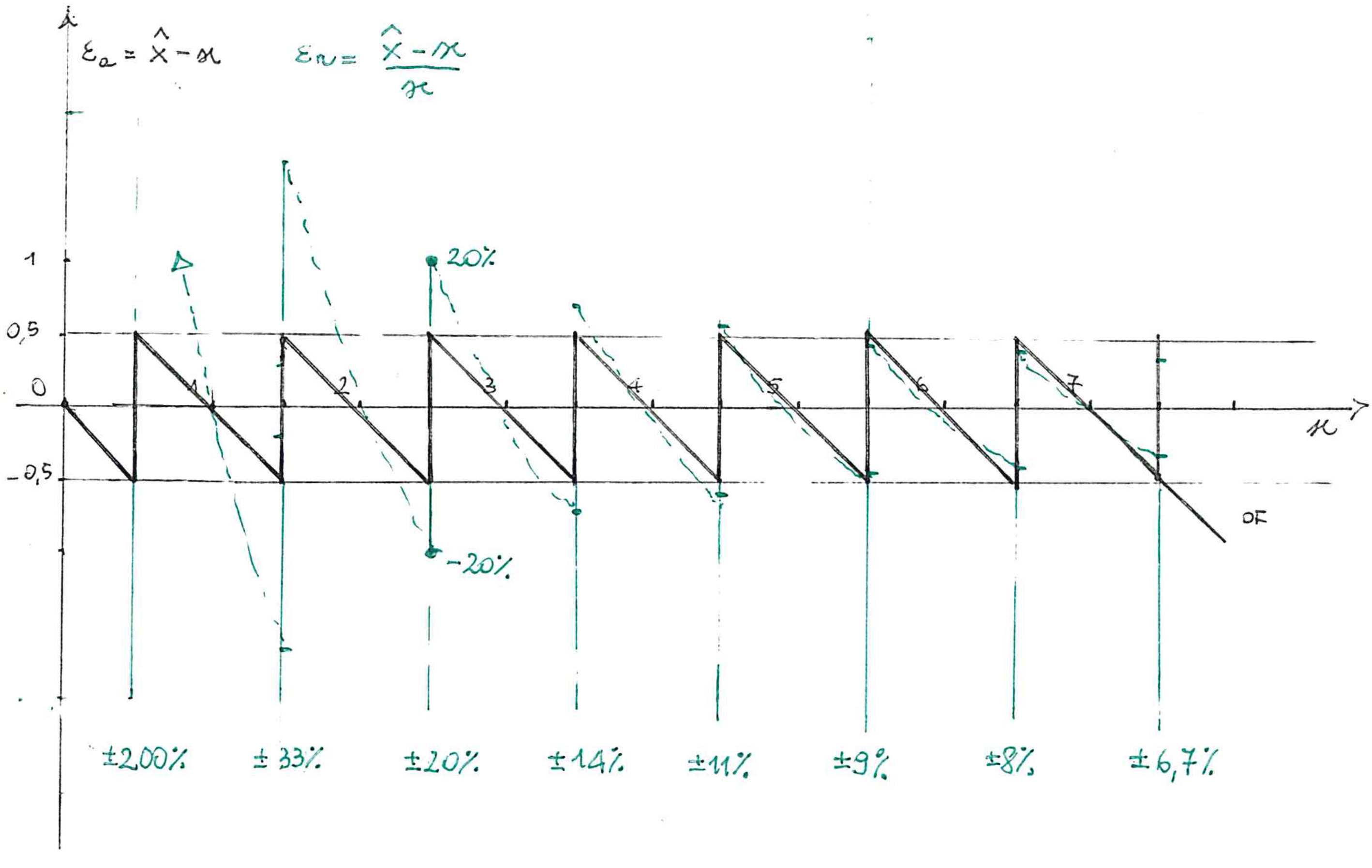
- I numeri frazionali visti possono essere usati per rappresentare numeri reali (un reale può sempre essere espresso come valore a cui converge una successione di razionali).
- Ci sono però dei limiti di cui occorre tenere conto
  - LEGAME range, numero di bit, errore risoluzione

> l'errore di rappresentazione è legato alla più piccola grandezza rappresentabile LSB.

$\frac{LSB}{2} < \varepsilon \leq \frac{LSB}{2}$ arrotondamento

> il range coperto dai codici è $2^n LSB$

> L'errore ASSOLUTO è COSTANTE: nel mondo reale però ci interessa di più l'errore RELATIVO
  - Un millimetro di errore è trascurabile nella misura di qualche metro è intollerabile se misuriamo pochi millimetri
  - Andamento di ε_A e ε_r nel caso proposto



- Quindi se intendiamo rappresentare grandezze da 1 mm a 10 m, con un errore relativo inferiore al 10%, quanti bit servono? (1)

Se l'unità di misura desiderata è il metro, che rappresentazione useremo? (2)

(1) Usando l'arrotondamento, dovremo avere una risoluzione di 0,2 mm (l'errore è METÀ della risoluzione con arrotondamento)

Serviranno $\frac{10m}{0,2mm} = 50 \times 10^3$ codici come minimo, quindi 16 b

in questo caso l'unità di misura è 0,2 mm e la rappresentazione non avrà parte frazionale.

(2) Se vogliamo la misura in metri serviranno 4 b per la parte intera $(10<16)$ e 13 b per la parte frazionale

$$\frac{1m}{8192} < 0,2mm \qquad \text{quindi } (4.13)$$

Osservazione: le rappresentazioni tra 0 e 1 mm avranno errore relativo maggiore, divergente se $\mapsto \emptyset$

Un modo più efficace di usare i bit per ridurre ε_r

- Dedicare un po' di bit a specificare l'ORDINE DI GRANDEZZA (come nella NOTAZIONE SCIENTIFICA)

Facciamo riferimento al problema precedente. Per avere il 10% (arrotondamento) tra 1 e 2 mm ci servono 3 bit.

$1.\boxed{b_{-1}\, b_{-2}\, b_{-3}}$

Per arrivare a 10 m ci sono 4 Odg_{10} $10^4 < 2^{14}$

Possiamo specificare l'OdG_2 con 4 bit. Con 7 b otteniamo la specifica richiesta abbondantemente

$$\underbrace{\boxed{f_{-1}}\boxed{f_{-2}}\boxed{f_{-3}}}_{f} \quad \underbrace{\boxed{e_3\, e_2\, e_1\, e_0}}_{e} \qquad x = \left(1 + \frac{f}{8}\right) \cdot 2^e$$

$1.000 \div 1.875$ range con $e = \emptyset$ (in mm)

$2.000 \div 3.750$ con $e = 1$

⋮

$32,8 \div 61,4$ con $e = 15$ (in metri)

Notazione in "virgola mobile". Standard

- La tecnica di separare esponente, cifre significative e con l'eventuale aggiunta del segno prende il nome di rappresentazione in virgola mobile (FLOATING POINT)
  - > Molto diffusa per il calcolo numerico
  - > Per rappresentare segnali con grande range dinamico
- Da diversi decenni, per rendere INTEROPERABILI sistemi diversi e permettere lo scambio di informazione, sono stati proposti STANDARD per la rappresentazione F.P.
  - Uno STANDARD è costituito da un insieme di prescrizioni o definizioni a cui il sistema (o il LINGUAGGIO SOFTWARE) si deve conformare
  - Nel caso degli standard di rappresentazione numerica sono importanti sia l'HW sia il SW che insieme concorrono a definire il comportamento del sistema

Lo standard IEEE 754 - 2008. Panoramica

- Per approfondire i concetti delle rappresentazioni FP, usiamo come riferimento lo standard corrente usato dalla maggior parte dei sistemi.
  - > 58 pagine, 11 capitoli, 2 appendici informative

Oggetto: formati e metodi per la gestione dell'aritmetica F.P. in sistemi elettronici di elaborazione
- precisione singola, doppia, estesa, estendibile
- formati di scambio
- condizioni di "eccezione" e modi di gestirle

Scopo: Sistemi conformi che elaborano gli STESSI DATI dovrebbero portare allo STESSO RISULTATO (indipendentemente dall'implementazione)

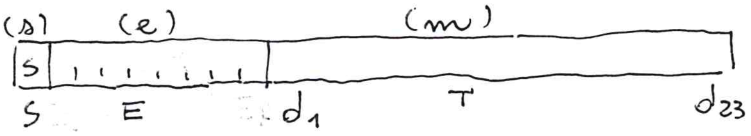
cosa è incluso:
- > Formati (base 2 e base 10) per ELABORAZIONE e per lo SCAMBIO dei DATI
- > Comportamento di addizione, sottrazione, moltiplicazione, divisione, mac, $\sqrt{\cdot}$, confronto e altre operazioni
- > Conversione INTERI ⇄ F.P.
- > Conversione tra F.P. diversi
- > Conversione tra F.P. e rappresentazioni esterne (es: stringhe di caratteri)
- > Eccezioni e relativa gestione

Formati proposti:

- Base 2 o 10
- Diverse cifre per la parte significativa
- Diverso range per l'esponente
- Rappresentazione di $\pm\infty$
- NaN (di vario tipo) quiet e signaling

BINARY32

- 32 bit divisi in 3 campi



legge di rappresentazione

$$x = (-1)^s \cdot 2^e \cdot m$$

"s" è $\emptyset$ (positivi)
1 (negativi)

"e" esponente $e_{min} \leq e \leq e_{max}$, -126, 127 ($e_{min} + e_{max} = 1$ per tutti i formati)

"m" significando frazionale $d_0 . d_1 d_2 \dots d_{24}$

- Per rendere univoca la rappresentazione, "m" è massimizzato (riducendo allo stesso tempo "e") fino a che
  $m \geq 1$ oppure $e = e_{min}$
- La rappresentazione di e è in traslazione
  $e = E - 127$
  quindi i valori "normali" di E sono $E = e + 127$
  $$1 \leq E \leq 254$$

Ricapitolando

S	E	T	valore
S	1 .. 254	T	$x = (-1)^S \cdot 2^{E-127} \cdot (1 + T\, 2^{-23})$
S	255	$\emptyset$	$x = \pm\infty$ secondo S
S	255	$\neq \emptyset$	NaN (sNaN: $d_1 = \emptyset$; qNaN: $d_1 = 1$)
S	0	$\neq 0$	$x = (-1)^S\, 2^{-126}\, T\, 2^{-23}$ (non normal.)
S	0	0	$x = \pm\emptyset$ secondo S

#6.5

Arrotondamento

- lo standard prevede "attributi sulla direzione dell'arrotondamento". Quindi i sistemi devono essere in grado di implementare diverse strategie.
  > L'esigenza di arrotondare nasce dal fatto che i risultati di molte operazioni non sono NUMERI DI MACCHINA e quindi occorre rappresentarli con approssimazione
  > L'idea è quella di avere il RISULTATO ESATTO, con infinite cifre e senza limiti di RANGE e poi applicare l'arrotondamento (con eventuale OVERFLOW)

Operazioni

- lo standard prevede diverse operazioni ed esamina il comportamento da intraprendere in seguito a possibili eccezioni. Qui ci limitiamo a riflettere sugli algoritmi da implementare e sui possibili eventi di cui tenere conto nel caso di NUMERI NORMALIZZATI.

PRODOTTO A·B

$$S_P = \begin{cases} \varnothing & \text{se uguali } S_A = S_B \\ 1 & \text{se diversi } S_A \neq S_B \end{cases}$$

$E_P = E_A + E_B - 127$ ci può essere overflow o underflow se $E_P > 254$ oppure $E_P < 1$

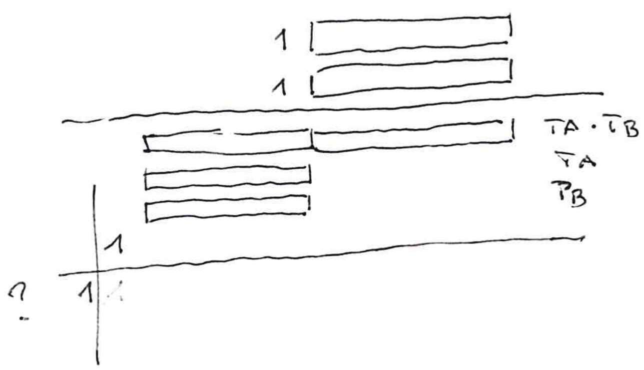
$$(1 + T_A \cdot 2^{-23})(1 + T_B 2^{-23}) = 1 + (T_A + T_B) 2^{-23} + T_A T_B 2^{-46}$$

questo risultato può venire da 1 a poco meno di 4

se viene $1 \le R < 2$ si arrotonda

$$T_P = T_A + T_B + T_A T_B \, 2^{-23}$$

se viene $2 \le R < 4$ si rinormalizza incrementando E_P



#6.6

DIVISIONE A/B

- $A = (-1)^{S_A}\, 2^{E_A-127} \left(1 + T_A 2^{-23}\right)$

  $B = (-1)^{S_B}\, 2^{E_B-127} \left(1 + T_B 2^{-23}\right)$

$$\frac{A}{B} = (-1)^{S_R}\, 2^{E_A-E_B+127-127} \qquad \frac{1+T_A 2^{-23}}{1+T_B 2^{-23}}$$

con

$S_R = 1$ se $S_A \neq S_B$
$\quad\;\; 0$ se $S_A = S_B$

$E_R = E_A - E_B + 127$ ci può essere overflow o underflow

il significando può andare

$$\frac{1}{2} < m_R < 2$$

quindi potrebbe essere necessario $(T_B > T_A)$ rinormalizzare DECREMENTANDO E

#6.7

SOMMA e DIFFERENZA $A \pm B$

Le due operazioni sono simili (per la differenza basta cambiare il segno di B). Il comportamento cambia se i segni sono CONCORDI o DISCORDI

> Segni concordi $S_R = S_A = S_B$

Sia $E_A \geq E_B$

$$A+B = (-1)^{S_R}\, 2^{E_A-127} \left[1 + T_A 2^{-23} + (1 + T_B 2^{-23})\, 2^{E_B - E_A} \right]$$

se la parentesi viene ≥ 2 (resterà sempre < 4) si incrementa E_R e si rinormalizza. Potrebbe esserci OVERFLOW

> Segni discordi con $E_A > E_B$; $S_R = S_A$

$$A-B = (-1)^{S_R}\, 2^{E_A-127} \left[1 + T_A 2^{-23} - \underbrace{(1 + T_B 2^{-23})\, 2^{E_B - E_A}} \right]$$

→ valore minimo

$E_B - E_A = -1$; $T_A = \emptyset$; $T_B = 2^{23} - 1$ si ha

$$\left[\cancel{1} - \frac{\cancel{1}}{2} - \frac{\cancel{1}}{2} + 2^{-23} \right]$$

Occorre rinormalizzare ottenendo $E_R \simeq E_A - 23$ (rischio underflow)

> Segni discordi con $E_A = E_B$; $T_A > T_B$ (altrimenti si potrebbe avere $\emptyset$)

$S_R = S_A$

$$A-B = (-1)^{S_R}\, 2^{E_A-127} \cdot (T_A - T_B) \cdot 2^{-23}$$

→ valore minimo $T_A - T_B = 1$ ed è simile al caso limite precedente con $E_R = E_A - 23$