Universidad del Valle de Guatemala
Redes
Sec. 20

Andres Quezada 21085
Javier Chavez 21016
25/07/24

Laboratorio #2 – Parte 2

Esquemas de detección y corrección

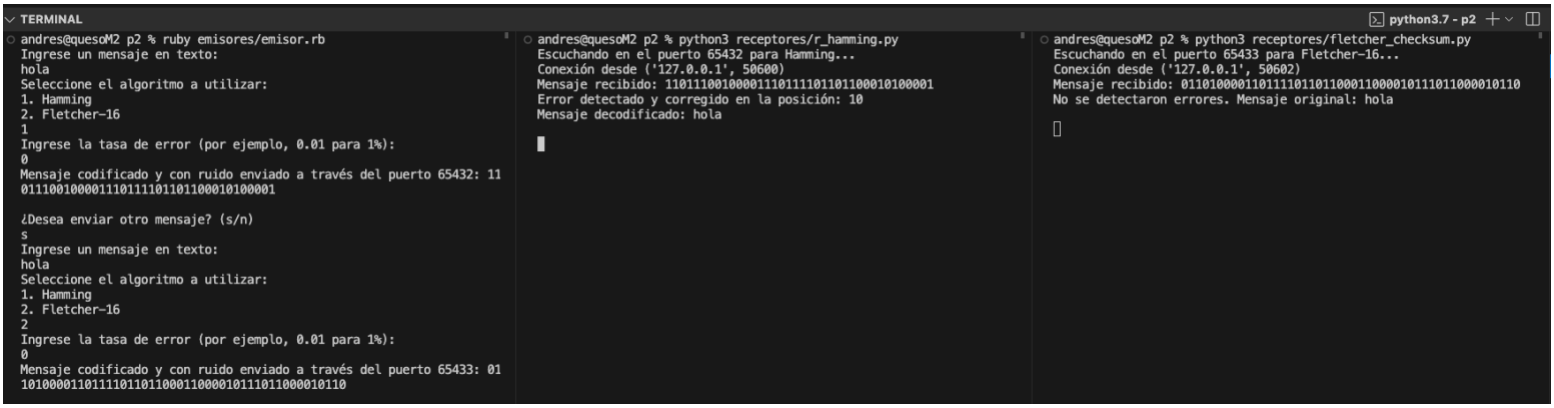
Descripción de la práctica

Esta práctica consiste en la implementación de un sistema de comunicación que utiliza los algoritmos de Hamming y Fletcher-16 para la codificación, detección y corrección de errores en la transmisión de mensajes binarios. Busca similar el comportamiento de una arquitectura de capas de comunicación que implementan distintos servicios para transmitir mensajes.

Resultados

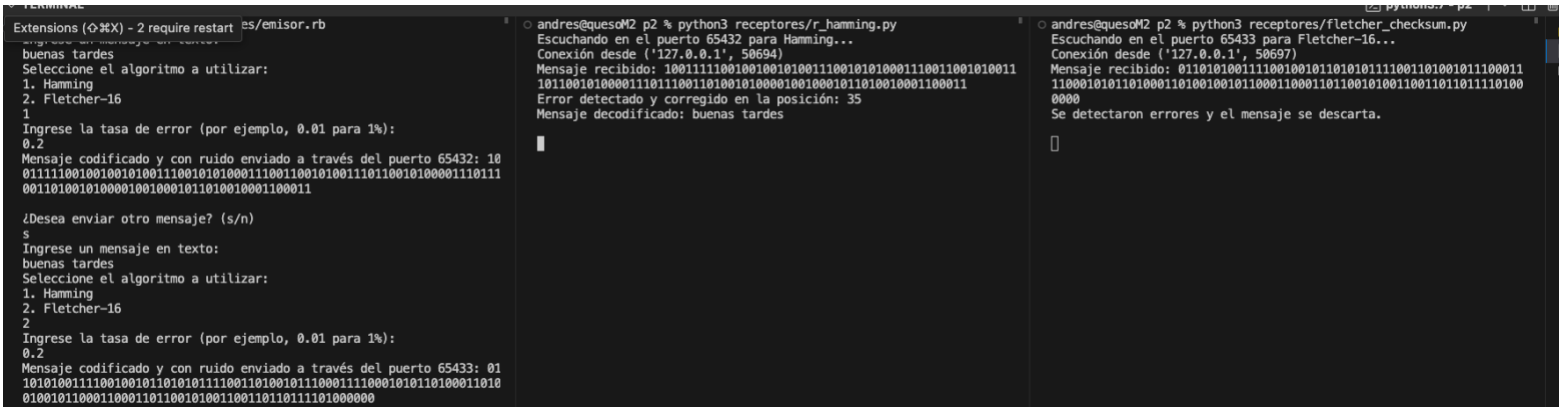
Demostraciones de ejecución:

- Mensaje: “hola” , error_rate: 0



- Mensaje: “hola como estas”, error_rate: 0

- Mensaje: “buenas tardes”, error_rate: 0.2



Análisis de resultados:

Para esta parte decidimos realizar un script de testing que manda 24 mensajes, 12 a cada algoritmo. Variando los siguientes parámetros:

- Longitud de cadena: 16, 32, 64 o 128
- Error rate: 0, 0.01, 0.05, 0.1

Log de lo enviado:

Longitud	Tasa de Error	Algoritmo
16	0	Hamming
16	0	Fletcher-16
16	0.01	Hamming
16	0.01	Fletcher-16
16	0.05	Hamming
16	0.05	Fletcher-16
16	0.1	Hamming
16	0.1	Fletcher-16
32	0	Hamming
32	0	Fletcher-16
32	0.01	Hamming
32	0.01	Fletcher-16
32	0.05	Hamming
32	0.05	Fletcher-16
32	0.1	Hamming
32	0.1	Fletcher-16
64	0	Hamming
64	0	Fletcher-16
64	0.01	Hamming
64	0.01	Fletcher-16
64	0.05	Hamming
64	0.05	Fletcher-16
64	0.1	Hamming
64	0.1	Fletcher-16
128	0	Hamming
128	0	Fletcher-16
128	0.01	Hamming
128	0.01	Fletcher-16
128	0.05	Hamming
128	0.05	Fletcher-16
128	0.1	Hamming
128	0.1	Fletcher-16

Y luego tabulamos automáticamente en los receptores, los siguientes datos:

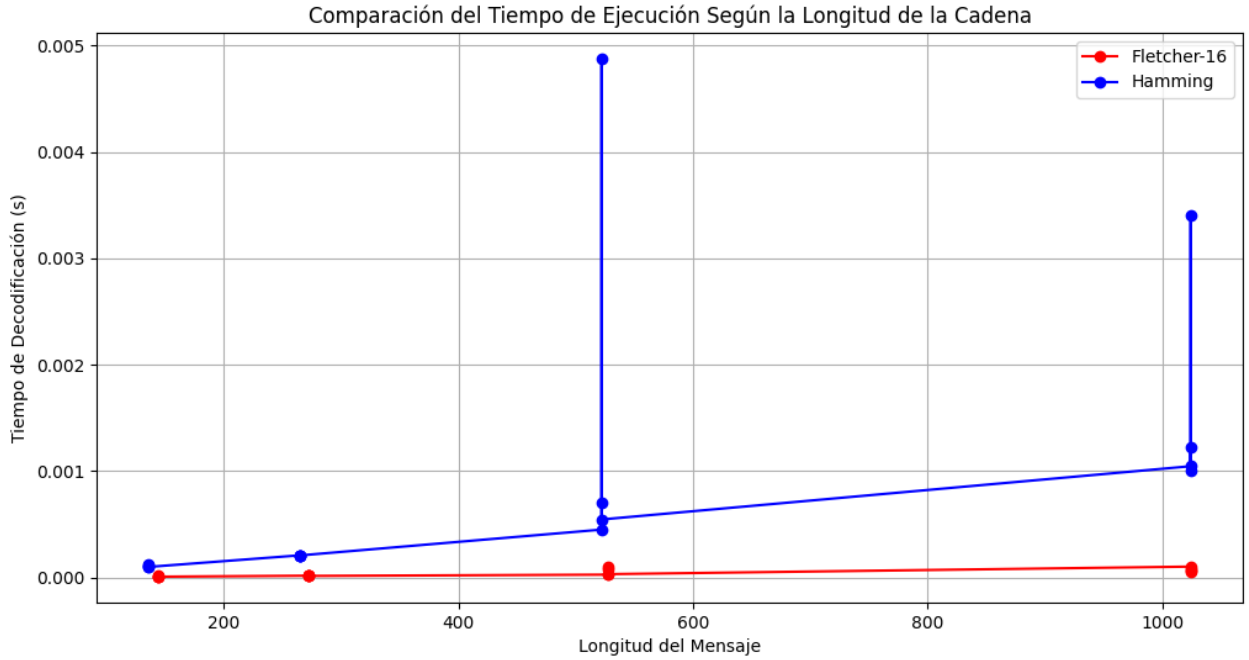
Receptor Hamming:

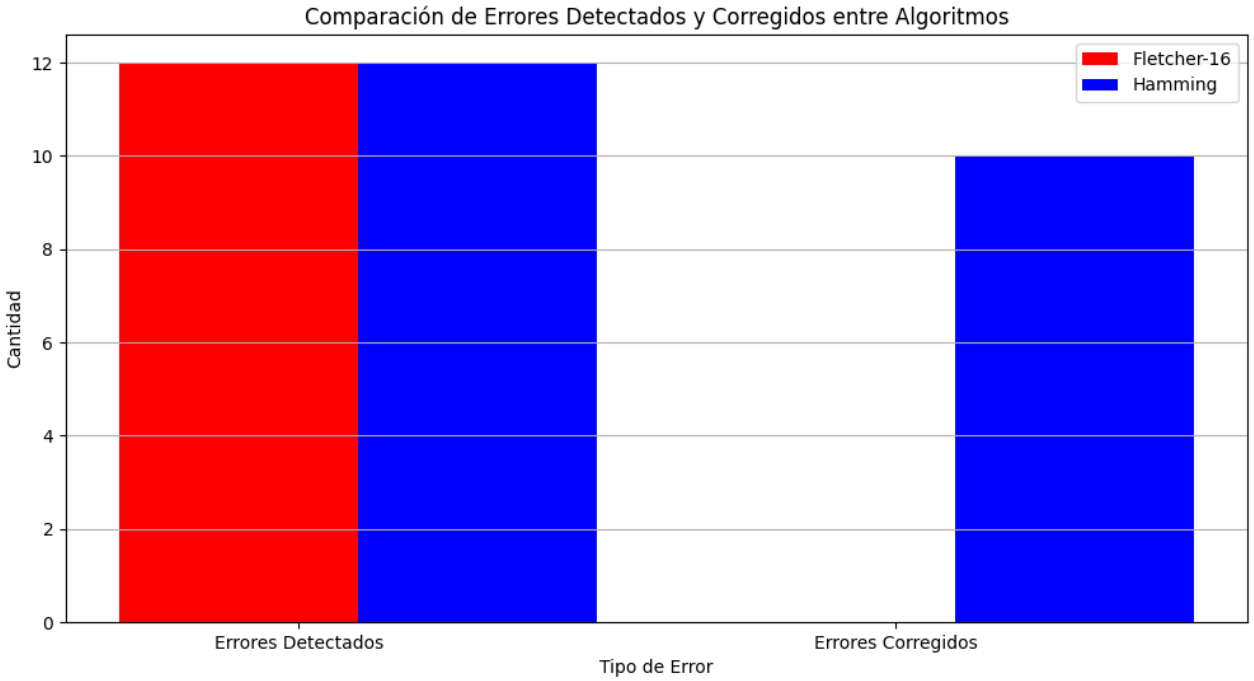
Longitud	Tasa de Error	Errores Detectados	Errores Corregidos	Tiempo de Decodificación
136	0.426470588	FALSE	TRUE	0.000126839
136	0.397058824	FALSE	TRUE	0.00010705
136	0.404411765	TRUE	FALSE	9.99E-05
136	0.419117647	TRUE	FALSE	9.89E-05
265	0.430188679	FALSE	TRUE	0.000208139
265	0.422641509	TRUE	TRUE	0.000207186
265	0.4	TRUE	TRUE	0.000205755
265	0.441509434	TRUE	TRUE	0.000206947
522	0.400383142	FALSE	TRUE	0.000449896
522	0.440613027	TRUE	TRUE	0.000698805
522	0.392720307	TRUE	TRUE	0.004878044
522	0.431034483	TRUE	FALSE	0.000545025
1024	0.420898438	TRUE	TRUE	0.001045942
1024	0.427734375	TRUE	FALSE	0.003405094
1024	0.434570313	TRUE	FALSE	0.001008034
1024	0.44921875	TRUE	FALSE	0.001229048

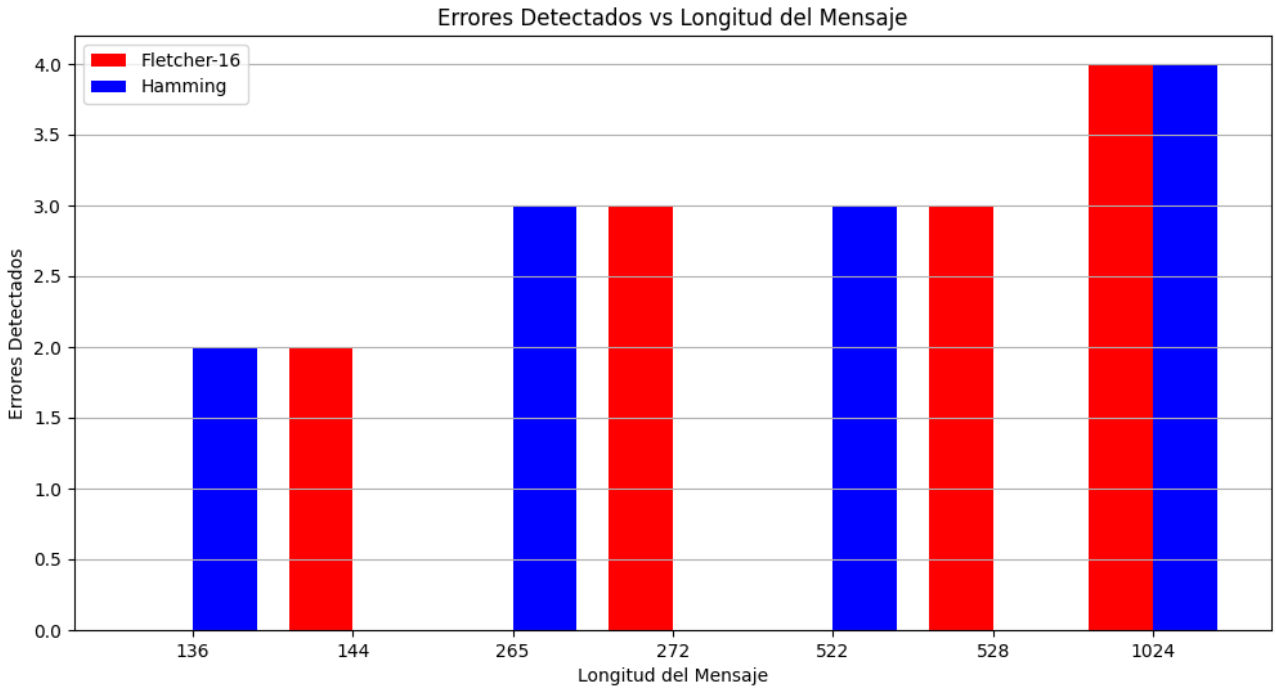
Receptor Fletcher:

Longitud	Tasa de Error	Errores Detectados	Errores Corregidos	Tiempo de Decodificación
144	0.395833333	FALSE	FALSE	1.48E-05
144	0.368055556	FALSE	FALSE	9.06E-06
144	0.423611111	TRUE	FALSE	6.91E-06
144	0.409722222	TRUE	FALSE	6.91E-06
272	0.400735294	FALSE	FALSE	1.62E-05
272	0.422794118	TRUE	FALSE	1.48E-05
272	0.404411765	TRUE	FALSE	1.53E-05
272	0.426470588	TRUE	FALSE	1.41E-05
528	0.386363636	FALSE	FALSE	2.62E-05
528	0.393939394	TRUE	FALSE	9.58E-05
528	0.428030303	TRUE	FALSE	7.92E-05
528	0.445075758	TRUE	FALSE	3.10E-05
1024	0.411132813	TRUE	FALSE	0.000101805
1024	0.416015625	TRUE	FALSE	6.29E-05
1024	0.436523438	TRUE	FALSE	5.82E-05
1024	0.422851563	TRUE	FALSE	7.70E-05

Y luego creamos un script en Python que nos permitió generar gráficas para analizar los resultados:






Errores Detectados vs Longitud del Mensaje
Fletcher-16
Hamming
Errores Detectados
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
136
144
265
272
522
528
1024
Longitud del Mensaje

Discusión

Observando los resultados obtenidos de los datos tabulados y las gráficas generadas, se puede determinar que el algoritmo Fletcher-16 muestra un rendimiento superior en términos de tiempo de decodificación en comparación con el Hamming. Fletcher-16 tiene tiempos de decodificación significativamente menores, lo que indica que es más eficiente en la detección de errores en todos los tamaños de mensaje y tasas de error evaluadas. Esto sugiere que Fletcher-16 puede ser más adecuado para aplicaciones donde el tiempo de procesamiento es crítico.

En cuanto a la flexibilidad para aceptar mayores tasas de error, el algoritmo Hamming obviamente es el más tolerante. A pesar de que Fletcher-16 es más rápido en general, Hamming maneja tasas de error más altas de manera más robusta. Las tasas de error elevadas, como 0.1, no afectan tanto a la capacidad de corrección de Hamming. Los resultados muestran que, con tasas de error más altas, Hamming aún mantiene una capacidad de corrección aceptable, mientras que Fletcher-16 nunca va a poder corregir errores, solamente detectarlos.

La elección entre un algoritmo de detección de errores y uno de corrección de errores depende en gran medida del contexto y los requisitos específicos de la aplicación. Los algoritmos de detección de errores, como Fletcher-16, son ideales en situaciones donde se requiere alta velocidad y el costo de reintentos o retransmisiones es bajo. Estos algoritmos identifican errores en los datos, pero no pueden corregirlos, lo que es adecuado cuando el sistema puede manejar la retransmisión de datos incorrectos o cuando se tiene una alta capacidad de corrección en el protocolo de comunicación.

Por otro lado, los algoritmos de corrección de errores, como Hamming, son preferibles en entornos donde la integridad de los datos es crucial y la corrección automática de errores es necesaria. Por que estos no tienen necesidad de retransmisión, y esto es útil en comunicaciones con alta probabilidad de error o en sistemas donde la retransmisión de datos no es práctica o es demasiado costosa. Por ejemplo, en sistemas de transmisión en tiempo real, como las comunicaciones de voz o video, la corrección de errores permite una mayor robustez y fiabilidad de la comunicación.

Comentario grupal

Para este laboratorio trabajamos en algoritmos de detección y corrección de errores, nos pareció interesante la conexión entre los sockets, la parte de simular la arquitectura nos hizo una idea de como funcionan las redes en la vida real.

Conclusiones:

- Los resultados muestran que Fletcher-16 tiene tiempos de decodificación significativamente menores en comparación con Hamming. Por ejemplo, con una longitud de mensaje de 1024 bits y una tasa de error de 0, Fletcher-16 toma aproximadamente 6.29E-05 segundos, mientras que Hamming toma alrededor de 0.001045942 segundos.

- Hamming tiene la capacidad de corregir errores que Fletcher-16 no puede. En mensajes con longitud de 1024 bits y tasas de error alrededor del 0.4, Hamming detecta y corrige errores en la mayoría de los casos, mientras que Fletcher-16 solo detecta errores pero no los corrige. Por ejemplo, a una tasa de error de 0.434570313, Hamming detecta y corrige errores, mientras que Fletcher-16 solo los detecta.

- Fletcher-16 es más eficiente en términos de tiempo de decodificación, lo que lo hace bueno para aplicaciones donde la velocidad es importante.

- Hamming es más robusto para la corrección de errores, ya que logra manejar mejor las tasas de error altas y puede evitar la necesidad de retransmisiones.

Referencias:

GeeksForGeeks .(2024) Hamming Code in Computer Network. Recuperado de: https://www.geeksforgeeks.org/hamming-code-in-computer-network/

Nakassis, A. (2010) Fletcher's Error Detection Algorithm: How to implement it efficiently and how to avoid the most common pitfalls. Recuperado de: https://dl.acm.org/doi/pdf/10.1145/53644.53648