Лекция №3
по курсу
«Машинная арифметика в рациональных чисел»

Москва, 2020

Представление данных

Вещественные числа

Стандарт IEEE 754 (1985 г.).
Профессор Университета Беркли Вильям Каган.

Одинарная точность

знак	порядок		мантисса	
31	30	23	22	0

Порядок 8 бит со сдвигом 127, мантисса не 23, а 24 бита, старшая 1 подразумевается.

Двойная точность

знак	порядок		мантисса	
63	62	52	51	0

Порядок 11 бит со сдвигом 1023.

Специальные числа

- **NaN** (not a number)
  Порядок — все единицы
  Мантисса — не нулевая
- **Бесконечность**
  Порядок — все единицы
  Мантисса — 0
- **Нуль**
  Порядок — 0
  Мантисса — 0

Округление чисел с плавающей точкой

Первый случай – находится за пределами диапазона представления нормализованных чисел с плавающей точкой.

Второй случай – требует более p бит для представления числа с плавающей запятой.

Определим x-, как число с плавающей точкой, ближайшее к x, которое меньше или равен x, и определим x+, как число с плавающей точкой, ближайшее к x, которое больше чем или равно x.

Пусть имеет место второй случай и x>0, т.е.

$$x = (1.b_1 b_2 \ldots b_{p-2} b_{p-1} b_p b_{p+1} \ldots)_2 \cdot 2^E, \qquad (2.4.0)$$

Ближайшее меньшее определяется следующим образом:

$$x- = (1.b_1 b_2 \ldots b_{p-2} b_{p-1})_2 \cdot 2^E, \qquad (2.4.1)$$

Округление чисел с плавающей точкой

ближайшее большее:

$$x+ = ((1.b_1b_2...b_{p-2}b_{p-1})_2 + (0.00...01)_2)\cdot 2^E, \qquad (2.4.2)$$

Если $x > N\max$, то $x- = N\max$, $x+ = \infty$.

Рассмотрим округление в соответствии со стандартом IEEE 754.

Если x число с плавающей точкой, то результат округления

$round\ (x) = x$

В противном случае результат округления зависит от одного из четырех режимов округления:

1.Округление вниз (иногда называется округлением в сторону $-\infty$)

$round\ (x) = x-$

2.Округление вверх (иногда называется округлением в сторону $+\infty$)

$round\ (x) = x+$

Округление чисел с плавающей точкой

3.Округление в сторону нуля

$round\ (x) = x -$, если $x > 0$ и $round\ (x) = x +$, если $x < 0$

4. Округление к ближайшему

$round\ (x)$ ближайшее или $x -$ или $x +$.

Режим округления, который почти всегда используется на практике, округляется до ближайшего. Стандарт IEEE требует, чтобы округление по умолчанию до ближайшего.

Другие режимы округления являются необязательными и находят применение в интервальной арифметике.

Существует исключение из правила округления до ближайшего, когда x> Nmax. В этом случае, round (x) определяется как Nmax, если x <Nmax + ulp (- / Nmax) / 2 и ∞ в противном случае.

ОШИБКИ ПРИ КОМПЬЮТЕРНЫХ ВЫЧИСЛЕНИЯХ

$$2^{-(p-1)} \cdot 2^{E} \qquad (2.5.1)$$

и

$$abserr\ (x) = |round\ (x) - x| < 2^{-(p-1)} \cdot 2^{E}$$

При округлении к ближайшему, абсолютная погрешность округления половина разности между x- и x+

$$abserr\ (x) = |round\ (x) - x| < 2^{-p} \cdot 2^{E}$$

Формула для относительной погрешности имеет следующий вид:

$$\delta = \frac{|round\ (x) - x|}{x} \qquad (2.5.2)$$

Пусть x имеет вид (2.4.0) и не является число с плавающей точкой, тогда справедливо, что:

$$|x| > 2^{E}$$

ОШИБКИ ПРИ КОМПЬЮТЕРНЫХ ВЫЧИСЛЕНИЯХ

Для всех режимов округления относительная погрешность округления удовлетворяет неравенству:

$$rel = |\delta| = \frac{|round\ (x) - x|}{|x|} < \frac{2^{-(p-1)} \cdot 2^{E}}{2^{E}} = 2^{-(p-1)} = \varepsilon$$

Для округления к ближайшему справедливо неравенство:

$$rel = |\delta| = \frac{|round\ (x) - x|}{|x|} < \frac{2^{-p} \cdot 2^{E}}{2^{E}} = 2^{-p} = \frac{1}{2}\varepsilon$$

Тогда

$$-\log_2 rel > p - 1$$

и для округления к ближайшему:

$$-\log_2 rel > p$$

или

$$-\log_{10} rel > -\log_{10} \varepsilon$$

$$\log_{10} rel < \log_{10} \varepsilon$$

ОШИБКИ ПРИ КОМПЬЮТЕРНЫХ ВЫЧИСЛЕНИЯХ

Из этого неравенства следует, что $-\log_2 rel$ измеряет количество бит в которых round(x) и x совпадают. Это соответствуют, по крайней мере, примерно 7 цифрам, для одинарной точности и до 16 цифр в случае двойной точности.

Из (2.5.2) следует, что:

$$round\ (x) = x(1+\delta) \qquad (2.5.3)$$

Таким образом, справедливо утверждение:

Пусть x - любое вещественное число в двоичной системе плавающей точкой с точностью p Тогда

$$round\ (x) = x(1+\delta)$$

для некоторого

$$|\delta| < 2^{-(p-1)} = \varepsilon$$

ε – машинное эпсилон (разница между 1 и ближайшим большим числом с плавающей точкой)

ОШИБКИ ПРИ КОМПЬЮТЕРНЫХ ВЫЧИСЛЕНИЯХ

Из этого неравенства следует, что $-\log_2 rel$ измеряет количество бит в которых round(x) и x совпадают. Это соответствуют, по крайней мере, примерно 7 цифрам, для одинарной точности и до 16 цифр в случае двойной точности.

Из (2.5.2) следует, что:

$$round\ (x) = x(1+\delta) \qquad (2.5.3)$$

Таким образом, справедливо утверждение:

Пусть x - любое вещественное число в двоичной системе плавающей точкой с точностью p Тогда

$$round\ (x) = x(1+\delta)$$

для некоторого

$$|\delta| < 2^{-(p-1)} = \varepsilon$$

ε – машинное эпсилон (разница между 1 и ближайшим большим числом с плавающей точкой)

Арифметические операции с плавающей точкой

В соответствии с форматом с плавающей точкой:

$x \oplus y = \text{round}\,(x+y)$,

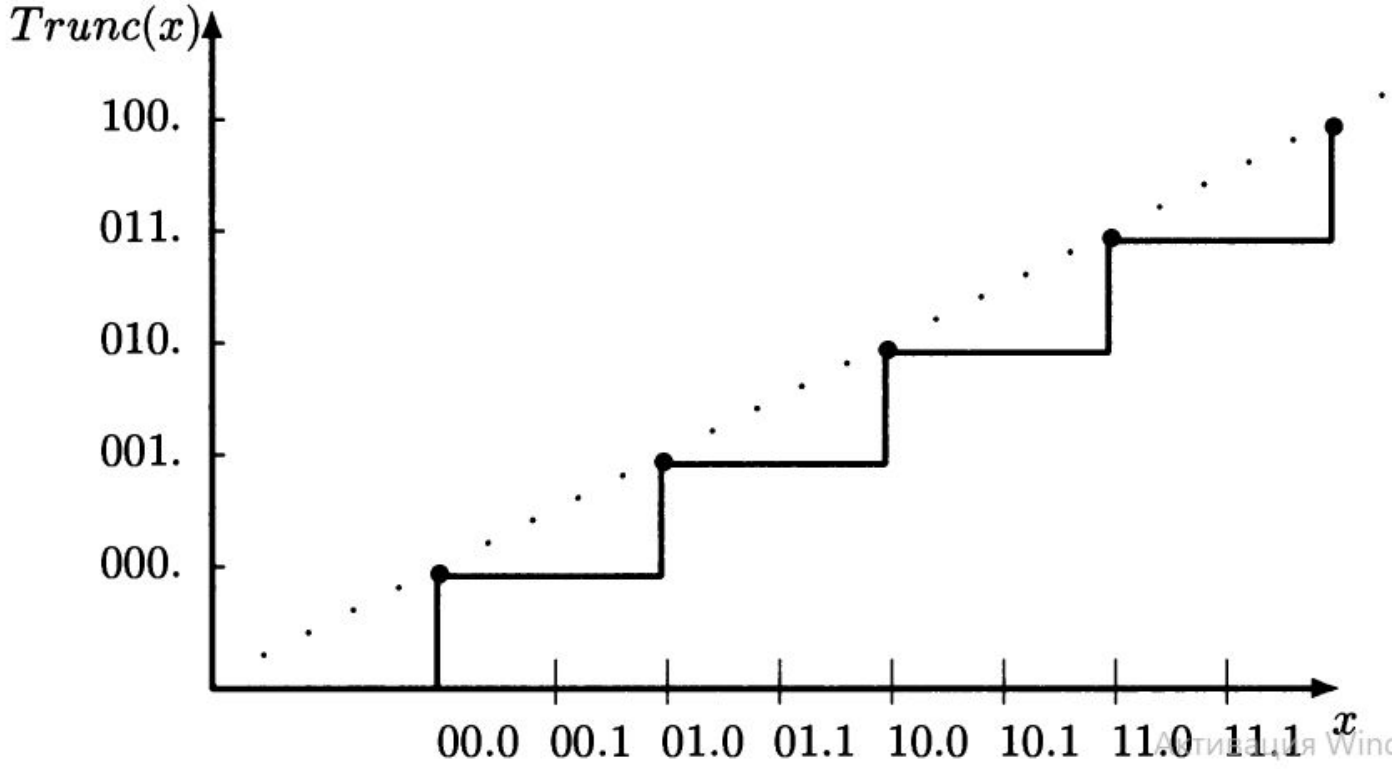
$x \ominus y = \text{round}\,(x-y)$,

$x \otimes y = \text{round}\,(x*y)$,

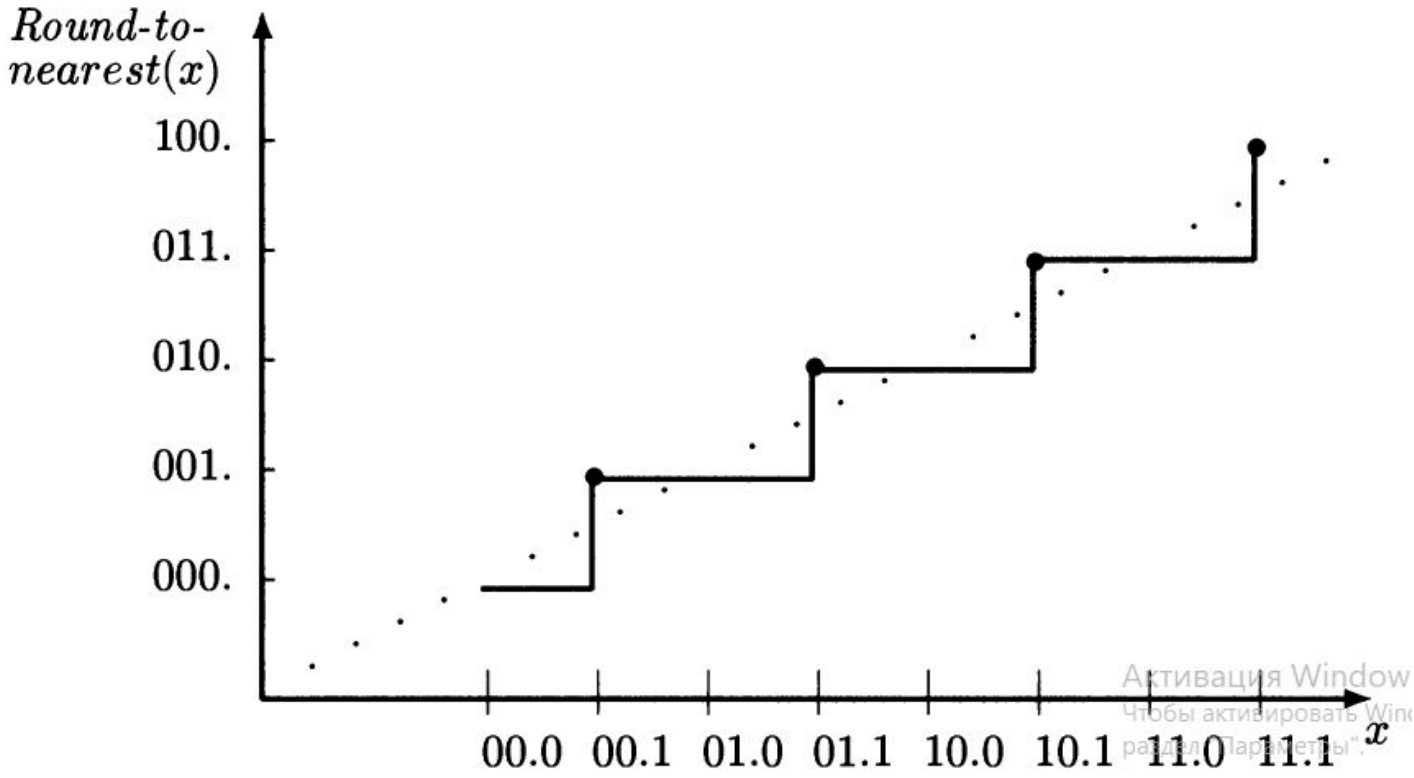
$x \oslash y = \text{round}\,(x/y)$

$x \oplus y = (x+y)(1+\sigma)$,
$|\sigma| < \epsilon$,
где
ϵ – машинное эпсилон.
ulp – “resolution of floating point number”

Ошибка усечения



Округление к ближайшему



Защитные биты

Для сохранения промежуточных результатов

$2.95 \cdot 10^2$
$2.39 \cdot 10^0$

Округление не должно производиться до нормализации результата.

В общем, необходимо использовать только три дополнительных бита, чтобы реализовать правильно округленное сложение и вычитание с плавающей запятой: два защитных бита и одним битом.

Умножение и деление

Пусть даны два числа в формате с плавающей точкой одинарной точности $x = S \cdot 2^E$, $y = T \cdot 2^F$, то в результате их умножения получим:

$$x \cdot y = S \cdot T \cdot 2^{E+F},$$

В 1994 году была обнаружена аппаратная ошибка с плавающей запятой в Pentium.

4195835

_____

3145727

возвращала результат с точностью 4 знака

Некоторые другие операции

- Преобразование от одинарного до двойного
- Преобразование от двойного к одинарному
- Преобразование между форматами с плавающей запятой и целыми числами
- Преобразование двоичного числа в десятичное и десятичного в двоичное

Упражнения

1. Найти абсолютную погрешность для четырех режимов округления

2. Привести пример, когда результатом сложения двух чисел с плавающей точкой не является число с плавающей точкой.

Бесконечность при делении на ноль

До стандарта IEEE было разработано два стандартных ответа на деление положительного числа на нуль.

- Генерация наибольшего числа

  1/0 – 1/0 - ?

- прерывание или прекращение работы программы

$$R = \frac{1}{\frac{1}{R_1} + \frac{1}{R_2}}$$

Бесконечность при делении на ноль

Если , то R1 = R2
Если R1<<R2, то и ток в большей степени проходит через первый резистор, в малой через второй.
Если R1=0, то интуитивно ясно, что первый резистор имеет нулевое сопротивление и весь ток проходит через него и общее сопротивление схемы . Т.е. R = 0

$$R = \frac{1}{\frac{1}{0} + \frac{1}{R_2}} = \frac{1}{\infty + \frac{1}{R_2}} = \frac{1}{\infty} = 0$$

Стандартный ответ деление на ноль - это получение бесконечного результата и продолжения выполнения программы.

Специальные числа

$0 \cdot \infty$

$0 / 0$

Не имеют математического смысла. Попытка вычислить их является недопустимой операцией, и стандартный ответ IEEE для такой операция состоит в том, чтобы установить результат в NaN.

$\infty - \infty$

Специальные числа

Проверить на компьютере с использованием современного языка программирования значения выражений $\infty / 0$, $0 / \infty$ и ∞ / ∞ ? Обосновать ответы.

Ноль со знаком и бесконечность со знаком

Возникает вопрос: почему 1/0 должно иметь значение ∞ , а не - ∞ ? Это одна из причин для существования числа с плавающей точкой -0, так что соглашения x/0 и x/-0 могут выполняться, если x - положительное число.

Предикат 0 = -0 верен, но предикат ∞ = - ∞ ложно

Предикаты

x = y

1/x = 1/y

имеют различные значения если x = 0, y =-0

Упражнения

Упражнение

Есть ли другие случаи, когда предикаты $a = b$ и $\frac{1}{a} = \frac{1}{b}$ имеют противоположные значения, кроме того, что a, b являются нулями противоположного знака?

Упражнение :

Каковы значения выражений $0 / (-0)$, $\infty / -\infty$ и $-\infty / -0$?

Упражнение

Каков результат для формулы параллельного сопротивления, если $R_1 = 1$, а $R_2 = -0$?

Переполнение и исчезновение порядка

В IEEE стандарте, ответом на переполнение является предоставление правильно округленного результата, либо ± Nmax или ±∞. Диапазон чисел, округляемых до ±∞, зависит от режима округления.

В арифметике IEEE стандартным ответом на недостаточное значение является возвращение правильно округленного значения, которое может быть ненормализованным числом с плавающей точкой, ± 0 или ± - Nmin. Это известно как постепенная потеря значимости

$$
\begin{aligned}
& (\ 1.01000000000000000000000\ )_2 \times 2^{-126} \\
- \quad & (\ 1.00000000000000000000000\ )_2 \times 2^{-126} \\
= \quad & (\ 0.01000000000000000000000\ )_2 \times 2^{-126} \\
\text{Нормализация:} \quad & (\ 1.00000000000000000000000\ )_2 \times 2^{-128}
\end{aligned}
$$

Переполнение и исчезновение порядка

Последняя строка показывает нормализованное представление, меньшее, чем Nmin.

Без постепенного потери значимости результат следует округлить до нуля. При постепенном значимости, ответ 2^{-128} может храниться точно, с субнормальным представлением:

0	00000000	01000000000000000000000

Стандартный ответ на исключения

Таблица Стандартный ответ на исключения

Неверная операция	Установка результата NaN
Деление на ноль	Установка результата ±∞
Переполнение	Установка результата ±∞ или ±Nmax
Исчезновение порядка	Установка результата ±0 или ±Nmin или субнормальное число
Неточный результат	Корректное округление

Стандартный ответ на исключения

Программист должен иметь возможность либо перехватывать исключение, предоставляя специальный код выполняться, когда возникает исключение, или маскировать исключение, и в этом случае программа продолжает выполнение со стандартным ответом, показанным в таблице. Появление NaN на выходе программы является верным признаком того, что пошло не так. Появление ∞ на выходе может указывать или не указывать на ошибку программирования в зависимости от контекста. При написании программ, где деление на нуль возможно, программист должен быть осторожен. Операции с ∞ не должны использоваться без анализа результатов вычислений.

Библиотечные математические функции

fabs
sqrt
exp
log
log10
sin
cos
atan
pow

Log(-0), Log(0), Log(+1), Log(-1)

Log(+∞), Log(-∞)

Include <math.h>