Apunte de Códigos Autocorrectores

Los errores de transmisión 3
Tipos de errores 3
Los errores y su tratamiento 3
La tasa de error en el diseño de sistemas de transmisión de datos 4
Técnicas de corrección de errores 4
Consideraciones generales 4
Control de paridad 5
Paridad par e impar 5
Control de paridad vertical 5
Control de paridad longitudinal o bidimensional 6
Control de paridad entrelazada o cíclica 6
Códigos Pre-Hamming 6
Paridad 6
Dos entre cinco 7
Repetición 7
Distancia de Hamming 7
Código de Hamming 8
Implementación 10
Como detectar un error: 11
Código de Hagelbarger 11
Circuito codificador Hagelbarger 11
Circuito decodificador Hagelbarger 12
Código de Bose-Chaudhuri 12

Los errores de transmisión

Se denomina error de transmisión a

Toda alteración o mutilación que hace que un mensaje recibido no sea una réplica fiel del mensaje transmitido.

Los errores pueden alterar el contenido del mensaje de dos formas:

1. Que el mensaje quede totalmente invalidado.
2. A pesar de los errores, el mensaje es útil porque aún puede interpretarse.

Un ejemplo del caso 1 sería enviar un número *"01000000"* y recibir *"01010000"*; un ejemplo del caso 2 sería enviar un mensaje de texto *"recibirá el formulario con la mercadería"* y recibir *"recibirá el fprmulario con la mercOdería"*.

Existen dos tipos de transmisiones a considerar:

1. **De operador a operador**: la intervención de los operadores permite la detección y corrección de los errores.
2. **De máquina a máquina**: en este caso se usa la información almacenada en el sistema para poder detectar automáticamente los errores, analizarlos, aceptarlos o rechazarlos parcial o totalmente.

Tipos de errores

Clasificación (según su distribución en el tiempo)

- Errores aislados o simples
  Aquellos que afectan a un solo bit cada vez y son independientes entre sí en cuanto al momento de ocurrencia.
- Errores en ráfagas
  Afectan a varios bits consecutivos y ocurren en períodos indeterminados de tiempo.
- Errores agrupados
  Ocurren en tandas sucesivas de una cierta duración y que no afectan necesariamente a varios bits seguidos.

Los errores y su tratamiento

Una forma es enviar datos adicionales en el contenido del mensaje (el contenido superabundante de información a efectos de la detección y/o corrección de errores se denomina redundancia): aunque se logra una mayor

protección contra errores, cuanto mayor es la cantidad de bits adicionales que no llevan información, la eficiencia del proceso de transmisión disminuye. Debe existir un compromiso entre el tamaño de los bloques de cada mensaje y la eficiencia de la transmisión. Cuanto más pequeños son los bloques menos probable será la necesidad de retransmitirlos, pero la eficiencia de la transmisión disminuye. Por otra parte, cuando los bloques de mensaje son largos, con el objeto de aumentar la eficiencia, una mayor proporción de estos bloques tendrán errores y será necesario retransmitirlos, lo que a largo plazo puede disminuir aún más su eficiencia.

La tasa de error en el diseño de sistemas de transmisión de datos

La presencia de errores en el diseño de los sistemas de transmisión de datos no puede ser omitida nunca. Por este motivo, cuando se diseña o se implanta un determinado sistema con su correspondiente tecnología asociada, deben tenerse presente los siguientes aspectos:

- La tasa de error (tanto para las comunicaciones locales como para las remotas).
- Los medios para recuperar la información afectada por errores.
- La cantidad de información a transmitir por unidad de tiempo; que se corresponde con el concepto definido como **velocidad real de transferencia de datos**.
- La velocidad de transmisión que es necesaria para satisfacer los requerimientos del sistema.

Basándose en los aspectos anteriores se deberá determinar:

- El ancho de banda del canal que será necesario usar.
- El tipo de control de errores que se requiere.
- Los medio de comunicaciones y elementos técnicos que podrán cumplir con dichos requerimientos.

Técnicas de corrección de errores

Consideraciones generales

La corrección de errores es un hecho casi imprescindible. Existen dos estrategias fundamentales para la corrección de errores:

1. **Corrección hacia atrás**

Consiste en el uso de sistemas de detección de errores. Cuando se detecta un error en el equipo receptor, este solicita al equipo transmisor la repetición del bloque de datos transmitido, de ahí la llamada corrección **hacia atrás**.

Este sistema implica la retransmisión de datos tantas veces como sea necesario hasta que sean recibidos libres de errores.

Si se eligen velocidades muy altas, respecto de las que el canal de comunicaciones realmente puede soportar, los errores provocan una pérdida de tiempo mayor, que la ganancia que presuntamente se iba a obtener con una velocidad superior.

2. Corrección hacia adelante

Esta técnica denominada **Forward Error Correction (FEC)**, se basa en el uso de **códigos autocorrectores** que se diseñan sobre la base de sistemas de codificación redundante y corrigen los errores detectados en la misma estación que recibe el bloque de datos.

Aunque estos métodos hacen innecesario la retransmisión, no son neutros al usuario, pues para posibilitar la corrección en destino deben enviar un número de bits varias veces superior al que se necesita cuando se utilizan códigos convencionales.

En muchas aplicaciones no resulta conveniente pedir la retransmisión de datos, por razones de seguridad o por necesidades de la misma operación del sistema por lo que se hace necesario el uso de este tipo de códigos.

Control de paridad

Paridad par e impar

En la paridad par, si el número de unos de la palabra de información a transmitir es impar, el bit de control que se debe agregar será un uno, para que la suma total de ellos, resulte un número par. Para el caso de la paridad impar, se busca que el número de unos sea impar. Ejemplo de paridad par: *Bloque a transmitir: “01101101100” Bit de paridad: “0”.*

Control de paridad vertical

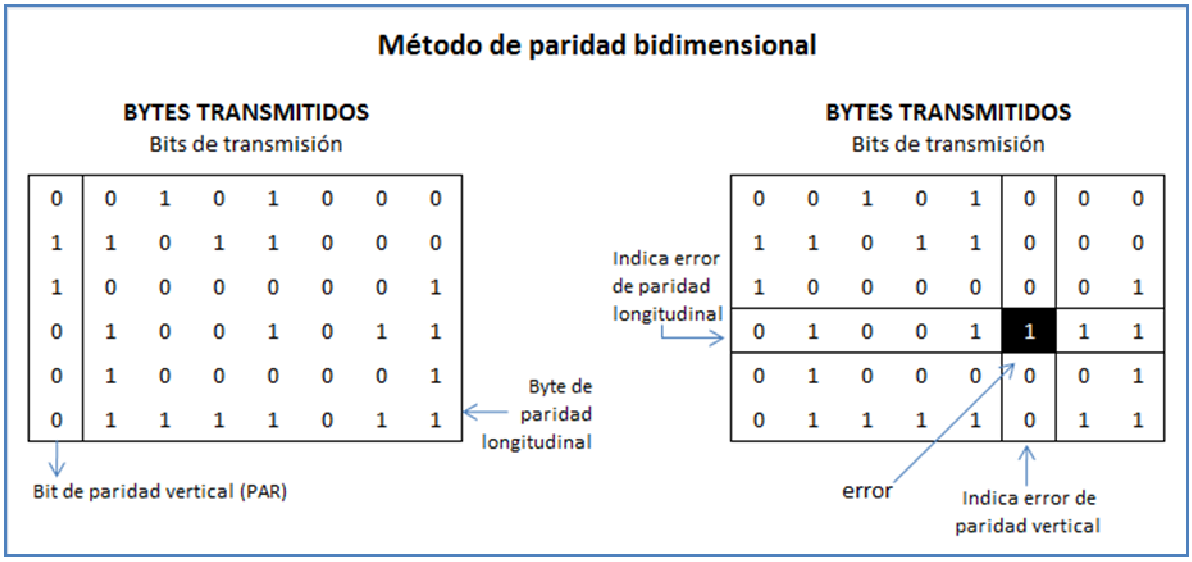
Añade un bit adicional al conjunto de 7 bits. Su uso está relacionado fundamentalmente con el código ASCII.

Control de paridad longitudinal o bidimensional

Se aplica a un conjunto de N caracteres de 7 bits cada uno. Por cada carácter de 7 bits transmitido se agrega un **bit de paridad vertical**.

Al finalizar el bloque de N caracteres, se transmitirá un carácter completo denominado **Carácter de Control del Bloque** (Block CheckCharacter BCC). El BCC se calcula de la siguiente manera:

- Se consideran los bits en la posición uno de cada carácter y se calcula el bit de paridad para todo el bloque correspondiente a esa posición. Luego se realiza el mismo proceso por cada bit del carácter incluyendo el bit de paridad.



Pueden detectar y corregir un error.

Control de paridad entrelazada o cíclica

Proporciona un nivel de detección de los errores de mayor calidad que el de **Paridad Vertical** y menor que el de **Paridad Longitudinal**.

Requiere de dos bits adicionales para el control de paridad. El primer bit de paridad proporciona la paridad de los bits primero, tercero y quinto, mientras que el segundo proporciona la paridad de los bits segundo, cuarto y sexto.

Códigos Pre-Hamming

Paridad

- Añade un bit (bit de paridad): 0 si la cantidad de 1s es par, 1 en caso contrario.
- Si un bit cambia por error se detecta el error pero no el bit erróneo.

- Si cambian dos bits, el bit de paridad será válido y el error no será detectado.
- No es un chequeo muy bueno pero produce poca sobrecarga (un único bit).

Dos entre cinco

- Basado en que cada bloque de cinco bits (penta-bit) tuviera exactamente dos unos y tres ceros (lo que forma 10 posibles estados, pudiendo representar todos los dígitos del rango 0-9).
- Si un bit cambia por error se detecta el error pero no el bit erróneo.
- Permite detectar cambios en un solo bit, pero si en un mismo penta-bit un 0 cambia a 1 y un 1 cambia a cero, la regla se cumple y el error queda sin descubrir.

Repetición

- Consiste en repetir cada bit de datos varias veces.
- Por ejemplo: si el bit de dato a enviar fuera un 1 con código de repetición n=3, enviaría “111”. Si los tres bits recibidos no eran idénticos había un error. En un ambiente con poco ruido en el que a veces cambiaría un solo bit, podrían corregirse datos 001, 010 y 100 por el bit 0. Como si el bit original se obtuviera por mayoría. Este código es corrector de errores.
- Si el error en la transmisión provocara el cambio simultáneo de dos bits y el receptor recibiera “001”, el sistema corregirá inadecuadamente el error.
- Aumentando la cantidad de repeticiones a 4, es posible detectar errores de dos bits pero no corregir; con 5 es posible corregir errores de dos bits pero no detectar los de tres. Y así sucesivamente.
- Este código es extremadamente ineficaz, pues reduce la velocidad de transmisión por n.

Distancia de Hamming

Los métodos más conocidos son **Hamming**, **Hagelbarger** y **Bose-Chaudhuri** entre otros.

Para interpretar el funcionamiento de estos códigos debemos analizar el concepto denominado **Distancia de Hamming**:

> ***Número de bits en que difieren dos secuencias binarias, S1 y S2, de la misma longitud.***

Si dos palabras de código difieren en una distancia d, se necesitan d errores para convertir una en la otra. Por ejemplo:

La distancia Hamming entre **1011101** y **1001001** es 2.
La distancia Hamming entre **2143896** y **2233796** es 3.
La distancia Hamming entre "**toned**" y "**roses**" es 3.

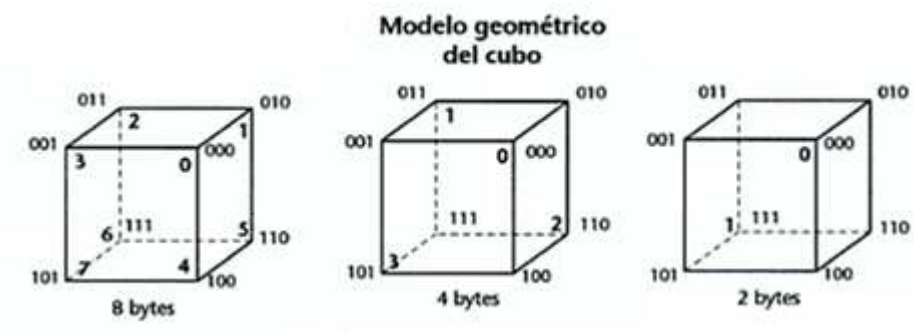
La **Distancia de Hamming** está relacionada con la probabilidad de error. Cuanto mayor sea la distancia mínima entre los símbolos de un código dado, menor será la probabilidad de cometer errores. Sin embargo, aumentar la distancia de Hamming mínima entre símbolos significa codificar menos símbolos con igual número de bits. En otras palabras, aumentando la redundancia se logra disminuir la probabilidad de cometer errores.

Para demostrar este concepto, usaremos el **ejemplo del cubo**.

1. En el primer cubo, a cada byte disponible se le ha asignado un símbolo (cifras de 0 a 7). La distancia de Hamming mínima es H = 1, que se da entre los bytes situados en vértices adyacentes. Con este valor no es posible detectar ni corregir ningún error, pues la alteración de un bit dará por resultado una nueva combinación que tiene asignado otro símbolo.

2. En el segundo cubo se han asignado símbolos (cifras de 0 a 3) a 4 bytes. En este caso H = 2. En este caso es posible detectar la presencia de un bit erróneo, pero no corregirlo. La alteración de un solo bit resultará en una combinación que no tiene símbolo asignado, por lo que se puede deducir que se trata de un error.

3. En el último cubo, H = 3. Con esta distancia de Hamming mínima es posible detectar el error y también corregirlo.



Código de Hamming

Los bloques de bits transmitidos estarán compuestos por n bits: k bits de datos y r bits de redundancia.

- Podrán formarse 2^n códigos.
- Podrán formarse 2^k mensajes, es decir, 2^k códigos válidos.
- Si n = k + r, entonces habrá n códigos a distancia 1 de un código válido.

- Al haber n+1 códigos asociados a cada mensaje (o código válido) (n códigos inválidos a distancia 1 del mismo más el propio código válido) entonces: $(n+1) * 2^k <= 2^n$

$$(n+1) * 2^k <= 2^n \textbf{ -> } (r + k +1) * 2^k <= 2^{k+r} \textbf{ -> } r + k + 1 <= 2^r \textbf{ -> } \mathbf{k <= 2^r - r - 1}$$

La cantidad de bits que puede tener el mensaje para ser corregido con r bits de redundancia ha de ser menor o igual a $\mathbf{2^r - r - 1}$.

Tabla de ejemplo:

r	k	n
3	4	7
4	11	15
5	26	31
6	57	63
7	120	127
8	247	255

r	k	n
9	502	511
10	1013	1023
11	2036	2047
12	4083	4095
13	8178	8191
14	16369	16383

Recordar que k indica la cantidad máxima de bits que puede tener el mensaje. Si mis mensajes fueran de 10 bits, necesitaría 4 bits de redundancia.

Es un método general propuesto por R. W.Hamming usando una ***distancia mínima r***. Con este método, ***por cada entero r existe un código de Hamming de 2^r-1 bits que contiene r bits de paridad y 2^r-1-r bits de información***.

Los bits de información y los bits de paridad se encuentran entremezclados de la siguiente forma: ***si se numeran las posiciones de los bits desde 1 hasta 2^r-1, los bits en la posición 2^i, donde 0 ≤i≤r-1, son los bits de paridad y los bits restantes son bits de información***.

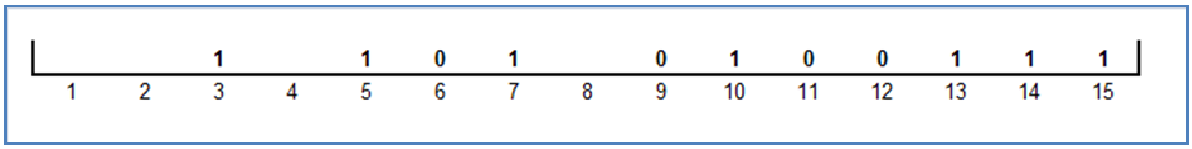
El valor de cada bit de paridad es la paridad par de un número determinado de bits. Estos grupos se escogen de tal forma que ningún bit de información se cubra con la misma combinación de bits de paridad. Es lo anterior lo que proporciona al código su capacidad de corrección.

Para cada bit de paridad en la posición 2^i, su grupo de bits de información correspondiente incluye todos esos bits de información correspondiente cuya representación binaria tenga un uno en la posición 2^i.

Ej.: código de Hamming de 7 bits o sea de la forma 2^r-1 con r = 3. En este ejemplo, los bits de información son 4 y los bits de paridad son 3. Los bits de información están en las posiciones 7, 6, 5 y 3. Los bits de paridad están en las posiciones 1, 2 y 4.

Implementación

Los bits redundantes ocupan las posiciones potencia de 2(para el caso de 15 bits: 1,2,4 y 8), el resto son los bits de datos.



Bits				
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1

Para generar el valor de los bits de redundancia se toman los bits cuya posición contengan bits en común. Es decir, para generar el valor del bit 1, se tomarán los posiciones de los bits 3, 5, 7, 9, 11, 13 y 15 que todas tienen un 1 en el último bit. Para el valor del bit 2 se tomarán los bits 3, 6, 7, 10, 11, 14 y 15. Entonces quedaría así con paridad impar:

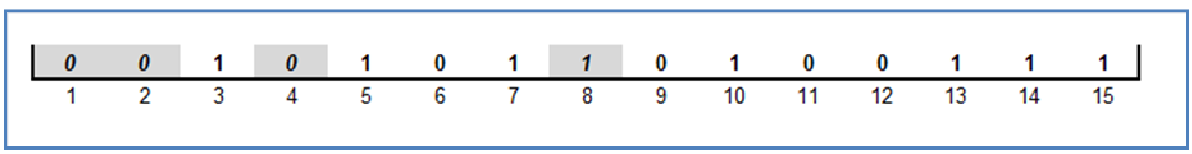
Bit 1: (1, 3, 5, 7, 9, 11, 13, 15) =>(**0**, 1, 1, 1, 0, 0, 1, 1)

Bit 2: (2, 3, 6, 7, 10, 11, 14, 15) =>(**0**, 1, 0, 1, 1, 0, 1, 1)

Bit 4: (4, 5, 6, 7, 12, 13, 14, 15) =>(**0**, 1, 0, 1, 0, 1, 1, 1)

Bit 8: (8, 9, 10, 11, 12, 13, 14, 15) =>(**1**, 0, 1, 0, 0, 1, 1, 1)

Resultando:



Como detectar un error:

Supongamos un error: el bit 10 pasa a valer 0.

El bit 1 y el 4 verificarán, pero no el 2 y el 8.

Bit 2: (2, 3, 6, 7, 10, 11, 14, 15) =>(**0**, 1, 0, 1, 0, 0, 1, 1) NO VERIFICA

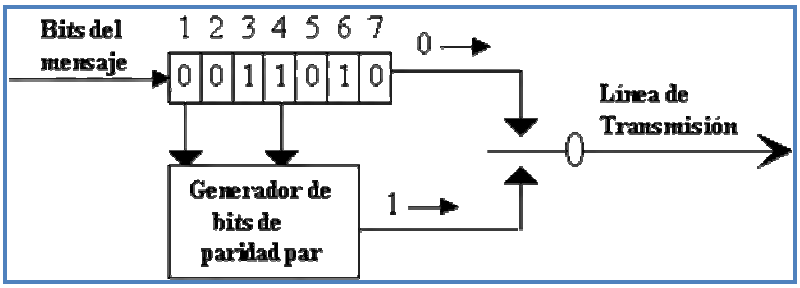
Bit 8: (8, 9, 10, 11, 12, 13, 14, 15) =>(**1**, 0, 0, 0, 0, 1, 1, 1) NO VERIFICA

Para saber cual bit sufrió el error, se suman los números de las posiciones de los bits que no verifican (2 + 8 = 10).

Código de Hagelbarger

Este código detector-corrector de errores es capaz de corregir hasta 6 bits erróneos siempre y cuando los sucedan al menos 19 bits válidos. La implementación del código de Hagelbarger implica el empleo de un circuito codificador y un circuito decodificador.

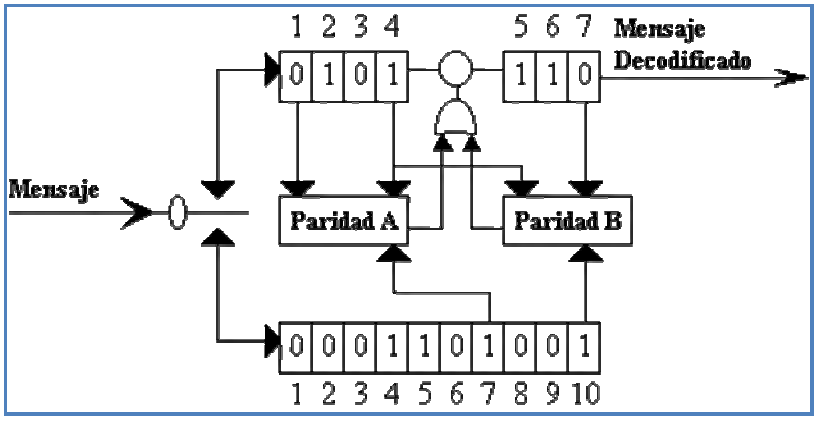
En la siguiente figura se muestra el codificador Hagelbarger:



Circuito codificador Hagelbarger

El codificador Hagelbarger recibe los bits del mensaje y los pasa uno a la vez a través del registro de 7 bits. Cada vez que entra un nuevo bit se genera un bit de paridad par a partir de los bits contenidos en la primera y cuarta posición del registro. Los bits que salen del registro de 7 bits y los bits de paridad generados se envían alternadamente por la línea de transmisión—es decir, este código presenta una redundancia del 100%. De esta forma, todos los bits del mensaje se encuentran enlazados, por medio de la paridad par, con el cuarto bit que le precede y sucede.

En la siguiente figura se muestra el decodificador Hagelbarger:



Circuito decodificador Hagelbarger

El decodificador Hagelbarger recibe a la sucesión de bits y separa a los bits del mensaje de los bits de paridad, enviando a estos últimos hacia un registro circulante. Después de recibir a cada par de bits (uno del mensaje y otro de paridad) se efectúan 2 verificaciones de paridad par, denominadas verificaciones de paridad A y B. La verificación de paridad A comprueba si las posiciones 1 y 4 del registro de mensajes son iguales mientras que la verificación de paridad B verifica la igualdad de las posiciones 4 y 7 del mismo registro. Estas verificaciones de paridad nos indican si debemos o no de invertir los valores de los bits de las posiciones 4 del registro de mensajes y 7 del registro circulante antes de pasar a la siguiente posición correspondiente, de acuerdo con la siguiente tabla de verdad.

Cambio en la posición 4 del registro de mensajes sí:	PPA = 1 y PPB = 1 ó PPA = 0 y PPB = 1
Cambio en la posición 7 del registro circulante sí:	PPA = 1 y PPB = 0 ó PP(PPA, posición 7 del registro circulante) = 1

Código de Bose-Chaudhuri

Tiene una distancia H = 5; consecuentemente puede detectar hasta cuatro errores y corregir hasta dos bits. Existen varias versaciones del código, pero la primitiva preveía la introducción de 10 bits adicionales, por cada 21 bits de información transmitida.