

浮点数表示

一、二进制实数表示

第*i*位权重为 2^i 。

$$\text{十进制转化公式: } d = \sum_{k=-j}^{i-1} b_k \cdot 2^k$$

限制: 只能表示 $\frac{1}{2^k}$ 的数字(精确)

2. 在**w**位的情况下, 表示的范围有限.

二、IEEE 浮点数

数学形式:

$$(-1)^S \cdot M \cdot 2^E$$

S 表示正负; $M \in [1.0, 2.0)$; 指数 E

单浮点数: 32 bits 1 8 23

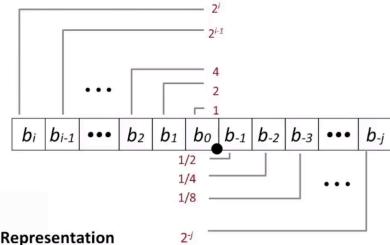
双 ~ : 64 bits 1 11 52

英特尔拓展: 80 bits 1 15 63/64

规格化表示:

1. $exp \neq 000\ldots 0$ 且 $exp \neq 111\ldots 1$

Fractional Binary Numbers



2. E 作为「被偏移值」 $E = \text{Exp} - \text{Bias}$ $\rightarrow 2^{\text{Bias}} \times (\text{K 为 exp 的系数})$

单精度: 127 ($\text{Exp} \in [1, 254]$, $E \in [-126, 127]$)

双精度: 1023 ($\text{Exp} \in [1, 2046]$, $E \in [-1022, 1023]$)

使用补码的形式可以让我们把两个数字作为无符号整数进行比较。

3. "1" 作为“起始位”，隐含的，无需存储。

T3y:

■ Value: float F = 15213.0;
■ $15213_{10} = 11101101101101_2 = 1.1101101101101_2 \times 2^{13}$

■ Significand
 $M = 1.1101101101101_2$
 $\text{frac} = 11011011011010000000000000_2$

■ Exponent
 $E = 13$
 $\text{Bias} = 127$
 $\text{exp} = 140 = 10001100_2$

■ Result:
$$\begin{array}{c|c|c|c} 0 & 10001100 & 110110110110100000000000 \\ \hline s & \text{exp} & \text{frac} \end{array}$$

 $\text{exp} = E + \text{Bias}, \text{exp} \in [0, 254].$

非规格化浮点数: ($\text{exp} = 0 \dots 0$)

$$E = 1 - \text{Bias}, M = 0.x_1x_2\dots x_n$$

* 会出现士0的情况

* 特殊值:

$$\text{exp} = 111\ldots1 \quad \text{frac} = 000\ldots0$$

• 表示 ∞ • 表示溢出 • 同时为正、负值 • $1.0/0.0 = +\infty \quad 1.0/-0.0 = -\infty$.

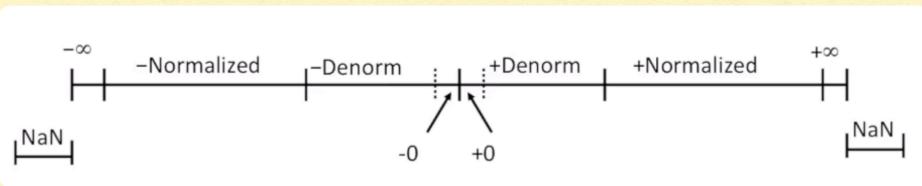
情况：

$$\text{exp} = 111\ldots1 \quad \text{frac} \neq 000\ldots0$$

非数(NaN), 表示无法推断出的数值, 如

$\sqrt{-1}$, $\infty - \infty$, $\infty \times 0$.

规范化数字：



例：8位浮点数：

$$\begin{aligned} \text{denormal: } & 0, \frac{1}{512}, \dots, \frac{7}{512} \quad \text{normalized: } \frac{8}{512} \\ E = 1 - \text{Bias} & \uparrow \quad \downarrow \quad \boxed{E = \text{Exp} - \text{Bias} \times \text{frac}} \\ \frac{1}{8} \times 2^{-6} & = \frac{1}{8} \times \frac{1}{64} \quad \frac{8}{8} \times 2^{-6} = \frac{8}{8} \times \frac{1}{64} \\ \frac{(0001)_{\text{frac}}}{2^3} & \quad \boxed{\frac{(0000)_{\text{frac}}}{2^3}} \end{aligned}$$

如果将其绘至于数轴上：



| 规格化 | 非规格化 |

IEEE 编码的特性：

- 为0时：所有比特均为0；-（几乎）可以用于无符号整数比较。

- 比较符号位
- 考虑 $-0 = 0$
- 考虑 NaN

三. 浮点数运算

思想：先算出具体结果，对结果进行合理的舍入。

舍入：

Rounding Modes (illustrate with \$ rounding)

	\$1.40	\$1.60	\$1.50	\$2.50	-\$1.50	
向零舍入	Towards zero	\$1	\$1	\$1	\$2	-\$1
向负无穷舍入	Round down ($-\infty$)	\$1	\$1	\$1	\$2	-\$2
向正无穷舍入	Round up ($+\infty$)	\$2	\$2	\$2	\$3	-\$1
向最近的偶数舍入	Nearest Even (default)	\$1	\$2	\$2	\$2	-\$2

IEEE 754 使用。

如图，为四种不同的舍入方式

为什么向最近的偶数舍入 (Round-To-Even)

根据统计学原理，一组未知数据约有50%的几率向上/向下舍入，使得没有平均值偏差/小的偏差。

应用于浮点数时：当正好处于半中央时，才对数字进行舍入至最近的偶数。

例： $7.8949999 \rightarrow 7.89$ $7.8950001 \rightarrow 7.90$ $7.8950000 \rightarrow 7.90$

$7.885000 \rightarrow 7.88$

二进制：舍入到最近的 $\frac{1}{4}$ (根据所需权重最高的一位，如下图中红色部分单靠左
右一位)。当权重最高的一位为 0 为舍入为偶数，为 100...11 时，则为半中央。

Value	Binary	Rounded	Action	Rounded Value
2 3/32	10.00011 ₂	10.00 ₂	(<1/2-down)	2
2 3/16	10.00110 ₂	10.01 ₂	(>1/2-up)	2 1/4
2 7/8	10.11100 ₂	11.00 ₂	(1/2-up)	3
2 5/8	10.10100 ₂	10.10 ₂	(1/2-down)	2 1/2

乘法。

$$\text{Result: } (-1)^S M \cdot 2^E \quad S \rightarrow s_1 s_2 \quad M \rightarrow M_1 \times M_2 \quad E \rightarrow E_1 + E_2$$

浮点数加法的数字特征：

相加于阶只对齐，将尾数加法可交换，但不可结合。

$$T_{\text{例}}: (3.14 + 1e10) - 1e10 = 0, 3.14 + (1e10 - 1e10) = 3.14.$$

乘法：同理

单调性：除 ∞ 、 NaN 外都可。

四、C 语言中的浮点数

`double float` \rightarrow `int`

舍弃 `frac`、类似于舍入至 0、`NaN/00`；因为 TMin

`int → double`

完整地转成，因为 `int` 只有 ≤ 53 bits

`int → float`

会进行舍入，因为 `int` 的 bit > `float` 的。