Métodos Numéricos

2do cuatrimestre 2020

Trabajo Practico n°1

Grupo 15

Barrera Pedro Lucas	58746
Bazano Valentina	59553
Cereigido Diego	59513

Sobre el trabajo

El trabajo consiste en crear una función que convierta un numero binario punto flotante IEEE 754 de 16 bits a un número en notación decimal y luego otra función que verifique el buen funcionamiento de la primer función creada.

En el programa se importa librería numpy para mayor eficiencia.

La funcion creada requiere que se ingrese un vector de 16 posiciones con caracteres iguales a 1 y/o 0 y todos separados con una coma. En caso contrario salta error.

Primero se define al exponente: en IEEE 754 de 16 bits el exponente ocupa 5 posiciones. Se convierten las cinco posiciones binarias a decimal y se resta el sesgo, lo que nos da un numero de exponente que nos servirá para clasificar el numero binario en los diferentes tipos de numero (normal, subnormal, infinito, etc.). El exponente indica dónde se coloca el punto decimal (o binario) en relación con el inicio de la mantisa. Exponentes negativos representan números menores que uno.

Luego se define a la mantisa: en IEEE 754 de 16 bits la mantisa ocupa 10 posiciones. Se convierten las 10 posiciones binarias a decimal. La mantisa contiene los dígitos del número. Mantisas negativas representan números negativos.

Se le asigna un valor positivo al signo y en caso de que el número de la posicion cero del vector sea igual a cero, el signo pasa a ser negativo.

Si el valor del exponente es -15 y la mantisa 0, el numero es de tipo cero (numero=0). Esto significa que el numero binario esta codificado en 15 de sus 16 bits con cero, sin importar el primer valor.

Si el valor del exponente es -15 y la mantisa es diferente de cero decimos que el número es de tipo subnormal (es decir que es un número muy pequeño), ya que dicha condición equivale a que los bits correspondientes al exponente sean todos ceros y la mantisa este compuesta por unos y ceros, o todos unos.

Si el exponente está entre -15 y 16, el número es de tipo normal. Esto equivale a que el exponente este compuesto por unos y ceros sin importar qué hay en la mantisa.

Si el exponente es 16 y la mantisa 0, el número es de tipo infinito. Esto equivale a que el exponente este compuesto por todos unos y la mantisa por ceros. El singo es variable.

Si el exponente es 16 y la mantisa es diferente a 0, no se trata de un número (tipo NaN). Esto equivale a que el exponente este compuesto por todos ceros y la mantisa este compuesta por unos y ceros, o todos unos.

La funcion test, testea la función creada anteriormente creada con números conocidos utilizando el assert para verificar la coincidencia del numero binario de 16 bits ingresado, con su resultado correcto. En caso de que haya algún error el programa deja de correr automaticamente.
Se prueba tanto un numero conocido positivo y negativo, de tipo sub-normal, de tipo NAN, y de tipo infinito tanto positivo como negativo.

Para ejecutar las funciones se crea un vector V que será el vector de 16 dígitos de tipo int donde se ingresan unos y ceros correspondientes al numero que se quiere traducir. El np.array hace mas eficiente al vecror, mejorando al programa.

Luego se aplica el testeo y la respuesta mostrada es el numero traducido ejecutando la función creada.

Código

```
import numpy as np


def binf2dec (v: np.array):


    if len(v)!=16:
        return "su vector debe contener 16 posiciones"


    for i in range (16):
        if (v[i]!=0) & (v[i]!=1):
            return "Error, ingrese unicamente 0 y 1 en las posiciones"


    exp=-15
    for i in range (5):
        index=(5-i)
        exp+=(2**i)*v[index]


    mantisa=0


    for i in range (10):
        index=(15-i)
        mantisa+=(2**i)*v[index]
    signo=1


    if v[0]==1:
```

```
        signo=-1


    if exp == -15 and mantisa == 0:
        numdec = 0
    elif exp == -15 and mantisa != 0:
        numdec = signo*(mantisa/1024)*2**float(-14)
    elif -15 < exp < 16:
        numdec = signo*(1 + mantisa/1024)*2**float(exp)
    elif exp == 16 and mantisa == 0:
        numdec = signo * float('Inf')
    elif exp == 16 and mantisa != 0:
        numdec = float("NaN")


    return numdec


def test():
    assert binf2dec(np.array([0,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1])) == 65504.0, ""
    assert binf2dec(np.array([1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1])) == -65504.0, ""
    assert binf2dec(np.array([0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1])) == 0.00006097555160522461, ""
    assert np.isnan(binf2dec(np.array([0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1])))
    assert binf2dec(np.array([0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0])) == float('Inf')
    assert binf2dec(np.array([1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0])) == -float('Inf')




v= np.array(list(map(int, input("ingrese vector de 16 caracteres, separados por coma y con 0 y
1 unicamente (ej:1,0,0,1,1...): ").split(','))))
test()
respuesta= binf2dec(v)


print(respuesta)
```