

Universidad del Valle de Guatemala
Facultad de Ingeniería



Laboratorio #2
Esquemas de detección y corrección de errores

Marco Antonio Jurado Velasquez 20308
Cristian Eduardo Aguirre Duarte 20231

Introducción

Los esquemas de detección y corrección de errores son utilizados en comunicación de datos para garantizar la integridad de la información . Estos algoritmos permiten detectar si ha ocurrido algún error durante la transmisión o el almacenamiento como el algoritmo CRC-32 y, en algunos casos, como lo es con el algoritmo de Hamming incluso corregirlos.

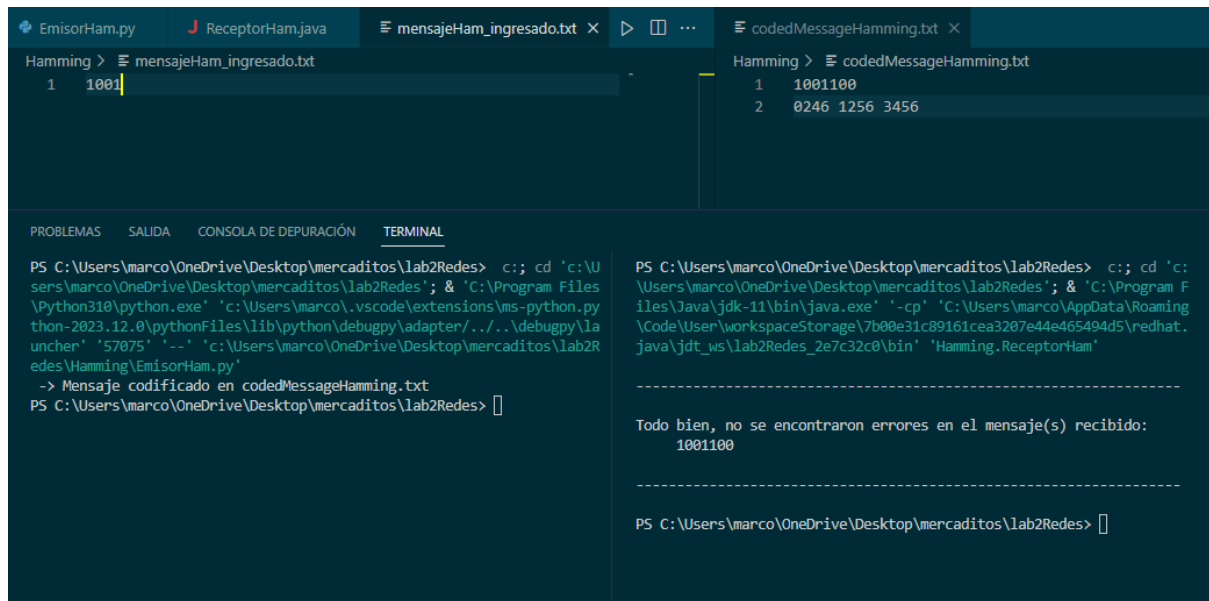
El algoritmo de Hamming es de detección y corrección de errores desarrollado por Richard Hamming. Este algoritmo realiza la adición de bits de paridad a un mensaje para detectar y así poder corregir errores de un solo bit en la cadena de mensaje codificada. Se colocan o se suman estos bits en casillas específicas para así poder diferenciar a la hora de encontrar un error en que bit específico se encuentra el error. Es utilizado por ejemplo en la memoria RAM de las computadoras.

CRC (Cyclic Redundancy Check) es un algoritmo de detección de errores que utiliza operaciones matemáticas de división. CRC-32 es una variante común del algoritmo que utiliza polinomios de 32 bits para calcular el valor de comprobación. En este caso se utilizó la variante vista en el curso. Al recibir el mensaje, el receptor también calcula el valor de verificación y lo compara con el valor recibido. Si ambos valores coinciden, se asume que el mensaje no tiene errores. CRC-32 se utiliza en sistemas de almacenamiento.

Resultados

Hamming

- Prueba con mensaje 1001
 - Sin cambiar bits



```
EmisorHam.py | ReceptorHam.java | mensajeHam_ingresado.txt | codedMessageHamming.txt
Hamming > mensajeHam_ingresado.txt
1 1001

Hamming > codedMessageHamming.txt
1 1001100
2 0246 1256 3456

PROBLEMAS | SALIDA | CONSOLA DE DEPURACIÓN | TERMINAL
PS C:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes> c:: cd 'c:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes'; & 'C:\Program Files\Python310\python.exe' 'c:\Users\marco\.vscode\extensions\ms-python.python-2023.12.0\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '57075' '--' 'c:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes\Hamming\EmisorHam.py'
-> Mensaje codificado en codedMessageHamming.txt
PS C:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes>

PS C:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes> c:: cd 'c:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes'; & 'C:\Program Files\Java\jdk-11\bin\java.exe' '-cp' 'C:\Users\marco\AppData\Roaming\Code\User\workspaceStorage\7b00e31c89161cea3207e44e465494d5\redhat.java\jdt_ws\lab2Redes_2e7c32c0\bin' 'Hamming.ReceptorHam'

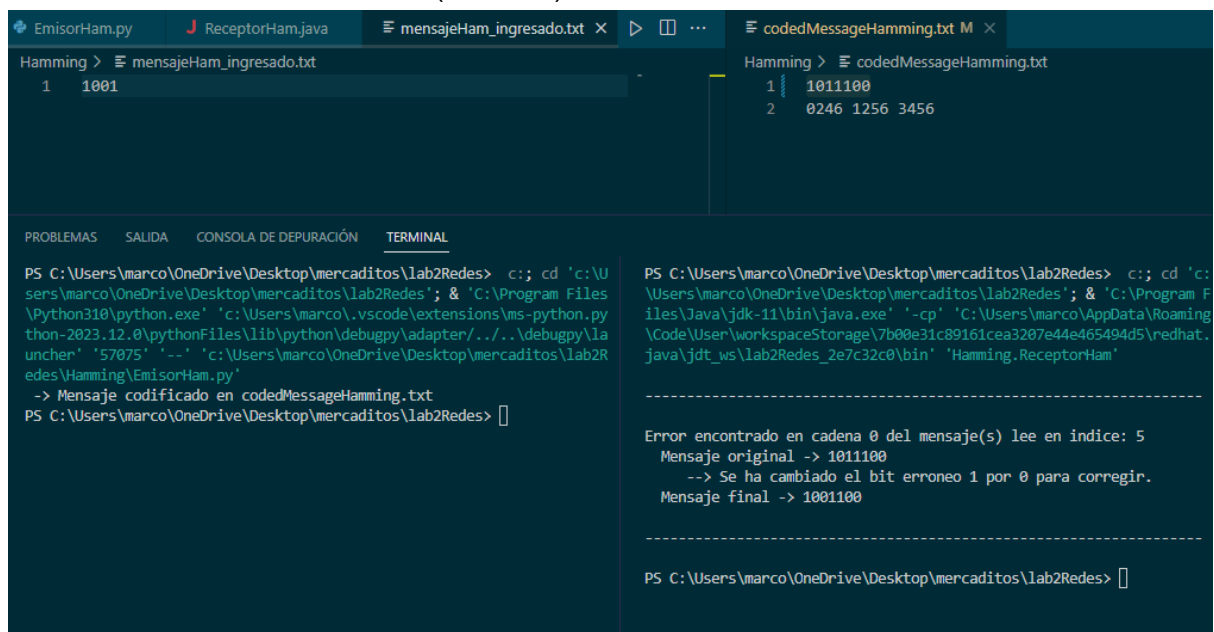
-----

Todo bien, no se encontraron errores en el mensaje(s) recibido:
1001100

-----

PS C:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes>
```

- Cambiando un bit (1011100)



```
EmisorHam.py | ReceptorHam.java | mensajeHam_ingresado.txt | codedMessageHamming.txt
Hamming > mensajeHam_ingresado.txt
1 1001

Hamming > codedMessageHamming.txt
1 1011100
2 0246 1256 3456

PROBLEMAS | SALIDA | CONSOLA DE DEPURACIÓN | TERMINAL
PS C:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes> c:: cd 'c:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes'; & 'C:\Program Files\Python310\python.exe' 'c:\Users\marco\.vscode\extensions\ms-python.python-2023.12.0\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '57075' '--' 'c:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes\Hamming\EmisorHam.py'
-> Mensaje codificado en codedMessageHamming.txt
PS C:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes>

PS C:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes> c:: cd 'c:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes'; & 'C:\Program Files\Java\jdk-11\bin\java.exe' '-cp' 'C:\Users\marco\AppData\Roaming\Code\User\workspaceStorage\7b00e31c89161cea3207e44e465494d5\redhat.java\jdt_ws\lab2Redes_2e7c32c0\bin' 'Hamming.ReceptorHam'

-----

Error encontrado en cadena 0 del mensaje(s) lee en indice: 5
Mensaje original -> 1011100
--> Se ha cambiado el bit erroneo 1 por 0 para corregir.
Mensaje final -> 1001100

-----

PS C:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes>
```

- Prueba con mensaje
 - Sin ningún error

```
isiorHam.py  ReceptorHam.java  mensajeHam_ingresado.txt M  codedMessageHamming.txt M
Hamming > mensajeHam_ingresado.txt
1 1100

PS C:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes> c:: cd 'c:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes'; & 'C:\Program Files\Python310\python.exe' 'c:\Users\marco\.vscode\extensions\ms-python.python-2023.12.0\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '57247' '--' 'c:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes\Hamming\EmissorHam.py'
-> Mensaje codificado en codedMessageHamming.txt
PS C:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes> c:: cd 'c:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes'; & 'C:\Program Files\Python310\python.exe' 'c:\Users\marco\.vscode\extensions\ms-python.python-2023.12.0\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '57255' '--' 'c:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes\Hamming\EmissorHam.py'
-> Mensaje codificado en codedMessageHamming.txt
PS C:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes>

PS C:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes> c:: cd 'c:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes'; & 'C:\Program Files\Java\jdk-11\bin\java.exe' '-cp' 'C:\Users\marco\AppData\Roaming\Code\User\workspaceStorage\7b00e31c89161cea3207e44e465494d5\redhat.java\jdt_ws\lab2Redes_2e7c32c0\bin' 'Hamming.ReceptorHam'

-----
Todo bien, no se encontraron errores en el mensaje(s) recibido:
0011110
-----

PS C:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes>
```

- Prueba con mensaje
 - Sin errores

```
isiorHam.py  ReceptorHam.java  mensajeHam_ingresado.txt M  codedMessageHamming.txt M
Hamming > mensajeHam_ingresado.txt
1 1011

PS C:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes> c:: cd 'c:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes'; & 'C:\Program Files\Python310\python.exe' 'c:\Users\marco\.vscode\extensions\ms-python.python-2023.12.0\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '57366' '--' 'c:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes\Hamming\EmissorHam.py'
-> Mensaje codificado en codedMessageHamming.txt
PS C:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes>

PS C:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes> c:: cd 'c:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes'; & 'C:\Program Files\Java\jdk-11\bin\java.exe' '-cp' 'C:\Users\marco\AppData\Roaming\Code\User\workspaceStorage\7b00e31c89161cea3207e44e465494d5\redhat.java\jdt_ws\lab2Redes_2e7c32c0\bin' 'Hamming.ReceptorHam'

-----
Todo bien, no se encontraron errores en el mensaje(s) recibido:
0101101
-----

PS C:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes>
```

- Pruebas cambiando un bit
 - 1101
 - 1011100

```
ReceptorHam.java
1 package Hamming;
2 import java.io.BufferedReader;
3 import java.io.FileReader;
4 import java.io.IOException;
5 import java.util.ArrayList;
6 import java.util.List;
7 import java.util.Arrays;
8
9 public class ReceptorHam {

mensajeHam_ingresado.txt
1 1101

codedMessageHamming.txt
1 1011100
2 0246 1256 3456

Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Instale la versión más reciente de PowerShell para obtener nuevas características y mejoras. https://aka.ms/PSWindows

PS C:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes> & 'C:\Program Files\Python310\python.exe' 'c:\Users\marco\.vscode\extensions\ms-python.python-2023.14.0\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '59628' '-.' 'c:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes\Hamming\EmisorHam.py'
-> Mensaje codificado en codedMessageHamming.txt
PS C:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes>

Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Instale la versión más reciente de PowerShell para obtener nuevas características y mejoras. https://aka.ms/PSWindows

PS C:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes> & 'C:\Program Files\Java\jdk-11\bin\java.exe' '-cp' 'C:\Users\marco\AppData\Roaming\Code\User\workspaceStorage\7b00e31c89161cea3207e44e465494d5\redhat.java\jdt_ws\lab2Redes_2e7c32c0\bin' 'Hamming.ReceptorHam'

-----
Error encontrado en cadena 0 del mensaje(s) lee en indice: 5
Mensaje original -> 1011100
-> Se ha cambiado el bit erroneo 1 por 0 para corregir.
Mensaje final -> 1001100
-----

PS C:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes>
```

- 1011
 - En esta situación se ha cambiado de 1100100 a 1101100 esto sin embargo altera al algoritmo de hamming y este reconoce que la cadena no esta correcta pero lo modifica no al original pero a otra trama. Eso se debe a la verificación de los bits de paridad que estos tambien coinciden con la corrección para generar dicha trama resultante

```
mensajeHam_ingresado.txt M X
Hamming > mensajeHam_ingresado.txt
1 1011

codedMessageHamming.txt M X
Hamming > codedMessageHam
1 1101100
2 0246 1256 3456

PS C:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes> c::; cd 'c:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes'; & 'C:\Program Files\Java\jdk-11\bin\javap' 'C:\Users\marco\AppData\Roaming\Code\User\workspaceStorage\7b00e31c89161e465494d5\redhat.java\jdt_ws\lab2Redes_2e7c32c0\bin' 'Hamming.ReceptorHam'

-----
Error encontrado en cadena 0 del mensaje(s) lee en indice: 2
Mensaje original -> 1100100
--> Se ha cambiado el bit erroneo 0 por 1 para corregir.
Mensaje final -> 1100110

-----
PS C:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes> c::; cd 'c:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes'; & 'C:\Program Files\Java\jdk-11\bin\javap' 'C:\Users\marco\AppData\Roaming\Code\User\workspaceStorage\7b00e31c89161e465494d5\redhat.java\jdt_ws\lab2Redes_2e7c32c0\bin' 'Hamming.ReceptorHam'

-----
Error encontrado en cadena 0 del mensaje(s) lee en indice: 3
Mensaje original -> 1101100
--> Se ha cambiado el bit erroneo 1 por 0 para corregir.
Mensaje final -> 1101000

-----
PS C:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes>
```

- 1111
 - cambia de 1110100 a 1111100

The image shows a code editor with two tabs: `mensajeHam_ingresado.txt` and `codedMessageHamming.txt`. The `mensajeHam_ingresado.txt` tab contains the text `1 1111`. The `codedMessageHamming.txt` tab contains the text `1 1111100` and `2 0246 1256 3456`. Below the editor is a terminal window showing a command prompt session. The command `c:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes> c:: cd 'c:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes'; & 'C:\Program Files\Java\jdk-11\bin\java.exe' '-cp' 'C:\Users\marco\AppData\Roaming\Code\User\workspaceStorage\7b00e31c89161cea3207e44e465494d5\redhat.java\jdt_ws\lab2Redes_2e7c32c0\bin' 'Hamming.ReceptorHam'` is executed. The output shows an error: `Error encontrado en cadena 0 del mensaje(s) lee en indice: 6`, `Mensaje original -> 1111100`, `--> Se ha cambiado el bit erroneo 1 por 0 para corregir.`, and `Mensaje final -> 1011100`. The terminal prompt is `PS C:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes>`.

De la misma manera los bits de paridad satisfacen dicha trama generada.

- Cambiar dos bits
 - 10011
 - En este caso la trama cambia de 1001100 0001111a 1001111 0001111

```

: \Users\marco\OneDrive\Desktop\mercaditos> python\debugpy\ad
thon.exe' 'c:\Users\marco\OneDrive\Desktop\mercaditos

: \Users\marco\OneDrive\Desktop\mercaditos> python\debugpy\ad
thon.exe' 'c:\Users\marco\OneDrive\Desktop\mercaditos

: \Users\marco\OneDrive\Desktop\mercaditos> python\debugpy\ad
thon.exe' 'c:\Users\marco\OneDrive\Desktop\mercaditos

PS C:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes> c.; cd 'c:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes'; & 'C:\Program Files\Java\jdk-11\bin\java.exe' '-cp' 'C:\Users\marco\AppData\Roaming\Code\User\workspaceStorage\7b00e31c89161cea3207e44e465494d5\redhat.java\jdt_ws\lab2Redes_2e7c32c0\bin' 'Hamming.ReceptorHam'

-----

Error encontrado en cadena 0 del mensaje(s) lee en indice: 6

-----

Error encontrado en cadena 1 del mensaje(s) lee en indice: 1
Mensaje original -> 1001111
Mensaje original -> 0001111
--> Se ha cambiado el bit erroneo 0 por 1 para corregir.
--> Se ha cambiado el bit erroneo 1 por 0 para corregir.

```

correctamente se detectan y se cambia

- 11011

■ la trama cambia de 1010100 0001111 a 1010110 0011111

```

python\debugpy\ad
esktop\mercaditos

Users\marco\OneDrive\Desktop\mercaditos> python\debugpy\ad
esktop\mercaditos

Users\marco\OneDrive\Desktop\mercaditos> python\debugpy\ad
esktop\mercaditos

PS C:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes> c.; cd 'c:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes'; & 'C:\Program Files\Java\jdk-11\bin\java.exe' '-cp' 'C:\Users\marco\AppData\Roaming\Code\User\workspaceStorage\7b00e31c89161cea3207e44e465494d5\redhat.java\jdt_ws\lab2Redes_2e7c32c0\bin' 'Hamming.ReceptorHam'

-----

Error encontrado en cadena 0 del mensaje(s) lee en indice: 6

-----

Error encontrado en cadena 1 del mensaje(s) lee en indice: 4
Mensaje original -> 1010110
Mensaje original -> 0011111
--> Se ha cambiado el bit erroneo 0 por 1 para corregir.
--> Se ha cambiado el bit erroneo 1 por 0 para corregir.

```

cambiando correctamente ambos bits.

- 11001

- la trama cambia de 0011110 0001111 a 0010010 0001111

```

mensajeHam_ingresado.txt M X
Hamming > mensajeHam_ingresado.txt
1 11001

codedMessageHamming.txt M X
Hamming > codedMessageHamming.txt
1 0010010-0001111
2 0246 1256 3456

PS C:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes> c:: cd 'c:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes'; & 'C:\Program Files\Java\jdk-11\bin\java.exe' '-cp' 'C:\Users\marco\AppData\Roaming\Code\User\workspaceStorage\7b00e31c89161cea3207e44e465494d5\redhat.java\jdt_ws\lab2Redes_2e7c32c0\bin' 'Hamming.ReceptorHam'

-----
Error encontrado en cadena 0 del mensaje(s) lee en indice: 7
-----

Error encontrado en cadena 1 del mensaje(s) lee en indice: 1
Mensaje original -> 0010010
Mensaje original -> 0001111
Error: el índice 0 está fuera del rango del arregloMensajes.
--> Se ha cambiado el bit erroneo 1 por 0 para corregir.
Mensaje final -> 00100100001110
-----

```

Cambiando ambos bits en la trama.

- Un error que Hamming no podrá detectar es cuando se cambian bits en alguna trama que no está cubierta por los bits de paridad
 - mensaje 1010

```

ReceptorHam.java mensajeHam_ingresado.txt M X
Hamming > mensajeHam_ingresado.txt
1 1010

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL

```

luego se cambia los bits a 1100001

```
orHam.py  ReceptorHam.java  mensajeHam_ingresado.txt M X  codedMessageHamming.txt M X
Hamming > mensajeHam_ingresado.txt
1 1010

Hamming > codedMessageHamming.txt
1 1100001
2 0246 1256 3456

PROBLEMAS  SALIDA  CONSOLA DE DEPURACIÓN  TERMINAL

-----

PS C:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes> c:; cd 'c:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes'; & 'C:\Program Files\Java\jdk-11\bin\java.exe' '-cp' 'C:\Users\marco\AppData\Roaming\Code\User\workspaceStorage\7b00e31c89161cea3207e44e465494d5\redhat\lab2Redes_2e7c32c0\bin' 'Hamming.ReceptorHam'

-----

Error encontrado en cadena 0 del mensaje(s) lee en indice: 1
Mensaje original -> 0100101
--> Se ha cambiado el bit erroneo 1 por 0 para corregir.
Mensaje final -> 0100100

-----

PS C:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes> c:; cd 'c:\Users\marco\OneDrive\Desktop\mercaditos\lab2Redes'; & 'C:\Program Files\Java\jdk-11\bin\java.exe' '-cp' 'C:\Users\marco\AppData\Roaming\Code\User\workspaceStorage\7b00e31c89161cea3207e44e465494d5\redhat\lab2Redes_2e7c32c0\bin' 'Hamming.ReceptorHam'

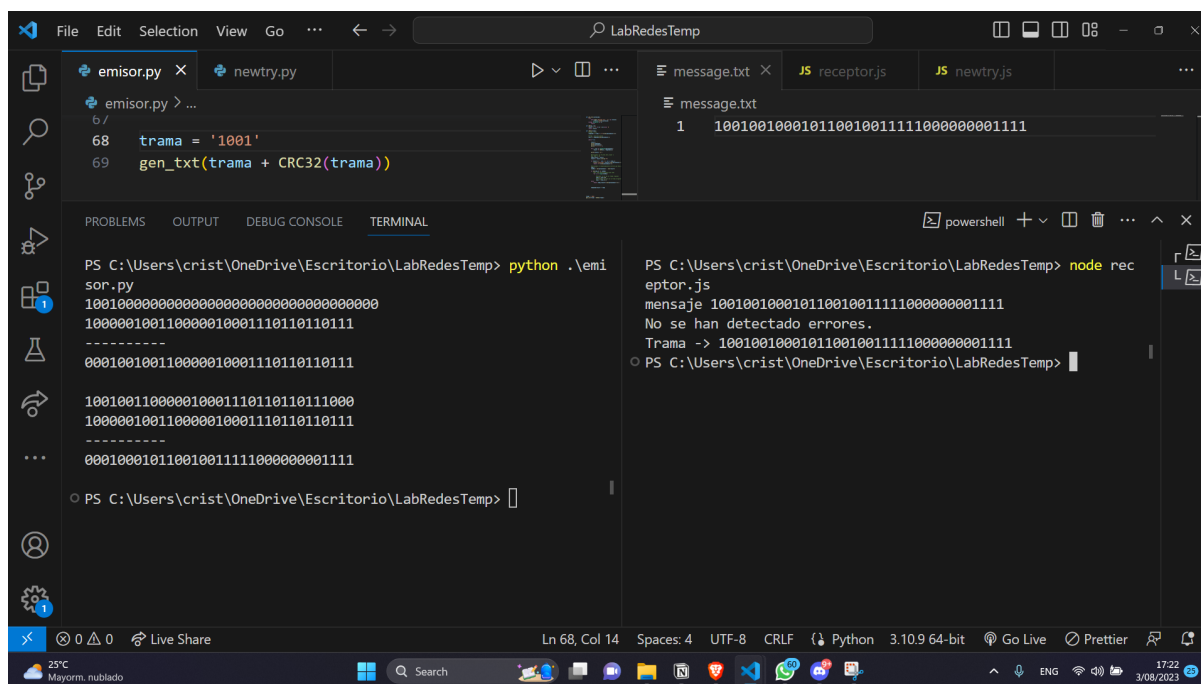
-----

Todo bien, no se encontraron errores en el mensaje(s) recibido:
1100001
-----
```

Podemos ver que Hamming dice que no hay errores pues si tenemos los errores presentes pues cambiamos 3 bits

CRC-32

- Trama:



File Edit Selection View Go ... LabRedesTemp

emisor.py x

```
emisor.py > ...
68
69 trama = '1100'
70 gen_txt(trama + CRC32(trama))
```

message.txt x JS receptor.js

message.txt

```
1 110000001101010000110010011011011001
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

PS C:\Users\cris\OneDrive\Escritorio\LabRedesTemp> python .\emisor.py

PS C:\Users\cris\OneDrive\Escritorio\LabRedesTemp>

PS C:\Users\cris\OneDrive\Escritorio\LabRedesTemp> node receptor.js

```
mensaje 110000001101010000110010011011011001
No se han detectado errores.
Trama -> 110000001101010000110010011011011001
```

PS C:\Users\cris\OneDrive\Escritorio\LabRedesTemp>

25°C Mayorm. nublado

Ln 1, Col 37 Spaces: 4 UTF-8 CRLF Plain Text Go Live Prettier

17:34 3/08/2023

File Edit Selection View Go ... LabRedesTemp

emisor.py x

```
emisor.py > ...
68
69 trama = '1010'
70 gen_txt(trama + CRC32(trama))
```

message.txt x JS receptor.js

message.txt

```
1 1010000101111100010101101101101011
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

PS C:\Users\cris\OneDrive\Escritorio\LabRedesTemp> python .\emisor.py

PS C:\Users\cris\OneDrive\Escritorio\LabRedesTemp>

PS C:\Users\cris\OneDrive\Escritorio\LabRedesTemp> python .\emisor.py

PS C:\Users\cris\OneDrive\Escritorio\LabRedesTemp>

PS C:\Users\cris\OneDrive\Escritorio\LabRedesTemp> node receptor.js

```
mensaje 1010000101111100010101101101101011
No se han detectado errores.
Trama -> 1010000101111100010101101101101011
```

PS C:\Users\cris\OneDrive\Escritorio\LabRedesTemp>

25°C Mayorm. nublado

Ln 69, Col 12 Spaces: 4 UTF-8 CRLF Python 3.10.9 64-bit Go Live Prettier

17:35 3/08/2023

Tramas modificadas para detección de errores

```
emisor.py
68
69 trama = '1001'
70 gen_txt(trama + CRC32(trama))

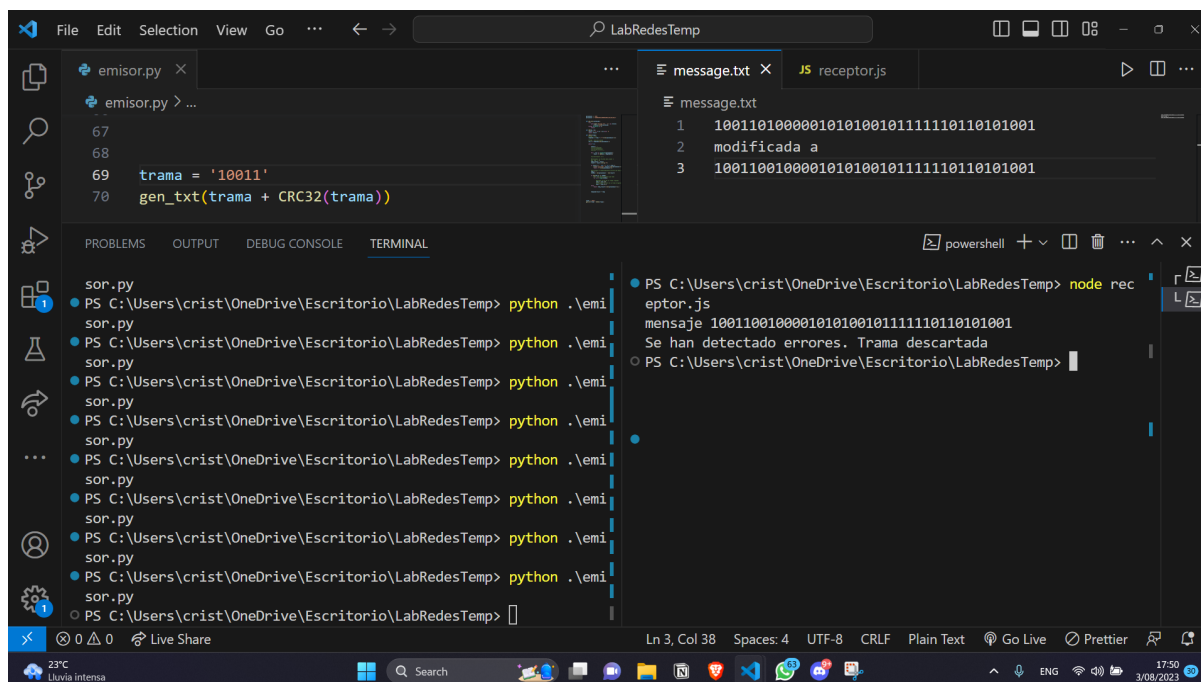
message.txt
1 10010010001011001001111100000001111
2 modificado a ->
3 00010010001011001001111100000001111

PS C:\Users\cris\OneDrive\Escritorio\LabRedesTemp> python .\emisor.py
PS C:\Users\cris\OneDrive\Escritorio\LabRedesTemp> python .\receptor.js
mensaje 10100001011110001010101010101011
No se han detectado errores.
Trama -> 10100001011110001010101010101011
PS C:\Users\cris\OneDrive\Escritorio\LabRedesTemp> node receptor.js
mensaje 10010010001011001001111100000001111
No se han detectado errores.
Trama -> 10010010001011001001111100000001111
PS C:\Users\cris\OneDrive\Escritorio\LabRedesTemp> node receptor.js
mensaje 00010010001011001001111100000001111
Se han detectado errores. Trama descartada
PS C:\Users\cris\OneDrive\Escritorio\LabRedesTemp>
```

```
emisor.py
68
69 trama = '1100'
70 gen_txt(trama + CRC32(trama))

message.txt
1 110000001101010000110010011011011001
2 mensaje modificado a
3 100000001101010000110010011011011001

PS C:\Users\cris\OneDrive\Escritorio\LabRedesTemp> python .\emisor.py
PS C:\Users\cris\OneDrive\Escritorio\LabRedesTemp> python .\receptor.js
mensaje 100000001101010000110010011011011001
Se han detectado errores. Trama descartada
PS C:\Users\cris\OneDrive\Escritorio\LabRedesTemp>
```



The screenshot shows a Visual Studio Code editor window titled 'LabRedesTemp'. The editor has three tabs: 'emisor.py', 'message.txt', and 'receptor.js'. The 'emisor.py' tab is active, showing a Python script with the following code:

```
67  
68  
69 trama = '10001'  
70 gen_txt(trama + CRC32(trama))
```

The 'message.txt' tab is also visible, showing a text file with the following content:

```
1 1000101001000110100001100011011000111  
2 modificada a  
3 1011101001000000100001100011011000111
```

The 'receptor.js' tab is also visible, showing a JavaScript script with the following code:

```
1  
2  
3
```

The terminal window at the bottom shows the output of the Python script. It displays the command 'python .\emisor.py' being executed multiple times, and the output of the 'gen_txt' function, which is '1011101001000000100001100011011000111'. The terminal also shows the command 'node receptor.js' being executed, and the output of the 'mensaje' variable, which is '1011101001000000100001100011011000111'. The terminal also shows the command 'Trama -> 1011101001000000100001100011011000111' and the command 'PS C:\Users\cris\OneDrive\Escritorio\LabRedesTemp>'.

Discusión

En los Esquemas de detección y corrección de errores, pudimos observar cómo se comportan los algoritmos de Hamming y CRC-32 durante su ejecución. Sin embargo, encontramos ciertas limitaciones en el algoritmo de Hamming en lo que respecta a la detección de errores, ya que depende en gran medida de los bits de paridad y su lógica de funcionamiento. En el algoritmo de Hamming notamos que cambiar dos o más bits en una trama puede generar situaciones en las que el mensaje parece estar correcto según Hamming. Un ejemplo de esto ocurre cuando ingresamos la cadena 1010 y aplicamos el algoritmo de Hamming. Inicialmente, el mensaje no muestra errores en su codificación. Después de aplicar la codificación de Hamming, obtenemos el código 0101101. Sorprendentemente, si modificamos algunos bits de este código y lo transformamos en 1100001, Hamming no detectará ningún error, ya que este nuevo código cumple con las reglas del algoritmo y se considera válido.

Esta particularidad del algoritmo de Hamming nos lleva a reflexionar sobre la importancia de complementar su uso con otros métodos de detección y corrección de errores, como el algoritmo CRC-32, que pueden proporcionar una mayor robustez en la detección de alteraciones en los datos transmitidos. Al emplear una combinación de estos algoritmos, podemos mejorar la fiabilidad y seguridad de nuestras comunicaciones y asegurar una detección más efectiva de posibles errores.

Comentario

- El algoritmo de Hamming a pesar de tener una explicación un poco más sencilla consta de muchas verificaciones de los bits de paridad lo cual lo hace más robusto pero no infalible.

Conclusiones

1. Hamming es efectivo para detectar y corregir errores en la transmisión de datos, pero no es infalible pues podemos encontrar que al tener errores que satisfacen las reglas estos no son perceptibles por el algoritmo.
2. Es esencial no solo hacer uso de un algoritmo pues tenemos implementaciones de código más robustas como el crc-32 que llega a ser un poco más difícil de engañar con errores en las tramas.

Citas y referencias bibliográficas

Conway, J. H., & Sloane, N. J. A. (1998). Sphere Packings, Lattices and Groups (3rd ed.). New York: Springer-Verlag. ISBN 0-387-98585-9. (requiere registro).

History of Hamming Codes. (n.d.). Archivado desde el original el 25 de octubre de 2007. Consultado el 3 de abril de 2008.