Floating-point numbers

Ágnes Baran, Csaba Noszály

Floating point numbers

Example 1

$a = 10$

$$0.3721 = \frac{3}{10} + \frac{7}{10^2} + \frac{2}{10^3} + \frac{1}{10^4}$$

$$21.65 = 0.2165 \cdot 10^2 = \left(\frac{2}{10} + \frac{1}{10^2} + \frac{6}{10^3} + \frac{5}{10^4}\right) \cdot 10^2$$

$a = 2$

$$0.1101 = \frac{1}{2} + \frac{1}{2^2} + \frac{0}{2^3} + \frac{1}{2^4}$$

$$0.001011 = 0.1011 \cdot 2^{-2} = \left(\frac{1}{2} + \frac{0}{2^2} + \frac{1}{2^3} + \frac{1}{2^4}\right) \cdot 2^{-2}$$

Floating point numbers

The form of non-zero floating point numbers:

$$\pm a^k \left(\frac{m_1}{a} + \frac{m_2}{a^2} + \cdots + \frac{m_t}{a^t} \right)$$

or in a shorter notation

$$\pm a^k \cdot 0.m_1 \ldots m_t$$

or if a is known

$$\pm |k| m_1, \ldots, m_t$$

where

$a > 1$ is an integer, the `base`,

$t > 1$ is an integer, the length of the `mantissa`

$k_- \leq k \leq k_+$ are integers, k is the `characteristic`, $k_- < 0$ and $k_+ > 0$ are fixed.

$0 \leq m_i \leq a - 1$ is an integer, for $i = 1, \ldots, t$

If $m_1 > 0$ then the number is in *normalized* form. It makes the representation *unique*. Usually we will consider only normalized floating point numbers. The number 0 is not a normalized floating point number!

The set of the representable numbers is uniquely determined by the numbers

$$a, t, k_-, k_+$$

This set is denoted by $\mathcal{F}_{a,k_-,k_+,t}$ or simply by $\mathcal{F}$ if the parameters are fixed.

Example 2

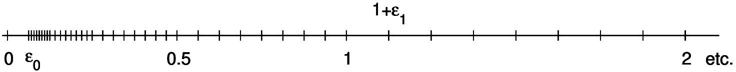
Let $a = 2$, $t = 4$, $k_- = -3$, $k_+ = 2$.

- Compute the floating-point form the numbers below:

$$0.6875, \quad 0.8125, \quad 3.25, \quad 0.875$$

- Of how many positive, normalized numbers can be represented in the given system?

For $a = 2,\ t = 4,\ k_- = -3,\ k_+ = 3$ the (partial) pictorial representation of the corresponding system is:



$\varepsilon_0 = a^{k_- - 1} = 2^{-4} = \frac{1}{16}$,

$\varepsilon_1 = a^{1-t} = 2^{-3} = \frac{1}{8}$

Fact 1

For a given a, t, k_-, k_+

- the `largest` (positive) representable number:

$$M_\infty = a^{k_+}\left(\frac{a-1}{a}+\frac{a-1}{a^2}+\cdots+\frac{a-1}{a^t}\right) = \\ = a^{k_+}\left(1-\frac{1}{a}+\frac{1}{a}-\frac{1}{a^2}+\cdots+\frac{1}{a^{t-1}}-\frac{1}{a^t}\right) = \\ = a^{k_+}\left(1-a^{-t}\right)$$

- the `smallest` (positive) representable number:

$$\varepsilon_0 = a^{k_-}\left(\frac{1}{a}+0+\cdots+0\right) = a^{k_- -1}$$

- `subnormal` numbers: if $k = k_-$ and $m_1 = 0$.

Fact 2

The number `1` is always representable:

$$\boxed{1 = a^1 \cdot \frac{1}{a}}$$

or

$$\boxed{1 = [+|1|1, 0, \dots, 0]}$$

The `right` neighbour of 1 (denoted by 1_+):

$$\boxed{1 + \varepsilon_1 = [+|1|1, 0, \dots, 0, 1]}$$

or

$$\boxed{1 + \varepsilon_1 = a\left(\frac{1}{a} + 0 + \dots + 0 + \frac{1}{a^t}\right) = 1 + a^{1-t}}$$

that is $\varepsilon_1 = a^{1-t}$ (**the machine epsilon**)

The `left` neighbour of 1 (denoted by 1_-):

$$1_- = a^0 \cdot 0.(a-1)\ldots(a-1) = 1 - a^t$$

That is, the largest number from the previous characteristic. For powers of `a` the left and right neighbours are in different distance!

The IEEE floating point standard:

	single precision	double precision
size	32 bits	64 bits
mantissa	23+1 bits	52+1 bits
characteristic	8 bits	11 bits
ε_1	$\approx 1.19 \cdot 10^{-7}$	$\approx 2.22 \cdot 10^{-16}$
M_∞	$\approx 10^{38}$	$\approx 10^{308}$

Note that here m_1 is 1 (a constant), so it is not stored explicitly. For the sign 1 bit is reserved.

Example 3

The set of all positive normalized numbers in the system

$$a = 2,\ t = 4,\ k_- = -3,\ k_+ = 2$$

	$k=-3$	$k=-2$	$k=-1$	$k=0$	$k=1$	$k=2$
0.1000	$\frac{8}{128}$	$\frac{8}{64}$	$\frac{8}{32}$	$\frac{8}{16}$	$\frac{8}{8}$	$\frac{8}{4}$
0.1001	$\frac{9}{128}$	$\frac{9}{64}$	$\frac{9}{32}$	$\frac{9}{16}$	$\frac{9}{8}$	$\frac{9}{4}$
0.1010	$\frac{10}{128}$	$\frac{10}{64}$	$\frac{10}{32}$	$\frac{10}{16}$	$\frac{10}{8}$	$\frac{10}{4}$
0.1011	$\frac{11}{128}$	$\frac{11}{64}$	$\frac{11}{32}$	$\frac{11}{16}$	$\frac{11}{8}$	$\frac{11}{4}$
0.1100	$\frac{12}{128}$	$\frac{12}{64}$	$\frac{12}{32}$	$\frac{12}{16}$	$\frac{12}{8}$	$\frac{12}{4}$
0.1101	$\frac{13}{128}$	$\frac{13}{64}$	$\frac{13}{32}$	$\frac{13}{16}$	$\frac{13}{8}$	$\frac{13}{4}$
0.1110	$\frac{14}{128}$	$\frac{14}{64}$	$\frac{14}{32}$	$\frac{14}{16}$	$\frac{14}{8}$	$\frac{14}{4}$
0.1111	$\frac{15}{128}$	$\frac{15}{64}$	$\frac{15}{32}$	$\frac{15}{16}$	$\frac{15}{8}$	$\frac{15}{4}$

$M_\infty = 2^2(1 - 2^{-4}) = \frac{15}{4}$ and $\varepsilon_0 = 2^{-3-1} = \frac{1}{16}\ \left(= \frac{8}{128}\right)$

Let $y = a^k \cdot 0.m_1 m_2 ... m_t$.

The closest number that is greater than y is denoted by y_+ and:

$$y_+ = y + a^k \cdot \frac{1}{a^t} = y + a^{k-t}$$

This number is called the *stepsize* of the given characteristic.

Larger characteristic (exponent) means larger distance (stepsize) between neighbouring numbers.

If $k > t$, then the stepsize is larger than 1.

For double precision ($t = 53$):

y	distance of the right neighbour
1	$\approx 2.22 \cdot 10^{-16}$
16	$\approx 3.5527 \cdot 10^{-15}$
1024	$\approx 2.27 \cdot 10^{-13}$
$2^{20} \approx 10^6$	$\approx 2.33 \cdot 10^{-10}$
$2^{52} \approx 4.5 \cdot 10^{15}$	1
$2^{60} \approx 1.15 \cdot 10^{18}$	256
$2^{66} \approx 7.38 \cdot 10^{19}$	16384

Rounding

Not all numbers has an exact representation in a floating point number system.

Example 4

The binary representation of $\frac{1}{10}$:

$$0.0001100110011001100\ldots.$$

The binary representation of $\frac{1}{3}$:

$$0.0101010101010\ldots.$$

Rounding

Let $x \in [-M_\infty, M_\infty]$ a real number, and denote by $fl(x)$ the corresponding floating-point number.

`Regular rounding`

$$fl(x) = \begin{cases} 0, & \text{if } |x| < \varepsilon_0 \\ \text{among the nearest floating point} & \\ \text{numbers to } x\text{, the larger} & \\ \text{in absolute value,} & \text{if } |x| \geq \varepsilon_0 \end{cases}$$

`Cutting, choping`

$$fl(x) = \begin{cases} 0, & \text{if } |x| < \varepsilon_0 \\ \text{the nearest floating point} & \\ \text{number towards zero,} & \text{if } |x| \geq \varepsilon_0 \end{cases}$$

Remark 1

The rounding rules implemented in todays processors are more involved. For simplicity we will use the rules above.

Example 5

Let $a = 2$, $t = 4$, $k_- = -3$, $k_+ = 2$. What is $fl(0.1)$ in case of choping and regular rounding?

From the binary expansion of 0.1, we get the form:

$$2^{-3} \cdot 0.1100110011001100....$$

Regular rounding:

$$fl(0.1) = 2^{-3} \cdot 0.1101$$

Choping:

$$fl(0.1) = 2^{-3} \cdot 0.1100$$

Rounding

Estimating the `absolute` error
in case of `regular rounding`:

$$|fl(x) - x| \leq \begin{cases} \varepsilon_0, & \text{ha } |x| < \varepsilon_0 \\ \frac{1}{2}\varepsilon_1 |x|, & \text{ha } |x| \geq \varepsilon_0 \end{cases}$$

in case of `choping`:

$$|fl(x) - x| \leq \begin{cases} \varepsilon_0, & \text{ha } |x| < \varepsilon_0 \\ \varepsilon_1 |x|, & \text{ha } |x| \geq \varepsilon_0 \end{cases}$$

Rounding

Estimating the `relative` error
in case of `regular rounding`:

$$\boxed{\frac{|fl(x) - x|}{|x|} \le \frac{1}{2}\varepsilon_1}$$

in case of `choping`:

$$\boxed{\frac{|fl(x) - x|}{|x|} \le \varepsilon_1}$$

Addition

Example 6

Let $a = 10$, $t = 3$. Assuming 1 spare digit compute $fl(x+y) =!$
$x = 0.425 \cdot 10^{-1}$, $y = 0.677 \cdot 10^{-2}$

$y \to y = 0.0677 \cdot 10^{-1}$ $\quad$ (**1 spare digit**)

$x + y = 0.425 \cdot 10^{-1} + 0.0677 \cdot 10^{-1} = 0.4927 \cdot 10^{-1}$

$$fl(x+y) = \begin{cases} 0.492 \cdot 10^{-1}, & \text{choping} \\ 0.493 \cdot 10^{-1}, & \text{regular rounding} \end{cases}$$

Error and operations

Denote by $\triangle$ one of the `+,-,*/`, let x and y floating point numbers. Assuming that the computer performs the operations exactly and assigns a floating point number to the result. Then
in case of regular rounding we have:

$$|fl(x\triangle y) - x\triangle y| \leq \begin{cases} \varepsilon_0, & \text{if } |x\triangle y| < \varepsilon_0 \\ \frac{1}{2}\varepsilon_1|x\triangle y|, & \text{if } |x\triangle y| \geq \varepsilon_0 \end{cases}$$

in case of choping we have:

$$|fl(x\triangle y) - x\triangle y| \leq \begin{cases} \varepsilon_0, & \text{if } |x\triangle y| < \varepsilon_0 \\ \varepsilon_1|x\triangle y|, & \text{if } |x\triangle y| \geq \varepsilon_0 \end{cases}$$

$$|x \triangle y| > M_\infty \quad \Longrightarrow \quad \textbf{overflow}$$
$$|x \triangle y| < \varepsilon_0 \quad \Longrightarrow \quad \textbf{underflow}(fl(x \triangle y) = 0)$$