Universidad del Valle de Guatemala
Facultad de ingeniería

Laboratorio 2 Segunda Parte: Esquemas de detección y corrección de errores

Javier Ramírez Cospin 18099
Priscilla Gonzalez 20689

Guatemala 03 de julio del 2023

Descripción de la práctica

En esta práctica, se tendrá como objetivo principal comparar el tiempo de ejecución y eficiencia de un algoritmo de detección de errores (CRC-32) y un algoritmo de corrección de errores (Hamming (7, 4)), además de la simulación de un modelo de capas para la transmisión y recepción de mensajes por medio de un socket.

Cada mensaje consiste en una trama codificada, seguida de dos puntos y el algoritmo utilizado para codificar dicha trama. La trama es una serie de bits que representan un carácter ascii, además de contener 32 bits agregados al final en caso de haber utilizado CRC-32 como algoritmo de codificación, o bits de paridad calculados para bloques de 4 bits en caso que el algoritmo haya sido Hamming (7, 4).

El emisor se encarga de generar y codificar una determinada cantidad de mensajes, aplicar error a cada bit de la trama codificada y enviar la información al receptor por medio del socket establecido. El algoritmo utilizado para codificar la trama se solicita mediante una entrada de usuario. El error en la trama consiste en intercambiar cada bit de la trama dependiendo de una probabilidad determinada, con la intención de verificar la eficiencia por parte del receptor para verificar y/o corregir dicha trama.

El receptor es quien se encarga de recibir los mensajes transmitidos por el emisor, detectar y/o corregir los errores en cada trama con el algoritmo utilizado para codificar dicho mensaje y reportar el porcentaje de mensajes identificados correctamente. En caso de que el algoritmo utilizado por el emisor para codificar el mensaje sea de detección de errores, el receptor se limita a verificar la integridad del mensaje. En caso que el algoritmo sea el de Hamming (7, 4), el receptor intenta corregir el mensaje y verifica nuevamente su integridad de manera cíclica.

Resultados

1. Prueba de 10k mensajes y error de 1/100
   - Algoritmo de CRC-32

Prueba	Respuestas correctas	Tiempo de ejecución (s)
1	7,796.0	6.500
2	7,866.0	6.553
3	7,919.0	6.552
4	7,765.0	6.468
5	7,856.0	6.547
Promedio	**7,840.4**	**6.524**

Cuadro 1: Resultados para el algoritmo CRC-32 para 10k pruebas y un error de 1/100.

   - Algoritmo de Hamming (7, 4)

Prueba	Respuestas correctas	Tiempo de ejecución (s)
1	10,000.0	8.431
2	10,000.0	8.575
3	10,000.0	8.546
4	10,000.0	8.549
5	10,000.0	8.524
Promedio	**10,000.0**	**8.525**

Cuadro 2: Resultados para el algoritmo Hamming (7, 4) para 10k pruebas y un error de 1/100.

2. Prueba de 10k mensajes y error de 2/100
    - Algoritmo de CRC-32

Prueba	Respuestas correctas	Tiempo de ejecución (s)
1	6,202.0	6.413
2	6,245.0	6.587
3	6,312.0	6.536
4	6,249.0	6.571
5	6,275.0	6.550
Promedio	**6,256.6**	**6.531**

Cuadro 3: Resultados para el algoritmo CRC-32 para 10k pruebas y un error de 2/100.

- Algoritmo de Hamming (7, 4)

Prueba	Respuestas correctas	Tiempo de ejecución (s)
1	10,000.0	8.137
2	10,000.0	8.710
3	10,000.0	8.821
4	10,000.0	8.846
5	10,000.0	8.837
Promedio	**10.000.0**	**8.670**

Cuadro 4: Resultados para el algoritmo Hamming (7, 4) para 10k pruebas y un error de 2/100.

3. Prueba de 10k mensajes y error de 5/100
    - Algoritmo de CRC-32

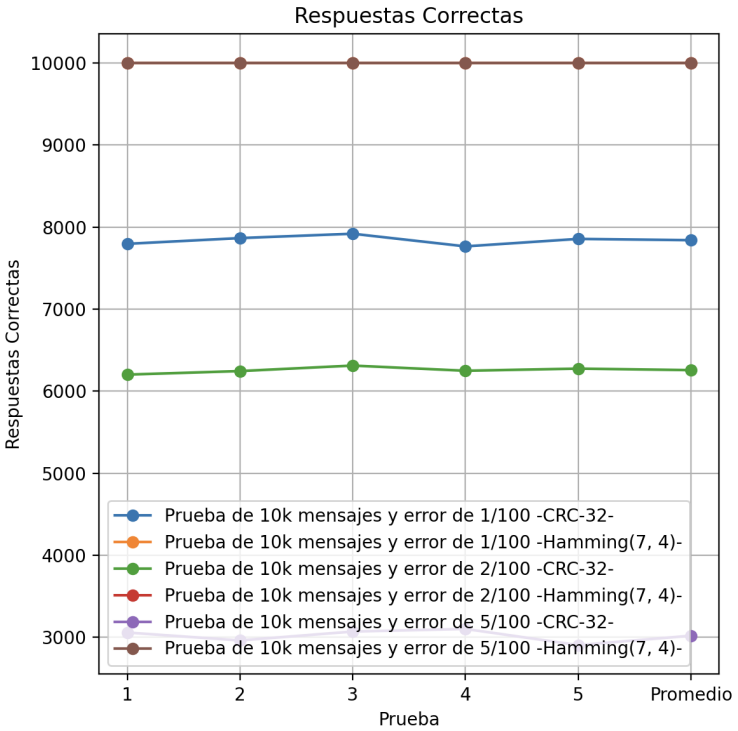
Prueba	Respuestas correctas	Tiempo de ejecución (s)
1	3,058.0	6.561
2	2,962.0	6.481
3	3,070.0	6.487
4	3,102.0	6.486
5	2,905.0	6.522
Promedio	**3,019.4**	**6.507**

Cuadro 5: Resultados para el algoritmo CRC-32 para 10k pruebas y un error de 5/100.

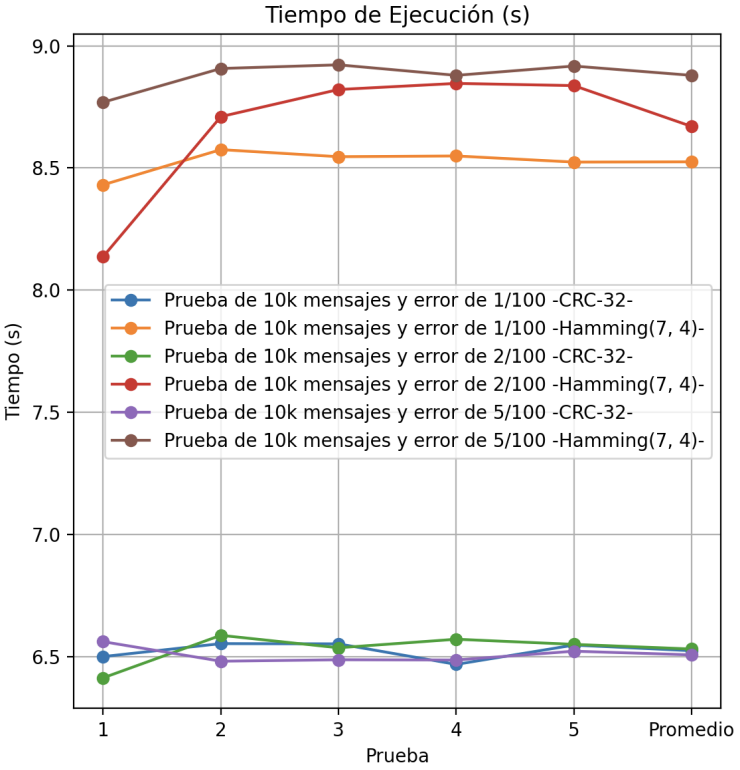
    - Algoritmo de Hamming (7, 4)

Prueba	Respuestas correctas	Tiempo de ejecución (s)
1	10,000.0	8.769
2	10.000.0	8.907
3	10,000.0	8.922
4	10,000.0	8.879
5	10,000.0	8.917
Promedio	**10,000.0**	**8.879**

Cuadro 6: Resultados para el algoritmo Hamming (7, 4) para 10k pruebas y un error de 5/100.



Gráfica 1: Respuestas correctas tanto para CRC-32 como Hamming(7, 4)



Gráfica 2: Tiempo de ejecución tanto para CRC-32 como Hamming(7, 4)

Discusión

Para comprobar la eficiencia de los algoritmos CRC-32 y Hamming (7, 4), se realizaron diversas pruebas con 10k mensajes y una probabilidad de error distinta.

El algoritmo de CRC-32 analiza cada mensaje y verifica si sus últimos bits contienen algún 0 luego de ser decodificados. En caso de haber algún 1, el algoritmo detecta el mensaje como erróneo, lo descarta de los mensajes identificados como correctos y procede a analizar el siguiente mensaje.

El algoritmo de Hamming (7, 4) verifica la integridad de cada mensaje al hacer una comparación de bits de paridad para cada bloque de 7 bits, y en caso de detectar alguna anomalía en alguna posición, procede a intercambiar el bit erróneo con su contraparte y revisa el mensaje corregido nuevamente. Esta implementación permite que los errores puedan identificarse correctamente y que un mensaje no sea descartado automáticamente al ser detectado como erróneo.

Para el primer caso, en el que se realizó una prueba de 10k mensajes y un error de 1/100, se puede observar en los resultados anotados en los cuadros 1 y 2, que el algoritmo de Hamming (7, 4) fue capaz de corregir todos los errores detectados. Por el contrario, el algoritmo de CRC-32 solamente tomaba en cuenta los mensajes que no tenían ningún error al momento de realizar la decodificación. Debido a los cálculos extra que realiza el algoritmo de Hamming, el tiempo promedio es de 2 segundos más que el del algoritmo de CRC-32.

Para las pruebas que fueron hechas con 10k mensajes y un error de probabilidad de 2/100, los resultados fueron anotados en los cuadros 3 y 4, siendo estos de los algoritmos CRC-32 y Hamming (7, 4) respectivamente. Al igual que en el caso anterior, el algoritmo de Hamming fue capaz de corregir todos los mensajes de manera correcta, mientras que el algoritmo de CRC-32 detectó menos mensajes correctos. El tiempo de ejecución promedio para el algoritmo de Hamming subió 100 milisegundos aproximadamente, mientras que el tiempo de ejecución para el algoritmo CRC-32 fue solamente 7 milisegundos mayor al del caso anterior.

Para el último caso en el que se probó enviar 10k mensajes con un porcentaje de error de 5/100 para cada bit. El cuadro 5 muestra los resultados del algoritmo CRC-32 y el cuadro 6 muestra los resultados del algoritmo de Hamming (7, 4). Al igual que en el caso anterior, la precisión del algoritmo de Hamming (7, 4) para detectar y corregir errores fue muy exacta, mientras que su tiempo de ejecución subió 200 milisegundos aproximadamente. Para los resultados del algoritmo CRC-32, sus resultados correctos bajaron considerablemente, mientras que su tiempo de ejecución se mantuvo constante, al igual que en los casos anteriores.

Debido a que el algoritmo de CRC-32, no puede detectar correctamente un error en ciertas tramas (i.e una trama en la que su última operación xor sea igual al mismo polinomio), los resultados obtenidos para dicho algoritmo pueden contener mensajes erróneos. Por esta razón, los resultados obtenidos no deben considerarse como resultado final sin antes aplicar algún otro tipo de algoritmo de detección o corrección de errores para complementar y confirmar que los mensajes no tengan errores.

Finalmente, al observar tanto la gráfica 1 y 2 y los resultados obtenidos, es posible tomar decisiones sobre cuál de los dos métodos de detección y corrección de errores es más adecuado para un escenario en particular. Si la prioridad es la precisión en la detección de errores, Hamming(7, 4) parece ser más confiable en las pruebas realizadas. Mientras que si el tiempo de ejecución es una consideración clave, CRC-32 podría ser preferible en algunas circunstancias.

Comentario grupal

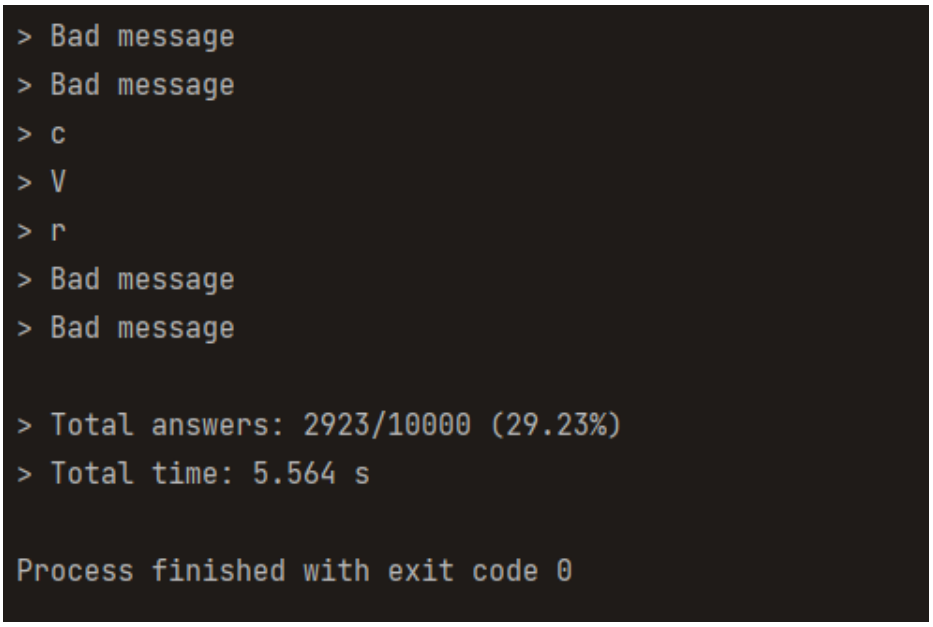
Con esta práctica, se logró implementar un modelo de capas de redes básico, agregando potenciales errores dentro de los mensajes para comprobar y comparar la eficiencia y tiempos de ejecución para los algoritmos CRC-32 y el Algoritmo de Hamming (7, 4). Adicionalmente, se logró implementar transmisión de mensajes entre dos lenguajes de programación distintos, logrando una buena comunicación y tomando en cuenta todos los potenciales errores que pudieran sufrir de parte del emisor o del receptor.

Conclusiones

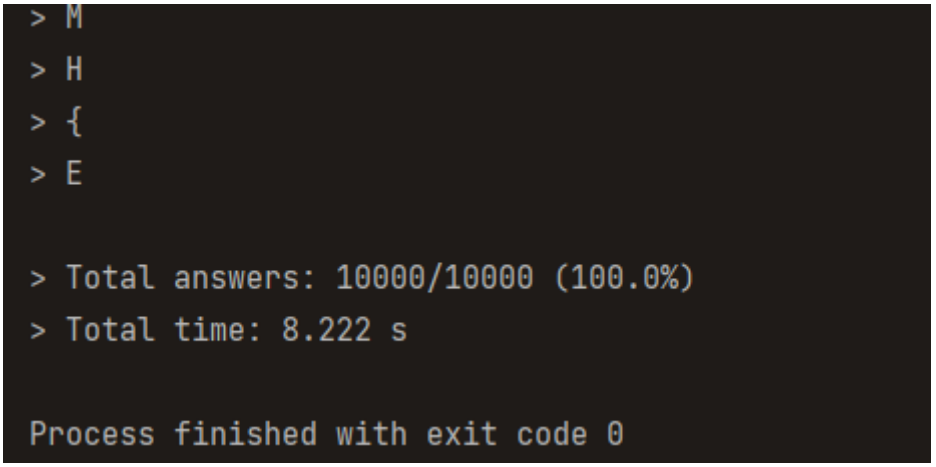
1. El algoritmo de CRC-32, al ser un algoritmo de detección de errores, tiene un tiempo de ejecución constante independiente de la cantidad de errores en un mensaje.
2. El algoritmo de Hamming (7, 4), es capaz de detectar y corregir errores en mensajes erróneos, teniendo un tiempo de ejecución que incrementa proporcionalmente con el número de errores que tengan los mensajes.
3. Debido a los errores que no pueden detectarse con el algoritmo de CRC-32 al aplicarse a ciertas tramas, los resultados de CRC-32 tienen un porcentaje de error más grande que el del algoritmo Hamming (7, 4).
4. Se puede notar que a medida que se introduce más error en las pruebas, la tasa de respuestas correctas disminuye y el tiempo de ejecución tiende a aumentar en ambos métodos.

Anexo

- Ejemplo de ejecución con 10k pruebas y aplicando el algoritmo de CRC-32



- Ejemplo de ejecución con 10k pruebas y aplicando el algoritmo de Hamming (7, 4)



Citas y referencias

- Pandey, H. (2023) *Hamming Code in Computer Network*. GeekforGeeks. https://www.geeksforgeeks.org/hamming-code-in-computer-network/