UNIVERSIDAD DEL VALLE DE GUATEMALA

Redes

Sección 20

“Laboratorio 2 Parte 2”

ESTEBAN ALDANA GUERRA 20591

JESSICA PAMELA ORTIZ IXCOT 20192

GUATEMALA, Agosto 2023

Descripción de la práctica y metodología de la práctica

En esta práctica, implementamos un sistema de comunicación que utiliza el Cyclic Redundancy Check (CRC) para detectar errores en los mensajes transmitidos entre un emisor y un receptor. El CRC es un método popular y eficiente para detectar errores de transmisión, como los causados por ruido o interferencias.

El sistema se compone de dos partes principales:

Emisor: Convierte mensajes de texto a formato binario y calcula un código CRC para el mensaje. Luego envía el mensaje concatenado con su código CRC al receptor.
Receptor: Recibe el mensaje con el CRC, recalcula el CRC para el mensaje recibido y lo compara con el CRC recibido para verificar si hay errores.
Adicionalmente, para simular un escenario realista, introducimos un módulo de ruido que altera aleatoriamente bits del mensaje con una cierta probabilidad antes de que llegue al receptor.

Se utiliza una gráfica para mostrar el porcentaje de éxito en función de la longitud del mensaje.

Esta metodología permite evaluar la eficacia del CRC para detectar errores en la transmisión y entender cómo el ruido puede afectar la integridad de los mensajes transmitidos. Es una práctica esencial para quienes trabajan en comunicaciones y buscan garantizar la integridad de los datos en entornos ruidosos.

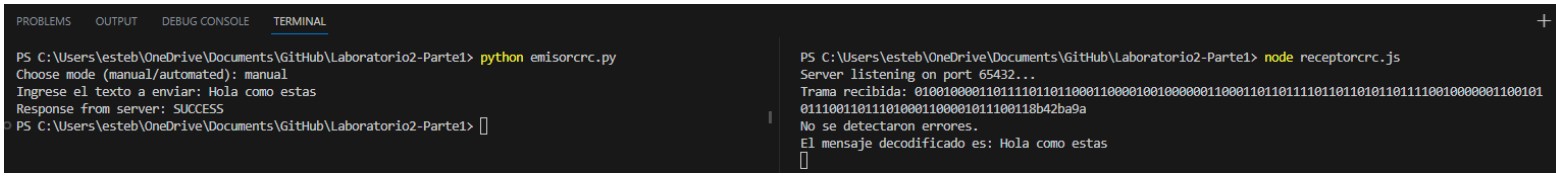
La simulación implementada en los scripts emisorhamming.py y receptorhamming.js representa el proceso de transmisión de mensajes utilizando el código Hamming, un método de corrección de errores. En este proceso, el emisor transforma un mensaje textual a formato binario, se codifica usando el código Hamming y, posteriormente, aplica intencionalmente errores o "ruido" basados en una probabilidad definida. Tras estas transformaciones, el mensaje se envía al receptor.

El emisor, tras codificar el mensaje y aplicar el ruido, transmite el dato a través de una conexión de socket. Por otro lado, el receptor, escuchando en un puerto específico, recibe este mensaje con ruido y lo decodifica usando el código Hamming, intentando corregir los errores introducidos. Si el código detecta y corrige los errores con éxito, el receptor transforma el mensaje binario de vuelta a formato textual.

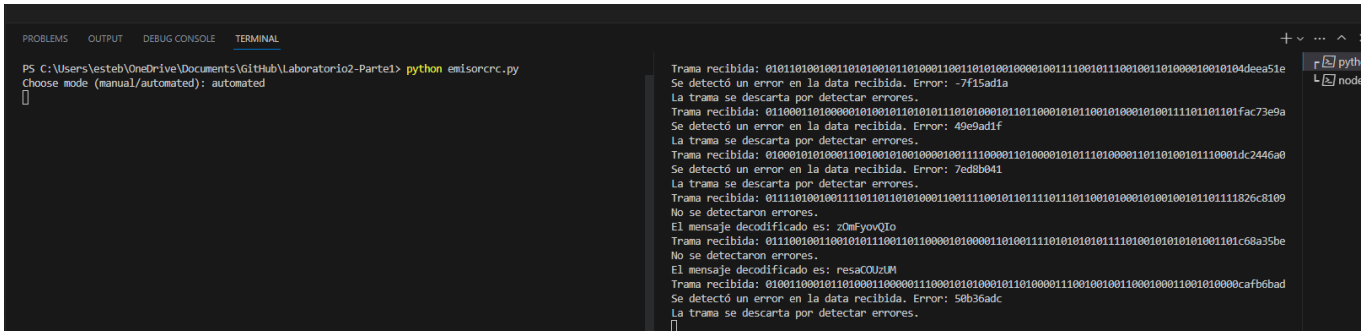
Resultados

Para el algoritmo crc-32

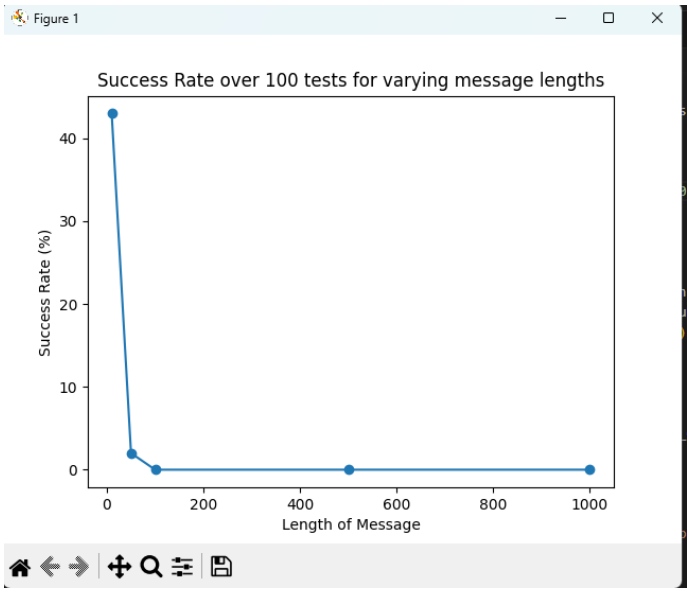
Correr manualemente el mensaje



Correr las pruebas con 1000 iteraciones con diferentes tamaños de lenght y añadieno ruido



Grafica



Para el algoritmo hamming

Correr manualmente el programa:

Las pruebas no se lograron realizar para hamming

Discusión

En la implementación del algoritmo CRC, observamos su comportamiento al introducir un mensaje manualmente y transmitirlo al receptor. Cuando se introduce ruido en la comunicación, es raro que el CRC no detecte estos errores. De hecho, cada vez que se perturba el mensaje con ruido, el CRC identifica y descarta la trama afectada.

En cuanto a las pruebas automatizadas, se destaca que, con mensajes de pocos caracteres, incluso en presencia de ruido, la tasa de error es mínima, manteniendo un alto índice de éxito. Sin embargo, a medida que aumenta la longitud del mensaje y se intensifica el ruido, la tasa de transmisiones exitosas disminuye significativamente.

La implementación del código Hamming en la transmisión de mensajes resalta la importancia de la corrección de errores en comunicaciones digitales. A pesar de los desafíos presentados por el ruido y las interferencias inevitables en muchos sistemas de transmisión, la metodología empleada en los scripts permite al usuario visualizar cómo, al ingresar un mensaje manualmente, el emisor puede modificarlo y, después de la introducción intencional de errores, el receptor tiene la capacidad de detectar y corregir dichas anomalías. Esto valida la robustez del código Hamming en tales situaciones y su relevancia en la garantía de integridad de la información.

Conclusiones

- La implementación con sockets nos permitió entender la esencia de un modelo de capas, demostrando cómo cada capa contribuye a una transmisión confiable, especialmente en canales no seguros.

- Si bien el CRC es efectivo en detectar errores, su eficacia disminuye con mensajes más largos y mayor ruido. Esto destaca la importancia de adaptar algoritmos y considerar las capacidades del canal al transmitir información.

Citas y Referencias

Wikipedia. Algoritmo de los códigos de redundancia cíclica. Wikipedia. https://es.wikipedia.org/wiki/Algoritmo_de_los_c%C3%B3digos_de_redundancia_c%C3%ADclica

Python Software Foundation.. Sockets. Documentación oficial de Python. https://docs.python.org/es/3/howto/sockets.html

GeeksforGeeks. (2023). Hamming code in computer network. *GeeksforGeeks*.

https://www.geeksforgeeks.org/hamming-code-in-computer-network/