

Representação de Dados Ponto Flutuante

Noemi Rodriguez
Ana Lúcia de Moura
Raúl Renteria

<http://www.inf.puc-rio.br/~inf1018>

Representação em Ponto Flutuante

Tipos C **float** e **double**

Codifica números racionais na forma $x * 2^y$

- adequado para valores grandes ou muito pequenos

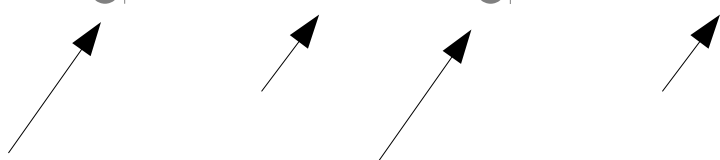
Padrão IEEE-754 (~1985)

- representação de número de ponto flutuante e suas operações
- portabilidade

Ponto Flutuante

Base 10

- números racionais $p/q \rightarrow$ mantissa $\times 10^{\text{exp}}$
- ponto (vírgula) "flutua", compensado pelo expoente

$$129 = 12\overset{\circ}{.}9 \times 10\overset{\circ}{1} = 1\overset{\circ}{.}29 \times 10\overset{\circ}{2}$$


aproximação $\rightarrow$ representação de **alguns** números

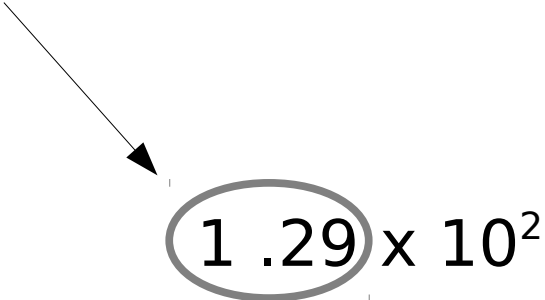
Normalização

Múltiplas representações possíveis

$$129 \times 10^0 = 12.9 \times 10^1 = 1.29 \times 10^2 = 0.129 \times 10^3 = 1290 \times 10^{-1}$$

Escolha de uma representação padrão (**normalizada**)

Base 10: $1 \leq \text{mantissa} < 10 \leftarrow \text{base}$


$$1.29 \times 10^2$$

Números Fracionários

Base 10

$$12.34_{10} = 1 * 10^1 + 2 * 10^0 + 3 * 10^{-1} + 4 * 10^{-2}$$

$\longleftarrow$ potências POSITIVAS de 10 $\longrightarrow$ potências NEGATIVAS de 10

Base 2

$$101.11_2 = 1 * 2^2 + 0 * 2^1 + 1 * 2^0 + 1 * 2^{-1} + 1 * 2^{-1}$$

$\longleftarrow$ potências POSITIVAS de 2 $\longrightarrow$ potências NEGATIVAS de 2

$$= 4 + 1 + \frac{1}{2} + \frac{1}{4} = 5.75$$

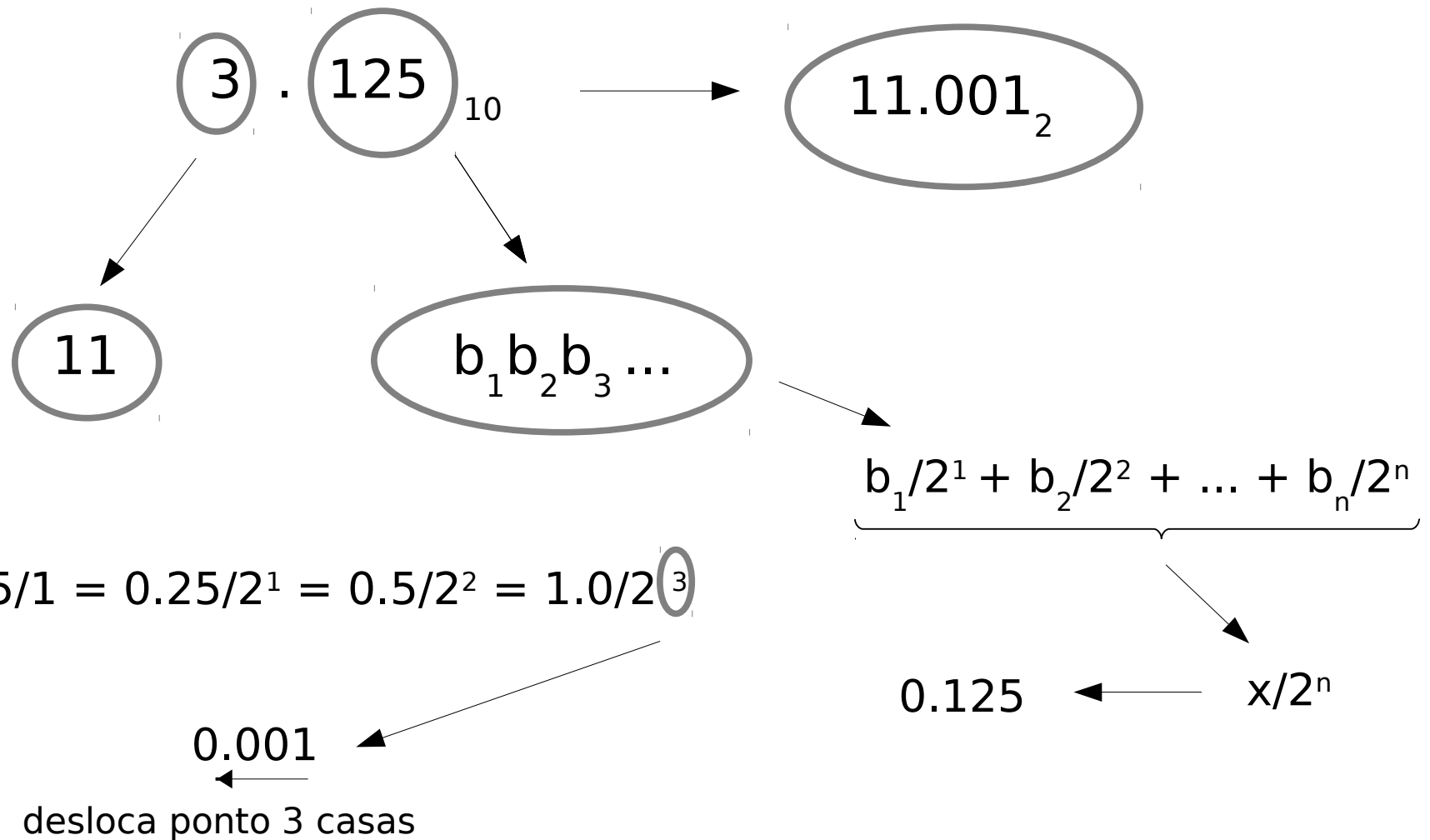
Conversão binário → decimal

$$1.001 = 1 * 2^0 + 1 * 2^{-3} = 1 + 1/8 = 1.125$$

$$101.111 = 1 * 2^2 + 1 * 2^0 + 1 * 2^{-1} + 1 * 2^{-2} + 1 * 2^{-3} = \\ 5 + 1/2 + 1/4 + 1/8 = 5.875$$

$$11.0011 = 1 * 2^1 + 1 * 2^0 + 1 * 2^{-3} + 1 * 2^{-4} = \\ 3 + 1/8 + 1/16 = 3.1875$$

Conversão decimal → binário



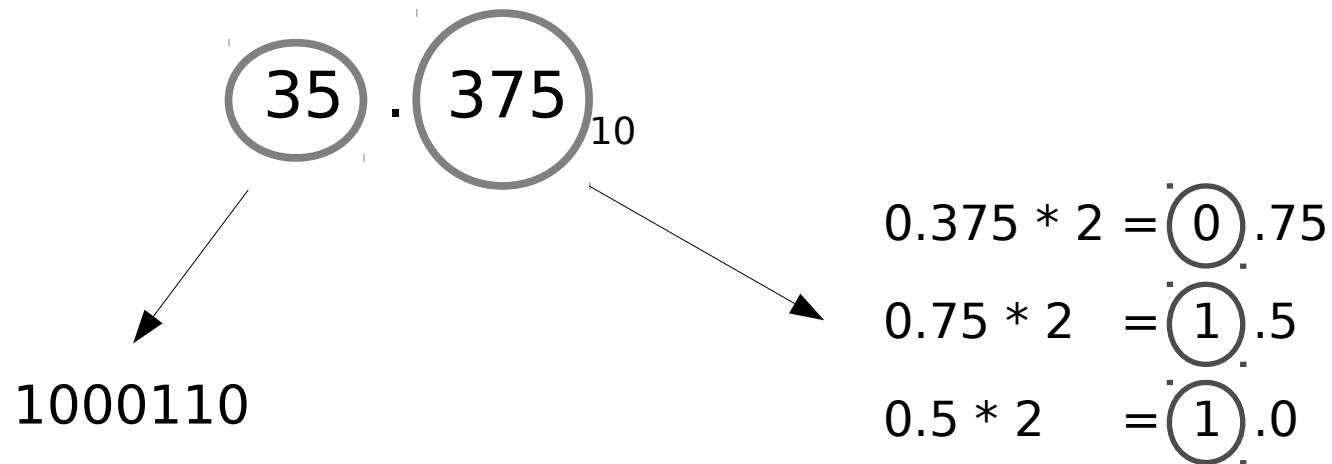
Algoritmo de Conversão

$$\begin{array}{lcl} 0.625 & = & 0.b_1b_2b_3\dots b_n \\ \downarrow *2 & & \downarrow *2 \\ \textcircled{1}.25 & = & \textcircled{b1}.b_2b_3\dots b_n \longrightarrow \boxed{0.1} \end{array}$$

$$\begin{array}{lcl} 0.25 & = & 0.b_2b_3\dots b_n \\ \downarrow *2 & & \downarrow *2 \\ \textcircled{0}.5 & = & \textcircled{b2}.b_3\dots b_n \longrightarrow \boxed{0.10} \end{array}$$

$$\begin{array}{lcl} 0.5 & = & 0.b_3\dots b_n \\ \downarrow *2 & & \downarrow *2 \\ \textcircled{1}.0 & = & \textcircled{b3}.b_4\dots b_n \longrightarrow \boxed{0.101} \end{array}$$

Conversão decimal → binário



1000110.011_2

Limitações da Representação

A notação decimal representa números que podem ser escritos como $\sum_{k=-n}^m d_k \cdot 10^k$

não há representação precisa para $1/3$ ou $5/7$

A notação binária representa números que podem ser escritos como

$$\sum_{k=-n}^m b_k \cdot 2^k$$

Outros números devem ser **aproximados**

parte fracionária possui sequências repetidas indefinidamente

Exemplo de Sequência Infinita

$$0.3 * 2 = 0.6$$

$$0.6 * 2 = 1.2$$

$$0.2 * 2 = 0.4$$

$$0.4 * 2 = 0.8$$

$$0.8 * 2 = 1.6$$

$$0.6 * 2 = 1.2$$

...

0.0 1001 1001 1001

mais bits, maior precisão

Representação IEEE 754

Representa números na forma $x * 2^y$

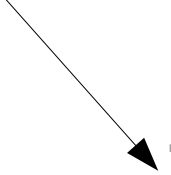
com um par adequado (x,y)

Existem várias representações possíveis

$$1.5_{10} = 1.1_2 \times 2^0 = 11.0_2 \times 2^{-1} = 0.11_2 \times 2^1$$

Normalização

Base 2: $1 \leq \text{mantissa} < 2$ ← base


$$1.1_2 \times 2^0$$

Codificação: Forma Numérica

$$(-1)^s M 2^E$$

Bit de sinal **s** determina se número é negativo ou positivo

Mantissa **M** é um valor **fracionário** $1 \leq M < 2$

Expoente **E** indica potência positiva ou negativa de 2

Codificação

$$(-1)^s M 2^E$$

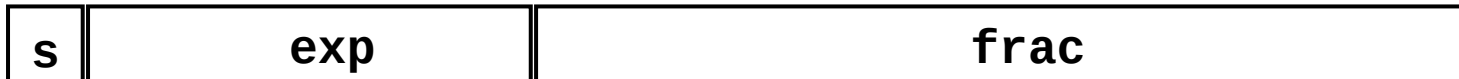


Bit **s** indica sinal (1 : n° negativo, 0 : n° positivo)

Campo **exp** codifica **E**

Campo **frac** codifica **M**

Padrão IEEE: Precisão



float (32 bits): **exp** = 8 bits, **frac** = 23 bits, **s** = 1 bit

faixa de valores: 2^{-126} até 2^{127}

double (64 bits): **exp** = 11 bits, **frac** = 52 bits, **s** = 1 bit

faixa de valores: 2^{-1022} até 2^{1023}

1

Representação de Valores



frac = M - 1 ($1 \leq M < 2$) $\longleftarrow$ normalização

1 é implícito $\longrightarrow M = 1 . \underbrace{b_1 b_2 b_3 \dots}_{\text{frac}}$

exp = E + bias $\longleftarrow$ representação em excesso 2^n

exp é um valor sem sinal e **bias** = $2^{n-1} - 1$

float (precisão simples) $\rightarrow$ **bias** = **127** ($2^{8-1} - 1 = 2^7 - 1 = 127$)

double (precisão dupla) $\rightarrow$ **bias** = **1023** ($2^{11-1} - 1 = 2^{10} - 1 = 1023$)

Exemplo 1: Precisão Simples

Como representar um número **X** em float?

float f = 15213.0;

$$15213_{10} = 11101101101101_2 = \underbrace{1.1101101101101_2}_M * 2^{\textcircled{13}} \quad \swarrow E$$

← 13 casas para a esquerda

frac (23 bits) = 110110110110100000000000₂

exp (8 bits) = E + bias = 13 + 127 = 140 = 1001100₂

s = 0

31	30	23	22	0
0	10001100	1101101101101	000000000000	

Binário	0100	0110	0110	1101	1011	0100	0000	0000
Hex:	4	6	6	D	B	4	0	0

Exemplo 2: Precisão Simples

Como encontrar o valor de um número **X** representado em IEEE 754?

Hex:	4	6	4	0	E	4	0	0
Binário	0100	0110	0100	0000	1110	0100	0000	0000

31	30	23	22	0
0	10001100	100000011100	100000000000	

$$\mathbf{frac} = 1000000111001_2 \rightarrow \mathbf{M} = 1.1000000111001_2$$

$$\mathbf{exp} = 10001100_2 \rightarrow 140 \rightarrow \mathbf{E} = 140 - 127 = 13$$

$$\mathbf{s} = 0$$

$$\mathbf{V} = (-1)^0 * (1 + 4153/8192) * 2^{13} = 12345_{10}$$

Exemplo 3: Precisão Dupla

double d = 178.125;

$$178.125_{10} = 10110010.001_2 = \underbrace{1.0110010001_2}_M * 2^{\overset{\cdot}{\underset{\cdot}{\textcircled{7}}}}_E$$

← 7 casas para a esquerda
M
E

frac (52 bits) = 0110010001000...000₂

exp (11 bits) = E + bias = 7 + 1023 = 1030 = 10000000110₂

s = 0

63	62	52	51	0
0	10000000110	0110010001000000...0000		

Binário	0100	0000	0110	0110	0100	0100	0000	0000	...	0000
Hex:	4	0	6	6	4	4	0	0	00	00 00 00 00

Valores Especiais

zero	$S = 0$	$\text{exp} = 0$	$M = 0$
$+\infty$	$S = 0$	$\text{exp} = 111\dots 111$	$M = 0$
$-\infty$	$S = 1$	$\text{exp} = 111\dots 111$	$M = 0$
NaN	S	$\text{exp} = 111\dots 111$	$M \neq 0$

infinito: permite representação de *overflow*

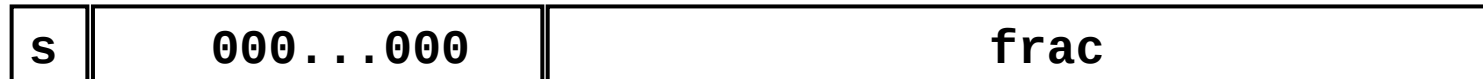
- multiplicação, divisão por 0

NaN (*not a number*)

- operações que não resultam em um número real ($\sqrt{-1}$, ∞ , $-\infty$)

em math.h: NAN, INFINITY

Valores Denormalizados



Representação de 0 e números muito próximos de 0

M = frac → não há "1" implícito ($0 \leq M < 1$)

E = 1 - bias → -126 ou -1022

$$(-1)^s * 0.b_1 b_2 b_3 \dots * 2^{-126}$$

$$(-1)^s * 0.b_1 b_2 b_3 \dots * 2^{-1022}$$