Clase 3

Práctico de Diseño Lógico

Clase 3 – temas

- Códigos
  - Distancia
  - Corrección y detección de errores
  - Código de Hamming
- Álgebra de Boole
- Funciones Lógicas

Códigos para detección y corrección de errores.

- Distancia entre palabras: cantidad de bits que difieren entre sí.
- Distancia del código: mínima distancia entre palabras válidas.
- Error simple: cambia un bit.
- Error doble: cambian dos bits.
- Error triple: cambian tres bits.
- Etc.

Códigos para detección y corrección de errores.

- Distancia entre palabras: cantidad de bits que difieren entre sí.
- Distancia del código: mínima distancia entre palabras válidas.
- Error simple: cambia un bit.
- Error doble: cambian dos bits.
- Error triple: cambian tres bits.
- Etc.

Ejemplo

Ejercicio 2.6

Ejercicio 6. Se consideran los cuatro códigos siguientes:

código A	código B	código C	código D
0001 0010 0100 1000	código Gray de 3 bits	01011 01100 10010 10101	000000 001111 110011

a) ¿Cuál de las siguientes propiedades son satisfechas por cada uno de ellos?

- Detectar errores simples.
- Detectar errores dobles.
- Detectar errores triples.
- Corregir errores simples.
- Corregir errores dobles.
- Corregir errores simples y detectar errores dobles.

b) ¿Cuántas palabras pueden agregarse al código A sin cambiar sus propiedades de detección y corrección de errores?. Dé un posible conjunto de dichas palabras. ¿Es este conjunto único?

Código B

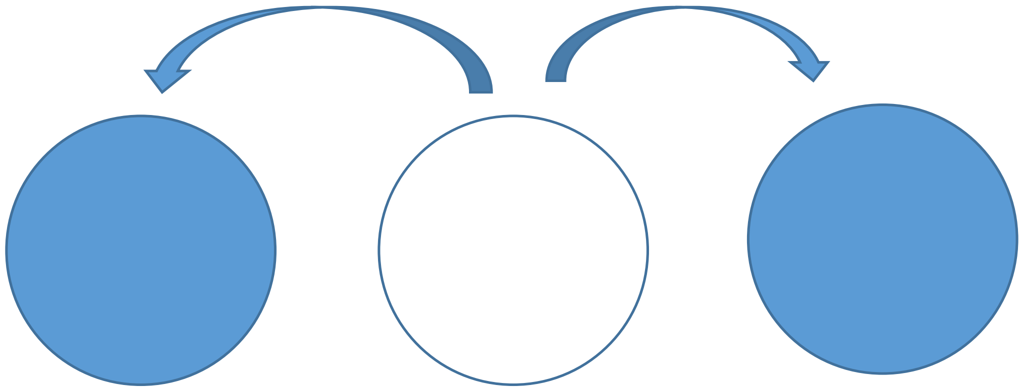
000
001
011
010
110
111
101
100

dist = 1

Código A

código A
0001
0010
0100
1000

Distancia 2



Puedo detectar errores simples, pero no puedo corregirlos.

Código C

01011 3
01100

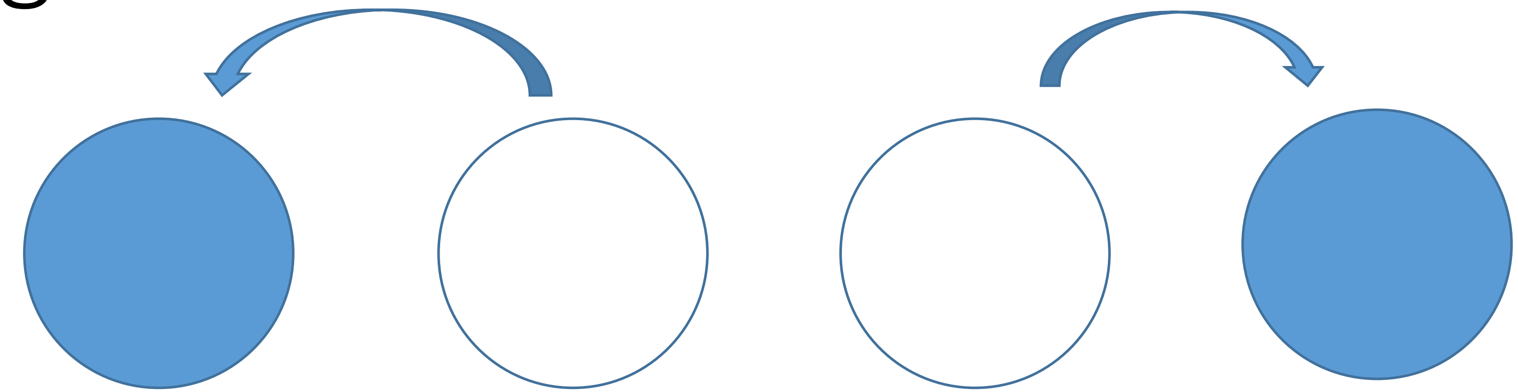
01011 3
10010

01011 4
10101

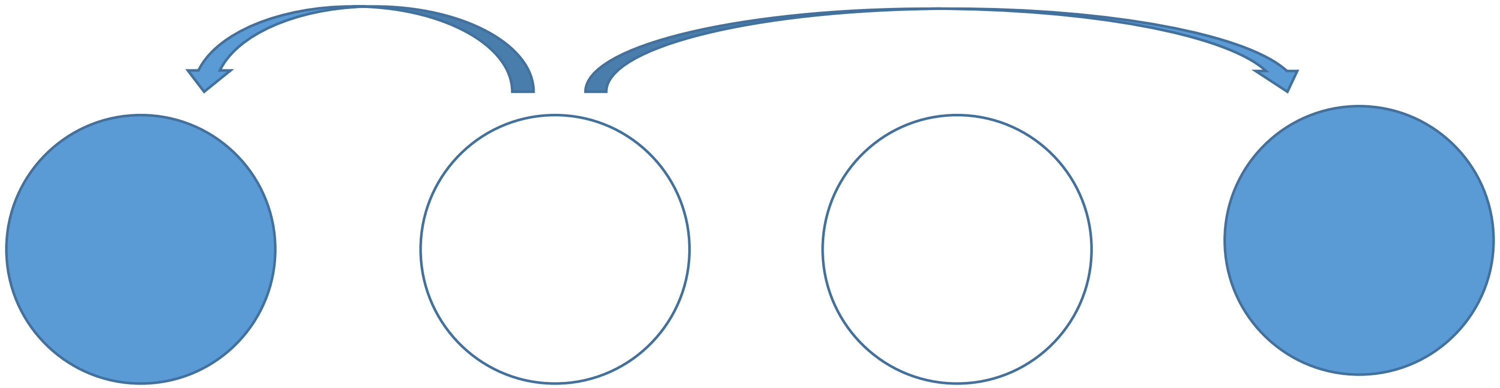
01100 4
10010

01100 3
10101

10010 3
10101



Opción 1: asumo que solo hay errores simples, entonces puedo corregirlos



Opción 2: asumo que hay errores dobles, entonces no puedo corregirlos porque no sabría cuál fue la palabra original, pero puedo detectarlos

Distancia 3

Ejercicio 2.6

Ejercicio 6. Se consideran los cuatro códigos siguientes:

código A	código B	código C	código D
0001 0010 0100 1000	código Gray de 3 bits	01011 01100 10010 10101	000000 001111 110011

a) ¿Cuál de las siguientes propiedades son satisfechas por cada uno de ellos?

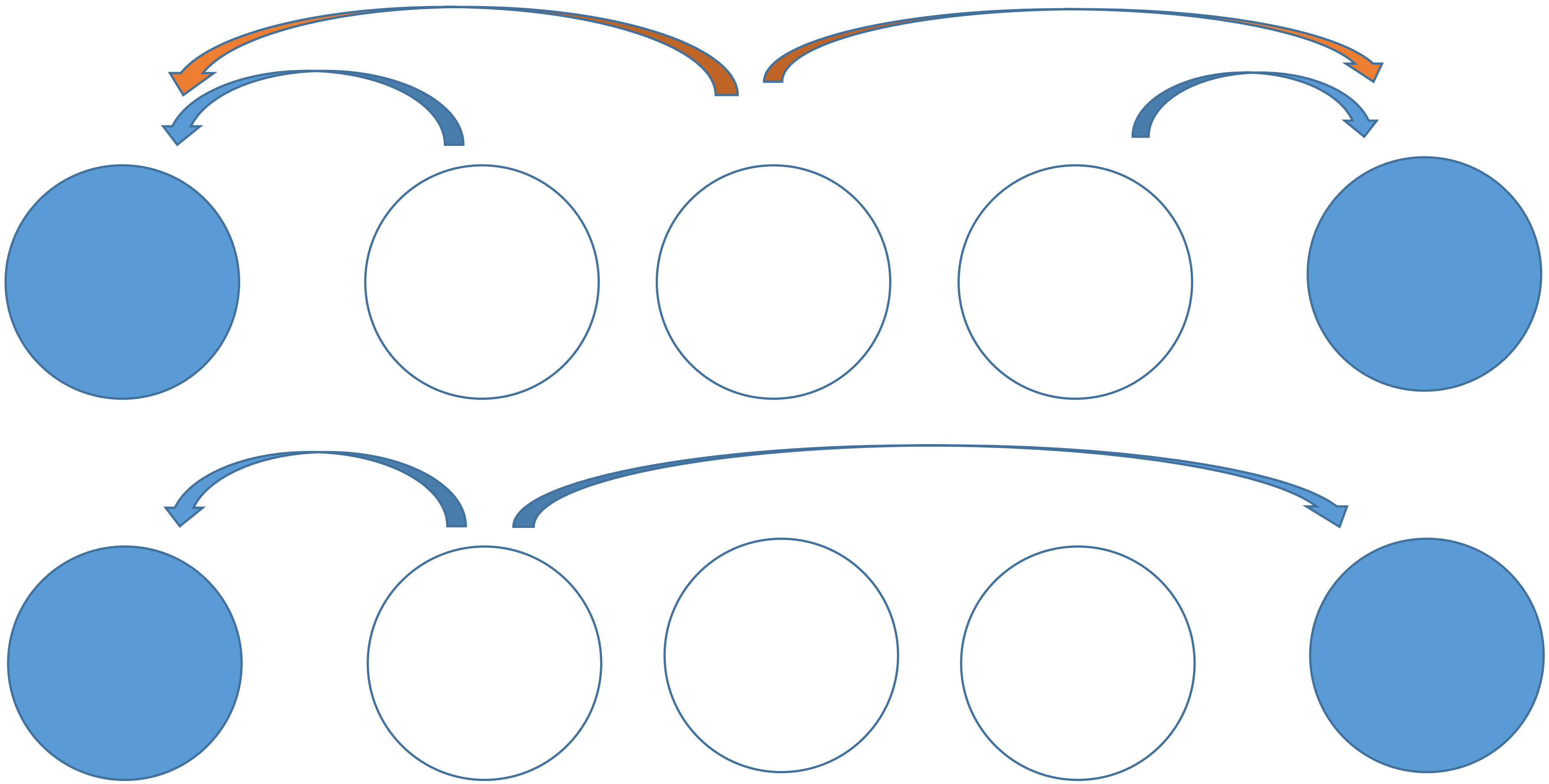
- Detectar errores simples.
- Detectar errores dobles.
- Detectar errores triples.
- Corregir errores simples.
- Corregir errores dobles.
- Corregir errores simples y detectar errores dobles.

b) ¿Cuántas palabras pueden agregarse al código A sin cambiar sus propiedades de detección y corrección de errores?. Dé un posible conjunto de dichas palabras. ¿Es este conjunto único?

Código D

Distancia 4

Código D
000000
001111
110011



Opción 1: asumo que a lo sumo hay errores dobles, entonces puedo detectar errores dobles y corregir errores simples.

Opción 2: asumo que hay errores triples, entonces no puedo corregir errores simples, pero puedo detectar errores triples.

Ejercicio 2.6 parte b)

código A
0001
0010
0100
1000

- Cantidad máxima de palabras distintas posibles con 4 bits: $2^4 = 16$
- Para garantizar distancia 2, solo puedo usar la mitad de las palabras, por lo que en mi código como máximo voy a tener 8 palabras válidas.
- La palabra 0000 no la puedo agregar porque está a distancia 1.
- Palabras que tengan 2 unos y 2 ceros tampoco puedo utilizar, porque forzosamente va a estar a distancia 1 de dos de las 4 palabras ya existentes.
- Tengo que usar las palabras que tienen 3 unos y 1 cero: 1110 – 1101 – 1011 – 0111. La palabra 1111 no la puedo usar porque está a distancia 1 de este conjunto de palabras (forma otro conjunto).

Codificación con paridad

- Paridad par: agrego un bit de paridad tal que la cantidad de 1s de la nueva palabra sea par.

- Paridad impar: agrego un bit de paridad tal que la cantidad de 1s de la nueva palabra sea impar.

Cant. de bits para detectar y corregir errores

- En general para corregir hasta en c bits: **dist= 2c+1**
- **dist=2c+1+d** - corrijo hasta en c y detecto d más de los que corrijo.
- Para códigos con distancia 3 se cumple:

$\mathbf{2^m <= 2^n/(n+1)}$

Con m: cantidad de bits de información

n: cantidad de bits a trasmitir n=m+r

r: cantidad de bits de redundancia

- Para m=4 y r=3 se cumple la igualdad de $\mathbf{2^m = 2^n/(n+1)}$ $2^4 = 2^7/(7+1)$
- Una implementación de esta codificación es el código de Hamming.

Código de Hamming

Posición	1	2	3	4	5	6	7
Palabra	P1	P2	m3	p3	m2	m1	m0

- Los bits de paridad van en las posiciones que son potencias de 2.

7= 4 + 2 + 1

6= 4 + 2

5= 4 + 1

3= 2 + 1

Por lo tanto 1 (p1) aparece para formar el número de las posiciones 7, 5 y 3 que corresponden a los bits m0, m2 y m3 respectivamente. Las paridades las calculo con el exor de los bits de las posiciones que contienen esa paridad.

p1 = m3 $ m2 $ m0

p2= m3 $ m1 $ m0

p3= m2 $ m1 $ m0

- Para chequear si hubo error, en la recepción chequeo:
- v1= p1 $ m3 $ m2 $ m0 = 1 $ 3 $ 5 $ 7
- v2= p2 $ m3 $ m1 $ m0 = 2 $ 3 $ 6 $ 7
- v3= p3 $ m2 $ m1 $ m0 = 4 $ 5 $ 6 $ 7
- La posición del bit erróneo es = v3 v2 v1. (Si no hay error, v3 v2 v1 = 000).

Ejercicio 2.7

Ejercicio 7. Se desea transmitir dígitos decimales en código BCD a través de un canal ruidoso. Para eso se genera a partir del código BCD un código Hamming de 7 bits utilizando paridad par. Decodifique el siguiente mensaje recibido, asumiendo que a lo sumo ha ocurrido un único error en cada palabra del código:

$\boxed{1001001}$ 0111001 1110110 0011011

Orden de los bits: p1 p2 m3 p3 m2 m1 m0 Dígito BCD: m3 m2 m1 m0

- $v1 = p1 \$ m3 \$ m2 \$ m0 = 1 \$ 3 \$ 5 \$ 7$
- $v2 = p2 \$ m3 \$ m1 \$ m0 = 2 \$ 3 \$ 6 \$ 7$
- $v3 = p3 \$ m2 \$ m1 \$ m0 = 4 \$ 5 \$ 6 \$ 7$
- $v1 = 1 \$ 0 \$ 0 \$ 1 = 0$
- $v2 = 0 \$ 0 \$ 0 \$ 1 = 1$
- $v3 = 1 \$ 0 \$ 0 \$ 1 = 0$
- Error en la posición 2 (bit de paridad), el mensaje m3m2m1m0 = 0001 es decir dígito 1.

Ejercicio 2.7

Ejercicio 7. Se desea transmitir dígitos decimales en código BCD a través de un canal ruidoso. Para eso se genera a partir del código BCD un código Hamming de 7 bits utilizando paridad par. Decodifique el siguiente mensaje recibido, asumiendo que a lo sumo ha ocurrido un único error en cada palabra del código:

1001001 0111001 1110110 $\boxed{0011011}$

Orden de los bits: p1 p2 m3 p3 m2 m1 m0 Dígito BCD: m3 m2 m1 m0

- v1= p1 $ m3 $ m2 $ m0 = 1 $ 3 $ 5 $ 7
- v2= p2 $ m3 $ m1 $ m0 = 2 $ 3 $ 6 $ 7
- v3= p3 $ m2 $ m1 $ m0 = 4 $ 5 $ 6 $ 7
- v1= 0 $ 1 $ 0 $ 1 = 0
- v2= 0 $ 1 $ 1 $ 1 = 1
- v3= 1 $ 0 $ 1 $ 1 = 1
- Error en la posición 6 (bit m1), el mensaje recibido m3m2m1m0 = 1011 tiene error en m1. Corrijo: 10**0**1 es decir enviaron dígito 9.

Álgebra de Boole

1) $x+0=x$ $\quad$ $x+1=1$

   $x*0=0$ $\quad$ $x*1=x$

2) $x+x=x$

   $x*x=x$

3) $x + !x = 1$

   $x. !x = 0$

4) $!(!x)=x$

5) $x.y = y.x$ $\quad$ $x+y= y+x$

6) $x(yz) = (xy)z$

   $x+(y+z) = (x+y)+z$

7) $x(y+z) = xy + xz$

   $x + yz = (x + y).(x + z)$

8) $x.(!x + y) = xy$

   $x + !xy = x + y$

   $x.(x+y) = x$

   $x+ (x.y) = x$

De Morgan

$!(x+y) = !x . !y$

$!(x.y) = !x + !y$

Función como suma de minitérminos o producto de maxitérminos

Ejercicio 4. (E1.5) Dada la siguiente tabla de verdad, hallar una función lógica equivalente S(A,B,C,D):

A	B	C	D	S
1	1	0	0	1
0	0	1	1	1
0	0	0	0	1
En otro caso				0

NOTA: el resto de las combinaciones dan todas cero.

- S = A.B.!C.!D + !A.!B.C.D + !A.!B.!C.!D
- O como producto de 13 maxitérminos

Ej. 3.2

Ejercicio 2. (Kohavi 3.22) Cinco estudiantes de ingeniería A, B, C, D y E quieren acampar en sus vacaciones, pero deben cumplirse las siguientes condiciones:

1. Deben ir A o B o ambos.
2. Deben ir C o E, pero no ambos.
3. Deben ir A y C conjuntamente, o ninguno de los dos va.
4. Si va D, entonces E también debe ir.
5. Si va B, entonces A y D también deben ir.

x + !xy = x +y

Reemplazo x por !D

- C1) A + B
- C2) C.!E + E.!C
- C3) A.C + !A.!C
- C4) DE + !D --> (* prop 8) !D + E
- C5) !B + BAD --> (* prop 8) !B + AD
- Se tienen que cumplir las 5 condiciones
- F= (A + B).(C.!E + E.!C).(AC + !A.!C).(E + !D).(!B + AD)
- F= (AC!E + BC!E + AE!C + BE!C).(ACE +!A!CE+ AC!D +!A!C!D).(!B + AD)
- F= (AC!D!E + ABC!D!E + !AB!CE + !AB!C!DE).(!B + AD)
- F= A!BC!D!E es decir que van A y C.

Propiedad

- xy + x!y =
- x.(y +!y) =
- x.1 = x
- **xy + x!y = x**

Ejercicio minimización

Recordar:
xy + x!y = x

- Minimizar la siguiente función utilizando Algebra de Boole:
- F(a,b,c)= !a!b!c + !ab!c + !abc + ab!c + a!b!c
- Solución:
- 1. F(a,b,c)= !a!b!c + !ab!c + **!ab!c** + !abc + ab!c + a!b!c [[x + x= x]]
- 1. F(a,b,c)= !a!b!c + !ab!c + !ab!c + !abc + ab!c + a!b!c
- 2. F(a,b,c)= !a!c + !ab + a!c [[xy + x!y = x]]
- 3. F(a,b,c)= !ab + !c [[xy + x!y = x]]