UNIVERSIDAD DEL VALLE DE GUATEMALA **María Mercedes Retolaza Reyna, 16339**

Facultad de Ingeniería **César Rodas, 16776**

Ingeniería en Ciencia de la Computación y Tecnologías de la Información

REDES

Ing. Jorge Yass

Esquemas de detención y corrección de errores

Laboratorio 2

Antecedentes

Actualmente, las empresas o instituciones tienen errores en todo tipo de comunicación, y constantemente buscamos innovar en los sistemas que se encargan de implementar algoritmos que detectan los errores y nos permitan manejar de forma adecuada las fallas o poder identificarlas previo a que puedan ocurrir. Por lo tanto, a lo largo de la evolución del internet se han desarrollado distintos mecanismos que sirven tanto para la detención como para la corrección de los errores.

Debido a esto, es importante que, como programadores, comprendamos en que momento pueden existir errores en la transferencia de mensajes dentro de una arquitectura de un modelo de capas.

Descripción de la práctica

Durante la práctica de laboratorio se elabora un modelo básico de implementación del intercambio de mensajes entre un cliente y un servidor. Para poder elaborar el modelo debemos de conocer la arquitectura que debemos de implementar. Dicha arquitectura es el modelo de 4 **Capas** propuesto en la práctica de laboratorio.

Dicho modelo de capas consiste en lo siguiente:

- Emisor
    - Aplicación
    - Verificación
    - Ruido
    - Transferencia

- Receptor
    - Transmisión
    - Codificación
    - Verificación
    - Aplicación

Al implementar nuestro algoritmo debemos de cumplir con los siguientes objetivos:

- Identificar ventajas y desventajas de implementación para la detección y corrección de errores
- Comprender los elementos de una arquitectura de capas

Para poder llevar a cabo la implementación de nuestro laboratorio se establece un desarrollo por **3 fases** la primera comprende en el desarrollo de la arquitectura para envío y recepción de mensajes, la segunda comprende en la implementación de los algoritmos que vamos a analizar: Corrección y detección de errores y la tercera fase comprende en el análisis de pruebas de algoritmos implementados.

A continuación, se detallan los resultados que fueron obtenidos durante la práctica de laboratorio al momento de elaborar un chat simple con los siguientes algoritmos de detención y corrección de errores: **CRC-32 y Hamming**

Resultados

Para poder determinar que mecánica de control debemos de implementar en nuestros modelos de arquitectura de redes debemos de conocer los tiempos de implementación de c/u de los tipos de algorimos que se eligieron y se grafica su desempeño.

UNIVERSIDAD DEL VALLE DE GUATEMALA **María Mercedes Retolaza Reyna, 16339**

Facultad de Ingeniería **César Rodas, 16776**

Ingeniería en Ciencia de la Computación y Tecnologías de la Información

REDES

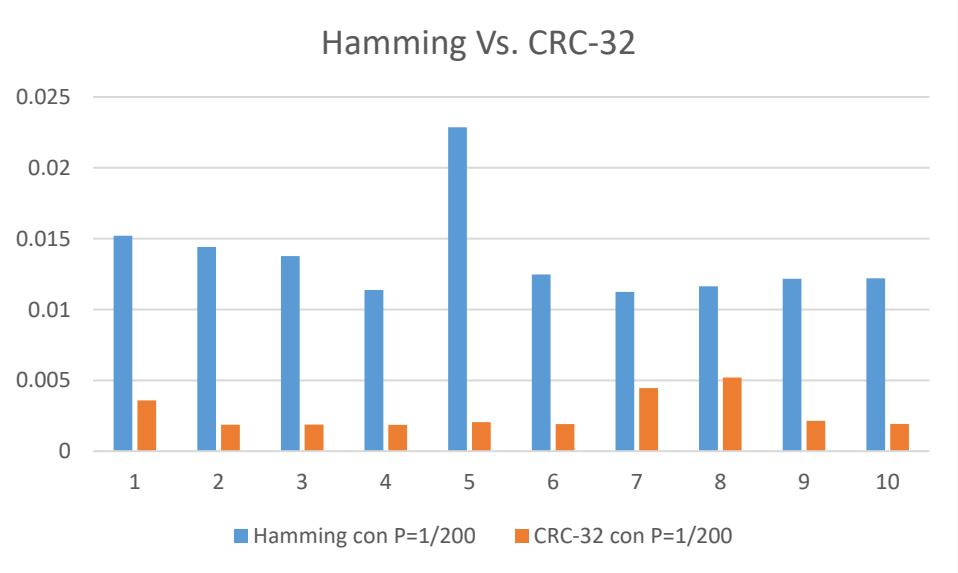
Ing. Jorge Yass

Ejecución 1: Medición de tiempos de ejecución de algortimos

Tabla de resultados #1 con P = 1/200

Hamming con P=1/200	CRC-32 con P=1/200
0.015204906	0.003582954
0.014416218	0.001863718
0.013767481	0.001877785
0.011372805	0.001859903
0.022862196	0.0020504
0.012471437	0.001917839
0.011242151	0.004456282
0.011631966	0.005197763
0.012166739	0.002144337
0.012199879	0.001925468

Gráfica de resultados #1 con P = 1/200



UNIVERSIDAD DEL VALLE DE GUATEMALA **María Mercedes Retolaza Reyna, 16339**

Facultad de Ingeniería **César Rodas, 16776**

Ingeniería en Ciencia de la Computación y Tecnologías de la Información

REDES

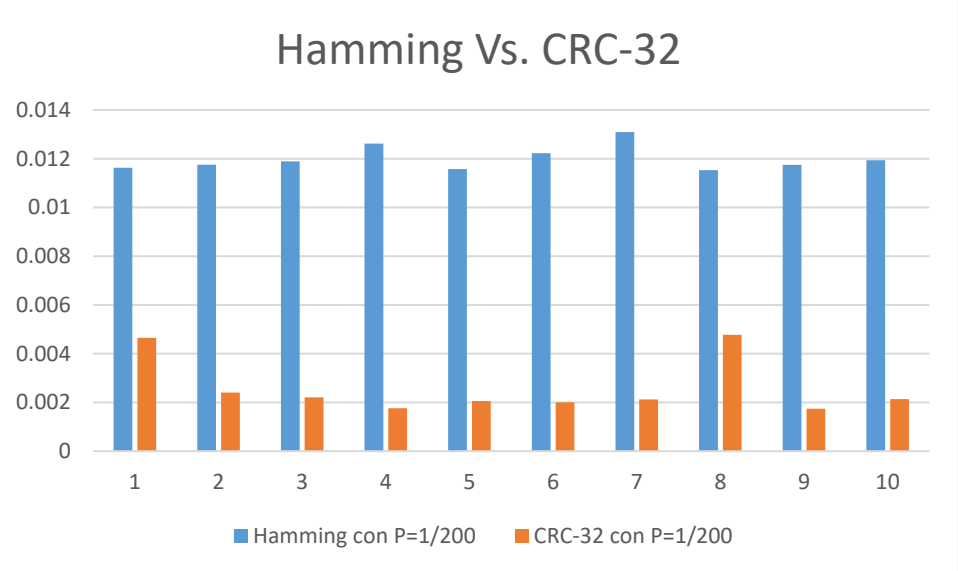
Ing. Jorge Yass

Ejecución 2: Medición de tiempos de ejecución de algortimos

Tabla de resultados #2 con P = 1/200

Hamming con P=1/200	CRC-32 con P=1/200
0.011630774	0.004653454
0.01175189	0.002405167
0.011890411	0.002207041
0.012623787	0.001759529
0.011577129	0.002056837
0.012228012	0.0020082
0.013092995	0.002121449
0.011533499	0.004773617
0.011750937	0.001745701
0.01193881	0.0021348

Gráfica de resultados #2 con P = 1/200



UNIVERSIDAD DEL VALLE DE GUATEMALA **María Mercedes Retolaza Reyna, 16339**

Facultad de Ingeniería **César Rodas, 16776**

Ingeniería en Ciencia de la Computación y Tecnologías de la Información

REDES

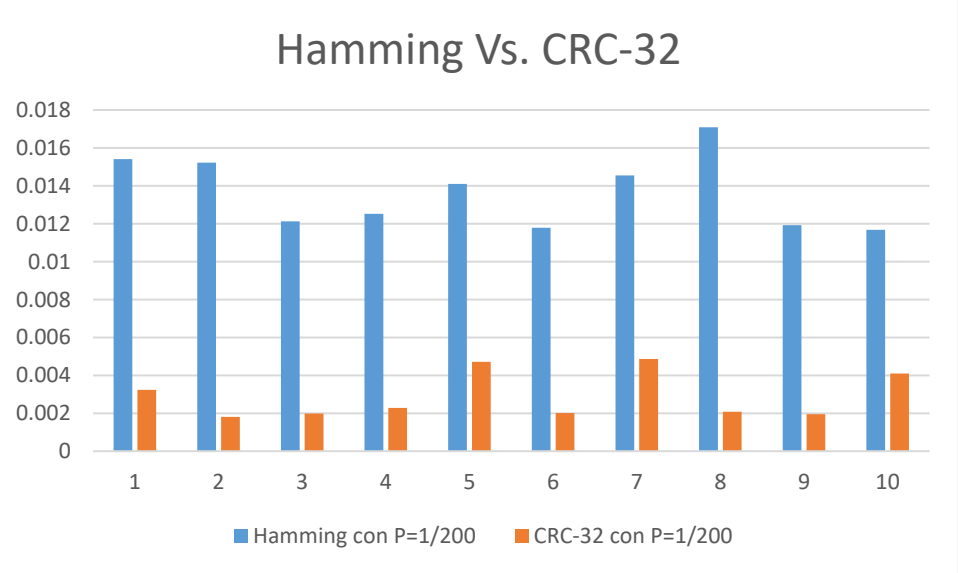
Ing. Jorge Yass

Ejecución 3: Medición de tiempos de ejecución de algortimos

Tabla de resultados #3 con P = 1/200

Hamming con P=1/200	CRC-32 con P=1/200
0.015411377	0.003235817
0.015218258	0.001806021
0.012122869	0.001984358
0.012519598	0.002286911
0.014098167	0.004716873
0.011786699	0.002009392
0.01454854	0.004864216
0.017086983	0.002078056
0.011923313	0.001951933
0.011676073	0.004100323

Gráfica de resultados #3 con P = 1/200



UNIVERSIDAD DEL VALLE DE GUATEMALA **María Mercedes Retolaza Reyna, 16339**

Facultad de Ingeniería **César Rodas, 16776**

Ingeniería en Ciencia de la Computación y Tecnologías de la Información

REDES

Ing. Jorge Yass

Discusión de resultados

Al momento de elaborar el sistema de intercambio de mensajes simple con una arquitectura de 4 capas se realizó la implementación de dos tipos de algoritmo de detención de errores. Los algoritmos que se implementaron son los siguientes:

- Hamming
- CRC- 32

Nuestro objetivo es establecer la importancia de proteger los datos, tanto de errores, mediante el control, como de acciones no autorizadas mediante la seguridad en redes.

Al momento de llevar a cabo una transferencia de mensajes, debemos de conocer el proceso de protección de datos y que prioridad tiene al momento de establecer una comunicación. Entonces debemos de saber que, los datos, cuya utilidad radica en su integridad y en su confidencialidad, están sujetos a dos tipos de amenazas: Errores y Acciones no autorizadas.

Durante la práctica de laboratorio nos enfocamos en **detección de errores** que esta práctica consiste en monitorear la información recibida a través de técnicas implementadas en el verificador, de nuestra arquitectura implementada.

Al momento de evaluar nuestros resultados, realizamos **3 escenarios de pruebas** en donde efectuábamos 10 corridas de nuestro sistema con diferentes cadenas de mensaje y evaluábamos el comportamiento y rendimiento de nuestros algoritmos, según nuestras tablas de resultados tenemos que el **rendimiento promedio** que muestran las gráficas 1, 2 y 3 podemos observar que en promedio el algoritmo CRC-32 tiene un mejor rendimiento en comparación al algoritmo de Hamming. Ya que los tiempos **no exceden** los 0.006 segundos.

Al evaluar los dos algoritmos, tenemos que los códigos de Hamming se pueden utilizar tanto para detectar como para corregir errores, mientras que el CRC 32 solo sirven para detectarlos, pero este se utiliza en la comunicación; mientras que Hamming se utiliza para detectar errores en discos de memoria. Entonces, el algoritmo que tiene una tasa de mayor detección de errores es CRC-32

porque, Hamming se utiliza en caso que se detecte y corrija una **longitud fija de datos** mientras que CRC funciona para **cualquier longitud de datos.**

Es importante conocer que ambos algoritmos agregan un valor de verificación basado en alguna operación aritmética de los bits en el mensaje que se transmite. Para Hamming, pueden ser bits de paridad pares o impares agregados al final de un mensaje y para CRC, es el resto de una división polinomial de su contenido.

Para poder determinar en que momento debemos de utilizar un tipo en especifico de algoritmo debemos de conocer nuestro entorno, que funcionalidades debe de cumplir el sistema y sobre todo analizar el entorno de transferencia de información (contenido) que tendremos en nuestro cliente - servidor o comunicación entre servidores. Entonces el CRC se concibe como el resto de una división polinomial. Es eficaz para detectar errores, cuando el resto calculado no coincide. Dependiendo del tamaño de CRC, puede detectar ráfagas de errores (10 bits puestos a cero, por ejemplo), lo cual es excelente para verificar las comunicaciones.

Mientras que, Hamming son códigos de detección y corrección. Al agregar los bits del código de Hamming, se asegura cierta distancia (como el número de bits diferentes) entre los códigos válidos. Por ejemplo, con una distancia de 3 bits, puede corregir cualquier error de 1 bit O detectar cualquier error de 2 bits.

Por lo que, para poder determinar en que momento es mejor utilizar un algoritmo de detención vs uno de corrección es evaluar nuestra comunicación, arquitectura y entorno del sistema a implementar.

Conclusiones

1. Al momento de realizar la evaluación de los algoritmos se puede concluir que la funcionalidad del algoritmo de Hamming se debe de implementar en caso que se detecte y corrija una longitud fija de datos mientras que CRC funciona para cualquier longitud de datos.

2. CRC-32 tiene una mayor detección de tasa de errores debido a que puede recibir una cadena de Longitud desconocida
3. El algoritmo CRC-32 tiene un mejor rendimiento a nivel de funcionamiento. Con un promedio de tiempo de 0.006 segundos
4. Al momento de reducir ambos algoritmos a un solo bit de verificación, los códigos de Hamming y CRC son idénticos (x + 1 polinomio) a la paridad.

Fuentes consultadas

Difference between CRC and Hamming Code. (2016, 28 febrero). Computer Science Stack Exchange. https://cs.stackexchange.com/questions/53719/difference-between-crc-and-hamming-code

What is the hamming distance, and how do I determine it for a CRC scheme? (2010, 27 septiembre). Stack Overflow. https://stackoverflow.com/questions/3800788/what-is-the-hamming-distance-and-how-do-i-determine-it-for-a-crc-scheme