Números em ponto flutuante

INF01107 – Introdução à Arquitetura de Computadores

Representação de números grandes

- Quantos bits são necessários para representar 1 quintilhão?

$$1 \text{ quintilhão} = 1{,}0 \times 10^{18} =$$

$$1\,000\,000\,000\,000\,000\,000$$

- Forma alternativa para representar números muito grandes e/ou números muito pequenos $\rightarrow$ notação científica

  Representação de números em ponto flutuante = versão binária da notação científica

Números em ponto flutuante

- mantissa ‘m’
- expoente ‘e’
- base ‘b’
- Nos computadores atuais: a base = binária (b=2)

$$N = m \times b^e$$

- base = constante para um determinado sistema
- representação em ponto flutuante: (m,e)
- ‘m’: fração ou inteiro, positivo ou negativo
- ‘e’: inteiro, positivo ou negativo

Representações variadas

- Precisão de um número em ponto flutuante: determinada pelo número de bits utilizados pela mantissa.
- Faixa de representação R: depende do número de bits do expoente.

- Um dado número pode ser representado de mais de uma maneira:

  Ex: 1 quintilhão =

  $$1{,}0.10^{18} = 0{,}1.10^{19} = 100.10^{16}$$

  $\rightarrow$ forma normalizada

Mantissa normalizada

- Apenas parte fracionária (não existe parte inteira)
- Primeiro dígito à direita da vírgula $\neq 0$

  Ex: forma normalizada para um quintilhão = $0{,}1.10^{19}$

- Base binária: mantissa normalizada tem seus dois bits mais significativos diferentes.
- Números positivos: mantissa inicia sempre por $0{,}1_2$.

- Números em 2's: bit mais significativo da mantissa $\neq$ bit de sinal. Logo:

  Números positivos: a mantissa inicia por $0{,}1_2$

  Números negativos: a mantissa inicia por $1{,}0_2$

  Assim: $1/2 \leq |M| \leq 1$

Estouro

- Estouro na capacidade de representação dos números em ponto flutuante: estouro de representação no expoente
- Estouro da mantissa pode ser recuperado, renormalizando a mantissa e somando um ao expoente
- Se o expoente ultrapassar o maior expoente positivo: “overflow”
- Se o expoente ultrapassar o menor negativo: “underflow” (resultado = zero)

Representação do número zero

- mantissa deve ser igual a zero
- expoente pode apresentar qualquer valor ($0.b^e = 0$)

Condições para representação:

- A mantissa do número zero tem todos os bits em 0, para facilitar o teste por zero (isto contraria a definição formal de mantissa normalizada)

- O número zero é a menor quantidade absoluta possível de representação (expoente escolhido para o 0 deve ser o menor número negativo representável). Para expoente de ‘k’ bits, representado em complemento de dois, expoente para o 0 = -2^{k-1}

Representação do número zero

- Assim, o zero é representado por $0.b^{-2^{k-1}}$. Para facilitar o teste de zero, codifica-se o expoente "em excesso de 2^{k-1}", ou seja, um expoente em zero significa na realidade a maior magnitude negativa representável.

- Codificação em excesso: indica que existe uma quantidade a mais somada ao expoente

  Ex: Para um expoente de 8 bits em complemento de dois: codificação em "excesso de 128":

  expoente –128 é representado por 0

  o expoente 0 é representado por 128

  o expoente 127 (o maior possível) é dado por 255

Formatos de números em PF: série S/360-370 da IBM

- O número utiliza 32 bits: 1 bit para o sinal da mantissa, 7 bits para o expoente e 24 bits para o valor da mantissa
- a mantissa é representada em sinal magnitude; o expoente é representado em excesso-de-64
- o bit mais significativo representa o sinal da mantissa; os sete bits seguintes representam o expoente; e os 24 bits menos significativos representam o valor da mantissa
- a base utilizada é a base 16 (hexadecimal)

Formatos de números em PF: série S/360-370 da IBM

- Representação de: $0{,}125 \times 16^{5}$:
  0 1000101 00100000000000000000000
- Representação de: $-0{,}125 \times 16^{-5}$:
  1 0111011 00100000000000000000000

Obs.: como a base é 16, a mantissa é normalizada para dígitos hexadecimais (não para dígitos binários)

Formato recomendado pelo IEEE

(Inst. of Electrical and Electronic Engineers)

Existem 3 formatos:

simples - de 32 bits

duplo - de 64 bits e

quádruplo - de 128 bits (não usado)

Bit de sinal (S)	**Expoente (E)**	**Mantissa (F)**

Formato recomendado pelo IEEE (Inst. of Electrical and Electronic Engineers)

	Simples	**Duplo**
Campos:		
S = sinal	**1 bit**	**1 bit**
E = expoente	**8 bits**	**11 bits**
L = primeiro bit da mantissa	**(não representado)**	**(não representado)**
F = fração	**23 bits**	**52 bits**
Expoente		
Excesso-de	**127**	**1023**
Maior valor	**255**	**2047**
Menor valor	**0**	**0**

Formato recomendado pelo IEEE

Podem ser representados 5 grupos de números:

- **Números normalizados:**
  expoente vai de 1 a 254 (ou 1 a 2046, ou 1 a 32766)
  primeiro bit da mantissa (L) é sempre zero (não é representado)
  L.F representa a mantissa (L = primeiro bit, F = fração)
  Cálculo do valor do número: $(-1)^{s} \cdot 2^{\text{E-excesso}} \cdot (L.F)$

- **Zero:**
  representado por um número todo em zero: E=F=L=0
  Obs.: o zero pode ter sinal

Formato recomendado pelo IEEE

Podem ser representados 5 grupos de números:

- **Números não normalizados:** possuem o expoente em zero (E=0) e uma fração não zero.

  Objetivo: representação do números que causam "underflow", se normalizados

- **Infinito:** representado pelo maior valor do expoente (E=255 ou 2047 ou 32767) e por uma fração em zero (F=L=0)

  Obs.: o infinito pode ter sinal

- **Não-números (Not a Number):** representados pelo maior expoente e por uma fração diferente de zero.

  Objetivo: indicação de códigos de erro, por exemplo

Soma e subtração de números em ponto flutuante

- Números devem apresentar o mesmo expoente.
- Soma (subtração) é realizada sobre as mantissas.
- Resultado: a mantissa resultante e o expoente comum aos operandos.

- Expoentes diferentes devem ser igualados:

  - menor dos expoentes deve ser tornado igual ao maior

  - a mantissa correspondente a este expoente deve ser deslocada para a direita

Soma e subtração de números em ponto flutuante

- X e Y são números em ponto flutuante: X_m e Y_m são as mantissas e X_e e Y_e são os expoentes

  Se $X_e = Y_e$, então $X \pm Y = (X_m \pm Y_m) \cdot 2^{Xe}$
  Se $X_e < Y_e$, então $X \pm Y = (X_m . 2^{(Xe-Ye)} \pm Y_m) \cdot 2^{Ye}$
  Se $X_e > Y_e$, então $X \pm Y = (X_m \pm Y_m . 2^{(Ye-Xe)}) \cdot 2^{Xe}$

- Após a operação, a mantissa do resultado deve ser normalizada, através de deslocamentos para a esquerda (direita) e decrementos (incrementos) do expoente
- Pode ocorrer estouro de representação

Multiplicação de números em ponto flutuante

- Basta multiplicar as mantissas e somar os expoentes:

$$X \times Y = (X_m \times Y_m) \cdot 2^{(Xe+Ye)}$$

- Após a multiplicação, o resultado deve ser normalizado. Pode ocorrer estouro de representação do expoente

Divisão de números em ponto flutuante

- A mantissa do dividendo é expandida para o dobro do seu nº de bits (0s à direita)
- As duas mantissas são divididas, usando-se o mesmo algoritmo utilizado para números inteiros
- Divide-se as mantissas e subtraem-se os expoentes

  Obs.: o resto é normalmente desprezado

$$X \div Y = (X_m \div Y_m) \cdot 2^{(Xe - Ye)}$$

- Após a divisão, o resultado deve ser normalizado

  - Pode ocorrer estouro de representação do expoente