## Section 1.2 Binary Numbers

How are numbers stored on the computer?
First, remember what we call "Scientific Notation". For any decimal number $x$ (assume finite nonzero digits) we can write:

$$x = a * 10^b \tag{1}$$

where $1 \leq |a| < 10$
Exception: when $x = 0$, we simply set $a = b = 0$
Example:

| x (decimal notation) | x (scientific notation) |
|---|---|
| 2011 | $2.011 * 10^3$ |
| 412 | $4.12 * 10^2$ |
| 3.14 | $3.14 * 10^0$ |
| 0.000789 | $7.89 * 10^{-4}$ |
| 0.2091 | $2.091 * 10^{-1}$ |

Every decimal (or base-10) number can be written as:

$$a_k ... a_3 a_2 a_1 a_0 . a_{-1} a_{-2} a_{-3} ... a_{-l} = \sum_{i=-l}^{k} a_i * 10^i$$

where $a_i \in \{0, 1, 2, ..., 9\}$
Example:

| | $a_3$ | $a_2$ | $a_1$ | $a_0$ | . | $a_{-1}$ | $a_{-2}$ | $a_{-3}$ | |
|---|---|---|---|---|---|---|---|---|---|
| 3.14 | | | | 3 | . | 1 | 4 | | $= 3 * 10^0 + 1 * 10^{-1} + 4 * 10^{-2}$ |
| 0.037 | | | | | . | 0 | 3 | 7 | $= 3 * 10^{-2} + 7 * 10^{-3}$ |
| 2011 | 2 | 0 | 1 | 1 | . | | | | $= 2 * 10^3 + 1 * 10^1 + 1 * 10^0$ |

Binary numbers (base-2) are written as:

$$b_k...b_3b_2b_1b_0.b_{-1}b_{-2}b_{-3}...b_{-l} = \sum_{i=-l}^{k} b_i * 2^i \qquad (2)$$

where $b_i \in \{0, 1\}$
Examples:

$$\begin{aligned} 5.75 &= 4 + 1 + 0.5 + 0.25 \\ &= 1 * 2^2 + 1 * 2^0 + 1 * 2^{-1} + 1 * 2^{-2} = 101.11_{(2)} \end{aligned}$$

$$\begin{aligned} 17.5 &= 16 + 1 + 0.5 \\ &= 1 * 2^4 + 1 * 2^0 + 1 * 2^{-1} = 10001.1_{(2)} \end{aligned}$$

Note that certain numbers which are finite in decimals, actually are periodic in binary
e.g. $0.4_{(10)} = 0.011\underbrace{0011}\underbrace{0011}...{}_{(2)} = 0.011\overline{0011}_{(2)}$

## Machine numbers (binary floating point numbers)

The numbers stored on the computer are, essentially "binary numbers" in scientific notation

$$x = \pm a * 2^b$$

a is called the mantissa
b is called the exponent

The convention here is that $\frac{1}{2} \leq a < 1$. The idea is that, for any number x, we can always divide it by an appropriate power of 2, such that the result will be within $[\frac{1}{2}, 1)$.

e.g. $x = 5.0$
Try

$$b = 0 \quad 5 = x = a * 2^0 \Rightarrow a = 5 \text{ too much}$$
$$b = 2 \quad 5 = x = a * 2^2 \Rightarrow a = 1.25 \text{ still too much}$$
$$b = 3 \quad 5 = x = a * 2^3 \Rightarrow a = 0.625 \text{ OK!}$$

Lastly, we have to write $a = 0.625$ in binary

$$0.625 = 0.5 + 0.125 = 1 * 2^{-1} + 1 * 2^{-3} = 0.101_{(2)}$$

Thus

$$5.0 = x = 0.101_{(2)} * 2^3$$

How about a different way?
Remember, in decimal:

$$3.14 = 31.4 * 10^{-1} = 0.314 * 10^1 = ...etc$$

In binary

$$\begin{aligned} 5_{(10)} &= 101_{(2)} \\ &= 10.1 * 2^1 = 1.01 * 2^2 = 0.101 * 2^3 \end{aligned}$$

In general, a machine number is stored as:

$$x = \pm 0.1 a_1 a_2 ... a_{k-1} a_k * 2^b \qquad (3)$$

Single Precision

$$k = 23 \qquad -126 \leq b \leq 127$$

(max number $\approx \pm 3.4 * 10^{38}$)

Double Precision

$$k = 52 \qquad -1022 \leq b \leq 1023$$

(max number $\approx \pm 1.8 * 10^{308}$)

Single precision "23 binary significant digits". How much is that in decimal?

Rule of thumb:

$$2^{10} \approx 10_3$$

i.e. 10 binary significant digits $\approx$ 3 decimal

Thus: single precision $\Rightarrow$ about 7 decimal significant digits

double precision $\Rightarrow$ a bit more than 15

## Section 1.3 Error Analysis

All computations are approximate, due to limited precision on computer.

Exact quantity: p

Quantity on the computer: $\hat{p}$

2 measures of Error:

Absolute error $\qquad E_p = |p - \hat{p}|$

Important when we want to examine accuracy within a certain interval

Relative error $\qquad R_p = \frac{|p-\hat{p}|}{|p|}$

Important when we care about error % of actual value

Example:
Let $x = 1,000,000$ and $\hat{x} = 999,996$

$$E_x = |x - \hat{x}| = |1,000,000 - 999,996| = 4$$

and

$$R_x = \frac{|x-\hat{x}|}{|x|} = \frac{4}{1,000,000} = 0.000004$$

Let $y = 0.000012$ and $\hat{y} = 0.000009$

$$E_y = |y - \hat{y}| = |0.000012 - 0.000009| = 0.000003$$

and

$$R_y = \frac{|y-\hat{y}|}{|y|} = \frac{0.000003}{0.000012} = 0.25$$

**Definition:** The number $\hat{p}$ is said to approximate $p$ to $d$ significant digits if $d$ is the largest positive integer for which

$$\frac{|p-\hat{p}|}{|p|} < \frac{10^{-d}}{2}$$

In previous example,
$\frac{|x-\hat{x}|}{|x|} = 0.000004 < \frac{10^{-5}}{2}$ therefore $\hat{x}$ approximates $x$ to 5 significant digits.
$\frac{|y-\hat{y}|}{|y|} = 0.25 < \frac{10^{-0}}{2}$ therefore $\hat{y}$ approximates $y$ to no significant digits.

Let $z = 3.141592$ and $\hat{z} = 3.14$ then $\frac{|z-\hat{z}|}{|z|} = 0.000507 < \frac{10^{-2}}{2}$ therefore $\hat{z}$ approximates $z$ to 2 significant digits.

## Chopping off Versus Rounding off

Consider any real number $p$ in decimal form

$$p = \pm 0.d_1 d_2 d_3 ... d_k d_{k+1} ... * 10^n$$

where $1 \leq d_1 \leq 9 \qquad 0 \leq d_j \leq 9$

Suppose that $k$ is the maximum number of decimal digits carried in the computations, then

chopped floating point

$$fl_{chop}(p) = \pm 0.d_1 d_2 d_3 ... d_k * 10^n$$

rounded floating point

$$fl_{round}(p) = \pm 0.d_1 d_2 d_3 ... r_k * 10^n$$

Example:

$$p = 0.3142857142857142857... * 10^1$$

has the following six-digit representations:

$$fl_{chop}(p) = 0.314285 * 10^1$$
$$fl_{round}(p) = 0.314286 * 10^1$$

## $O(h^n)$ Order of Approximation

Definition: Assume that $f(h)$ is approximated by a function $p(h)$ and there exists a real number $M$ and a positive integer $n$ so that

$$\frac{|f(h)-p(h)|}{|h^n|} \leq M \qquad \text{for sufficiently small } h$$

We say that $p(h)$ approximates $f(h)$ with order of approximation $O(h^n)$ and write

$$f(h) = p(h) + O(h^n)$$