2020

Números de Punto Flotante

Introducción a la Informática

JUAN DIEGO VÉLEZ SANCHEZ

DICIEMBRE DE 2020

UTP | Pereira

1 CONTENIDO

1 CONTENIDO 1
2. PRESENTACIÓN 2
3. PUNTO FIJO Y FLOTANTE 3
4. SUMA Y RESTA 4
5. DESBORDAMIENTO 6
6. RANGOS NUMÉRICOS 7
7. ESTRUCTURA Y PRESICIÓN FLOTANTE 8
8. REAL A BINARIO CON PUNTO FLOTANTE 11
9. CONCLUSIÓN 12
10. BIBLIOGRAFÍA 13

2. PRESENTACIÓN

La presente monografía describe la teoría e implementación de los números con punto flotante, que nos dan soporte al entendimiento de una notación que está presente en la computación actual.

En los siguientes párrafos se presenta una descripción básica del significado de lo que es un número con punto flotante.

Los números con punto flotante es una notación que se usa para representar números reales, esto pueden ser demasiados largos, tanto números con un gran valor como 93 billones, o aquellos cuyo valor es más pequeño 0.001245…., esta notación es una forma eficiente y además se puede realizar operaciones aritméticas.

Hay que tener en cuenta que el sistema numérico utilizado en computación es el sistema binario cuya base es 2.

3. PUNTO FIJO Y FLOTANTE

Los números con punto fijo, son aquellos que tienen un número fijo de dígitos, ejemplo 345, 230, -10 o 123 con una escala del 1 / 100 == 1,23 donde su notación costa de tres partes:

1. Signo
2. Parte entera
3. Parte decimal

Las operaciones toman menos tiempo pero sus valores son restringidos a valores enteros.

Otra forma de representación es la del punto flotante, donde se representa con tres elementos que son:

1. El cociente o mantisa
2. La base
3. El exponente

El cociente o mantisa está formado por un número que puede ser negativo o positivo, seguido de un punto o coma con varios dígitos que representan el número real ejemplo 3,14

La base equivale al sistema numérico en el que está escrito dicho número, ejemplo la base puede ser 10 este es el sistema decimal, o base 2 que es el sistema binario.

El exponente es un número entero positivo o negativo que representa el desplazamiento de la coma tantas veces como indique el exponente.

Ejemplos punto flotante:

2.71×10^{-2} Representación de 0.0271

Si notamos el punto se ha desplazado dos posiciones hacia la izquierda ya que el exponente es negativo, su mantisa en el número 2.71, base 10 y exponente -2.

2.71×10^{2} Representación del 271

Como el exponente es positivo el punto se desplaza hacia la derecha, como el exponente lo indica se corre dos posiciones lo cual queda 271

2.715×10^{2} Representación del 271.5

Las operaciones toman más tiempo y los resultados incluyen una parte entera.

4. SUMA Y RESTA

x+y ==> x+y = $2.56\ 10^{3} + 5,16\ 10^{5}$

$(2.56\ 10^{3-5} + 5,16)\ 10^{5}$

$(2,56\ 10^{-2} + 5,16)\ 10^{5}$

$(0,0256 + 5,16)\ 10^{5}$

$(5,1825)\ 10^{5}$

Como podemos ver en el anterior proceso donde se suman números en base decimal, lo primero que se debe hacer es igualar los exponentes ya que las bases son iguales, esto se logra restando el exponente menor con el mayor, cuando hallamos realizado este proceso sumamos los números que nos han quedado, hay que recalcar que el número que se realizó la resta de exponentes se debe resolver antes de la suma, mientras que el otro queda normal, al final se suman los dos números y el resultado queda con la misma base y el exponente mayor.

x-y ==> x-y = $2.56\ 10^{3} - 5,16\ 10^{5}$

$(2.56\ 10^{3-5} - 5,16)\ 10^{5}$

$(2,56\ 10^{-2} - 5,16)\ 10^{5}$

$(0,0256 - 5,16)\ 10^{5}$

-5.1344 * 10^{5}

En la resta es el mismo proceso, igualar exponentes y resolver al final que con la misma base y exponente mayor.

Pero como sabemos las computadoras funcionan con números binarios por lo cual la suma quedaría de la siguiente forma:

Para la duma tenemos cuatro reglas:

0 + 0 = 0

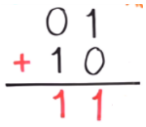
1 + 0 = 1

0 + 1 = 1

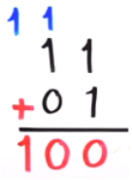
1 + 1 = 10

Estas son las reglas para efectuar una suma, en la cuarta regla el resultado no es diez (10), sino que 1 en binario y 0 en binario.

A = 01 B = 10 Resultado = 11



A = 11 B = 01 Resultado = 100



Como 1 + 1 = 10, se pone cero en la suma y se lleva 1 para la siguiente cifra, como tenemos 1 + 1 nos da nuevamente 10, lo cual ponemos cero y llevamos el 1 a la siguiente cifra, como el 1 no está sumando nada es lo mismo tener 1 + 0, lo cual nos da 1.

```
  010,010
 +100,011
 ---------
  110,101
```

Sin embargo estamos representando números en una forma científica lo cual la representación del resultado en la suma puede ser:

$$110.101 = 0.110101 * 2^{11} = 1.10101 * 2^{10}$$

Este proceso es la normalización de un número en base 2, consiste en desplazar el punto decimal, hasta que el primer digito sea UNO (1).

5. DESBORDAMIENTO

Un desbordamiento es cuando una operación intenta obtener o crear un valor numérico que esta fuera de su rango computacional, hay diferentes tipos de desbordamiento:

1. **Desbordamiento del exponente:** Es cuando un número positivo o negativo cuyo exponente supera su capacidad, cuyo valor es el máximo o mínimo permitido, cuando ocurre esto la computadora lo representa como infinito o menos infinito, $+\infty$ o $-\infty$

2. **Desbordamiento de mantisa:** Es cuando dos mantisas producen un arrastre del bit, este se puede solucionar desplazando a la derecha y ajustando el exponente. Llamado normalización:

   0.000111 Al ser normalizado 1.11×2^{-4}

3. **Subdesbordamiento del exponente:** Es cuando un número es demasiado pequeño pero no nulo, cuando su exponente es negativo y excede el máximo permitido por lo cual se puede considerar como igual a cero (0).

4. **Subdesbordamiento de mantisa:** Es cuando los dígitos se desplazan, y lo que sucede es que se pierden gran cantidad de estos, por lo cual al final se redondea el resultado.

6. RANGOS NUMÉRICOS

Existen diferentes rangos numéricos o mejor dicho capacidad para guardar un número en el sector de la computación.

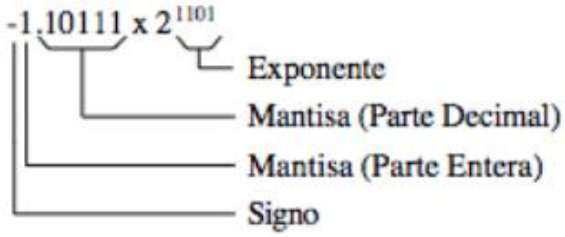
Tipo	Representación / Valor	Tamaño (en bits)	Valor mínimo	Valor máximo	Valor por defecto
`boolean`	`true` o `false`	1	N.A.	N.A.	`false`
`char`	Carácter Unicode	16	\u0000	\uFFFF	\u0000
`byte`	Entero con signo	8	-128	128	0
`short`	Entero con signo	16	-32768	32767	0
`int`	Entero con signo	32	-2147483648	2147483647	0
`long`	Entero con signo	64	-9223372036854775808	9223372036854775807	0
`float`	Coma flotante de precisión simple Norma IEEE 754	32	±3.40282347E+38	±1.40239846E-45	0.0
`double`	Coma flotante de precisión doble Norma IEEE 754	64	±1.79769313486231570E+308	±4.94065645841246544E-324	0.0

Estos son los tipos de datos numéricos, que usualmente encontramos en los lenguajes de programación, el primero en un booleano, el segundo es de caracteres como la “A”, “C” solo puede representar un elemento, y el resto son de tipo numérico, que están divididos en numéricos enteros y numéricos con punto flotante. Cada uno consumo una cantidad de bits, y tienen un valor por defecto.

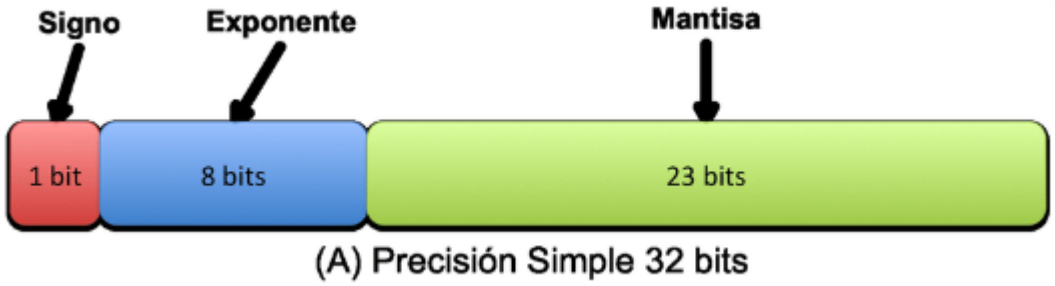
7. ESTRUCTURA Y PRESICIÓN FLOTANTE

Existen diferentes rangos numéricos o mejor dicho capacidad para guardar un número en el sector de la computación

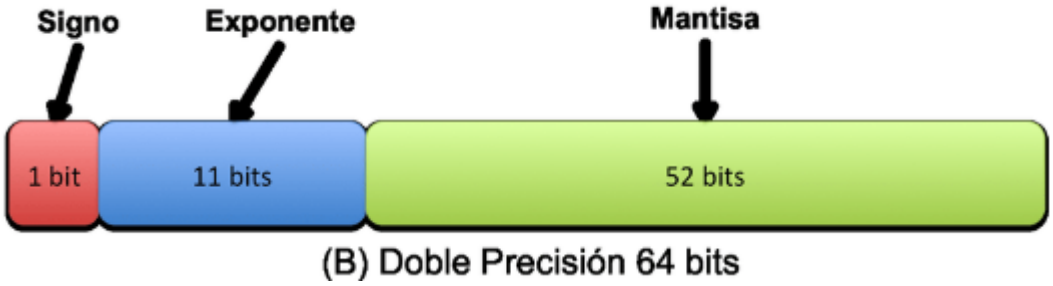
signo	exponente con signo	Mantisa
1	8	23



Podemos ver que un sistema de 32 bits, las diferentes partes de un número con puntos flotante se agrupan en paquetes o se dividen la cantidad de bits para guardar el respectivo número, o sea cada parte del número de punto flotante tiene una capacidad de bits en el cual puede ser almacenado, los signos consume un bit, la mantisa de parte entera y decimal tiene una capacidad de 23 bits y el exponente de 8 bits. Así se agrupa un número binario en un sistema de 32 bits.



(A) Precisión Simple 32 bits

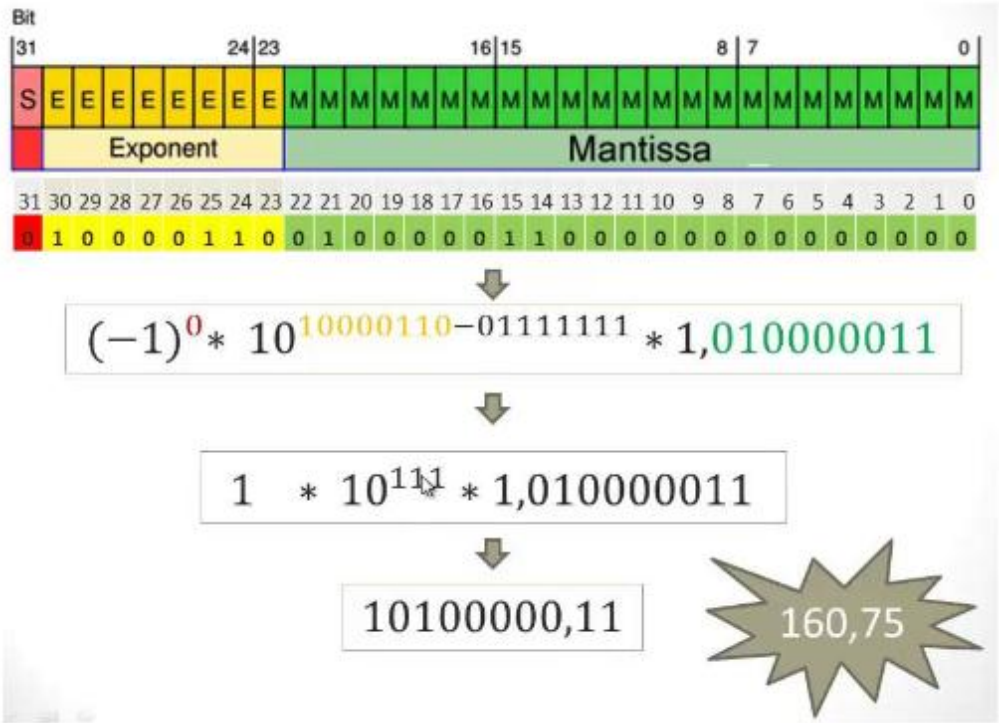


(B) Doble Precisión 64 bits

Mientras que en sistemas de 64 bits, tenemos más capacidad de cómputo, por ende podemos asignar más bits para almacenar estos datos numéricos.

Así es como se representa el número -248,75 en binario y como se almacena en un sistema de 32 bits.

Si notamos el signo está representado por un cero o uno, donde el 1 es para valores negativos y el cero (0) para positivos, seguido del exponente que nos dice cuántas posiciones debemos desplazar la coma, y de la mantisa tanto entera como decimal.



En los dos ejemplos vemos que la mantisa comienza con en cero, pero es así para ahorrarse un bit de más y tener más capacidad, además porque el número esta normalizado, se ignora el primer símbolo de la mantisa que es uno (1), sin embargo no se puede olvidar de este ya que es esencial para el resultado.

Toca recalcular que los bits son agrupados de derecha a izquierda y contados desde la posición 0 hasta la 31, esto quiere decir que el signo de la mantisa está representado en el bit 31, el exponente desde el 23 al 30, y la mantisa comienza desde el bit 0 hasta el 22, considerando que el valor (1) está oculto.

Estos tamaños de exponente y mantisa permiten tener fracciones tan pequeñas como:

$2.0_{diez} \times 10^{-38}$

Y tan grandes como:

$2.0_{diez} \times 10^{38}$.

Mientras que en la precisión doble (64 bits) permite almacenar números tan pequeños como:

$2.0_{diez} \times 10^{-308}$

Y tan grandes como:

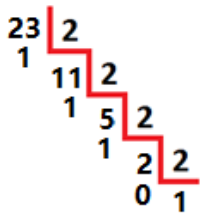
$2.0_{diez} \times 10^{308}$

8. REAL A BINARIO CON PUNTO FLOTANTE

Para representar un número real en el sistema numérico binario, debemos separarlo en dos partes, su parte entera y decimal.

La parte entera se calcula normalmente, por medio de la división sucesiva.

Pero en la parte decimal, se debe poner la parte entera de la multiplicación entre 2, por ejemplo el número 23.85:



Respuesta: 10111

.85 x 2 = 1.70
.70 x 2 = 1.40
.40 x 2 = 0.80
.80 x 2 = 1.60
.60 x 2 = 1.20
.20 x 2 = 0.40
.40 x 2 = 0.80

Luego 0.85 = 0.1101100 ....

Por tanto 23.85 = 10111.1101100....

La parte decimal se sigue multiplicando por 2 hasta que el valor sea precisamente 0 o 1, sin embargo en este caso tenemos valores repetidos ya que si vemos anteriormente ya nos había dado 0.40 y 0.80

9. CONCLUSIÓN

Los números con punto flotante, es una forma científica de escribir un número tan grande o pequeño como sea, este tiene una cantidad finita de bits que puede utilizar para ser almacenado, constando de partes esenciales como exponente, base, mantisa y signo. Hay dos formas de precisión, la precisión simple y precisión doble, 32 bits o 64 bits.

$$(-1)^{signo} \times 1.mantisa \times 2^{exponente - desplazamiento}$$

En precisión simple el exponente puede ir desde + - 127, mientras que en la doble + - 256,

1×2^{-127} Es considerado como el 0.0

10^{-39} Es el número más pequeño que podemos tener con precisión simple.

10^{38} Es el número más grande que podemos obtener con precisión simple.

Conversiones:

1	10000001	01000000000....
signo	exponente	mantisa

$$(-1)^{signo} \times 1.mantisa \times 2^{exponente - desplazamiento} = (-1)^{1} \times 1.0100_{dos} \times 2^{129-127}$$
$$= -1.01_{dos} \times 2^{2} = -101_{dos} = -5.0_{diez}$$

Anteriormente aplicación lo visto en las anteriores secciones del documento, normalización, número asignado al signo, exponente, mantisa.

-0.75_{diez} en simple precisión

$$-0.75_{diez} = -3/4_{diez} = -3/2^{2}_{diez} = -11_{dos} / 2^{2}_{diez} = -0.11_{dos}$$

normalizada se tendría: -1.1×2^{-1}

1	01111110	10000000000000....
signo	exponente	mantisa

10. BIBLIOGRAFÍA

https://es.wikipedia.org/wiki/Coma_flotante