12-09-22

Objetivos:

1. Representação de números no computador
    1. Inteiros
    2. Reais
    3. IEEE 754 - revisão de 2008
2. Limitações
    1. Range
    2. Erros de arredondamento
    3. Overflow
    4. Underflow

Por que você deveria se importar?


```
(base) filipem@pop-os:~$ python3
Python 3.9.12 (main, Apr  5 2022, 06:56:58)
[GCC 7.5.0] :: Anaconda, Inc. on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> print(0.1+0.2)
0.30000000000000004
>>> 
```

Por que aconteceu?

Tipo de representação escolhido! Imagine o que possa acontecer se você acumular a diferença de milhares de operações dessas?

Como podemos representar números?

Inteiros:

Número fixo de bits: 32 bits

6 bits:

$$[1]\ [0]\ [1]\ [0]\ [1]\ [0] = [1 \times 32]\ [0 \times 16]\ [1 \times 8]\ [0 \times 4]\ [1 \times 2]\ [0 \times 1] = 42_{10}$$

Reais

Diferente inteiros entre $(0, 1)$ existem **infinitos números!**

Temos que escolher o que queremos representar: **nem muito grandes nem muito pequenos**

Comparação: Notação científica

$$300000000 \times 0.00000015 = 45$$

Notação científica:

$$Mantissa \times Base^{expoente}$$

> Do latim, mantissa significa a parte quebrada, ou o excedente do peso.

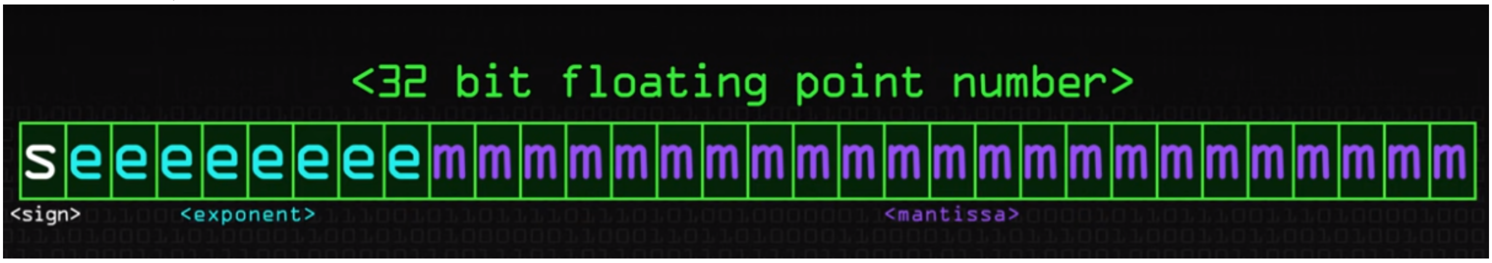
$$3 \times 10^8 * 1.5 \times 10^{-7} = (3 * 1.5) \times 10^{(8+(-7))} = 45$$

Facilita as operações, introduz a noção de algarismos significativos.
Com notação cientifica ampliamos os ranges de números que podemos representar, mas criamos novas limitações também
Limitações pelo número de **algarismos significativos** que podemos representar

Essas limitações aparecem no dia a dia, quando aproximamos constantes físicas $9.98 \approx 10$.

Convencionalmente representamos os números com floats de 32 bits, com 23 bits de mantissa. E 56 em floats de 64 bits. Mas a base da notação é 2 e não 10.

A representação completa seria: 1 bit sinal + 8 bits expoente + 23 bits de mantissa



23 bits define o número máximo de algarismos significativos que podemos representar! Depois disso há arredondamento!

Erro de arredondamento

$$0.\overline{3333333333} + 0.\overline{3333333333} + 0.\overline{3333333333} = 0.\overline{9999999999}$$

Uma forma de resolver seria remapear esses intervalos de décimos e centésimos para inteiros e depois dividir pelo número adequado.

Valores máximos e mínimos

Underflow

Valor abaixo do mínimo representável

$$1 + \varepsilon = 1$$

Overflow

Valor maior que o máximo representável. Coisas estranhas acontecem!

Referências

[Floating point numbers: Floating and Fixed]
[IEEE 754 Revisão de 2008]