

# Práctica 4

Gonzalo Munoz Rubio

December 2022

## 1 El desarrollo del cálculo de la menor codificación del programa WHILE "diverger".

El programa con menor código que representa a la función diverger es el siguiente:

Q = (0, s)

s:

```
X1 := X1 + 1;  
while X1 ≠ 0 do  
    X1 := X1;  
od
```

A continuación, calcularé paso a paso el número correspondiente a su codificación, haciendo uso de las siguientes funciones: While2N, Sent2N y Code2N.

1)  $while2N(Q) = \sigma^2_1(n, code2N(c)) = \sigma^2_1(0, code2N(s)) = \sigma^2_1(0, 11.6262) = 67.599.377$

2)  $code2N(s) = \Gamma(sent2N(X_1 := X_1 + 1), sent2N(\text{while } X_1 \neq 0 \text{ do } X_1 := X_1)) = \Gamma(2, 14) = 11.626$

3)  $sent2N(X_1 := X_1 + 1) = 5\sigma^2_1(1-1, 1-1)+2 = 2$   
 $\sigma^2_1(0, 0) = 0$

4)  $sent2N(\text{while } X_1 \neq 0 \text{ do } X_1 := X_1) = 5\sigma^2_1(0, code2N(X_1 := X_1))+4 = 5\sigma^2_1(0, 1)+4 = 5*2+4 = 14$

$code2N(X_1 := X_1) = 5\sigma^2_1(1-1, 1-1)+1 = 1$

$\sigma^2_1(0, 0) = 0$

$\sigma^2_1(0, 1) = 2$

$\Gamma(2, 14) = \sigma^2_1(2-1, \sigma^2_1(2, 14)) = \sigma^2_1(1, 150) = 11.626$

$\sigma^2_1(2, 14) = (\text{Definición 11.1.12 de la teoría}) = 150$

$\sigma^2_1(1, 150) = (\text{Definición 11.1.12 de la teoría}) = 11.626$

$\sigma^2_1(0, 11.6262) = (\text{Definición 11.1.12 de la teoría}) = 67.599.377$

## 2 El código Octave que hace un print de todos los vectores, y una captura de ejemplo de ejecución.

```
function element = allVectors()

n = 0;

while n >= 0
    element = godeldecoding(n);
    n = n+1;
endwhile

end
```



The screenshot shows the execution of the `allVectors` function in an Octave terminal. The output displays the value of `element` for each iteration of `n` from 0 to 4. Each `element` is a binary vector represented as a row of 0s and 1s. The vectors correspond to the binary representations of the numbers 0 through 4, padded to a length of 5 bits.

```
octave:1> allVectors
element = [](5x0)
element = 0
element =
    0 0
element = 1
element =
    0 0 0
element =
    1 0
element = 2
element =
    0 0 0 0
element =
    1 0 0
element =
    0 1
element = 3
element =
    0 0 0 0 0
element =
    1 0 0 0
element =
    0 0 1
element =
    2 0
element = 4
element =
    0 0 0 0 0 0
```

## 3 El código Octave que hace un print de todos los programas WHILE, y una captura de ejecución.

```
function element = allWhile()

n = 0;

while n >= 0
    element = N2WHILE(n);
```

```

n = n+1;
endwhile

end

```

```

octave:1> allwhile
element = (0, X1=0)
element = (1, X1=0)
element = (0, X1=0; X1=0)
element = (2, X1=0)
element = (1, X1=0; X1=0)
element = (0, X1=0;1)
element = (3, X1=0)
element = (2, X1=0; X1=0)
element = (1, X1=0;1)
element = (0, X1=0; X1=0; X1=0)
element = (4, X1=0)
element = (3, X1=0; X1=0)
element = (2, X1=0;1)
element = (1, X1=0; X1=0; X1=0)
element = (0, X1=0;1; X1=0)
element = (5, X1=0)
element = (4, X1=0; X1=0)
element = (3, X1=0;1)
element = (2, X1=0; X1=0; X1=0)
element = (1, X1=0;1; X1=0)
element = (0, X1=0;1;1)
element = (6, X1=0)
element = (5, X1=0; X1=0)
element = (4, X1=0;1)
element = (3, X1=0; X1=0; X1=0)
element = (2, X1=0;1; X1=0)
element = (1, X1=0;1;1)
element = (0, X1=0; X1=0; X1=0; X1=0)
element = (7, X1=0)
element = (6, X1=0; X1=0)
element = (5, X1=0;1)
element = (4, X1=0; X1=0; X1=0)
element = (3, X1=0;1; X1=0)
element = (2, X1=0;1;1)
element = (1, X1=0; X1=0; X1=0; X1=0)
element = (0, X1=0;1; X1=0; X1=0)
element = (8, X1=0)
element = (7, X1=0; X1=0)
element = (6, X1=0;1)
element = (5, X1=0; X1=0; X1=0)
element = (4, X1=0;1; X1=0)
element = (3, X1=0;1;1)
element = (2, X1=0; X1=0; X1=0; X1=0)
element = (1, X1=0;1; X1=0; X1=0)
element = (0, X1=0; X1=0;1)
element = (9, X1=0)
element = (8, X1=0; X1=0)
element = (7, X1=0;1)

```