PUCE Virtual

CLASE

6

Teoría de la Información y la Comunicación

Transmisión de datos y detección de errores

QS WORLD UNIVERSITY RANKINGS

Educación de calidad

INTRODUCCIÓN DE LA CLASE

GRADO / POSGRADO / TECNOLOGÍAS

Estudia a tu tiempo
sin interrupciones

EDUCACIÓN EN LÍNEA DE CALIDAD

En la evolución de los sistemas de comunicación digital, comprender cómo se representa y comprime la información es solo una parte del desafío. En escenarios reales, los datos no se transmiten en condiciones ideales: existen interferencias, pérdidas y alteraciones que pueden comprometer la integridad de la información. Desde la perspectiva de la ciberseguridad, este problema adquiere mayor relevancia, ya que la corrupción de datos no solo afecta la eficiencia, sino también la confiabilidad y seguridad de los sistemas.

Durante esta sexta semana, el enfoque se desplaza hacia el comportamiento de la información cuando es transmitida a través de canales con imperfecciones. Se analizará cómo el ruido puede modificar los datos y cómo los sistemas modernos incorporan mecanismos de detección de errores para garantizar que la información recibida sea confiable. Este análisis se vincula directamente con los principios estudiados previamente, especialmente con la redundancia, que ahora se interpreta no solo como un recurso para comprimir, sino también como un elemento clave para proteger la información.

Desde una visión aplicada a la seguridad informática, los métodos de detección de errores permiten identificar manipulaciones accidentales o intencionales en los datos, lo cual es fundamental en procesos como autenticación, integridad de archivos y transmisión segura en redes. La correcta implementación de estos mecanismos contribuye a evitar vulnerabilidades que podrían ser explotadas en ataques cibernéticos. Finalmente, estos contenidos fortalecen el desarrollo del Reto 2, ya que permiten al estudiante comprender el equilibrio entre compresión y redundancia. Mientras la compresión busca eliminar redundancia, la detección de errores requiere introducirla de forma controlada, lo que implica tomar decisiones estratégicas para optimizar la eficiencia sin comprometer la integridad de la información.

6. TRANSMISIÓN DE DATOS Y DETECCIÓN DE ERRORES

En los sistemas modernos de comunicación digital, la transmisión de datos no ocurre en condiciones ideales. A diferencia de los modelos teóricos estudiados previamente —donde la información se analiza en términos de entropía, redundancia y compresión—, en la práctica los datos deben atravesar canales que introducen imperfecciones. Estas alteraciones pueden deberse a interferencias físicas, limitaciones del medio o incluso acciones maliciosas, lo que convierte la transmisión de datos en un proceso vulnerable desde la perspectiva de la ciberseguridad.

En este contexto, garantizar que la información llegue de forma íntegra al receptor se vuelve un requisito fundamental. La integridad de los datos no solo afecta la calidad del servicio, sino que también impacta directamente en la confiabilidad de sistemas críticos, como redes financieras, sistemas de autenticación o plataformas de almacenamiento seguro. Por ello, los mecanismos de detección de errores constituyen una herramienta esencial para validar la información recibida.

Es importante destacar que, mientras en la compresión de datos el objetivo es eliminar redundancia para optimizar el uso de recursos, en la detección de errores ocurre lo contrario: se introduce redundancia de forma intencional para poder verificar la integridad de los datos. Este contraste representa un punto clave en el análisis del RDA2, ya que el estudiante debe comprender cómo gestionar la redundancia dependiendo del objetivo del sistema.

A lo largo de este tema se estudiarán los principios de la transmisión de datos en presencia de ruido, así como los fundamentos de los principales mecanismos de detección de errores, estableciendo una conexión directa con la eficiencia, seguridad y confiabilidad en los sistemas de comunicación.

6.1 Transmisión de Datos y Canal con Ruido

La transmisión de datos implica el envío de información desde una fuente hacia un destino a través de un medio físico o lógico. En este proceso, el canal de comunicación juega un papel determinante, ya que puede introducir alteraciones en la señal original debido a interferencias externas, limitaciones físicas o ataques maliciosos (Alencar, 2015).

Todo canal real introduce perturbaciones que pueden modificar los datos transmitidos.

En condiciones reales, los canales de comunicación no son perfectos. La presencia de perturbaciones externas introduce errores en los datos transmitidos, fenómeno conocido como ruido (Csiszár & Körner, 2011).

Un canal con ruido se caracteriza por la probabilidad de que los símbolos transmitidos sean alterados antes de llegar al receptor.
Tipos de ruido

- Ruido térmico: Generado por el movimiento de electrones
- Interferencia electromagnética: Proveniente de otros dispositivos
- Ruido impulsivo: Alteraciones repentinas de alta intensidad
- Ruido intencional: Asociado a ataques en ciberseguridad

El ruido introduce incertidumbre y puede alterar la información transmitida.

Impacto del ruido en la información
Desde el punto de vista de la teoría de la información:

- Aumenta la incertidumbre en la recepción
- Disminuye la confiabilidad del sistema
- Incrementa la probabilidad de error

Esto implica que los sistemas deben incorporar mecanismos para mitigar estos efectos.

Un **canal con ruido** se caracteriza por la presencia de errores aleatorios que afectan los símbolos transmitidos. Desde el punto de vista de la teoría de la información, estos errores se modelan probabilísticamente, en el que cada símbolo transmitido tiene cierta probabilidad de ser alterado; permitiendo analizar su impacto en la confiabilidad del sistema (Csiszár & Körner, 2011).

En términos de ciberseguridad, el ruido no siempre es accidental. En algunos casos, puede representar intentos de manipulación de datos o ataques de tipo “injection”, donde se alteran los mensajes durante la transmisión.

Modelo de transmisión
Un sistema de comunicación incluye:

- Fuente de información
- Codificador
- Canal (con ruido)
- Decodificador
- Receptor

El canal introduce una probabilidad de error que puede representarse como:

- Error de bit (bit flip)
- Pérdida de datos
- Alteración de secuencias

La probabilidad de error define la confiabilidad del sistema de transmisión.

6.2 Fundamentos de Códigos Detectores de Errores

Un error ocurre cuando el dato recibido es diferente al dato enviado. En sistemas digitales, estos errores suelen manifestarse como cambios en los bits (0 ↔ 1).
Tipos de errores

- Error de un solo bit: Se altera un único bit
- Errores múltiples: Se afectan varios bits
- Errores en ráfaga: Secuencia continua de bits alterados

Los errores pueden afectar desde un bit hasta bloques completos de datos.

Probabilidad de error
La probabilidad de error depende de:

- Calidad del canal

- Nivel de ruido
- Tipo de modulación

Este parámetro es fundamental para diseñar sistemas confiables.

Detección de errores

La detección de errores consiste en identificar si los datos han sido alterados durante la transmisión. Para lograrlo, se utilizan técnicas que añaden información adicional a los datos originales (Gorzalka & Deloumeaux, 2012).

Principio básico

1. El emisor añade redundancia
2. El receptor verifica esa redundancia
3. Se detectan inconsistencias

La detección de errores permite verificar la integridad de los datos transmitidos.

Importancia en ciberseguridad

La detección de errores permite:

- Identificar corrupción de datos
- Detectar alteraciones maliciosas
- Garantizar confiabilidad en la comunicación

Esto es clave en protocolos de red y sistemas de autenticación.

Redundancia y su papel en la detección

Mientras que en la compresión:

- Se elimina redundancia

En detección de errores:

- Se añade redundancia

La redundancia puede ser eliminada o añadida dependiendo del objetivo del sistema.

Tipos de redundancia

- Redundancia espacial: Bits adicionales
- Redundancia temporal: Repetición de datos
- Redundancia estructural: Patrones verificables

La clave está en utilizar la redundancia de forma estratégica.

Para contrarrestar los efectos del ruido, se emplean códigos que permiten identificar si la información ha sido alterada durante la transmisión. Estos códigos se basan en la introducción controlada de redundancia en los datos (Gorzalka & Deloumeaux, 2012).

Aquí aparece un concepto clave: Redundancia controlada

A diferencia de la redundancia eliminada en la compresión, esta redundancia se añade intencionalmente para verificar la integridad de la información.

Principio de funcionamiento

El emisor:

- Añade bits adicionales (redundancia)
- Transmite el mensaje extendido

El receptor:

- Recalcula la redundancia
- Compara con la recibida
- Detecta inconsistencias

Los códigos detectores no corrigen errores, solo indican su presencia.

Relación con la entropía

Desde la teoría de la información:

- La redundancia reduce la incertidumbre
- Permite identificar desviaciones del patrón esperado

Esto conecta directamente con el RDA2, ya que el estudiante debe comprender cómo manipular la redundancia para diferentes objetivos: compresión vs. confiabilidad.

6.3 Bit de Paridad

El bit de paridad es uno de los métodos más simples para detectar errores en la transmisión de datos. Consiste en añadir un bit adicional a un conjunto de bits con el objetivo de verificar si el número de unos es par o impar (Alencar, 2015).

Enlaces complementarios

Título: Bit de paridad

Descripción: Muestra cómo funciona la paridad par e impar mediante ejemplos binarios. Es útil para explicar la detección básica de errores.

Enlace: https://es.wikiital.com/wiki/Bit_di_parità?utm_source=chatgpt.com

Tipos de paridad

- Paridad par
- Paridad impar

Paridad par

En la paridad par, el objetivo es que el número total de unos (1) sea par.

Ejemplo

Datos originales: 1011

Contamos los unos: Hay 3 unos.

El número 3 es impar, pero en la paridad par necesitamos que el total sea par.

Entonces:

- Se añade un 1 extra para convertir 3 en 4.

Resultado: 10111 hay 4 unos, y 4 es un número par. Este proceso se describe en la Figura 1.

En la paridad par, el total de unos debe terminar siendo un número par.

Paridad impar

En la paridad impar, el objetivo es que el número total de unos sea impar.

Usamos los mismos datos: 1011

Cantidad de unos: 3
Como 3 ya es impar, NO hace falta cambiar nada.
Entonces:

- Se añade un 0.

Resultado: 10110 Siguen existiendo 3 unos, que continúa siendo impar. Este proceso se evidencia en la Figura 2.

En la paridad impar, el total de unos debe terminar siendo un número impar.

¿Cómo detecta errores?
Supongamos que enviamos: 10111 (Paridad par)
Durante la transmisión ocurre un error y llega: 10011
Ahora contamos los unos:
1 0 0 1 1
Hay solo 3 unos.
Pero el receptor esperaba un número par de unos.
Entonces: El sistema detecta que ocurrió un error.

El bit de paridad permite detectar errores simples en la transmisión.

Figura 1
Funcionamiento de la paridad par en detección de errores

Nota. La figura representa el proceso de cálculo de paridad par, donde se añade un bit adicional para que el número total de unos sea par. Elaboración propia con base en Forouzan, B. A. (2017). *Transmisión de datos y redes de comunicaciones* (5.ª ed.). McGraw-Hill Education. Miguel Ortiz Navarrete

Figura 2

Detección de errores mediante paridad impar

Nota. La figura muestra el funcionamiento de la paridad impar para detectar errores en la transmisión de datos. Se añade un bit de paridad con el objetivo de que el número total de unos sea impar. Elaboración propia con base en Forouzan, B. A. (2017). *Transmisión de datos y redes de comunicaciones* (5.ª ed.). McGraw-Hill Education. Miguel Ortiz Navarrete

Limitaciones

- Detecta solo errores de un bit
- No identifica la posición del error
- Puede fallar si ocurren múltiples errores

Aplicación en ciberseguridad

El bit de paridad se utiliza en:

- Memorias RAM (detección básica de fallos)
- Sistemas embebidos
- Comunicaciones simples

Aunque es un método limitado, representa la base conceptual de sistemas más avanzados.

6.4 Checksum y CRC

Checksum

El checksum es un mecanismo de detección de errores basado en operaciones aritméticas simples. Su funcionamiento consiste en calcular una suma utilizando los datos del mensaje y enviar ese resultado junto con la información original (Forouzan, 2017).

El receptor realiza exactamente el mismo cálculo:

- Si el resultado coincide → el mensaje es válido.
- Si existe diferencia → se detecta un error.

Si hay diferencia → existe error

El checksum detecta errores mediante operaciones aritméticas sobre los datos.

Funcionamiento del Checksum

El proceso general incluye:

1. Dividir los datos en bloques
2. Sumar los bloques
3. Generar un valor checksum
4. Enviar datos + checksum
5. Recalcular en la recepción

Ventajas

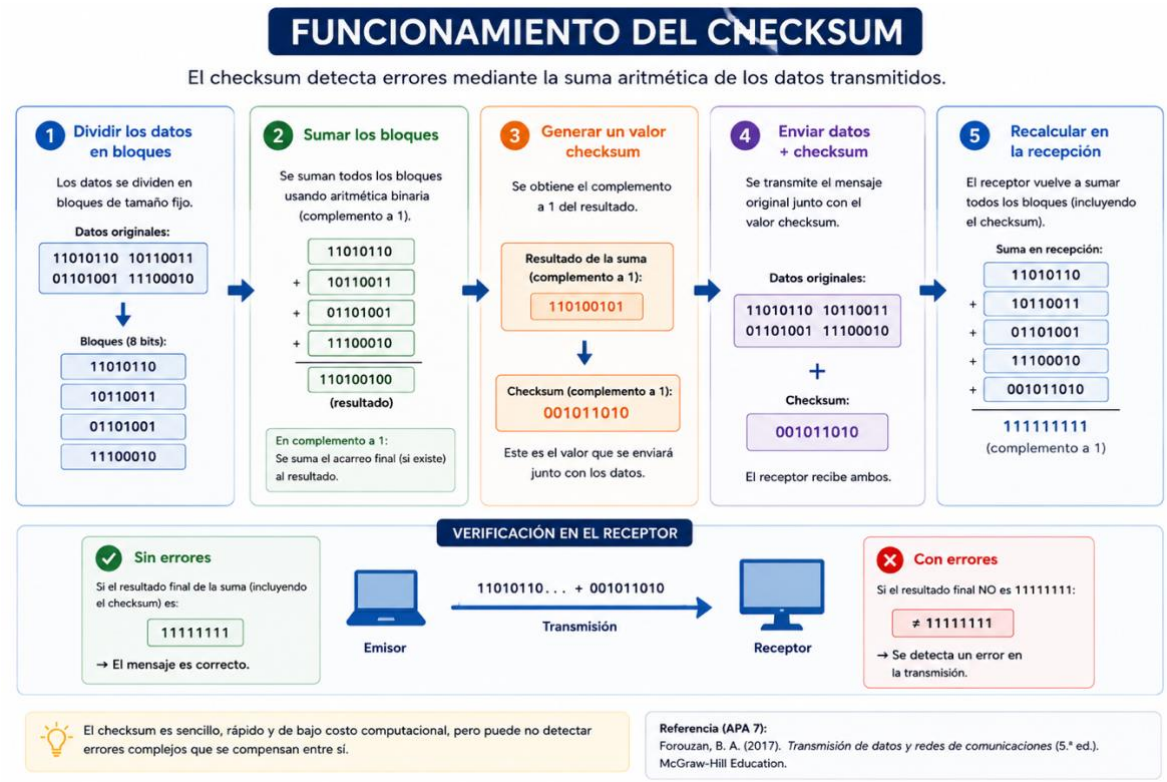
- Fácil implementación
- Bajo costo computacional
- Alta velocidad

Desventajas

- Baja precisión frente a errores complejos
- Vulnerable a alteraciones compensadas
- Menor confiabilidad

Figura 3

Proceso de funcionamiento del checksum



Nota. La figura representa las etapas del algoritmo checksum, incluyendo la división de datos en bloques, la suma aritmética, la generación del valor checksum y la verificación de integridad en recepción. Elaboración propia con base en Forouzan, B. A. (2017). *Transmisión de datos y redes de comunicaciones* (5.ª ed.). McGraw-Hill Education. Miguel Ortiz Navarrete

Supongamos los siguientes bloques binarios:
1010
1100
1001

Paso 1: Convertir a decimal

Binario	Decimal
1010	10
1100	12
1001	9

Paso 2: Sumar valores: 10 + 12 + 9 = 31

Paso 3: Enviar checksum: 31

El mensaje transmitido contiene:

- Datos originales
- Valor checksum

Verificación en la recepción
El receptor recibe:
1010
1100
1001
Checksum = 31

Vuelve a sumar: 10 + 12 + 9 = 31

Comparación: 31 = 31

Simulación de error
Supongamos que durante la transmisión ocurre una alteración:
Mensaje recibido:
1010
1110
1001

Conversión decimal

Binario	Decimal
1010	10
1110	14
1001	9

Suma: 10 + 14 + 9 = 33
Checksum esperado: 31

Comparación: 33 ≠ 31 → Se detecta error.

Aplicaciones del Checksum

El checksum se utiliza en:

- Protocolos IP
- Archivos descargados
- Verificación rápida de integridad
- Sistemas embebidos simples

CRC (Cyclic Redundancy Check)

El **CRC (Cyclic Redundancy Check)** es un método utilizado para detectar errores durante la transmisión o almacenamiento de datos. Su objetivo principal es verificar que la información recibida sea exactamente igual a la información enviada. A diferencia del bit de paridad, el CRC puede detectar errores múltiples y errores en ráfaga, por lo que es ampliamente utilizado en redes, protocolos de comunicación y sistemas de almacenamiento. (Csiszár & Körner, 2011).

El CRC añade información de verificación al mensaje para detectar alteraciones producidas durante la transmisión.

Aquí introducimos el concepto: Código cíclico

El mensaje se interpreta como un polinomio binario y se divide por un polinomio generador.

El residuo de esta división se añade al mensaje.

Funcionamiento

1. Convertir datos en polinomio
2. Dividir por polinomio generador
3. Añadir residuo
4. Verificar en la recepción

El CRC es altamente eficiente para detectar errores múltiples y ráfagas de errores.

1. Convertir datos en polinomio

En el CRC, los datos binarios se interpretan matemáticamente como un polinomio.

Por ejemplo, consideremos 1101 como mensaje a enviar

Tabla 1

Representación del mensaje binario 1101 como polinomio

Bit	Potencia
1	X^3
1	X^2
0	X^1
1	X^0

Nota. La tabla muestra cómo cada bit de un mensaje binario se asocia con una potencia de $x$para construir el polinomio utilizado en el cálculo del CRC. Elaboración propia con base en Csiszár, I., & Körner, J. (2011). *Information theory: Coding theorems for discrete memoryless systems* (2nd ed.). Cambridge University Press. Miguel Ortiz Navarrete

Por tanto: 1101 → $x^3 + x^2 + 1$
El bit con valor 0 no se incluye porque su coeficiente es cero.

2. Dividir por polinomio generador

El siguiente paso consiste en dividir el mensaje por un **polinomio generador**, también expresado en binario. G(x) = 1011

Este polinomio representa: $x^3 + x + 1$

Antes de dividir:

- Se añaden ceros al final del mensaje.
- La cantidad de ceros depende del grado del polinomio generador.
- Esta es la redundancia controlada, bits adicionales que permiten verificar la integridad del mensaje

Como el generador tiene grado 3: 1101 → 1101000

¿Cómo se realiza la división?
La división CRC no utiliza resta tradicional. En su lugar emplea: Operaciones XOR; las reglas de la operación XOR se muestran en la Tabla 2.

Tabla 2
Regla de funcionamiento de la operación XOR

A	B	Resultado
0	0	0
0	1	1
1	0	1
1	1	0

Nota. La tabla presenta la operación lógica XOR utilizada en la división binaria del algoritmo CRC para la detección de errores. Elaboración propia con base en Forouzan, B. A. (2017). *Transmisión de datos y redes de comunicaciones* (5.ª ed.). McGraw-Hill Education. Miguel Ortiz Navarrete

Si los bits son iguales → resultado 0. Si son diferentes → resultado 1.

Durante la división:

- Se alinean los bits
- Se aplica XOR
- Se continúa hasta obtener el residuo final

Dividimos 1101000 ÷ 1011

Se alinean los bits los 4 primeros con os bits del polinomio generador y aplicamos XOR

1101
1011
0110

Bajamos el siguiente bit para este caso el bit quinto (0) y aplicamos XOR

1100
1011
0111

Bajamos el siguiente bit para este caso el sexto bit (0) y aplicamos XOR

1110
1011
0101

Bajamos el último bit (0) y aplicamos XOR

1010
1011
0001

Resultado final: 001

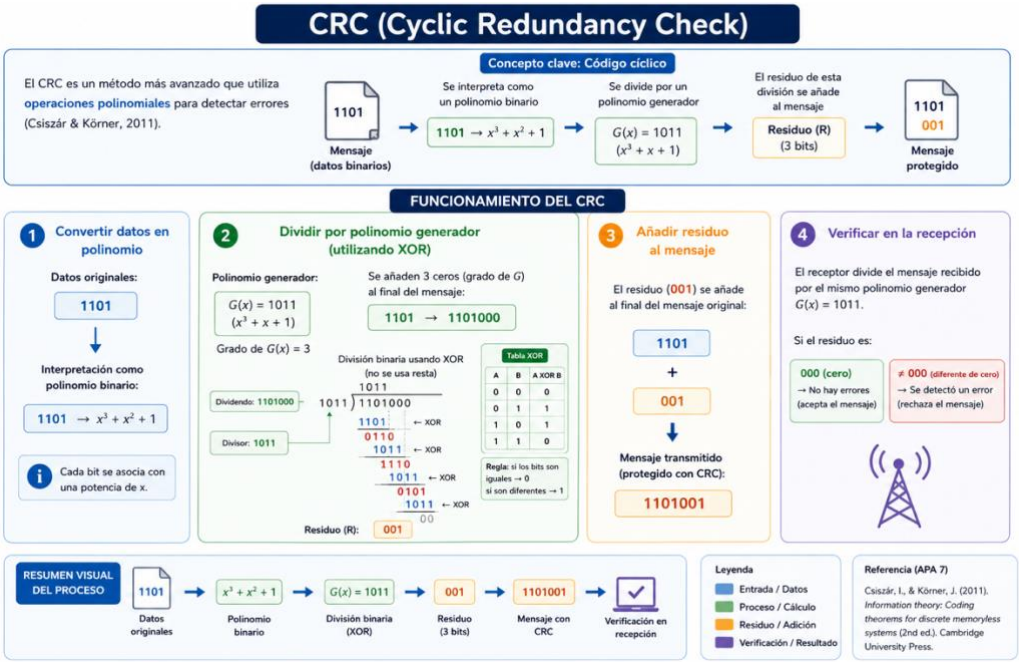
Construimos el mensaje final: 1101001

¿Qué hace el receptor?
El receptor toma: 1101001
Y vuelve a dividir usando: 1011
Si el resultado final es: 000
Entonces: No hubo errores.
Si el residuo es distinto de cero: Se detecta error.

La división CRC utiliza XOR en lugar de resta para generar un residuo que permite detectar errores en los datos transmitidos.

Figura 4

Funcionamiento del CRC (Cyclic Redundancy Check



Nota. La figura muestra el proceso de detección de errores mediante CRC, incluyendo la conversión de datos en polinomios binarios, la división utilizando operaciones XOR, la generación del residuo y la verificación de integridad en recepción. Elaboración propia con base en Csiszár, I., & Körner, J. (2011). *Information theory: Coding theorems for discrete memoryless systems* (2nd ed.). Cambridge University Press. Miguel Ortiz Navarrete

Aplicaciones

- Redes (Ethernet, Wi-Fi)
- Protocolos de comunicación
- Integridad de archivos

Tabla 3

Descripción de puertos del módulo CRC

Señal	Nombre y descripción
reset_n	Reinicia el módulo CRC antes de iniciar un nuevo cálculo.
crc_in	Indica al módulo CRC que capture los datos presentes en las líneas CRC_DATA. Debe permanecer en estado HIGH únicamente durante un ciclo BCLK.
CALC_CRC	El módulo CRC coloca el valor calculado del CRC en estas líneas después de cada señal crc_in.
CRC_DATA	Línea donde se ingresan los datos de entrada al módulo CRC.

Nota. Adaptado de *Cyclic redundancy code checksum generator*, por M. Hodson (s.f.), Geocities, https://www.geocities.ws/hodmas/crc.htm

Relación con la teoría de la información

Desde la teoría de la información:

- La entropía mide incertidumbre
- La redundancia reduce incertidumbre

- El ruido aumenta incertidumbre

Por tanto:
Sistema eficiente = equilibrio entre:

- Compresión (eficiencia)
- Redundancia (seguridad)

El diseño de sistemas requiere equilibrar eficiencia y confiabilidad.

Relación con RDA2

El CRC demuestra cómo la redundancia puede ser utilizada estratégicamente:

- Compresión → eliminar redundancia
- Seguridad → añadir redundancia

Este equilibrio es clave en el diseño de sistemas eficientes.

Alencar, M. S. (2015). Information theory. Momentum Press.

Csiszár, I., & Körner, J. (2011). Information theory: Coding theorems for discrete memoryless **memoryless systems (2nd ed.).** Cambridge University Press.

Gorzalka, J. D., & Deloumeaux, P. (2012). Information theory: New research. Nova Science Publishers.

PUCE Virtual

síguenos en:

www.pucevirtual.puce.edu.ec