Aritmética finita

Computación 2021

FaMAF

17 mar. 2021

Contenidos

Aritmética finita

Claramente, ustedes no dudarían en afirmar que

- $(\sqrt{3})^2 = 3$
- $0.3 - 0.1 = 0.2$
- $1 + 2^{-53} > 1$
- $1 + 2^{-53} + 2^{-53} = 2^{-53} + 2^{-53} + 1$

Pero en la computadora esto puede no ser así. Veamos que ocurre en Python (ver notebook)

Aritmética de punto flotante

Como ya lo mencionamos anteriormente, los números se almacenan en la memoria de la computadora como una cadena de ceros y unos. El sistema más usado es la **aritmética de punto flotante de doble precisión**. Existen dispositivos que utilizan aritmética de punto fijo y otros aritmética de punto flotante de precisión simple, pero no trabajaremos con ellos en este curso.
En la aritmética de punto flotant de doble precisión, cada número se representa utilizando 64 bits

$$\underbrace{0}_{s}\ \underbrace{10000000011}_{c}\ \underbrace{1011100100010000000000000000000000000000000000000000}_{f}$$

partido en 3 binarios: * el signo s de 1 dígito, * el exponente c, un entero con representación binaria de 11 dígitos excluyendo 00000000000 y 11111111111, y * la mantisa f, real menor a uno con representación fraccionaria binaria de 52 dígitos.
Para la conversión de binarios a decimales es bueno recordar que

$$\sum_{i=0}^{n} r^i = \frac{1-r^{n+1}}{1-r}, \quad \sum_{i=1}^{n} r^i = \frac{r(1-r^n)}{1-r}.$$

Aritmética de punto flotante

Como el exponente c cumple

$$0 < c < \sum_{i=0}^{10} 2^i = 2^{11} - 1 = 2047,$$

para poder representar números con exponente negativo, a c se le resta 1023, obteniendo exponentes enteros en el intervalo $(-1023, 1024)$. Además, para tener siempre una mantisa con parte entera igual a uno, sumaremos 1 a f. Con esto, los números no nulos representables en esta aritmética son

$$x = (-1)^s 2^{c-1023}(1+f).$$

Se define al cero tomando $c = 0$ y $f = 0$, obteniendo dos representaciones: cuando $s = 0$ y cuando $s = 1$.

Aritmética de punto flotante: ejemplo

Escribamos el número real cuya representación en 64 bits es

$$\underbrace{0}_{s}\ \underbrace{10000000011}_{c}\ \underbrace{1011100100010000000000000000000000000000000000000000}_{f}$$

Por un lado tenemos el exponente

$$\begin{aligned} c &= 1\cdot 2^{10}+0\cdot 2^{9}+0\cdot 2^{8}+0\cdot 2^{7}+0\cdot 2^{6}+0\cdot 2^{5}+0\cdot 2^{4}+0\cdot 2^{3}+0\cdot 2^{2}+ \\ &= 1024+2+1=1027. \end{aligned}$$

Por otro, la mantisa

$$\begin{aligned} f &= 1\cdot\left(\frac{1}{2}\right)^{1}+0\cdot\left(\frac{1}{2}\right)^{2}+1\cdot\left(\frac{1}{2}\right)^{3}+1\cdot\left(\frac{1}{2}\right)^{4}+1\cdot\left(\frac{1}{2}\right)^{5}+0\cdot\left(\frac{1}{2}\right)^{6} && (3)\\ &\quad +0\cdot\left(\frac{1}{2}\right)^{7}+1\cdot\left(\frac{1}{2}\right)^{8}+\ldots+1\cdot\left(\frac{1}{2}\right)^{12}+\ldots+0\cdot\left(\frac{1}{2}\right)^{52} && (4)\\ &= \frac{1}{2}+\frac{1}{8}+\frac{1}{16}+\frac{1}{32}+\frac{1}{256}+\frac{1}{4096} && (5) \end{aligned}$$

De esto obtenemos que el número real es

$$\left(\quad\right)$$

SSS

Claramente $\sqrt{3}$ no tiene una cantidad finita de dígitos decimales y por lo tanto no puede representarse en esta aritmética. En este caso Python utiliza el representante más próximo dentro de la aritmética.

```
from math import sqrt
a = sqrt(3)
print(a)
```

El cuadrado de este representante no es igual a 3. Por otro lado, como 3^2 si pertenece a la aritmética, tenemos que:

```
sqrt(3**2)==3
```

Ejemplo

Con un par de cálculos algebráicos podemos ver que

$$\sum_{i=1}^{\infty} \left(\frac{1}{2}\right)^{4i} + \left(\frac{1}{2}\right)^{4i+1} = \frac{3}{2} \sum_{i=1}^{\infty} \left(\frac{1}{2^4}\right)^{i} = \frac{3}{2(2^4-1)} = \frac{1}{10}.$$

Por lo tanto 0.1 no tiene representación binaria finita y no puede pertenecer a esta aritmética.

$$0.1 = \left(\frac{1}{2}\right)^{4} + \left(\frac{1}{2}\right)^{5} + \left(\frac{1}{2}\right)^{8} + \left(\frac{1}{2}\right)^{9} + \left(\frac{1}{2}\right)^{12} + \left(\frac{1}{2}\right)^{13} + \ldots$$

Una nube de puntos, eso es lo que obtendríamos si graficamos sobre la recta real los números representables en la aritmética de punto flotante de doble precisión. Para tener una mejor descripción de esta nube, veamos algunas propiedades:

* el número positivo más pequeño se obtiene tomando $s = 0$, $c = 1$, $f = 0$ y es

$$2^{-1022} \approx 2.2251 \times 10^{-308},$$

* el número positivo más grande se obtiene tomando $s = 0$, $c = 2046$, $f = 1 - 2^{-52}$ y es

$$2^{1023}(1 + 1 - 2^{-52}) \approx 1.7977 \times 10^{308},$$

* es simétrica en relación al cero.

Cuando el resultado de un cálculo es un número de magnitud menor a la representable, se está en presencia de un **underflow**. En general el valor se suele sustituir por cero y los cálculos continúan. Cuando el resultado es un número de magnitud mayor a la representable, se está en presencia de un **overflow**. Generalmente los cálculos se detienen.

¿Qué podemos decir sobre la separación entre puntos de esa nube? Para entender esto deberemos introducir un valor conocido como *épsilon de la máquina*, que suele denotarse con ϵ. Formalmente, ϵ es el número positivo

Error de representación

El representante en punto flotante de un número x, denotado por $\mathrm{fl}(x)$, se define como aquél que minimiza la distancia entre x y los puntos de la aritmética. En el caso de existir dos posibilidades (cuando x es justo el punto medio entre dos representantes) se toma el representante de mayor magnitud. Este método para elección de representante se conoce como **redondeo** y es el más usado.
Por ejemplo, si tenemos tres números positivos consecutivos en la aritmética $a < b < c$, tendremos

$$\mathrm{fl}(x) = b \quad \text{si } x \in \left[\frac{a+b}{2} , \frac{b+c}{2} \right) .$$

Por ejemplo, si tenemos tres números positivos consecutivos en la aritmética $a < b < c$, tendremos

$$\text{fl}(x) = b \quad \text{si } x \in \left[\frac{a+b}{2} , \frac{b+c}{2} \right) .$$

Otro método existente para elegir un representante, conocido como **truncamiento** asigna siempre el representante de menor magnitud. O sea, dados dos números positivos en la aritmética $a < b$, tendremos

$$\text{fl}(x) = a \quad \text{si } x \in [a , b) .$$

Cada vez que un número x no pertenezca a la aritmética, estaremos generando un error de redondeo. Llamaremos * **error absoluto** al valor $|x - \text{fl}(x)|$, * **error relativo** al valor $\dfrac{|x - \text{fl}(x)|}{|x|}$.

Por otro lado, las operaciones se van ejecutando de izquierda a derecha y el resultado de cada operación es un elemento de la aritmética. Con esto, la operación

$$x + y + z$$

dará como resultado

$$\mathrm{fl}\left(\mathrm{fl}\left(\mathrm{fl}(x) + \mathrm{fl}(y)\right) + \mathrm{fl}(z)\right).$$

Ejemplo:
Al ejecutar $1 + 2^{-53} + 2^{-53}$ se obtiene

$$\text{fl}\left(\text{fl}\left(\text{fl}(1) + \text{fl}(2^{-53})\right) + \text{fl}(2^{-53})\right) = \text{fl}\left(\text{fl}\left(1 + 2^{-53}\right) + 2^{-53}\right) = \text{fl}\left(1 + 2^{-53}\right) =$$

ya que 2^{-53} es menor que el épsilon de la máquina.
Al ejecutar $2^{-53} + 2^{-53} + 1$ se obtiene

$$\text{fl}\left(\text{fl}\left(\text{fl}(2^{-53}) + \text{fl}(2^{-53})\right) + \text{fl}(1)\right) = \text{fl}\left(\text{fl}\left(2^{-53} + 2^{-53}\right) + 1\right) = \text{fl}\left(2^{-52} + 1\right) =$$

ya que 2^{-52} es el épsilon de la máquina.

```
a=2**(-53)
a+a+1>1+a+a
```

Bucles y listas

Funciones