UNIVERSIDAD DEL VALLE DE GUATEMALA

Redes

Sección 10

Tarea 1

Maria Ramirez #21342

Gustavo González #21438

25 de julio de 2024

Link al repositorio
https://github.com/mariaRam2003/Laboratorio2-Redes

Escenarios de Prueba

Corrección de Errores
(sin errores)
Mensajes:

1. 0100101
2. 101
3. 00101

- Emisor 1,2, y 3:

```
Mensaje no.  1
    original:  [0, 1, 0, 0, 1, 0, 1]
    codificado:  [1, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1]
Mensaje no.  2
    original:  [1, 0, 1]
    codificado:  [1, 0, 1, 1, 0, 1]
Mensaje no.  3
    original:  [0, 0, 1, 0, 1]
    codificado:  [1, 1, 0, 1, 0, 1, 0, 1, 1]
```

- Receptor 1,2, y 3

```
~~~~~~~~~~~~ TEST CASE 1~~~~~~~~~~~~
Mensaje obtenido:
[1, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1]
No error
~~~~~~~~~~~~ TEST CASE 2~~~~~~~~~~~~
Mensaje obtenido:
[1, 0, 1, 1, 0, 1]
No error
~~~~~~~~~~~~ TEST CASE 3~~~~~~~~~~~~
Mensaje obtenido:
[1, 1, 0, 1, 0, 1, 0, 1, 1]
No error
```

(un error)
Mensajes:

1. 01011
2. 1100
3. 011001

- Emisor 4,5, y 6:
- Receptor 4,5, y 6:

```
Mensaje no.  4
    original:  [0, 1, 0, 1, 1]
    codificado:  [1, 1, 0, 0, 1, 0, 1, 1, 1]
Mensaje no.  5
    original:  [1, 1, 0, 0]
    codificado:  [0, 1, 1, 1, 1, 0, 0]
Mensaje no.  6
    original:  [0, 1, 1, 0, 0, 1]
    codificado:  [1, 0, 0, 0, 1, 1, 0, 1, 0, 1]
```

```
~~~~~~~~~~~~ TEST CASE 4~~~~~~~~~~~~
Mensaje obtenido:
[1, 0, 0, 0, 1, 0, 1, 1, 1]
El error esta en el indice: 2

La version corregida del mensaje es: 110010111
~~~~~~~~~~~~ TEST CASE 5~~~~~~~~~~~~
Mensaje obtenido:
[0, 1, 0, 1, 1, 0, 0]
El error esta en el indice: 3

La version corregida del mensaje es: 0111100
~~~~~~~~~~~~ TEST CASE 6~~~~~~~~~~~~
Mensaje obtenido:
[1, 0, 0, 0, 1, 1, 0, 1, 0, 0]
El error esta en el indice: 10
```

(dos + errores)
Mensajes:

1. 00101
2. 110
3. 0011010

- Emisor 7,8, y 9:

```
Mensaje no.  7
    original:  [0, 0, 1, 0, 1]
    codificado:  [1, 1, 0, 1, 0, 1, 0, 1, 1]
Mensaje no.  8
    original:  [1, 1, 0]
    codificado:  [0, 1, 1, 1, 1, 0]
Mensaje no.  9
    original:  [0, 0, 1, 1, 0, 1, 0]
    codificado:  [1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 0]
```

- Receptor 7,8, y 9:

```
~~~~~~~~~~~ TEST CASE 7~~~~~~~~~~~
Mensaje obtenido:
[0, 1, 0, 1, 0, 1, 0, 1, 0]
Multiple errors
~~~~~~~~~~~ TEST CASE 8~~~~~~~~~~~
Mensaje obtenido:
[1, 1, 1, 1, 1, 1]
Multiple errors
~~~~~~~~~~~ TEST CASE 9~~~~~~~~~~~
Mensaje obtenido:
[1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1]
Multiple errors
```

Detección de Errores

(sin errores)

Mensajes:

4. 0100101
5. 101
6. 00101

Respuestas del emisor y receptor:

- Emisor 1:

```
PS C:\Users\maria\Documents\UVG\Semestre 8\Redes\Laboratorio2-Re
des\DeteccionDeErrores> ./emisor
Ingrese un mensaje en binario: 010101
Mensaje original: 010101
Mensaje con checksum: 010101000101010001010100
```

- Receptor 1:

```
PS C:\Users\maria\Documents\UVG\Semestre 8\Redes\Laboratorio2-
Redes\DeteccionDeErrores> python Receptor.py
Ingrese el mensaje original en binario (sin checksum): 010101
Ingrese el mensaje en binario con el checksum: 010101000101010
001010100
No se detectaron errores. Mensaje original: 010101
```

- Emisor 2:

```
PS C:\Users\maria\Documents\UVG\Semestre 8\Redes\Laboratorio2-Re
des\DeteccionDeErrores> ./emisor
Ingrese un mensaje en binario: 101
Mensaje original: 101
Mensaje con checksum: 10100000101000010100000
```

- Receptor 2:

```
Redes\DeteccionDeErrores> python Receptor.py
Ingrese el mensaje original en binario (sin checksum): 101
Ingrese el mensaje en binario con el checksum: 101000001010000
010100000
No se detectaron errores. Mensaje original: 101
```

- Emisor 3:

```
PS C:\Users\maria\Documents\UVG\Semestre 8\Redes\Laboratorio2-Re
des\DeteccionDeErrores> ./emisor
Ingrese un mensaje en binario: 00101
Mensaje original: 00101
Mensaje con checksum: 0010100000101000000101000
```

- Receptor 3:

```
Redes\DeteccionDeErrores> python Receptor.py
Ingrese el mensaje original en binario (sin checksum): 00101
Ingrese el mensaje en binario con el checksum: 001010000010100
000101000
No se detectaron errores. Mensaje original: 00101
```

(un error)
Mensajes:

4. 01011
5. 1100
6. 011001

- Emisor 1:

```
PS C:\Users\maria\Documents\UVG\Semestre 8\Redes\Laboratorio2-Re
des\DeteccionDeErrores> ./emisor
Ingrese un mensaje en binario: 01011
Mensaje original: 01011
Mensaje con checksum: 0101100001011000001011000
```

- Receptor 1:

```
PS C:\Users\maria\Documents\UVG\Semestre 8\Redes\Laboratorio2-
Redes\DeteccionDeErrores> python Receptor.py
Ingrese el mensaje original en binario (sin checksum): 01011
Ingrese el mensaje en binario con el checksum: 010110000101100
001011001
Se detectaron errores en el mensaje. Mensaje descartado.
```

- Receptor 2:

```
PS C:\Users\maria\Documents\UVG\Semestre 8\Redes\Laboratorio2-Re
des\DeteccionDeErrores> ./emisor
Ingrese un mensaje en binario: 1100
Mensaje original: 1100
Mensaje con checksum: 110000001100000011000000
```

- Emisor 2:

```
PS C:\Users\maria\Documents\UVG\Semestre 8\Redes\Laboratorio2-
Redes\DeteccionDeErrores> python Receptor.py
Ingrese el mensaje original en binario (sin checksum): 1100
Ingrese el mensaje en binario con el checksum: 110000001100000
011100000
Se detectaron errores en el mensaje. Mensaje descartado.
```

- Receptor 3:

```
PS C:\Users\maria\Documents\UVG\Semestre 8\Redes\Laboratorio2-Re
des\DeteccionDeErrores> ./emisor
Ingrese un mensaje en binario: 011001
Mensaje original: 011001
Mensaje con checksum: 011001000110010001100100
```

- Emisor 3:

```
PS C:\Users\maria\Documents\UVG\Semestre 8\Redes\Laboratorio2-
Redes\DeteccionDeErrores> python Receptor.py
Ingrese el mensaje original en binario (sin checksum): 011001
Ingrese el mensaje en binario con el checksum: 011001000110010
001100101
Se detectaron errores en el mensaje. Mensaje descartado.
```

(dos + errores)
Mensajes:

4. 00101
5. 110
6. 0011010

- Emisor 1:

```
PS C:\Users\maria\Documents\UVG\Semestre 8\Redes\Laboratorio2-Re
des\DeteccionDeErrores> ./emisor
Ingrese un mensaje en binario: 00101
Mensaje original: 00101
Mensaje con checksum: 0010100000101000000101000
```

- Receptor 1:

```
PS C:\Users\maria\Documents\UVG\Semestre 8\Redes\Laboratorio2-
Redes\DeteccionDeErrores> python Receptor.py
Ingrese el mensaje original en binario (sin checksum): 00101
Ingrese el mensaje en binario con el checksum: 001010000010100
000101101
Se detectaron errores en el mensaje. Mensaje descartado.
```

- Emisor 2:

```
PS C:\Users\maria\Documents\UVG\Semestre 8\Redes\Laboratorio2-Re
des\DeteccionDeErrores> ./emisor
Ingrese un mensaje en binario: 110
Mensaje original: 110
Mensaje con checksum: 11000000110000011000000
```

- Receptor 2:

```
PS C:\Users\maria\Documents\UVG\Semestre 8\Redes\Laboratorio2-
Redes\DeteccionDeErrores> python Receptor.py
Ingrese el mensaje original en binario (sin checksum): 110
Ingrese el mensaje en binario con el checksum: 110000001100000
011000110
Se detectaron errores en el mensaje. Mensaje descartado.
```

- Emisor 3:

```
PS C:\Users\maria\Documents\UVG\Semestre 8\Redes\Laboratorio2-Re
des\DeteccionDeErrores> ./emisor
Ingrese un mensaje en binario: 0011010
Mensaje original: 0011010
Mensaje con checksum: 0011010000110100000110100
```

- Receptor 3:

```
PS C:\Users\maria\Documents\UVG\Semestre 8\Redes\Laboratorio2-
Redes\DeteccionDeErrores> python Receptor.py
Ingrese el mensaje original en binario (sin checksum): 0011010

Ingrese el mensaje en binario con el checksum: 001101001011010
000110110
Se detectaron errores en el mensaje. Mensaje descartado.
```

Preguntas

Detección de Errores

¿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.

Creo que si es posible manipular los bits de tal forma que el algoritmo de Fletcher checksum no sea capaz de detectar el error. Esto puede ser debido a la naturaleza de los algoritmos de checksum, que no son perfectos y pueden tener colisiones. Por ejemplo, diferentes mensajes pueden tener el mismo checksum, lo que puede llevar a falsos negativos (un error en el mensaje que no se detecta).

Implementación:
Mensaje: 1010101

Emisor:

```
PS C:\Users\maria\Documents\UVG\Semestre 8\Redes\Laboratorio2-Re
des\DeteccionDeErrores> ./emisor
Ingrese un mensaje en binario: 1010101
Mensaje original: 1010101
Mensaje con checksum: 10101010101010101010101010
```

Receptor:

```
PS C:\Users\maria\Documents\UVG\Semestre 8\Redes\Laboratorio2-
Redes\DeteccionDeErrores> python Receptor.py
Ingrese el mensaje original en binario (sin checksum): 1010101

Ingrese el mensaje en binario con el checksum: 101010101010101
010101010
No se detectaron errores. Mensaje original: 1010101
```

¿Qué ventajas y desventajas posee cada algoritmo con respecto a los otros dos?

- Ventajas:
  - Mejor capacidad de detección de errores: Puede detectar múltiples errores en el mensaje y es más robusto que el bit de paridad.
  - Uso extendido: Es utilizado en varios protocolos de red y aplicaciones debido a su buen balance entre complejidad y capacidad de detección.
- Desventajas:
  - Redundancia mayor: Requiere dos sumas adicionales (16 bits en total), lo que agrega más overhead al mensaje en comparación con el bit de paridad.
  - Mayor complejidad: La implementación y el cálculo son más complejos que los del bit de paridad. La complejidad computacional es O(n).
  - No proporciona corrección de errores: Solo detecta errores pero no puede corregirlos ni indicar dónde están los errores.