Lecture 6

Friday, January 17, 2020

Last time we learned the addition, multiplication, rounding and chopping of numbers in the binary system. Due to finite memory, a computer can represent a binary number by finitely many bits. Let's consider a simple example: suppose that a computer can represent a number by only 8 bits:

$$a_1\ a_2\ a_3\ a_4\ a_5\ a_6\ a_7\ a_8 \qquad (*)$$

If we are to design the most effective representation, what binary number should this bit sequence represent? Here are some of the options.

① The most natural choice is that the bit sequence $(*)$ represent the binary number whose digits are $a_1, a_2, \ldots, a_8$. For example, the sequence

$$1\ 0\ 1\ 0\ 0\ 1\ 0\ 1$$

represents the binary number $(10100101)_2$, which is equivalent to 165 in decimal system. This way of representation helps us represent any whole number between 0 and 255. However, it can't represent negative numbers nor fractional numbers.

② The sequence $(*)$ represent the number $(a_1a_2a_3a_4.a_5a_6a_7a_8)_2$ in binary system. For example, the bit sequence 10100101 represents the number

$$(1010.0101)_2$$

which is equivalent to $2^3+2^1+2^{-2}+2^{-4} = 12.3125$ in decimal system. This way of representation can represent fractional numbers and small numbers, the smallest positive number being $(0000.0001)_2 = 0.0625$. However, the largest number it can represent is only $(1111.1111)_2 = 15.9375$. Also, this representation can't represent negative numbers.

③ The sequence $(*)$ represents the binary number

$$(-1)^{a_1}(a_2a_3a_4.a_5a_6a_7a_8)_2$$

For example, the sequence 10100101 represents number

$$(-1)^1 (010.0101)_2 = -(010.0101)_2$$

which is equivalent to -2.3125 in decimal system.

What is the largest positive number can a bit sequence represent? What is the smallest positive number?

This way of representation can represent both positive and negative numbers, both whole and fractional numbers

The dynamic range of a representation system is defined as the quotient of the largest positive number to the smallest positive number. Dynamic range of a number format system indicates how wide the range of numbers it can represent.

For example, the dynamic ranges of system ①, ②, ③ are

$$d_1 = \frac{(11111111)_2}{(00000001)_2} = \frac{255}{1} = 255,$$

$$d_2 = \frac{(1111\ 1111)_2}{(0000.0001)_2} = 255,$$

$$d_3 = \frac{(111.1111)_2}{(000.0001)_2} = 127.$$

The representation system (number format) ①, ②, ③ are fixed-point number formats because the position of the binary point is fixed. Fixed point number formats can represent numbers that are equally spaced.

$$\underbrace{1}_{1}\ \underbrace{2}_{1}\ \underbrace{3}_{1}\ 4\ \ldots\ 254\ \underbrace{255}_{1}$$

$$\underbrace{(0000.0001)_2}_{2^{-4}}\ \underbrace{(0000.0010)_2}_{2^{-4}}\ (0000.0011)_2\ \ldots\ (1111.1111)_2$$

The drawback of fixed-point number formats is that their dynamic ranges are small. Consider another number format as follows.

(4) The bit sequence (*) represents the number

$$(-1)^{a_1}(1.a_2a_3a_4)_2 \times 2^{(-1)^{a_5}(a_6a_7a_8)_2}$$

For example, the sequence 1 0 1 0 1 1 0 1 represents the number

$$(-1)^1(1.010)_2 \times 2^{(-1)^1(101)_2}$$

$$= -125 \times 2^{-5}$$

$$= -0.0390625$$

The dynamic range of this format is

$$d_4 = \frac{(1.111)_2 \times 2^{(111)_2}}{(1.000)_2 \times 2^{-(111)_2}} = 15 \times 2^{11} = 30720.$$

This number format is a (binary) floating-point number format.

$$x = \sigma \cdot \bar{x}\ 2^e$$

$\sigma = \pm 1$ is the sign

$\bar{x}$ is called significand or mantissa.

e is called the exponent.

With relatively moderate size of $\bar{x}$ and e, the number x can be quite big or small. Compared to the fixed-point format, floating point format has a much bigger dynamic range Thus, it is capable of representing very big as well as very small numbers.

The decimal floating-point format is prevalent in science. In fact, most physical constants are written in such format

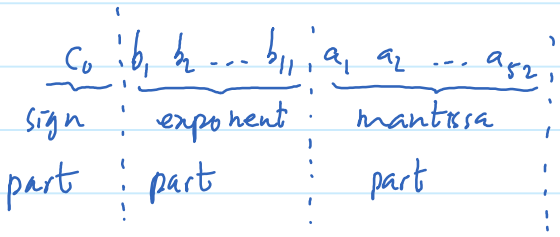
$$x = \sigma \cdot \bar{x} \cdot 10^e$$

For example,

- The Avogadro constant 6.022×10^{23},
- Gravitational constant 6.674×10^{-11},
- Planck constant 6.626×10^{-34},
- Electron mass 9.109×10^{-31}.

In 1985, the IEEE standardized a floating-points format known as IEEE 754-1985 or double precision floating-point format

This format is widely used in mathematical software today (Matlab, Maple, Mathematica, ...) Accordingly, each real number is represented by 64 bits



This bit sequence represents a number $x = \sigma \cdot \bar{x} \cdot 2^e$ where σ, $\bar{x}$, e are given as follows:

$$\sigma = \begin{cases} 1 & \text{if } c_0 = 0 \\ -1 & \text{if } c_0 = 1 \end{cases}$$ (flag up if the number is negative)

$$E = (b_1 b_2 \ldots b_{11})_2$$

If $1 \le E \le 2046$ then

$$e = E - 1023$$
$$\bar{x} = (1.a_1 a_2 \ldots a_{52})_2$$

If $E = 0$ then

$$e = -1022$$
$$\bar{x} = (0.a_1 a_2 \ldots a_{52})_2$$

If $E = 2047$ then

- If $a_1 = a_2 = \ldots = a_{52} = 0$ then
  $x = \pm\infty$ (depending on σ)
- otherwise (at least one of $a_1, a_2, \ldots, a_{52}$ is equal to 1)
  $x =$ NaN ("not a number")

[See practice problems on the worksheet.]