UNIVERSIDAD DEL VALLE DE GUATEMALA

Facultad de Ingeniería

Redes – Jorge Yass

Laboratorio 2. Parte 1

Francis Aguilar – 22243

Angela García – 22869

Introducción

En cualquier sistema de comunicación digital, los errores de transmisión representan uno de los principales desafíos a enfrentar. Estos errores pueden surgir por múltiples factores como interferencias, ruido o fallos en el medio de transmisión, lo que compromete la integridad de los datos enviados. Por esta razón, a lo largo del tiempo y con la evolución del Internet, se han desarrollado diversos algoritmos y mecanismos diseñados específicamente para detectar y corregir errores durante la transmisión de datos.

Descripción de la práctica

Este laboratorio tiene como propósito profundizar en el análisis y la implementación de estos algoritmos, permitiendo comprender su funcionamiento, ventajas y limitaciones. Además, se busca poner en práctica los conocimientos adquiridos sobre los servicios de la capa de Enlace, específicamente en lo referente a la detección y corrección de errores. A través del desarrollo colaborativo de programas que simulan tanto el emisor como el receptor, y utilizando distintos lenguajes de programación para cada uno, se refuerza la importancia de estos mecanismos en entornos de comunicación no confiables.

Capturas y evidencia

Tramas con solo 1 error

```
angel@galleta:~/redes/Lab2-redes$ ./correccion

=== CORRECION DE ERRORES ===
Nota: Ingrese un codigo Hamming que ya contenga un posible error, con su parieda

--- Emisor ---
Ingrese el codigo Hamming : 10101011011
Longitud de la información: 11 bits
En decimal: 1371
Codigo recibido en binario: 10101011011

--- DECODIFICACION ---
Codigo recibido (11 bits): 11011010101
Bits de datos: 7, Bits de paridad: 4

Calculo de sindrome:
S1 (P1): 1 3 5 7 9 11 -> 5 unos (impar) -> S1=1
S2 (P2): 2 3 6 7 10 11 -> 3 unos (impar) -> S2=1
S4 (P4): 4 5 6 7 -> 3 unos (impar) -> S4=1
S8 (P8): 8 9 10 11 -> 2 unos (par) -> S8=0
Sindrome: 7
Error detectado - Sindrome: 7
Posicion indicada por sindrome: 7

--- Receptor ---

/// ADVERTENCIA ///

El codigo Hamming solo puede:
1. Corregir 1 error de forma confiable
2. Detectar, pero no corregir, 2 errores
3. Con 3+ errores: comportamiento impredecible y usualmente lo suele empeorar

Intentar correccion? (s/n): s
Correccion aplicada en posicion 7
Trama corregida: 1010010
```

```
angel@galleta:~/redes/Lab2-redes$ ./correccion

=== CORRECION DE ERRORES ===
Nota: Ingrese un codigo Hamming que ya contenga un posible error, con su pariedad

--- Emisor ---
Ingrese el codigo Hamming : 1000101
Longitud de la información: 7 bits
En decimal: 69
Codigo recibido en binario: 1000101

--- DECODIFICACION ---
Codigo recibido (7 bits): 1010001
Bits de datos: 4, Bits de paridad: 3

Calculo de sindrome:
S1 (P1): 1 3 5 7 -> 3 unos (impar) -> S1=1
S2 (P2): 2 3 6 7 -> 2 unos (par) -> S2=0
S4 (P4): 4 5 6 7 -> 1 unos (impar) -> S4=1
Sindrome: 5
Error detectado - Sindrome: 5
Posicion indicada por sindrome: 5

--- Receptor ---

/// ADVERTENCIA ///

El codigo Hamming solo puede:
1. Corregir 1 error de forma confiable
2. Detectar, pero no corregir, 2 errores
3. Con 3+ errores: comportamiento impredecible y usualmente lo suele empeorar

Intentar correccion? (s/n): s
Correccion aplicada en posicion 5
Trama corregida: 1011
```

```
angel@galleta:~/redes/Lab2-redes$ ./correccion

=== CORRECION DE ERRORES ===
Nota: Ingrese un codigo Hamming que ya contenga un posible error, con su pariedad

--- Emisor ---
Ingrese el codigo Hamming : 10001010101
Longitud de la información: 11 bits
En decimal: 1109
Codigo recibido en binario: 10001010101

--- DECODIFICACION ---
Codigo recibido (11 bits): 10101010001
Bits de datos: 7, Bits de paridad: 4

Calculo de sindrome:
S1 (P1): 1 3 5 7 9 11 -> 5 unos (impar) -> S1=1
S2 (P2): 2 3 6 7 10 11 -> 3 unos (impar) -> S2=1
S4 (P4): 4 5 6 7 -> 2 unos (par) -> S4=0
S8 (P8): 8 9 10 11 -> 1 unos (impar) -> S8=1
Sindrome: 11
Error detectado - Sindrome: 11
Posicion indicada por sindrome: 11

--- Receptor ---

/// ADVERTENCIA ///

El codigo Hamming solo puede:
1. Corregir 1 error de forma confiable
2. Detectar, pero no corregir, 2 errores
3. Con 3+ errores: comportamiento impredecible y usualmente lo suele empeorar

Intentar correccion? (s/n): s
Correccion aplicada en posicion 11
Trama corregida: 0001011
```

Tramas con solo 2 o más errores

```
angel@galleta:~/redes/Lab2-redes$ ./correccion

=== CORRECION DE ERRORES ===
Nota: Ingrese un codigo Hamming que ya contenga un posible error, con su pariedad

--- Emisor ---
Ingrese el codigo Hamming : 1110111
Longitud de la información: 7 bits
En decimal: 119
Codigo recibido en binario: 1110111

--- DECODIFICACION ---
Codigo recibido (7 bits): 1110111
Bits de datos: 4, Bits de paridad: 3

Calculo de sindrome:
S1 (P1): 1 3 5 7 -> 4 unos (par) -> S1=0
S2 (P2): 2 3 6 7 -> 4 unos (par) -> S2=0
S4 (P4): 4 5 6 7 -> 3 unos (impar) -> S4=1
Sindrome: 4
Error detectado - Sindrome: 4
Posicion indicada por sindrome: 4

--- Receptor ---

/// ADVERTENCIA ///

El codigo Hamming solo puede:
1. Corregir 1 error de forma confiable
2. Detectar, pero no corregir, 2 errores
3. Con 3+ errores: comportamiento impredecible y usualmente lo suele empeorar

Intentar correccion? (s/n): s
Correccion aplicada en posicion 4
Trama corregida: 1111
```

```
=== CORRECION DE ERRORES ===
Nota: Ingrese un codigo Hamming que ya contenga un posible error, con su pariedad

--- Emisor ---
Ingrese el codigo Hamming : 11101010
Longitud de la información: 8 bits
En decimal: 234
Codigo recibido en binario: 11101010

--- DECODIFICACION ---
Codigo recibido (8 bits): 01010111
Bits de datos: 4, Bits de paridad: 4

Calculo de sindrome:
S1 (P1): 1 3 5 7 -> 1 unos (impar) -> S1=1
S2 (P2): 2 3 6 7 -> 3 unos (impar) -> S2=1
S4 (P4): 4 5 6 7 -> 3 unos (impar) -> S4=1
S8 (P8): 8 -> 1 unos (impar) -> S8=1
Sindrome: 15
Error detectado - Sindrome: 15
Posicion indicada por sindrome: 15

--- Receptor ---

/// ADVERTENCIA ///

El codigo Hamming solo puede:
1. Corregir 1 error de forma confiable
2. Detectar, pero no corregir, 2 errores
3. Con 3+ errores: comportamiento impredecible y usualmente lo suele empeorar

Intentar correccion? (s/n): s
Correccion aplicada en posicion 15
Trama corregida: 1100
```

```
=== CORRECION DE ERRORES ===
Nota: Ingrese un codigo Hamming que ya contenga un posible error, con su pariedad

--- Emisor ---
Ingrese el codigo Hamming : 0101001101101
Longitud de la información: 13 bits
En decimal: 2669
Codigo recibido en binario: 0101001101101

--- DECODIFICACION ---
Codigo recibido (13 bits): 1011011001010
Bits de datos: 9, Bits de paridad: 4

Calculo de sindrome:
S1 (P1): 1 3 5 7 9 11 13 -> 3 unos (impar) -> S1=1
S2 (P2): 2 3 6 7 10 11 -> 4 unos (par) -> S2=0
S4 (P4): 4 5 6 7 12 13 -> 4 unos (par) -> S4=0
S8 (P8): 8 9 10 11 12 13 -> 2 unos (par) -> S8=0
Sindrome: 1
Error detectado - Sindrome: 1
Posicion indicada por sindrome: 1

--- Receptor ---

/// ADVERTENCIA ///

El codigo Hamming solo puede:
1. Corregir 1 error de forma confiable
2. Detectar, pero no corregir, 2 errores
3. Con 3+ errores: comportamiento impredecible y usualmente lo suele empeorar

Intentar correccion? (s/n): s
Correccion aplicada en posicion 1
Trama corregida: 010101101
```

Trama sin errores

```
angel@galleta:~/redes/Lab2-redes$ ./correccion

=== CORRECION DE ERRORES ===
Nota: Ingrese un codigo Hamming que ya contenga un posible error, con su pariedad

--- Emisor ---
Ingrese el codigo Hamming : 10100011011
Longitud de la información: 11 bits
En decimal: 1307
Codigo recibido en binario: 10100011011

--- DECODIFICACION ---
Codigo recibido (11 bits): 11011000101
Bits de datos: 7, Bits de paridad: 4

Calculo de sindrome:
S1 (P1): 1 3 5 7 9 11 -> 4 unos (par) -> S1=0
S2 (P2): 2 3 6 7 10 11 -> 2 unos (par) -> S2=0
S4 (P4): 4 5 6 7 -> 2 unos (par) -> S4=0
S8 (P8): 8 9 10 11 -> 2 unos (par) -> S8=0
Sindrome: 0
No hay errores detectados
Trama corregida: 1010010
```

```
angel@galleta:~/redes/Lab2-redes$ ./correccion

=== CORRECION DE ERRORES ===
Nota: Ingrese un codigo Hamming que ya contenga un posible error, con su pariedad

--- Emisor ---
Ingrese el codigo Hamming : 11110000111
Longitud de la información: 11 bits
En decimal: 1927
Codigo recibido en binario: 11110000111

--- DECODIFICACION ---
Codigo recibido (11 bits): 11100001111
Bits de datos: 7, Bits de paridad: 4

Calculo de sindrome:
S1 (P1): 1 3 5 7 9 11 -> 4 unos (par) -> S1=0
S2 (P2): 2 3 6 7 10 11 -> 4 unos (par) -> S2=0
S4 (P4): 4 5 6 7 -> 0 unos (par) -> S4=0
S8 (P8): 8 9 10 11 -> 4 unos (par) -> S8=0
Sindrome: 0
No hay errores detectados
Trama corregida: 1110001
```

```
angel@galleta:~/redes/Lab2-redes$ ./correccion

  === CORRECION DE ERRORES ===
  Nota: Ingrese un codigo Hamming que ya contenga un posible error, con su pariedad

  --- Emisor ---
  Ingrese el codigo Hamming : 00000000000
  Longitud de la información: 11 bits
  En decimal: 0
  Codigo recibido en binario: 00000000000

  --- DECODIFICACION ---
  Codigo recibido (11 bits): 00000000000
  Bits de datos: 7, Bits de paridad: 4

  Calculo de sindrome:
  S1 (P1): 1 3 5 7 9 11 -> 0 unos (par) -> S1=0
  S2 (P2): 2 3 6 7 10 11 -> 0 unos (par) -> S2=0
  S4 (P4): 4 5 6 7 -> 0 unos (par) -> S4=0
  S8 (P8): 8 9 10 11 -> 0 unos (par) -> S8=0
  Sindrome: 0
  No hay errores detectados
  Trama corregida: 0000000
```

Respuesta a las preguntas mencionadas

¿Es posible manipular los bits de tal forma que el algoritmo seleccionado no sea capaz de detectar el error? ¿Por qué sí o por qué no? En caso afirmativo, demuestrelo con su implementación

Si es posible manipular los bits de tal forma que el algoritmo de Hamming no detecte el error correctamente, eso puede ser si el resultado del síndrome es igual a 0, manipulando 2 bits para que el resultado sea 0 como si no existiera un error. La otra forma es que detecte mal un síndrome incorrecto, que este indique una posición incorrecta.

Input con dos errores que no se pueden detectar: 1100110

```
angel@galleta:~/redes/Lab2-redes$ ./correccion

=== CORRECION DE ERRORES ===
Nota: Ingrese un codigo Hamming que ya contenga un posible error, con su pariedad

--- Emisor ---
Ingrese el codigo Hamming : 1100110
Longitud de la información: 7 bits
En decimal: 102
Codigo recibido en binario: 1100110

--- DECODIFICACION ---
Codigo recibido (7 bits): 0110011
Bits de datos: 4, Bits de paridad: 3

Calculo de sindrome:
S1 (P1): 1 3 5 7 -> 2 unos (par) -> S1=0
S2 (P2): 2 3 6 7 -> 4 unos (par) -> S2=0
S4 (P4): 4 5 6 7 -> 2 unos (par) -> S4=0
Sindrome: 0
No hay errores detectados
Trama corregida: 1101
```

Como se ve en la imagen, los errores múltiples dan como resultado un síndrome incorrecto.

En base a las pruebas que realizó, ¿qué ventajas y desventajas posee cada algoritmo con respecto a los otros dos?

Hamming:

Ventajas	Desventajas
- Corrige cualquier error de 1 bit, sin la necesidad de retransmisión - Bajo costo computacional - Corrección en tiempo real - Fácil implementación	- Solo puede corregir error de 1 bit, si hay dos o más errores puede dar resultados incorrectos o no detectar el error - No detecta todos los errores múltiples - Es ineficiente en canales muy ruidosos - Tiene poca escalabilidad

Discusión

Durante el desarrollo del laboratorio fue posible comprobar que, si bien los algoritmos de detección y corrección de errores son herramientas fundamentales en las comunicaciones digitales, presentan limitaciones importantes que deben tomarse en cuenta dependiendo del contexto en el que se utilicen. En nuestro caso, implementamos el algoritmo de Hamming en el lenguaje C, lo cual nos permitió tener un mayor control sobre la manipulación de bits y observar de forma precisa el comportamiento del algoritmo en diferentes escenarios. Se realizaron pruebas prácticas en tres casos distintos.

Durante las pruebas realizadas, se evaluó el comportamiento del algoritmo de Hamming implementado en C bajo distintos escenarios. En los casos sin errores, se enviaron mensajes al emisor, se copió el mensaje codificado sin alteraciones y se entregó al receptor, quien mostró correctamente el mensaje original, confirmando que el sistema funciona adecuadamente en condiciones ideales. Al introducir un solo error, es decir, cambiar un único bit en el mensaje codificado, el algoritmo fue capaz de detectar y corregir correctamente el error, indicando su posición y recuperando el mensaje original en los tres casos probados con diferentes longitudes. Sin embargo, al introducir dos o más errores, el algoritmo falló: en algunos casos generó un síndrome incorrecto que llevó a una corrección errónea, y en otros, no detectó ningún error, ya que los cambios anulaban el efecto del síndrome, como se evidenció con el ejemplo 1100110. Estas pruebas confirmaron la eficacia del algoritmo ante errores aislados y sus limitaciones frente a errores múltiples.

Conclusiones

- El algoritmo de Hamming es útil en escenarios donde predominan errores aislados
- La elección de un algoritmo debe considerar tanto su capacidad de corrección como su costo computacional y facilidad de implementación.
- La eficacia depende del tipo de error y del entorno de transmisión.

Referencias

Hamming, R. W. (1950). Error detecting and error correcting codes. *Bell System Technical Journal*, 29(2), 147–160. https://doi.org/10.1002/j.1538-7305.1950.tb00463.x

GeeksforGeeks. (2021, July 28). *Hamming Code - Error detection and error correction*. https://www.geeksforgeeks.org/hamming-code-error-detection-and-error-correction/