Floating Point Numbers

Floating point numbers are used approximate the real numbers.

Scientific notation is the basis for the floating point representation.

For instance, we can write

$$3.1415 \times 10^0 = 31.415 \times 10^{-1} = 314.15 \times 10^{-2} = 0.031415 \times 10^2$$

and float the decimal point by changing the value of the exponent.

Normalized Floating Point Numbers

A real number x, written in scientific notation is <u>normalized</u> if it has a single non-zero digit to the left of the point.

For instance,

$$3.1415 \times 10^0 \quad \text{and} \quad 6.022 \times 10^{23} \quad \text{are normalized.}$$

But,

$$314.15 \times 10^{-2} \quad \text{and} \quad 0.6022 \times 10^{24} \quad \text{are not.}$$

In binary,

$$1.1010 \times 2^{-5} \quad \text{is normalized} \quad \text{but} \quad 0.1010 \times 2^{-5} \quad \text{is not.}$$

0.1010×2^{-5} can be normalized as 1.010×2^{-6}.

Encoding a Number Written in Scientific Notation

Write a number x in normalized scientific notation as

$$x = \pm d.f \times 10^e$$

You must know

- The leading sign $+$ or $-$ of x.
- The normalizing digit d.
- The fractional part f.
- And the exponent e.

Encoding a Number Written in Binary Scientific Notation

Now, write x in normalized <u>binary</u> form.

$$x = \pm 1.f \times 2^e$$

You still must know: The sign, the fractional part f, and the exponent e.

But because x is normalized, the normalizing bit must be 1.

Floating Point Numbers

The numbers that can be written in floating point notation is limited by the size of their representation.

For instance,

There needs to be 1 bit to encode the sign.

If there are 8 bits for the exponent, then there are $2^8 = 256$ different exponents e.

If there are 23 bits for the fractional part, then there are 2^{23} different fractions f.

So there are approximately

$$2 \times 2^8 \times 2^{23} = 2^{32} \quad \text{different } 32 \text{ bit floating point numbers.}$$

I say approximately because special numbers such as zeros, infinities, and NaNs (Not a Numbers)s must be counted.

IEEE Standard for Floating-Point Arithmetic (IEEE 754)

To minimize the early chaos in approximating real arithmetic a standard was invented in the 1980's and today's floating point units implement it.

Learning the IEEE 754 standard is beyond the scope of this course.

But you will learn a pidgin version that explains some basic ideas.

Binary Floating Point Numbers (Pidgin Version)

A normalized 8-bit binary floating point number x is parsed into three parts as shown below.

Sign Bit	Exponent Bias $b = 3$	Fraction
s	$e_2 e_1 e_0$	$f_{-1} f_{-2} f_{-3} f_{-4}$

Then x can be written as

$$x = (-1)^s \pm 1.f \times 2^{e-b}$$

The Floating Point Sign

Let x be a normalized floating point number.

$$x = (s\ e_2e_1e_0\ f_{-1}f_{-2}f_{-3}f_{-4})_{fp}$$

In scientific notation,

$$x = (-1)^s \pm (1.f_{-1}f_{-2}f_{-3}f_{-4}) \times 2^{e_2e_1e_0-b}$$

If the sign $s = 0$, then x is positive.

$$x = (0\ e_2e_1e_0\ f_{-1}f_{-2}f_{-3}f_{-4})_{fp} \rightarrow x \geq 0$$

If the sign $s = 1$, then x is negative.

$$x = (1\ e_2e_1e_0\ f_{-1}f_{-2}f_{-3}f_{-4})_{fp} \rightarrow x < 0$$

The Floating Point Exponent

The exponent is 3 bits: 000 to 111.

You must to represent positive and negative exponents.

Biased notation is used, because aligning exponents can be easily implemented in hardware with biased notation.

With 3 bits, 8 numbers can be represented. Let's choose -4 to 3 using a bias $b = 4$ to shift the range 0 to 7 onto -4 to 3.

The Floating Point Fractional Part

With 4 bits to represent the fractional part, you can represent 15 numbers:

$$(1.0001)_2 = \frac{17}{16} \quad \text{to} \quad (1.1111)_2 = \frac{31}{16}$$

in increments of $1/16$.

An Example of Pidgin Floating Point Notation

Let's see how this works.

Consider the floating point number

$$x = (1\ 110\ 1101)_{fp} = -\,(1.1101)_2 \times 2^{(110)_2-4}$$

$$x = -\left(1 + \frac{13}{16}\right) \times 2^{6-4}$$

$$x = -\frac{29}{16} \times 2^2 = -\frac{29}{4}$$

An Example of Pidgin Floating Point Notation

Here is another example.

Consider the floating point number

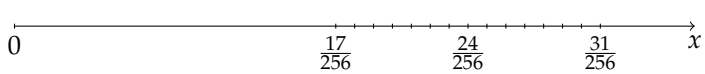
$$x = (0\ 010\ 0011)_{fp} = +\,(1.0011)_2 \times 2^{(010)_2 - 4}$$
$$x = +\left(1 + \frac{3}{16}\right) \times 2^{2-4}$$
$$x = +\frac{19}{16} \times 2^{-2} = +\frac{19}{64}$$

The Distribution of Floating Point Numbers

Consider how the floating point numbers are distributed.

The smallest positive numbers are

$$(0\ 000\ 0001)_{fp} = 17/256 \quad \text{to} \quad (0\ 000\ 1111)_{fp} = 31/256$$

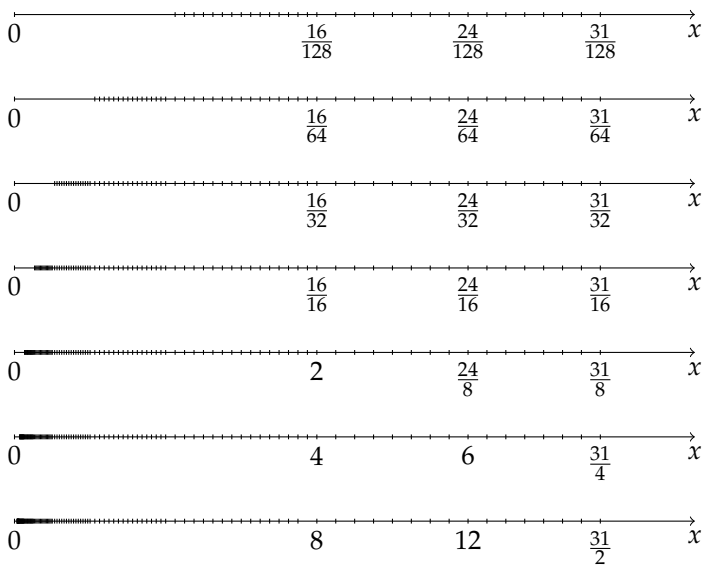


The next smallest range is from

$$(0\ 001\ 0000)_{fp} = 16/128 \quad \text{to} \quad (0\ 001\ 1111)_{fp} = 31/128$$

The Distribution of Floating Point Numbers

Let's finish out the the graphs.



Floating Point Arithmetic

The rules of arithmetic fail for floating point numbers.

For instance, the associative law fails.

$$8 + \left(\frac{1}{4} + \frac{1}{4}\right) = 8 + \frac{1}{2} = \frac{17}{2}$$

But,

$$\left(8 + \frac{1}{4}\right) + \frac{1}{4} = 8 + \frac{1}{4} = 8$$

Learning about floating point errors and how to guard against them or compensate for them is beyond the scope of this class.