Unidade de Traballo 3

Codificación da información e detección e corrección de erros

Comentario [CCA1]: García T. (135), Tanenbaum, (242), Equipos (Tema 3), Ciclo (47)

INDICE.

Unidade de Traballo 3

Codificación da información e detección e corrección de erros

Comentario [CCA2]: García T. (135), Tanenbaum, (242), Equipos (Tema 3), Ciclo (47)

4.1.- INTRODUCCIÓN.

- **Código:** Correspondencia entre un conxunto F, (alfabeto fonte) e S (Conxunto de símbolos).

  A cada elemento de F asígnaselle un grupo de símbolos **(Palabra)**.

  Para que o código sexa útil a correspondencia debe ser biunívoca, recíproca e inequívoca

- **Codificación:** proceso de conversión do conxunto F ó conxunto S

Palabra Fonte	**Palabra Código**.
A B	A B A+B
0 0	0 0 0
0 1	0 1 1
1 0	1 0 1
1 1	1 1 0
Función de codificación: f(A,B) = (A,B,A+B)	

- **Redundancia dun código**: diferencia entre a información máxima que podería proporcionar un código e a que realmente proporciona. Úsanse os díxitos que non transportan información como detectores e incluso correctores de erros.
- **Taxa de erros**: Relación entre os bit recibidos erróneos e os bits transmitidos, pois sempre existe a posibilidade de que se introduzan erros na información transmitida (*ruídos*).
- **Tipos de ruídos:**

  Ruído Impulsivo: Probabilidade de que un bit sexa erróneo. Por exemplo 10^{-3}.

  Ruído por Ráfagas: O erro comeza nun determinado bit e prodúcense erros aleatorios ó longo de toda a ráfaga. Erro por ráfaga de 100 bits.

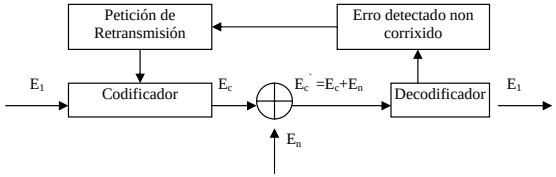
Exercicio:

1.- Deséxanse transmitir 100.000 bits. Estudia como afecta os distintos tipos de ruído a eses 100.000 bits, nos seguintes casos: Que se transmitan os 100.000 nun só bloque, que se transmitan en bloques de 1.000 bits ou que se transmitan en bloques de 100 bits.

- **Probabilidade de erro:** Depende das condicións dos elementos que interveñen na canle de transmisión, a saber

Canle	Probabilidade de erro/bit
Liñas telefónicas conmutadas	10^{-5}
Liñas telefónicas dedicadas a 1.200 bits/seg	10^{-6}
Liñas telefónicas dedicadas a 4.800 bits/seg	10^{-5}

- **Esquema básico dunha transmisión de información:**



E_1 é a mensaxe inicial e E_c, a mensaxe inicial codificada. A causa do ruído aditivo na liña aparece: $E´_c = E_c + E_n$.

O chegar $E´_c$ ó decodificador poden ocorrer dúas cousas:

Que $E´_c$ pertence ó conxunto de posibles palabras do código: A transmisión considerase boa e a decodificación da como resultado E_1.

Que $E´_c$ non sexa unha palabra do código: Detéctase ó erro, se este é corrixido pola lóxica do decodificador, extráese E_1 . Se só é detectado, pero non corrixido, pídese a repetición da mensaxe.

Segundo a potencia do código, os sistemas poden detectar erros, corrixilos ou ben ámbalas dúas cousas, corrixindo algúns tipos de erros e detectando outros.

Para que a detección e corrección de erros se leve a cabo é preciso unha serie de bits (r bits), *bits de redundancia*. Co cal a palabra código estará composta por n=m+r bits (m bits de información).

Comentario [CCA3]: Equipos (Tema 3), Alxebra (Tema Códigos Lineais)

3.2.- DISTANCIA HAMMING

A ***distancia Hamming*** entre dúas palabras dun código é o número de bits en que difiren ambas: por exemplo d(010010110, 110110111)=3

010010110 110110111	Estas dúas palabras difiren en 3 bits. H=3 (Distancia Hamming = 3 bits)
100100001	1 indica onde son distintos os bits

Defínese ***peso (w)*** dunha palabra X como d(X,0), noutras palabras é o número de 1s da palabra.

Dúas palabras serán máis fáciles de distinguir canto maior sexa a súa distancia Hamming, debido a que si a distancia é d, fan falla d bits erróneos para transformar unha palabra noutra. Co cal a eficacia dun código depende da *distancia Hamming mínima* que poida encontrase entre dúas das súas palabras:

- Un código C detecta p ou menos erros si $d_{min} \geq p+1$
- Un código C corrixe p ou menos erros si $d_{min} \geq 2p+1$

Exemplos

Código 1:

Mensaxes	Palabras Código = x
AB	AB+
00	000
01	011
10	101
11	110
Función de codificación: f(A,B) = (A, B, A+B)	

As palabras código son : {000, 011, 101, 110}. Si se observa vese que a $d_{min}= 2$, co cal este código detecta erros simples (nun só bit), pero non capaz de corrixilos.

O transmisor desexa enviar 01 para iso ***codifica*** como 011 e transmítea. O receptor recibe 111. A palabra 111 non é unha palabra código, co cal non pode ***decodificala*** e así obter a mensaxe orixinal. Esta palabra 111 non puido ser transmitida, co cal produxeuse un erro nun bit pero o decodificador non pode determinar en cal. Non podemos recuperar a mensaxe orixinal, esto implica que temos que solicitar unha retransmisión.

Código 2:

Mensaxes	Palabras Código = x
AB	ABABAB
00	000000
01	010101
10	101010
11	111111
Función de codificación: f(A,B) = (A, B, A, B, A, B)	

A distancia mínima deste código 2 é $d_{min}= 3$, co cal o decodificador corrixe erros simples e detecta erros dobres e simpres.

Podemos supor que os erros ocorren aleatoriamente e independentemente, e que a probabilidade de erro é igual en calquera dos díxitos (si $q \leq 0.5$, é máis probable que se produza un erro que 2, 2 que 3, ...). Por iso úsase a **decodificación de máxima verosimilitude**: e dicir, para decodificar a palabra recibida buscarase aquela palabra código máis probablemente transmitida, ou sexa a que defira no menor número de díxitos da palabra recibida.

Por exemplo si o código e C={000000, 010101, 101010, 111111} e si y=010111 é a palabra recibida, calcúlase:

.d(000000, 010111)=4
.d(010101, 010111)=1
.d(101010, 010111)=5
.d(111111, 010111)=2

A menor distancia Hamming entre as palabras código é a palabra recibida prodúcese no 2º caso (*decodificación de máxima verosimilitude*). Co cal, a palabra código máis probablemente transmitida sexa: x=010101. Neste caso o decodificador detecta e corrixe o erro.

O erro é e=000010, w(e)=1, y= 010101 + 000010
x ∈ C + e

Código 3:

Mensaxes	Palabras Código = x
ABC	
000	
001	
010	
011	
100	
101	
110	
111	
Función de codificación: f(A,B,C) = (A, B, C, A+B, B+C, A+B+C)	

Comentario [CCA4]: A distancia mínima é 3, existe unha folla de calculo chamda distancia min, para o calculo

Exercicios:

Comentario [CCA5]: Existe na páxina 244 Tanenbaum un posible exercicio de exame

2.- ¿Que fai o decodificador no código 2 si o receptor recibe 000011?

3.- Desexase enviar 01 e o receptor recibe 010000, ¿Que pasou e como actúa o decodificador?

4.- Calcula as palabras códigos do código 3.

5.- ¿Cantos bits erróneos pode detectar e corrixir o código 3?

6.- ¿Cal dos tres códigos é mais efectivo atendendo a relación bits de información bits de redundancia?

7.- Transmítese unha palabra código, no código 3, producese un erro, ¿Como actúa o decodificador?. Faino poñendo un exemplo.

8.- Temos m bits para as mensaxes e r para a redundancia. ¿Cantos mensaxes podemos ter, e cantas palabras código legais?.¿Úsanse tódalas combinacións posibles dos n bits, sendo n=m+r?

3.3.- CÓDIGOS DE CONTROL DE PARIDADE

3.3.1.- Paridade simple

O **bit de paridade** é aquel que se selecciona en función do número de 1s da palabra mensaxe. Así temos:

Tipo de paridade	Definición	Exemplo
Par	Número de 1s é par: engádese un 0	0011011 (0)
Impar	Número de 1s é impar: engádese un 0	0011011 (1)

O cálculo, realmente, faise coa función EXOR (OR-Exclusivo), definida como segue:

AB	$A \oplus B = \overline{A}.B + A.\overline{B}$
00	0
01	1
10	1
11	0

A partires de agora, salvo que se diga o contrario, cando se fale de paridade estarase falando de paridade par.

Este código ten unha distancia mínima de 2, co cal só detecta erros simples. Detecta tamén erros si o número de bits erróneos é impar, pois si fose para un erro compensaría có outro.

Exemplo:

Deséxase enviar a seguinte palabra 1001 o transmisor calcula o bit de paridade (0), deste xeito a palabra código que se envía é 10010. O receptor recibe 10011, este calcula de novo o bit de paridade e dálle (0) e el recibiu (1) co cal detecta o erro pero nono pode corrixir.

3.3.1.- Paridade de bloque

Trátase de organizar a información por bloques, compoñendo unha táboa de **k** x **m** bits (k palabras de m bits cada unha). Logo calcúlase os bits de paridade de cada unha das m filas (***paridade horizontal***) e k columnas (***paridade vertical***). Por último calculase o ***bit de paridade cruzada*** a partires da columna ou da fila de paridade calculada anteriormente.

Exemplo:

	Mensaxe	Paridade Horizontal
	101110	0
	011010	1
	111000	1
	101010	1
Paridade Vertical	100110	1 Bit de Paridade Cruzada

Neste caso o codificador fai bloques de 4 palabras de 6 bits cada unha, e calcula as distintas paridades, logo envía ó receptor, unha fila tras outra. Ter en conta que envía 5 palabras código de 7 bits cada unha.

O decodificador reconstrúe o bloque e volve a calcular os bits de paridade correspondentes e compáraos cos bits de paridade recibidos.

Este código ten unha distancia mínima de 4, co cal corrixe erros simples e detecta erros dobres, triples e cuádruples se estes non forman un rectángulo na matriz de díxitos.

Exercicios:

9.- Baseado no exemplo anterior. O emisor envía esas 4 palabras de 6 bits cada unha. Na canle producese ruído no 1º bit da 1ª palabra. ¿Qué pasos realiza o decodificador no caso de paridade simple e no caso de paridade de bloque?

10.- Igual que no exercicio anterior, pero esta vez tamén se produce un erro no 2ºbit da 2ª palabra. ¿Qué pasos realiza o decodificador no caso de paridade simple e no caso de paridade de bloque?

11.- ¿Cal é a efectividade deste código?

3.4.- CÓDIGOS HAMMING

Comentario [CCA6]: Alxebra (Tema Códigos Lineais), Tanenbaum (244), García Tomas (140)

Son un subconxunto dos códigos de control de paridade. Os díxitos de paridade dispóñense de maneira que localicen a presencia de erros dentro da palabra. Estes códigos teñen, xeralmente, distancia mínima 3.

Si temos palabras código de N bits. Para corrixir un erro ou detectar a ausencia de erros, precisamos, ó menos, R deses N bits como bits de control, de tal xeito que:

$N=2^R-1$

Desta fórmula dedúcese que o código máis pequeno terá 3 bits(2 de control e 1 de información).

Regras relativas o control de paridade nos códigos Hamming:

1.- Dous bits non poden controlar a paridade dun mesmo conxunto de bits da mensaxe.

2.- Non se debe incluír no conxunto de bits controlados por un bit de paridade a outros bits de paridade, pois el detecta o seu propio erro.

3.- Un erro nun bit de información debe afectar a un ou mais díxitos de paridade.

Exemplo:

Temos un código Hamming con 3 ($c_4\ c_2\ c_1$)bits de paridade. A lonxitude da palabra será 7 e haberá 4 bits de información ($b_7\ b_6\ b_5\ b_3$). Co cal a palabra código será da seguinte forma:

$b_7\ b_6\ b_5\ c_4\ b_3\ c_2\ c_1$

Haberá tantas ecuacións de paridade como díxitos de control. Para obter estas usase a seguinte táboa. Notar que cada bit de control ocupa unha posición potencia de 2, deste xeito el mesmo controla o seu propio erro.

Pos	c_4 c_2 c_1	ERRO	Ecuacións Codificador	Ecuacións Decodificador
0	0 0 0	NON ERRO	$c_1 = b_3 \oplus b_5 \oplus b_7$	$e_1 = b_3 \oplus b_5 \oplus b_7 \oplus c_1$
1	0 0 1	c_1 *Está na posición* 2^0	$c_2 = b_3 \oplus b_6 \oplus b_7$	$e_2 = b_3 \oplus b_6 \oplus b_7 \oplus c_2$
2	0 1 0	c_2 *Está na posición* 2^1	$c_4 = b_5 \oplus b_6 \oplus b_7$	$e_3 = b_5 \oplus b_6 \oplus b_7 \oplus c_4$
3	0 1 1	b_3		
4	1 0 0	c_4 *Está na posición* 2^2		
5	1 0 1	b_5		
6	1 1 0	b_6		
7	1 1 1	b_7		

Un emisor desexa enviar 0101, o codificador realiza o seguinte en base as ecuacións anteriores:

```
b7 b6 b5 c4 b3 c2 c1
0  1  0  1  1  0  1
```

Esta palabra transmítese pola canle e sufre un erro en b_2. O decodificador realizará a seguinte operación ó recibir a palabra código:

```
b7 b6 b5 c4 b3 c2 c1    e3 e2 e1
0  0  0  1  1  0  1     1  1  0  (=6 en decimal).
```

Como se pode observar o decodificador detectou en que posición se produxo o erro.

Exercicios:

12.- Baseado no exemplo anterior. O emisor envía 1011, pero pola canle o bit c_2 cambia. ¿Qué pasos realiza o decodificador?

13.- Igual que no exercicio anterior, pero esta vez tamén se produce un erro no bit b_7 ¿Qué pasos realiza o decodificador?

14.- ¿Cal é aCÓDIGOS efectividade deste código?

Comentario [ISC7]: Ciclo (48, 89), Tanenbaum(246-250)

3.5.- CODIGOS DE REDUNDANCIA CÍCLICA (CRC)

Baséanse no tratamento de series de bits como si foran representacións de polinomios, con coeficientes de valor 0 e 1, unicamente. Así 1011001 (son 7 bits) daría lugar a un polinomio de grado 6 (7-1):

$$1.x^6 + 0.x^5 + 1.x^4 + 1.x^3 + 0.x^2 + 0.x^1 + 1.x^0 = x^6 + x^4 + x^3 + 1$$

Emisor e receptor deben deben poñerse de acordo no polinomio divisor ou xerador. O emisor divide o polinomio-mensaxe entre o polinomio-xerador, obtendo un cociente que se ignora e un polinomio resto. Este polinomio é un secuencia de bits que se engaden o polinomio-mensaxe, este campo é chamado SVT (Servicio de Verificación de Tramas), constituíndo así a trama a enviar ó receptor.

Cando o receptor recibe a trama, volve a dividir a parte da trama que corresponde o polinomio-mensaxe entre o polinomio-xerador, obtendo un polinomio-resto. Este polinomio e comparado co SVT que recibiu na mesma trama. Si non coinciden interpretase que existiu un erro.

Estes algoritmos soen estar implementados en hardware, o que implica unha maior velocidade a hora de detectar un erro.

Os métodos CRC soen detectar os seguintes tipos de erros:

- Erros simples
- Erros de máis de 1 bit si o polinomio divisor é suficientemente grande.
- Un ráfaga de erros de lonxitude menor o SVT
- Outros moitos non descritos aquí.

Exemplo

O tamaño das mensaxes é de 6 bits e o polinomio xerador é x^2+1, co cal os seus coeficientes son 101. Desto dedúcese que o tamaño da trama (mensaxe+SVT) é de 8 bits (6+2 do grado do polinomio xerador).

O emisor desexa enviar 101111. O algoritmo para calcular a redundancia (SVT) é.

1º. Sexa r o grado do polinomio xerador, pois engadir r ceros no extremo inferior da trama: *no noso caso: 10111100*

2º. Facer a división binaria do polinomio resultante do paso anterior entre o xerador.

3º. Suma o resto resultante do paso anterior o polinomio do paso 1º.

```
10111100 | 101
---------|-----
101      | 100101
0001
 000
 0011
  000
```

```
 0111
  101
 0100
  000
 1000
  101
 0011
```

O resto desta división é 11, que será o SVT da trama a enviar, co cal esta queda da seguinte forma: 10111100.

Si pola canle de transmisión se produce un erro no 1º bit a esquerda do SVT, o receptor procedería da seguinte forma:

```
10111000 | 101
---------+------
101      | 100100
0001
 000
 0011
  000
  0110
   101
   0010
    000
    0100
     000
     0100
```

O resto desta división é 100 que comparado co SVT da trama recibida pódese comprobar que non coinciden co cal. Deste xeito o receptor detecta o erro producido.

Na maioría dos casos os códigos CRC só detectan os erros e non os corrixen. Nestes casos o receptor solicítalle o transmisor que lle retransmita a trama recibida erroneamente.

Existen tres polinomios xeradores que se converteron en normas internacionais:

CRC-12 = $x^{12} + x^{11} + x^3 + x^2 + x^1 + 1$

CRC-16 = $x^{16} + x^{15} + x^2 + 1$

CRC-CCITT = $x^{16} + x^{12} + x^5 + 1$

Os dous últimos capturan tódolos erros dos seguintes tipos:

- Erros simples
- Erros dobres
- Tódolos erros con un número impar de bits.
- Tódolos erros de ráfagas con CORRECCIÓNlonxitude de 16 bits ou menos.
- 99.997% dos erros de ráfagas de 17 bits.

3.6.- A CORRECIÓN DE ERROS

Unha vez que o receptor detectou unha situación de erro, esta situación debe ter unha solución. Neste senso existen dous métodos para corrixir unha situación de erro, a saber: Corrección de erros no destinatario e corrección de erros por retransmisión.

3.6.1.- Corrección de erros no destinatario

Neste caso cando o receptor detecte un erro intentará corrixilo coa información redundante que recibiu do emisor na propia trama. Son exemplos deste método:

- Os métodos de paridade de bloque.

- Os códigos Hamming.
- Códigos construídos tendo en conta as distancias Hamming.

Estes dous últimos códigos son moi seguros, pero precisan dun canal con maior capacidade (nº de bits pos segundo que pode transmitir o canal) para poder transmitir a mesma información que un canal con menor capacidade e que usa outro código con menor redundancia.

Por exemplo si temos un emisor que transmite palabras de 4 bits, teríamos os seguintes datos:

Exemplo 1º

	Paridade de bloque (bloques de 5 palabras)	**Códigos Hamming. (3 bits de redundancia)**
Nº Bits do bloque	6 filas x 5 columnas = 30 bits	5 palabras x 7 Bits=35 bits
Nº Bits de información	5 palabras x 4 bits = 20 bits	5 palabras x 4 bits = 20 bits
Nº Bits de redundancia	5 filas x 1 bit + 5bits hrzt =10	5 palabras x 3 bits = 15
Aproveito da canle **Fórmula xeral**	20/30 = 0,66 = 66% 4.palabras / 5.palabras + 5	20/35=0,57 = 57% 4 / 7
Capacidade da canle para poder transmitir a mesma info no mesmo tempo. C(bits/seg)	30 bits/seg. En 1 seg transmite 20 bits de información e 10 de control	35 bits/seg. En 1 seg transmite 20 bits de información e 15 de control

Exemplo 2º

	Paridade de bloque (bloques de 10 palabras)	**Códigos Hamming. (3 bits de redundancia)**
Nº Bits do bloque	11 filas x 5 columnas = 55 bits	10 palabras x 7 Bits=70 bits
Nº Bits de información	10 palabras x 4 bits = 40 bits	10 palabras x 4 bits = 40 bits
Nº Bits de redundancia	10 filas x 1 bit + 5bits hrzt =15	10 palabras x 3 bits = 30
Aproveito da canle	40/55 = 0,72 = 72% Pode chegar a aproveitar o 80%	40/70=0,57 = 57%
Capacidade da canle para poder transmitir a mesma info no mesmo tempo. C(bits/seg)	55 bits/seg. En 1 seg transmite 40 bits de información e 15 de control	70 bits/seg. En 1 seg transmite 40 bits de información e 30 de control
Outro exemplo de capacidade si temos unha velocidade 10 veces maior que a anterior		
Capacidade da canle para poder transmitir a mesma info no mesmo tempo. C(bits/seg)	1100 bits/seg. ≈ 1.1 Kbps En 1 seg transmite 800 bits de información e 300 de control	1400 bits/seg. ≈ 1.4 Kbps En 1 seg transmite 800 bits de información e 600 de control

3.6.2.- Corrección de erros por retransmisión

É moito máis sinxelo detectar o erro que corrixilo, pois a operación de corrección require calcular cales son as posicións dos bit erróneos. Na meirande parte das comunicacións actuais a corrección de erros faise por retransmisión das tramas, nas que se detectaron erros.

Obviamente este método require a comunicación bidireccional, semidúplex e preferiblemente full-duplex.

Dentro deste método existen 2 técnicas:

3.6.2.1.- Envío e espera

O transmisor envía a trama e non envía a seguinte ata que o receptor llo comunique. E dicir que o transmisor está as ordes do receptor. Pódese usar dúplex ou semidúplex. Poden acontecer dous casos:

- Se o receptor lle indica o transmisor que a trama anterior lle chegou incorrecta (NACK) o transmisor retransmite esa trama e espera ata nova orde.
- Se o receptor lle indica o transmisor que a trama anterior lle chegou correcta (ACK) o transmisor transmite a seguinte trama que corresponda e espera ata nova orde.

3.6.2.2.- Envío continuo

Esta técnica precisa comunicación dúplex, pois mentres o transmisor envía tramas o receptor vaille contestando cales recibiu correctas e cales incorrectas. Neste caso o transmisor vai enviando tramas sen parar e no caso de que reciba un NACK pode actuar de dúas formas.

Para elo imaxínese o seguinte que o transmisor ten que enviar 10 tramas (1-10), no momento en que xa ía enviando a trama 7 recibe un NACK2 (e dicir que o receptor comunicalle ó transmisor que a trama 2 estaba incorrecta), co cal o transmisor ten que retransmitir a trama 2, pois procede segundo da técnica que estea a usar:

- **Rexeite non selectivo:** retransmite todo dende a trama errónea. No caso do exemplo. Transmite 1, 2, 3, 4, 5, 6, 7 neste momento e cando recibe NACK2 e retransmite: 2, 3, 4, 5, 6, 7 e continúa con 8, 9, 10.
- **Rexeite selectivo:** Retransmítese só a trama errónea unha vez detectado polo transmisor o seu NACK. Neste caso o transmisor faría: 1, 2, 3, 4, 5, 6, 7, **2**, 8, 9, 10.

3.6.3.- Comparación dos códigos detectores o dos correctores de erros.

Sexa un sistema de transmisión coas seguintes características:

Pe_{bit} = 10^{-6} (e dicir, por cada 1.000.000 de bits un é erróneo)

Bloque = 1.000 bits

Bits a transmitir = 10^6 (e dicir, 1.000 bloques de 1.000 bits cada un, e ademais vai ocorrer un erro nun bit neste millón de bits)

	Código detector	**Código Corrector**
Nº Bits de redundancia	1 bit /bloque	10 bits / bloque
Nº total bits redundancia	1.000 bits	10.000 bits
Nº total de bits	Bits información 1.000.000 Bits redundancia 1.000 Bits retransmitidos 1.001 Total 1.002.001	Bits información 1.000.000 Bits redundancia 10.000 Total 1.010.000
Efectividade	99,8%	99 %

Conclusión: Si a probabilidade de erro é baixa e mellor usar códigos detectores que códigos correctores, xa que o tamaño da redundancia é menor.